Sitecore CMS 6

Client Configuration Cookbook

Tips and Techniques for CMS Architects and Developers

Table of Contents

Chapter 1

Introduction

This cookbook provides tips and techniques for CMS architects and developers configuring Sitecore clients to optimize the Sitecore user experience. For more information about the topics described in this document, see the Client Configuration Reference manual.[1]

This document contains the following chapters:

- **Chapter 1 — Introduction**
  A brief introduction to this cookbook and a description of its intended audience.
- **Chapter 2 — Data Templates and Items**
  Procedures to optimize the editing experience for Sitecore users.
- **Chapter 3 — Data Validation**
  Provides procedures to configure data validation.
- **Chapter 4 — Simplify Repetitive User Operations**
  Procedures to optimize repetitive operations.
- **Chapter 5 — The Page Editor**
  Procedures to configure the Page Editor.

This chapter contains the following section:

- **Common Procedures**

[1] http://sdn5.sitecore.net/Reference/Sitecore%206.aspx.

1.1 Common Procedures

This section describes procedures used in various sections of this and other documents.

1.1.1 How to Select a Database in the Sitecore Desktop

To select a database in the Sitecore Desktop:

1. On the Sitecore login page (http://localhost/sitecore), click the Advanced tab, and then click the Desktop option.
2. Log in as an administrator. The Sitecore Desktop appears in the browser.
3. In the lower right corner of the Sitecore Desktop, click the database icon, and then click the database name. The Desktop refreshes, closing any open applications. If you open the Content Editor or other applications within the Desktop, they will access the database you selected until you log out or select a different database.

Important
To reduce the potential for making changes to a database unintentionally, always select the master database after working with one of the other databases.

Important
Unless otherwise specified, all procedures in this and other Sitecore documentation assume that you have selected the master database.

1.1.2 How to Show or Hide the Standard Fields

To show or hide the standard fields:

1. In the Template Manager or Content Editor, click the View tab.
2. In the View group, select or clear the Standard Fields checkbox.

Note
The Content Editor may not respond as quickly when showing the standard fields.

1.1.3 How to Show or Hide Raw Values

You may investigate the text value of a field by viewing its raw value. For example, you may view raw values to determine the attributes of a field such as image that stores a single XML element.

To show or hide raw values:

1. In the Template Manager or Content Editor, click the View tab.
2. In the View group, select or clear the Raw Values checkbox.

1.1.4 How to Enable the Developer Tab

The developer tab in the ribbon provides convenience features for developers working with Sitecore solutions. To enable the developer tab, in the Content Editor, right-click the tab strip, and then select the Developer option.

1.1.5 How to Copy the Path to an Item to the Windows Clipboard

To copy the path to an item to the Windows clipboard:

1. In the Content Editor, select the item.

2. In the edintg pane, click the Content tab.
3. In the Quick Info section, click and drag the mouse across the value of the Item Path property.
4. To copy the path to the Windows clipboard, press CTRL-C, or right-click the selected text and then select copy.

1.1.6 How to Enter a Class Signature

A class signature identifies a class in a .NET assembly.

To enter a class signature:

1. Enter the following prototype:

```
Namespace.Class,Assembly
```

2. Replace `Namespace` with the namespace containing the class.
3. Replace `Class` with the name of the class.
4. Replace `Assembly` with the name of the assembly containing the class, without the `.dll` extension.

Chapter 2

Data Templates and Items

This chapter provides tips and describes techniques for configuring data templates and items.

This chapter contains the following sections:

- **Item Appearance**
- **Data Template Sections**
- **Data Template Fields**
- **Item Editors**
- **Insert Options**

2.1 Item Appearance

This section provides procedures to control the appearance of items in the Content Editor.

2.1.1 Icons

This section provides tips for controlling the icons displayed for items in the Content Editor. The icon for each item is the relative or absolute path to an image file, which is stored in a field defined by the Sitecore standard template.

Note
Sitecore supports theming. By default, relative icon paths are relative to the `/sitecore/shell/themes/standard` directory under the document root associated with the web site. An icon of `applications/16x16/star_yellow.png` is the same as an icon of `/sitecore/shell/themes/standard/applications/16x16/star_yellow.png`.

Note
An icon can be an arbitrary URL of an image.

How to Set the Icon for an Item

To set the icon for an item:

1. In the Content Editor, edit the item.
2. Click the Configure tab.
3. In the Appearance group, click the Icon command. An icon selection menu appears.
4. In the icon selection menu, select an icon, or click More Icons and use the icon selection dialog.

Note
The icon selection dialog may load slowly when presenting a large number of images in a single directory.

Tip
As a shortcut to access the icon selection dialog, in the editing pane, click the Content tab, and then click the icon in the item title bar.

Important
Rather than setting the icon for individual items, set the icon for each data template to apply that icon to the data template, its standard values item, and all other items based on that template.

How to Set the Default Icon for All Items Based on a Data Template

To set the default icon for all items based on a data template:

1. In the Template Manager, edit the data template definition item.
2. Set the icon in the data template definition item as described in the section How to Set the Icon for an Item.

Important
Set the icon in the definition item for each data template, not in the standard values item for the data template. If you only set the icon in the standard values item for the data template, the icon does not apply to the data template itself; the icon shown for the data template in the content tree remains the default icon.

How to Change the Default Icon for All Items

To set the default icon for all items for which the item, its data template, and the standard values item associated with its data template do not define an icon, in the `web.config` file, in the `/configuration/sitecore/settings/setting` with the name `DefaultIcon`, set the `value` attribute to the default icon path.

How to Configure the Icon Selection Menu

To configure the icons that appear in the icon selection menu, edit the file `/App_Config/Icons.config`. Each collection element defines a row, and contains a pipe-separated list of absolute or relative paths to icon files.

2.1.2 Hidden Items

Hidden items do not appear in the content tree for users who do not have permission to show hidden items, or have not chosen to show hidden items.

How to Show or Hide Hidden Items

To show hidden items, the user must be an administrator, or a member of either the Sitecore Client Developing role or the Sitecore Client Maintaining role.

To show hidden items:

1. In the Content Editor, click the View tab.
2. In the View group, select or clear the Hidden Items checkbox.

How to Show or Hide an Item

To hide or show an item, the user must be an administrator, or a member of the Sitecore Client Configuring role.

To show or hide an item:

1. In the Content Editor, edit the item.
2. Click the Configure tab.
3. In the Attributes group, click the Hide Item command to hide the item, or click the Make the Item Visible command to show the item.

2.1.3 Protected Items

No user can edit protected items through Sitecore user interfaces.

How to Protect or Unprotect an Item

To protect or unprotect an item, the user must be an administrator or a member of the Sitecore Client Configuring role.

To protect or unprotect an item:

1. In the Content Editor, edit the item.
2. Click the Configure tab.
3. In the Attributes group, click the Protect Item command to protect the item, or click the Unprotect Item command to unprotect the item.

2.1.4 How to Configure the Style of an Item Name in the Content Tree

You can use the tree node style to configure the styling of item names in the content tree. To configure the tree node style for item names:

1. In the Template Manager or Content Editor, edit the standard values item or the individual item.
2. Click the Configure tab.
3. In the Appearance section, click the Tree Node Style command. A CSS style dialog appears.
4. In the CSS style dialog, enter CSS styles, or click the Edit button to use the CSS style wizard. For example, enter `COLOR:red;` to set the color of the font to red.

Note
Sitecore uses `COLOR:green` for system items and `COLOR:gray` for proxy shadow items. Avoid using these tree node styles for other purposes.

2.1.5 How to Set the Display Name for an Item

You may set the display name for an item to cause what the text that appears in the content tree to differ from the item name and hence its URL.

To set the display name for an item:

1. In the Content Editor, edit the item.
2. Click the Home tab.
3. In the Rename group, click the Display Name command. The display name prompt appears.
4. In the display name prompt, enter the display name for the item.

2.1.6 How to Set Item Context-Sensitive Help

To set context-sensitive help for a data template or an individual item:

1. In the Template Manager or Content Editor, edit the standard values item or the individual item.
2. Click the Configure tab.
3. In the Appearance group, click the Help command. The help properties dialog appears.
4. For Short Description, enter a short description of the type of item or the individual item.
5. For Long Description, enter a long description of the type of item or the individual item.

2.2 Data Template Sections

This section provides procedures to optimize the appearance of data template sections and fields.

2.2.1 How to Set a Data Template Section Icon

To set the icon for a data template section:

1. In the Template Manager, edit the data template section definition item.
2. Set the icon as described How to Set the Icon for an Item

2.2.2 Data Template Section Sort Order

You can sort data template sections using Template Builder, or by setting the sort order property of a data template section definition item.

Note
To control the order of sections when a data template and its base templates define different data template sections, set the sort order property of data template section definition items.

How to Sort Data Template Sections

To sort the data template sections in a data template:

1. In the Template Manager, select the data template definition item.
2. In the editing pane, click the Builder tab. The Template Builder appears.
3. Click the Builder Options tab.
4. In the Template Builder, click the mouse in the section name to select that section.
5. In the Sorting group, click the Up, Down, First, and Last commands to sort the section relative to the other sections in the data template.

How to Set the Sort Order Property of a Data Template Section Definition Item

To set the sort order property of a data template section definition item:

1. In the Template Manager, edit the data template section definition item.
2. Show the standard fields as described in the section How to Show or Hide the Standard Fields.
3. In the Appearance section, in the Sortorder field, enter a numerical value.
4. Hide the standard fields.

2.3 Data Template Fields

This section provides procedures to optimize the appearance of data template fields.

2.3.1 Data Template Field Headers and Context-Sensitive Help

The following sections provide procedures to define data template field headers including context-sensitive help.

How to Set the Title for a Data Template Field

To set the title for a data template field, causing the field label in the Content Editor to differ from the name of the field definition item:

1. In the Template Manager, edit the data template field definition item.
2. In the Data section, in the Title field, enter the title of the field.

How to Set Context-Sensitive Help for a Data Template Field

To set context-sensitive help for a data template field:

1. In the Template Manager, edit the data template field definition item.
2. Click the Configure tab.
3. In the Appearance group, click the Help command. The help properties dialog appears.
4. For Short Description, enter a short description of the field.
5. For Long Description, enter a long description of the field.
6. For Help Link, enter the URL of a resource containing helpful information about the field.

2.3.2 How to Style a Data Template Field

To style a data template field:

1. In the Template Manager, edit the data template field definition item.
2. Click the Configure tab.
3. In the Appearance group, click the Tree Node Style command. The CSS style dialog appears.
4. In the CSS style dialog, in the text field, enter CSS styles, or click the Edit button to use a CSS style wizard. For example, enter `HEIGHT:600px;` to set the height of a Rich Text Editor, or `FONT-WEIGHT:bold;` to style the text in a single-line text field.

2.3.3 Data Template Field Sort Order

Explicitly set the sort order property of data template field definition items to control the order of fields when a data template and its base templates define the same data template sections.

How to Sort Data Template Fields

To sort data template fields:

1. In the Template Manager, edit the data template definition item.
2. In the editing pane, click the Builder tab. The Template Builder appears.
3. In the Template Builder, click the mouse in the field name to select that field.

4. In the Sorting group, click the Up, Down, First, and Last commands to sort the field relative to the other fields in the section.

How to Set the Sort Order Property of a Data Template Field Definition Item

To set the sort order property of a data template field:

1. In the Template Manager, edit the data template field definition item.
2. Show the standard fields as described in the section How to Show or Hide the Standard Fields.
3. In the Appearance section, in the Sortorder field, enter a numerical value.
4. Hide the standard fields as described in the section How to Show or Hide the Standard Fields.

2.3.4 Rich Text Editor (RTE) Configuration

The following sections describe procedures to configure the Rich Text Editor (RTE).

How to Configure RTE Profiles

To configure RTE Profiles:

1. In the Sitecore Desktop, select the core database.
2. Open the Content Editor and select `/Sitecore/System/Settings/Html Editor Profiles`.
3. In each custom RTE profile definition branch, remove options that are unnecessary for all users, and control the visibility of other options using the `item:read` access right.
4. In the Sitecore Desktop, select the master database.

Important
Do not edit the default RTE profiles under `/sitecore/system/Settings/Html Editor Profiles` in the core database. Instead, duplicate an existing RTE profile, and configure RTE data template field definition items to use that profile.

How to Determine the Path to an RTE Profile

To determine the path to an RTE profile:

1. In the Content Editor in the core database, select the RTE profile definition item.
2. Copy the path to the RTE profile definition item to the Windows clipboard as described in the section How to Copy the Path to an Item to the Windows Clipboard.
3. In the Sitecore Desktop, select the master database.

How to Set the Profile for an RTE Template Field Definition Item

To set the RTE profile for an RTE template field definition item:

1. In the Template Manager, edit the RTE field definition item.
2. In the Data section, in the Source field, enter the path to the RTE profile.
3. In the Sitecore desktop, select the master database.

How to Add Features to an RTE Profile

To add features to an RTE profile, copy features from the `/sitecore/system/Settings/Html Editor Profiles/Rich Text Full` profile.

How to Set the CSS Used by RTE Fields

To set the Cascading Style Sheet (CSS) file used to populate the CSS drop-down in all RTE fields:

1. Close the Rich Text Editor.
2. In the web.config file, in the `/configuration/sitecore/settings/setting` element with the name `WebStylesheet`, set the `value` attribute to the path to a CSS file.
3. Open the Rich Text Editor to see the change.

How to Limit the CSS Styles Visible in RTE Fields

The Rich Text Editor will automatically list all of the styles in the CSS file specified by the `value` attribute of the `/configuration/sitecore/settings/setting` element with the `name` `WebStylesheet` in `web.config`. Use the CSS `import` directive in cases where not all of the styles on the site are relevant to the Rich Text Editor. For example, if the file `contentstyles.css` containing Rich Text Editor styles is in the same directory as the file `sitestyles.css` containing site styles, `sitestyles.css` can import `contentstyles.css` with the following code:

```
@import url(contentstyles.css);
```

Warning
Due to various levels of caching, changes to CSS files may not be visible in the browser immediately. If changes to CSS files do not appear, perform the following operations in order until the change appears: reload the Rich Text Editor, clear the browser cache, restart IIS, and change the value of the `value` attribute of the `/configuration/sitecore/settings/setting` element with the `name` `WebStylesheet` in `web.config`.

How to Configure the RTE HTML Element Types Drop-Down Menu

To configure the HTML element types listed in the RTE:

1. In the Content Editor in the core database, under the RTE profile definition item, select `/Paragraphs`.
2. Insert an item using the `/System/Html Editor Profiles/Html Editor List Item` data template.
3. In the Data section, in the Header field, enter the value to display in the elements drop-down.
4. In the Value field, enter the HTML element to insert when the user chooses this element type.
5. In the Sitecore Desktop, select the master database.

How to Enable Snippets in an RTE Profile

To enable snippets in an RTE profile:

1. In the Content Editor in the core database, select `/Sitecore/System/Settings/Html Editor Profiles/Rich Text Full/Toolbar 1`.
2. Copy the `Insert Snippet` item to the corresponding location in the RTE profile.
3. In the Sitecore Desktop, select the master database.

How to Add Snippets to an RTE Profile

To add snippets to an RTE profile:

1. In the Content editor in the core database, select the RTE profile definition item.
2. Under the `Snippets` item, insert items using the `/System/Html Editor Profiles/Html Editor Snippet` data template.

3. In the Data section, in the Header field, enter the text to display in the snippets drop-down list.
4. In the Value field, enter markup to insert when the user chooses the snippet.
5. In the Sitecore Desktop, select the master database.

How to Control the Markup Inserted by the Enter Key

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name HtmlEditor.LineBreak` controls the markup that the Rich Text Editor inserts when a user presses the ENTER key. By default, the `value` of this `setting` is `p`, causing the Rich Text Editor to surround text with paragraph elements (“`<p>...</p>`”). To insert a line break element (“`<br>`”) at the end of the line instead of wrapping the text with paragraph elements, change the value of this setting to `br`.

Note
If the `value` of the `/configuration/sitecore/settings/setting` element with `name HtmlEditor.LineBreak` is `p`, you can insert a line break element (`<br>`) by pressing CTRL-ENTER. If the value of this setting is `br`, you can insert paragraph elements (`<p>...</p>`) by pressing CTRL-M.

How to Disable the RTE HTML Tab for Some or All Users

To remove the HTML tab from the RTE:

1. In the Content Editor in the core database, select the RTE editor profile definition item.
2. Delete or restrict read access rights to the `/Buttons/HTML View` item.
3. In the Sitecore Desktop, select the master database.

2.3.5 Data Template Field Security

By default, users with read or write access rights to an item have read or write access rights to all fields of that item. Developers can apply data template field security to restrict which users have read and write access rights to individual fields of items to which they have read or write access rights.

How to Configure Data Template Field Security

To configure data template field security:

1. In the Template Manager, edit the data template field definition item.
2. Click the Security tab.
3. In the Security group, click the Assign command. The access rights dialog appears.
4. In the access rights dialog, grant and deny the field read and field write access rights for one or more accounts.

2.3.6 How to Create an Iframe Data Template Field

To create an iframe data template field:

1. Create a web page containing the iframe user interface. For information about the query string parameters passed to the iframe field, see the Client Configuration Reference manual.[2]
2. In the Template Manger, select the data template definition item.
3. In the editing pane, click the Builder tab. The Template Builder appears.

[2] http://sdn5.sitecore.net/Reference/Sitecore%206.aspx.

4. In the Template Builder, add a field of type iframe, and then save the data template.
5. Edit the iframe data template field definition item.
6. In the Data section, in the Source field, enter the URL of the web page created previously containing the iframe user interface.

2.4 Item Editors

The following sections describe procedures to configure item editors.

2.4.1 How to Configure Item Editors

To configure item editors:

1. In the Template Manager or the Content Editor, edit the standard values item or the individual item.
2. Click the Configure tab.
3. In the Appearance group, click the Editors command. The custom editors dialog appears.
4. In the custom editors dialog, in the All tree, expand branches and double-click item editors to add them to the Selected list.
5. To change the order of item editor tabs, in the Selected list, select an item editor and click the arrows at the right to sort the selection.
6. To remove an item editor, in the Selected list, double-click the item editor.

2.4.2 How to Create an Item Editor

To create an item editor:

1. Create a web page containing the item editor user interface. For information about the query string parameters passed to the custom editor, refer to the Client Configuration Reference manual.[3]
2. In the Sitecore Desktop, select the core database.
3. In the Content Editor in the core database, select `/Sitecore/Content/Applications/Content Editor/Editors/Items`.
4. Insert an item editor definition item using the `/Sitecore Client/Content Editor/Editor` data template.
5. In the Data section, in the header field, enter a value that Content Editor should display in the tab that activates the item editor.
6. In the Icon field, enter the icon to display in the tab that activates the item editor.
7. In the URL field, enter the URL of the web page created previously containing the item editor user interface.
8. Save the item editor definition item.
9. In the Sitecore Desktop, select the master database.

[3] http://sdn5.sitecore.net/Reference/Sitecore%206.aspx.

2.5 Insert Options

Use insert options to control the types of items users can insert beneath existing items. For more information about insert options, see the Data Infrastructure Reference manual.[4]

[4] http://sdn5.sitecore.net/Reference/Sitecore%206.aspx.

Chapter 3

Data Validation

This chapter provides procedures to configure data validation. Developers use data validation to enforce rules for data entry. This chapter contains the following sections:

- **Validation Error Levels**
- **Configuring Validation Rules**
- **Registering Validators**
- **Custom Validators**
- **Validation Actions**

Note
For each item, Sitecore invokes the item validation rules defined in the item or the standard values item associated with its data template, as well as global validation rules. For each data template field, Sitecore invokes the validation rules defined in the data template field definition item, as well as the validation rules defined in the data template field type validation rules definition item.

Tip
Avoid use of the Validation and Validation Text fields in the Data section of field definition items available in earlier versions of Sitecore in favor of the data validation features described in this document.

Important
Do not edit the default validator definition items under `/sitecore/system/Settings/Validation Rules`. Instead, register additional validators by creating additional validator definition items.

3.1 Validation Error Levels

Validation error levels control the actions Sitecore takes on validation results. Each validator returns one of the following error levels defined in `Sitecore.Data.Validators.ValidatorResult`.

Error Level	**UI Color**	**Function**
`Unknown`	Gray	Validation incomplete or result otherwise unknown.
`Valid`	Green	Valid.
`Suggestion`	Green	Suggestions appear in the user interface.
`Warning`	Orange	Warnings appear in the user interface.
`Error`	Red	Errors prevent the user from completing workflow commands associated with the workflow validation action.
`CriticalError`	Red	Critical errors cause a modal warning when the user attempts to save an item and prevent the user from completing workflow commands associated with the workflow validation action.
`FatalError`	Red	Fatal errors cause a modal warning and prevent the user from saving the item.

3.2 Configuring Validation Rules

You can configure Sitecore to invoke validation rules in the Quick Action Bar, in the Validator Bar, when the user chooses the Validate command in the Proofing group on the Review tab, and when the user chooses a specific workflow command.

Note
After you stop editing a field value, Sitecore automatically invokes validation rules asynchronously, and then updates the user interface after validation completes. The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Validators.UpdateDelay` controls the length of time between the cessation of editing activity and the invocation of the validators. To disable this automatic revalidation feature, set the value attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Validators.AutomaticUpdate` to `False`.

Note
The default Sitecore configuration disables validation in the Quick Action Bar. To enable Quick Action Bar validation, right-click the Quick Action Bar, and then select Validation Rules.

Note
In general, select the same validation rules for all four types of validation.

3.2.1 How to Configure Quick Action Bar Validation Rules

To configure validation rules to invoke in the Quick Action Bar:

1. In the Content Editor, edit the appropriate validation rules definition item as described in the following sections.
2. In the Validation section, in the Quick Action Bar field, select validation rules.

3.2.2 How to Configure Validate Button Validation Rules

To configure validation rules to invoke when the user clicks the Validate command in the Proofing group on the Review tab:

1. In the Content Editor, edit the appropriate validation rules definition item as described in the following sections.
2. In the Validation section, in the Validate Button field, select validation rules.

3.2.3 How to Configure Validation Bar Validation Rules

To configure validation rules to invoke in the Validation Bar:

1. In the Content Editor, edit the appropriate validation rules definition item as described in the following sections.
2. In the Validation section, in the Validation Bar field, select validation rules.

Note
The validation bar does not appear in Content Editor if the `/configuration/sitecore/settings/setting` with `name` ContentEditor`.ShowValidatorBar` has a `value` of `False`.

3.2.4 How to Configure Workflow Validation Rules

To configure validation rules to invoke when the user chooses a workflow command:

1. In the Content Editor, edit the appropriate validation rules definition item as described in the following sections.
2. In the Validation section, in the Workflow field, select validation rules.

3.2.5 How to Create a Workflow Command or State Validation Action

To configure Sitecore to invoke the validation workflow action when the user chooses a workflow command or when the user chooses a workflow command:

1. In the Content Editor, select the workflow state or command.
2. Insert a validation workflow action definition item using the `/System/Workflow/Validation Action` data template.
3. In the Data section, in the Type field, enter the following:

```
Sitecore.Workflows.Simple.ValidatorsAction,Sitecore.Kernel.
```

4. In the Max Result Allowed field, enter the maximum `Sitecore.Data.Validators.ValidatorResult` value the workflow validation action can generate. If validation generates a higher validation error level, the workflow validation action will prevent the user from completing the workflow command. The default value for this field is `Warning`.
5. In the Unkown Result field, enter a message to display to the user if the validator returns `Unknown` as the validation result.
6. In the Warning Result field, enter a message to display to the user if the validator returns `Warning` as the validation result.
7. In the Error Result field, enter a message to display to the user if the validator returns `Error` as the validation result.
8. In the Critical Error Result field, enter a message to display to the user if the validator returns `CriticalError` as the validation result.
9. In the Fatal Error Result field, enter a message to display to the user if the validator returns `FatalError` as the validation result.

3.2.6 How to Configure Validation Rules for All Instances of a Data Template Field Type

To configure validation rules to invoke for all instances of a specific data template field type:

1. In the Content Editor, select `/Sitecore/System/Settings/Validation Rules/Field Types`.
2. Select the existing data template field type validation rules definition item, or insert a data template field type validation rules definition item using the `/System/Validation/Field Type Validation Rules` data template. For the name of the field type validation definition item, use the name of the field type as it appears in the Type field in the Data section of the field type definition item.
3. In the data template field type validation rules definition item, in the Validation Rules section, configure validation rules.

3.2.7 How to Configure Validation Rules for All Items:

To configure validation rules for all items, in the Content Editor, select `/Sitecore/System/Settings/Validation Rules/Global Rules` and configure validation rules.

3.2.8 How to Configure Validation Rules for an Individual Item or All Items Based on a Specific Data Template

To configure validation rules for an individual item or all items based on a specific data template:

1. In the Content Editor or the Template Manager, select the individual item or the standard values item for the data template.
2. Show the standard fields as described in the section How to Show or Hide the Standard Fields.
3. Configure validation rules.
4. Hide the standard fields.

3.2.9 How to Configure Validation Rules for a Specific Data Template Field

To configure validation rules for a specific data template field:

1. In the Template Manager, edit the data template field definition item.
2. Configure validation rules.

3.2.10 Default Item Validators

You can configure item validation rules using the following default validators.

Validator	**Function**
Broken Links	Identifies invalid references.
Duplicate Name	Identifies items with a name or display name that matches the name or display name of another item under the same parent (case-insensitive).
Full Page XHtml	Checks the validity of the output of requesting the item using the default device.
Media Size Too Big	Identifies media too large to serialize for database storage.
Url Characters	Checks item names for characters requiring escape sequences in URLs.

Note
XHTML validators validate the XML against the schema defined by the `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `XHtmlSchemaFile`.

3.2.11 Default Field Validators

You can configure field validation rules using the following default validators.

Validator	**Function**
Broken Links	Identifies invalid references.
Is Email	Identifies invalid email addresses.
Is Integer	Identifies invalid integers.
Is XHtml	Checks validity of field against local XHTML schema.

Validator	Function
Max Length 40	Identifies field values containing more than 40 characters.
Rating 1 to 9	Identifies invalid integers, negative numbers, and values greater than nine.
Required	Identifies empty fields.
Spellcheck	Identifies fields containing spelling errors.
W3C XHtml Validation	Identifies invalidity against remote W3C validation service.
Alt Required	Identifies missing Alt text in media library image data templates.
Extension May Not Start with a Dot	Identifies media items with a dot character in the field named extension, which is invalid.
External Link Target	Identifies external links in Rich Text Editor fields that should open in a new browser window and provide a title.
Image Has Alt Text	Identifies image fields that do not contain alternate text and that reference media items that do not contain alternate text.
Image Has Alt Text from Media Library	Identifies image fields that do not specify alternate text and that reference media items that do contains alternate text.
Image Size	Identifies image field that references a media item that exceeds the size limit.
Rich Text Image Size	The width of images used in the Rich Text Editor must not exceed the `value` of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `Media.MaxImageWidth`.

The `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `HtmlEditor.ValidatorServiceUrl` controls the URL Sitecore uses for W3C XHtml Validation.

3.3 Registering Validators

You can configure validation rules definition items to use the default validators, or register custom validators.

Important
To maximize the performance of the Sitecore user interfaces, validators that consume significant resources and especially processing time, such as to invoke external services or other potentially long-running operations, should run in separate threads. Using a separate thread allows the system to invoke additional validators before a validator completes. Sitecore will update the user interface as the different validators complete.

3.3.1 How to Register a Validator

To register a field validator:

1. In the Content Editor, select the appropriate project-specific folder under `/Sitecore/System/Settings/Validation Rules/Field Rules` or `/Sitecore/System/Settings/Validation Rules/Item Rules`.
2. Insert a validator definition item using the `/System/Validation/Validation Rule` data template.
3. In the Data section, in the Type field, enter the class signature.
4. In the Parameters field, enter URL-escaped key=value parameters, separated by ampersand characters (“`&`”).
5. If the validator should run in a separate thread, in the Data section, in the Use Thread field, select the checkbox.

3.3.2 How to Register a Regular Expression Field Validator

To register a regular expression field validator:

1. Register the field validator.
2. In the Data section, in the Type field, enter the following:

```
    Sitecore.Data.Validators.FieldValidators.RegexValidator,Sitecore.Kern
el
```

3. In the Parameters field, enter the following:

```
    Pattern=RegularExpression&Text=Message
```

4. In the Parameters field, replace `RegularExpression` with a regular expression.
5. In the Parameters field, replace `Message` with a message to display if the field does not match the pattern. Sitecore will call `String.Format()` to substitute `{0}` in the message with the name of the field.

3.3.3 How to Register an Integer Field Validator

To register an integer field validator:

1. Register the field validator.
2. In the Data section, in the Type field, enter the following:

```
    Sitecore.Data.Validators.FieldValidators.IntegerFieldValidator,
Sitecore.Kernel
```

3. In the Parameters field, enter the following:

```
        AllowNegative=AllowNegative=Boolean&AllowZero=Boolean
```

4. In the Parameters field, replace `Boolean` with `True` or `False` as appropriate.

3.3.4 How to Register an Integer Range Field Validator

To register an integer field range validator:

1. Register the field validator.
2. In the Data section, in the Type field, enter the following:

```
        Sitecore.Data.Validators.FieldValidators.IntegerRangeValidator,
Sitecore.Kernel
```

3. In the Parameters field, enter the following:

```
        Min=Minumum&Max=Maximum
```

4. In the Parameters field, replace `Minimum` with the minimum allowed value for the field.
5. In the Parameters field, replace `Maximum` with maximum allowed value for the field.

3.3.5 How to Register a Maximum Length Field Validtor

To register a maximum field length validator:

1. Register the field validator.
2. In the Data section, in the Type field, enter the following:

```
        Sitecore.Data.Validators.FieldValidators.MaxLengthFieldValidator,
Sitecore.Kernel
```

3. In the Parameters field, enter the following"

```
        MaxLength=MaximumLength
```

4. In the Parameters field, replace `MaximumLength` with the maximum allowed length of the field value.

3.3.6 How to Register Validators for Specific Items and Items Based on Specific Templates

To register validation rules for specific items and items based on specific templates:

1. In the Content Editor, select `/Sitecore/System/Settings/Validation Rules/Item Rules`.
2. Insert a validation rule definition item using the `/System/Validation/Validation Rule` data template.
3. In the Data section, in the Type field, enter the class signature.
4. In the Parameters field, enter URL-escaped key=value parameters, separated by ampersand characters (“`&`”).

3.3.7 How to Disable Default Validation Rules

To disable default validation rules for all items:

1. In the Content Editor, select `/Sitecore/System/Settings/Validation Rules/Global Rules`.
2. In the Validation Rules section, configure validation rules.

3. Select `/Sitecore/System/Settings/Validation Rules/Field Types`.
4. Configure validation in field type validation rules definition items.
5. Select `/Sitecore/Templates/System/Media`.
6. Show the standard fields as described in the section How to Show or Hide the Standard Fields.
7. For each of the data templates used for media, click on the data template definition item, then click the Content tab in the editing pane, and in the Validation Rules section, configure validation.
8. Hide the standard fields.

Important
No character in an item name can match the regular expression specified by the `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name InvalidItemNameChars`. An item name must match the regular expression specified by the `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name ItemNameValidation`. Do not change these two settings.

3.3.8 How to Override the Default Error Level for a Validator

To override the default error level for a validator:

1. In the Content Editor, edit the validator definition item.
2. In the Data section, in the Parameters field, add the following:

```
Result=ErrorLevel
```

3. Replace `ErrorLevel` with the name of the error level in `Sitecore.Data.Validators.ValidatorResult`.

3.4 Custom Validators

This section describes procedures to implement a custom validator.

3.4.1 How to Implement a Custom Validator

A custom validator involves two components: a .NET class and a validator definition item.

3.4.2 How to Implement a Custom Validator

To implement a custom validator:

1. Create a class using the following prototype:

```
        using System.Runtime.Serialization;
        using Sitecore.Data.Validators;
        namespace Namespace.Data.Validators.ItemValidators//TODO:namespace (FieldValidators)
        {
          [Serializable]
          public class ClassName : Sitecore.Data.Validators.StandardValidator
          {
            public ClassName(){}//TODO:class name
            public ClassName(SerializationInfo info,StreamingContext
context):base(info,context){}
            public override string Name
            {
              get
              {
                return(GetType().ToString());//TODO:validator name
              }
            }
            protected override ValidatorResult GetMaxValidatorResult()
            {
              return(GetFailedResult(ValidatorResult.Error));//TODO:error level
            }
            protected override ValidatorResult Evaluate()
            {
              if(false)//TODO:validate ControlValidationValue
              {
                return(ValidatorResult.Valid);
              }
              else
              {
                Text = "error message";//TODO:error message
                return(GetFailedResult(ValidatorResult.Error));//TODO:error level
              }
            }
          }
        }
```

2. Replace `Namespace.Data.ItemValidators` with the appropriate namespace.
3. Replace all instances of `ClassName` with the class name.
4. Replace `GetType().ToString()` with the friendly name of the validator.
5. Replace all instances of with the appropriate validation error level.
6. Replace `false` with logic to validate the field value in `ControlValidationValue` or the item returned by `GetItem()`.
7. Replace `"error message"` with an error message.
8. Register the validator.

Important
If the result of `GetMaxValidatorResult()` is `FatalError` or `CriticalError`, Sitecore will block operations such as save or workflow commands in the user interface as appropriate until validation completes. To avoid blocking user interface actions during validation, the `GetMaxValidtorResult()` method of expensive validators should not return `FatalError` or `CriticalError`. Validation Bar and Quick Action Bar validators never block user interface operations.

3.5 Validation Actions

Validation actions represent clickable operations in the user interface to correct validation errors.

3.5.1 How to Create a Validation Action

To Create a Validation Action for a Single-Line Text data template field:

1. Create a class using the following prototype:

```
using Sitecore.Shell.Framework.Commands.ContentEditor.Validators;
namespace Namespace.Shell.Framework.Commands.ContentEditor.Validators//TODO:namespace
{
  public class ClassName:ValidatorCommand//TODO:class name
  {
    public override void Execute(CommandContext context)
    {
      var validator=GetValidator(context);
      if(validator!=null)
      {
        var control = GetControlToValidate(validator);
        if (control!= null)
        {
          control = control as Sitecore.Web.UI.HtmlControls.Control;
          if(control!=null)
          {
            control.Value=Value;//TODO:logic
            Validate();
          }
        }
      }
    }
  }
}
```

2. Replace `Namespace.Shell.Framework.Commands.ContentEditor.Validators` with the namespace of the class.
3. Replace `ClassName` with the name of the class.
4. Replace `Value` with the validated value for the field.
5. In the Content Editor, select `/Sitecore/System/Settings/Validation rules/Field Rules`.
6. Insert a validation action definition item using the `/System/Menus/Menu` Item data template.
7. In the Data section, in the Display Name field, enter the label the user should select in the user interface to invoke the validation action.
8. In the `/App_Config/commands.config` file, insert a new command based on the following prototype:

```
<command name="validator:ClassName" type="Namespace.Class,Assembly"/>
```

9. Replace `ClassName` with the name of the class, and replace `Namespace.Class,Assembly` with the namespace and the class signature.

Important
Because unsaved values are only available in the UI and not in the database, `Sitecore.Data.Items.Item` and other APIs are not available. The API used to validate fields depends on the field type, and may involve JavaScript, such as for values of Rich Text Editor fields.

3.5.2 How to Use a Validation Action

To Use a Validation Action:

1. In the Content Editor, edit an item that violates a validation rule that provides a validation action.
2. In the Validation Bar, right-click the validation indicator, and then select the validation action.

Chapter 4

Simplify Repetitive User Operations

This chapter provides procedures to simplify repetitive user operations.

This chapter contains the following sections:

- **Security Presets**
- **Layout Presets**

4.1 Security Presets

This section describes procedures for working with security presets.

Important
To minimize long-term administration, avoid security presets using security inheritance wherever possible.

4.1.1 How to Create a Security Preset

For the name of the security preset definition item, use a value appropriate for the command in the ribbon that will invoke this security preset.

To create a security preset by defining access rights through the user interface:

1. In the Content Editor, select `/Sitecore/System/Settings/Security/Presets`.
2. Insert a security preset definition item using the `/System/Security/Security Preset` data template.
3. In the Data section, in the Security Preset field, define access rights.

To create a security preset by copying access rights from an existing source item:

1. In the Template Manager or the Content Editor, select the source standard values item or the individual item.
2. Show the standard fields as described in the section How to Show or Hide the Standard Fields.
3. Show raw values as described in the section How to Show or Hide Raw Values.
4. In the Security section, in the Security field, select the value, and then copy it to the Windows clipboard.
5. Hide the standard fields.
6. Select `/Sitecore/System/Settings/Security/Presets`.
7. Insert a security preset definition item using the `/System/Security/Security Preset` data template.
8. In the Security section, in the Security field, paste the value from the Windows clipboard.
9. Hide the standard fields and raw values.

4.1.2 How to Apply a Security Preset

By default, to apply security presets, a user must be an administrator, or a member of the Sitecore Client Securing role.

Apply security presets using Content Editor or Security Editor.

Tip
Use Security Editor when applying access rights to multiple items.

To apply a security preset using Content Editor:

1. In the Content Editor, edit the item.
2. Click the Security tab.
3. In the Presets group, click a security preset command. Sitecore copies the access rights defined in the Security Preset field of the security preset definition item to the selected item.

To apply a security preset using the Security Editor:

1. In the Sitecore Desktop, in the lower left corner, click the Sitecore button, and then click Security Editor. The Security Editor appears.
2. In the Security Editor, select the item.
3. In the Presets group, click the security preset command. Sitecore copies the access rights defined in the Security Preset field in the security preset definition item to the selected item.

4.2 Layout Presets

This section describes procedures for working with layout presets.

Important
Consider creating additional data templates, command templates, or another solution to avoid layout presets, which duplicate layout details in numerous items.

4.2.1 How to Create a Layout Preset:

For the name of the layout preset definition item, use a value appropriate for the command in the ribbon that will invoke this layout preset.

To create a layout preset by copying layout details from an existing source item:

1. In the Template Manager or the Content Editor, edit the source standard values item or individual item.
2. Configure and test layout details for the item.
3. Show the standard fields as described in the section How to Show or Hide the Standard Fields.
4. Show raw values as described in the section How to Show or Hide Raw Values.
5. In the Layout section, in the Renderings field, select the value, and then copy it to the Windows clipboard.
6. Select `/Sitecore/System/Settings/Layouts/Presets`.
7. Insert a layout preset definition item using the `/System/Layout/Layout Preset` data template.
8. In the Data section, in the Layout field, paste the value from the Windows clipboard.
9. Hide the standard fields and raw values.

4.2.2 How to Apply a Layout Preset

By default, to apply layout presets, a user must be an administrator, or a member of the Sitecore Client Designing, Sitecore Client Developing, or Sitecore Client Maintaining role.

To apply a layout preset:

1. In the Template Manager or the Content Editor, edit the standard values item or the individual item.
2. Click the Presentation tab.
3. In the Presets group, click the layout preset command. Sitecore copies layout details defined in the layout preset definition item to the selected item.

Chapter 5

The Page Editor

This chapter provides procedures to configure the Page Editor.

This chapter contains the following sections:

- **Placeholder Settings**
- **How to Show or Hide Features Depending on Mode**

5.1 Placeholder Settings

This section provides procedures to configure placeholder settings.

Layout details reference placeholders either by placeholder key or using a fully qualified placeholder key. For example, the fully qualified placeholder key for a placeholder with key `content` in a sublayout bound to a placeholder with key `main` in a layout would be `/main/content`.

Note
The Design Pane of Page Editor uses fully qualified placeholder keys, but layout details do not require fully qualified placeholder keys.

Placeholder settings definition items with names matching placeholder keys automatically apply to those placeholders unless layout details specify placeholder settings. The placeholder settings definition item named `content` automatically applies to all placeholders with the key `content`, including nested placeholders, except where layout details specify placeholder settings. For the name of the default placeholder settings definition item for a placeholder, use the placeholder key. For example, for the placeholder with key `content`, edit the placeholder settings definition item named `content`. Alternatively, by convention, use the fully qualified placeholder key, replacing slash characters (“`/`”) with dash characters (“`-`”). For example, to control the placeholder with key `content` in a sublayout intended bound to a placeholder with key `main` in a layout, insert a placeholder settings definition item named `content`, or insert a placeholder settings definition item named `main-content` and associate this placeholder settings definition item with the fully qualified placeholder key `/main/content` in layout details.

Important
Sitecore does not automatically apply placeholder settings using fully qualified placeholder keys to the corresponding nested placeholders. For example, for the placeholder with fully qualified key `/main/content`, if it exists, the placeholder settings definition item with name `content` applies, but the placeholder settings definition item with name `main-content` does not. To cause different uses of placeholders to apply different placeholder settings definition items, configure placeholder settings for placeholder keys or fully qualified placeholder keys using layout details.

5.1.1 How to Create a Placeholder Settings Definition Item

To create a placeholder settings definition item:

1. In the Content Editor, select `/Sitecore/Layout/Placeholder Settings`.
2. Insert a placeholder settings definition item using the `/System/Layout/Placeholder` data template.
3. In the Data section, in the Allowed Renderings field, select the presentation components users can bind to the placeholder.
4. In the Description field, enter a summary of the placeholder. Consider including a graphic indicating the location of the placeholder within the layout or sublayout.
5. Set the icon for the placeholder settings definition item as described in the section How to Set the Icon for an Item to control the image that appears next to the name of the placeholder in the Page Editor Design Pane.
6. Configure write access to the placeholder settings definition item to control which users can bind sublayout and renderings to the placeholder.

5.1.2 How to Configure Placeholder Settings for a Data Template or an Individual Item

To configure placeholder settings for a data template or an individual item:

1. In the Template Manager or the Content Editor, edit the standard values item or the individual item.
2. Click the Presentation tab.
3. In the Layout group, click the Details command. The layout details dialog appears.
4. In the layout details dialog, under the device, click the Edit button. The Device Editor appears.
5. In the Device Editor, click the Placeholder Settings tab.
6. Click the Add button. The Placeholder Settings dialog appears.
7. In the Placeholder Settings dialog, for Key, enter the placeholder key or fully qualified placeholder key.
8. For Settings Item, click the Browse button, and then select the placeholder settings definition item.

5.2 How to Show or Hide Features Depending on Mode

In presentation components, you may wish to show or hide features depending on the mode in which users access Sitecore. For example, a page may use a field for the HTML `<title>` element. If the page does not use the field elsewhere, a rendering can include a conditional element to output an inline editing control for the field if the user is in Page Editor, simplifying the process to update the title of the page.

As another example, consider an HTML element used to enclose a field value, such as `<span>`. A rendering should not output the `<span>` element if the field does not contain a value. To provide inline editing for the field, the rendering may output the span and potentially the field editing control if the field has a value or if the user is in the edit mode of page editor.

As another example, consider when a rendering should output a field from the context item unless that field is empty, in which case it should output the value from an ancestor. It may be necessary to output an inline editing control for the context item only if it has a value.

To show or hide features depending on the mode of Page Editor in .NET, use `Sitecore.Context.PageMode`:

```
        if(Sitecore.Context.PageMode.IsPageEditorEditing)
```

In XSL, use or `sc:pageMode()`:

```
        <xsl:if test="sc:pageMode()/pageEditor/edit">
```

The following table provides syntax to determine the Page Editor mode:

`Sitecore.Context.PageMode.`	**`sc:pageMode()`**	**Mode**
`IsPageEditor`	`/pageEditor`	
`IsPageEditorClassic`	`/pageEditor/classic`	Sitecore 5 WebEdit.
`IsPageEditorDesigning`	`/pageEditor/design`	Design Pane visible.
`IsPageEditorEditing`	`/pageEditor/edit`	Inline editing controls visible.
`IsPageEditorNavigating`	`/pageEditor/navigate`	Navigating without editing.
`IsPreview`	`/preview`	Preview.
`IsDebugging`	`/debug`	Debugger.
`IsProfiling`	`/profile`	Debugging and profiling.
`IsNormal`	`/normal`	Published web site.