Oracle® WebCenter Sites

Administrator's Guide

11*g* Release 1 (11.1.1)

April 2012

Oracle® WebCenter Sites: Administrator’s Guide, 11g Release 1 (11.1.1)

Primary Author: Sean Cearley

Contributor: Kannan Appachi, Ravi Khanuja, Tatiana Kolubayev, Melinda Rubenau, Bonnie Sedotto

Table of

Contents

Part 1. Introduction

Part 2. CM Site Configuration Procedures

Part 3. Export and Mirror Publishing

Part 4. RealTime Publishing

Part 5. System Configuration Procedures

Part 6. Performance Enhancement Features

Part 7. Diagnostic Utilities

Appendices

About This Guide

This guide describes the Oracle WebCenter Sites administrator's environment, accessible from the Admin interface. It begins with an overview of Oracle WebCenter Sites, its add-on products, content management concepts, and the process you will follow to create your content management (CM) framework.

The rest of the guide explains your main tasks: how to configure Oracle WebCenter Sites so that writers, editors, and other content providers can electronically create and manage content, collaborate in workflows, and publish to the online site. Other sections describe Oracle WebCenter Sites' database and utilities for tuning and maintaining Oracle WebCenter Sites installations.

The *Oracle WebCenter Sites* application, discussed in this guide, is a former FatWire product.

- *Oracle WebCenter Sites* is the current name of the application previously known as *FatWire Content Server*. In this guide, *Oracle WebCenter Sites* is also called *WebCenter Sites*.
- *Oracle WebCenter Sites: Satellite Server* is the current name of the application previously known as *FatWire Satellite Server*. In this guide, *Oracle WebCenter Sites: Satellite Server* is also called *Satellite Server*.
- *Sites Explorer* is the current name of the application previously known as *FatWire CS Explorer*.
- *Sites DocLink* is the current name of the application previously known as *CS-DocLink*. In this guide, *Sites DocLink* is also called *DocLink*.
- *Sites Desktop* is the current name of the application previously known as *FatWire CS-Desktop*. In this guide, *Sites Desktop* is also called *Desktop*.
- *Oracle WebCenter Sites: Engage* is the current name of the application previously known as *FatWire Engage*. In this guide, *Oracle WebCenter Sites: Engage* is also called *Engage*.
- *Oracle WebCenter Sites: Web Experience Management Framework* is the current name of the application previously known as *FatWire Web Experience Management Framework*. In this guide, *Oracle WebCenter Sites: Web Experience Management Framework* is also called *WEM Framework*.

Audience

This reference is written especially for administrators. It is assumed that administrators have a clear knowledge of their company's business needs, and a basic understanding of their roles in the site development process. This guide is also useful to developers, who collaborate with administrators by designing and coding the data model, presentation templates, security system, and caching options.

Administrators are not required to have programming experience, although a technical background is assumed. Developers, however, must know Java, JavaScript Pages, XML, and HTML.

Related Documents

For more information, see the following documents:

- *Oracle WebCenter Sites User's Guide*
- *Oracle WebCenter Sites Developer's Guide*
- *Oracle WebCenter Sites Property Files Reference*

How This Guide Is Organized

Information in this guide is organized by parts, where each part presents a set of chapters that are related to a particular task or function.

Part 1, "Introduction" presents an overview of Oracle WebCenter Sites, its add-on products, the interface you will be using, and the process you will follow to create your content management (CM) framework.

Part 2, "CM Site Configuration Procedures" describes Oracle WebCenter Sites' administrative environment. It provides procedures for creating and managing users, assembling and organizing content management sites (the back end of the online site), managing users' access to CM sites, creating workflow processes, and replicating sites.

Part 3, "Export and Mirror Publishing" describes Oracle WebCenter Sites' publishing and approval systems. In particular, it provides an in-depth discussion of Export to Disk publishing. This part also contains procedures for configuring Mirror to Server, Export to Disk, and Export to XML publishing processes.

Part 4, "RealTime Publishing" describes Oracle WebCenter Sites' RealTime publishing method, including instructions on configuring the RealTime destination and using the RealTime publishing interface.

Part 5, "System Configuration Procedures" presents information about configuring user interfaces, managing the publishing system, revision tracking, and implementing search engines.

Part 6, "Performance Enhancement Features" contains information about the inCache framework and configuring the framework to support page and asset caching.

Part 7, "Diagnostic Utilities" contains information about system tools that enable administrators to diagnose the Oracle WebCenter Sites system. For example, administrators can configure log4j logging, search the contents of the log file, manage page and asset caches, and test the performance of the shared file system.

The final part, "Appendices," contains reference information about system defaults, the Oracle WebCenter Sites database, and LDAP authentication and deployment options.

At the end of this guide is a standard index and an index of procedures to help you quickly locate procedures for completing administrative tasks from the administrator's interface.

As you read this guide, keep a copy of the *Oracle WebCenter Sites Property Files Reference* handy. The *Property Files Reference* provides detailed descriptions of the properties that are mentioned in this guide.

Graphics in This Guide

Many steps in this guide include screen captures of dialog boxes and similar windows that you interact with in order to complete the steps. These screen captures are presented to help you follow the installation process. They are not intended to be sources of specific information, such as parameter values, options to select, or product version number.

Conventions

The following text conventions are used in this guide:

- **Boldface** type indicates graphical user interface elements that you select.
- *Italic* type indicates book titles, emphasis, or variables for which you supply particular values.
- `Monospace` type indicates file names, URLs, sample code, or text that appears on the screen.
- **`Monospace bold`** type indicates a command.

Third-Party Licenses

Oracle WebCenter Sites and its applications include third-party libraries. For additional information, see *Oracle WebCenter Sites 11gR1: Third-Party Licenses*.

Part 1

Introduction

This part of the administrator's guide introduces you to Oracle WebCenter Sites, its system of managing content, the underlying concepts and constructs, and how they are implemented. The chapters in this part of the guide also provide you with guidelines for modeling your online site in the Oracle WebCenter Sites interface, as well as planning the configuration tasks that you will routinely perform in the Oracle WebCenter Sites environment.

This part contains the following chapters:

- Chapter 1, "Overview"
- Chapter 2, "Administrator's Interface"
- Chapter 3, "Site Configuration Guidelines"

Chapter 1

Overview

Welcome to Oracle WebCenter Sites' content management system. In this chapter, you will learn about CM sites in the Oracle WebCenter Sites environment. The sites can be thought of as the back end of your online site, a place where business users work to create their electronic assets, manage them, and deploy them to their audiences.

As the administrator, you will be working with CM sites throughout your job, configuring the sites, replicating them, and managing their day-to-day use. Because they are a critical part of your work, you need to understand what CM sites are, and how they relate to the online site. This chapter provides an overview.

This chapter also outlines the administrator's job in the Oracle WebCenter Sites environment and offers guidelines for planning configuration tasks.

This chapter contains the following sections:

Introduction

As the Oracle WebCenter Sites administrator, your job is to build the foundation of your online site, specifically, its back end. The back end itself is a site (or set of sites) in the Oracle WebCenter Sites installation—a site where business users work to create their electronic assets, manage them, and deploy them to their audiences. Building the back end involves configuring the business users, linking them to content-entry forms as well as other authoring tools, and providing the users with publishing and delivery systems for serving the online site to browsers.

When the back end is configured, its users log on and fill in content-entry forms. The content they enter is then saved to tables in the Oracle WebCenter Sites database. When ready for delivery, the content is drawn programmatically from the database tables, formatted, and served to browsers as the online site. Figures 1 and 2 illustrate the process flow, using a Mirror to Server publishing scenario.

In our scenario, Figure 1 depicts the smallest unit that can function as a back end: a user who has permissions to write to the Oracle WebCenter Sites database and to invoke the publishing-delivery systems. This unit, Oracle WebCenter Sites defines as a "CM site." Oracle WebCenter Sites imposes no limit on the number of CM sites that you can configure for the back end of the online site, or the number of ways in which the sites can be configured.

Throughout your job as the Oracle WebCenter Sites administrator, you will be working with CM sites—configuring them, replicating them, and managing their day-to-day use. Because they are a critical part of your work, you need to understand how CM sites are defined, how they must be configured, and how they relate to the online site. This chapter provides an overview.

Online Sites

Although most Internet users don't need a definition of "online site," it is important to describe what an online site powered by Oracle WebCenter Sites is. Later in this chapter, we explain how online sites can be modeled by CM sites.

A Sites-powered online site is the set of pages that an organization displays to its target audience of customers, clients, and casual visitors via Oracle WebCenter Sites. The online site can be accessible to the general public or it can be a password-protected site. It can also be a completely exclusive site, such as a corporate intranet or departmental network, operating strictly within the private domain.

Regardless of its nature, an online site originates from either a single CM site, or many CM sites, depending on which model you choose. Throughout our product guides, we use the term "online site" generically to refer to websites.

Figure 1: CM Site

This content was created by a user populating one of the content-entry forms below.

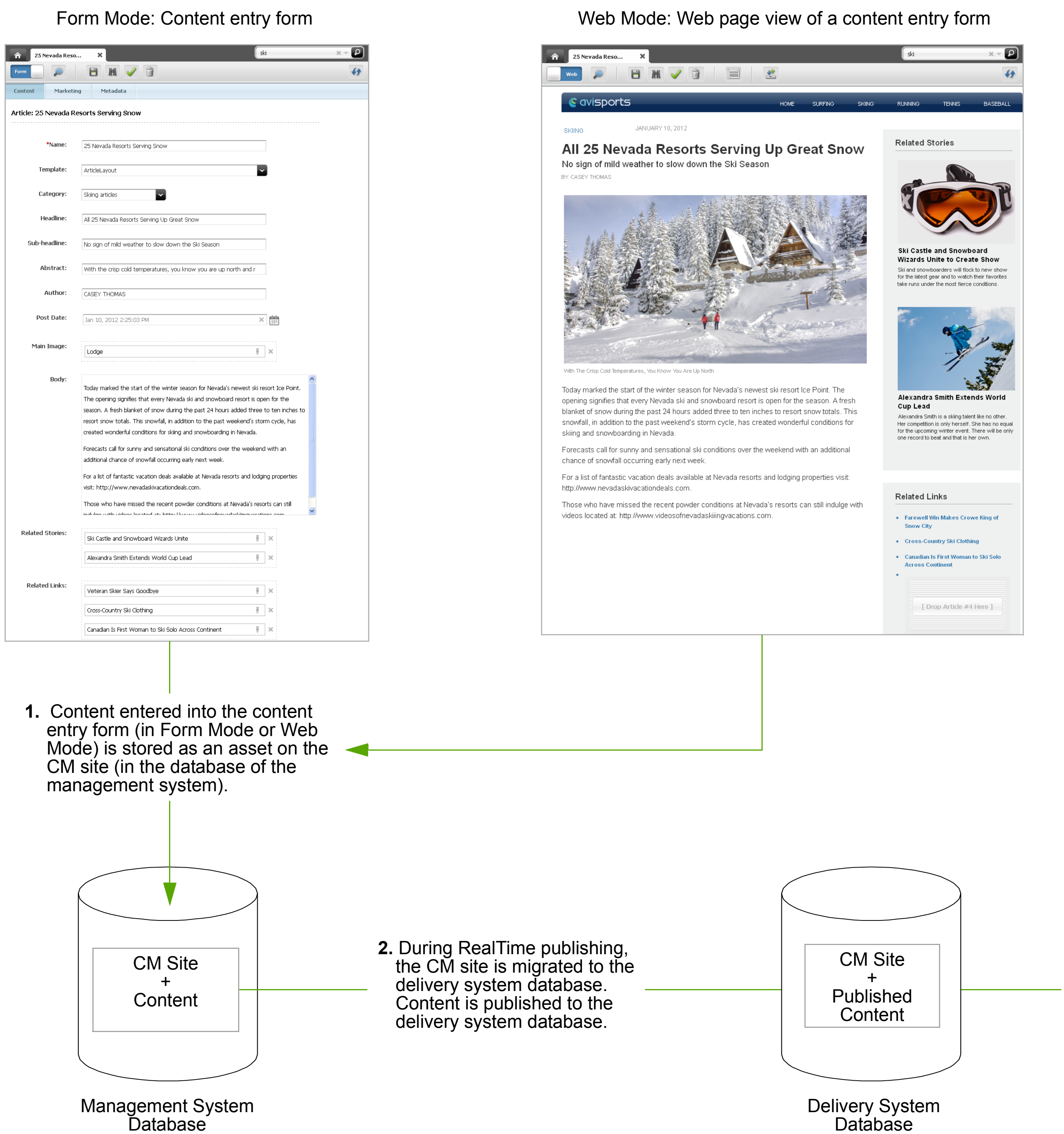
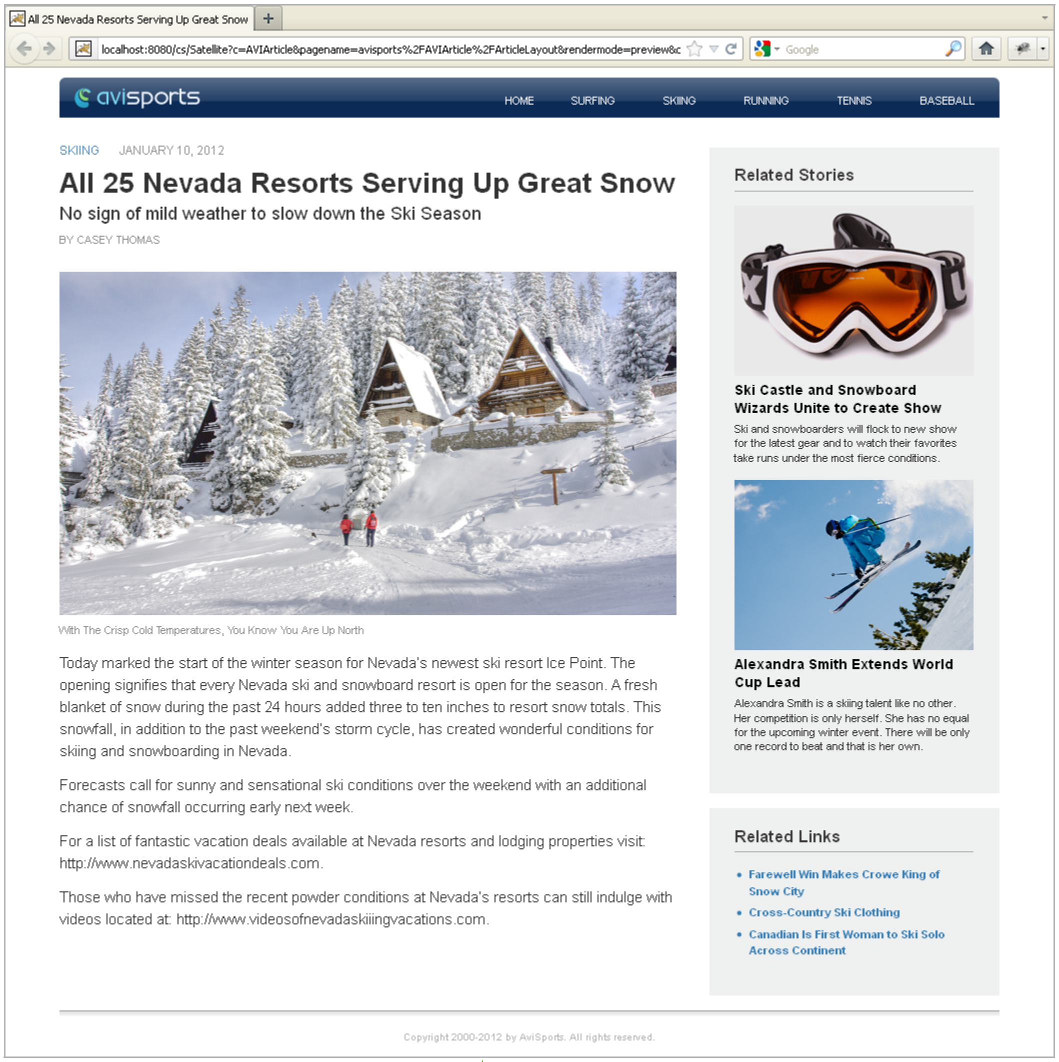


Figure 2: Online Site

This online site, a single page, was created from the asset shown in Figure 1.



3. Published content is formatted by templates, laid out by templates, and delivered to the online site.

CM Sites

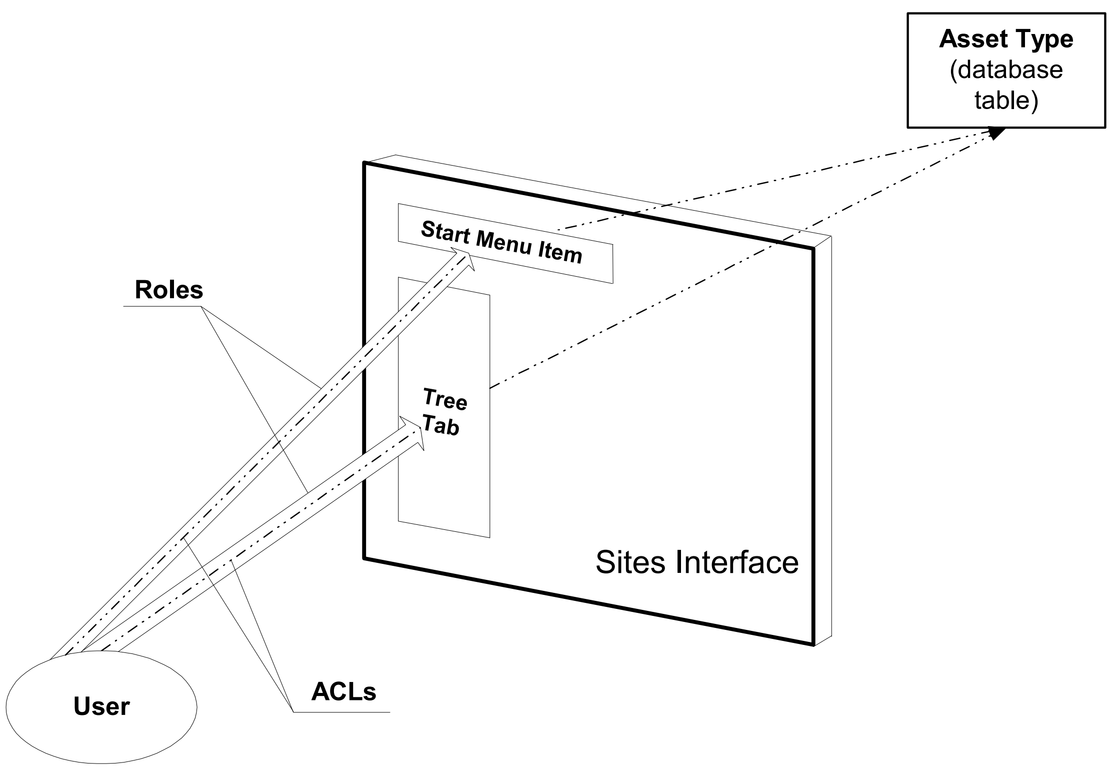
A CM site is a content management unit within Oracle WebCenter Sites. It is the source of content for the online site and can represent either an entire online site or one of its sections.

Note

From this point forward, when the term "site' is used without a qualifier, it means "CM site."

A site is an object that you must configure in order to (1) define the authors and managers of the online site, and (2) provide them with the permissions and content management tools they will need: content-entry forms, content-rendering templates, workflow processes, start menu items, publishing methods, and a delivery system. The process of configuring a site involves creating not only the site components, but also associating the components with each other. Making the associations defines a site, similar to the one shown in Figure 3.

Figure 3: CM Site



The site is identified by a name (also configured by the administrator), stored in Oracle WebCenter Sites' database, and listed in authorized users' interfaces. Users with the correct roles have access to the site. Within the site, roles manage the users' access to specific functions (such as a "Start Menu Item") in the Oracle WebCenter Sites interface. ACLs manage the user's permissions to content (database tables). Through the interface functions (made accessible by roles), the user is able to actuate his permissions (as defined in the ACLs), and therefore operate on the database tables in order to author and manage specific types of content.

A Oracle WebCenter Sites system typically has many such sites, each one unique in its function and composition. When users log in to Oracle WebCenter Sites, they must also select a site where they will work. The site can be independent of other sites, or it can share assets with other sites, providing that the sites have a common set of users.

Note

In this release, Oracle WebCenter Sites offers a site-replication utility called "Site Launcher" that speeds up the site creation process. Instead of creating sites from scratch, you can replicate established sites as necessary, modify the replicates, and spin them off as new sites in the Oracle WebCenter Sites environment.

Configuration Components

Configuring a site involves using Oracle WebCenter Sites' administrator interface to access a system-wide configuration pool, select (or create) components that will make up the site, and associate the components with each other so they can function together to produce either the online site or one of its sections.

Table 1 lists site components, most of which are required. The optional components vary according to business needs and user preferences. Whereas developers are responsible for the code-based components, the administrator is responsible for all other components.

Site components are generally complex constructs, especially the data model. They are described in the sections that follow. Data modeling is described in detail in the *Oracle WebCenter Sites Developer's Guide.*

Table 1: Site Components

Component	Required	Created by	See page …
CM site definition	✓	Sites administrator	28
Data model	✓	Developers	29
Users	✓	Sites administrator	30
Roles	✓	Sites administrator	31
Start menu items	✓	Sites or its administrator	32
Workflow processes	–	Sites administrator	33
Publishing system	✓	Sites administrator	33
User interface options	–	Sites administrator	33

CM Site Definition

A CM site definition is one of the components of a CM site. It consists of a site name and optionally a description, both specified by the administrator. The site name can represent an online site, a business topic, the work of a department, or yet another type of content. In any case, the site name establishes a business theme that users in the site are responsible for developing and maintaining.

When you specify a site name, Oracle WebCenter Sites creates a node to represent the site in its interface. It also appends several default sub-nodes for linking components to the site: an “Asset Types” sub-node for linking the data model to the site, a “Users” sub-node for linking users to the site, and two other nodes for enabling the optional user interfaces Desktop and DocLink. Descriptions of these components are given in the next sections.

For content providers, the site is a visibility control mechanism. The site provides authorized users access to certain content in the Oracle WebCenter Sites installation. When a site is properly configured, its name is displayed in the content providers’ interfaces, allowing the content providers to select the site and navigate within it according to their roles and permissions.

Data Model

The data model is a component of a site. It comprises a set of asset types (database tables) and asset type definitions, coded by developers for content providers’ use. To help developers equip content providers with the broadest possible set of content management options, Oracle WebCenter Sites supports three kinds of asset types.

- Content asset types, which are structured repositories for content and by design reflect the business theme established by the site definition. For example, if you defined a site named “Social Events,” suitable content asset types could be “Selected Moments in History,” “The 20th Century’s Greatest Events,” and so on, since they pertain to the theme suggested by the site name.

  From asset types, developers code asset type definitions. These are expressed in the Sites interface as content-entry forms (such as the one in Figure 1), whose fields prompt users for information that will be delivered to the online site (or reserved for internal use, if necessary). The set of fields defines the asset type; users’ field entries define the asset (an instance of the asset type).
- Design asset types, which are used by developers to code template assets, which render the content assets. (We distinguish template assets from content assets in order to distinguish presentation code from content.)
- Management asset types, which are also used by developers to create tools such as simple searches and database queries that help content providers manage their content.

Ordinarily, developers test the asset types they create and pass them on to you so that you can link them to the site definition (through the “Asset Types” node) and complete another step in the site configuration process.

Note

Oracle WebCenter Sites provides a number of default asset types and allows developers to create their own. The asset types, their definitions, and the assets themselves are stored in the Oracle WebCenter Sites database as tables or table entries, and loosely referred to as “content” in this guide.

For information about data modeling, see the *Oracle WebCenter Sites Developer’s Guide.* We recommend that administrators read the guide to gain a basic understanding of asset types.

Users and Their Table-Level Permissions

Users are site components. In this guide, they are often referred to as “content providers”—people who use the developers’ data model to author and manage content.

Content providers are the subject matter experts. They can be:

- Authors of online content. Copywriters and designers would fall into this group.
- Reviewers, who examine and edit the content that other users submit to them in order to ensure its quality. Examples of content reviewers include editors and art directors who review and modify the copy and designs that are submitted to them.
- Content publishers, who ensure that content is ready to be delivered to the online site, and approve the content for delivery. An editor-in-chief could be a content publisher.
- Content managers, who oversee the authoring, review, and publishing processes.

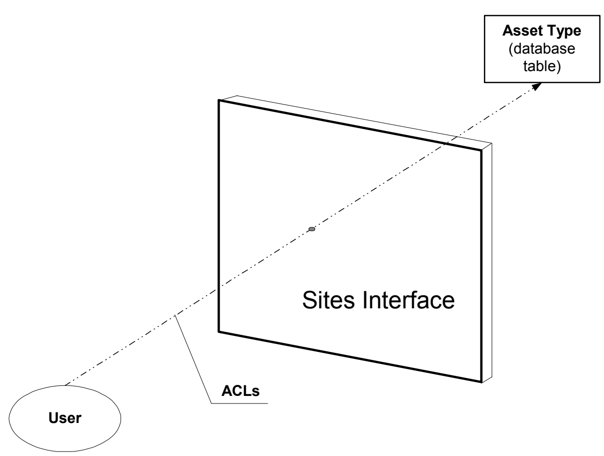
Each content provider must be identified to Oracle WebCenter Sites through a user account, which consists of a user name, a password, and Access Control Lists (ACLs), the foundation of Oracle WebCenter Sites’ security system.

An ACL is a set of permissions to database tables. The permissions (such as read and write) are granted when the same ACL is assigned to both the table and the user. If no ACLs are common to a table and a user, the user has no permissions to the table.

For example, the system table named `SystemUsers` contains user account information. The table is assigned three ACLs: SiteGod, UserReader, and UserEditor. If a user is assigned one of the ACLs—UserReader, in our example—he can read the table. If the same user is assigned a second ACL—UserEditor—he can also edit the table. If the user is not assigned any of the ACLs, he has no permissions to the table.

Be aware, that while ACLs give the user *permissions* to operate on tables, they do not give the user *the means* to operate on the tables. For example, in our preceding scenario, the user’s permissions to read and edit the table translate into permissions to use the “view” and “edit” functions in the Oracle WebCenter Sites interface. However, the functions are hidden from the user, unless the same roles are assigned to both the user and the functions. (For information about roles, see the next section, “Roles,” on page 31.)

ACLs give the user permission (but not the means) to operate on database tables



In general, Oracle WebCenter Sites uses ACLs at two levels:

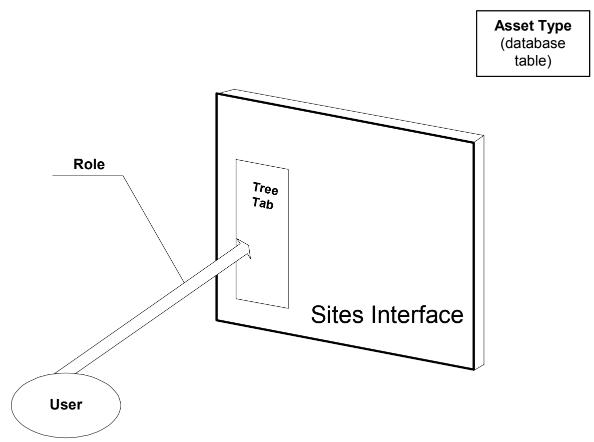
- At the system security level, to provide authentication functionality and therefore prevent hackers from entering the Oracle WebCenter Sites environment.
- At the interface level, to control users’ permissions to database tables and, therefore, control the ability to use (but not view) interface functions through which the permissions are actuated.

Oracle WebCenter Sites provides a number of default system ACLs and pre-assigns them to system tables. You can re-use the ACLs, or configure your own and assign them to custom tables, as necessary. For more information about ACLs, see Chapter 4, "Working with ACLs and Roles" and Appendix A, "System Defaults."

Roles

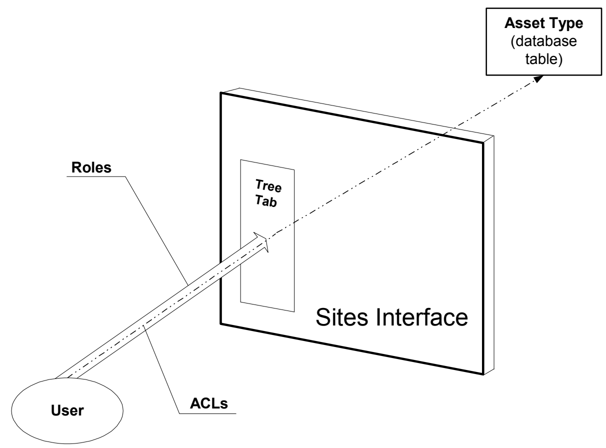
Whereas ACLs give the user permission to operate on database tables, roles give the user the means to operate on database tables. Roles determine whether the user has access to a site, and whether interface functions, such as "edit," "delete," and "start workflow" are exposed in the user's interface. If the functions are hidden, the user's permissions to database tables (as specified in the ACLs) cannot be actuated, leaving the user unable to operate on the tables. Roles also define groups of users, such as "authors" and "editors." They are used to describe the groups' permissions (and therefore the users' permissions) to sites, the sites' content, to collateral (such as start menus for creating and locating content), and to workflow processes.

Roles, without ACLs, give the user the means (but not the permission) to operate on database tables



Roles are implemented in the same way as ACLs; that is, for a function to be displayed in the user's interface, the function and the user must be assigned the same role. To illustrate, we continue our previous scenario (on page 30), where a user is given editorial permissions to the `SystemUser` table through the UserReader and UserEditor ACLs. To exercise those permissions, the user needs access to the "edit" function in the Oracle WebCenter Sites interface. The "edit" function is located in the "Admin" tab. To view the "Admin" tab, however, the user must be assigned the same role as the tab. By default, the tab is assigned the GeneralAdmin role; the same role must also be assigned to the user.

ACLs + roles give the user the permission *and* the means to operate on database tables



To summarize, the user gains full access to the database table only when he is assigned the ACLs of the table, *and* the role of the “Admin” tab. In practice, as in our scenario, roles and ACLs must be compatibly assigned—role assignments must support the permissions that are granted by ACL assignments.

In addition to displaying interface functions to site users, roles provide a way of grouping users according to their responsibilities on the site. Users with similar responsibilities can be assigned the same role(s). For example, administrative users need access to the “Admin” tab. All the administrators can be assigned the GeneralAdmin role and thereby be given access to the “Admin” tab.

Oracle WebCenter Sites defines several default system roles, all of which are pre-assigned to various functions in the Sites interface. Oracle WebCenter Sites also allows you to configure and assign your own roles. When choosing role names, consider the responsibilities of the users on the site and select the role names accordingly. Note that unlike ACLs (which are mapped to database tables), roles are mapped to sites and functions on the sites (which means that roles must be assigned on a per-user, per-site basis).

Note

Unlike ACLs, roles are exposed to content providers for enlisting other content providers into workflow processes.

Start Menus

A start menu is a site component. Items on the menu provide a way of coupling a user with the asset types he is to work with on the site. The coupling is accomplished by means of roles.

Oracle WebCenter Sites defines several start menus: **New**, which allows the user to create assets on the site; **Search**, which allows the user to look for and edit assets on the site; start menus for **Sites Desktop**, which enables Microsoft Word as an alternative interface to Sites content-entry forms; and start menus for **Sites DocLink** which enables the document management interface.

A start menu item specifies:

- The site(s) to which the start menu item applies
- The asset type that users on the sites can work with
- The roles that are allowed to create or search for assets of those types
- The workflow processes (if any) in which the roles can participate

In other words, a start menu item determines which roles can create and search for assets of a specific type on a site, and which workflow processes the roles can participate in.

Oracle WebCenter Sites creates start menu items automatically and gives you the option to configure your own.

Workflow Processes

Workflow processes are optional constructs, mandated only by business needs, and used to regulate collaborations among content providers.

A workflow process enlists qualified users with complementary expertise to perform a sequence of operations that starts with the creation of a content asset, continues with review of the asset, and culminates in approval of the asset. Once approved, the asset is published to the delivery system by one of the users and finally, delivered as content to the online site.

Once workflows are configured, they can be specified in start menu items as processes that automatically begin the moment newly created assets are saved. Workflow processes can also be omitted from start menu items, to be invoked by users as necessary. For more information about workflow processes, see Chapter 9, "Creating and Managing Workflow Processes."

Publishing System

The publishing system is a site component. The publishing system gives users the means to migrate sites and their content from one system to another. The publishing methods that can be configured are: RealTime, Mirror to Server, Export to Disk, and export to XML. The structure of the site (that is, the database schema) is migrated by means of either the Mirror to Server or the RealTime publishing method. The content itself can be published at the administrator's discretion.

User Interface Options

Oracle WebCenter Sites supports a number of optional interfaces through which users can interact with content:

- **Oracle WebCenter Sites Desktop (Sites Desktop)**, which provides users with the familiar Microsoft Word interface as an alternative to the content-entry forms that are native to Oracle WebCenter Sites. Users can create their content in MS-Word documents, import the Word documents as assets into the Sites database, and from there, recall and edit the documents.

  Note that Sites Desktop requires the content in Word documents to be structured. For example, when using Oracle WebCenter Sites Desktop to author content, the user opens a Word document, enters content, and structures the content by tagging it with the same field names as defined in the equivalent content-entry form. The tagging utility is embedded in the Word interface, and the selection of fields is determined by the administrator. When the Word document is saved, the content in its fields is parsed to the appropriate database table(s).

- **Oracle WebCenter Sites DocLink (Sites DocLink)**, which supports unstructured content in the flex asset family.

  Sites DocLink provides a drag-and-drop interface for uploading and downloading unstructured content—documents, graphics, or other single binary files that are managed as flex assets. Sites DocLink also presents the hierarchical structure of any flex asset family in the Oracle WebCenter Sites database as folders and files in the Windows Explorer application.

- **Web Mode of the Contributor interface**, which supports the editing of content directly on the rendered page. Regular Oracle WebCenter Sites users can make quick edits in context, while infrequent users can accomplish their work without having to learn the Oracle WebCenter Sites interface.

Summary

It is important to understand that *the components of a site are not site-specific; the associations among them are.*

Site components are constructs that either exist or must be created in a system-wide pool. From this pool, you select the components that must interact with each other in order to form a site; you then associate the components with each other, and assign them a site name. Because the components are system-wide, they are re-usable—components that are used to form one site can be used to form other sites.

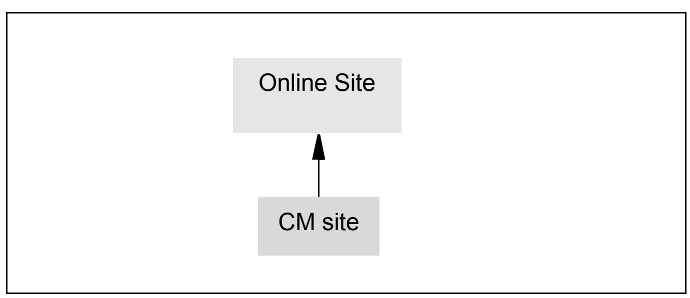
Note that to content providers, site components appear to be site-specific. However, any assets that content providers create on the site are indeed specific to that site. The assets can be copied to other sites, or shared among them.

The options of sharing and copying site components play an important role in the modeling and replication of online sites, as explained in the next section "Content Management Models."

Content Management Models

As the administrator, one of your biggest decisions, whether you make it alone or with collaborators, is how to model the online site in the Oracle WebCenter Sites interface: as a single site or a set of sites. Which content management model you choose depends largely on the size of your online site and the nature of its content.

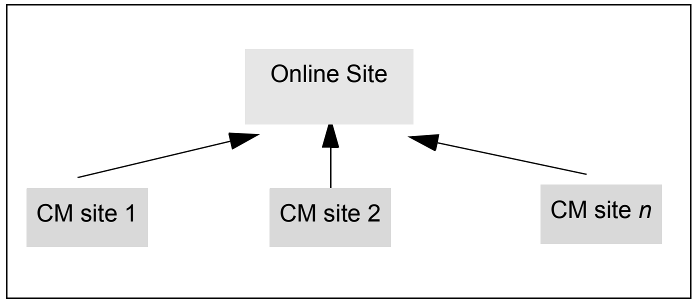
1:1 Model



The 1:1 model maps the online site directly to a single CM site. The CM site is the sole source of the online site.

The 1:1 model works efficiently for small online sites, where content tends to be limited, uniform, and managed by few users. For enterprise-level sites, however, pages often number in the millions and differ significantly in subject matter, content, presentation, and scope. Organizing large sites requires a model that can handle the complexities of the task.

1:*n* Model



The 1:*n* model maps the online site to multiple CM sites. Each section of the online site, whether logical or physical, maps to a certain CM site as its source. The CM sites can function independently of each other or overlap each other by sharing components and content.

For example, in a certain catalog site, content contributors who enter data about household goods never enter data about yard goods, so two separate sites are used to represent household goods and yard goods. Similarly, in a publication site where writers work independently of each other, the sports writers have a site that represents the sports news section, while the financial writers have a separate site that represents the financial news section. In each design, two sites contribute to one online site.

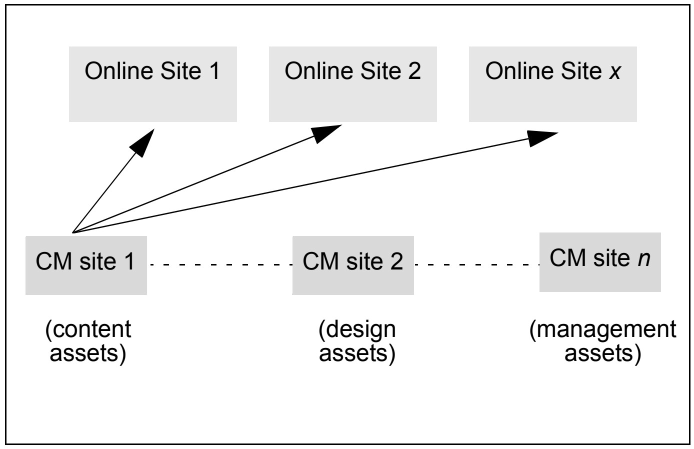
In a 1:*n* model, each site is unique; it has its own content types, templates, content providers, roles, workflow processes, and publishing mechanisms. Each site is the source of content for its corresponding online section.

When using a 1: *n* model, administrators have the option to configure sites to be independent of each other or to overlap each other by sharing components and content. An independent model works well when certain content needs to be segregated from other content. For example, the content is sensitive and must be handled only by certain content providers; or the content is so dissimilar that it needs to be handled by different specialists.

Overlapping sites, on the other hand, are used to support collaborations among remotely related content providers, such as those residing in different departments, or business units.

For example, consider the following scenario. A growing e-business specializing in mountain climbing disseminates mountain-climbing news and publishes a photograph album. It also sells gear and answers FAQs. The online site is clearly segmented into four sections: news, photos, gear, and FAQs. On many occasions, however, the news and photo sections share photographic content. In this scenario, one of your options is to create four sites: news, photos, gear, and FAQs such that news and photos share content. (Another one of your options is to create three independent sites: one for news and photos, one for gear, and one for FAQs.)

x:*n* Model



The *x:n* model maps multiple online sites to multiple CM sites. Of the three possibilities, this model is the most complex, but offers the greatest advantage to enterprise-level e-businesses.

In this model, content assets are localized on a given site, while the design and management assets are configured on separate sites from which they are shared to all other sites. This type of design allows the administrator to separate content from presentation and business logic.

Note

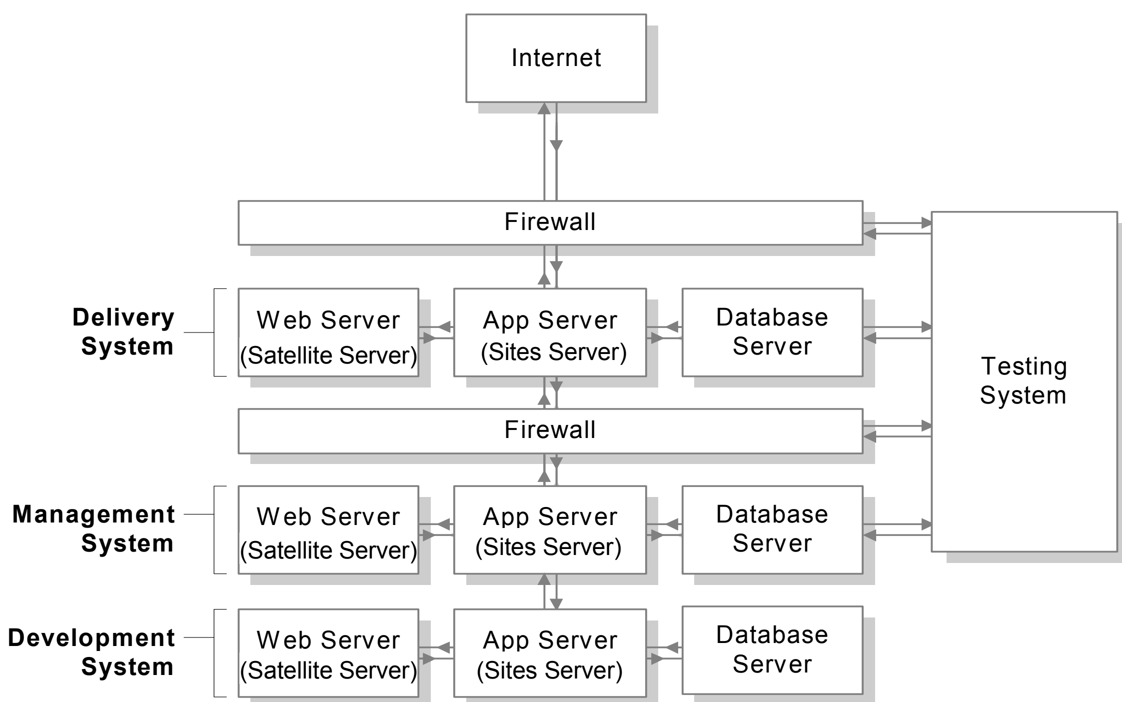
To help administrators and other users understand sites and their relation to the online site, Oracle WebCenter Sites provides sample sites named *avisports* and *FirstSiteII*. Each sample site maps 1:1 to its online site. If the sample sites are installed, you can log in to the sites by following the login procedure in Chapter 2.

Oracle WebCenter Sites Environment

Site configuration is a collaborative process that takes place in a Oracle WebCenter Sites environment rather than an isolated Oracle WebCenter Sites system. An enterprise-level environment typically consists of four different Sites-powered systems: development, management, delivery, and testing, as shown in Figure 4. Each system runs on its own database and at some point interacts with the other systems. One of the systems, testing, is an optional system that large organizations typically install for increased quality assurance.

As the Sites administrator, you will work with all of the systems at one time or another. Your collaborators are developers, who work exclusively at the development system with you and with users whom you appoint as collaborators. This section summarizes the possible systems and your function in each of them.

Figure 4: Oracle WebCenter Sites Environment



> **Note**
>
> The names of the systems in your Sites environment might vary from the names used in this guide. Generally, the management system is also called “staging”; the delivery system is also called “production”; and the testing system is also called “QA” or “QA testing.”

- The development system is responsible for planning and creating the framework of the online site: the data model that will be used by the CM sites and the data presentation templates.
- The management system is the staging area, responsible for configuring site components (other than the data model), assembling the components (including the data model) into sites, making the sites available to users, and managing users’ day-to-day activities on the sites. The users’ responsibility is to develop and manage content for delivery to the online site.
- The delivery system is the production area, which receives content for the online site from the management system and serves it to the target audience.
- The testing system is where QA tests both the management and delivery systems, and the online site itself before its launch. When the testing system is absent, the development system doubles as the testing system.

The rest of this chapter describes your collaborators, your planning strategies, and your functions at each of the systems.

Administrator’s Job

Collaborating

Before development begins you typically collaborate with a team comprising many different specialists:

- Site designers
- XML and JSP developers
- Java application developers
- Database administrators
- System network administrators
- Marketers and advertising staff
- Business managers
- Product managers, if you are developing a commerce site
- Content providers

The job of the team is to establish functional requirements and design specifications for the management and delivery systems:

- Page design
- Caching strategy
- Security strategy
- Format vs. content
- Data model
- Content management model

Much of the preceding information is outlined in Chapter 1 of the *Oracle WebCenter Sites Developer’s Guide*. The information is presented in this *Administrator’s Guide* in the context of the administrator’s job.

Planning

Planning is critical, as the decisions that you and your collaborators make will determine how developers will code system security, the data model, and the delivery system. Your job, in particular, concerns the management system, as described below.

During the planning stage, your job is to establish with collaborators (mostly business managers) the requirements and design of the management system, and relay the information to developers. Items to consider are listed below. The list is not meant to be exhaustive, but to help you start gathering and organizing information that is most critical to a successful start of the development cycle.

- Content management model
  - Determine which of the models (1:1, *n*:1, and *x*:*n*) is most appropriate for your operation
  - Determine which content assets must be shared and which must copied. Do the same for template assets.
- Site replication

  Determine whether you will be replicating sites. If so, you must configure the source sites to conform to the requirements that are set by the replication process.

  Developers must know your decision in order to code correctly. For example, if sites are to be replicated, the names of templates that will be copied to the new sites must not be hard-coded.
- Data modeling
  - What types of assets should be created
  - Which asset types should use revision tracking
  - Which users must have access to which asset types? on which sites?
  - What types of templates need to be created by developers
- User management methods

  In addition to its native user manager, Oracle WebCenter Sites supports LDAP plug-ins, both hierarchical and flat-schema. Determine which method is appropriate for your environment. (Procedures in this guide use Oracle WebCenter Sites' native user manager system throughout.)
- User management models

  Determine the following:
  - How many users are needed
  - Who the users are
  - What permissions the users must be given to the database tables:
    - Which system ACLs the users must be assigned (without ACLs, no user accounts can be created).
    - Whether custom ACLs need to be created.

      (Note that you may also need to create ACLs to assign to visitors of the online site in order to restrict them from accessing certain database tables.)
  - How many roles are needed

    - What types of roles are needed (writers, editors, illustrators). Make sure that the roles are compatible with the users' ACLs.
    - To which users the roles should be assigned.
- Workflow processes

  Workflow processes can be optionally attached to asset types to ensure that newly created assets of those types are automatically engaged in workflow.
  - Should workflow be implemented
  - If so, which asset types need workflow processes
  - How must the workflow processes be designed
  - Which users and roles need to participate in the workflow processes
- Publishing system options
  - Which type of publishing needs to be configured
  - Whether publishing schedules need to be approved
- User interface options

  Determine which interfaces the users will need:
  - Contributor (Web Mode and Form Mode)
  - Admin
  - WEM Admin
  - Sites Desktop
  - DocLink

Use both this book and the *Oracle WebCenter Sites Developer's Guide* to help you make these decisions.

Development

When the specifications of the online and sites are established, you are ready to begin your work. As the Oracle WebCenter Sites administrator, you will be working on all systems in the Sites environment during the development stage. (Once the online site is running, however, you will spend most of your time at the management system, a smaller fraction at the delivery system, and perhaps none at all at the development system.)

At each system, you have a specific set of jobs. While developers are responsible for the data model and other code-based components, you are responsible for all other aspects.

At the Development System

The development system is where coding takes place.

- Your job is to provide developers with the specifications of the management system and to assist with content management operations, such as creating sites.
- The developers' job is to:
  - Create sites with the same names as those that will be used on the management system
  - Code the data model (bearing in mind site-replication requirements:

    - Content asset types
    - Design asset types
    - Management asset types
  - Create sample assets of each type.
  - Test the data model.
  - Code templates and, in general, the framework of the online site.

Once the data model is complete and tested to the satisfaction of all collaborators, developers migrate the code to the management system.

Note

In your WebCenter Sites environment, developers might choose to not migrate the data model, but to re-create the data model on the management system.

Note that once development is complete, the development system continues to operate. One of its ongoing functions is to revise the data model in response to the evolving needs of online site visitors, content providers, and administrators.

At the Management System

Configuration tasks that require no coding are typically completed at the management system. Here, your administrative work is not collaborative unless you choose to make it so. For example, because no data modeling or coding takes place at the management system, you have no need for assistance from developers. However, you can appoint developers (or other users) to be the site or workflow administrators, to manage specific sites or workflow processes for the system.

Note

On occasion, you will need help from developers; for example, you might want to add custom functionality to the tree tabs in Oracle WebCenter Sites' interface.

At the management system, your job is the following:

- Create the sites

Note

In this release of Oracle WebCenter Sites, you can quickly spin off new sites by using the Site Launcher feature to replicate source sites. You can then modify the replicates as necessary. For information about site replication and selecting or creating the appropriate source sites, see Chapter 10, "Replicating CM Sites."

- Configure the users
  - Create their ACLs in Oracle WebCenter Sites, as necessary

  - Create user accounts, either through Oracle WebCenter Sites, or external user managers, such as LDAP plug-ins
  - Create roles in Oracle WebCenter Sites, as necessary
  - Assign users their roles for each site
- Assemble the sites by
  - Associating the users with the sites
  - Associating the correct asset types and publishing system with the sites
- Enable the users by
  - Assigning users their roles for each site
  - Using roles to associate users and asset types with start menu items
  - Using roles to associate users with interface functions such as tree tabs
  - Creating tree tabs, as necessary
- Create and manage workflows
- Enable revision tracking
- Configure site and workflow administrators, as necessary
- Configure the publishing process
- Configure the users' interfaces, as necessary

At the Delivery System

The delivery system is configured by developers and typically does not involve an administrator until the system is ready for use. For Mirror to Server and RealTime delivery, the delivery system is the entire Oracle WebCenter Sites installation. For Export to Disk and Export to XML delivery, the delivery system is simply the web server component of the Oracle WebCenter Sites installation.

At the Testing System

The testing system is where QA analysts test the performance of both the management system and the delivery system, as well as the online site itself before its launch. If the testing system is absent, the development system doubles as the testing system. Ordinarily, you are not involved in testing.

Implementation

When the management and delivery systems are in use and the online site is running, you work almost exclusively at the management system, where you:

- Manage the users
  - Adjust their permissions to database tables
  - Add, delete, and re-organize users
  - Create workflows
  - Configure the users' interfaces
- Manage the publishing system
- And otherwise respond to the needs of online site visitors, content providers, and the e-business

The Focus of This Guide

This guide focuses on the configuration tasks that you complete at the management system during the development stage, as described in "Development," on page 40. It is assumed that the data model resides on the management system by the time you begin your configuration tasks.

Administrator's Prerequisites

To function in the jobs that were described in the previous section, you must be equipped with a certain amount of information. This section summarizes the information you need to know or be familiar with.

- Technical aspects of content management

  Although you are not required to have a specific technical knowledge of programming and system architecture, a basic knowledge of the following is desirable:

  - Java
  - JSPs
  - Publishing
  - Databases and database management
  - Browsers

- Your collaborators

  For detailed information, see "Collaborating," on page 38 and the *Oracle WebCenter Sites Developer's Guide*.

- Online site specifications

  Online site specifications are typically prepared by site designers or the corporate communications department to define the site's audience, the site's size, its message and goals, the nature of the content, and the presentation style. Having an understanding of the online site will help you model the online site, plan sites, and collaborate efficiently with developers.

- The Oracle WebCenter Sites environment

  As the Oracle WebCenter Sites administrator, you can work with up to four different systems: development, management, delivery, and testing. Make sure you have a good understanding of the systems and your function at each of them. For detailed information, refer to the sections "Oracle WebCenter Sites Environment" and "Administrator's Job" (both in this chapter), and the *Oracle WebCenter Sites Developer's Guide*.

- Oracle WebCenter Sites' administrative interface and system defaults

  The administrative interface provides you with site configuration tools, and access to system defaults such as ACLs, roles, and asset types. For information about the administrator interface, see Chapter 2, "Administrator's Interface." For information about system defaults, see Appendix A, "System Defaults" and Appendix B, "System Data: WebCenter Sites Database."

Chapter 2

Administrator's Interface

Oracle WebCenter Sites defines three kinds of administrators, each with a different range of administrative rights. This chapter summarizes the administrators and shows you how to log in to Oracle WebCenter Sites as the general administrator. It describes the administrator's interface, its tabs, the nodes within the tabs that allow you to manage sites, and important system defaults that are accessible either from the interface or the back end.

This chapter contains the following sections:

Types of Sites Administrators

Oracle WebCenter Sites defines three types of administrators:

- The general administrator, for whom this guide is written.

  The general administrator typically manages all systems in the WebCenter Sites environment and therefore has unrestricted access to each system's interfaces. The jobs of the general administrator at each system are outlined in Chapter 1, "Overview."
- Site administrators, who administrate only certain sites on the management system.
- Workflow administrators, who create workflow processes on the management system

Each administrator is given permissions to relevant parts of the Sites interface.

Procedures in this guide are specific to the general administrator. Whenever steps can be executed by the site or workflow administrators, this guide indicates which steps.

Finally, this guide is not restricted to any particular system in the Sites environment. The concepts and procedures it contains are valid at all systems in the Sites environment.

Note

As this guide is written for the general administrator, the word "you," whenever it is used, refers strictly to the general administrator.

Logging in to the Administrator’s Interface

This section shows you how to log in to Oracle WebCenter Sites as the general administrator.

> **Note**
>
> If you are *not* the general administrator and need to log in as a site administrator or workflow administrator, obtain your user name and password from the general administrator before following the steps below. Also bear in mind that because your permissions to the Oracle WebCenter Sites interfaces are limited by your role, only certain parts of this guide are relevant to you.

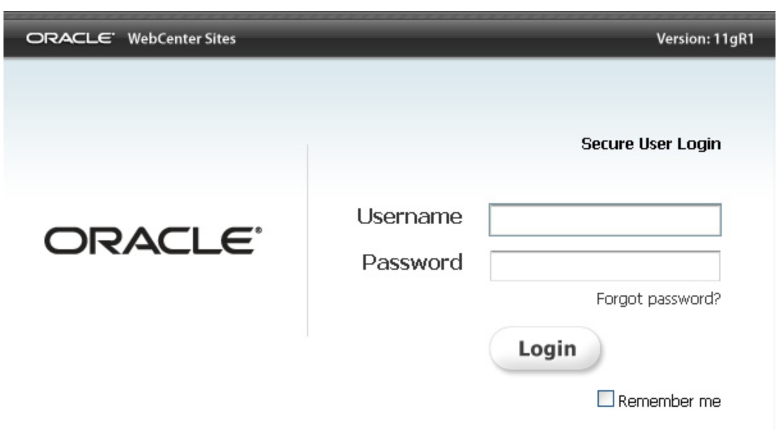
To log in to Oracle WebCenter Sites as the general administrator

1. Open your browser.
2. Set your browser so that it checks for newer versions of stored pages with every visit to the page.
3. Navigate to the following URL:

   http://*<server>*:*<port>*/*<context>*/login

   where *<server>* is the host name or IP address of the machine running Oracle WebCenter Sites, *<port>* is the port number for the server, and *<context>* is the context path of the Oracle WebCenter Sites web application. Depending on how the system was set up, you might also need to include the port number. The system can be development, management, delivery, testing, or another system in your WebCenter Sites environment.

   Oracle WebCenter Sites displays the following log in form:



4. Enter your user name and password.

> **Note**
>
> If you are the general administrator of a new installation that had no previous administrator, log in as the default system user, with user name **fwadmin** and password **xceladmin** and then immediately change the password. For instructions on changing your password, follow the procedure in "Creating a New User," on page 86.
>
> Because you have logged on as the default administrator, your user name has the `xceladmin` ACL (Access Control List) assigned to it, in addition to being a part of the REST_Admin security group. See Chapter 5, "Configuring Users, Profiles, and Attributes" for information about user accounts and their ACLs.
>
> If you create any other general administrators, they must have the `xceladmin` ACL assigned to them, and they must be part of the REST_Admin security group.

5. Click **Login**.
6. WebCenter Sites responds according to the number of sites that have been created in its interface. To continue, complete one of the following steps:
   - If no sites have been created, you must log in to the AdminSite, which will be listed in the drop-down list. (Note that the situation of no sites is rare. Most new installations run the sample sites that are provided with Oracle WebCenter Sites.) Continue with step 7.
   - If one site has been created, you still need to select the site from the list to log in to that site. Continue with step 7.
   - If two or more sites have been created, they are listed in your Sites interface, along with the sample sites that your installation engineer has elected to install.
     a) Select the site that you want to work on.
     b) Continue with step 7.

7. Once a site is selected, you will see two icons appear. The left icon is to log in to the Admin interface, and the right icon is to log in to the Contributor interface.

Click to enter the Admin interface.

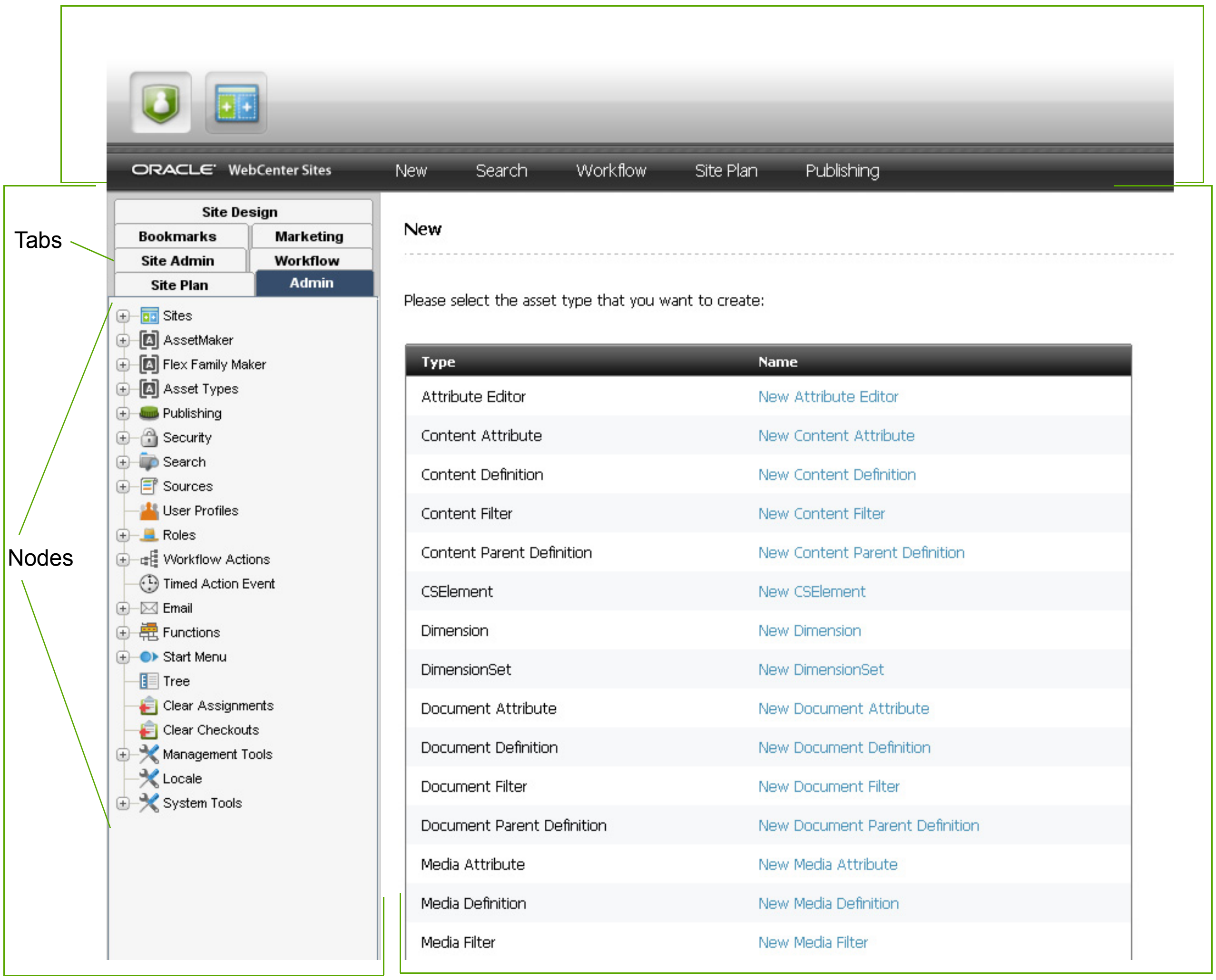
If you select AdminSite, the icons will be slightly different.

8. At this point, you are logged in to the site of your choice. All the tasks that you complete pertain to the site, until you switch to a different site. The one exception is the set of administrative tasks that you execute from the “Admin” tab, which is not site specific. The “Admin” tab contains data for all sites in the system.

Within your selected site, Oracle WebCenter Sites displays the administrator’s interface, consisting of the three frames shown in the next figure:

Top Frame: Button bar

Displays icons and buttons for navigating WebCenter Sites.



Left Frame: WebCenter Sites Tree

The WebCenter Sites tree is a collection of tabs with nodes. Each node is a building block of the site.

Right Frame: CM Site-Building Form

This frame displays the forms that WebCenter Sites calls when you select nodes from the left frame. Each form is used to enter data into the selected node and manage the data.

Note

At this point, the first time you log in after selecting a site, you will be prompted to create a new asset type, with the “New” screen displaying.

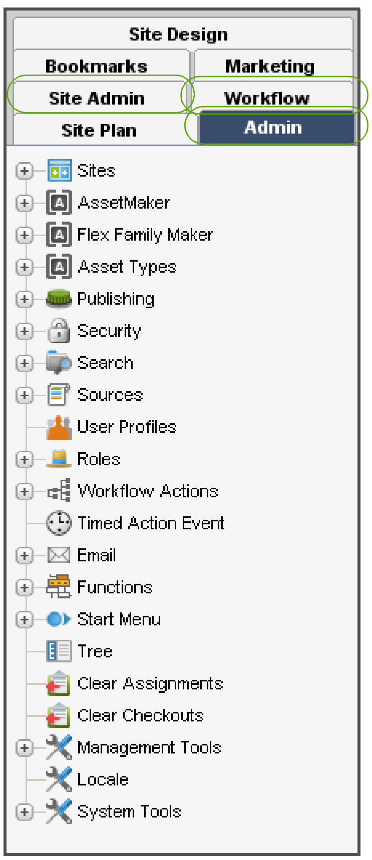
If you are a first time Sites user, we recommend that you familiarize yourself with the following features:

- The administrator's interface—the "Admin" tab in particular. You will be using the "Admin" tab almost exclusively throughout this guide to create, configure, and manage sites. For more information, see "Administrator Tabs," on page 51.
- System defaults, such as ACLs, which you will need in order to configure users, their access rights, and so on. For more information about system defaults, see Appendix A, "System Defaults."

Administrator Tabs

Up to three administrator tabs can be displayed in Oracle WebCenter Sites' tree:

- Admin
- Site Admin
- Workflow



The tabs are permissions-based. To be exposed to the user, they require the user to have the ACLs and roles shown below:

Tab	ACL	Role
Admin	xceladmin	GeneralAdmin
Site	xceladmin	SiteAdmin
Workflow	xceladmin	WorkflowAdmin

The ACLs and Roles are covered in further depth in “Configuring Users, Profiles, and Attributes,” on page 83, and “System Defaults,” on page 521.

The tabs are also conditional on the type of site in which you are working:

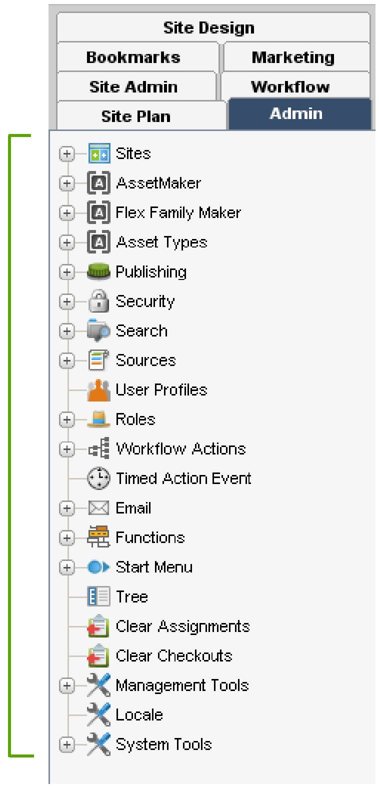
- If you log in to a Oracle WebCenter Sites installation where no sites have been created, you are required to log in to the AdminSite site. Here, you see only the “Admin,” “Site Admin,” and “Workflow” tabs.
- If you are logged in to a custom site, both the “Admin” and the “Site Admin” tabs are displayed in your interface. The “Admin” tab remains system-wide. The “SiteAdmin” tab is specific to the site you are logged in to. It is therefore a sub-set of the “Admin” tab.
- If workflow processes have been created for any site (except the default site), the “Workflow” tab is also displayed in your interface. The “Workflow” tab lists all workflow processes that have been configured for all sites in the system.

Whether administrator tabs are exposed to other users depends on whether you assign the users the associated ACLs and roles. For more information about ACLs and roles, see Chapter 4, “Working with ACLs and Roles.”

‘Admin’ Tab

The “Admin” tab is the main administrative tab, used to create, configure, and manage sites. The “Admin” tab accounts for all content management components in the Oracle WebCenter Sites installation.

The “Admin” tab displays the following nodes:



- **Sites**, that is, CM sites (described in Chapter 1, “Overview”).

  The **Sites** node has the following functions:

  - Lists sites that have been created in the system.
  - Enables you to define sites and associate to them the components from the system-wide configuration pool, below the “Sites” node.

    Oracle WebCenter Sites provides several sample sites as learning tools.

  The configuration pool consists of the following nodes:

- **AssetMaker**, for site developers to create asset types using the basic asset data model. See the *Oracle WebCenter Sites Developer’s Guide* for information.
- **Flex Family Maker**, for site developers to create new flex families and flex asset types. See the *Oracle WebCenter Sites Developer’s Guide* for information.
- **Asset Types**, for configuring asset types:
  - To add subtypes or asset associations for basic asset types.
  - To create start menu items for asset types that appear on the **New** and **Search** menus.
  - To enable or disable revision tracking for specific asset types. See Chapter 24, “Revision Tracking” for information.
  - To enable assets for Sites Desktop. For information, see Chapter 22, “Configuring the User Interfaces.”
  - To enable assets for Sites DocLink. For information, see Chapter 22, “Configuring the User Interfaces.”
- **Publishing**, for setting up the publishing system on the development and management systems. See Part 3, “Export and Mirror Publishing” for information about static and dynamic publishing. See Part 4, “RealTime Publishing” for information about interactive dynamic publishing.
- **Security**, for configuring the security groups and assigning users to those groups. Adding someone to the security groups is tied to the WebCenter Sites interface. See Chapter 6, “Setting Up External Security” and *Oracle WebCenter Sites Developer’s Guide* for information.

- **Search**, for configuring asset types so that they are indexed correctly when you are using one of the supported third-party search engine modules. See Chapter 23, "Configuring the Lucene Search Engine" for information.
- **Sources**, which you or your site developers use to add new content sources if your asset types are designed to use them. See the *Oracle WebCenter Sites Developer's Guide* for information.
- **User Profiles**, which you use to specify e-mail addresses for your users. See Chapter 5, "Configuring Users, Profiles, and Attributes" for information.
- **Roles**, which you use both for determining the recipients of workflow assignments and to control users' access to functions in the Oracle WebCenter Sites interfaces.
- **Workflow Actions**, which you use to create the workflow building blocks called actions and conditions. These items can be used by any workflow process. See Chapter 9, "Creating and Managing Workflow Processes" for information.
- **Timed Action Event,** which you use to configure the frequency with which due dates are calculated for assets that are in workflow. See Chapter 9, "Creating and Managing Workflow Processes" for information.
- **Email**, which you use to create the workflow e-mail messages that can be used by any workflow process. See Chapter 9, "Creating and Managing Workflow Processes" for information.
- **Functions**, which administrators can enable or disable in the Oracle WebCenter Sites interfaces of all users or selected users.
- **Start Menu**, which you use to create links in the form that is displayed when users click the **New** button or **Search** button in the Oracle WebCenter Sites interface. You use the links to create new assets or search for existing assets. Start Menu items can also be created for managing access to the Sites Desktop and Sites DocLink interfaces. See Chapter 8, "Managing Access to CM Site Components" for information.

Note

This release of Sites supports automatic creation of the **New** and **Search** Start Menu items.

- **Tree**, which you use to control access to the existing tree tabs, create new ones, or view assets in the tree in the Contributor interface. See Chapter 8, "Managing Access to CM Site Components" for information.
- **Clear Assignments**, which you use to clear workflow assignments. See Chapter 9, "Creating and Managing Workflow Processes" for information.
- **Clear Checkouts**, which you use to unlock and check back in to the database the assets that are checked out by specific users. See Chapter 24, "Revision Tracking" for information.
- **Management Tools**, which you use to create ACLs and users, to enable or disable revision tracking for non-asset tables, and to perform various other operations on WebCenter Sites' database. For information, see Chapter 4, "Working with ACLs and Roles," Chapter 5, "Configuring Users, Profiles, and Attributes," and Chapter 24, "Revision Tracking."

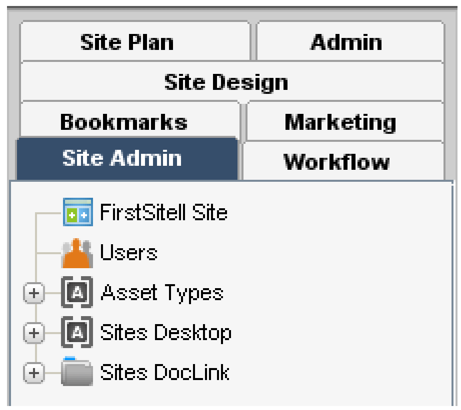
- **Locale**, which you use to set the default locale for the Sites system. See Chapter 22, "Configuring the User Interfaces" for information.
- **System Tools**, which you use to troubleshoot problems directly from the administrator's interface. Features include configuring log4j logging, accessing various types of system information, managing caches, searching logs, and testing the performance of the shared file system. See Chapter 28, "System Tools" for information.

'Site Admin' Tab

The "Site Admin" tab is site-specific. It is used to manage the site that you are logged in to.

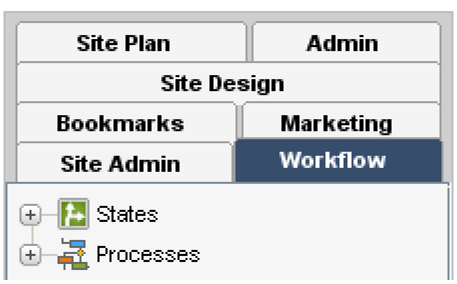
The "Site Admin" tab holds a subset of the administrative functions in the "Admin" tab. It allows you to manage existing users of the site, as well as enable or disable asset types. It does not allow you to create users, asset types, or sites.

For any other user to access this tab, the user must be assigned the `xceladmin` ACL and the **SiteAdmin** role for the site(s) that the user logs in to.



'Workflow' Tab

The "Workflow" tab is used to configure workflow processes from the workflow building blocks that were created using the "Admin" tab.



Note

While the "Workflow" tab itself is site-specific, its content is not.

The tab. When a user selects a site, the "Workflow" tab is displayed if the user is assigned the `xceladmin` ACL and WorkflowAdmin role *for that site*. In this respect, the "Workflow" tab is site-specific.

The tab's content. The "Workflow" tab displays system-wide information, that is, all workflow processes for all sites in the installation. Therefore, the content of the "Workflow" tab is not site-specific.

Non-Administrative Tabs

Your interface displays default non-administrative tabs, and custom tabs if they have been configured by another administrator. You can create additional tabs as necessary.

For information about the default tabs, see "Default Tree Tabs," on page 532. For information about creating and configuring tabs, see "Managing Access to the Tree (Administrator's interface only)," on page 142.

System Defaults

WebCenter Sites has a number of system defaults:

- Non-administrative tabs, described in the previous section
- ACLs
- Roles
- Users
- Asset types
- System Tables

You must be familiar with the defaults in order to configure users and sites. Information about system defaults is given in Appendix A, "System Defaults" and Appendix B, "System Data: WebCenter Sites Database."

Sample Sites

To help you better understand WebCenter Sites, we have packaged it with sample sites named *avisports* and *FirstSiteII*. You can use the sites to learn about and experiment with WebCenter Sites without affecting your own sites. The sample sites vary in complexity and purpose. In addition to providing code samples for developers, they exemplify how sites are configured.

If your installation engineer has elected to install the sample sites, they will be displayed in your interface.

To work on the sample sites, follow the steps in "Logging in to the Administrator's Interface," on page 47. You can access the corresponding online sites by pointing your browser to each site's URL, available from your installation engineer.

Chapter 3

Site Configuration Guidelines

Site configuration in WebCenter Sites is based on a relational system, where components in a system-wide configuration pool must be associated with each other and with a site name in order to function as a site. Making the correct associations creates a functional site—one that hosts the correct users, provides them with the correct asset types for authoring and managing content, supports workflow as necessary, publishes correctly, and delivers approved content to the intended audiences.

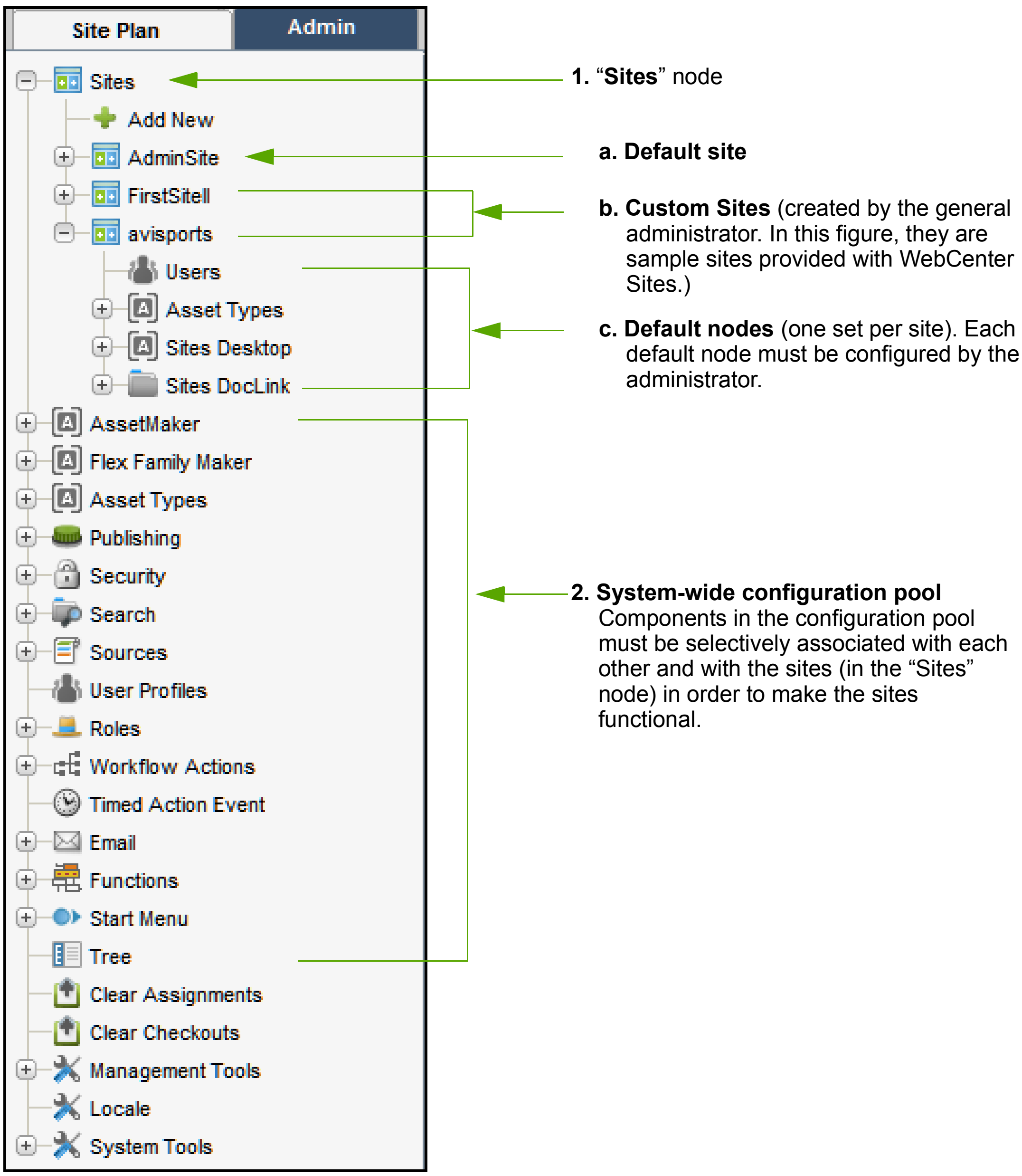
This chapter contains procedures that show first-time administrators how to configure sites. It also contains recommendations for the more experienced administrators to help them manage and maintain established sites.

This chapter contains the following sections:

Overview

When creating and configuring sites, you will be using the "Admin" tab, shown in the figure below. The tab is logically separated into two sections: (1) the "Sites" node, where sites are first established, and (2) the system-wide configuration pool, where site components are created, associated to each other, and managed. The configuration pool begins with the "AssetMaker" node and ends with "Management Tools."

Initially, when a new site is defined and added to the "Sites" node, it is empty and non-functional, because it is not associated in any way with the configuration pool. You make the site functional by associating with it the various components in the configuration pool. The components are sharable among sites. Any component that you associate with one site, you can associate with as many other sites as necessary. More information about the "Sites" node and the configuration pool is given on the next page.



Sites are configured in four *basic* steps from the “Admin” tab, as shown below. We emphasize *basic* to indicate that the steps are intended to provide only an overview of the site configuration process. Detailed procedures are given in the rest of this guide.

Note

In the procedure below, steps 1 and 2 are interchangeable.

Step	Description
1. Add sites to WebCenter Sites: **a.** In the **Admin** tab, expand **Sites** and double-click **Add New**. **b.** Enter the site name and description. **c.** Select a preview method by which you will preview the site’s content (when the content is created).	The “Sites” node (rather than the configuration pool) can be the starting point for creating sites. When you execute step 1, WebCenter Sites adds the site to the “Sites” node. The Contributor and Admin interfaces are automatically assigned to each newly added site. Under the newly added site, WebCenter Sites creates a set of sub-nodes that prompt you for site components: • The “Users” sub-node prompts you for users. • The “Asset Types” sub-node prompts you for asset types (created by developers). • The “Sites Desktop” sub-node prompts you to enable Sites Desktop. • The “Sites DocLink” sub-node prompts you to enable Sites DocLink. Each sub-node is equipped with a linking mechanism that allows you to associate the sub-node, and therefore the site, with data in the configuration pool. Because of this linking mechanism, the sub-nodes cannot be deleted or supplemented with custom sub-nodes. All data required by the sub-nodes must be entered in to the configuration pool, described next.
2. Populate the configuration pool with components, such as users, asset types, and roles.	The configuration pool begins with the **AssetMaker** node and ends with **Management Tools**, as shown in the figure on page 58. The configuration pool enables you to select and create components for the site and to associate the components with each other. Data that you establish in the configuration pool remains independent of any sites (and therefore inaccessible to users) until you associate the data with the sites.

Step	Description
3. Associate and enable components in the configuration pool with the site: **a.** In the **Admin** tab, expand **Sites** and double-click the desired site. **b.** Add users and asset types to the site. **c.** Enable the users and asset types by means of start menu items, access permissions, and permissions to tree tabs.	The "Sites" node is the endpoint of the site building process. In this step, you once again use the "Sites" node, this time to associate components in the configuration pool to the site that you added in step 1. Note that components in the configuration pool are reusable; they can be associated with as many sites as necessary.

Site Replication Option

If you plan to create many sites that are similar in content and structure, you can shorten the site creation process. You can configure the appropriate source sites, then replicate the sites as often as necessary, and finally modify the replicates as necessary. The source sites must conform to certain replication requirements. For more information, see Chapter 10, "Replicating CM Sites."

Site Configuration Steps

In the site configuration process, certain components in the configuration pool are required by other components and must be created first if re-tracing of steps is to be minimized. For example, before users can be created, their ACLs must first exist.

In general, the order of steps in the site configuration process is flexible. One possible order is given in the procedure below (starting with "Assumptions"), where the critical site components are created first, then assembled into a site, and finally enabled for use; the optional components (with the exception of workflows) are considered last, after the publishing process is configured.

If you are a first-time administrator, we suggest that you familiarize yourself with the procedure below in order to obtain a basic understanding of the configuration process and gather all the information that you will need. Once you understand how sites are configured, you can modify the procedure to suit your needs.

We also suggest that you make a copy of the procedure below and keep it handy when configuring sites. Each step refers you to sections of this guide (or the *Oracle WebCenter Sites Developer's Guide*) where you can find detailed information about the particular step.

Assumptions

We assume that you and developers will code the data model at the development system, and after successful testing, migrate the data model to the management system. At the management system, you will complete the site configuration tasks, test the management system, and finally deploy the system. Before beginning the configuration tasks, you should read Chapter 1, “Overview” for a general understanding of WebCenter Sites concepts, components, and content management models.

Pre-Configuration Decisions

Step 1. Determine the user management method

Your options are WebCenter Sites’ native user manager, an LDAP plug-in, or an external manager.

While this guide is based on the native WebCenter Sites user manager, it does describe system behavior for LDAP-integrated systems. For information, see Appendix C, “Managing Users, Sites, and Roles in LDAP-Integrated Sites Systems.” For information about configuring users in LDAP, refer to the LDAP product documentation.

Step 2. Determine users’ table-level permissions (ACLs)

Regardless of which user management method you will be implementing, you must have ACLs defined in WebCenter Sites. Examine the list of system ACLs (and custom ACLs, if any have been created) to determine which ACLs to assign to users, and whether you need to create additional ACLs. Typically, the system ACLs are sufficient. For general information about ACLs, see Chapter 4, “Working with ACLs and Roles.” For specifications, see Appendix A, “System Defaults.”

Step 3. Determine the users’ roles

WebCenter Sites defines several system roles and additional roles for the sample sites. You can reuse the roles for your sites, or create your own. Typically, you will need to create roles to cover the full range of users’ responsibilities on any given site.

For information about roles, see the following:

- Chapter 1, “Overview” and the section “Working with Roles,” on page 81 for general information.
- Appendix A, “System Defaults,” for the list of system roles and their specifications.

Step 4. Determine whether you will be replicating the sites you are configuring

If you plan to replicate the sites you are configuring, make sure that template-coding and other requirements are satisfied for the source sites (the sites you are currently configuring). For an overview of site replication as well as replication options and requirements, see Chapter 10, “Replicating CM Sites.”

Step 5. Determine system security

Before developers begin designing the online site, or contemplating changes to the user interface on the management system, you must determine and implement your security protocols. Decisions that you make about security affect the way that developers code and implement the online site. For information about security, see Chapter 6, “Setting Up External Security.”

Configuration Steps at the Development System

Configuration steps at the development system are also given in detail in "Oracle WebCenter Sites Development Process," Chapter 3 of the *Oracle WebCenter Sites Developer's Guide*. When coding at the development system is complete, the data model is migrated to the management system, where site configuration is completed.

Step 1. Create the sites that will be used on the management system

In this step, you will create sites with the names that will be used on the management system. For information and procedures on creating sites, see Chapter 7, "Assembling CM Sites" (in particular "Creating a Site," on page 106).

Step 2. Create the data model (asset types and their definitions)

For developers to create a data model or any site used on the management system, the developers must be created and configured, preferably as an administrative user, at the development system. For more information on this, see "Configuration Steps at the Management System," on page 63, and create administrators on the developers' system.

1. Create content asset types.
2. Create design asset types (templates), bearing in mind site-replication requirements, if any.
3. Create management asset types.
4. Create sample assets of each type.

For information about creating asset types, see the *Oracle WebCenter Sites Developer's Guide*.

Step 3. Test the data model

Step 4. Migrate the data model to the management system

To migrate the data model, publish it to the management system. For instructions, see Part 4, "RealTime Publishing."

Configuration Steps at the Management System

Step 1. Create ACLs, if necessary

For procedures on creating ACLs, see Chapter 4, "Working with ACLs and Roles" (in particular "Creating a New ACL," on page 73).

Notes

- Under no circumstances should you modify or delete a system ACL.
- **Database tables.** When the data model is migrated to the management system, assign ACLs to the custom database tables (asset types), as necessary. For procedures on assigning ACLs to database tables, see Chapter 4, "Working with ACLs and Roles" (in particular "Assigning ACLs to Custom Tables," on page 76).
- **WebCenter Sites page entries.** Typically there is no need to assign ACLs to WebCenter Sites page entries on the management system because you use roles to control access to assets and interface functions. However, if you need instructions for assigning ACLs to WebCenter Sites page entries, see "Assigning ACLs to WebCenter Sites Pages (SiteCatalog Page Entries)," on page 78.

Step 2. Configure the external user manager, if you are using one

For procedures on configuring an LDAP server, see *Oracle WebCenter Sites Configuration Guide for Supporting Software*.

Step 3. Create roles

For procedures on creating roles, see Chapter 4, "Working with ACLs and Roles" (in particular, "Creating a Role," on page 81).

Step 4. Create the users (the content providers)

For instructions on creating user accounts in WebCenter Sites, see "Creating a New User," on page 86. If you are creating user accounts in external managers, refer to the product documentation.

Step 5. Create user profiles, if necessary

User profiles are required for users who will be working with:

- Sites content applications (Engage, Analytics)
- Language packs and setting a default language
- Workflow processes, in which e-mail messages will be sent to notify workflow participants of their assignments. The user profile supports workflow actions by mapping a user name to an e-mail address.

User profiles must be created in WebCenter Sites. For instructions, see "Working with User Profiles and User Attributes," on page 93.

Step 6. Set user attributes, if necessary

If users require attributes beyond the locale and e-mail attributes that are specified in the user profile, you can create the attributes. For instructions, see "Modifying, Adding, and Deleting User Attributes," on page 94.

Step 7. Create workflow processes for asset types, as necessary

Implementing workflow processes is a business decision, and not a WebCenter Sites requirement. If you need to create workflow processes, do the following:

1. Create the "Workflow" tab for the site, if one does not already exist. For instructions, see "Creating a New Tree Tab," on page 144.
2. Plan your workflow process by sketching it. See Chapter 9, "Creating and Managing Workflow Processes" for help with this step. Then refer to your sketch and notes throughout this section.
3. If you have not already done so, create the roles that are needed for the workflow processes. in you need instructions, see "Creating a Role," on page 81.
4. Ensure that users are set up to participate effectively in the workflow:
   a. Ensure that users who will participate in workflow have user profiles created for them. Otherwise, they will not receive e-mail messages from the workflow process.
   a. For shared assets, if you want the pool of workflow candidates to include people from all the sites that an asset is shared to, enable the cross-site assignments feature. See "Cross-Site Assignments and Participants," on page 162 for details.
5. Create the e-mail objects that you need for your actions and enable the `xcelerate.emailnotification` property in the `futuretense_xcel.ini` file. For instructions, see Chapter 1, "Overview."
6. Create the step actions, timed actions, deadlock actions, group deadlock actions, and delegate actions that you need. For instructions, see "Setting Up the Workflow Actions and Conditions," on page 181.
7. If your states have deadlines, be sure to configure the Timed Action Event so that the deadlines of assets are calculated regularly and the appropriate timed actions (if any) are invoked in a timely way. For instructions, see "Setting Up the Timed Action Event," on page 184.
8. Create your states. For instructions, see "Setting Up the States," on page 185.
9. Create your process. While creating your workflow process, you create the steps for that process. The steps link together the states so they occur in the proper order. Additionally, while creating your process, you configure any function privileges that you need. For instructions, see "Setting Up the Workflow Processes," on page 187.
10. Test your workflow processes. For instructions, see "Testing Your Workflow Process," on page 197.

Set up your start menu shortcuts so that workflow processes are assigned automatically to assets when they are created. For help with start menu items, see "Creating Start Menu Items," on page 124.

Step 8. Re-create the sites that were created on the development system

Again, for information and procedures on creating sites, see *"Assembling CM Sites," on page 105* (in particular "Creating a Site," on page 106).

Step 9. Migrate the data model, if you have not already done so

At this point, you will need the data model in order to complete most of the remaining steps. Real-time publish the data model from the development system to the management system by using procedures in Part 4, "RealTime Publishing."

Step 10. Assign custom ACLs (if any) to the custom tables, if you have not already done so

For procedures on assigning ACLs to database tables, see Chapter 4, "Working with ACLs and Roles" (in particular "Assigning ACLs to Custom Tables," on page 76).

Step 11. Assemble the sites

1. Grant users access to the sites. For instructions, see "Granting Users Access to a Site (Assigning Roles to Users)," on page 112.
2. Enable asset types for the sites by associating the asset types with the sites. For instructions, see "Enabling Asset Types for a Site," on page 116.

> **Note**
>
> This step provides you with the option of allowing WebCenter Sites to automatically create "Start Menu" items for the asset types you are enabling. The "Start Menu" items **New** and **Search** items allow site users to create and search for assets of those types. You can also create your own "Start Menu" items, as summarized in the next step.

Step 12. Make site components available to users

1. If you have chosen to create your own "Start Menu" items for various asset types, go to "Managing Access to Asset Types via Start Menus," on page 122 for instructions.
2. Set access permissions on assets. For instructions, see "Setting Access Permissions to Assets," on page 134.
3. Give users access to the tabs in WebCenter Sites' tree. For instructions, see "Managing Access to the Tree (Administrator's interface only)," on page 142.

Step 13. Configure the Sites publishing process

1. Determine which type of publishing your management system is to execute:
   - RealTime
   - Export to Disk
   - Mirror to Server
   - Export Assets to XML

2. Configure the publishing process so the information that content providers create can be made available to the delivery system.

For general information and instructions on configuring the publishing process, see Part 3, "Export and Mirror Publishing." For information on the RealTime publishing process, see Part 4, "RealTime Publishing."

Step 14. Configure user interfaces, as necessary

The user interface options are Contributor, Admin, WEM Admin, Sites Desktop, and Sites DocLink. For instructions on making these interfaces available to content providers, see Chapter 22, "Configuring the User Interfaces."

Step 15. Enable asset types for search engines, as necessary

In some cases, such as upgrades, it may be necessary to configure the asset types to be searched so that they can be indexed correctly and found by your search engine. For instructions on configuring the asset types, see Chapter 23, "Configuring the Lucene Search Engine."

Step 16. Set up revision tracking, as necessary

Revision tracking prevents assets from being edited by more than one user at a time. When you enable revision tracking for a database table, WebCenter Sites maintains multiple versions of the assets in that table. For instructions on setting up revision tracking, see Chapter 24, "Revision Tracking."

Step 17. Set up the delivery system

For more information, see "Oracle WebCenter Sites Development Process," Chapter 2 in the *Oracle WebCenter Sites Developer's Guide*.]

Step 18. Test all systems

Step 19. Give users the URL to the Sites installation and their log in information

Part 2
CM Site Configuration Procedures

This part contains detailed procedures for configuring and replicating sites on your WebCenter Sites system.

This part contains the following chapters:

Chapter 4

Working with ACLs and Roles

Several user management components are used to control access to a WebCenter Sites system: ACLs, user accounts, user profiles, roles, and sites. Site configuration, itself, begins with ACLs, with the creation of user accounts, and with the creation of roles.

ACLs can be used on both your management and your delivery systems. However, the main focus of this chapter is user management on the management system. For information about user management on your delivery system, see the "Site Development" section in the *Oracle WebCenter Sites Developer's Guide*.

This chapter contains the following sections:

Overview

ACLs and roles are of paramount importance in WebCenter Sites.

- ACLs are used to regulate entry in to the WebCenter Sites system. The assignment of ACLs to users, database tables, and WebCenter Sites Pages determines users' permissions to operate on WebCenter Sites' database tables. If the user's ACLs match the database ACLs, the user has certain permissions (defined by the ACLs) to operate on the database tables. ACLs therefore serve as the foundation of the security and user management model. Without ACLs user accounts cannot be created.
- Roles are used to manage access to sites and their components. The assignment of roles to users and interface functions on a given site determines whether the interface functions are enabled for the users or hidden from them. If the user's roles match the roles that are assigned to the interface functions, the functions are enabled for the user. Otherwise, the functions are hidden.

Before you can establish users you must determine which ACLs and roles the users need to be assigned. If the ACLs and roles do not already exist, you must create them in WebCenter Sites, even if you intend to use LDAP plug-ins to create the user accounts.

This section provides you with a basic explanation of ACLs and roles, and shows you how to create, modify, and delete ACLs and roles.

ACLs

Access Control Lists, called **ACLs** for short, are named sets of database operation permissions such as read, write, create, and retrieve. Because just about everything in WebCenter Sites (and the WebCenter Sites content applications) is represented as one or more rows in one or more database tables, user management on any of your Sites systems starts with ACLs.

With ACLs, you can limit access to the following items:

- Individual database tables
- Individual WebCenter Sites pages

WebCenter Sites and the Sites content applications use ACLs to enforce access restrictions to the various functions of those applications by controlling the user's access rights to the database tables that represent those functions. How? By verifying that the users attempting the function have the same ACL assigned to their **user accounts** as are assigned to the database table.

For example, user account information is contained in the system tables named `SystemUsers` and `SystemUserAttrs`. Because certain ACLs are assigned to those system tables, only a user with the same ACLs assigned to his or her user account is able to create new users or edit existing user information.

ACLs serve as the foundation of the security and user management model in your Sites system by providing authorization functionality. Even if you are using an external user manager like LDAP to store user information, you must use WebCenter Sites ACLs.

A user must always have at least the Browser ACL in order to view Sites pages. However, additional ACL restrictions are enforced only when the `cc.security` property in the `futuretense.ini` file is set to `true.` For information about the `cc.security` property, see `futuretense.ini` in the *Property Files Reference*. For information about configuring security on your Sites system, see Chapter 6, "Setting Up External Security."

ACLs are assigned to three things: user accounts, database tables, and page entries in the `SiteCatalog` table (that is, WebCenter Sites pages).

> **Note**
>
> **User Management on the Delivery System.** User management on your delivery system is also based on ACLs. If your online site is designed to require visitors to register or log in before they can access areas of the site, you create the ACLs that are needed on the delivery system and then assign them to the appropriate database tables.
>
> Typically your site designers take care of assigning ACLs to WebCenter Sites pages. The *Oracle WebCenter Sites Developer's Guide* discusses how to design a user management process on the delivery system and provides code samples for pages that log visitors in to the site and verify their identities.

User Accounts

Every user must be assigned at least one ACL and the ACLs assigned to a user define that user's access to the Sites system.

While users have one user account and one set of ACLs, no matter how many sites they have access to, they can have one set of roles for one site and a different set of roles for another site. Therefore, users must be assigned all the ACLs necessary to provide them the permissions that they need to fulfill all of their site-specific roles.

For example, if you create a role that allows a user with the role the ability to create template assets, the user assigned that role must also be assigned the ElementEditor ACL, because creating templates writes data to the `ElementCatalog` table.

Database Tables

To restrict access to the data in a table, assign an ACL to it through the **WebCenter Sites Database** forms available through the **Admin** tab. Then, only those users with the same ACL have access to the data in that table.

If you assign more than one ACL to a table, a user needs only one of those ACLs to access the table. The user's access rights to that table (read, write, create, and so on) are the ones defined by the ACL.

All of the WebCenter Sites system tables (and several of the Sites content applications tables) have ACL restrictions. The `SystemInfo` table lists all the tables in the WebCenter Sites database and the ACLs assigned to them.

> **Note**
>
> Do not add ACLs to database tables through the WebCenter Sites Explorer application. Instead, use the **WebCenter Sites Database** form in the **Admin** tab.

With one exception—to register a foreign table in the WebCenter Sites database—never attempt to change the information in the `SystemInfo` table even if your user account has the ACL that allows you to do so.

For information about:

- Assigning ACLs to database tables, see "Assigning ACLs to Custom Tables," on page 76
- Registering foreign tables, see the "Database" chapter in the *Oracle WebCenter Sites Developer's Guide*

Page Entries in the SiteCatalog Table

The `SiteCatalog` table holds page entries for all the pages displayed for the Sites content applications as well as the pages displayed for your online site (that is, the site that you are delivering from the delivery system.) If you want to restrict access to a page, assign an ACL to it.

Typically, site developers determine how page restrictions should be configured on the delivery system. If you are customizing the user interface on the management system, however, you might need to use ACLs to restrict access to your custom pages.

System ACLs

WebCenter Sites and the Sites content applications use a number of system ACLs to control user access to their features and functions. Different combinations of these ACLs must be assigned to users. Information about system ACLs and their permissions are given in Appendix A, "System Defaults."

Sample ACLs

A newly installed WebCenter Sites system contains only system ACLs. No additional ACLs have been created for any of the sample sites that are packaged with WebCenter Sites.

Custom ACLs

Because WebCenter Sites provides a comprehensive set of ACLs, it is unlikely that you will need to create your own. However, situations can arise where user management needs on the management or delivery systems require you to create ACLs. For example,

- If your online site requires user registration, you may need to create a set of ACLs for site visitors.
- If your developers create new functions and put them on new tabs to customize the management system, you might need to create additional ACLs (or roles) to support the new functions and tabs.

Although creating and applying ACLs is an administrative task, you must first work with your site designers and developers to determine which ACLs you need and how to apply them. Once you have determined the ACLs, create them by following procedures in the rest of this chapter.

Working with ACLs

This section shows you how to create ACLs, edit, and delete custom ACLs; apply ACLs to the database tables and WebCenter Sites pages; and customize "access restricted" messages.

Note

When using an LDAP integration option, be aware of system response to user and site management operations. For information about system response, see Appendix C, "Managing Users, Sites, and Roles in LDAP-Integrated Sites Systems."

Creating a New ACL

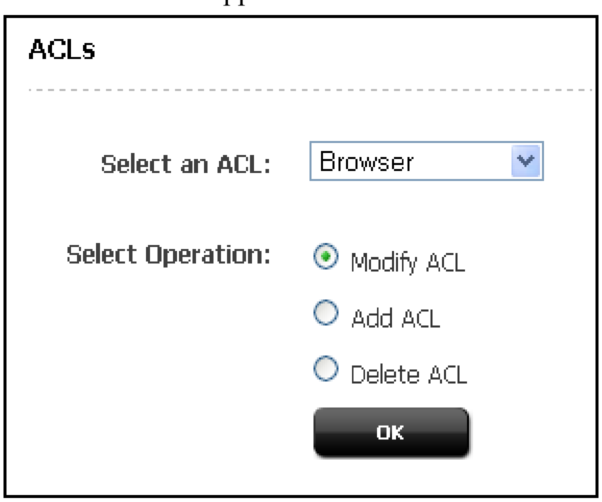
Note

When creating ACLs, consider the roles you will be using in order to ensure that the ACLs are commensurate with the roles. For example, if you will be creating a role that allows a user to create template assets, the user who is assigned that role must also be assigned the ElementEditor ACL, because creating templates writes data to the `ElementCatalog` table.

To create a new ACL

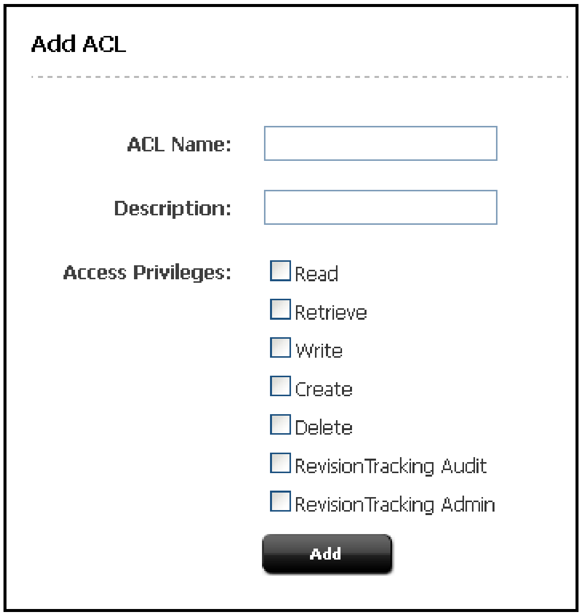
1. In the **Admin** tab, expand **Management Tools**, then double-click **ACLs**.

   The ACLs form appears.



2. Select **Add ACL** and click **OK**. The value in **Select an ACL** does not matter.

The Add ACL form appears.



3. In the **ACL Name** field, enter a unique name.
4. Select the access privileges you want to assign to this ACL. For information on each privilege, see "Permissions," on page 522).
5. Click **Add**.

   WebCenter Sites creates the ACL and writes it to the `SystemACL` table. The new ACL appears in the drop-down list in the form described in step 2 of this procedure.
6. If you are using LDAP, create a group (on your LDAP server) that exactly matches the ACL you just created. After you create the group, assign it to the appropriate users.

Editing a Custom ACL

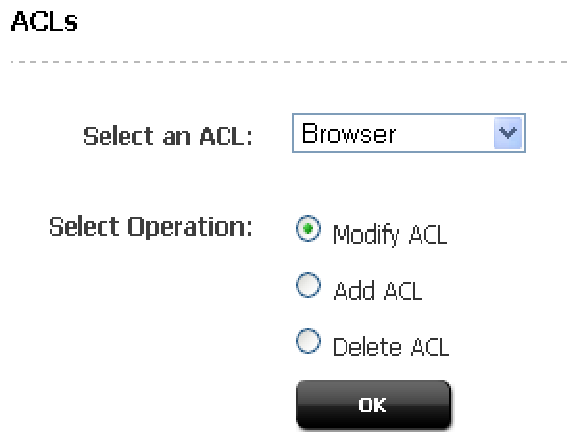
Caution

Never modify any of the system ACLs. For a list of these ACLs, see "System ACLs," on page 524.

To edit a custom ACL

1. In the **Admin** tab, expand **Management Tools**, then double-click **ACLs**.

The ACLs form appears.



2. In the **Select an ACL** field, select the ACL to edit.
3. Select **Modify ACL** and click **OK**.

   If the ACL you selected is a system ACL, an alert will display. Do not modify system ACLs.
4. In the Modify ACL form, make the desired changes to the Description and Access Privileges. For information on the displayed options, see “Permissions,” on page 522.
5. Click **Modify**.

   WebCenter Sites writes your changes to the `SystemACL` table.

Deleting a Custom ACL

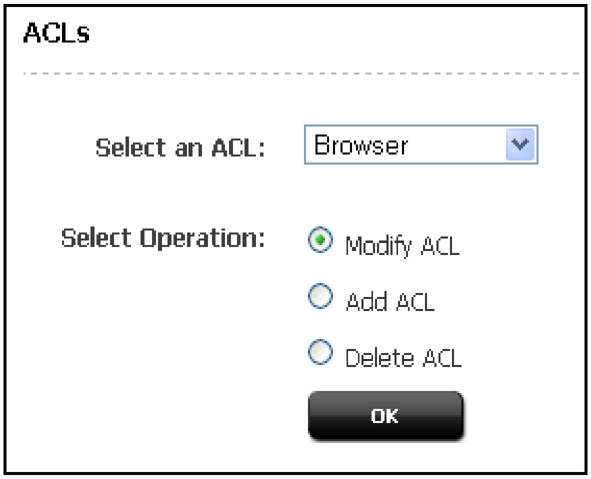
Caution

Never delete a system ACL. For a list of these ACLs, see “System ACLs,” on page 524.

To delete a custom ACL

1. If you are using LDAP, delete (from your LDAP server) the group corresponding to the ACL you will be deleting.
1. In the **Admin** tab, expand **Management Tools**, then double-click **ACLs**.

The ACLs form appears.



2. In the **Select an ACL** field, select the ACL to delete.
3. In the form that appears, select the desired ACL from the drop-down list and click **OK**.

   WebCenter Sites displays a warning message.
4. Click **OK**.

   The ACL has been deleted.

Assigning ACLs to Custom Tables

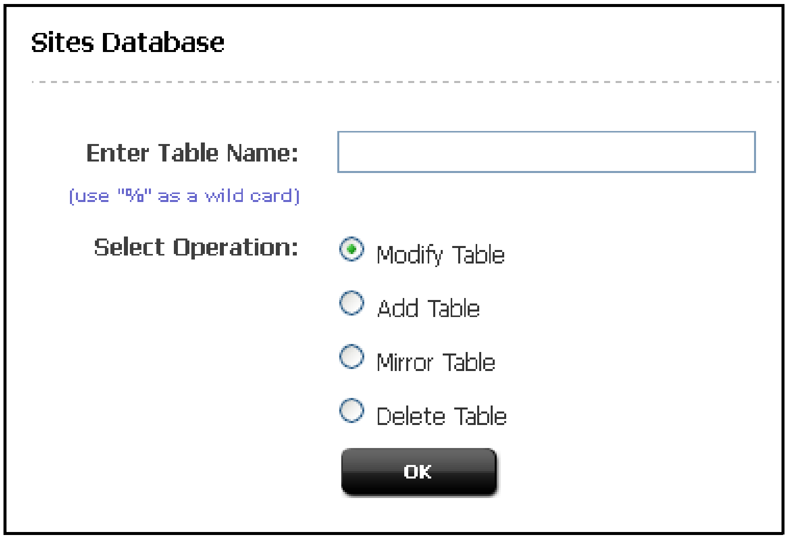
If you or the site designers create new tables, you might need to restrict access to those tables by assigning ACLs to them. Typically, you assign ACLs to new tables when you create those tables. (For more information, see the *Oracle WebCenter Sites Developer's Guide*.)

> **Note**
>
> Do not assign additional ACLs (beyond the ones assigned by default) to system or core product tables.

To assign ACLs to an existing table

1. In the **Admin** tab, expand **Management Tools**, then double-click **Sites Database**.

WebCenter Sites displays the following form.



2. Enter the name of the table to which you want to assign ACLs. If you do not know the name of the table you want to work with, do one of the following:
   - Leave the field blank. WebCenter Sites will return a list of all tables in the database.
   - Enter a partial name, ending with the wildcard character (**%**). WebCenter Sites will return a list of tables named similarly to your criteria.
3. Select **Modify Table** and click **OK**.
4. In the list of tables, select the desired table.

   WebCenter Sites displays the "Modify Catalog" form.
5. In the **ACL** field, select the ACL(s) you want to assign to the selected table. To select multiple ACLs, **Ctrl-click** each desired ACL. You can also select a range of ACLs by **Shift-clicking** the first and last ACL in the range.

   Note

   Do **not** change the value of the **File Storage Directory** field. For information about this field, look up the `defdir` property in the *Oracle WebCenter Sites Developer's Guide*.

6. Click **Modify**.

Assigning ACLs to WebCenter Sites Pages (SiteCatalog Page Entries)

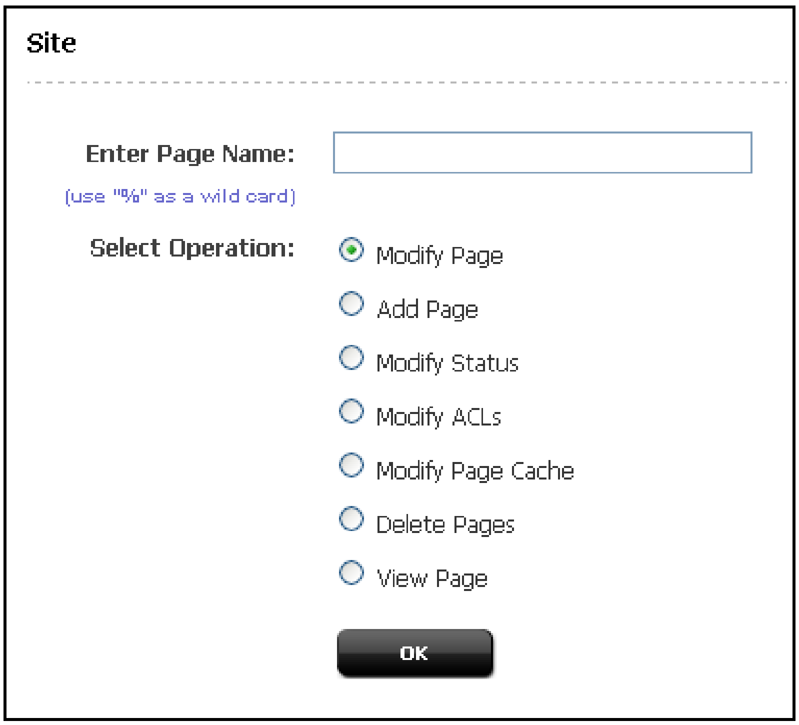
There are at least two ways to assign ACLs to `SiteCatalog` page entries.

- When developers create SiteEntry or template assets, they can assign ACLs to the page entry created for that asset through a field in the "Create" or "Edit" form.
- For page entries that are not associated with a SiteEntry or template asset, developers can use the Management Tools.

To assign ACLs to a page entry that is not associated with a SiteEntry or template asset

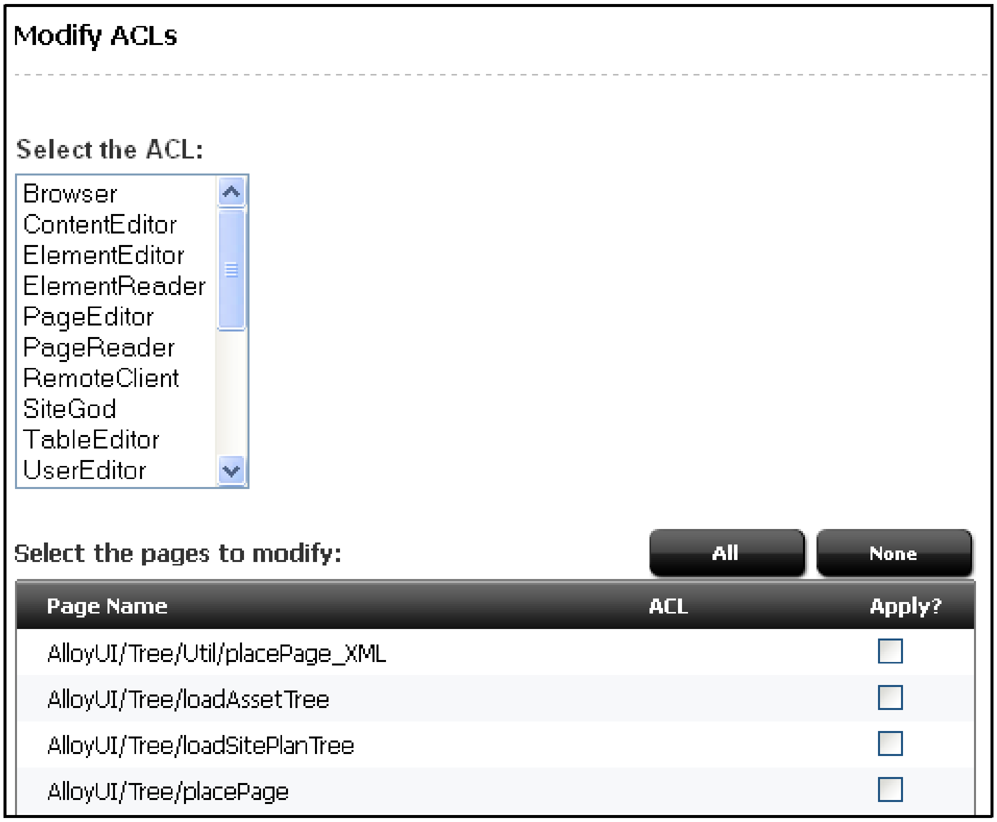
1. In the **Admin** tab, expand **Management Tools**, then double-click **Site**.

   WebCenter Sites displays the following form:



2. Enter the full path and name of the page to which you want assign ACLs. If you do not know the name of the page you want to work with, do one of the following:
   - Leave the field blank. WebCenter Sites will return a list of all page entries in the `SiteCatalog` table.
   - Enter a partial name, ending with the wildcard character (**%**). WebCenter Sites will return a list of pages named similarly to your criteria.
3. Select **Modify ACLs** and click **OK**.

The Modify ACLs screen appears.



4. In the scrolling list at the top of the form, select the ACLs you want to assign to one or more pages. To select multiple ACLs, **Ctrl-click** each desired ACL. You can also select a range of ACLs by **Shift-clicking** the first and last ACLs in the range.
5. In the list of pages, select the **Apply?** check box next to each page to which you want to assign the ACLs you selected in step 4. ACLs currently assigned to each page are shown in the **ACL** column. (To select all of the pages in the list, click **All**. To deselect all of the pages in the list, click **None**.)
6. At the bottom of the screen, click **Apply**.

Setting the ACL Restriction Error Message

When users attempt to access a page for which they do not have the appropriate permissions, WebCenter Sites displays an error message. This message is stored in the following file:

```
<cs_install_dir>/futuretense_cs/formpriv.html
```

You can customize this and other error message pages in the `futuretense_cs` directory, with the following restrictions:

- Do not rename any of the files.
- Do not alter any occurrences of the `{0}` string in any of the files. WebCenter Sites uses the `{0}`string to automatically generate these error messages.

Roles

User management includes the concept of roles. Roles complement users in the following ways:

- Whereas the user definition (account) describes an individual's access to the underlying Sites functionality (database tables) through ACLs, roles are used to manage site-specific access to WebCenter Sites' interface functions.
- A role represents a job description or the title of individuals with similar functions: for example, content provider, editor, site designer, and administrator.
- Each Sites user has one user definition (account) no matter how many sites have been created. However, the user's roles can vary by site.
- When you enable a user for a site, you enable that user within the context of the roles that user is to fulfill for that site.

The roles assigned to a user for a site determine the following:

- Which assets the user can create on that site.
- Which assets the user can search for on that site.
- Which tabs are displayed in the tree when the user logs in to the site.
- Whether the user is eligible for participation in any workflow processes, and, if so, for which steps in those workflow processes.
- Which functions a user can or cannot perform on an asset while it is moving through a workflow process.
- Whether the user can administer a workflow process or create or modify a workflow group on that site.

System Roles

Several system roles are installed by WebCenter Sites. One role is required for the Sites content applications to function, and three are required for the WebCenter Sites administrators to function. For more information, see "System Roles," on page 529.

Sample Roles

If you installed one or more sample sites, you will have access to a number of sample roles that are included with the sites. The roles permit the sample site users to access different tree tabs. You can use the sample roles as examples of how you can configure access control on your sites.

Custom Roles

Unlike ACLs, roles are objects that you will most likely need to create in order to account for the full range of users' responsibilities on your sites. To create roles, follow instructions in the next section, "Working with Roles."

Working with Roles

This section shows you how to create, edit, and delete roles.

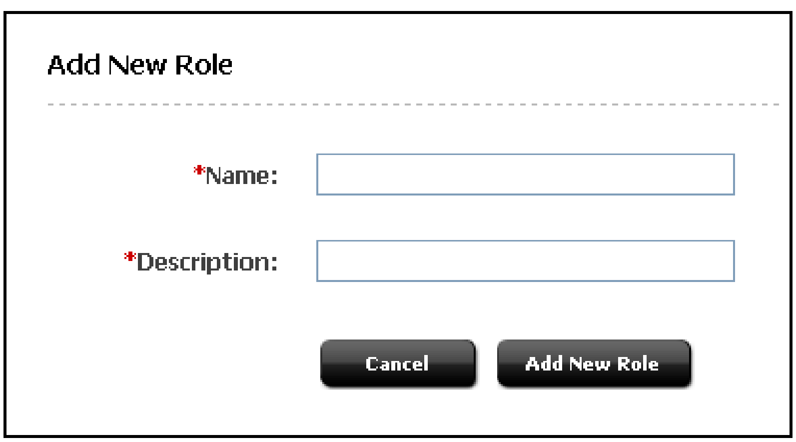
> **Note**
>
> If you are using LDAP, be aware of system responses to user and site management operations. For information about system responses, see Appendix C, "Managing Users, Sites, and Roles in LDAP-Integrated Sites Systems."

Creating a Role

To create a new role

1. In the **Admin** tab, expand **Roles**, then double-click **Add New**.

   WebCenter Sites displays the "Add New Role" form.



2. In the **Name** field, enter a unique name of up to 32 characters.
3. In the **Description** field, enter a short and informative description of no more than 255 characters.
4. Click **Add New Role**.

5. Add the role to the default tree tabs (**Workflow**, **Bookmarks**, **Site Admin**, **Admin**, **Site Plan**, and **Active List**), as appropriate. For instructions, see "Editing Tree Tabs," on page 147.

Editing a Role

Although you cannot change the name of a role after you have created it, you can edit the description of the role.

To edit the description of a role

1. In the **Admin** tab, double-click **Roles**. A list of all roles appears in the main window.
2. In the list of roles, navigate to the role you want to edit and click its **Edit** (pencil) icon.
3. In the "Edit Role" form, make your changes, then click **Save**.

Deleting a Role

Caution

Do not delete any of the system default roles: GeneralAdmin, SiteAdmin, WorkflowAdmin, AdvancedUser, SitesUser.

To delete a role

1. In the **Admin** tab, double-click **Roles**. A list of all roles appears in the main window.
2. In the list of roles, navigate to the role you want to delete and click its **Delete** (trash can) icon.

   WebCenter Sites displays a confirmation message.
3. Click **Delete Role**.

   The role has been deleted.

Chapter 5

Configuring Users, Profiles, and Attributes

Users can be created through one of the following options: WebCenter Sites' native user manager (in the administrator's interface), or via LDAP. However, to configure user profiles and attributes, you must use the administrator's interface.

This chapter outlines your options for creating users. This chapter also shows you how to create users in WebCenter Sites, and configure their profiles and user attributes. For information about creating users in external sources, refer to the product documentation.

This chapter contains the following sections:

Overview

Sites users can be created through WebCenter Sites' native user manager, through WEM Admin, or through external user managers such as LDAP.

Every Sites user is defined by the following set of data:

- User account, which gives the user access to the WebCenter Sites system and its database tables
- User profile, which is required for users who will be working with:
  - Sites applications
  - Language packs and setting a default language
  - Workflow processes, in which e-mail messages will be sent to notify workflow participants of their assignments. The user profile supports workflow actions by mapping a user name to an e-mail address.
- User attributes (in addition to the e-mail and locale attributes in the user profile), if actions and events in addition to workflow must be supported.

Once users are created and configured, they must be associated, by means of roles, with the sites they are to work in. This chapter shows you how to create and configure users. Procedures for associating users with sites are given in Chapter 7, "Assembling CM Sites."

User Management Options

WebCenter Sites' Directory Services API enables your Sites system to connect to external directory servers or user managers that contain authentication information, user information, and so on. The following connection options are available:

- Native system—The WebCenter Sites native user manager, which uses the native WebCenter Sites user management tables `SystemUsers` and `SystemUserAttrs`.
- The LDAP plug-in—With this option, user names and attributes are stored in the directory server rather than in the WebCenter Sites database.

Because WebCenter Sites security is based on ACLs, any external user management system (such as LDAP) must be configured to match the WebCenter Sites ACLs.

Information about configuring the plug-ins is given in our guide, *Configuring Third-Party Software*. Properties that configure the plug-ins are located in the files `futuretense.ini` (the **Authentication** tab), `ldap.ini`, and `dir.ini` files. The files are described in the *Property Files Reference*.

Note

This guide uses the native WebCenter Sites user manager throughout.

Native WebCenter Sites User Manager

If you are using the native WebCenter Sites user manager, follow the guidelines in Chapter 3, "Site Configuration Guidelines" to create and configure users, and then grant them access to the management system.

LDAP Plug-In

If you are using LDAP to manage your users on either the management or the delivery system, you create user accounts with LDAP rather than with the WebCenter Sites administrator's interface. However, you must still use the administrator's interface to create ACLs and roles in the WebCenter Sites database (It is also possible to use WEM Admin to create roles). Instructions for granting users access to a WebCenter Sites management system integrated with LDAP are given in our guide, *Oracle WebCenter Sites Configuration Guide for Supporting Software*.

Configuring Users in WebCenter Sites

Each WebCenter Sites user is completely defined by a user account, user profile, and, if necessary, user attributes.

- A user account is required for anyone who is to work with WebCenter Sites.
- A user profile is required for users who will be working with WebCenter Sites modules and products, setting a default language, and participating in workflow processes in which e-mail messages will be sent.
- User attributes, in addition to the locale and e-mail attributes in the user profile, may also be required for your operation. If so, the additional attributes can be created.

Once you have created the user, you must enable that user for the appropriate sites by assigning roles to the user name for each site the user will work in. For information about enabling users after you have created them, see "Granting Users Access to a Site (Assigning Roles to Users)," on page 112.

After you have created and enabled a new user, be sure to give that user the following information:

- The user name/password combination of the user account.
- The URL to the WebCenter Sites:

  `http://<server>:<port>/<context>/login`

  where

  `<server>` is the host name or IP address of the machine running WebCenter Sites. Depending on how the system was set up, you might also need to include the port number— `server:8080` for example; and

  `<context>` is the name of the web application on the same server.

The rest of this chapter shows you how to create user accounts, profiles, and attributes, as well as modify and delete them.

Working with User Accounts

This section shows you how to create, edit, and delete user accounts in WebCenter Sites.

> **Note**
>
> This section provides procedures for creating, editing, and deleting users in the WebCenter Sites administrator's interface. If you are using LDAP, refer to the LDAP product documentation. If you are using WEM Admin, refer to the *Oracle WebCenter Sites Administrator's Guide for the Web Experience Management Framework*.
>
> Also, be sure to substitute the word "group" for the word "ACL" when reading that guide, and create users who belong to the groups with these names.

Creating a New User

> **Note**
>
> This section shows you how to create users in the WebCenter Sites administrator's interface. If you are using LDAP, refer to the LDAP product documentation, if you are using WEM Admin, refer to the *Oracle WebCenter Sites Administrator's Guide for the Web Experience Management Framework*.

Before creating a user

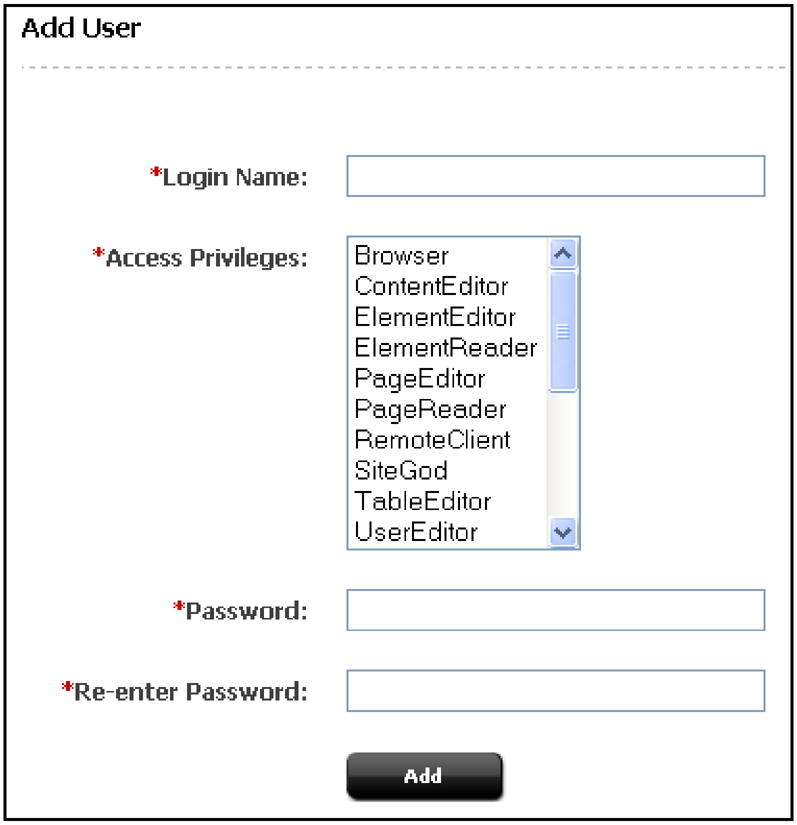
1. Before creating a user, determine the user's:
   - Log in name
   - Password
   - ACLs, which regulate the user's access to WebCenter Sites' database tables.
     - To determine the user's required ACLs, see "Required ACLs for Custom Users," on page 528.
     - To determine the user's additional system ACLs, see "System ACLs," on page 524.
     - To determine which ACLs are assigned to the database tables the user must access, follow the steps in "Assigning ACLs to Custom Tables," on page 76.

To create a user

2. In the **Admin** tab, expand **Management Tools**, then double-click **User**.
3. In the form that appears, select **Add User** and click **OK**.

   You can add a user name before clicking **OK**; it will appear auto-filled in the **Login Name** field on the next screen.

4. Fill in the “Add User” form as follows:



a. In the “Login Name” field, enter a unique name. Do not include spaces or special characters, such as punctuation. The underscore character (_) is allowed.

b. In the “Access Privileges” list, select ACLs for the user. To select multiple ACLs, **Ctrl+Click** each desired ACL; you can also select a range of ACLs by **Shift+Click**ing the first and last ACL in the range.

c. Enter the same password into the **Password** and **Re-Enter Password** fields.

d. Click **Add**.

   The user has been created.

e. You can create a general administrator, For more information about making the user an administrator or a content contributor, see “Creating Common User Types,” on page 88.

To follow up with post-creation procedures

5. If the user will be implementing any of the following options:

   - Oracle WebCenter Sites products such as Oracle WebCenter Sites: Engage
   - Language packs and different languages
   - Workflow processes that send e-mail messages

   create a profile for the user. For instructions, see “Creating and Editing a User Profile,” on page 93.

6. If the user requires attributes in addition to or in place of locale and e-mail (specified in the user profile), create the attributes. For instructions, see "Modifying, Adding, and Deleting User Attributes," on page 94.
7. Once the user has been completely defined, you must associate the user with a site by means of roles.
   a. If you have not already done so, create roles for the user, following instructions in "Creating a Role," on page 81.
   b. To associate the user to the site, create the site and add the user to the site. For instructions, see "Creating a Site," on page 106, and "Granting Users Access to a Site (Assigning Roles to Users)," on page 112.

Creating Common User Types

When creating a user, it is important to consider the roles that the user will have on separate web sites. In many cases, a user may be an administrator on one site and a content contributor on another. This section details how to make each user a specific type of user.

This section describes the following topics:

Making the User a General Administrator

The general administrator for a site controls all aspects of the site. You should be aware that when a user is a general administrator, their administrative actions can only be controlled when they are logged in to the one site, however, they can administrate all sites when logged in.

1. For the newly created user, double-check that it has the correct ACLs assigned to it.

   To determine the ACLs for a user, see "Required ACLs for Custom Users," on page 528. Looking at the table there, you can see that the ACLs for a general administrator are Browser, Element Reader, PageReader, UserReader, xceleditor, xceladmin, TableEditor, UserEditor, and VisitorAdmin.
2. On the **Admin** tab expand security, expand **Assign Users to Groups**, and double-click **Add New**. The "Assign Groups to User" screen appears.
3. Select the user to make a general administrator, and then select **RestAdmin** in the Groups field.

   You can use **Ctrl+Click** to select more than one user or group, but as you will likely make few general administrators, it is suggested you add them one at a time to ensure you do not accidentally add a user to the RestAdmin group.
4. Open the **Sites** tree, and select the site to add the user to.
5. Click **Users** to open the "User Role Management" screen.

   This screen is also available under **Site Admin** tab by double clicking **Users** if you are logged in to the specific site.

6. Enter the log in information for the user, and click the **Edit** icon to open the "Edit Roles for User" screen. Select **GeneralAdmin** as well as any other roles you wish the user to have on this web site.

> **Note**
>
> The names of roles used in this documentation are the names of the roles used in the FirstSiteII example web site that comes installed. You might choose to use different roles or role names with your published web site.

When adding a general administrator to a site, you may want to consider if you will want the same person to have different roles on a different site. You will need to assign these roles to the user when logged in to the site or sites you want to assign these different roles for the user on.

When a user has general administrator rights, the Admin tab will only appear when they are logged into the web site (or web sites) where they have the general administrator role.

Making the User a Site Administrator

1. For the newly created user, double-check that it has the correct ACLs assigned to it.

   To determine the ACLs for a user, see "Required ACLs for Custom Users," on page 528. Looking at the table there, you can see that the ACLs for a site administrator are Browser, Element Reader, PageReader, UserReader, xceleditor, and xceladmin.
2. Open the **Sites** tree, and select the site to add the user to.
3. Click **Users** to open the "User Role Management" screen.

   This screen is also available under the **Site Admin** tab by double-clicking **Users** if you are logged in to the specific site.
4. Enter the log in information for the user, and click the **Edit** icon to open the "Edit Roles for User" screen.
5. Select the user to make a site administrator, and then select **SiteAdmin_AdminSite** in the Groups field.

   You can use **Ctrl**+**Click** to select more than one user or group, but as you will likely make few site administrators, it is suggested you add them one at a time to ensure you do not accidentally add a user to the SiteAdmin_AdminSite group.
6. Select **SiteAdmin** and **Advanced User** as well as any other roles you wish the user to have on this web site. The SiteAdmin role grants the user access to the **Site Admin** tab, the AdvancedUser role grants access to the Admin interface.

> **Note**
>
> The names of roles used in this documentation are the names of the roles used in the FirstSiteII example web site that comes installed. You might choose to use different roles or role names with your published web site.

When adding a site administrator to a site, you may want to consider if you will want the same person to have different roles on a different site. You will need to assign these roles to the user when logged in to the site or sites you want to assign these different roles for the user on.

When a user has site administrator rights, the **Site Admin** tab will only appear when they are logged into the web site (or web sites) where they have the site administrator role.

Making the User a Workflow Administrator

1. For the newly created user, double-check that it has the correct ACLs assigned to it.

   To determine the ACLs for a user, see "Required ACLs for Custom Users," on page 528. Looking at the table there, you can see that the ACLs for a workflow administrator are Browser, Element Reader, PageReader, UserReader, xceleditor, and xceladmin.
2. Open the **Sites** tree, and select the site to add the user to.
3. Click **Users** to open the "User Role Management" screen.

   This screen is also available under **Site Admin** tab by double-clicking **Users** if you are logged in to the specific site.
4. Enter the log in information for the user, and click the **Edit** icon to open the "Edit Roles for User" screen.
5. Select **WorkflowAdmin** and **AdvancedUser**, as well as any other roles you wish the user to have on this web site. The WorkflowAdmin role grants the user access to the **Workflow** tab, the AdvancedUser role grants access to the Admin interface.

Note

The names of roles used in this documentation are the names of the roles used in the FirstSiteII example web site that comes installed. You might choose to use different roles or role names with your published web site.

When adding a workflow administrator to a site, you may want to consider if you will want the same person to have different roles on a different site. You will need to assign these roles to the user when logged in to the site or sites you want to assign these different roles for the user on.

When a user has workflow administrator rights, the **Workflow** tab will only appear when they are logged into the web site (or web sites) where they have the workflow role.

Making the User a Content Contributor

1. For the newly created user, double-check that it has the correct ACLs assigned to it.

   To determine the ACLs for a user, see "Required ACLs for Custom Users," on page 528. Looking at the table there, you can see that the ACLs for all users are Browser, ElementReader, PageReader, UserReader, and xceleditor. You may want to add additional ACLs needed for a content contributor, depending on the type of content you want the user to contribute.

For example, the PageEditor ACL would allow the user to add pages to the site, and modify the content based on the defined elements. The ElementEditor ACL would allow the contributor to create templates. For more information on the specific ACLs and how they impact the contributor's editing access, see "System ACLs and Their Descriptions," on page 525.

The user does not need additional ACLs if you only want them to modify existing content on a page. Simple editing is contained in the assigned role.

2. Open the **Sites** tree, and select the site to add the user to.
3. Click **Users** to open the "User Role Management" screen.

   This screen is also available under **Site Admin** tab by double clicking **Users** if you are logged in to the specific site.
4. Enter the log in information for the user, and click the **Edit** icon to open the "Edit Roles for User" screen. Select **SitesUser** for the role.

   It may be necessary to add different roles depending on how the roles are connected to the assets. When assets are created, they are connected to roles. These roles are the only roles that have access to the asset.

Note

Typically you would not assign a user the SitesUser role as well as the AdvancedUser role for the same web site. The SitesUser role grants access to the Contributor interface; the AdvancedUser role grants access to the Admin interface.

The names of roles used in this documentation are the names of the roles used in the FirstSiteII example web site that comes installed. You might choose to use different roles or role names with your published web site.

For most content contributors, you will not want to assign them roles on the web site beyond SitesUser, depending on how you have set up the different assets. Content contributors will make changes to the content only through the Contributor interface, and not through the Admin interface that administrators use.

Editing a User

Note

This section shows you how to edit users in the WebCenter Sites administrator's interface. If you are using LDAP, refer to the LDAP product documentation.

To edit a user

Caution

Do **not** change the names or ACLs of WebCenter Sites system users (`DefaultReader`, `ContentServer`, `xceladmin`).

1. In the **Admin** tab, expand **Management Tools**, then double-click **User**.
2. In the form that appears, enter the name of the user you want to work with. If you do not know the user name, leave the field blank; WebCenter Sites will return a list of all users in the system.
3. Select **Modify User** and click **OK**.
4. In the list of users, select the user you want to work with.
5. In the "Modify User" form, make the desired changes, then click **Modify**.

Deleting a User from the System

Note

This section shows you how to delete WebCenter Sites users using the WebCenter Sites administrator's interface.
If you are using LDAP, refer to the LDAP product documentation.

To delete a user from the system

Caution

Do **not** delete any of the WebCenter Sites system users (`fwadmin` - or the user used as admin, `ContentServer`, or `DefaultReader`).

1. Delete the user profile, as shown in "Deleting a User Profile," on page 94, then continue with the next step.
2. Delete the user:
   a. In the **Admin** tab, expand **Management Tools**, then double-click **User**.
   b. In the form that appears, enter the name of the user you want to delete. If you do not know the user name, leave the field blank; WebCenter Sites will return a list of all users in the system.

c. Select **Delete User** and click **OK**.
d. In the list of users, select the **Delete** radio button next to the user you want to delete.
e. Click **Delete**.

   WebCenter Sites displays a warning message.
f. Click **OK**.

   The user has been deleted.

Working with User Profiles and User Attributes

A user profile is required for any user who will be working with the following:

- Sites modules and products
- Language packs (and languages must be enabled)
- Workflow processes in which e-mail messages will be sent to notify workflow participants of their assignments. The user profile supports workflow actions by mapping a user name to an e-mail address.

A **user profile** holds a set of **user attributes**. By default, the only user attributes a user profile holds are:

- The **email attribute**, which is used to support workflow actions and takes the user's e-mail address as a value. (You can create workflow actions that send workflow participants e-mail about the assets that are assigned to them.)
- The **locale attribute**, which is used to determine which language to use when your WebCenter Sites system has a language pack installed. This attribute takes the user's preferred location as a value.

If the user was created in WEM, the avatar is also as attribute. You can add more user attributes and store values for them in the WebCenter Sites user management tables if you want to. However, to use these values in the WebCenter Sites interfaces requires you to customize the elements that display the user profile forms. For information about customizing elements for the WebCenter Sites user interface, see the *Oracle WebCenter Sites' Developer's Guide*.

Creating and Editing a User Profile

> **Note**
>
> If you are using LDAP, be aware of system responses to user and site management operations. For information about system responses, see Appendix C, "Managing Users, Sites, and Roles in LDAP-Integrated Sites Systems."

To create or edit a user profile

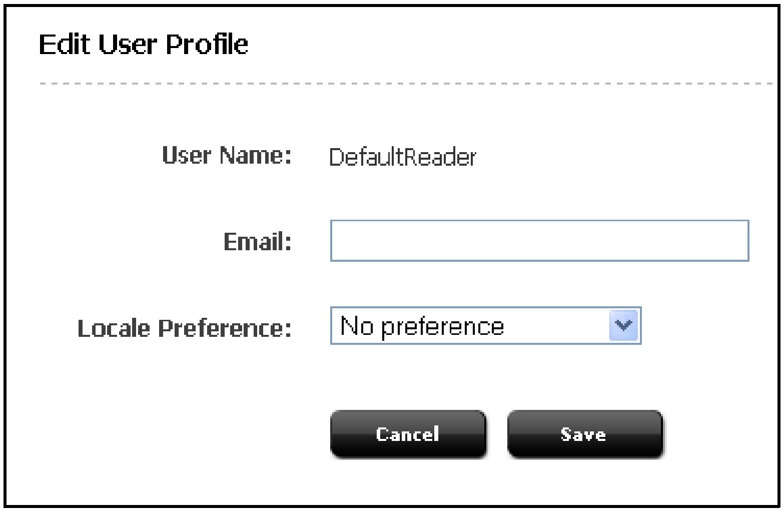
1. In the **Admin** tab, double-click **User Profiles**.

   WebCenter Sites displays the "User Profile Management" form.
2. In the form, enter the desired user name and click **Select**.

WebCenter Sites displays the profile of the selected user.

3. Click the **Edit** icon.

   WebCenter Sites displays the "Edit User Profile" form.



4. In the **Email** field, enter the user's e-mail address.
5. (Optional) If one or more language packs are installed on your Sites system, select a locale preference for this user from the "Locale Preference" drop-down list.
6. Click **Save**.
7. Enable this user for the sites the user needs to work with. For instructions, see "Granting Users Access to a Site (Assigning Roles to Users)," on page 112.

Deleting a User Profile

To delete a user profile

1. In the **Admin** tab, double-click **User Profiles**.
2. In the "User Profile Management" form, click **Delete**.

   WebCenter Sites displays a warning message.
3. Click **Delete User Profile**.

   WebCenter Sites displays a message confirming the deletion.

Modifying, Adding, and Deleting User Attributes

By default, the only user attributes that the Sites content applications need are an e-mail address and locale preference. Users created in WEM will additionally have an avatar attribute. You use the user profile feature to assign these attributes to a user, as shown in "Creating and Editing a User Profile," on page 93. If you need to, you can store and use additional user attributes for your users in this table, even if you are using LDAP.

The **Modify User Attributes** option allows you to modify the attributes that are used in the user profile. It also allows you to add and delete attributes.

To modify a user's attributes

1. In the **Admin** tab, expand **Management Tools**, then double-click **User**.

   In the form that appears, enter the name of the user you want to work with. If you do not know the user name, leave the field blank; WebCenter Sites will return a list of all users in the system.

2. Click the user name in the list that whose attributes you want to modify. The User Attributes form will display.

3. Fill in the User Attributes form, as explained below:

   Do one of the following, as required:

   - Change the current value (or values) assigned to an attribute by editing the contents of the **Attribute Values** field.
   - Add a new attribute by entering its name and at least one value in the fields at the bottom of the form.
   - Delete any unwanted attributes by deleting the associated value (in the **Attribute Values** field).

4. Click **Modify**.

   Your changes are committed to the database.

Chapter 6

Setting Up External Security

Determining what your security protocols should be and then implementing them is an important part of a Sites administrator's duties. You and the site developers must discuss and determine how security will be configured on both the management and the delivery systems before they start coding templates and designing asset types.

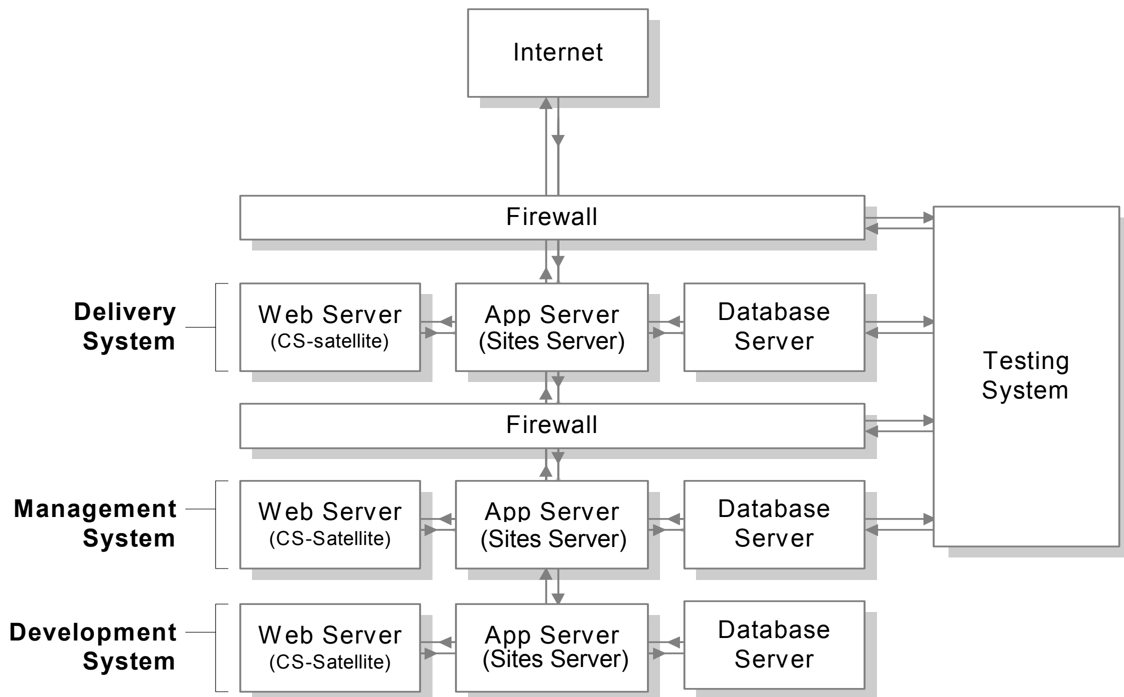
This chapter contains the following sections:

- Overview
- Implementing Security
- Testing Security

Overview

Before developers begin designing the online site or contemplating changes to the user interface on the management system, you must determine and implement your security protocols. The decisions you make about security configuration affects the way that you code and implement your online site.

At regular intervals, you should review your systems to determine whether they are working as they should. The following figure shows an example of a secure Sites system:



ACLs and Security

As mentioned in Chapter 4, “Working with ACLs and Roles,” ACLs (access control lists) serve as the foundation of the security and user management model in your Sites system. ACLs are sets of permissions that restrict access to both database tables and WebCenter Sites pages.

ACL restrictions are enforced only when the `cc.security` and the `security.checkpagelets` properties in the `futuretense.ini` file are set to `true`. These properties are set to `true` by default. If you need to disable security for any reason, open the Property Editor and set the `cc.security` property to `false`. However, Oracle recommends that you keep this property set to `true` on all systems to ensure that site development is designed to work with security enabled.

When planning security measures for your system, examine the list of default ACLs (they are described in Appendix A, “System Defaults”), and determine whether you need additional ACLs. You might need to create an ACL with a different combination of permissions than any of the system ACLs provide if your developers plan to create new tables or to make additions to the user interface.

Note

Under no circumstances should you modify a system default ACL or modify the ACLs assigned to a WebCenter Sites system table or a Sites content application table.

DefaultReader, secure.CatalogManager, and secure.TreeManager

WebCenter Sites and the Sites content applications are delivered with several default user accounts, one of which is named DefaultReader. This account is assigned to all unauthorized site visitors (because anyone who visits a site hosted by WebCenter Sites must have a WebCenter Sites user identity).

By default, the DefaultReader user account has the following ACLs: `Browser` and `Visitor`. `Visitor` enables the tracking of Engage asset visitors for demographic purposes. Security issues arise because many of the database tables also have the `Browser` ACL assigned to them, and various Engage tables are assigned the `Visitor` ACL. Therefore, anyone using the DefaultReader account can examine information in the tables (although they cannot write to the tables as this user). To prevent someone from using the DefaultReader user account to view tables in your WebCenter Sites database, set the following properties in the `futuretense.ini` file to `true`:

- `secure.CatalogManager`
- `secure.TreeManager`

When these properties are set to `true`, the DefaultReader user cannot access the CatalogManager and TreeManager servlets—not even for read-only data.

Note

More information about Oracle WebCenter Sites: Engage ACLs (`Visitor` and `VisitorAdmin`) is available in Appendix A, “System Defaults.” See Table A-2 for permissions that are associated with each ACL and Table A-3 for descriptions of the ACLs.

BlobServer and Security

When you want to implement security for your blobs you must enable the security feature for the BlobServer servlet. You do this by setting the `bs.security` parameter in the `futuretense.ini` file to `true`.

When `bs.security=true`, BlobServer refuses to serve a blob unless the URL for the blob contains evidence that the person requesting it has been authenticated as a valid user. What evidence? A value in the URL from the `csblobid` parameter whose value matches a session variable named `csblobid`. Therefore, when BlobServer security is enabled, your developers must code links to blobs differently than usual.

For information about how those links should be coded, see the *Oracle WebCenter Sites Developer's Guide*.

Security Goals

Typically, you have a different set of security goals for each of your Sites systems: the development, management, and delivery systems.

Security Goals for the Development System

Even though development systems are typically behind firewalls, you should implement the same security configuration on the development system that will be in place on the system the developers are designing for (management or delivery). Why? To be sure that the code will work properly on the target system, because the same conditions exist on both systems:

- When you plan to use BlobServer security on the delivery system, templates that create URLs for blobs must be coded differently. Therefore, you must enable BlobServer security on the development system as well.
- If your online site will require that visitors identify themselves, you must have the same security configuration in place on the development system that you will use on the delivery system.

Security Goals for the Management System

Security on the management system encompasses two main concepts:

- Ensuring that only valid Sites users can access the system (described in this chapter).
- Ensuring that those valid users can access only the functions that are appropriate for them. For information, see Chapter 4, "Working with ACLs and Roles."

Security Goals for the Delivery System

The precautions that you take on the delivery system are more stringent by nature because when you deliver a site to the public, you must ensure that while visitors can access the content on your site, hackers cannot access areas that you do not want made public.

When you configure your delivery system, you disable the WebCenter Sites user interfaces to prevent anyone from adding assets or code through the interfaces or with any of the developer tools. Additionally, you restrict access to some of the WebCenter Sites servlets.

Implementing Security

This section describes the steps you must take to configure the security measures you have decided to implement: setting properties, changing passwords for default user accounts, using SSL for systems accessible outside your company's network, mapping URLs for specific WebCenter Sites servlets, and disabling certain parts of the applications on the delivery system.

Each section states which steps should be performed on which system or systems.

Properties That Configure Security Settings

This section describes how to configure the properties in the `futuretense.ini` file that implement various kinds of security.

For All Systems

Use the Property Editor to open the `futuretense.ini` file and verify that the following properties are set to `true`:

- `cc.security`
- `security.checkpagelets`
- `secure.CatalogManager`
- `secure.TreeManager`

If you plan to use BlobServer security, set the following property to `true`:

`bs.security`

To ensure that the session timeout value is appropriate for each system, set the following property, as well:

`cs.timeout`

On the development and management systems, set the timeout value for as long as you think that you safely can. On the delivery system, set the timeout value for as short a time as you can without frustrating your visitors.

For the Delivery System

In addition to the properties described in the preceding section, there is one more property to specify for the delivery system: `cs.wrapper`.

"Wrappers" are the HTML files located in the `futuretense_cs` directory on the web server. Also included in that directory is a subdirectory named Dev, which you should remove from your delivery system.

However, if you decide to remove the entire `futuretense_cs` directory from the web server on the delivery system, you must set the `cs.wrapper` property to `false`.

For more information, see "Sites Forms and Pages (Delivery System Only)," on page 103.

Users and Passwords

Be sure that the default user accounts were made secure after Sites was installed on all systems.

For All Systems

Complete the following steps on all systems—development, management, and delivery:

- Change the default password for the `fwadmin` user account (**User** node under **Management Tools** in the **Admin** tab in the administrator's interface).
- If the WebCenter Sites user that was created during the installation is named `ContentServer`, verify that its password is not the default password (`password`). If the password used is the default password, change it (**User** node under **Management Tools** in the **Admin** tab in the administrator's interface).
- Change the default administrator username/password for your application server.

- Change the default administrator username/password for your web server.
- If you have the sample sites installed, a mirror user was created for the Mirror to Server publishing method (name: `mirroruser`; password: `mirroruser`). Change the password for this user.
- For any user accounts that have the SiteGod ACL, change the password frequently (**User** node under **Management Tools** in the **Admin** tab in the administrator's interface). Handle any user account that has the SiteGod ACL as you would the UNIX `root` user.

Note

Do **not** change the default password for the `DefaultReader` user.

For the Management and Delivery Systems

If any of the sample sites **except** AviSports or FirstSiteII were installed on either the management or delivery systems, **delete all the sample users including editor**, but do **not** delete the xceleditor ACL and do **not** delete the fwadmin user. Additionally, change the password for the mirroruser user.

For the Delivery System

Do **not** assign the xceleditor ACL to any user on the delivery system other than the system administrator of that system. This ACL allows access to the WebCenter Sites content applications if they are installed on that delivery system.

SSL and Digital Certification

For any system accessible outside your company's network or containing proprietary data, such as a management system open to remote employees, it is recommended to use SSL to establish a secure, encrypted connection.

You must **use a digital certificate approved by a trusted authority** rather than a self-signed certificate. Using a self-signed certificate can have the following consequences:

- Internal calls to the system made through Java may fail.
- Some features of the user interface may not function correctly, for example, the left-hand navigation tree or publishing.

URLs and the Web Server (Delivery System Only)

On the web server of the delivery system, be sure to give access to the following WebCenter Sites servlets only. How? By mapping only their URLs to the application server:

- ContentServer
- BlobServer
- Satellite
- CookieServer

Do **not** map URLs to the application server for any of the other WebCenter Sites servlets. Instead, map the URLs for the following servlets to an error page such as the "404 Not Found" page:

- HelloCS
- CatalogManager
- TreeManager
- DebugServer
- CacheServer
- Inventory

Sites Forms and Pages (Delivery System Only)

On the delivery system, be sure to disable or completely remove the following pieces of the Sites applications:

- Remove the `futuretense_cs/Dev` directory from the **web server**. This directory holds forms that are useful for developers, which means that you do not want them on your delivery system.

  Note

  Do not remove this directory from the application server. Remove it from the **web server only**.

  You can optionally remove the entire `futuretense_cs` directory from the web server. (Do **not** remove it from the application server.) If you do, be sure to set the `cs.wrapper` property in the `futuretense.ini` file to `false` (the wrappers are the HTML files from that folder). If you forget, visitors on the site will see "404 Page Not Found" rather than some of the system error messages.
- Go to `SiteCatalog/OpenMarket/Xcelerate/UIFramework` and rename all of the pages. At the very least, rename `LoginPage` and `LoginPost`.

Testing Security

After you have implemented your security measures, test your systems.

Security Tests for All Systems

Complete the following steps on your development, management, and delivery systems:

1. Try to log in to the database with Sites Explorer using the default user accounts:
   - `DefaultReader`

     If you can log in using `SomeReader` as the password, the `secure.CatalogManager` and `secure.TreeManager` properties are set to `false`. Change them to `true`.
   - `ContentServer`

     If you can log in using `password` as the password, change the password immediately.
   - `editor`

     If you can log in using `xceleditor` as the password, change the password immediately.
   - `fwadmin`

     If you can log in using `xceladmin` as the password, change the password immediately.
2. Verify that the sample site users do not exist on the management or delivery systems.
3. Verify that you cannot log in with `ContentServer/password` using a CatalogManager URL:

```
http://<server>:<port>/<context>/
   CatalogManager?ftcmd=login&username=ContentServer&password=
   password
```

4. Verify that you cannot flush the entire cache as `ContentServer/password` using a CacheServer URL:

```
http://<server>:<port>/<context>/
   CacheServer?all=true&authusername=ContentServer&authpassword
   =password
```

5. Verify that you cannot log in to the application server as the default administrator user.
6. Verify that you cannot log in to the database as the default administrator user.
7. Verify that you cannot log in to the web server as the default administrator user.

Chapter 7

Assembling CM Sites

CM sites can be assembled once their basic components are created. The basic components are the users (including their ACLs and roles) and the data model. Optionally, you can enable auxiliary user interfaces such as Sites Doclink and Sites Desktop. This chapter shows you how to work with sites (create, edit, and delete sites); how to manage site users (add, edit, and delete users); and how to manage asset types for the sites (enable and remove asset types).

This chapter contains the following sections:

Overview

Assembling a site requires you to assign users to the site, assign content (asset types) to the site, and optionally enable the Sites Desktop and Sites DocLink interfaces. In this chapter, you will create sites, associate users to the sites, and assign asset types to the sites. In later chapters, you will develop each site by tuning site users' permissions to content on the site.

Working with Sites

- Creating a Site
- Obtaining Site Configuration Information
- Editing a Site
- Deleting a Site

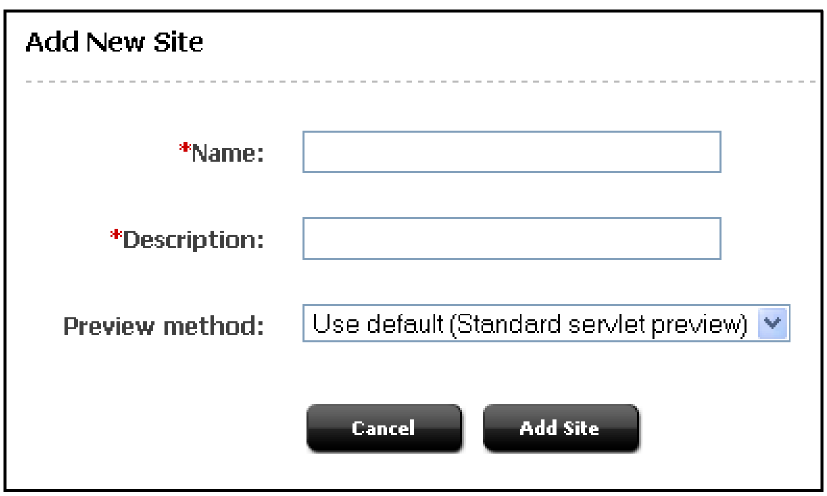
Creating a Site

Creating a site means specifying a site name, description, and a method for previewing the content that the site will display.

To create a site

1. In the **Admin** tab, expand **Sites** and double-click **Add New**.

The "Add New Site" form appears.



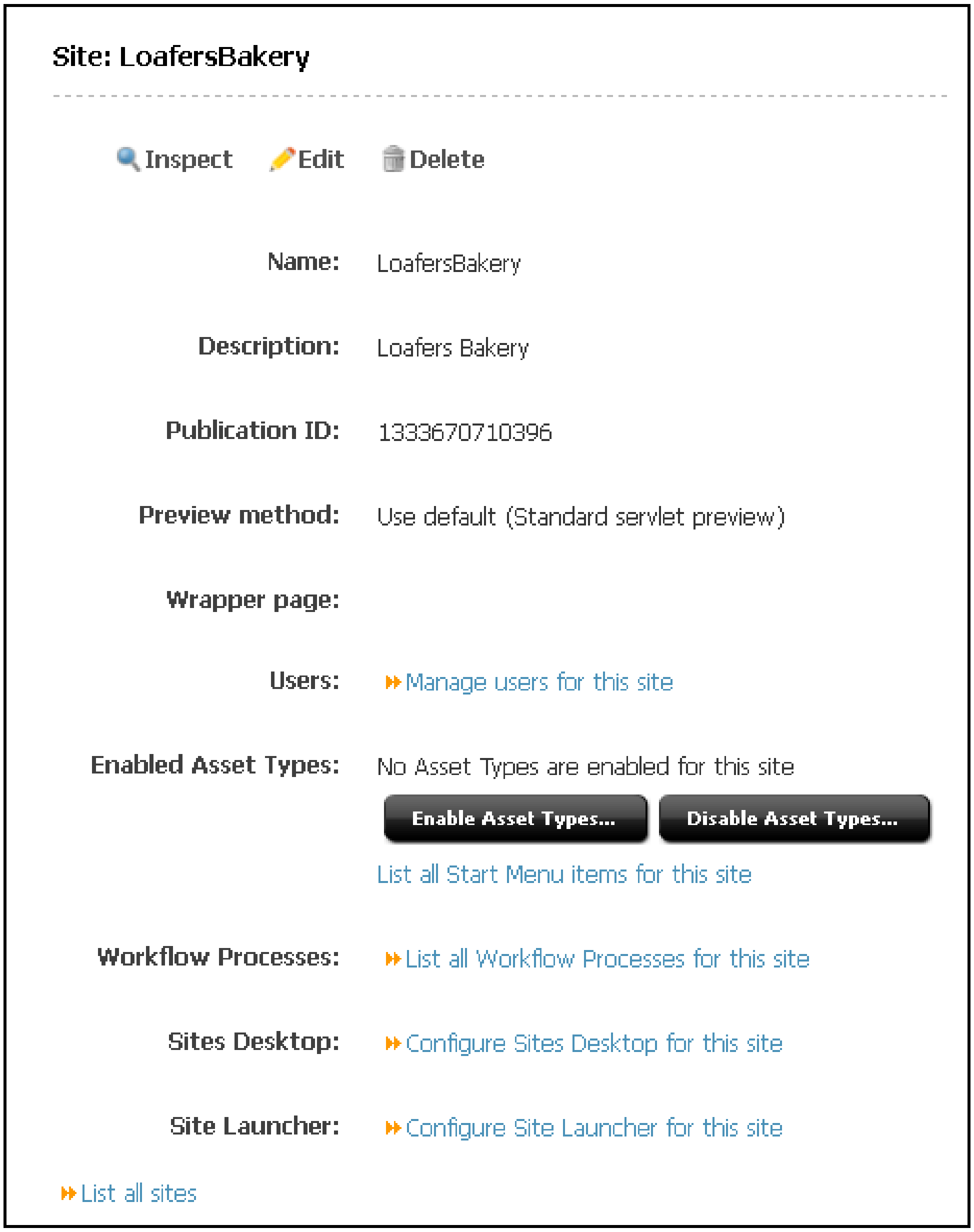
2. In the **Name** field, enter a unique name of up to 255 characters.

Note

Be certain that you have entered a correct name for the site. Although you can edit the description of a site, you cannot change the name of a site after you have created it.

3. In the **Description** field, enter the name of the site as you want it to appear in the WebCenter Sites interfaces (up to 64 characters). This display name will be used in site lists throughout the user interface.
4. In the **Preview method** field, select the method how you will preview the site content.
5. Click **Add Site**.

   The “Site” form appears, where you can execute a number of operations; for example, inspect, edit, or delete the site, manage users, or configure Site Launcher to replicate the site.



When the new site is added to the **Sites** node, WebCenter Sites creates, below the site name, a set of sub-nodes that prompt you for site components:

- Click **Manage users for this site** to manage site users.
- Click **Enable Asset Types** to select asset types for the Web site; click **Disable Asset Types** to remove asset types. Asset types are created by developers.
- Click **List all Start Menu items for this site** to display the Start Menu items which have been created for the site.
- Click **List all Workflow Processes for this site** to manage the workflows for the site.
- Click **Configure Sites Desktop for this site** prompts you to configure and enable Sites Desktop.
- Click **Configure Site Launcher for this site** prompts you to configure and enable Site Launcher.

Each sub-node is equipped with a linking mechanism that allows you to associate the sub-node, and therefore the site, with data in the configuration pool. Because of this linking mechanism, the sub-nodes cannot be deleted or supplemented with custom sub-nodes. Selecting a sub-node allows you to create All data required by the sub-nodes must be entered in to the configuration pool, described next.

Note that the procedure above only adds new sites.

- To make a site usable, you must associate users and asset types (created by developers) with the site. If you plan to use workflow processes, you should create those processes as well.
- To copy or migrate the site to another WebCenter Sites system, use the RealTime publishing method. See Part 4, “RealTime Publishing.

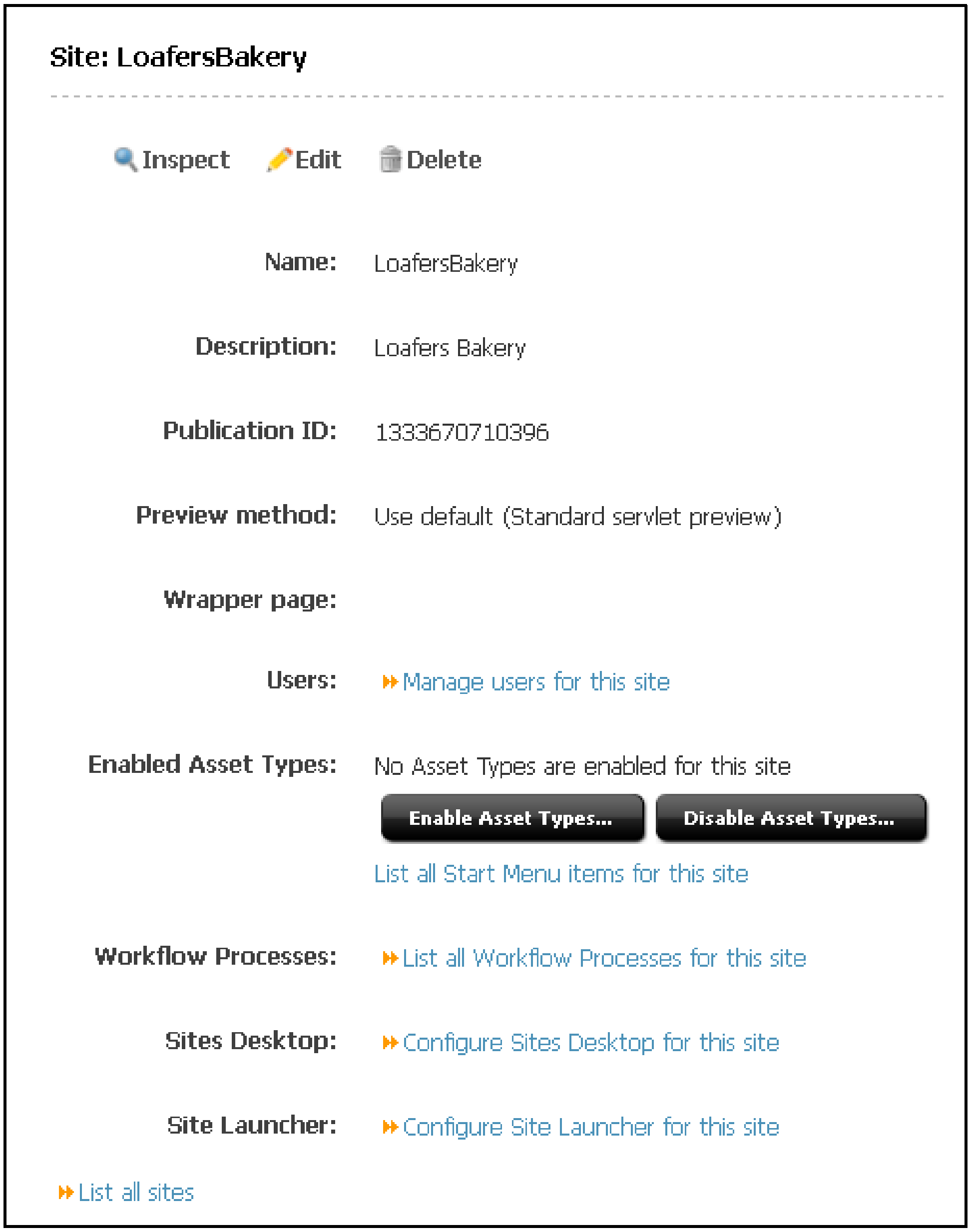
Obtaining Site Configuration Information

In certain situations, you will need to know how a site has been configured—for example, which asset types and workflow processes are enabled, and who are the site users.

To obtain site configuration information

1. In the **Admin** tab, expand **Sites** and double-click the desired site.

2. In the “Site” form, select the option that provides the information you are looking for.



Editing a Site

Editing a site means changing its description and/or preview method. If you need to change the name of a site, you must delete the site and create a new one with the correct name.

To edit a site description

1. In the **Admin** tab, expand **Sites** and double-click the desired site.
2. In the “Site” form, click **Edit**.

3. In the “Edit” form, do the following:
   a. In the **Description** field, make the appropriate changes.
   b. In the **Preview Method** field, select a different preview method, if necessary.
4. Click **Save**.

Deleting a Site

When you delete a site, all configuration information for that site is either removed or unshared (if the site shared data with other sites). The following configuration information is affected by the deletion of a site:

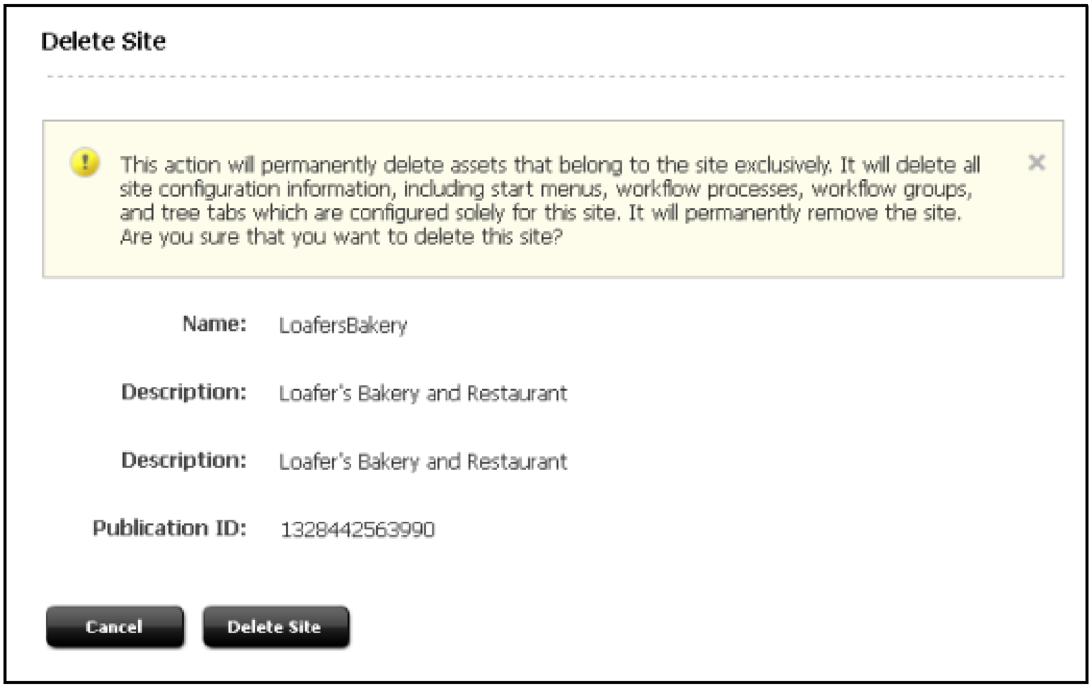
- Start Menu items that were enabled for the site are deleted or unshared
- Smartlists are deleted or unshared
- Publishing destinations are unshared
- Users are removed from the site
- Workflows that were shared are unshared or deleted
- Template assets that were shared have their SiteEntry removed
- Asset subtypes that were enabled for that site as well as tree tabs are removed or unshared

Therefore, before deleting a site, be sure that you have shared the assets that you need to keep.

To delete a site

1. In the **Admin** tab, double-click **Sites**.
2. In the list of sites, navigate to the site you want to delete and click its **Delete** icon.

A warning is displayed to inform you that site components will be deleted. The warning identifies the components and prompts you to confirm your intention to delete the site.



3. Click **Delete Site**.

Assigning and Managing Site Users

This section shows you how to grant WebCenter Sites users access to a site, view site users, change users' role assignments on a site, and remove users from a site.

Granting Users Access to a Site (Assigning Roles to Users)

For a user to gain access to a site, the user must be assigned at least one role on the site. Assigned roles associate the user with the site and allow the user to log in to the site. When one of the user's assigned roles is also specified for an interface function on the site, the user gains access to the interface function.

Note

In this section, we assume that you have created at least one user, a role, and a site. If you need to create one or another, follow the steps in:

- "Creating a New User," on page 86
- "Creating a Role," on page 81
- "Creating a Site," on page 106

To give a user access to a site

1. In the **Admin** tab, expand **Sites**, then expand the desired site.
2. Double-click **Users**.

   The "User Role Management" form appears.

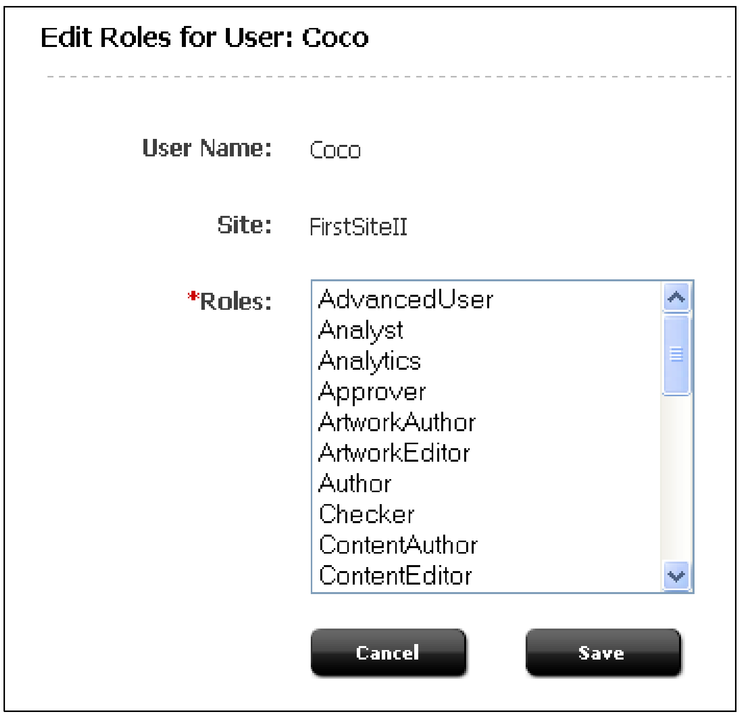
3. Enter the desired user name and click **Select**.

The "User Role Management" form displays the user name. No roles are listed next to the user name.

> **Note**
>
> If you need a list of system-wide users, open the list as follows:
>
> Expand **Management Tools** and double-click **User**. Leave the "Enter User Name" field blank, select **Modify User,** and click **OK**.

4. Click the **Edit** icon next to the user name.

   WebCenter Sites displays the "Edit Roles" form:



5. Select one or more roles from the list. To select multiple roles, **Ctrl-click** each desired role. To select a range of roles **Shift-click** the first and last role in the range.

Note

- Consider the user's type. A general administrator, site administrator, and regular user require different sets of system-defined roles. For example, a regular user requires access to the content contributor's interface. In this instance, select the **SitesUser** role. For additional help with selecting system-defined roles, see "System Roles," on page 529.
- Selecting a role (or roles) makes the user a member of the site that you chose to work with in step 1. It does not necessarily enable the user to manage the site's content. To manage content, the user must have access to interface functions (such as **Search**) on the site.

  If you know which roles are allowed to invoke which interface functions on the user's site, select the roles in this step. Otherwise, the roles that you select here you will also specify (in later chapters) for interface functions that the user needs. The functions may be start menu items, tree tabs, workflow processes, and asset approval and publishing functions.

  For more information about assigning user roles, see "Working with Roles," on page 81. See also Chapter 8, "Managing Access to CM Site Components" and chapters dealing with workflow and publishing.

6. Click **Save**.
7. Give the user access to site components by following procedures in Chapter 8, "Managing Access to CM Site Components."

> **Note**
>
> Remember that the user cannot perform the functions available to her roles unless the user also has adequate ACL permissions to operate on the database tables. (While roles allow access to interface functions, ACLs activate those functions by allowing access to the database tables.) For more information about roles and their relationship to ACLs, see page 31.

Viewing Site Users and Reassigning Roles

To view the list of users and their roles on a given site, and to reassign roles

1. In the **Admin** tab, expand **Sites**, then expand the desired site.
2. Under the desired site, double-click **Users**.
3. In the “User Role Management” form, leave the user name field blank and click **Select**.

   WebCenter Sites displays a list of all the users currently assigned roles in the selected site:


User Role Management

Site: HelloAssetWorld

Select the user to modify:

User Name	Roles
Bobo	AdvancedUser, WorkflowAdmin, SitesUser, GeneralAdmin
Coco	HelloDesigner, AdvancedUser, SitesUser, GeneralAdmin
Flo	AdvancedUser, SitesUser, HelloEditor
fwadmin	HelloDesigner, AdvancedUser, SiteAdmin, WorkflowAdmin, SitesUser, HelloEditor, HelloAuthor, GeneralAdmin
Joe	AdvancedUser, SitesUser, HelloAuthor
Moe	AdvancedUser, SitesUser, HelloAuthor

Manage users for this site

4. (Optional) If you want to reassign the user’s roles, do the following:
   a. Navigate to the desired user name and click its **Edit** icon.
   b. In the “Edit Roles” form, select (and/or deselect) one or more roles from the list, then click **Save**.
   c. Repeat steps a and b for each additional user, as necessary.

Deleting Users from a Site

You can delete a user from a site without deleting that user from the system.

To delete a user from a site

1. In the **Admin** tab, expand **Sites**, then the desired site.
2. Under the desired site, double-click **Users**.
3. In the "User Role Management" form, enter the desired user name and click **Select**.
4. In the form that appears, click the **Delete** (trash can) icon next to the user.
   WebCenter Sites displays a warning message.
5. Click **Delete User from Site**.

Enabling and Managing Asset Types

This section shows you how add asset types to a site (enable the asset types), and delete them from a site.

Enabling Asset Types for a Site

Enabling an asset type means associating the asset type with a chosen site or sites. Once the assignment is made, the asset type itself must be made available to users through an interface function such as a start menu item or a tree tab, as shown in Chapter 8, "Managing Access to CM Site Components."

> **Note**
>
> The following procedure describes how to enable an asset type for use. Enabling an asset type for use with Sites Desktop is described in "Configuring the Word Asset Types for Sites Desktop," on page 408.

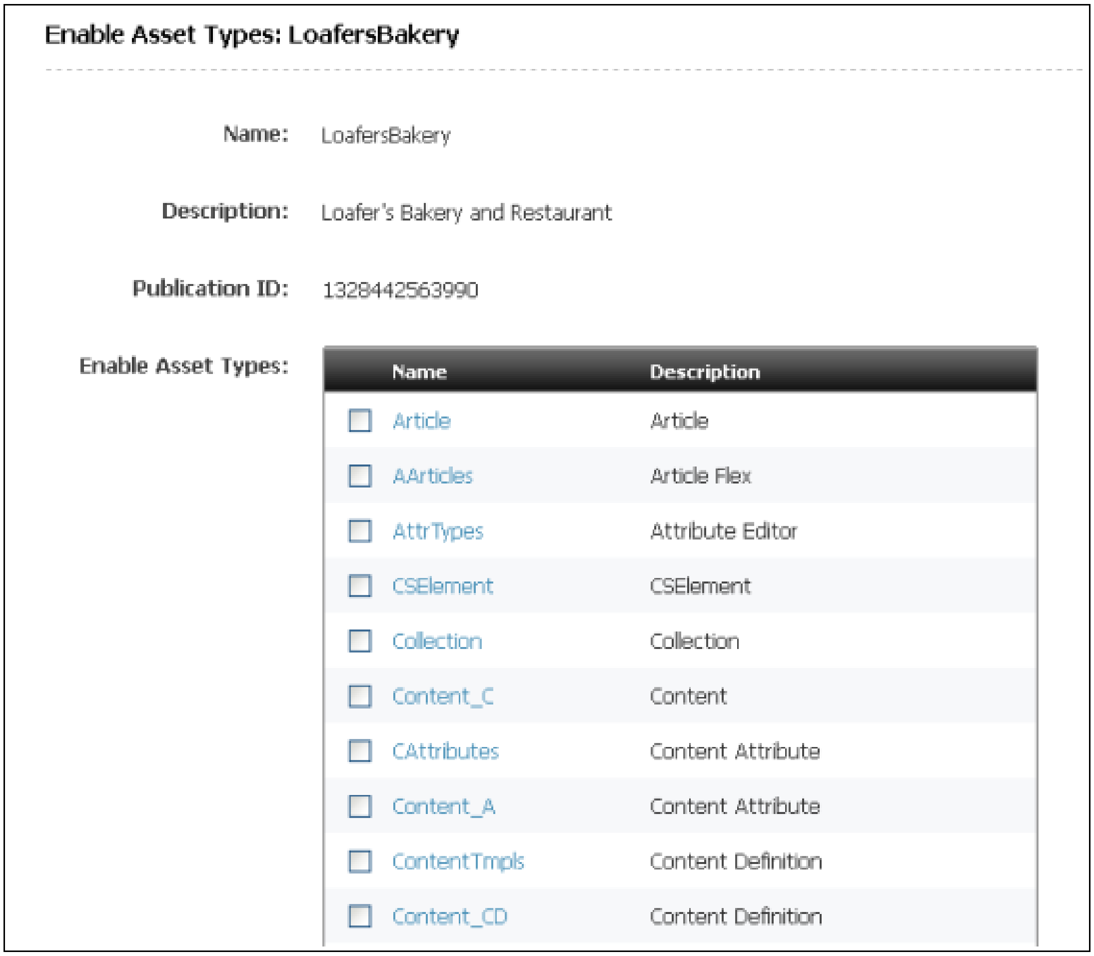
To enable asset types for a site

1. In the **Admin** tab, expand **Sites**, then expand the desired site.

> **Note**
>
> If you have the SiteAdmin role, but not the GeneralAdmin role, select the **Site Admin** tab.

2. Under the desired site, expand **Asset Types** and double-click **Enable**.

WebCenter Sites displays the "Enable Asset Types" form, listing asset types that have not yet been enabled in this site:


Enable Asset Types: LoafersBakery

Name: LoafersBakery

Description: Loafer's Bakery and Restaurant

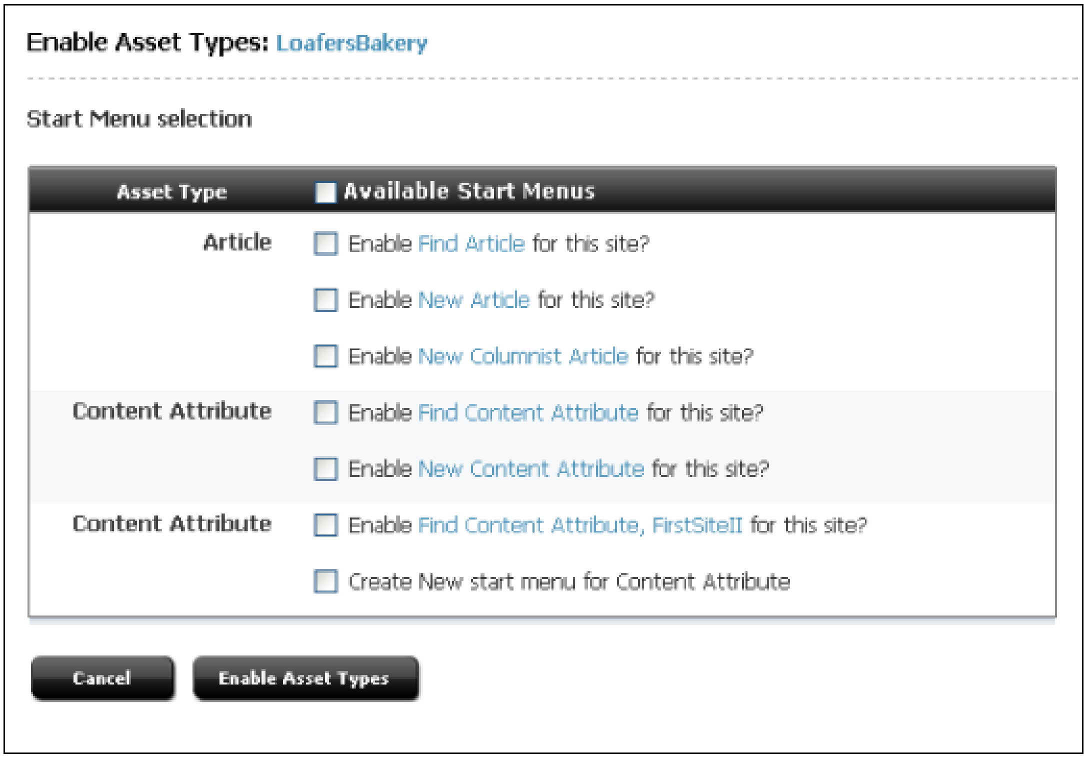
Publication ID: 1328442563990

Enable Asset Types:

Name	Description
Article	Article
AArticles	Article Flex
AttrTypes	Attribute Editor
CSElement	CSElement
Collection	Collection
Content_C	Content
CAttributes	Content Attribute
Content_A	Content Attribute
ContentTmpls	Content Definition
Content_CD	Content Definition

3. Select the asset types which you want to enable for this site.
4. Click **Enable Asset Types**.

5. (Optional) In the form that follows, check the box next to each start menu item that you would like WebCenter Sites to create. (For more information on start menu items, see “Managing Access to Asset Types via Start Menus,” on page 122.)



If you choose not to generate start menu items at this time, you must manually create them later. No one can create assets of the asset types you just enabled until the corresponding start menu items are created.

6. Click **Enable Asset Types**.
7. The asset types are now enabled for the site. If you need to create start menu items for them, continue to “Managing Access to Asset Types via Start Menus,” on page 122.

> **Note**
>
> To view the assets in the Contributor interface’s trees (**Site Tree**, **Content Tree**, and **My Work** tree) you also need to manually create the tree tabs for the assets.

Removing Asset Types from a Site

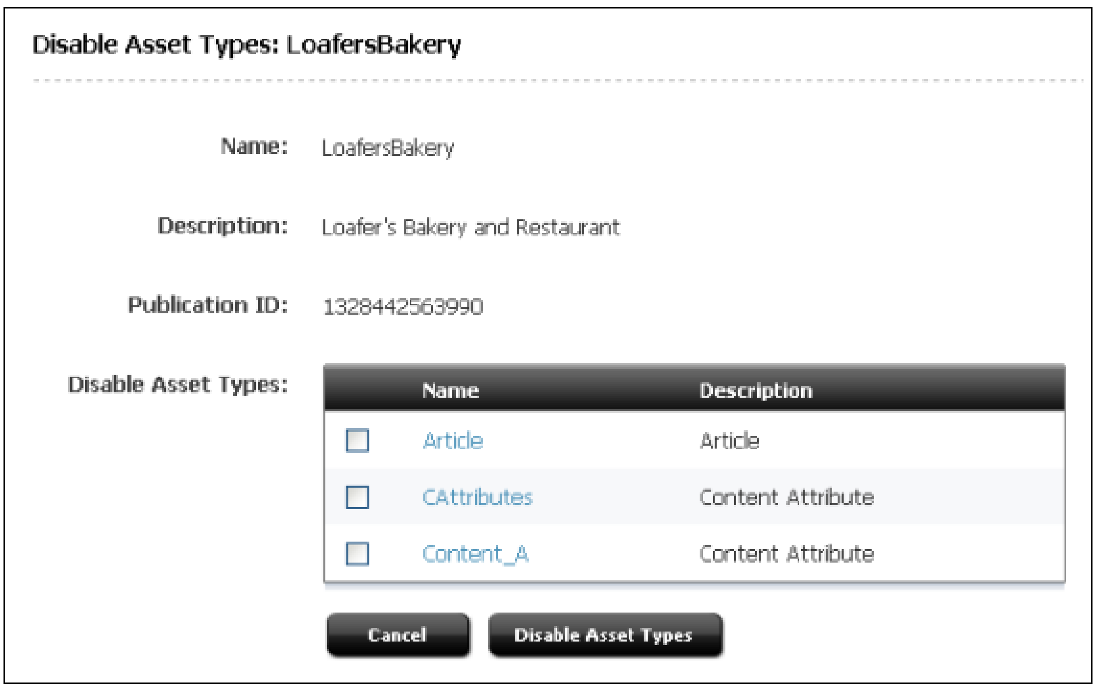
To remove asset types from a site

1. In the **Admin** tab, expand **Sites**, then the desired site.

> **Note**
>
> If you have the SiteAdmin role, but not the GeneralAdmin role, select the **Site Admin** tab.

2. Under the desired site, expand **Asset Types** and double-click **Disable**.

   WebCenter Sites displays the "Disable Asset Types" form, showing asset types that have been enabled for the selected site:



3. Select the asset types that you want to remove from the site.
4. Click **Disable Asset Types**.

> **Note**
>
> If start menu items or workflow processes exist for the asset type that you disabled on a site, edit the start menu items or workflow processes so that they reflect the change.

Enabling and Managing User Interfaces

If you plan to enable the Sites Desktop and Sites DocLink interfaces for users on certain sites, follow instructions in Chapter 22, "Configuring the User Interfaces."

Chapter 8

Managing Access to CM Site Components

This chapter shows you how to manage access to site components. It is assumed that users have already been given the permission to access the site containing those components.

This chapter contains the following sections:

Overview

Once users are granted permission to access a site (as shown in Chapter 7, "Assembling CM Sites") their ability to work with the site's content is managed through:

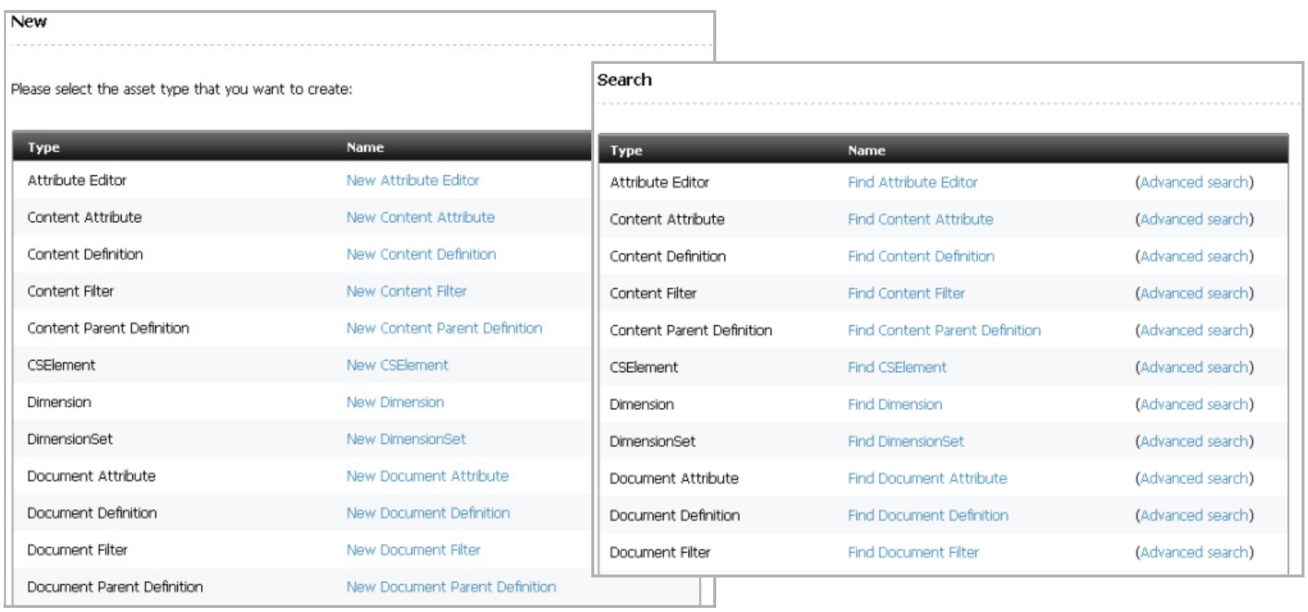
- Start menus, which determine whether the users can create, search for, and edit assets of certain types on the site to which the users are assigned.
- Permissions to assets (that are not part of a workflow process).
  For such assets, you can create policies that define which roles have permissions to perform which functions on assets.
- Permissions to WebCenter Sites' tree.
  You can control users' permissions to the tree, tabs in the tree, nodes in the tabs, and items in the nodes.

Roles are the components that you invoke in order to associate users with interface functions and the content that the users must manage.

Managing Access to Asset Types via Start Menus

A start menu is a list of hyperlinked asset types, such as you see in Figure 5.

Figure 5: New and Search Start Menus in the Admin Interface



New

Please select the asset type that you want to create:

Type	Name
Attribute Editor	New Attribute Editor
Content Attribute	New Content Attribute
Content Definition	New Content Definition
Content Filter	New Content Filter
Content Parent Definition	New Content Parent Definition
CSElement	New CSElement
Dimension	New Dimension
DimensionSet	New DimensionSet
Document Attribute	New Document Attribute
Document Definition	New Document Definition
Document Filter	New Document Filter
Document Parent Definition	New Document Parent Definition

Search

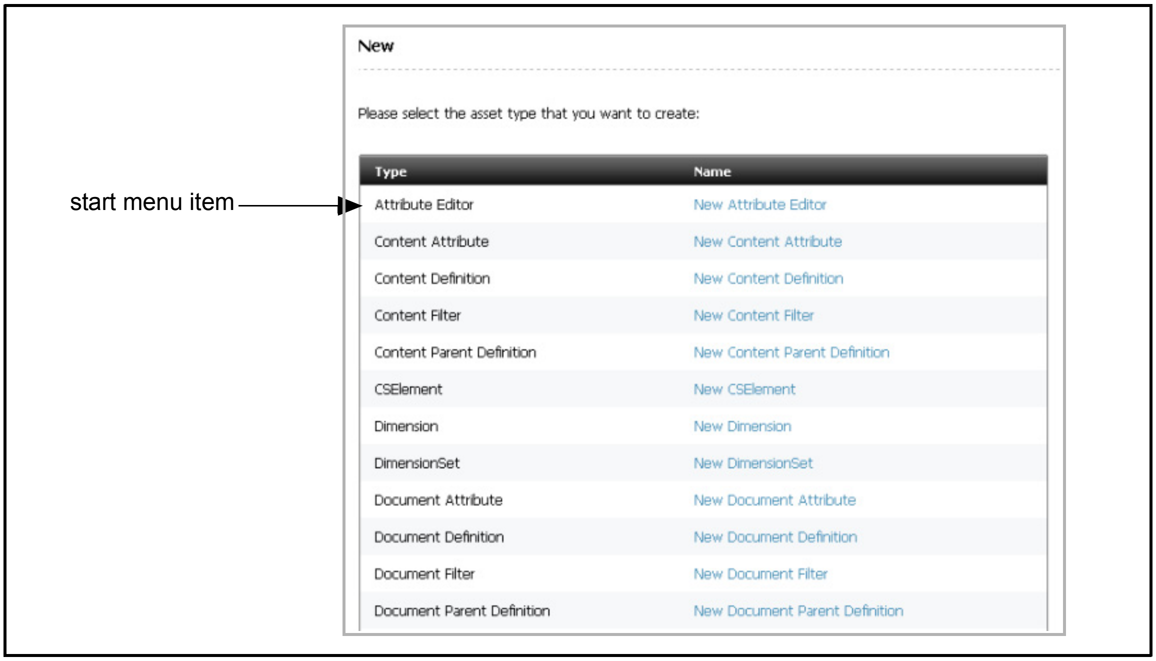
Type	Name	
Attribute Editor	Find Attribute Editor	(Advanced search)
Content Attribute	Find Content Attribute	(Advanced search)
Content Definition	Find Content Definition	(Advanced search)
Content Filter	Find Content Filter	(Advanced search)
Content Parent Definition	Find Content Parent Definition	(Advanced search)
CSElement	Find CSElement	(Advanced search)
Dimension	Find Dimension	(Advanced search)
DimensionSet	Find DimensionSet	(Advanced search)
Document Attribute	Find Document Attribute	(Advanced search)
Document Definition	Find Document Definition	(Advanced search)
Document Filter	Find Document Filter	(Advanced search)

Clicking an asset type name opens a form for either creating an asset of the selected type, or searching for assets of the selected type. For example, clicking an asset type in a **New** start menu opens a form for creating an asset of the selected type. Clicking an asset type in the **Search** start menu opens a form that is used to search for and edit assets of the selected type.

The start menus shown in Figure 5 are accessible from the Admin Interface by the user clicking the **New** and **Search** buttons, at the top left of the interface. You, as the administrator, grant a user permission to a start menu item by assigning the same role to the user and the start menu item.

Start Menu Items

A **start menu item** is an option in a start menu. A start menu item names the type of asset a user can create, copy, or search for. For example, the **New** start menu shown below belongs to the Admin Interface and displays 16 start menus items to the user who is authorized to work with the items. The user can create assets of the types that are listed in the menu.



In the Admin Interface, start menu items are displayed in a list when users click **New** or **Search** in the button bar.

You configure the start menu items to determine the following about the new assets that are created with them:

- Which roles on which sites have access to the start menu item. Specifying roles controls which content providers can create or search for assets of this type on the specified sites.
- Field/value pairs. You can supply values for certain fields so that those fields are automatically filled in whenever a content provider creates a new asset with the shortcut.

  For example, for basic assets, it might be useful to have the start menu item set a pre-determined value for the **Dimension**, **Description**, **End Date**, **Start Date**, **Subtype**, **Template** or **User** fields (or some combination). If your basic assets have associations that can be different based on the subtype of the asset, be sure to create start menu items that set a subtype for new assets so that the only the appropriate associations appear in the “New” form.

  For flex assets, it is useful to have start menu items that set the flex definition for new flex assets so the content providers have fewer steps.

- Which workflow process to assign by default to new assets created with the start menu item and the users who participate in that workflow process.

Start menu items also determine whether certain operations can be performed on an asset type that is listed in a tree tab (in the Admin interface). If an asset type is displayed on a tree tab, but there is no start menu item created for it, the only options listed in the right-mouse menu for the asset type are **Refresh, Edit**, and **Inspect**. To have the **New** option included in the right-mouse menu, a **New** start menu item must be created for the asset type.

For Different Kinds of Users

If you have different groups of content providers who create different types of content, you could create start menu items for each type.

For example, suppose that you have a group of writers who create business articles. You could create a start menu item that creates a new article, sets the Category to Business, sets the template to the correct template for articles on the business page, and places it in a workflow for business articles. Additionally, if you create a role for business writers, you can configure the start menu item so that only users with the business writer role can use this start menu item.

For Flex Assets

For flex assets, you can set the definition of a flex asset with the start menu item.

For example, suppose that your public site were a catalog of household goods and that your flex assets were products. You could create a start menu item named "Toaster" that sets the product definition to "toaster" for new toaster products, a start menu item named "glassware" that sets the appropriate product definition for new glassware products, and so on. A start menu item that sets the flex definition in this way eliminates a step for your content providers, thereby reducing the chance that they will select the wrong flex definition by mistake.

Creating Start Menu Items

Managing user's access to asset types on a site entails creating start menu items for the asset types. Start menu items determine which users can create and search for assets of specific types on a specific site. WebCenter Sites supports four kinds of start menu items:

- **New** – creates a link for the asset type on the list that is displayed when you click **New** in the button bar. If you do not create **New** items for the asset types that are enabled for a site, no one can create assets of those types. For information about creating **New** start menu items, see "Creating "New" Start Menu Items," on page 125.
- **Search** – creates a link for the asset type on the list that is displayed when you click **Search** in the button bar. If you do not create **Search** items for the assets that are enabled for a site, no one can search for and edit assets of those types in any of the Sites interfaces. For information about creating **Search** start menu items, see "Creating "Search" Start Menu Items," on page 129.
- **Sites Desktop** – makes the asset type available to content providers who use the Sites Desktop interface instead of one of the browser-based interfaces. For information about creating Sites Desktop start menu items, see "Configuring the Word Asset Types for Sites Desktop," on page 408.
- **Sites DocLink** – makes the asset type available to content providers who use the Sites DocLink interface instead of one of the browser-based interfaces. For information

about creating Sites DocLink start menu items, see "Configuring the Sites DocLink Asset Types," on page 416.

Creating "New" Start Menu Items

Before you begin creating start menu items, note the following:

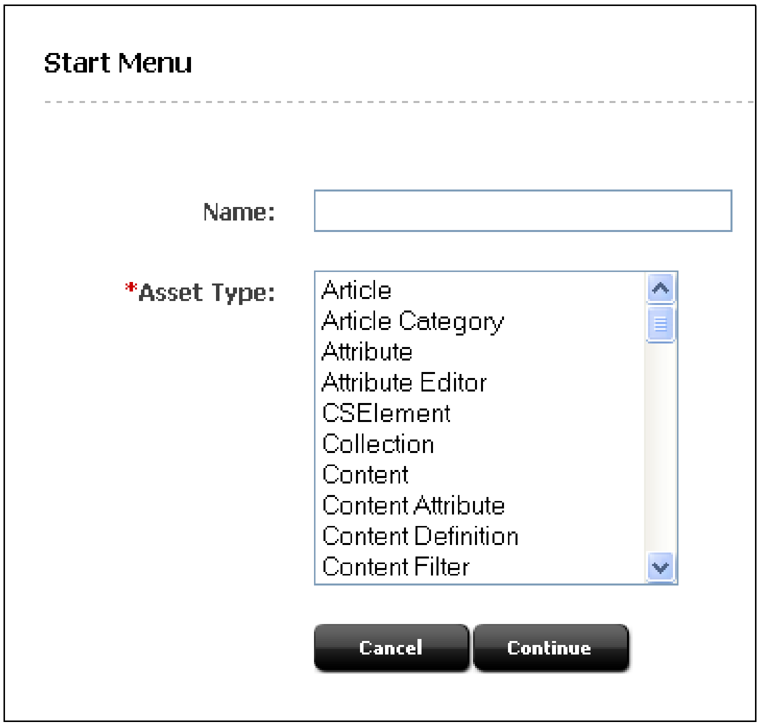
- Create start menu items for only the asset types that are enabled for that site. If the assets are not enabled, go back to Chapter 7, "Assembling CM Sites" and complete the steps in "Enabling Asset Types for a Site," on page 116.
- If you want to specify a default workflow process, create the workflow process first. For information about workflow, see Chapter 9, "Creating and Managing Workflow Processes."
- Talk with your developers to ascertain which fields are used by queries, collections, or other design elements for the online site. That way, you can specify useful default values for those fields.
- Talk with your developers about specific dates when the site will change so that you can create start menu items with appropriate default start and end dates for Site Preview. Also note that for published assets, the end date determines when the assets automatically expire from the web site. Speak with developers about their needs when formulating end date defaults.
- Talk with your developers to find out which templates are appropriate for the various subtypes of asset types that you are creating start menu items for.
- You **cannot** set values for the following kinds of fields:
  - An upload field or any field that writes data to a URL column.
  - Fields that accept more than one value.
  - Association fields.
  - Flex attribute fields for flex and flex parent assets.

  To illustrate, the following asset types cannot have set values in the Start Menu:
  - Visitor Attribute
  - CS Element
  - Dimension
  - Dimension Set
  - Flex Attribute
  - Flex Child Definition
  - Flex Filter
  - Flex Parent Definition
  - History Attribute
  - History Definition
  - Page Attribute
  - Page Definition
  - Page Filter
  - SiteEntry

- Template
- Attribute Editor

To create a "New" start menu item

1. In the **Admin** tab, expand **Start Menu**, then double-click **Add New**.
2. In the "Start Menu" form, do the following:



   a. In the **Name** field, enter the name of the item, using up to 64 characters. Although it is not required, it is a good idea to start the name with the word "New" (for example, "New Article" or "New Template"). This is to help the user later realize that the start menu item is for a new asset.

   The name used in the New field will be displayed on the New and Search screens if the **Description** field (described in step 4) is left blank.

   b. In the "Asset Type" field, select the type of asset for which you are creating the Start Menu item.

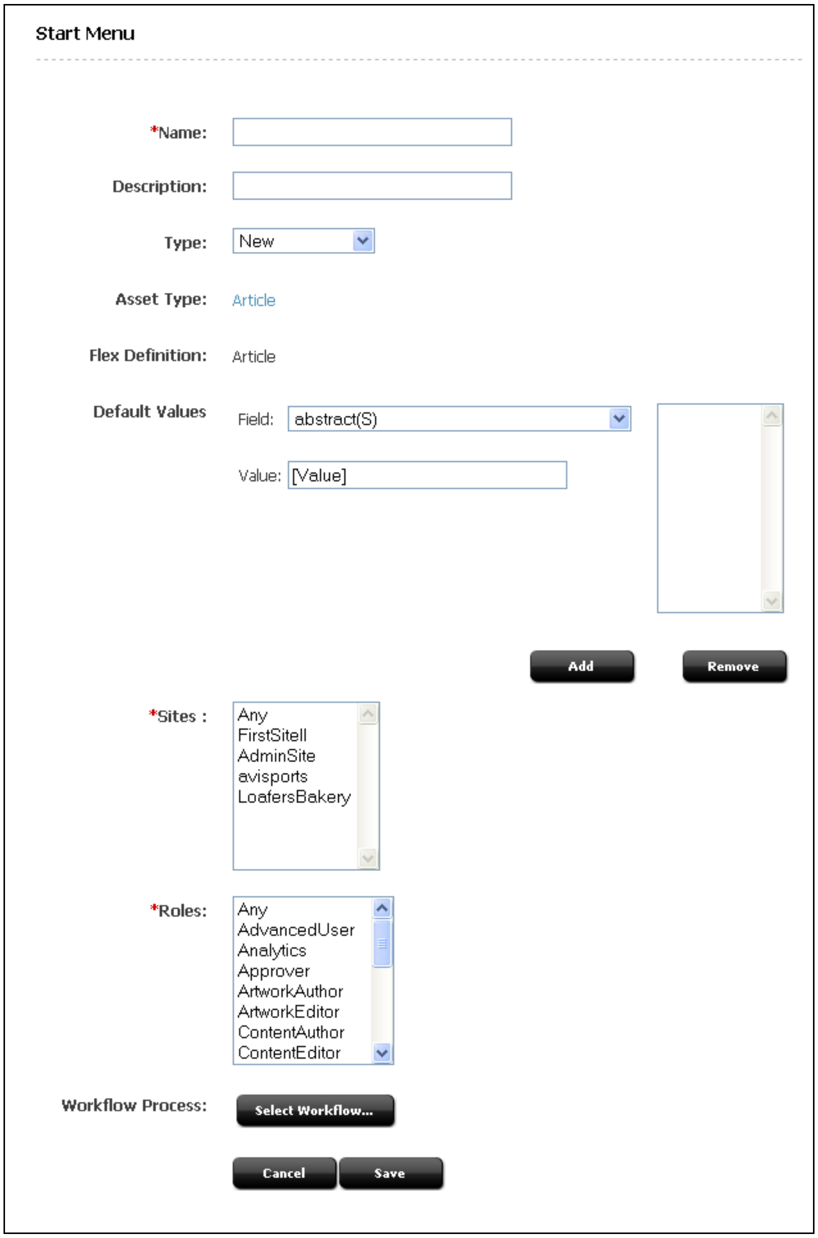
   c. Click **Continue**.

3. If you have chosen a flex asset type, the **Flex Definition** field will appear. If you did not select a flex asset type, skip to step 4.

   a. In the **Flex Definition** field, select the asset subtype.

   b. If the screen does not automatically reload, click **Continue**.

4. In the next “Start Menu” form, fill in the fields as explained below:



a. In the **Description** field, enter a short description, using up to 255 characters. The name entered in the **Description** field will be displayed on the New and Search screens.

b. In the **Type** field, select **New**.

c. Note that the **Asset Type** field is an information-only field that displays your choice of asset type from the previous screen.

d. (Optional) In the **Default Values** section, complete the following steps if you want field values to be set automatically when a content provider creates an asset with this start menu item.

1) Select a field from the **Field** drop-down list. This list displays all the fields for assets of this type; for example, category, source, and template.
2) In the **Value** field, enter or select a value.

> **Notes**
>
> - All values are case-sensitive.
> - WebCenter Sites does not validate the value against the field that you are setting it for. Therefore, you must be sure that you have entered an appropriate value.

3) Click the **Add** arrow to add the value to the field.
4) Repeat the previous three steps, as necessary.

> **Note**
>
> When you set a field value through a start menu item, the field does not become a read-only field. That is, no matter what value you set in the "Start Menu" form, that value can be overridden by the content provider who creates an asset through the start menu item. The values that you specify are for convenience only.

e. In the **Sites** field, select which sites can use this start menu item. Use **Ctrl-click** to select more than one site.

f. In the **Roles** field, select the roles that can have access to the start menu item. Use **Ctrl-click** to select more than one role.

> **Note**
>
> If you are assigning a default workflow with this start menu item, the roles that you select in this step must match the roles that are (or will be) authorized for the start step in the workflow process.

g. (Optional) To configure workflow process details for assets created with this start menu item, complete the following steps:

1) Click **Select Workflow**.
2) In the "Select Workflow" form, select the appropriate workflow process from the drop-down list and then click **Select Workflow**.
3) In the "Set Participants" form, select at least one user for each role that appears, and click **Set Participants**.
4) Click **Continue**.

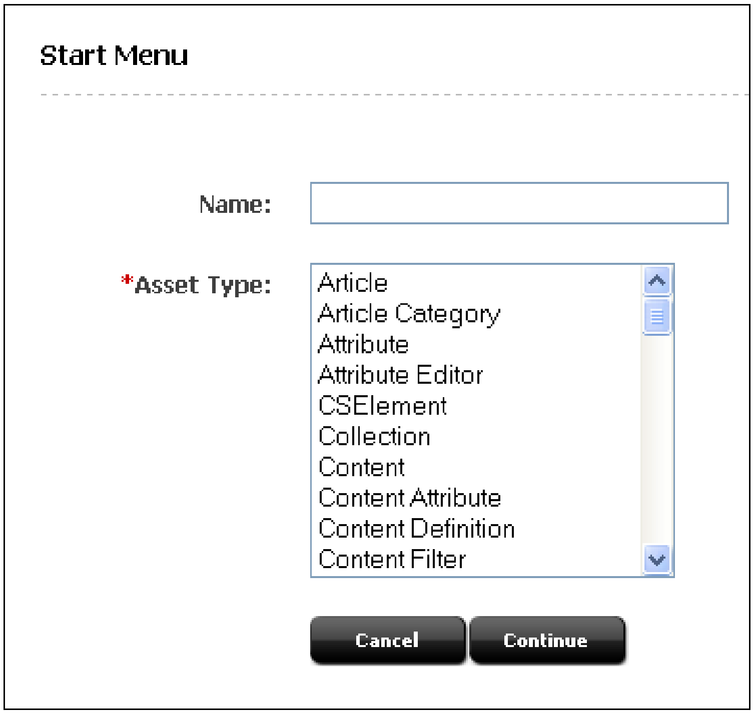
The system saves the workflow information and redisplays the "Start Menu" form.

5. Click **Save**.

Creating "Search" Start Menu Items

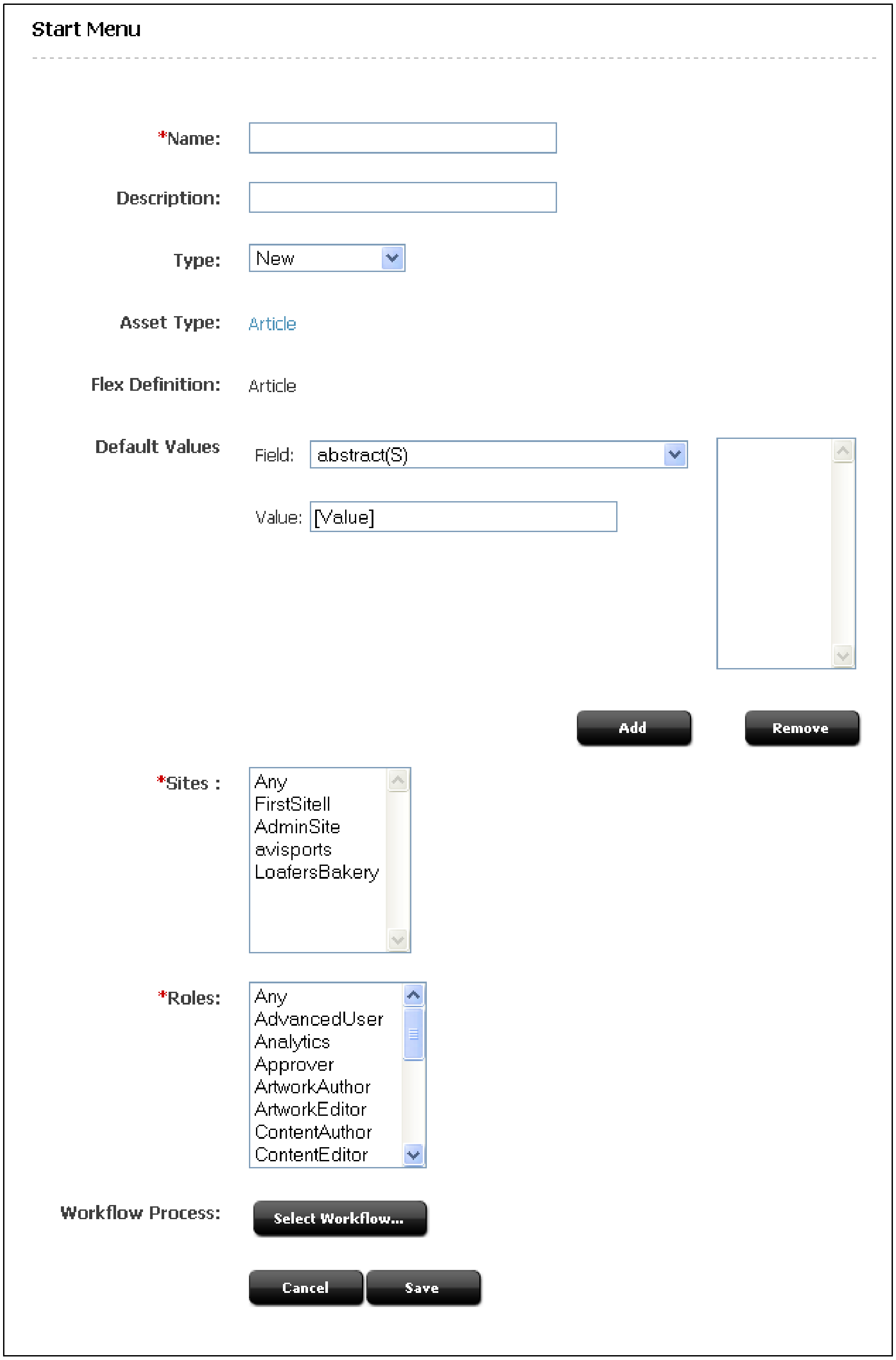
To create a "Search" start menu item

1. In the **Admin** tab, expand **Start Menu**, then double-click **Add New**.
2. In the "Start Menu" form, do the following:



   a. In the **Name** field, enter the name of the item, using up to 64 characters. Although it is not required, it is a good idea to start the name with the word "Find" (for example, "Find Article" or "Find Template"). This is the name that is displayed on the **Search** list.
   b. In the "Asset Type" field, select the type of asset for which you are creating the Start Menu item.
   c. Click **Continue**.
3. If you have chosen a flex asset type, the **Flex Definition** field will appear. If you did not select a flex asset type, skip to step 4.
   a. In the **Flex Definition** field, select the asset subtype.
   b. If the screen does not automatically reload, click **Continue**.

4. In the next “Start Menu” form, fill in the fields as explained below:



5. In the **Description** field, enter a short description, using up to 255 characters.

6. In the **Type** field, select **Search**.

7. Note that the **Asset Type** field is an information-only field that displays your choice of asset type.
8. (Optional) In the **Default Values** section, complete the following steps if you want field values to be set automatically when a content provider creates an asset with this start menu item.
    a. Select a field from the **Field** drop-down list. This list displays all the fields for assets of this type; for example, category, source, and template.
    b. In the **Value** field, enter or select a value.

    Notes

    - All values are case-sensitive.
    - WebCenter Sites does not validate the value against the field that you are setting it for. Therefore, you must be sure that you have entered an appropriate value.

    c. Click the **Add** arrow to add the value to the field.
    d. Repeat the previous three steps, as necessary.

    Note

    When you set a field value through a start menu item, the field does not become a read-only field. That is, no matter what value you set in the “Start Menu” form, that value can be overridden by the content provider who creates an asset through the start menu item. The values that you specify are for convenience only.

9. In the **Sites** field, select which sites can use this start menu item. Use **Ctrl-click** to select more than one site.
10. In the **Roles** field, select the roles that can have access to the start menu item. Use **Ctrl-click** to select more than one role.

    Note

    If you are assigning a default workflow with this start menu item, the roles that you select in this step must match the roles that are (or will be) authorized for the start step in the workflow process.

11. (Optional) To configure workflow process details for assets created with this start menu item, complete the following steps:
    a. Click **Select Workflow**.
    b. In the “Select Workflow” form, select the appropriate workflow process from the drop-down list and then click **Select Workflow**.

c. In the “Set Participants” form, select at least one user for each role that appears, and click **Set Participants**.

d. Click **Continue**.

   The system saves the workflow information and redisplays the “Start Menu” form.

12. Click **Save**.

Editing Start Menu Items

To edit a Start Menu item

1. In the **Admin** tab, expand **Start Menu**, then expand the node the desired Start Menu item is located under. Double-click the desired Start Menu item.

   The “Start Menu Item” screen appears.

2. Click **Edit**.

The “Start Menu” form with the stored information for the item displays.

3. Make the changes to the item.
4. Click **Save**.

Creating Start Menu Items for Sites Desktop

For information about creating Sites Desktop start menu items, see “Configuring the Word Asset Types for Sites Desktop,” on page 408.

Creating Start Menu Items for Sites DocLink

For information about creating Sites DocLink start menu items, see "Configuring the Sites DocLink Asset Types," on page 416.

Displaying a List of All Start Menu Items

In addition to the individual start menu nodes that appear for each of the asset types that exist on your Sites system, there is a top-level **Start Menu** node in the **Admin** tab. Use this node when you want to see a list of all the start menu items in the system.

Additionally, there is a link to the list of all the start menu items from the "Inspect" form for any individual start menu item.

Setting Access Permissions to Assets

WebCenter Sites provides a system by which you can set access permissions to assets that are not part of a workflow process. For such assets, you can create policies and, within them, define which roles have permissions to perform which functions on assets. For example, a policy for the function "copy" might specify which roles are allowed to copy an asset. A different policy for "copy" might specify which roles are denied the ability to copy an asset. For flex assets, inheritable policies can be set on parents, meaning that the policies will be inherited by the children.

Functions that can be performed on assets, by default, are the following: checkout, copy, edit, delete, rollback, share, approve, and build. (For definitions of the functions, see "VI. Determine the Function Privileges," on page 173).

All of the above-listed functions are defined in the `futuretense_xcel.ini` file, as values of the property `xcelerate.authorizefunction` (located in the **Authorization** tab, if you are using the Property Editor). You can specify additional functions (which are listed in the **Authorization** tab).

To summarize, access permissions serve a three-fold purpose:

- Extend the workflow function privileges system, making it possible to set permissions for assets when they are not in workflow
- Introduce new functions (authorize, inspect, and preview) for viewing the contents of assets that are not in workflow
- Introduce a hierarchical permission inheritance model for assets that are not in workflow.

Note

The "access permissions" feature does not apply to other entities, such as pagelets.

Permission Structure

Permission structure determines which functions (for example, edit and view) a role can perform. A user's permissions are managed by assigning him the roles that can perform the functions. WebCenter Sites implements a three-level permission structure: "Grant," "Deny," and "Inherited." "Grant" and "Deny" denote the explicit permissions, that is, they grant or deny the permission to perform a function. "Inherited" means that permissions are neither granted nor denied. Instead, the permissions are inherited from the parent asset's policy, if one exists. If the parent's policy does not exist, then the asset's permissions are inherited from the policy of the parent's parent, and so on.

If permissions are ambiguous, the following considerations may help resolve the ambiguity:

- A permission is granted if at least one of the user's roles grants the permission.
- "Grant" overrides "Deny."
  If one role grants a permission, but another role denies the same permission, the permission is granted.
- A permission is denied if at least one of the user's roles denies the permission, and none of the user's roles grant the permission.

  The above scenarios are summarized in the following table:

	Role 1	Role 2	Result
Permission	Granted		Granted
	Granted	Denied	Granted
	Denied		Denied

- Assets can inherit permissions from parent assets.

  For example, if a parent asset named "Money Market Funds" grants the "delete" permission to a role named `Editor`, and user Bob has that role, then Bob can delete the "Money Market Funds" asset and all of its child assets.
- Asset permissions can override the parent assets' permissions.

  A "Prime Money Market semi-annual report" asset denies the "delete" permission to a certain role. That same role is granted the "delete" permission in the "Prime Money Market" parent asset. This means that Bob cannot delete the report, even though the parent asset grants the "delete" permission.
- If there are no applicable permissions found on a particular asset, then the administrator needs to examine the `futuretense_xcel.ini` file for this permission. In the `futuretense_xcel.ini` file, permissions for a particular function are set for all assets, rather than for a single asset.

  For more information about setting permissions in the `futuretense_xcel.ini`, see "To set access permissions from futuretense_xcel.ini," on page 140.
- Asset permissions overrides `futuretense_xcel.ini` settings.

  For example, if there is a permission `xcelerate.copy.grant` in the `futuretense_xcel.ini` file and the permission to delete asset A is revoked, then the permission to delete asset A is denied.

Types of Authorization

Three types of authorization can be defined on a particular function and a particular asset: workflow function privileges, access permissions, and authorization properties in `futuretense_xcel.ini`, all independent of each other. (For information about workflow function privileges, see Chapter 9, "Creating and Managing Workflow Processes.")

When a user attempts to perform a function on an asset, the system performs a series of authorization checks in the following order, for that function:

For assets in workflow:

1. The system checks the workflow function privileges for the function in the workflow process in which the asset is entered.
2. If no function privilege is defined for the function, the system checks the access permission policy for that function, as defined in the asset's access permissions interface.
3. If no access permission is set for the function (including inheritance of the parent's policy) as defined in step 2, then the system checks the authorization properties in `futuretense_xcel.ini`.
4. If none of the above types of authorization are defined, permission to perform the function on the asset is granted, by default.

Note

The above types of authorization are checked in the order shown. If two or more types of authorization are defined, the authorization that is found first overrides all other types of authorization. For example, for a specific function such as Edit, if both workflow function privileges and access permissions are defined on an asset, then function privileges on Edit override access permissions on Edit.

For assets that are not in workflow:

1. The system checks the access permissions policy for the function, as defined in the asset's access permissions interface.
2. If no access permission is set for the function (including inheritance of the parent's policy), then the system checks the authorization properties in `futuretense_xcel.ini`.
3. If none of the above levels of authorization are defined, permission to perform the function on the asset is granted, by default.

Summary

The different types of authorization described above are independent of each other, which means that only one type of authorization can apply to an asset at any given time for any given function. The following guidelines can help you decide which type of authorization to use:

1. If a function does not require authorization, do not use any of the authorization types. All roles will have permission to perform that function.
2. If you need a simple authorization policy across multiple assets, set authorization properties in `futuretense_xcel.ini`.
3. If you need a simple authorization policy on a particular asset, set access permissions.

4. If you need a more complex type of authorization that allows workflow to control the permissions on an asset, set workflow function privileges.

Setting Access Permissions

Setting access permissions means creating a policy for an asset by specifying which roles can perform which function(s) on the asset.

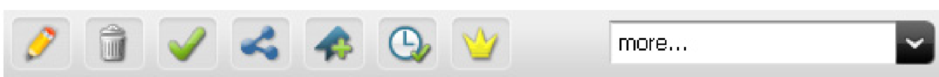
There are several ways to set access permissions on assets:

- Using the administrator's interface, to set permissions on individual assets
- Using the `futuretense_xcel.ini` file, to set permissions on all assets at once

> **Note**
>
> If your asset is entered in to a workflow for which function privileges are defined, the function privileges override the asset's access permissions. For information about function privileges, Chapter 9, "Creating and Managing Workflow Processes."

Asset Toolbar

When an asset is viewed, a toolbar appears at the top to aid in working with the asset. This is where you can perform certain function with each asset.



The icons in the toolbar perform the following actions, described from left to right:

- **Edit** - Clicking this icon opens the "Edit" screen for the asset.
- **Delete** - Clicking this icon deletes the asset.
- **Approve for Publish** - Clicking this icon approves the asset for publication the next time publication for this asset occurs. See RealTime Publishing for more information.
- **Share Content** - Clicking this icon allows you to share the content with other sites as enabled by the system administrator.
- **Add to My Bookmarks** - Clicking this icon adds a bookmark to this asset in the **Bookmarks** tab. If the asset is already added to the bookmarks tab, it will be unavailable.
- **Status** - Clicking this icon opens a screen detailing the workflow approval state of the asset.
- **Access Permissions** - Clicking this icon opens the "Access Permissions" screen for the asset. See Setting Access Permissions from the Administrator's Interface for more information.
- Drop-down menu - this menu allows you additional ways to work with the asset, including using the asset to copy and create a new asset, as well as create a wholly new asset and search for other assets.

Setting Access Permissions from the Administrator's Interface

To set access permissions from the Administrator's interface

1. Locate the asset for which you will set access permissions.
2. In the asset's "Inspect" form, select **Access Permissions** from the icon menu bar.
3. Select one of the radio boxes, click **View by role** or **View by function**. In this example, we use **View by Function**.

> **Note**
>
> In this procedure, we are using **View by function**, which allows you to select a function and then specify the roles that are granted or denied permission to perform it. If you select **View by role**, the order will be reversed: you will have to select the role first, then specify the functions it is granted or denied.

4. Select a function.

   Roles that can perform the function are now displayed in the "Inherited" list box, by default. "Inherited" refers to the permissions policy that the asset inherits from its parent, or its parent's parent, and so on, depending on which asset in the hierarchy has a policy.

5. Select a role (or roles) to which you will deny or grant the permission to perform the function. (To select a block of roles, **Shift-click** the extremes of the block. To select non-adjacent roles, **Ctrl-click** each role).

6. Select either **Grant** or **Deny** to allow or forbid the role(s) to perform the function:

   - **Grant** explicitly allows the selected role(s) to perform the selected functions on the specific asset. All other roles are denied permission to perform the selected function on that asset unless explicitly granted permission.
   - **Deny** explicitly forbids the selected role(s) to perform the selected function on the specific asset. All other roles are denied permission to perform the selected function on that asset unless explicitly granted permission
   - Selecting neither **Grant** nor **Deny** leaves the roles in the "Inherited" list box to indicate that access permissions are inherited.

7. Repeat steps 4–6 for each function and role that must be granted or denied or which you want to grant or deny permission.

When you have completed selecting functions and roles, your access permissions screen will look similar to the one below:

8. Click **Save** when finished.

Users who now access this asset will be able to perform the functions that their roles allow.

Setting Access Permissions from Property Files

To set access permissions from `futuretense_xcel.ini`

In `futuretense_xcel.ini`, either grant or deny permissions by setting the following properties:

- `xcelerate.grant.`*`functionname`* = *`rolelist`*
- `xcelerate.deny.`*`functionname`* = *`rolelist`*

  where

  `functionname` is the name of the function, as shown in the `futuretense_xcel.ini` file, on the "Authorization" tab if you are using the Property Editor.

  `rolelist` is a comma-separated list of roles for which the permission is either denied or granted.

If the permission is granted or denied in the `futuretense_xcel.ini`, it is granted or denied for all assets. These permissions can be overridden by specifically setting access permissions on an asset-by-asset basis, using WebCenter Sites' access permissions feature.

Configuring Other Options for Asset Types

In the **Admin** tab, there are several other items that configure asset types. These items contribute to the design and implementation of the asset types, which means that they are typically used by the developers. Administrators typically do not manage the items.

However, after an asset type is designed and installed, you might be asked to maintain the following items:

- sources
- categories
- subtypes
- associations
- mimetypes

For information about these items and procedures for creating and editing them, see the section called "Data Design" in the *Oracle WebCenter Sites Developer's Guide*.

Managing Access to the Tree (Administrator's interface only)

You have several ways to configure the tree that is displayed in the left-hand panel of the administrator's interface:

- You can configure the tree to be either displayed by default in the main window or hidden by default.
- You can control whether users who toggle a tree off are allowed to toggle the tree back on.
- You can configure tabs within the tree, adding items or functionality to them, and determining which users have access to them based on their roles.
- You can configure the tabs' nodes to specify the number of items that are displayed under any given node.

WebCenter Sites supports two general categories of tree tabs:

- System default tabs, which provide administrative functionality (**Admin**, **Site Admin**, **Workflow**) and access to individual assets (**History**, **Bookmarks**, **Site Plan**).

  Assets and asset types intended for developers (for example, Template) are either available only for inspection or inaccessible in the Contributor interface. Conversely, the following assets are available only through the Contributor interface: All flex assets and their parent assets, all basic assets, Engage assets (such as recommendations, segments and promotions), Query, Collection, and Page assets.
- Custom tabs
  - Tabs that you create to make it more convenient for users in different roles to create the most commonly used asset types.
  - Tabs that developers create to offer customized behavior or new functionality.

Note

The **Workflow Groups** tab is a "hybrid" between a system default and a custom tab. Although the **Workflow Groups** tab is not installed by default and you must create it to use it, the element that provides its logic is installed. That is, while you must manually create this tab, your developers do not have to code a new element for you.

This section describes how to add roles to the system default tabs, how to create basic tree tabs that provide access to asset types, and how to configure whether the tree is displayed by default.

For information about creating custom tree tabs that implement new functionality, see the section called "Management System Features" in the *Oracle WebCenter Sites Developer's Guide*.

Displaying and Hiding the Tree

Any user can toggle the tree on and off whenever he needs less or more space in the main window to display the assets that he is working on.

If you want, you can configure your Sites system so that the tree is toggled off by default when users first log in to the system. You can also control whether users are allowed to toggle the tree back on.

The WebCenter Sites features for working with assets are available in right-mouse menus on the tabs as well as through the icons at the top of the window and the action bars displayed in the **New**, **Edit**, and **Inspect** forms for individual assets managed solely in the Admin interface. Therefore, it is not necessary to display the tree for content providers. However, because the administrative functions are available only through the **Admin**, **Site Plan**, and **Workflow** tabs, administrative users must always have access to the tree.

To configure display of the tree and toggling rights

1. Start the Property Editor and open the `futuretense_xcel.ini` file.
2. Select the **Preference** tab.
3. To specify that the tree should be toggled off by default when users first log in to the administrator's interface, set the value of the `xcelerate.showSiteTree` value to `false`.
4. (Optional) To configure the system so that only administrative users (that is, users who have the xceladmin ACL assigned to their user accounts) are allowed to toggle the tree back on, set the `xcelerate.restrictSiteTree` property to `true`.
5. Save the property file and close the Property Editor.
6. Restart the application server.

Creating Tree Tabs

- Creating a New Tree Tab
- Creating the 'Workflow Groups' Tab

A tree tab is a tab (such as **Admin**) that appears in the left-hand panel of the administrator's interface. Access to tree tabs is controlled through roles.

Several default tree tabs are core to WebCenter Sites. Before creating a new tree tab, review the default tabs to gain an understanding of WebCenter Sites' existing capabilities. The default tree tabs are:

- **Bookmarks** – displays the assets that you selected and placed in your bookmarks. All users see this tab as soon as they use the **Add to My Bookmarks** button to place an asset in their bookmarks.
- **Admin** – displays the administrative functions that affect all of the sites in the system. By default, only users with the default system role named GeneralAdmin have access to this tab.
- **Design** — serves as the source for creating pages on your site. Some of these sources are: Templates, Product Definition, Content Definition, and other sources for the creating pages.

- **History** – displays the assets that you worked with during the current session. All users see this tab as soon as they create, inspect, edit, or copy their first asset. ***This tab is not configurable.***
- **Site Admin** – holds a subset of the administrative functions that apply only to the site that you are currently logged in to. By default, only users with the default system role named SiteAdmin have access to this tab. This tab is useful if you have individuals who manage which existing users have access to individual sites, but who do not need to create new users or new sites.
- **Site Plan** – displays a graphical and hierarchical representation of the pages, collections, queries, and content assets that make up the online site that you are making available on the delivery system. By default, all of the FirstSiteII sample site roles and system default roles grant access to this tab.
- **Workflow** – has the workflow configuration functions. By default, only users with the Workflow Admin role have access to this tab.

Creating a New Tree Tab

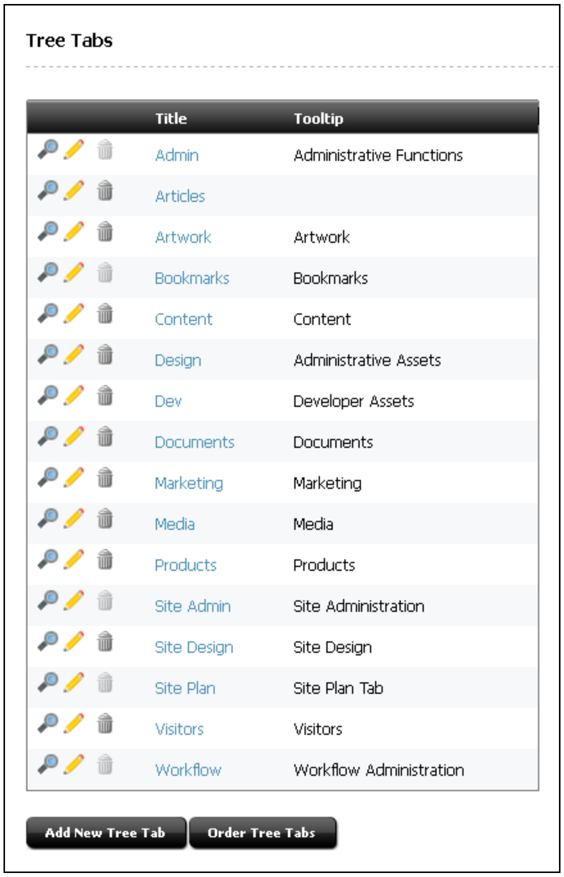
Note

Before creating a tree tab, familiarize yourself with the default tree tabs to avoid duplicating their functions. The default tree tabs are described on page 143.

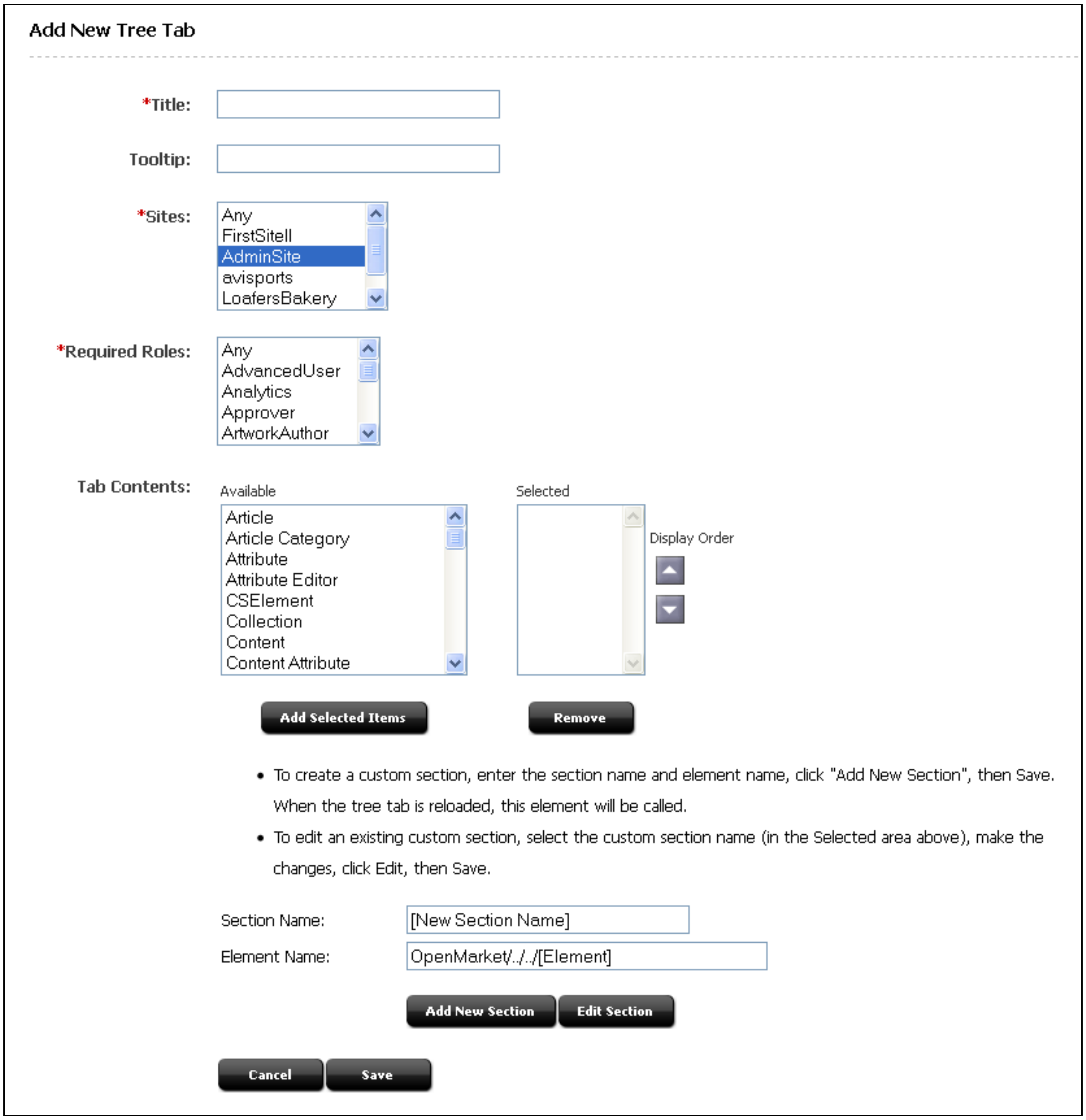
To create a new tree tab

1. In the **Admin tab,** double-click **Tree**.

2. In the "Tree Tabs" form, click **Add New Tree Tab**.



3. Fill in the "Add New Tree Tab" form:



a. In the **Title** field, enter a short, descriptive name of up to 64 characters.

b. In the **Tooltip** field, enter a short, informative description of the tab, up to 255 characters. This description is displayed when the mouse hovers over the tab.

c. In the **Site** field, select the sites that will display the tab.

d. In the **Required Roles** field, select which roles can access the tab.

e. In the **Tab Contents** field, select the asset types that will be displayed on the tab and then click **Add Selected Items**.

f. (Optional) If you want to add custom functionality to this tab, use the **Section Name** and **Element Name** fields at the bottom of the form. You will need help from your developers for this step. See also the *Oracle WebCenter Sites Developer's Guide* for more information.

4. Click **Save**.

   The new tab is displayed in the tree (refresh the display, if necessary).

Creating the 'Workflow Groups' Tab

The **Workflow Groups** tab displays the workflow groups that exist for a site and the assets that are included in those groups. For information about workflow groups, see "Workflow Groups," on page 156. This tab does not appear by default. If you want to use it, you must create it.

To create the Workflow Groups tab

1. In the **Admin** tab, double-click **Tree**.

   WebCenter Sites displays the "Tree Tabs" form.

2. In the form, click **Add New Tree Tab**.
3. In the **Title** field, enter a short, descriptive name of up to 64 characters.

   For example: **Workflow Groups**.

4. In the **Tooltip** field, enter a short, informative description of the tab, up to 255 characters.

   For example: **Workflow Groups**.

5. In the **Site** field, select the sites that will display the tab.
6. In the **Required Roles** field, select which roles can access the tab.
7. In the **Tab Contents** area, scroll down to the section area. In the **Section Name** field, enter the following name exactly as it appears here: `Groups`.
8. In the **Element Name** field, enter the following name exactly as it appears here: `OpenMarket/Gator/UIFramework/LoadWorkflowGroups`
9. Click **Add New Section**.

   The word "`Groups`" appears in the **Selected** column.

10. Click **Save**.

Editing Tree Tabs

Editing a tree tab means that you can:

- Rename the tree tab
- Edit its tooltip
- Enable the tree tab for a different set of sites
- Display the tree tab to a different set of roles
- Select a different set of content for the tree tab, and/or rearrange the content's display order

Note

- A tree tab is displayed in the user's interface when two conditions are met:
  - The tree tab is enabled on the user's site, and
  - Both the tree tab and the user have at least one role in common.
- When you create site users with new roles, you must select the new roles in the default tree tabs so that the users have access. (Default tree tabs are not automatically enabled for new roles.) The list of default tabs can be found on page 143.

To edit a tree tab

1. In the **Admin** tab, double-click **Tree**.
2. In the "Tree Tabs" form, navigate to the desired tab and click its **Edit** icon.
3. In the "Edit Tree Tab" form, make the necessary changes.

   For example:

   - If you wish to enable the tree tab for a new site, select the site from the **Sites** list.
   - If your new site has users with new roles, and you wish to enable the tree tab for those users, select the users' roles from the **Roles** list.
4. Click **Save**.

Deleting Tree Tabs

To delete a tree tab

1. In the **Admin** tab, double-click **Tree**.
2. In the "Tree Tabs" form, navigate to the desired tree tab and click its **Delete** icon.

   WebCenter Sites displays a warning message.
3. Click **Delete Tree Tab**.

   The tree tab has been deleted.

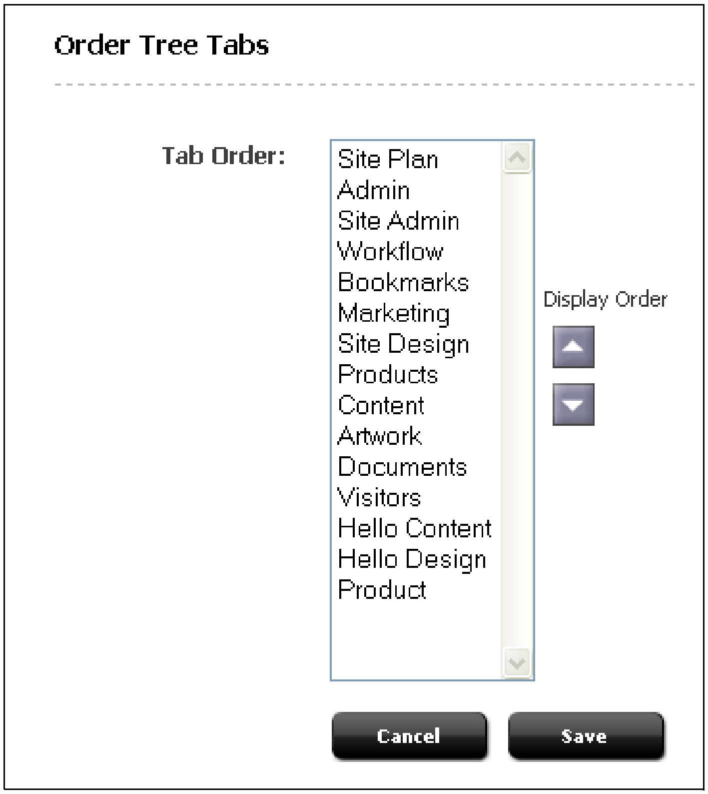
Changing Tree Tab Order

You can change the order in which the tree tabs are displayed.

To change the order of tree tabs

1. In the **Admin** tab, double-click **Tree**.
2. At the bottom of the "Tree Tabs" form, click **Order Tree Tabs**.

WebCenter Sites displays the "Order Tree Tabs" form.



The form lists the tabs in the order in which they will appear in the tree.

The tree arranges its tabs into rows. When there are more tabs than a single row can accommodate, an additional row is formed behind it. The tabs in each row are ordered from left to right. Rows are ordered front-to-back.

3. To move a tab in the tree, select the tab from the list and then use the arrow buttons to move the tab to the correct position. Repeat this step for each tab that you want to reposition.

4. Click **Save**.

   The new tab order is displayed in the tree (refresh the display, if necessary).

Configuring the Number of Items Displayed Under Nodes on Tabs

To optimize WebCenter Sites' response time, you may want to limit the number of assets that are listed under their respective node in a tree tab. Specifically, when the WebCenter Sites database holds many assets of any one type, WebCenter Sites can take a long time to load all the assets and then list them under their node in the tab. To minimize the loading time, you can limit the number of items to be listed under the node by setting a threshold. Then, when the number of assets (or other items) for a node exceeds the threshold, and a user attempts to access the assets, the administrator's interface prompts the user to enter a search criterion that restricts the number of items to be returned.

For example, suppose that a node contains more than 300 articles, and the display threshold is set to 100. When a user expands the node icon that represents the articles, WebCenter Sites displays a small search criteria form that prompts the user to enter values which WebCenter Sites uses to limit the number of articles that it returns.

To configure the number of items under a node in a tree tab

1. Start the Property Editor and open the `futuretense_xcel.ini` file.
2. Select the **Preference** tab.
3. Select the `xcelerate.treeMaxNodes` property and specify a value for it. By default, this property is set to 100, which means that up to 100 items can be displayed under a node.
4. Save and close the property file.
5. Restart the application server.

Chapter 9

Creating and Managing Workflow Processes

Workflow is a feature provided by WebCenter Sites that you use to manage the work on an asset when more than one person participates in its creation. For example, if assets of a certain type must be reviewed by an editor or a legal representative before it can be approved for publishing, the workflow feature can route those assets to the appropriate people at the appropriate time.

This chapter contains the following sections:

Overview

The end goal for any content provider is to have content assets published. Before an asset can be published, it must be approved for publishing. The workflow feature routes assets through whatever series of tasks you deem necessary in a **workflow process** that ushers assets from creation to approval.

You can configure a workflow process with as many or as few tasks as necessary to reflect the way the work at your organization is accomplished. You can configure e-mail messages that WebCenter Sites sends to notify people when assets are assigned to them and to remind them that a deadline is approaching or has been missed.

Because there are so many configuration possibilities, it is typical to create a separate workflow process for each asset type that you plan to use workflow with rather than attempt to create one process for more than one asset type.

Workflow Participants

When you begin creating a workflow process, the first general question is this: what are the job titles of the people who work on assets of this type? For example, are they authors, editors, marketers, graphic artists, product managers, lawyers?

The job titles of the people who participate in a workflow process are considered **roles** in the WebCenter Sites interfaces. Roles describe the function of an individual on a site. When you enable a user for a site, you assign that user the roles that he fulfills for that site.

When you create a workflow process, you determine which roles are appropriate for each task. Then, when an individual asset is going through that workflow process, only the users who have the appropriate role are allowed to complete the task. The individual user who is selected from the pool of users who have the correct role at any point is called a workflow **participant**.

Workflow States

Next, what are the tasks that are performed for assets of this type? For example, writing, pricing, editing, fact checking, legal review, and so on.

These tasks are called workflow **states**. A state is a point in the workflow process that represents the status of the asset at that point. For example, writing article, reviewing image, legal review, and so on. Participants complete the work that the state represents while the asset is in that state.

An asset that a participant is working on (or is supposed to be working on) is called an **assignment**. A user's **assignment list** is displayed In the "My Assignments" form in the administrator's interface. An asset appears on a user's assignment list as soon as it enters a state for which the user has a role to fulfill as a workflow participant.

Should an asset in a specific state remain in that state for only a specific amount of time? If so, assign a deadline to the state. You can then configure the workflow process to send e-mail messages that remind a participant when the deadline is approaching or has been missed. These e-mail messages are examples of **timed actions**.

You can create one or more timed action for each state.

Workflow Steps: Moving Assets from State to State

Next, how does the asset move from state to state? Does a marketing writer send a prospectus asset to a legal reviewer? Does a graphic artist send an image asset to an editor? And then what?

The movement of an asset between states is called a **step**. Because creating the steps in a process links together states in a specific order, creating the steps in your workflow process is what organizes your process.

When you create a step in a process, you specify which state a step moves the asset from (the **From State**), which state that step moves the asset to (the **To State**), and which roles can take the step.

How Steps Work

A step places an asset in a state and notifies participants from the appropriate roles. This operation creates the assignment. You then need a step for the participants from those roles to take that moves the asset to the next state. In other words, the roles notified by the previous step should be the roles authorized to take the next step. For example:

How does a user take a step in a workflow process? By specifying that he or she has finished the assignment (except for the start step, which is described below). When a user selects the **Finish My Assignment** option for an asset that is assigned to him or her, that option invokes the step, moves the asset to the next state, and assigns the asset to the next participants.

When more than one participant is assigned an asset in the same state, using the **Finish My Assignment** option is also referred to as **voting**. Each participant "votes" to move the asset to the next state.

The first step in a workflow process is a **start step**. A start step is one which has no From State. That is, the start step begins the workflow process, moving the asset to the first state in the series of states. This is the step that is invoked when the workflow process is assigned to the asset. Only users who have the roles authorized for the start step can start the workflow process for an asset.

Step Actions and Conditions

Step actions are events that occur when the step is completed. For example, when a step occurs, the asset is assigned to a participant. Should the participant be notified that an asset has been assigned to him or her? If so, you can configure a step action to send an e-mail message to the new participant. You can specify one or more step actions for each step.

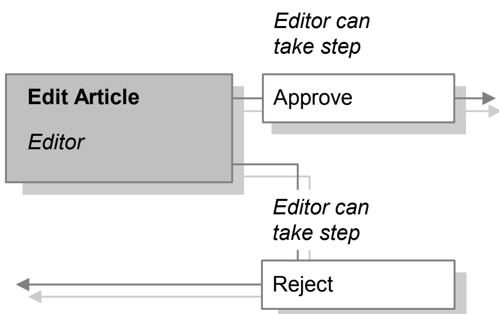
Another example of a step action is the **ApproveForPublish** action. It is a default action delivered with WebCenter Sites that approves the asset. Typically you use this action in the final step of a workflow process.

Finally, are there any requirements that must be met for an asset in a state before the step can move the asset to the next state? If so, configure and assign a **step condition** to the step. For example, you could configure a condition that verifies that the asset has all of its association fields filled—that an article has the associated images that it needs—before it progresses to the next state.

Multiple Paths for the Asset

When a workflow process includes a state in which people are reviewing an asset, typically there is more than one path possible for an asset in that workflow because the reviewer can either accept it the way it is or reject it.

In such a case, you create two steps for moving the asset from the review state. When the asset is in that state, the “Finish My Assignment” form lists both options and the participants select the appropriate step when they finish their assignments. For example:



What happens if more than one participant are reviewing the same asset in the same state and they choose different steps (that is, vote differently)? That depends on how you configured each step. There are several possibilities:

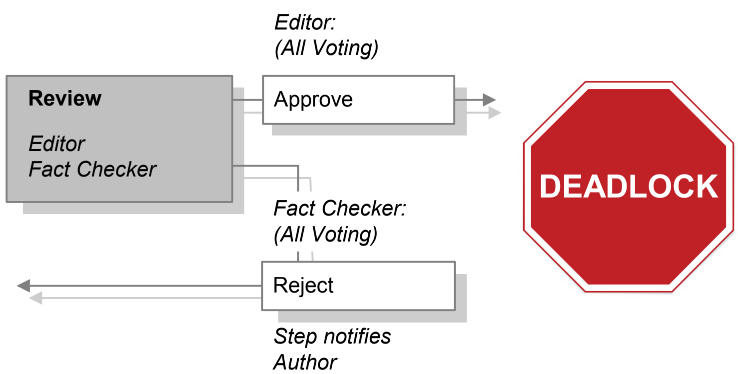
- Configure the steps so that the first participant to finish the assignment (vote) determines the direction the asset takes.
- Configure the steps so that all participants have to select the same step (vote the same way) in order for the asset to progress in either direction. A step that all the participants must select before that step can be completed is called an **all-voting step**. Note that this option can result in **deadlocks**, described in the next section.
- Use a combination of the preceding possibilities: configure one all-voting step (all participants must agree), and configure the other step(s) so that if any participant selects it, the step completes and the asset takes that path.

Managing Deadlocks

A deadlock occurs when the following conditions are true:

- There is more than one step from a state.
- Two or more of the steps require the participants to perform that step. That is, they are all-voting steps.
- An asset in that state is assigned to more than one participant.
- The participants select different all-voting steps when they finish the assignment.

This diagram illustrates a deadlock:



Note that if even one of the steps is not all-voting and a participant selects that step, the asset will not become deadlocked.

Resolving Deadlocks

An asset that is in a deadlocked state cannot progress through the workflow process until the deadlock is resolved. To resolve the deadlock, the participants must confer with each other and agree on that path that the asset should take. Then, the participants who must change their selection can do one of the following to resolve the deadlock:

- Finish the assignment and select the step that they all agreed to take.
- Select the **Abstain from Voting** option from the **Workflow Commands** drop-down list on the asset's "Status" form.

Preventing Deadlocks

Before configuring a workflow process that can result in a deadlock, be sure that it is absolutely necessary to have complete agreement on all the possible steps from the state. As you can see from this description, deadlocks cause additional work for all the participants so be sure that you use this feature only when you need to.

For example, consider a review state with two possible steps: "return for revisions" and "approve for publish." If you configure the steps so that "return for revisions" does not require a unanimous vote but "approve for publish" does, you have created a desirable control—all the reviewers must agree before the asset can be published and any rejection stops the asset from being published—without risking a deadlock.

Notifying Participants When Deadlocks Occur

If you do need to create a workflow process that can result in a deadlock, be sure to configure and assign a **deadlock action** that notifies the participants of the deadlock for each of the steps that can cause the deadlock. The default deadlock action is an e-mail message that WebCenter Sites sends to the appropriate participants when a step causes a deadlock.

Workflow Groups

Is there ever a situation in which several assets are so closely connected that they need to be thought of as one unit of work or they need to be approved at the same time? In such a case, you can use the **workflow group** feature.

Using Workflow Groups

Workflow groups enable content providers to send a defined set of assets though the workflow together. While it is the content providers who create workflow groups and select the assets that are assigned to the group, you, the administrator, still need to know what kind of assets will be included in workflow groups. Why? So that you can configure the workflow processes appropriately.

For example, you can configure workflow steps that allow each asset in the group to progress to the next state when it is finished or you can configure a step that requires all the assets in the group to reach that point before any of them can progress. (This second example, called a **synchronize step**, is described next.)

Adding a Synchronize Step

When creating a workflow process that will be used with workflow groups, it is usually best to configure the process so that it has only one synchronize step. Multiple synchronize steps can slow down the work on those assets unnecessarily. Assess the business process that is reflected by the workflow process and determine which steps must truly be synchronized: perhaps all the assets should go to legal review in one batch, or perhaps all the assets should be approved for publishing at the same time, for example.

Managing Group Deadlocks

If you create a workflow process to be used with workflow groups and any of the steps can result in a deadlock, be sure to configure and assign a **group deadlock action** that notifies participants when there is a group deadlock and assign it to the process.

Delegating and Clearing Assignments

People go on vacation, get reassigned to new work groups, and move on to different jobs. What happens to the assets that they are working on? They can **delegate** their assignments to other participants who have the appropriate roles.

Additionally, each workflow process can have a **workflow administrator**. The administrator of a workflow process can delegate assignments on behalf of the other participants.

When an asset is delegated to a new participant, should that person receive an e-mail notice? If so, configure a **delegate action** to send an e-mail message to new assignees when assets are delegated to them. You can specify one or more delegate actions for each workflow process.

If an asset no longer needs to be assigned or if it is easiest to clear the assignment and then start over, you can use the **Clear Assignments** feature in the **Admin** tab.

Placing an Asset in Workflow

An asset begins its participation in a workflow process in one of the following ways:

- A user selects a workflow process from the **Workflow Commands** field on the "Status" form for the asset.

  Selecting the workflow process invokes the start step for the process, which places the asset into the first state.

- A user creates an asset and the start menu **New** item for assets of that type is configured such that there is a default workflow process.

  In this case, saving the asset invokes the start step for the process, which automatically places the asset into the first state.

Because steps are enabled for specific roles, only the users who have a role that is assigned to the start step of a process can select that process. This means that if you are using start menu items to place assets in workflow, you must be sure that the roles assigned to the start menu item are the same roles that are assigned to the start step of the default workflow.

Restricting Access to Assets While They Are in Workflow

Although workflow routes an asset through a business process, sending it to the appropriate users at the appropriate times, the fact that the asset is assigned to a specific user does not stop other users from modifying or even deleting that asset.

If you want to restrict who has access to an asset while it is in workflow, use the workflow feature called **function privileges**. These are restrictions set on functions such as edit, copy, approve, delete, show versions, and so on in the context of workflow states and workflow roles.

There are three parts to a function privilege:

- The function being restricted.
- The roles allowed or not allowed to perform the function.
- The state during which users with those roles are allowed or not allowed to perform the function.

When a function privilege is in effect, it means that a user can perform that function only when the following conditions are true:

- The user has an appropriate role.
- The asset is in the correct state.
- The asset is **assigned** to the user.

This means that even if the user has the correct role and the asset is in the correct state, the user cannot perform that function on that asset unless the asset is assigned to that user.

Function Privileges and Step Actions

Function privileges restrict access to a function from the user interface only. This means that you can program step actions that invoke a function when a step is taken regardless of what the function privilege is set to at that moment.

The ApproveForPublish step action is an example. Even if you specify function privileges that restrict users from using the **Approve** option in the user interface, those same users can approve an asset with a workflow step if the workflow step invokes the Approve For Publish action and the user has the correct role to take the step.

In other words, you can use function privileges to prevent users from selecting and changing assets by mistake and use actions to invoke those functions in a highly controlled way.

Function Privileges: All or None

If you create even one function privilege that allows or restricts access to a function for a given role, you must create function privileges that cover all the other roles for that function.

For example, suppose that the only function that you want to restrict is the Delete function—you want to allow only the editors to delete article assets and only then if the article is in the Review state. If you create a function privilege that allows editors to delete article assets that are in the Review state but you do not create any other function privileges, nobody else can delete article assets that are in the Review state until you explicitly grant other roles access to the Delete function in the Review state.

Additionally, if you create a function privilege that **denies** editors access to the Delete function for assets in the Review state, **no user** can delete article assets that are in the Review state until you explicitly grant other roles access to the Delete function in the Review state.

Deadlines

There are two kinds of workflow deadlines:

- **Assignment deadlines** – which specify how long an asset should remain in an individual state. When you set a value for the **Estimated Time** field of a state, that creates a deadline for the assets that are assigned to workflow participants when the assets are in that state.
- **Process deadlines** – which specify how long it should take for an asset to go through the entire process.

Note that these different types of deadlines do not interact with each other—they are calculated separately and are mutually exclusive. Most likely you will use either one kind or the other, but not both for the same workflow process.

Setting and Overriding State Deadlines

When you create a workflow state, you can set an Estimated Time for it and determine whether reminder e-mails should be sent when assignments miss the deadline that is calculated from the Estimated Time.

When you create a workflow process that uses the state, you specify whether the Estimated Time for the state can be changed (overridden) when a user takes the step that moves the asset into that state.

If the deadline can be changed, any participant who takes the step can override the deadline unless you configure function privileges that restrict their ability to do so. Note that any participant who has a role that was designated as an administrator role for the workflow process can always override an assignment deadline if the deadline can be changed.

Setting Process Deadlines

When you create a workflow process, you determine whether a process deadline can be set. A process deadline is set when an asset is first placed in workflow, in the “Select Workflow” form. Unlike an assignment deadline, however, you cannot configure the process to send reminder e-mails when a process deadline is approaching.

If a process deadline can be set, any participant who places the asset in the workflow can set a process deadline for that asset—unless you configure function privileges that restrict their ability to do so. Additionally, any participant who has a role that was designated as an administrator role for the workflow process can always set a process deadline, if a deadline can be set.

Scheduling a Deadline Calculation

There are two kinds of actions:

- Actions that are invoked by a step. These actions are events that WebCenter Sites completes when a step is taken.
- Actions that are triggered by a deadline. These actions are queued and are triggered only after WebCenter Sites calculates the deadlines for the assets and determines which timed actions (if any) should be invoked.

You specify how often the deadlines are calculated by configuring the **Timed Action Event**. This is an event that invokes a background calculation process of all deadlines. It is similar to the publishing event that invokes the background approval calculation process.

Just as the publishing process calculates approvals to determine which assets should be published, the deadline calculation process that is invoked by the **Timed Action Event** calculates the deadlines for all assets that are participating in workflow processes—to determine whether any reminder messages should be sent for assignment deadlines and to determine the times that should be displayed for assets in the Due and Process Deadline columns of assignment lists.

You can configure the **Timed Action Event** on your management system to run as often as you find it necessary.

How Does a Workflow Process End?

A workflow process ends when there are no more states for the asset to progress through. This occurs when the final participant takes the **end step** for the workflow process.

An end step is the opposite of a start step—it has a From State but no To State. When a user takes the end step, it is moved to a “stateless” state, which means the asset is no longer in workflow and any function privileges set for that workflow process no longer apply.

Roles Required to Configure Workflow Processes

The workflow building blocks are located on two tree tabs in the administrator’s interface:

- **Admin** – holds e-mail objects, actions, conditions, and the **Timed Action** Event.
- **Workflow** –holds workflow states and processes.

For access to the **Admin** tab, you must be assigned the GeneralAdmin role and have the xceladmin ACL assigned to your user account. For access to the **Workflow** tab, you must be assigned the WorkflowAdmin role for the site you want to create workflow processes for.

For more information about access rights in the WebCenter Sites interfaces, see Chapter 4, "Working with ACLs and Roles" and Chapter 5, "Configuring Users, Profiles, and Attributes."

Planning Your Workflow Processes

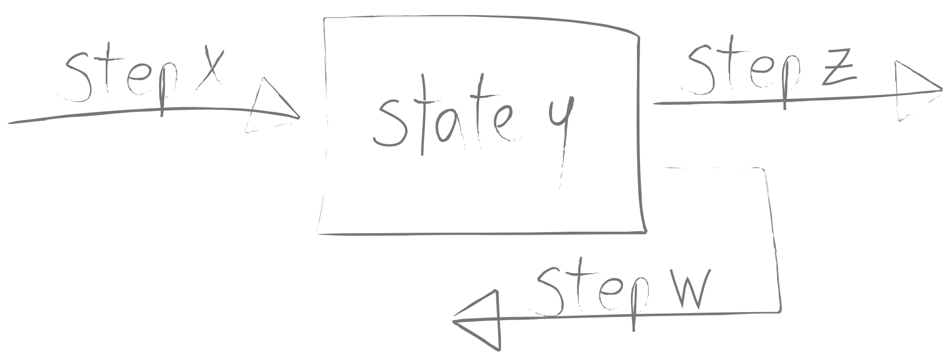
When you create a workflow process, you create steps that link together the states for that process. This means that you or someone else must create the workflow components—roles, e-mail messages, the various kinds of actions, step conditions, and states—before you can create a workflow process.

This section provides more details about the configuration of each of the workflow components so that you can plan and implement your workflow processes. When planning a workflow, you will do the following:

I. Start with a Sketch

Where do you begin? With sketches of the business processes that you need to implement as workflow processes:

- Use boxes to represent the states.
- Use arrows to represent the steps that connect the states.



As you read through the descriptions in this section, write on your sketches the details about which roles, actions, conditions, deadlines, and so on are appropriate for each state or step in the process.

Then, refer to your sketches as you use the administrator's interface to create your workflow processes, described in "Configuring Your Workflow Processes," on page 178.

II. Determine Roles and Participants

When you start planning your workflow processes, begin with the roles. What kind of functional groups participate in workflow? Then, determine how the individual users from those roles will become the participants in a workflow process for a specific asset.

Planning Roles

To determine the roles that you should create for your processes, ask yourself the following questions:

- What are the job titles or roles of the content providers who are using your Sites management system?
- What do the people in each role do? (You can map the tasks that they complete to your states.)
- How are the roles organized? For example, is there one group of reviewers who review everything but several groups of content providers who are organized by subject (for example, sports writers, financial writers, marketing writers, and so on)
- Do certain groups of people work with specific asset types but not with others? If so, which roles should have access to each asset type that you plan to create a workflow process for?
- Would you ever need to notify someone who is not participating in a workflow about the status of an asset that is in workflow? If so, that person will need a role so a step or timed action can send an e-mail message to that person.

On your sketches of your workflow processes, write the names of the roles that will have the asset assigned to them in each state.

For information about creating roles, see Chapter 4, "Working with ACLs and Roles."

Selecting Individual Participants

When you configure a workflow process, there are several ways to determine which specific users participate in the workflow for each individual asset that goes through the workflow:

- When the workflow is assigned.

  You can configure the process so that the person who first assigns the workflow to the asset must select all the users who will participate. For each state, they select users from a list. The list includes all the users who have the correct role for that state.

- When a participant finishes the assignment.

  You can configure the process so that each participant determines who the next participant is when he or she finishes the assignment. That participant selects a user name from a list that includes all the users who have the correct role for the next state.

You specify how participants will be selected when you configure the steps.

Cross-Site Assignments and Participants

Assets can be shared between sites. If a shared asset is entered into a workflow process, should users from all the sites that have access to the asset be considered as candidates for participating in the workflow? If the answer is yes, enable the cross-site assignments feature.

When you use the cross-site assignments feature, users see all of their assignments from all the sites that they have access to in their assignment list no matter which site they are currently logged into. Having one, consolidated assignment list is very convenient for your users.

Keep in mind, however, that if your users have different roles in different sites and you are using function privileges in your workflow processes, they might not be able to work on an asset that they can see in their assignment lists. For example, say that some of your users have the author role in one site and the editor role in another. If you are using function privileges to restrict editing to editors in a certain state, they can see an asset that they aren't allowed to edit in their assignment list when they are logged in to the site where they function as authors.

To enable this feature, you set the value of the `xcelerate.crosssiteassign` property to true. This property is in the `futuretense_xcel.ini` property file. For information about the property, see the *Oracle WebCenter Sites Property Files Reference*.

III. Determine the E-mail Objects, Actions, and Conditions

An **action** is an event that is triggered in one of three ways:

- A step invokes it (step action).
- A deadline triggers it (timed action).
- A workflow situation triggers it (deadlock, group deadlock, and delegate actions)

A **condition** is an event that is assessed when a step is attempted. If the condition is not met, the step cannot be completed.

When you create an action or a condition, you identify an element. **Elements** are named pieces of code that are stored in the `ElementCatalog` table. It is the element that invokes the function represented by the action. If the element is coded to expect variables or arguments, you identify values for them when you create the action and those variables are then passed to the element when the action is triggered.

As an administrator, you are not responsible for coding elements. If your workflow needs cannot be met by the default workflow elements that are provided, your developers can code the functionality that you need. See the *Oracle WebCenter Sites Developer's Guide* for information about customizing workflow.

WebCenter Sites provides the following default workflow elements:

Element Name	Variables It Expects
OpenMarket/Xcelerate/Actions/Workflow/StepActions/ApproveForPublish	`target` The name of the publishing destination that the asset is to be approved for.
OpenMarket/Xcelerate/Actions/Workflow/StepActions/SendEmailToAssignees	`emailname` The name of the e-mail object to send.
OpenMarket/Xcelerate/Actions/Workflow/StepConditions/ExampleStepCondition	
OpenMarket/Xcelerate/Actions/Workflow/AssignmentActions/SendEmail	`emailname` The name of the e-mail object to send.
OpenMarket/Xcelerate/Actions/Workflow/DeadlockActions/SendEmailToAssignees	`emailname` The name of the e-mail object to send.

Element Name	Variables It Expects
OpenMarket/Xcelerate/Actions/Workflow/GroupActions/SendEmailToAssignees	`emailname` The name of the e-mail object to send.

These elements are described in detail in the *Oracle WebCenter Sites Developer's Guide*.

With the exception of `ApproveForPublish` and `ExampleStepCondition`, these elements take a variable called `emailname` and send the e-mail message that is identified by that variable.

How do you determine the value of the `emailname` variable? By creating workflow **e-mail objects**. The names of the e-mail objects that you create in the **Admin** tab in the WebCenter Sites administrator's interface are the names that you specify as the arguments with the `emailname` variable when you create actions that send e-mail messages.

About E-Mail Objects

E-mail objects are building blocks separate from the actions so that you can use them with more than one action. For example, the default deadlock action and the default group deadlock action use the same e-mail message (named Deadlock Message).

You create e-mail objects by giving them a name, a description, a subject line, and text for the body. When the message is sent, the text provided for the subject is placed in the subject line and the text provided for the body is placed in the body of the message.

Workflow E-Mail Variables

There are several variables that you can use in the subject line and body text which makes it easier for you to write e-mail messages that are personalized for each recipient.

The following table lists the default workflow variables that you can use with any e-mail message:

Variable Name	Description
`Variables.assetname`	The name of the asset. Use this variable in every e-mail message so the recipient knows which asset is being referred to.
`Variables.assigner`	The user name of the participant who assigned the asset to the person receiving the e-mail. Use this variable in e-mail messages for step actions that notify participants that a new asset has been assigned to them.
`Variables.time`	The time specified in the **Estimated Time** field for a state. Use this variable for timed actions.

Variable Name	Description
`Variables.instruction`	The text that the previous participant entered in the **Action to Take** field of the “Finish My Assignment” form when he or she finished the assignment. Use this variable in e-mail messages for step actions that notify participants that a new asset has been assigned to them.

For examples of custom e-mail variables, see the e-mail object named Deadlock Message and the corresponding deadlock action element named `OpenMarket/Xcelerate/Actions/Workflow/DeadlockActions/SendEmailToAssignees`.

Default Workflow E-Mail Objects

The default workflow e-mail objects are these:

- Assignment Due Reminder – specifies the asset and the time that it is due.
- Assignment Message – specifies the asset, the person who assigned it, and in the message from the **Action To Take** box on the “Finish My Assignment” form.
- Deadlock Message – describes how the deadlock occurred by listing the users and the steps that they took.
- Rejection Message –is similar to the Assignment Message, but states that the asset was rejected by the previous participant (the assigner).

Planning Your E-mail Objects

To determine the e-mail objects that you need to create, you must also determine the actions that will send the e-mail messages held in the objects. By using the e-mail variables in the subject and body, it is likely that you can use the same e-mail object with more than one timed action or step action.

Typically, you need to compose e-mail messages that specify the following kinds of things:

- An asset has been assigned to a participant. (For step actions.)
- An asset has been delegated to a new participant. (For delegate actions.)
- A deadline is approaching for an asset. (For timed actions.)
- A deadline for an asset has been missed. (Also for timed actions.)
- An asset is in a deadlock. (For a deadlock action.)
- A group of assets are in a deadlock. (For a group deadlock action.)

In the **Admin** tab, double-click **Email**. Examine the default e-mail objects to determine whether you can use them. You can modify the default messages or create your own email messages.

In one corner of your sketches of the workflow processes, list the e-mail messages that you will need for that workflow.

About Timed Actions

Timed actions are actions that are based on the deadline of a state. If you specify a deadline for a state, you can specify a timed action to remind the participants about the deadline.

You create a timed action by giving it a name and a description, specifying an element, and then providing argument values for the element, if necessary.

To create a timed action that sends e-mail notices about a deadline, specify the `OpenMarket/Xcelerate/Actions/Workflow/AssignmentActions/SendEmail` element and use the **Argument** field to specify which e-mail message to send.

When you create a state, you specify which timed actions to use and when to trigger them, relative to the deadline specified for the state. You can specify more than one timed action for each state.

When the Timed Action Event runs and calculates the deadlines for all the assets currently participating in workflow, the appropriate timed actions are triggered.

Default Timed Actions

There is one default timed action: Send Email. It uses the `OpenMarket/Xcelerate/Actions/Workflow/AssignmentActions/SendEmail` element to send the Assignment Due Reminder e-mail message.

Planning Timed Actions

When planning your timed actions, you must also consider the following workflow components:

- The states that you will create, because these actions are triggered in terms of the deadlines that you set for your states.
- The e-mail objects that you need to create, because it is likely that your timed actions will send e-mail messages

Because the time that a timed action is triggered is determined outside of the action itself (you do this in the form for the state), you can probably create a small number of generic timed actions—whose only difference is the e-mail message they send—and use them repeatedly with your states.

In the **Admin** tab, expand **Workflow Actions**, then **Timed Actions**. Examine the Send Email action to determine whether you can use it. You can modify the default timed action or create your own.

On each of your sketches of your workflow processes, list the timed actions that you need for that process near the list of the e-mail messages.

About Delegate Actions

When you assign a delegate action to a workflow process, the action is triggered whenever a participant (or the workflow administrator) delegates an assignment to another participant.

You create a delegate action by giving it a name and a description, specifying an element, and then providing argument values for the element, if necessary.

You can specify one delegate action per workflow process.

Default Delegate Action

There are no default delegate actions provided.

Planning Delegate Actions

It is likely that your delegate action will send an e-mail message to the participant to whom an assignment is delegated. If this is the case, you can create a delegate action that uses any of the default workflow elements that send e-mail and create a new e-mail object for that element to send.

On each of your sketches of your workflow processes, write the name of the delegate action that you will use for that process. (You can assign one for each process.)

About Step Actions

When you assign a step action to a step, the action is triggered when a participant takes the step.

You create a step action by giving it a name and a description, specifying an element, and then providing argument values for the element, if necessary.

There are two general categories of step actions: those that complete functions and those that send e-mail messages to the participants who are being assigned the asset by the step.

Step actions are completed regardless of function privileges. That is, if you create a step action that performs a WebCenter Sites function, the action does not check the function privileges of the participant taking the step. This means that you can restrict access to a function from the user interface and require participants to use a step in a workflow in order to complete a specific function (approve for publish, for example).

You can specify one or more step actions for each step.

Default Step Actions

The default step actions are these:

- Approve for Publish, which uses the `OpenMarket/Xcelerate/Actions/Workflow/StepActions/ApproveForPublish` element to approve the asset. Then, the next time a publishing process runs, the asset is published (as long as all of its dependencies are also approved).

  To use this action for your workflow processes, you must specify the publishing destination(s) that the asset is to be approved for by using the `targets` argument.
- Send Assignment Email, which uses the `OpenMarket/Xcelerate/Actions/Workflow/StepActions/SendEmailToAssignees` element to send the Assignment Message e-mail object to all the participants assigned the asset (that is, who are specified in the **Assignment Method** for the step).
- Send Rejection Email, which uses the `OpenMarket/Xcelerate/Actions/Workflow/StepActions/SendEmailToAssignees` element to send the Rejection Message e-mail message to the new assignee (the participants specified in the **Assignment Method** for the step).

Planning Step Actions

To determine what kind of step actions you need to create, you must consider the following workflow components:

- The steps that you will create for the process. Do people need to be notified that a step has assigned an asset to them? If so, you need a step action that sends an e-mail message.
- The states that the steps move the assets into. Does the state represent the asset after a function has been completed? If so, you need a step action that implements that function.

In the **Admin** tab, select **Workflow Actions**, then **Step Actions**. Examine these actions to determine whether you can use them. You can both modify the default step actions and create your own.

On each of your sketches of your workflow processes, write the name of the step actions next to the appropriate steps. You can assign one or more step actions for each step.

About Step Conditions

When you assign a step condition to a step, the condition is assessed when the step is taken. The condition determines whether the step can be completed or not.

You create a step condition by giving it a name and a description, specifying an element, and then providing argument values for the element, if necessary. If you want to use conditions in your workflow processes, talk to your developers about the conditions you need them to implement through elements.

You can specify one or more conditions for each step.

Default Step Conditions

There is one default step condition: Example Step Condition. It is a “hello world”-style example that illustrates how to code an element to check for a condition. You and your developers should examine it to learn how it works and then your developers should create their own condition elements.

Planning Step Conditions

When thinking about step conditions, you must consider the steps and the state. If you determine that conditions are necessary for your workflow processes, ask your developers to create them.

On each of your sketches of your workflow processes, write the name of any step conditions that you need next to the appropriate steps.

About Deadlock Actions

When you assign a deadlock action to a step that can result in a deadlock, the action is triggered whenever an asset becomes deadlocked during the step.

You create a deadlock action by giving it a name and a description, specifying an element, and then providing argument values for the element, if necessary.

You can assign one or more deadlock actions per step.

Default Deadlock Action

There is one default deadlock action: Send Deadlock Email. It uses the `OpenMarket/Xcelerate/Actions/Workflow/DeadlockActions/SendEmailToAssignees` element to send the Deadlock Message e-mail message.

Planning Deadlock Actions

When thinking about possible deadlock actions, you need to consider the steps that you will create for your workflow processes. If you plan to design your steps so that deadlocks cannot occur, there is no need to create any deadlock actions.

In the **Admin tab**, expand **Workflow Actions**, then **Deadlock Actions**. Examine this action to determine whether you can use it. You can modify the default deadlock action or create your own.

In your sketches of your workflow processes, write the name of the appropriate deadlock action next to any step that can result in a deadlock.

About Group Deadlock Actions

A group deadlock action is triggered when workflow group becomes deadlocked. Whoever creates the workflow group selects the group deadlock action. That person can select one or more for each group.

The group deadlock actions are invoked in addition to any other actions that are associated with the states or the steps in the workflow process.

Default Group Deadlock Action

There is one default group deadlock action: Send Deadlock Email. It is identical to the default deadlock action.

Planning Group Deadlock Actions

The same considerations apply for group deadlock actions as for deadlock actions. However, because it is a content provider who selects the group deadlock action for a workflow group, be sure that you give a group deadlock action a meaningful, unambiguous name.

IV. Determine the States

When you create a state, you specify the following kinds of information:

- Name and description. The name of a state should be meaningful and should represent the kind of work that is being done while the asset is in that state.
- Amount of time an asset should remain in this state (the **Estimated Time**). The deadline is calculated in terms of the number of hours or days since the asset entered the state.

  Note that you set the estimated time (the deadline) in terms of hours or days rather than as a specific date because a state deadline is calculated for each asset as it passes through the state.
- The timed actions that should occur and when they should occur, in terms of days or hours before or after the deadline.

There are no default states, although there are example states provided with all of the sample sites. On a system where the sample sites are installed, select **Workflow > States** and then select a state to examine to learn about how the sample site states are configured.

Planning Your States

To determine the states that you need to create, ask yourself the following kinds of questions:

- Which roles participate in each state? On your sketches, write the names of the roles next to the states.
- Do any of these states apply to more than one asset type? If so, it's possible that you can reuse the same state in more than one workflow process.
- How long should it take to complete each state?
- Does this amount of time differ by asset type or by site? If yes, you cannot reuse the same state in more than one workflow process or share the workflow process with another site.
- If there is a deadline, should WebCenter Sites notify the participant when the deadline is approaching? If yes, when? The day before? Several hours before? And which timed action (which determines which e-mail message) should be used?
- Should there be a notice when a deadline is missed? If yes, when? And which timed action should be used?

On the sketches of your workflow processes, list all of this information next to each state.

Depending on your goals, you might consider creating a workflow process that does not really end. For example, the Hello Asset World sample site workflow process ends after a HelloArticle asset has been approved. However, in a case like this, someone could open and save the asset by mistake, which would mean that it is no longer approved and will not get published.

To keep assets from being edited after they have been approved and before they have been published, you could create a final state that holds assets after they have been approved with a function privilege on the state that restricts anyone from editing the asset.

Be sure that if you do create a state like this that you create a step that can move the asset back into an editing state in the workflow so someone can edit it on purpose.

V. Determine the Steps

You create steps within a process. You use the steps to link together the states. This is how the process is actually created.

You specify the following kinds of information for each step in your process:

- The name of the step. The name should be meaningful and should describe the path being taken. For example, Send for Review, Send for Approval, and so on.
- The **From State**, which is the state that the step is moving the asset from. If the step has no **From State**, it is the start step that begins the workflow process. A **From State** is required for each step other than the start step.
- The **To State**, which is the state that the step is moving the asset to. If the step has no **To State**, it is the end step the ends the workflow process.
- Which roles are authorized for the step.

  If this is the very first step, the roles you specify determine which users can select this workflow for an asset. If you are using a Start Menu item to assign the workflow process to an asset, be sure that the roles assigned to this first (start) step match the roles assigned to the Start Menu item.

  For subsequent steps after the first (start) step, the roles that you select must either match or be a subset of the roles that were selected to be notified by the previous step.

Otherwise, the asset cannot leave the state because no one who is assigned the asset is allowed to move it to the next state.

- Who gets the asset next? (That is, when this step is completed). You specify the assignee for the To State by selecting one of the following assignment methods:
  - Retain "From State" assignees.

    This option assigns the asset to its current assignees—that is, to the user who completes the step.

    This option is useful when you are creating a step that returns to the same state (which creates an iterative state) or when it is the start step and you want the asset to be assigned to the person who selects the workflow process (whether through a start menu item or by selecting it on the "Status" form).
  - No assignments; control actions with function privs.

    This option keeps the asset in the workflow process, which means that function privileges are enforced, but the asset is not actually assigned to anyone.

    Note

    When an asset is assigned to the current assignees (Retain "From State" assignees option) or assigned to no one ("No assignments" option), the workflow system records workflow history for that step.
  - Assign from a list of participants.

    With this option, you select roles. When a user assigns the workflow process to an asset, a list of users with the roles that you selected is displayed. The user selects one or more users from the list, and those users are assigned the asset when it is in the state that this step moves the asset to.
  - Choose assignees when the step is taken.

    You also select roles with this option. This option means that the person who takes this step (by completing the assignment) during the workflow process selects the user who is assigned the asset next from a list of users with the roles selected for the step.
- Whether the estimated time for the state that this step moves the asset to can be changed or not. (Called an Assignment Deadline.)
- Any step actions that should occur.
- Any step conditions that should be assessed.
- If the step can be taken by more than one user (assignee), whether all the assignees must take this step before the asset can move to the next state. (The **All assignees must vote** field.)
- If the step can result in a deadlock, what the deadlock action should be.
- If the process will be used for a workflow group, whether all the assets in the group must complete this step before any of the assets can move to the next state. (The **Step is group synchronized** field.)

There are example steps in all of the sample site workflow processes. On a system where the sample sites are installed, select **Workflow > Processes** and examine a process to learn about how the sample workflow steps are configured.

Planning Your Steps

When determining what your steps should be, consider the following questions:

- Every process needs a start step. Before you create a start step, you must first determine how the asset is created and then placed into workflow.

  Does the first participant create the asset, place it into workflow, and then keep working on the asset? If so, configure the step so that it is assigned to the person taking the start step (choose the **Retain "From State" assignees** option) and remember to create a start menu item that assigns the workflow to the asset by default.

  Or does a supervisor create the asset and then assign to it a participant? If so, configure the start step so that the person taking the start step has to select the assignees.
- What is the order of the states?
- How many paths do you need between each state? This answer determines how many steps you need.
- Do you need to create an iterative state? If so, configure the step that takes the asset back to that state so that it is assigned to the person taking the step (choose the **Retain "From State" assignees** option).
- For steps that move an asset from a state in which more than one person is working on that asset, must all the participants complete their assignment before the step can be taken? If so, configure the step to be an all-voting step.
- Does the asset have more than one possible path from a state? (That is, there needs to be more than one step with the same From State.) If yes, do any of the steps from the state require that all participants vote the same way before the asset can progress? If yes, try to design the process so only one of the steps that lead from that state require that all the participants vote the same way. If you have two all-voting steps from the same state, deadlocks can occur.
- For each step, should WebCenter Sites notify the affected participants when the step is completed and the asset progresses to the next state? If yes, which step action (which determines the e-mail message) should the step use?
- Do groups of related assets ever need to be worked on at the same time? If so, can you configure your steps to work appropriately for both a single asset and a group of assets? If not, create separate workflow processes for the workflow groups.
- For a workflow group, are there any states that all the assets in the group must enter at the same time? (For example, all the assets should be approved at the same time.) If so, configure that step to be a synchronize step. Note that it's best to have only one synchronize step per workflow process.
- What should happen to the asset if you decide not to publish it, after all? Should you have a cancel step? How many cancel steps do you need?

On your sketches of your workflow processes, right next to the box that represents a step, list the step actions, the roles who can take the step, the roles who should be notified when the step is taken, whether the step is an all-voting step, and whether the step is a synchronize step.

VI. Determine the Function Privileges

Administrators grant or deny roles the privilege to use functions (such as create and edit) on a given site. When the privilege is granted, the functions are displayed to site users with those roles. Users can invoke the functions when an asset is in the state specified by the privilege. (By default, function privileges are granted to all roles on a given site.)

You can set function privileges for each function in a workflow process, as necessary. Instructions for setting function privileges can be found on page 194 ("Step E: (Optional) Configure Function Privileges").

The following table lists the functions that you can allow or disallow users to perform during a workflow process.

Function	If the Function Privilege is Granted...
Abstain from Voting	Users have the option to not vote in a workflow step. Displayed as an option in the **Workflow Commands** drop-down list on the asset's "Status" form.
Approve for Publish	Users can mark an asset as approved for publishing. Displayed as an option in the **more...** drop-down menu on the asset's "Inspect" and "Status" forms.
Authorize	Users can configure permissions to an asset.
Build	Users can build a collection asset. Displayed as an option in the **more...** drop-down menu on an asset's "Inspect" and "Status" forms.
Checkout	Users can check out an asset, as long as revision tracking is enabled for the asset type. Displayed as the **Check Out** button at the top of an asset's "Edit" and "Status" forms.
Copy	Users can copy an asset. Displayed as an option in the **more...** drop-down menu on an asset's "Inspect" and "Status" forms.
Delegate Assignment	Users can delegate an asset (assigned to themselves) to another user (one who has the appropriate role for the asset while it is in that state). Displayed as an option in the **Workflow Commands** drop-down list on the asset's "Status" form.
Delete	Users can delete an asset. Displayed as an icon in the action bar on an asset's "Inspect" and "Status" forms. Also displayed as an icon next to the asset in lists.
Edit	Users can open an asset in its "Edit" form. Displayed as an icon in the action bar on an asset's "Inspect" and "Status" forms. Also displayed as an icon next to the asset in lists.

Function	If the Function Privilege is Granted...
Inspect	Users can open an asset in its “Inspect” form. **Note:** Users without permission to inspect an asset cannot view any data specific to the asset type. Instead, they see a limited version of the “Inspect” form, which displays only standard, unrestricted fields (such as name, description, and ID).
Make Root of Translation	Users can set an asset as the master asset of translations. (Used by multilingual assets.) Displayed as the **Make Master** link in the “Translations” field of a multi-lingual asset’s “Inspect form.
Place Page	Users can place a page on the **Site Plan** tab. Displayed as an option in the right-mouse menu on the **Site Plan** tab when the user right-clicks **Placed Pages**.
Preview	Users can view an asset in the context of its page. Displayed as the **Preview** icon in the asset’s toolbar.
Remove from Group	Users can remove the asset from a workflow group. Displayed as an option in the **Workflow Commands** drop-down list on the asset’s “Status” form.
Remove from Workflow	Users can remove the asset from a workflow process. Displayed as an option in the **Workflow Commands** drop-down list on the asset’s “Status” form.
Rollback	Users can return an asset to one of its previous versions (all of which are stored in the revision tracking system). **Note:** This function is displayed as the **Rollback** button at the top of an asset’s “Edit” and “Status” forms when revision tracking is enabled and at least two versions of the asset exist.
Set Export Destination Path/ Filename	Users can fill in the **Path** and **Filename** fields on an asset’s “New” or “Edit” forms.
Set Participants	Users can set participants for a workflow process. Displayed as an option in the **Workflow Commands** drop-down list on the asset’s “Status” form. Additionally, this function prompts the user to select participants when the workflow process is configured. In this manner, the user finishing an assignment must select the next participant (person being assigned the asset).

Function	If the Function Privilege is Granted...
Set Process Deadline	Users can set a deadline indicating the date and time by which the workflow process must be completed. **Note:** This function is displayed only if the workflow is configured to allow its users to set a process deadline. This function then appears as an option in the “Select Workflow” form for assets, and when a workflow group is created or edited.
Set Assignment Deadline	Users can set a deadline indicating the date and time by which the next state must be completed. A deadline set with this option overrides any deadlines set for the state in the workflow process. **Note:** This function is displayed only if the workflow is configured to allow its users to set an assignment deadline. This function then appears as an option in the asset’s “Select Workflow” form and in the “Finish My Assignment” form.
Share Assets	Users can share the asset with another CM site. Displayed as the **Share *Asset*** option in the **more...** drop-down menu on an asset’s “Edit,” “Inspect,” and “Status” forms when the asset type is enabled for more than one CM site.
Show Participants	Users can view a list of the participants in the workflow process that is currently assigned to the asset. Displayed as an option in the **Workflow Commands** drop-down list on the asset’s “Status” form.
Show Status	Users can view the “Status” form for an asset. Displayed as an option in the **more...** drop-down menu on an asset’s “Inspect” and “Status” forms.
Show Version	Users can view information about each version of the asset that the revision tracking system is storing. Displayed as the **Show Versions** button at the top of an asset’s “Edit” and “Status” forms when revision tracking is enabled for this asset type.
Set Nested Workflow	Users can create a workflow within an existing workflow. **Note:** The administrator’s interface does not support nested workflow, but functionality can be obtained with the use of tags.
Translate	Users can create a translation of the master asset. (Used by multilingual assets.) Displayed as an option in a multi-lingual asset’s **more...** drop-down menu, on the “Edit,” “Inspect,” and “Status” forms.

Planning Function Privileges

For each workflow process, determine which functions you want to restrict access to, during which states you want to restrict those functions, and which roles should or should not have access to those functions when assets that are assigned to them are in those states.

List any restrictions that you want to enforce next to the appropriate states on your sketches of your workflow processes.

Remember that in order for your function privileges to enforce the restrictions that you intend, the roles of the users specified for the state in the function privilege must match the roles of the users associated with those states in the workflow process itself. If the roles don't match, the result can be a condition in which no one can move the asset out of a state because the users who are allowed to work with the asset by the privilege are not allowed to by the state.

Also, remember that if you create even one function privilege that allows or restricts access to a function for a given role, you must create function privileges that cover all the other roles for that function.

Implementing Simplified Access Control

Because function privileges are associated with workflow processes, you can restrict individual users' access to specific functions only when an asset is participating in a workflow process.

What can you do if you do not want to design and implement workflow processes but you still want to control access to specific functions? Create a simplified workflow process with one state and use the **No assignments; control actions with function privs** assignment method.

When a step moves an asset into a state using the **No assignments** assignment method, the asset does not appear on anyone's assignment list, it just stays in the state which means that any function privileges assigned to that state are enforced.

To implement access control in this way, follow these general steps:

1. Create a state. It needs a name and a description. It does not need a deadline or any timed actions.
2. Create a new workflow process. Select all the roles you want to enforce restrictions for and all the asset types you want to use this workflow process for.
3. Create one step— the start step that puts assets of those types into the state. For the step, select the **No assignments** option. Enable the step for the roles that you want to be able to create assets of this type and assign this workflow to.
4. Configure the function privileges that you want to enforce for assets in the state that you created.
5. Create start menu items for the asset types that automatically assign this workflow process. Be sure that the roles who can use the start menu item match the roles who are assigned to the start step in the workflow process.

Now when users select **New**, then select ***asset type***, the new asset is automatically placed into your single-state workflow. Because the workflow has only the start step which places all assets of that type into the single state, those assets do not leave the state and the function privileges are always enforced.

VII. Determine Additional Workflow Process Details

If you have been sketching your workflow processes, filling in all the details about actions, conditions, e-mail messages, states, steps, and function privileges, by now you have planned nearly the entire design of your workflow processes.

Although the main task when creating a process is to create the steps that link the states (which is how your business process is represented), you must also specify the following kinds of information for each workflow process:

- Its name and description. The name should be descriptive so that users select the correct workflow process for the correct asset types.
- Which sites can use the process.
- Which asset types can use the process.
- Which roles can participate. This is a superset of all the roles that are designated in the steps.
- Which role will serve as the administrator of the workflow. A workflow administrator can delegate assignments on behalf of other participants.
- Whether a process deadline can be set for assets that participate in this workflow process.
- What the delegate action is.
- What the steps are. For each step, you specify the information described in the section named "V. Determine the Steps," on page 170.
- Whether any of the Sites content application functions should be restricted to users in certain roles while they are working on assets in specific states. For function privileges, you specify the information described in "VI. Determine the Function Privileges," on page 173.

There are example workflow processes delivered with all of the sample sites. They are described in detail in the next section.

Configuring Your Workflow Processes

This section presents the procedures for creating all the components for a workflow process and then stitching those components together into your workflow processes.

Remember that to create e-mail objects, actions, and conditions or to schedule the timed action event, you need access to the **Admin** tab, which means that you need the GeneralAdmin role and the xceladmin ACL. To create workflow states and workflow processes, you need access to the **Workflow** tab, which means that you need the WorkflowAdmin role.

Overview

Before you create a workflow process, you must create the individual workflow components that you need for that process. Here are the general steps that you must take presented in the order you perform them:

1. Plan your workflow process by drawing it. See section "Planning Your Workflow Processes," on page 161 for help with this step. Then refer to your sketch and notes throughout this section.
2. Create the roles that you need for your workflow processes. See Chapter 4, "Working with ACLs and Roles," for help with this step. Be sure that any users who will participate in workflow have user profiles created for them. Otherwise, they will not receive e-mail messages from the workflow process.

   Note

   If you want the pool of users who are candidates to be participants in a workflow to include folks from all the sites that an asset is shared to, be sure to enable the cross-site assignments feature. See "Cross-Site Assignments and Participants," on page 162 for details.

3. Create the e-mail objects that you need for your actions and enable the `xcelerate.emailnotification` property in the `futuretense_xcel.ini` file. See "Setting Up E-Mail Objects," on page 179.
4. Create the step actions, timed actions, deadlock actions, group deadlock actions, and delegate actions that you need. See "Setting Up the Workflow Actions and Conditions," on page 181.
5. Create your states. See "Setting Up the States," on page 185.
6. Create your process. While creating your workflow process, you create the steps for that process. The steps link together the states so they occur in the proper order. Additionally, while creating your process, you configure any function privileges that you need. "Setting Up the Workflow Processes," on page 187.
7. If your states have deadlines, be sure to configure the Timed Action Event so that the deadlines of assets are calculated regularly and the appropriate timed actions (if any) are invoked in a timely way. "Setting Up the Timed Action Event," on page 184.
8. Test your workflow processes. See "Testing Your Workflow Process," on page 197.
9. Set up your start menu shortcuts so that workflow processes are assigned automatically to assets when they are created. For help with start menu items, see "Creating Start Menu Items," on page 124.

Setting Up E-Mail Objects

You create e-mail objects so that your step and timed actions can send them to the appropriate participants at the appropriate times. Examine the sketches of your workflow processes, determine the e-mail messages you need, and then use the procedures in this section to create and edit them.

Your user account must give you access to the **Admin** tab in order for you to create e-mail objects.

Enabling the E-mail Feature

To ensure that your workflow process can successfully send e-mail messages, the following conditions must be true:

- In the `futuretense.ini` file, the properties on the **Email** tab must be configured to provide information about your e-mail server.
- In the `futuretense_xcel.ini` file, the `xcelerate.emailnotification` property must be set to `true`.

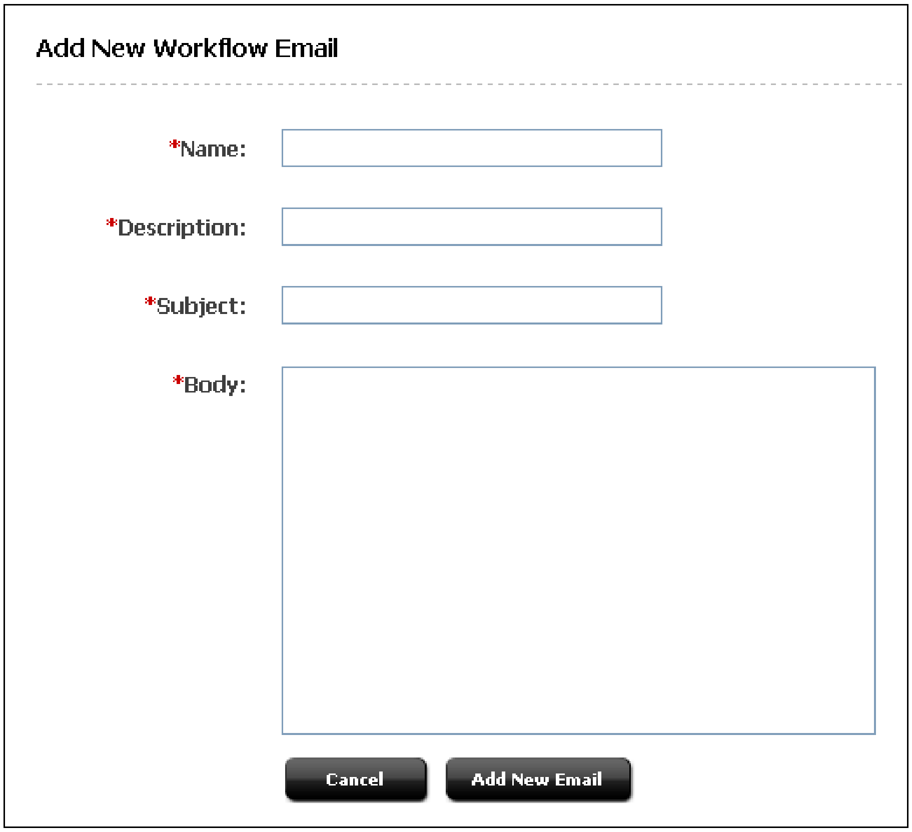
  For information about the properties cited above and using the Property Editor, see the *Property Files Reference*.
- The workflow participants must have e-mail addresses specified in their user profiles. (For information about creating user profiles, see "Working with User Profiles and User Attributes," on page 93.)

Creating E-Mail Objects

To create e-mail objects

1. In the **Admin** tab, expand **Email** and double-click **Add New**.

WebCenter Sites displays the “Add New Workflow Email” form:



2. In the **Name** field, enter a unique name of up to 36 characters. This is the name that you provide to the `emailname` variable when you use this e-mail object with an action.
3. In the **Description** field, enter a short, informative description of up to 36 characters.
4. In the **Subject** field, enter a short, informative subject for the e-mail message. (See “About E-Mail Objects,” on page 164 for a list of variables that you can use.)
5. In the **Body** field, enter the text for the message. (See “About E-Mail Objects,” on page 164 for a list of variables that you can use.)
6. Click **Save**.

Editing E-Mail Objects

To edit e-mail objects

1. In the **Admin** tab, expand **Email**, then double-click the desired e-mail object.
2. In the action bar, click **Edit**.
3. In the “Edit Workflow Email” form, make the necessary changes. (See “About E-Mail Objects,” on page 164 for a list of variables that you can use.)
4. Click **Save**.

Deleting E-Mail Objects

To delete e-mail objects

1. In the **Admin** tab, expand **Email**, and double-click the desired e-mail object.
2. In the action bar, click **Delete**.

   WebCenter Sites displays a warning message.
3. Click **Delete Email**.

Setting Up the Workflow Actions and Conditions

When you create any kind of action or a step condition, you identify an element and you supply values for the variables that the element expects. If you are creating an action that sends an e-mail message, you identify which e-mail object to send with the `emailname` variable.

Before you begin creating actions or step conditions, be sure that you have the elements and e-mail objects that you need. WebCenter Sites provides several default action elements and e-mail objects. Use the administrator's interface to examine the e-mail objects and use WebCenter Sites Explorer to examine the `ElementCatalog` table. Determine which elements you plan to use and then write down the entire name of those elements. If you need additional e-mail messages, consult the previous section, "Setting Up E-Mail Objects," on page 179 and create the e-mail messages that you need.

The following procedures describe how to create, edit, and delete the workflow actions—step, timed, delegate, deadlock, and group deadlock action—and step conditions. Your user account must give you access to the **Admin** tab in order for you to create actions or conditions.

Creating Workflow Actions and Conditions

To create workflow actions and conditions

1. In the **Admin** tab, expand **Workflow Actions**.
2. Under **Workflow Actions**, expand the category describing the action you are creating, then double-click **Add New**.

   The available categories are: **Step Actions**, **Step Conditions**, **Timed Actions**, **Delegate Actions**, **Deadlock Actions**, or **Group Deadlock Actions**.

WebCenter Sites displays the “Add New” form corresponding to the type of action you selected. The example below shows the “Add New Step Action” form:

3. In the **Name** field, enter a unique name of up to 40 characters.
4. In the **Description** field, enter a short, informative description of up to 40 characters.
5. In the **Element Name** field, enter the name of the element in its entirety.

   For example, to use the default workflow element `SendEmailToAssignees`, enter:

   ```
   OpenMarket/Xcelerate/Actions/Workflow/StepActions/
      SendEmailToAssignees
   ```
6. In the **Arguments** field, use the following convention to supply values for the arguments or variables that the element needs to function correctly:

   `name=value`

   For the `SendEmailToAssignees` element, for example, you must provide a value for the `emailname` variable (that is, the name of an e-mail object). For example:

   ```
   emailname=AssignmentDueReminder
   ```

   If an element takes more than one variable, separate the name/value pairs with the ampersand (&) character. For example:

   ```
   name1=value1&name2=value2
   ```
7. Click **Add New Action**.

Configuring Approve for Publish Step Actions

The approval process approves an asset to a specific publishing destination. To use the default Approve for Publish step action with your workflow processes, you specify the publishing destination for the assets that are approved by the action.

If all the assets for all of your workflow processes are published to the same destination, you can simply configure the existing Approve For Publish action. However, if some types of assets are published to a different destination than the others, you must create an additional Approve for Publish action for each publishing destination, or combination of publishing destinations.

The Approve for Publish step action uses the `OpenMarket/Xcelerate/Actions/Workflow/StepActions/ApproveForPublish` element which takes the `targets` variable. You can use this same element with as many additional approval step actions as you need.

For the default Approve for Publish step action, provide a value for the `targets` variable. Possible values for the `targets` variable are the names of any of the publishing destinations that have been created on this Sites system.

For example:

`targets=serverX.`

To specify more than one publishing destination, separate each with a comma. For example:

`targets=serverX,serverY`

> **Note**
>
> If the name of a publishing destination that is referenced by the `targets` variable is changed, you must also change the value of the `targets` variable in your Approve for Publish steps.

Editing Workflow Actions and Conditions

To edit workflow actions and conditions

1. In the **Admin** tab, expand **Workflow Actions**.
2. Under **Workflow Actions**, expand the category describing the action you want to edit.

   The available categories are: **Step Actions**, **Step Conditions**, **Timed Actions**, **Delegate Actions**, **Deadlock Actions**, or **Group Deadlock Actions**.
3. Under the selected category, double-click the action you want to edit.
4. In the action bar, click **Edit**.
5. Make your changes, then click **Save**.

Deleting Workflow Actions and Conditions

To delete workflow actions and conditions

1. In the **Admin** tab, expand **Workflow Actions**.
2. Under **Workflow Actions**, expand the category describing the action you want to delete.

   The available categories are: **Step Actions**, **Step Conditions**, **Timed Actions**, **Delegate Actions**, **Deadlock Actions**, or **Group Deadlock Actions**.
3. Under the selected category, double-click the action you want to delete.
4. In the action bar, click **Delete**.

   WebCenter Sites displays a warning message.
5. Click **Delete Action**.

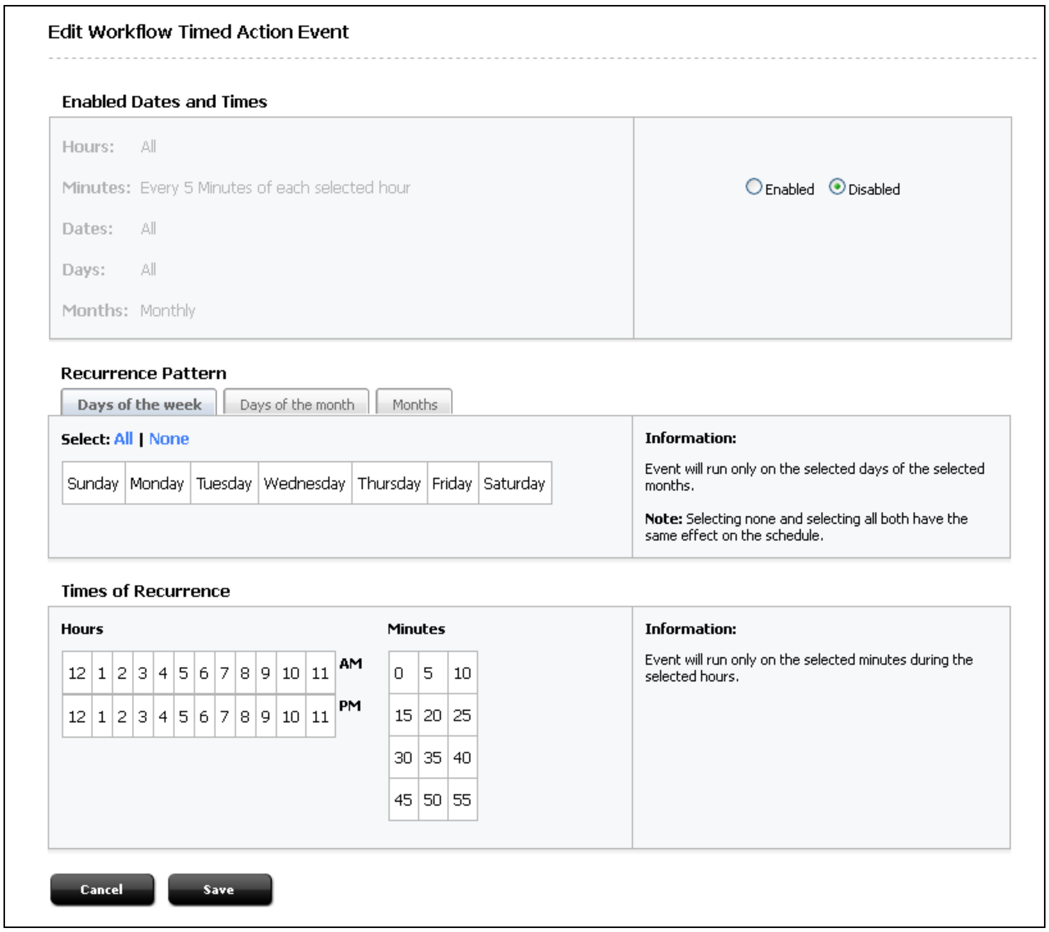
Setting Up the Timed Action Event

Before any of your timed actions can be triggered, you must configure the timed action event so that it calculates deadlines at regularly occurring intervals. Note that there can be only one timed action event per Sites system.

To set up timed action events

1. In the **Admin** tab, double-click **Timed Action Event**.
2. In the action bar, click **Edit**.

   WebCenter Sites displays the “Edit Workflow Timed Action Event” form.



3. Set the schedule for how often the state deadlines should be calculated in terms of months, days, hours and minutes, depending on the selected interval.

   Selected values will be highlighted in blue. All values are toggled separately, so it is not necessary to use **Ctrl** to select multiple values.

   The method of setting a timed action event is similar to the method for setting a publication event. See “Example Schedule,” on page 320 for more information.
4. In the **Enabled Times and Dates** field, select **Enabled**.
5. Click **Save**.

   A summary of the schedule is displayed.

WebCenter Sites uses the same abbreviations and codes to summarize the schedule for the timed action event that it does for your publishing events. For information about how to read the schedule, see "Reading the Schedule Abbreviations," on page 322.

Setting Up the States

When you create a workflow state, you specify a deadline, select a timed action, and configure when the timed action should run. Therefore, before you begin creating your workflow states, be sure that you have created the timed actions that you need.

To work with workflow states, your user name must be assigned the WorkflowAdmin role for the site that you are working with.

Creating Workflow States

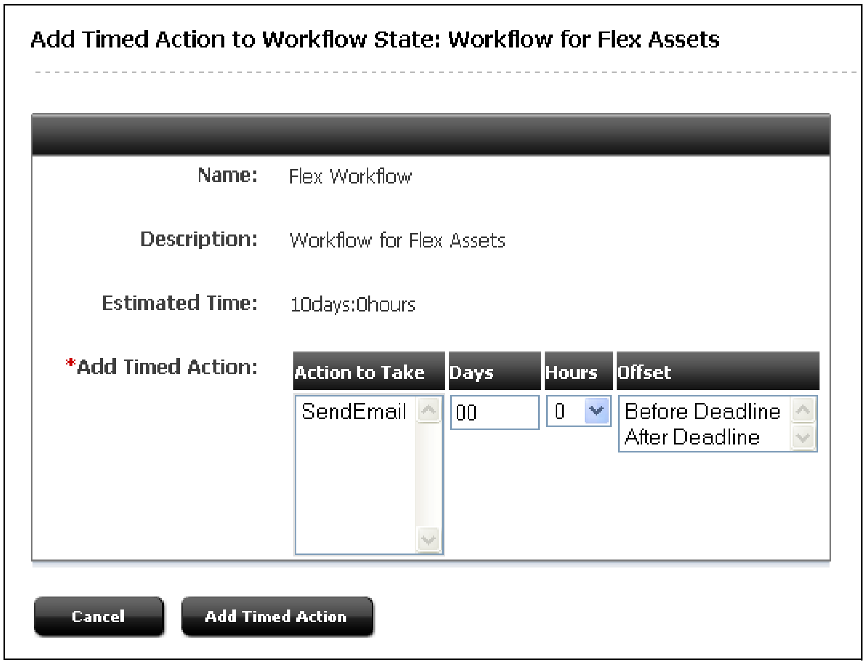
To create workflow states

1. In the **Workflow** tab, expand **States** and double-click **Add New**.

   WebCenter Sites displays the "Add New Workflow State" form:

2. In the **Name** field, enter a unique, meaningful name of up to 40 characters.
3. In the **Description** field, enter a short, informative description of up to 40 characters.
4. (Optional) In the **Estimated Time** fields, configure the deadline for assets in this state. You can specify a deadline in terms of days, hours, or a combination of the two.

5. (Optional) Click **Add Timed Actions** and complete the following steps:



   a. In the **Action to Take** list, select a timed action.
   b. In the **Offset** field, specify whether the action should be triggered before the deadline or after the deadline.
   c. In the **Days** field and/or the **Hours** field, specify how many hours or days before or after the deadline (that you specified in step 5) that the action is to be triggered.
   d. Click **Add Timed Action**.
   e. Repeat this entire step for each timed action that you want to set up for this state.

6. Click **Add New State**.

Editing Workflow States

To edit workflow states

1. In the **Workflow** tab, expand **States** and double-click the state you want to edit.
2. In the action bar, click **Edit**.
3. In the “Edit” form, make the desired changes.
4. (Optional) To change the timed action or the time that it is scheduled to run, click **Add Timed Actions**, make the appropriate changes, and click **Add Timed Actions** again.
5. Click **Save**.

Deleting Workflow States

To delete workflow states

1. In the **Workflow** tab, expand **States** and double-click the state you want to delete.
2. In the action bar, click **Delete**.

3. WebCenter Sites displays a warning message.
4. Click **Delete State**.

Setting Up the Workflow Processes

When you create a workflow process, you specify global process information and then you create steps, assigning step actions to them as needed. The steps in the process create the flow of the process by linking the states in a specific order.

Before you can begin creating a workflow process, you must have already created your step actions, step conditions (if necessary), and states.

To work with workflow processes, your user name must be assigned the Workflow Admin role for the site that you are working with.

> **Note**
>
> If your content providers will use workflow groups, be sure to enable the Workflow Groups tab for them. For information, see "Creating the 'Workflow Groups' Tab," on page 147.

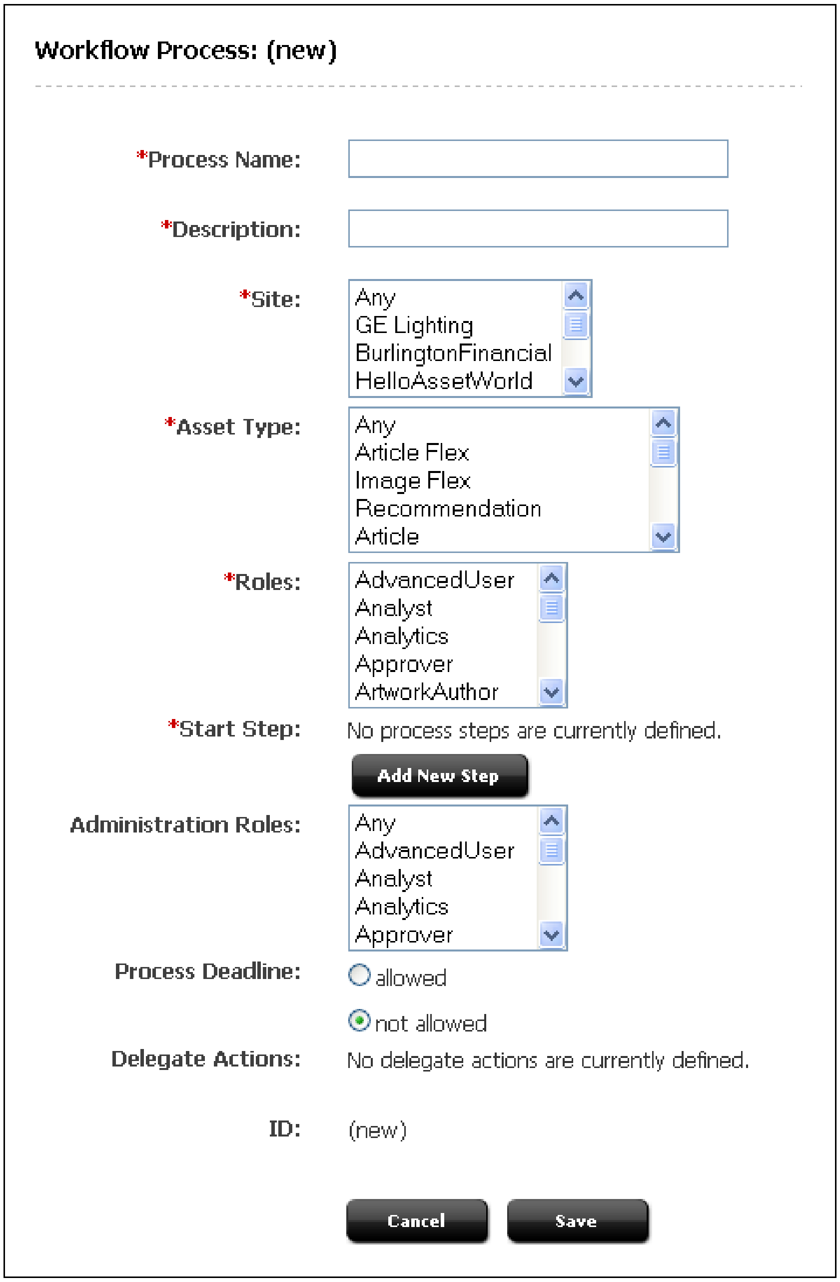
Creating Workflow Processes

When you create a workflow process, you configure three categories of information: global process settings, steps, and function privileges.

Step A: Name and Set Global Settings for the Process

1. In the **Workflow** tab, expand **Processes** and double-click **Add New**.

WebCenter Sites displays the “Workflow Process: (new)” form.

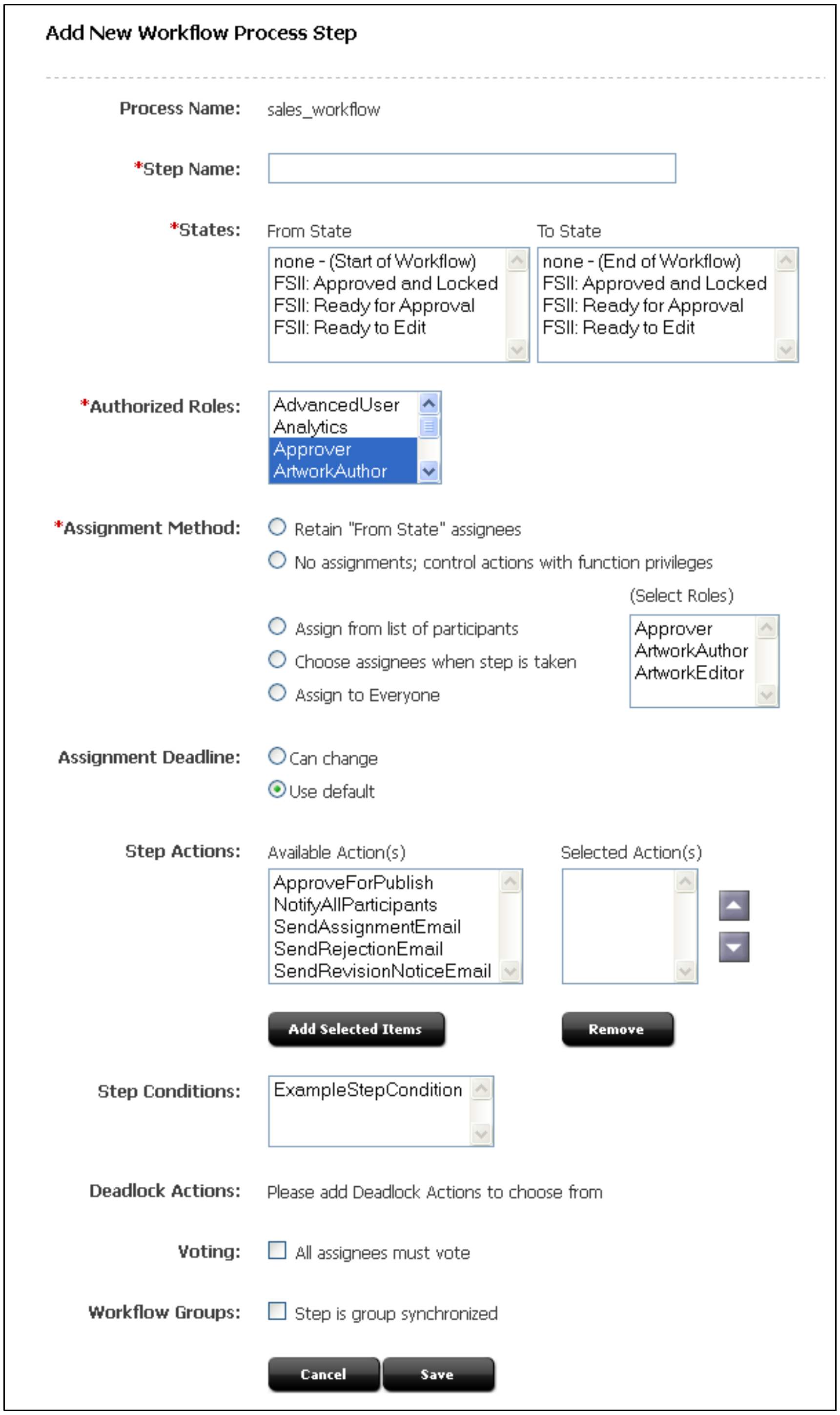


2. In the **Name** field and enter a unique name of up to 25 characters.
3. In the **Description** field, enter a short, informative description of up to 64 characters.
4. In the **Sites** list, select the sites that can use this workflow process.
5. In the **Asset Type** list, select the asset type(s) that this workflow process is for.
6. In the **Roles** list, select the roles that will participate.
7. In the **Administration Roles** list, select the roles that can act as the administrator for this workflow process when an asset is using it.

8. In the **Process Deadline** section, specify whether the workflow administrator (or other user if you configure function privileges that allow it) can set a process deadline for the assets that participate in this workflow process.
9. If you have configured one or more delegate actions, select the appropriate actions in the **Delegate Actions** list.
10. In the **Start Step** section, click **Add New Step**.

    WebCenter Sites displays the "Add New Workflow Process Step" form.

Step B: Create the Start Step



1. In the **Step Name** field, enter a unique, meaningful name of up to 64 characters.
2. Configure the states as follows:
   a. In the **From State** list, select **none (start of workflow)**.
   b. In the **To State** list, select the name of the first state in the workflow process.

3. In the **Authorized Roles** list, select the roles that are allowed to take this step by assigning this workflow to an asset.
4. In the **Assignment Method** section, specify which roles will be assigned the asset by selecting one of the following options:
   - **Retain "From State" assignees**, which, for a start step, means that the user who assigns the workflow to the asset is assigned the asset
   - **No assignments; control access with function privs**
   - **Assign from list of participants**
   - **Choose assignees when step is taken**

     If you select either **Assign from list of participants** or **Choose assignees when step is taken**, select the appropriate roles from the list that is to the right of those options. (For definitions of these options, see "V. Determine the Steps," on page 170.)
   - **Assign to everyone**
5. In the **Assignment Deadline** section, determine whether the workflow administrator (or another user holding the appropriate function privileges) can override the Estimated Time (deadline) set for the state that this step moves the asset to.
6. (Optional) In the **Step Actions** list, select one or more step actions that should be invoked when this step is taken.
7. (Optional) In the **Step Conditions** list, select a step condition, if appropriate.
8. (Optional) Check **Voting** to select that all assignees must vote to approve.
9. (Optional) Check **Workflow Groups** to have the step group synchronized.
10. Click **Save**.

    The start step is added and the "Steps for Workflow Process" form is displayed.
11. Click **Save**.

    WebCenter Sites saves the process and displays it in the "Inspect" form. Note that this step appears as the **Start Step** in the form.
12. Continue to the next procedure.

Step C: Create the Subsequent Steps

Use the following procedure to create the rest of the subsequent steps, including the end step:

1. In the "Inspect" form of the workflow process, click **Edit** in the action bar.
2. At the bottom of the "Edit" form, click **Add/Edit Steps**.
3. In the "Steps for Workflow Process" form, click **Add New Step**.

   WebCenter Sites displays the "Add New Workflow Process Step" form.
4. In the **Step Name** field, enter a unique, meaningful name of up to 64 characters.
5. Configure the states as follows:
   a. In the **From State** list, select the name of the state that you selected as the **To State** for the previous step.

b. In the **To State** list, select the name of the next state in the workflow process. (Note that if you want to create an iterative step, select the same state in the **From State** and the **To State** lists.)

6. In the **Authorized Roles** list, select the roles that are allowed to take this step by finishing the assignment. The roles that you select from this list should either match or contain a subset of the roles that you selected for the **Assignment Method** in the workflow step that immediately precedes this workflow step.
7. In the **Assignment Method** section, specify which roles will be assigned the asset by selecting one of the following options:
   - **Retain "From Step" assignees**

     If the **From State** and the **To State** are different, the first person who takes this step (finishes the assignment) is assigned the asset, even if the previous step assigned the asset to more than one participant.

     If the **From State** and the **To State** are the same, every user who was assigned the asset by the last step is assigned the asset again with this step.
   - **No assignments; control access with function privs**
   - **Assign from list of participants**
   - **Choose assignees when step is taken**

     If you select either of the last two options above, select the appropriate roles from the list that is to the right of those options. (For definitions of these options, see "V. Determine the Steps," on page 170.)
   - **Assign to everyone**
8. In the **Assignment Deadline** section, determine whether the workflow administrator (or other user if you configure the appropriate function privileges) can override the Estimated Time (deadline) set for the state that this step moves the asset to.
9. (Optional) In the **Step Actions** list, select one or more step actions that should be invoked when this step is taken.
10. (Optional) In the **Step Conditions** list, select a step condition, if appropriate.
11. If the step moves the asset from a state in which more than one participant was working on the asset and you want all participants to finish their assignments before the step can complete, select the **All Assignees must vote** option.
12. If there are any other steps in this process with the same **From State** and any of those steps are also all-voting steps (**All Assignees must vote** is selected), select a **Deadlock Action**. For information about deadlocks and avoiding them, see "Managing Deadlocks," on page 154 and "About Delegate Actions," on page 166.
13. If this workflow process will be used for workflow groups and you want all the assets to progress at the same time to the **To State** that you selected in step 4 of this procedure, select the **Step is group synchronized** option. it is recommended that you create only one synchronized step in the process.
14. Click **Save**.

    The step is saved and the "Steps for Workflow Process" form is displayed.
15. Click **Save**.

    WebCenter Sites saves the process and displays it in the "Inspect" form.

16. Repeat this procedure for each step—except the end step—that you want to create for this process.

    To create the end step, continue with the next procedure.

    To configure function privileges, continue to "Step E: (Optional) Configure Function Privileges," on page 194.

Step D: (Optional) Create the End Step

An end step ends the workflow which means that the asset is no longer in a process and no longer has function privileges controlling access to it, if you are using function privileges. Use the following procedure to create an end step for the workflow process:

1. In the "Inspect" form of the workflow process, click **Edit** in the action bar.
2. In the "Edit Process" form, click **Add/Edit Steps** (a button at the bottom of the form).
3. In the "Steps for Workflow Process" form, click **Add New Step**.

   WebCenter Sites displays the "New Workflow Process Step" form.
4. In the **Step Name** field, enter a unique, meaningful name of up to 64 characters.
5. Configure the states as follows:
   a. In the **From State** list, select the name of the state that you selected as the **To State** for the previous step.
   b. In the **To State** list, select none **(end of workflow)**.
6. In the **Authorized Roles** list, select the roles who are allowed to take this step by finishing an assignment. The roles that you select from this list should either match or contain a subset of the roles that you selected for the **Assignment Method** of the workflow step that immediately precedes this workflow step in the process.
7. (Optional) In the **Step Actions** list, select one or more step actions that should be invoked when this step is taken.
8. (Optional) In the **Step Conditions** list, select a step condition, if appropriate.
9. If the step moves the asset from a state in which more than one participant was working on the asset and you want all participants to finish their assignments before the step can complete, select the **All Assignees must vote** option.
10. If there are any other steps in this process with the same **From State** and any of those steps are also all-voting steps (**All Assignees must vote** is selected), select a **Deadlock Action.** For information about deadlocks and avoiding them, see "Managing Deadlocks," on page 154 and "About Delegate Actions," on page 166.
11. If this workflow process will be used for workflow groups and you want this step to be performed on all the assets in the group at the same time, select the **Step is group synchronized** option. it is recommended that you create only one synchronized step in the process.
12. Click **Save**.

    The start step is added and the "Steps for Workflow Process" form is displayed.
13. Click **Save**.

    WebCenter Sites saves the process and displays it in the "Inspect" form.

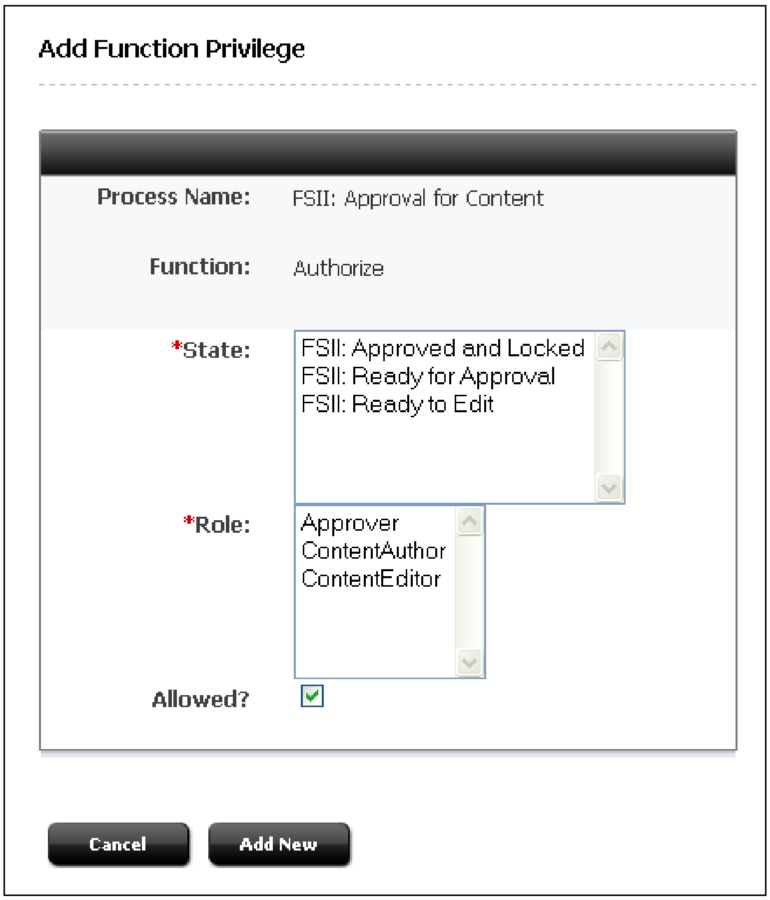
If you do not need to configure function privileges, this workflow process is completed. If you do need to configure function privileges, continue to the next procedures.

Step E: (Optional) Configure Function Privileges

To configure function privileges

1. Find the desired workflow process and open its "Inspect" form.
2. In the action bar, click **Edit**.
3. Click **Add/Edit Function Privileges** at the bottom of the form.
4. In the "Functions for Workflow Process" form, scroll down to the function for which you want to set a privilege.
5. Click **New** next to the desired function.

   WebCenter Sites displays the "Add Function Privilege" form:



6. Select the appropriate state from the **State** list.
7. In the **Roles** list, select the roles that are allowed or not allowed to perform this function when an asset is in this state.
8. Do one of the following:
   - To allow users with the selected roles to perform the function, select the **Allowed** check box.
   - To restrict users with the selected roles from performing the function, clear the **Allowed** check box.
9. Click **Add New**.

WebCenter Sites saves the privilege and redisplays the “Functions for Workflow Process” form.

10. Repeat steps 3–8 for each function privilege that you need to configure. For more information about function privileges, see “VI. Determine the Function Privileges,” on page 173.
11. After you have configured all of your function privileges, click **Save**.

    WebCenter Sites saves the workflow process and redisplays the “Workflow Process” form.

The workflow process is complete.

Editing Workflow Processes

To edit workflow processes

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to edit.
2. In the action bar, click **Edit**.
3. Make your changes as follows:
   - To modify the name, description, or any of the other global settings, make your changes directly in the form.
   - To edit the steps, see “Editing Steps,” on page 195
   - To edit the function privileges, see “Editing Function Privileges,” on page 195.
4. When you are finished, click **Save**.

Editing Steps

To edit a workflow step

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to work with.
2. In the action bar, click **Edit**.
3. In the “Workflow Process” form, click **Add/Edit Steps** (at the bottom of the form).
4. In the “Steps for Workflow Process” form, click **Edit** next to the step you want to edit.
5. In the “Edit Workflow Process Step” form, make your changes, then click **Save**.

   WebCenter Sites saves the process and displays it in the “Inspect” form.

Editing Function Privileges

To edit a function privilege, you must delete it and then re-create it.

To edit a function privilege

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to work with.
2. In the action bar, click **Edit**.
3. In the “Workflow Process” form, click **Add/Edit FunctionPrivs** (at the bottom of the form).

4. In the “Functions for Workflow Process” form, click **Remove** next to the function that you want to change.
5. In the pop-up dialog box that appears, click **OK**.
6. Click **New** next to the function you just removed.
7. In the “Add Function Privilege” form, make your selections and click **Save**.
8. In the “Functions for Workflow Process” form, click **Save**.

   WebCenter Sites saves the workflow process and displays it in the “Inspect” form.

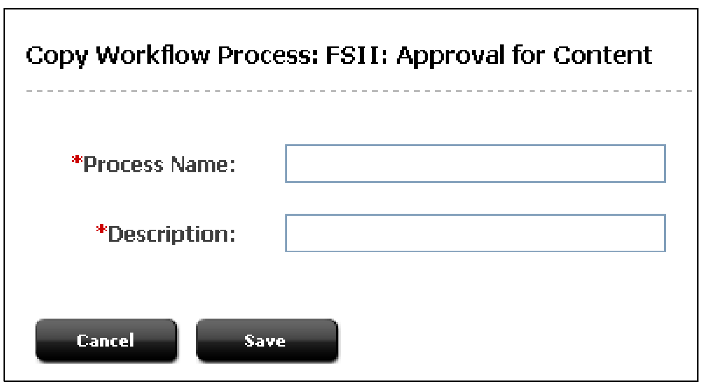
Copying Workflow Processes

You can copy a workflow process, which can save you some steps in configuring additional processes.

To copy a workflow process

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to copy.
2. In the action bar, select **Copy Process** from the drop-down list.

   WebCenter Sites displays the “Copy Workflow Process” form.



3. In the **Process Name** field, enter a unique name for the process.
4. In the **Description** field, enter a short, informative description of the process.
5. Click **Save**.
6. Edit the process as necessary. See the following procedures for help:
   - “Editing Workflow Processes,” on page 195.
   - “Editing Steps,” on page 195.
   - “Editing Function Privileges,” on page 195.

Deleting Workflow Processes

To delete a workflow process

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to delete.
2. In the action bar, click **Delete**.

   WebCenter Sites displays a waning message.

3. Click **Delete Process**.

   The process has been deleted.

Deleting Steps

To delete workflow steps

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to work with.
2. In the action bar, click **Edit**.
3. In the "Edit Process" form, click **Add/Edit Steps** (at the bottom of the form).
4. In the "Steps for Workflow Process" form, click **Remove** next to the step you want to delete.
5. In the pop-up dialog box that appears, click **OK**.
6. Click **Save**.

   WebCenter Sites saves the workflow process and displays it in the "Inspect" form.

Deleting Function Privileges

To delete function privileges

1. In the **Workflow** tab, expand **Processes** and double-click the workflow process you want to edit.
2. In the action bar, click **Edit**.
3. In the "Workflow Process" form, click **Add/Edit FunctionPrivs** (at the bottom of the form).
4. In the "Functions for Workflow Process" form, click **Remove** next to the function privilege you want to delete.
5. Click **Save**.

   WebCenter Sites saves the workflow process and displays it in the "Inspect" form.

Testing Your Workflow Process

Before you move your workflow process to the management system and implement it there, test it on your development system.

To test your workflow process

- Log in as a user with administrator access and configure start menu items that assign workflow processes to assets, if necessary.
- Log in as a user who has a role that has been authorized to take the start step of the workflow process. Create an asset, select the workflow process (only if the start menu item does not assign it), and then finish the assignment.
- Log in as the participant who has the assignment now. Finish the assignment and then log in as the next participant. Continue through the entire workflow in this manner.
- Verify that the e-mail messages that should be sent are being sent.
- Verify that your workflow sends the asset through the process correctly.

Moving Your Work

Typically you create and fine-tune a workflow process on a development system to ensure that it functions exactly as you need it to before you introduce that workflow process to the management system.

When your workflow is ready to be used by content providers, use the Initialize Mirror Target feature to move your workflow components to the management system.

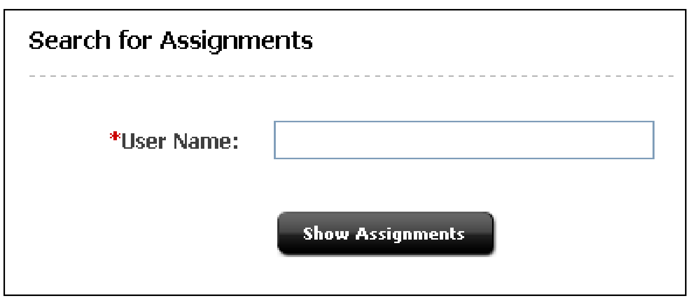
For information about this feature, see "Migrating a Site from One System to Another," on page 314.

Clearing Workflow Assignments

People go on vacation, get reassigned to new work groups, and move on to different jobs. What should happen to their workflow assignments in these situations? One option for handling work assignments that cannot be finished by the original assignee is to delegate the assignments to someone else. However, even with the delegate feature, you, the administrator, might still be called on to clear assignments from some user's assignment list.

To clear assignments for a user

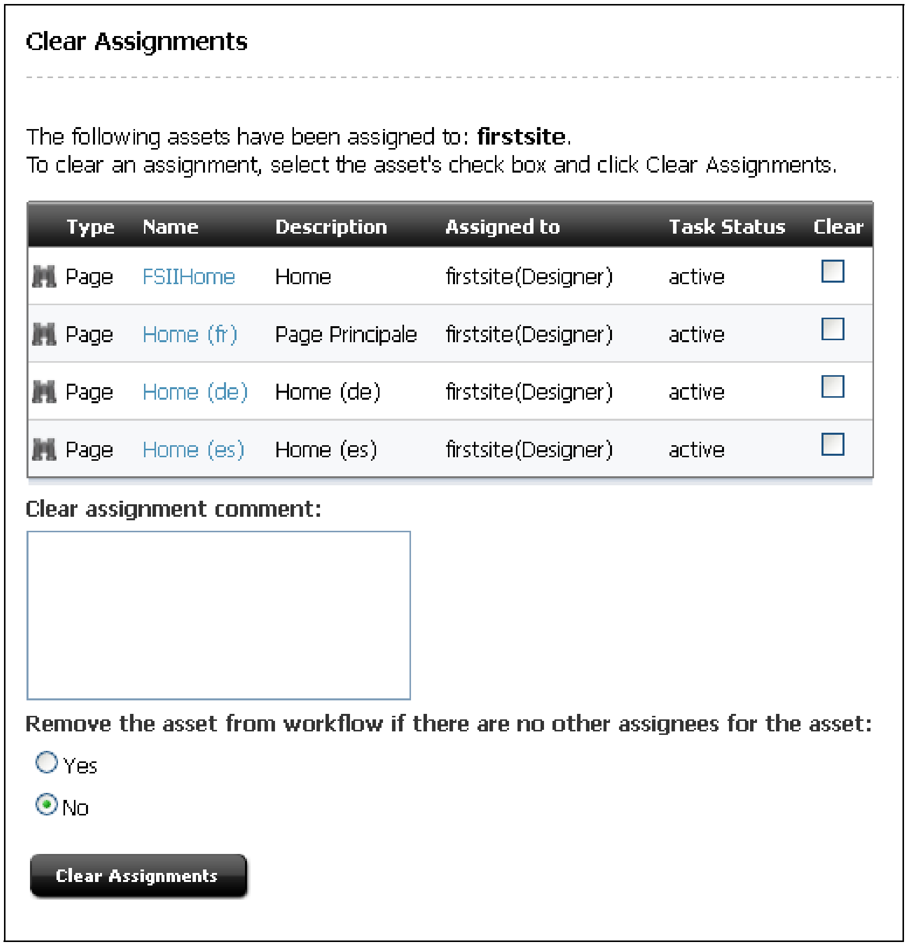
1. In the **Admin** tab, double-click **Clear Assignments**.
2. In the "Search for Assignments" form, enter the desired user name. For example:



3. Click **Show Assignments**.

   WebCenter Sites displays a list of assets that are currently assigned to that user.

For example:



4. In the “Clear Assignments” form, select the **Clear** check box next to each assignment you want to clear.
5. (Optional) In the **Clear assignment comment** field, enter a brief explanation of the reason for which you are clearing the selected assignments. The explanation you enter in this field appears in the **Action Taken** field in the **Workflow History** section of the asset’s “Status” form.
6. Do one of the following:
   - If you want the asset to be removed from workflow if it is not assigned to anyone else, select **Yes**.
   - If you want the asset to remain on the assignment list of any other user who is also assigned the asset, select **No**.
7. Click **Clear Assignments**.

   WebCenter Sites displays a “Clear Assignments Report” summarizing your changes.

Chapter 10

Replicating CM Sites

To speed up the deployment of online sites, WebCenter Sites provides Site Launcher. This new feature enables you to replicate a suitably-prepared CM site and, in the process, either share or copy its components to the new site. You can then modify the new site as necessary, and deploy it. This chapter provides an overview of Site Launcher, replication guidelines, requirements and options, and procedures for enabling and using Site Launcher.

This chapter contains the following sections:

Site Launcher Overview

To minimize your effort in creating new sites, WebCenter Sites provides a site-replication utility called “Site Launcher.” This utility is designed not for backing up CM sites, but for spinning them off.

For example, your management system hosts a dedicated CM site for a department named “Products.” A new department named “Services” has been recently established and must be quickly introduced to the public. “Services” is similar to “Products” in size and structure, although its members are public relations specialists rather than advertising staff. Instead of creating a “Services” CM site from scratch, you can replicate the “Products” CM site directly on the management system, and modify the replicate to accommodate the newly established “Services” department.

Site Launcher replicates source sites directly on their native WebCenter Sites system, re-using the existing database schema. The sites are replicated quickly and easily, without the need for coding. However, while replication itself is a quick and straightforward procedure, it does require preparation and follow-up on the part of the administrator. How much, depends on the content management needs.

The rest of this chapter provides the steps that you need to follow in order to successfully replicate sites for use in any content management model—1:1, 1:*n*, or *x*:*n,* described in “Content Management Models,” on page 34.

Note

Site replication can be carried out only by the general administrator. Site and workflow administrators cannot replicate the sites they manage, as they have no access to Site Launcher.

Preparing for Replication

Before attempting to replicate a site, you need to consider several recommendations regarding the nature of the site and its replicate. You must also make decisions about which components to copy or share, and finally ensure that the source site meets system requirements for replication. This section outlines our recommendations, your options, and the system requirements.

Ensuring the Source Site Meets Replication Requirements

Site Launcher can be used to replicate almost any CM site: small, large, functional, incomplete, independent of other sites, and overlapping other sites by the sharing of components.

However, to use Site Launcher most effectively, it is best for you to start with a site that is small, functional, and similar to the site that you need to have. In general, the source site should be a skeletal one, meaning that it has structure and design, but little content. These characteristics, beyond ensuring a tractable replicate, will help you conserve resources and minimize replication time.

In addition, make sure that the source site resides on the management system. (If you need to first create a replicable source site, follow the steps in "Site Configuration Steps," on page 60.)

Planning the New Site

This section outlines your site replication options and the system requirements.

Copying vs. Sharing

When designating a source site, you must decide whether to copy or share certain site components, and therefore, determine from the start, how the new site will function in relation to its source site—as a duplicate or a subset; as an independent site or an overlapping site. (On a bigger scale, you will also need to determine how the new site will function in relation to other sites in your content management model.)

The table below shows that when the source site is being replicated, most if its components are shared by default. Other components are shared or copied at your discretion. Users are neither copied nor shared.

Source Site Component	Replication method
asset subtypes	Shared
asset types	Shared
assets	Either copied or shared (admin decides)
associations	Shared
Sites Desktop and Sites DocLink configuration	Copied
publishing destinations	Shared
roles	Shared
start menu items	Shared
templates (design assets)	Either copied or shared (admin decides)
tree tabs	Shared
users	Neither shared nor copied
workflow processes	Shared

Note that copying and sharing of assets is selective at the asset type level, but not at the asset level—when you select an asset type to be copied (or shared), *all* assets of that type are copied (or shared). The same holds for templates (design assets).

Concerning assets and templates, your specific tasks are to determine:

- Which asset types must have their assets copied or shared.
- Whether template assets must be copied or shared.
- Whether assets will be previewed on the new site as formatted content.

To determine whether assets and templates must be copied or shared, use the following guidelines:

- Sharing an asset does not require you to share the template that renders the asset. The template can be copied.

  Whether you share or copy an asset depends on how the rendering logic in the template treats asset names. For example, template logic can be written in such a way as to assume either shared templates (constructed with explicit template names in the `render:satellitepage` and `render:callelement` tags) or copied templates (constructed with a template name using `Variables.site` prepended to the name in the `render:satellitepage` and `callelement` tags).

  The same reasoning applies to other asset types. If in your rendering logic you use asset names such as flex attribute name, or image name, then the template logic will treat the asset type as either shared (explicit name) or copied (constructed name), and you need to designate the source site accordingly.

  If you decide to copy template assets, make sure that the template assets do not contain hard-coded template names, and a template is not named in the "Default Values" list of any start menu item.

- When assets are shared to a new site by Site Launcher, their data is not changed. The single exception is Template assets. A Template asset maintains a SiteCatalog entry for each site to which it is shared. Therefore, when a template is shared to a new site, the template's list of SiteCatalog entries is updated (the new site is added to the list of SiteCatalog entries).

- Because data in a shared asset is not changed, asset references of that asset also remain unchanged. Therefore, if assets with asset references are to be shared, the asset references must also be shared. For example, if you have Article assets that have references to ImageFile assets, you may

  - share both asset types,
  - copy both asset types, and
  - copy the Articles and share the ImageFiles.

  However, you must not share the Article and copy the ImageFiles. Doing so will cause errors to be displayed when a copy of the site is launched.

- Start menus are shared among the source site and its replicates. Therefore, if a workflow process is set in the source site's start menu, that workflow will be set in the replicate sites. When users in the new site invoke the start menu item to create an asset, the workflow will take effect.

Your decisions in the planning stage are governed by the specifics of your installation as well as your own methods of managing the WebCenter Sites environment. The site that you create by replication must ultimately fit in to the content management model that is being (or has been) established on the management system. For example, if you have central sites and centrally managed sites, you need to determine how your new site will function in relation to them. For information about site modeling, see "Content Management Models," on page 34.

Naming Assets

Assets that are copied to the new site are named algorithmically, by use of prefixes as follows:

- The prefix for the source is specified when you designate a site as a site launcher source site and is, by default, the name of the site.
- The prefix for the new site is specified in Site Launcher itself and is, by default, the name of the new site. When specifying a prefix, make sure it is short.
- If the source asset has a prefix, or the asset type requires a unique name, then a prefix will be used in the name of the new asset. If the source asset has a prefix, the new asset will have the new prefix in place of the old prefix.
- If the source asset does not have a prefix, then the prefix is prepended to the asset name to make the new asset name.

Planning Users

Given that users are not copied or shared to the new site during replication, you will have to manually add them to the new site.

If the users will be different from those on the source site, you will need to create new user accounts. In the process, consider the users' functions on the new site, the roles they will need, their access to start menu items, workflows, tree tabs, and so on.

Bear in mind that the original associations among components are preserved on the new site. If you redefine users' functions, you may have to redefine their associations with other components on the site. Accounting for new associations in the planning stages will help you to assess your post-replication workload and create a functional site that also meets your expectations.

Choosing the Replication Time

Because normal operations affect site replication, it is best to run Site Launcher when the source site is not being actively edited. Otherwise, the replicate site might not match the source site. Also, make sure that prior to replication, all assets in the source site are checked in to the database. Otherwise, if revision tracking is enabled and an asset to be shared is checked out, the sharing of the asset will fail.

Site Replication Steps

This section shows you how to enable and use Site Launcher. The procedure consists of three basic steps:

I. Ensure that Replication Requirements Are Satisfied

II. Enable the Source Site for Site Launcher, on page 206

III. Replicate the Source Site, on page 207

The steps are given in detail in the rest of this chapter.

I. Ensure that Replication Requirements Are Satisfied

1. Prepare for replication by ensuring that source sites and planned sites meet the requirements. For more information, see "Preparing for Replication," on page 202.
2. Go to section "II. Enable the Source Site for Site Launcher" to enable the source site.

II. Enable the Source Site for Site Launcher

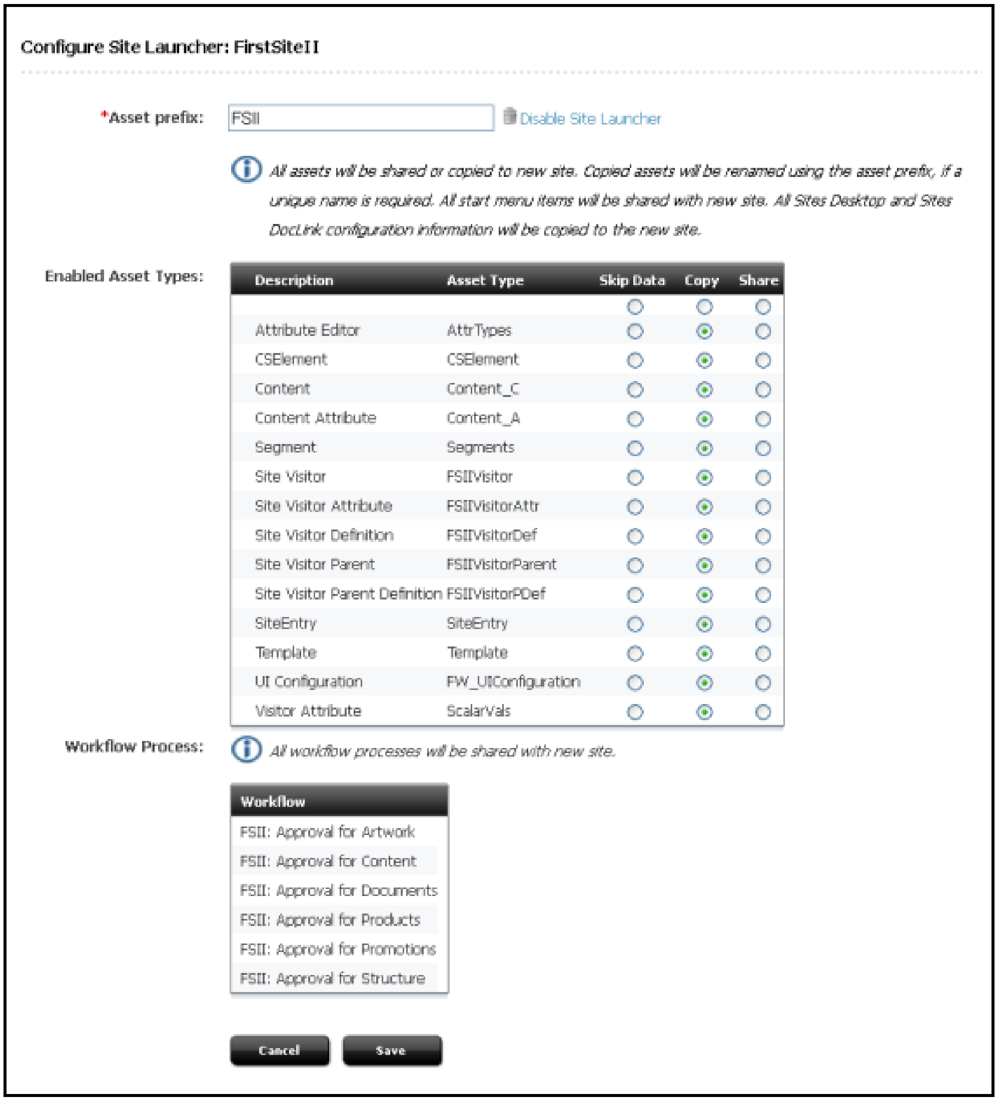
In order to replicate a source site, you must first configure Site Launcher.

To configure Site Launcher

1. Make sure that you have completed the steps shown above ("I. Ensure that Replication Requirements Are Satisfied").
2. When you complete the steps in this section, be prepared to follow up immediately with the steps in the next section, "III. Replicate the Source Site." Immediate follow-up ensures that the condition of the source site has not changed.
3. In the **Admin** tab, expand **Sites** and double-click the site you want to replicate. This is your source site.
4. In the site's "Inspect" form, click **Configure Site Launcher for this site** (at the bottom of the form).
5. In the "Configure Site Launcher" form, do the following:
   a. If you want to copy assets, go to the **Asset prefix** field and either enter a prefix for the assets, or accept the default (the name of the source site).
   b. In the "Enabled Asset Types" panel, locate the asset types whose assets you want to copy or share, and click their **Copy** or **Share** radio buttons.

> **Note**
>
> If you want to copy or share all assets of all types, click the **Copy All** or **Share All** button.

c. Click the **Save** button to save your configuration options.



6. Go to the next section, "III. Replicate the Source Site," to complete the site replication procedure.

III. Replicate the Source Site

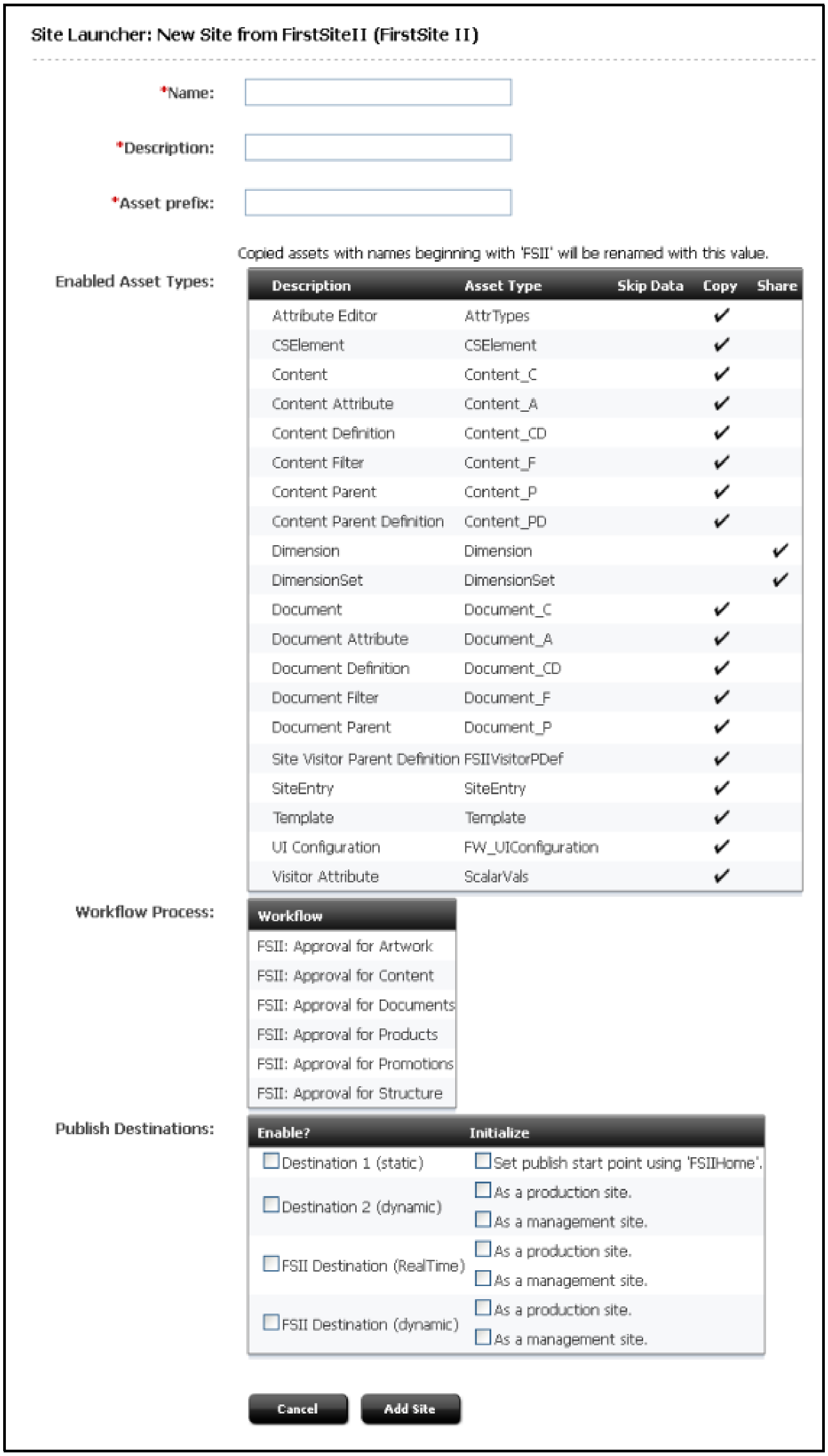
Once you have enabled the source site for replication, you can use Site Launcher to create as many copies of the source site as you wish.

To replicate the source site

1. Make sure that you have completed the steps above ("I. Ensure that Replication Requirements Are Satisfied," and "II. Enable the Source Site for Site Launcher").
2. At the bottom of the "Inspect" form, click **Launch Copy**.
3. In the "Site Launcher" form, do the following:
   a. In the **Name** field, enter the name for the new site.

   b. In the **Description** field, enter the description for the new site.
   c. In the “Publish Destinations” area:
      1) Select the publish destinations you would like to be available for the new site.
      2) Initialize the destinations by selecting the checkboxes, as necessary.

d. Click **Add Site** to copy the site.



Site Launcher: New Site from FirstSiteII (FirstSite II)

*Name:

*Description:

*Asset prefix:

Copied assets with names beginning with 'FSII' will be renamed with this value.

Enabled Asset Types:

Description	Asset Type	Skip Data	Copy	Share
Attribute Editor	AttrTypes		✔	
CSElement	CSElement		✔	
Content	Content_C		✔	
Content Attribute	Content_A		✔	
Content Definition	Content_CD		✔	
Content Filter	Content_F		✔	
Content Parent	Content_P		✔	
Content Parent Definition	Content_PD		✔	
Dimension	Dimension			✔
DimensionSet	DimensionSet			✔
Document	Document_C		✔	
Document Attribute	Document_A		✔	
Document Definition	Document_CD		✔	
Document Filter	Document_F		✔	
Document Parent	Document_P		✔	
Site Visitor Parent Definition	FSIIVisitorPDef		✔	
SiteEntry	SiteEntry		✔	
Template	Template		✔	
UI Configuration	FW_UIConfiguration		✔	
Visitor Attribute	ScalarVals		✔	

Workflow Process:

Workflow
FSII: Approval for Artwork
FSII: Approval for Content
FSII: Approval for Documents
FSII: Approval for Products
FSII: Approval for Promotions
FSII: Approval for Structure

Publish Destinations:

Enable?	Initialize
☐ Destination 1 (static)	☐ Set publish start point using 'FSIIHome'.
☐ Destination 2 (dynamic)	☐ As a production site. ☐ As a management site.
☐ FSII Destination (RealTime)	☐ As a production site. ☐ As a management site.
☐ FSII Destination (dynamic)	☐ As a production site. ☐ As a management site.

Cancel Add Site

After the new site is created, WebCenter Sites displays a summary that indicates:

- Which assets were copied or shared. For each asset that is shared, WebCenter Sites adds a row in the `AssetPublication` table, indicating the site id (publication id) of the new site. For each asset that is copied, WebCenter Sites enters the copy into the table where the original asset is stored.
- The number of start menu items, workflow processes, and publish destinations that were shared with the new site.

4. At this point, you need to add users and otherwise ensure that the new site is properly configured. For instructions and guidelines, see the next section, "Post-Replication Tasks and Guidelines."

Post-Replication Tasks and Guidelines

When the new site is established, you need to complete its configuration by completing the following steps:

1. Add existing users to the new site, or establish new users, depending on how you planned the new site.
   - To add existing users, follow instructions in "Granting Users Access to a Site (Assigning Roles to Users)," on page 112.
   - To establish new users, follow the guidelines in Chapter 3, "Site Configuration Guidelines."
2. If workflow processes exist, make sure that the correct asset types and roles are associated with the processes through Start Menu items.
3. Test the new site.
4. If necessary, create a site administrator. Be sure to assign the xceladmin ACL to the user account and give the user the SiteAdmin role for the new site.

Part 3

Export and Mirror Publishing

This part describes Oracle WebCenter Sites' Export and Mirror publishing systems and the approval system which validates content intended for publication. This part also contains procedures for configuring and managing these publishing methods.

This part contains the following chapters:

Chapter 11

Publishing with Oracle WebCenter Sites

Content is made available to the visitors of your online site by copying from the management system to the delivery system. Copying content from one system to another is called **publishing**.

The WebCenter Sites publishing system supports three publishing methods: Export to Disk, Mirror to Server, and Export to XML. WebCenter Sites also provides an approval system that determines which content gets published, a scheduling function that enables you to set the publication time, and a utility for configuring site-specific publishing destinations.

This chapter describes the publishing and approval systems. It contains the following sections:

- Overview
- Publishing Methods
- Publishing Destinations
- The Approval System
- The Publishing Schedule
- What Happens During a Publishing Session?
- Obtaining Information About a Publishing Session

Overview

Before assets can be published, either you or WebCenter Sites must determine the following information:

- Which **publishing method** WebCenter Sites should use: Export to Disk, Mirror to Server, or Export Assets to XML.

  As an administrator, you specify the publishing method for the destination when you configure the publishing destination. For more information, see "Publishing Methods."

- To which **publishing destination** WebCenter Sites should publish during a given session.

  As an administrator, you configure the publishing destinations for your system. For more information, see "Publishing Destinations," on page 217.

- Which assets have been approved and are ready to be published to the current destination. The **approval system** determines this information.

  When an asset is deemed ready for publication, a content provider marks it as "approved" for a specific publishing destination. The approval system validates the "approved" label to determine whether publishing the asset is likely to create broken links on the live site. If the potential for broken links exists, the asset is held back from the publishing session until its dependencies on other assets are resolved by the user approving those assets, as well. If the approved asset either has no dependencies or its dependencies are satisfied, the approval system releases the asset to the publishing system.

  The approval system's process is complex and varies from one publishing method to another. For detailed information about the workings of the approval system, see Chapter 12, "The Approval System" and Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process."

- When assets should be published.

  As an administrator, you set up the **publishing schedule**. The publishing process runs as a batch process that you schedule to occur at regularly-occurring intervals. On an as-needed basis, you can also override the schedule and publish on demand. For more information about publishing schedules, see "The Publishing Schedule," on page 218.

During a publishing session, the session information is recorded in log files. You can monitor the session through the **Publish Console** to determine both the publishing history and the status of currently running publishing sessions. You can also use your browser to complete other administrative tasks, as the publishing process runs in the background. For information about the events that occur during a publishing session, see Chapter 11, "What Happens During a Publishing Session?" For information on how to obtain publishing information, see Chapter 11, "Obtaining Information About a Publishing Session."

Publishing Methods

This section describes the publishing methods that WebCenter Sites supports. Figure 6, on page 216 summarizes the publishing methods.

- For **Export to Disk publishing**, the delivery system is a web server. Publishing to this type of system is done as follows:

  Approved assets in the CM system database are rendered by templates into HTML files. The files are saved to a file system and subsequently published to the web server by an administrator using a transfer protocol, such as FTP.

  When published content is requested by site visitors, the HTML files are served as pages to the browser.

- For **Mirror to Server** publishing, the delivery system is a WebCenter Sites system. Publishing to this type of system is done as follows:

  Approved assets and their database tables are mirrored from the CM system database to the delivery system database. Throughout the publishing session, the publishing system communicates and cooperates with the **CacheManager** on the delivery system. The CacheManager is a WebCenter Sites servlet that manages a system's page cache. CacheManager ensures caching of the pagelets or pages that refer to the assets which will be mirrored. After the publishing session concludes, CacheManager generates those pages again to display the updated content, and caches the new pages and pagelets.

  When published content is requested by site visitors, it is provided on-the-fly. The content is drawn from the delivery system database by templates and served to the browser, where it is finally rendered as pages.

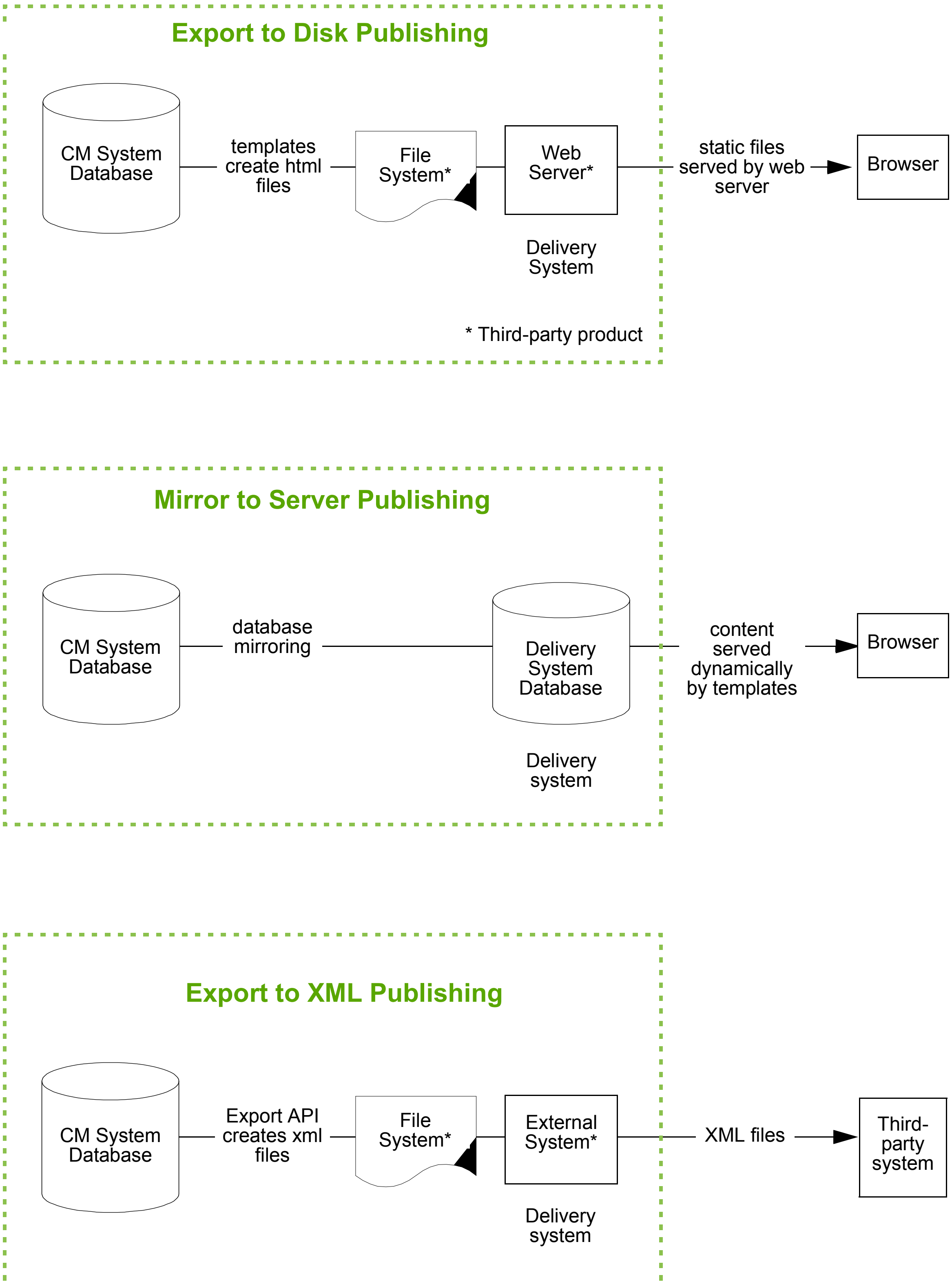
- For **Export Assets to XML**, the delivery system is a database or an external system running an application other than WebCenter Sites. The publishing method is a data transformation method that outputs XML files. Rather than creating pages that are ready to be displayed by a web server, this publishing method uses the Export API to create one XML file for each approved asset.

  When published content is requested by site visitors, it is provided on-the-fly. The content is drawn from the delivery system database by templates and served to the browser, where it is finally rendered as pages.

These chapters describe each publishing method in detail:

- Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process"
- Chapter 14, "The Export to Disk Publishing Process"
- Chapter 15, "The Mirror to Server Publishing Process"
- Chapter 16, "The Export Assets to XML Publishing Process"

Figure 6: WebCenter Sites' Publishing Methods



Publishing Destinations

A publishing destination is either the WebCenter Sites database to which you are mirroring content, or a directory to which you are exporting. A publishing destination is expressed as a named object that defines the following parameters:

- A publishing method (Export to Disk, Mirror to Server, or Export Assets to XML)
- A location
  - For Export to Disk publishing and Export Assets to XML, the publishing system converts the approved assets into HTML or XML files. The location is the root directory to which the files are saved. You set this directory with the `cs.pgexportfolder` property in the `futuretense.ini` file.
  - For Mirror to Server publishing, the destination is the **server name in URL format** (including the port number) of the server that is supposed to deliver the content. (You set the server name in the **Destination address** field of the "Destination" form.)
- The names of CM sites whose assets can be published to the destination.

For any publishing method, you can configure more than one publishing destination for your Sites system. Depending on how your online sites are designed by the developers, you may need to publish both static content (HTML files, using Export to Disk) and dynamic content (pages generated on the fly by the Mirror to Server or Export to XML publishing method). For this kind of dual implementation, consult with your developers to determine how to configure your publishing destinations.

In this document, the following terms are used when discussing publishing destinations and configuration:

- **Source.** The WebCenter Sites database that serves as the source for a publishing session. Because you can mirror assets from any Sites system to any other Sites system, the source is not always the content management system.
- **Destination.** Either the WebCenter Sites database that you are mirroring to or the directory that you are exporting to.

For information about creating publishing destinations, see "Step 3. Configure an Export Destination," on page 275 and "Step 5: Create a Mirror Destination," on page 293.

The Approval System

When assets are approved for publication on a specific destination, WebCenter Sites verifies that each asset is ready to be published by invoking the approval system. The approval system runs a pre-publication process that protects the online site against the possibility of broken links and outdated content. In this process, the approval system determines answers to the following questions:

1. Which assets are deemed to be ready for publishing—that is, which assets are marked by the user as approved?
2. Do the approved assets have dependencies? That is, must any other assets also accompany those approved assets? or, must any other assets also exist on the destination to ensure that all the links on the site will work correctly?

For example, you may have a site set up that has article assets that refer to image files. An approved article with an unapproved associated image file cannot be published until the image file is also approved.

Depending on the answers to these questions, the approval system either allows the publishing system to publish approved assets at the next publishing session, or it stalls the publication of assets that fail the dependency test.

Note

- In Export to Disk publishing, the choice of templates can cause assets to bypass the approval system. For more information about the approval system in Export to Disk publishing, see Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process."
- It is typical to build an approval step into your workflow processes. For information about workflow, see Chapter 9, "Creating and Managing Workflow Processes."
- For information about how to approve assets, see the chapter on publishing in the *Oracle WebCenter Sites User's Guide*.

Detailed information about the workings of the approval system is given in Chapter 12, "The Approval System" and Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process."

The Publishing Schedule

A publishing session (no matter what the publishing method) runs as a background, batch process called a **publish event**. As the administrator, you configure the schedule for publishing events.

You can schedule publishing events to occur daily, weekly, hourly, every 15 minutes, or in various combinations of these time increments. When scheduling multiple publishing sessions, bear in mind several rules:

- Publishing to a given destination must be done serially.

  Because each publishing session is a background event, a given destination can support only a single publishing session at a time. If publishing to a specific destination is still in progress when the next event for that destination is scheduled, the second event attempts to run and then fails, reporting that a publishing session to that destination is already underway.

  Similarly, multiple sources must not be configured to publish simultaneously to the same destination. Doing so will cause publishing errors and problems with data integrity.

- Never schedule a publishing session to a destination for a time when the destination system could be publishing to another destination.

  For example, you publish from the development system to the management system, and from the management system to the delivery system. Do not schedule a publishing session from the development system to the management system while the management system is publishing to the delivery system.

- Publishing to multiple destinations can be done simultaneously. You can publish to multiple destinations at the same time by setting up events for the destinations and selecting the same time for them.

Because a publishing event completes database transactions, the publishing feature must have a user account specified for it. This user is called the **batch user** and you use the `xcelerate.batchuser` and `xcelerate.batchpass` properties in the `futuretense_xcel.ini` file to identify the batch user account for your Sites system.

Note

Publishing also requires a mirror user, different from the batch user. For more information, see "Users and Mirror to Server Publishing," on page 288.

What Happens During a Publishing Session?

When the publishing system begins publishing approved assets, WebCenter Sites does the following:

- Creates a publishing session by adding a row to the `PubSession` table and assigns the session a unique ID, called the PubSession ID.
- Runs a query to gather all the assets that are approved and are ready to be published to the destination that the session is publishing to.
- Locks the assets that are returned by the query, and also notifies the CacheManager about these assets, so that they cannot be edited while the publishing session is underway.
- Invokes the appropriate element for the publishing method, passing it the list of assets that should be published.

Then, those assets are either rendered into files or mirrored to a destination database, depending on the publishing method in use. For information about what the Export to Disk, Mirror to Server, and Export Assets to XML publishing methods do with the list of approved assets, see the following chapters:

- Chapter 14, "The Export to Disk Publishing Process"
- Chapter 15, "The Mirror to Server Publishing Process"
- Chapter 16, "The Export Assets to XML Publishing Process"

When the publishing session concludes, the publishing system notifies the approval system of which assets were published and then it updates the page cache.

Obtaining Information About a Publishing Session

WebCenter Sites writes information about each publishing session to log files and to several tables in the WebCenter Sites database. Much of this information is displayed in the **Publishing Console** as the publishing history for a session. To display the **Publishing Console**, click the **Publish** button located in the top button bar in the main window. See also "Troubleshooting," on page 329.

Note

You must have the `xcelpublish` ACL to view the **Publish Console**.

Chapter 12

The Approval System

This chapter explains the basic workings of WebCenter Sites' approval system. It does not contain procedures on performing specific operations. Rather, its purpose is to help the administrator gain a basic understanding of the approval system's process, which is both complex and highly interactive, especially in the case of Export to Disk publishing.

Each section of this chapter is written to be as self-contained as possible, with many examples to illustrate the concepts. For continuity, users can read the sections sequentially, and for reference, turn to the section of interest. As this chapter progresses, it builds on the introductory material (especially approval-system terms and definitions), and ends with a comprehensive picture of how the approval system performs its function.

Note

Readers who would like a summary of the approval process can skip to the section "Putting It All Together," on page 243.

This chapter contains the following sections:

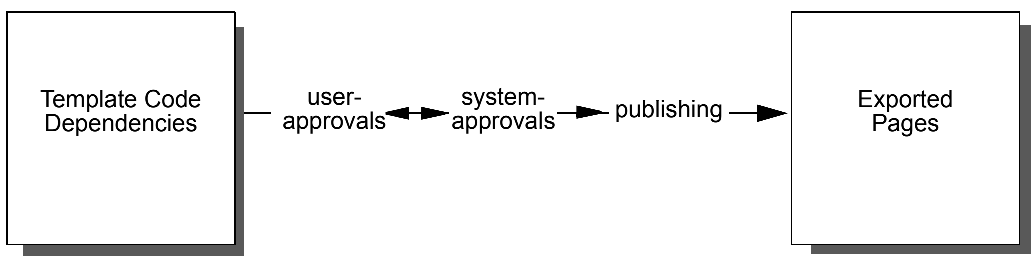
Overview

Before a publishing session can begin, the user is required to approve assets for the publishing destination. This task is far more intricate than it appears to be. Enterprise sites host enormous numbers of assets, making it impossible for users to track which assets are bound to each other by dependencies and therefore must be approved together in order to be published. As a result, users typically approve only some of the assets that require approval. The unapproved assets compromise the integrity of the approved assets by causing broken links among them. This is where the approval system takes over.

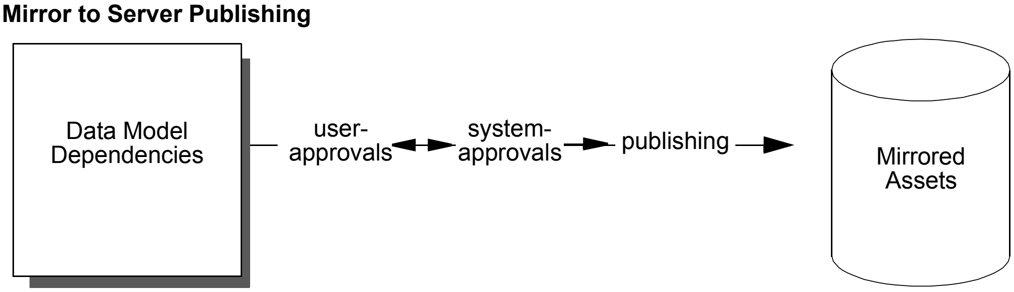
The ultimate purpose of the approval system is to protect the integrity of published content. To this end, the approval system does not automatically allow the publication of user-approved assets. Instead, it evaluates the condition of the user-approved assets in order to determine whether they are indeed ready for publication. If necessary, the approval system then assists the user in approving the supporting assets.

During its operation, the approval system ensures that asset dependencies, encoded in templates or the data model, are kept intact among approved assets, so the dependencies can be reproduced on the publishing destination when the assets are published. If an approved set of assets is free of broken links, it is likely to be free of broken links when published. If an approved set of assets upholds version matching, it is likely to uphold version-matching when published.

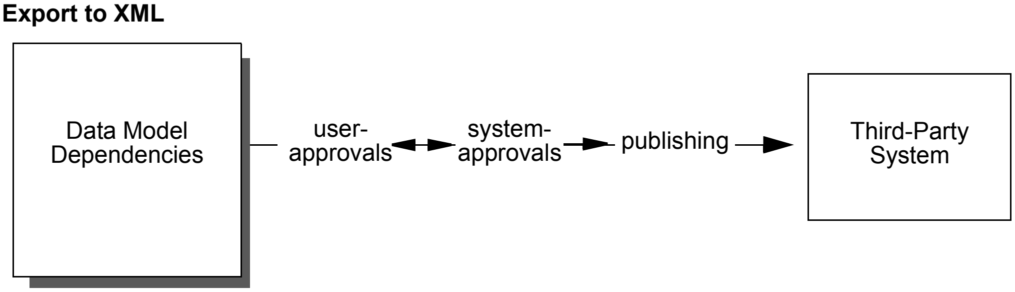
Export to Disk Publishing



Mirror to Server Publishing



Export to XML



A Brief Look at Dependency Analysis

When analyzing approved assets for broken links and inconsistent versions the approval system performs a dependency test:

1. The approval system determines whether dependencies among *user-approved* assets match dependencies that are specified in either the template code or the data model, taking into consideration the publishing method. As shown in Figure 8, on page 228:
   - In Export to Disk publishing, the approval system tests dependencies (among approved assets) against template code. The dependencies are determined as the code in the template is evaluated. If a default template has been assigned to assets of a given type, the approval system uses that template to determine the dependencies. If there is no default template, the approval system uses the template that is specified for the given asset. (For detailed information about approvals and Export to Disk publishing, see Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process.")
   - In Mirror to Server publishing and Export to XML, the approval system tests dependencies (among approved assets) against the data model.
2. If the approval system finds that dependencies among user-approved assets are satisfied, it allows the assets to be published and releases them to the publishing system for inclusion in the next publishing session.

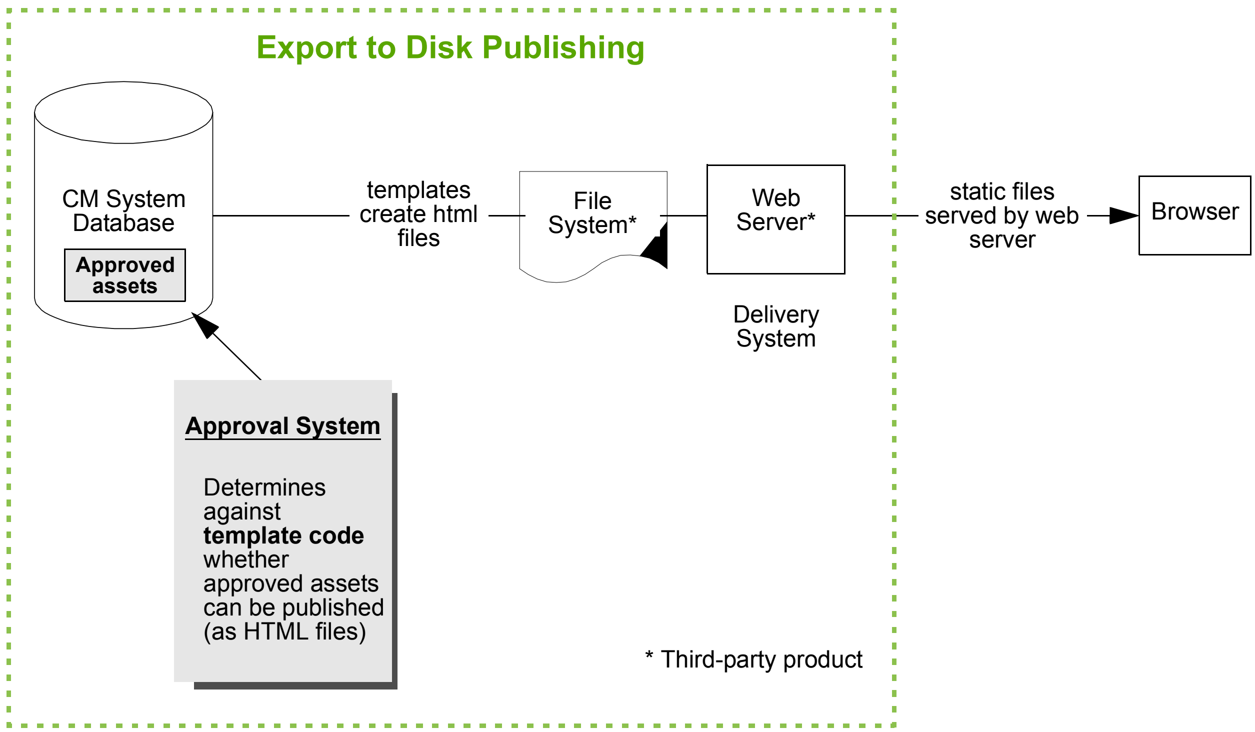
   However, if it finds the dependencies to be breached, the approval system looks for published assets that will satisfy those dependencies at publication time and therefore turns to published content. The approval system scans the `PublishedAssets` table to determine what has been published to the target destination:
   - If an asset requiring approval does not exist on the target destination (and in the correct version, if applicable), the approval system prompts the user to approve the asset.
   - If the asset requiring approval already exists on the target destination (and in the correct version, if applicable), the asset will satisfy the dependency at publication time and therefore does not require approval (and re-publication).

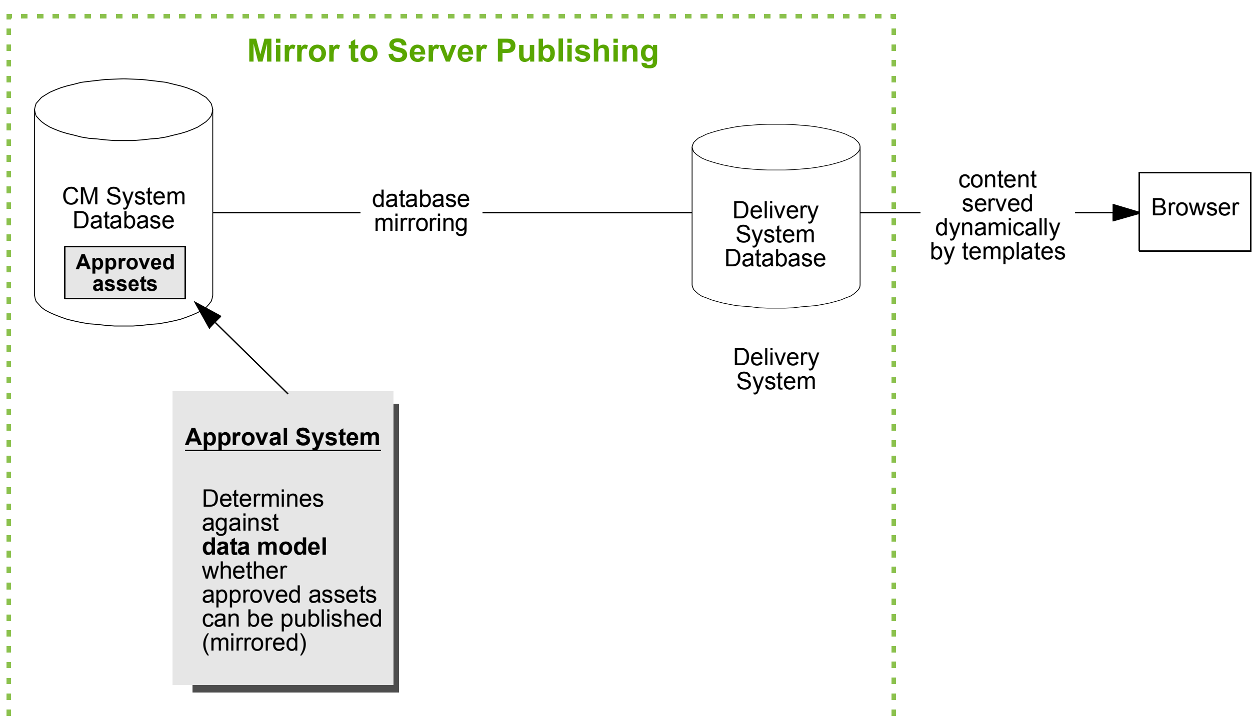
Note

Version matching can be imposed by the Sites administrator, in which case user-approved assets must match the versions of published assets in order to satisfy the dependency criteria (which are called `exists` and `exact`).

For Mirror to Server publishing, version matching is specified in the `mwb.conservativedependencies` property, in `gator.ini`. For Export to Disk publishing it is specified in the template code. For more information see "Ensuring Version-Matched Content," on page 237.

Figure 7: Approvals and Publishing Methods





3. On completing its analysis of all the user-approved assets, the approval system either approves or disallows the publishing of certain assets:

   The approval system supports fractional publishing rather than "all or none" publishing. Assets that pass the dependency test are released to the publishing system for inclusion in the next publishing session. Assets that fail the dependency test are held back from the publishing system, and the user is informed as to which additional assets also require approval. Dependencies among approved assets are hence the cause of lighter-than-expected publishing sessions.

Dependencies among approved assets have a specific treatment and a name ("approval dependencies"), both of which are critical to the understanding of how the approval system works. Approval dependencies are the main subject of the rest of this chapter. They are described and explained in the section "Terms and Definitions," on page 225, along with the equally important terms, "approvals" and "approval states." (A summary of the approval process can be found in the section "Putting It All Together," on page 243.)

Terms and Definitions

Three especially important terms in the WebCenter Sites publishing model are "approval," "approval dependencies," and "approval states." These terms deal with dependencies that are critical to the understanding of how the approval system works. They are also used throughout this guide, as well as the WebCenter Sites publishing interface. We recommend that you familiarize yourself with the terms, their definitions, and the supporting examples. For information about these terms, see the following sections:

- Approval: Intent to Publish vs. Permission to Publish
- Approval Dependencies
- Approval States

Approval: Intent to Publish *vs.* Permission to Publish

For an asset to be published, it requires approval first from the user, then from the approval system. To distinguish one type of approval from the other, we use the terms **user-approval** and **system-approval**.

- The difference between user-approval of assets and system-approval of assets is the following:

  Whereas the user's approval of an asset signals the user's *intent to publish* the asset, the system's approval of the asset grants the publishing system *permission to publish* the asset.

  The user's approval of an asset is equivalent to submitting the asset to the approval system for a dependency test. The system's approval of a user-approved asset is equivalent to (1) confirming that the user-approved assets pass the dependency test, and (2) releasing the asset to the publishing system for publication in the next publishing session.

- The approval system never approves assets before the user has had a chance to do so:

  When we say that an asset is system-approved, it is understood that the user has already approved the asset.

- When a user approves an asset, it is always to a given publishing destination.

  When we say that an asset is user-approved, the publishing destination is understood (and explicitly identified, if necessary).

- System-approvals are conditional:

  When approving assets, the user can assign only one state to them, and that is "Approved." The approval system responds by treating the approval as tentative and requiring verification. Only when the approval system completes its dependency test and confirms that the "Approved" state is valid does the system approve (release) the assets for publication.

  If, however, the approval system cannot confirm the "Approved" state, it rejects the state, assigns its own approval state to the assets, informs the user as to which supporting assets also require approval, and holds the current user-approved assets from publication until the user can approve the supporting assets. The system's approval of a user-approved asset is thus conditional on the user also approving the supporting assets. Conditional approvals are caused by approval dependencies, which are explained in the next section.

- The approval system can unapprove assets:

  The approval system supports an approval queue to handle asset modification events as a way of keeping approval tables up to date. For example, if an asset is approved for publication, but then changed, the approval queue handles this change by unapproving the asset—that is, rejecting it from the publishing queue.

  The approval queue is not accessible to the user. It runs, by default, every 5 minutes. However, when you invoke a feature that uses approval functionality (such as the **Publishing** tab), the approval queue is forced to run before the screen is rendered in order to keep approval information up to date.

Approval Dependencies

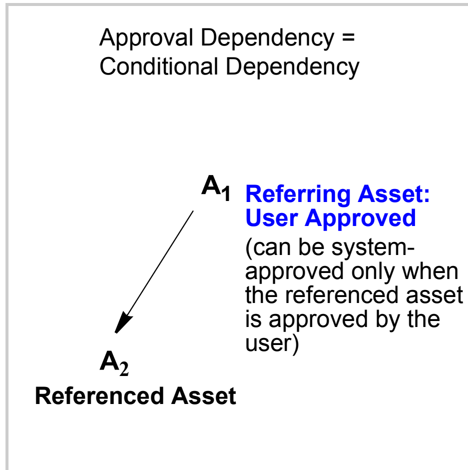
Note

Before reading this section, make sure you understand the distinction between the terms "user-approval" and "system-approval." For more information, refer to the section "Approval: Intent to Publish vs. Permission to Publish," on page 225.

An **approval dependency** is a conditional dependency and results in a conditional approval:

Condition 1. An approval dependency is created if the user approves an asset that refers to another asset for content.

Condition 2. When an approval dependency is created, the system's approval of the referring asset is conditional on the approval of the referenced asset.
In other words, the approval system can approve a referring asset only if the referenced asset is also approved. The referring asset is said to have an approval dependency on the referenced asset.



Note

From this point forward, we define "referring asset" to be the parent asset. The "referenced asset" is the child asset. For simplicity, we also assume that parent assets are solely parents, and child assets are solely child assets.

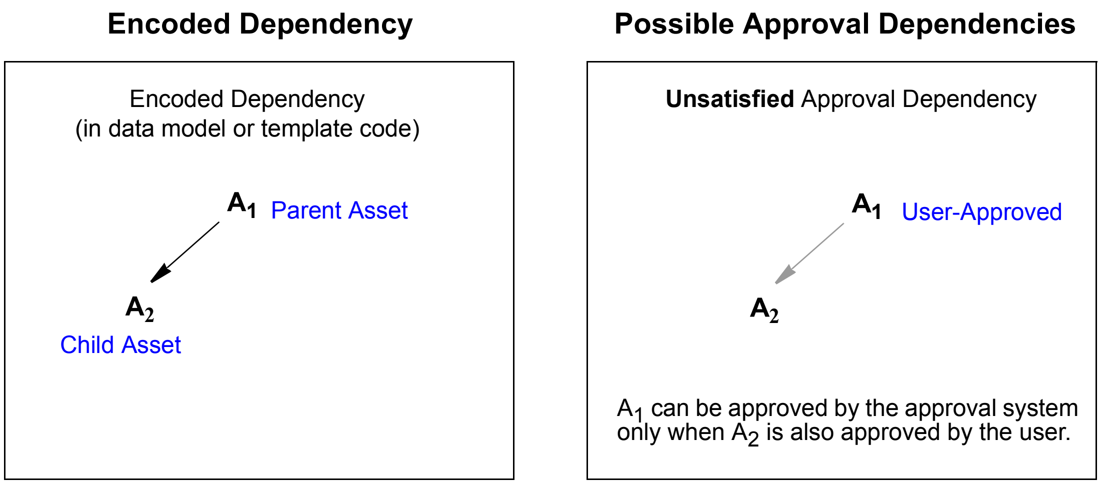
An important implication of approval dependencies is the following:

While approval dependencies originate in one of two sources (the data model or template code), **the number of approval dependencies is not necessarily equal to the number of dependencies in the source.** As shown in Figure 8, the existence of a dependency in the source can lead to an unsatisfied approval dependency, a satisfied approval dependency, or no approval dependency at approval time. Which outcome is observed depends on two factors:

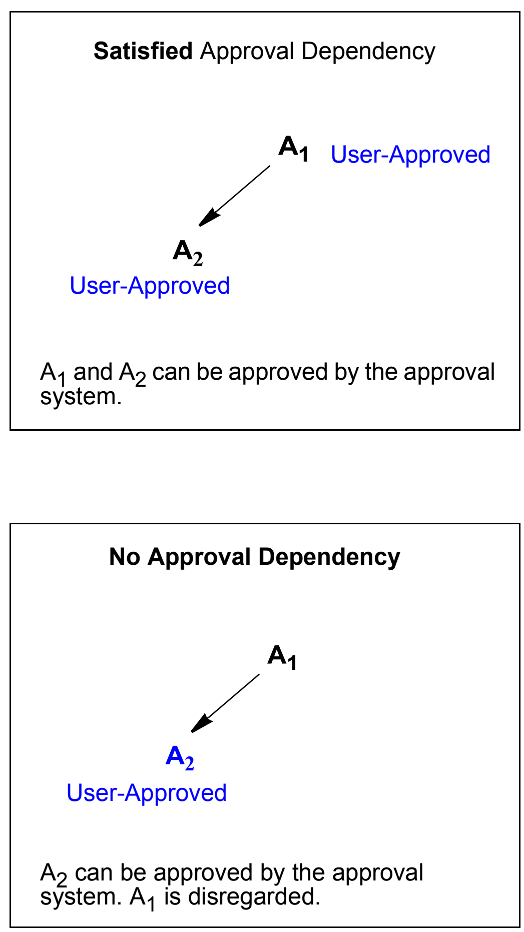
- Which asset(s) the user has approved.
- How the approval system treats approval dependencies. The approval system does not simply determine that a dependency exists for a given pair of assets. It explicitly accounts for the direction of the dependency by testing whether the approved asset is a parent or a child. If it determines that a parent asset has been approved, it logs an approval dependency. If it determines that a child asset has been approved, it does not log an approval dependency.

For detailed examples of how approval dependencies are determined, see page 229.

Figure 8: Encoded Dependencies and Approval Dependencies



For detailed examples of how approval dependencies are determined, see "Examples of Approval Dependencies," on page 229.

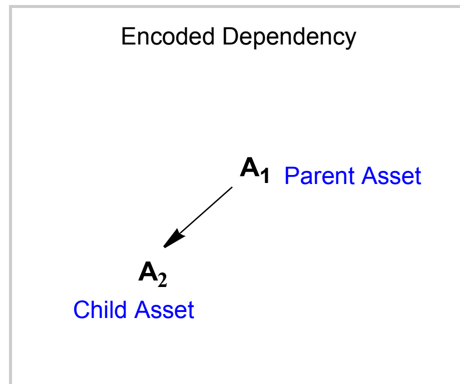


Examples of Approval Dependencies

This example supplements the previous section (especially Figure 8, on page 228), by illustrating in greater depth how approval dependencies are determined.

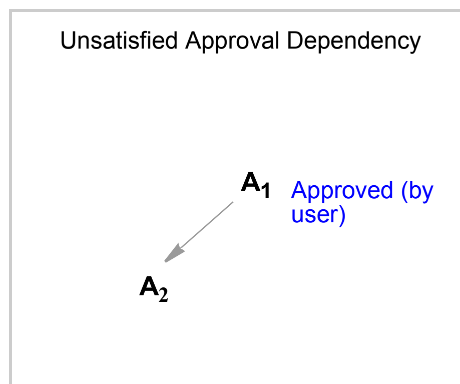
In this example, we have a two-asset data model. Asset A_1 refers to asset A_2 for content. While the data model specifies one dependency ($A_1 \rightarrow A_2$), the user has three different ways to approve the assets and, therefore, create different approval dependencies:

- Unsatisfied approval dependency
- Satisfied approval dependency
- No approval dependency



1. **Unsatisfied approval dependency**

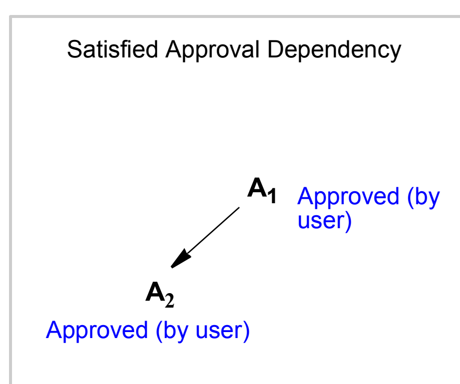
   In this instance, the user approves the parent asset (A_1), but not its child (A_2). The approval system crawls the data model (using the approved asset as a seed), discovers the $A_1 \rightarrow A_2$ dependency, determines the dependency to be unsatisfied (by the user's approval of A_1 alone), classifies the dependency to be an unsatisfied approval dependency, rejects the "Approved" state for A_1, disallows the publication of A_1 until A_2 is also approved, and informs the user that A_2 needs approval.

   In this instance, the system's approval of A_1 is conditional on the user approving A_2. A_1 is said to have an approval dependency on A_2. (Because the approval dependency is unsatisfied, the approval system notifies the user.)

2. **Satisfied approval dependency**

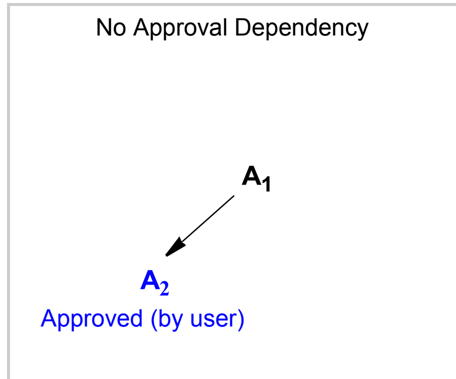
   In this instance, the user approves both the parent asset and its child. The approval system finds the data model dependency to be satisfied (by the user's approval of both A_1 and A_2), classifies the dependency to be a satisfied approval dependency, confirms the "Approved" states, and allows publication of both A_1 and A_2 when the next publishing session begins.

As in the previous instance, the system's approval of A_1 is conditional on the user approving A_2. This time, however, the approval dependency is satisfied. No notice is returned to the user. The asset will be published in the next publishing session.

3. **No approval dependency**

   In this instance, the user approves the child asset (A_2), but not its parent (A_1). Here, the approval system does not recognize an approval dependency (because A_2 has no dependency on A_1). The approval system confirms the user's "Approved" state. As a result, A_2 will be published (but A_1 will not).



Summary

- Whereas data models and template code distinguish between the types of dependencies they specify (for example, associations and recommendation associations), the approval system simplifies these dependencies by treating them as parent-child approval dependencies. The parent is the referring asset and depends on its child for content.

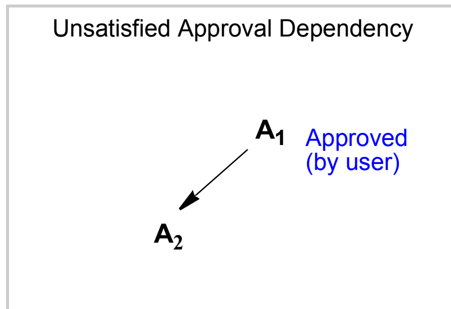
  More information about this rule (and other rules regarding dependency analysis) is given later in this chapter. See "Rules for Dependency Analysis," on page 234.

- Whereas data model dependencies and template dependencies are created by the developer to specify how and which assets refer to each other for content, approval dependencies are created by users who approve assets. Approval dependencies are analyzed by the approval system and used to flag which user-approved assets can be released for publication and which must be held back from publication; which assets have yet to be approved by the user, which assets have been published, and so on.
- Data model and template code dependencies are fixed for the given data model and template. Approval dependencies can vary, even for a fixed set of assets and a given publishing destination. This is so because the combination of assets approved by the varies from one approval attempt to another.

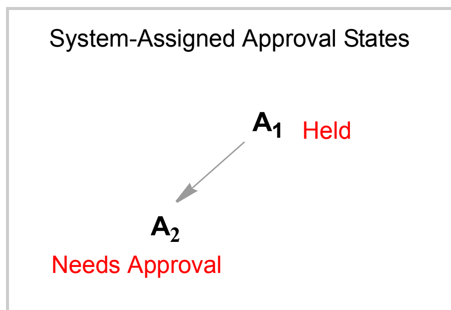
Approval States

When the approval system detects an approval dependency, it must either confirm the user's "Approved" label for each of the assets, or flag the assets with approval states of its own. An **approval state** is a label assigned to user-approved assets by the approval system in order to identify their approval status: why an approved asset cannot be published, whether the asset has been approved and published, whether an asset is eligible for publishing (that is, approved but not yet published because the publishing session has not yet been started), and so on. A complete listing of approval states is given in Table 2, on page 246.

For example, in the figure at the right, the user marked asset A_1 as "Approved." The system, however, treats the approval as tentative. Following a dependency analysis, the system determines that A_1—even though approved— has an unsatisfied approval dependency on A_2 and must be held from publication (until A_2 is also approved).



The system overrides the user's approval of A_1 by assigning A_1 the "Held" approval state. It assigns A_2 the "Needs Approval" state.



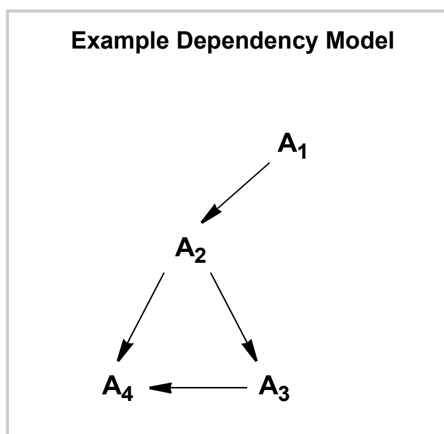
Approval State Scenarios

Even for a fixed set of assets and a given publishing destination, the list of approval states can (and typically does) vary from one approval attempt to another. This is so, because the combination of assets approved by the user tends to vary from one approval attempt to another. To illustrate, we use a simple dependency model and three approval scenarios, all involving one and the same user:

In the figure at the right, each arrow represents an approval dependency. For all four assets to be published, four approval dependencies must be satisfied at publication time:

$A_1 \rightarrow A_2$
$A_2 \rightarrow A_3$
$A_2 \rightarrow A_4$
$A_3 \rightarrow A_4$

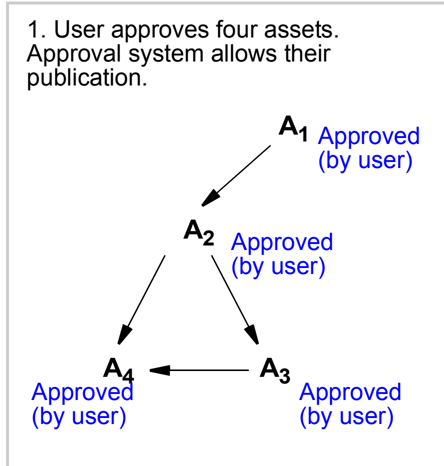
(None of the assets have been previously published.)



Now, consider three scenarios in which the user approves different combinations of assets:

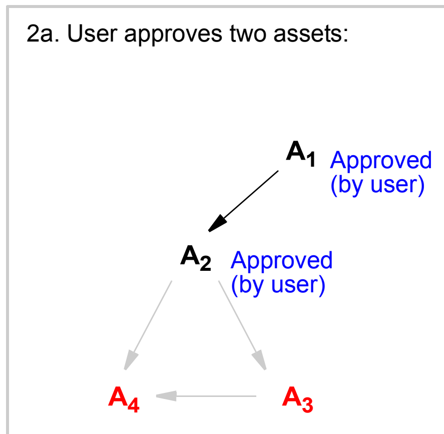
1. In the first (and ideal) attempt, the user approves all four assets, which satisfies all four approval dependencies at once.

In this scenario, the system logs four approval dependencies, determines they are satisfied, confirms the assets as approved, and allows their publication when the next publishing session begins.



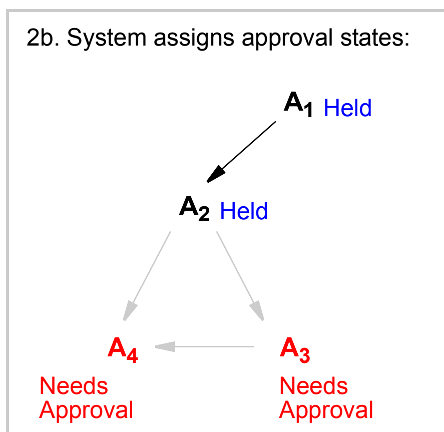
2a. In the second scenario, the user approves two assets—A_1 and A_2—which satisfies one of the four approval dependencies. Three approval dependencies remain to be satisfied:

$A_2 \rightarrow A_3$
$A_2 \rightarrow A_4$
$A_3 \rightarrow A_4$



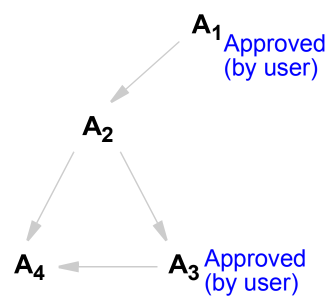
2b. The approval system logs three unsatisfied approval dependencies, determines that none of the assets are ready for publication, assigns its own approval states to all the assets, and returns to the user the following list:

- A_3 and A_4, in the “Needs Approval” state
- A_1 and A_2, in the “Held” state; these assets cannot be published until A_3 and A_4 are approved



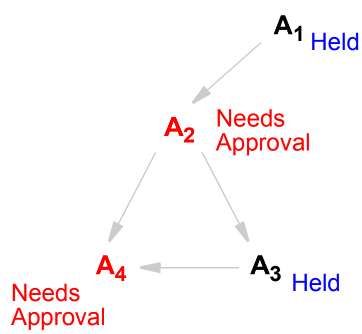
3a. In the third scenario, the user also approves two assets—A_1 and A_3—but this time, satisfies none of the four approval dependencies:

$A_1 \rightarrow A_2$
$A_2 \rightarrow A_3$
$A_2 \rightarrow A_4$
$A_3 \rightarrow A_4$



3b. The approval system logs four unsatisfied approval dependencies, and determines that none of the assets are ready for publication. The approval system returns a list of assets and approval states different from the list in scenario 2. The list identifies:

- A_2 and A_4 in the "Needs Approval" state
- A_1 and A_3 in the "Held" state



Note

If the user had approved a different set of assets, the system would have determined different approval dependencies, and therefore different approval states for the participating assets:

- If the user had approved A_2, the approval system would have determined only three approval dependencies ($A_2 \rightarrow A_3$, $A_2 \rightarrow A_4$, and $A_3 \rightarrow A_4$), even though the data model in our example specifies four dependencies.
- If A_3 were approved, one approval dependency would have been determined ($A_3 \rightarrow A_4$).
- If A_4 were approved, no approval dependencies would have been determined.

For more information as to how approval dependencies are determined, see "Approval Dependencies," on page 227.

Rules for Dependency Analysis

To perform its function, the approval system follows several rules, summarized below and covered in more detail in the sections that follow:

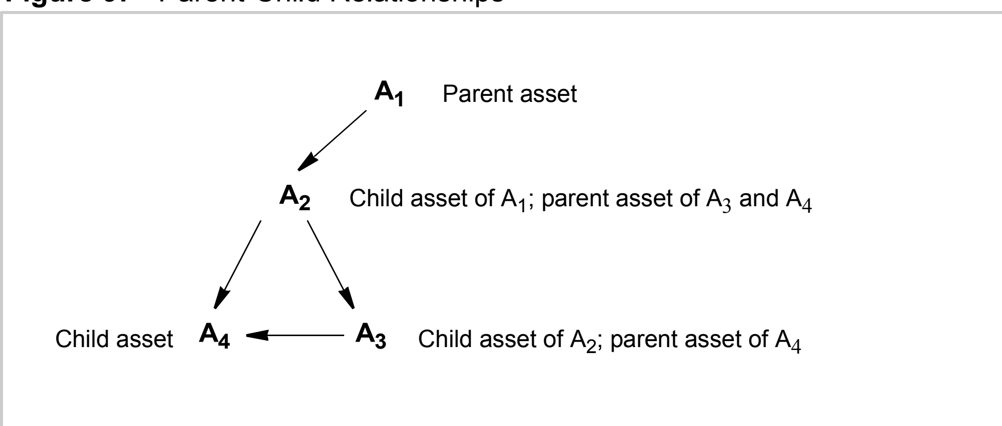
- The approval system simplifies all approval dependencies by treating them as parent-child relationships. For more information, see "Approval Dependencies and Parent-Child Relationships," on page 234.
- In Export to Disk publishing, the approval system tests approval dependencies against template code. For more information, see "Export to Disk Publishing," on page 235.
- In Mirror to Server publishing and Export to XML, the approval system tests approval dependencies against the data model. For more information, see "Mirror to Server Publishing and Export to XML," on page 237.
- If version matching needs to be controlled, the administrator can set an `exists`, `exact`, or `none` dependency to impose or cancel version matching. For more information, see "Ensuring Version-Matched Content," on page 237.
- If the approval system determines the existence of an unsatisfied approval dependency, it searches for published content to avoid redundant approvals. For more information, see "Evaluating Published Content," on page 242.

Approval Dependencies and Parent-Child Relationships

The approval system simplifies the many types of dependencies that are specified in data models and template code (for example, associations and recommendation associations). Regardless of their origin and type, the approval system classifies all *approval dependencies* as parent-child relationships and adheres to the following rules.

- An asset that refers to another asset is always the parent asset; the referenced asset is the child asset. Figure 9 shows an example of parent-child relationships, in which one asset is solely a parent asset, another is solely a child asset, and the rest are a mixture.

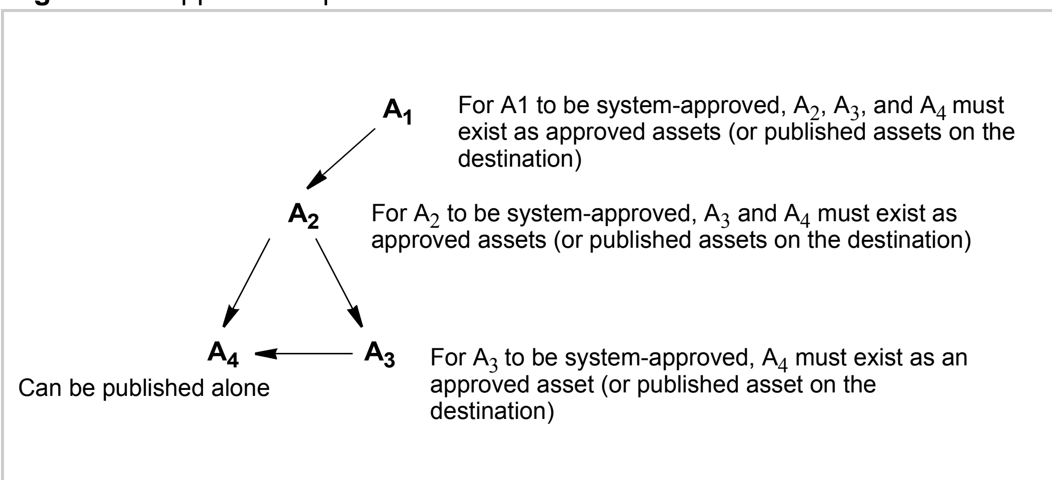
Figure 9: Parent-Child Relationships



- A parent asset is dependent on the child asset for content.
- The approval system approves parent assets only if the child assets are either user-approved or they already exist on the publishing destination.
  - If the child assets are also parents, then their child assets must be approved (or must exist on the destination), and so on.
  - If the publishing method is Mirror to Server or Export Assets to XML, version matching can be a requirement (at the Sites administrator's discretion), in which case the approval system checks assets for version. For more information about version matching, see "Exists/Exact Dependency Rules," on page 239.
- The approval system approves child assets for publication independently of their parent assets. On the live site, the link from parent to child will not appear to be broken; it simply will not appear. (In this respect, parent data can be inadvertently dropped from the live site.)

Figure 10 displays a hierarchical dependency among four assets to illustrate which assets must be approved concurrently (as a result of approval dependencies), and which can be approved alone.

Figure 10: Approval Dependencies



Export to Disk Publishing

For Export to Disk publishing, the approval system tests approval dependencies against template code.

> **Note**
>
> Approval templates represent the developer's approval strategy for Export to Disk publishing. Because approval for Export to Disk publishing is complex, it is given a separate treatment in Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process." The section below introduces the basic concepts.

Evaluating Approval Dependencies

For both flex and basic assets that are published using Export to Disk, there are two types of approval dependencies: template and reference.

Template dependencies are strictly independent of the data model, regardless of how (or whether) the assets are associated in the data model. For example, your data model can specify an article-image association. In Mirror to Server publishing, the article depends on the image for content, and cannot be published without the image. In Export to Disk publishing, however, the template chosen to render the asset defines its own dependencies. Some possibilities are:

- The approval template code re-creates the article-image association as a dependency, thereby creating on the published page a relationship that already exists in the data model.
- The approval template code creates dependencies on yet other assets (such as audio files), thereby creating relationships that do not exist in the data model, but will exist on the published page.
- The approval template code does not create any dependency between the two assets, thereby creating no relationship at all on the published page. The assets are treated as stand-alone content and published independently of each other, even though the data model specifies an association.

Tags that generate template dependencies are:

- `<asset:load>`
- `<asset:loadall>`
- `<assetset:setasset>`
- `<assetset:setlistedassets>`
- `<render:logdep>`

Reference dependencies are generated when a link is created from one page to another. They are registered as reference dependencies between the primary assets of the two pages. For example, if we create a hyperlink from the approval template of asset `A` to a page where asset `B` is the primary asset, the approval system will register this as asset `A`'s reference dependency on `B`. Tags that generate this kind of dependency are:

- `<render:getpageurl>`
- `<render:gettemplateurl>`
- `<render:gettemplateurlparameters>`

Approval dependencies for Export to Disk publishing are a complex topic. Additional information about approval dependencies can be found in Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process."

Test Rendering

When an asset is user-approved to a Export to Disk publishing destination, the approval system determines approval dependencies for that asset by evaluating the code in the template that is assigned to the asset. The approval system also performs a "test-render" in which it evaluates the template code for compositional dependencies, which are manifested when the content is published.

Compositional dependencies are the dependencies of a generated page on the assets that were used to generate that page. They are determined by the logic in that page's template.

The same tags that create template and approval dependencies at approval time also create compositional dependencies at publication time.

If a default template has been assigned to assets of that type, the approval system uses it to determine the dependencies. If there is no default template, the approval system uses the template that is specified for the given asset.

Your developers create default templates for asset types for the following reason: when the Export to Disk publishing method actually publishes an asset, it does not necessarily use the template that is assigned to the asset—the code in another element could determine that a different template renders that asset in certain cases. If this is the case for your online site, it is likely that the developers who created the templates also designed default templates for the approval system to use when determining approval dependencies.

You or your site designers can set default approval templates for each asset type and for each publishing destination. See "Assigning Approval or Preview Templates," on page 324.

Mirror to Server Publishing and Export to XML

For **Mirror to Server** and **Export to XML** publishing methods, the approval system tests approval dependencies against the data model.

Note

Mirror to Server publishing and Export to XML allow for the possibility of unsatisfied **compositional dependencies** on the live site, because templates that render the page are not evaluated during Mirror to Server and Export to XML publishing.

Compositional dependencies appear when a dynamic page is assembled. They involve a set of assets that are used to generate the page. For more information about compositional dependencies, see "Test Rendering," on page 236 and Chapter 13, "Various Topics in Export to Disk Publishing and the Approval Process."

Ensuring Version-Matched Content

Any publishing session (such as the one in Figure 10, on page 235) can be made more stringent by the administrator requiring parent assets to match child assets in terms of version, thereby ensuring self-consistent content. Dependency on version is not encoded in either the data model or the template code:

- For Mirror to Server and Export to XML publishing, it is specified in one of WebCenter Sites' properties—`mwb.conservativedependencies` (in the `gator.ini` file)—by setting the value of the property to `exists` or `exact`.
- For Export to Disk publishing, it is specified by setting the `deptype` attribute in the relevant tags to `exists`, `exact`, or `none` (the tags are listed in "Evaluating Approval Dependencies," on page 236).

Dependency on version is qualified by the terms `exists`, `exact`, or `none`.

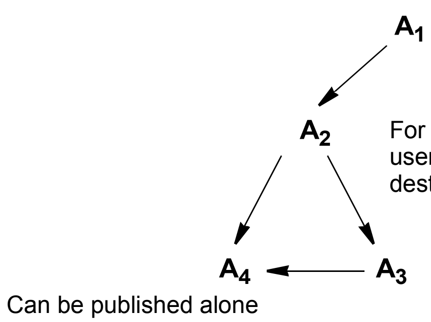
- An **`exists` dependency** does not require version matching. For a parent to be published, its child assets must simply exist either as approved assets or as published assets on the publishing destination. The version of the parent asset *need not* match any versions of its child assets.

- An **`exact` dependency** requires version matching. This dependency is identical to the `exists` dependency, except that the version of the parent must match the versions of its child assets (which means that all assets in the dependency must match each other's version).
- A **`none` dependency** causes the tag in which it is used to specify no *approval* dependency, at all.

Figure 11 summarizes and illustrates `exists` and `exact` dependencies.

Figure 11: Exists/Exact Dependencies

`Exists` dependencies do not require version matching:

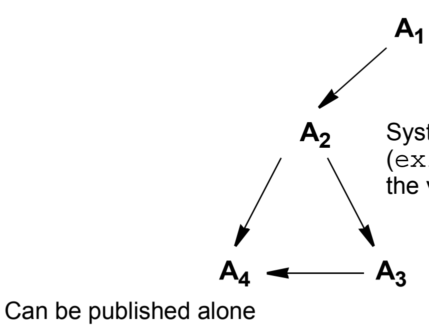


For A_1 to be system-approved, A_2, A_3, and A_4 must exist either as user-approved assets (or as published assets on the target destination)

For A_2 to be system-approved, A_3 and A_4 must exist either as user-approved assets (or as published assets on the target destination)

For A_3 to be system-approved, A_4 must exist either as a user-approved asset (or as a published asset on the target destination)

`Exact` dependencies require version matching:



System-approval of A_1 is the same as in Figure 11 (`exists` dependency, *but* the version of A_1 must match the version of its child asset, A_2

System-approval of A_2 is the same as in Figure 11 (`exists` dependency), *but* the version of A_2 must match the version of its child assets, A_3 and A_4

System-approval of A_3 is the same as in Figure 11 (`exists` dependency), *but* the version of A_3 must match the version of its child asset, A_4

Exists-Exact Dependencies in Mirror to Server and Export to XML Publishing

For Mirror to Server and Export to XMl publishing of flex assets, whether the `exists` or `exact` dependency is in effect depends on the value of the property `mwb.conservativedependencies`, in the `gator.ini` property file.

- The default value (`false`) sets `exists` dependencies.
- A value of `true` sets `exact` dependencies.

> **Note**
>
> When `mwb.conservativedependencies` is set to `false` (exists) and an attribute is in use, the following fields of the attribute are locked and cannot be changed: Value Type, Storage Style, External ID, External Table, and External Column.
>
> If you change the value of `mwb.conservativedependencies`, you must re-approve the assets that were affected by the change.

Exists-Exact Dependencies in Export to Disk Publishing

For Export to Disk publishing, the developer sets `exists`, `exact`, or `none` dependencies using the `deptype` attribute in applicable tags, such as `getpageurl` and `assetload`. (For a listing of the tags, see "Test Rendering," on page 236).

Template dependencies are *by default* `exact` dependencies—if you wish to approve asset A you must approve B if B has been changed. Reference dependencies are *always* `exists` dependencies—if you approved and published B once, you are not required to approve it again in order to re-publish A.

The exception is when you set `deptype="none"` on any of the tags. As a result, no *approval* dependency at all is created by the tag. This means no record is created for the dependency *during approval.* In all other contexts, such as Export to Disk publishing and live sites, the `deptype` attribute is ignored.

Exists/Exact Dependency Rules

When an asset is approved to a Mirror to Server or **Export Assets to XML** destination, the approval system determines approval dependencies against the data model.

> **Note**
>
> When reading this section, bear in mind that in the context of an approval dependency, the term "dependent asset" is equivalent to the "parent asset." For rules governing parent-child relationships in approval dependencies, see "Approval Dependencies and Parent-Child Relationships," on page 234.

Basic Asset Types

For basic assets, the approval system follows these dependency rules:

Relationship to Approved Asset	Dependency	
	Exists	Exact
Association	✓ by default, unless configured to be `exact`	
For page asset only: another page asset at a lower level in site plan	✓	
Embedded link	✓	
Embedded pagelet		✓

- Rules for associations:
  - Depending on how your asset associations are designed, an approved asset has either an `exists` or `exact` dependency on assets that it references through named asset associations.
- Rules for page assets:
  - An approved page asset has an `exists` dependency on page assets that are lower than itself in the hierarchy of the site plan (which is reflected in the `SitePlanTree` table).
- Rules for embedded links:
  - An approved asset has an `exists` dependency on assets that it references by an embedded link. For information about embedded links, see the *Oracle WebCenter Sites Developer's Guide*.
- Rules for embedded pagelets:
  - An approved asset has an `exact` dependency on assets that it references as embedded pagelets. For information about embedded pagelets, see the *Oracle WebCenter Sites Developer's Guide*.

CSElement and SiteEntry Assets

The root element for a SiteEntry asset is represented by a CSElement asset. The following rule applies:

An approved SiteEntry asset has an `exists` dependency on the CSElement that it references.

Flex Families

For flex family members, the approval system follows these dependency rules:

Approved Asset	Approved Asset's Dependencies						
	Flex Parent Definition	Flex Parent	Flex Definition	Flex Attributes	Flex Filter	Attribute Editor	Template
Flex Parent Definition	exists			exists			
Flex Parent	exists	exists		exists			
Flex Definition	exists			exists			
Flex Asset		exists	exists	exists			exists
Flex Attribute					exists (by default)	exists	

- Rules for flex parent definitions:
  - An approved flex parent definition has an `exists` dependency on its flex parent definition(s) and flex attributes.
- Rules for flex parents:
  - An approved flex parent has an `exists` dependency on its flex parent definition, flex parent, and flex attributes.
- Rules for flex definitions:
  - An approved flex definition has an `exists` dependency on its flex parent definition, flex parent, and flex attributes.
- Rules for flex assets:
  - An approved flex asset has an `exists` dependency on its flex parents, flex asset definitions, flex attributes, and template.
- Rules for flex attributes:
  - An approved flex attribute has either an `exists` or an `exact` dependency on its flex filter, depending on how the flex filter is coded. The default flex filter is coded for an `exists` dependency. If the attribute is of type "asset," the user can decide whether there is an `exists` or `exact` dependency on that asset.
  - An approved flex attribute has an `exists` dependency on its attribute editor (if one is assigned).

For information regarding `exists` and `exact` dependencies among flex family members, see "Flex Families," on page 241. For more information about the functions of the flex family members, see chapter 1 of the *Oracle WebCenter Sites Developer's Guide*.

Oracle WebCenter Sites: Engage Visitor Data Assets

The approval system uses the following rules to determine approval dependencies for the Engage visitor data asset types:

Approved Asset	Approved Asset's Dependencies				
	History Definition	History Attribute	Visitor Attribute	Recommendation	Related Flex Asset
History definition		exact			
Segment	exists		exists		
Recommendation					
Promotion				exists	
Flex Asset					exists

- History definitions have `exact` dependencies on their history attributes.
- Segments have `exists` dependencies on the history definitions and visitor attributes used in them.
- Promotions have `exists` dependencies on the recommendation assets that they override.
- Flex assets have `exists` dependencies on any assets that are designated as related items (for Related Items recommendations)

Evaluating Published Content

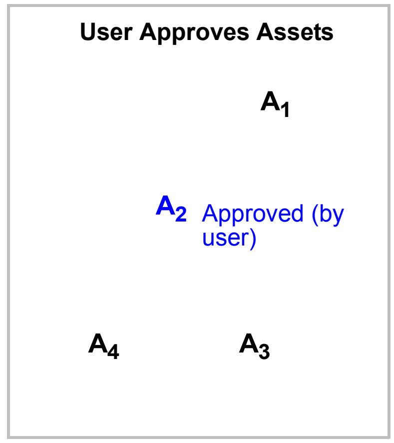
If the approval system determines the existence of an unsatisfied approval dependency, it searches for published content to avoid redundant approvals.

Specifically, the approval system reads the `PublishedAssets` table. If assets currently requiring approval are listed as published to the given destination, the approval system considers it unnecessary to approve (and re-publish) the same assets, assuming version-matching is not a requirement. If an `exact` dependency is specified in the data model or template code, the approval system checks the versions of the published assets. Information about enforcing version-matched content is given on page 237.

Putting It All Together

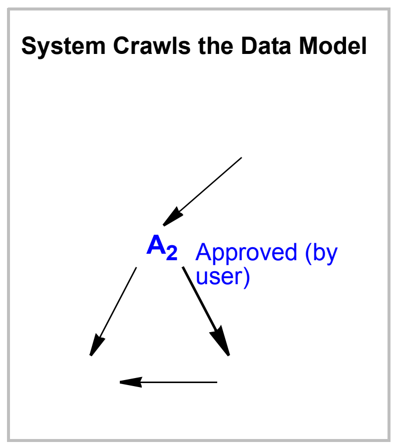
This section contains a scenario that provides a comprehensive illustration as to how the approval system works. The scenario begins with the user approving an asset, continues with the approval system's dependency analysis, and ends with the approval system's response. This scenario touches on all the concepts that were discussed throughout this chapter: approvals, approval dependencies, and approval states, using the Mirror to Server publishing example and an `exists` dependency.

1. In this scenario, a user edits and approves a single asset — A_2 (out of a possible four) — to be published dynamically under an `exists` dependency. Only A_4 has been previously published to the target destination.

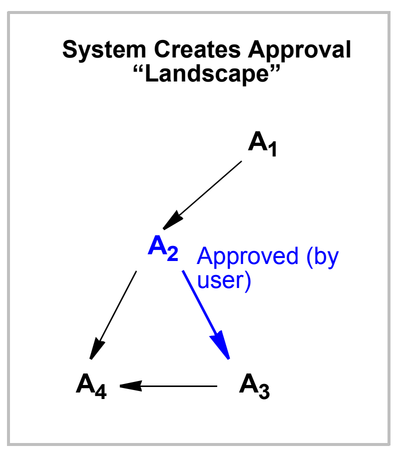


2. When the user attempts to publish, the approval system crawls the data model to determine the encoded dependencies (represented by arrows in the figure to the right).

   Using the approved asset as a seed, the approval system follows the asset's links to and from other assets in the data model, follows the other assets' links to and from yet other assets, and so on until the system determines the boundaries of the seed's dependency "network" and no more dependencies remain to be detected.



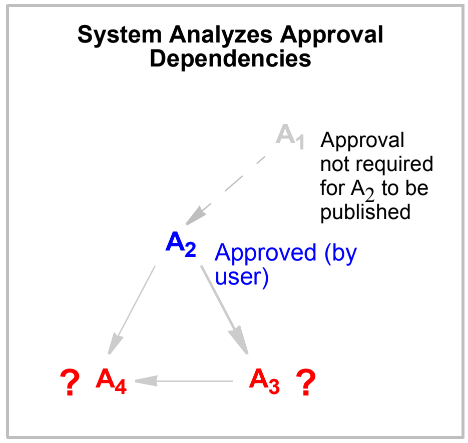
3. From the results of step 2, the approval system constructs an approval "landscape," which identifies all the interdependent assets and relationships among them.



4. The approval system tests for approval dependencies and their impact on the publishing session. In this example, the approval system determines the following:

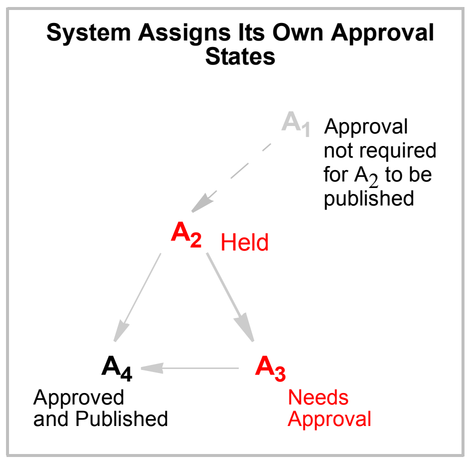
- A_1 does not require approval because the approved asset A_2, as its child, can be published independently. However, A_2 is also a parent asset and cannot be published until the status of its child assets (A_3 and A_4) is determined.

  The approval system now searches for previously published assets. Referring to the `PublishedAsset` table, the approval system determines:



- A_3 has never been published to the target destination. A_3 requires approval to satisfy the $A_2 \rightarrow A_3$ dependency.
- Because A_4 has been previously published to the target destination, its re-approval (and re-publication) is unnecessary. The dependency of A_2 and A_3 on A_4 will be satisfied at publication time.

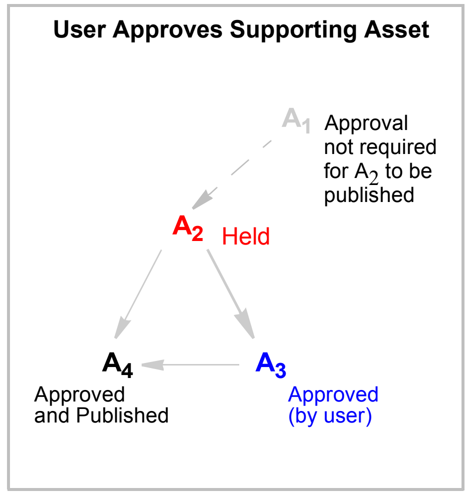
5. From step 4, the approval system concludes that three approval dependencies must be satisfied in order for the approved asset to be published, and, accordingly, assigns its own approval states to the assets:



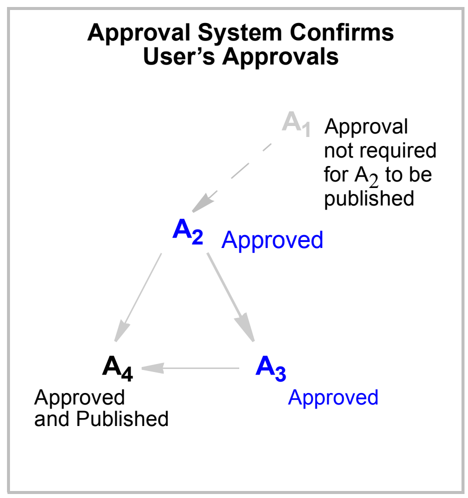
- The $A_2 \rightarrow A_3$ approval dependency needs to be satisfied (by the user approving A_3). The system assigns A_3 the **Needs Approval** state and A_2 the **Held** state.
- $A_3 \rightarrow A_4$ needs to be satisfied (by the user approving A_3, which is already assigned the **Needs Approval** state).
- $A_2 \rightarrow A_4$ is automatically satisfied by the pre-publication of A_4, which is now assigned the **Approved and Published** state.

(A complete list of approval states is available in Table 2, on page 246.)

6. Finally, the approval system displays the list of assets and their approval states (shown in the figure above). In response, the user approves the asset that was assigned the **Needs Approval** state (A_3).



7. To complete the approval process, the approval system updates the approval state of A_2 to **Approved** and releases both A_2 and A_3 to the publishing system for publication in the next publishing session.



Reference: Approval States

Table 2 lists the possible approval states that can be assigned by the approval system to assets in an approval dependency and displayed to the user.

> **Note**
>
> When referring to Table 2, bear in mind that in the context of an approval dependency, the term "dependent asset" is equivalent to the "parent asset." For rules governing parent-child relationships in approval dependencies, see "Approval Dependencies and Parent-Child Relationships," on page 234.

Table 2: Approval States Assigned by the Approval System

Approval State	Description
Approved. Approved and ready to publish to *destination*.	(Informational) This asset will be published at the next publishing event to this destination, unless the asset is changed, or an `exact` dependency changes.
Approved and published. Asset version is the same as that on *destination*.	(Informational) An asset with an `exact` dependency has been published to this destination.
Currently checked out. Will not be published to *destination*.	(Action may be required) The asset is checked out under revision tracking. Although approved, it cannot be published until revision tracking relinquishes control: • Check in – The asset must be re-approved. • Undo Checkout – The asset remains approved and can be published. • Rollback – The asset must be re-approved.
Approved for inclusion as a link in pages exported to *destination*.	(Informational) This asset is approved for Export to Disk publishing, if it is linked to from the page that is being exported.
Asset has been modified since approved for publish to *destination*.	(Action required) The asset must be re-approved.
Approved, but approval for publish to *destination* was based on versions of the child assets that no longer exist.	(Action required) The asset must be re-approved so that its version matches the version of its child assets.

Table 2: Approval States Assigned by the Approval System

Approval State	Description
Held. Approved, but child assets have not been approved for publish to *destination*.	(Action required) The asset is held from a publishing session when any one of the following conditions is true: • The asset was approved, but was then re-edited and has not yet been re-approved. • The asset has a child asset with an `exact` dependency, and that child asset is not approved. • The asset has a child asset with an `exists` dependency and that child asset has never been published. • The asset has a child asset with an `exact` dependency. The child asset has been published, but it has since been edited and is not yet approved. • The asset has an `exact` dependency on a child of another asset that was previously approved, but has since been edited and is not yet re-approved. • The asset is checked out (revision tracking).
Needs Approval. Not yet approved for publish to destination.	(Action required) The asset must be approved.
This asset cannot be published until the assets it references have been approved.	(Action required) A referenced asset must be approved before this asset can be published. Related assets that are held are also listed and may require approval.

Chapter 13

Various Topics in Export to Disk Publishing and the Approval Process

This chapter presents the main concepts and features of Export to Disk publishing, including the approval process as its relates to Export to Disk publishing. When reading this chapter, keep in mind the approval and publishing contexts. While they tend to be discrete—for example, approval templates, pubkeys, and starting points apply to only one context, but not the other—approval and publishing systems depend on each other and therefore interact to some extent.

This chapter contains the following sections:

Export to Disk Publishing Terminology

Compared to mirror publishing, in which dependencies in both the approval and publishing stages are determined by the asset model, Export to Disk publishing is an entirely template-driven process, in both the approval and publishing stages. If the approval template and publishing templates differ with respect to the dependencies they specify—a condition we do not recommend—the likelihood of publishing unapproved content increases.

Coding templates is the responsibility of the developer. However, the WebCenter Sites administrator (and any other users who publish) must understand which templates to invoke and what outcome to expect in order to prevent adverse effects on publishing sessions.

This chapter begins by explaining the most commonly used terms in Export to Disk publishing, then delves into the workings of the approval system. It concludes with a series of examples illustrating how various relationships among assets and templates determine approval and publishing results.

Note

The terminology presented here refers only to Export to Disk publishing. Terms such as **publish key**, **dependencies**, and **references**, have completely different meanings in mirror publishing and are beyond the scope of this document.

Page

In the Export to Disk publishing context, a **page** is typically an HTML file that is exported to a disk. (The file is not to be confused with a Page asset type.) Each exported page is represented by a Publish Key.

Publish Key

A **publish key** (or pubkey) is strictly a publish-time term. A pubkey represents an atomic publishing unit, which in the case of Export to Disk publishing is an exported file (equivalent to a Page, not to be confused with a Page asset type).

A pubkey is defined by two constructs: an asset (called a Primary Asset) and a publish-time template (not to be confused with an Approval Template, because they may be different).

The initial pubkeys in Export to Disk publishing are the Starting Points, defined by the user who initiates the publishing session. All other pubkeys are discovered and logged during publishing.

Publish Queue

A publish queue is a list of pubkeys awaiting publication. For a pubkey to be added to the publish queue, its Primary Asset must be approved.

Primary Asset

The **primary asset** participates in the definition of a pubkey. In layman's terms, its ID and type become the `cid` and `c` parameters for the template. A primary asset must be approved before its page(s) can be exported.

> **Note**
>
> If it is determined that a page's primary asset is not approved, the page will not be exported. This can lead to the creation of a broken link on the publishing destination, depending on which approval and publishing templates were used.
>
> Ultimately, it is the developer's responsibility to code approval templates such that they can validate the dependencies that are expected to exist on the target destination.

Approvals and Export to Disk Publishing

Exporting a page first requires approval of the page's *primary asset.* Approval of the primary asset is contingent on the approval states of its dependent assets, which are dictated by the primary asset's **approval template**. The dependent assets' dependencies are further dictated by their own approval templates.

> **Note**
>
> Because the approval template is not necessarily the one used to render the asset, it is possible to publish assets without passing them through the approval system. This implication and others are discussed and illustrated in "Behavior and Functionality," on page 255 and "Approval and Publish-Time Templates: What Happens When They Differ?," on page 257.
>
> Also note that compositional dependencies (involving non-primary assets of a page, as explained on page 253) need not be approved in order to appear on an exported page (to filter out unapproved assets, see the `<render:filter>` tag in the *Developer's Tag Reference* for more information).

Approval Template

The purpose of approval in Export to Disk publishing is to allow the developer to define dependencies in a way as to ensure that the correct content is published and it is published intact.

Compared to mirror publishing, in which asset dependencies in both the approval stage and publishing stage are determined by the asset model, Export to Disk publishing is entirely template-oriented. As a result, approval behavior (as well as publish-time behavior) is very much defined by the developer who writes the approval templates (as well as the publish-time templates). WebCenter Sites then uses the approval template to discover an asset's dependencies. If an asset is published against the approval template, its approval dependencies are likely to be the same as its Compositional Dependencies.

Approval templates are assigned by use of the "Set Default Templates" feature in the publishing destination screen.

- If a default template is not explicitly chosen, the approval system will choose the asset's default template.
- If neither the approval nor default template is specified, the asset will be approved without an approval template (i.e., it will have no approval dependencies).

If the approval template is shared among sites, the approval system will choose the site entry corresponding to the current site.

Approval Queue

The approval queue handles asset modification events to keep approval tables up to date. For example, if an asset is approved for publication, but you then change it, the approval queue will handle this by unapproving the asset—rejecting it from the publish queue.

You never work directly with the approval queue. The queue runs every 5 minutes by default. However, if you invoke a feature that uses approval functionality (such as the **Publishing** tab), the approval queue is forced to run before the screen is rendered to keep approval information up to date.

Approval Dependencies

When you approve an asset, its Approval Template is used to determine its approval dependencies. The following tags create approval dependencies:

- `<asset:load>`
- `<asset:loadall>`
- `<assetset:setasset>`
- `<assetset:setlistedassets>`
- `<render:logdep>`
- `<render:getpageurl>`
- `<render:gettemplateurl>`
- `<render:gettemplateurlparameters>`

If you approve asset `A` whose template (or any element called by the template) references asset `B` using one of the tags above, an approval dependency is created from `A` to `B`. This generally means that when you want to approve `A`, you must also approve `B`.

Types of Approval Dependencies

Approval dependencies are created and used at approval-time, not to be confused with publish-time dependencies (even though they can be the same). Approval for Export to Disk publishing involves two types of approval dependencies:

- Template dependencies
- Reference dependencies

Template dependencies are created when an asset's approval template uses another asset to define the content. For example, if asset `A`'s approval template loads asset `B`, then `A` has a template dependency on `B`. In more practical terms, if you have an approval template for a Page asset that shows an Article asset, the Article asset is used on the Page, so the dependency is of a template kind. The following tags generate template dependencies:

- `<asset:load>`
- `<asset:loadall>`
- `<assetset:setasset>`

- `<assetset:setlistedassets>`
- `<render:logdep>`

Reference dependencies are generated when a link is created from one page to another. They are registered as reference dependencies between the primary assets of the two pages. For example, if we create a hyperlink from the approval template of asset `A` to a page where asset `B` is the primary asset, the approval system will register this as asset `A`'s reference dependency on `B`. Tags that generate this kind of dependency are:

- `<render:getpageurl>`
- `<render:gettemplateurl>`
- `<render:gettemplateurlparameters>`

Template dependencies (in Export to Disk publishing) are *by default* `exact` dependencies — if you want to approve `A` you must approve `B` if `B` has been changed. Reference dependencies are *always* `exists` dependencies—if you approved and published `B` once, you are not required to approve it again in order to re-publish `A`.

The exception is when you set `deptype="none"` on any of the tags. In that case, no *approval* dependency at all is created by that tag. This means no record is created for it *during approval;* in all other contexts, such as Export to Disk publishing and live sites, the `deptype` attribute is ignored.

Export to Disk Publishing

Three main components play a role in the Export to Disk publishing process:

- Publish-Time Template
- Starting Points
- Compositional Dependencies

Publish-Time Template

The purpose of the publish-time template is to render content as files. Typically, publish-time templates are identical to the approval templates. However, when they differ, content can be published without first being approved. Bypassing the approval system requires the developer to provide a means by which publishable content can be validated.

Starting Points

The **starting point** is the pubkey (or pubkeys) where export begins, which is at publish-time. Typically, starting points are selected to link to most, if not all the pages on your site. (The page may not necessarily be the home page; for example, it could be a sidebar with many links. The choice is yours.) The Export to Disk publishing system will crawl your site, beginning with the starting points, and will keep logging new pubkeys as it discovers them.

Compositional Dependencies

Compositional dependencies are the dependencies of a generated page on the assets that were used to generate that page. They are determined by the logic in that page's Publish-Time Template. Compositional dependencies dictate what is rendered on the exported page as prescribed by the template, completely ignoring any `deptype` attributes. The same tags that created template and reference dependencies in the approval template also create compositional dependencies at publish-time. The tags are listed in "Approval Dependencies," on page 252.

Frequently Asked Questions

How Do I Select an Approval Template?

The approval template (if specified) is the only template ever used by the approval system to validate a given asset. This template may or may not be used at publish-time, depending on what the starting point is set to. The best advice is to set approval templates that most closely represent the asset's intended dependencies.

What if the template that contains the most representative set of dependencies is not the template that you want to publish the asset with? Set it as the approval template for assets of that type, and use any desired template(s) as starting point(s).

Are Data Model Dependencies Accounted For in Any Way?

Associations, attributes, and so on, are not used in Export to Disk publishing. The only dependencies that matter are those established by templates.

Why Do We Track Publish-Time Compositional Dependencies?

After you export a page, its content is frozen. However, the assets used to generate that page may evolve, making the affected pages obsolete.

Because it records dependencies, WebCenter Sites is able to remind you which pages need updating, assuming you will want to republish those pages. WebCenter Sites gives you two republishing options: automatic refresh and manual refresh.

- With automatic refresh, pages are queued automatically for the next publishing session, without your having to approving compositional dependencies that have changed.

  If you choose this option, you will not have to approve the modified assets in order for the page to be queued for publishing; in other words, the article page will be republished as soon as you change the image asset. To set up automatic refresh, specify `exists` or no dependencies between the article and the image; this disables the approval system from prompting you to approve the assets.

- With manual refresh, you review the updated pages before publishing them.

  If you choose this option, the affected pages will be `Held` until approved; in other words, you must reapprove the article page every time the image changes. For this to work, you need to set up `exact` dependencies.

Behavior and Functionality

Following is a compilation of simple scenarios which demonstrate Export to Disk publishing behavior:

- Example 1: Template Dependencies
- Example 2: Reference Dependencies
- Approval and Publish-Time Templates: What Happens When They Differ?

The participating asset types are Page and HelloArticle (from HelloAssetWorld). The templates are `Ttemplate`, `RTemplate`, and `dummyTemplate`.

For clarity, the asset IDs have been hardcoded into the templates. Normally, assets would be loaded somehow instead of being hardcoded, so developers would need to leverage the `deptype='none'` attribute in their tags if they do not wish to set up additional dependencies. See "Sample Templates," on page 259 for the template code.

Example 1: Template Dependencies

Setup: Page `P` asset uses `Ttemplate` for both approval and publishing. `Ttemplate` loads HelloArticle `A`, therefore establishing a template dependency.

Starting Point: `P+Ttemplate`

Pubkeys (exported files): `P+Ttemplate`

Approval Dependencies: Page `P` has template dependencies on `Ttemplate` because of `<render:logdep>`, and on HelloArticle `A` because of `<asset:load>`. Because `Ttemplate` does not create any links, there are no reference dependencies.

Publish Dependencies: Starting Point is the only pubkey; therefore only one file will be exported.

Note that the exported page does not have any publish-time compositional dependencies because it is never actually used by the template. There is only reason why `P` had to be approved—`P` is the exported page's primary asset. So, the only compositional dependencies determined at publish-time are the exported page's dependencies on `A` and `Ttemplate`.

Action	Approval Status	Publish Status	Comments
Initial	Approved: `P`, `A`, `Ttemplate`	Published: `A` and `Ttemplate`	`P` was **not** published because it was not referenced by the template.
Edit `P`	`P` status becomes `Changed`	Not affected by change.	Since there are no compositional dependencies on `P`, the change had no effect.

Action	Approval Status	Publish Status	Comments
Edit A	P status becomes Held. A status becomes Changed – need to approve P again.	After approval, page is published again	Template dependency on A causes P to be held, and subsequent approval makes the page publishable again.

Example 2: Reference Dependencies

Setup: Page P asset uses Rtemplate for both approval and publishing. Rtemplate references HelloArticle A, which is in turn rendered by dummyTemplate.

Starting Point: P+Rtemplate

Pubkeys (exported files): P+Rtemplate, A+dummyTemplate

Approval Dependencies: Page P has the routine <render:logdep> template dependency on its Rtemplate. In addition, it establishes a reference dependency on A because of <render:getpageurl> (but **not** on dummyTemplate, because dependencies between pages are logged only between their primary assets). Note that dummyTemplate does not participate in an approval dependency because it is not referenced by any tags.

Publish Dependencies: The relevant tags are <render:logdep> and <render:getpageurl>. These dictate the publish-time compositional dependencies as Rtemplate and A. As before, note that Page P does not have a compositional dependency because it is not used by any of the templates. When export encounters <render:getpageurl>, it detects that another page is involved, so it creates the A+dummyTemplate pubkey on-the-fly, and runs dummyTemplate to generate its contents. Note that dummyTemplate does not officially have a compositional dependency because it is not used by any tags—it participates only as a pubkey member; so, if dummyTemplate is changed, the approval/publishing cycle will not recognize the change. For this reason, we advise not altering <render:logdep>, which appears on every newly created template.

Action	Approval Status	Publish Status	Comments
Initial	Approved: P, A, Rtemplate	Published: A, Rtemplate	P was **not** published because it was not referenced by any template. The result is two HTML files: P+Ttemplate and A+dummyTemplate
Edit P	P status becomes Changed – approve it to go back to status quo	Not affected by change.	Since there are no compositional dependencies on P, the change had no effect.

Action	Approval Status	Publish Status	Comments
Edit A	A status becomes Changed; P is still Approved, **not** Held because it has only an exists dependency on A (no reason to hold P).	P+Rtemplate was automatically queued for re-publishing because <render:getpageurl> generated a compositional dependency. (If you are not sure as to why, keep reading); A+dummyTemplate was not affected because it does not use A.	A+dummyTemplate was not affected because it does not have tags that use A.

Approval and Publish-Time Templates: What Happens When They Differ?

Generally, using the same template for approval and for publishing does not create problems, because approval dependencies match compositional dependencies. Problems start to occur when you use a different template for approval than for publishing. This can happen in several situations. (For template code, see "Sample Templates," on page 259.)

Example 1

Asset A uses T1 for approval. T1 does not reference any assets.
Asset A uses T2 for publishing. T2 references asset B using <asset:load>.
T2 loads asset B, which was never approved, but B will nevertheless be on the exported A+T2 page!

Does this mean you can publish assets that were never approved?

Yes! Some users prefer this, only to avoid approving assets before publishing. Approval can be a taxing operation because exact dependencies cause primary assets to be given Held status if the dependent assets change. As a result, pages of those primary assets cannot be exported. However, a better way to handle such situations is to make good use of the deptype attribute with exists or none values to avoid unwanted approval dependencies.

Now, what happens after asset B is published and then we change B? In this case, there is no approval dependency between asset A and asset B, but there is a compositional dependency. This means that whenever asset B is changed, the affected pages will be automatically re-published. (Because there are no dependencies, the automatic refresh option is chosen). So, A+T2 will be placed in the publish queue automatically as soon as asset B is saved—not approved. (For information about republication options, see "Why Do We Track Publish-Time Compositional Dependencies?," on page 254.)

Example 2

Asset A uses T1 for approval. T1 references asset B using <asset:load> under an exact dependency. Asset A uses T2 for publishing. T2 does not reference asset B.

In this case, there is an approval dependency between asset A and asset B, but there is no compositional dependency. This means that whenever asset B is changed, asset B has to be approved before asset A can be approved, even though the published page does not make any use of asset B.

Example 3

Asset A uses T1 for approval. T1 does not reference any assets.
Asset A uses T2 for publishing. T2 references asset B using <asset:getpageurl>.

In this case, there are no approval dependencies, so A is the only approved asset. During publishing of A+T2, we realize there is another page to be created, with B as primary asset. Now, remember that **primary assets must be approved before their page is exported**. However, B is not approved, so its page will not be exported. A+T2 is exported, but has a broken link.

Unknown dependencies and Export to Disk publishing. Situations become complicated when you start using <render:unknowndeps>. This tag removes compositional dependencies, and it records unknown dependencies to the pubkey. Assets thus remain in the publish queue permanently (meaning they are refreshed with every publish cycle). This behavior is expected, because if unknown dependencies exist (that is, dependencies may change at any moment, such as query results), the system cannot determine which assets a given asset depends on. To be on the safe side, the system always republishes the assets.

Many `exact` dependencies. While this is not always avoidable, keep in mind that when a primary asset has exact dependencies on other assets, a change to any of the dependent assets will cause the approval system to change the primary asset's status to Held. The primary asset's page is not publishable unless approved again. If the modified asset is used by many pages, they will all need re-approval. You may consider setting the deptype to exists, and make the dependent pages automatically publishable—but the modified asset must be ready when the publishing session begins.

Note

The preferred way to customize export URLs (without using simplename) is to use PREFERREDFILE and PREFERREDDIR as parameters to <render:getbloburl> and <render:getpageurl> to specify arbitrary names. For example:

```
 <render:getbloburl
         blobtable='MungoBlobs' blobcol='urldata'
         blobkey='id' blobwhere='1088466917821'
         outstr='pagelogoURL' csblobid='1088466917821'>
   <render:argument name='PREFERREDFILE'
         value='myBlob.out'/>
   <render:argument name='PREFERREDDIR' value='myDir'/>
</render:getbloburl>
```

This code exports the blob whose id is 1088466917821 to myDir/myBlob.out.

Sample Templates

This section lists the template implementations used in the examples on pages 255 and 257. Taglib and import definitions have been skipped.

Ttemplate:

```
<%-- Record dependencies for the Template --%>
<ics:if
   condition='<%=ics.GetVar("tid")!=null%>'><ics:then><render:l
   ogdep cid='<%=ics.GetVar("tid")%>' c="Template"/></
   ics:then></ics:if>
<asset:load name='myArticle' type='HelloArticle'
   objectid='1156878442427'/>
```

Rtemplate:

```
<%-- Record dependencies for the Template --%>
<ics:if
   condition='<%=ics.GetVar("tid")!=null%>'><ics:then><render:l
   ogdep cid='<%=ics.GetVar("tid")%>' c="Template"/></
   ics:then></ics:if>
<render:getpageurl outstr="myURL" pagename='HelloAssetWorld/
   dummyTemplate' cid='1156878442427' c='HelloArticle'/>
Got URL: <a href='<%=ics.GetVar("myURL")%>'> Click here</a><br/
   >
```

dummyTemplate:

completely blank, no `logDep`.

Chapter 14

The Export to Disk Publishing Process

This chapter explains how Export to Disk publishing works and how it is configured.

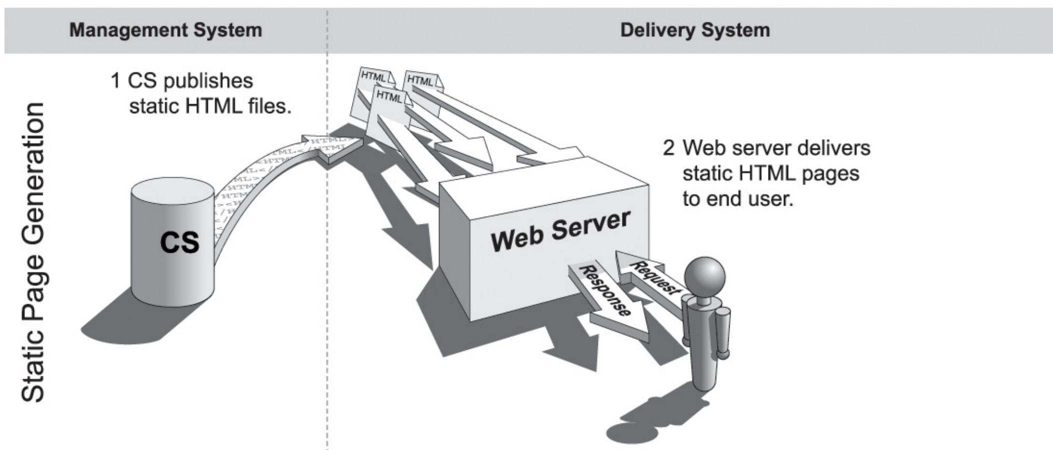
This chapter contains the following sections:

Rendering Export to Disk Pages

In extremely simple terms, WebCenter Sites renders a Export to Disk page as follows:

1. A page name is submitted by the client browser to WebCenter Sites (through HTTP or another protocol).
2. WebCenter Sites locates the page in the `SiteCatalog` table, invokes the root element, and renders the page to an `html` file.
3. You publish the file (either through FTP or another file transfer protocol) to the web server that is hosting your online site.

Typically, administrative users of the Export to Disk publishing method set up a quality assurance process to test the rendered files before moving them to the web server of the delivery system.



How Export to Disk Publishing Works

When content is published using Export to Disk, the approval system, the publishing schedule, and your destination configurations all contribute to the process of exporting your approved assets into static HTML files.

When a Export to Disk publishing session runs, this is what happens:

1. The publishing system notifies the CacheManager servlet that a session is about to begin for a specific destination. CacheManager clears all the pages that were previously exported to this destination from the page cache on the source system.

   During an export publishing session, exported files are cached to the page cache. If publishing sessions occur more frequently than the cache is configured to be cleared (expiration time), there could be exported pages in the cache that have the same names as the newly exported pages. To ensure that old files are not used, the CacheManager servlet clears the page cache of any pages that were exported to during previous publishing sessions before a new publishing session starts.
2. WebCenter Sites creates an export queue and adds all references that can be published to it. A reference that can be published is one of the following:

- A pubkey that has been designated as an **export starting point**. (See "Export Starting Points," on page 272.)

  There must be a starting point for the site and its asset must be approved, or the export session cannot begin. Every site that is exported must have at least one export starting point designated for it.
- Any previously exported page or pagelet (reference) whose assets have changed and been approved again since the last publishing session to this destination. (After a site has been published, the publishing system knows which references have been published. It reads the data in the `PublishedAssets` table and then adds any references whose assets have been changed and approved again for this destination to the export queue.)
- Any asset whose template uses `RENDER.UNKNOWNDEPS` tag.

  This tag alerts the approval system that the dependencies for the asset being rendered by the template cannot be calculated because they are unknown. It is typically used for queries. When the dependencies are unknown, the system cannot determine whether changed dependents would require the asset to be published. Therefore, the presence of this tag means that the asset must be published during every publishing session.
- Any page or pagelet (reference) that has not yet been published and is connected to another page or pagelet through a `RENDER.GETPAGEURL` tag. (That is, it is referenced from another page and has not yet been published.)
- Any page or pagelet (reference) identified by a `RENDER.SATELLITEPAGE` or `satellite.page` tag.

3. For the first export starting point, WebCenter Sites determines whether its asset has been approved:
   - If the asset has not been approved, the publishing session ends.
   - If the asset is approved, WebCenter Sites determines the page name of the template assigned to the asset.
4. WebCenter Sites renders the export starting point by passing the following information to WebCenter Sites:
   - The page name of the template for the starting point.
   - The `rendermode` variable set to `export[`*`destinationID`*`]`.
   - The name and location of the export directory where the rendered files should be saved.
5. WebCenter Sites invokes the root element of the page name, and begins rendering every approved asset that is connected to that export starting point as a reference. WebCenter Sites writes the resulting rendered pages to files rather than posting them to the browser.
6. After each file is rendered, WebCenter Sites writes a message about that reference to the publish log for the session.
7. If there is more than one export starting point, the process cycles back to step 3 in this description.
8. When the publishing session is successful, WebCenter Sites concludes the session by writing information about the published references to the `PubKey` and `PublishedAssets` tables.

9. WebCenter Sites also writes information about the assets that were published to the `ApprovedAssets` and `ApprovedAssetDeps` tables. It logs the date from the assets' updated field at the time that the assets were published so the approval system can calculate dependencies correctly the next time an asset is approved.

Path Naming Convention

How WebCenter Sites names paths to exported files depends on the type of asset that is published and path information that might have been set for the asset. The export path can be as simple as a single directory or it can be a path of several directories, limited only by the number of characters that are supported by the operating system.

The export path naming convention is:

```
<cs.pgexportfolder>/<DIR>/<file_dir>
```

where `<cs.pgexportfolder>` is a required directory. The subdirectories are conditional. Whether `<file_dir>` is used depends on a publishing argument named `SIMPLEDIR`.

Before summarizing the possible export paths, we define the export path variables and the `SIMPLEDIR` publishing argument.

Export Path Variables and SIMPLEDIR

Export path variables are defined as follows:

- `<cs.pgexportfolder>` defines the root directory for all exported files. `<cs.pgexportfolder>` is the value of the `cs.pgexportfolder` property in the `futuretense.ini` file. It is a required value and applies to all Export to Disk publishing destinations.

  Typically, `<cs.pgexportfolder>` names a testing location (a file system where the exported files are verified) rather than a location directly on the delivery system.

- `<DIR>` is the value of the publishing argument `DIR`, which is used to create subdirectories of the root directory (specified by `cs.pageportfolder`). When `<DIR>` is specified, WebCenter Sites creates `<cs.pgexportfolder>/<DIR>`.

  Note

  The `DIR` publishing argument is available to administrators as the "Base Directory" field in the "Add New Destination" screen (page 276). Throughout this section, we use `<DIR>` to mean "Base Directory."

  This argument is typically used to organize the contents of the root export directory when multiple export destinations exist for your sites.

  Note

  The directory specified by `<cs.pgexportfolder>` or `<cs.pgexportfolder>/<DIR>` is called the **destination directory**.

- `<file_dir>` takes one of the values listed below, depending on whether the exported asset is a page asset and whether path information is available. `<file_dir>` is determined hierarchically, as WebCenter Sites looks for the values listed below, in the order shown. It is assumed that `SIMPLEDIR=false`.

Possible <file_dir> Values	Description
`<ForDestination:Path>`	Value of the **For Destination: Path** field, shown on page 282, step 8 (the field is displayed in the "Specify Path/Filename *for destination*" form for a specific destination). Setting a value in the **ForDestination: Path** field forces the asset to take the specified path when the asset is exported to the specified destination. If `<ForDestination:Path>` is not set, WebCenter Sites uses `<parent_page's_path>`
`<parent_page's_path>`	Value of the **Path** field for the exported asset's ***parent*** page. **Note for Page Assets:** When exported, a page asset takes the path that is set for its parent page, not the one set for itself. (As a rule, a page asset's **Path** is applied only to child assets.) If `<parent_page's_path>` is not set, WebCenter Sites uses `<ID_of_parent_page>`.
`<asset's_path>`	Value of the exported asset's **Path** field. **Note:** `<asset's_path>` is used only for assets that are not page assets. • If `<asset's_path>` is not set, WebCenter Sites uses `<parent_page's_path>` as the value of `<file_dir>`. • If both `<parent_page's_path>` and `<asset's_path>` are not set, WebCenter Sites uses <ID_of_parent_page>.
`<ID_of_parent_page>`	ID of the asset's parent page. This value is used if `<parent_page's_path>` and `<asset's_path>` are unknown.

- **`SIMPLEDIR`**. Although this publishing argument is not explicitly part of the export path, it determines whether `<file_dir>` is used when path information is unavailable. That is:
    - If `SIMPLEDIR` is set to `true` ***and*** path information is unavailable, `<file_dir>` is omitted from the export path. Setting a path overrides `SIMPLEDIR`.

- If SIMPLEDIR is set to false ***and*** path information is unavailable, <file_dir> takes <ID_of_parent_page> as its value.

Note

The SIMPLEDIR publishing argument is available to administrators as the "Use simple directory naming" checkbox in the "Add New Destination" screen (see page 276).

Export Path Construction

This section summarizes the export paths that WebCenter Sites can construct using the convention:

<cs.pgexportfolder>/<DIR>/**<file_dir>** [1]

as explained in "Path Naming Convention," on page 264. (Note that only <file_dir> varies in all the examples below).

Forced Export Path

When <ForDestination:Path> (defined on page 265) is specified, the asset being published to the named destination is forced to take the specified path, even if other path information is set. The asset's full export path is:

<cs.pgexportfolder>/<DIR>/**<ForDestination:Path>** [2]

Export Path for Page Assets

If <ForDestination:Path> is not set, WebCenter Sites constructs the export path as shown below (for examples, see Table 3, on page 268).

- When <parent_page's_path> is specified, the page asset's export path is:

  <cs.pgexportfolder>/<DIR>/**<parent_page's_path>** [3]

  (even if <asset's_path> is specified).

Reminder

When exported, a page asset takes the path that is set for its parent page, not the one set for itself (that is, <file_dir> takes the value <parent_page's_path>).

As a rule, a page asset's **Path** information is applied only to child assets.

- When <parent_page's_path> is not specified:
  - If SIMPLEDIR=false, WebCenter Sites uses <ID_of_parent_page>:

    <cs.pgexportfolder>/<DIR>/**<ID_of_parent_page>** [4]
  - If SIMPLEDIR=true, WebCenter Sites omits <file_dir>:

    <cs.pgexportfolder>/<DIR> [5]

Export Path for Assets Other than Page

For an asset that is not a page, WebCenter Sites constructs one of the following export paths depending on which information is available. For examples, see Table 4, on page 268.

- When <asset’s_path> is specified, the export path is:

  <cs.pgexportfolder>/<DIR>/**<asset’s_path>** [6]

- When <asset’s_path> is not specified, WebCenter Sites uses <parent_page’s_path>:

  <cs.pgexportfolder>/<DIR>/**<parent_page’s_path>** [7]

- When both <asset’s_path> and <parent_page’s_path> are unspecified:
  - If SIMPLEDIR=false, WebCenter Sites uses <ID_of_parent_page>:

    <cs.pgexportfolder>/<DIR>/**<ID_of_parent_page>** [8]

  - If SIMPLEDIR=true, WebCenter Sites omits <file_dir>:

    <cs.pgexportfolder>/<DIR> [9]

Sample Export Paths

Table 3: Sample Export Paths for Page Assets when `SIMPLEDIR=false`

Page Asset	<cs.pg exportfolder>	<DIR>	<file_dir>			Export path
			<asset's _path>	<parent_page's _path>	<ID_of_parent _page>	
Home	/export		/abstracts			/export
Home	/export	/Japan	/abstracts			/export /Japan
World News	/export	/Japan	/abstracts	/news/	998877665	export/Japan/news
World News	/export		/abstracts	/news/	998877665	/export/news
World News	/export	/Japan	/abstracts		998877665	/export/Japan/ 998877665
World News	/export		/abstracts		998877665	/export/998877665

Table 4: Sample Export Paths for Assets Other Than Page when `SIMPLEDIR=false`

Not Page Asset	<cs.pg exportfolder>	<DIR>	<file_dir>			Export path
			<asset's _path>	<parent_page's _path>	<ID_of_parent _page>	
Oil Price	/export	/Japan	/energy		997766554	/export/Japan/ energy
Oil Price	/export	/Japan	/energy	/news/	997766554	/export/Japan/ energy
Oil Price	/export		/energy	/news/	997766554	/export/energy
Oil Price	/export	/Japan		/news/	997766554	/export/Japan/ news
Oil Price	/export	/Japan			997766554	/export/Japan/ 997766554
Oil Price	/export				997766554	/export/997766554

Paths for Links Within Exported Files

URLs in links for HREFs within the exported files do not include the values that specify the destination directory (<cs.pgexportfolder>/<DIR>). Instead, they begin within the destination directory and are relative to that directory.

The "URL Prefix" setting on the destination configuration screen is used to resolve URLs by specifying the location of those files on the delivery system. That is, URLs for links within the exported files are created like this:

```
<URLPREFIX>/<path_of_parent_page>
```

This means that the value for the "URL Prefix" parameter must match the value of the web alias that identifies the directory where the files are found. If you are using only `cs.pgexportfolder` to create the destination directory, the web alias that the "URL Prefix" represents must point to that location. And if you have set the "Base Directory" field on the destination configuration screen to add a subdirectory to the root directory specified by `cs.pgexportfolder`, be sure that the web root or the web alias represented by the "URL Prefix" parameter points to this directory.

File Naming Conventions

How WebCenter Sites names an exported file depends on whether a file name is provided by the asset (in the "Inspect" form) and whether the "Use Simple File Naming" parameter is selected (on the Destination Configuration form; for more information on this setting see "Configuring Your System for Export to Disk Publishing," on page 273). Possible file names are shown in Table 5. The naming convention is explained below the table.

Table 5: File Naming Conventions

File Name	Description
`<pagename>_<packedargs(if any)>_<filename>.<SUFFIX>`	Used for assets with a **populated** "Filename" field and "Use Simple File Naming" is not selected.
`<packedargs(if any)>_<filename>.<SUFFIX>`	Used for assets with a **populated** "Filename" field and "Use Simple File Naming" is selected.
`<pagename>_<packedargs(if any)>_<assetID>.<SUFFIX>`	Used for assets with an **empty** "Filename" field and "Use Simple File Naming" is not selected.
`<packedargs(if any)>_<assetID>.<SUFFIX>`	Used for assets with an **empty** "Filename" field and "Use Simple File Naming" is selected.

> **Note**
>
> When export files are named, the following syntax changes are made:
>
> - Slash characters (/) in page names are converted to hyphens (-) in file names.
> - Equal signs (=) in packed arguments are converted to hyphens (-) in file names.
> - Ampersand characters (&) in packed arguments are converted to underscores (_) in file names.

- `<pagename>` is the name of the SiteCatalog page entry of the template that is used to render the asset. (All template assets have SiteCatalog page entries.)

> **Note**
>
> `<pagename>` is dropped when the "Use Simple File Naming" parameter is selected in the Destination Configuration form. Use "Use Simple File Naming" selectively. If your assets are rendered by more than one template, do not select "Use Simple File Naming." For more information about "Use Simple File Naming," see "Configuring Your System for Export to Disk Publishing," on page 273.

- `<packedargs(if any)>` are passed in from the page entry's root element. If any packed arguments are passed in from the root element of the asset's rendering template, the values of those arguments are also included in the name of the file generated for the asset.

  For information about packed arguments and how they relate to URLs, see the *Oracle WebCenter Sites Developer's Guide*.

- `<filename>` is the value of the asset's "Filename" field. If the "Filename" field is empty, WebCenter Sites uses the `asset_ID` specified in the asset.

> **Note**
>
> A file name set for an asset for a specific destination in the "Specify Path/ Filename *for destination*" form supersedes the file name provided in the "Filename" field on the asset's "New" or "Edit" form when the asset is exported to that destination.

- `<asset_ID>` is the ID that is specified in the asset. `<asset_ID>` is used if the asset's "Filename" field is empty or if the following note applies.

  Note

  WebCenter Sites needs both the asset's ID and type to determine whether a file name is provided for the asset. If the identity of the asset's type is not provided, the file name cannot be used. In such a case, WebCenter Sites uses the asset's object ID in place of its file name.

- `<SUFFIX>` is a destination configuration option that specifies the extension of the file name. For more information about the "suffix" parameter, see "Configuring Your System for Export to Disk Publishing," on page 273.

Table 6 provides examples of how file names are created during Export to Disk publishing. In the examples "Use Simple File Names" is not selected in the destination configuration.

Table 6: Export File Naming Conventions

Asset Name	SiteCatalog Pagename	packedargs	filename	asset_ ID	SUFFIX	Exported File Name
Home	BF/Page/Home	cid=123 c=Page		123		BF-Page-Home_123.html
Home	BF/Page/Home	cid=123 c=Page		123	.htm	BF-Page-Home_123.htm
Home	BF/Page/Home	cid=123 c=Page	home.html	123		BF-Page-Home_home.html
Home	BF/Page/Home	cid=123 c=Page	home.html	123	.htm	BF-Page-Home_home.html
Home	BF/Page/Home	cid=123 c=Page PACKEDARGS= "topicword=oil"	home.htm	123		BF-Page-Home_topicword_oil_home.htm

Export Starting Points

The Export to Disk publishing method cannot begin rendering files unless it knows where to start. You tell it where to start by designating at least one starting point: that is, a page asset and the template that should be used to render it. WebCenter Sites invokes the root element of the template's page name, and begins rendering every approved asset that is connected to that export starting point—assets connected through an association, a hyperlink, a navigation bar, a query, and so on.

Note

There must be at least one export starting point for an exported site. Typically it is your home page. However, depending on how your online site is designed, you may need to designate more than one export starting point. Why? Because if there is a section on your site that isn't connected to the rest of the site, it will not be rendered. For example:

- If your online site has more than one top-level page asset, you need an export starting point for each one.
- If your site uses a combination of static pages and dynamic pages and there is a hard-coded static URL in an HREF on a dynamically-generated page, the asset that the URL points to should be designated as an export starting point.

Configuring Your System for Export to Disk Publishing

The main steps when configuring a system for Export to Disk publishing are these:

- Step 1. Create the Batch User Account (If One Does Not Exist)
- Step 2. Specify the Root Export Directory
- Step 3. Configure an Export Destination
- Step 4. Map a URL Prefix for Your Web Server
- Step 5. Create the Export Starting Points
- Step 6. Approve Your Assets
- Step 7. Publish and Test the Results
- Step 8. Set Up the Schedule

Note

Configuring a system for Export to Disk publishing requires you to provide information that is specific to your WebCenter Sites installation and your business practices. Before starting the procedures in this section, it is best to read them in order to gather and confirm the information they prompt you to provide.

Step 1. Create the Batch User Account (If One Does Not Exist)

Note

The steps in this section must be completed only once regardless of the number and types of publishing methods you are setting up. If an account already exists for your installation, skip to "Step 2. Specify the Root Export Directory," on page 275.

The purpose of creating a batch user account is two-fold:

- Configure a **batch user** account for the publishing system on the source to use.
- Specify where the publish **logs** should be stored.

The procedures in this section require you to set several properties in the `futuretense.ini`, `batch.ini`, and `futuretense_xcel.ini` files. Additional properties are also available to you for fine-tuning your system configuration. For a complete list of properties, see the *Property Files Reference*.

To create the batch user account

1. Log in to the administrator's interface on your source system.
2. In the **Admin** tab, expand **Management Tools** and double-click **User**.
3. Create a user account for the publishing system on the source system to use (that is, the batch user) and assign it the following ACLs:

- Browser
- ElementEditor
- PageEditor
- TableEditor
- UserReader
- Visitor (if your installation includes Oracle WebCenter Sites: Engage)
- VisitorAdmin (if your installation includes Oracle WebCenter Sites: Engage)
- xceladmin
- xceleditor

If you need help with this step, see "Creating a New User," on page 86.

4. Start the Property Editor:
   - Windows: `propeditor.bat`, typically located in `<sites_install_dir>/`
   - UNIX: `propeditor.sh`, typically located in `$HOME/<sites_install_dir>`

   a. Open the `futuretense_xcel.ini` file for the source system.

      1) In the **Publishing** tab, set values for the following properties:

      - `xcelerate.batchhost`

        Set this property to the host name of the application server that is hosting the **source** system. The source system is the batch host. If the port number of the web server is anything other than 80, you must also include the port number. For example, `myserver:7001`. In a clustered WebCenter Sites environment, only one batch host is supported. The `xcelerate.batchhost` property must be set on each cluster member to point to the dedicated host.

      - `xcelerate.batchuser`

        Set this property to the name of the user that you created in step 3 of this procedure.

      - `xcelerate.batchpass`

        Set this property to the password of the user that you created in step 3 of this procedure.

   b. Save and close the property file.

   c. Open the `batch.ini` file for the source system.

      1) In the **Results** tab, set the `request.folder` property to point to the directory that you want to hold your publish log files. By default, this property is set to the `dispatcher` subdirectory in the application server installation directory.

      Note

      Be sure to set this property to a directory that can be written to. Otherwise, no session information will be displayed in the **Publish History** tab of the Publish Console.

2) Save the property file and close the Property Editor.

5. Restart the application server.
6. Continue with the configuration procedure for Export to Disk publishing. Go to "Configuring Your System for Export to Disk Publishing," on page 273 the next procedure in this section.

Step 2. Specify the Root Export Directory

As mentioned previously in this chapter, the directory that WebCenter Sites exports files to is determined by the `cs.pgexportfolder` property in the `futuretense.ini` file. (If you wish to add subdirectories to this directory, you enter a value in the "Base Directory" field in the Destination Configuration screen.)

To set the root directory for the exported files

1. Start the Property Editor on the source system. If you need help with this step, see the *Property Files Reference*.
2. Open the `futuretense.ini` file.
3. In the **Export/Mirror** tab, select the `cs.pgexportfolder` property and set the value to the file directory (location) to which all files should be exported.

   This is a global setting for the system. If you have multiple destinations and want the files published to those destinations to be stored in separate subdirectories within this directory, enter a value in the "Base Directory" field when you configure those destinations (in "Step 3. Configure an Export Destination").

   For more information about export directories, see "Path Naming Convention," on page 264.
4. Save the property file and close the Property Editor.
5. Restart the application server.

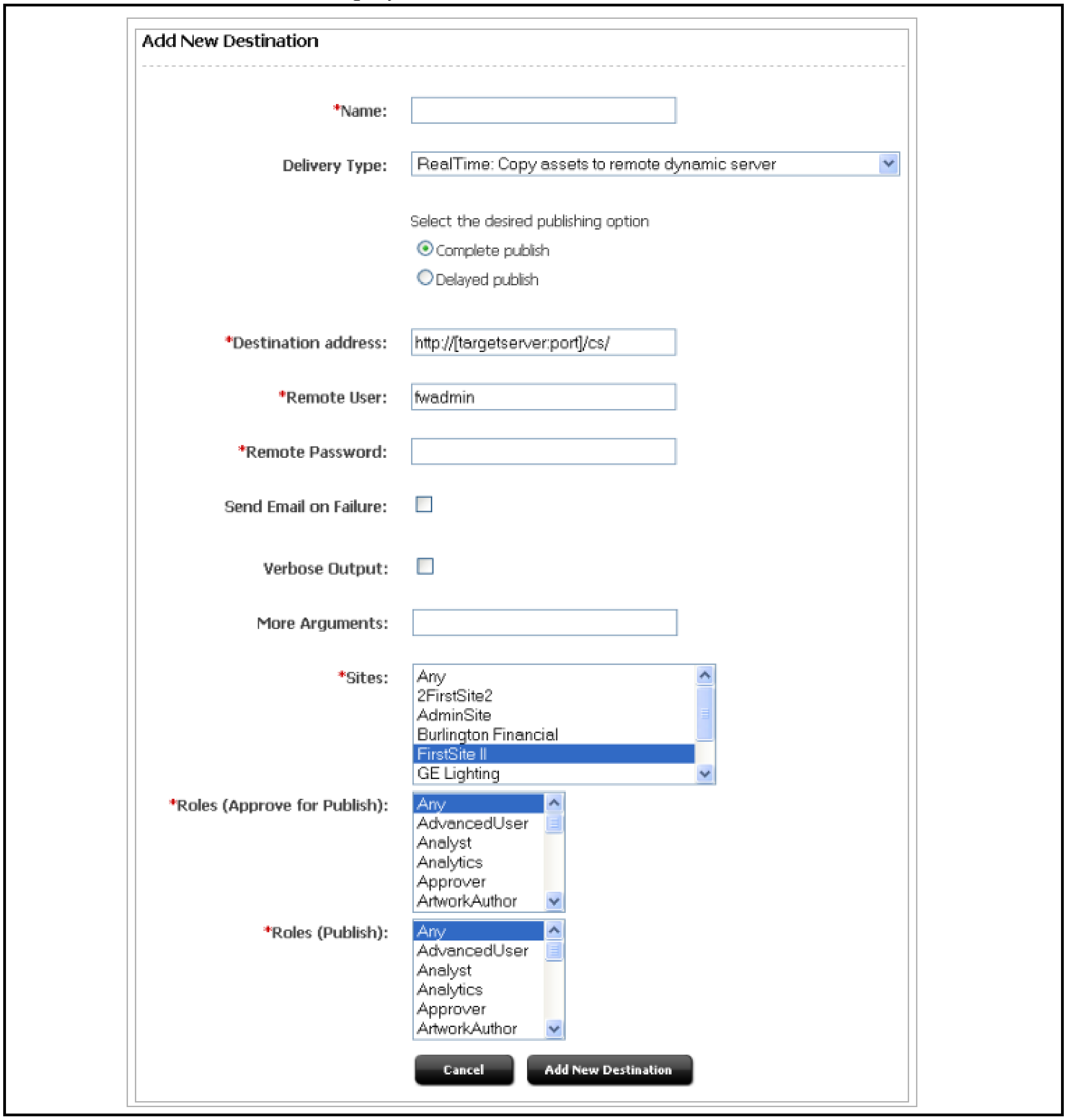
> **Note**
>
> Before you run an export publishing session, make sure that this root export directory exists and that it has enough space for the HTML pages that will be created there.

Step 3. Configure an Export Destination

To create an export destination

1. Log in to the administrator's interface on the source Sites system.
2. In the **Admin** tab, expand **Publishing**, then **Destinations**, and double-click **Add New**.

WebCenter Sites displays the “Add New Destination” form.



3. Fill in the fields on the “Add New Destination” form as follows:

- In the **Name** field, enter a unique name for the destination.
- In the **Delivery Type** drop-down list, select **Export to Disk: Export Web Files to Disk**. The available fields on the screen will change slightly.
- **Base Directory**: Enter a subdirectory name for the files published to this export destination. The base directory is created as a child directory of the root directory that is specified by the `cs.pgexportfolder` property.

  When you enter a value in this field, the export path for the publishing destination is as follows:

  ```
  <cs.pgexportfolder>/<DIR>
  ```

  where `DIR` is the base directory.

> **Note**
>
> URLs that are contained in exported files do not include this destination directory path in the URL. Exported files have URLs that begin below this level, inside the destination directory. Therefore, if you enter a value in this field and you enter a value in the "URL Prefix" field as well, be sure that the web root or the web alias represented by the "URL Prefix" field points to this directory.

This parameter is typically used to organize the contents of the root export directory when there are multiple export destinations for your sites.

- **URL Prefix**: Enter the prefix that the export process will add to the beginning of the URLs that are used as links (HREFs) within the exported files.

  Specify the web server alias of your delivery system in the URLs so that the links in the HTML files can be resolved when the files are moved to that system. If you leave this field blank, URLs will be relative.

  The names of the generated files do not include this prefix—only the links within the files use it.

  Because you should always test the HTML files before you move them to the delivery system, be sure that the name of the web alias on the testing system is the same as the name of the web alias on the delivery system.

- **Suffix**: In this text box, specify the file suffix to use for the generated files when the **Filename** field for the asset is not used or it does not specify a suffix. The default is **html**.

- **Use simple file naming**: If this parameter is selected, WebCenter Sites overrides the normal file-naming conventions by ignoring the `SiteCatalog` page entry of the template that is used to render the asset and uses only the value entered in the asset's **Filename** field as the file name. If the asset does not have a filename, WebCenter Sites uses the asset's ID, instead. (For a description of the file-naming conventions, see "File Naming Conventions," on page 269.)

> **Note**
>
> There is one exception to this rule: assets with upload fields (assets that are blobs) will always receive a standard (long) file name, even when this value is set to `true`.

> **Caution!**
>
> The **Use simple file naming** parameter must be used with caution. If your site is designed such that individual assets are rendered by multiple templates, you must not use this parameter. Without the `SiteCatalog` page entries used in the file name, file names cannot be guaranteed to be unique when an asset is rendered by more than one template.

- **Use simple directory naming**: When selected, this parameter overrides the normal directory-naming convention. (For a description of the file-naming conventions, see "Path Naming Convention," on page 264.)

  When **Use simple directory naming** is selected, WebCenter Sites puts the rendered HTML file directly into the default export directory when there is no path information available from the parent asset. (Normally, WebCenter Sites uses the page asset's ID when there is no path information.) Note that if there is a value in the Path parameter of the asset's parent page asset, this selection is ignored.

- **Old Template**: Select this parameter to use the rendering methodology that was standard in 3.5 and earlier versions of the product.

  Note

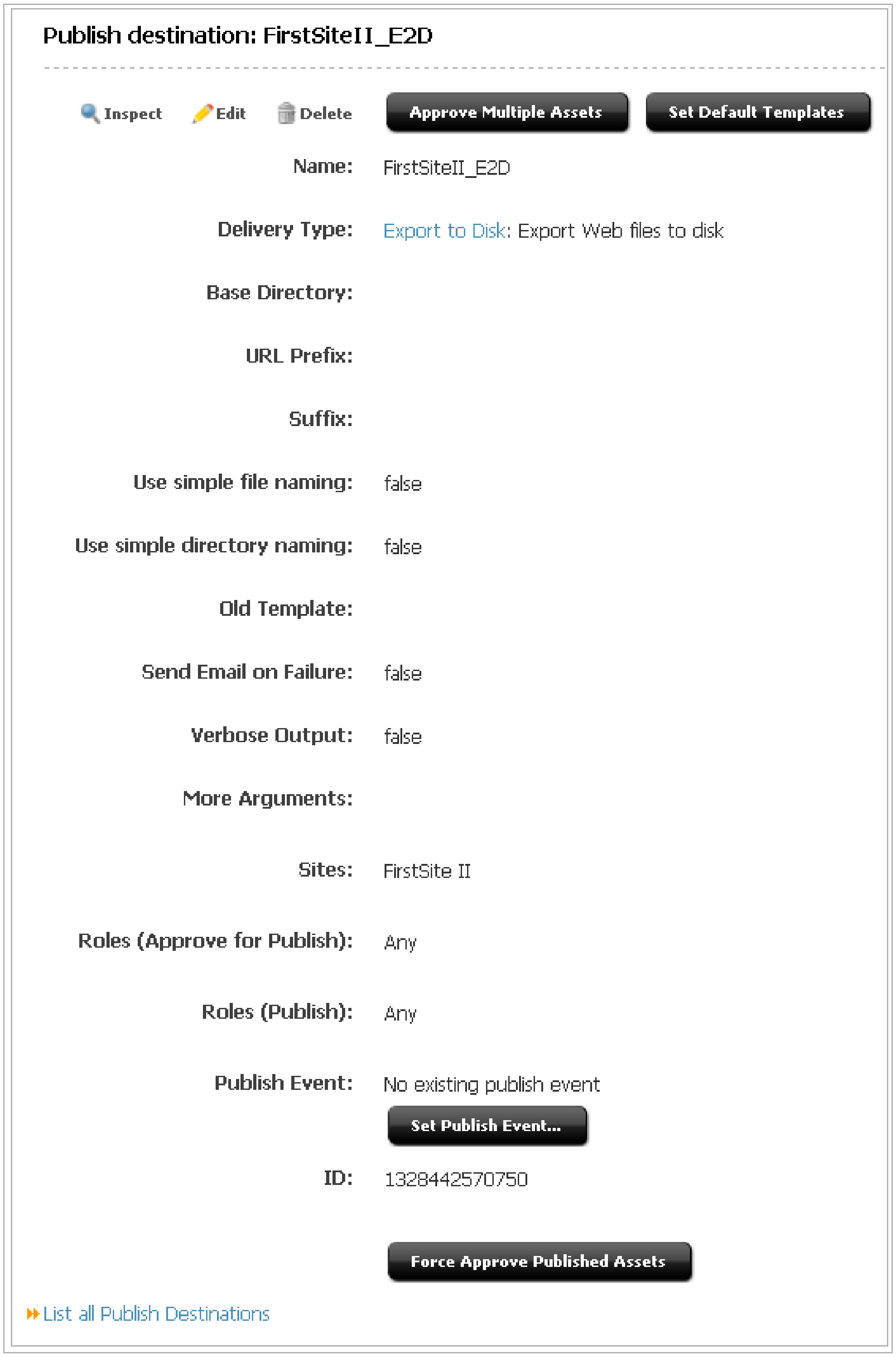
  If the online site that you present on your delivery system was designed for the rendering model used in Sites versions prior to 3.6 and it has not yet been redesigned for the 3.6 rendering model, you must select this parameter.

  If you select this parameter, do not set an export starting point for your site. The export starting point is determined by the templates themselves.

- **Send Email on Failure**: If publishing fails and email notices to that effect are required, select this option. Additional fields appear.
  - **Email Addresses:** Enter the recipient's email address, and click **Add**. (This field is displayed only when **Send Email on Failure** is selected). All email addresses that will receive an email on failure are displayed.
- **Verbose Output**: Activates detailed error logging during the publishing process. When selected, messages in addition to error messages are written to the PubMessage table. Because additional information lengthens the publishing process, be sure to select this option only for troubleshooting.
- **More Arguments**: No additional arguments may be specified at this time.
- **Sites**: Select the sites whose assets can be approved for and published to this destination.
- **Roles (Approve for Publish)**: Select the roles to which you are assigning approval privileges. All users assigned these roles can approve assets.
- **Roles (Publish)**: Select the roles that have publish privileges. All users assigned these roles can publish.

4. Click **Add New Destination**.

The "Publish Destination" screen appears showing the new destination.



5. Repeat steps 2–4 for each additional export destination you need to configure.

Step 4. Map a URL Prefix for Your Web Server

To make your exported content available to the public at its final destination (your delivery system), you must configure your web server to map a URL path prefix to the place where the files will reside so that URLs can be resolved.

Refer to the vendor documentation for your web server for information about the appropriate procedure for mapping URL prefixes on your web server. Remember that the name you specify for this prefix must match the name that you specified with the URL Prefix destination configuration parameter for the destination. If you enter a URL Prefix, be sure that the web root or alias includes the directory you entered.

Step 5. Create the Export Starting Points

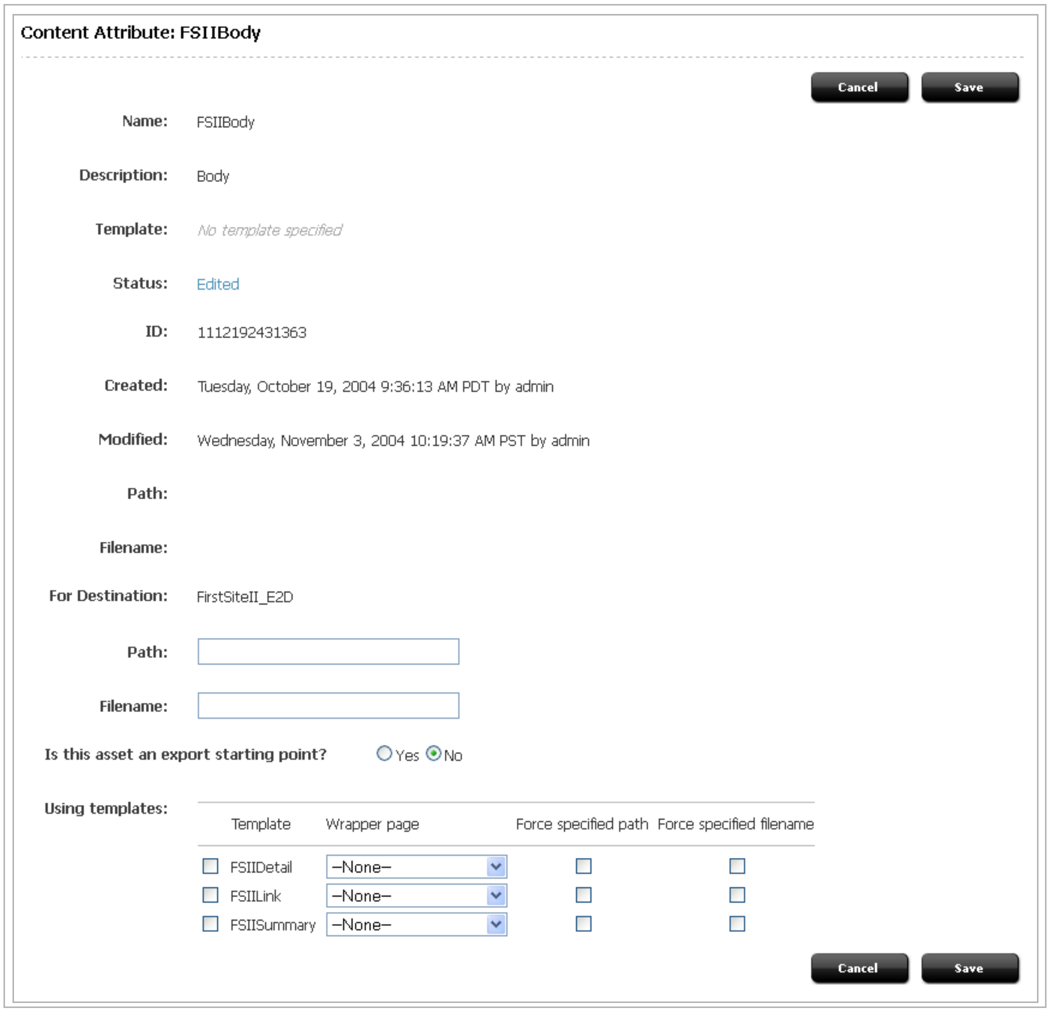
To create an export starting point

1. Using the administrator's interface on the source system, find the asset that you want to specify as an export starting point.
2. Open the asset in its "Status" form and scroll to the "Publishing Destination" section.
3. Do one of the following:
   - If the asset is not yet approved for the destination you are currently configuring, select **Approve for Publish** from the toolbar (The green check mark icon) and go to step 4.
   - If this asset is already approved for the destination you are currently configuring, go to step 6.
4. In the publishing approval form, select the destination for which you are approving the asset and click **Approve**.

   WebCenter Sites calculates dependencies for all the assets linked to this asset and displays the results.
5. When WebCenter Sites displays the approval results, click the **Status** icon (a clock face with a green check mark) from the toolbar.
6. Scroll down to the section of the "Status" form that displays the selected destination.

7. In the **File/Path** field, click **Specify Path/Filename, Start points** to configure the export starting point:

8. In the next screen, fill in the fields as instructed below:



a. (Optional) **For Destination** section:

1) **Path**: If you want to specify unique path information for the asset being published to this destination (the path information is not provided elsewhere for this asset), enter the path in the **Path** field.

2) **Filename**: If you want to specify file name information that is different for this destination, enter it in the **Filename** field. A file name entered in this form for this destination overrides any file name information provided elsewhere for the asset.

b. Select **Yes** for **Is this asset an export starting point?**

c. (Optional) **Using Templates** section:

1) Select a template.

2) Select a wrapper page.

3) Select **Force specified path** if you would like to force this asset to be exported to the path that is named in the **For Destination** section.

4) Select **Force specified filename** if you would like to force the use of the file name that is specified in the **For Destination** section.

9. Click **Save**.
10. Repeat steps 1–9 for each top-level page asset in the site.

Step 6. Approve Your Assets

If you want to perform a simple test of your configuration, select the export starting point with the fewest number of dependent assets and approve each of those assets.

If you want to complete a test publish of your entire site, approve all the assets for the site. See “Approving Multiple Assets,” on page 315 for help with approving many assets at once.

Step 7. Publish and Test the Results

Once the assets you want to publish to the selected destination are approved, run a test publishing session:

1. In the administrator’s interface on the source system, click **Publishing** in the button bar.
2. In the **Publish Console**, select your destination from the drop-down list and click **Select Destination**.

   WebCenter Sites displays information about the assets that are ready to be published to this destination. (If you have not yet created an export starting point for this destination, the form states this fact. You must create at least one export starting point before the publishing session can begin.)
3. Click **Publish**.

   A confirmation message appears. Click **Yes** to proceed with the publish.

   The publishing system exports all the approved assets for this destination into files.
4. In the **Publish Console Active** tab you can observe that the session is running.
5. When the session completes, click the **History** tab to view the session summary for this session.

   Mouse over the session information to view summary information;
6. In the Publish Console **History** tab, select the link to the publishing destination.

   A session summary displays a list of all the files that were created during the export process.
7. Click the **Preview** (binoculars) icon next to the file representing the top-level page in your site.

   WebCenter Sites opens the page in a new browser window.
8. Scan the page for errors and test all the links. Make sure to test links to all of the other files that were generated.
9. Test the results by directly entering the URL of the rendered home page into your browser and navigating the entire site. To do this, you must first set up a web server root on the Sites management system that points to the export directory.
10. If you decide that the files are ready for delivery, copy them to your delivery system.

Step 8. Set Up the Schedule

After you have ensured that:

- Your destination is configured correctly,
- The file names and directory names are created according to your needs, and
- Your web server is delivering the files correctly,

you can finish configuring your publishing system by completing the following steps on the source system:

- Create scheduled publish events for the destination. For help with this step, see "Scheduling Publish Events," on page 319.
- Plan how you will copy the generated files to your web servers, for both the testing area and the delivery system. For example, you can set up a script that automatically copies the files via FTP to a testing area after a publishing session completes.

Chapter 15

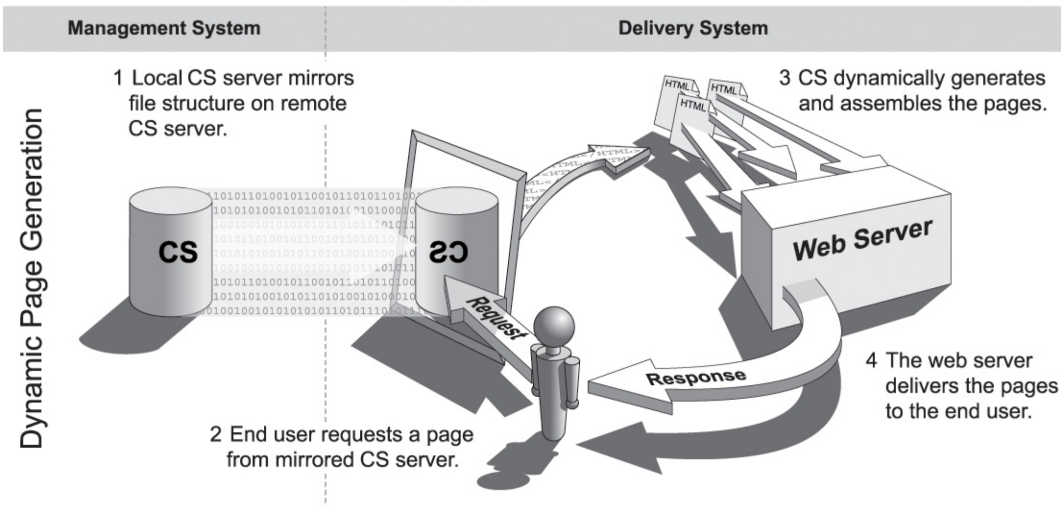
The Mirror to Server Publishing Process

This chapter explains how Mirror to Server publishing works and how it is configured.

This chapter contains the following sections:

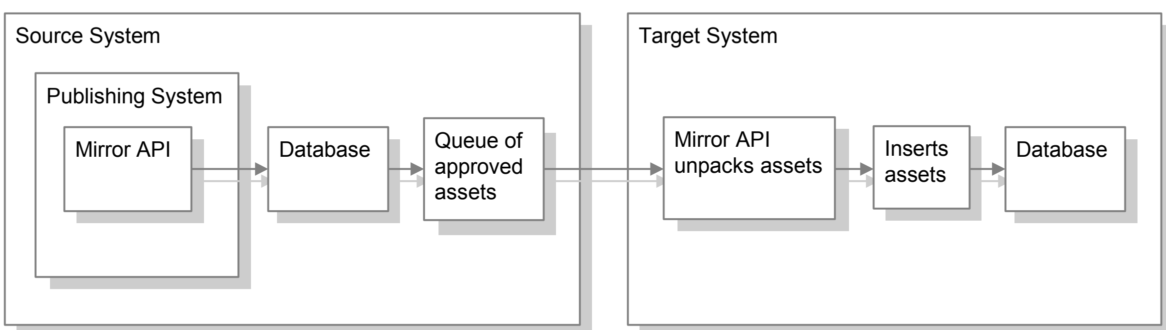
How Mirror to Server Publishing Works

The Mirror to Server publishing method gathers information from the approval system, the publishing schedule, and the destination configurations, copies data to the destination, unpacks that data on the destination system, and then invokes the CacheManager servlet to refresh any pages that should be regenerated to take advantage of the new content.



Whether or not your Sites delivery system delivers content dynamically, you probably use the Mirror to Server publishing method to move data from your development systems to your management system. For information about that process, see "Troubleshooting," on page 329.

When a Mirror to Server publishing session runs, this is what occurs:



1. On the source system, Mirror to Server publishing uses the list of approved assets that the publishing system passed to it to create a mirror queue for the destination.

   Mirror Queue for Basic Assets

   For basic assets, the following information is added to the queue:

   - The asset's main table row. For example, for page assets, the asset's row in the `Page` table.

- The appropriate rows in the AssetPublication table. These rows list sites and which assets belong to them.
- Rows in the AssetRelationTree table that refer to any assets that are associated with the asset being published.
- The associated assets referenced by those rows in the AssetRelationTree table. The asset table rows, AssetPublication rows, and associated assets of any dependent assets are also mirrored, if they are approved and have not yet been published.

Mirror Queue for Flex and Complex Assets

For flex asset types and the other multi-table asset types like template and CSElement, the information from the appropriate rows from all of their tables are serialized into an object and stored in the _Publish table for that asset type.

For example, when a template is to be published, the appropriate rows from the Template, SiteCatalog, and ElementCatalog tables are serialized into an object and stored in the Template_Publish table.

Each item in a _Publish table is added to the mirror queue.

2. WebCenter Sites uses the AssetPublishList table to create a list of all the assets that are in the mirror queue.
3. The mirror operation starts.

   First, the AssetPublishList is mirrored from the source to the destination.
4. The mirror queue is delivered and the publishing system unpacks the assets in the queue.

   For flex assets, Mirror to Server publishing de-serializes the objects in the _Publish tables, and inserts the results in the appropriate tables.

   For basic assets, each row in the queue is copied.
5. When the items in the mirror queue are unpacked, the publishing system sends messages that the mirror publish concluded successfully. The destination system responds as follows:
   - The newly-published assets are marked as changed on the destination system, which means that before they can be published from that system to another destination, they must be approved. Note that this feature is enabled by default. You can turn it off if you need to. See "Step 12: Turn Off Asset Invalidation on the Delivery System," on page 302 for details.
   - The CacheManager servlet on the destination regenerates the appropriate pages in the cache so that all pages that refer to the assets that were just published are updated. It also rebuilds any pages that have unknown compositional dependencies.
   - CacheManager then communicates a message about which pages must be refreshed to each Satellite Server identified by the satellite.ini file on the destination system. It communicates with the co-resident version and any remote instances that are identified as belonging to this Sites system. The Satellite Server applications then use that information to refresh the Satellite Server page caches.
   - The AssetPublishList table is cleared, making it ready for the next publishing session.

6. On the source system, the publishing system updates the publish log file. Unlike the Export to Disk publishing method, which writes to the publish log after each asset is exported, Mirror to Server publishing waits until the entire mirror queue has been successfully mirrored before writing the results to the publish log.
7. When the publishing session is successful, WebCenter Sites concludes the session by writing information about the published references to the `PubKey` and `PublishedAssets` tables and by clearing the `AssetPublishList` table on the source system.
8. WebCenter Sites also writes information about the assets that were published to the `ApprovedAssets` and `ApprovedAssetDeps` tables so the approval system can calculate dependencies correctly the next time an asset is approved.

Before Configuring Mirror to Server Publishing

Before you configure the Mirror to Server publishing process, take into the consideration the topics that are presented in this section. Prior knowledge of the information you will be providing will help to ensure a smooth configuration process:

- Users and Mirror to Server Publishing
- CacheManager
- Configuring a Mirror Destination

Users and Mirror to Server Publishing

In addition to the batch user account on the source system that a publish event uses to process a publishing session, the Mirror to Server publishing method requires another user account: the **mirror user account**, which is located on the destination server. This is the user account that the publishing system uses to unpack the mirror queue on the destination system.

When you set up a Mirror to Server publishing destination, you must create the mirror user account on the destination system that the destination represents.

CacheManager

The CacheManager is a WebCenter Sites servlet that maintains the page cache on any dynamic Sites system, including the management system.

CacheManager is important to the publishing system because it locks the appropriate assets on the destination during a mirror publish and ensures the integrity of the page cache both before and after a mirror publishing session.

For more information about the CacheManager, see the *Oracle WebCenter Sites Developer's Guide.*

Configuring a Mirror Destination

The Mirror to Server publishing method copies information for approved assets from one Sites database to another. This information must be configured for the mirror destination. Configuring a mirror destination involves two steps: initializing the destination, and configuring the data for the destination.

Whenever you create a new mirror destination, you must initialize it before you can publish to it. To initialize a mirror destination, you specify the following information:

- The sites. Based on the sites that you select, the appropriate rows from the `Publication`, `SitePlanTree`, and `PublicationTree` tables are mirrored to the destination.
- Any custom table that supports or is directly related to your asset types. In other words, a table for assets that was not created by either AssetMaker or Flex Family Maker. Examples of these are `Source` and `Mimetype`. These tables are called "auxiliary tables."

Mirror destination configuration is the process of indicating which data (for example, assets types, associations, and start menu items) will be mirrored to the target destination. Mirror destination configuration moves configuration data and rows from auxiliary tables—which are non-asset database tables that are used for the dynamic display of the assets—from one Sites database to another.

When to Use Mirror Destination Configuration

You use the Mirror Destination Configuration feature in several situations:

- To set up any new mirror destination.
- To move configuration items that support asset types (start menu items, associations, and so on) from a development system to a management system or to a delivery system.
- To move workflow configuration data from a development system to a management system.
- If you or your developers add any additional sites, asset types, or auxiliary tables to your system.
- If you or your developers add any additional categories or subtypes for existing asset types.
- When you need to troubleshoot your configuration. If you can successfully initialize a mirror destination, that means that the source and the destination systems are communicating.

Configuring Your System for Mirror to Server Publishing

The main steps when configuring a system for Mirror to Server publishing are as follows:

- Step 1: Create the Batch User Account (If One Does Not Exist)
- Step 2: Set Up the Destination System
- Step 3: Identify the Mirror User to the Source System
- Step 4: Identify the Local Proxy Server to the Source System (If One Exists)
- Step 5: Create a Mirror Destination
- Step 6: Initialize the Destination
- Step 7: Configure the Mirror Destination

- Step 8: Approve Your Assets
- Step 9: Publish the Assets
- Step 10: Test the Results
- Step 11: Set Up the Schedule
- Step 12: Turn Off Asset Invalidation on the Delivery System

Note

Configuring a system for Mirror to Server publishing requires you to provide information that is specific to your WebCenter Sites installation and your business practices. Before starting the procedures in this section, it is best to read them in order to gather and confirm the information they prompt you to provide.

Step 1: Create the Batch User Account (If One Does Not Exist)

Note

This procedure needs to be performed only once on each source system, regardless of the number and types of publishing destinations you are configuring.

If a batch user account already exists on your source system, skip to "Step 2: Set Up the Destination System."

- If you have not already done so, create the batch user account. For instructions, see "Step 1. Create the Batch User Account (If One Does Not Exist)," on page 273.
- If a batch user account already exists for your installation, skip to "Step 2: Set Up the Destination System."

Step 2: Set Up the Destination System

To mirror publish assets from one Sites system to another, you must ensure that the sites and asset types are the same on both the source and the destination system. You must also create an additional user on the destination system, called the **mirror user** (as opposed to the batch user, which exists on the source). This user completes the mirror publish database transactions on the destination system.

Note

Database properties on the source and destination systems, if not an exact match, must be compatible, particularly database schema options (set in `futuretense.ini`). Be sure to restart the application server if you make changes to `futuretense.ini`.

To set up your destination system

1. If you have any custom support tables—a lookup table for a field in an asset type, for example—create those tables on the destination system.

2. On the destination system, create the mirror user and note the following:

> **Note**
>
> The mirror user has the same privileges as the RealTime user configured for RealTime publishing. If a RealTime user is configured for RealTime publishing, the same user may serve as the mirror user.

- This user must have the following ACLs:
  - Browser
  - ElementEditor
  - PageEditor
  - TableEditor
  - Visitor (if Oracle WebCenter Sites: Engage is installed)
  - VisitorAdmin (if Oracle WebCenter Sites: Engage is installed)
  - xceladmin
  - xceleditor
- Because the mirror user account is used by the CacheManager to regenerate the page cache after a publishing session, the mirror user must have sufficient privileges to regenerate all the pages in the cache. Therefore, the mirror user account must be given to all ACLs which are assigned to page entries in the `SiteCatalog` table or database tables that are holding data to be rendered.

> **Note**
>
> If any of the sample sites are installed on the destination system, a user named "mirroruser" already exists. For security, if you decide to keep this user as your mirror user, be sure to **change the password** for this user; or if you decide to create a different mirror user, be sure to delete the sample site "mirroruser."

- If you need help with creating the mirror user, see "Creating a New User," on page 86.

Step 3: Identify the Mirror User to the Source System

Next, you identify the name and password of the mirror user on the destination system to the source system by setting property values in the `futuretense.ini` file on the source system.

> **Note**
>
> Because the proxy server is used by both Mirror to Server and RealTime publishing, it may have been set up for RealTime publishing. In such a case, review the properties in the steps below to ensure they are set correctly.

To identify the mirror user to the source system

1. Start the Property Editor and open the `futuretense.ini` file on the source Sites system. If you need help with this step, see the *Property Files Reference*.
2. In the **Export/Mirror** tab specify values for the following properties:
   - `cs.mirroruser`

     Set this property to the name of the user that you created on the destination system in the preceding procedure.
   - `cs.mirrorpassword`

     Set this property to the password of the user that you created on the destination system in the preceding procedure.
3. Save the property file.
4. Do one of the following:
   - If there is a firewall separating your source system from the destination system, go to "Step 4: Identify the Local Proxy Server to the Source System (If One Exists)," on page 292.
   - If there is no firewall separating your source and destination systems, close the Property Editor. Stop and restart the application server. Then go to "Step 5: Create a Mirror Destination," on page 293.

Step 4: Identify the Local Proxy Server to the Source System (If One Exists)

Note

This procedure needs to be performed only once on each source system, regardless of the number and types of publishing destinations you are configuring.

Skip to Step 5: Create a Mirror Destination if:

- The local proxy has already been identified to your source system, or
- A proxy server is not used.

To identify the local proxy to the source system:

1. Open `futuretense.ini` in the Property Editor on the source system, select the **Export/Mirror** tab, and specify values for the following properties:
   - `cs.mirrorproxyserver`

   Set this property to either the name or the IP address of the local proxy server.
   - `cs.mirrorproxyserverport`

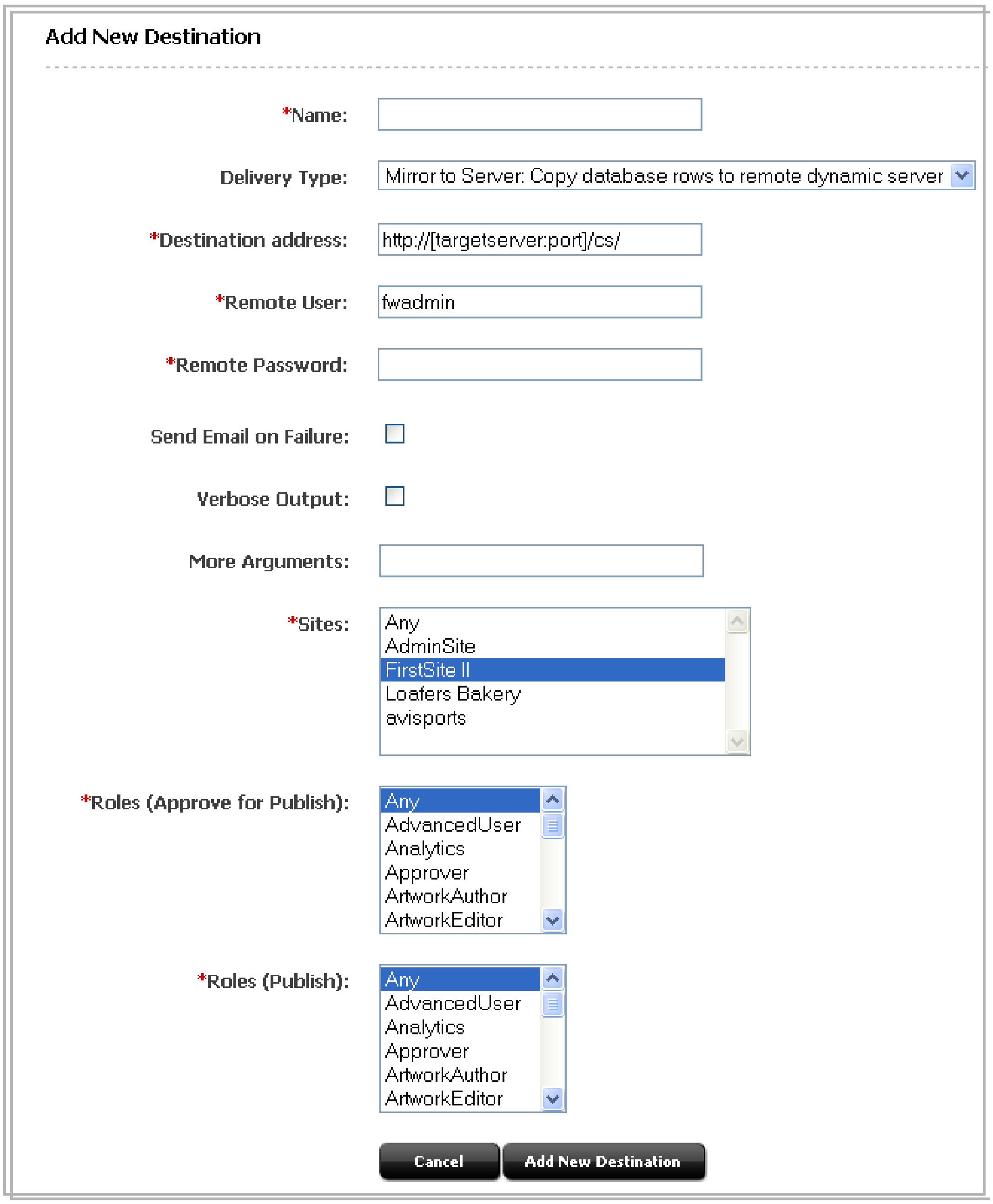
   Set this property to the port number of the local proxy server.
2. Save the property file and close the Property Editor.
3. Restart the application server.

Step 5: Create a Mirror Destination

To create a mirror destination

1. Log in to the administrator's interface on the source Sites system.
2. In the **Admin** tab, expand **Publishing**, then **Destinations**.
3. Under **Destinations**, double-click **Add New**.

   The "Add New Destination" form appears.

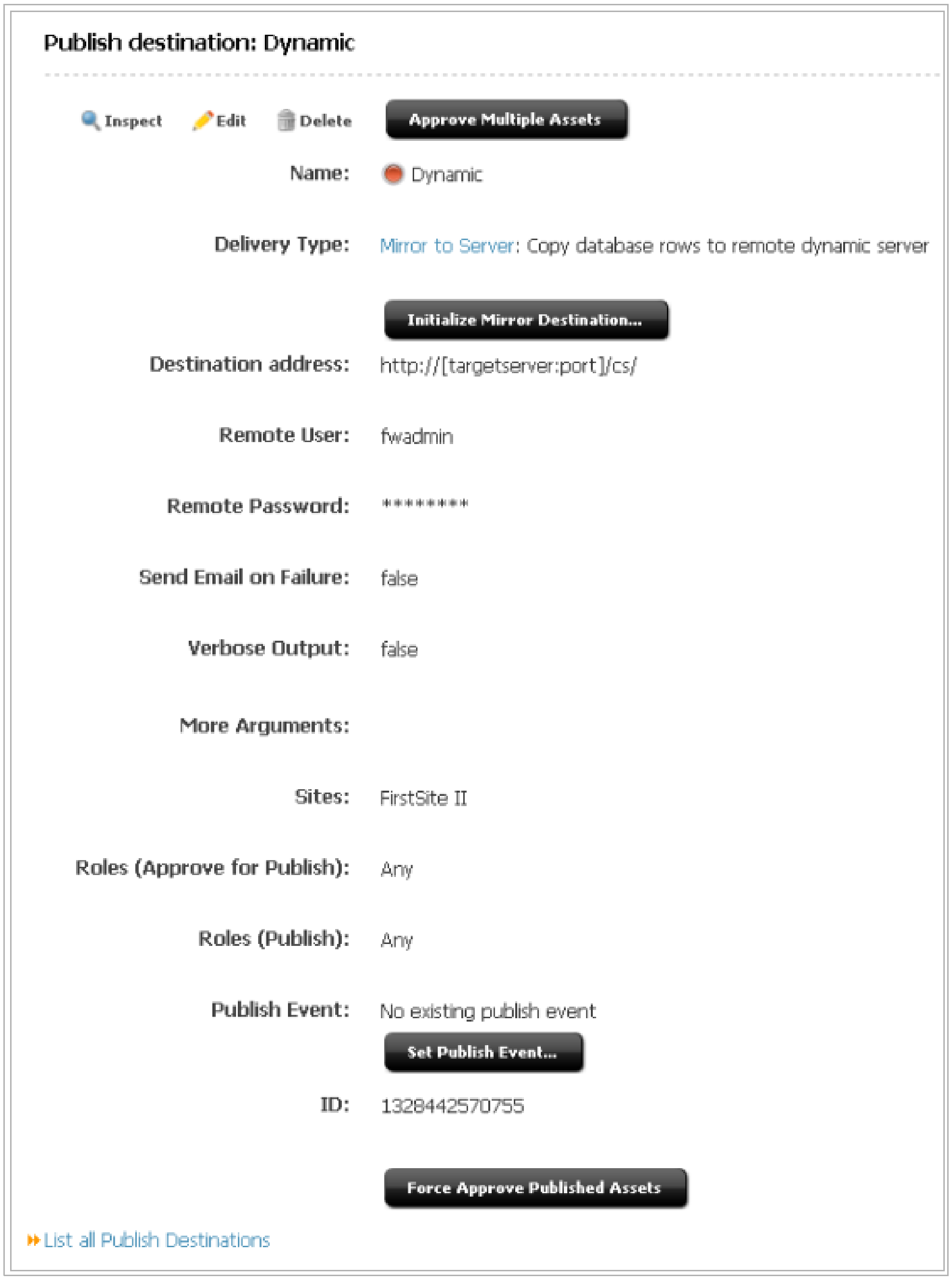


4. Configure the fields on this screen as follows:
   - **Name**: Enter a unique name for the destination.

- **Delivery Type**: Select Mirror to Server: Copy database rows to remote server.
- **Destination Address**: Enter the URL of the remote server in the format shown. For [`targetserver:port`], enter the hostname or IP address of the target system and the port to be used by the destination. (A slash is required at the end of the URL because this URL is appended dynamically.)
- **Remote User**: Enter the name of the mirror user created in "Step 2: Set Up the Target System," on page 330. This user will be invoked by the publishing system to unpack the mirror queue on the target system.
- **Remote Password**: Enter the password of the mirror user.
- **Send E-mail on Failure**: If publishing fails and e-mail notices to that effect are required, select the checkbox.
  - **Email Addresses**: Enter the recipient's e-mail address. (This field is available only when Send E-mail on Failure is selected).
- **Verbose Output**: Select this option to activate detailed error logging during the publishing process. When selected, messages in addition to error messages are written to the PubMessage table. Because additional information lengthens the publishing process, select this parameter only for troubleshooting.
- **More Arguments**: This parameter is reserved for future use; no additional arguments may be specified at this time.
- **Sites**: Select the sites whose assets can be approved for and published to this destination.
- **Roles (Approve for Publish)**: Select the roles to which you are assigning approval privileges. All users who are assigned these roles will be allowed to approve assets.
- **Roles (Publish)**: Select the roles to which you are assigning publish privileges. All users assigned these roles can publish.

5. Click **Add New Destination**.

WebCenter Sites displays the Publish Destination screen.



6. If you are ready to initialize this destination system (you must initialize the destination system before you can publish to it), click the **Initialize Mirror Destination** button in the “Publish Destination” form, then proceed to Step 6: Initialize the Destination” for detailed instructions (begin at step 4 of that procedure).

If you need to create more mirror destinations, repeat steps 2 through 5 for each additional mirror destination that you need to create for your source system.

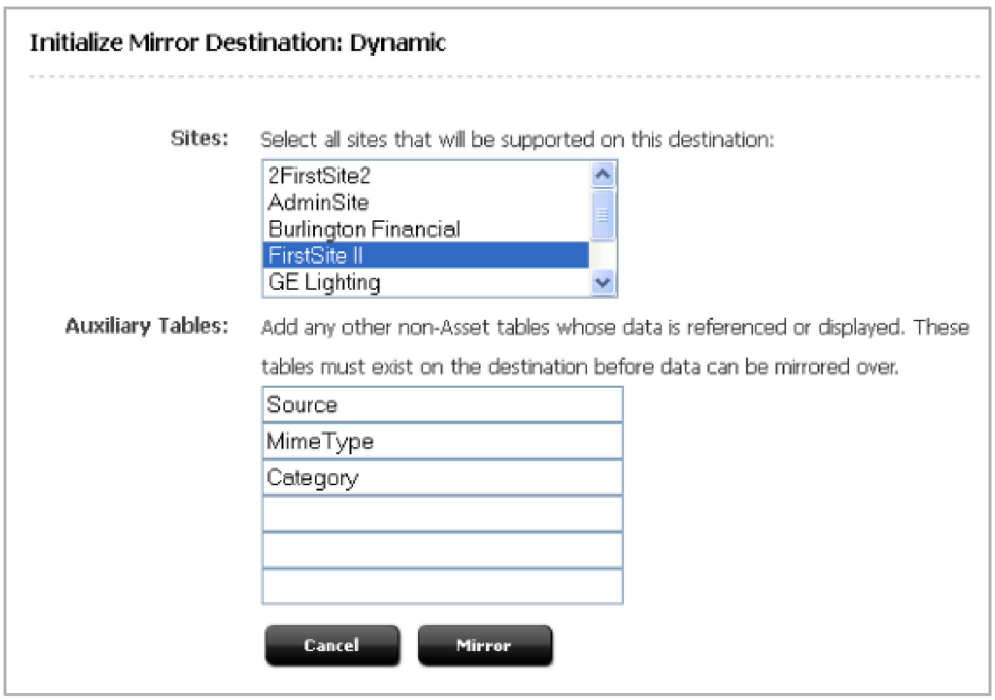
Step 6: Initialize the Destination

You must initialize the destination system before you can publish to it. This creates the basic site information on the destination system. Specifically, the `Publication` and `PublicationTree` tables are updated with the site names and asset types published to the target.

To initialize the destination system

1. In the **Admin** tab, expand **Publishing**, then **Destinations**.
2. Under **Destinations**, double-click the destination you want to initialize.
3. In the "Publish Destination" form, click **Initialize Mirror Destination**.

   The **Initialize Mirror Destination** screen appears.



4. Select the sites whose assets you want to publish to this destination.
5. In the **Auxiliary tables** fields, enter the names of the following tables:
   - `Source`, if you are using the source feature for any of your asset types
   - `Mimetype`, if you are using Sites DocLink, flex filter assets, the ImageFile asset type, or if you are using this table for any of your custom asset types.
   - `Filters`, if you are using flex filters on your site. (The `Filters` table lists the classes that are used by the flex filters. You will need to copy `jar` files and classes manually.)

- Any other auxiliary tables (such as lookup tables) for your asset types.

> **Note**
>
> Only five fields are provided for table names. If you have to enter more than five table names, complete this procedure through step 6 for the first five tables, then repeat steps 3–6 for the remaining tables.

6. Click **Mirror**.

   If the initialization is successful, WebCenter Sites displays a confirmation message; otherwise, an error message is displayed.

   Based on the sites that you selected in step 4, the corresponding rows from the `Publication`, `SitePlanTree`, and `AssetType` tables are copied to the destination.

   Additionally, all the rows in the tables that you specified as auxiliary tables are copied to the destination.

> **Note**
>
> You can also initialize the target destination from the **Publish Console**:
>
> 1. In the button bar, click **Publishing**.
> 2. In the **Publish Console**, select your mirror destination from the drop-down list and then click **Select Destination**.
> 3. Click the **Create Site on Target** button. When the confirmation screen appears, click **Yes**. This creates the basic site information on the destination system.

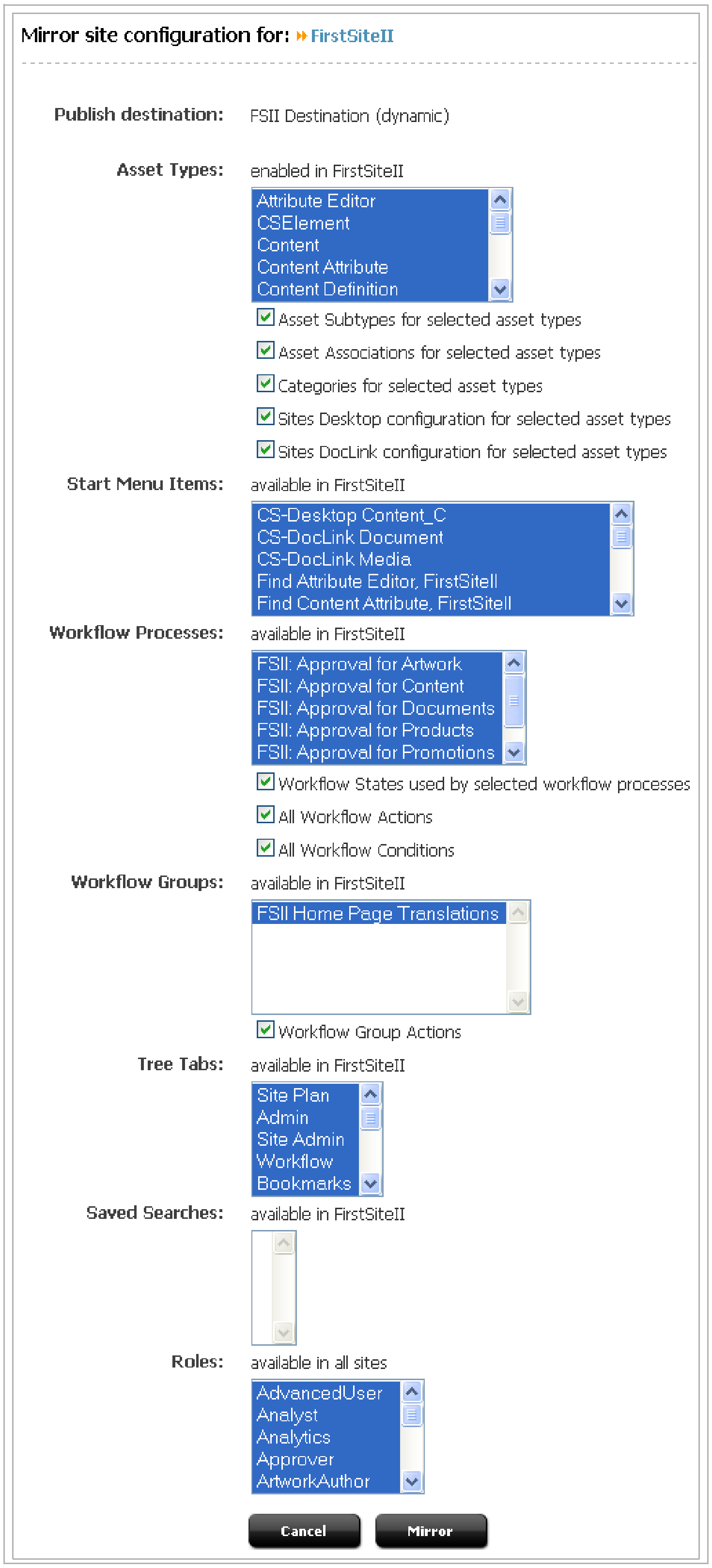
Step 7: Configure the Mirror Destination

Now you need to configure the data that will be on your target destination. This configuration will vary depending on the purpose of the target destination. For example, if the target destination is strictly a delivery machine, it makes sense to only include asset types when mirroring. Before proceeding with configuring the destination, you may find it helpful to refer to “Migrating a Site from One System to Another,” on page 314.

> **Note**
>
> If Revision Tracking is turned on at the publishing target destination for either ElementCatalog or SiteCatalog, then the publishing of template will fail and the template may become corrupted.
>
> It is not recommended to enable Revision Tracking on the target destination.

1. In the in the **Admin** tab on the source system, expand **Sites** and double-click the site whose data you want to publish on the target destination.
2. In **Publish Destinations** (near the bottom of the screen), click **Mirror site configuration for *destination* (dynamic)**, where ***destination*** is the target destination.

WebCenter Sites displays the “Mirror site configuration” form.



Select the data you wish to make available on the target destination:

- **Asset Types**: Select the asset types that you want to make available on the target destination.

  If your asset types have subtypes and categories, choose the appropriate options from the list in the **Asset Types** section and select the corresponding check boxes. For example, the `AssocNamed` table contains information about associations for asset types. The table will be mirrored only if you select **Asset associations for selected asset types** and then the relevant asset types.

  Select the Sites Desktop / Sites DocLink configuration check boxes to configure the selected asset types for Sites Desktop and Sites DocLink.

- **Start Menu Items**: Select any start menu items that you or the developers designed for your content providers to use.
- **Workflow Processes**: If there are workflow processes for your asset types, select these processes and the appropriate workflow items.
- **Workflow Groups**: Select any workflow groups that you or the developers designed for your content providers to use.
- **Tree Tabs**: Select the tree tabs that you want to make available on the target destination.

> **Note**
>
> If any of the following tree tabs already exist on the target destination, mirroring the tree tabs will fail and potentially corrupt the tree tabs on the destination: Site Plan, Admin, Site Admin, Workflow, Bookmarks.
>
> It is not recommended to mirror tree tabs that already exist on the target destination.

- **Saved Searches**: Select any saved searches that you or the developers designed for your content providers to use.
- **Roles**: Select the roles you want to exist on the target destination.

3. Click **Mirror**.

   Based on which asset type configuration options you selected in step 2, appropriate rows from the `AssocNamed`, `AssocNamed_Subtypes`, and `Category` tables are copied to the destination.

   If you selected **Start Menu Items** or **Saved Searches**, appropriate rows from the tables that implement those features are copied to the destination.

   Based on which workflow configuration options you selected in step 2, appropriate rows from the workflow tables are copied to the destination.

4. Repeat steps 1–3 for each site whose assets you want to publish on the target destination.

Step 8: Approve Your Assets

To truly test your published site, you must approve and publish all the assets for the site. See "Approving Multiple Assets," on page 315 for help with approving many assets at once.

If you want to perform a simple test of your configuration, temporarily create a home page asset with just a few dependents.

Step 9: Publish the Assets

After you have approved assets that can be published to your destination, you can run a test publishing session:

1. On the source system, click **Publishing** in the button bar.
2. In the **Publish Console**, select your mirror destination from the drop-down list and click **Select Destination**.

   WebCenter Sites displays information about the assets that are ready to be published to this destination.
3. If for some reason the target destination has not yet been initialized, it must be initialized now or the publishing session will fail. To initialize the destination, click **Create Site on Target** in the **Publish Console**. This creates the basic site information on the destination system.
4. Click **Publish**.

   A confirmation message appears. Click **Yes** to proceed with the publish.

   The publishing system mirrors all the approved assets for this destination to the WebCenter Sites database on the destination system.
5. In the **Publish Console** Active tab you can observe that the session is running.
6. When the session completes, click the **History** tab to view the session summary for this session.

   Mouse over the session information to view summary information; click **View Log** to view the session log files. For information on configuring the logs to view destination information, see "Retrieving Logs From Delivery Sites Systems," on page 303.

Step 10: Test the Results

To test the results, point your browser to the home page asset on the destination system and examine the site.

If you have not already done so, you must determine the asset's URL. A WebCenter Sites URL is constructed by concatenating the following values:

- The hostname or IP address (and sometimes the port number) of the destination system.
- The CGI path, which the system obtains from the `ft.cgipath` property in the `futuretense.ini` file. For example, for WebLogic and other application servers with servlet architectures, this default path is `/servlet/`.
- The string `ContentServer?pagename=`
- A page name from a `SiteCatalog` entry.

- Additional information that is passed in with the WebCenter Sites page criteria variables, `c`, `cid`, `tid`, and `p` (see the *Oracle WebCenter Sites Developer's Guide* for information about these variables).

Complete the following steps to determine the URL of your home page and test the site:

1. Start the Property Editor and open the `futuretense.ini` file for the destination system. (If you need help with this step, see the *Property Files Reference*.)
2. In the **App Server** tab, locate the `ft.cgipath` property and write down its value.
3. In the **Compatibility** tab, locate the `ft.approot` property and write down its value.
4. Save the property file and close the Property Editor.
5. Open a text editor. Enter the server name, a slash (/), and then the cgipath. Precede the server name with the proper protocol—`http://` or `https://`—as in the following example:

   WebLogic and WebSphere:

   `http://bigfatsun.example.com:8080/servlet/`
6. At the end of the string, type a slash, and then add the following text:

   `ContentServer?pagename=`*`your_home_page`*

   Now the URL should look similar to the following examples:

   Example for WebLogic and WebSphere:

   ```
   http://bigfatsun.example.com:8080/servlet/
      ContentServer?pagename=ExampleSite/Page/Home
   ```
7. Point your browser to the URL you assembled.
8. Scan the page for errors and test all links to ensure they work.

> **Note**
>
> It is recommended that you conduct a complete test of the system under peak load conditions after you have mirrored the entire site for the first time, and at regular points thereafter.

Step 11: Set Up the Schedule

After you have ensured that your destination is configured correctly, you can finish configuring your publishing system by completing the following steps on the source system:

- Create scheduled publish events for the destination. For help with this step, see "Scheduling Publish Events," on page 319.
- If you are using images that are not assets in the design of your site—that is, your site designers want to store all images on the web server rather than manage them as assets —plan how you will move the image files from the web server for the management system to the web server for the delivery system. For example, you can set up a regular FTP transfer.
- If you are using elements and SiteCatalog page entries that are not CSElement and SiteEntry assets, you must use the CatalogMover tool to mirror them to the destination

system. For help with CatalogMover, see the *Oracle WebCenter Sites Developer's Guide*.

Step 12: Turn Off Asset Invalidation on the Delivery System

By default, the publishing system is configured to mark an asset as changed on the destination system when you publish an asset from one system to another (source to destination). Then, the newly-published asset must be approved on the destination system before it can be published to another destination.

The default configuration is appropriate for development and management systems. However, when you are publishing to the delivery system, there is no need for the publishing system to take the time to mark the change—assets are published to the delivery system but not from it.

Therefore, on the delivery system, turn off this publishing feature by completing the following steps:

1. Start the Property Editor and open the `futuretense_xcel.ini` file. If you need help with this step, see the *Property Files Reference*.
2. In the **Publishing** tab, locate the `xcelerate.publishinvalidate` property and set its value to `false`.
3. Save the file and close the Property Editor.

Approval Query for Performance Enhancement

Dynamically publishing thousands of assets takes a long time because of the time required to compute which assets are publishable for the given destination. The query is executed twice: When the user selects the publishing destination and when the user clicks the **Publish** button to publish the assets. The current query is also complex, given that it is written to work for both mirror and static publishing. To improve publishing performance, we simplified the query by considering only mirror publishing. The same query is used to improve the performance of RealTime publishing. The simplified query, with comments, looks as follows:

```
select pkt.localkey, pkt.urlkey from PubKeyTable pkt JOIN
   ApprovedAssets aa ON pkt.assetid=aa.assetid -- match asset id's
   WHERE aa.targetid=<targetid>
   AND pkt.targetid=<targetid> -- match target id's
   AND aa.tstate='A' -- asset must be approved
   AND (aa.lastassetdate IS NULL OR aa.assetdate >
   aa.lastassetdate) -- compare assetdate to lastassetdate
```

To implement the approval query

1. On the source WebCenter Sites system, create the file `Shared/Storage/pubquery.txt` and paste in the following query (same as above, but without comments):

```
select pkt.localkey, pkt.urlkey from PubKeyTable pkt JOIN
   ApprovedAssets aa ON pkt.assetid=aa.assetid
   WHERE aa.targetid=__TARGETID__
   AND pkt.targetid=__TARGETID__
   AND aa.tstate='A'
   AND (aa.lastassetdate IS NULL OR aa.assetdate >
   aa.lastassetdate)
```

2. Restart your source system and enable database debugging to verify that the query runs during mirror and RealTime publishing sessions. (To enable database debugging, go to the **Admin** tab, expand **System Tools**, and double-click **Configure log4j**. After making changes there, system restart is not necessary)
3. Inspect the logs to verify that the query ran during publishing and that all publishable assets were published correctly.
4. Resume normal operations (disable database debugging by setting `com.fatwire.logging.cs.db` to its normal value and restart the system).

Retrieving Logs From Delivery Sites Systems

WebCenter Sites provides a way to retrieve publishing-related logging information from the delivery system and display it in the Publishing console. To achieve this, WebCenter Sites inserts the publishing session ID into each relevant log entry. The management interface has the ability to retrieve the relevant entries for a publishing session and display them in the Log screen for that session.

To enable remote publishing logging

1. Activate the logger by setting the following property in `commons-logging.properties` on both source and target:

```
org.apache.commons.logging.LogFactory=com.fatwire.cs.core.
   logging.ContextAwareLogFactory
```

2. Add the following lines to your `web.xml` on the target system:

```
<filter>
      <filter-name>ContextHeaderFilter</filter-name>
      <filter-class>com.fatwire.cs.core.logging.context.
         filter.ContextHeaderFilter</filter-class>
</filter>
<filter-mapping>
      <filter-name>ContextHeaderFilter</filter-name>
      <servlet-name>ContentServer</servlet-name>
</filter-mapping>
```

3. On the target system, set the following `futuretense.ini property`:

```
log.file.location=<path to Sites log file>
```

Chapter 16

The Export Assets to XML Publishing Process

This chapter explains how Export Assets to XML works and how it must be configured.

This chapter contains the following sections:

The Export Assets to XML Publishing Method

The **Export Assets to XML** publishing method is a hybrid between the Export to Disk and dynamic publishing methods. Assets are rendered into files, but this publishing method uses the dynamic dependency calculation method rather than the method of Export to Disk publishing.

The Export Assets to XML publishing method differs from the other two in that it is really a data transformation method. While Export to Disk publishing creates one HTML file per page, which could mean that several assets are rendered into one file, Export Assets to XML creates **one XML file** for **each asset** that is approved to a destination configured to use this publishing method.

Export to Disk publishing creates Export to Disk files that are to be delivered from a Export to Disk Sites delivery system. In contrast, Export Assets to XML creates **XML files** for delivery to a database or non-Sites system.

When an Export Assets to XML publishing session runs, this is what happens:

1. On the source system, the approval system provides a list of the assets approved for this destination to Export Assets to XML.
2. Export Assets to XML renders each asset in the list to an XML file. The output file describes the asset, stating all the values for all of the asset's fields.

When you use the Export Assets to XML publishing method, typically you set up a quality assurance process to test the generated files before you move them to the external (non-Sites) system.

The XML Output

The output from Export Assets to XML is well-formed XML that describes an asset object in terms of its field values. Fields are described as attributes.

For each of the asset's fields, there is a name/value pair. The statement of the value includes the data type of the field. For example, the **Name** field holds characters. So the name/value pair for an asset's **Name** field would appear as follows in the output:

```
<attribute name="Name">
      <string value="nameOfAsset"/>
</attribute>
```

Flex attributes serve as the fields for flex assets. In the output XML for a flex asset, the flex attribute values are prepended with `Attribute_`. Here's an example of an attribute value for the price of a sample site product asset:

```
<attribute name="Attribute_price">
      <decimal value="5.9"/>
</attribute>
```

Note that the XML output uses the column names rather than the field name of each field/attribute and that column names and field names can be different.

The following is a description of the XML file created by this publishing method, presented as a pseudo-dtd file

```
<!-- ASSET:
-- ASSET defines an asset object
-- an asset is made up of attributes
```

```
-->
<!ELEMENT ASSET (ATTRIBUTE)*>
<!ATTLIST ASSET TYPE CDATA #REQUIRED>
<!ATTLIST ASSET ID CDATA #IMPLIED>
<!ATTLIST ASSET SUBTYPE CDATA #IMPLIED>

<!ELEMENT ATTRIBUTE (STRING | DATE | INTEGER | DECIMAL | BINARY |
FILE | ASSETREFERENCE | ARRAY | STRUCT | LIST)>
<!ATTLIST ATTRIBUTE NAME CDATA #REQUIRED>

<!ELEMENT STRING>
<!ATTLIST STRING VALUE CDATA #REQUIRED>

<!ELEMENT DATE>
<!ATTLIST DATE VALUE CDATA #REQUIRED>

<!ELEMENT INTEGER>
<!ATTLIST INTEGER VALUE CDATA #REQUIRED>

<!ELEMENT DECIMAL>
<!ATTLIST DECIMAL VALUE CDATA #REQUIRED>

<!ELEMENT BINARY>
<!ATTLIST BINARY VALUE CDATA #REQUIRED>

<!ELEMENT FILE (CDATA) >
<!ATTLIST FILE NAME CDATA #REQUIRED>

<!ELEMENT FILE>
<!ATTLIST FILE NAME CDATA #REQUIRED>

<!ELEMENT ASSETREFERENCE>
<!ATTLIST ASSETREFERENCE TYPE CDATA #REQUIRED>
<!ATTLIST ASSETREFERENCE VALUE CDATA #REQUIRED>

<!ELEMENT ARRAY (STRING | DATE | INTEGER | DECIMAL | BINARY | FILE
| ASSETREFERENCE | ARRAY | STRUCT | LIST)+>

<!ELEMENT STRUCT (FIELD)+>

<!ELEMENT FIELD (STRING | DATE | INTEGER | DECIMAL | BINARY | FILE
| ASSETREFERENCE | ARRAY | STRUCT | LIST)+>
<!ATTLIST FIELD NAME CDATA #REQUIRED>

<!ELEMENT LIST (ROW)+>

<!ELEMENT ROW (COLUMN)+>

<!ELEMENT COLUMN ( STRING | INTEGER | DECIMAL | DATE |
ASSETREFERNCE)>
<!ATTLIST COLUMN NAME CDATA #REQUIRED>
```

File Naming Conventions for Export Assets to XML

Exported XML files are named according to the following convention:

assetID.xml

For example, if the asset ID is 3344556677, the file name for the asset's XML file is: 3344556677.xml

The Export Assets to XML publishing method does not use any information that is entered in an asset's **Path** or **Filename** field.

Configuring Your System for Export Assets to XML

The main steps when configuring for the Export Assets to XML delivery type are as follows:

- Step 1: Create the Batch User Account (If One Does Not Exist)
- Step 2: Specify the Root Export Directory
- Step 3: Configure an Export Assets to XML Destination
- Step 4: Approve Your Assets
- Step 5: Publish and Test the Results
- Step 6: Set Up the Schedule

Step 1: Create the Batch User Account (If One Does Not Exist)

Creating a batch user account is required before any type of publishing can take place. This procedure needs to be performed only once, regardless of the number and types of publishing destinations you are creating.

- If you have not already done so, create the batch user account. For instructions, see "Step 1. Create the Batch User Account (If One Does Not Exist)," on page 273.
- If a batch user account already exists for your installation, skip to "Step 2: Specify the Root Export Directory."

Step 2: Specify the Root Export Directory

As mentioned earlier in this chapter, the directory to which WebCenter Sites exports files is determined by the `cs.pgexportfolder` property in the `futuretense.ini` file.

To set the root directory for exported files

1. Start the Property Editor. If you need help with this step, see the *Property Files Reference*.
2. Open the `futuretense.ini` file on the source Sites system.
3. On the **Export/Mirror** tab select the `cs.pgexportfolder` property and set its value to the full path of the directory to which files should be exported. This is a global setting for the system.

   Save the file and close the Property Editor.

4. Restart the application server.

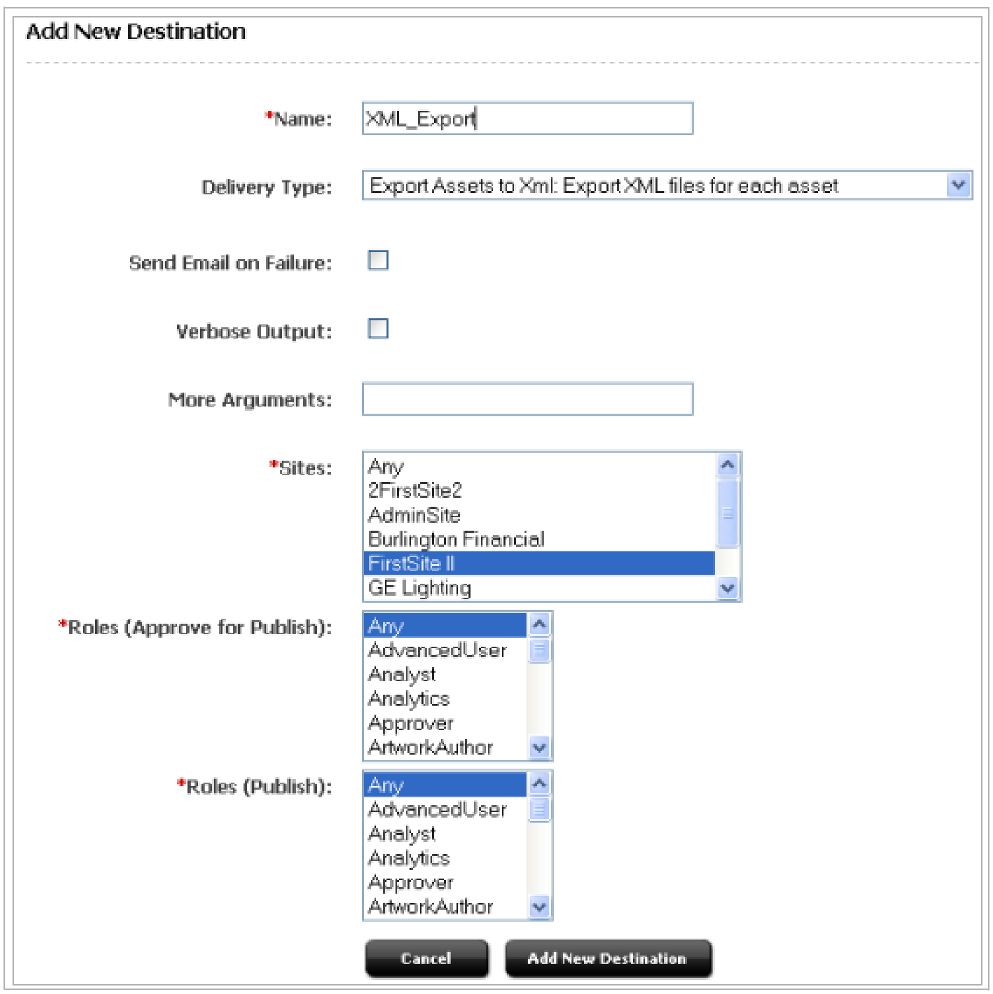
> **Note**
>
> Before you run an Export to XML publishing session, make sure that this destination directory exists and that is has enough space for the XML files that will be created there.

Step 3: Configure an Export Assets to XML Destination

Next, create the Export to XML destination by completing the following steps:

1. In the **Admin** tab on the source Sites system, expand **Publishing,** then **Destinations**.
2. Double-click **Add New**.

   The “Add New Destination” form appears:



   Fill in the fields as described below:

   - **Name:** Enter a unique name for the destination.
   - **Delivery Type:** Select Export Assets to XML: Export XML file for each asset.

- **Sites:** Select the sites whose assets can be approved for and published to this destination.
- **Roles (Approve for Publish):** Select the roles to which you are assigning asset approval privileges. All users who are assigned these roles can approve assets.
- **Roles (Publish)**: Select the roles to which you are assigning publish privileges. All users assigned these roles can publish.

3. Repeat step 2 for each additional export destination that you need to configure for your source system.

Step 4: Approve Your Assets

If you want to perform a simple test of your configuration, select an asset and approve it and any of its dependent assets.

If you want to complete a test publish of your entire site, approve all the assets for the site. See "Approving Multiple Assets," on page 315 for help with approving many assets at once.

Step 5: Publish and Test the Results

After you have approved assets that can be published to your destination, you can run a test publishing session:

1. On the source Sites system, click **Publishing** in the button bar.
2. In the **Publish Console**, select your destination from the drop-down list and then click **Select Destination**.

   WebCenter Sites displays information about the assets that are ready to be published to this destination.
3. Click **Publish**.

   A confirmation message appears. Click **Yes** to proceed with the publish.

   WebCenter Sites exports all assets approved for this destination to files. The files are stored in the directory you specified earlier.
4. In the **Publish Console** Active tab you can observe that the session is running.
5. When the session completes, click the **History** tab to view the session summary for this session.

   Mouse over the session information to view summary information; click **Log** to view the session log files.
6. The log file displays a list of all the resulting XML files, verify that the files are correct.

Step 6: Set Up the Schedule

After you have ensured that your destination is configured correctly, that the directory names are created according to your needs, you can finish configuring your publishing system by completing the following steps on the source system:

- Create scheduled publish events for the destination. For help with this step, see "Scheduling Publish Events," on page 319.

- Plan how you will move the generated files to your external, non-Sites systems. For example, you can set up a regular FTP transfer that automatically moves files to a testing area after a publishing session completes.

Chapter 17

Additional Publishing Procedures

In addition to the configuration steps described in the preceding sections of this chapter, administrators and developers also perform, whenever necessary, the publishing procedures described in this chapter.

This chapter contains the following sections:

Migrating a Site from One System to Another

During the development phase for your sites, your site designers and developers develop sites and asset types with all their supporting configuration (subtypes, associations, start menu items, and so on), code templates, and assist the administrators in customizing the user interface, when necessary, by writing elements for workflow and so on. Your site designers and developers do this work on the development system.

When the site is ready for the content providers, you can use the **Mirror Destination Configuration** feature to migrate it to the management system and the delivery system.

Moving a Site from a Development System to a Management System

Complete the following procedure **once**:

1. Log in to the WebCenter Sites interface on the management (destination) system as the **fwadmin** user.
2. If you have any custom support tables—a lookup table for a field in an asset type, for example—create those tables on the management (destination) system.
3. On the management system, create all the users.
4. On the source system, log in to the WebCenter Sites interface.
5. Create a **Publishing Destination** for the destination system that uses the RealTime publishing method.

   For help with this step and the following two steps, see "Configuring Your System for RealTime Publishing," on page 351.
6. Initialize the publishing destination.
7. Configure the publishing destination.
8. On the destination (management) system, configure the users for the site. For help with this step, see "Granting Users Access to a Site (Assigning Roles to Users)," on page 112.
9. If you created a flex family, log back in to the source system. Approve and then publish all of the data structure flex assets to the management (destination) system.

   You can either approve each asset individually or use the **Approve Multiple Assets** feature on the **Admin** tab. For information about this feature, see "Approving Multiple Assets," on page 315.
10. Approve and then publish all of the rest of the appropriate assets for the destination system: template, page, collection, query, CSElement, and SiteEntry assets, and so on.
11. If there are any elements or page entries in the `SiteCatalog` table that are not CSElement or SiteEntry assets, use CatalogMover to mirror those items to the management system. For help with CatalogMover, see the *Oracle WebCenter Sites Developer's Guide.*

Moving a Site to a Delivery System

When you move a site to a delivery system, there is no need to mirror start menu items, workflow, saved searches, and so on.

Complete the following steps:

1. Log in to the WebCenter Sites interface on the delivery (destination) system as the **fwadmin** user.
2. If you have any custom support tables—a lookup table for a field in an asset type, for example—create those tables on the delivery (destination) system.
3. On the source system, log in to the WebCenter Sites interface.
4. Create a **Publishing Destination** for the destination system that uses the Mirror to Server publishing method.

   For help with this step, see "Step 5: Create a Mirror Destination," on page 293.
5. Initialize the mirror destination.

   For help with this step, see "Step 6: Initialize the Destination," on page 296.
6. Configure the mirror destination.

   For help with this step, see "Step 7: Configure the Mirror Destination," on page 297.

   Do **not** select start menu items, saved searches, or any of the workflow options.
7. If you created a flex family, log back in to the source system. Approve and then publish all of the data structure flex assets to the management (destination) system.

   You can either approve each asset individually or use the **Approve Multiple Assets** feature on the **Admin** tab. For information about this feature, see the "Approving Multiple Assets," on page 315.
8. Approve and then publish all of the rest of the appropriate assets to the destination (delivery) system: template, page, collection, query, CSElement, and SiteEntry assets, and so on.
9. If there are any elements or page entries in the `SiteCatalog` table that are not CSElement or SiteEntry assets, use CatalogMover to mirror those items to the management system. For help with CatalogMover, see the *WebCenter Sites Developer's Guide.*
10. If there are any images in the online site that are not assets, be sure to copy them to the delivery system.

Approving Multiple Assets

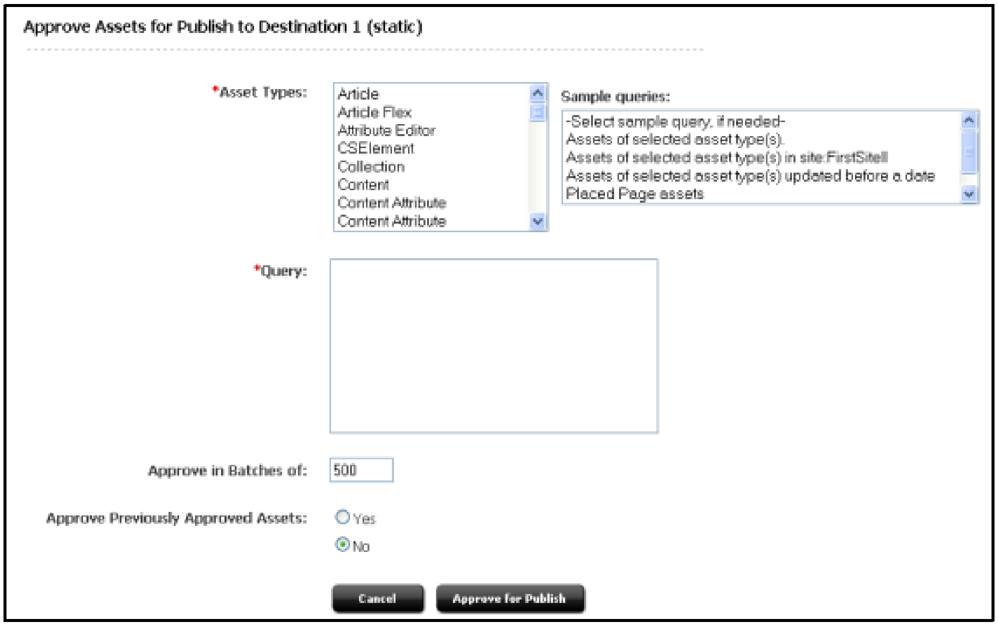
Each publishing destination has an option to **Approve Multiple Assets**. It is especially useful for upgrades and for the first publishing session to a destination.

Note that the Approve Multiple Assets feature is **not** the BulkApprover utility. The BulkApprover utility approves only those assets that have been imported into the WebCenter Sites database with the BulkLoader utility. Both BulkLoader and BulkApprover are described in the *Oracle WebCenter Sites Developer's Guide*.

To approve a group of assets, follow these steps:

1. In the **Admin** tab, expand **Publishing**, then expand **Destinations**, and then expand the destination that you want to approve assets for.
2. Under that destination, double-click the **Approve Multiple Assets** option.

   The "Approve Assets" form appears:



3. In the **Asset Types** section, from the list on the left, select the asset types that you want to approve.
4. From the **Sample Queries** list on the right, select a query. If the exact query that you need is not present, select the query that is most like the one that you want.

   WebCenter Sites creates a SQL query based on the items that you selected and displays it in the **Query** box.
5. (Optional) If necessary, edit the SQL query.
6. Click in the **Approve in Batches of** field and enter a numeric value. The default is set to 500.

> **Note**
>
> When you click the **Approve for Publish** button, the approval system approves batches of assets. The number of assets in each batch is determined by value in the **Approve in Batches of** field. Because the approval process is not a background process, it is possible that the browser session could time out while the batch is being approved.
>
> When using the **Approve Multiple Assets** feature, to ensure that the session does not time out, adjust the following settings:
>
> - The `cs.timeout` property in the `futuretense.ini` file, which sets the browser session timeout value. Start by setting this value to 1800 (which means 30 minutes). (See the *Property Files Reference*.)
> - The value you specify in the **Approve in Batches of** field. Start by using the default of 500 and lower it if necessary.

7. Specify the **Approve Previously Approved Assets** option as follows:

- If you want WebCenter Sites to ignore all previous approvals and re-approve all selected assets, select **Yes**. This is the option to use if previous approvals may have become invalid for some change such as a change in default templates for that destination.
- If you want WebCenter Sites to approve only the assets that have not yet been approved, select **No**. This is the correct choice when you are sure that previously approved assets are still valid.

8. Click **Approve for Publish**.

Creating Destinations

The procedures for creating destinations are described in the previous section. For information about creating new destinations, use one of the procedures in the following list, as appropriate:

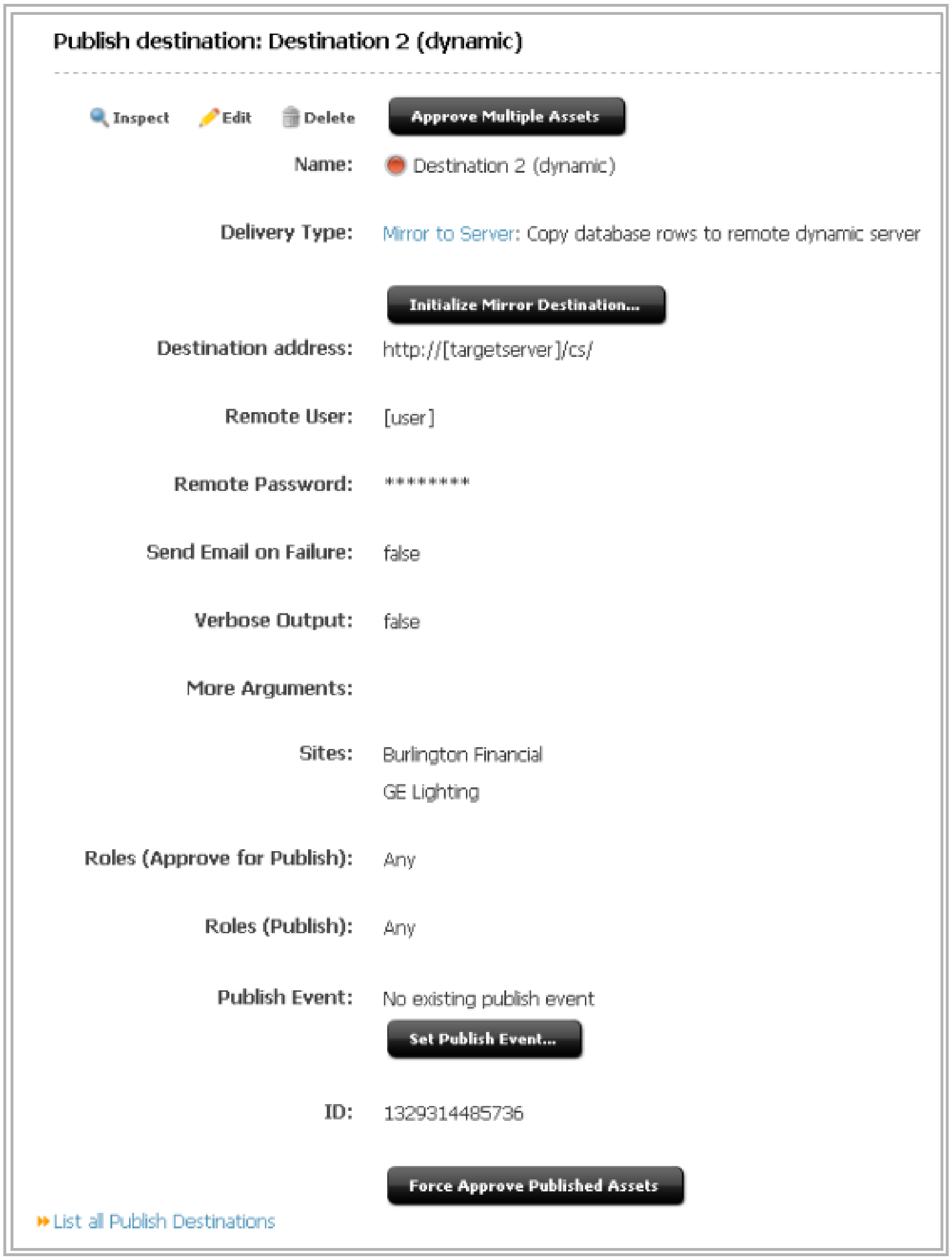
- To create an export destination, see "Step 3. Configure an Export Destination," on page 275.
- To create a mirror destination, see "Step 5: Create a Mirror Destination," on page 293.
- To create an export assets destination, see "Step 3: Configure an Export Assets to XML Destination," on page 309.

Editing Destinations

To edit a destination

1. In the **Admin** tab, expand **Publishing**, then expand **Destinations**.
2. Double-click on the destination that you want to edit.

   The Publish destination screen appears for the selected destination.



3. Click the **Edit** icon, and then make the appropriate changes.

   For information about the values that can be entered in the fields in this form, see the delivery-type specific procedures in the previous sections:

- For an Export to Disk destination, see "Step 3. Configure an Export Destination," on page 275.
- For a Mirror to Server destination, see "Step 5: Create a Mirror Destination," on page 293.

- For an Export Assets to XML destination, see "Step 3: Configure an Export Assets to XML Destination," on page 309.

Deleting Destinations

To delete a destination

1. In the **Admin** tab, expand **Publishing**, then expand **Destinations**.
2. Double-click on the destination that you want to delete.
3. Click the **Delete** icon.

   WebCenter Sites displays a verification message.
4. Click **Delete Destination**.

Creating Export Starting Points

Export starting points are defined in "Export Starting Points," on page 272. For information about creating them, see "Step 5. Create the Export Starting Points," on page 280.

Scheduling Publish Events

You can set up publishing events for all of your publishing destinations. Because publish events run as background processes and there is no publish queue, you can set up publish events for different destinations that run at the same time, if necessary.

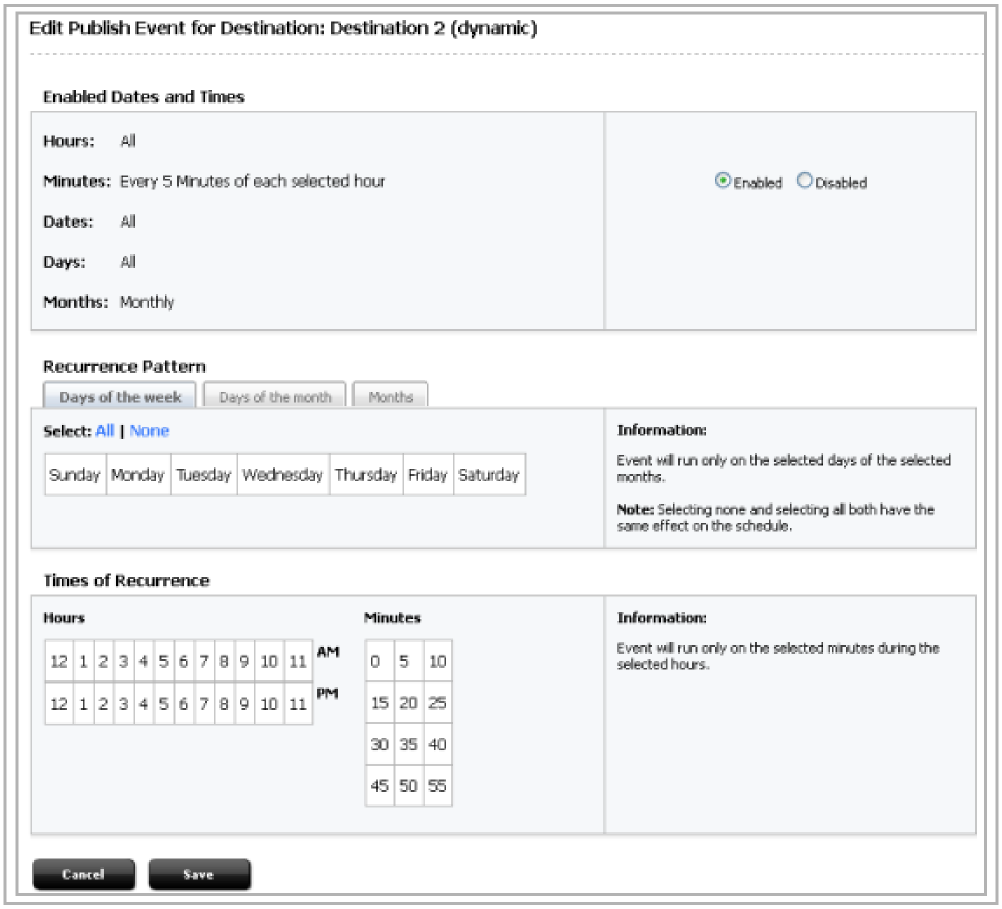
Note the following:

- There can be only **one** publish event schedule for **each** destination.
- If any of your systems serve as both source and destination, be sure that incoming and outgoing publishing sessions **do not overlap**. For example, you publish from the development system to the management system and you publish from the management system to the delivery system. Never schedule a publish event from the development system to the management system for a time when the management system is publishing to the delivery system.

To schedule a publish event, complete the following steps:

1. In the **Admin** tab, expand **Publishing**, then expand **Destinations**.
2. Double-click on the destination that you want to schedule a publish event for.
3. In the "Publish Destination" form, click the **Set Publish Event** button.

WebCenter Sites displays the "Edit Publish Event" form:



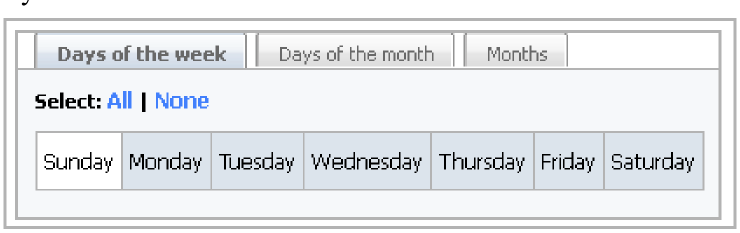
Now you select days, months, hours, and minutes to set a schedule.

The screen for setting a publish event is similar to the screen for setting a timed action event in workflow. See "Setting Up the Timed Action Event," on page 184 for more information.

Example Schedule

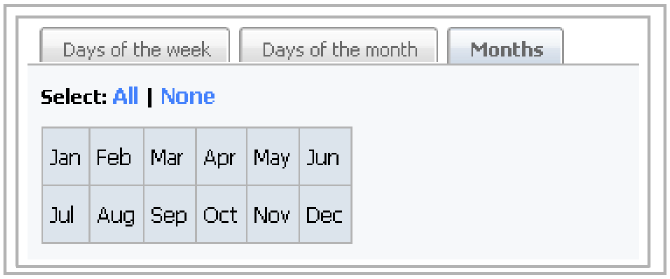
For example, if you want to set a schedule so that WebCenter Sites publishes the assets approved for this destination at 7 a.m., 4 p.m., and 8 p.m. every day of the week except Sunday, you would complete these steps:

1. In the **Recurrence Pattern, Days of the week** tab, click on **Monday** and drag to **Saturday** to select all days of the week except Sunday. The days highlight in gray as you select them.

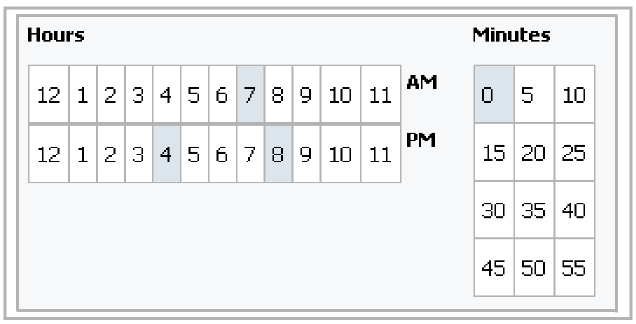


2. Skip the **Days of the month** tab, since you have already selected the days you want.

3. In the **Months** tab, click **All** to select all months. All months become highlighted in gray.

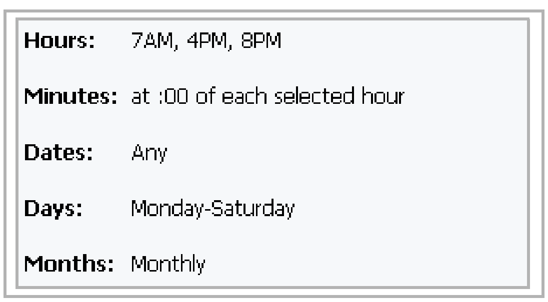


4. In the **Times of Recurrence** section, **Hours** selector, click on **7am**, **4pm,** and **8pm**.
5. In the **Minutes** selector, click on 0.



6. Double-check the dates and times selected in the **Enabled Dates and Times** section of the screen.

   It should look like the sample shown below:



7. In the Enabled Dates and Times section, click **Enabled**.
8. Click the **Save** button.

   A summary of the schedule now appears in the **Publish Event** section of the “Publish Destination” form.

Reading the Schedule Abbreviations

Once scheduled events are created, these events are displayed in string format in the WebCenter Sites interface.

In the "Edit Publish Events" screen the schedule is displayed in easy-to-read format on the Enabled Dates and Times section at the top of the page. In the "Publish Destination" screens, the schedule is displayed in the Publish Event field as a string.

Here is the key to the schedule string:

```
hours:minutes:seconds weekdays/days of month/months of year
```

This information is displayed as follows:

- Hours are displayed as numbers from 0 (midnight) through 23 (11 p.m.) and are separated from the minutes with a colon.

  When multiple hours are selected, the selected hours display with commas separating them.

  In the previous example (see "Example Schedule," on page 320), 7 a.m., 4 p.m., and 8 p.m. were set as the times to publish. This displays as **7,16,20:**

- Because you can set minutes in increments of 5, minutes are displayed as numbers 0 – 55 in five minute increments. Minutes are separated from the seconds with a colon.

  If the schedule describes more than one minutes increment, the scheduled increments are listed with commas separating them.

  When minutes are displayed as a **0**, it means the event runs on the hour.

  The hours and minutes for the example are displayed like this: **7,16,20:0:**

- Seconds are always set to zero. Therefore, the complete expression of the time in the display of the example is: **7,16,20:0:0**

- The time and the days are separated with a space.

  - Days of the week are expressed as numbers 0 (Sunday) through 6 (Saturday) and end with the slash (/) character.
  - If either no days or all days were selected, the publish event runs every day of the week. An asterisk (*) character appears in the field in that case.
  - If more than one day is scheduled, the selected days are displayed with commas separating them.

  The example schedule includes all the days of the week except Sunday. Therefore, the days of the week are appended to the display as follows: **7,16,20:0:0 1,2,3,4,5,6/**

- Days of the month are also displayed as numbers (1–31) and end with the slash (/) character.

  - The list of days is separated with a comma.
  - If no days of the month were selected, the publishing event runs every day of the month. An asterisk (*) character appears in the field in that case.
  - If you specify both days of the week and days of the month, the event runs when either setting matches the current day.

  The example schedule does not specify days of the month. Therefore, the value for this item is appended to the display as an asterisk: **7,16,20:0:0 1,2,3,4,5,6/*/**

- At the end of the string are the months, displayed as numbers from 1 (January) through 12 (December).
    - The list of months is separated with a comma.
    - If all the months are selected, the publishing event will run every month. An asterisk (*) character appears in the field in that case.

  The example schedule specified all the months. Therefore, the final result is:

  7,16,20:0:0 1,2,3,4,5,6/*/*

Editing Publish Events

To edit a publish event, complete the following steps:

1. In the **Admin** tab, expand **Publishing**, then expand **Destinations**.
2. Double-click on the destination that you want to schedule a publish event for.
3. In the “Publish Destination” form, click **Set Publish Event**.

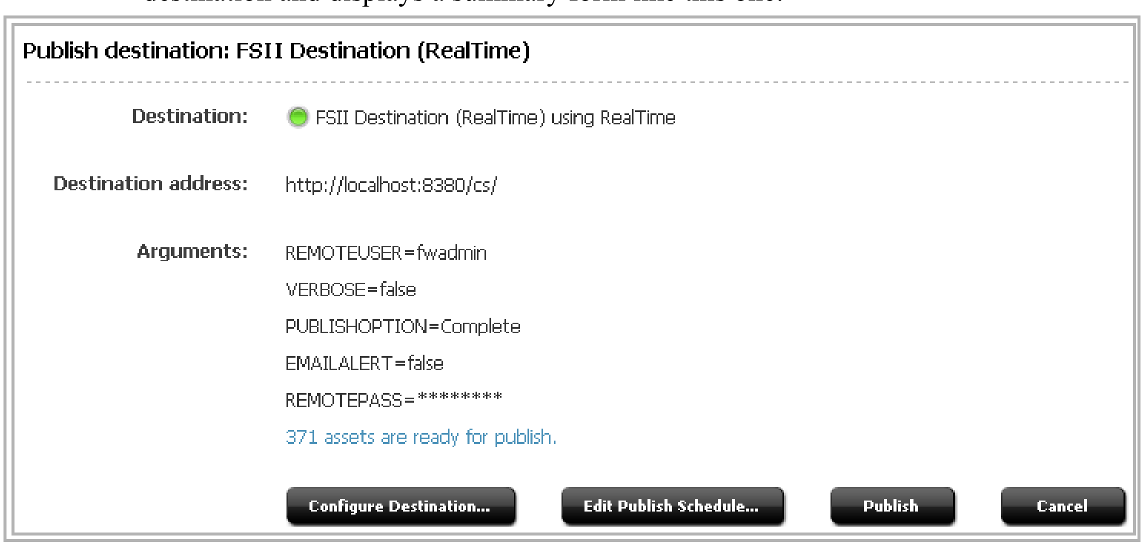
   WebCenter Sites displays the “Publish Event” form. The text at the top of the form describes how the event is currently configured.
4. Edit the event as needed and click **Schedule**. To clear your selections, click **Cancel**.

Overriding the Schedule

If you want to start a publishing session immediately, complete the following steps:

1. In the WebCenter Sites interface, click the **Publishing** button in the toolbar.
2. In the **Publishing Console**, from the drop-down list, select the destination that you want to publish to.
3. Click the **Select Destination** button.

   WebCenter Sites determines whether there are any assets that can be published to the destination and displays a summary form like this one:



If there are assets that can be published to the destination, the **Publish** button is displayed. If there are no assets to publish, there is no **Publish** button.

4. Click **Publish**.

   WebCenter Sites starts the publishing session.

   Note that if a publishing session for this destination is already underway, the publish event that you just tried to run fails and WebCenter Sites displays a status message describing that another session for the same destination is in progress. If you still want to run this event, wait until the current session completes and then repeat this procedure.

Assigning Approval or Preview Templates

When assets are approved for a Export to Disk publishing destination, the approval system examines the template assigned to the asset to determine its dependencies.

However, when the system publishes the asset, it does not necessarily use the template that is assigned to the asset. Why? Because the code in another element could determine that a different template is used for that asset in certain cases.

Consider the Burlington Financial sample site. An article asset from this sample site can be rendered by several different templates, depending on the context.

So when someone approves an article asset for the Burlington Financial sample site, which template should the approval process use to determine the dependencies for the article? The one that contains the most representative set of dependencies for all of the templates.

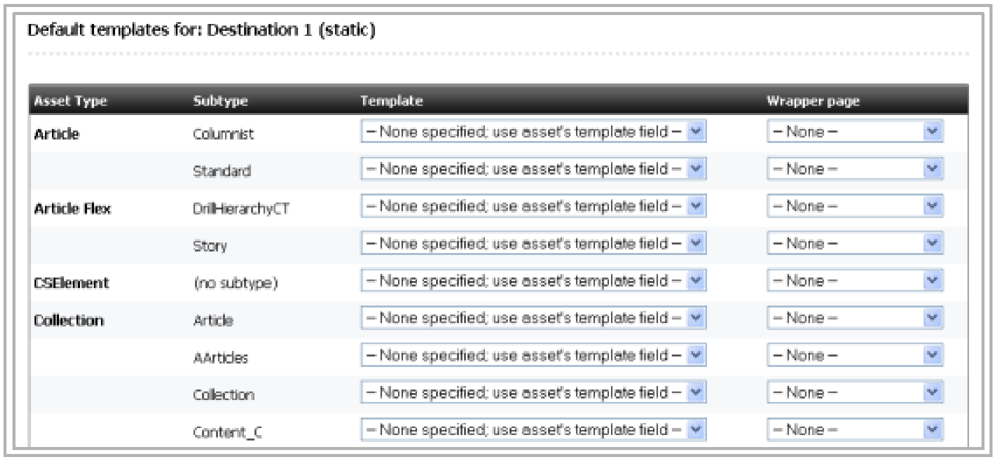
What if the template that contains the most representative set of dependencies is not the template that you want to assign to the asset? Set it as the Default Approval Template for assets of that type.

Note that when you assign default templates to assets that are published to mirror destinations, those templates are not used to calculate dependencies, but they are used when someone previews the asset from the “Status” form.

To set a default template, complete the following steps

1. In the **Admin** tab, expand **Publishing**, then expand **Destinations**.
2. Expand the destination that you want to configure default templates for and then select the Set Default Templates item.
3. WebCenter Sites displays a “Default Templates” form.
4. Click **Edit Configuration**.

   The system displays a form like this one:



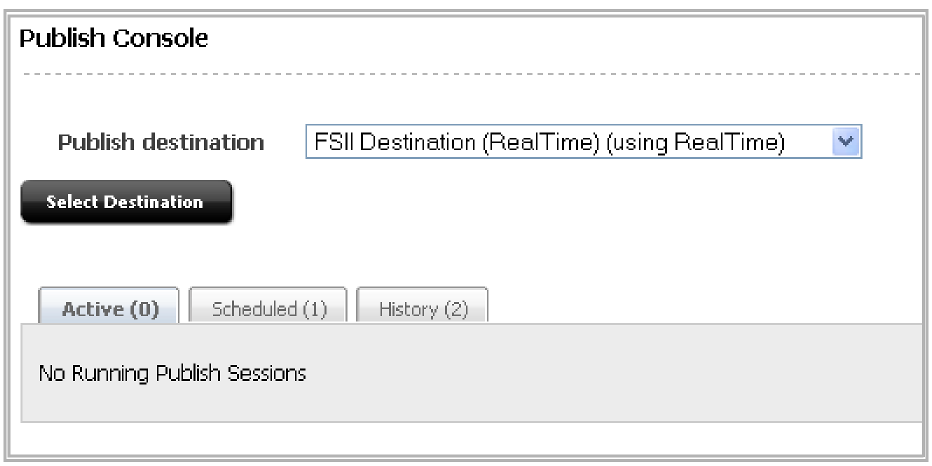
5. For each asset type that you need to configure, select the template and wrapper page that you want WebCenter Sites to use as the approval or preview template for assets of that type. If there are subtypes configured for an asset type, you can specify a default template for each subtype of that asset type.

6. Click the **Save** button.

   You are returned to the Default Templates screen for that destination.

Monitoring Sessions in the Publishing Console

You use the **Publish Console** to monitor publishing sessions. To open the **Publish Console**, click the **Publishing** button at the top of the WebCenter Sites interface:



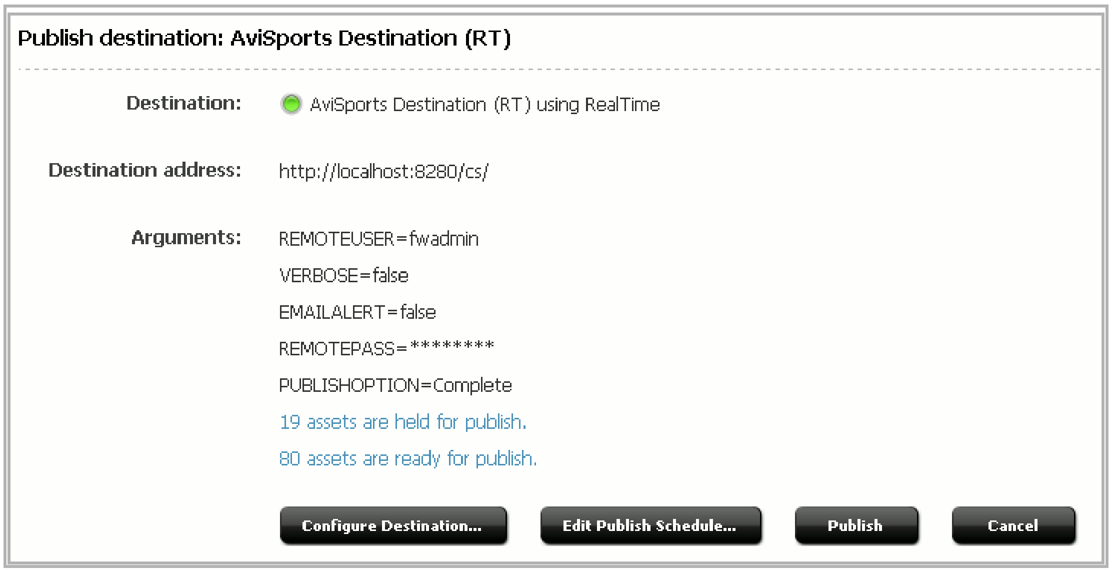
The console displays a summary of all the publishing activity, including any sessions that are currently running, scheduled, or completed.

Verifying Publishing Readiness

To determine whether a publishing session is ready or whether there are any problems with unapproved assets, follow these steps:

1. In the Publish Console, select the destination from the drop-down list.
2. Click the **Select Destination** button.

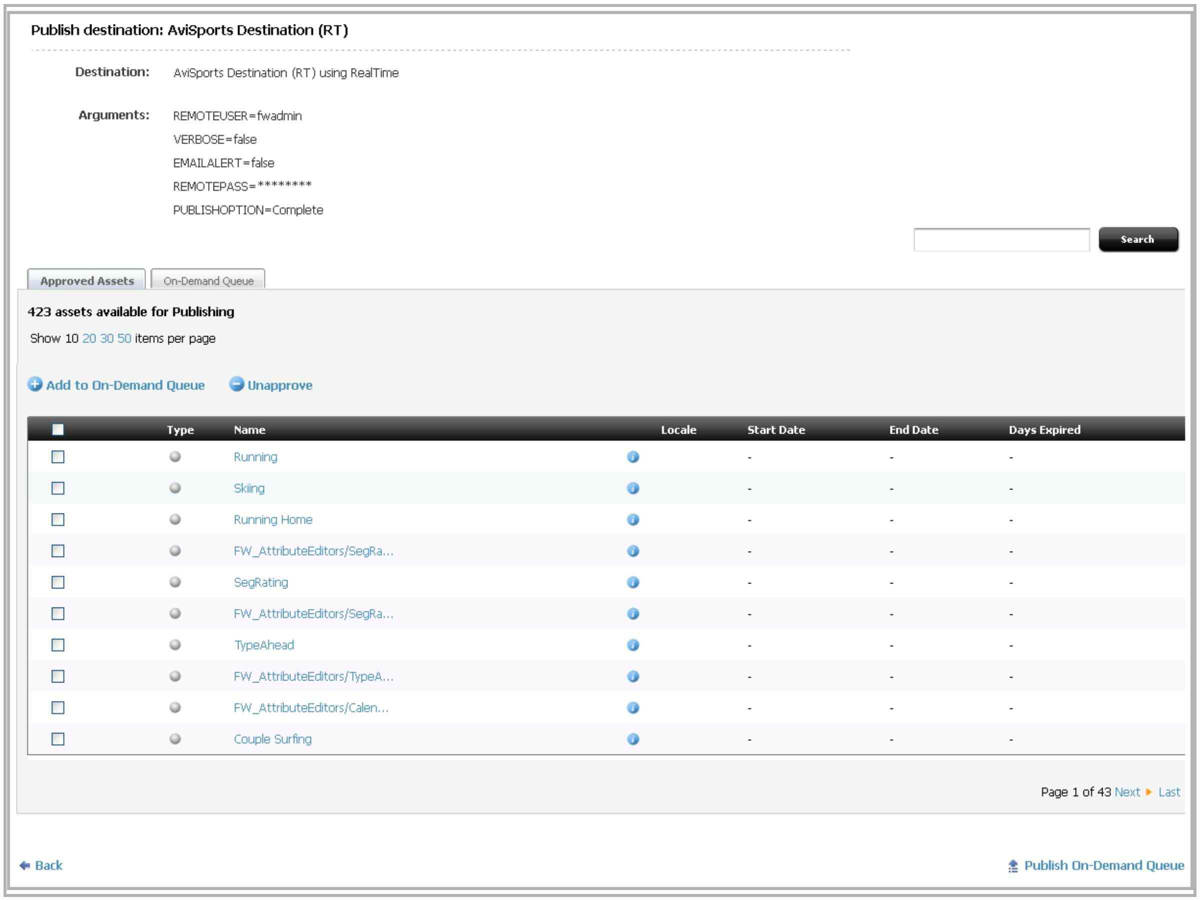
   A summary form appears. It lists any assets that need to be published but are not ready (they are held) and assets that can be published. For example:



3. To see the details, click on the link.

   WebCenter Sites displays lists of the assets that are either approved or approved and still held for dependency reasons. On one tab is the list of Approved Assets, and the other tab are the assets that are in the On-Demand Queue.

   For example:



If you need to resolve an issue, click the link to the asset and make any changes that are necessary (and perhaps approve it):

- To place assets in the On-Demand Queue, select them and click **Add to On-Demand Queue**. To remove assets from the On-Demand Queue (from the On-Demand Queue tab) select the assets and click **Remove from On-Demand Queue**.
- To publish the assets in the On-Demand Queue, click **Publish On-Demand Queue** from either tab.
- For information about why assets can be held, see page 247.
- For procedures that describe how to work with assets, see the *Oracle WebCenter Sites User's Guide.*

Managing Publishing History Information

Publishing sessions that are completed are listed in the **Publish History** section of the **Publish Console**, which is displayed when you click the **Publishing** button. Each publish session in the list has a status; Running, Done, or Failed.

Each item in the list has two icons:

- The inspect icon (the circled letter "i"), which displays a summary of that session. Examining the summary is the first step in troubleshooting a failed session.

- The delete icon (the trash can), deletes the session. When you click this icon, WebCenter Sites deletes the row from the `PubSession` table, the publishing log file for the session, and any messages for the session from the `PubMessage` table.

  Note that the greater the number of publishing sessions listed in this section, the longer it takes WebCenter Sites to open the Publish Console.

If a session shows its status to be Failed, click the inspect icon, note any error messages, and then consult "Troubleshooting," on page 329.

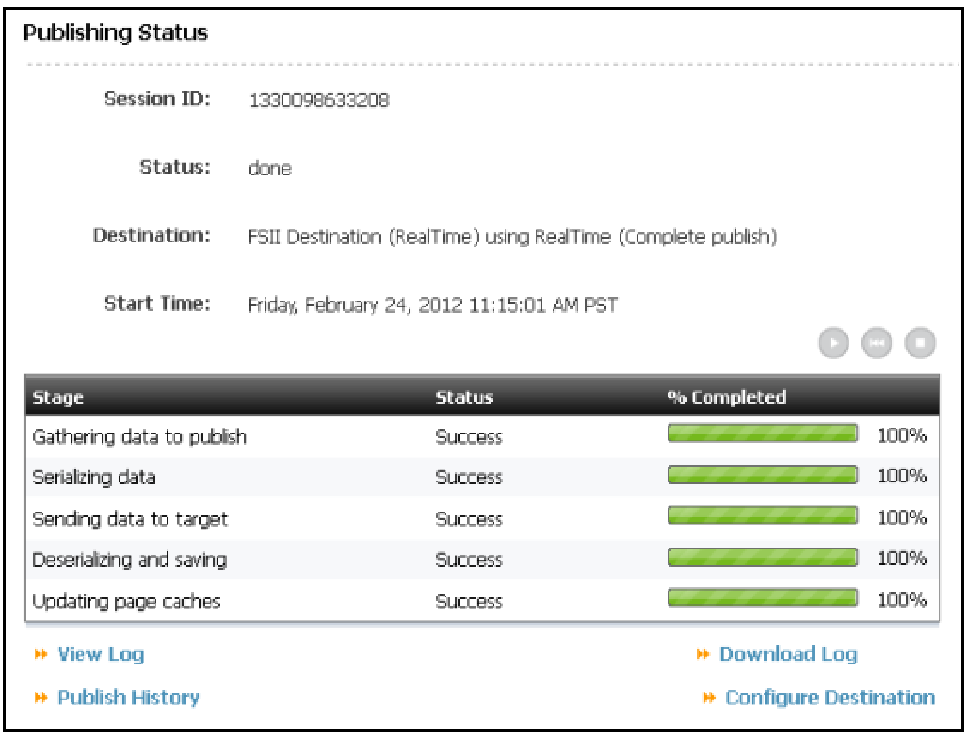
About Session History

The publishing history log files are displayed in the History tab of the Publishing Console as summaries of publishing sessions.

The text in a history file is a list of all the references that were published during that session. The term reference means something different for each publishing method:

- For Mirror to Server publishing, a reference is an asset. Each asset that is published is listed by its asset type and its ID.
- For Export to Disk publishing and Export Assets to XML, a reference is a generated file. Each file that is created is listed by its file name.

To see the Publishing History log, select the particular session from the History tab to view the Publishing Status screen for that session.



Click **View Log** to see the history log. You can also click **Download Log** to download the log and view it separately.

If there was a problem with the session, an error message is written to the `PubMessage` table and it is displayed as the history instead. If you see an error message in the history for a publishing session, see the section called "Troubleshooting," on page 329 and attempt to resolve the problem.

Publishing All Approved Assets

The publishing system conducts incremental publishing sessions. That is, during any session, only those assets that have been approved since the last session for this destination are published.

However, you may find a situation in which you need to publish all the approved assets in your entire site. (For example, after data repair on the delivery system.) When this is the case, use the **Force Publish All Items Next Time** button in the "Edit Destination" form.

When you click this button, it is the equivalent of changing the status of all approved assets to that of "never been published." That means that the next publishing session to the destination will publish all approved assets, regardless of whether they have already been published and have not changed since they were published.

Troubleshooting

This section describes error messages and other system indicators that reveal configuration, system, or data errors and suggests corrective actions for each. It contains the following topics:

- About Publishing System Error Messages
- Numeric Messages
- Other Indicators of System or Configuration Issues

About Publishing System Error Messages

The publishing system reports information about publishing sessions in the following ways:

- Displays a status for each session in the **Publish History** list in the **Publish Console**. If you see a status of "Failed," "Not Found," or "false," there was a problem with the publishing session.
- Writes information about the assets that were published to the log file for the publishing session. When you click the inspect icon for a publishing session in the Publish Console, the history log file for that session is displayed.

  The `request.folder` property in the in the `batch.ini` file defines where the publishing history log files are located. Each log file is named as follows:

  `PubSession ID` `+ Output.html`

  For example, if the PubSession ID for the session you are interested in is 9876544, then the log file is named `9876544Output.html`, and it is located in the directory specified by the `request.folder` property.
- Writes error messages about the publishing session to the `PubMessage` table. These messages are displayed in the **Messages** text area when you examine the publishing history in the Publishing Console for a specific publishing session.

You can use the PubSession ID to look up messages in the `PubMessage` table about a specific publishing session. Note that when you select the Verbose parameter on the destination configuration, status messages are also written to the `PubMessage` table.

- Writes other information about the session to the `PubSession` table. For example, to determine what time a publishing session started, look up the session by its PubSession ID in the `PubSession` table.
- Writes error messages to the `futuretense.txt` file on both the **source** and the **destination** (target) systems.

You can also find error messages about failed publishing sessions in the following locations:

- The application server log files on both the source and the destination systems.
- The `stdout` and `stderr` logs on both the source and the destination system.

Numeric Messages

This section contains descriptions and suggested corrective actions for the numeric error messages that the publishing system might report in any of the log files mentioned in the list above.

> **Note**
>
> The following list is not a complete list of all the error messages that your WebCenter Sites system can report. For a complete list of all error messages, see the error conditions appendix in the *Developer's Tag Reference*.

-2

This error indicates that either the mirror user name or password is not identified correctly. The publishing system on the source began the publishing process, but the destination system couldn't authenticate the user that is identified as the mirror user.

Corrective Action

Correct the mirror user information. See the procedure "Step 3: Identify the Mirror User to the Source System," on page 291 for help.

-3

This error indicates that the mirror user does not have the correct permissions to save the assets to the WebCenter Sites database on the destination system. In other words, the destination system user identified to the source as the mirror user does not have all the ACLs it needs.

Corrective Action

Edit the mirror user's account and assign it the appropriate ACLs.

See the procedure "Step 2: Set Up the Destination System," on page 290 for a list of the ACLs that the mirror user needs.

If you still cannot determine which ACL is missing from the mirror user's account, examine the futuretense.txt file on the destination system. There should be an entry that describes which table the mirror process failed to update and the name of the table may help you determine which ACL is missing.

For example, if the mirror session is failing on a visitor table, it is likely that your mirror user does not have the Visitor or VisitorAdmin ACL assigned to it.

-103

This error indicates that one of the tables whose data is being mirror published does not exist on the destination system. This error typically occurs when there is an asset type on the source that does not yet exist on the destination. It is also possible that someone created a custom table to support a function that was custom designed for your site but that table has not yet been created on the destination system.

Corrective Action

Examine the `sites.log` file on the **destination** system and look for the lines that list the -103 error. The text in the message should mention which table is missing.

Then use the appropriate tool to create the table on the destination system. That is, if it is a basic asset type, use AssetMaker. If it is a flex asset type, use Flex Family Maker. If it is a custom table, use Sites Explorer.

-611

This error message is one of the two generic mirror publishing error messages (the other is -12011). Typically -611 means that there was a problem when the publishing system tried to access the destination system and there should be other error messages reported, too.

Corrective Action

Examine the `sites.log` file and application server logs on the **destination** system to look for additional messages.

-612

This error message indicates that the definition of a mirror destination is incorrect in some way. Perhaps the syntax of the destination address is incorrect or there is a typographical error of some kind.

Corrective Action

Log into the WebCenter Sites interface on the **source**, examine the definition of the destination, and determine that it is correct. For help, see "Step 5: Create a Mirror Destination," on page 293.

-12011

This error message is one of the two generic messages (along with -611) which mean that the mirror process failed. It is unlikely that this message will appear without other error messages being reported. Typically the other errors will give more information about what went wrong.

Corrective Action

Examine the messages in the `PubMessage` table and the `sites.log` file on the **source** to look for additional error messages.

-12044

This error indicates that the publishing system could not begin the publishing session because it could not contact the destination to start the session. That is, there was no response from the destination; some part of the destination system is offline.

Corrective Action

Start by verifying that the web server and application server are running on the **destination** system. If they are not, start them. Verify that the two systems can connect to each other through the network (perhaps the network went off-line, for example).

-12045

This error indicates that the publishing system was able to start the session, but it then failed in some way.

Corrective Action

Examine the `sites.log` file on the **destination** system and look for error and exception messages in the log.

-12046

This error indicates that there was a problem at the end of a publishing session when the publishing system began to clean up the temporary tables that it creates for each session. When the cleanup process began, the connection with the destination failed. That is, the source system could get no response from the destination when the publishing process tried to clean up the temporary tables.

Corrective Action

Determine whether the web server and application server on the **destination** system are running. If they are not, start them. Check for network connection problems, as well.

-12047

This error indicates that the publishing system was able to begin the cleanup process but it then failed in some way during the operation.

Corrective Action

Examine the `sites.log` file on the destination and look for error exception messages and exceptions that describe the problem.

-13054

This error occurs for complex and flex assets only. In this case, the publishing system began publishing a complex or flex asset but the data did not reach the destination. It can mean that the data itself was corrupt or that the HTTP request connection failed.

Corrective Action

Check the `sites.log` file on the source system. If the problem was the data itself, there will be additional messages in this log file that can help you determine which asset caused the problem. However, if the HTTP request was dropped before the destination system could respond, there will **not** be an additional message in the log.

-13055

This error occurs for complex and flex assets only. In this case, the data for the flex or complex asset was delivered, but the destination system could not save the asset to its WebCenter Sites database.

This message indicates that there is a problem with your data. For example, if you are attempting to publish from more than one source to the same destination, there can be ID collisions and other problems with data integrity that will cause this error.

Corrective Action

Examine the `sites.log` and application server log files on the **destination** system. There is often a stack trace message in the `stderr` log that describes what went wrong.

Other Indicators of System or Configuration Issues

This section describes symptoms your Sites system may exhibit when the publishing system discovers configuration errors or system problems.

Publishing Session Does Not End and Displays an Odd Status

When a publishing session cannot begin or end, typically the batch user has not been configured correctly. Examine the **Status** listed for the publishing session. If the **Status** is "Not Found" or "false," it is likely that your batch host is not configured correctly.

Note that the Initialize Mirror Destination feature does not use the batch user. If you can initialize a destination correctly, but the publishing session has errors, it is likely that the batch user settings are incorrect.

Corrective Action

See "Step 1. Create the Batch User Account (If One Does Not Exist)," on page 273 and verify that you have configured the batch user correctly:

- The batch user identified by the `xcelerate.batchuser` property must have the appropriate read/write privileges. These are listed in that procedure.
- The name of the batch user identified by the `xcelerate.batchuser` property must be spelled correctly and the password identified by the `xcelerate.batchpass` property must be correct.
- The server identified by the `xcelerate.batchhost` property must identify the correct server. This property should identify the **application server** that hosts the source system, **not** the destination, and **not** the load balancer if you are using a load balancer. Note that if the port number is something other than 80, you must also specify the port number. For example: `myserver:7001`

  If the WebCenter Sites environment is clustered, only one batch host is supported. The `xcelerate.batchhost` property must be set on each cluster member to point to the dedicated host.

  If you see the message "Cannot retrieve output for publish session" in the publishing history, it is likely that the server name is incorrect.
- The port number for the batch host server must also be identified correctly. If you see the message "Cannot retrieve output for publish session" in the publishing history, it is likely that either the port number is incorrect, or if a port number is not specified, that the default port (80) is incorrect.

The System Stops Altogether

If your system simply stops, this behavior indicates that there may not be enough disk space available for the publishing system to function.

Corrective Action

Check the `stdout` and `stderr` files. If there are any Java write errors, you are out of disk space.

- For Export to Disk publishing and Export Assets to XML, you need enough disk space in your file system to store all the files that are being generated.
- For Mirror to Server publishing, you need space equal to four times the size of the data that is being mirrored available on both the source and the destination system or the mirror operation will fail.

Note that the publishing process will also fail if the Java temp directory is not large enough.

Pages Are Not Refreshed After the Publishing Session Ends

If you are using Satellite Server—either the co-resident Satellite on a management or development system or a remote Satellite Server for a delivery system—and you notice that cached pages are not being regenerated after a publishing session, it is likely that you have configured Satellite Server incorrectly.

Corrective Action

Examine the `sites.log` file on the **destination** system and look for messages like these:

- "Number of satellite servers must match number of usernames and passwords."

  This message indicates that there is something wrong with the values specified for the `cs.satellitehosts`, `cs.satelliteuser`, and `cs.satellitepassword` properties in the `futuretense.ini` file.
- "-100.FormPoster failed flushing URL."

  This message indicates that either a Satellite servlet wasn't running or that the destination system couldn't reach a Satellite servlet that it tried to reach:

Open the `futuretense.ini` file, examine the properties on the **Satellite** tab, and note the values set for the user names, passwords, and host names. Open the satellite.ini files for the co-resident Satellite Server and your remote Satellite Server applications. Compare the values specified for the `cs.satellitehosts`, `cs.satelliteuser`, and `cs.satellitepassword` properties in the `futuretense.ini` file to the values set for the `username` and `password` properties in the `satellite.ini` files. **They must match**.

Remember that the order in which you specify host names in the `futuretense.ini` file must match the order in which you identify user names and passwords.

DB2 Systems, Troubles When Publishing Assets with Associations

When you notice that there are problems with publishing assets that have associations and your system uses a DB2 database, it is likely that the `LOCKLIST` parameter is not set correctly.

Corrective Action

The WebCenter Sites installation guide that describes DB2 installations recommends that this property be set to at least 1000. If you find that you are having difficulties publishing assets with associations, increase this value.

Part 4

RealTime Publishing

This part describes WebCenter Sites' RealTime publishing system, the publishing methods it supports, and procedures for configuring and managing the supported publishing methods.

This part contains the following chapters:

Chapter 18

RealTime Publishing Overview

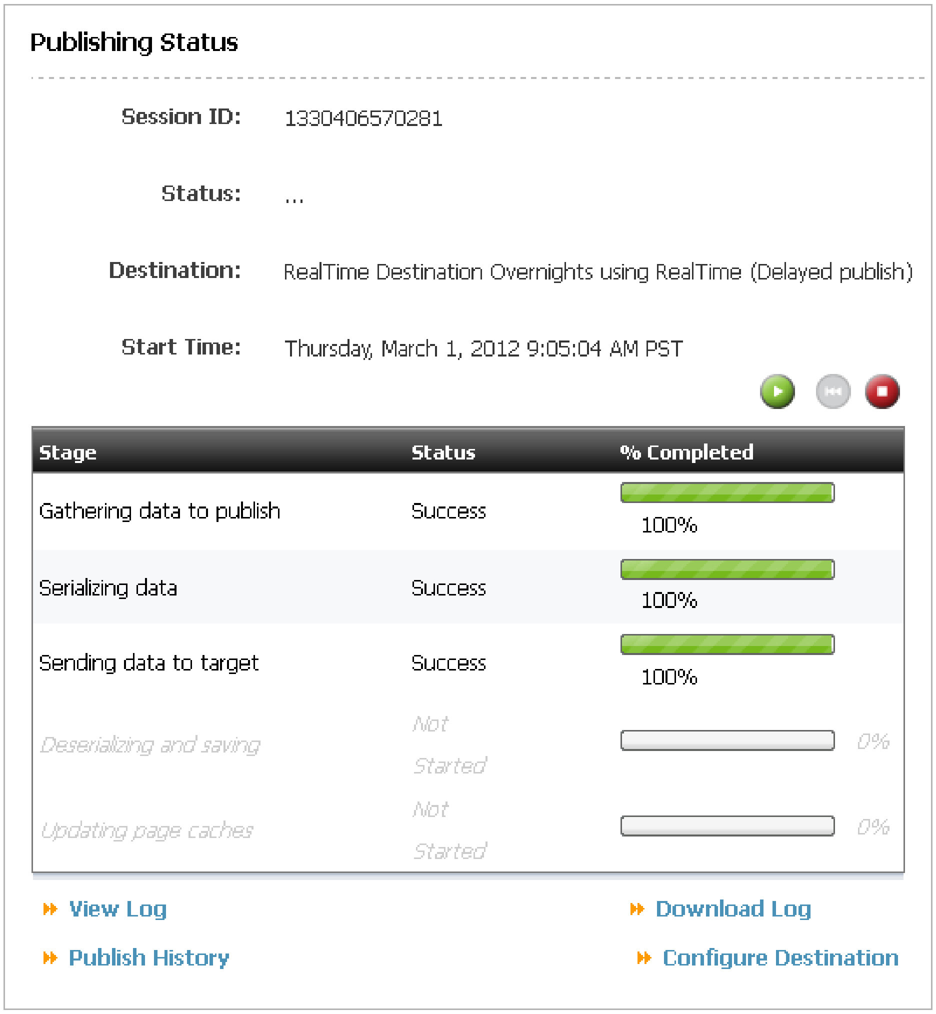
RealTime publishing is a dynamic publishing process, and the preferred method for publishing. Using the RealTime user interface, an administrator can monitor the progress of a publishing session, publish selected assets on demand, bulk unapprove assets approved for publishing, cancel publishing, and redo publishing sessions.

This chapter contains the following sections:

RealTime Publishing Overview

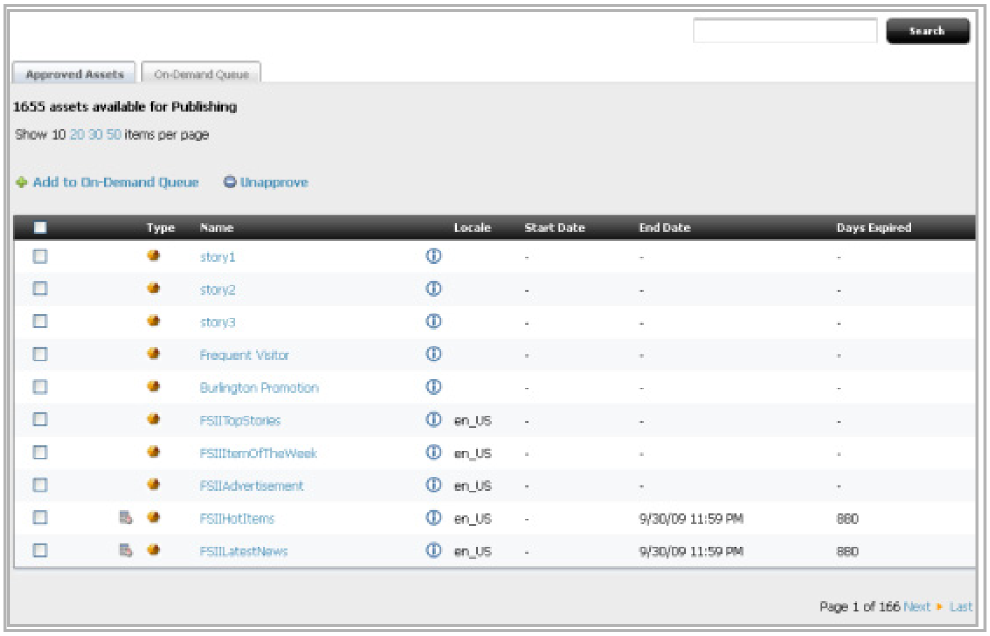
RealTime publishing is a dynamic publishing method named for its user interface. In the user interface, publishing status messages update in real time: Progress bars and mouse-over tooltips illustrate the progression of a publishing session as it unfolds, relaying status information from both source and destination systems.

Figure 12: The 'Publishing Status' screen communicates up to the moment status information about active publishing sessions



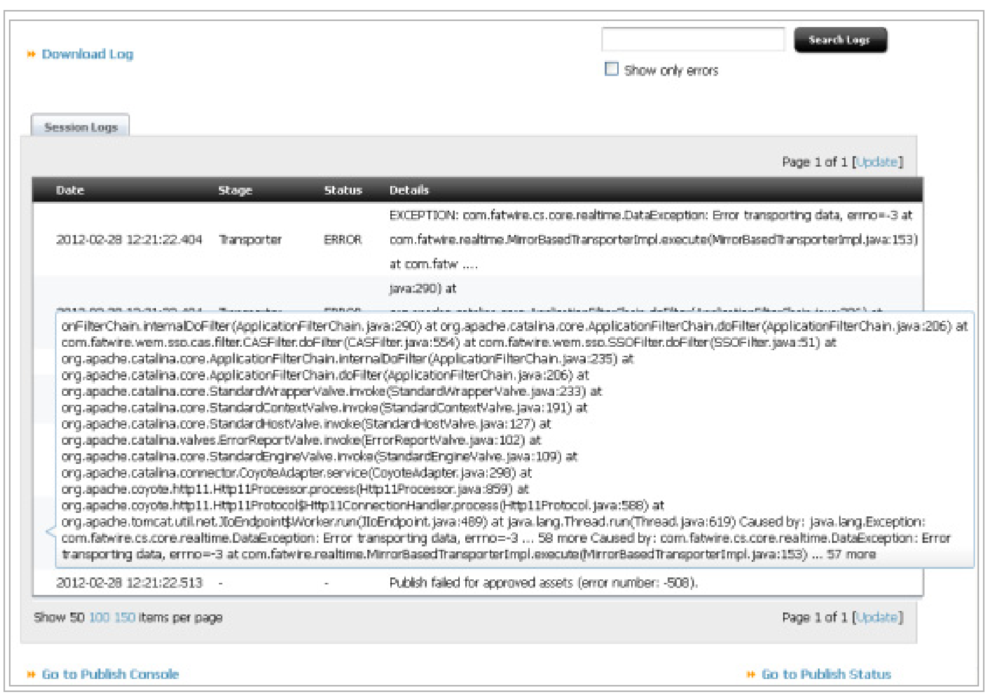
The RealTime user interface is also interactive. For example, you can interact with the list of approved assets. From the approved assets list you can select as many items as you like for immediate publication by placing the items into the **On Demand queue**. You can also remove one or more assets from the approved assets list using **Unapprove**. In both On Demand publishing and asset unapproval, asset dependencies are taken into account to ensure against broken links or missing assets on the destination system web site.

Figure 13: In the 'Approved Assets' list, you can place assets into the On-Demand Queue, and Unapprove them



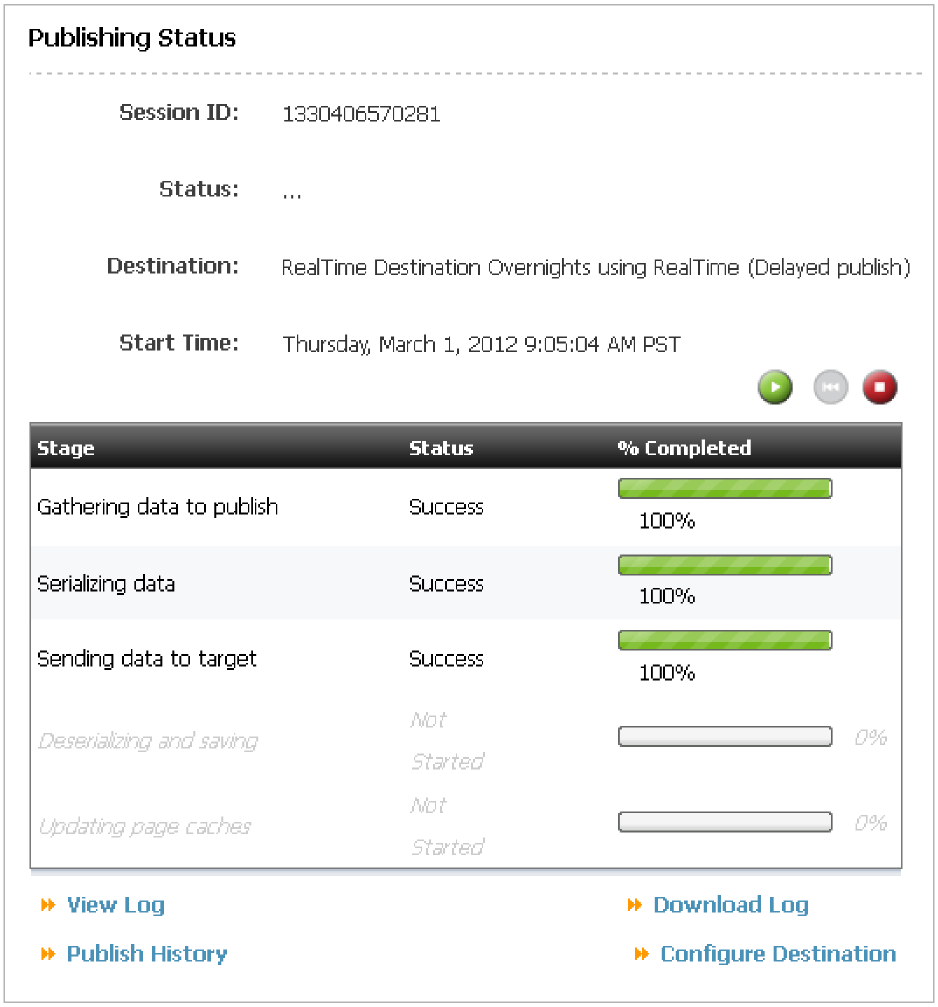
In addition to status messages and asset approval functions, the RealTime user interface features interactive logging screens. On the logging screen, you can search, page through, and filter log entries. Mousing over a summary log message's "**show all**" link expands the entry to a detailed report.

Figure 14: The Session Log is interactive; view a detailed entry by clicking the "show all" link next to the summary entry



Underlying the RealTime user interface is a multi-stage publishing process. WebCenter Sites replicates, transfers, and stores web site data in five stages. The “Publishing Status” screen below shows the five stages of publishing. These stages are described in detail in the section “How RealTime Publishing Works,” on page 344.

Figure 15: The five stages of RealTime publishing



One advantage of breaking RealTime publishing into discrete stages is that it introduces the option of stopping the publishing process after a stage completes and picking up the publishing session from the next stage.

RealTime publishing includes two modes of publishing: complete and delayed. In **complete publishing** mode, WebCenter Sites runs all stages of publishing in one continuous sequence. In **delayed publishing** mode, there is a pause midway through the process so the administrator can control when the system begins writing assets to the WebCenter Sites destination database.

Delayed publishing can be used to minimize the impact of publishing on web server performance. Using delayed publishing, the administrator can time a destination system update to occur during off-peak traffic hours, an option that is particularly useful when publishing a large number of assets.

Another advantage of the multi-stage publishing process is the ability to cancel a running session at will and then, if you wish, restart the cancelled session from just after the last successfully completed stage of publishing. You can also redo a publishing session that does not complete due to errors or power interruptions.

Figure 16: You can cancel a running session from the 'Publish Console,' then restart it from just after the last successfully completed stage.

Summary

When you use the RealTime publishing method, you can:

- Monitor an active publishing session. Progress bars on the Publishing Status screen show the current progress of the session on both source and destination WebCenter Sites systems.
- Selectively publish approved assets using On-Demand publishing.
- Selectively unapprove assets.
- Review interactive, sortable session logs.
- Configure your destination for Complete Publishing mode to run the entire publishing process in an uninterrupted session.
- Configure your destination for Delayed Publishing mode to pause the publishing session midway so you can control when the system begins writing assets to the destination database.
- Cancel an active publishing session.
- Restart a cancelled publishing session from just after the last completed stage.
- Redo a failed publishing session.

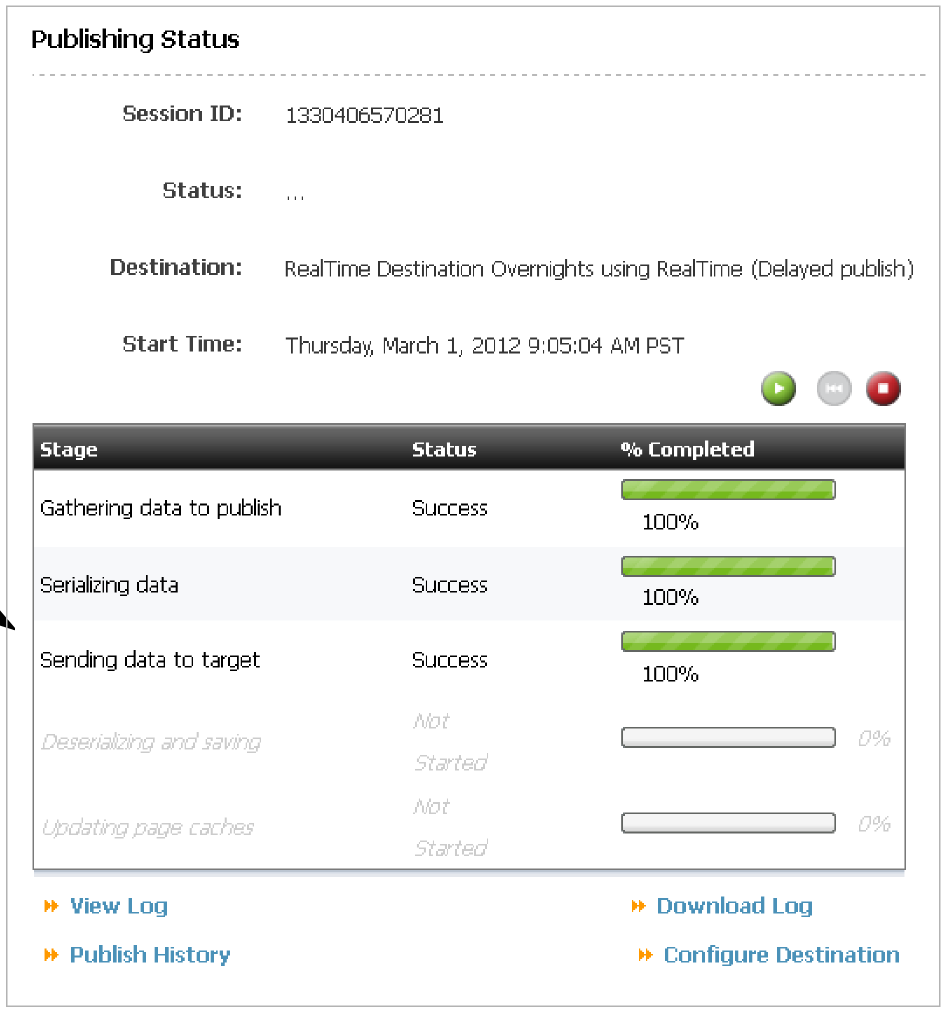
How RealTime Publishing Works

Dynamic publishing is the process of copying assets and their underlying schema from one WebCenter Sites system to another. In RealTime publishing, this process is broken up into five stages.

WebCenter Sites always performs the five stages in order. When the destination is configured for complete publishing mode, the process proceeds without interruption. When configured for delayed publishing mode, the process pauses before the fourth step. User interaction is needed for the session to complete.

As the publishing session proceeds, you can monitor the progress of each stage on the 'Publishing Status' screen. Instructions for monitoring a publishing session can be found in Chapter 20, "Working with RealTime Publishing Modes."

The five stages of publishing



The stages of publishing are:

1. **Gathering data to publish**:

   a. The WebCenter Sites source system locks the assets that are approved for this publishing session. Locking approved assets prevents them from being edited during publishing.

   b. On the source system, RealTime publishing uses data from the approval system to create a manifest, a list of assets to publish. The manifest identifies resource groups, groups of assets that must exist together on the destination to prevent broken links and incomplete pages.

2. **Serializing data**:
   a. The WebCenter Sites source system mirrors the following items:
      - Asset types
      - Auxiliary tables (identified in the destination configuration)
      - Approved assets
   b. The source system serializes (translates to binary format) the mirrored data.

> **Note**
>
> This step can require considerable space to complete. It is recommended that the temporary space assigned to the JVM (via `java.io.temp`) is at least three times the size of the data to be published. The source and destination databases must also have enough space for the table `FW_PUBDATASTORE` to grow. Since this can be equal to the size of the entire source site (both objects stored on disk and the database) during an initial publish, this means that the space required is likely to exceed what WebCenter Sites is currently using (particularly when large binaries are included).

3. **Sending data to the target:**
   a. The destination WebCenter Sites database locks assets that are approved for this publishing session. When assets on the target are locked, they cannot be edited.
   b. The WebCenter Sites source system sends the manifest and all serialized data to the destination. The data has not yet been committed to the destination database.

   A delayed publishing session pauses at this point. It continues to stage 4 only after the administrator selects the **Resume** button on the “Publish Console” or “Publishing Status” screen.

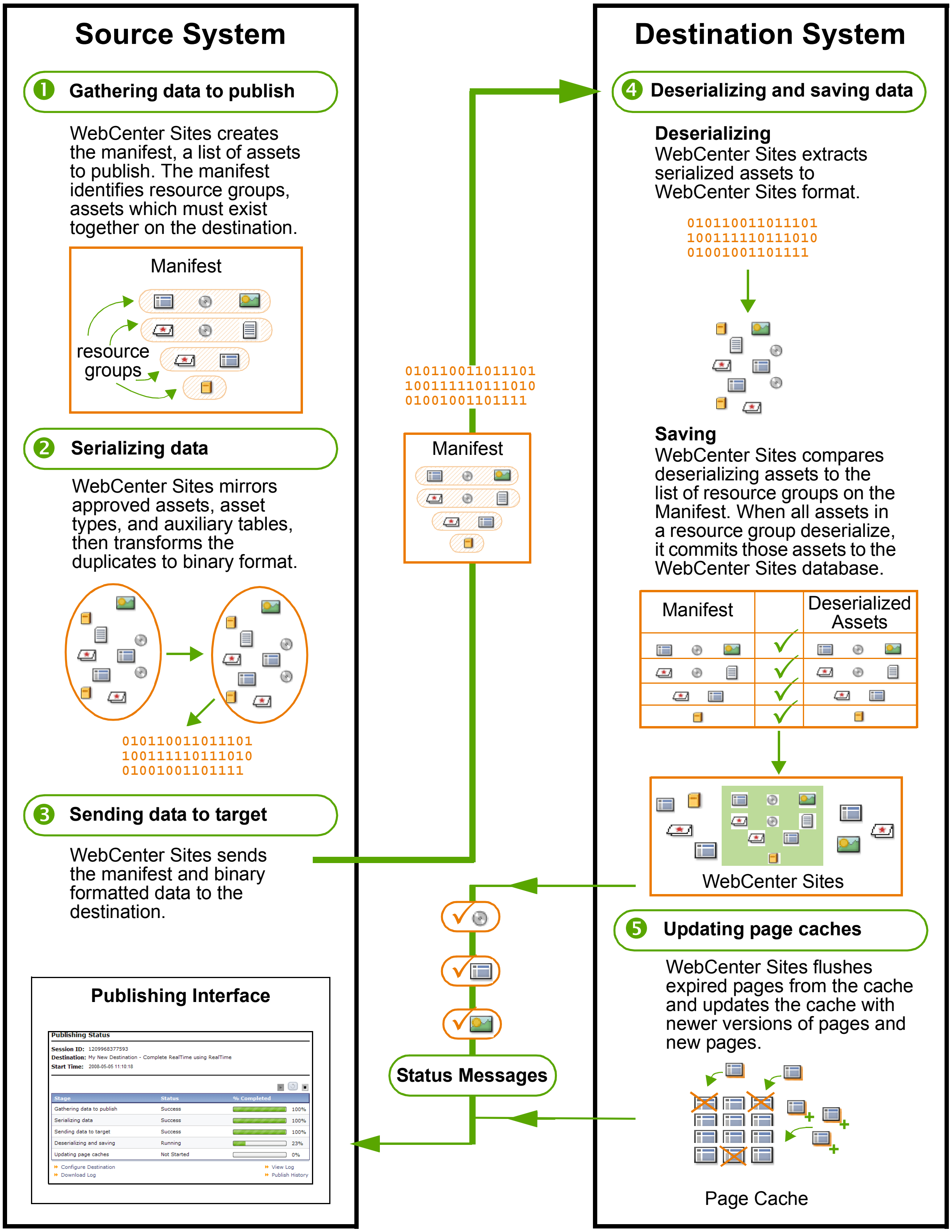
4. **Deserializing and saving data**:
   a. The WebCenter Sites destination system deserializes (inflates back to WebCenter Sites format) the mirrored asset types, auxiliary tables, and approved assets.
   b. The destination database updates asset types and auxiliary tables with the deserialized ones, or adds them as new data.
   c. The destination system uses the manifest to determine when to commit assets to its database. Assets in a resource group are committed only when every asset in the group has deserialized.
   d. When the destination system commits all assets in a resource group:
      1) It unlocks the assets in its database for editing. As the publishing session runs, the number of published, editable assets in the destination database increases.
      2) The destination system sends a message to the source system identifying assets that have been successfully published. Assets in that resource group are unlocked on the source system for editing. As the publishing session runs, the number of editable assets on the source system increases.

5. **Updating page caches**:

The destination system completes the publication process:

   a. Flushes the page cache, which removes expired pages.
   b. Adds new pages to the cache.

Figure 17: The Five Stages of Publishing



Chapter 19

Configuring the RealTime Publishing Process

When you configure the RealTime publishing process, you define the publishing destination and mirror site configuration data to that destination. In addition, you can set the publishing schedule. This chapter also includes instructions for using the RealTime process to synchronize site data.

This chapter contains the following sections:

Overview

Dynamic publishing is the process of copying assets and their underlying schema from one WebCenter Sites system to another. In order for the RealTime publishing process to work, both the source and target systems must be set up for RealTime publishing.

Source System Configuration

The source system requires the following items to be configured:

- **A batch user account**. This user allows publishing to run as a background process and designates the location of publishing logs.
- If there is a proxy, the identity of the local **proxy server**. This is identified to WebCenter Sites in the `futuretense.ini` file.
- For large publishes (such as the initial publication of a site), the session timeout may need to be increased beyond the default 900 seconds (for example, 9000 seconds). To do so open `futuretense.ini` in property editor and change `cs.timeout` to 9000, then restart WebCenter Sites. When the publish is completed this value can be reset.
- The **publishing destination**, which identifies:
  - The destination URL and port
  - The publishing method
  - Publishing log and email notification options
  - User roles with approval and publishing privileges
  - The RealTime user, who has publishing privileges on the destination

Additionally, though not required for publishing to work, you can configure a **publishing schedule** which will publish approved assets on a regular basis.

Destination System Configuration

The destination system requires:

- A **RealTime user account,** which has publishing privileges on the destination.
- **Mirrored site configuration data** from the source system for the sites to be published:
  - Sites
  - `Publication` and `SitePlanTree` table data
  - Auxiliary tables (such as `Source` and `MimeType`)
  - Asset types
  - Roles
  - Start menu items*
  - Workflow groups and processes*
  - Tree tabs*
  - Saved searches*

  * *Items marked with an asterisk are typically not necessary on a delivery system.*

Additionally, if you will not publish from the destination system, we recommend that you **turn off asset invalidation**.

Follow the steps in this chapter to configure your system for RealTime publishing.

Configuring Your System for RealTime Publishing

> **Note**
>
> Configuring a RealTime publishing environment requires you to provide information that is specific to your WebCenter Sites installation and your business practices. Before starting procedures in this section, it is best to read them in order to gather and confirm the information they prompt you to provide.

The main steps for configuring a RealTime publishing environment are as follows:

Step 1: Create the Batch User Account (If One Does Not Exist)

> **Note**
>
> **This procedure needs to be performed only once on each source system,** regardless of the number and types of publishing destinations you are configuring.
>
> If a batch user account already exists on your source system, skip to "Step 2: Set Up the Destination System."

A batch user account is required before any type of publishing can take place.

The purpose of a batch user account is two-fold:

- Enable the publishing process to run as a background process on the source system.
- Specify the location where publish logs will be stored.

If you have not already done so, create the batch user account. For instructions, see "Step 1. Create the Batch User Account (If One Does Not Exist)," on page 273.

Step 2: Set Up the Destination System

Data cannot be dynamically published unless the same database schema exists on the source and destination systems. In this step, you will ensure that the structure of auxiliary tables on the source database is reproduced in the destination database. The data, itself, you will mirror in a later step ("Step 5: Initialize the Destination Database," on page 358).

The destination also requires a mirror user to complete database transactions on the destination system during publishing. You will ensure that a mirror user exists on the destination. (The mirror user is different from the batch user, which exists on the source system.)

To set up the destination system

1. Determine which auxiliary tables on the source system support the types of assets that users will publish, then manually re-create the table structure on the destination system.

   Note

   Auxiliary tables are custom-defined tables that support asset types. The tables are not created via WebCenter Sites' AssetMaker or Flex Family Maker. Examples of auxiliary tables are `Source`, `MimeType`, and lookup tables (for different attributes in asset types).

   Database properties on the source and destination systems, if not an exact match, must be compatible, particularly database schema options (set in `futuretense.ini`). Be sure to restart the application server if you make changes to `futuretense.ini`.

2. Create the RealTime user on the destination system (using the administrator's interface) and note the following:

   Note

   The RealTime user has the same privileges as the mirror user configured for Mirror to Server publishing. If a mirror user is configured for Mirror to Server publishing, the same user may serve as the RealTime user.

   - **Basic privileges:** The RealTime user must be assigned the following ACLs:
     - Browser
     - ElementEditor
     - PageEditor
     - TableEditor
     - Visitor
     - VisitorAdmin
     - xceladmin
     - xceleditor
   - **Additional privileges:** Because the RealTime user account is used by the CacheManager to regenerate the page cache after a publishing session, the RealTime user must have sufficient privileges to regenerate all the pages in the

cache. Therefore, the RealTime user account must be assigned all the ACLs that are assigned to page entries in the `SiteCatalog` table or database tables that hold data to be rendered.

If you need help with creating the mirror user, see "Creating a New User," on page 86.

Step 3: If a Proxy Server Exists, Identify it to the Source System

> **Note**
>
> **This procedure needs to be performed only once on each source system,** regardless of the number and types of publishing destinations you are configuring.
>
> Skip to Step 4: Create a RealTime Destination Definition on the Source System if:
>
> - The local proxy has already been identified to your source system, or
> - A proxy server is not used.

To identify the local proxy server to the source system for all destinations

1. Open `futuretense.ini` in the Property Editor on the source system, select the **Export/Mirror** tab, and specify values for the following properties:
   - `cs.mirrorproxyserver`

     Set this property to either the name or the IP address of the local proxy server.
   - `cs.mirrorproxyserverport`

     Set this property to the port number of the local proxy server.
2. Save the property file and close the Property Editor.
3. Restart the application server.

To identify the local proxy server to the source system per destination

1. In the Admin interface, select the **Admin** tab. Expand **Publishing**, then expand **Destinations**, and open the destination.

   The destination opens in the main screen.
2. Click **Edit**.

   The "Edit Destination" screen for the destination displays.
3. In the **More Arguments** field, enter

   PROXYSERVER=*wcs.example.com*&PROXYPORT=*8080*

   Where *wcs.example.com* is the local proxy server, and *8080* is the port the proxy is running on.
4. Click **Save**.

Step 4: Create a RealTime Destination Definition on the Source System

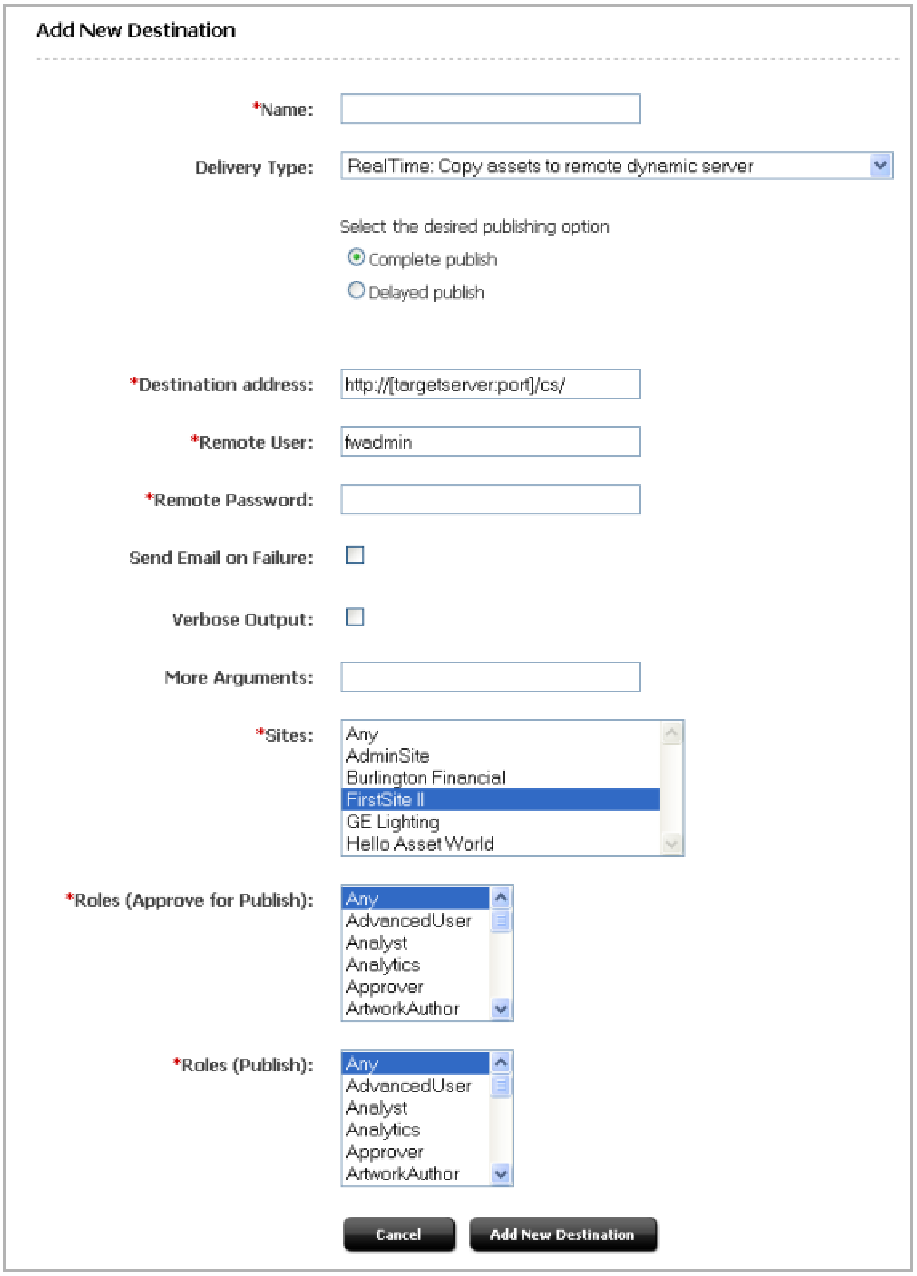
In this step, you will define the publishing destination to the source system.

> **Note**
>
> While you can define multiple destinations, bear in mind that only sequential publishing is supported.

To create a RealTime destination definition

1. On the source system, log in to the administrator's interface.
2. In the **Admin** tab, expand **Publishing**, then **Destinations**.
3. Under **Destinations**, double-click **Add New**.

The “Add New Destination” form appears.



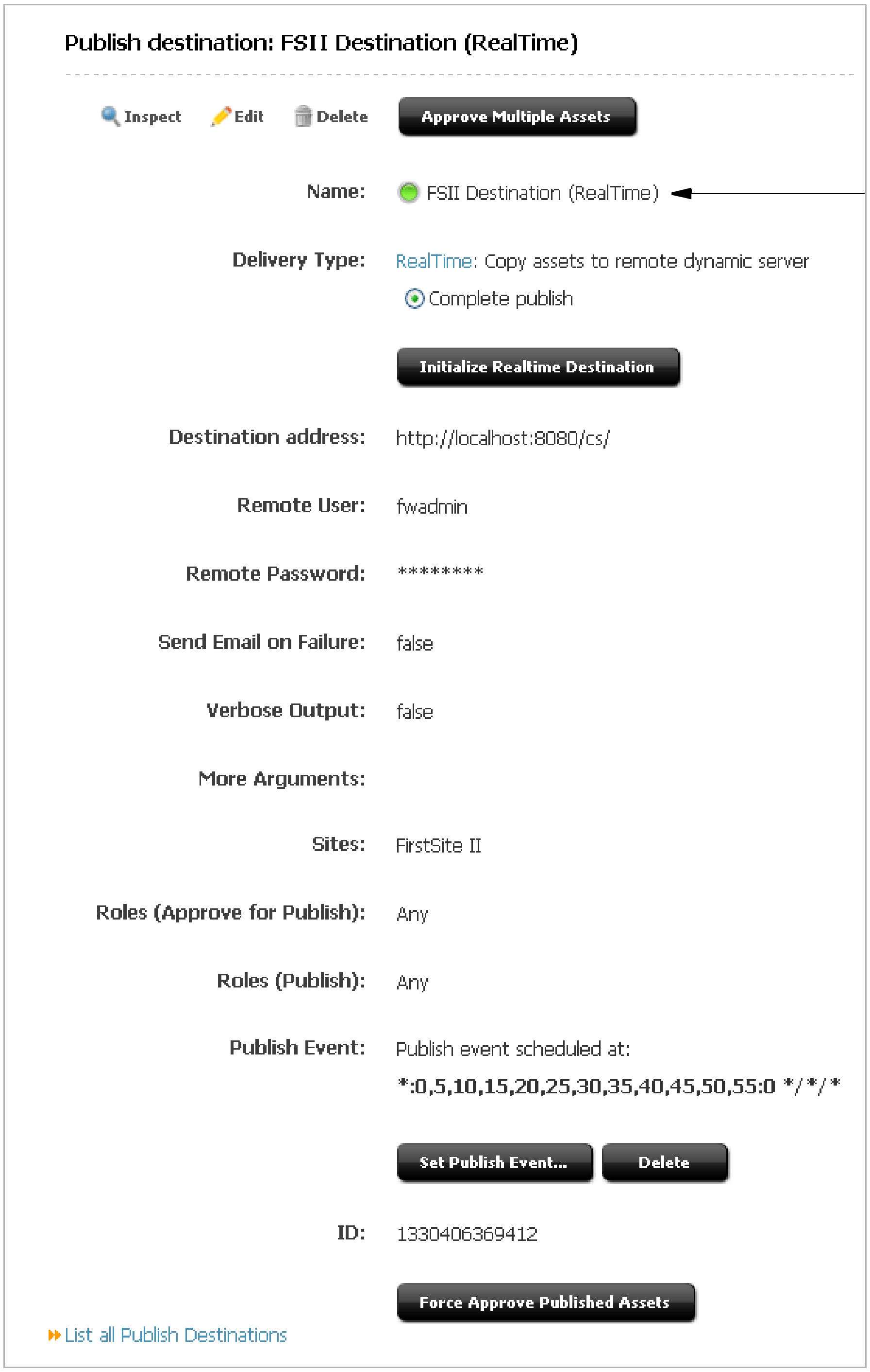
4. Fill in the “Add New Destination” form. For guidelines, use the information below.
   - **Name**: Enter a unique name for the destination.
   - **Delivery Type**: Select **RealTime: Copy assets to remote dynamic server**, then choose one of the following options:
     - **Complete publish** to copy assets to the destination address without interruption to the publishing process.
     - **Delayed publish** to pause the publishing process before assets are committed to the destination database. To resume the session, a user must initiate the

fourth stage of publishing (deserializing [inflating] data and saving it to the destination database). More information on delayed publishing can be found in “Resuming a Delayed Publishing Session,” on page 371.

- **Destination Address**: Enter the URL in the format shown. For `[targetserver:port]`, delete the brackets, enter the hostname or IP address of the destination system, and specify the port to be used by the destination. (**A slash is required at the end of the URL** because this URL is appended dynamically.)
- **Remote User**: Enter the name of the RealTime user that you created in “Step 2: Set Up the Destination System,” on page 352. This user will be invoked by the publishing system to unpack the RealTime queue on the destination system.
- **Remote Password**: Enter the password of the RealTime user.
- **Send Email on Failure**: If publishing fails and email notices to that effect are required, select the checkbox and fill in the field that is displayed:
  - **EmailIDs**: Enter the recipient’s email address.
    (This field is available only when **Send Email on Failure** is selected.)
- **Verbose Output**: Select this option to activate detailed error logging during the publishing process. When this option is selected, messages in addition to error messages are written to the `PubMessage` table. Because additional information lengthens the publishing process, select this parameter only for troubleshooting.
- **More Arguments**: This parameter is reserved; no additional arguments may be specified at this time.
- **Sites**: Select the sites whose assets can be approved for and published to this destination.
- **Roles (Approve for Publish)**: Select the roles to which you are assigning asset approval privileges. All users assigned these roles will be able to approve assets.
- **Roles (Publish)**: Select the roles to which you are assigning publish privileges. All users assigned these roles will be able to publish and view the “Publish Console.”

5. Click **Add New Destination**.

WebCenter Sites writes your information to the `Pubdestination` table and displays the destination definition in the “Inspect” form.



Note this field.
If a red button is displayed, it means that the target server cannot be located. Refer to the note below for possible causes.

> **Note**
>
> Before proceeding to the next step, check the "Name" field. A red button next to the destination name means that the target server cannot be located. Common causes are the following: The target URL is incorrect or the target server is not running.

6. If you need to create more destination definitions, repeat steps 3 through 5 for each new definition.
7. When you are ready to initialize the destination system, proceed to the next section, "Step 5: Initialize the Destination Database." Begin at step 3 of that procedure.

Step 5: Initialize the Destination Database

You must initialize the destination database before you can publish to it. To initialize, you will mirror two constructs:

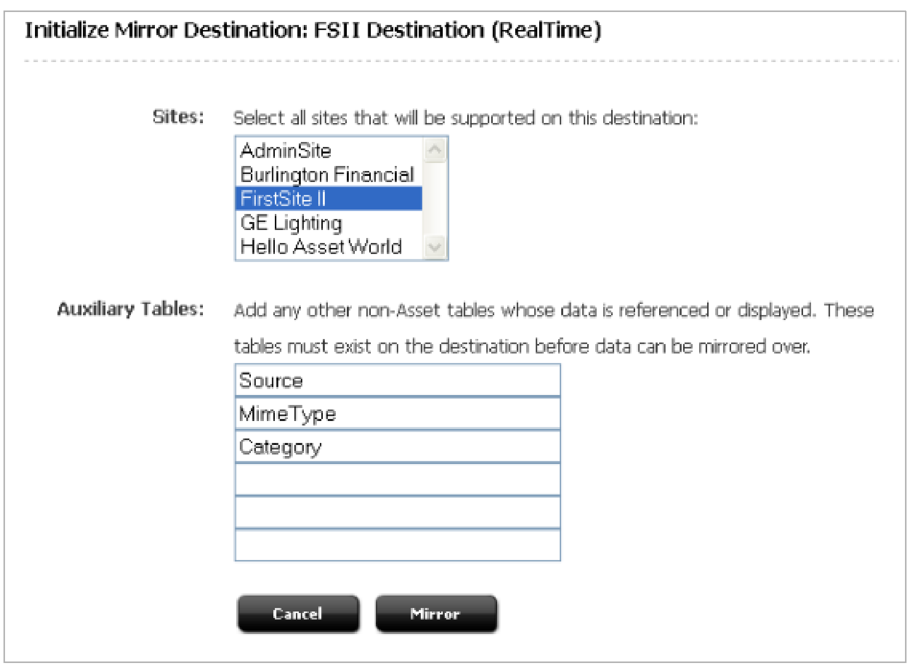
- Sites
- Data in auxiliary tables (which support the types of assets users will be publishing)

The `Publication` and `PublicationTree` tables will be updated with the site names and table data you have chosen to mirror.

To initialize the destination

1. In the **Admin** tab on the source system, expand **Publishing**, then **Destinations**.
2. Under **Destinations**, double-click the RealTime destination you want to initialize.
3. In the "Publish Destination" form, click **Initialize RealTime Destination**.

   The "Initialize RealTime Destination" screen appears:



4. In the **Sites** field, select the sites whose assets you want to publish to this destination.
5. In the **Auxiliary Tables** field, enter the names of the following tables:
   - `Source`, if you are using the source feature for any of your asset types.
   - `MimeType`, if you are using Sites DocLink, flex filter assets, the ImageFile asset type, or if you are using the `MimeType` table for any of your custom asset types.
   - `Filters`, if you are using flex filters on your site. (The `Filters` table lists the classes that are used by the flex filters. You will need to copy `jar` files and classes manually.)
   - Any other auxiliary tables (such as lookup tables) for your asset types.

   Note

   Only five fields are provided for table names. If you have to enter more than five table names, complete this procedure through step 6 for the first five tables, then repeat steps 3–6 for the remaining tables.

6. Click **Mirror**.

   If initialization is successful, WebCenter Sites displays a confirmation message (otherwise, an error message is displayed).
   - Based on the sites that you selected in step 4, the corresponding rows from the `Publication`, `SitePlanTree`, and `AssetType` tables are copied to the destination.
   - If you specified auxiliary tables in step 5, all rows in those tables are copied to the destination.

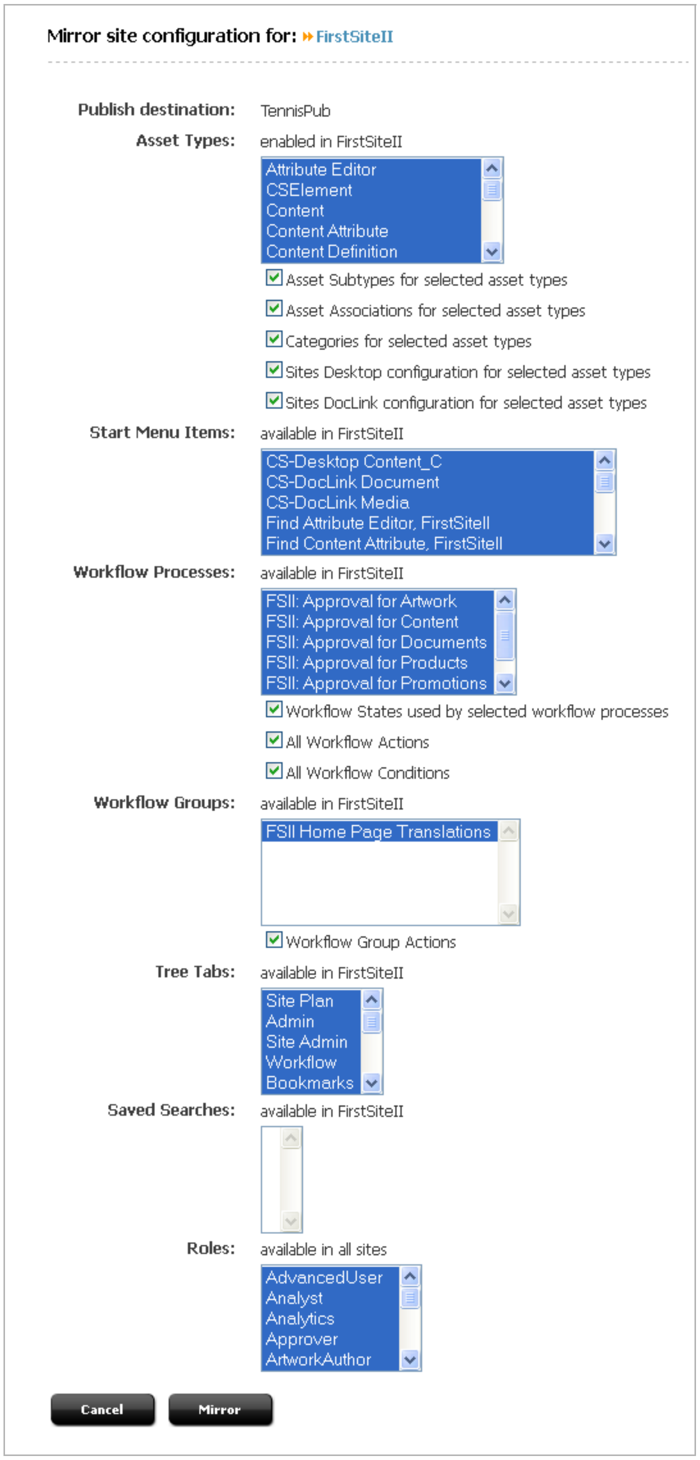
Step 6: Mirror the Site Configuration Data to the Destination Database

In this step, you will mirror the source's site configuration data (such as asset types and start menu items) to the destination database.

To mirror the site configuration data

1. In the **Admin** tab on the source system, expand **Sites** and double-click the site whose data you want to publish to the destination.
2. In "Publish Destinations" (near the bottom of the screen), click **Mirror site configuration for *destination*.**

WebCenter Sites displays the "Mirror site configuration for" form.



3. Select the data you wish to mirror to the destination:

> **Note**
>
> Which data you choose to mirror typically varies with the purpose of the destination system:
>
> - If the destination system is strictly for delivery, it makes sense to mirror only asset types.
> - When content management functions are to be supported, mirror the remaining data as well.

   - **Asset Types**: Select the asset types that you want to make available on the destination.

     If your asset types have subtypes and categories, choose the appropriate options from the list in the **Asset Types** section and select the corresponding check boxes. For example, the `AssocNamed` table contains information about associations for asset types. The table will be mirrored only if you select **Asset associations for selected asset types** and then the relevant asset types.

     Select the Sites Desktop / Sites DocLink configuration check boxes to configure the selected asset types for Sites Desktop and Sites DocLink.
   - **Start Menu Items**: Select any start menu items that you or the developers designed for your content providers to use.
   - **Workflow Processes**: If there are workflow processes for your asset types, select these processes and the appropriate workflow items.
   - **Workflow Groups**: Select any workflow groups that you or the developers designed for your content providers to use.
   - **Tree Tabs**: Select the tree tabs that you want to make available on the destination.
   - **Saved Searches**: Select any saved searches that you or the developers designed for your content providers to use.
   - **Roles**: Select the roles that must exist on the destination.

4. Click **Mirror**.

   Based on which asset type configuration options you selected in step 3, appropriate rows from the `AssocNamed`, `AssocNamed_Subtypes`, and `Category` tables are copied to the destination:

   - If you selected **Start Menu Items** or **Saved Searches**, appropriate rows from the tables that implement those features are copied to the destination.
   - Based on which workflow configuration options you selected in step 3, appropriate rows from the workflow tables are copied to the destination.

5. Repeat steps 1–4 for each site whose assets you want to publish to the destination.

Step 7: Test the Publishing Process

You can test your publishing process by using all publishable assets or a smaller collection.

A. Approve Assets

Do one of the following:

- To fully test your publishing process, you must approve all the assets that you intend to publish. See "Approving Multiple Assets," on page 315 for help with approving multiple assets at once.
- If you wish to perform a simple test of your configuration, temporarily create a home page asset with just a few dependents.

B. Publish the Assets

To run a test publishing session on the approved assets

1. On the source system, click **Publishing** in the button bar.
2. In the **Publish Console**, select your RealTime destination from the drop-down list and click **Select Destination**.

   WebCenter Sites displays information about the assets that are ready to be published to this destination.
3. If for some reason the destination has not yet been initialized, it must be initialized now or the publishing session will fail. To initialize the destination, click **Create Site on Target** in the **Publish Console** (or follow instructions in " Step 5: Initialize the Destination Database," on page 358). This creates the basic site information on the destination system.
4. Click **Publish**.

   The publishing system mirrors all the approved assets for this destination to the WebCenter Sites database on the destination system.

Step 8: Test the Published Site

To test your published site, you will point your browser to the home page asset on the destination system and examine the site.

If you have not already done so, you must first determine the home page asset's URL. A WebCenter Sites URL is constructed by concatenating the following values, as shown:

```
http://<targetserver:port>/<cgipath>/
   ContentServer?pagename=<your_home_page>/<other_info>
```

where

- `<targetserver:port>` is the hostname or IP address (and sometimes the port number) of the destination system.
- `<cgipath>` is the value of the `ft.cgipath` property in the `futuretense.ini` file. For example, `<cgipath>` is `/servlet/` for WebLogic and other application servers with servlet architectures.
- `ContentServer?pagename=` sets the name of the page to be displayed at this URL.

- `<your_home_page>` is the home page's name from the `SiteCatalog` entry.
- `<other_info>` is additional information that is passed in with the WebCenter Sites page criteria variables, `c`, `cid`, `tid`, and `p` (see the *Oracle WebCenter Sites Developer's Guide* for information about these variables).

For example, the URL for the Burlington Financial home page looks like this when it is rendered by the WebCenter Sites JumpStart Kit:

```
http://localhost:7001/servlet/
   ContentServer?pagename=BurlingtonFinancial/Page/Home
```

To determine the URL of your home page and test the site

1. Start the Property Editor on the destination system (if you need help with this step, see the *Oracle WebCenter Sites Property Files Reference*).
   a. Open the `futuretense.ini` file.
   b. In the **App Server** tab, locate the `ft.cgipath` property and write down its value.
   c. In the **Compatibility** tab, locate the `ft.approot` property and write down its value.
   d. Save the property file and close the Property Editor.
2. Open a text editor and assemble the URL:
   a. Enter `http://` or `https://` followed by: `<targetserver:port>/<cgipath>/`

      For example:

      ```
      http://bigfatsun.example.com:8080/servlet/
      ```

      (for WebLogic and WebSphere)
   b. At the end of the string, add the following text:

      ```
      ContentServer?pagename=<your_home_page>
      ```

   Now the URL should look similar to the following example (for WebLogic and WebSphere):

   ```
   http://bigfatsun.fatwire.com:8080/servlet/
      ContentServer?pagename=ExampleSite/Page/Home
   ```
3. Point your browser to the URL you assembled.
4. Scan the page for errors and test all links to ensure they work.

Note

If you detect an error related to content that is not stored as assets (the content is stored outside WebCenter Sites), use the appropriate transfer protocol to publish that content. For more information, see "Step 9: Set Up the Schedule."

It is recommended that you conduct a complete test of the system at strategic times: under peak load conditions after you have mirrored the entire site for the first time, and at regular points thereafter.

Step 9: Set Up the Schedule

Once you have ensured that your destination is correctly set up, finish configuring your source's publishing system.

To finish configuring the publishing system

- On the source system, schedule times running publishing sessions to the destination. For help with this step, see "Scheduling Publish Events," on page 319.
- If your site uses image files that are stored not as assets but as images on the web server, plan how you will move the image files from the CM system web server to the delivery system web server. For example, you can set up a regular FTP transfer.
- If you are using elements and SiteCatalog page entries that are not CSElement and SiteEntry assets, you must use the CatalogMover tool to mirror them to the destination system. For help with CatalogMover, see the *Oracle WebCenter Sites Developer's Guide*.

Step 10: Turn Off Asset Invalidation on the Delivery System

By default, the destination invalidates chain publishing. That is, the publishing system (on the destination) marks each published asset as changed, due solely to the fact that the asset was published. Hence, an asset *must be approved from the current destination* before it can be published to yet another destination.

For assets on a final destination—such as a delivery system— asset invalidation should be turned off. It is an unnecessary (and time consuming) operation, as assets will not be published from that destination.

To turn off asset invalidation

1. Start the Property Editor on the final destination system (delivery, in this example) and open the `futuretense_xcel.ini` file. (If you need help with this step, see the *Oracle WebCenter Sites Property Files Reference*.)
2. In the **Publishing** tab, locate the `xcelerate.publishinvalidate` property and set its value to `false`.

Adding a New RealTime Destination Definition

Once you have configured your source system to publish to one RealTime destination, you can define other RealTime destinations, if necessary. Follow only the steps below:

- Step 2: Set Up the Destination System
- Step 4: Create a RealTime Destination Definition on the Source System
- Step 5: Initialize the Destination Database
- Step 6: Mirror the Site Configuration Data to the Destination Database
- Step 7: Test the Publishing Process
  - A. Approve Assets
  - B. Publish the Assets
- Step 8: Test the Published Site
- Step 9: Set Up the Schedule
- Step 10: Turn Off Asset Invalidation on the Delivery System

Synchronizing Site Configuration Data

In addition to publishing assets, you can synchronize site configuration between the source and destination. For example, you can synchronize site configuration data from a development system to a management system or to a delivery system.

Use the **Mirror Type Configuration** page:

- To move configuration items that support asset types (start menu items, associations, asset types, and so on).
- If you or your developers add categories or subtypes for existing asset types.
- To move workflow configuration data from a development system to a management system.

See Step 6: Mirror the Site Configuration Data to the Destination Database for instructions on using the Site Configuration form.

Use the **Initialize Mirror Destination** page:

- If you or your developers add sites, or auxiliary tables to your system that you wish to publish.
- When you need to troubleshoot your configuration. If you can successfully initialize a RealTime destination, the source and the destination systems are communicating.

See Step 5: Initialize the Destination Database for instructions on using the Initialize Mirror Destination form.

Performance Enhancement

To improve the efficiency of high-volume dynamic publishing, an approval query can be implemented. For more information, see “Approval Query for Performance Enhancement,” on page 302.

Chapter 20

Working with RealTime Publishing Modes

One of the advantages of RealTime publishing is the feedback that the system generates and displays to the publishing administrator throughout the publishing process. The publishing administrator can monitor sessions in progress, review publishing schedules, inspect history records, and explore logs.

This chapter contains the following sections:

Complete and Delayed Publishing

A publishing session can run in one of two modes: complete or delayed. The mode must be explicitly specified, as shown in "Configuring Your System for RealTime Publishing," on page 351.

- In a complete publishing session, the five stages of RealTime publishing run in one uninterrupted sequence. You have the option to stop and resume a session, or cancel it.
- In delayed publishing, the session pauses after the second stage (data serialization). For the session to run to completion, you must resume the session. The final stages follow automatically.

  Delayed publishing enables you to control when the system begins writing to the destination database. For example, if you are publishing a large number of assets to a heavily trafficked web site, delayed publishing offers you the option to wait for a low-traffic period, when saving data to the destination database is a more efficient process.

This chapter contains instructions for monitoring complete and delayed publishing sessions, adjusting publishing schedules, and retrieving the histories of past sessions.

Publish Console

Throughout RealTime publishing, you will use the "Publish Console." In the "Publish Console," the system communicates information about current, scheduled, and past publishing sessions for all configured publishing destinations.

To open the "Publish Console": Log in to the administrator's interface and click **Publishing** on the button bar.

- The **Active tab** shows a summary of ongoing publishing sessions. While a publishing session is underway, you can open the "Publishing Status" screen to learn detailed information about the publishing process in real-time. You can also explore interactive on screen logs. Upon completion of the publishing session, you can download text formatted logs. More information about using the functions on the Active tab can be found in "Working with Active Publishing Sessions," on page 369. Information on working with logs can be found in "Viewing and Searching Logs," on page 382.
- The **Schedule tab** displays the number of approved assets for each destination and the schedule of publishing sessions. More information about using the functions on the **Schedule** tab can be found in "Working with Scheduled Publishing Sessions," on page 377.

- The **History tab** shows summary information on prior publishing sessions. More information on working with the **History** tab can be found in "Working with Prior Publishing Sessions," on page 378. From this tab, you can also access logs of prior publishing sessions. For information about logs, see "Working with Publishing Logs," on page 382.

Working with Active Publishing Sessions

The **Active** tab of the "Publish Console" shows information about all publishing sessions currently running. From the **Active** tab, you can click a destination name to access the "Publishing Status" screen for the selected session. On the "Publishing Status" screen you can monitor the progress of the session as it runs.

Instructions in this section apply to both modes of publishing: complete and delayed (described on page 368). The following topics are covered in this section:

- Monitoring a Publishing Session
- Resuming a Delayed Publishing Session
- Canceling a Publishing Session

Note

When a delayed publishing session pauses, it remains listed in the **Active** tab. Canceled sessions are moved to the **History** tab.

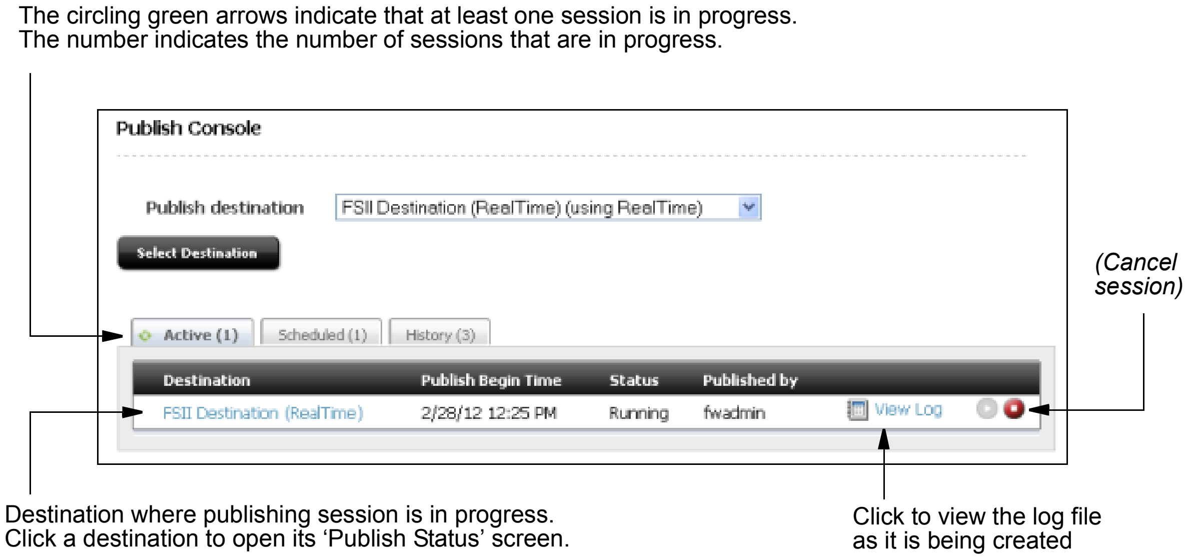
- Viewing log and downloading logs (see "Working with Publishing Logs," on page 382)

Monitoring a Publishing Session

To monitor a publishing session

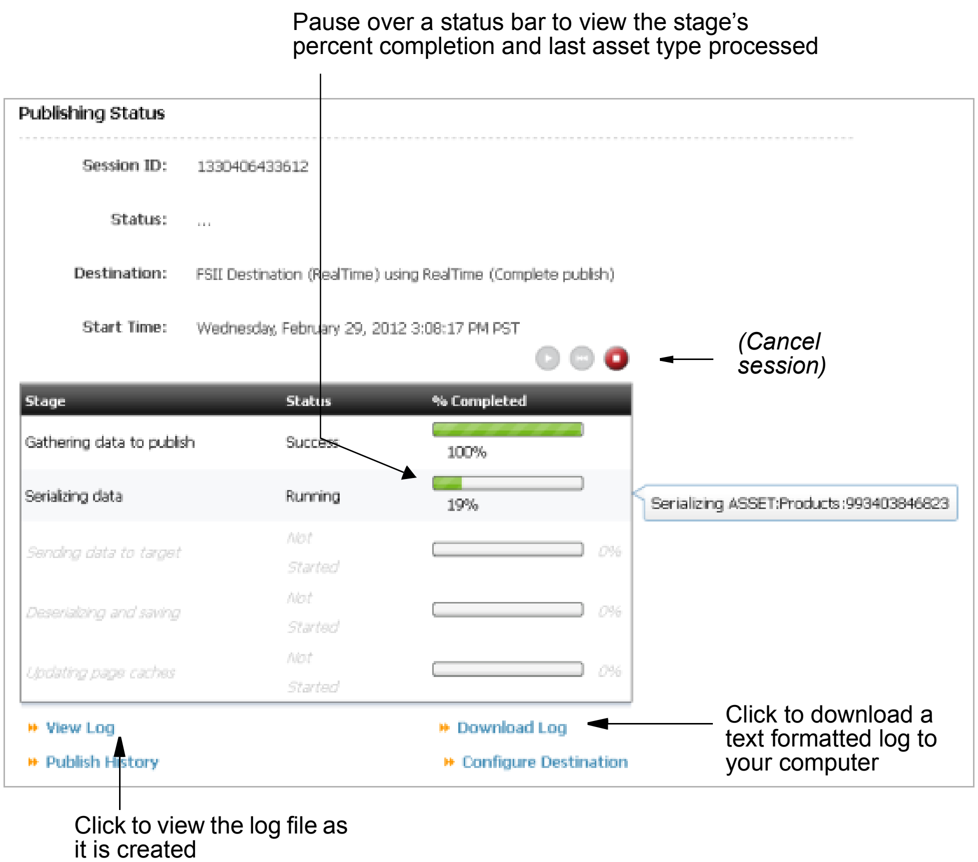
1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).

The “Publish Console” opens with the **Active** tab at the front.



2. To monitor a session, stay in the **Active** tab and select the destination where the session is running.

   The “Publishing Status” screen opens.



Notice the five status bars, one for each stage of publishing. The status bars are interactive.

- To view the percent completion of a stage, hold your cursor over the stage's status bar.
- As you hover, another tooltip opens showing the current asset type being processed. If you continue to hover, you will see the tooltip updating in real-time, always displaying the publishing system's last action.

(While a session is running, you can also view its log.)

Resuming a Delayed Publishing Session

Delayed publishing allows you to control when the system begins writing to the destination WebCenter Sites database. This mode of publishing can be used if you are publishing a large number of assets and wish to wait for a low traffic period on your web site before you begin saving data to the destination database.

When a session runs in delayed publishing mode, it pauses after the second stage (data serialization). Although the session pauses, no other publishing session can run to the destination until the current session completes. For the session to complete, you must resume the session. The final stages follow automatically.

To resume a delayed publishing session

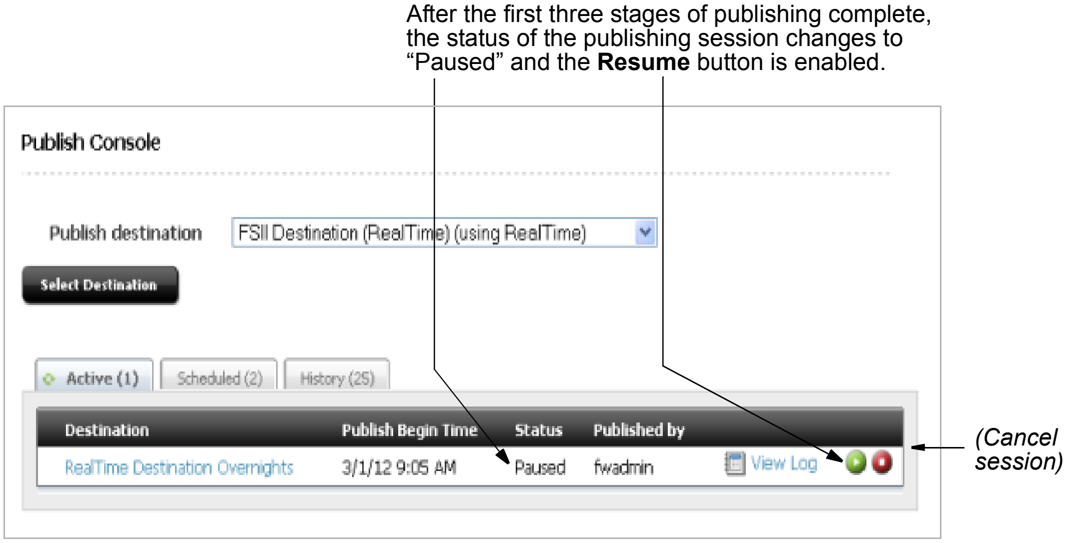
1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).
2. Stay in the **Active** tab and navigate to the destination where the publishing session is running. (If the destination name is not displayed, no session is running on that destination.)

When a publishing session is running, it appears in the **Active** tab of the "Publish Console"

(Cancel session)

3. Read the destination's "Status" column and do one of the following:

   - If the status is "Paused," it means that the first three stages of publishing are complete. To continue the session, click the **Resume** button (which is enabled only during a pause).



   - If the status is "Running," you can determine which stage is in progress. Open the "Publishing Status" screen:

     1) Select the destination where the session is running.

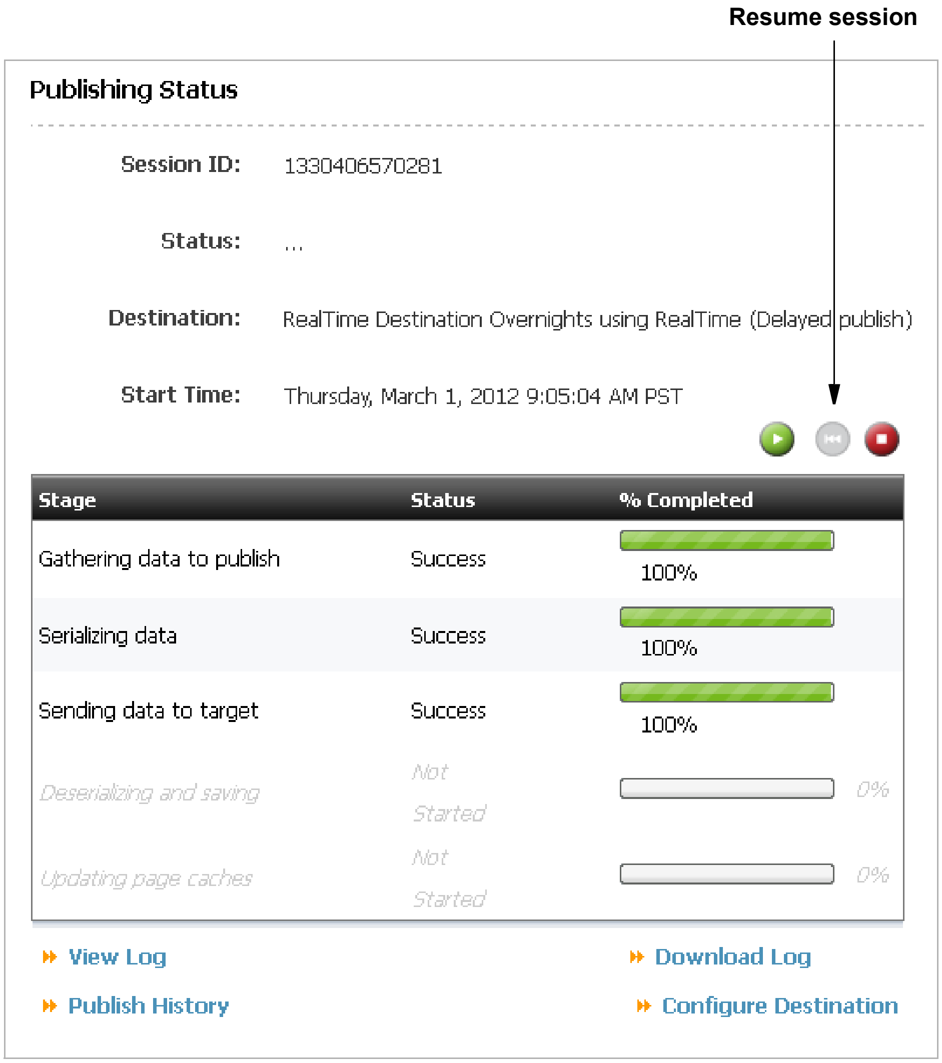
     2) In the "Publishing Status" screen, note the progress bars **and wait for the second stage to complete**. If the second stage is not allowed to complete, the session cannot be resumed.

        Note that the status bars are interactive:

        - To view the percent completion of a stage, hold your cursor over the stage's status bar.
        - As you hover, another tooltip opens showing the current asset type being processed. If you continue to hover, you will see the tooltip updating in real-time, always displaying the publishing system's last action.

        (While a session is running, you can also view its log.)

3) When the second stage is complete, click the **Resume** button.



4) When the publishing session is complete, all five stages display “Success” in the “Status” column.

4. Once a session is complete, you can access its summary information and complete log on the **History** tab of the “Publish Console.” To access the **History** tab:

   a. Return to the “Publish Console” (click **Publishing**, on the button bar).

   b. Click the **History** tab. More information about options on the **History** tab can be found in “Working with Scheduled Publishing Sessions,” on page 377.

Canceling a Publishing Session

Ongoing publishing sessions can be canceled (and restarted) by any user with publishing rights. When a session is canceled, session information is moved from the **Active** tab to the **History** tab. Here, the session can be rerun, but only if *at least* the second publishing stage was completed successfully before the cancellation **and** the session is the most recently canceled.

Note

You can cancel a publishing session from the "Publish Console" or from the "Publishing Status" screen. Be careful about which option you choose. If you prematurely cancel a session that you intend to redo, that session cannot be redone. When in doubt, cancel sessions from the "Publishing Status" screen, where you can verify that at least the second stage has been completed.

To cancel a publishing session from the 'Publish Console'

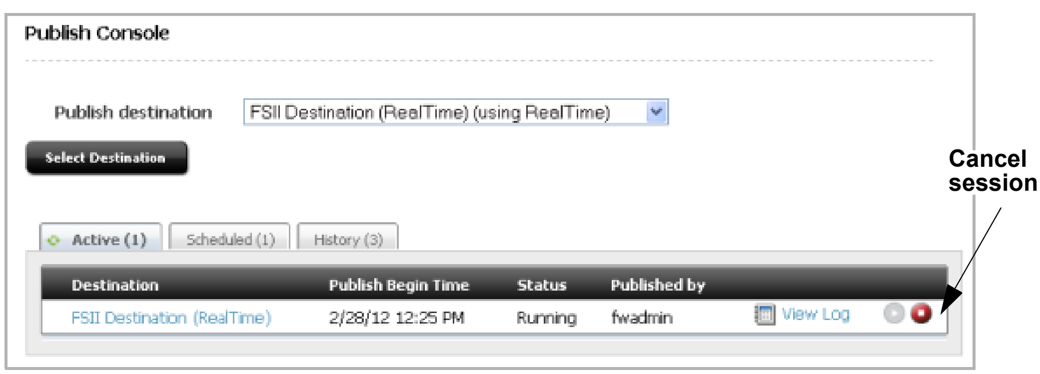
Note

Use the following procedure only if you are sure that the canceled session will not be rerun. Otherwise, if you plan to rerun the session, follow the steps on page 375 to cancel the session from the "Publishing Status" screen.

1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).

   The "Publish Console" appears, displaying the active publishing sessions in the **Active** tab. (If the **Active** tab is empty, no publishing sessions are in progress.)

2. Stay in the **Active** tab, navigate to the destination where the session is running and click its **Cancel** button.



   A message that the cancellation request has been sent appears above the list of active publishing sessions. WebCenter Sites looks for a stable stopping point; it may take a few moments for the session to cancel. Session information is then moved from the **Active** tab to the **History** tab.

To cancel a publishing session from the 'Publishing Status' screen

Note

You must use the following procedure if you plan to rerun the session you wish to cancel. The "Publish Status" screen enables you to monitor the publishing process to ensure that at least the second stage (data gathering and serialization) runs to completion. Canceling prior to that stage invalidates rerunning the session.

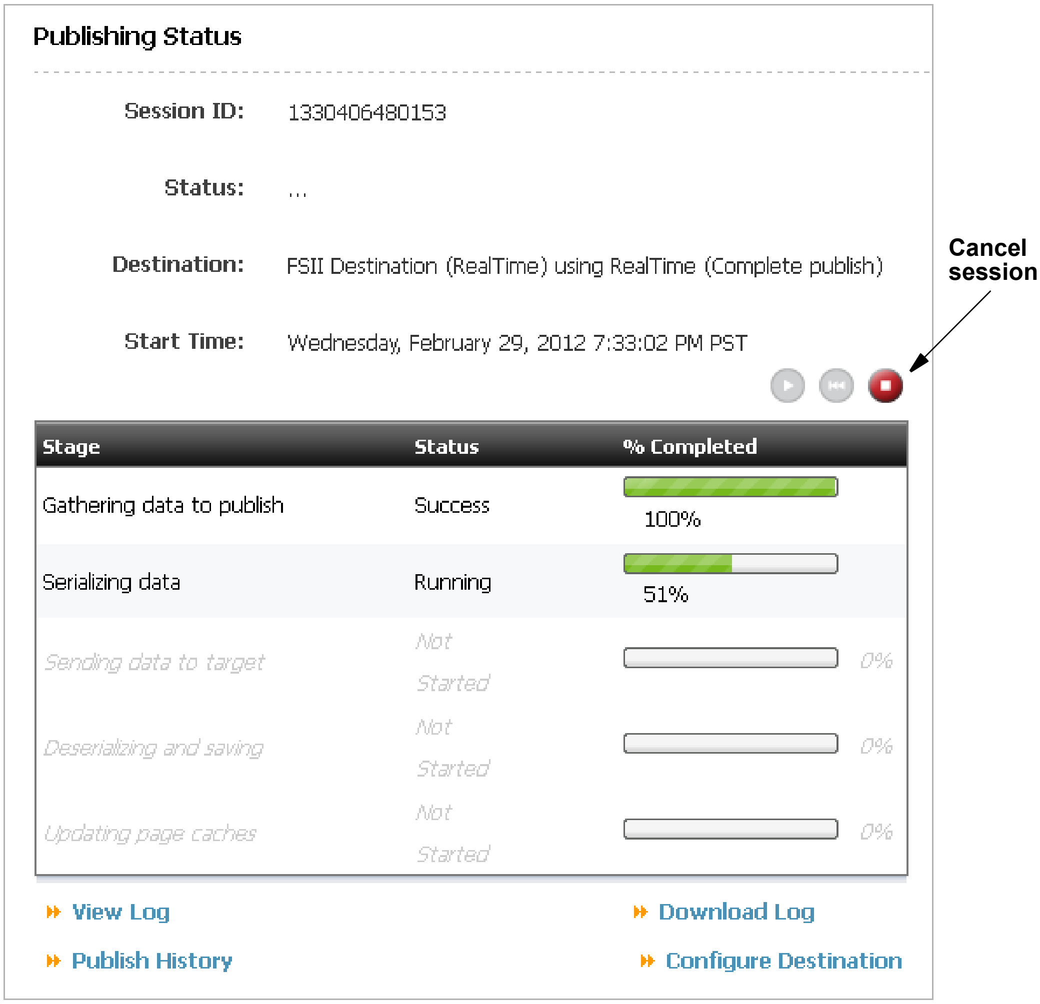
Note also that the option to redo a session is available if ***all*** of the following conditions hold:

- The first two stages of publishing were successfully completed.
- The session is the most recently canceled publishing session.

1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).

   The "Publish Console" appears, displaying the active publishing sessions in the **Active** tab. If the **Active** tab is empty, no publishing sessions are currently in process.
2. Stay in the **Active** tab and select the destination where the session is running.
3. In the "Publishing Status" screen, note the progress of the session and wait for at least the second stage to complete.

4. When the second (or later) stage has completed, click the **Cancel** button.



A message appears above the status column indicating that the cancellation request has been sent. WebCenter Sites looks for a stable stopping point; it may take a few moments before the session is canceled. Session information is then moved from the **Active** tab to the **History** tab.

5. When you are ready to rerun the session, select the **History** tab, navigate to the session, and click the **Redo** button. For more information, see "Redoing a Publishing Session," on page 380.

Working with Scheduled Publishing Sessions

If a destination is configured for scheduled publishing, its schedule is listed in the **Schedule** tab of the "Publish Console." (For instructions on configuring publishing schedules, see "Scheduling Publish Events," on page 319.)

Use the **Schedule** tab to:

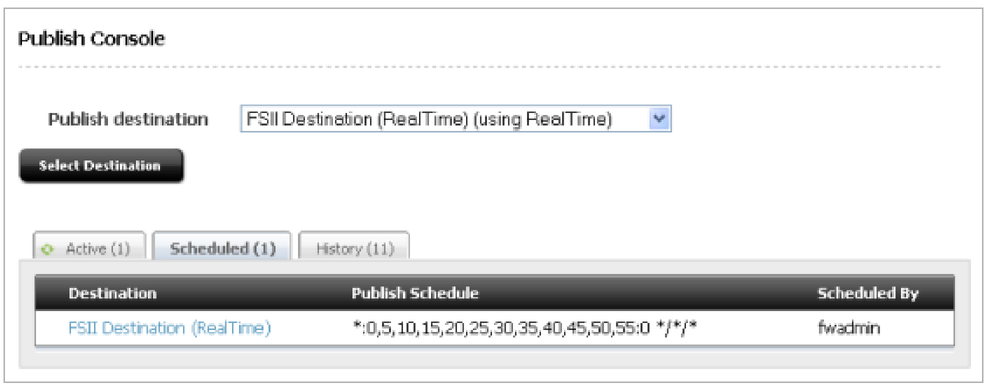
- View a destination's publishing schedule
- Edit publishing schedules

Viewing and Editing Publishing Schedules

To view publishing schedules for all configured destinations

1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).
2. Click the **Scheduled** tab.

   The **Scheduled** tab displays only destinations with a publishing schedule.



> **Note**
>
> A publishing schedule is expressed in the string format that is used by the `SystemEvents` table:
>
> `hours:minutes:seconds weekdays/days of month/months of year`
>
> For information about syntax, see "Reading the Schedule Abbreviations," on page 322.

3. To view a more user-friendly version of the publishing schedule, hold your cursor over the schedule.
4. To edit a schedule, click the destination name and specify the publishing conditions. (You can also disabled the schedule.)

Working with Prior Publishing Sessions

The Publishing Console's **History** tab provides summary information and detailed logs on every publishing session that has run in the past. Past sessions include completed, canceled, and failed sessions, but not delayed publishing sessions that have paused.

Note

When delayed publishing sessions pause, they remain listed on the **Active** tab. They are considered to be active sessions.

The **History** tab is interactive. You can hold your cursor over each destination to view the destination's summary information in a pop-up window, or you can select a destination to obtain its session log. To locate information quickly, you can sort the list of destinations by the following criteria: Destination, Publish End Time, Status, and Published by

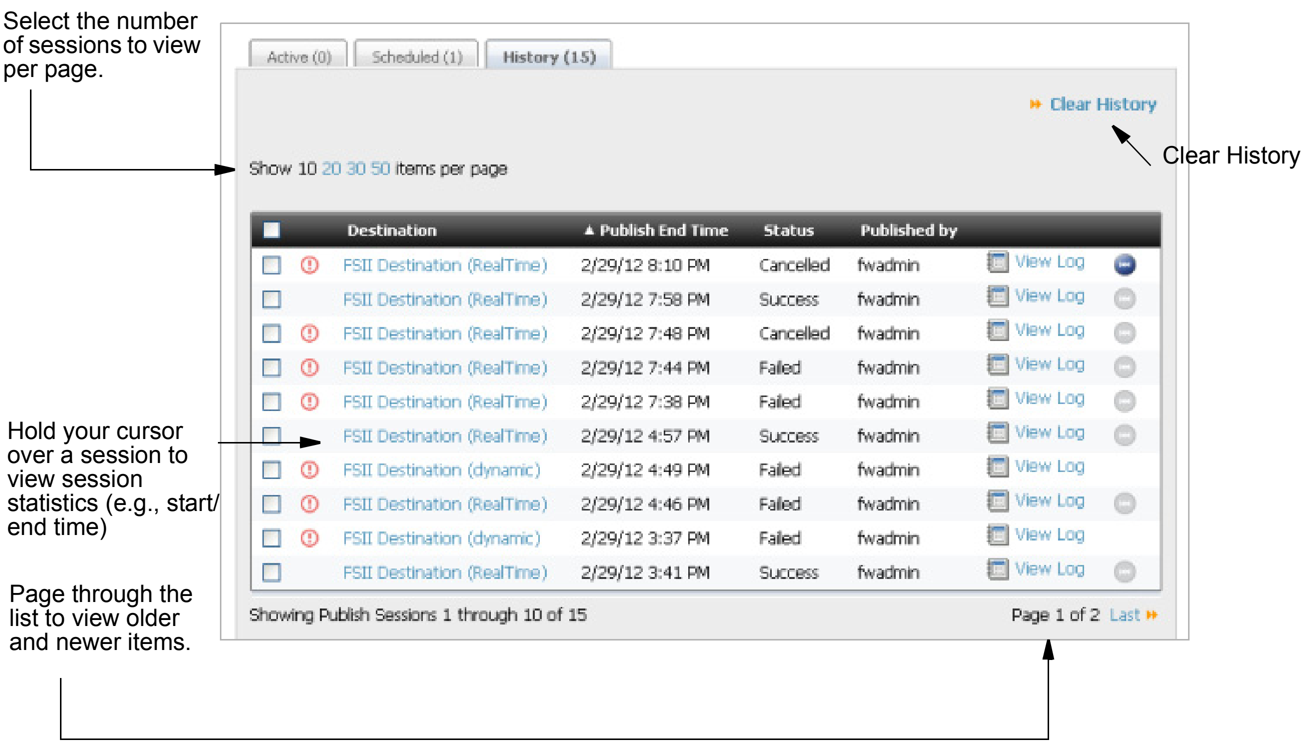
On the **History** tab, you can:

- View summary information about past publishing sessions.
  See "Viewing Past Publishing Sessions," on page 379
- Access the final "Publishing Status" screen for completed sessions.
  See "Viewing Past Publishing Sessions," on page 379
- Redo a canceled session.
  See "Redoing a Publishing Session," on page 380
- Access session logs
- Clear the entire history list or delete selected sessions

Viewing Past Publishing Sessions

To view past publishing sessions

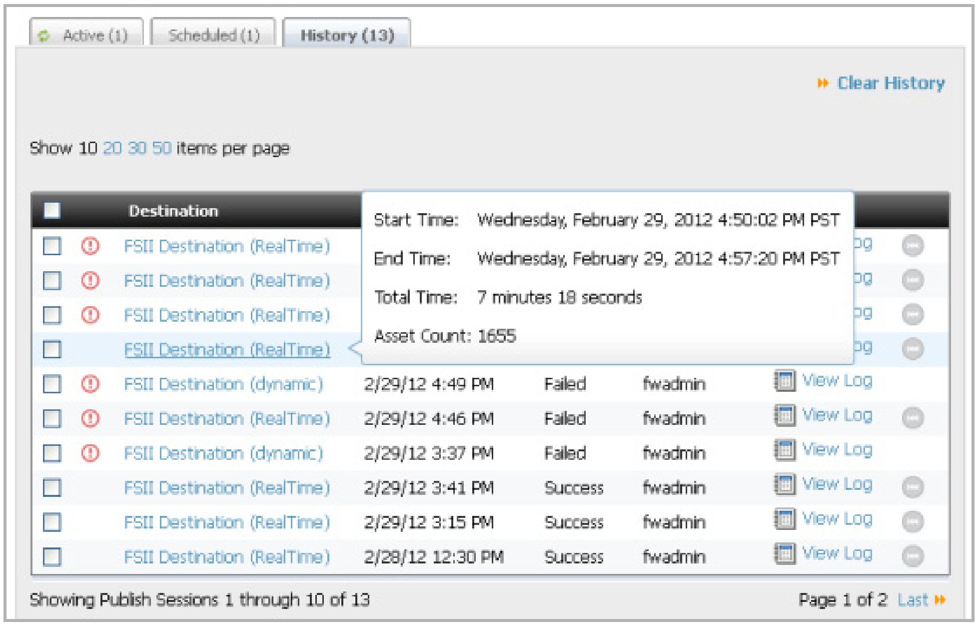
1. Open the “Publish Console” (in the administrator’s interface, click **Publishing** on the button bar).
2. Click the **History** tab.



3. To view a session’s summary information, hold your cursor over the destination where the session ran. (You can sort the **Destination** column by clicking its title.)

   A pop-up window displays summary information, as shown next.

The pop-up window shows when the session started and ended, how long it took to publish the assets, and the number of assets published.



4. To view and download a session log:
   a. Select the destination where the session ran.
   b. In the “Publishing Status” screen, click **View Log**.

      If a log entry exceeds 200 characters, click the **show all** link to display the entire entry. You can also search the log for specific entries.
   c. To download the log, click the **Download Log** link.

Redoing a Publishing Session

When a session stops (because of system error or user cancellation), you can rerun the session, meaning that you start the session again. When the session starts, it continues the publishing process from the last successfully completed stage. The option to redo a session is available if ***all*** of the following conditions hold:

- The data to be published was successfully gathered and serialized (that is, the first two stages of publishing were successfully completed).
- The session is the most recently failed or canceled publishing session.

The “Redo” option is useful when a disruption, such as a power failure, interferes with publishing.

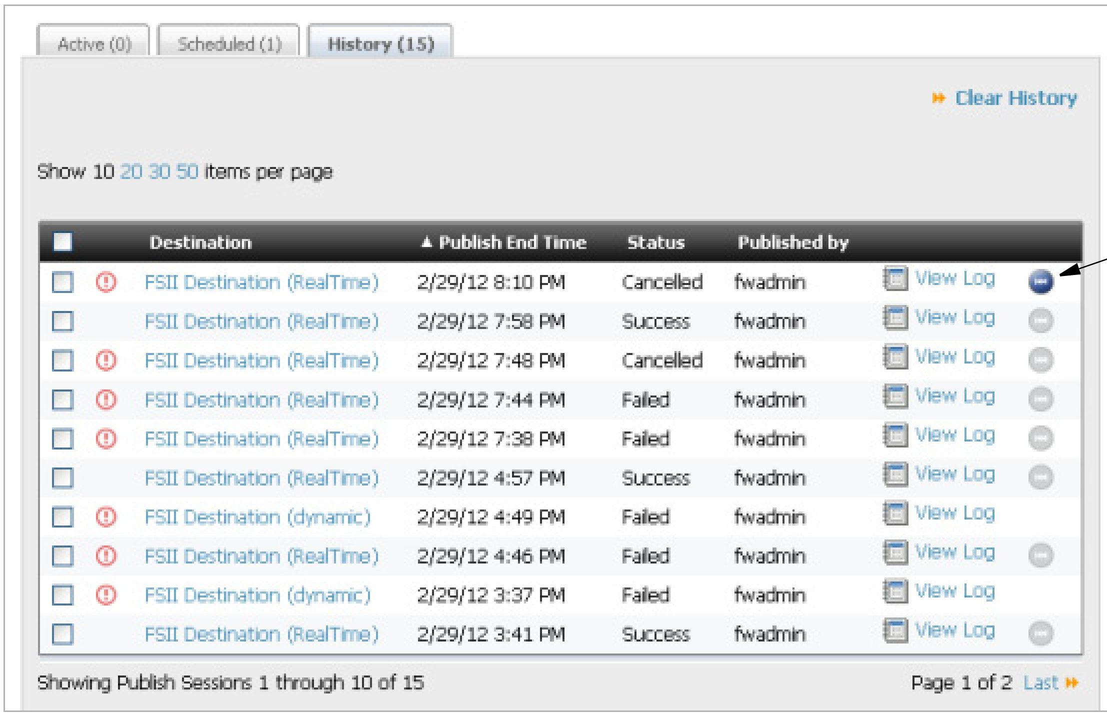
To redo a publishing session

1. Open the “Publish Console” (in the administrator’s interface, click **Publishing** on the button bar).
2. Click the **History** tab to view past publishing sessions.

   The list of past publishing sessions appears. The **Redo** button is active (in the right-hand column) only for the sessions that can be redone.

If the **Redo** button is disabled, then either this publishing session did not progress far enough to be redone, or it is not the most recently canceled (or failed) session.

Click the **Redo** button to begin a session at stage 3 of publishing (sending data to target)



3. Do one of the following:
   - Click the **Redo** button to restart the publishing session.
   - Redo the session from the “Publishing Status” screen:
     1) Click the destination link for the session you wish to redo.

        The final “Publishing Status” screen for the session opens.
     2) Click the **Redo** button to restart the publishing session.

   The message “`Restart request has been sent`” appears and the session moves from the **History** tab to the **Active** tab. The circling green arrows rotate in the corner of the **Active** tab, indicating a publishing session is active.
4. To monitor the session as it completes, click the **Active** tab followed by the destination name. (For more information, see “Working with Active Publishing Sessions,” on page 369.)

Working with Publishing Logs

Viewing and Searching Logs

RealTime publishing logs store a detailed account of the transactions that take place during a publishing session. The logs are viewable for active and past publishing sessions. They are also available for download, at any time, while sessions run and when they end.

Note

WebCenter Sites can be configured to generate verbose log files, containing more detailed information than regular log files. Because verbosity lengthens the publishing process, this option should be used only when needed (typically for troubleshooting).

WebCenter Sites can also be configured to automatically email text-formatted log files at the completion of a publishing session.

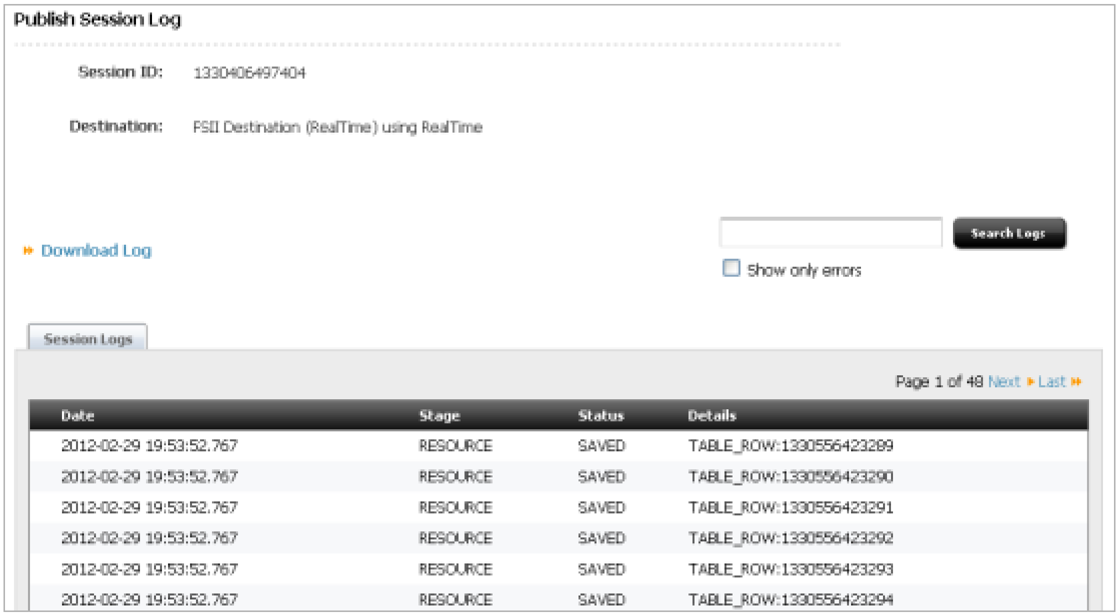
To configure log files and specify the email option, see "Step 4: Create a RealTime Destination Definition on the Source System," on page 354.

To view publishing logs

1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).

   The "Publish Console" appears with the **Active** tab open, displaying all currently running RealTime sessions.

2. To view the log of a current session, stay in the **Active** tab, navigate to the session, and click **View Log**. To view the log of a past session, click the **History** tab, navigate to the session, and click **View Log**.
   - The “Publish Session Log” screen opens. Additional pages are added as the session continues.



   - If a log entry exceeds 200 characters, an ellipsis (three periods) will appear at the end. To see the entire entry, hover the mouse over the entry. The entire entry opens in a window that floats over the log file. Note that the cause of the log event is identified.

3. You can search the log file in several ways:

- To search for specific entries, enter a search term in the “Search” field (at the top right of the screen).

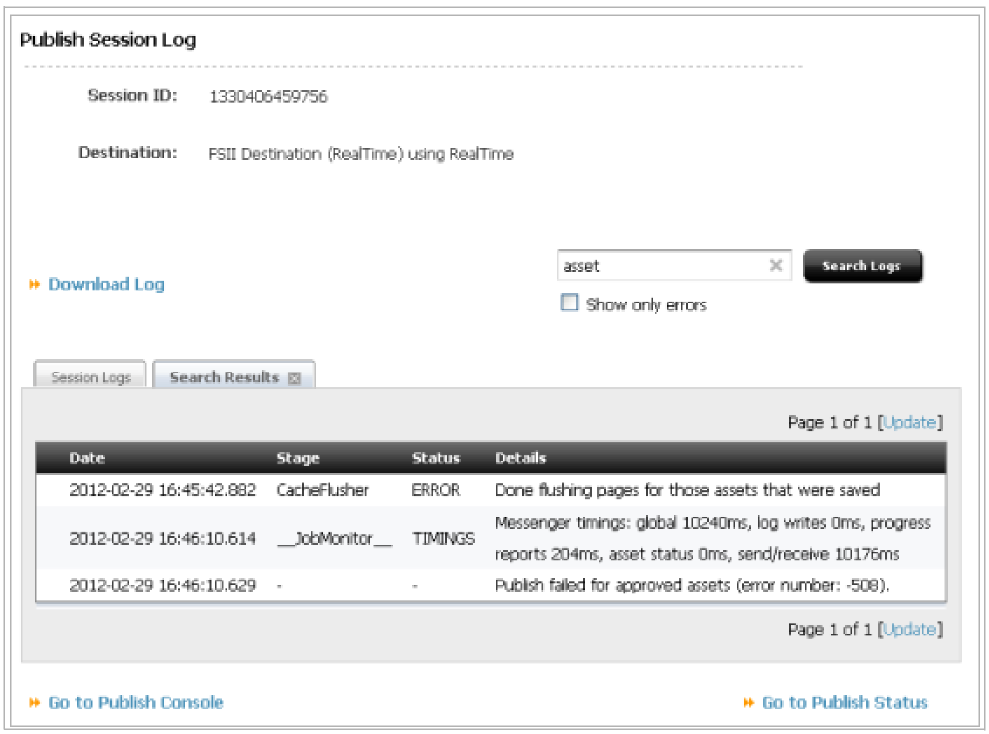
  Note

  The search utility recognizes terms that are used in the following columns: Stage, Status, and Details. Wildcards and boolean operators are not recognized.

- To view only error messages, select the **Show only errors** checkbox and leave the “Search” field empty.
- To filter error messages by specific terms, select the **Show only errors** checkbox and enter a search term in the “Search” field.

a. Click **Search**.

The **Search Results** tab opens (next to the **Session Logs** tab).



b. To exit the **Search Results** tab, click the **X** on the tab.

You are returned to the **Session Logs** tab.

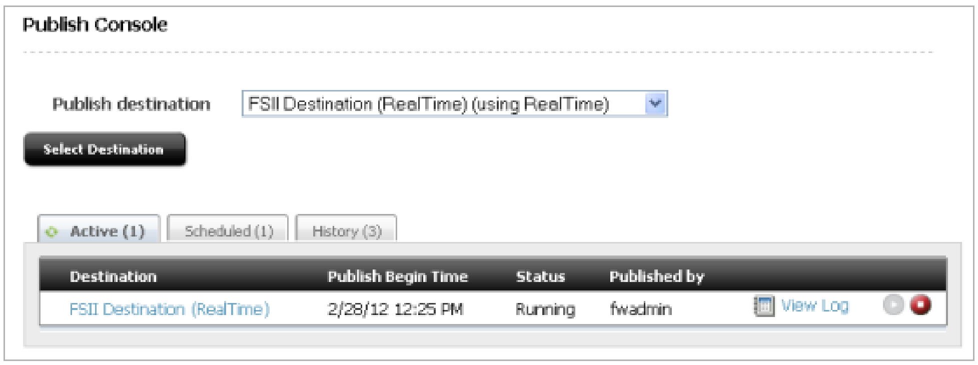
Downloading Logs

A publishing log is available for download at any time during the session and when the session ends. You can download a log file from several places in the RealTime publishing interface, as shown in the steps below.

To download a session log

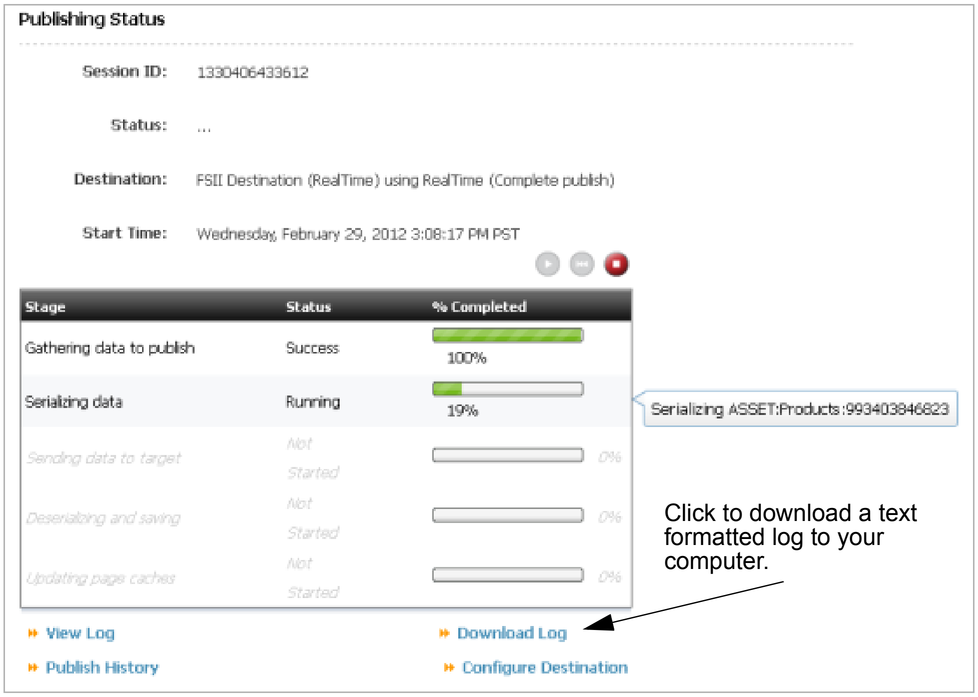
1. Open the "Publish Console" (in the administrator's interface, click **Publishing** on the button bar).

   The "Publish Console" opens with the **Active** tab at the front. The **Active** tab lists publishing sessions currently in progress.



2. Download the log file for an active session or a past session:
   - **Active** tab: Select a destination name, and click **Download Log**.
   - **History** tab: Select a destination name, click **View log** and select **Download Log**.

The “Publishing Status” screen opens.



3. When the session is complete, click **Download Log** and follow the prompts.

   The log file is in text format and can be opened in any text editor.

Deleting Publish History

It is recommended that you delete the publish history regularly ensure proper operation and performance, as all data related to publishing is stored in the database, and the size of these tables can become a performance problem over time. While regular publishes will not greatly increase the size of this table, large publishes (such as the initial publish of a site, especially when it has failed) can leave large amounts of data behind.

Deleting a publish session will do the following:

- Delete all messages associated with the session from the database tables.
- Clear out any data associated with it from `FW_PUBDATASTORE` on this system (data is often left in this table when a publish fails).
- Attempt to access the destination and clear associated data from `FW_PUBDATASTORE` (data is often left in this table when a publish fails).

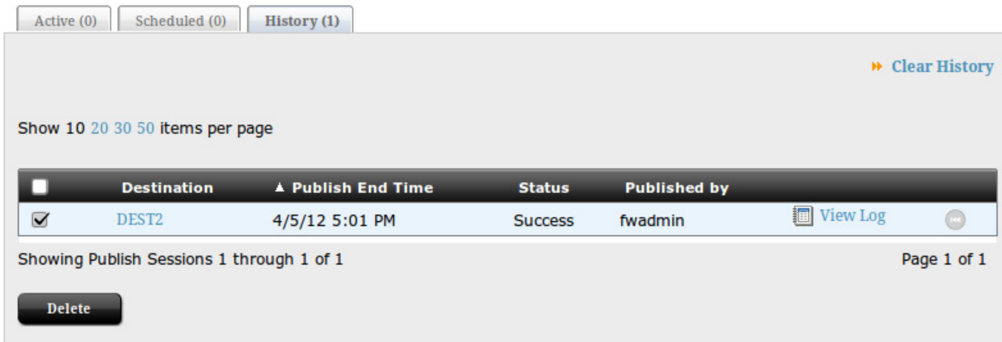
To delete a session log

1. Open the “Publish Console” (in the administrator’s interface, click **Publishing** on the button bar).

   The “Publish Console” opens with the **Active** tab at the front.

2. Select the **History** tab.

All available logs are displayed.



3. Check the box next to each log or logs you want to delete.
4. Click **Delete**.

   The selected logs are deleted.

> **Note**
>
> As data is in `FW_PUBDATASTORE` is stored using Large Binary Objects it may be necessary have the DBA manually clean up the Large Binary Data after performing this operation.

Chapter 21

On Demand Publishing and Asset Unapproval

If you are using the RealTime publishing method, you can selectively publish approved assets. You can also unapprove multiple assets in bulk.

This chapter contains the following sections:

- Using On-Demand Publishing
- Unapproving Selected Assets from a Publishing Session

Using On-Demand Publishing

In RealTime publishing, you can publish assets on-demand. That is, you can immediately and selectively publish any of the assets on the approved assets list.

As with scheduled RealTime publishing, an asset will be published with all its dependent assets. Therefore, the dependent assets are also placed in the On-Demand Queue.

The list of approved assets is searchable, making it easier to find the assets you need to publish. After the selected assets are published, they are removed from the approved assets list for the next publishing session.

Note

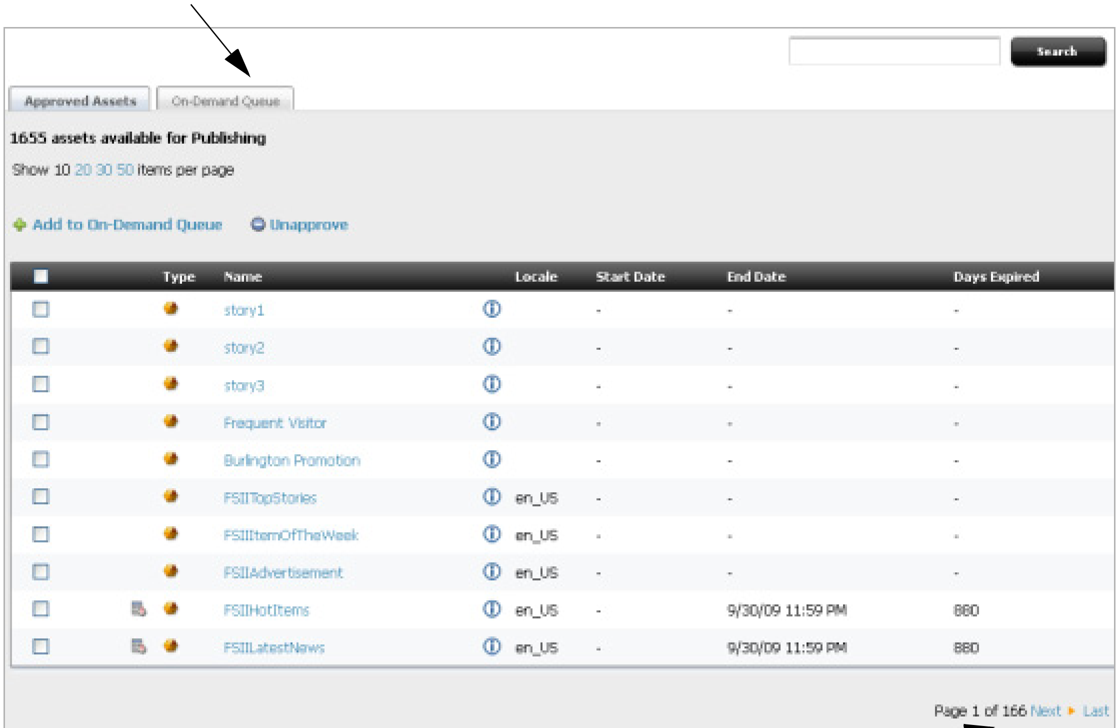
On Demand publishing can be either complete or delayed. For more information about publishing modes, see Chapter 20, "Working with RealTime Publishing Modes."

To select approved assets for immediate publication

1. Open the Publish Console by clicking **Publishing** on the button bar.
2. From the "Publish Destination" drop-down list:
   a. Select the RealTime publishing destination to which you will publish the assets.
   b. Click **Select Destination**.
3. On the "Publish Destination" screen, click the ***x*** **assets are ready for publish** link.

The **On Demand Publishing** tab opens. The **Approved Assets** tab lists all assets approved for the upcoming publishing session.

Select to view the On-Demand Queue.



Page through the list of approved assets.

4. (Optional) Search through the approved assets.

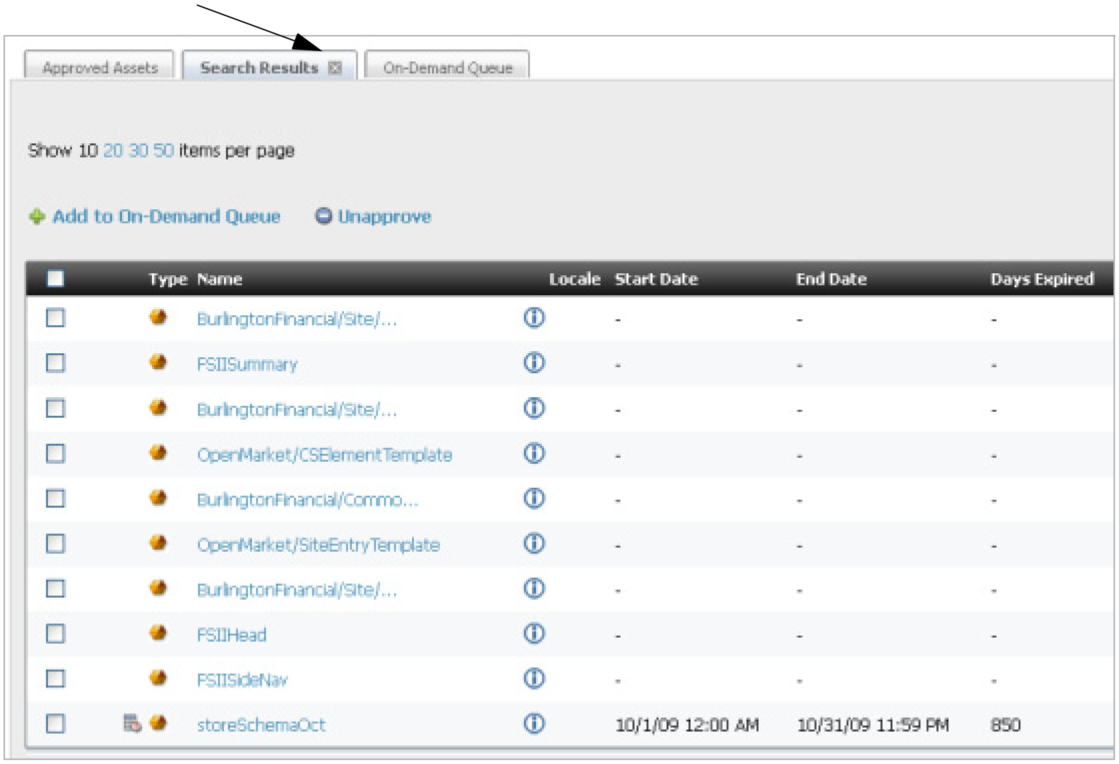
a. Enter a search term in the Search text box.

- Search runs across asset names and descriptions.
- You cannot use wildcard and boolean operators.

b. Click **Search**.

The **Search Results** tab opens, displaying the results of your search.

Click the **X** to close this tab and
return to the Approved Assets tab.



5. Add assets to the On-Demand Queue.

> **Note**
>
> On-Demand supports immediate publication. Therefore, selections that you place into the On-Demand queue remain there only throughout the current WebCenter Sites session. Once you log out (or the system logs you out), the On-Demand queue is cleared.

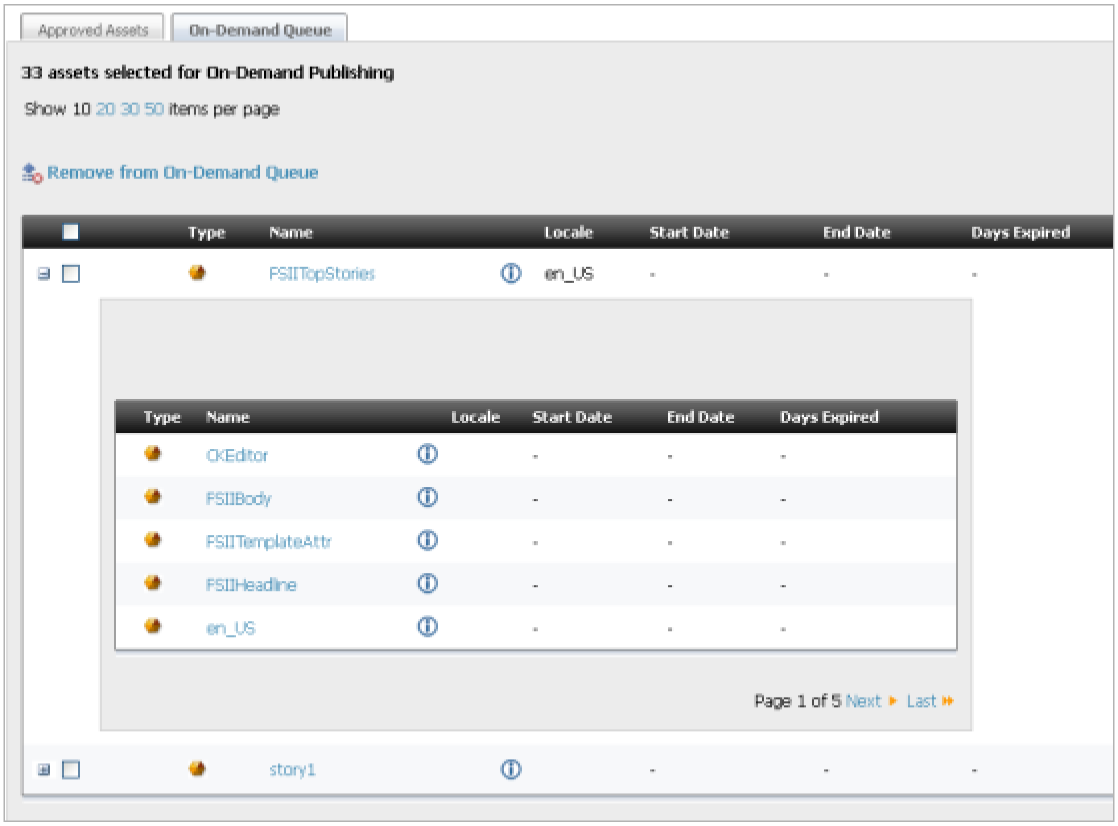
On either the **Approved Assets** tab or the **Search Results** tab, select the assets you wish to add to the On-Demand Queue, then click **Add to On-Demand Queue**.

A message is displayed while the dependencies are calculated. Once this process completes, the selected assets are removed from the **Approved Assets** tab (and **Search Results** tab, if open). The On-Demand Queue is updated with the selected assets and their dependent assets.

6. Publish the On-Demand Queue. Do one of the following:

- If you are sure that the assets you selected can be published, click **Publish On-Demand Queue** (on the **Approved Assets** tab).

- If you wish to review the assets before you publish them, switch to the On-Demand Queue by clicking the **On Demand Queue** tab.



- To view an asset's dependencies, click the plus mark at the far left of the asset's row.
- If you decide not to publish assets after you place them in the On-Demand Queue, select them and click **Remove from On-Demand Queue**. The assets will be removed from the On-Demand Queue and placed back in the **Approved Assets** list. (A dialog box prompts you to confirm your action.)

7. Click **OK**.

   The Publish Console opens showing the publishing session in process.

8. For instructions on observing the publishing session, see Chapter 20, "Working with RealTime Publishing Modes."

Unapproving Selected Assets from a Publishing Session

If you change your mind about publishing approved assets, you can selectively remove them from the approved assets list.

To unapprove assets

1. Open the "Publish Console" by clicking **Publishing** on the button bar.

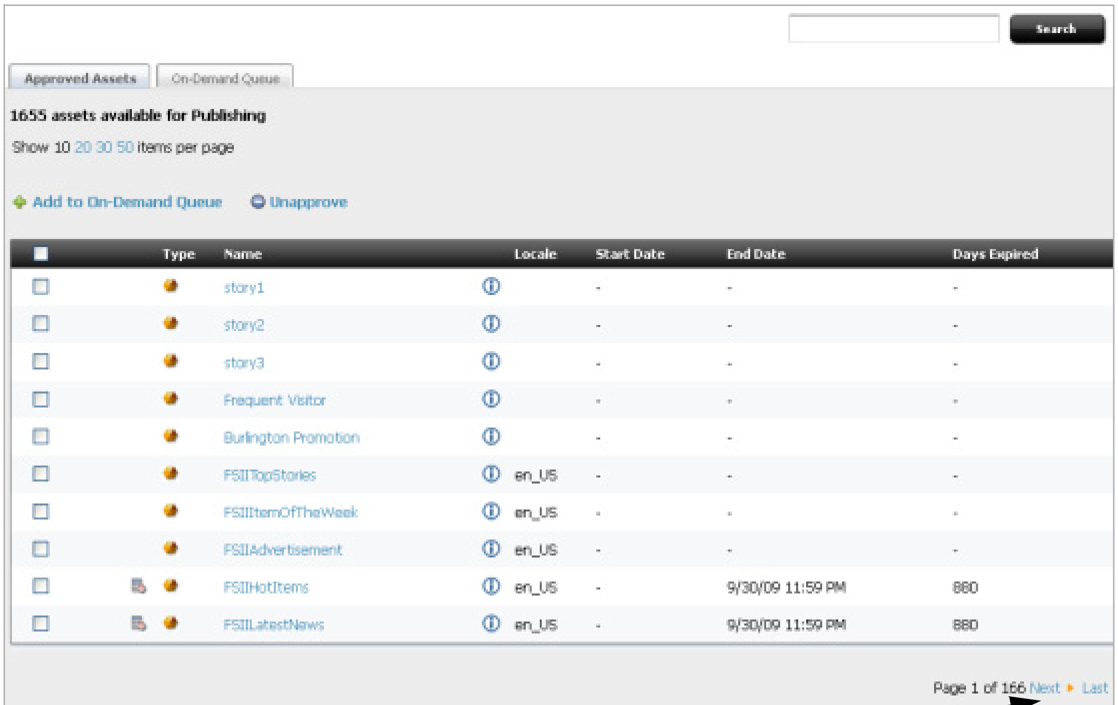
2. From the "Publish Destination" drop-down list:

   a. Select the RealTime publishing destination you wish to publish assets to.

   b. Click **Select Destination**.

3. On the "Publish Destination" screen, click the link ***x* assets ready for publish**.

The On Demand Publishing screen opens. This screen displays the **Approved Assets** tab and the On-Demand Queue. The **Approved Assets** tab lists all assets approved for the upcoming publishing session.



Page through the list of approved assets.

4. (Optional) Search through the approved assets.

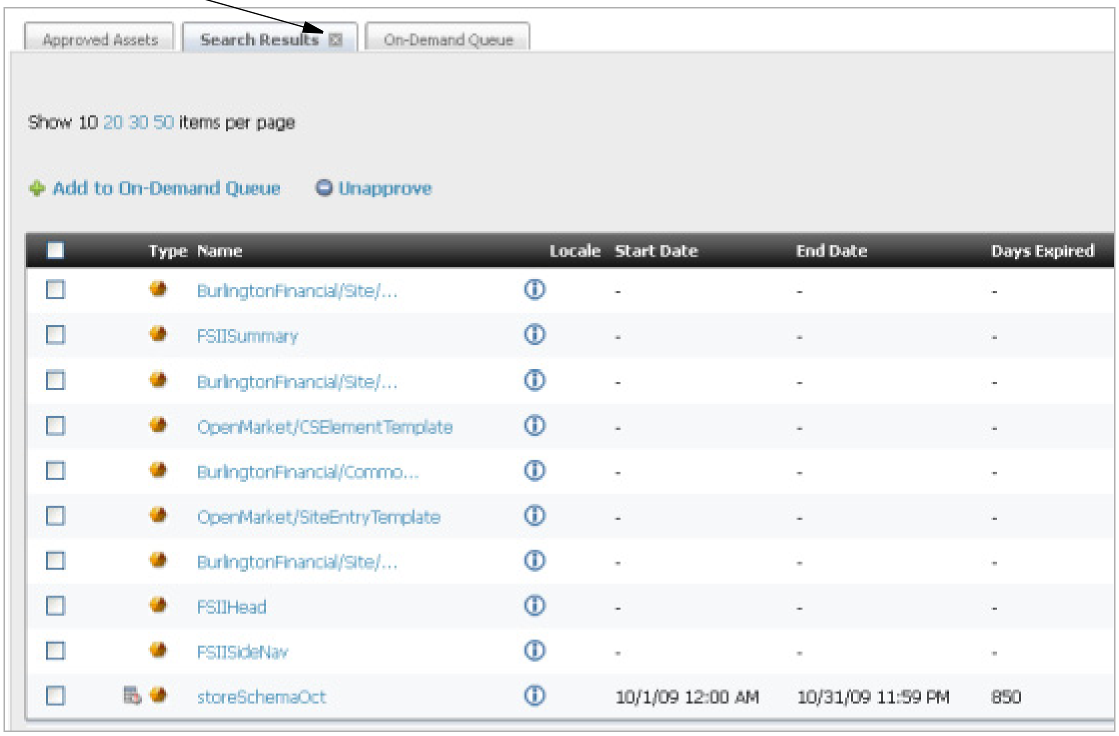
   a. Enter a search term in the Search text box.

      - Search runs across asset names only.
      - You cannot use wildcard and boolean operators.

   b. Click **Search**.

The **Search Results** tab opens, displaying the results of your search.

Click the **X** to close this tab and return to the **Approved Assets** tab.



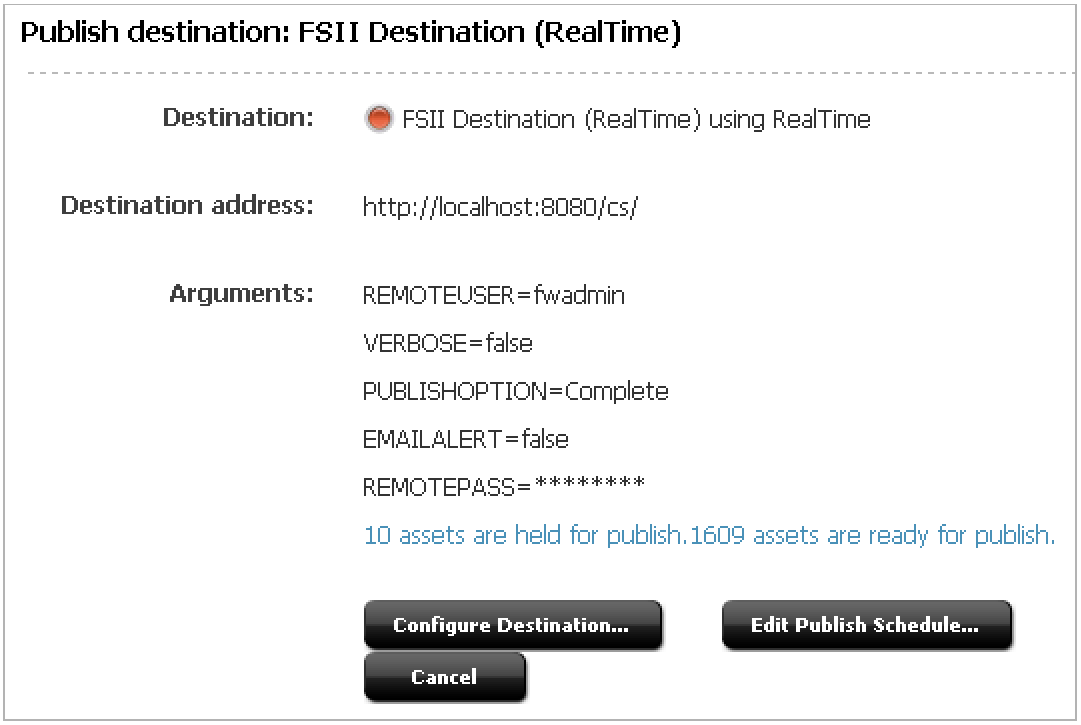
5. On either the **Approved Assets** tab or the **Search Results** tab, select assets to unapprove. Click **Unapprove**.

   The assets you selected are removed from the Approved Assets list.

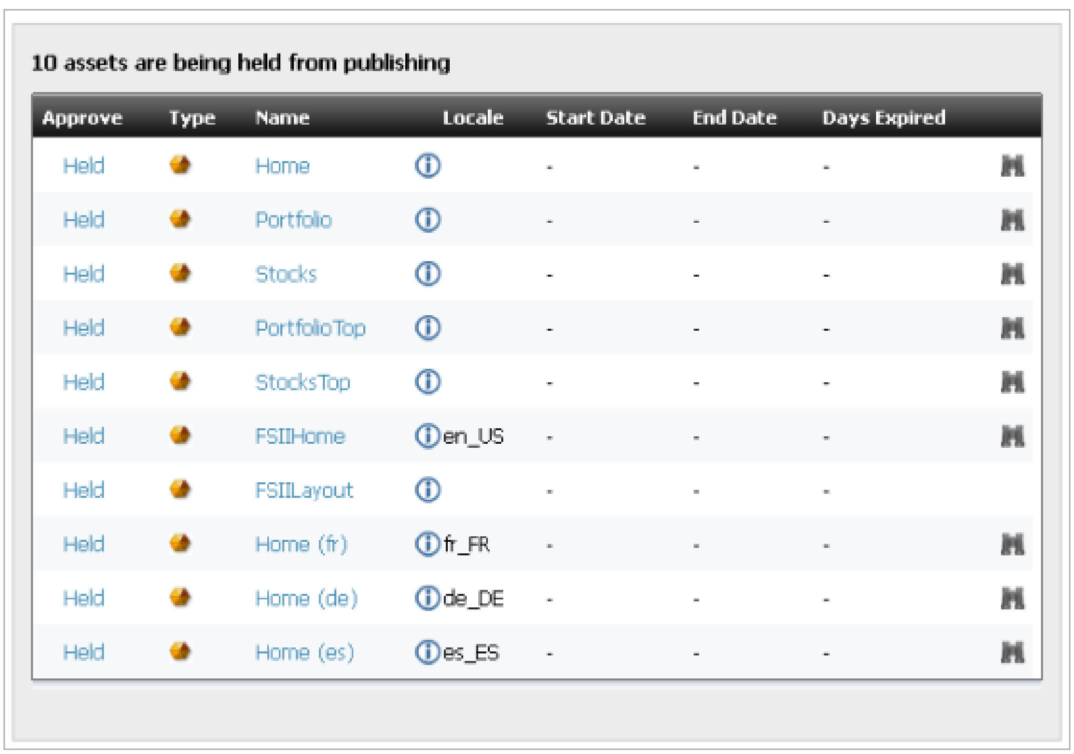
   If the asset you unapprove has a dependent asset that is approved for this publishing session, it is placed into a hold queue.

6. (Optional) Approve assets held from publishing:

a. Click **Back** to return to the Publish Destination screen. Note that there are now two text links. One lists the assets held for publishing, the other the assets ready for publishing. Click the ***x* assets are held for publish** link.



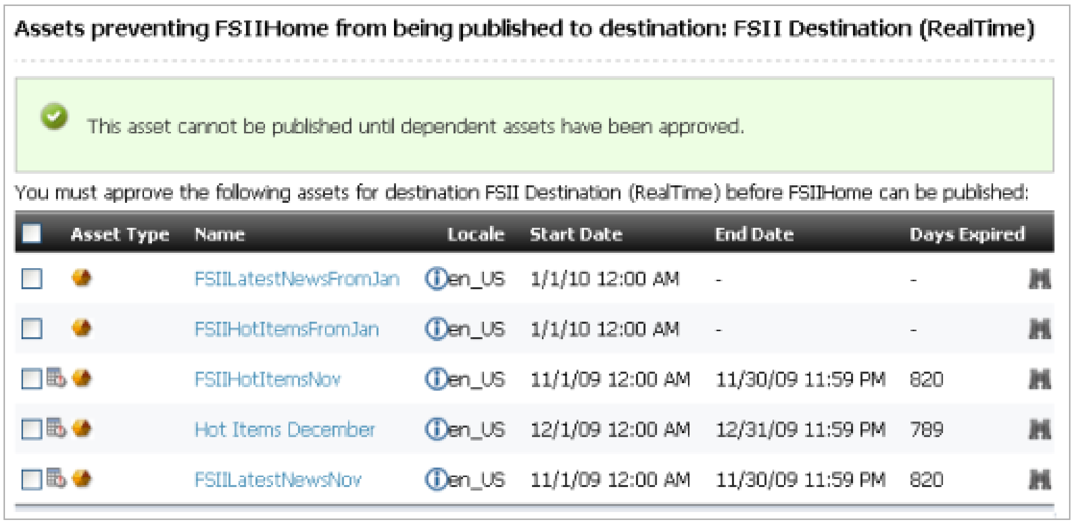
The held queue opens.



b. Click the **Held** link for the asset you wish to approve for the publishing session.

The **Assets Preventing *`Asset Name`* From Being Published to *`Destination`*** screen opens.

- If the asset has no dependencies, a message appears stating that the asset is approved.
- If the asset has dependencies, a list of those dependent assets is shown.



After reviewing the list of dependent assets, select all of the assets on the list by checking the check box in the header row of the table. Then click **Approve**.

The following occurs:

- A message indicates that the assets have been approved and are ready for publishing.
- The asset and its dependent assets are placed in the approved assets queue.
- You are returned to the held queue.

c. (Optional) Repeat step b to approve another held asset.

Part 5

System Configuration Procedures

This section describes the configuration options that affect all the sites on your Sites system.

It contains the following chapters:

Chapter 22

Configuring the User Interfaces

There are several Sites user interface features that must be configured before content providers can use them.

Some of these features are alternate interfaces to the WebCenter Sites interfaces: Sites Desktop, Sites DocLink, and the Web Mode of the Contributor interface. Others change the WebCenter Sites interfaces (locale settings) and the behavior of the preview function.

This chapter contains the following sections:

Setting the Locale for the WebCenter Sites Interfaces

If your organization purchased and installed one of the Sites language packs, you must configure a default locale for your Sites system to use.

There are several options and properties that you use to set the locale:

- The **Locale** item in the **Admin** tab sets the default locale for the WebCenter Sites interfaces. Individual users can override the default and use a different locale (if one is present) by setting a preference.
- The `cs.emailcharset` and `cs.emailcontenttype` properties in the `futuretense.ini` file determine the character set that is used in the e-mail messages that the workflow system sends. These properties—which must be set to a character set that can appropriately represent characters for all of the locales that your content providers can select—are usually set during installation.
- If your system is using the UTF-8 character set and you are using the Mirror to Server publishing delivery type, you must set the value for the `cs.mirrorrowsperpost` property in the `futuretense.ini` file to 4 or lower.

System Default Locale for the Sites Users

When a language pack is installed, the installer writes the language and country code for the locale of the language pack to the `LocaleMap` database table. For example, Canadian French is `fr_ca`. When there is more than one locale listed in the `LocaleMap` table, you, as the administrator, need to specify which should be the default locale.

You can override the default locale for individual users by setting it in their user profile. See "Working with User Profiles and User Attributes," on page 93 for information about how to set a user's locale preference.

WebCenter Sites determines which language to use for an individual user through its universal session variable named `locale`. This variable contains the language and country code associated with the user who is logged in.

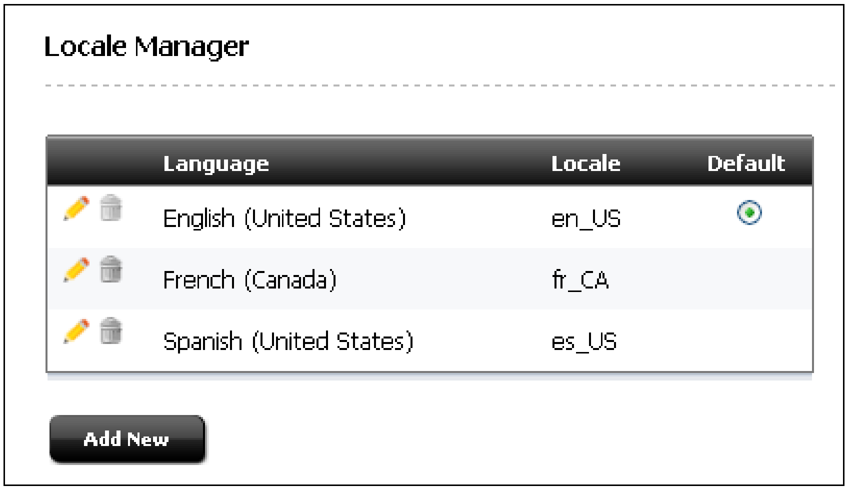
When a user logs in to WebCenter Sites, Sites obtains the value for the `locale` variable as follows:

- If the user's user profile has a value for the locale user attribute, WebCenter Sites obtains and uses it.
- If the user profile does not specify a preference, WebCenter Sites attempts to match the locale setting of that user's browser to a locale from the `LocaleMap` table.
- If the user profile does not set a preference and WebCenter Sites cannot match that user's browser locale, WebCenter Sites uses the default locale that is set for the system.

To set the system default locale

1. In the **Admin** tab, double-click **Locale**.

   WebCenter Sites displays the “Locale Manager” form.



2. In the “Locale Manager” form, do the following:
   a. Click the **Edit** icon next to the desired locale.

      A radio button will appear on the screen in the Default column.
   b. Select the **Default** radio button next to the desired locale.
3. Click **Save**.

System Locale Properties

The properties in the following list should have been configured during the installation of the language pack on your Sites system. They are listed here for troubleshooting purposes. For example, if the workflow e-mails are displaying odd characters, perhaps the e-mail properties were not set correctly.

For instructions on how to start and use the Property Editor to verify or modify property values, see the *Oracle WebCenter Sites Property Files Reference*.

Property	Property File	Description
`cs.contenttype`	`futuretense.ini`	Determines the HTTP header character set for all WebCenter Sites pages, as well as other HTTP interaction (Sites Explorer, CatalogMover, and so on). This value is set to `text/html; charset=UTF-8` by default. This value must be set to an appropriate value for the online site that your Sites system is delivering.

Property	Property File	Description
cs.emailcharset	futuretense.ini	Determines the character set used in the subject lines of e-mail messages sent by WebCenter Sites. (Typically these are workflow e-mails.)
cs.emailcontenttype	futuretense.ini	Determines the character set used in the body of e-mail messages sent by WebCenter Sites.
cs.mirrorrowsperpost	futuretense.ini	Specifies the number of table rows that can be mirrored during each HTTP POST while a mirror publish session is underway. If you are using UTF-8 and your database serves non-ASCII text, this value must be set to 4 or lower.
xcelerate.charset	futuretense_xcel.ini	Determines the HTTP header character set used for all WebCenter Sites pages. This value must be set to UTF-8.

Single-Language Restrictions

Although you can configure a multi-lingual management system, certain parts of the user interface can be displayed in one language only.

For example, the names of tables and columns in the WebCenter Sites database as well as individual items such as categories and source codes can have one name only. This means that although much of the text on an individual WebCenter Sites form can be displayed in any of the languages that you have installed on your Sites system, items such as field names and asset type names can be displayed in one language only.

Following is a list of items in Sites that can have one name only, which means that they can be displayed in one language only:

- Asset type names
- Field names
- Asset names
- Categories
- Source codes
- Tree tab names
- Site names
- Names of workflow building blocks (actions, e-mail objects, conditions, states, steps, processes)
- Role names
- Start menu items—both Search and New

On a system that provides two or more languages, you must determine which language is going to be used by the majority of content providers and then use that language to name your sites, tabs, asset types, and so on.

Locale and Asset Types

When your site developers create asset types (either basic or flex), they can specify the field names for those asset types in whatever language they have available on the Sites system. However, there are several core columns/fields that WebCenter Sites creates for all asset types by default:

- id
- name
- description
- status
- createdby
- createddate
- updatedby
- updateddate
- startdate
- enddate
- subtype
- filename
- path
- template
- category

These column names are created in English, by default. When WebCenter Sites displays them as fields in its interfaces, it uses the language that is set as the system default language.

For descriptions of the default columns for asset types, see the "Data Design" section in the *Oracle WebCenter Sites Developer's Guide.*

Locale and the Article Asset Type

If your site developers decided to use the sample site asset type named Article, you must also modify the value for the `xcelerate.body.length` property in the `futuretense_xcel.ini` file. This property determines how many characters are stored in the `urlbody` column of the `Article` table.

Because `urlbody` is a URL column, the data entered in it is actually stored as a file outside of the WebCenter Sites database. However, the first *n* number of characters, where *n* equals the value specified for the `xcelerate.body.length` property, is also stored in the `urlbody` column so that you can search for text in the body of an article asset with the search feature in the administrator's interface.

When this column holds non-ASCII text, the value for this property should be set to 500.

Sites Desktop

Sites Desktop enables Microsoft Word users to create, import, and edit Word documents as WebCenter Sites assets using the familiar Word user interface. They save these assets to, and open them from, the WebCenter Sites database.

> **Note**
>
> Sites Desktop supports the use of standard Word documents and templates. In Microsoft Word 2000, web page templates that enable Word users to create HTML pages using Word were introduced.
>
> The Word web page templates are neither supported nor needed by Sites Desktop.

Overview

When Sites Desktop is configured for a system, users can log in to WebCenter Sites from a special toolbar in the Microsoft Word user interface.

You, the administrator, configure which asset types will be available in the Word interface. You enable the asset types for Sites Desktop and then determine which fields the Word users can enter data for. This information determines the appearance and the functionality of the WebCenter Sites toolbar when users create assets of those types in Word. Your content providers use the WebCenter Sites toolbar to insert field markers that mark the data for each field.

When someone saves the Word document as an asset, Sites Desktop sends that asset to WebCenter Sites. A temporary copy of the document is also saved locally on the user's hard drive.

The file conversion subsystem converts the binary form of the document into XML, which, in turn, is parsed into a form suitable for storing in the WebCenter Sites database. The asset is then managed as follows:

- The Word document is also stored with the asset. That document is retrieved when a user opens the asset in Word.
- When a visitor to your online site (on the delivery system) requests the asset, WebCenter Sites serves the asset, not the Word document.

> **Note**
>
> If you use WebCenter Sites' interfaces to edit an asset that was created in Sites Desktop, the link between the asset and Sites Desktop will be irreversibly severed when you save the asset. You will no longer be able to edit the asset in Sites Desktop. WebCenter Sites warns you of this fact when you open the asset in the "Edit" form. A link to the Sites Desktop client installer is also provided at that time.

Sources of Data for a Word Asset

The Microsoft Word document is the complete source for a word asset. There are several ways to enter data for an asset's field in the Word interface:

- A user enters text and then marks it with the appropriate field marker.

- Certain required fields such as **Name** and **Description** are hidden in Word. For these fields, the user selects the field name from the WebCenter Sites toolbar and then enters the data in a dialog box.

  This kind of data is considered to be Word metadata.

  In Word, you can select **File > Properties** and click the **Custom** tab to see these hidden fields and other metadata that results from storing the Word asset in the WebCenter Sites database. Do **not** modify this data.

- Obtain information from Word template `.dot` files. You can create Word templates for the Word assets that provide field/value pairs for specific fields. (You can also use Word templates to create a form for your content providers to use to enter information about their Word assets.)

Preserving the Formatting of the Word Data

You, the administrator, determine how the data from an individual field is managed by using the **Preserve formatting** option when you configure asset types for Sites Desktop.

When you select **Preserve formatting** for a field, WebCenter Sites converts the Word format to the corresponding HTML format and stores the HTML in the column that represents the field in the WebCenter Sites database. Paragraph breaks are converted to `<p>` tags, bold text is marked with `<b></b>` pairs, tables are stored as HTML tables, and so on.

Therefore, it is best if you use the **Preserve formatting** option for fields that contain text only. Do not use the **Preserve formatting** option for fields that contain files (image or other types of files).

It is also best to keep text and images separate—that you do not allow the Sites Desktop users to save both text and images to the same blob attribute or upload field. For example, if you have an article asset type that contains an image with a caption, use one attribute or field for the caption and another attribute or field for the image file.

By keeping text and images in separate fields and controlling whether the formatting is preserved or not, your template writers can know what kind of data will be extracted from a particular field and can code their templates to display that data appropriately. If you decide to allow users to save both text and image to the same blob attribute, both you and your template developers should note the following:

Table 7: Storage of Content Created in Sites Desktop

Format Preserved	Word Content Marked to Be Saved in the Blob Attribute	How Word Content Is Stored in the Database
Yes	text and image	The image is written to the directory identified by the `transformer.imgdir` property in `futuretense_xcel.ini`. The text is stored as HTML in the URL column; the HTML has an unmanaged link to the image in the `<transformer.imgdir>` directory. In such a case, you must implement a process that regularly mirrors the `transformer.imgdir` directory to the delivery system because the publishing system does not mirror this directory.
Yes	text only	The text is stored as HTML in the URL column.
Yes	image only	The image file is written to the URL column.
No	text and image	The text is stripped out and the image file is written to the URL column.
No	text only	The formatting of the text is stripped out and the text is written to the URL column.
No	image only	The image file is written to the URL column.

Configuring the Word Asset Types for Sites Desktop

The term "Word asset type" refers to any WebCenter Sites asset type that is configured to be created and edited in Microsoft Word. It is not a new asset type, but rather, an asset type already known to WebCenter Sites that you (or another administrator) have enabled for Word (Sites Desktop).

While there are no restrictions on which asset types might fall into this category, practical limits require that you restrict the qualifying asset types to those that have text content or are otherwise document-related.

To configure the asset types for Sites Desktop

- Enable the asset types that you want to make available through Sites Desktop.
- For each subtype for the enabled basic asset types and for each definition for the enabled flex asset types, specify which fields will appear on the WebCenter Sites toolbar within the Microsoft Word interface.
- For each field that you enable, determine whether WebCenter Sites should retain the formatting of the data that was entered in the Microsoft Word application. For fields that hold blobs, use the formatting option to determine how text and files are managed when they are written to that field.
- Create Sites Desktop start menu items for each of the Word asset types.

The following sections describe each step in detail.

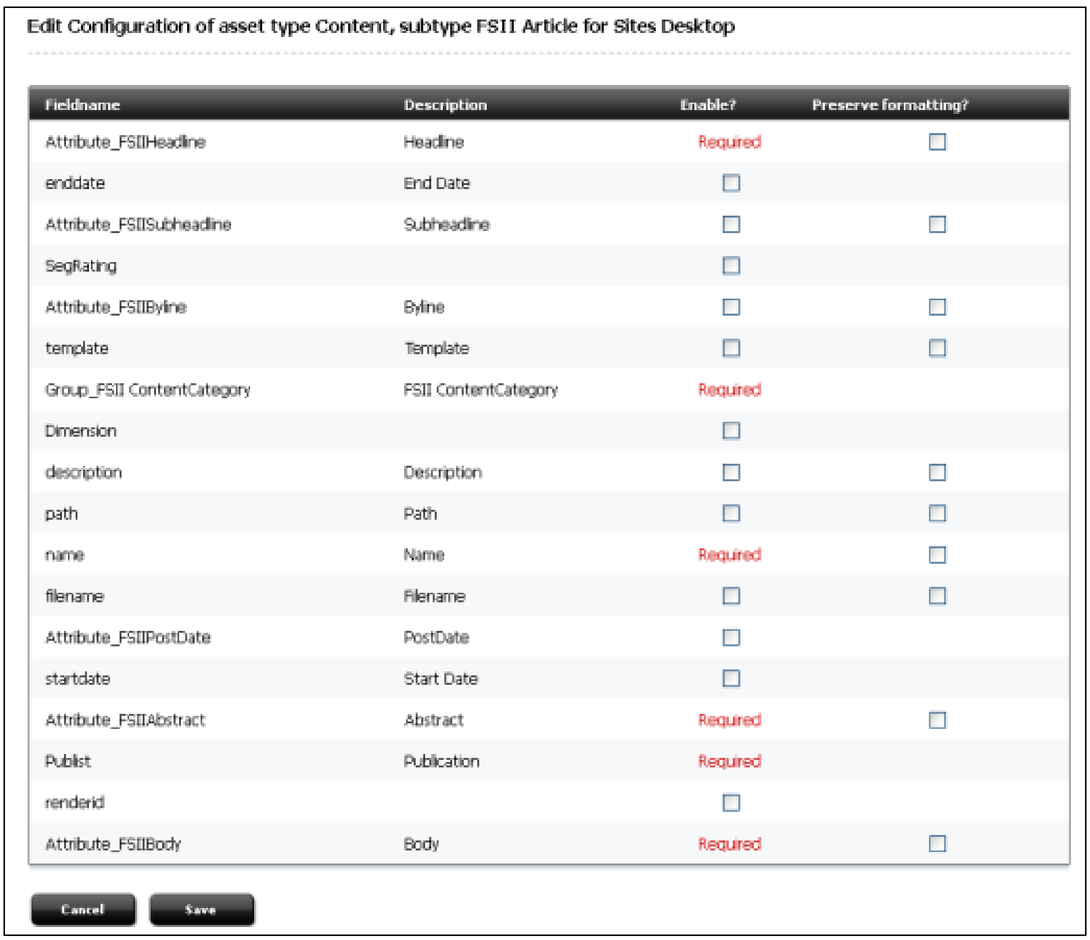
Enabling Asset Types for Sites Desktop

You enable an asset type for Sites Desktop by adding it to those that are selected for Sites Desktop.

To enable asset types for Sites Desktop

1. In the tree, do one of the following:
   - In the **Admin** tab, expand **Sites**, then the site for which you are configuring Sites Desktop.
   - If you have logged in to the site that you are configuring and you have access to the **SiteAdmin** tab, select the **SiteAdmin** tab.
2. Expand **Sites Desktop** and double-click **Enable Sites Desktop**.

   WebCenter Sites displays the "Enable Asset Types for Sites Desktop" form, showing asset types that have not yet been enabled for Sites Desktop in this site:

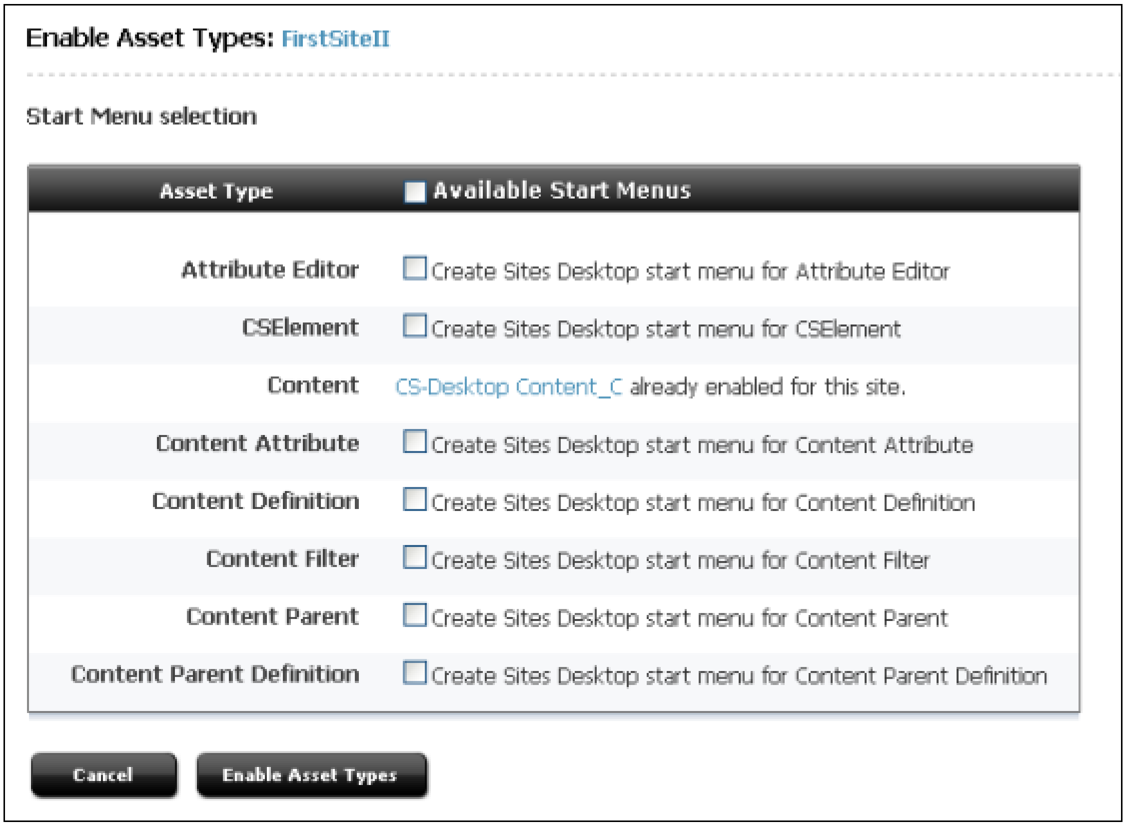


Edit Configuration of asset type Content, subtype FSII Article for Sites Desktop

Fieldname	Description	Enable?	Preserve formatting?
Attribute_FSIIHeadline	Headline	Required	☐
enddate	End Date	☐	
Attribute_FSIISubheadline	Subheadline	☐	☐
SegRating		☐	
Attribute_FSIIByline	Byline	☐	☐
template	Template	☐	☐
Group_FSII ContentCategory	FSII ContentCategory	Required	
Dimension		☐	
description	Description	☐	☐
path	Path	☐	☐
name	Name	Required	☐
filename	Filename	☐	☐
Attribute_FSIIPostDate	PostDate	☐	
startdate	Start Date	☐	
Attribute_FSIIAbstract	Abstract	Required	☐
Publist	Publication	Required	
renderid		☐	
Attribute_FSIIBody	Body	Required	☐

Cancel Save

3. Select the asset types you want to enable for Sites Desktop in this site.
4. Click **Enable Asset Types**.
5. The "Enable Asset Types Start Menu selection" screen displays.



6. Select the asset types to include in the Start Menu.
7. Click **Enable Asset Types**.
8. Continue to step 4 in "Specifying Which Fields Are on the WebCenter Sites Toolbar," on page 410.

Specifying Which Fields Are on the WebCenter Sites Toolbar

Before you begin this procedure, be sure that you have enabled the asset type(s) that you want to configure for Sites Desktop. If you have not yet enabled the asset type(s), see "Enabling Asset Types for Sites Desktop," on page 409.

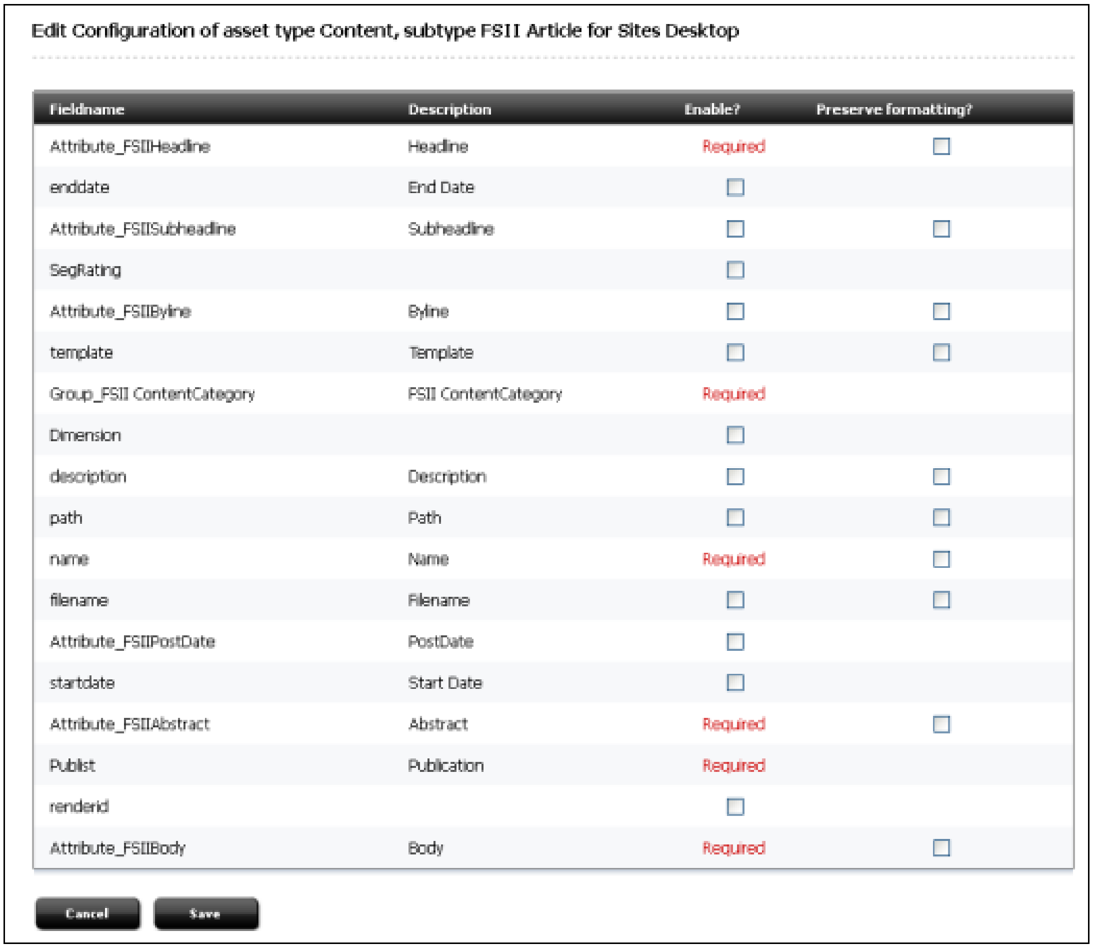
To specify which fields of an asset type will be displayed on the Sites toolbar in the MS Word interface

1. In the tree, do one of the following:
   - In the **Admin** tab, expand **Sites**, then the site for which you are configuring Sites Desktop.
   - If you have logged in to the site that you are configuring and you have access to the **SiteAdmin** tab, select the **SiteAdmin** tab.
2. Expand **Sites Desktop**, then expand asset type you want to configure.
3. Double-click the desired subtype (basic asset types) or definition (flex and flex parent asset types).

The Configuration page for the asset opens.

4. Click **Edit Configuration**.

   WebCenter Sites displays a configuration form similar to this one:



Edit Configuration of asset type Content, subtype FSII Article for Sites Desktop

Fieldname	Description	Enable?	Preserve formatting?
Attribute_FSIIHeadline	Headline	Required	☐
enddate	End Date	☐	
Attribute_FSIISubheadline	Subheadline	☐	☐
SegRating		☐	
Attribute_FSIIByline	Byline	☐	☐
template	Template	☐	☐
Group_FSII ContentCategory	FSII ContentCategory	Required	
Dimension		☐	
description	Description	☐	☐
path	Path	☐	☐
name	Name	Required	☐
filename	Filename	☐	☐
Attribute_FSIIPostDate	PostDate	☐	
startdate	Start Date	☐	
Attribute_FSIIAbstract	Abstract	Required	☐
Publist	Publication	Required	
renderid		☐	
Attribute_FSIIBody	Body	Required	☐

Cancel Save

5. In the “Edit Configuration” form, select the **Enable** check box for each field you want to appear in the Sites Desktop toolbar in Word. As a general rule, select only text-entry fields. Note that required fields are pre-elected and cannot be deselected.
6. (Optional) If you want the format of the text that is entered in a field through Word to be preserved when that data is stored in the WebCenter Sites database, select the **Preserve formatting** option for that field.

   As a general rule, enable the **Preserve formatting** option only for fields that store text. Do not enable the **Preserve formatting** option for fields that hold uploaded files.
7. Click **Save**.
8. (Optional) Repeat steps 3–7 for each additional subtype or definition you want to configure.
9. (Optional) Repeat steps 2–8 for each additional asset type you want to configure.

Creating Sites Desktop Start Menu Items

The subtype configuration determines which fields are available on the WebCenter Sites toolbar in the Microsoft Word interface, but your Word asset types do not become available in Word until you create start menu items for them.

To create Sites Desktop start menu items

1. In the **Admin** tab, expand **Asset Types**, then the desired asset type.
2. Under the selected asset type, expand **Start Menu** and double-click **Add New**.
3. In the **Name** field, enter a short, descriptive name for the item. This is the name that is displayed in the list of asset types that are available from the Sites Desktop toolbar in Word.
4. In the **Description** field, enter a short, informative description for the start menu item.
5. In the **Type** field, select **Sites Desktop**.

   > **Note**
   >
   > You cannot set default field values for assets whose source is Word. Ignore the **Default Values** section when creating start menu items for Word asset types.

6. In the **Sites** field, select which sites can use this start menu item. Use **Ctrl-click** to select more than one site.
7. In the **Roles** field, select all the roles that can have access to the start menu item. Use **Ctrl-click** to select more than one role.
8. (Optional) To configure workflow process details for the assets that are created with this start menu item, complete the following steps:
   a. Click **Select Workflow**.
   b. In the “Select Workflow” form, select the appropriate workflow process from the drop-down list and click **Select Workflow**.
   c. In the “Set Participants” form, select at least one user for each role that appears and then click **Set Participants**.
   d. Click **Continue**.

      WebCenter Sites saves your changes and redisplays the “Start Menu” form.
9. In the “Start Menu” form, click **Save**.
10. Repeat steps 1–9 for each additional asset type you want to configure.

User Accounts and Sites Desktop

Like the content providers who use the WebCenter Sites interfaces, the content providers who use the Sites Desktop application must have WebCenter Sites user accounts as well. User accounts for the individuals who use Sites Desktop must have the following ACLs:

- Browser
- ElementReader
- PageReader

- RemoteClient
- xceleditor
- UserReader
- Visitor

For information about creating user accounts, see "Editing a User," on page 92, and "Required ACLs for Custom Users," on page 528.

Give the Sites Desktop users the following information:

- Their user name/password combination
- The appropriate URL for logging in to your Sites management system

They need both pieces of information to log in to WebCenter Sites through Sites Desktop.

The WebCenter Sites URL uses the following convention:

```
http://<server>:<port>/<context>
```

where:

- `<server>` is the host name or IP address of the machine running WebCenter Sites.
- `<port>` is the port on which WebCenter Sites is listening for connections.
- `<context>` refers to the application context root (URI) assigned to the WebCenter Sites application when it was installed.

Installing the Sites Desktop Client Application

In order to use the Sites Desktop feature, the Sites Desktop client must be installed on the client machine.

> **Note**
>
> Before installing Sites Desktop, make sure that the client machine:
>
> - Is running supported versions of Microsoft Windows and Word, and
> - Supports the entry and display of text using the UTF-8 character set.

To install the Sites Desktop client

1. In the shipped zip file, the Sites Desktop installer is located on the following path:

   ```
   ContentServer/Xcelerate/cs_client
   ```

   where there are `/32bit` and `/64bit` folders to choose depending on your system.
2. Extract the downloaded archive to a temporary directory and open that directory.
3. Launch the Sites Desktop installer (`.msi`) and follow the instructions it displays.

   > **Note**
   >
   > If one or more language packs are installed on your Sites system, the page lists localized versions of the Sites Desktop client corresponding to the installed language packs. In such case, download the version appropriate to the user's language preference. (Language packs are available with patchset 1)

Specifying Locale for the Client Application

The Download page is generated dynamically based on the language packs that are installed on your Sites system. If your Sites system has a language pack installed, the Download page presents a Sites Desktop option in that language and the csdesktop.exe file installs a version of the help file in that language.

The first time a Sites Desktop user logs in to WebCenter Sites, the Log In dialog box is in English. The user clicks the Select Language link to select the language that matches the version of Sites Desktop help file that he or she installed.

After a user selects a language, Sites Desktop copies the appropriate XML language file from WebCenter Sites to that user's hard drive. Sites Desktop uses the information in that language file for menu names and dialog box text until the user selects a different language.

Additionally, whenever the language file on the WebCenter Sites server is updated, be sure to remind your Sites Desktop users to update their language file on their hard disks by using the Select Language link again the next time they log in to WebCenter Sites.

Testing the Sites Desktop Configuration

To determine whether your Sites Desktop application is configured correctly, you must create and save an asset from the Microsoft Word interface and then examine it through WebCenter Sites. Complete the following steps:

1. Open Microsoft Word.
2. On the button bar, select **WebCenter Sites**, then select **Login**.
3. In the **Login** dialog box, enter the following information:
   - The name and password of a user who has the RemoteClient ACL.
   - The server URL, as follows:

     `http://<server>:<port>/<context>`
     where:

     - `<server>` is the host name or IP address of the machine running WebCenter Sites.
     - `<port>` is the number of the port on which WebCenter Sites is listening for connections.
     - `<context>` refers to the application context root (URI) assigned to the WebCenter Sites application when it was installed.
4. Select the site, asset type, and subtype, as necessary, to set the field markers list.
5. Enter text in the Word document.
6. Select (highlight) the entire range of text.
7. Click the **Field Markers** button on the toolbar and select the marker for the field that you are entering text for.
8. Repeat steps 5–7 for all required asset fields that require you to enter text in the Word document.

9. Set values for the field markers with ellipses (…). These field markers represent hidden fields; that is, their values do not appear in the Word document, but are part of the asset definition, and are thus stored in the WebCenter Sites database. When you click a field marker with an ellipsis, a dialog box opens. Enter the appropriate data.
10. In the button bar, select **WebCenter Sites**, then **Save to WebCenter Sites**.
11. Log in to the administrator's interface as a user who has access to the type of asset that you just created.
12. Select the site that you created the asset for and then click **Search**.
13. Find the asset that you entered through Sites Desktop.
14. Click **Inspect**. (Do not select **Edit**.)
15. Examine the data listed for the fields and verify that the data that you marked in Sites Desktop matches the data displayed in the "Inspect" form.

Configuring Word Templates for the Word Assets

The field markers on the WebCenter Sites toolbar are Word bookmarks. Therefore, you can create `.dot` files for your Word assets, marking the appropriate areas of those templates with either the WebCenter Sites field markers or your own Word bookmarks.

Create a new Word document and mark placeholder text with the appropriate field markers, just as you would for an asset of that type (that is, be sure that you do not mark text for the metadata fields such as Name and Description and so on). Then save the document as a template (a .dot file) rather than to WebCenter Sites.

Sites DocLink

Oracle WebCenter Sites DocLink (Sites DocLink) provides a drag-and-drop interface for uploading and downloading documents, graphics, or other files that are managed as flex assets by WebCenter Sites. This application presents the hierarchical data structure of the flex parents and flex assets in your WebCenter Sites database as folders and files in the Windows Explorer application.

Overview

Sites DocLink enables your content providers to use their third-party tools to create content for your online sites, and then save it to the WebCenter Sites database without having to log in to the WebCenter Sites interfaces.

Sites DocLink enables content providers to drag and drop Word files, Excel spreadsheets, graphic files, and so on—that is, single binary files—into the WebCenter Sites database by dropping each file onto the appropriate Microsoft Windows Explorer folder that represents the asset type or parent asset type that the document should be saved as.

After a content provider drops the file onto a folder, Sites DocLink displays a dialog box that prompts the user to enter any metadata that is required, depending on how you, the administrator, have configured your system. Examples of such metadata include name, description, and so on.

Sites DocLink differs from Sites Desktop for Word as follows:

- With Sites Desktop, content providers produce the content and tag the data in that document that should be inserted into various fields from within the Word interface. The tagging process creates a structured asset when the document is stored in the database as an asset.
- With Sites DocLink, content providers produce their content in their third-party applications but they do not use those applications to provide structure to the data.

  They drop a binary file into the Windows Explorer interface, and then provide information about the file through the Sites DocLink dialog boxes that appear in response to the drop. The information required depends on the kind of asset type the file is going to be stored as, based on the place in the graphical representation of the database that the user dropped the file.

Configuring the Sites DocLink Asset Types

The term "Sites DocLink asset type" refers to a flex or flex parent asset type that is configured to be accepted by Sites DocLink. Sites DocLink asset types must be flex asset types that store an uploaded, binary file. That is, one (and **only** one) of the flex attributes that define the asset **must be** of type **blob**.

To configure the Sites DocLink asset types

- Verify that the flex or flex parent asset type is enabled for the site. Be sure that the asset type has **one** flex attribute of type **blob**.
- Enable the flex or flex parent asset type for Sites DocLink.
- For each definition for those enabled asset types, specify which fields will be available in Sites DocLink and specify which field is the upload field.
- Create Sites DocLink start menu items for each of the Sites DocLink asset types.

Enabling Asset Types for Sites DocLink

You enable an asset type for Sites DocLink by adding it to those that are selected for Sites DocLink.

To enable asset types for Sites DocLink

1. In the tree, do one of the following:
   - In the **Admin** tab, expand **Sites**, then the site for which you want to enable Sites DocLink.
   - If you have logged in to the site that you are configuring and you have access to the **SiteAdmin** tab, select the **SiteAdmin** tab.
2. Under the selected asset type, expand **Sites DocLink**, then double-click **Enable Sites DocLink**.

WebCenter Sites displays the “Enable Asset Types” form, listing asset types that have not yet been enabled for Sites DocLink in the selected site.

3. Select the check boxes next to the asset types that you want to enable for Sites DocLink in the selected site.
4. Click **Enable Asset Types**.
5. The “Start Menu Selection” screen appears. Select the items to include in the Start Menu.

   The enabled asset types (the items selected on the “Enable Asset Types” screen in step 2 appear in the **Admin** tab under the Sites DocLink node for the selected site.
6. Continue to step 3 in “Specifying Which Fields Are on the WebCenter Sites Toolbar,” on page 410.

Specifying Which Fields Are Available in Sites DocLink

Before you begin this procedure, be sure that the following conditions are true:

- You have enabled the asset type that you want to configure for Sites DocLink. If you have not yet enabled the asset type, see “Enabling Asset Types for Sites DocLink,” on page 416.
- The asset type that you want to configure has one attribute of type blob.

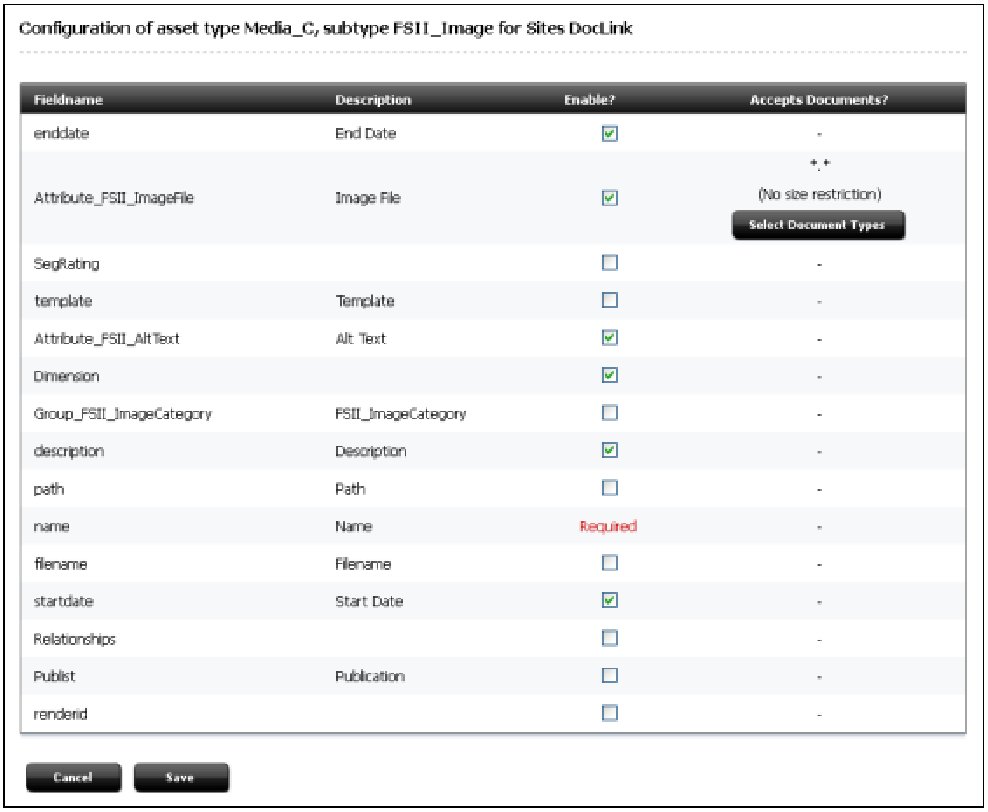
To specify which of an asset type’s fields will be available to content providers who are using Sites DocLink to create or edit assets of that type

1. In the tree, do one of the following:
   - In the **Admin** tab, expand **Sites**, then the site for which you are configuring Sites DocLink.
   - If you have logged in to the site that you are configuring and you have access to the **SiteAdmin** tab, select the **SiteAdmin** tab.
2. Under the selected site, expand **Sites DocLink**, then expand the desired asset type and double-click the subtype.

The configuration screen for the asset subtype is displayed.

3. Click **Edit Configuration**.

   WebCenter Sites displays a form similar to this one:



Configuration of asset type Media_C, subtype FSII_Image for Sites DocLink

Fieldname	Description	Enable?	Accepts Documents?
enddate	End Date	☑	-
Attribute_FSII_ImageFile	Image File	☑	*.* (No size restriction) Select Document Types
SegRating		☐	-
template	Template	☐	-
Attribute_FSII_AltText	Alt Text	☑	-
Dimension		☑	-
Group_FSII_ImageCategory	FSII_ImageCategory	☐	-
description	Description	☑	-
path	Path	☐	-
name	Name	Required	-
filename	Filename	☐	-
startdate	Start Date	☑	-
Relationships		☐	-
Publist	Publication	☐	-
renderid		☐	-

Cancel Save

4. In the “Configuration of Asset Type” form, select the **Enable** option for each field that you want to be displayed in the Sites DocLink interface. An enabled field means that users can view and edit the data in that field.

   As a general rule, enable the following kinds of fields:

   - The blob attribute that will hold the file (just one)
   - Text entry fields (**Description**, for example)

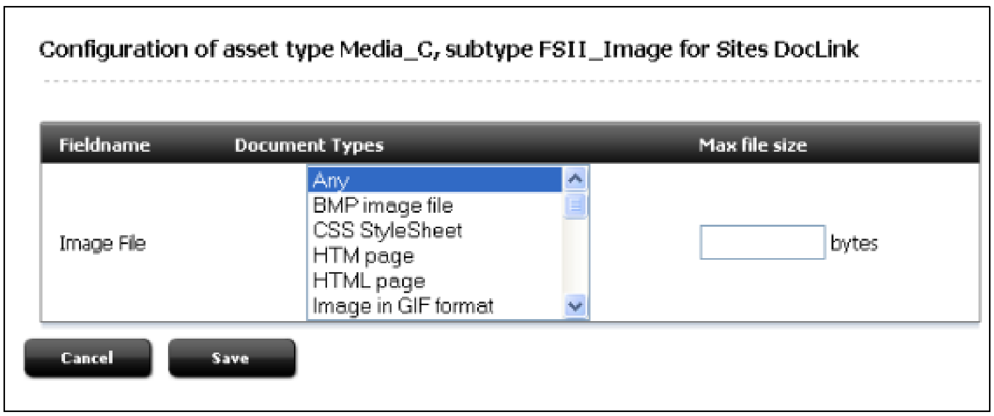
Note

- Do **not** enable the **Publist** field—assets cannot be shared with other sites through the Sites DocLink interface.
- If users are required to enter a value into a field, they will be prompted for the value ***when they create the asset***—whether or not you have selected the **Enable** option for that field. If you wish to have a required field write-protected once the asset is saved, do not enable the field. When Sites DocLink users right-click the asset and select **Properties**, the required field will not be displayed to them.

5. Click **Select Document Types** next to the upload field (flex attribute) you want to specify as the field in which documents are to be stored.

> **Note**
>
> If your developers have created a flex filter for assets of this type, the flex attribute that you specify as the upload field must match the field that is designated as the input attribute for the flex filter.
>
> For information about flex filters, see the *Oracle WebCenter Sites Developer's Guide* and talk to your developers.

WebCenter Sites displays a form similar to this one:



6. In the **Document Types** field, select the types of documents that this upload field can store. Use **Ctrl-click** to select more than one.

   The values displayed in the **Document Types** field come from the `description` column of the entries in the `MimeType` table. If the document type that you want to select does not appear in the list, use WebCenter Sites Explorer to add it to the `MimeType` table.

> **Note**
>
> When you select the **Any** option, it means that Sites DocLink accepts any type of file for assets of this type, whether or not there is an entry for it in the `MimeType` table.
>
> However, when there is no entry for a file type in the `MimeType` table, a file of that type cannot be displayed correctly in the WebCenter Sites interfaces.

7. In the **Max file size** field, specify a size limit (if there is one) for the documents that can be stored in this upload field.
8. Click **Save**.
9. In the "Configuration of Asset Type" form, click **Save** again.
10. Repeat steps 2–8 for each additional subtype or definition you want to configure.
11. Repeat steps 2–10 for each additional asset type you want to configure.

Creating Sites DocLink Start Menu Items

The Sites DocLink configuration determines which fields are available in the Sites DocLink interface, but your content providers cannot create assets of this type in Sites DocLink until you create start menu items for those asset types.

To create Sites DocLink Start Menu items

1. In the **Admin** tab, expand **Asset Types**, then the desired asset type.
2. Under the selected asset type, expand **Start Menu** and double-click **Add New**.
3. In the **Name** field, enter a name for the item.
4. In the **Description** field, enter a short, informative description for the start menu item.
5. In the **Type** field, select **Sites DocLink**.

> **Note**
>
> You cannot set default field values for assets whose source is Sites DocLink in this version. Ignore the **Default Values** section when creating Sites DocLink start menu items.

6. In the **Sites** field, select which sites can use this start menu item. Use **Ctrl+click** to select more than one site.
7. In the **Roles** field, select all the roles that can have access to the start menu item. Use **Ctrl+click** to select more than one role.
8. (Optional) To configure workflow process details for the assets that are created with this start menu item, complete the following steps:
   a. Click **Select Workflow**.
   b. In the “Select Workflow” form, select the appropriate workflow process from the drop-down list and then click **Select Workflow**.
   c. In the “Set Participants” form, select at least one user for each role that appears and then click **Set Participants**.
   d. Click **Continue**.

      WebCenter Sites saves the workflow information and redisplays the “Start Menu” form.
9. In the “Start Menu” form, click **Save**.
10. Repeat this procedure for each additional asset type you want to configure.

Users and Sites DocLink

Like the content providers who use the WebCenter Sites interfaces, the content providers who use the Sites DocLink application must have WebCenter Sites user accounts as well. Accounts for users who use Sites DocLink must have the following ACLs:

- Browser
- ElementReader
- PageReader
- RemoteClient

- UserReader
- xceleditor

For information about creating user accounts, see “Creating a New User,” on page 86.

Give the Sites DocLink users the following information:

- Their user name/password combination
- The appropriate URL for logging in to WebCenter Sites

They need both pieces of information to log in to WebCenter Sites through Sites DocLink.

The WebCenter Sites URL uses the following convention:

`http://<server>:<port>/<context>`

where:

- `<server>` is the host name or IP address of the machine running WebCenter Sites.
- `<port>` is the port on which WebCenter Sites is listening for connections.
- `<context>` refers to the application context root (URI) assigned to the WebCenter Sites application when it was installed.

Installing the Sites DocLink Client Application

In order to use the Sites DocLink feature, the Sites DocLink client must be installed on the client machine.

To install the Sites DocLink client

1. In the shipped zip file, the Sites DocLink installer is located on the following path:

   `ContentServer/Xcelerate/cs_client`

   where there are `/32bit` and `/64bit` folders to choose depending on your system.
2. Extract the downloaded archive to a temporary directory and open that directory.
3. Launch the Sites DocLink installer (`.msi`) and follow the instructions it displays.

> **Note**
>
> If one or more language packs are installed on your WebCenter Sites installation, the page lists localized versions of the Sites DocLink client corresponding to the installed language packs. In such case, download the version appropriate to the user’s language preference. (Language packs are available with patchset 1)

Testing the Sites DocLink Configuration

To determine whether your Sites DocLink application is configured correctly, you must create and save an asset from the Sites DocLink interface and then examine it through WebCenter Sites.

To test the Sites DocLink configuration

1. Open WebCenter Sites Explorer.
2. Double-click **Sites DocLink**.

3. In the **Login** dialog box, enter the following information:
   - The Sites DocLink URL. See "Users and Sites DocLink," on page 420 for the appropriate URL for your system.
   - The user name and password of a user who has the necessary ACLs (listed in "Users and Sites DocLink," on page 420).
4. Drag a document from the file system to the file structure represented under the Sites DocLink node and drop it in the appropriate place.
5. If there are required fields that must be filled in, the **Properties** dialog box appears. Enter the required information and save the asset.
6. Open your browser and log in to the administrator's interface as a user who has access to the type of asset that you just created.
7. If prompted, select the site in which you created the asset.
8. Find the asset that you created through Sites DocLink and open it in the "Inspect" form.
9. Examine the data listed for the fields and verify that the data that you specified in Sites DocLink matches the data displayed in the "Inspect" form.

Web Mode

Web Mode is a WebCenter Sites feature that enables infrequent users or users who perform a limited role to find, edit, and submit content directly from the rendered (Preview) version of an asset.

Enabling this feature requires three general steps:

- You, the administrator, set the `xcelerate.enableinsite` property in the `futuretense_xcel.ini` file to `true` on the Sites management system.
- Your developers code templates that invoke the Web Mode feature for the fields that you want content providers to be able to edit in this way. Note that these same templates do not display Web Mode on the Sites delivery system because the `xcelerate.enableinsite` is set to `false` on that system.
- You determine the appropriate URL and then make it available to the Sites users who need to use Web Mode.

Enabling Web Mode

To enable Web Mode

1. Start the Property Editor and open the `futuretense_xcel.ini` file.
2. On the **xcelerate** tab, select `xcelerate.enableinsite` and set its value to `true`.
3. Save the property file and close the Property Editor.

To configure your servlet and host name used in Web Mode and Preview

1. Start the Property Editor and open the `futuretense_xcel.ini` file.
2. On the **xcelerate** tab, do one or both of the following:
   a. Set the `xcelerate.previewhost` property as

`http://<server>:<port>/servlet/`

b. Change the value of the `xcelerate.previewservlet` property from `ContentServer` to `Satellite`

For Example:

- If content providers are not required to include the port number in the URL that they use to log in to Sites, include it in the value for the `xcelerate.previewhost` property.
- If content providers are required to include the port number, specify a different port number for the property. Use either the web server port number or the port number of a redirect request to the web server.
- Create an alias and use it only for the `xcelerate.previewhost` property.
- Use the `xcelerate.previewservlet` property to specify the Satellite Server servlet rather than the WebCenter Sites servlet.

Maintaining Separate Browser Sessions for Preview

If the online site that you are serving from your Sites delivery system requires visitors to log in, or it logs them in by default, your content providers need a separate browser session for preview if they must log in to the previewed site on the Sites management system with a user name that is **different** from the one they are using in WebCenter Sites.

Without a new, separate browser session for preview, the WebCenter Sites session ends when the preview session begins, which means that content providers can lose any unsaved work.

To ensure that a preview session does not end the WebCenter Sites session, you can do the following:

- Have your site designers configure the online site so that your content providers are logged in to the site that they are previewing with their WebCenter Sites user accounts.
- Configure your Sites management system so that the web server opens a new browser session for the preview window without closing the current session in the WebCenter Sites window.

To configure your Sites management system so that the web server opens a new browser session for the previewed asset, you configure properties in the `futuretense_xcel.ini` file: `xcelerate.previewhost` and `xcelerate.previewservlet`.

You use these properties to point to the URL for the Sites management system, but the URL must be different in some way from the URL that your content providers use to log in to the management system. For example:

- If content providers are not required to include the port number in the URL that they use to log in to Sites, include it in the value for the `xcelerate.previewhost` property.
- If content providers are required to include the port number, specify a different port number for the property—the web server port number or the port number of a redirect request to the web server.

- Create an alias and use it only for the `xcelerate.previewhost` property.
- Use the `xcelerate.previewservlet` property to specify the Satellite Server servlet rather than the WebCenter Sites servlet.

Start the Property Editor, open the `futuretense_xcel.ini` file, select the **xcelerate** tab, and do one or both of the following:

- Set the `xcelerate.previewhost` property as follows:

  For most application servers, including Sun JES Application Server:

  `http://<server>:<port>/servlet/`
- Change the value of the `xcelerate.previewservlet` property from `ContentServer` to `Satellite`

Chapter 23

Configuring the Lucene Search Engine

Lucene is a third-party search engine that is integrated with WebCenter Sites. Lucene powers the search feature in WebCenter Sites' Contributor interface and enables the Public Site Search API to support search capabilities on Web sites. Before you can search for assets in the Contributor interface, you must set up and configure the Lucene search engine. This chapter shows you how to set up and maintain Lucene on your system.

This chapter contains the following sections:

Overview

When you install WebCenter Sites, the search feature in the WebCenter Sites database and Lucene are available. Lucene is enabled and configured "out of the box." The Lucene engine is set up as Sites is installed, allowing content contributors, website visitors, and third party applications will be able to search for assets.

This chapter includes information detailing how to use the Lucene search engine, how to make additional assets searchable and how to pause or disable the search engine.

Indexing for Search Functions

Contributor searches are run against a search index that is powered by Lucene, not WebCenter Sites' database. A search index is built by an automated process called "indexing," which collects, parses, and stores asset data in a form that can be quickly retrieved during a search query.

Search results are returned based solely on the data that is available in the index at the time the search is performed. The more assets you include in the index, the longer it takes to build and to search.

You select the types of assets to index on the search configuration screens. Selected assets will be indexed and therefore, searchable. Asset types you omitted from indexing will not be indexed and, therefore, will not be searchable.

Once the index is built, the Lucene search engine runs an event every thirty seconds that checks for changes made to assets of the types selected for indexing. If changes were made (such as creating a new index item, editing an existing entry, or deleting an entry) Lucene updates the index automatically. By default, index data is stored in the <cs_shared_dir>/lucene directory (where <cs_shared_dir> is the Sites shared file system directory).

WebCenter Sites Search Functions

WebCenter Sites includes the following search functions: **Global** search, **Asset Type** search, and the most specific of the searches, **Configure Attributes for Asset Type Index**. This search option is a subset of the **Asset Type Index** option, as it allows you to specify attributes that will be searchable for the indexing-enabled asset types.

You can enable all of the searches on your system. The searches differ in how they store the indexed user-defined attributes.

Global Search indexes system-defined attributes individually, allowing users to search by specific attributes. All of an asset's *user-defined* attribute values are stored together in one table cell; attribute names are omitted. This means, when Global search is configured, users are restricted to searching across *all user-defined* attributes per asset type.

For example, suppose you have an article asset. You could search for the string "Jane Doe." However, you could not limit your search to just one specific *user-defined* attribute, because all of the user-defined attribute data are stored together in a single cell that does not differentiate one attribute from another. See table "A. Global Search: Indexed "Article" Data," on page 428.

Asset Type Search indexes each attribute value, for both system-defined and user-defined attributes, in its own individual cell, by attribute name. Asset type searches are used for the Public Site Search API, which enables search capabilities on the website. For more information on public site search, see the *Oracle WebCenter Sites Developers Guide*.

Using the example above of an article asset, with Asset Type search enabled you could, as with Global search, look for the string "Jane Doe" in the system-defined attribute "Name." You could also search for the string in the user-defined attribute "Byline," or in any other attribute for that asset type. Since user-defined attributes are now stored under their attribute names, you can search by those specific attributes. See table "B. Asset Type Search: Indexed "Article" Data," on page 428.

Asset type search, therefore, has a level of specificity to its search results that Global search does not. In addition to the obvious benefits of this targeted search capability, an asset type search can return search results more quickly than a Global search. By limiting searches to only the relevant attributes, Asset Type search can eliminate the need for a search to run across unnecessary index data.

Configure Attributes for Asset Type Search enables you to limit a search to only the attributes that you specify for the asset types that are enabled by **Asset Type Search**. Using the example above of an article asset, if you have configured indexing for specific attributes, such as "Headline" and "Byline," users can search for the string "Jane Doe" in only those attributes. See table "C. Configured Attributes for Asset Type Search: Indexed "Article" Data," on page 428.

The tables on page 428 illustrate the differences in Lucene-based searches and their levels of granularity. Each table represents an index for the same article type asset, but for a different type of search function. Across the tables, only the *system-defined* attribute data is stored in the same way.

Note

The tables on page 428 illustrate the granularity of global searches, asset type searches, and attribute-specific searches. They are not meant to indicate how the search engine actually stores indexed data.

A. Global Search: Indexed "Article" Data

System-defined Attributes | User-defined Attributes

Global Index creates one index for all asset types.

ID#	Name	Modified Date	Start Date	End Date	User-defined Attribute Values
1234	Article 1	01/01/08 00:00:00	02/02/08 00:00:00	03/02/08 00:00:00	Four Tips Beautiful Hair Jane Doe 1. Condition 2. Rinse with beer 3. Use shampoo sparingly 4. Wash daily
1235	Article 2	01/02/08 00:00:00	02/03/08 00:00:00	03/03/08 00:00:00	Four Tips Fuel Efficiency Jane Doe 1. Start slowly 2. Drive the speed limit 3. Get regular oil changes 4. Drive a small car
1236	Article 3	01/03/08 00:00:00	02/04/08 00:00:00	03/04/08 00:00:00	Four Tips Saving Money on Groceries Jane Doe 1. Clip coupons 2. Use a list 3. Shop weekly specials 4. Buy in bulk

Values of user-defined attributes are stored together in a single cell (per asset), without attribute names.

B. Asset Type Search: Indexed "Article" Data

System-defined Attributes | User-defined Attributes

Asset Type Index creates one index per asset type.

ID#	Name	Modified Date	Start Date	End Date	Headline	Subhead	Byline	Article text
1234	Article 1	01/01/08 00:00:00	02/02/08 00:00:00	03/02/08 00:00:00	Four Tips	Beautiful Hair	Jane Doe	1. Condition 2. Rinse with beer 3. Use shampoo sparingly 4. Wash daily
1235	Article 2	01/02/08 00:00:00	02/03/08 00:00:00	03/03/08 00:00:00	Four Tips	Fuel Efficiency	Jane Doe	1. Start slowly 2. Drive the speed limit 3. Get regular oil changes 4. Drive a small car
1236	Article 3	01/03/08 00:00:00	02/04/08 00:00:00	03/04/08 00:00:00	Four Tips	Saving Money on Groceries	Jane Doe	1. Clip coupons 2. Use a list 3. Shop weekly specials 4. Buy in bulk

Values of user-defined attributes are stored in separate cells, under their attribute names.

C. Configured Attributes for Asset Type Search: Indexed "Article" Data

System-defined Attributes | User-defined Attributes

Attributes for Asset Type Index creates an index of searchable attributes per asset type.

ID#	Name	Modified Date	Start Date	End Date	Headline	Subhead	Byline	Article text
1234	Article 1	01/01/08 00:00:00	02/02/08 00:00:00	03/02/08 00:00:00	Four Tips	Beautiful Hair	Jane Doe	1. Condition 2. Rinse with beer 3. Use shampoo sparingly 4. Wash daily
1235	Article 2	01/02/08 00:00:00	02/03/08 00:00:00	03/03/08 00:00:00	Four Tips	Fuel Efficiency	Jane Doe	1. Start slowly 2. Drive the speed limit 3. Get regular oil changes 4. Drive a small car
1236	Article 3	01/03/08 00:00:00	02/04/08 00:00:00	03/04/08 00:00:00	Four Tips	Saving Money on Groceries	Jane Doe	1. Clip coupons 2. Use a list 3. Shop weekly specials 4. Buy in bulk

Only specified attributes are searchable (Headline and Byline, in this example).

Besides the way they index data, the searches enable different functions, as summarized below:

	Global Index	Asset Type Index	Attributes for Asset Type Index
Enables	Searches in the Contributor interface		
	Public searches on the live website		

For more information about the Public Site Search API, see the *Oracle WebCenter Sites Developer's Guide*.

Setting Up Search Indices

The steps for setting up Lucene are as follows:

1. Enabling the Lucene Search Engine. Before you can configure Lucene, you must enable it on your system.
2. Adding Asset Types to the Search Index. This indicates to Lucene which assets it should index. You can add asset types to the global index and the asset type index.

   Once you have selected which assets to index and enabled binary file indexing (if desired), you can start the indexing process.

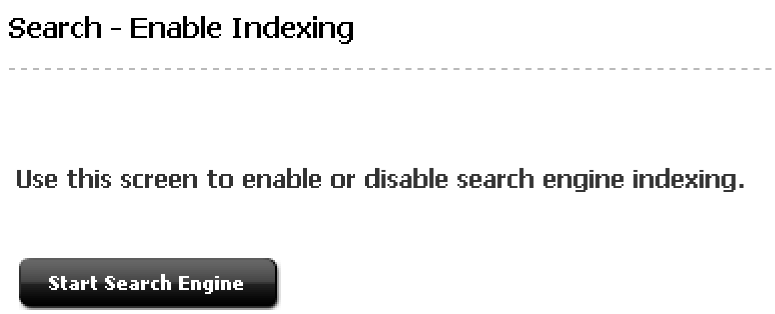
   During indexing, Lucene examines the contents of assets of the selected asset types (and the binary files the assets reference, if applicable) and creates entries for those assets in the index. A global index creates one index for all assets while an asset type index creates one index per asset type. Once the global index has been created and users conduct a search, assets will be returned by the search feature in WebCenter Sites' Contributor interface or on the live site. Once the asset type index is created, and users conduct a search, items will be returned by the search feature on the live site.
3. Configuring Attributes for Asset Type Index. Once you have created asset type indexes, you can select specific user-defined attributes to search on. Once an index for these specified attributes is created and populated, and you conduct a search, the specified attributes will be returned by the search feature on the live site.
4. Enabling Indexing of Binary Files. If one or more asset types which you added to the index are set up to reference binary files, you can configure Lucene to convert the contents of those files to text when indexing assets that reference them.

Enabling the Lucene Search Engine

This section shows you how to enable the Lucene engine.

To start the Lucene Engine

1. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
2. Click **Start Search Engine**.



Lucene is now enabled to start indexing selected data. The time it takes to index data varies with the number of assets being indexed and the speed of your system.

Once the Lucene Search Engine is started, it will continue to run until it is disabled. While indexing is running, changes to selected asset types are detected and the index is updated. The status of the asset type is listed as "Enabled," while the index is running. If you wish to remove search capability, in addition to stopping indexing, you will also need to delete index data. See "Deleting Index Data," on page 441.

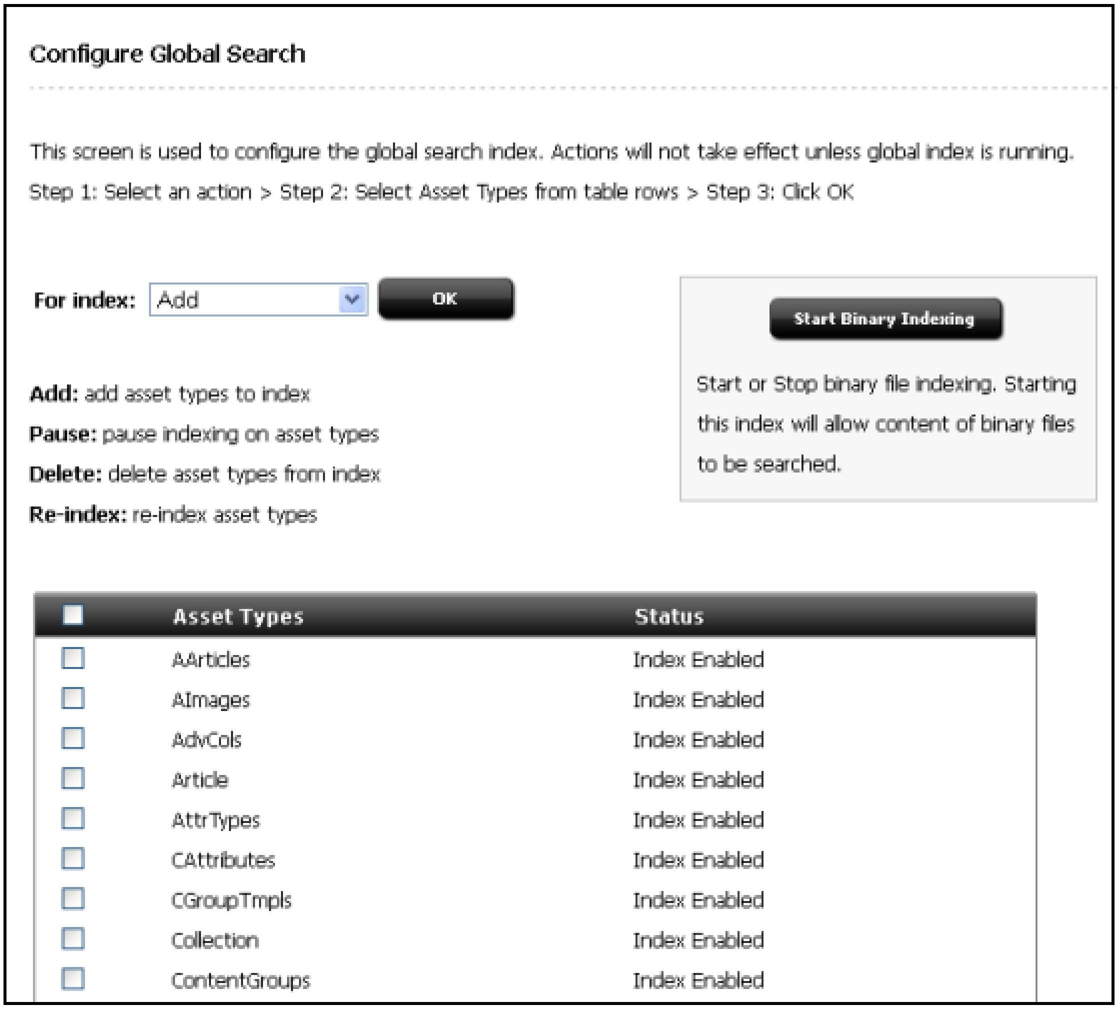
Adding Asset Types to the Search Index

This section shows you how to add asset types to the Global Search Index and the Asset Type index. Once each initial index has been created, Lucene checks for changes every 30 seconds. By default, index data is stored in the `<cs_shared_dir>/lucene` directory (where `<cs_shared_dir>` is the Sites shared file system directory). Once the data is added, it will be maintained until indexing is stopped entirely or paused for a selected asset type Assets of the selected types will not be returned by the search feature in WebCenter Sites' Contributor interface or on the live site until Lucene has indexed them.

To add new asset types to the search index

1. Enable the Lucene engine:
   a. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
   b. Click **Start Search Engine**.
2. Add asset types to the global search index:
   a. Double-click **Configure Global Search**.

b. In the "For index:" drop-down list, select **Add**. WebCenter Sites displays a list of asset types that are not currently being indexed.



c. In the "Asset Types" list, select the asset types you want to index.

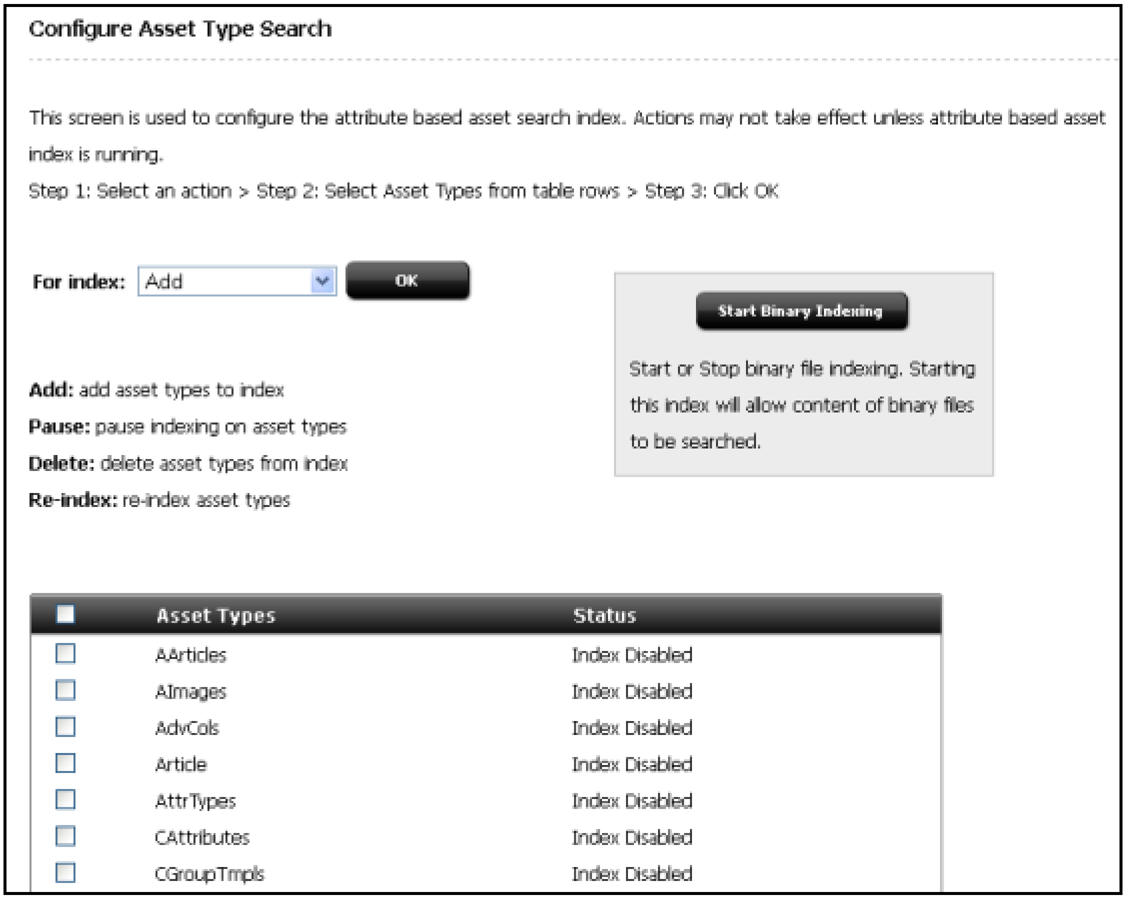
d. Click **OK**.

e. In the confirmation pop-up dialog, click **OK.** The asset type status changes to "Enabled" and indexing is enabled for the selected asset type. The index is created for that asset type as soon as the first asset of that type is created.

3. Add asset types to the asset type search index.

a. Double-click **Configure Asset Type Search**.

b. In the “For index:” drop-down list, select **Add**. WebCenter Sites displays a list of asset types that are not currently being indexed.



c. In the list, select the asset types you want to index.

d. Click **OK**. In the confirmation pop-up dialog that appears, click **OK.**

The asset type status changes to “Enabled” and indexing is enabled for the selected asset type. The index is created for that asset type as soon as the first asset of that type is created.

4. Enable binary file indexing, if desired (click **Start Binary Indexing**). For more information on binary file indexing, see “Binary File Indexing.”

Note

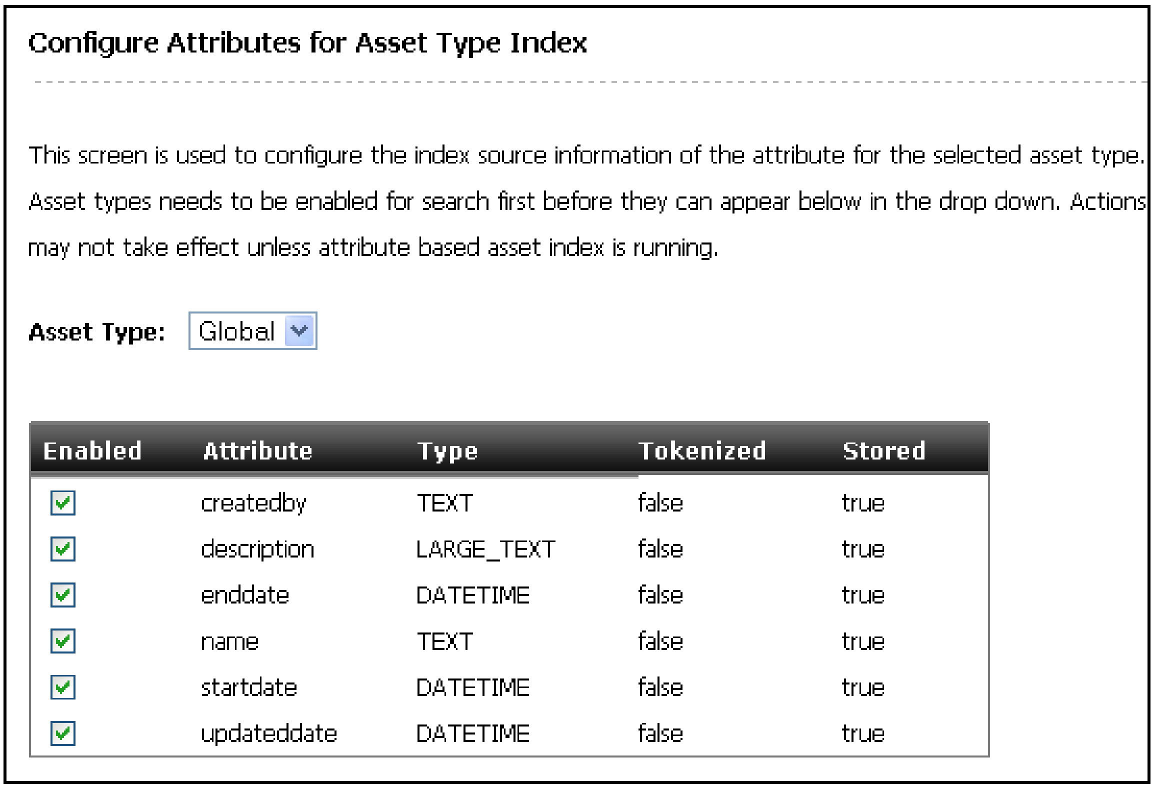
Global search indexing creates one index for all asset types. Asset type index creates one index for each individual asset type. For more information, see the tables on page 428.

Configuring Attributes for Asset Type Index

You can configure indexing on specific attributes for specific asset types. The selected asset type needs to be enabled for indexing first before you can select any specific attributes for the asset type. Once you have enabled Lucene and conducted a search on the live site, assets with indexed attribute data matching the search terms will be returned.

To configure attributes for a selected asset type

1. If you have not already done so, add the asset type you want to configure to the asset type index. For instructions, see "To add new asset types to the search index," on page 430.
2. In the **Admin** tab, expand **Search** and double-click **Configure Attributes for Asset Type Index**.
3. In the "Asset Type:" drop-down list, select the asset type you want to configure. WebCenter Sites displays a list of attributes for the selected asset type.



Enabled	Attribute	Type	Tokenized	Stored
	createdby	TEXT	false	true
	description	LARGE_TEXT	false	true
	enddate	DATETIME	false	true
	name	TEXT	false	true
	startdate	DATETIME	false	true
	updateddate	DATETIME	false	true

Note

To enable all attributes for a selected asset type, click **Select All.**

4. The following information is also displayed:
   - **Enabled**: Indicates if the specified attribute is enabled for indexing for that specific asset type. To disable an attribute, deselect its checkbox.
   - **Type**: Indicates the type of data the attribute stores. For example, "Numeric" can indicate that price information is stored. You can also have "Text" and "DateTime" data.

- **Tokenized**: Select **True** if you want this data to be converted to text before it is indexed. Data that is not tokenized is indexed as a single word and may not be interpreted by the reader.
- **Stored**: Indicates if the original entire text of the specified attribute has been added to the index. Select **True** to store the entire text to the index.

> **Note**
>
> At least one attribute must be enabled for each listed asset type.

5. Once you have made your changes to the selected asset types, click **Save.**

Binary File Indexing

Binary files are files of type other than text, such as Word and PDF documents. You may choose to not enable this option if your assets do not reference binary files, or if the files they reference contain content that is not indexable, such as images and videos.

Enabling Indexing of Binary Files

If one or more asset types which you added to the indexing queue are set up to reference binary files stored in the WebCenter Sites file system, you can configure Lucene to convert the contents of those files to text when indexing the assets that reference them. (By default, Lucene is set up to ignore all binary files referenced by assets being indexed.)

To enable binary file indexing

1. If you have not already done so, enable the Lucene engine:
   a. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
   b. Click **Start Search Engine**.

   The button name changes to **Stop Search Engine**.
2. In the **Admin** tab, expand **Search**.
3. Do one of the following:
   - To enable binary file indexing for Global search, double-click **Configure Global Search**.
   - To enable binary file indexing for Asset Type search, double-click **Configure Asset Type Search**.
4. Click **Start Binary Indexing**.

   Lucene will now convert to text all binary files that are referenced by the assets it indexes.

Disabling Indexing of Binary Files

If you decide that you no longer want Lucene to convert the contents of binary files referenced by assets it indexes, you can disable this feature to improve performance.

To disable binary file indexing

1. If you have not already done so, enable the Lucene engine:
   a. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
   b. Click **Start Search Engine**.

   The button name changes to **Stop Search Engine**.
2. In the **Admin** tab, expand **Search.**
3. Do one of the following:
   - To disable binary file indexing for Global search, double-click **Configure Global Search**.
   - To disable binary file indexing for Asset Type search, double-click **Configure Asset Type Search**.
4. Click **End Binary Indexing**.

   Lucene will now ignore all binary files referenced by the assets it indexes.

Disabling the Lucene Search Engine

You can stop the Lucene engine if you want to improve performance. Once the engine is stopped, you will no longer be able to add or delete assets, or pause indexing. You will also no longer be able to re-index assets.

To stop indexing

1. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
2. Click **Stop Search Engine**.

   The button name changes to **Start Search Engine**.

   Indexing with the Lucene engine is now disabled. The index data is preserved; a search on the Contributor interface or public site will return assets included during the last time the index was built. If you wish to delete the index data (and therefore remove search capability), see "Deleting Index Data," on page 441.

Note

If you stop Global search indexing, the search index used for the Contributor interface search function and the public site will not be updated. Therefore, search results will not be accurate.

- To remove search capability from Contributor, you will also need to delete all assets from the Global index. See "Deleting Index Data," on page 441 for instructions.
- If you add or remove assets while indexing is stopped, you need to rebuild the index to create an accurate search index when you restart indexing. See "Re-indexing," on page 439 for information on rebuilding indexes.

Maintaining Search Indexes

Once you have set up Lucene, you may need to perform tasks such as temporarily suspending indexing in order to perform bulk operations on assets, re-indexing, deleting index data, or writing code to specifically query the search engine. These maintenance tasks are described in the following sections:

- Pausing and Resuming Indexing
- Re-indexing
- Deleting Index Data
- Writing Code that Queries the Search Index

Pausing and Resuming Indexing

Pausing and stopping indexing are similar functions. When you pause indexing, you have the ability to pause indexing for selected asset types, whereas when you stop the search index you stop indexing on *all* assets.

When you add and delete large numbers of assets, you can speed up the process by temporarily pausing indexing on the assets of the type you are adding or deleting. To reflect these changes in your search index, you will then need to index all assets of the type that you added or deleted, using the re-indexing function.

- Pausing Global and Asset Type Indexing
- Resuming Global and Asset Type Indexing

Pausing Global and Asset Type Indexing

When indexing is enabled, every asset that is added or updated to the WebCenter Sites database is indexed after it is saved. Saving a large number of assets will proceed faster if you pause the indexing of assets of that type. For example, you can pause indexing when performing a bulk import of assets into the WebCenter Sites database. You can then resume indexing and re-index all assets of that type after the assets are added to the database, indexing all the new (and existing) assets at one time.

When you pause indexing for an asset type, Lucene does the following:

- Stops indexing assets of the selected type.
- Preserves the index data for assets of the selected type.

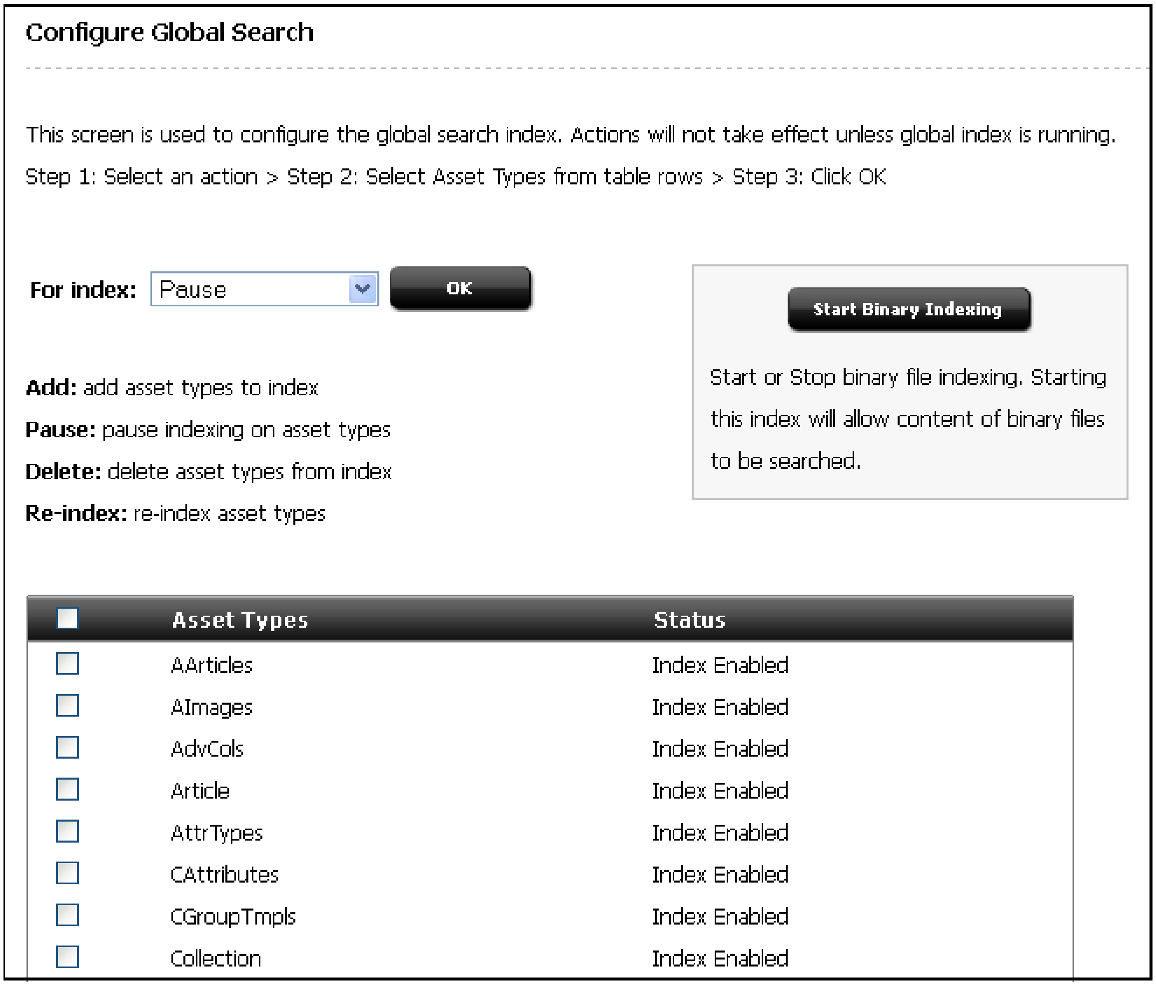
When indexing is paused, searches continue to return results against the existing index. However, changes to the database made after indexing is paused are not indexed. Therefore, search results will not reflect changes made to the database after indexing was paused.

To pause indexing

1. If you have not already done so, enable the Lucene engine:
    a. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
    b. Click **Start Search Engine**.

    The button name changes to **Stop Search Engine**.
2. In the **Admin** tab, expand **Search.**
3. Do one of the following:
    - To pause indexing for Global search, double-click **Configure Global Search**.
    - To pause indexing for Asset Type search, double-click **Configure Asset Type Search**.

4. In the “For index:” drop-down list, select **Pause**. WebCenter Sites displays the list of asset types for which you can pause indexing.



5. In the “Asset Types” list, select the asset types for which you want to pause indexing.

> **Note**
>
> If no asset types are displayed when you select **Pause** from the drop-down list, stop here. Either indexing is already paused for all asset types, or no asset types have yet been selected for indexing.

6. Click the **OK** button next to the drop-down list of operation selections.
7. In the confirmation pop-up dialog that appears, click **OK.**

   The Lucene Search Engine pauses indexing on assets of the selected types and preserves their index data. The status of the asset type changes to “Paused.”

   Changes to the database after indexing was paused are not indexed; therefore, search results will not reflect changes made to the database after indexing pauses.

Resuming Global and Asset Type Indexing

After pausing or disabling indexing, you will need to re-index to ensure that all your asset data is in the index. Follow the steps below to restart indexing.

To resume indexing

1. Restart indexing on the paused asset types. For instructions, see "Adding Asset Types to the Search Index," on page 430.
2. If you added assets to the database while indexing was paused, you must re-index to ensure that new data is included in the index. Proceed to "Re-indexing."
3. If you deleted assets while indexing was paused, the regular indexing process will detect which assets were deleted and remove that data from the index. However, if a large number of assets were deleted, it may be faster to delete the entire index for assets of the type you deleted and then re-index.
    - For more information on deleting index data, see "Deleting Index Data," on page 441.
    - Information on re-indexing can be found in the section "Re-indexing."

Re-indexing

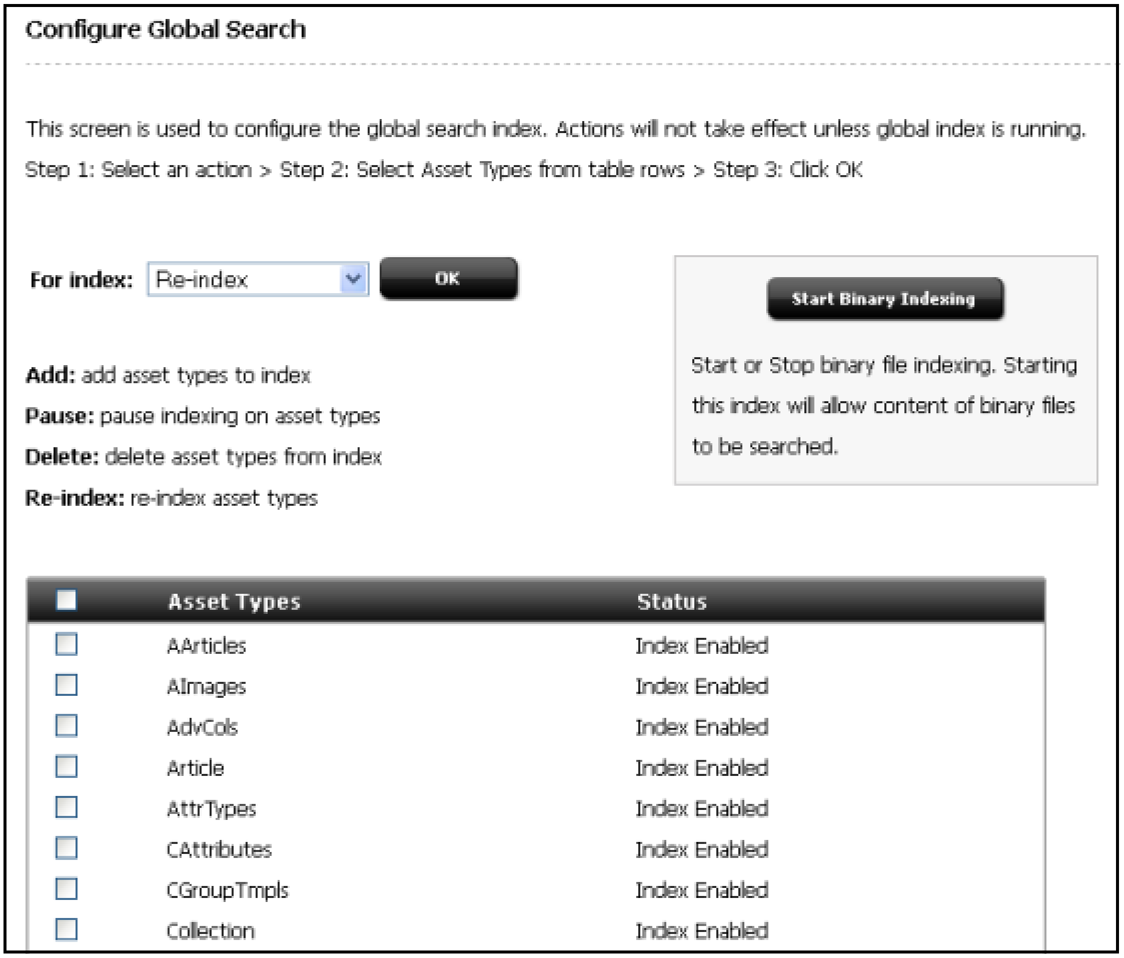
While indexing is paused or stopped, WebCenter Sites does not track the additional assets added to the database. Therefore, to search those assets, all assets of the type for which indexing was paused must be re-indexed.

The time it takes to re-index assets varies with the number of assets being indexed and your system configuration. Updated search results for assets of the selected types will be returned only after the Lucene search engine has indexed them.

To re-index assets

1. If you have not already done so, enable the Lucene engine:
    a. In the **Admin** tab, expand **Search** and double-click **Start/Stop Search Engine Indices**.
    b. Click **Start Search Engine**.
2. In the **Admin** tab, expand **Search.**
3. Do one of the following:
    - To re-index assets for Global search, double-click **Configure Global Search**.
    - To re-index assets for Asset Type search, double-click **Configure Asset Type Search**.
4. In the "For index:" drop-down list, select **Re-index**.

WebCenter Sites displays the asset types currently selected for indexing.



Note

If no asset types are displayed when you select **Re-index** from the drop-down list, stop here. No asset types are in the indexing queue or indexing has been paused for all asset types in the queue.

5. In the list, select the asset types whose index data you want to build (or rebuild).
6. Click **OK**.
7. In the confirmation pop-up dialog that appears, click **OK**.

   Indexing begins.

   The status of the selected asset types changes to “Enabled.”

   Updated search results for assets of the selected types will be returned only after the Lucene search engine has indexed them.

Deleting Index Data

If you no longer need to perform searches on assets of a particular type, search results will be returned more quickly if the unnecessary data is removed from the index.

You may also wish to delete indexes if you stopped indexing and then deleted a large number of assets. In this case, it may be faster to delete the relevant index data and then re-index the remaining assets than to allow the regular indexing process to run through its normal process to detect which assets have been deleted. More information on pausing and restarting indexing can be found in "Pausing and Resuming Indexing," on page 436.

When you delete index data, WebCenter Sites does the following:

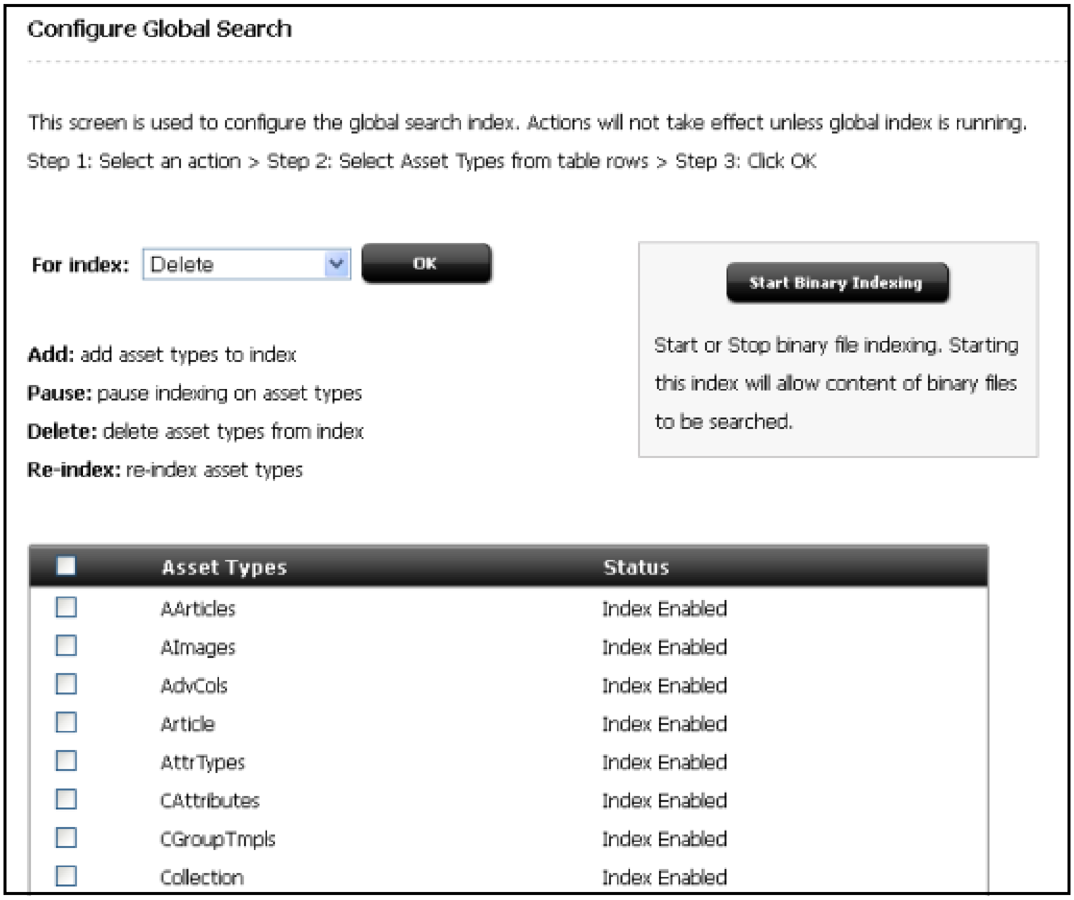
- Pauses indexing on assets of the selected asset types.
- Deletes the index data for assets of the selected asset types.

After you perform the steps below, index data is no longer available for assets of the selected types. Search results will no longer return data from assets of the selected types.

To delete data from the index

1. In the **Admin** tab, expand **Search.**
2. Do one of the following:
   - To delete assets from Global search, double-click **Configure Global Search**.
   - To delete assets from Asset Type search, double-click **Configure Asset Type Search**.
3. In the "For index:" drop-down list, select **Delete**.

4. WebCenter Sites displays the asset types currently being indexed.



> **Note**
>
> If no asset types are displayed when you select **Delete** from the drop-down list, stop here. No assets are being indexed.

5. In the list, select the asset types whose index data you want to delete.
6. Click **OK**.
7. In the confirmation dialog box that appears, click **OK**.

   WebCenter Sites pauses indexing on assets of the selected types and deletes their index data.

   In "Configure Global Search", the status of the asset types changes to "Paused." This status indicates that no new assets will be added to the existing index.

   In "Configure Asset Type Search", the status of the asset types changes to "Disabled". This status indicates that this asset is no longer eligible for indexing.

   The assets will no longer be returned by the search feature in WebCenter Sites' Contributor interface or the public site.

To make the assets searchable again, you must add the asset types back to indexing. For instructions, see "Adding Asset Types to the Search Index," on page 430.

Writing Code that Queries the Search Index

The following sample code illustrates how to query the Lucene search engine index. This code is based on the assumption that the user wants to search against a particular site and a particular asset type, where the site is passed in as variable `currentSite` and type is passed in as `assetType`. The user would use the following code to write a query against the Global index. The Lucene search engine would return all the assets or the maxResults (if total is greater than maxResults) specified of the type that belongs to the specified site.

```
ICS ics = Factory.newCS();
IndexSourceConfig srcConfig = new IndexSourceConfigImpl(ics);
SearchEngineConfig engConfig = new SearchEngineConfigImpl(ics);
IndexSourceMetadata sourceMd =
   srcConfig.getConfiguration("Global");
String engineName = sourceMd.getSearchEngineName();
SearchEngine eng = engConfig.getEngine(engineName);
String currentSite = (String)
   props.get(SearchIndexFields.Global.SITEID);
   QueryExpression siteExpr = new
   QueryExpressionImpl(SearchIndexFields.Global.SITEID,
   Operation.CONTAINS, currentSite);
siteExpr = siteExpr.or(SearchIndexFields.Global.SITEID,
   Operation.EQUALS, "0");
QueryExpression typeQ = new
   QueryExpressionImpl(SearchIndexFields.Global.ASSET_TYPE,
   Operation.EQUALS, assetType);
QueryExpression qe = typeQ.and(siteExpr);
qe.setMaxResults(maxResults);
SearchResult<ResultRow> res =
   eng.search(Collections.singletonList("Global"), qe);
```

Chapter 24

Revision Tracking

WebCenter Sites provides revision tracking functionality that prevents a row in a table from being edited by more than one user at a time. When you enable revision tracking for a table, WebCenter Sites maintains multiple versions of a row in that table.

WebCenter Sites applies this revision tracking functionality to your asset types. You decide which asset types should be tracked and determine how many revisions to store. The content providers then check their assets in and out and can compare versions, if necessary.

This chapter describes how to enable revision tracking and how to manage versions of assets.

This chapter contains the following sections:

Overview

WebCenter Sites provides revision tracking functionality through its revision tracking API. This API is used to provide additional revision tracking functionality for asset type tables.

When you enable revision tracking for an asset type, WebCenter Sites creates a new table, called a **tracker** table, for assets of that type. You specify how many versions you want to keep and you also specify a storage directory that the tracker table uses to store supporting files for the assets that it is tracking.

WebCenter Sites' implementation of revision tracking provides the following features:

- **Check out** and **check in**.

  Check out locks an asset so that only one user can edit it at a time.

  Check in releases the lock on the asset, increments the version number, and determines whether the number of versions falls within the configured limit. If the new version exceeds the limit, the oldest version is deleted to make room for the next version.

- Storage of **multiple versions** of an asset.

  When you enable revision tracking for an asset type, WebCenter Sites stores versions in a **tracker** table; upload data is stored in a storage directory that you specify, as shown in "The RTInfo Table," on page 447. Because past versions are stored (that is, a history exists), a user can **roll back** an asset to a previous version or examine the **differences** between two versions of the asset.

- Administrative or maintenance features.

  An administrator can **delete past versions** of an asset or **clear the checkout** for an asset by overriding the check out on it and checking it back in.

When an asset type is being tracked, WebCenter Sites provides checkin, checkout, and other revision tracking features on the **New** and **Edit** forms for assets of those types.

Tracker Tables and Storage Directories

When you enable revision tracking for an asset type, WebCenter Sites creates a tracker table that stores revision information for the records in the source table. A tracker table has the same name as the main storage table for the asset type with `_t` appended to it. For example, the tracker table for the article asset type would be named `Article_t`. The tracker table for the attribute type asset type would be named `AttrTypes_t`.

For each record in a tracked asset type table, there are several rows in the corresponding tracker table that stores its version information. Tracker tables have two kinds of columns:

- Columns that store the system information that the revision tracking system needs to keep track of all the versions
- Columns that hold the IDs of text files that are stored in a storage directory

When a new version of an asset is checked in, WebCenter Sites creates a separate text file to hold the data in each of the asset type's upload (URL) fields and in any text fields that are configured to hold more than 64 characters. These files are stored in `{shared}/rtStorage/{AssetType}`. There is a set of these text files stored in the storage directory for each version of the asset.

Note

Because tracker tables hold system information only, they are hidden in Sites Explorer. Do **not** attempt to modify the information in any of the tracker tables with a database tool.

The RTInfo Table

While the tracker tables are kept hidden, the `RTInfo` table is visible through WebCenter Sites Explorer. This table holds information about which tables are being revision tracked. It has the following columns:

Column	Description
`tblname`	The name of a table that is being revision tracked. For asset types, this is the name of the main asset type table.
`versions`	The number of versions to store for each asset of this type.
`storage`	The path to the storage directory that holds the text files for each version of assets of this type. The path is in the shared folder, located in *`{shared}`*`/rtStorage/`*`{AssetType}`*.
`recordupdate`	A timestamp of the last time a version was stored in the tracker table.
`trackingupdate`	The time at which revision tracking was enabled for the source table.

Revision Tracking and the Two Asset Models

Because the data model for basic assets is different from the data model for flex assets, the revision tracking system works differently for the two asset models:

- For basic assets, only the row in the main asset storage table is tracked.
- For flex assets and the other multi-table asset types (template and CSElement), the information from the appropriate rows from all of their tables are serialized into an object and stored in the tracker table for that asset type.

For example, if you enable revision tracking for template assets, the appropriate rows from the `Template`, `SiteCatalog`, and `ElementCatalog` tables are serialized into an object and stored in the `Template_t` table.

Implicit vs. Explicit Checkin and Explicit Checkout

When revision tracking is on for an asset type, WebCenter Sites provides both implicit (or automatic) and explicit (or manual) checkout, and explicit checkin functionality. When users create or edit assets of a type that is being revision tracked, they do not have to manually check out the asset: it is automatically assigned to them. However, implicit checkin is not allowed; when they click **Save**, the asset is not checked back in. Users must manually check assets in.

This may or may not be the behavior that you want. For example, if an author is making extensive revisions to an asset that was checked out implicitly, the changes will not be available until the user also explicitly checks in the asset. This will help prevent the user from overwriting an older version that he or she really wanted to keep.

When a content provider implicitly or explicitly checks out an asset, a version is not stored—no matter how many times he or she saves it—until it is manually checked back in.

Revision Tracking and Non-Asset Tables

In addition to using revision tracking for your asset types, you can implement revision tracking on your non-asset tables.

To do so, you use the **Management Tools** feature in the **Admin** tab to enable tracking for the table. WebCenter Sites then creates a corresponding tracker table to support the tracked table. It is named the same way as a tracker table for an asset type: `nameOfTable_t`. For example, if you enabled revision tracking for the Source table, the tracker table would be called `Source_t`.

When revision tracking is enabled for a non-asset table, you can use either the revision tracking features accessible from the menus in Sites Explorer or the "Management Tools" node on the "Admin" tab to lock (check out) a row and then unlock the row (check it back in) when you are finished with it.

If you need to provide additional support outside of the Sites Explorer tool for revision tracking of a non-asset table, your developers can code additional forms, using the WebCenter Sites revision tracking API and revision tracking XML or JSP tags. For information, see the *Oracle WebCenter Sites Developer's Tag Reference* and the *Oracle WebCenter Sites Java API Reference*.

> **Note**
>
> If you need to delete a non-asset table from the WebCenter Sites database and that table is being revision tracked, be sure to untrack the table before deleting it.

How Many Versions?

Each revision of an asset or a database row occupies disk space. Therefore, your decision about how many revisions to keep must be based on the following factors:

- The amount of disk space that you have available
- The typical data size of the asset (or row)
- The likelihood that there could be a need for a rollback of several versions

For example, an asset that consists of a small amount of ASCII data occupies so little space that a large number of revisions would take little disk space. However, each version of an asset that holds a large amount of binary data could occupy a significant amount of disk space. In the second case, you must strike the appropriate balance, storing the fewest number of versions necessary for rollback purposes.

Enabling Revision Tracking

This section shows you how to enable revision tracking for assets and for non-asset tables.

Note

If Revision Tracking is turned on at the publishing target destination for either ElementCatalog or SiteCatalog, then the publishing of template will fail and the template may become corrupted.

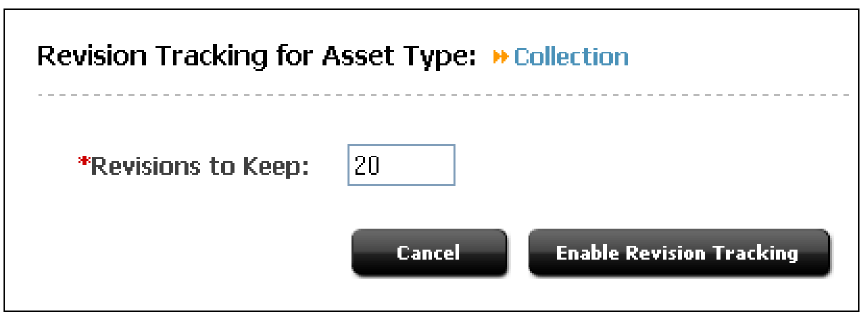
It is not recommended to enable Revision Tracking on the target destination.

Enabling Revision Tracking for Assets

To enable revision tracking for assets

1. In the **Admin** tab, expand **Asset Types,** then expand the asset type for which you want to enable revision tracking.
2. Under the selected asset type, expand **Revision Tracking** and double-click **Track**.

   The "Track Asset Type" form appears:



3. In the **Revisions to Keep** field, enter the number of revisions you want WebCenter Sites to store for each asset of the selected asset type. Once this many revisions are stored, the oldest revision is overwritten by the next revision.

   If you are using revision tracking solely for its record-locking feature and you do not need the ability to roll back to previous versions, you can set this field to 1.
4. Click **Enable Revision Tracking**.

Enabling Revision Tracking for Non-Asset Tables

To enable revision tracking for database tables that do not hold assets, use the **Management Tools** node.

To enable revision tracking for non-asset tables

1. In the **Admin** tab, expand **Management Tools** and double-click **Revision Tracking**.
2. In the form that appears, select **Track Tables** and click **OK**.
3. In the **Enter root storage directory** field, enter the full path to the directory in which revisions of table rows will be stored. Do not add a slash (or backslash) character after the directory name.

4. In the **Enter number of revisions to keep** field, enter the number of revisions WebCenter Sites should store for rollback purposes.
5. Select the **Track?** check box next to each table for which you are enabling revision tracking.
6. Click **Track Tables**.

Editing Revision Tracking Settings

You must have the SiteGod ACL to either inspect or edit revision tracking settings. If you attempt to examine the current revision tracking settings for a tracked asset type or non-asset table and you do not have the SiteGod ACL, the system does not display the name of the root storage directory or the number of revisions.

Because revision tracking settings directory affect the database, you must be careful when changing these settings.

Changing the Root Storage Directory

Caution

If you change the root storage directory for an asset type or non-asset table, you will lose all the versions currently stored for it.

If you must change the root storage directory, follow these steps:

1. Disable revision tracking for the asset type or table. When you do this, all the version data for the asset type is orphaned.
2. Enable revision tracking for the asset type or table, entering the new root storage directory.

Changing the Number of Revisions

Increasing

If you want to increase the number of revisions to be stored for an asset type or table, simply increase the value in the **Revisions to Keep** field:

- **For asset types:** in the **Admin** tab, expand **Asset Types**, the asset type you want to modify, **Revision Tracking,** and double-click **Set Revisions**. Use the form that appears to increase the value.
- **For non-asset tables:** in the **Admin** tab, expand **Management Tools**, **Revision Tracking**, and double-click **Set Table Revisions**. Use the form that appears to increase the value.

Decreasing

Although you might need to decrease the number of versions while you are testing configuration settings on a Sites development system or while you are fine-tuning the Sites management system, it is best if you do not decrease the number of versions being stored by Revision Tracking on a fully functioning management system.

If you decrease the value in the **Revisions to Keep** field to a number that is less than the number of revisions currently being stored for that asset type or table, the following occurs:

- The text files in the storage directory for the extra versions are orphaned.
- The rows in the tracker table for the extra versions are orphaned.

You can avoid creating orphan rows in the tracker table by deleting the extra versions (the oldest ones) before you decrease the number of revisions. However, you cannot avoid creating orphan text files in the storage directory and you should not attempt to delete them because it is very difficult to determine which ones to delete.

If you must decrease the number of revisions, complete the following steps (in this order):

1. Follow the steps in the procedure "Deleting Revisions."
2. To decrease the number of revisions, do one of the following:

- **For asset types:** in the **Admin** tab, expand **Asset Types**, the asset type you want to modify, **Revision Tracking,** and double-click **Set Revisions**. Use the form that appears to decrease the value.
- **For non-asset tables:** in the **Admin** tab, expand **Management Tools**, **Revision Tracking**, and double-click **Set Table Revisions**. Use the form that appears to decrease the value.

Deleting Revisions

Under certain conditions, you might need to delete old versions for an asset type or table that is being revision tracked.

To delete revisions

1. In the **Admin** tab, expand **WebCenter Sites Administrator Tools** and double-click **Revision Tracking**.
2. Enter the name of the table you want to work with. If you do not know the name of the table, do one of the following:
    - Leave the field blank. WebCenter Sites will return a list of all tables in the database.
    - Enter a partial name, ending with the wildcard character (**%**). WebCenter Sites will return a list of tables named similarly to your criteria.
3. In the **Enter Value for Key** field, enter the ID of the asset or table row whose versions you want to delete. (You can obtain the ID of an asset by inspecting it in the WebCenter Sites interfaces. You can obtain the ID of a row by examining the table with WebCenter Sites Explorer.)
4. Select **Delete Revisions** and click **OK**.
5. In the "Delete Revisions" form, select the option next to the versions that you want to delete. You can use the the **All** button to select all of the boxes and the **None** button to clear all of the boxes.
6. Click **Delete Revisions**.

Disabling Revision Tracking

When you disable revision tracking for an asset type or a non-asset table, the following occurs:

- The tracker table is inactivated, but not deleted. This means that the versions stored in it are orphaned, but not deleted. And, after revision tracking is disabled for a table, you can no longer delete versions by using the WebCenter Sites revision tracking forms.
- All links to the text files in the storage directory are broken, but the files themselves are not deleted. They, too, are orphaned.

If you later decide to enable revision tracking for that asset type or non-asset table, the old versions and text files are ignored. They remain orphaned.

Because these orphaned versions and text files take up disk space and there could be a large number of them stored on disk, it is best practice to complete the following tasks in the following order when you want to disable revision tracking:

1. Use the Management Tools revision tracking forms to delete all the versions stored for the table. See the procedure "Deleting Revisions," on page 451.
2. Disable revision tracking for the table. See the procedures in this section.
3. Manually delete all the text files from the revision tracking directory located inside the storage directory for that asset type or database table.

   For example, if the storage directory for the asset type is `/Storage/AssetType`, then delete all the files from the `/Storage/AssetType/AssetType` revision tracking directory.

Note

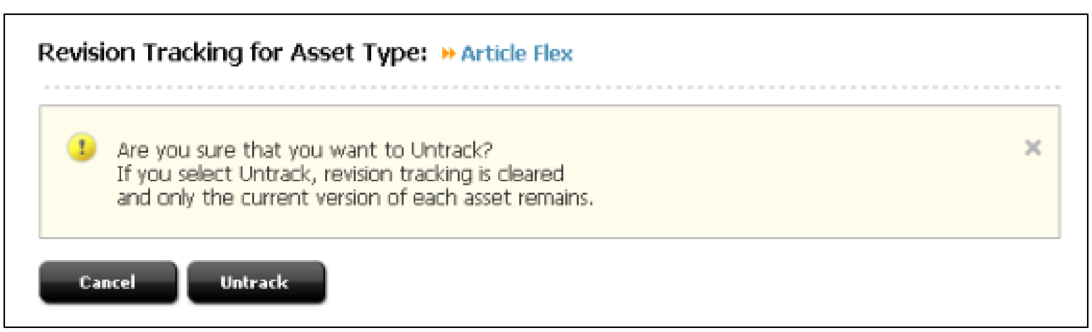
Do **not** delete any files from the asset type storage directory.

Disabling Revision Tracking for Asset Types

To disable revision tracking for asset types

1. In the **Admin** tab, expand **Asset Types**, then the asset type for which you want to disable revision tracking.
2. Under the selected asset type, expand **Revision Tracking** and double-click **Untrack**.

   WebCenter Sites displays a warning message:



3. Click **Untrack**.

Disabling Revision Tracking for Non-Asset Tables

To disable revision tracking for non-asset tables

1. In the **Admin** tab, expand **Management Tools** and double-click **Revision Tracking**.
2. In the “Revision Tracking” form, select **Untrack Tables** and click **OK**.
3. In the “Untrack Tables” form, select the **Untrack?** check box next to each table for which you want to disable revision tracking.
4. Click **Untrack Tables**.

Unlocking Revisions

Occasionally, user operations may leave an asset in an inappropriate locked state. Resolving these states is an administrative responsibility.

- For asset types, use the **Clear Checkouts** function in the **Admin** tab.
- For non-asset tables, use the **Revision Tracking** function under **Management Tools**.

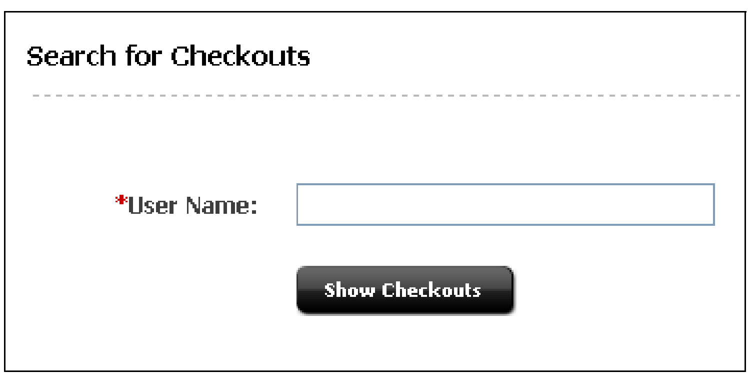
Clearing Checkouts for Assets

Before you begin, find out the name of the user who has the asset locked. This information is listed on the “Inspect” form for the asset.

To clear checkouts for assets

1. In the **Admin** tab, double-click **Clear Checkouts**.

   WebCenter Sites displays the “Search for Checkouts” form:



2. Enter the name of the user who has left the asset locked and click **Show Checkouts**.
3. In the list of assets checked out to the selected user, select the **Clear** check box next to each asset for which you want to undo the checkout.
4. Click **Clear Checkouts**.

Unlocking Versions for Non-Asset Tables

To unlock rows for a non-asset table that you are revision tracking, you need to determine the object ID of the row that you want to unlock. You can use WebCenter Sites Explorer to examine the table and determine the object row.

Once you have obtained the table's object ID, complete the following steps:

1. In the **Admin** tab, expand **Management Tools** and double-click **Revision Tracking**.
2. In the form that appears, enter the table's object ID, select **Unlock Rows**, and click **OK**.
3. In the "Unlock Rows" form, select the **Unlock** check box next to each row you want to unlock. Use the **All** button to select all rows in the list; use the **None** button to clear your selection.
4. Click **Unlock Records**.

Additional Revision Tracking Functions for Non-Asset Tables

The administrator's interface allows you to change the status of revisions in every way that a user normally does. This allows you to correct inappropriate states by administrative intervention when some error occurs.

The options all appear as radio buttons on the **Revision Tracking** screen:

- Lock
- Commit
- Release
- Rollback
- History
- Track Tables (see "Enabling Revision Tracking," on page 449)
- Untrack Tables (see "Disabling Revision Tracking," on page 452, above)
- Set Table Revisions (see "Editing Revision Tracking Settings," on page 450)
- Delete Revisions (see "Deleting Revisions," on page 451
- Unlock Rows (see "Unlocking Revisions," on page 453)

A summary of each of the other operations appears below.

Lock

The "Lock" form presents a list of rows that is restricted by any criteria that you specified on the "Revision Tracking" form:

- A red lock icon appears next to the rows that are locked by another user.
- A blue lock icon appears next to rows locked by you.

You can lock any asset by clicking the check box in the **Lock?** column and clicking the **Lock** button. The attempt fails if you do not have permission to lock the checked item. If it succeeds, the item is now checked out to you.

Commit

The “Commit” form presents a comment box and a list of rows checked out to you.

You can enter a comment and check the box in the **Commit** column for each item that you want to update. The check box in the **Keep Locked?** column specifies whether you want the item to be unlocked after it is committed (checked in).

The Commit action updates the revision to the version you have been editing.

Release

The “Release” form presents a list of rows that are checked out to you.

You can check the box in the **Release?** column for each item that you want to release. A release differs from a commit in that any changes that you have made to the asset since locking it are lost.

This form differs from the “Unlock Rows” form in that it only lists assets checked out to you.

Rollback

The “Rollback” form presents a list of rows and their versions. The list is restricted by any criteria that you entered on the “Revision Tracking” form.

Note

To roll back an asset to a previous version, use the buttons on the asset’s “Inspect” or “Status” form.

You can select any version for the rollback operation by clicking the radio button in the **Rollback?** column. A rollback creates a new version. It matches the version that you rolled back to but adds the comment “Version created by Rollback.”

History

The “History” form is similar to the “Rollback” form, except that it is informational only. For each row that it shows, it provides the date and revision information.

Part 6

Performance Enhancement Features

This part describes the inCache framework and the process of configuring page caching as well as asset caching.

This part contains the following chapters:

- Chapter 25, "inCache Framework"
- Chapter 26, "inCache for Page Caching"
- Chapter 27, "inCache for Asset Caching"

Chapter 25

inCache Framework

Oracle WebCenter Sites and Satellite Server ship with a caching system called the *inCache framework*, or simply *inCache*. This system is built on top of Ehcache, an open source, standards-based product from Terracotta. Compared to the legacy method of caching to the database, inCache offers improved website performance. This chapter introduces the inCache framework.

This chapter contains the following sections:

What is inCache?

inCache is a high performance, memory-based page and asset caching system that eliminates the need to cache Oracle WebCenter Sites' data in a central, shared repository and shared file system. inCache is based on Ehcache, an open source Java caching framework from Terracotta, and can be implemented on top of any caching system. (For more information about Ehcache, see `http://ehcache.org/`.)

The inCache framework replaces the legacy page caching framework (which can be restored as described on page 466). inCache also supports asset caching, by default. The inCache framework offers the following benefits:

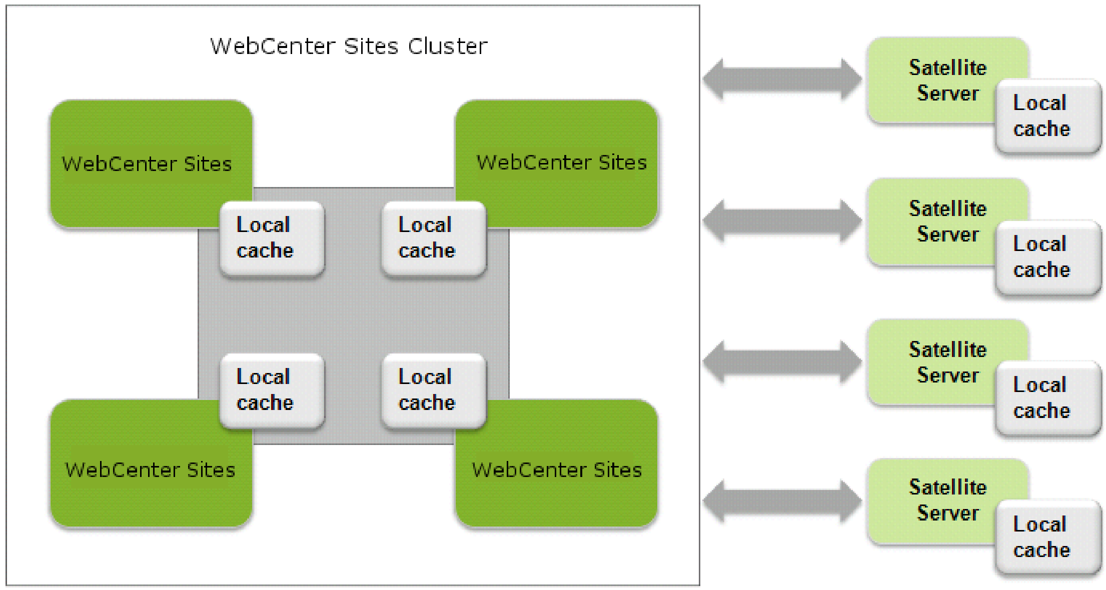
- High performance. Fast response times allow for a greater frequency of publishing sessions. Efficient page and asset invalidation ensures that updated content is quickly served.
- Decentralized architecture. Nodes maintain their own local caches. The inCache framework eliminates database caching and shared disks.
- Broadcasting system. Each individual node no longer has to have a complete view of the entire cache. Nodes listening for changes to content respond to broadcasts as necessary, depending on their individual content.
- Improved linear scalability, enhanced for data on disk and in memory.
- Failover and persistence. Nodes that are shut down retain data in their local caches and update themselves upon restart against a centrally managed record of invalidations.
- On-demand page evaluation and invalidation. Nodes validate and update their currently cached content only when the content is requested.

This chapter describes mainly page caching. Throughout, the term "page" is used to mean rendered page rather than page asset. To follow the discussion of asset caching (chapter 27), we recommend first reading the rest of this chapter, especially "How inCache Works," for a basic understanding of cache containers and how they function in relation to each other.

How inCache Works

In a clustered environment, inCache is implemented in a decentralized architecture outside the cluster's database. Each node has its own local cache, as shown in Figure 18, on page 461. Each Oracle WebCenter Sites node keeps its local cache current by listening for broadcasts from other Oracle WebCenter Sites nodes and communicates updates to remote Satellite Server via HTTP.

Figure 18: Local Caches



A node's local cache is partitioned:

- The `pageByQry` cache stores the node's web pages.
- The `dependencyRepository` cache stores identifiers of the assets that make up the web pages.

> **Note**
>
> The `pageByQry` cache is also called "page cache."
>
> The `dependencyRepository` cache is also called "dependency cache."

- The `notifier` cache broadcasts identifiers of assets that have been modified by editorial or publishing processes. Broadcasts are initiated by the nodes on which assets are modified. How a listening node responds to a broadcast depends on the node. For example:

  1. A WebCenter Sites user at node `A` edits asset `A`.
  2. The notifier cache on node `A` broadcasts a notice of change to all other nodes.
  3. Every WebCenter Sites node that contains asset `A` responds by invalidating the asset in its own local cache. Each node refers to its own `dependencyRepository` cache, marks the asset's identifier as invalid, and increments the dependency generation counter for the entire cache. Because the invalidated asset is no longer available to pages that reference the asset, the pages themselves are invalidated. However, **the node does not evaluate the pages until they are requested**. Herein lies the performance benefit.

     When a node responds to a request for a page that has a dependency on the invalidated asset, the node refers to its `pageByQry` cache, evaluates the page, determines the page to be invalid, flushes the page, generates the new page, serves the page to the visitor's browser, and records the new page along with its **dependencies** in the local cache (dependencies are recorded as identifiers of the assets that make up the page). From that point on, the same page is served from the node's local cache until the page expires or its assets are again invalidated.

WebCenter Sites and Satellite Server use the same caching framework, with one main difference: Satellite Server nodes still communicate with WebCenter Sites nodes via HTTP.

Note

While the `pageByQuery` cache is used only for page caching, it illustrates how a cache container works. For asset caching, the inCache framework introduces a counterpart container called "`AssetCache`." It works in a similar manner to `pageByQuery` and interacts with the `dependencyRepository` and `notifier` caches, described above. Additional information about asset caching is available in Chapter 27, "inCache for Asset Caching."

Restarting a Node

When a WebCenter Sites or Satellite Server node is shut down, data in its cache persists, but can become quickly outdated as the active nodes continue to invalidate assets. As a result, restarting a node requires updating its cache. Update on restart is ensured by a common invalidation memory, which is stored as a table in the database of the WebCenter Sites cluster and kept available to all nodes in the cluster for use upon restart. The table is named `FW_InvalidationMemory`.

The invalidation memory stores records of asset invalidations, specifically, identifiers of assets that have been modified or deleted during a content management or publishing process. Table growth is checked by a timer-based cleanup mechanism that runs at 15-minute intervals to purge invalidation records for the oldest period of time.

When a node is restarted, it attempts to recover information that it missed during its inactive period and therefore refers to the invalidation memory:

- If invalidation records exist for the node's inactive period, the node replays them on itself.
- If no assets were invalidated during the inactive period, the node continues to operate as if it were never shut down.
- If invalidation records were purged by the cleanup mechanism while the node was inactive, the node's cache self-destructs and must be rebuilt.

When a remote Satellite Server is restarted, it obtains the information it missed by sending a request to WebCenter Sites for an update.

Page Regeneration During RealTime Publishing

Enabling inCache deactivates the `donotregenerate` flag for the RealTime publishing process. Because the flag is no longer recognized, crawling is used to regenerate pages during RealTime publishing sessions. Crawling is a computationally expensive option. If it is not implemented, pages will be regenerated only when they are requested.

The crawling option requires specifying a set of URLs to be analyzed by the WebCenter Sites page regenerator. Typically, they are the URLs of the home page and other high-traffic pages. In addition, you can specify the depth to which those pages will be crawled. For example, a depth of 1 means that the specified pages and pages they link to will be crawled, while a depth of 0 means that only the specified pages will be crawled. Crawled pages are regenerated only if their component assets have been invalidated during the publishing session, or the pages are not cached. All pagelets on the specified pages are regenerated in the process.

The list of URLs to crawl and the crawl depth must be specified in the `FW_RegenCriteria` table, which is created on the delivery system during the first publishing session after inCache is configured. The `ft_ss` parameter can be included in the URL to specify that requests are handled either directly by WebCenter Sites or by remote Satellite Server. Instructions for configuring inCache and enabling page regeneration can be found in Chapter 26, "inCache for Page Caching."

Double-Buffered Caching

WebCenter Sites' double-buffered page caching method uses the WebCenter Sites and Satellite Server caches in tandem on live web sites. Double buffering ensures that pages are always kept in cache, either on WebCenter Sites or Satellite Server, to protect WebCenter Sites from an overload of page requests and prevent the live web site from displaying blank pages and broken links.

To maintain the traditional system of double-buffered caching for the inCache framework, we ensured that remote Satellite Server continues to communicate with WebCenter Sites via http requests. Satellite Server still reads page data via http requests and caches in the usual way. However, page data now includes dependency information in the form of a comma-separated list of asset identifiers which is also streamed to remote Satellite Server.

inCache Features for Remote Satellite Server

Remote Satellite Server is used only in page caching. It can be configured to support advanced functionality such as page propagation and page regeneration in background.

Page Propagation

The page propagation option enables all WebCenter Sites nodes and Satellite Server nodes to host the same pages without each node having to regenerate the pages. Instead of referring to the database to regenerate pages, nodes receive newly generated and regenerated pages into their local caches from the nodes on which the pages were (re)generated and cached. Caching the pages triggers their propagation. Instructions for configuring page propagation can be found in Chapter 26, "inCache for Page Caching."

Page Regeneration in Background

Remote Satellite Server can be configured to serve invalidated pagelets while they are being regenerated by a background process. See "Configuring for Pagelet Regeneration in Background," on page 475 for more information.

Summary

The inCache framework significantly increases performance. Nodes can retain cache on disk and recover from failure. Decentralized architecture prevents bottlenecks (although the lack of a central repository for cached items can make it difficult to determine the overall state of all caches). Page propagation eliminates the need to regenerate pages, while page regeneration in background enables remote Satellite Servers to continue serving pages while their replacements are being generated.

This new framework deactivates the `donotregenerate` flag for RealTime publishing. Pages that must be regenerated during a publishing session must be specified in the `FW_RegenCriteria` table. Otherwise, pages are regenerated when they are requested.

Next Steps

For more information about page caching and configuration methods, see Chapter 26, "inCache for Page Caching." The same chapter contains information about options such as striping the disk, enabling page regeneration to occur during RealTime publishing, setting up page propagation, configuring page regeneration in background, and returning to the traditional system of page caching.

For information about asset caching, see Chapter 27, "inCache for Asset Caching."

Chapter 26

inCache for Page Caching

Overview

inCache for page caching is enabled by default when WebCenter Sites is installed or upgraded. inCache for page caching overrides the legacy method of page caching. Two configuration files, `cs-cache.xml` and `ss-cache.xml`, are provided with WebCenter Sites for configuring local caches and peer-to-peer communication.

inCache supports disk striping and affects RealTime publishing by deactivating the `donoteregenerate` flag. The option to enable page regeneration during RealTime publishing requires populating the `FW_RegenCriteria` table, on the delivery system, with the URLs of pages to be crawled and regenerated. Page propagation is also an option, used to ensure that all nodes host the same pages without each node having to regenerate the pages. In addition, remote Satellite Server can be configured to continue serving stale pages for a short duration while their replacements are being regenerated.

Note

You can return to the legacy page caching method, as necessary, by:

1. Setting the VM argument on all WebCenter Sites and Satellite Server nodes as follows: `–Dcs.useEhcache=false`
2. Reconfiguring all WebCenter Sites and Satellite Server nodes to use the database and shared file system.
3. You will also have to purge the legacy page cache before resuming operations, given that it is likely to have become outdated since the time it was disabled.

In general, switching between caching methods requires you to reconfigure the WebCenter Sites and Satellite Server nodes to use the same cache, and to purge the cache before its use, as recommended by best practices.

Configuring Your System for inCache Page Caching

The configuration process consists of a set of required steps and optional optimization steps, shown below. For example, you will enable multiple WebCenter Sites nodes to communicate with each other, and set storage properties for each local cache on the WebCenter Sites and remote Satellite Servers.

To configure your system for inCache page caching

1. Ensure that inCache for page caching is enabled. Look for the following VM argument: `–Dcs.useEhcache` and verify that it is either set to `true` or not set (in which case its value is assumed to be `true`).
2. Configure each WebCenter Sites and Satellite Server node to retain cache content on system restart. Pass the following VM argument:

   `–Dnet.sf.ehcache.enableShutdownHook=true`

3. Optional optimization. Set the `diskStore` path for each WebCenter Sites and Satellite Server node. If multiple nodes are on the same machine (vertical cluster) ensure that each node has a unique `diskStore` path. In the `cs-cache.xml` and `ss-cache.xml` files, set the property `diskStore path="<path to disk store>"`. Both `xml` files are stored on each WebCenter Sites node: `cs-cache.xml` for WebCenter Sites and `ss-cache.xml` for co-resident Satellite Server. Only the `ss-cache.xml` file is stored on remote Satellite Server. The files are located under the `WEB-INF/classes` folder.

   For more information about the `diskStore` property, refer to the documentation on the Ehcache web site (at the time of this writing, the URL is `http://ehcache.org/documentation/configuration.html`).

   Note

   inCache exists on WebCenter Sites, remote Satellite Servers, and co-resident Satellite Server. Architecturally and functionally inCache is identical on co-resident and remote Satellite Servers. However, the following recommendations are applied to co-resident Satellite Server:

   1. Disk persistence is turned off.
   2. The size of each cache is kept smaller than the size of WebCenter Sites' cache. This is to prevent overloading memory. We recommend keeping the sizes small except under special circumstances, such as the requirement for full double-buffered caching on co-resident Satellite Server.

4. Required for multiple nodes. Configure automatic node detection for WebCenter Sites clusters (Satellite Server nodes cannot be clustered). As an example, this step uses three WebCenter Sites nodes named CS1, CS2, and CS3.

   a. On each WebCenter Sites node ensure that `cs-cache.xml` and `ss-cache.xml` (in `WEB-INF/classes`) specify a `cacheManagerPeerListenerFactory`, which will be used to create a `CacheManagerPeerProvider`. The provider detects other nodes in the cluster. Configure automatic detection as shown in the following example:

```
<cacheManagerPeerProviderFactory
class="net.sf.ehcache.distribution.RMCacheManagerPeer
  ProviderFactory"
properties="peerDiscovery = automatic,
multicastGroupAddress = 230.0.0.1,
multicastGroupPort = 4444, timeToLive = 32"/>
```

> **Note**
>
> The multicastGroupPort property must be set identically across cluster members. For example, if CS1 specifies multicastGroupPort=4444 in its cs-cache.xml file, then CS2 and CS3 must specify the same setting in their files.
>
> The timeToLive property specifies the number of hops allowed to a packet, which determines how far the packet propagates. Valid values range between 0 and 255 with the following restrictions:
>
> | | | | |
> |---|---|---|---|
> | 0 | - same host | 64 | - same region |
> | 1 | - same subnet | 128 | - same continent |
> | 32 | - same site | 255 | - unrestricted |

b. On all Satellite Servers, co-resident and remote, do one of the following:

- Set the multicastGroupPort property in each ss-cache.xml file to a unique value.
- To support cache replication, set the multicastGroupPort property in each ss-cache.xml file to the same value.

> **Note**
>
> The value(s) that you set in the ss-cache.xml files must be different from the value for WebCenter Sites' multicastGroupPort property in the cs-cache.xml files.

When inCache is configured, the system will start using the configurations specified in cs-cache.xml and ss-cache.xml. Caches are initialized upon the first call to any page in WebCenter Sites, cached or not.

5. Optional optimization. Configure the local caches as necessary, using Table 8, on page 470 as a property reference. The local cache for each WebCenter Sites and remote Satellite Server node is partitioned. Each part is defined in its own <cache> tag in the cs-cache.xml and ss-cache.xml files. The parts are named as follows:

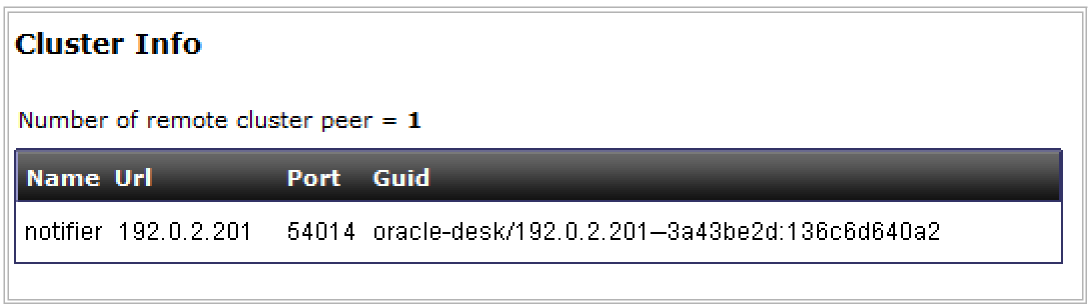
- pageByQry: This is the cache for the page data itself, keyed by the query url.
- dependencyRepository: This is the cache of dependencies on which the pages are built. When pages are added to the pageByQry cache, entries are automatically created in the dependencyRepository cache. Each entry in this cache is an asset id or unknowndep or unknowndep-<type>. Therefore, this cache contains at most the number of assets in the system and a handful of items for different variations of unknowndeps.

  When a page with no dependencies is cached, that page logs a dependency on _NODEP_ (a single item in the dependencyRepository cache used to identify pages with no dependencies). The _NODEP_ item remains associated with the page until that page either expires or is manually flushed. On remote Satellite Servers, you can disable caching of pages without dependencies by setting the

JVM option
`-Dignore_nodep_pages` to `true` (per Satellite Server).

- `notifier`: This cache is used by active WebCenter Sites cluster members to notify other WebCenter Sites cluster members of changes to content.

6. Restart all configured WebCenter Sites and Satellite Server nodes.
7. Verify that all active cluster members are in the caching network and recognize each other. Use the “Cluster Info” diagnostic tool, which lists all WebCenter Sites members that have a notifier cache. To invoke “Cluster Info”:
   a. Bootstrap inCache by rendering a cacheable page.
   b. Log in to the WebCenter Sites Admin interface as a general administrator (`fwadmin/xceladmin`, by default).
   c. Verify that WebCenter Sites cluster members and co-resident Satellite Servers are communicating with each other. Open the **Admin** tab and go to **System Tools** > **Cache Management** > **Sites Cache** > **Cluster Info** (also > **Co-Res SS Cache** > **Cluster Info**).


Cluster Info

Number of remote cluster peer = **1**

Name	Url	Port	Guid
notifier	192.0.2.201	54014	oracle-desk/192.0.2.201—3a43be2d:136c6d640a2

Various types of page caching statistics are available in the Cache Configuration tool. For more information, see Chapter 28, “System Tools.”

8. If you wish to stripe disks, enable page regeneration during RealTime publishing, or enable page propagation, see “Tuning Options,” on page 471.

Table 8: Cache Configuration Properties

Property	Required?	Description
`name`	Y	Specifies the name of the cache. **Legal values:** `pageByQry` `dependencyRepository` `notifier` For descriptions, see step 5 on page 468.
`diskPersistent`	N	Specifies whether to persist data on disk between restarts of the JVM. **Default / recommended value:** `true` **Note:** The `diskPersistent` setting must be consistent across the `pageByQry`, `dependencyRepository`, and `notifier` caches to prevent potential conflicts in generation count.
`maxElementsInMemory`	Y	Specifies the maximum number of objects to store in memory. **Default value:** `200000`
`maxElementsOnDisk`	Y	Specifies the maximum number of objects to store on disk. Disks can be striped. For information see pg. 471. **Default value:** `1000000`
`eternal`	Y	Specifies whether cache will be cleared by WebCenter Sites (it is never cleared by inCache). **Default:** `true` **Do not change the value of this property.**
`overflowToDisk`	Y	Specifies whether the memory cache is allowed to overflow to disk-based cache. **Default value:** `true` **Do not change the value of this property.**
`diskSpoolBufferSizeMB`	N	Specifies the size of the disk buffer, in megabytes. If you expect many disk I/Os (i.e., disk cache is much larger than memory cache) set the buffer to 20MB. **Default value:** `5`
`memoryStoreEvictionPolicy`	N	Specifies how to remove items from cache. The recommended and default value is `LFU` (Least Frequently Used). `LRU` (Least Recently Used) is also a legal value. **Default / recommended value:** `LFU`
`clearOnFlush`	N	Specifies whether to clear memory as it is flushed to disk. **Default value:** `false` **Do not change the value of this property.**

Tuning Options

Striping the Disk Cache

The inCache framework supports striping of the `pageByQry` cache to reduce the contention that occurs when a large portion of the cache must be written to disk.

Complete the following steps on WebCenter Sites nodes and remote Satellite Servers in the inCache framework:

1. To enable striping, add the following VM argument:

   ```
   -DnumOfDiskStores=X
   ```

   where `X` is the number of stripes. Set the number of stripes to the number of unique spindles that are available to stripe over (for instance if you have 5 drives, then set `X` to `5`).

   > **Note**
   >
   > Drives used for striping the disk-based cache should not be used for any other purpose.
   >
   > The size of each `DiskStore` is the size specified by the property `maxElementsOnDisk` in the `xml` configuration file (see Table 8, on page 470). For example, if you are using 5 stripes and `maxElementsOnDisk` is set to `100000` items, a total of `500000` items can be stored.

2. Create a symbolic link or else mount the drive physically in the correct location so that the stripes are properly distributed.

   For each defined cache, the system creates a directory under the `diskStore` path (configured in step 3 on page 467). Under each directory it also creates a group of numbered directories, starting at `0`. Each directory points to a different drive.

   For example, items would be stored as follows in a disk-based cache on WebCenter Sites for `-DnumOfDiskStores = 5`:

   ```
   <custom_path>/cs-cache:
      Directory: 0
      Directory: 1
      Directory: 2
      Directory: 3
      Directory: 4
      File: dependencyRepository.data
      File: dependencyRepository.index
      File: notifier.data
      File: notifier.index
   <custom_path>/cs-cache/0:
   ```

```
   File: pageByQry.data
   File: pageByQry.index
<custom_path>/cs-cache/1:
   File: pageByQry.data
   File: pageByQry.index
<custom_path>/cs-cache/2:
   File: pageByQry.data
   File: pageByQry.index
<custom_path>/cs-cache/3:
   File: pageByQry.data
   File: pageByQry.index
<custom_path>/cs-cache/4:
   File: pageByQry.data
   File: pageByQry.index
```

In this example, the root cache contains dependency and notifier caches. Spread among the directories `0–4` are the stripes of the `pageByQry` cache. The directories `0–4` in this example were placed on five separate drives by the use of symbolic links.

Configuring for Page Regeneration During RealTime Publishing

Pages are regenerated only when they are requested. To configure page regeneration during RealTime publishing, do the following:

1. By default, the `PageCacheUpdater` is set by the WebCenter Sites installer to use `ParallelCacheRegenerator`:

   Open the `AdvPub.xml` file in the `WEB-INF/classes` folder on the delivery WebCenter Sites system and verify that the `PageCacheUpdater` section has the lines shown below:

   Note

   Do not change any of the values in the `PageCacheUpdater` section except for:

   - `numThreadsPerServer`, to specify the number of simultaneous threads for crawling
   - `regenServers`, to point to the WebCenter Sites nodes by `<address>` and `<port of server where pages will be regenerated>`

```
<bean id="PageCacheUpdater"
 class="com.fatwire.realtime.regen.ParallelRegeneratorEh"
 singleton="false">
   <property name="messenger">
     <ref local="EmbeddedMessenger_flusher" />
   </property>
   <property name="numThreadsPerServer" value="3" />
   <property name="regenServers">
     <list>
       <value>http://{address}:{port of server where pages will
          be regenerated}/servlet/ContentServer</value>
     </list>
```

```
    </property>
</bean>
```

2. Restart the delivery system.
3. RealTime publish to the delivery WebCenter Sites to create the `FW_RegenCriteria` table on that system.
4. Open the `FW_RegenCriteria` table, using Sites Explorer. Specify the pages to be regenerated and the depth of links to be followed. The `ft_ss` parameter can be included in the URLs to specify whether page requests will be handled by WebCenter Sites directly or by remote Satellite Server.

   > **Note**
   >
   > If a URL assembler is used, specify the internal URLs of the pages to crawl. The regenerator does not recognize URL-assembler URLs.

   Example:

   ```
   pagename=SiteName/HomePage&ft_ss=true
   Level=1
   ```

   WebCenter Sites will regenerate `HomePage` and the pages that are linked from `HomePage`. Given that `ft_ss=true`, the requests are treated as if they are generated from Satellite Server.
5. RealTime publish to the delivery WebCenter Sites.

   During the publishing session, the delivery system crawls all of the pages specified in the `FW_RegenCriteria` table, but regenerates only pages for which component assets were invalidated (it also generates the uncached pages).

   For example:

   a. Updated assets are published to the delivery system.
   b. The delivery system invalidates the existing dependency information for the re-published assets by marking their asset identifiers as invalid in the `dependencyRepository` cache and incrementing the dependency generation counter. Because the invalidated assets are no longer available to pages that reference the assets, the pages are invalidated.
   c. The page regenerator crawls pages with the URLs that are specified in the `FW_RegenCriteria` table and regenerates the invalidated pages. It also generates the uncached pages.

Setting Up Page Propagation

Page propagation enables all nodes in a WebCenter Sites cluster (including the Satellite Server nodes) to host the same pages without each node having to regenerate the pages.

When configured to use inCache, each WebCenter Sites has a separate JVM and thus maintains a separate, local cache. If one WebCenter Sites generates and caches a new page, none of the other WebCenter Sites have the newly cached page, nor are they informed of that page. Upon receiving a request for the same page, each node must generate the page by referring to the database and shared file system, thus putting extra load on both components. Page propagation prevents different nodes from regenerating the same page by propagating the page across the cluster.

Page propagation is triggered when pages are loaded into a node's local cache. It works as illustrated in the scenario below, starting with basic inCache functionality.

1. **inCache Page Caching for Newly Generated Pages:**
   a. Node A, a WebCenter Sites node, receives a request for a new page.
   b. Node A generates the requested page.
   c. Node A caches the new page with complete dependency information (identifiers of component assets) into its local page and dependency caches.
   d. When page propagation is enabled, step 3 follows.
2. **inCache Page Caching for Regenerated Pages:**
   a. Node B, also a WebCenter Sites node, receives a request for a page that has been invalidated (a component asset was modified. The asset is marked as invalid in the node's local dependency cache and in the dependency caches of all other nodes containing the asset.)
   b. Node B regenerates the requested page.
   c. Node B caches the regenerated page and updates its dependency (in step 2a) by incrementing the generation counter.
   d. When page propagation is enabled, step 3 follows.
3. **inCache Page Caching with Page Propagation:**

   When the node caches the (re)generated page, it also propagates the page to all other nodes. Propagated information consists of:

   - The page's complete dependency information. For example, if the page has *x* dependencies (asset identifiers), all *x* dependencies are propagated from the local dependency cache to the dependency caches of other WebCenter Sites nodes.
   - The page itself. The page is propagated from the local page cache to the page caches of other WebCenter Sites nodes.

   If the page already exists on a receiving node, the node ignores the propagation.

The following list summarizes page propagation events and conditions:

- When a page is cached on a WebCenter Sites node, its complete page information (described in step 3, above) is propagated to all other WebCenter Sites nodes.
- When a page is cached on a remote Satellite Server node, its complete page information (described in step 3, above) is propagated to other remote Satellite Servers over Java Remote Method Invocation (RMI).
- When a page is propagated:
  - The page's "last updated" time stamp is preserved on all WebCenter Sites and Satellite Server nodes.
  - The generation count on a given node remains independent of generation count on all other nodes. For example, Node A has a generation count of 10, but other nodes use a generation count of 10 for a different type of dependency. Even though the same object is propagated across WebCenter Sites or Satellite Servers, it may be assigned different generation counts by each WebCenter Sites or

Satellite Server node. The object and its “last updated/modified” time remain the same across nodes.

- The propagation is ignored by nodes on which the page already exists.

- If any of a cached page’s dependencies fail to propagate to a node where the page does not exist, the page is not cached on that node until it is requested and generated on that node.
- Because blobs are still stored in the Satellite Server caches, they are replicated across Satellite Servers.

Enabling Page Propagation

Start with a system that is configured to use inCache.

To enable page propagation

1. On all WebCenter Sites nodes:
   a. Set `propagatecache=true` in the `futuretense.ini` property file.
   b. In `cs-cache.xml`, verify that `multicastGroupPort` is the same for all WebCenter Sites nodes in the cluster to ensure they can communicate with each other. (The file is located in `WEB-INF\classes`.)
2. On all remote Satellite Servers:
   a. Set `propagatecache=true` in the `satellite.properties` file.
   b. In `ss-cache.xml`, ensure `multicastGroupPort` is identical for all Satellite Servers intended for cache propagation, but different from `multicastGroupPort` for the WebCenter Sites nodes.
3. On all nodes, initialize page propagation by rendering any page and verify that the system responds as described below:
   - On WebCenter Sites nodes, caching of the page triggers its propagation from the local cache to the caches of other WebCenter Sites nodes (as described in step 3 on page 474) while preserving the page’s “last update/modified” time stamp across all WebCenter Sites. Propagation is ignored by WebCenter Sites nodes on which the page already exists.
   - The response on Satellite Servers proceeds as on WebCenter Sites nodes, but over Java RMI.

Setting Up Page Propagation on Restart

If a node enabled for page propagation is restarted, its local caches must be re-initialized in order for the node to recognize page propagations. To re-initialize a local cache, render a cacheable page on the node; caching triggers the node to propagate the page to other nodes and recognize pages that are propagated by other nodes. When a node is restarted, the propagations it missed during its period of inactivity are not reproduced on the node, even though its local caches are re-initialized.

Configuring for Pagelet Regeneration in Background

Remote Satellite Server can be configured to serve invalidated pagelets while they are being regenerated by a background process. To enable serving of invalidated pagelets, add `serveStale=true` to remote Satellite Server's `satellite.properties` file. If a pagelet is then invalidated, it will be regenerated in one of the following ways:

- If a browser requests the pagelet *within* the next 30 minutes, remote Satellite Server starts a background process that sends a request to WebCenter Sites to regenerate the page. While the background process is running, the browser is served the invalidated pagelet from remote Satellite Server's cache. All subsequent requests will be served the invalidated pagelet until it is regenerated by the background process.
- If a browser requests the pagelet *after* 30 minutes, remote Satellite Server uses its normal process for regenerating pagelets; that is, it sends a request to WebCenter Sites to regenerate the pagelet. The requesting browser must wait until the page is regenerated.

To obtain information about the background process, set the `com.fatwire.logging.cs.cache.ehcache` logger to `DEBUG` in remote Satellite Server's `commons-logging.properties` file. `DEBUG` produces the following message at the start of the background process:

```
"Data for <cache key> is found to have been invalidated. A new
request has started processing new data in the background."
```

At the end of the process, the message reads:

```
"Background process for request <cache key> has completed
successfully, data in cache is updated."
```

Chapter 27

inCache for Asset Caching

The WebCenter Sites inCache framework supports asset caching by default. Asset caching can be customized.

This chapter contains the following sections:

Introduction to Asset Caching

Asset caching is automatically enabled on WebCenter Sites systems during installation and upgrade. Asset caching, a memory-based system, is built on the inCache framework to protect WebCenter Sites' performance by taking up load that would otherwise affect the database. Asset caching uses its own container – the `AssetCache` container – which functions by interacting with other inCache components, in particular the `dependencyRepository` and `notifier` caches.

To follow the discussion about asset caching, we recommend first reading Chapter 25, "inCache Framework" for a basic understanding of the inCache framework, especially how its cache containers – `pageByQuery`, `dependencyRepository`, and `notifier` – function in relation to each other in page caching. Note that while the `pageByQuery` cache is not used in asset caching, it serves to illustrate how a cache container works. Its counterpart `AssetCache` container works in a similar manner. Note, also, that asset caching and inCache page caching work independently of each other. Each type of caching can be enabled or disabled independently of the other.

The rest of this chapter describes the `AssetCache` container, discusses it functionality in terms of asset loading and invalidations, compares asset caching with page caching, and provides instructions for customizing caching of assets by asset type.

AssetCache Container

In WebCenter Sites, programmatic usage of assets consists of loading and reading their attributes. Given that assets are loaded by templates, which are stored in WebCenter Sites, `AssetCache` is used only on WebCenter Sites nodes.

- `AssetCache` is a component of the inCache framework. Similar to the `pageByQry` cache, `AssetCache` is independently available in the inCache framework as its own container on each WebCenter Sites node.
- The `AssetCache` container is enabled by default in each WebCenter Sites `cs-cache.xml` file, where it is named in the `<cache>` element. The `AssetCache` container functions by caching assets of all types and interacting with the inCache components listed below. Additional AssetCache containers for assets of specific types can be configured. For more information, see "Types of Asset Caching," on page 482 and "Customizing Asset Caching," on page 483.
- Interaction with the `dependencyRepository` cache
  When an asset is loaded into `AssetCache`, inCache logs a dependency on that asset by creating entries in the `dependencyRepository` cache, as shown in Figure 19, on page 480. Modifying or saving the asset removes its dependency information from the `dependencyRepository` cache and therefore invalidates the asset's entry in the `AssetCache`. In general, any save or delete operation on a cached asset invalidates that asset.

> **Note**
>
> When asset caching is enabled, assets must not be saved or deleted directly in database tables via Sites Explorer or the CatalogManager API. Save and delete operations made in this manner do not invalidate the asset cache.

- Interaction with the `notifier` cache
  Invalidations that happen on any one node are reflected on other nodes by means of the `notifier` cache. This cache broadcasts information as to which asset was changed. If a receiving node contains the asset in its `AssetCache`, the node updates its own `dependencyRepository` cache accordingly.

Asset caching works independently of page caching, but shares various cache containers, as shown in Table 9.

Table 9: Cache Containers

Function		**Cache Container**			
Asset Caching	**Page Caching**	**AssetCache**	**pageByQry**	**dependency Repository**	**notifier**
Off	Off	X	X	X	X
Off	On	X	✓	✓	✓
On	Off	✓	X	✓	✓
On	On	✓	✓	✓	✓

Asset Caching Operations

This section contains the following topics:

- Asset Loading
- Comparing Asset Caching with inCache Page Caching
- Flushing AssetCache

Asset Loading

An asset's data can be read in several ways, by using APIs or `asset:load` tags.

- The following APIs load assets into `AssetCache` in `readonly-complete` mode:
  - `AssetDataManager.read(AssetId)`
  - REST API GET for assets
- The following tag loads assets into `AssetCache` in different modes:
  - `asset:load(objectid="cid1" option="editable")`
  - `asset:load(objectid="cid1" option="readonly")`
  - `asset:load(objectid="cid1" option="readonly-complete")`

The `option` parameter determines whether an asset is cached, and how a cached asset's key is named in the `AssetCache` and in the `dependencyRepository`. Caching scenarios are summarized in Figure 19, on page 480. Note that in scenario D, you will find multiple entries in the `AssetCache` container for the same asset. This is due to the fact that an asset can be loaded in two modes: `readonly` and `read-only complete`.

Figure 19: Asset Caching Scenarios

The following modes determine whether an asset is cached and how its key is named in the `AssetCache` and `dependencyRepository`. The mode is defined in the APIs and tags listed on page 479.

A. `editable` mode. When an asset is loaded in `editable` mode it is not cached.

`AssetCache`

Key	

`dependencyRepository`

Key	

B. `readonly` mode. The asset is cached pointing to its dependency on itself. The `2` in `assettypecid1:2` specifies the `readonly` mode.

`AssetCache`

Key	
`assettype:cid1:2`	

`dependencyRepository`

Key	
`assettype:cid1`	

C. `readonly-complete` mode. The asset is cached pointing to its dependency on itself. The `3` in `assettypecid1:3` specifies the `readonly-complete` mode.

`AssetCache`

Key	
`assettype:cid1:3`	

`dependencyRepository`

Key	
`assettype:cid1`	

D. `readonly` mode at time t_1 and `readonly-complete` mode at time t_2. The asset is cached twice, pointing to its dependency on itself. The `2` in `assettypecid1:2` specifies `readonly` mode; the `3` in `assettypecid1:3` specifies the `readonly-complete` mode.

`AssetCache`

Key		**Key**
`assettypecid1:2`	t_1	
`assettypecid1:3`	t_2	

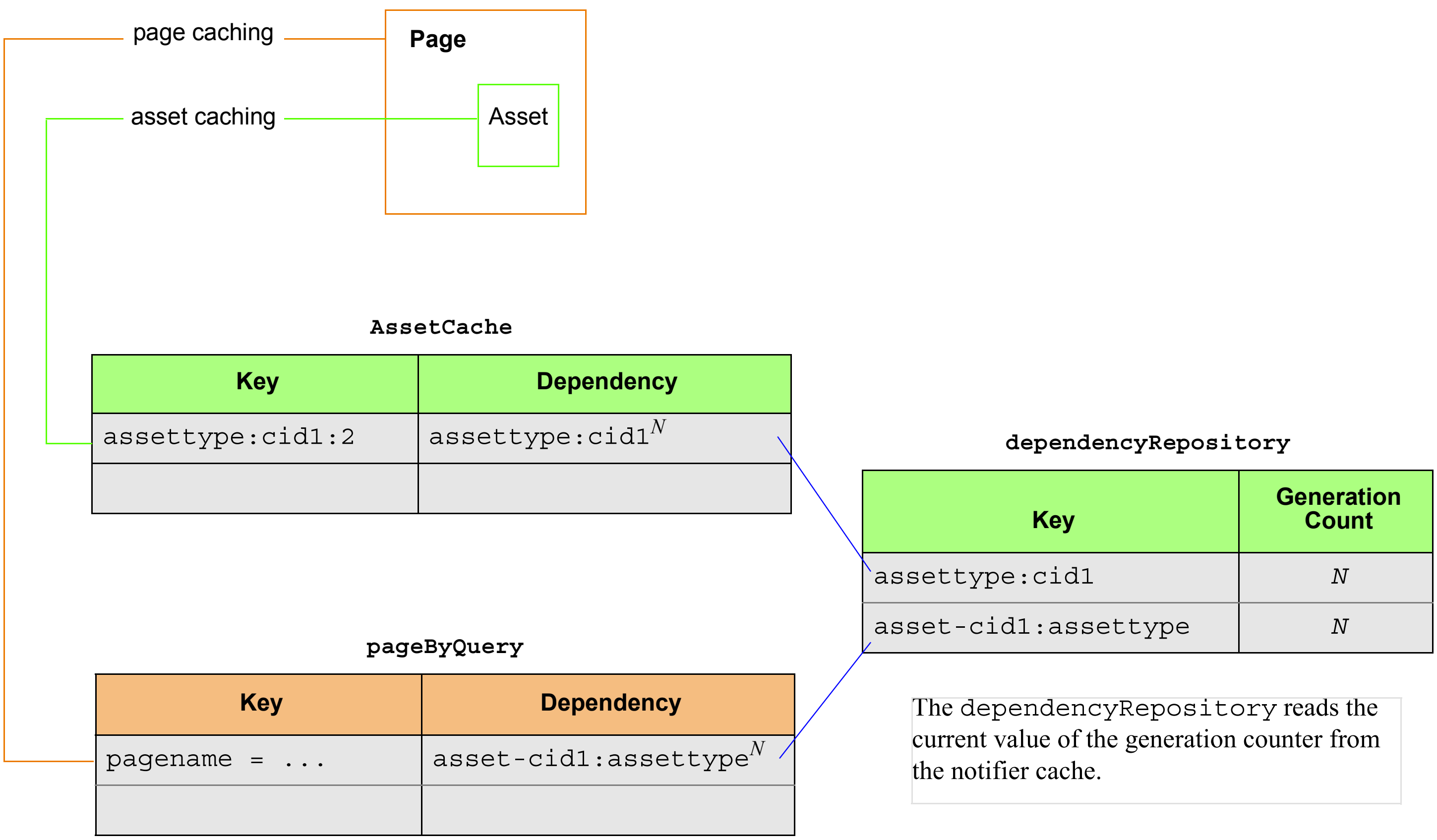
`dependencyRepository`

Key	
`assettype:cid1`	

Comparing Asset Caching with inCache Page Caching

Keys are named differently for cached assets and pages, and for their dependencies, as shown in Figure 20. Understanding the differences helps to interpret caching information displayed in the Cache Management tool (available under the **System Tools** node, in the **Admin** tab).

Figure 20: Dependency Naming Conventions in Asset Caching and Page Caching



For example, Figure 20 shows that in the `AssetCache` container, the key of a cached asset is named `assettype:cid1:2`, where the last integer represents the mode in which the asset was read (`2` for `readonly` mode). If the same asset is also loaded in `readonly-complete` mode, its key would be `assettype:cid1:3`. When an asset is called in multiple modes, you will find multiple entries in the `AssetCache` container for that asset (see Figure 19, on page 480).

The key of a cached page begins with `pagename`. Mode does not apply to pages; if a page is cached, it is cached only once.

For more information about asset loading modes, see "Asset Loading," on page 479. Information about the Cache Management tool is available in Chapter 28, "System Tools."

Flushing AssetCache

Assets can be flushed from the `AssetCache` if they are saved via the following APIs and tags:

- API
  - REST API PUT
  - AssetDataManagerWrite
- JSP Tags

  ```
  asset:save                asset:rollback
  asset:saveall             asset:deleterevision
  asset:delete              asset:undocheckout
  asset:deleteassettype     insite:edit
  asset:void
  ```

- XML Tags

  ```
  INSITE.EDIT               ASSET.VOID
  ASSET.SAVE                ASSET.DELETE
  ASSET.SAVEALL
  ```

Types of Asset Caching

When asset caching is enabled on a WebCenter Sites node, the `cs-cache.xml` file on that node contains at least one of the following `<cache>` elements:

- The default `<cache>` element, which specifies that assets of all types are cached in a generic cache container named `AssetCache`:

  ```
  <cache name="AssetCache"
     diskPersistent="true" maxElementsInMemory="100"
     maxElementsOnDisk="100" eternal="true" overflowToDisk="true"
     diskSpoolBufferSizeMB="20" memoryStoreEvictionPolicy="LFU"
     clearOnFlush="false" />
  ```

  The generic `AssetCache` container is automatically configured on all WebCenter Sites nodes during the WebCenter Sites installation process.

- A custom `<cache>` element, which specifies that assets of a given type are cached in their own container named **`AssetCache`*`Asset_type_name`***:

  ```
  <cache name="AssetCacheAsset_type_name"
     diskPersistent="true" maxElementsInMemory="100"
     maxElementsOnDisk="100" eternal="true" overflowToDisk="true"
     diskSpoolBufferSizeMB="20" memoryStoreEvictionPolicy="LFU"
     clearOnFlush="false" />
  ```

  All custom, type-specific `AssetCache` containers must be manually configured in the `cs-cache.xml` files. How many type-specific containers to create depends on which types of assets must be cached in their own containers and which conditions must be applied to caching in each container.

Note that the generic `AssetCache` container can either co-exist or be replaced with one or many type-specific asset caches. The contents of all asset caches are mutually exclusive.

That is, the generic `AssetCache` container excludes assets of the type that are cached in a type-specific container, and a type-specific container excludes assets of all other types.

Note, also, that when asset caching is enabled, the Cache Management tool (on the System Tools node of the Admin tab) displays dialogs showing the contents of `AssetCache` containers, corresponding entries in the `dependencyRepository`, and other types of information. Searches are also supported.

For more information about configuring asset caching, see "Customizing Asset Caching," on page 483. For more information about the Cache Management tool, see Chapter 28, "System Tools."

Customizing Asset Caching

The steps below show you how to configure type-specific asset caches.

To configure asset caching

1. When *default* asset caching is enabled on a WebCenter Sites node, you will see the following entry in the `cs-cache.xml` file on WebCenter Sites nodes, which specifies that assets of all types will be cached in a generic container named `AssetCache` (parameters, other than `name`, are defined in Table 8, on page 470):

```
<cache name="AssetCache"
   diskPersistent="true" maxElementsInMemory="100"
   maxElementsOnDisk="100" eternal="true"
   overflowToDisk="true" diskSpoolBufferSizeMB="20"
   memoryStoreEvictionPolicy="LFU" clearOnFlush="false" />
```

> **Note**
>
> Given that the generic `AssetCache` container can either co-exist or be replaced with one or many type-specific asset caches, you have the option to do any of the following:
>
> - You can keep the generic `AssetCache` container and add as many type-specific containers as necessary. Assets of all types will be cached. Assets having their own type-specific container will be cached *exclusively* in their own container, which can be individually tuned by use of the parameters shown in Table 8, on page 470.
> - You can replace the generic `AssetCache` container with one or more type-specific containers. Only assets of the named types will be cached.
>
> How many `AssetCache` containers to create depends on which types of assets must be cached in their own containers and which tunings must be applied. Configuration information is available in the steps below.

2. To cache assets of a given type in their own container, create a `<cache>` element for that asset type and set `name="AssetCache`*`Asset_type_name`*`"` (only one asset type is supported for each custom `<cache>` element). Configure the cache by using the parameters in Table 8, on page 470.

For example:

```
<cache name="AssetCacheContent_C"
   diskPersistent="true" maxElementsInMemory="100"
   maxElementsOnDisk="100" eternal="true"
   overflowToDisk="true" diskSpoolBufferSizeMB="20"
   memoryStoreEvictionPolicy="LFU" clearOnFlush="false" />
```

In this example, `name="AssetCacheContent_C"` creates a type-specific `AssetCache` container that holds only assets of type `Content_C`. If the generic `AssetCache` container is also configured, it will not cache assets of type `Content_C`.

Disabling Asset Caching

If your inCache framework is set up to support asset caching and you wish to disable asset caching, do the following:

On each WebCenter Sites node in the inCache framework, open the `cs-cache.xml` file (in the `WEB-INF/classes/` directory) and remove the `<cache>` elements whose name begins with `AssetCache`. You can remove selected `<cache>` elements or all `<cache>` elements. Each element represents an `AssetCache` container. For information about the types of `AssetCache` containers, see "Types of Asset Caching," on page 482.

Cache Management Tool

When asset caching is configured, the Cache Management tool displays dialogs showing the contents of `AssetCache` containers and corresponding entries in the `dependencyRepository`, among other types of information. Searches are also supported. For more information about the Cache Management tool, see Chapter 28, "System Tools."

Part 7

Diagnostic Utilities

This part describes the system tools that enable general administrators to troubleshoot problems directly from the WebCenter Sites administrator's interface. Features include configuring log4j logging dynamically, accessing various types of system information, managing caches, searching logs, and testing the performance of the shared files system.

This part contains the following chapter:

Chapter 28

System Tools

The **System Tools** node on the **Admin** tab provides a range of diagnostic utilities that general administrators can use to troubleshoot directly from the WebCenter Sites administrator's interface. Features include configuring log4j loggers, accessing various types of system information, managing caches, searching the contents of the WebCenter Sites log, and testing the performance of the shared file system.

This chapter contains the following sections, each covering a sub-node of **System Tools**:

- Configure log4j
- System Information
- Cache Management
- Log Viewer
- File System Test
- Resultset Cache

Configure log4j

If WebCenter Sites is set up to use the Apache log4j logging system, the **Configure log4j** tool enables you to view current logger levels, change logger levels, and add new loggers directly from the administrator's interface. In addition, since changes to logger levels and added loggers remain in effect only until WebCenter Sites is restarted, you can retain your changes by copying a test version of the logger properties to the `log4j.properties` file.

This section contains the following topics:

Setting Up Apache log4j

If WebCenter Sites was not set up to use log4j during the upgrade process, you can manually switch to log4j by configuring the `commons-logging.properties` and `log4j.properties` files in the `WEB-INF/classes` directory.

To set up log4j

1. In the `commons-logging.properties` file, update the following property to set log4j as WebCenter Sites' logging system:

```
org.apache.commons.logging.Log=
   org.apache.commons.logging.impl.Log4JLogger
```

2. Open the `log4j.properties` file and do the following:

   a. Add the `FWDefaultAppender` file appender properties:

```
log4j.rootLogger=INFO, FWDefaultAppender
log4j.appender.FWDefaultAppender.File=
   <path to futuretense.txt>/futuretense.txt
```

> **Note**
>
> The `futuretense.txt` file is typically located in the WebCenter Sites installation directory.

```
log4j.appender.FWDefaultAppender.Append=true
log4j.appender.FWDefaultAppender.BufferSize=256
log4j.appender.FWDefaultAppender.MaxFileSize=10MB
log4j.appender.FWDefaultAppender.encoding=UTF-8
log4j.appender.FWDefaultAppender.layout.ConversionPattern
   =[%d] [%c{4}] %m%n
log4j.appender.FWDefaultAppender=org.apache.log4j.Rolling
   FileAppender
log4j.appender.FWDefaultAppender.MaxBackupIndex=15
```

```
log4j.appender.FWDefaultAppender.layout=org.apache.log4j
   .PatternLayout
log4j.appender.FWDefaultAppender.bufferedIO=false
```

b. Copy the loggers in the `commons-logging.properties` file to the `log4j.properties` file. Prepend `log4j.logger.` to each logger name. For example: `com.fatwire.logging.cs.auth` becomes `log4j.logger.com.fatwire.logging.cs.auth`.

For information about the file appender properties and other log4j-specific properties, refer to the Apache log4j documentation. For information about WebCenter Sites loggers, see the `commons.logging-properties` documentation in the *Oracle WebCenter Sites Property Files Reference*.

Viewing Loggers

You can view a list of WebCenter Sites loggers and their verbosity levels. The list includes the following:

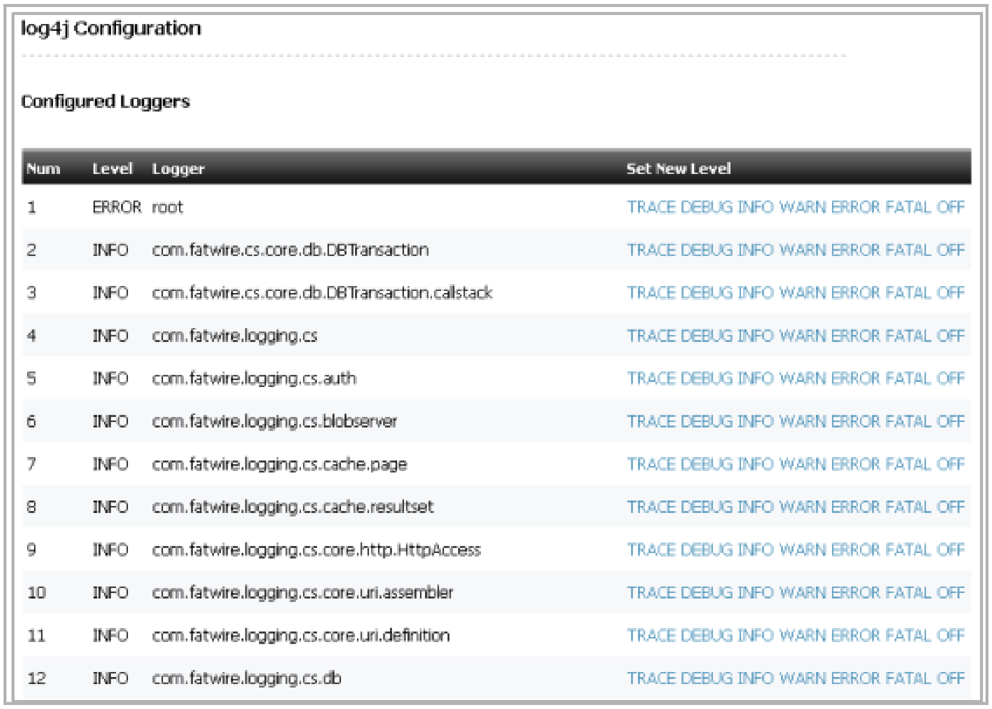
- Loggers that are registered in the `log4j.properties` file, in the `WEB-INF/classes` directory. The file is read only on system startup. If, during runtime, you make changes to the `log4j.properties` file and save them, your changes will not be displayed in the "log4j Configuration" screen. They will be applied and displayed when WebCenter Sites is restarted.
- Loggers that are detected during runtime. The loggers are detected when:
  - System features or add-on components (such as Oracle WebCenter Sites: Engage) are accessed for the first time.
  - The loggers are added to the system manually by developers using the **Add Logger** feature, as shown in "Adding New Loggers," on page 492.

The collapsed view of the list shows the root logger and parent loggers, all of which are explicitly assigned verbosity levels. If you require a more granular view, you can expand the list to also show loggers that inherit levels from the root logger or parent loggers.

To view loggers

1. On the **Admin** tab, expand **System Tools** and double-click **Configure log4j**.

The list of loggers with explicitly assigned levels is displayed.



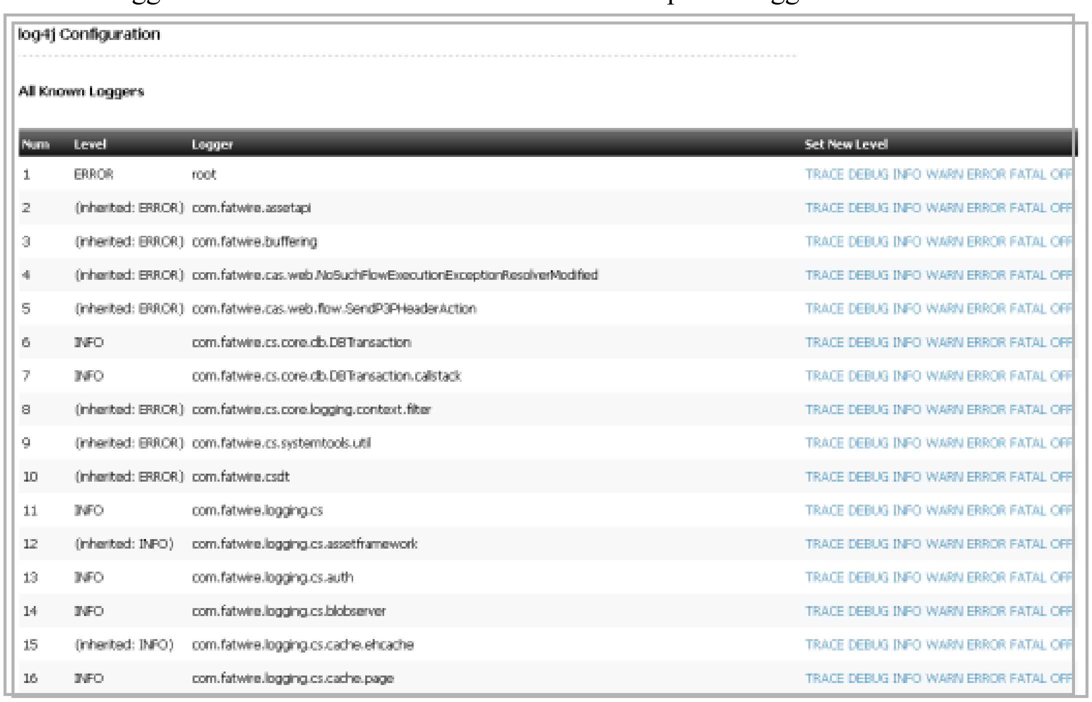
log4j Configuration

Configured Loggers

Num	Level	Logger	Set New Level
1	ERROR	root	TRACE DEBUG INFO WARN ERROR FATAL OFF
2	INFO	com.fatwire.cs.core.db.DBTransaction	TRACE DEBUG INFO WARN ERROR FATAL OFF
3	INFO	com.fatwire.cs.core.db.DBTransaction.callstack	TRACE DEBUG INFO WARN ERROR FATAL OFF
4	INFO	com.fatwire.logging.cs	TRACE DEBUG INFO WARN ERROR FATAL OFF
5	INFO	com.fatwire.logging.cs.auth	TRACE DEBUG INFO WARN ERROR FATAL OFF
6	INFO	com.fatwire.logging.cs.blobserver	TRACE DEBUG INFO WARN ERROR FATAL OFF
7	INFO	com.fatwire.logging.cs.cache.page	TRACE DEBUG INFO WARN ERROR FATAL OFF
8	INFO	com.fatwire.logging.cs.cache.resultset	TRACE DEBUG INFO WARN ERROR FATAL OFF
9	INFO	com.fatwire.logging.cs.core.http.HttpAccess	TRACE DEBUG INFO WARN ERROR FATAL OFF
10	INFO	com.fatwire.logging.cs.core.uri.assembler	TRACE DEBUG INFO WARN ERROR FATAL OFF
11	INFO	com.fatwire.logging.cs.core.uri.definition	TRACE DEBUG INFO WARN ERROR FATAL OFF
12	INFO	com.fatwire.logging.cs.db	TRACE DEBUG INFO WARN ERROR FATAL OFF

2. To expand the list to include loggers that inherit levels, click **Show All Known Loggers** at the bottom of the screen.

   Loggers that inherit levels are listed below their parent loggers.



log4j Configuration

All Known Loggers

Num	Level	Logger	Set New Level
1	ERROR	root	TRACE DEBUG INFO WARN ERROR FATAL OFF
2	(inherited: ERROR)	com.fatwire.assetapi	TRACE DEBUG INFO WARN ERROR FATAL OFF
3	(inherited: ERROR)	com.fatwire.buffering	TRACE DEBUG INFO WARN ERROR FATAL OFF
4	(inherited: ERROR)	com.fatwire.cas.web.NoSuchFlowExecutionExceptionResolverModified	TRACE DEBUG INFO WARN ERROR FATAL OFF
5	(inherited: ERROR)	com.fatwire.cas.web.flow.SendP3PHeaderAction	TRACE DEBUG INFO WARN ERROR FATAL OFF
6	INFO	com.fatwire.cs.core.db.DBTransaction	TRACE DEBUG INFO WARN ERROR FATAL OFF
7	INFO	com.fatwire.cs.core.db.DBTransaction.callstack	TRACE DEBUG INFO WARN ERROR FATAL OFF
8	(inherited: ERROR)	com.fatwire.cs.core.logging.context.filter	TRACE DEBUG INFO WARN ERROR FATAL OFF
9	(inherited: ERROR)	com.fatwire.cs.systemtools.util	TRACE DEBUG INFO WARN ERROR FATAL OFF
10	(inherited: ERROR)	com.fatwire.csdt	TRACE DEBUG INFO WARN ERROR FATAL OFF
11	INFO	com.fatwire.logging.cs	TRACE DEBUG INFO WARN ERROR FATAL OFF
12	(inherited: INFO)	com.fatwire.logging.cs.assetframework	TRACE DEBUG INFO WARN ERROR FATAL OFF
13	INFO	com.fatwire.logging.cs.auth	TRACE DEBUG INFO WARN ERROR FATAL OFF
14	INFO	com.fatwire.logging.cs.blobserver	TRACE DEBUG INFO WARN ERROR FATAL OFF
15	(inherited: INFO)	com.fatwire.logging.cs.cache.ehcache	TRACE DEBUG INFO WARN ERROR FATAL OFF
16	INFO	com.fatwire.logging.cs.cache.page	TRACE DEBUG INFO WARN ERROR FATAL OFF

3. To view only loggers with explicitly assigned levels once again, click **Show Configured Loggers** at the bottom of the screen.

Changing Logger Levels

You can temporarily change logger levels to define the type and number of statements that get written to the WebCenter Sites log specified in the `log4j.properties` file. Your changes are applied right away; as soon as the loggers are executed, you can track the impact of their new levels in the log. The new levels remain in effect only until WebCenter Sites is restarted, but if you wish to retain the changes, you can copy a text version of the logger properties to the `log4j.properties` file in the `WEB-INF/classes` directory on your application server.

This option is particularly useful when you want to temporarily override inherited logger levels. For example, if a parent logger is set to a high-verbosity level such as TRACE and you want to prevent one of its child loggers from inheriting that level, you can assign the child logger a different level.

The following logger levels are available, listed in order of decreasing verbosity:

TRACE, DEBUG, INFO, WARN, ERROR, FATAL, OFF

OFF indicates that a logger does not write any statements to the log. For more information about logger levels, refer to the Apache log4j documentation.

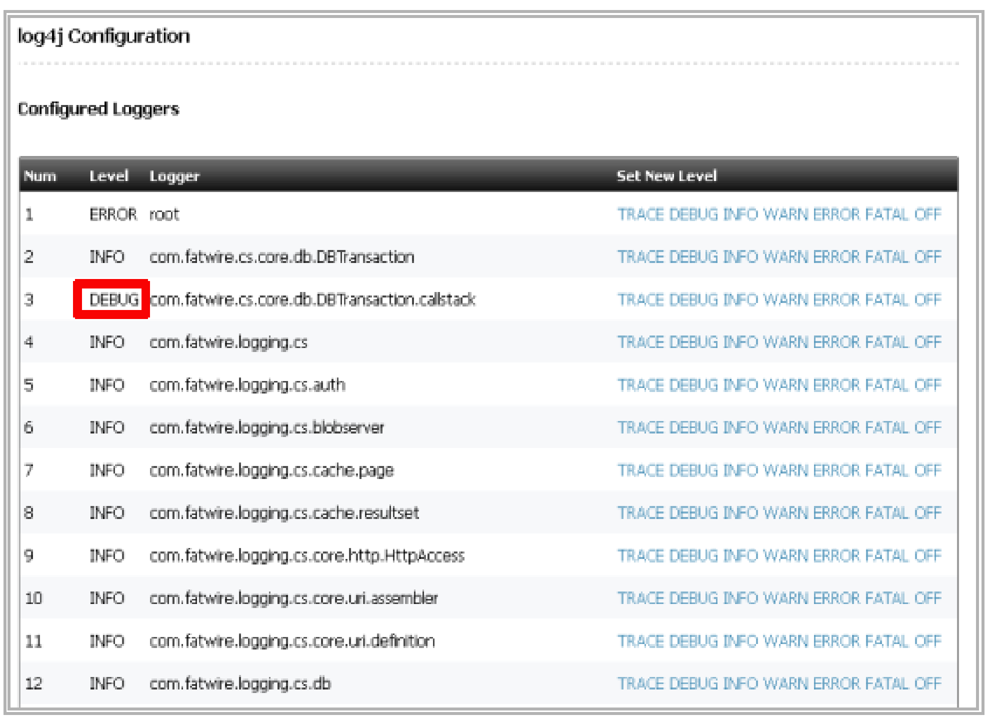
To change a logger level

1. On the **Admin** tab, expand **System Tools** and double-click **Configure log4j**.

   The list of loggers with explicitly assigned levels is displayed. If you require a more granular view, you can click **Show All Known Loggers** at the bottom of the screen to expand the list to include loggers that inherit levels from parent loggers.

2. Find the logger that you want to change and select a new logger level in the **Set New Level** column.

   The new level is displayed in the "Level" column. As soon as the logger is executed, you can track the impact of the new level in the log (see "Log Viewer," on page 506).

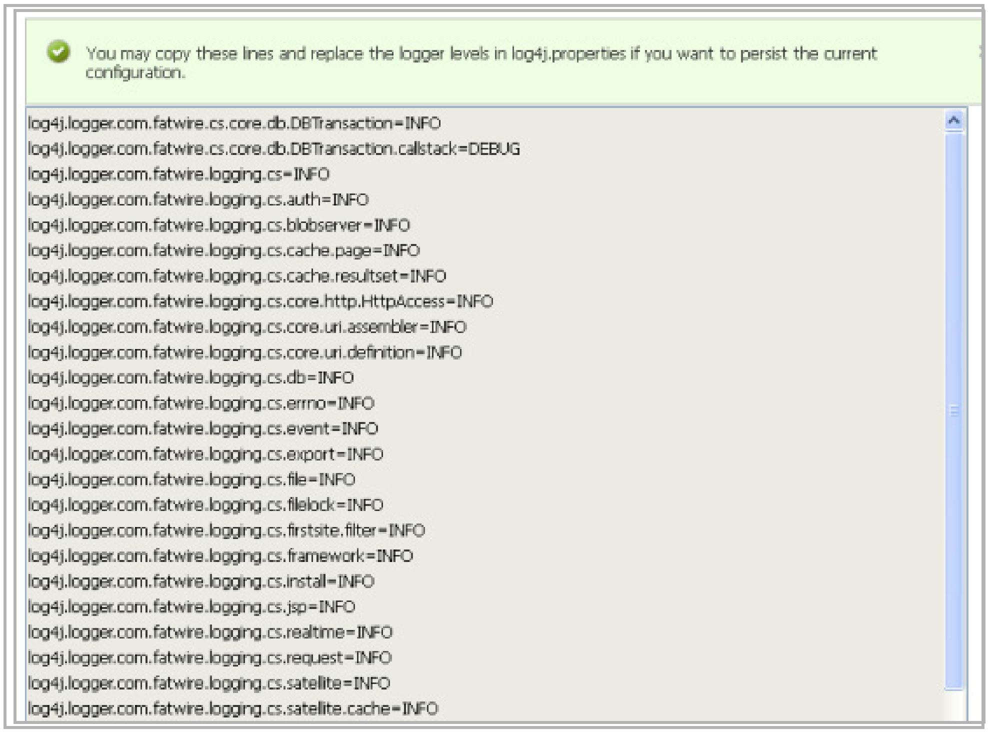

log4j Configuration

Configured Loggers

Num	Level	Logger	Set New Level
1	ERROR	root	TRACE DEBUG INFO WARN ERROR FATAL OFF
2	INFO	com.fatwire.cs.core.db.DBTransaction	TRACE DEBUG INFO WARN ERROR FATAL OFF
3	DEBUG	com.fatwire.cs.core.db.DBTransaction.callstack	TRACE DEBUG INFO WARN ERROR FATAL OFF
4	INFO	com.fatwire.logging.cs	TRACE DEBUG INFO WARN ERROR FATAL OFF
5	INFO	com.fatwire.logging.cs.auth	TRACE DEBUG INFO WARN ERROR FATAL OFF
6	INFO	com.fatwire.logging.cs.blobserver	TRACE DEBUG INFO WARN ERROR FATAL OFF
7	INFO	com.fatwire.logging.cs.cache.page	TRACE DEBUG INFO WARN ERROR FATAL OFF
8	INFO	com.fatwire.logging.cs.cache.resultset	TRACE DEBUG INFO WARN ERROR FATAL OFF
9	INFO	com.fatwire.logging.cs.core.http.HttpAccess	TRACE DEBUG INFO WARN ERROR FATAL OFF
10	INFO	com.fatwire.logging.cs.core.uri.assembler	TRACE DEBUG INFO WARN ERROR FATAL OFF
11	INFO	com.fatwire.logging.cs.core.uri.definition	TRACE DEBUG INFO WARN ERROR FATAL OFF
12	INFO	com.fatwire.logging.cs.db	TRACE DEBUG INFO WARN ERROR FATAL OFF

3. If you want the new logger level to remain in effect after WebCenter Sites is restarted, do the following:
   a. Click **Show Current Logger Levels As .properties** at the bottom of the screen.

      The loggers with explicitly assigned levels are displayed as text.



   b. Copy the changed logger to the `log4j.properties` file in the `WEB-INF/classes` directory on your application server.
   c. Save the `log4j.properties` file.

Adding New Loggers

If you wish to add new loggers to the log4j framework in your WebCenter Sites system and you do not want them to inherit verbosity levels from the root logger or parent logger, you can add the loggers via the administrator's interface and assign each of them a verbosity level. (If you add your own custom loggers, be sure to deploy the supporting `class` and `jar` files.)

Assigned verbosity levels remain in effect only until WebCenter Sites is restarted, but if you wish to apply the levels permanently, you can copy a text version of the logger properties to the `log4j.properties` file in the `WEB-INF/classes` directory on your application server.

To add a new logger

1. On the **Admin** tab, expand **System Tools** and double-click **Configure log4j**.

   The list of loggers with explicitly assigned levels is displayed.
2. In the field below the list of loggers, enter the name of the logger you want to add and select its verbosity level.

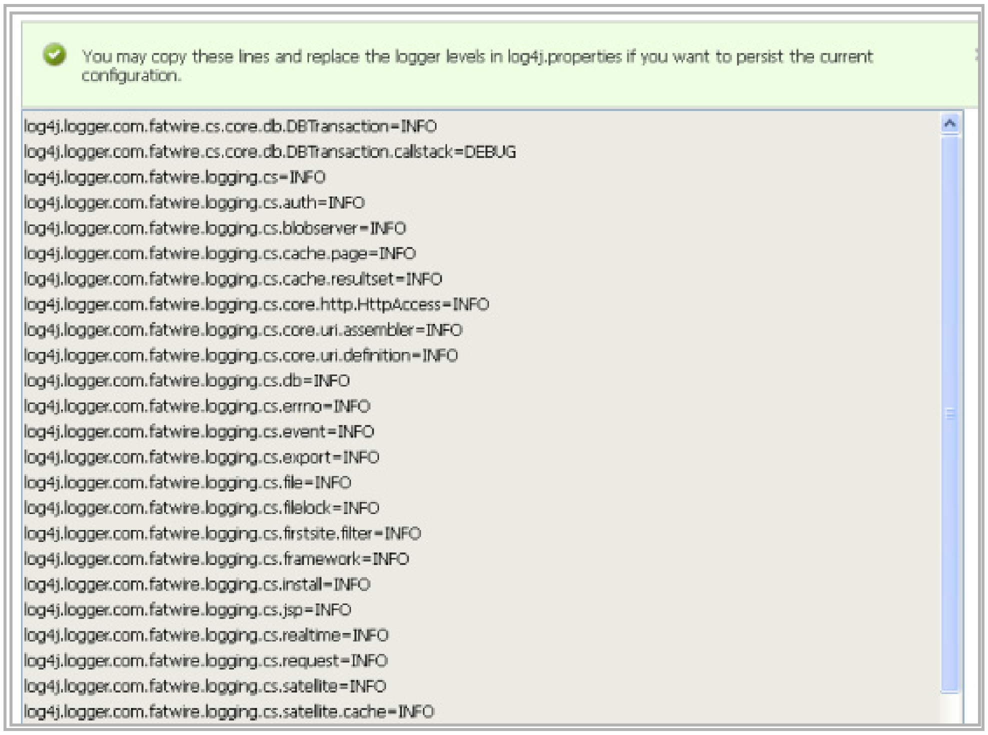
3. Click **Add Logger**.

   The new logger is added to the list. As soon as the logger is executed, you can track the impact of the assigned level in the log (see "Log Viewer," on page 506).

4. If you want the added logger to remain in effect after WebCenter Sites is restarted, do the following:

   a. Click **Show Current Logger Levels As .properties** at the bottom of the screen.

      The loggers with explicitly assigned levels are displayed as text.



   b. Copy the added logger to the `log4j.properties` file in the `WEB-INF/classes` directory on your application server.

   c. Save the `log4j.properties` file.

System Information

The **System Information** tool provides immediate access to various types of WebCenter Sites, database, and thread information for troubleshooting and checking the health of your application. You can download information to a file or view information directly in the administrator's interface.

This section contains the following topics:

- WebCenter Sites Information
- Database Information
- Thread Information

> **Note**
>
> You can download all available system information and a list of all WebCenter Sites directories and files on the application server by right-clicking the **System Information** tool and selecting **Download**. If Sigar is configured, you can also download Sigar information to send to Technical Support by right-clicking the tool and selecting **System Details (Sigar)**.

WebCenter Sites Information

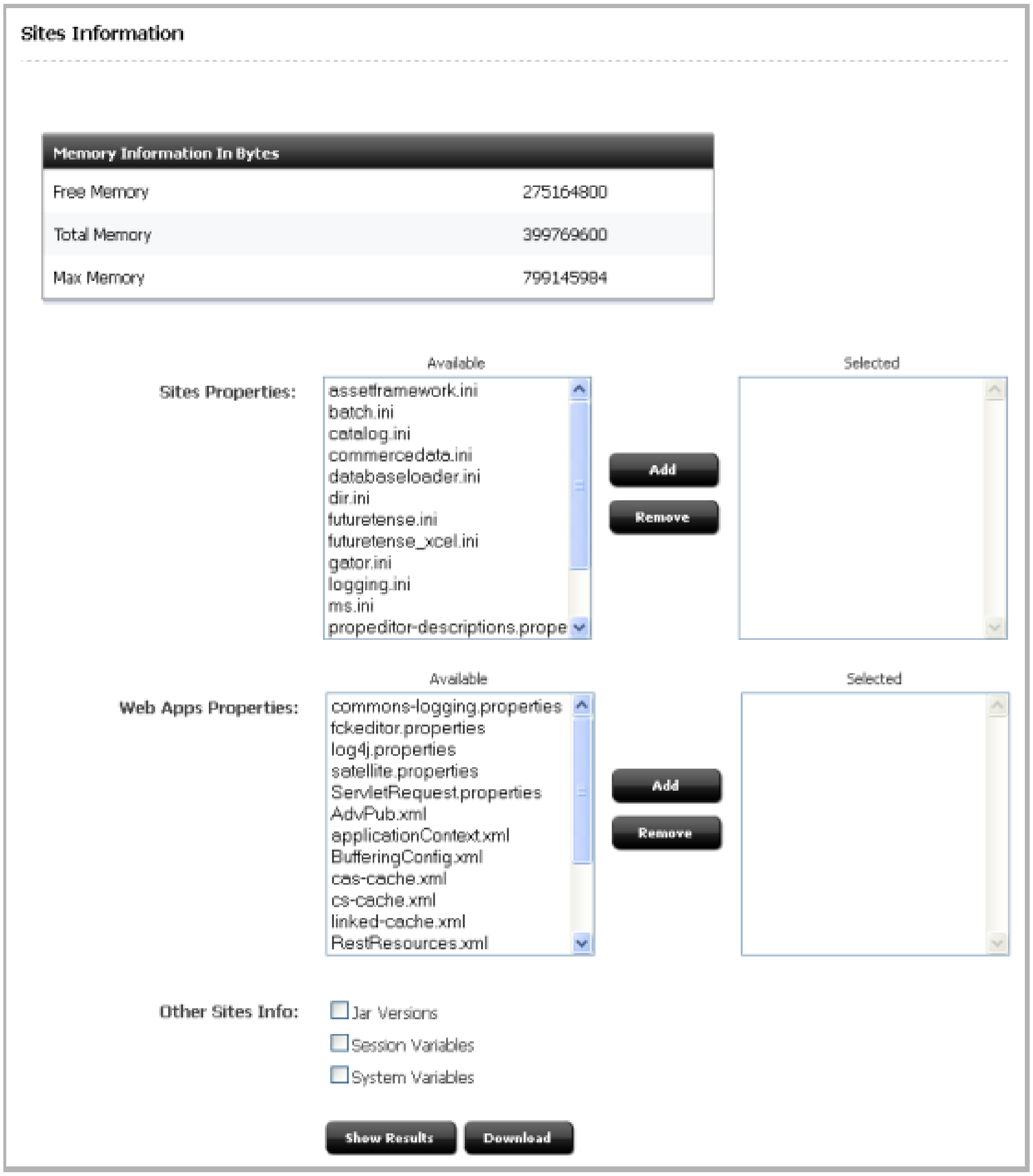
You can download or view the following types of WebCenter Sites information:

- **Memory Information** – The amount of free memory in the application server Java Virtual Machine, the total amount of memory free and in use, and the maximum amount of memory that the Java Virtual Machine will attempt to use.
- **Sites Properties** – The `ini`, `xml`, and `properties` files in the WebCenter Sites installation directory.
- **Web Apps Properties** – The `xml` and `properties` files in the `WEB-INF` and `WEB-INF/classes` directories on the application server.
- **Other Sites Info** – The `jar` files in the `WEB-INF/lib` directory on the application server, the session variables for the user currently logged in to WebCenter Sites, and the Java run-time properties.

To download or view WebCenter Sites information

1. On the **Admin** tab, expand **System Tools** > **System Information** and double-click **Sites Info**.

   The "WebCenter Sites Information" screen shows memory information as well as additional information that you can select to download or view.



2. In the "Sites Properties" and "Web Apps Properties" sections, select the files that you are interested in and click **Add**.
3. In the "Other Sites Info" section, do the following:
   - To obtain information about the `jar` files in the `WEB-INF/lib` directory on the application server, select the **Jar Versions** check box.

- To obtain the session variables for the user currently logged in to WebCenter Sites, select the **Sites Session Variables** check box.
- To obtain Java run-time properties, select the **System Variables** check box.

4. To download the selected WebCenter Sites information as a text file, click **Download** and then open or save `SysInfo.zip`. The text file is inside `SysInfo.zip`.
5. To view the selected WebCenter Sites information in the administrator's interface, click **Show Results**.
   a. The selected files and information are displayed directly in the administrator's interface.
   b. To view the details of a file or type of information, click its "plus" (⊞) icon.

   For `ini` and `properties` files, as well as Sites session variables and system variables, the properties and values are displayed.

⊟ batch.ini

Property	Value
debug	false
hidden.encrypted	true
request.folder	./dispatcher/
security.class	com.openmarket.Batch.DefaultSecurity
thread.count	32
thread.growcache	false
thread.idle	10
thread.wait	15

For `xml` files, the code is displayed.

⊟ applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
                        "http://www.springframework.org/dtd/spring-beans.dtd">
<beans/>
```

For Jar Versions, the `jar` names, versions, and implementation vendors are displayed. “NA : NA” indicates that a `jar` file does not provide its version and vendor.

Jar Versions

Jar Name	Version : Vendor
jersey-json-1.1.4.1.jar	1.1.4.1 : Sun Microsystems, Inc
analyticscs.jar	Build Date: Oct 13 2011 at 17:24:19 Analytics-bld- 10 Revision:140069 : FatWire Corporation
commons-io-1.4.jar	1.4 : The Apache Software Foundation
core-renderer-R8pre2.jar	NA : NA
httpcore-4.1.2.jar	4.1.2 : The Apache Software Foundation

Database Information

You can download or view the following types of database information:

- **Basic Information** – The database engine and version, default transaction isolation level, JDBC driver, URL, and version, and JNDI data source.
- **System Tables** – Tables that store core WebCenter Sites data. The `ElementCatalog` table, for example, lists all of the XML or JSP elements used in your system.
- **Tree Tables** – Tables that store parent-child relationships between objects. The `AssetRelationTree`, for example, stores information about the associations between assets.
- **Sites Object Tables** – Tables that store uniquely identified objects, such as asset types, asset attributes, and asset definitions.
- **Other Sites Tables** – Tables that store flat data that cannot be organized into parent-child relationships and the foreign tables that WebCenter Sites does not manage.

Note

You have permission to access a database table if at least one of the Access Control Lists (ACLs) assigned to the table is assigned to your user. For more information about database permissions, see “ACLs,” on page 70. For more information about the different types of WebCenter Sites database tables, see the *Oracle WebCenter Sites Developer’s Guide*.

To download or view database information

1. On the **Admin** tab, expand **System Tools** > **System Information** and double-click **DB Info**.

   The "Database Information" screen shows basic information about your database as well as the tables that you can select to download or view.

2. In the "System Tables," "Tree Tables," "Sites Object Tables," and "Other Sites Tables" sections, select the tables that you want to view and click **Add**.
3. To download the selected database information as a spreadsheet file, click **Download** and then open or save `SysInfo.zip`. The spreadsheet file is inside `SysInfo.zip`.
4. To view the selected database information in the administrator's interface, click **Show Results**.

   a. For each selected table that you have permission to access, the following information is displayed:

      - Table Name
      - Primary key
      - Number of rows
      - CSz (number of resultsets to cache in memory)
      - Timeout (number of minutes to keep a resultset cached in memory)
      - Abs (how to calculate the expiration time for resultsets)
      - Time to run SQL
      - Special table key

b. To view a table's column and indices, click its "plus" (⊞) icon. If an index is listed more than once, it extends across multiple columns.

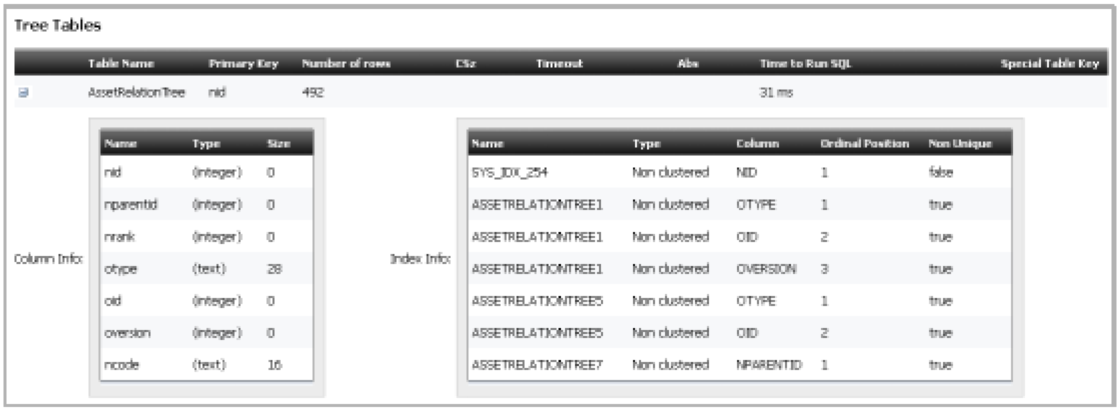

Tree Tables

	Table Name	Primary Key	Number of rows	CSz	Timeout	Abs	Time to Run SQL	Special Table Key
⊟	AssetRelationTree	nid	492				31 ms	

Column Info:

Name	Type	Size
nid	(integer)	0
nparentid	(integer)	0
nrank	(integer)	0
otype	(text)	28
oid	(integer)	0
oversion	(integer)	0
ncode	(text)	16

Index Info:

Name	Type	Column	Ordinal Position	Non Unique
SYS_IDX_254	Non clustered	NID	1	false
ASSETRELATIONTREE1	Non clustered	OTYPE	1	true
ASSETRELATIONTREE1	Non clustered	OID	2	true
ASSETRELATIONTREE1	Non clustered	OVERSION	3	true
ASSETRELATIONTREE5	Non clustered	OTYPE	1	true
ASSETRELATIONTREE5	Non clustered	OID	2	true
ASSETRELATIONTREE7	Non clustered	NPARENTID	1	true

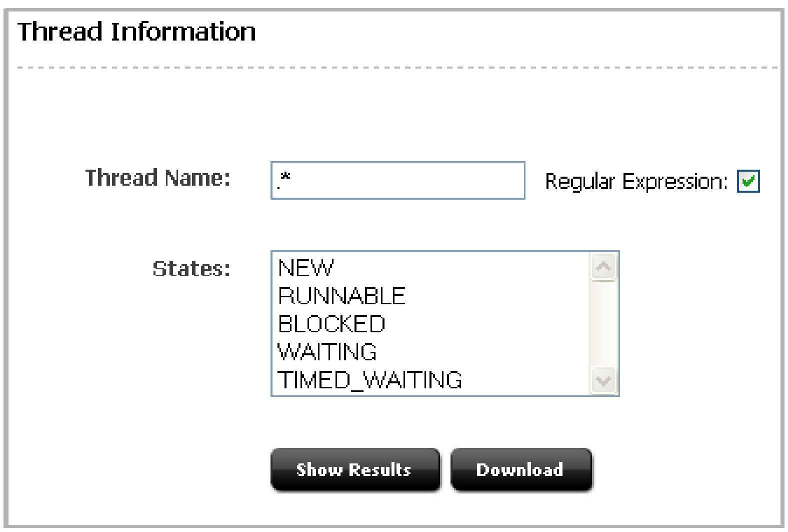
Thread Information

You can download or view thread dumps to analyze the runtime state of the application server. This can be especially useful when trying to detect problems that might result in resource starvation or thread hangs.

To download or view thread information

1. On the **Admin** tab, expand **System Tools** > **System Information** and double-click **Thread Info**.

   The "Thread Information" screen is displayed.



2. The **Regular Expression** check box is selected by default. This option enables you to use Java regular expressions to search for thread names.

   Examples:

   - **Thread.*** to find `Thread-0`, `Thread-1`, and `Thread-3`
   - **.*thread.*** to find `pool-6-thread-1`, `pool-8-thread-1`, and `pool-14-thread-2`

If you deselect the check box, you must search for exact thread names.

3. In the “Thread Name” field, enter the thread name or text string that you want to search for.
4. If you want to limit your search to one or more thread states, select the state(s) in the “States” field. If you do not select a state, all states will be searched.
5. To download the selected thread information as a text file for use with a thread dump analyzer, click **Download** and then open or save `SysInfo.zip`. The text file is inside `SysInfo.zip`.

Note

Downloading thread information is possible only in the following cases:

- WebCenter Sites’ JDK includes a compatible `jstack` utility.
- The first JDK listed in your operating system’s library path includes a `jstack` utility that is compatible with WebCenter Sites’ JDK.

If a compatible `jstack` is not found, the following message displays: “Thread dump feature is not supported for this platform.”

6. To view the selected thread information in the administrator’s interface, click **Show Results**. For each thread matching the search, the following information is displayed:
   - Name
   - ID
   - State
   - Total time the CPU executed the thread
   - Total time the thread was in user mode
   - Number of times it was blocked and waited
   - How long it was blocked and waited
   - Object for which it is waiting
   - Strack trace

Thread Dump

```
Full Thread Dump at Tue Feb 08 10:02:38 EST 2011
"Timer-0" id=39 in TIMED_WAITING on lock=java.util.TaskQueue@fd9d27
 total cpu time=0.00 ms, user time=0.00 ms
 blocked count=1, blocked time=0.00 ms, wait count=23, wait time=0.91 ms
    at java.lang.Object.wait(Native Method)
    at java.util.TimerThread.mainLoop(Timer.java:509)
    at java.util.TimerThread.run(Timer.java:462)

"ScannerThread" id=42 in TIMED_WAITING
 total cpu time=0.00 ms, user time=0.00 ms
 blocked count=49, blocked time=0.00 ms, wait count=252, wait time=0.96 ms
    at java.lang.Thread.sleep(Native Method)
    at org.jboss.deployment.scanner.AbstractDeploymentScanner$ScannerThread.loop
(AbstractDeploymentScanner.java:280)
    at org.jboss.deployment.scanner.AbstractDeploymentScanner$ScannerThread.run
(AbstractDeploymentScanner.java:225)
```

Cache Management

If the inCache framework is set up for page caching and/or asset caching, the **Cache Management** tool enables you to monitor the caching processes on WebCenter Sites cluster members and their co-resident Satellite Servers. Features include cache searching, force writing to disk, and flushing individual pages, assets, dependencies, and entire caches. For information about setting up the inCache framework, see Chapter 26, "inCache for Page Caching for page caching" and Chapter 27, "inCache for Asset Caching."

Note

The **Cache Management** tool provides caching information only for WebCenter Sites cluster members and their co-resident Satellite Servers. To access cache information for remote Satellite Servers, use the following URL:

```
http://<server>:<port>/<application_context_path>/
   Inventory?username=SatelliteServer&password=password&
   detail=debug
```

The **Cache Management** tool includes the following screens:

- 'Summary' Screen
- 'Page' Screen
- 'Dependency' Screen
- 'Asset Cache' Screen
- 'Cluster Info' Screen

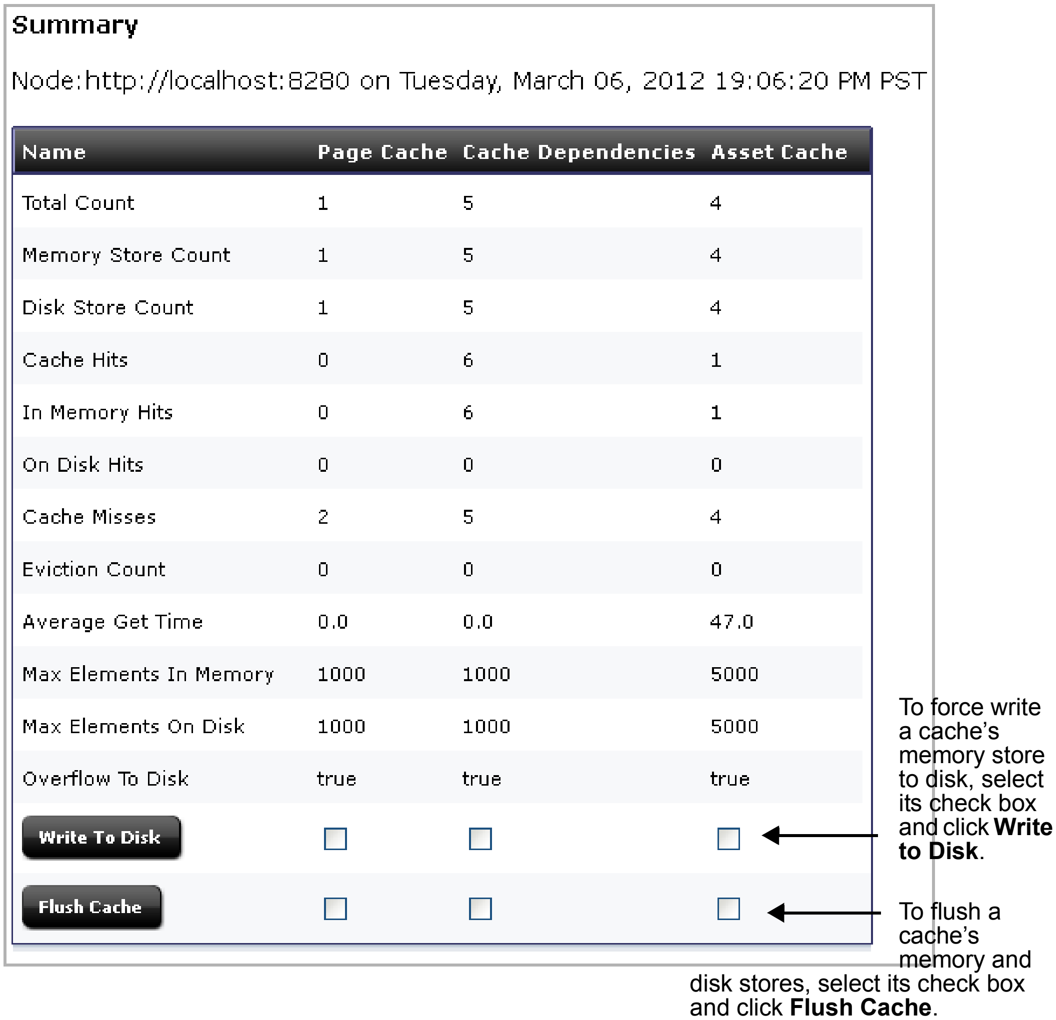
'Summary' Screen

This screen lists the following basic data for each cache (values are estimates):

- Total Count (total number of cached elements)
- Memory Store Count (number of cached elements in memory)
- Disk Store Count (number of cached elements on disk)
- Cache Hits (number of cached elements retrieved from both memory and disk)
- In Memory Hits (number of cached elements retrieved from memory)
- On Disk Hits (number of cached elements retrieved from disk)
- Cache Misses (number of times elements were not found in the cache)
- Eviction Count (number of elements evicted from the cache)
- Average Get Time (average number of milliseconds for retrieving an element)
- Max Elements In Memory (maximum number of elements allowed in memory)
- Max Elements On Disk (maximum number of elements allowed on disk)
- Overflow to Disk (whether elements are written to disk when memory is full)

This screen also provides the following options. **They should be used with caution**, especially on a delivery system, as they can impact the performance of your site.

- Writing a cache's entire memory store to disk – This option can be particularly useful if WebCenter Sites is not set up to automatically write a cache's memory store to disk before shutting down.
- Flushing a cache's entire memory and disk stores



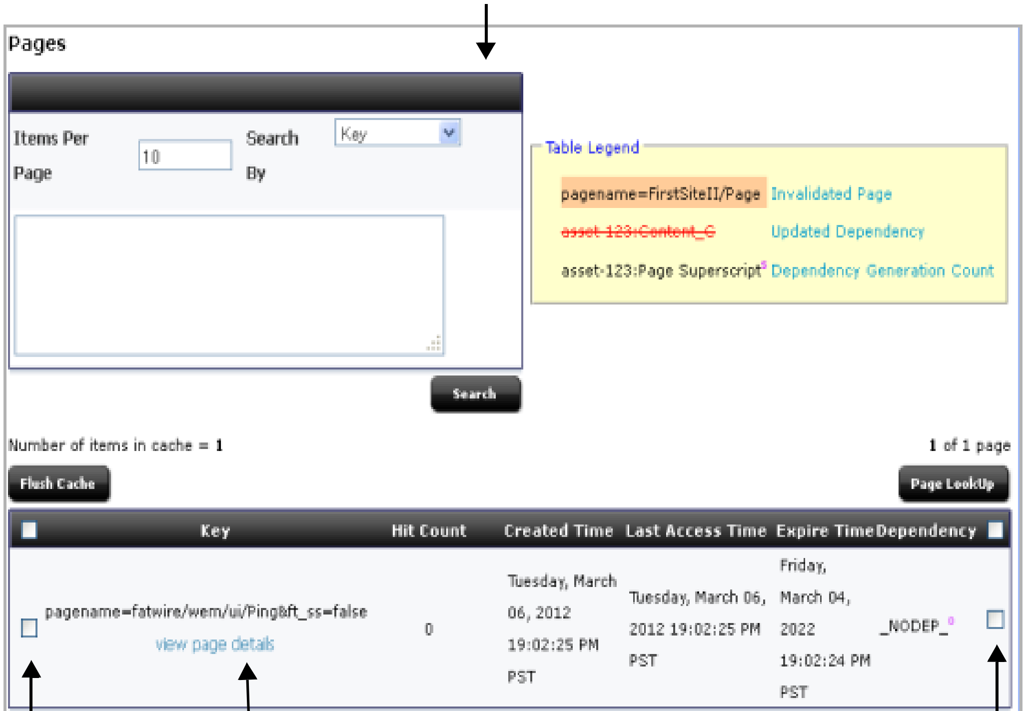
Name	Page Cache	Cache Dependencies	Asset Cache
Total Count	1	5	4
Memory Store Count	1	5	4
Disk Store Count	1	5	4
Cache Hits	0	6	1
In Memory Hits	0	6	1
On Disk Hits	0	0	0
Cache Misses	2	5	4
Eviction Count	0	0	0
Average Get Time	0.0	0.0	47.0
Max Elements In Memory	1000	1000	5000
Max Elements On Disk	1000	1000	5000
Overflow To Disk	true	true	true
Write To Disk	☐	☐	☐
Flush Cache	☐	☐	☐

‘Page’ Screen

This screen lists statistics for each cached page, such as hit count, time of creation/last access/expiration, and dependencies. The table legend identifies which types of information are tracked: invalidated pages, updates to dependencies, and changes to the dependency generation count. The screen also provides the following options:

- Searching the cache for specific pages – Searches are CPU intensive and should be used with caution on a delivery system.
- Flushing pages from the cache
- Viewing html information for pages
- Looking up page dependencies

Search for pages by key, expiration time, or dependency. If you search by expiration time, enter the information in “mm/dd/yyyy” or “mm/dd/yyyy hh:mm:ss” format.



To flush a page, select its check box and click **Flush Cache**.

To flush all pages on the screen, select the check box in the header and click **Flush Cache**.

To display a page’s html information, click **view page details**.

To view the pages for a specific dependency, select its check box and click **Page Lookup**.

To view all page dependencies on the screen, select the check box in the header and click **Page Lookup**.

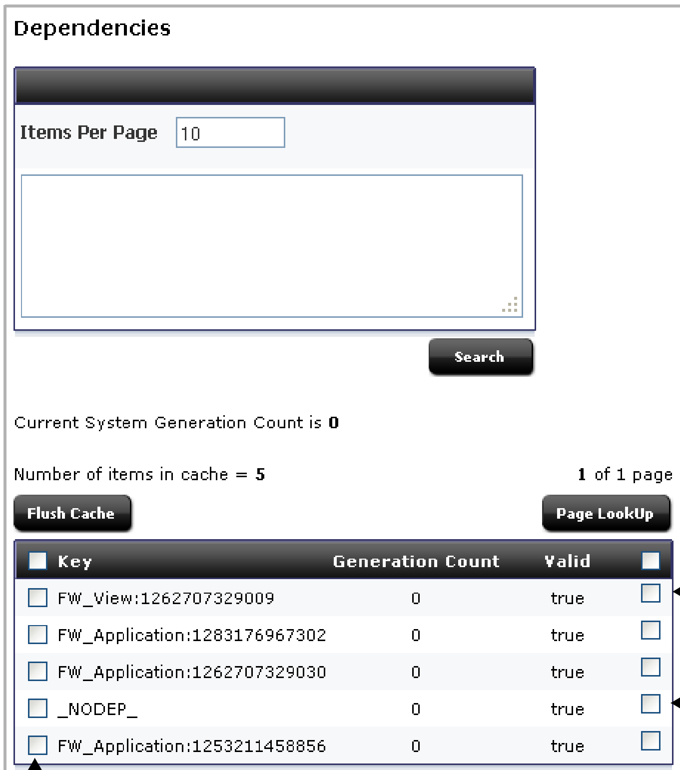
Note

After WebCenter Sites is restarted, this screen does not show page cache information until the cache engine is initialized. The cache engine is initialized when the first page or asset is requested.

‘Dependency’ Screen

This screen lists the contents of the dependency cache and the generation count for each dependency. In addition, it provides the following options:

- Searching the cache for specific dependencies – Searches are CPU intensive and should be used with caution on a delivery system.
- Flushing dependencies from the cache
- Looking up page dependencies



To view all page dependencies on the screen, select the check box in the header and click **Page Lookup**.

To view the pages for a specific dependency, select its check box and click **Page Lookup**.

To flush a dependency, select its check box and click **Flush Cache**.

To flush all dependencies on the screen, select the check box in the header and click **Flush Cache**.

Note

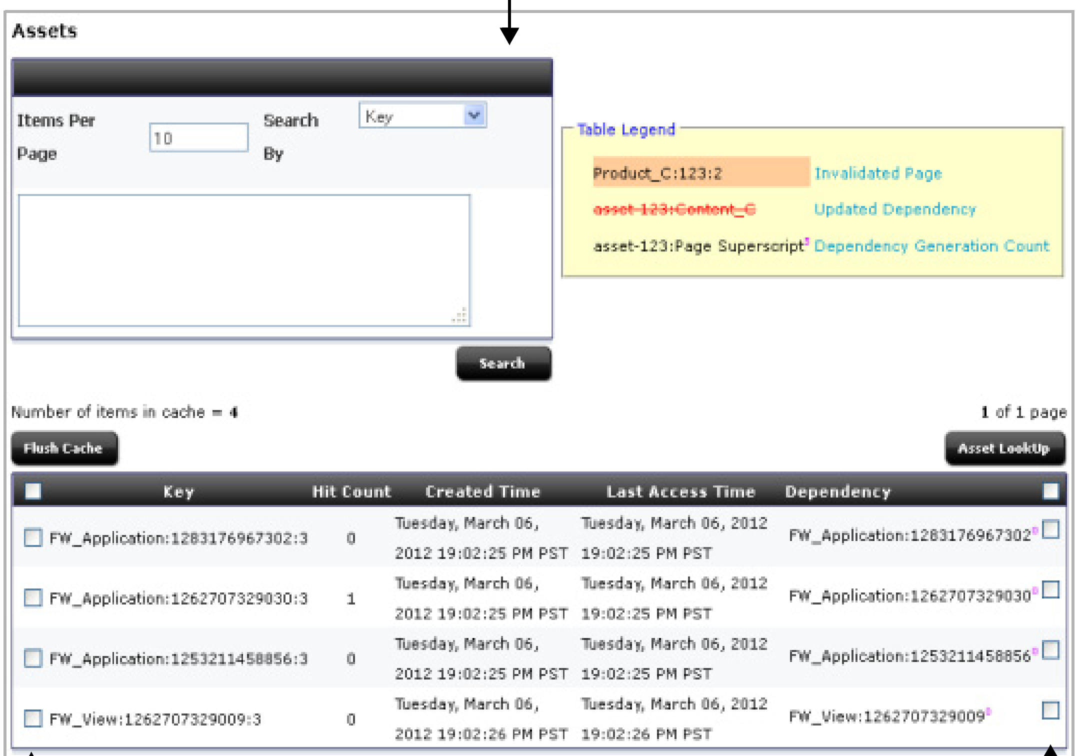
After WebCenter Sites is restarted, this screen does not show dependency cache information until the cache engine is initialized. The cache engine is initialized when the first page or asset is requested.

‘Asset Cache’ Screen

This screen lists statistics for each cached asset, such as hit count, time of creation/last access, and dependencies. The table legend identifies which types of information are tracked: invalidated assets, updates to dependencies, and changes to the dependency generation count. The screen also provides the following options:

- Searching the cache for specific assets – Searches are CPU intensive and should be used with caution on a delivery system.
- Flushing assets from the cache
- Looking up asset dependencies

Search for assets by key or dependency.



To flush an asset, select its check box and click **Flush Cache**.

To flush all assets on the screen, select the check box in the header and click **Flush Cache**.

To view the assets for a specific dependency, select its check box and click **Page Lookup**.

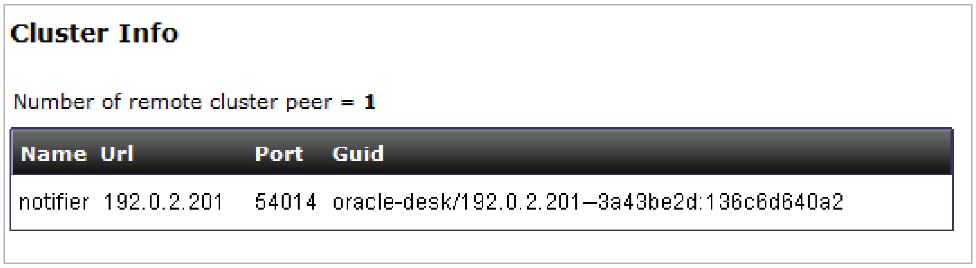
To view all asset dependencies on the screen, select the check box in the header and click **Page Lookup**.

Note

After WebCenter Sites is restarted, this screen does not show asset cache information until the cache engine is initialized. The cache engine is initialized when the first page or asset is requested.

'Cluster Info' Screen

If Sigar is configured, this screen lists the name, URL, port, and GUID of each additional WebCenter Sites cluster member.



Log Viewer

The **Log Viewer** tool enables you to view, tail, download, and search the contents of the WebCenter Sites log specified in the `log4j.properties` file directly from the WebCenter Sites administrator's interface. Since the total size of the log affects the tool's performance, if the log size exceeds 100 megabytes, you are informed and given the choice to proceed or cancel.

This section contains the following topics:

- Viewing the Log
- Tailing the Log
- Downloading the Log
- Searching the Log

Note

The **Log Viewer** tool is available only if WebCenter Sites is set up to use the Apache log4j logging system and the `log4j.properties` file includes the file appender `FWDefaultAppender`. For information about setting up log4j, see "Setting Up Apache log4j," on page 488.

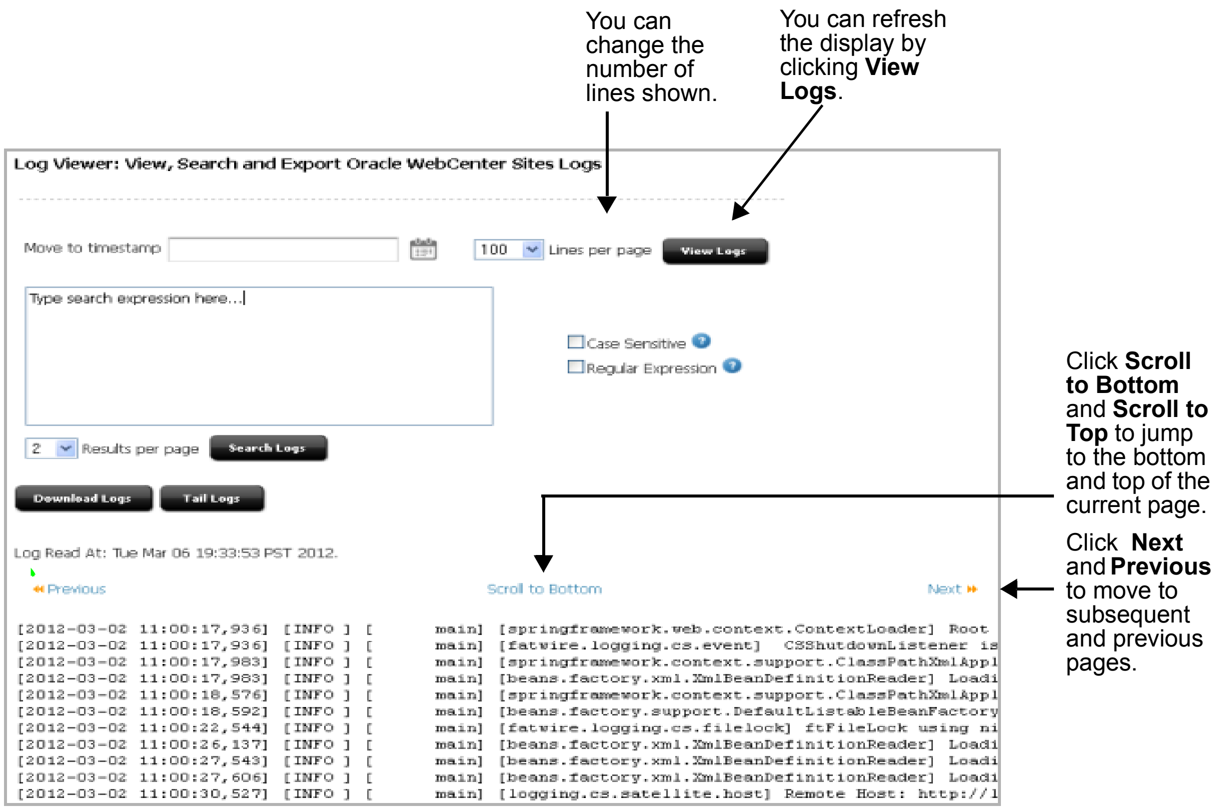
Viewing the Log

You can browse the entire contents of the WebCenter Sites log, starting with statements written most recently or written at a specific time.

To view the contents of the log

1. On the **Admin** tab, expand **System Tools** and double-click **Log Viewer**.

   The "Log Viewer" screen shows the most recent 100 lines in the log.



2. To jump to statements written at a specific time, do the following:

   a. Click the "calendar" ( ) icon and select a date and time, or enter the date and time manually in the "yyyy-mm-dd hh:mm:ss" format.

   > **Note**
   >
   > This feature takes into consideration only log statements that were written with the timestamping pattern currently specified in the `log4j.properties` file.

   b. Click **View Logs**.

   The "Log Viewer" screen shows the statements written at and/or after the specified date and time.

Tailing the Log

You can tail the WebCenter Sites log in a separate window. This option is particularly useful when you want to keep track of the statements written to the log while executing an action that causes an exception.

To tail the contents of the log

1. On the **Admin** tab, expand **System Tools** and double-click **Log Viewer**.
2. Click **Tail Logs**.

   The most recent 100 lines in the log are displayed in a new window and are refreshed every five seconds.

Downloading the Log

You can download a compressed file containing the entire contents of the WebCenter Sites log.

To download the contents of the log

1. On the **Admin** tab, expand **System Tools** and double-click **Log Viewer**.
2. Click **Download Logs**.
3. Open or save the compressed file containing the WebCenter Sites log. If the log is spread over several files, each file is included.

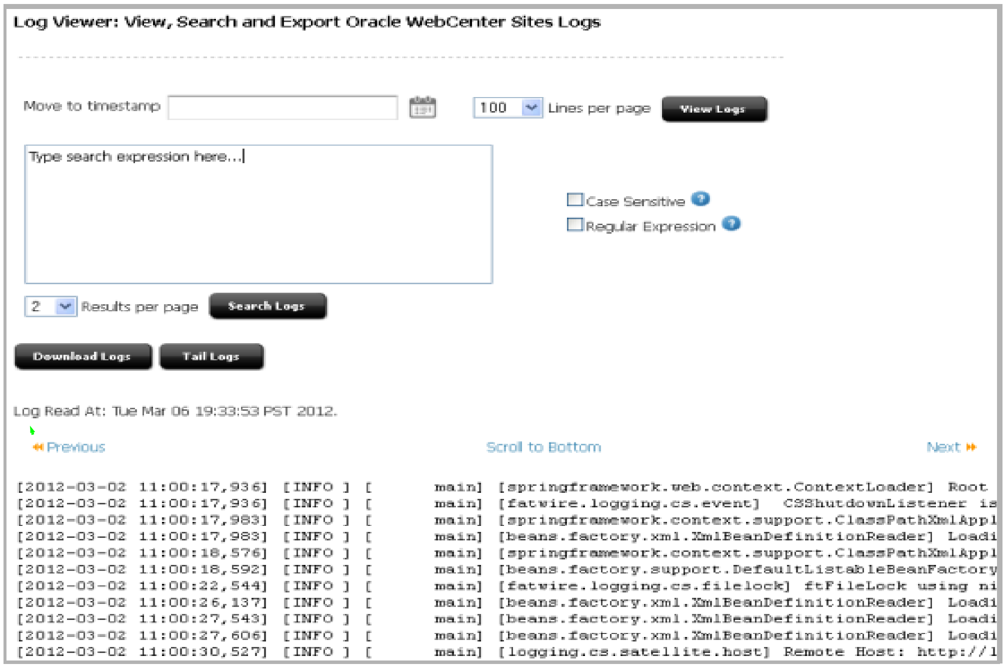
Searching the Log

You can search the contents of the WebCenter Sites log for specific information.

To search the contents of the log

1. On the **Admin** tab, expand **System Tools** and double-click **Log Viewer**.

The “Log Viewer” screen shows the most recent 100 lines in the log.



2. If you want to enable case-sensitive matching, select the **Case Sensitive** check box. By default, case-sensitivity covers only US-ASCII characters. If you want case-sensitivity to apply to Unicode characters, select the **Unicode Case** check box as well. The **Unicode Case** check box appears once the **Case Sensitive** check box is selected.

3. If you want the search to allow the usage of Java regular expressions, select the **Regular Expression** check box. Once the **Regular Expression** check box is selected, the **Dot All** and **Multi Line** check boxes appear. Use these to refine the regular expression settings:

- If you want a dot (.) to match any characters and line terminators, select the **Dot All** check box. If this check box is not selected, a dot matches only characters.
- If you want a caret (^) to match the beginning of a line and a dollar sign ($) to match the end of a line, select the **Multi Line** check box.

Note

If you search with regular expressions, a single search result is restricted to a maximum of 40 lines of the log.

4. In the search form, enter the expression that you want to find in the log.

5. In the “Results per page” field, select the number of results that you want to list on each page.

6. Click **Search Logs**.

The “Log Viewer” screen shows the lines in the log that match your search.

Click **Next** to move to the next page of the results.

7. To view a search result in its larger context, click **View log** below the result line.

   100 lines of the log, starting with the line matching the search, are displayed in a new window.

File System Test

The **File System Test** tool enables you to test the local application server directory and the central directories in the shared file system (`<cs_install_dir>/Shared`) in order to compare their performance with provided benchmarks.

Table 10: Tested File Systems

FileSystem	Description
spc	Shared page cache directory for all WebCenter Sites cluster members. Sample path: `\cs76\Shared\SystemPageCache\`
local	Temporary directory for servlet context information. Sample path: `\apache-tomcat-6.0.16\work\Catalina\localhost\cs`
data	Shared blob data directory for all WebCenter Sites cluster members. Sample path: `\cs76\Shared\ccurl`

Table 10: Tested File Systems

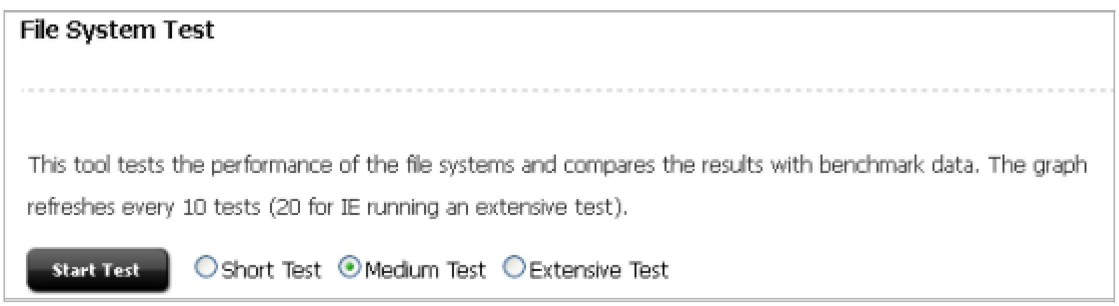
FileSystem	Description
sync	Shared synchronization directory for all WebCenter Sites cluster members. Sample path: `/cs76/Shared/clustersync`

A test suite is composed of a number of unique tests which are performed on the local application server directory and the shared file system directories. Each of these tests executes multiple threads. Each thread creates an empty file and then writes to, reads, and deletes the file. The following test suites are available:

- **Short Test** – runs a total of 16 tests (4 tests are performed on each of the directories listed in Table 10).
- **Medium Test** – runs a total of 64 tests (16 tests are performed on each of the directories listed in Table 10).
- **Extensive Test** – runs a total of 384 tests (96 tests are performed on each of the directories listed in Table 10).

To test the file system

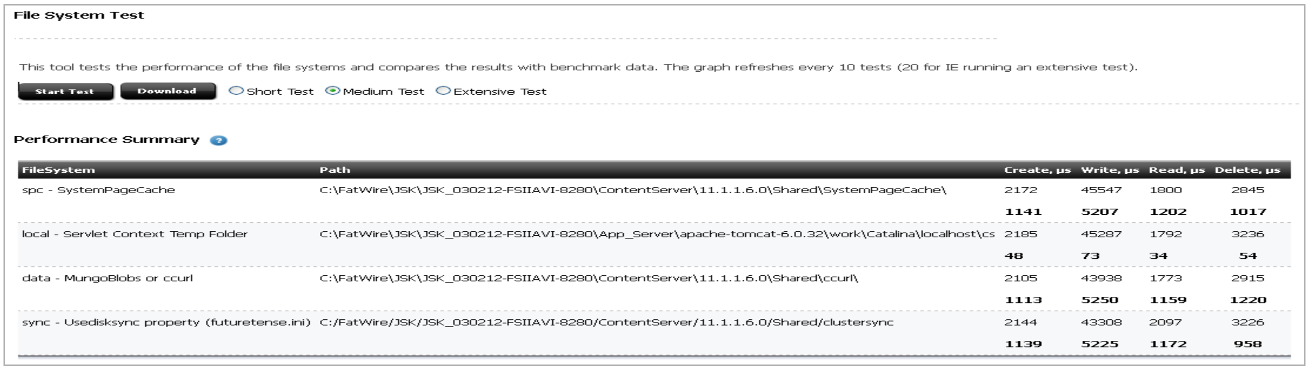
1. On the **Admin** tab, expand **System Tools** and double-click **File System Test**.
2. In the "File System Test" screen, select the type of test you want to perform and click **Start Test**.



The test is executed on the directories named in the "Performance Summary" table shown in step 3. You can follow the results of the file system test as it progresses.

> **Note**
>
> If the test takes too long or affects performance more than is acceptable, you can click **Stop Test**. The results of a file system test can be compared with the provided benchmark only when the test is completed.

3. Once the test is complete, review the overall results in the “Performance Summary” table.


File System Test

This tool tests the performance of the file systems and compares the results with benchmark data. The graph refreshes every 10 tests (20 for IE running an extensive test).

Start Test | Download | Short Test | Medium Test | Extensive Test

Performance Summary

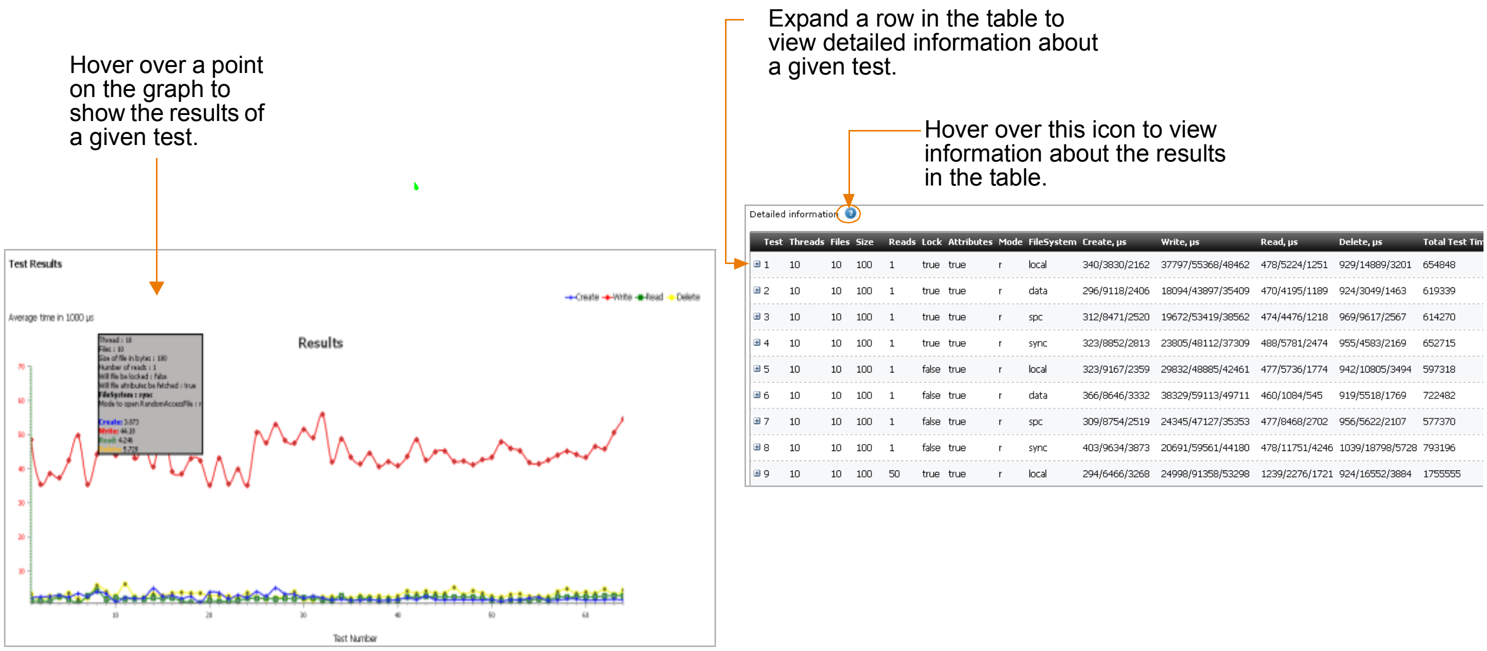
FileSystem	Path	Create, µs	Write, µs	Read, µs	Delete, µs
spc - SystemPageCache	C:\FatWire\JSK\JSK_030212-FSIIAVI-8280\ContentServer\11.1.1.6.0\Shared\SystemPageCache\	2172	45547	1800	2845
		1141	**5207**	**1202**	**1017**
local - Servlet Context Temp Folder	C:\FatWire\JSK\JSK_030212-FSIIAVI-8280\App_Server\apache-tomcat-6.0.32\work\Catalina\localhost\cs	2185	45287	1792	3236
		48	**73**	**34**	**54**
data - MungoBlobs or ccurl	C:\FatWire\JSK\JSK_030212-FSIIAVI-8280\ContentServer\11.1.1.6.0\Shared\ccurl\	2105	43938	1773	2915
		1113	**5250**	**1159**	**1220**
sync - Usedisksync property (futuretense.ini)	C:/FatWire/JSK/JSK_030212-FSIIAVI-8280/ContentServer/11.1.1.6.0/Shared/clustersync	2144	43308	2097	3226
		1139	**5225**	**1172**	**958**

The following information is shown for each directory:

- Numbers displayed in bold are benchmarks. The benchmarks were calculated by running standard tests on an Oracle Secure Linux 5.4 system connected to a 10K RPM SAS300 HD via NFS version 4. Both the client and server were on the same network switch and there was no additional traffic at the time of the test.
- **FileSystem** specifies the directory on which the test was performed.
- **Path** specifies the path to the directory on which a file system test was performed.
- **Create** specifies the average number of microseconds in which a create operation was performed.
- **Write** specifies the average number of microseconds in which a write operation was performed.
- **Read** specifies the average number of microseconds in which a read operation was performed.
- **Delete** specifies the average number of microseconds in which a delete operation was performed.

4. For detailed information about each test, review the graph (located under the “Performance Summary” table) and the test results table (located under the graph).

a. To view the results of a single test, either hover over a point on the graph or navigate to the desired row in the test results table.



Test	Threads	Files	Size	Reads	Lock	Attributes	Mode	FileSystem	Create, μs	Write, μs	Read, μs	Delete, μs	Total Test Tim
1	10	10	100	1	true	true	r	local	340/3830/2162	37797/55368/48462	478/5224/1251	929/14889/3201	654848
2	10	10	100	1	true	true	r	data	296/9118/2406	18094/43897/35409	470/4195/1189	924/3049/1463	619339
3	10	10	100	1	true	true	r	spc	312/8471/2520	19672/53419/38562	474/4476/1218	969/9617/2567	614270
4	10	10	100	1	true	true	r	sync	323/8852/2813	23805/48112/37309	488/5781/2474	955/4583/2169	652715
5	10	10	100	1	false	true	r	local	323/9167/2359	29832/48885/42461	477/5736/1774	942/10805/3494	597318
6	10	10	100	1	false	true	r	data	366/8646/3332	38329/59113/49711	460/1084/545	919/5518/1769	722482
7	10	10	100	1	false	true	r	spc	309/8754/2519	24345/47127/35353	477/8468/2702	956/5622/2107	577370
8	10	10	100	1	false	true	r	sync	403/9634/3873	20691/59561/44180	478/11751/4246	1039/18798/5728	793196
9	10	10	100	50	true	true	r	local	294/6466/3268	24998/91358/53298	1239/2276/1721	924/16552/3884	1755555

Note

Each point on the graph and row in the table shows information about a single test. However, to view the details of a single test when you hover over a point in the graph, you must be using a browser other than *Internet Explorer.*

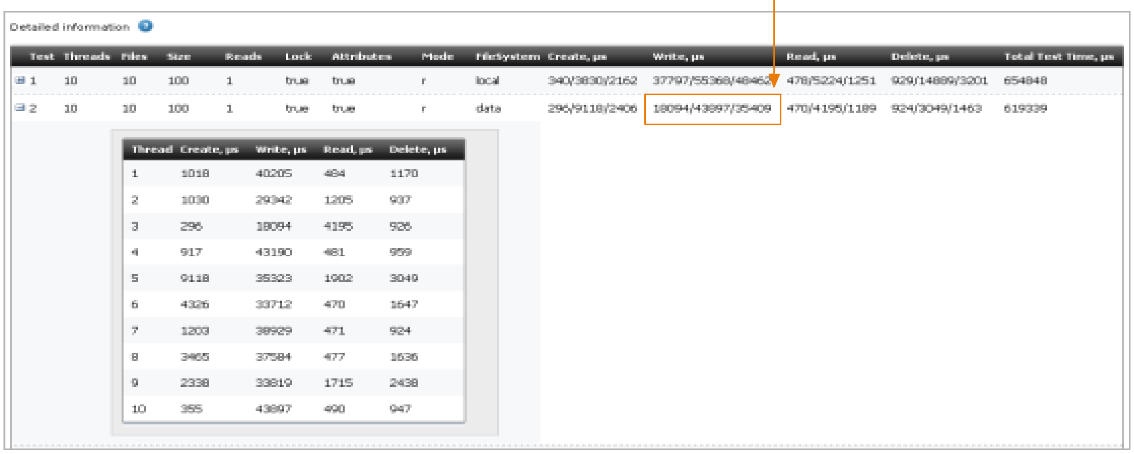
The following information is shown in the table for each test:

- **Test** specifies a serial number for each test.
- **Threads** specifies the number of threads created during the test.
- **Files** specifies the number of files created per thread.
- **Size** specifies the number of bytes per file.
- **Reads** specifies the number of times each file was read by a test.
- **Lock** specifies whether each file was locked (random access lock operation).
- **Attributes** specifies whether each file's attributes were read during the test.
- **Mode** specifies the Java I/O random access mode for accessing the files in the thread.
- **FileSystem** specifies the directory on which the test was performed.
- **Create** specifies the minimum, maximum, and average number of microseconds in which the test performed a create operation.
- **Write** specifies the minimum, maximum, and average number of microseconds in which the test performed a write operation.

- **Read** specifies the minimum, maximum, and average number of microseconds in which the test performed a read operation.
- **Delete** specifies the minimum, maximum, and average number of microseconds in which the test performed a delete operation.
- **Total Test Time** specifies the total number of microseconds in which a test was completed.

b. In the test results table, expand a row to view detailed information about each thread that was created during a specific test.

Slash separated values in the main table specify the minimum, maximum, and average values for the test.



The following information is shown in the thread detail table:

- **Thread** specifies a serial number for each thread.
- **Create** specifies the total number of microseconds in which the thread created an empty file.
- **Write** specifies the total number of microseconds in which the thread finished writing to the file.
- **Read** specifies the total number of microseconds in which the thread read the file it created.
- **Delete** specifies the total number of microseconds in which the thread deleted the file.

5. To download the test results as a spreadsheet, go to the top of the "File System Test" screen and click **Download**.

Resultset Cache

Oracle WebCenter Sites caches resultsets to help optimize system performance. Caching prevents overloading the database with repeated queries for the same information, and serving from cache is always faster than performing another database lookup.

By default, Oracle WebCenter Sites stores resultsets within the inCache framework, instead of hash tables. Both caching frameworks use Java memory. You can switch between frameworks by setting the `rsCacheOverinCache` property in `futuretense.ini` to either `true` or `false`.

When resultset caching over inCache is enabled, the **System Tools** node displays the **Resultset Cache** tool, which provides statistical information about the caches and their contents as described in this section. Additional information about resultset caching is available in the *Oracle WebCenter Sites Developer's Guide*.

Enabling and Configuring Resultset Caching over inCache

If you do not enable resultset caching over inCache, caching in hash tables is enabled by default.

To enable and configure resultset caching

1. Enable resultset caching:
   a. Make sure the `linked-cache.xml` configuration file is present in the application server's classpath (`WEB-INF/classes` directory).
   b. In the `futuretense.ini` file, set the `rsCacheOverinCache` property to `true`.
2. Configure resultset caching:

   In the `futuretense.ini` file, set values for the following resultset caching properties:

   - `cc.cacheResults` – Specify the default number of resultsets to cache.
   - `cc.cacheResultsTimeout` – Specify the number of minutes to keep a resultset cached.
   - `cc.cacheResultsAbs` – Specify how expiration time will be calculated for cached resultsets.

   For detailed information about configuring resultset caching properties, see the *Oracle WebCenter Sites Property Files Reference*.

Viewing Cache Information

The summary view of resultset cache displays statistical information about all caches. You can select a cache from the summary table to view the database queries it contains. You can further view the resultset elements of the database queries contained in a cache.

To view cache information

1. On the **Admin** tab, expand **System Tools > Cache Management > Resultset Cache** and double-click **Summary**.

The summary table displays the following information:

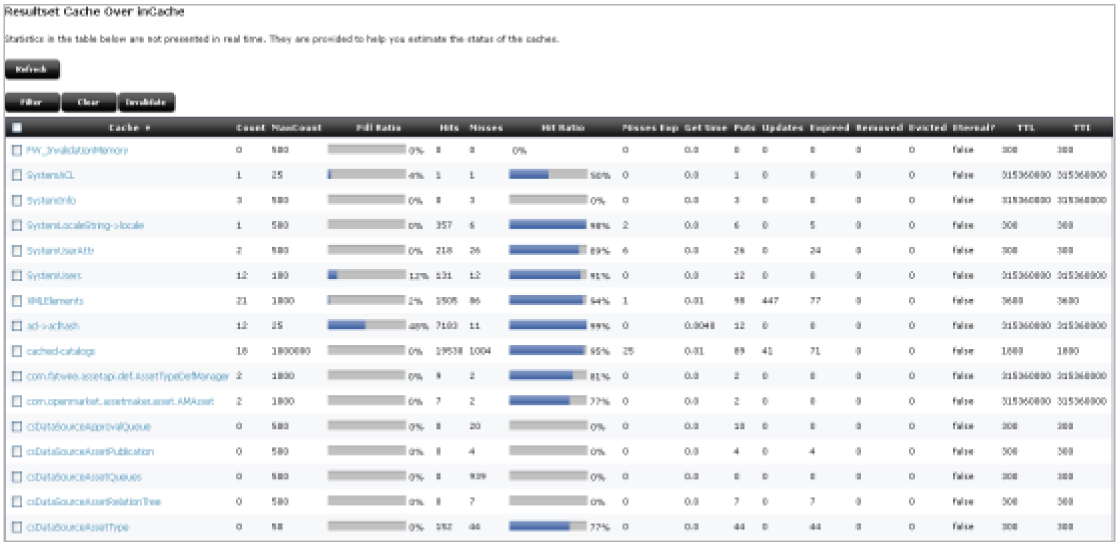


Table 11: Descriptions of Columns in the Summary Table

Column name	Description
Cache	Name of the cache.
Count	Number of resultset elements in the cache. A resultset element is a key-value pair, where the key is the query and the value is the result returned by the query.
MaxCount	Capacity of the cache (maximum number of resultset elements allowed in the cache).
Fill Ratio	Percentage of cache filled: `(Count / MaxCount) x 100`
Hits	Total number of times all resultset elements were read from the cache.
Misses	Number of times the resultset element was not found in cache.
Hit ratio	Percentage of hits: `[Hits / (Hits + Misses)] x 100`
Misses Exp	Number of misses due to resultset elements that are expired but not removed.
Get time	Average time (in ms) for `get` operations.
Puts	Number of times resultset elements were placed in the cache following a database query.
Updates	Number of times individual resultset elements in the cache have been updated.
Expired	Number of expired resultset elements. They are removed by a cleaning thread that runs periodically in the background.
Removed	Number of resultset elements removed for any reason (such as a manual clear operation or cleaning thread).

Table 11: Descriptions of Columns in the Summary Table (continued)

Column name	Description
Evicted	Number of evicted resultset elements. Eviction occurs when the capacity of the cache is exceeded.
Eternal	Indicates whether all resultset elements in the cache have infinite expiration time.
TTL	Time to live. Length of time (in seconds) that resultset elements will remain in cache.
TTI	Time to idle. If a resultset element is not hit (read from cache) within this time period, it is marked as idle.

2. To view detailed information about the contents of a cache, click the cache name in the summary table.

   The detail view opens to show the keys (queries), their related statistics, and a search field for locating the cache's content by key, expiration time, and dependency.

   To refresh your screen, click **Search**.

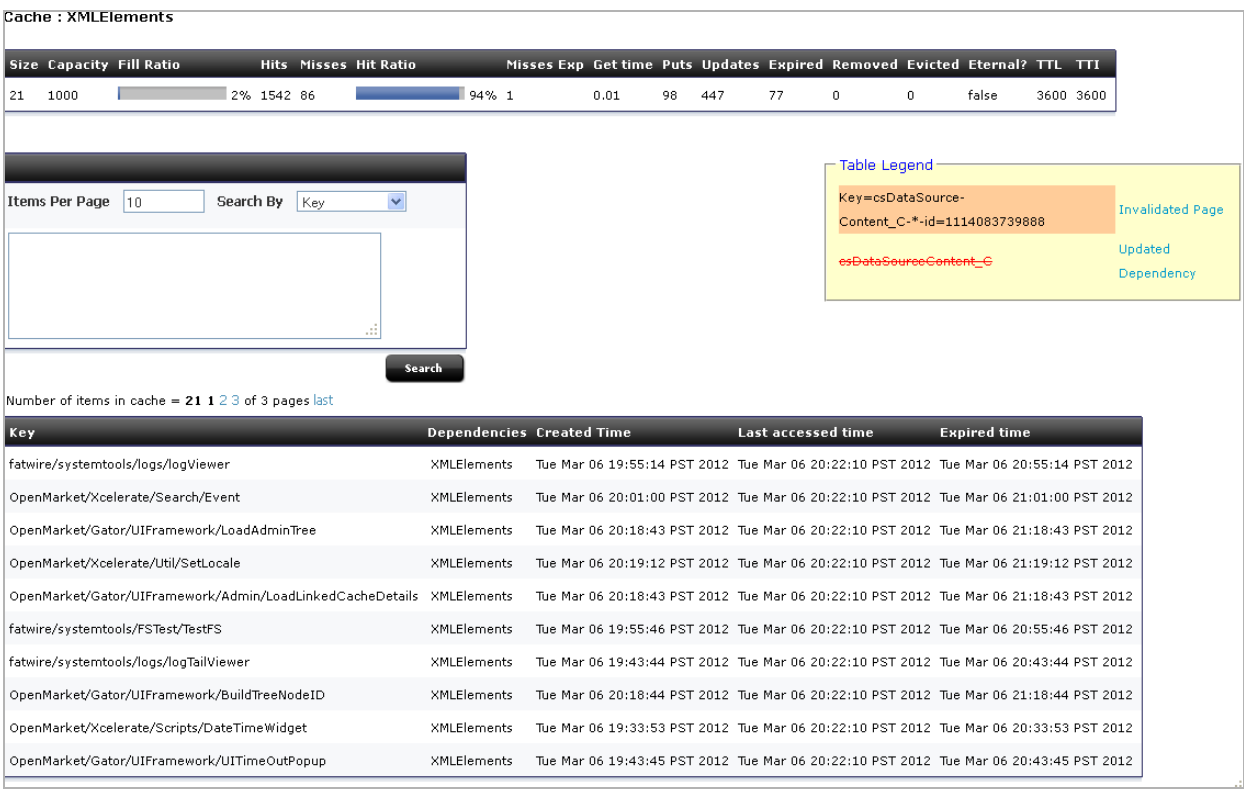
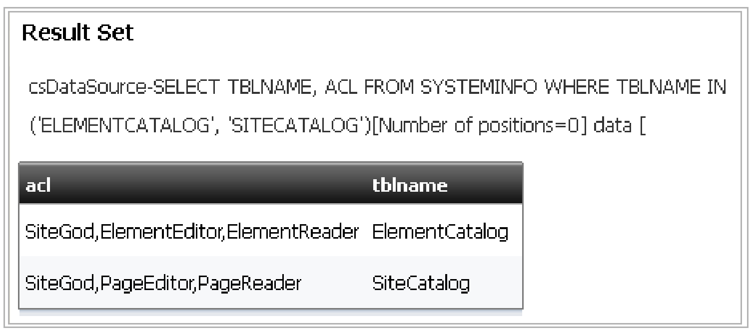


Table 12: Descriptions of Columns in the Cache Detail Table

Column name	Description
Key	Key (query) for the resultset element.
Dependencies	List of tables against which the query was run.
Created time	When this resultset element was placed in the cache.
Last accessed time	When this cached resultset element was last accessed.
Expired time	When this resultset element will expire.

3. To view the resultset element for a key, click the desired key (query).

   Columns in the table that opens are dynamic and depend on your site's data model and how the query is set up.



Operating on a Cache

In the summary table view, you can refresh, filter, clear, and invalidate the resultset elements contained within the caches listed in the table.

To work with caches

1. Go to the summary table. Expand **System Tools > Cache Management > Resultset Cache** and double-click **Summary**.
2. Click one of the following options, located above the summary table:
   - **Refresh**, refreshes the results of the summary table.
   - **Filter** (caches), narrows down the results shown in the summary table to only selected caches.
   - **Clear**, clears selected caches.
   - **Invalidate**, invalidates selected caches. All invalidated resultset elements are purged by the cleaning thread, which runs periodically.

     Each resultset element contains a list of tables on which it depends. When an invalidated resultset element is requested, it will be read from the database and reloaded into cache.

Appendices

This part contains information about WebCenter Sites' system defaults and sample sites. It provides LDAP configuration procedures and describes the system behavior that results when user and site management operations are carried out when the LDAP options are installed.

This part contains the following appendices:

- Appendix A, "System Defaults"
- Appendix B, "System Data: WebCenter Sites Database"
- Appendix C, "Managing Users, Sites, and Roles in LDAP-Integrated Sites Systems"

Appendix A

System Defaults

This appendix lists and describes the WebCenter Sites system defaults that you will be using routinely.

This appendix contains the following sections:

ACLs

WebCenter Sites and its applications use several default ACLs to control user access to their features and functions. This section summarizes the permissions that can be specified in an ACL, and describes WebCenter Sites' default system ACLs.

Permissions

An ACL specifies a set of permissions. When an ACL is assigned to a database table, only the permissions specified in the ACL can be exercised on the database table. Only a user with the same ACL as the table can exercise those permissions.

Table A-1 lists all the permissions that can be specified in an ACL.

Table A-1: Permissions Supported by WebCenter Sites

Permission	Bit Mask[1]	Action
Read	1	Read data from a table.
Retrieve	16	Retrieve the contents of a URL column, also known as an upload field. For information about URL columns, see the *Oracle WebCenter Sites Developer's Guide*
Write	2	Write information to a table.
Create	4	Create a table.
Delete	8	Delete information from a table.
Revision Tracking Audit	32	Access all the revision tracking information for the rows (records) in a tracked table.
Revision Tracking Admin	64	Assign or remove revision tracking on a table.

1. When an ACL is created, the bit mask numbers for each permission assigned to an ACL are added together and the totals are listed with the ACL in the `SystemACL` table.

Accessing ACLs

WebCenter Sites' ACLs and their permissions are accessible as either a listing or an individual entry.

- To obtain the list of ACLs and their permissions, open the `SystemACL` table directly.
- To obtain an individual ACL and its permissions, use the administrator's interface:

1. In the **Admin** tab, expand **Management Tools** and double-click **ACLs**.
2. In the drop-down list, select the ACL you want to work with.

3. Select **Modify ACL** and click **OK**.

Caution

Never modify a default system ACL. Never modify the ACLs assigned to any of the system tables.

For descriptions of the system ACLs, see "System ACLs," on page 524.

System ACLs

Table A-2 lists the system ACLs and their permissions. Each system ACL exists in order to control access to specific parts of the database tables, and subsequently, the product features that use those tables. Although several of the default ACLs have the same set of permissions, the ACLs are all necessary because they are assigned to different tables.

Table A-3 describes the functions of each ACL and how each ACL is used by WebCenter Sites and the Sites content applications.

Table A-2: System ACLs and Their Permissions

ACL Name	Permissions						
	Read	Retrieve	Write	Create	Delete	Rev. Track Audit	Rev. Track Admin
Browser	x						
ContentEditor	x	x	x	x	x	x	
ElementEditor	x	x	x	x	x	x	
ElementReader	x						
PageEditor	x	x	x	x	x	x	
PageReader	x						
RemoteClient	x	x	x	x	x	x	x
SiteGod	x	x	x	x	x	x	x
TableEditor	x	x	x	x	x	x	
UserEditor	x	x	x	x	x	x	
UserReader	x						
Visitor	x	x	x	x	x	x	
VisitorAdmin	x	x	x	x	x	x	x
WSAdmin							
WSEditor							
WSUser							
xceladmin	x	x	x	x	x	x	x
xceleditor	x	x	x	x	x	x	
xcelpublish	x	x	x	x	x	x	

Table A-3: System ACLs and Their Descriptions

ACL Name	Description
Browser	Allows read-only access to the content in the WebCenter Sites database. It is assigned to most of the system default and sample site users. WebCenter Sites requires that all visitors to an online site that it manages have user accounts. For this reason, WebCenter Sites is delivered with a default user account, named DefaultReader, that it assigns to all non-authenticated visitors, that is, those who do not have a user account of their own. The Browser ACL is assigned to the DefaultReader user account, which gives non-authenticated visitors read-only access rights to the content in the WebCenter Sites database.
ContentEditor	Used in a sample WebCenter Sites site. This ACL is assigned to the tables that support the sample site.
ElementEditor	Allows users to write data to the `ElementCatalog` and `SystemSQL` tables. Site designers and anyone who creates templates, CSElement, and SiteEntry assets need this ACL.
ElementReader	Allows users to read data in the `ElementCatalog` and `SystemSQL` tables. WebCenter Sites users need this ACL so they can inspect the templates assigned to their assets.
PageEditor	Allows users to create page entries in the `SiteCatalog` table. Site designers and anyone who creates a template, CSElement, or SiteEntry asset need this ACL.
PageReader	Allows users to read page entries from the `SiteCatalog` table. WebCenter Sites users need this ACL so they can inspect the templates assigned to their assets.
RemoteClient	Grants users the ability to log in to the WebCenter Sites management system through a remote client like Sites Desktop. All Sites Desktop users need this ACL.
SiteGod	Enables complete access to all the tables in the WebCenter Sites database. At least one user of the management system, typically an administrator, must have the SiteGod ACL.
TableEditor	Allows users to create and delete tables in the WebCenter Sites database. Site designers who create database tables or who create new asset types (which causes new tables to be created) need this ACL. Administrators or anyone else who will use the Initialize Mirror Destination feature also needs this ACL.
UserEditor	Allows users to manage user accounts. Administrators need this ACL.

Table A-3: System ACLs and Their Descriptions (continued)

ACL Name	Description
UserReader	Allows user account information to be recognized by WebCenter Sites. WebCenter Sites uses this ACL to determine which users have which roles on which sites. All users need this ACL to be able to access the content management sites to which they are assigned.
Visitor	Grants users the ability to write data to the Oracle WebCenter Sites: Engage tables that store visitor data, and to create recommendation assets. • Any Engage user who needs to create Recommendation assets needs this ACL. • Any authorized visitor (of Engage assets) whose data you are collecting on the delivery system must have this ACL assigned to their user account. • All unauthorized visitors of the online site are automatically assigned this ACL (in the DefaultReader user account) when Engage is installed. For more information about the DefaultReader account, see "DefaultReader, secure.CatalogManager, and secure.TreeManager," on page 99).
VisitorAdmin	Grants users the ability to create visitor attributes, history attributes, and history types. Any Engage user who needs to create assets of those types needs this ACL.
WSUser	Assigned to `SiteCatalog` page entries for the Web Services feature. Grants users the ability to access WebCenter Sites through the WebCenter Sites web services.
WSEditor	Assigned to `SiteCatalog` page entries for the Web Services feature. Grants users the ability to access WebCenter Sites through the WebCenter Sites web services.
WSAdmin	Assigned to `SiteCatalog` page entries for the Web Services feature. Grants users the ability to access WebCenter Sites through the WebCenter Sites web services.
xceladmin	Grants users the ability to create user profiles, roles, sites, asset types, and so on—that is, to use all the functions in the **Admin**, **Site Admin**, and **Workflow** tabs. System, site, and workflow administrators need this ACL. Also, because the **Admin** tab has both administrative and site design functions, site designers also need this ACL.
xceleditor	Grants users the ability to log in to the Sites content applications. The log in request code verifies whether or not a user has the ACL. All users of the management system need this ACL.
xcelpublish	Grants users the ability to view the Publish Console.

ACLs of Default Users

Table A-4: Default Users and Their ACLs

User Name	ACLs	Description
fwadmin	Browser ElementEditor PageEditor PageReader RemoteClient TableEditor UserEditor UserReader Visitor VisitorAdmin xceladmin xceleditor xcelpublish wsadmin wseditor wsuser	Basic administrator user that WebCenter Sites creates so that you can begin configuring your Sites content applications. Do **not** delete this user unless another user with identical ACLs already exists.
WebCenter Sites (the installation's user account)	Browser ContentEditor ElementEditor ElementReader PageEditor PageReader SiteGod TableEditor UserEditor UserReader	User account that the installation program creates during the installation of the products. The name of this account is whatever the installers chose for it.
DefaultReader	Browser Visitor (if Engage is installed)	`Browser` is the ACL that WebCenter Sites assigns to non-authenticated site visitors on the delivery system. The `Visitor` ACL is also automatically assigned if Engage is installed.

Required ACLs for Custom Users

Table A-5: System ACLs Required by Users

User	Required ACLs
All users	Browser, Element Reader, PageReader, UserReader, xceleditor
Workflow Administrator Site Administrator	xceladmin
General Administrator	xceladmin, TableEditor, UserEditor, VisitorAdmin (for Engage)
Site Designer	xceladmin, ElementEditor, PageEditor, TableEditor, Visitor (if Engage is installed), Visitor Admin (for Engage)
Engage Users	Visitor
Users of Web Mode in the Contributor interface	Browser, ElementReader, PageReader, RemoteClient, UserReader, Visitor (for Engage), xceleditor
Sites Desktop Sites DocLink	RemoteClient, Visitor (for Engage)

System Roles

Table A-6: System Roles

Role	Description
GeneralAdmin	Default system role for global WebCenter Sites administrators. Required for users who need access to the **Admin** tab (and all other possible functions) in the tree. **Note**: A user with the GeneralAdmin role must also have the `xceladmin` ACL in order to use any of the functions in the **Admin** tab.
SiteAdmin	Default system role for site administrators. Required for users who are administrators of selected sites and therefore need access to the **Site Admin** tab (which displays a subset of the functions in the **Admin** tab). Assign the SiteAdmin role to users who will manage, but not create, other site users. **Note**: A site user with the SiteAdmin role must also have the `xceladmin` ACL in order to use any of the functions on the **Site Admin** tab.
WorkflowAdmin	Default system role for workflow administrators. Required for users who need access to the **Workflow** tab in the tree. **Note**: A user with the WorkflowAdmin role must also have the `xceladmin` ACL in order to use any of the functions on the **Workflow** tab.
AdvancedUser	Grants WebCenter Sites users access to the administrator’s interface.
SitesUser	Grants WebCenter Sites users access to the content contributor’s interface.

System Asset Types

Table A-7 lists the default asset types. Unlike custom asset types, system asset types cannot be deleted.

Table A-7: System Asset Types

Asset Type	Description
Attribute Editor	An attribute editor specifies how data is entered for a flex attribute when that attribute is displayed on a "New" or "Edit" form for a flex asset or a flex parent asset. It is similar to a template asset. However, unlike a template asset, you use it to identify the code that you want WebCenter Sites to use when it displays an attribute in the Sites interface—not when it displays the value of an attribute on your online site.
CSElement	Stores code (XML or JSP and Java) does not render assets. Typically, you use CSElements for common code that you want to call from more than one template (a banner perhaps). You also use CSElements to provide the queries that are needed to create DynamicList recommendations in Engage.
Collection	Stores an ordered list of assets of one type. You "build" collections by running one or more queries, selecting items from their resultsets, and then ranking (ordering) the items that you selected. This ranked, ordered list is the collection. For example, you could rank a collection of articles about politics so that the article about last night's election results is number one.
Dimension	Represents a locale in a site. You must create a "Dimension" asset for each locale you want to enable on the management system. To enable publishing of content in a given locale, you must publish the corresponding "Dimension" asset to the delivery system, and enable the locale in the site's dimension set.
Dimension Set	Defines which locales and locale filter are enabled on the online site. For locale filtering to work on the delivery site, you must create and publish to the delivery system at least one "DimensionSet" asset. Has no effect on the management system.
History Attribute	Individual information types that you group together to create a vector of information that Engage treats as a single record. This vector of data is the history definition. For example, a history type called "purchases" can consist of the history attributes "SKU," "itemname," "quantity," and "price." Available in Engage.
History Definition	The vector of data in a History Attribute. This vector of data is the history definition. For example, a history type called "purchases" can consist of the history attributes "SKU," "itemname," "quantity," and "price." Available in Engage.
Link	Stores a URL to an external web site. You use this asset to embed an external link within another asset.

Table A-7: System Asset Types (continued)

Asset Type	Description
Page	Stores references to other assets. Arranging and designing page assets is how you represent the organization or design of your site. You design page assets by selecting the appropriate collections, articles, imagefiles, queries, and so on for them. Then, you position your page assets on the Site Plan tab that represents your site in the tree on the left side of the WebCenter Sites interfaces.
Promotion	Is a merchandising asset that offers some type of value or discount to your site visitors based on the flex assets (products, perhaps) that the visitor is buying and the segments that the visitor qualifies for. Available in Engage.
Query	Stores queries that retrieve a list of assets based on selected parameters or criteria. You use query assets in page assets, collections, and recommendations. The database query can be either written directly in the "New" or "Edit" form for the query asset as a SQL query, or written in an element (with WebCenter Sites query tags or a as a search engine query) that is identified in the "New" or "Edit" form.
Recommendation	This is like an advanced collection. It collects, assesses, and sorts flex assets (products or articles, perhaps) and then recommends the most appropriate ones for the current visitor, based on the segments that visitor belongs to. Available in Engage.
Segment	Assets that divide visitors into groups based on common characteristics (visitor attributes and history types). You build segments by determining which visitor data assets to base them on and then setting qualifying values for those criteria. For example, a segment could define people who live in Alaska and own fly fishing gear, or it could define people who bought a personal computer in the past six months, and so on. Available in Engage.
SiteEntry	Represents a WebCenter Sites page or pagelet and has a CSElement assigned as the root element that generates the page. Template assets do not have associated SiteEntry assets because they represent both an element and a WebCenter Sites page.
Template	Stores code (XML or JSP and Java) that renders other assets into WebCenter Sites pages and pagelets. Developers code a standard set of templates for each asset type (other than CSElement and SiteEntry) so that all assets of the same type are formatted in the same way. Content providers can select templates for previewing their content assets without having access to the code itself or being required to code.
Visitor Attribute	Holds types of information that specify one characteristic only (scalar values). For example, you can create visitor attributes named "years of experience," "job title," or "number of children." Available in Engage.

Default Tree Tabs

Table A-8 lists the default tabs in WebCenter Sites' tree. These tabs are critical to WebCenter Sites. All features which stem from WebCenter Sites can be accessed through these tabs; they are automatically created upon installation.

Table A-8: Default Tabs in WebCenter Sites

Tab	Description
Active List	Displays items that are in the process of being created or edited in WebCenter Sites.
Admin	Displays the administrative functions that affect all of the CM sites in the system. By default, only users with the default system role named GeneralAdmin have access to this tab.
Design	A source for creating pages on your site. Some of these sources are: Templates, Product Definition, Content Definition, and other sources for the creating pages.
History	Displays the assets that you worked with during the current session. All users see this tab as soon as they create, inspect, edit, or copy their first asset.
Query	Enables a user to query for certain types of articles and organize them in that fashion. This is accomplished using a SQL query.
Site Admin	Holds a subset of the system-wide administrative functions. The subset applies only to the CM site that the SiteAdmin is logged in to. By default, only users with the default system role named SiteAdmin have access to this tab. This tab is useful if to individuals who manage access to individual CM sites, but who do not need to create new users or new sites.
Site Plan	Represents the layout and overview of the site. This tab shows each site that is controlled by WebCenter Sites. It lists the "placed pages" and the "unplaced pages." The "placed pages" are pages which are created and have been integrated into the live site. "Unplaced Pages" are pages which are finished but are not integrated into the live site.
Workflow	Lists the workflow configuration functions. By default, only users with the Workflow Admin role have access to this tab.

Appendix B

System Data: WebCenter Sites Database

This appendix contains information about the dynamic tables in the WebCenter Sites database and how they grow. DBAs can use this information to determine how to size the WebCenter Sites database appropriately.

This appendix contains the following sections:

- Cache Management Tables
- Approval System Tables
- Publishing System Tables
- Workflow Tables
- Flex Asset Tables
- Visitor Tables (Oracle WebCenter Sites: Engage)

This appendix also contains information about how to purge inactive data from the visitor tables.

Cache Management Tables

WebCenter Sites delivers the CacheManager, a page caching utility that manages both the WebCenter Sites page cache and the Satellite Server caches.

Because cached pages need to expire both when their freshness date expires and when an asset that the page refers to in some way is changed, the CacheManager keeps track of expiration times as well as the dependencies that exist between the pages and pagelets stored in the cache. It stores this information in the `SystemPageCache` and `SystemItemCache` tables.

Every Sites system uses a CacheManager, which means that these tables grow dynamically on any system—development, management, or delivery. The cache-tracking tables grow at the following rate:

Table	Number of Rows
`SystemPageCache`	One row for every cached page. When a page expires, the row is removed.
`SystemItemCache`	One row for each asset, pagelet, or other item that is referenced by a cached page in the `SystemPageCache` table. For example, if a cached page was created from a page asset that has associations to three article assets, there would be four rows for that cached page.

Approval System Tables

The WebCenter Sites approval system keeps track of each asset that has been approved, the dependencies that approved assets have on other assets, and the targets for which assets are approved. It stores this information in the `ApprovedAssets` and `ApprovedAssetDeps` tables.

These tables have the potential to grow very large on a management system, but are not used on a delivery system. The approval system tables grow at the following rate:

Table	Number of Rows
`ApprovedAssets`	One row for each asset that has been approved, for each target destination. That is, if an asset has been approved for two destinations, that asset has two rows in this table.
`ApprovedAssetDeps`	One row for each asset dependency for each approved asset, for each target destination. For example, if an approved asset is dependent on four other assets, it has four rows in this table. If that same asset is approved to two destinations, it has one row each for each dependency for each destination.

Publishing System Tables

The WebCenter Sites publishing system keeps track of when assets were published, and where they were published to. It stores this information in the `PubKey` and the `PublishedAssets` tables.

As the number of assets in your Sites management system increases, so does the number of rows in these tables. These tables grow at the following rate:

Table	Number of Rows
`PubKey`	• For mirror publishing: one row for every asset that is mirrored, for each target destination. • For export publishing: one row for each page that is created during the export. That is, if 14 assets are rendered into 1 page, the table holds 1 row—not 14—for the entire group of assets.
`PublishedAssets`	• For mirror publishing: one row for every asset that is mirrored, for each target destination. • For export publishing: one row for each asset that is exported. To continue the preceding example, if 14 assets are rendered onto 1 page, the table holds 1 rows (one for each asset) for that page.

Workflow Tables

The workflow system keeps track of all the assets that are involved in a workflow process at any given time. It stores this information in the `Assignment` and `WorkflowObject` tables.

As the number of assets that are placed in a workflow process increases, so does the number of rows in these tables. These tables grow at the following rate:

Table	Number of Rows
`Assignment`	One row for each workflow assignment. For example, if an asset is assigned to four users during the course of a workflow process, that asset has four rows in the table. These rows are not deleted when the workflow process is completed for the asset.
`WorkflowObjects`	One row for each asset in workflow. When an asset leaves workflow, the row is deleted.

Basic Asset Tables

A basic asset type has one primary storage table. For example, the primary storage table for the article asset type is named `Article`; the primary storage table for the `HelloArticle` asset type is named `HelloArticle`.

As the number of assets of a single type increases, so does the size of the table that holds assets of that type. The primary storage table for a basic asset has one row for each asset of that type.

Flex Asset Tables

Each asset type in a flex family has several database tables. The three types of tables in any flex family that can potentially grow quite large are as follows:

- The primary storage table for the **flex asset** type. For example, the primary storage table for the avisports sample site asset type named article is `AVIArticle`.
- The `_AMap` tables for flex asset or flex parent asset types. (For example, `AVIArticle_AMap`.)
- The `_Group` tables for flex parent asset types. (For example, `ArticleCategory_Group`.)
- The `_Mungo` table for the **flex asset** type. (For example, `AVIArticle_Mungo`.)
- The `_Mungo` table for the flex **parent** asset type. (For example, `ArticleCategory_Mungo`.)
- The `Mungo_Blobs` table

These types of tables grow at the following rate:

Table	Number of Rows
`FlexAssetType` (for example: `AVIArticle`)	One row for every asset of this type.
`FlexAssetType``_AMap` (for example: `AVIArticle_AMap`)	One row per attribute value—whether the attribute value is inherited or directly assigned—for each of the assets of that type.
`FlexAssetType``_Group` (for example: `ArticleCategory_Group`)	One row per ancestor relationship between flex parent asset and flex asset—includes rows for grandparent, great-grandparent, and so on, relationships.
`FlexAssetType``_Mungo` (for example: `AVIArticle_Mungo`)	One row for every attribute value for each asset of this type. In some cases, this equation can result in more than 10 million rows.
`FlexParent_Mungo` (for example: `ArticleCategory_Mungo`)	One row for every attribute value for every parent asset of this type.

Table	Number of Rows
`Mungo_Blobs`	One row for every attribute value saved for an attribute of type `blob`.

Visitor Tables (Oracle WebCenter Sites: Engage)

Oracle WebCenter Sites: Engage captures visitor information and stores it in the visitor data tables. These tables store information such as session IDs for visitors so that they can be linked with their previous sessions and values for the attributes that represent the data you are collecting.

As the number of visitors who visit your online site increases, so do the rows in these tables. These tables grow at the following rate:

Table	Number of Rows
`scratch`	One row for each visitor context session object that is created for a visitor. Visitor context session objects are things like promotion lists, segment lists, shopping carts, and so on. There are at least five rows added to this table for each visitor in each session.
`VMVISITOR`	One row for each visitor for each browser session. Engage creates a unique visitor ID for each visitor for each session.
`VMVISITORALIAS`	In database-centric visitor tracking, one row is created for each visitor for each browser session, even for returning visitors. This method creates *one row per visit*. Memory-centric visitor tracking creates *one row per visitor*, regardless of the number of returning visits. **Note:** A row holds the name/value pair of an alias and the visitor ID (also listed in `VMVISITOR`) that marks a session.
`VMVISITORSCALARVALUE`	One row for each visitor attribute value (except for attributes of type binary) that is saved. **Note:** This table is not populated when memory-based visitor tracking is enabled.
`VMVISTORSCALARBLOB`	One row for each visitor attribute value of type binary (also referred to as scalar objects) that is saved. **Note:** This table is not populated when memory-based visitor tracking is enabled.

Table	Number of Rows
VMz ------------	These are dynamically generated tables that are created when values for a history attribute are saved. Engage creates one table for each history type and adds a row to the table each time a record of that type is saved.

Managing the Attribute Tables

Because the tables that hold attribute values can grow very quickly, you should purge inactive data from them regularly.

You use the following Engage XML object method or its JSP equivalent to delete inactive data from these tables:

```
<VDM.FLUSHINACTIVE STARTDATE="cutoffDate"/>
```

Inactive visitor data is data marked with a visitor ID that is not connected through an alias to data that you consider current. You set a cutoff date (`STARTDATE`) for Engage to use. All visitor data recorded before that date is deleted from the previously listed visitor tables **unless** it is linked through an alias with data recorded after that date.

There are several ways to use the `VDM.FLUSHINACTIVE` tag. For example:

- You can create an administrative element that invokes the tag and prompts you to enter the cutoff date.
- You can provide it with an equation that calculates a cutoff date based on some parameter so that you can set it up to run as an automatic event at a regularly scheduled time. To set it up as an automatic event, use the WebCenter Sites `APPEVENT` tag (which functions like a `kron` job). For information about this tag, see the *Oracle WebCenter Sites Developer's Tag Reference*, and the "Coding Basics" section of the *Oracle WebCenter Sites Developer's Guide*.

Note that the value that you pass to the `STARTDATE` parameter must be in epoch time. You can use the WebCenter Sites `DATE.CONVERT` tag to obtain an epoch value for the date that you want to use. For example:

```
<DATE.CONVERT VARNAME="flushtime"
   YEAR="four digit year" MONTH="number in the range 1-12"
   DAY="number in the range 1-31"
   [HOUR="number in the range 0-11 where 0 is midnight" AMPM="am
   or pm" MINUTE="number in the range 0-59" TIMEZONE="timezone"]/>
<VDM.FLUSHINACTIVE STARTDATE="Variables.flushtime"/>
```

For more information about these tags, see the *Oracle WebCenter Sites Developer's Tag Reference*.

Managing the Session Objects Table (scratch)

The `scratch` table can grow quickly because there are at least five session objects stored for each visitor in each session. Each object has a timestamp; you should purge old objects regularly, based on their timestamps.

You use the following Engage XML object method or its JSP equivalent to delete old session objects from the scratch table:

```
<SESSIONOBJECTS.FLUSH TIMESTAMP="cutoffTime"/>
```

Because “session objects” includes carts, you must set the cutoff time to represent the point at which you consider a cart to be abandoned.

Note

This object method does not affect the `VMSCALARBLOB` table, which means that it does not delete carts that you have stored as a visitor attribute of type binary (a scalar object).

Note that the value that you pass to the `STARTDATE` parameter must be in epoch time. You can use the WebCenter Sites `DATE.CONVERT` tag to obtain an epoch value for the date that you want to use. For an example, see the code example for the `VDM.FLUSHINACTIVE` tag provided in the preceding section.

For more information about these tags, see the *Oracle WebCenter Sites Developer’s Tag Reference*.

Deleting Unnecessary .class Files

Oracle WebCenter Sites: Engage is a Java-based application and it generates Java `.class` files each time one of the following events occurs:

- A segment, recommendation, or promotion is created.
- A product is configured for a related items recommendation.
- A segment, recommendation, or promotion is calculated or invoked by Engage.

Typically, old `.class` files are deleted when a segment, recommendation, promotion, or product is updated and are then replaced with new `.class` files. However, if the segment, recommendation, product, or promotion is in use when an updated version is published, Engage cannot delete the old `.class` file because it is locked.

The old `.class` files can build up, filling up the disk and using memory. Therefore, depending on how much development work you are doing and how frequently you publish to the delivery system, you must manually delete these `.class` files at a regularly scheduled time.

Complete the following steps to delete old `.class` files:

1. Use the Property Editor to examine the `vis.genclasspath` property and note the directory name designated by that parameter. This is the directory where Engage stores `.class` files.
2. During a quiet time on your site, shut down and restart each instance of JRE runtime that is running. This process releases any old `.class` files that the JRE runtime has locked.
3. Using any file management tool, navigate to the directory that holds the `.class` files and delete the contents of this directory.

Engage regenerates any `.class` files that it needs when it needs them.

Appendix C

Managing Users, Sites, and Roles in LDAP-Integrated Sites Systems

This appendix is for WebCenter Sites administrators who are managing users, ACLs, sites and roles from the administrator's interface of an LDAP-integrated Sites system and need to know how their operations affect both the Sites database and the LDAP server. This appendix summarizes:

- Which operations are propagated to the LDAP server (given the type of LDAP server, the operation, and whether you are connected as user with read-only or read-write permissions)
- Which operations produce errors
- How to correct errors

This appendix contains the following sections:

- Overview
- User Management Operations
- Site and Role Management Operations

Overview

WebCenter Sites can be integrated with LDAP servers that use the LDAP-2 protocol or any other protocol (although, in the latter case, write operations from the administrator's interface to the LDAP server are not supported).

This overview summarizes

- the types of LDAP schema that WebCenter Sites supports,
- operations that can be performed from the Sites interface when each type of schema is deployed, and
- the outcome of the operations, given the permissions of the WebCenter Sites administrator to the LDAP server and the nature of the LDAP server.

LDAP Schema

The following LDAP-schema scenarios are covered in this appendix: flat- and hierarchical -schema LDAP. Each schema supports selected operations and installations:

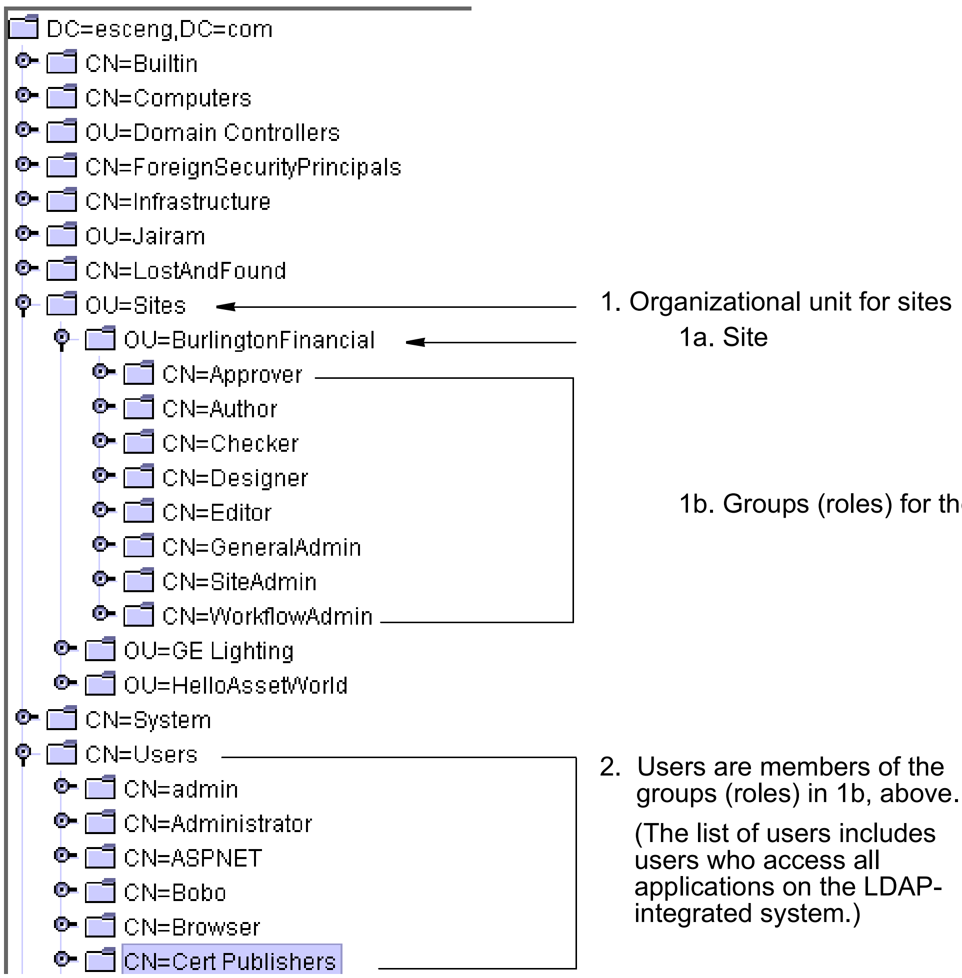
- Both flat- and hierarchical-schema LDAP support operations on ACLs and users (user accounts, user profiles, and user attributes) in WebCenter Sites.
- Flat-schema LDAP supports operations on sites and roles in WebCenter Sites and levels the sites and roles so that no hierarchy exists among them.
- Hierarchical-schema LDAP supports operations on sites and roles in WebCenter Sites. This schema requires LDAP users to define a site organizational unit in the LDAP server in which roles must be subordinated to their relevant sites (for an example, see Figure C-1, "LDAP Hierarchies").

Table C-1 summarizes LDAP schema and the possible operations. Using the correct schema, the Sites administrator can perform operations in the Sites interface and have them propagate to the WebCenter Sites database, or the LDAP server, or both, depending also on the conditions that are outlined in "LDAP Connectivity for Site and Role Management," on page 543.

Table C-1: LDAP Integration Scenarios

Create/Modify/Delete Operations on ...	Flat-Schema LDAP	Hierarchical-Schema LDAP
ACLs, Users, User Profiles, User Attributes	Table C-2	Table C-2
Sites and Roles	Table C-3	Table C-4

Figure C-1: LDAP Hierarchies



LDAP Connectivity for Site and Role Management

For operations on roles and sites to work as described in this appendix, LDAP connectivity must be enabled by setting the values of two properties in the `futuretense_xcel.ini` file:

```
xcelerate.usermanagerclass
xcelerate.rolemanagerclass
```

These properties allow you to manage sites and roles directly in the LDAP server (as shown in this appendix) or exclusively in the WebCenter Sites database.

- If the property values specify LDAP, then LDAP connectivity is established. Sites and roles can be managed in the LDAP server (and in the WebCenter Sites database).
  - For hierarchical-schema LDAP, only the `xcelerate.usermanagerclass` property must specify LDAP. The `xcelerate.rolemanagerclass` property uses the default value. For example:

```
xcelerate.usermanagerclass=com.openmarket.xcelerate.
 user.LDAPSchemaUserManager
```

```
xcelerate.rolemanagerclass=com.openmarket.xcelerate.
 roles.RoleManager
```

- For flat-schema LDAP, both properties must specify LDAP. For example:

```
xcelerate.usermanagerclass=com.openmarket.xcelerate.
 user.FlatLDAPSchemaUserManager
```

```
xcelerate.rolemanagerclass=com.openmarket.xcelerate.
 roles.FlatLDAPSchemaRoleManager
```

- If the property values do not specify LDAP (for example, `xcelerate.usermanagerclass=com.openmarket.xcelerate.user.usermanager)` then sites and roles can managed only in the WebCenter Sites database.

LDAP Users and Their Permissions to LDAP Servers

In an LDAP-integrated Sites system, the administrator of the WebCenter Sites system may or may not be an administrative user of the LDAP server, depending on the value of the `jndi.connectAsUser` property in the `dir.ini` file. The value determines how the WebCenter Sites administrator is connected to the LDAP server, and therefore defines the LDAP user:

- If `jndi.connectAsUser` is set to `true`, then WebCenter Sites defines the LDAP user to be the same one that is logged in to WebCenter Sites and connects that user to the LDAP server.
- If `jndi.connectAsUser` is set to `false`, then WebCenter Sites defines the LDAP user to be the one that is specified in the `jndi.login` property (in `dir.ini`) and connects that user to the LDAP server.

For the connection to take place, the user and his permissions must also be defined in the LDAP server. If the user has read-only permissions to the LDAP server, he is not an administrative LDAP user. If the user has read and write permissions to the LDAP server, he is an administrative LDAP user (or simply, an LDAP administrator).

Note

For more information about LDAP-related properties, see the *Oracle WebCenter Sites Property Files Reference*.

LDAP-Integrated Operations

For the WebCenter Sites administrator to successfully perform an operation (such as creating an ACL in both the WebCenter Sites database and the LDAP server), it is critical for WebCenter Sites to be properly integrated with the LDAP server. Barring integration issues, the outcome of an operation depends on the following factors:

1. The LDAP user, as defined by the `jndi.connectAsUser` property (see the previous section).
2. The LDAP user's permissions, as defined in the LDAP server.

If the Sites administrator is connected to the LDAP server as a user without administrative rights, his operations (such as deleting an ACL from the WebCenter Sites database) cannot be written to the LDAP server. The system responds by either writing the operations to the WebCenter Sites database, or not at all. The operations must then be repeated in the administrator's interface and performed manually in the LDAP server by an LDAP administrator.

3. The nature of the operation.

   Certain operations that are performed in the administrator's interface (such as editing an ACL) are not written to the LDAP server, even when the WebCenter Sites administrator is connected with write permissions.

4. Whether the LDAP server supports the LDAP-2 protocol.

   If the LDAP server does not support the LDAP-2 protocol, then write operations from the administrator's interface to the LDAP server are not supported, and the result of an operation cannot always be predicted.

Tables C-2 through C-4 summarize the results of operations that can be performed by a WebCenter Sites administrator who is connected to the LDAP server as a user with read-only and read/write permissions.

User Management Operations

Table C-2, on page 546 applies to flat- and hierarchical-schema LDAP web environments and summarizes system response to user management operations. The operations are performed by a WebCenter Sites administrator using the interfaces of three WebCenter Sites systems, each integrated with one of the following LDAP options:

- LDAP-2 server with read-only permissions for the LDAP user (defined on page 543)
- LDAP-2 server with write permissions for the LDAP user
- LDAP server other than LDAP-2, in which case write operations are not supported

The results of each user management operation are described on the pages that are noted in the left-hand column of Table C-2.

> **Note**
>
> The term "user management" in this appendix includes the management of ACLs, user accounts, user profiles, and user attributes.

Table C-2: System Response to User Management Operations in WebCenter Sites

Operation in WebCenter Sites	Result	System Response With: LDAP-2: User has Read-Only Permissions [1]	LDAP-2: User has Write Permissions [2]	Not LDAP-2: Write not Supported [3]
Creating an ACL (p. 547)	*ACL is created in:*	Returns error[4]	LDAP + Sites database	Unpredictable
Editing an ACL (p. 547)	*ACL is modified in:*	Sites database	Sites database	Sites database
Deleting an ACL (p. 547)	*ACL is deleted in:*	Returns error	LDAP + Sites database	Unpredictable
Assigning ACLs to Tables (p. 548)	*ACL is assigned in:*	Sites database	Sites database	Sites database
Assigning ACLs to WebCenter Sites Pages (Site Catalog Page Entries) (p. 548)	*ACL is assigned in:*	Sites database	Sites database	Sites database
Creating a User (Granting ACLs for Access Privileges) (p. 548)	*User is created in:*	Returns error	LDAP	Unpredictable
Editing a User (p. 549)	*User is modified in:*	Returns error	LDAP	Unpredictable
Deleting a User (p. 549)	*User is deleted from:*	Returns error	LDAP	Unpredictable
Creating/Editing a User Profile (p. 550)	*User profile is created/edited in:*	Returns error	LDAP	Unpredictable
Deleting a User Profile (p. 550)	*User profile is deleted from:*	Returns error	LDAP	Unpredictable
Creating User Attributes (p. 551)	*User profile is created in:*	Returns error	LDAP	Unpredictable
Editing/Deleting User Attributes (p. 551)	*User profile is deleted from:*	Returns error	LDAP	Unpredictable

1. ACLs and users cannot be stored in the LDAP server (the WebCenter Sites administrator is connected to the LDAP server as a user without write permissions).
2. ACLs and users can be stored in the LDAP server (the WebCenter Sites administrator is connected to the LDAP server as a user with write permissions).
3. The LDAP server does not support the LDAP-2 protocol (such as Netscape Server). For non-compliant LDAP servers, write operations from the administrator's interface to the LDAP server are not supported.
4. "Returns error" means that the operation is not performed (the system returns an error).

Creating an ACL

Operation

The WebCenter Sites administrator attempts to create an ACL by using the "Management Tools" node on the "Admin" tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot create the ACL. The LDAP administrator must first create the ACL manually in the LDAP server and then in the WebCenter Sites database.
2. LDAP-2 Server with Write Permissions—The system creates the ACL in both the LDAP server and the WebCenter Sites database.
3. Not LDAP-2: Write not Supported—Response is unpredictable. The LDAP administrator must first create the ACL in the LDAP server and then in WebCenter Sites.

Editing an ACL

Operation

The WebCenter Sites administrator attempts to modify an ACL by using the "Management Tools" node on the "Admin" tab.

> **Note**
>
> The WebCenter Sites administrator can modify either the description of the ACL, or the permission associated with the ACL, but not the ACL's name.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system modifies the ACL in the WebCenter Sites database. The modification is stored only in the database, even with LDAP integration.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Deleting an ACL

Operation

The WebCenter Sites administrator attempts to delete an ACL by using the "Management Tools" node on the "Admin" tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot delete the ACL. The LDAP administrator must first delete the ACL from the LDAP server and then from the WebCenter Sites database.

2. LDAP-2 Server with Write Permissions—The system deletes the ACL from the LDAP server and from the WebCenter Sites database.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must first delete the ACL from the LDAP server before attempting to delete the ACL from WebCenter Sites.

Assigning ACLs to Tables

Operation

The WebCenter Sites administrator attempts to assign an ACL to system tables, using the “Management Tools” node on the “Admin” tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system assigns the ACLs to the database tables. If the ACLs cannot be assigned, the system returns an error message.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Assigning ACLs to WebCenter Sites Pages (Site Catalog Page Entries)

Operation

The WebCenter Sites administrator attempts to assign an ACL to WebCenter Sites Pages by using the “Management Tools” node on the “Admin” tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system adds the selected ACLs for the specified WebCenter Sites Page. If the ACLs cannot be assigned, the system returns an error message.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Creating a User (Granting ACLs for Access Privileges)

Operation

The WebCenter Sites administrator attempts to create a user by using the “Management Tools” node on the “Admin” tab.

Note

WebCenter Sites displays only the ACLs that are stored in its database.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot add the user and the user's assigned ACLs. The LDAP administrator must use the LDAP server to manually add the user to the LDAP server and assign ACLs to the user.
2. LDAP-2 Server with Write Permissions—The system creates the user in the LDAP server and assigns ACLs to the user.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must use the LDAP server to manually add the user to the LDAP server and assign ACLs to the user.

Editing a User

Operation

The WebCenter Sites administrator attempts to modify the ACLs that are assigned to a user or to change the user's password by using the "Management Tools" node on the "Admin" tab

Note

The "Modify user" operation allows you to change only the user password and the ACLs that are associated with the user.

The "Modify user attributes" operation allows you to change all attributes that are stored in the database or in the LDAP server.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot modify the user's password and/or ACLs. The LDAP administrator must manually modify the user in the LDAP server.
2. LDAP-2 Server with Write Permissions—The system modifies the user's password and/or assigned ACLs in the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must manually modify the user in the LDAP server.

Deleting a User

Operation

The WebCenter Sites administrator attempts to delete a user by using the "Management Tools" node on the "Admin" tab.

Before deleting a user, the administrator must delete the user profile. If a user profile exists, an error message is triggered. The behavior described below is expected when no user profile exists for the user being deleted.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot delete the user. The LDAP administrator must manually delete the user from LDAP server.
2. LDAP-2 Server with Write Permissions—The system deletes the user from the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must manually delete the user from the LDAP server.

Creating/Editing a User Profile

Operation

The WebCenter Sites administrator attempts to create or modify a user profile by using the “User Profiles” node on the “Admin” tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot create/modify the user’s profile and/or Locale Preference. The LDAP administrator must manually create/modify the user profile in the LDAP server.
2. LDAP-2 Server with Write Permissions—The system creates/modifies the user’s profile and/or locale preference in the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must manually create/modify the user profile in the LDAP server.

Deleting a User Profile

Operation

The WebCenter Sites administrator attempts to delete a user profile by using the “User Profiles” node on the “Admin” tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot delete the user’s profile. The LDAP administrator must manually delete the user profile in the LDAP server.
2. LDAP-2 Server with Write Permissions—The system deletes the user’s profile from the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must manually delete the user profile from the LDAP server.

Creating User Attributes

> **Note**
>
> User attributes are stored in only one place—either in the database or in LDAP.

Operation

The WebCenter Sites administrator attempts to create new attributes for the user from the "Management Tools" node of the "Admin" tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot add the new attributes to the LDAP server. The LDAP administrator must manually add user attributes to the LDAP server.
2. LDAP-2 Server with Write Permissions—The system adds the user's new attributes to the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must manually add user attributes to the LDAP server.

Editing/Deleting User Attributes

The WebCenter Sites administrator attempts to modify the user attributes from the "Management Tools" node of the "Admin" tab.

System Response

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot modify or delete the user's attributes from the LDAP server. The LDAP administrator must manually modify user attributes in the LDAP server and delete user attributes from the LDAP server.
2. LDAP-2 Server with Write Permissions—The system modifies the user attributes in the LDAP server and deletes them from the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The LDAP administrator must manually modify user attributes in the LDAP server, and manually delete user attributes from the LDAP server.

Site and Role Management Operations

This section covers site and role management operations with flat- and hierarchical-schema LDAP servers.

Operations with Flat-Schema LDAP Servers

Table C-3 summarizes system response to site and role management operations. The operations are performed by a WebCenter Sites administrator using the interfaces of three WebCenter Sites systems, each integrated with one of the following LDAP options:

- An LDAP-2 server with read-only permissions for the LDAP user (defined on page 543)
- An LDAP-2 server with write permissions for the LDAP user
- An LDAP server other than LDAP-2, in which case write operations are not supported

The results of each operation are described on the pages that are noted in the left-hand column of Table C-3.

Table C-3: System Response to Site and Role Management Operations with Flat-Schema LDAP

Operation in WebCenter Sites	Result	System Response with Flat-Schema LDAP: LDAP-2: User has Read-Only Permissions [1]	LDAP-2: User has Write Permissions [2]	Not LDAP-2: Write not Supported [3]
Creating a Site (p. 554)	*Site is created in:*	Sites database	Sites database	Sites database
Editing a Site (p. 554)	*Site is edited in:*	Sites database	Sites database	Sites database
Deleting a Site (p. 555)	*Site is deleted from:*	Sites database	Sites database	Sites database
Creating a Role (p. 557)	*Role is created in:*	Returns error[4]	LDAP + Sites database[5]	Unpredictable
Editing a Role (p. 557)	*Role is edited in:*	Sites database	Sites database	Sites database
Deleting a Role (p. 558)	*Role is deleted from:*	Returns error	LDAP + Sites database	Unpredictable
Granting Users Access to Sites (p. 555)	*Access is granted in:*	Returns error	LDAP	Unpredictable
Removing Users' Access to Sites (p. 556)	*Access is removed from:*	Returns error	LDAP	Unpredictable

1. Sites and roles cannot be stored in the LDAP server (the WebCenter Sites administrator is connected to the LDAP server as a user without write permissions).
2. Sites and roles can be stored in the LDAP server (the WebCenter Sites administrator is connected to the LDAP server as a user with write permissions).
3. The LDAP server does not support the LDAP-2 protocol (such as Netscape Server). For non-compliant LDAP servers, write operations from the administrator's interface to the LDAP server are not supported.
4. "Returns error" means that the operation is not performed (the system returns an error).
5. For a role to be created in both the LDAP server *and* the Sites database, at least one site must exist in the LDAP server. If no site exists, the role is created only in the Sites database.

Operations with Hierarchical-Schema LDAP Servers

Table C-4 summarizes system response to site and role management operations. The operations are performed by a WebCenter Sites administrator using the interfaces of three WebCenter Sites systems, each integrated with one of the following LDAP options:

- An LDAP-2 server with read-only permissions for the LDAP user (defined on page 543)
- An LDAP-2 server with write permissions for the LDAP user
- An LDAP server other than LDAP-2, in which case write operations are not supported

The results of each operation are described on the pages that are noted in the left-hand column of Table C-4.

Table C-4: System Response to Site and Role Management Operations with Hierarchical-Schema LDAP (WebCenter Sites Web Applications)

Operation in WebCenter Sites	Result	System Response With Hierarchical-Schema LDAP: LDAP-2: User has Read-Only Permissions [1]	LDAP-2: User has Write Permissions [2]	Not LDAP-2: Write not Supported [3]
Creating a Site (p. 554)	*Site is created in:*	Sites database	Sites database	Sites database
Editing a Site (p. 554)	*Site is edited in:*	Sites database	Sites database	Sites database
Deleting a Site (p. 555)	*Site is deleted in:*	Sites database	Sites database	Sites database
Creating a Role (p. 557)	*Role is created in:*	Sites database	Sites database	Sites database
Editing a Role (p. 557)	*Role is edited in:*	Sites database	Sites database	Sites database
Deleting a Role (p. 558)	*Role is deleted in:*	Sites database	Sites database	Sites database
Granting Users Access to Sites (p. 555)	*Access is granted in:*	Returns error[4]	LDAP	Unpredictable
Removing Users' Access to Sites (p. 556)	*Access is removed from:*	neither the Sites database nor the LDAP server	neither the Sites database nor the LDAP server	neither the Sites database nor the LDAP server

1. Sites and roles cannot be stored in the LDAP server (the WebCenter Sites administrator is connected to the LDAP server as a user without write permissions).
2. Sites and roles can be stored in the LDAP server (the WebCenter Sites administrator is connected to the LDAP server as a user with write permissions).
3. The LDAP server does not support the LDAP-2 protocol (such as Netscape Server). For non-compliant LDAP servers, write operations from the administrator's interface to the LDAP server are not supported.
4. "Returns error" means that the operation is not performed (the system returns an error).

Creating a Site

Operation

The WebCenter Sites administrator attempts to create a site by using the “Sites” node on the “Admin” tab.

> **Note**
>
> If you are manually creating a site in an LDAP-integrated system, you must ensure that the database and the LDAP server are synchronized. Otherwise, sites will not be properly listed in the WebCenter Sites interfaces.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system creates the site in the WebCenter Sites database. The LDAP administrator must create the site entry in the LDAP server.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Flat Schema

1. LDAP server with Read-Only Permissions—The system creates the site in the WebCenter Sites database. The administrator must create an entry for all available roles by prefixing *each* site name to the role name in the LDAP server.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Editing a Site

Operation

The WebCenter Sites administrator attempts to edit the description of a site by using the “Site Edit” option on the “Admin” tab.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system modifies the site description in the WebCenter Sites database. (The site description is stored only in the WebCenter Sites database, even with LDAP integration.)
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Flat Schema

1. LDAP-2 Server with Read-Only Permissions—The system modifies the site description in the WebCenter Sites database. (The site description is stored only in the WebCenter Sites database, even with LDAP integration.)

2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Deleting a Site

Operation

The WebCenter Sites administrator attempts to delete the description of the site by using the "Site Edit" node on the "Admin" tab.

> **Note**
>
> If you are manually deleting a site with LDAP integration, you must ensure that the database and the LDAP server are synchronized. Otherwise, sites will not be properly listed in the WebCenter Sites interfaces.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system deletes the site from the WebCenter Sites database. The LDAP administrator must manually delete the site entry from the LDAP server.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Flat Schema

1. LDAP-2 Server with Read-Only Permissions—The system deletes the site from the WebCenter Sites database. The LDAP administrator must manually delete the site entry from the LDAP server.
2. LDAP-2 Server with Write permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Granting Users Access to Sites

Operation

The WebCenter Sites administrator attempts to grant users access to a site by using the "Site > Users" nodes on the "Admin" tab.

> **Note**
>
> WebCenter Sites displays only the roles that are stored in its database.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message because, in the LDAP server, it cannot assign any roles to the specified users of the selected site. The LDAP administrator must manually assign the roles.
2. LDAP-2 Server with Write Permissions—The system assigns roles, within the LDAP server, to the specified users of the selected site.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable.

Flat Schema

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message because, in the LDAP server, it cannot assign any roles to the specified users of the selected site. The LDAP administrator must manually assign the roles.
2. LDAP-2 Server with Write Permissions—The system assigns roles, within the LDAP server, to the specified users of the selected site.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable.

Removing Users' Access to Sites

Operation

The WebCenter Sites administrator attempts to remove user access to sites by using the "Site > Users" nodes on the "Admin" tab.

> **Note**
>
> WebCenter Sites displays only the sites that are stored in its interface.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The LDAP administrator must manually remove the user's permissions to the sites, directly in the LDAP server.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Behavior is unpredictable.

Flat Schema

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message. The LDAP administrator must manually remove the user's permissions to the sites, directly in the LDAP server.
2. LDAP-2 Server with Write Permissions—The system removes the user's permissions to the sites, directly in the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable.

Creating a Role

Operation

The WebCenter Sites administrator attempts to create a role by using the “Roles” node on the “Admin” tab.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system creates the role in the WebCenter Sites database. The LDAP administrator must manually create the role in the LDAP server.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Flat Schema

1. LDAP-2 Server with Read-Only Permissions—The system returns an error message; it cannot create the role in the LDAP server or in the WebCenter Sites database. The LDAP administrator must create the role in the LDAP server. The same role must be re-created in the WebCenter Sites database.
2. LDAP-2 Server with Write Permissions—The system creates the role in the WebCenter Sites database. The system also creates the role for all available sites in the LDAP server by pre-fixing the name of each existing site to the name of the role (*SiteA-Role*, *SiteB-Role*, and so on).

Note

For a role to be created in both the LDAP server *and* the Sites database, at least one site must exist in the LDAP server. If no sites exist, the role is created only in the Sites database.

3. Not LDAP-2: Write not Supported—Behavior is unpredictable. An error message can be returned.

Editing a Role

Operation

The WebCenter Sites administrator attempts to modify the description of a role by using the “Roles” node on the “Admin” tab.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system modifies the description of the role in the WebCenter Sites database. (The role description is stored only in the WebCenter Sites database, even with LDAP integration.)
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).

3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Flat Schema

1. LDAP-2 Server with Read-Only Permissions—The system modifies the description of the role in the WebCenter Sites database. (The role description is stored only in the WebCenter Sites database, even with LDAP integration.)
2. LDAP-2 Server with Read-Only Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Deleting a Role

Operation

The WebCenter Sites administrator attempts to delete a role by using the "Roles" node on the "Admin" tab.

System Response

Hierarchical Schema

1. LDAP-2 Server with Read-Only Permissions—The system deletes the role from the WebCenter Sites database. The LDAP administrator must manually delete this role from the LDAP server.
2. LDAP-2 Server with Write Permissions—Responds as above (scenario 1).
3. Not LDAP-2: Write not Supported—Responds as above (scenario 1).

Flat Schema

1. LDAP server with Read-Only Permissions—The system returns an error message; it cannot delete the role from either the LDAP server or the WebCenter Sites database. The LDAP administrator must create the sites and roles in the LDAP server. The same sites and roles must be re-created in the WebCenter Sites database.
2. LDAP-2 Server with Read-Only Permissions—The system deletes the role from both the WebCenter Sites database and the LDAP server.
3. Not LDAP-2: Write not Supported—Behavior is unpredictable. The system can return an error message.

Index of Procedures

Index

B

C

D

E

F

G

H

Q

R

S

T

X