Sitecore CMS 6

What's New?

An Overview of the New Functionality in Sitecore CMS 6

Table of Contents

Chapter 1

Introduction

Welcome to Sitecore CMS 6!

This document introduces new features in Sitecore 6 from the perspective of someone familiar with Sitecore 5.3. As someone who is familiar with Sitecore, you're going to love Sitecore 6! This release builds upon the solid architecture that you already know, but makes a number of improvements and refinements that make Sitecore even more productive and intuitive to use.

The documentation team has put together an extensive set of reference materials and cookbooks that describe how Sitecore 6 works in detail. This "What's New?" document, however, describes the changes made between Sitecore 5.3 and Sitecore 6. These will help you quickly upgrade your knowledge and skills so that you can begin using Sitecore 6 immediately.

Be sure to check the corresponding reference materials if you'd like more information about a feature described in this guide.

This document contains the following chapters:

Chapter 1 – Introduction
This chapter contains a brief introduction to this manual and its intended audience.

Chapter 2 – Content Definition Improvements
This chapter contains a description of the changes made to Sitecore's content definition features.

Chapter 3 – Productivity Enhancements for Business Users
This chapter contains a description of the improvements that have been made to the Page Editor. Sitecore 6 introduces inline editing and other features that improve business user productivity.

Chapter 4 – Security Related Improvements
This chapter contains details of the various improvements that have been made to the Sitecore security model.

Chapter 5 – Additional Application Enhancements
This chapter contains a description of the major structural improvements that have been made to increase performance and ease of use in Sitecore 6.

Chapter 6 – Miscellaneous Changes
This chapter contains details of the additional changes that have been made in Sitecore 6.

Chapter 2

Content Definition Improvements

This chapter describes the changes made to Sitecore's content definition features.

Sitecore 6 has expanded the role of templates, introduced two new template types, improved upon the list of standard field types, and retired the concept of masters. This chapter describes all these changes in detail.

This chapter contains the following sections:

- Templates
- Improved Field Types
- Branch and Command Templates

2.1 Templates

Sitecore 6 changes the definition of the term “template” to refer to any component used to create items.

In previous versions of Sitecore, content authors created subitems based on a list of masters assigned to a given item. Sitecore 6 supports the concept of assigning templates directly to an item, without the need of the intermediate step of creating a master. Content authors can then create items based on assigned templates.

Sitecore 6 retires the concept of masters. The functionality previously provided by masters is now available using templates.

Sitecore 6 includes three types of template: data templates, branch templates, and command templates. Each of these is described in more detail in the following sections.

2.1.1 Data Templates

What was referred to in Sitecore 5 as a “template” is now called a “data template”. Assigning a data template to an item means that content authors will be allowed to create items directly from the assigned template.

$name and Other Tokens

In Sitecore 6, developers can set default values in the data template standard values item as before, although standard values now support tokens such as $name, previously only supported in masters. Tokens such as $name, when provided as standard field values, are processed during item creation and the value copied into the target item’s field. Changes to the item’s field name will not automatically change fields with a standard value set to $name.

For field values in standard values, we support: $name, $id, $parentname, $parentid, $date, $time, $now, and any dynamic variables that have been setup using the ReplacerContext (typically accessed through the ReplacerContextSwitcher).

2.1.2 Organized Template Fields

Sitecore 6 organizes the list of field types displayed to make it easier for users to find the type they need and to understand how a specific type works.

The field type categories are:

- **Simple Types** — Fields that store simple data, such as simple and rich text, yes or no values, dates and times, and references to images and other files in the Media Library.
- **List Types** — Fields that offer a list of choices and store one or more selected options.
- **Link Types** — Fields that link to other pages, whether internal or external.
- **Developer Types** — Fields that are typically used by Sitecore internal developers or developers who are customizing the Sitecore user interfaces.
- **Deprecated Types** — Fields that existed in previous versions of the Sitecore, but have now been superseded by new field types.
- **System Types** — Fields intended for internal use only.

2.2 Improved Field Types

Sitecore 6 changes the name of some field types to make them easier to understand.

2.2.1 New Field Type Names

The following table lists the field types as they were in previous versions of Sitecore and the new field types that should be used in their place. While these field types still exist, Sitecore now recommends using the new field type names as soon as possible.

Sitecore 5	Sitecore 6
html	Rich Text
link	General link
lookup	Droplink
memo	Multi-Line text
reference	Droplink
server file	None
text	Single-Line Text
tree	Droptree
tree list	Treelist
valuelookup	Droplist

The complete list of field types includes:

Simple Types

- **Checkbox** — Can be only one of two possible values, such as Yes/No. Stores “1” if the checkbox is selected, blank otherwise, although any other value is interpreted as not checked.
- **Date** — Used to store dates using the format “yyyyMMdd”
- **Datetime** — Used to store both a date and time in the format “yyyyMMddTHHmmss”
- **File** — Used to store links to files in the media library.
- **Image** — Used to upload images into the media library and/or choose from existing media library images.
- **Multi-Line Text** — Used to store simple multi-line text entries with no rich text support
- **Password** — Used to store text content that is masked in the Content Editor
- **Rich Text** — Used to store Rich-text supported data.
- **Single-Line Text** — Used to store a single line of text, with no rich text support.

List Types

- **Checklist** — Shows a list of items as a list of multiple select checkboxes. Stores the GUID of selected checkboxes in a pipe (“|”) separated list.

- **Droplist** — Shows a list of items as a single select drop list. Stores the name of the item selected.
- **Grouped Droplink** — Shows a list of items as a single select drop list with options grouped into sections. Stores the GUID of the item selected.
- **Grouped Droplist** — Shows a list of items as a single select drop list with options grouped into sections. Stores the name of the item selected.
- **Multilist** — Allows users to select multiple values from a list of Items. Stores the GUID of selected items in a pipe (“|”) separated list.
- **Treelist** — Allows users to select multiple items from a directory tree structure. Stores the GUID of selected items in a pipe (“|”) separated list.

Link Types

- **Droplink** — Shows a list of items as a single select drop list. Stores the GUID of the item selected.
- **Droptree** — Shows an entire section of tree structure allowing single selection. Stores the GUID of the item selected.
- **General Link** — Shows a set of specialized link options to help users create links to other areas of the site, email addresses, and external URLs.

Developer Types

- **Icon** — Used for ‘themed’ icons which are provided by Sitecore.
- **Iframe** — Used to integrate external pages into the Sitecore client.
- **Tristate** — Provides a droplist with values Default, Yes, and No. Stores a blank, “1”, or “0”.

System Types

- **Attachment** — Stores uploaded files in the database. For internal use only.
- **Field Source** — Internal field, used by the Sitecore client. For internal use only.
- **Internal Link** — For links to internal Sitecore Items. For internal use only.
- **Layout** — Inserts a subsection similar to the one you see in the Layout section. For internal use only.
- **Reference** — Creates a field which allows a user to select an Item from a content tree. For internal use only.
- **Security** — Internal field, used by the Sitecore client. For internal use only.
- **Server File** — For links to local server files. For internal use only.

Deprecated Types

These field types, still with their old names, continue to be supported in the Deprecated Types category, but this is for backward compatibility and the use of these field types should be avoided if at all possible.

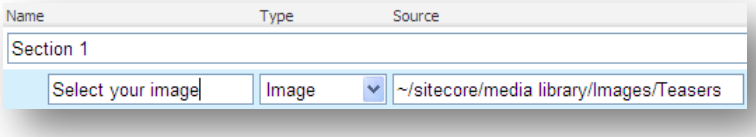
- **html** — use Rich Text.
- **lookup** — use Droplink.

- **tree** — use Droptree.
- **link** — use General Link.
- **valuelookup** — use Droplist.
- **memo** — use Multi-Line Text.
- **text** — use Single-Line Text.

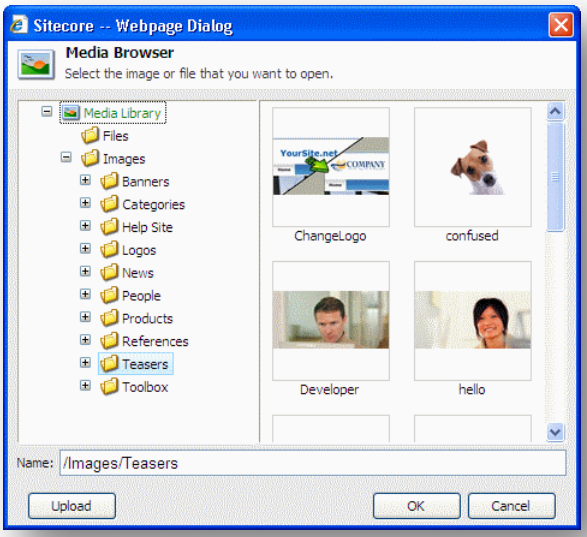
2.2.2 Improved Source Field Attribute

In cases where the Source field attribute allows the developer to specify the default folder to open (such as for Image fields), Sitecore 6 supports a tilde (~) prefix to the provided path. If the tilde prefix is provided, the dialog box will open with the desired folder selected, but will provide the user access to the entire content tree.

In the following example, you can see that in the first image we have defined an “Image” type field and in the source property we have specified the path prefixed with the tilde (~) character.



This displays the normal image select window. However, instead of the root of the available selections being the path, it displays the whole content tree of the media library with the root path as the highlighted selection, but not as the root, as in the following image:



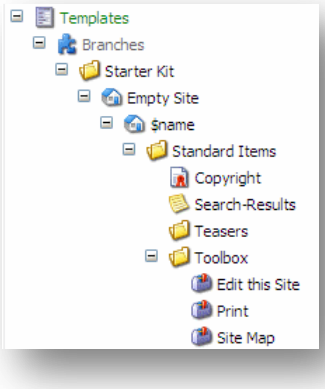
2.3 Branch and Command Templates

Sitecore 6 provides two new template types: Branch Templates, which define a set of items to be copied during item creation, and Command Templates, which define a class and method to be called during item creation. The following sections describe each in more detail.

2.3.1 Branch Templates

A branch template includes a definition item, used to assign security to branch templates, plus any number of immediate subitems and descendants, all of which will be copied during an insert operation.

The following image shows the “Empty Site” Branch template from the Sitecore Starter Kit. You can see that the root item of the Branch template is called “$name”. This means that on creation this will ask the user for a name, which it will use as the root name for the item. Underneath the “$name” item, you can see a variety of folders and Items, which will be created below the $name root to create an entire subset of content below the newly created item.



Branch subitem names will be preserved when copied during an insert operation except where $name is used.

Branch templates resemble masters in that Sitecore 6 copies the subitems of the branch definition item during item creation, similar to the way that Sitecore 5 copied the master item and any subitems during item creation. However, unlike Sitecore 5 masters, a branch template can create multiple items at the same level.

RETIRED: Masters

In Sitecore 6, the Branch Template replaces the concept of a master. This change simplifies the process of learning and using Sitecore, because developers no longer need to learn about masters immediately in order to begin working with Sitecore. Developers need only learn about branch templates when they need this functionality.

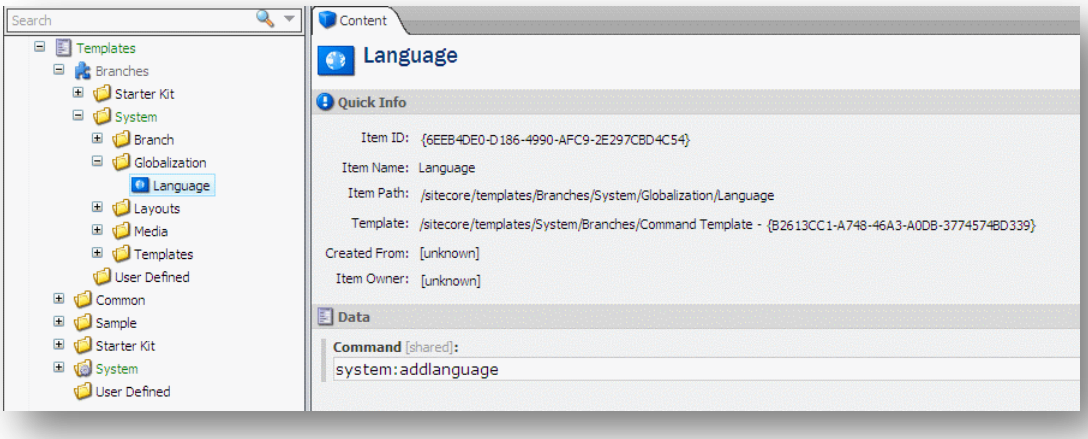
Developers can create a branch with a single sub-item to implement the same basic functionality of a master. This would only be necessary if the developer truly needs to copy initial field values into a created item rather than allowing the item to inherit the field value via template standard values. In most cases, developers only need to define data templates.

2.3.2 Command Templates

Command templates define a class and method to be called during an insert operation.

The intention is that the called method will open a wizard or dialog which will collect information and then create items programmatically.

In the following image, you can see the “Language” command template used when creating new languages. In the right-hand pane, you can see the “Command” field which contains the name (“system:addlanguage” in this case) of the command that is called when the command template is invoked.



2.3.3 Insert Options

Sitecore 6 has changed some of the UI labels and terminology associated with creating items:

- All menus previously labeled New, such as the Home ribbon’s **New** group, are now labeled Insert.
- The Configure ribbon’s **Masters** group used to assign masters to an item, is now labeled Insert Options.

The Insert Options associated with an item influence the templates shown in Insert menus. The insert options associated with an item can include a set of insert rules.

2.3.4 Insert Rules

An insert rule defines a class and method to be called as Sitecore constructs the list templates to include in the Insert menu associated with a selected item. The insert rule code may trim or expand the list of templates provided based on business requirements.

Chapter 3

Productivity Enhancements for Business Users

Sitecore 6 provides a redesigned Page Editor with in-line editing and other features that improve business user productivity.

Furthermore, Sitecore provides XSLT and .NET controls that both automatically provide the in-line editing controls, and make it simple for Web site developers to extend their controls to provide access to information and features that only appears while editing a page in the Page Editor. Developers can also configure Sitecore to allow business users to make limited design changes to a Web site without involving IT staff.

This chapter describes such improvements, as well as enhancements to the Content Editor and Developer Center in the following sections:

- The New Page Editor
- Content Editor Enhancements
- Developer Center Enhancements

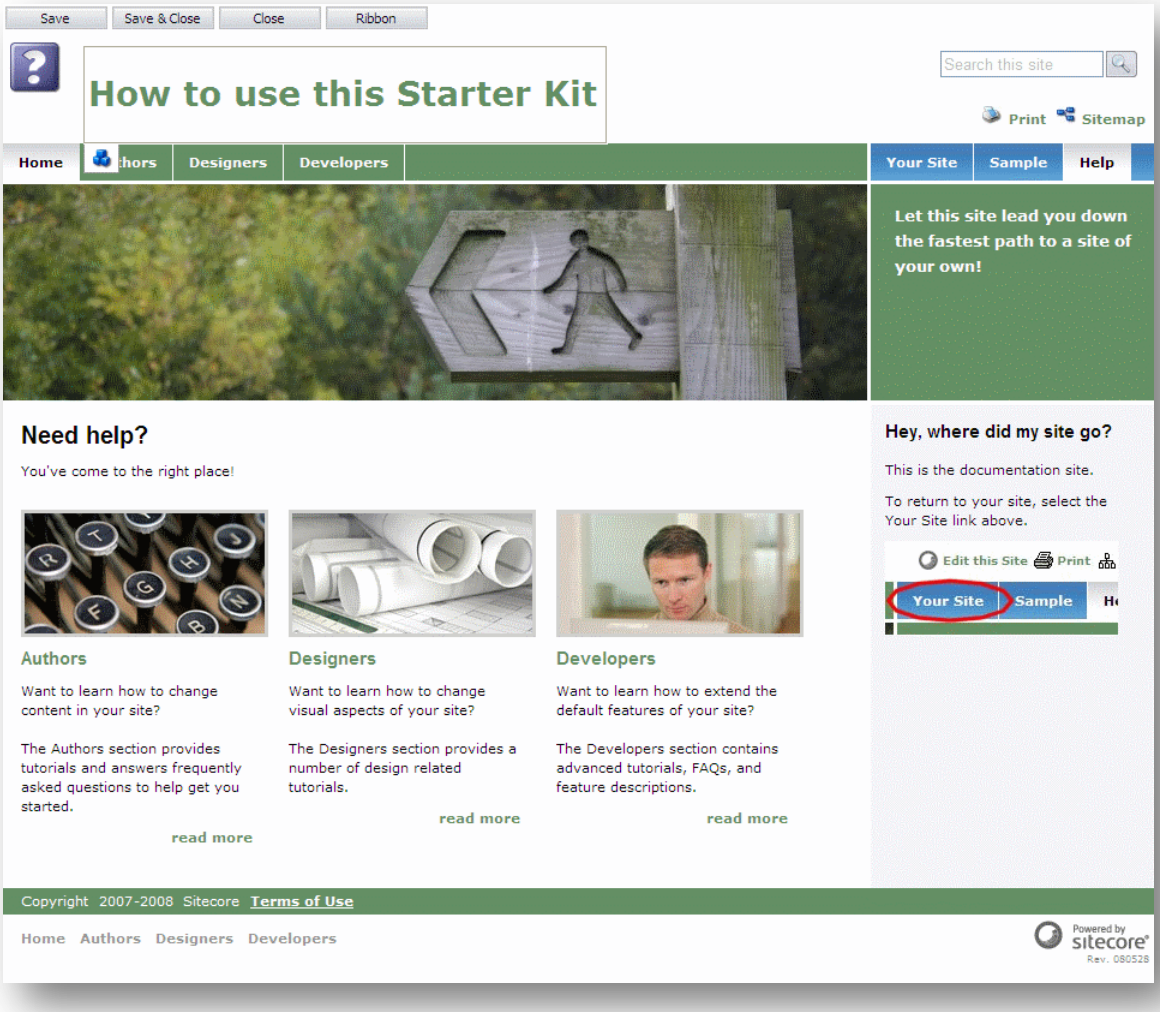
3.1 The New Page Editor

Sitecore 6 replaces the WebEdit client with the new Page Editor client. The Page Editor client supports inline editing to provide a more intuitive and productive environment for business users.

Once logged in to the client, a user sees flyover boxes appear over editable regions of the current page. The user can edit the text included inside such a flyover box directly. The flyover boxes may also provide content specific commands, such as the ability to choose an image or a link destination.

The Page Editor automatically locks any items that the user changes and denies access to any content locked by another user.

The following image shows the Help home page from the starter kit. Around the main title “How to use this Starter Kit” you can see a flyover box associated with the site name.



There is also support for editing fields that are currently blank and therefore do not generate tags on the published Web site, in Edit mode these field tags will appear of the Page Editor giving access to them for editing by a Page Editor.

The Page Editor also supports a collapsible ribbon (similar to the Content Editor ribbon) with advanced features.

The Page Editor can provide at least three functionality levels.

- **Normal** — Containing a full set of features designed for professional content authors provided by assigning the user the Sitecore Client Authoring role, required for any user who will use the Page Editor.
- **Limited** — Assigning the Sitecore Limited Page Editor role to the same user limits the functionality provided in the Page Editor to a simpler user interface appropriate for users with limited editing skills and requirements.
- **Minimal** — Alternatively, administrators can assign the Sitecore Minimal Page Editor role to the same user to limit the functionality available in the Page Editor to the bare minimum required to change pages. This option is intended for infrequent users or those with limited computer skills who require editing access to areas of the Web site.

The Page Editor also provides support for workflow operations (for the normal and limited functionality levels) and automated workflow operations (for the minimal functionality level). The minimal set of workflow features hides Workflow features in the Page Editor. There are also new workflow features such as the __**On Save** command and **Auto Submit Action**. These make it possible to configure automatic workflow behavior for users with minimal page editor features.

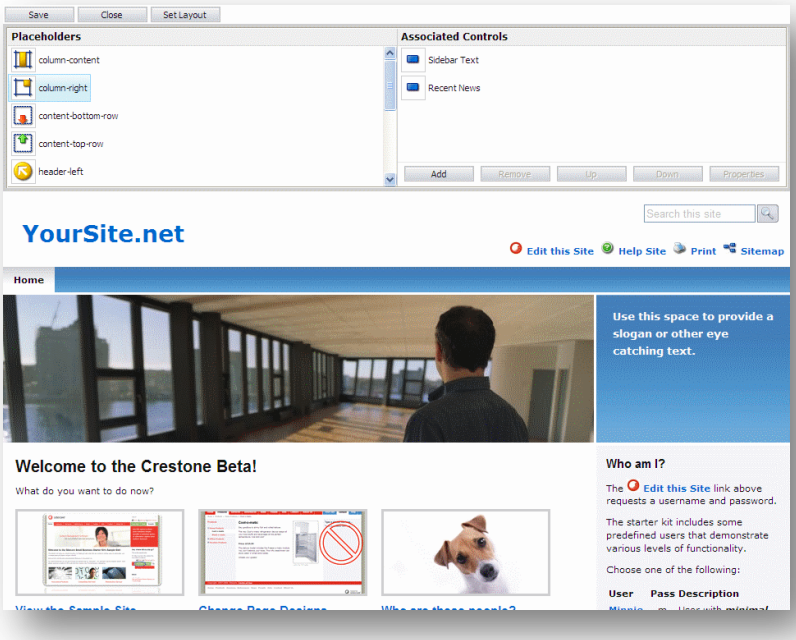
3.1.1 XSL Elements Support Page Editing

The Sitecore 6 Page Editor automatically makes content editable when an XSL rendering uses the sc:text, sc:image, sc:link, and related controls. However, to add flexibility developers may turn this off using the **disable-web-editing=”true”** parameter within the control.

3.1.2 The Page Editor Design Pane

The Page Editor Design Pane allows appropriately privileged users to change the design details of a page visually from within the Page Editor. Users require the Sitecore Client Designing role to access the Design Pane. Sitecore introduces Placeholder Settings to allow developers to configure which placeholders the Design Pane provides access to and which controls users may assign to each placeholder.

In the following image, you can see the Page Designer open with a variety of Placeholders and the associated page controls.



3.1.3 Configuring Placeholder Settings

In Sitecore 6, it is possible to associate various settings for placeholders in layouts and sublayouts. Using placeholder settings you can configure which placeholders and controls the Design Pane reveals to the current user. This includes the icon and flyover help associated with each placeholder in the placeholder list.

Security based permissions can also be used to assign controls to a given placeholder.

Sitecore automatically associates placeholders and placeholder settings based on the placeholder key; however, this can be overridden via the layout details associated with a specific item. The placeholder setting items are stored in the */sitecore/layout/placeholder settings* folder.

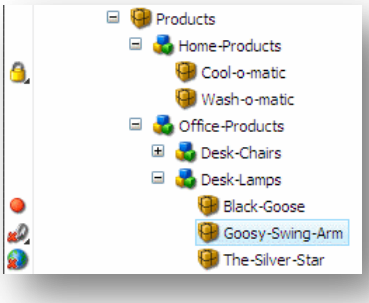
If you do not specify placeholder settings, the Page Editor displays the placeholder to all users and allows users to associate all controls (renderings and sublayouts).

3.2 Content Editor Enhancements

The Content Editor itself has been greatly improved to increase the amount of information getting to the author and thereby improving productivity. A variety of enhancements and improvement have been introduced to improve overall productivity for content authors.

3.2.1 Quick Action Bar

The Content Editor now displays a Quick Action Bar to the left of the content tree. This Quick Action Bar displays information about the corresponding item via a variety of icons. When a Quick Action Bar icon is displayed a simple right click on the quick action bar displays a popup menu which allows the user to change whatever information the bar displays. The following image shows the “Products” section of the content tree of the Starter Kit. You can see four Quick Action Bar icons, which are: the “Locked Items” icon, the “Missing Versions” icon, the “Broken Links” icon and the “Never Publish” icon.



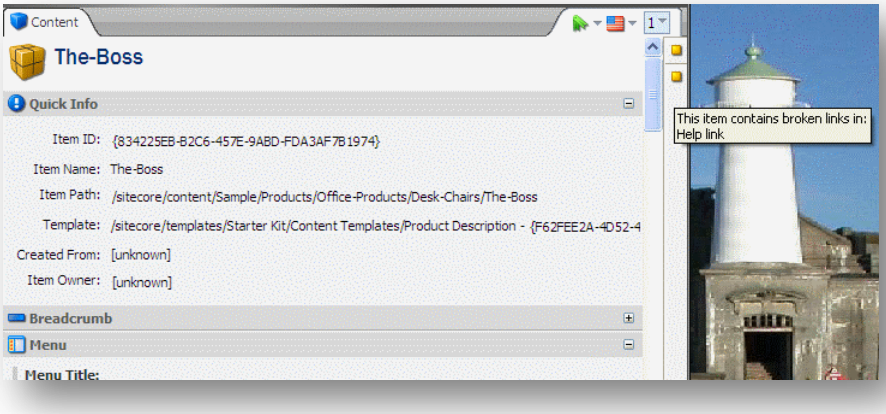
Note
Information icons in the quick action bar reflect the state of the latest version of the content to which they relate.

The information options available in the quick action bar include:

- **Broken Links** — Displays a broken link icon (“ ”) beside items that have broken links in any fields. Moving the mouse over the icon indicates which fields have broken links.
- **Locked Items** — Displays a lock icon (“ ”) beside all locked items. A click on the lock unlocks the item if the current user owns the lock.
- **Missing Versions** — Displays a red dot icon (“ ”) when the item is missing versions in a supported language. Moving the mouse over the icon indicates which languages are missing versions.
- **My Locked Item** — Displays a “my locked item” icon (“ ”) beside all items locked by the current user. A click on the lock unlocks the item.
- **Never Published Items** — Displays a do not publish icon (“ ”) beside items marked as not publishable.
- **Workflow State** — Displays the icon (“ ”) associated with the current item’s workflow state. A click on the icon displays a popup menu of workflow commands available in that state.

3.2.2 Validation Bar and Predefined Validation Rules

The Sitecore 6 Content Editor displays a color coded validation bar to the right of the field area. In the following image, you can see the starter kit item “The Boss” with yellow validation warnings and the tooltip help for the warning.

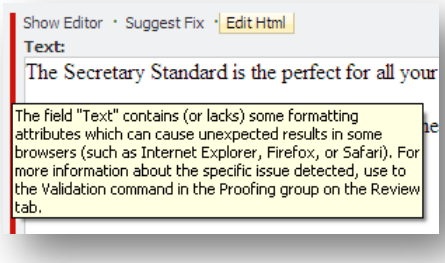


The Validation bar displays a color coded box at its top and can display the following colors:

- **Green** — Indicates that the current item has no associated warnings or errors.
- **Yellow** — Indicates that the current item has associated warnings, but no associated errors.
- **Red** — Indicates that the current item has associated errors (and potentially warnings as well).

The validation bar may display color or icon indicators along the height of the bar to indicate the relative position of the field associated with a given warning or error.

Each field also has a validation indicator bar displayed to the left of the entire field area which displays a color corresponding to the fields current state (gray for no errors, yellow for warnings, red for errors). In the following image, you can see a field validation warning (the red bar), with the tooltip explanation displayed, which shows more details regarding the error.



Sitecore 6 includes a number of predefined validation rules, as well as support for custom defined validation rules. Validation rules are also integrated with workflows. Developers can define the validation rules to be run which associate warnings and errors with an item, a field, or globally.

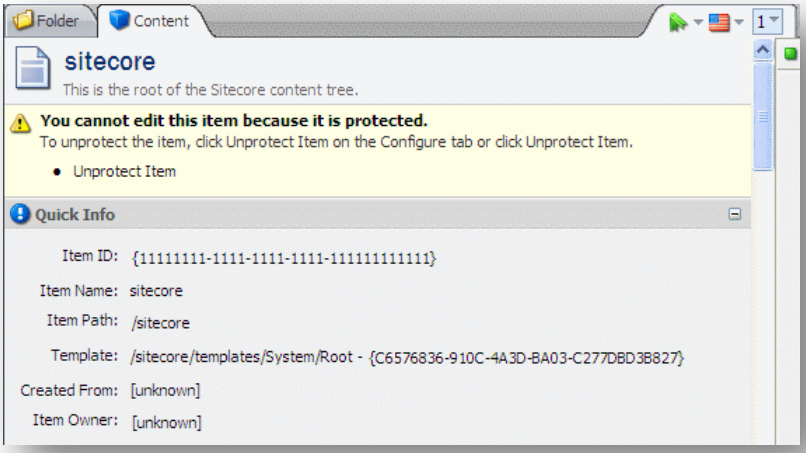
Also, validation rules have a number of execute options depending on their resource usage. These are:

- **While typing** — Validation acts on the entries as the entries are being made.
- **When leaving the field** — Validation acts when a field loses the focus, validating the data within that field.

- **When saving the item** — Validation runs when the item is saved.
- **When performing a Workflow** command — Validation runs when a workflow command is executed.

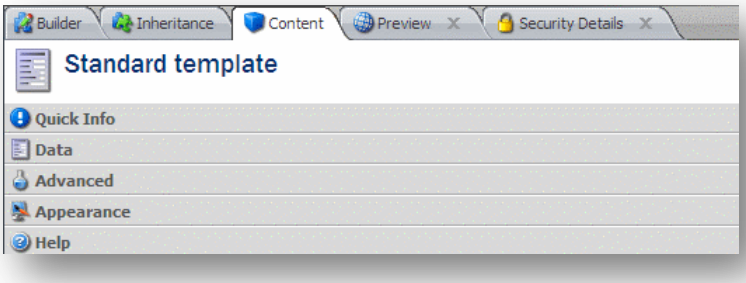
3.2.3 Item Related Warnings

The Content Editor now also displays appropriate warnings at the top of the field list to highlight important information to the content author. The following image shows the “sitecore” root item with the warning that the item is protected.



3.2.4 Content Editor Tabs

The Content Editor can now display one or more “tabs” in the field area of the editor. In the following image, you can see the tabs automatically open when you view the Standard template in the Content Editor:



The tabs available depend on the selected item. You can also create custom tabs to apply to items as required. Switching between tabs provides access to various editors associated with the selected item.

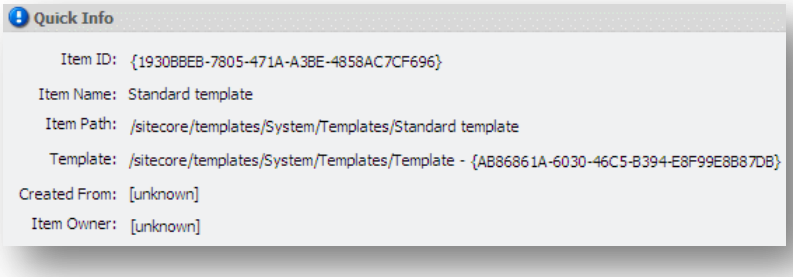
Common tabs include:

- **Builder** — Provided for data templates only. Supports creation and maintenance of fields and field sections for the selected data template.
- **Content** — Displays the content of the item selected with its field sections and fields.

- **Folder** — Provided for folders only. Shows folder related commands and subitems under the selected folder. These commands vary based on the folder type, but generally include the ability to create new subitems in the folder.
- **Inheritance** — Provided for data templates only. Displays a list of all base templates for the selected data template.
- **Grid Designer** — Provided for layouts and sublayouts only. Displays a tool used to define the layout and format of layouts and sublayouts.
- **Media Folder** — Provided for media folders only. Displays the media items listed under this item as subitems and allows for uploading and creation of subfolders.
- **Preview** — Previews the page the item is on with navigation facilities. Can be automatically opened using the **Preview** button on the **Presentation** tab.
- **Security Details** — Displays access rights that have been explicitly assigned to the selected item.

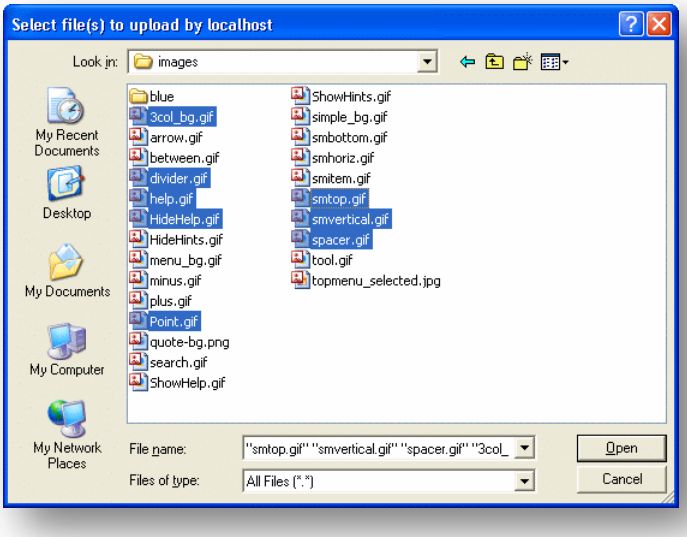
3.2.5 Quick Info Section

The Content Editor now displays information about the current item at the top of the fields section. The information includes: ID, key, path, template, and branch (if any). Shown for users assigned the Sitecore Client Developing role and Administrator users.



3.2.6 Multiple File Upload in the Media Library

The Media Library now supports the selection of multiple files in the content selection window.



3.2.7 Work In Progress Indicators in the Content Tree

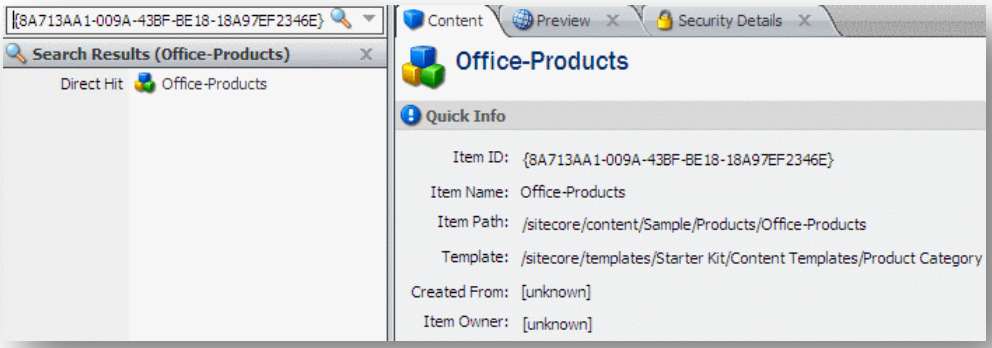
The Content Editor displays work in progress indicators while waiting for a response from the server when you open or select an item. These include:

- Hourglass to see items loading from the server ( ).
- Animation when opening tree branches to indicate processing ( ).

3.2.8 Search

A search facility has now been added to allow full searching of the content tree. This search facility allows for simple searching and customized searches with a filter system. It also supports searching for GUID's

and path entries in the search bar. In the following image, you can see a GUID entered into the search bar. Under this is the Direct Hit result from the search and to the right the Quick Info section of the item, clearly showing the GUID.



3.2.9 RETIRED: “Editable” Rich Text Editor Fields

The Sitecore 5 Content Editor previously provided an “Editable” option for Rich Text fields. This allowed users to change contents of Rich Text fields without opening a separate window, but had significant performance implications. This feature has been removed from the Content Editor in Sitecore 6.

3.3 Developer Center Enhancements

The Developer Center provides developers with in-browser tools for editing presentation components on a remote server. Sitecore 6 provides a new quick layout designer tool to simplify the initial development of sublayouts and a FieldRenderer control which can output content without coding.

3.3.1 Grid Tab

The Developer Center now displays a Grid tab when editing sublayouts. The tab provides a rapid development tool that simplifies the process of creating the structural aspects of sublayouts, such as the DIV and/or TABLE constructs used to position controls. Developers can easily add placeholders and static controls as well as edit the properties of any construct on the page.

While the Grid tab provides limited functionality to keep the interface simple, when the developer has finished an initial design, the Design and HTML tabs still offer access to advanced features.

3.3.2 FieldRenderer Web Control

Sitecore provides a FieldRenderer Web control which allows users with limited XSLT and .NET skills to output the value of a given field for the requested item or any data source by just adding the control and filling out a property sheet.

Chapter 4

Security Related Improvements

This chapter provides details of the various improvements that have been made to the Sitecore security model. It gives an overview of .NET security as opposed to the old Sitecore security model. It then goes on to discuss the various definitions in the security model including Domains, Providers, and Roles. It also gives a brief overview on the definition of an account and briefly details the new Security Explainer.

This chapter contains the following sections:

- Leveraging the .NET Security Model
- UI Enhancements
- Domains and Multi Domain Management
- Account Changes

4.1 Leveraging the .NET Security Model

Sitecore 6 replaces the standard Sitecore security model with the .NET security model. This provides a number of enhancements, including:

- The standard ASP.NET way of handling security.
- The ability to use plug and play security providers from Microsoft.
- Abstraction of data from the real data source.
- An easy option to replace or extend the default configuration with your own custom security providers.
- The possibility of using several providers simultaneously and thus keeping the accounts in identifiable storage areas.

The security model has been enhanced to allow roles in roles, a feature that is not available in the standard .NET security model.

These enhancements serve to allow the security of the Sitecore client to handle a lot more users, roles and domains than previously. Along with the new feature of roles in roles this serves to allow for scalability enhancements to support large scale security repositories.

4.1.1 Plug and Play Microsoft Providers

Sitecore's .NET Security model uses three pre supplied Plug and Play Microsoft Providers. These are:

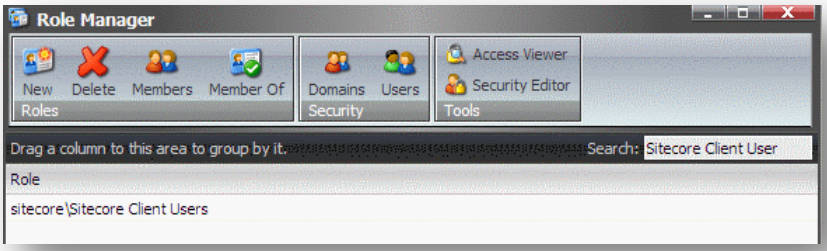
- **Membership provider** — The membership provider provides the interface with Sitecore Security data sources containing data regarding the registered users. It also provides methods for creating and deleting users, verifying login credentials, changing passwords and storing and retrieving membership information.
- **Role Provider** — The role provider provides the interface with data sources containing role data, mapping users to roles. It also provides methods for creating roles, deleting roles, adding users to roles, and so on.
- **Profile Provider** — The job of the profile provider is to write profile property values supplied by ASP.NET to Sitecore's profile data sources, and to read the property values back from the data source when requested by ASP.NET. It also implements methods that allow consumers to manage profile data sources-for example, to delete profiles that haven't been accessed since a specified date.

4.2 UI Enhancements

Sitecore 6 provides UI enhancements related to its security features and applications, including search features in the user and role manager applications and a new Security Explainer feature in the Access Viewer.

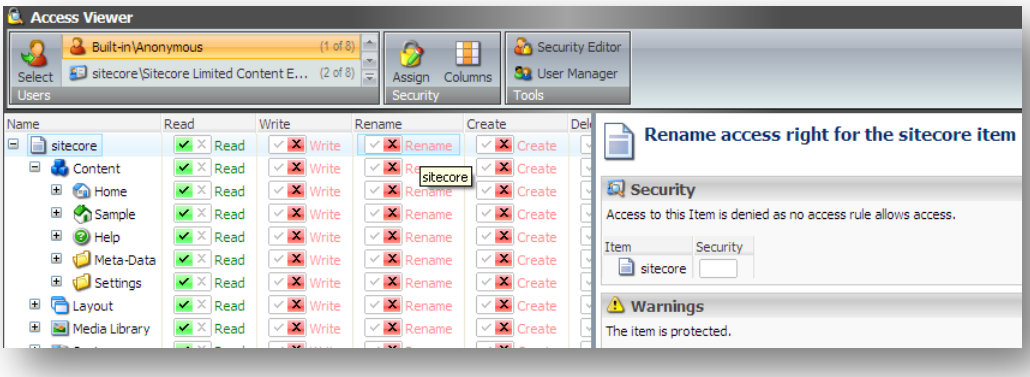
4.2.1 Search Functionality in User and Role Managers

Search functionality has been embedded in the user, role and domain managers to improve the ability to find users, roles and domains in large scale security setups. The following image shows the Role Manager. In the search bar, you can see the text that was entered and in the main window the results of the search.



4.2.2 Security Explainer in the Access Viewer

A new “Security Explainer” has been added to allow tracing of security within Sitecore. In the following image, you can see the Access Viewer opened and the Rename Access Right for the Sitecore root item selected. In the right-hand pane of the main window is the “Security Explainer” which shows how this access right has been set.



4.3 Domains and Multi Domain Management

The latest version of Sitecore comes with 3 predefined domains. These are:

- **Built-in** — A virtual “default” domain that only exists in memory. If a Web site does not specify a default domain, then unauthenticated visitors will appear as the **built-in\anonymous** user.
- **Extranet** — This is the standard Web site security domain. This domain defines who can access the information published on the Intranet, Extranet and/or Internet Web site.
- **Sitecore** — This is the normal internal security domain which handles security for the Sitecore clients.

4.3.1 Locally Managed Domains

Sitecore 6 also supports the concept of a “locally managed” domain. A locally managed domain is a domain whereby the users of that domain only see that specific domain that they are part of and not the other domains within the system. Locally managed domains are usually maintained by a Local Administrator, who also will not be able to see other domains within the system. Also, users defined in a locally managed domain will only see accounts that have been defined within their own domain. This simplifies the process of supporting multiple sites from within a single installation as each locally managed domain will be administered and used by accounts that cannot see the other domains.

Locally managed domains are part of “The Delegation Model” in Sitecore security. The delegation model includes additional features, such as:

- **Globally Visible Roles** — A list of roles that users in all domains can see.
- **Managed Domains** — A list of domains associated with a user, which allows users to see users in multiple locally managed domains.

You can define multiple domain names per site which saves caching, and for sites in live-mode the HTML caches can be set to be cleared no-more-often than every X seconds, for example, every 60 seconds — to avoid constant clears.

4.4 Account Changes

Sitecore collectively refers to users and roles as “accounts”. Thus, an “account” is either a user or a role.

4.4.1 Roles in Roles

“Roles in Roles” is the ability whereby a role may be a member of other roles. Users who are members of a role automatically inherit the membership of these additional roles. This is useful in administration for grouping commonly used roles together, rather than adding several roles to users. It also enhances the ability to provide for large scale security models.

4.4.2 Predefined Accounts

Sitecore 6 comes with a series of preinstalled users and roles. These are:

Preinstalled users

- **built-in\anonymous** — A virtual user which is assigned to unauthenticated users viewing a Web site that has no associated domain.
- **built-in\owner** — a virtual role that refers to the user referred to in an item’s current creator / owner field
- **extranet\anonymous** — An unauthenticated user who is viewing the default published Web site.
- **sitecore\anonymous** — An unauthenticated user who accesses the Sitecore client. This user only has access to the Sitecore login screen.
- **sitecore\Admin** — A predefined “Administrator” CMS user.

Predefined System Roles

- **-\everyone** — A virtual role that administrators can use to assign access rights to all users or all users in a specific domain. Everyone is available in all domains, plus as a global role.

Predefined Content Roles

- **Sitecore\Author** — Provides access to content in the content tree and automatically assigns membership to the Sitecore Client Authoring and Sitecore Client Users roles.
- **Sitecore\Designer** — Provides access to the areas of the content tree required to make design changes and automatically assigns membership to the Sitecore Client Designing and Sitecore Client Users roles.
- **Sitecore\Developer** — Provides access to the areas of the content tree required for development and automatically assigns membership to appropriate Sitecore Client roles.

New Sitecore Client Roles

- **Sitecore Client Account Managing** — This role allows a user to maintain users, roles, and domains through the use of the Access Manager, the Domain Manager, the Role Manager and the User Manager
- **Sitecore Limited Content Editor** — This role limits the amount of Content Editor functionality provided by the Sitecore Client Authoring role (which is still required for users given this role).

- **Sitecore Limited Page Editor** — This role limits the amount of Page Editor functionality provided by the Sitecore Client Authoring role (which is still required for users given this role).
- **Sitecore Minimal Page Editor** — This role restricts the amount of Page Editor functionality provided by the Sitecore Client Authoring role (which is still required for users given this role) to the absolute minimum.

Chapter 5

Additional Application Enhancements

The Sitecore 6 Recycle Bin has undergone major structural improvements to increase performance and ease of use, plus it has been joined by a twin application that provides similar functionality for archived items.

Sitecore 6 also offers improved taskbar based searching.

This chapter contains the following sections:

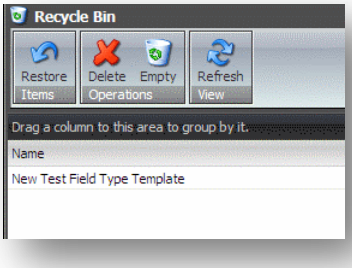
- Recycle Bin and the Archive
- Taskbar Quick Search

5.1 Recycle Bin and the Archive

Sitecore has improved the functionality of both the Recycle Bin and the Archive, removing individual databases in favor of a new model where each individual database contains its own Recycle Bin and Archive.

5.1.1 Recycle Bin Improvements

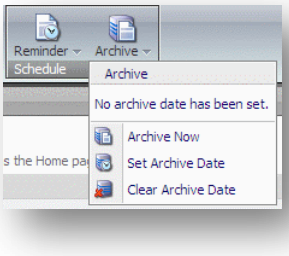
Sitecore 6 offers an improved Recycle Bin application. Deleted items are no longer organized by delete date. The application adds support for paging and a search facility. The Recycle Bin also respects security settings, so that users may only restore items for which they have access. The Recycle Bin now only shows items which have been deleted from the current database.



5.1.2 Archive Improvements

Sitecore 6 adds an Archive application which can retrieve archived items. Archiving can be done through the Archive button in the **Schedule** group of the **Review** tab in the Content Editor. In the following image, you can see the Archive button with the sub menu open showing the archiving options, which are:

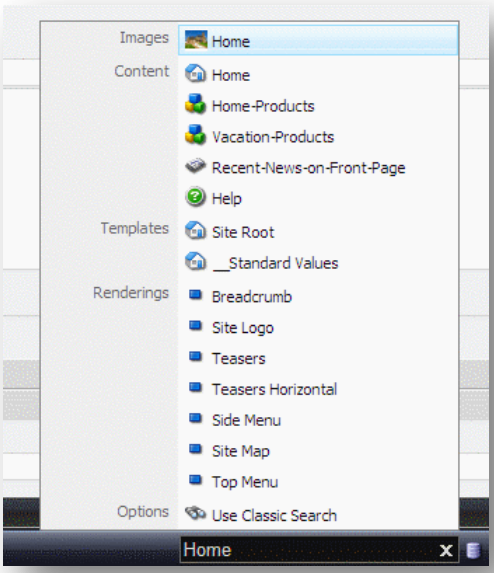
- **Archive Now** — Archives the Item
- **Set Archive Date** — Sets a future date and time to archive the item
- **Clear Archive Date** — Clears any set archive date.



The Archive application only shows items which have been archived from the current database.

5.2 Taskbar Quick Search

When using the Desktop user interface, CTRL+/ moves the cursor into the Taskbar search field and the search results are shown above the field. In the following image, you can see a taskbar search for the word “Home” with the results displayed in a results pane above the search bar.



Chapter 6

Miscellaneous Changes

This chapter provides details of additional miscellaneous changes provided in Sitecore 6.

This chapter contains the following sections:

- Database Changes
- Configuration Changes
- Performance Improvements

6.1 Database Changes

This section describes miscellaneous database related changes.

6.1.1 DB Consolidation

Sitecore 6 uses three databases by default, compared to the seven databases used by Sitecore 5 by default. The three databases are: Core, Master, and Web.

The “Sitecore” and “Extranet” security databases were removed as they are now handled by the .NET security model and stored in standard tables.

The “Archive” and “Recycle Bin” databases were removed as each database now has its own internal archive and recycle bin storage areas.

6.1.2 Field Types and Html Editor Profiles

Sitecore 6 stores field type and HTML editor profile definition items in the Core database.

6.1.3 Various System Paths Moved in the Core Database

Various system paths have been moved in the restructuring of the Sitecore model, these are described in the following table:

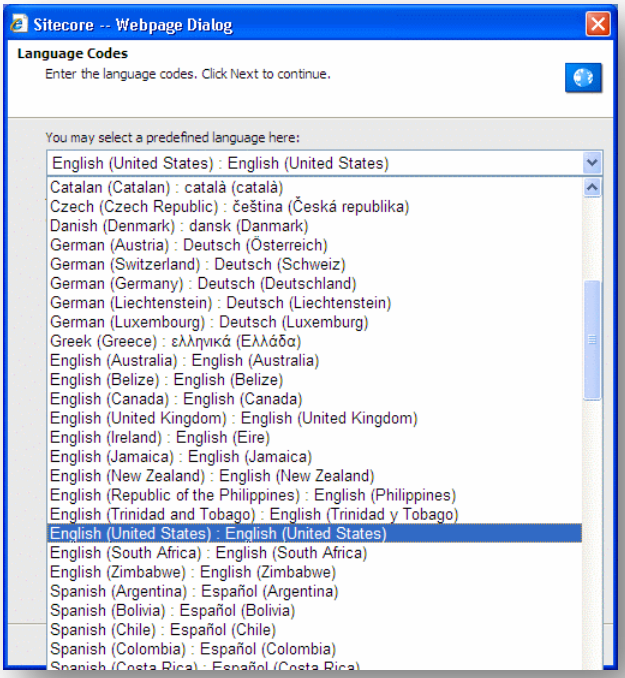
Old Path	New Path
/Applications/System/Tray	/Applications/Desktop/Tray
/Applications/System/Window Manager	/Applications/Desktop/Context Menus
/sitecore/system/Ribbons	/Sitecore/content/Applications/Content Editor/Ribbons
/Applications/Search	/Applications/Search/Search
/Applications/Dialogs/Rebuild Search Index	/Applications/Search/Rebuild Search Index
/Globals/Desktop Link Menu	/Applications/Desktop/Context Menus/Desktop Link Menu
/Globals/Desktop Background Menu	/Applications/Desktop/Context Menus/DesktopBackground Menu

6.2 Configuration Changes

Sitecore 6 introduces a number of miscellaneous configuration changes.

6.2.1 Improved Language Support

The Sitecore Import Languages Wizard now asks which languages, included in the specified file, to import. Sitecore 6 also supports the ability to select any ASP.NET defined language for editing in the Content Editor (not just the languages defined in the current database).



6.2.2 Workflow Changes

A new checkbox type field called **Suppress Comments** has been added as a field of the “workflow command” data template. If you select this checkbox, Sitecore suppresses the comment prompt window that is shown when a workflow command is executed.

Sitecore 6 introduces the __**OnSave** command, which does not appear as a workflow command in any user interface, but which is automatically triggered whenever a user saves an item in the corresponding state. The __OnSave command’s Next State field is ignored, but any actions assigned to the __OnSave command will be processed.

Sitecore 6 introduces two new default actions:

- **Auto Submit Action** — this action automatically moves an item to a new state if the action is triggered by a user that is a member of the specified role.
- **Validators Action** — this action checks whether an item has any issues reported by item validators and cancels the transfer of an item to the next state and displays an error to the user, if it does.

6.2.3 Spelling Dictionary Files

Sitecore 6 uses Telerik `.tdf` files instead of the old `.dic` files. For more information, see Telerik's Web site.

6.2.4 Web.config Patching with Include Config Files

Previous versions of Sitecore CMS forced administrators to make direct changes to configuration settings in the `web.config` file manually. This led to challenges locating local configuration changes as opposed to modifications made by Sitecore when upgrading to a new version of Sitecore.

Sitecore 6 offers a smart solution: `web.config` modifications can now be made in a separate XML file, stored under the */App_Config/Include* folder, which Sitecore reads in at startup time after loading the `web.config` file. The folder contains several example files which illustrate how to use this feature.

The Sitecore 6 configuration factory reads the include config files and overrides settings in the `web.config` file based on the settings in the include config file.

6.2.5 Audit Logging Entries Moved to Log File

Sitecore 6 has added a significant amount of audit information. However, Sitecore 6 does not generate audit files. Instead, Sitecore 6 writes all audit information to the standard daily log file.

6.2.6 RETIRED: Automatic Integration with W3C Tidy

W3C appears to have discontinued support for the Tidy component. Sitecore 6 therefore removes automatic integration with this component. Manual integration is still possible for those who which to use this feature.

6.3 Performance Improvements

Sitecore 6 introduces a number of miscellaneous performance improvements.

6.3.1 Media Streaming Enhancements for Large Files

Sitecore 5 read the entire media object from the database before streaming, which negatively impacted performance as the size of the media increased.

Sitecore 6 directly streams media from the database without the need to read the entire media object before streaming begins, thereby reducing performance overheads and allowing for large scale media databases without the previous loss in performance.

6.3.2 Enterprise Class Server Support

Sitecore 6 has added support for the following:

- 64-bit Windows.
- IIS 7
- IIS 7 Integrated mode support
- Windows Server 2008

6.3.3 Dynamic Links

In Sitecore 6, all internal links are stored as GUIDs, unlike Sitecore 5, which stored internal links as their text representations. This means that all operations in the link database have become significantly faster. All GUID links are resolved at runtime to their text representations when generating a Web page.

The links are resolved using a pipeline. Likewise, a reverse link (link -> guid) pipeline is also available for incoming HTTP requests. The new link pipelines support multilingual links. So a link to the same item can be represented in English, Danish, and so on. This greatly improves search engine optimization capabilities.

Furthermore, the default link generation is aware of subsites and can substitute a sub-site's domain name in place of a link path absolute to */Sitecore/content/….*

Default validators are provided which continuously checks for unique tree item names but which also checks for the uniqueness of display names whenever they are used to generate the link.

In IIS7 links can have any extension and no extension. For example, the following are all valid IIS 7 links:

- http://mysite.com/home/products.aspx
- http://mysite.com/home/products.html
- http://mysite.com/home/products

The `web.config` file contains a <linkManager> element which configures how Sitecore should generate links.

This element supports settings such as:

- addAspxExtension – true | false
- alwaysIncludeServerUrl – true | false

- encodeNames – true | false
- languageEmbedding – asNeeded | always | never
- languageLocation – filePath | queryString
- shortenUrls – true | false
- useDisplayName – false | true

6.3.4 Performance of SelectSingleItem vs. GetItem

SelectSingleItem can perform much better than **GetItem** in cases where a content tree has many items.