

Sitecore CMS 7.0 or later

Data Definition Reference

A Conceptual Overview for CMS Administrators, Architects, and Developers

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Table of Contents

Chapter 1 Introduction 5
1.1 Features 6
1.2 Common Terminology 7
1.2.1 Template 7
1.2.2 Data Infrastructure 7
1.2.3 Data Representation 7
1.2.4 Data Repositories 7
1.3 General Information 8
1.3.1 Naming Conventions 8
Display Name and Field Title 8
1.3.2 Template Storage Locations 8
Chapter 2 Data Templates 10
2.1 Data Templates 11
2.1.1 Data Template Fields 12
2.1.2 Data Template Sections 12
Base Templates 13
Multiple Inheritance 14
Circular Inheritance 14
2.2 Standard Values 15
2.2.1 The $name Token 16
2.2.2 Blank vs. NULL Field Values 16
Chapter 3 The Standard Template 17
3.1 Standard Template Overview 18
3.2 The Structure of the Standard Template 19
3.2.1 Advanced Section 19
3.2.2 Appearance Section 19
3.2.3 Help Section 20
3.2.4 Layout Section 20
3.2.5 Lifetime Section 20
3.2.6 Indexing Section 20
3.2.7 Insert Options Section 20
3.2.8 Item Buckets Section 21
3.2.9 Publishing Section 21
3.2.10 Security Section 21
3.2.11 Statistics Section 21
3.2.12 Tagging Section 22
3.2.13 Tasks Section 22
3.2.14 Validators Section 22
3.2.15 Workflow Section 22
Chapter 4 The Template Field 23
4.1 Understanding the Template Field Definition Item 24
4.1.1 The Template Field Template 24
Field Name 24
Type 24
Title 24
Source 24
Blob 25
Shared 26
Unversioned 26
Default Value 26

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Validation ... 26
Validation Text ... 26
Field Security ... 26
Reset Blank ... 27
Exclude from Text Search ... 27
Page Editor Buttons ... 27
4.2 The Data Template Field Types ... 28
4.2.1 The Analytics Field Types ... 28
4.2.2 The Simple Field Types ... 28
The Checkbox Field Type ... 28
The Date and Datetime Field Types ... 29
The File Field Type ... 29
The Image Field Type ... 30
The Integer, Number, Password, and Single-Line Text Field Types ... 31
The Multi-Line Text Field Type ... 31
The Rich Text Field Type ... 31
The Word Document Field Type ... 32
4.2.3 The List Field Types ... 32
The Multilist with Search Type ... 33
The Treelist with Search Type ... 33
The Checklist Field Type ... 33
The Droplist Field ... 33
The Grouped Droplink Field Type ... 34
The Grouped Droplist Field Type ... 34
The Multilist Field Type ... 34
The Name Value List Field Type ... 34
The Treelist and TreelistEx Field Types ... 35
4.2.4 The Link Field Types ... 35
The General Link with Search Type ... 35
The Droplink Field Type ... 36
The Droptree Field Type ... 36
The General Link Field Type ... 36
The Version Link Field Type ... 36
4.2.5 The Developer Field Types ... 36
The Icon Field Types ... 36
The IFrame Field Type ... 37
The Tristate Field Type ... 37
4.2.6 The System Field Types ... 37
The Query Datasource Type ... 38
The Attachment Field Type ... 38
The Custom Field Type ... 38
The File Drop Area (FDA) Field Type ... 38
The Internal Link Field Type ... 39
The Layout Field Type ... 40
The Rules Field Type ... 40
The Security Field Type ... 40
The Template Field Source Field Type ... 40
The Thumbnail Field Type ... 41
4.3 Sitecore Query ... 42
4.3.1 General Syntax ... 42
4.3.2 Axes ... 43
4.3.3 Operators ... 44
4.3.4 Internal Architecture ... 45
Chapter 5 Field and Item Validation ... 46

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

5.1 Understanding Validation Rules and Options 47
5.2 Default Validation Rules 48
5.2.1 Item Validation Rules 48
5.2.2 Field Validation Rules 48
5.2.3 System Field Validation Rules 49
5.3 Validation Options 50
Chapter 6 Insert Options 51
6.1 Insert Options 52
6.1.1 Effective Insert Options 52
6.1.2 Assigned Insert Options 52
6.1.3 Insert Rules 52
6.1.4 Insert Options Rules 53
6.1.5 The uiGetMasters Pipeline 53
6.2 Branch Templates 54
6.2.1 Using Branch Templates 54
What happens? 54
6.3 Command Templates 56
6.3.1 Using Command Templates 56

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Chapter 1

Introduction

This document describes the concepts that architects, developers, and CMS administrators need to understand when designing, implementing, and maintaining the information infrastructure associated with a Sitecore Web site.

This document contains the following chapters:

- Chapter 1 — Introduction
- Chapter 2 — Data Templates
- Chapter 3 — The Standard Template
- Chapter 4 — The Template Field
- Chapter 5 — Field and Item Validation

This chapter contains the following sections:

- Features
- Common Terminology
- General Information

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

1.1 Features

This document contains a detailed description of the components and concepts used when defining the information architecture of a Web site. In this document you will find details describing the function and purpose of the following:

Templates — Sitecore users create items using one of three template types: data templates, branch templates, and command templates. Data Templates form the framework around which items are built. They define fields used to control how data is entered and can inherit from other templates to enable reuse.

Fields — these are the areas within which the data entered into the system is controlled. Through these fields, grouped into **Sections**, we can organize and control the amount and type of data entered into the system. Fields are organized by their field type.

Field Types — these are the different ways through which data is entered or selected for fields. There are numerous flexible ways of entering data into the system and it is the field types which control the type of data that can be entered or selected in the fields.

Data is validated throughout the system. The different types of validation are discussed and we will describe how validation covers the entire data definition area within the system to control the validity and authenticity of data definition structure.

You can use standard values to ensure that default data is automatically inserted into newly created items. Standard values are default values that populate null fields of existing items or fill data into newly created items.

Branch templates — allow you to create a set of items rather than a single item at a time.

Command Templates — allow you to insert of items according to logic rather than predefined structures.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

1.2 Common Terminology

This section defines some of the common data definition concepts.

1.2.1 Template

Sitecore uses the term template to refer to the components within Sitecore that are used to create new data. The terms: data template, branch template, and command template refer to specific types of templates. Templates control the schema of data entered into the system.

1.2.2 Data Infrastructure

The term data infrastructure describes the aspects of a Sitecore solution that enforce data structures. Data infrastructure includes all types of templates including standard values, the information architecture of the Web site including security, insert options that control what users can create, and other structural aspects of the system. Sitecore differentiates data infrastructure from content, or data entered into that structure.

1.2.3 Data Representation

Sitecore data representation is:

- Flexible — with Sitecore you can define data structures through an easy-to-use browser-based user interface.
- Abstract — you can also change data structure definitions without impacting data.
- Hierarchical — the data is structured using implicit and explicit relations.
- XML-oriented — the system also provides XML representations of data.
- Object-oriented — Sitecore also provides hierarchical, object-oriented representations of data.

1.2.4 Data Repositories

The Sitecore data repositories provide a variety of data services. Data can be represented in a limitless variety of languages, and then controlled using strict versioning and security. The flow of data through the system can be set using a variety of custom rules developed into a workflow. Finally, publication of data to the “live” Web site can also be influenced by a variety of settings that strictly monitor and control how data is published.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

1.3 General Information

The following sections describe general information that pertains to data definition.

1.3.1 Naming Conventions

When naming templates, fields and field sections, try to use simple, relevant, and easy to understand names. By default, Sitecore displays the names you provide to both technical and non-technical users. Choose names that business users, such as content authors, will easily understand.

Display Name and Field Title

Where they are defined, Sitecore user interfaces show display names and field titles rather than actual item names. For more information about display names, see the *Client Configuration Cookbook*.

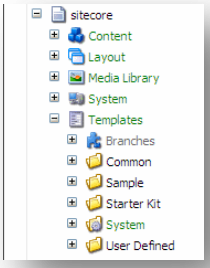
Warning
When you give a name to a template field this name is also written to the search index. Therefore avoid using generic names like *Title* in field names. For example, if you create two templates that both have a field called *Title*; the index considers this to be the same field. This can cause Sitecore severe confusion when searching indexes.

Note
Because code references fields by name, name fields as you would name the corresponding variables or object properties. Use field titles to provide users with friendly labels.

Tip
Avoid naming conventions that recommends prefixes or suffixes indicating types.

1.3.2 Template Storage Locations

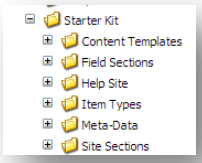
The various types of templates in Sitecore are stored as items under `/Sitecore/Templates` in the master database.



Warning
Except for anything beneath the Sample and User Defined folders, do not rename, remove, or otherwise modify any of the folders or data templates under the `/Sitecore/Templates` item.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Create template folders to classify templates by Web site, function, or other criteria. The following image indicates the Sitecore Starter Kit template folder structure.



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Chapter 2

Data Templates

This chapter describes data templates and standard values. A data template defines the structures of a type of item as a number of fields. The standard values item of a data template provides default values for the fields defined in that data template.

This chapter contains the following sections:

- Data Templates
- Standard Values

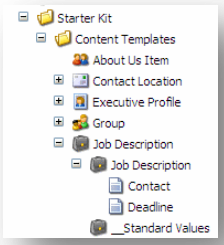
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

2.1 Data Templates

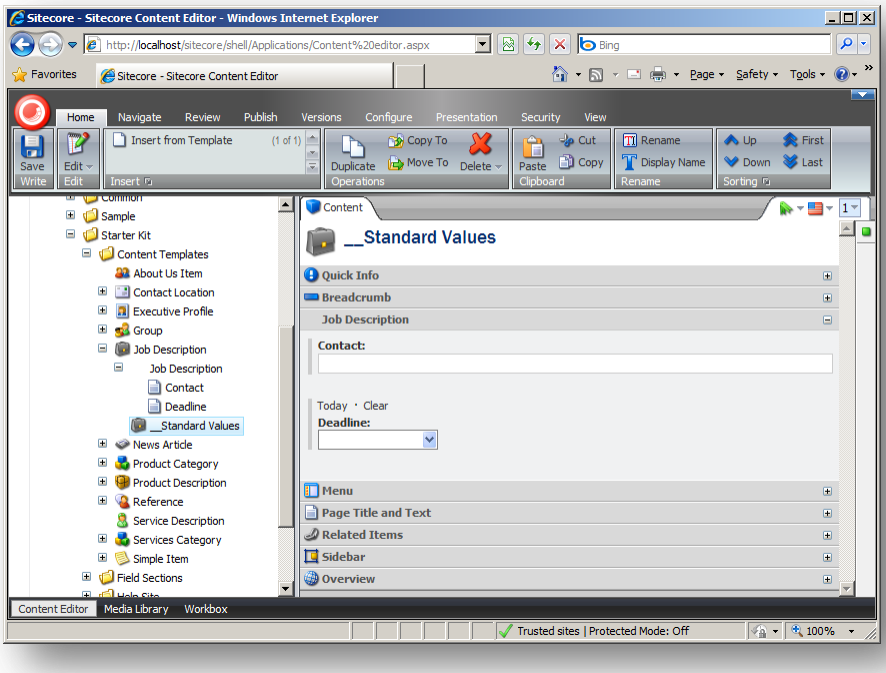
A data template defines a data type. Sitecore associates a data template with each item. The data template defines the structure of all items associated with that data template.

A data template contains a number of data template sections, each of which contains a number of data template fields. For more information about data template sections, see the section Data Template Sections. For more information about data template fields, see the section Data Template Fields. Data template sections organize data template fields visually in editing user interfaces. Data template fields define the structure of items based on that data template.

The system represents data templates as hierarchies of definition items; the root item defines the template, each child defines a field section and each grandchild defines a field.



In this example, you can see the Job Description data template opened with its field section Job Description immediately below it, and underneath this field section you can see the fields Contact and Deadline. These correspond to the sections and fields that you see if you edit an item associated with this data template. The following image shows the standard values item for the Job Description data template, which is an item based on (associated with) that data template.



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

In object-oriented programming, a data template is similar to a class, where data template fields are like properties of that class. In a relational database programming, a data template is similar to a table, where data template fields are like columns in that table.

Each data template inherits from zero or more base data templates, which in turn specify base templates. In the previous image, the user has collapsed the sections containing the fields defined in the base templates. For more information about base templates, see the section *Base Templates*.

In user interfaces, Sitecore combines the data template sections and fields in the data template and all base templates. Data templates support sequential and multiple inheritance.

Most data templates eventually inherit from the Sitecore standard template, which defines fields common to all items. For more information about the standard template, see the section *The Standard Template*.

Changes to a data template appear immediately in all items based on a data template or a data template that inherits from that data template.

2.1.1 Data Template Fields

A data template's fields define the properties of an individual data element created from that data template.

A data template field defines the user interface control and other properties that influence how the field behaves in the Content Editor and Page Editor. For more information about fields, see the section *The Template Field*.

Note
When defining field names, ensure that they are unique even between field sections. Both XSLT and .NET code use field names alone, without reference to sections, to extract content from fields.

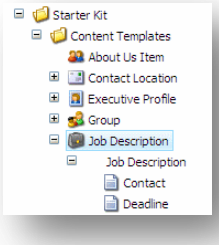
2.1.2 Data Template Sections

Data template sections organize the fields into related groups. Fields are always stored within a section. Sections, however, are only reflected in the Content Editor. Programmers do not need indicate a section when extracting content from a field.

Use sections to group fields logically, making them easier for content authors to find and use.

Consider creating data templates with a single section containing related fields to use as one of multiple base templates. This makes it easy to build up a new data template with multiple standard sections of fields.

Section definition items are stored under the corresponding data template definition item. In the following image, you can see the data template Job Description and immediately under it is the Job Description section, which, in turn, contains the Contact and Deadline fields.



When you edit an item associated with a data template, each section appears as a collapsible group of fields in the Content Editor.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

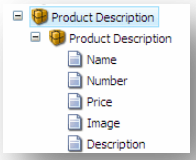
The Section Name field of the section definition is used as a label for the section in the Content Editor. Section names can be localized.

Base Templates

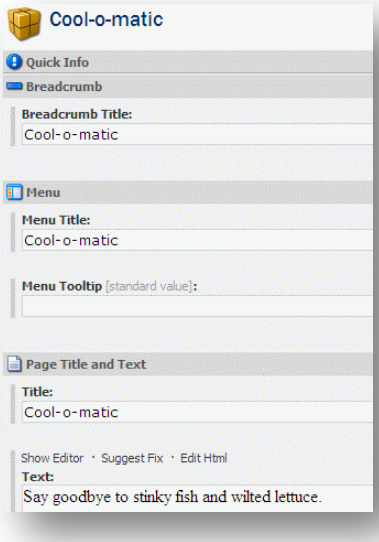
A data template inherits the sections and fields defined on its base templates. You can see the base templates associated with a data template in the Inheritance tab in the Template Manager or the Content Editor.

If a set of fields or sections are common to many templates then these can be gathered together in one data template, then that template can be added to other templates as desired.

As an example, consider the Product Description data template from the Sitecore Starter Kit. This data template defines a single section called Product Description and five fields: Name, Number, Price, Image, and Description.

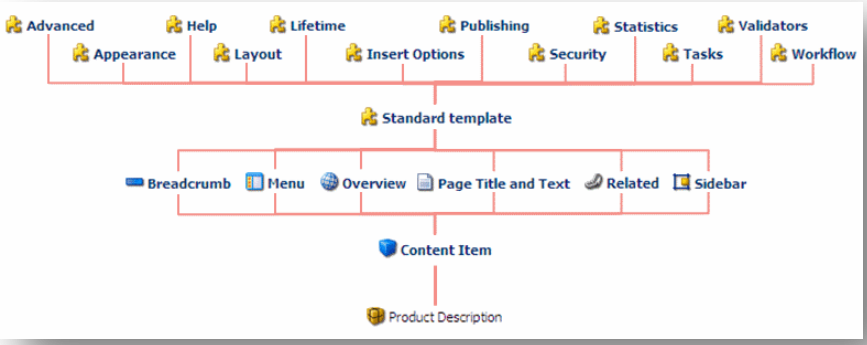


However, an item created from this template contains additional sections and fields.



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The item inherits these fields and sections from the Product Description data template's base templates. The following diagram indicates the hierarchy of base templates associated with the Product Description data template.



Multiple Inheritance

A data template can be based on any number of data templates, not just one. Occasionally more than one inherited template will contain the same field or field section. In this case, the UI will merge these fields or field sections to prevent duplication.

Note
If a data template inherits from a single data template more than once, Sitecore ignores all but the first occurrence of that base template.

Circular Inheritance

If a data template is based upon itself, either directly or indirectly, this is referred to as circular inheritance. Circular inheritance causes severe issues.

The symptoms include:

- The system becomes unresponsive, especially when working with data templates.
- ASP.NET raises application errors.
- Log entries indicate circular template inheritance detected.

Circular template inheritance often occurs when developers modify the base templates associated with the Standard template or its base templates. Avoid making modifications to the default templates provided with the system. Aside from the risk of circular inheritance, modifications to templates under the `/Sitecore/Templates/System` branch could complicate the process of upgrading Sitecore or raise other challenges.

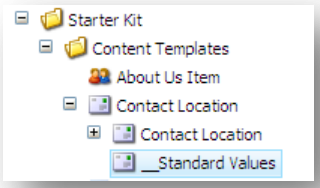
If any template does not explicitly inherit from another template, that template implicitly inherits from Sitecore's Standard template. The Standard template inherits from a number of templates defined under `/Sitecore/Templates/System/Templates/Sections`, each defining a section of the Standard template.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

2.2 Standard Values

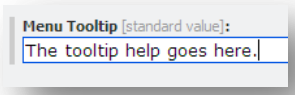
Standard values provide default values for data template fields. If the value of a field is NULL, the item contains the standard value for that field as defined in the data template for that item.

Sitecore stores the standard values for each data template as a child of the data template definition item named `__Standard Values`. Each `__Standard Values` item is based on that data template defined by its parent data template definition item. In the following image, the user has expanded the Contact Location data template. The data template contains the Contact Location data template section definition item and the `__Standard Values` item for the Contact Location data template.



The standard values item, like any item based on the data template, contains all the fields defined in the data template itself, plus any fields inherited from base templates. The standard values item contains default field values for items based on the data template. Standard values are used when a field value for an item is set to NULL.

When viewing an item in the Content Editor, any field that contains a standard value will be indicated by the token `[standard value]` immediately next to the field title (as shown).



Standard values can be inherited from any base template, not just the base template that an item is directly based on.

Tip
To reduce administration and centrally manage system settings, it is best to define the following in template standard values rather than in individual items:

- Layout settings.
- Initial workflow.
- Insert Options.

Note
If a standard value for a field is set on standard values items for multiple templates an item is based on, only the value from the first template in the inheritance list is used. The values from the other templates are ignored.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

2.2.1 The $name Token

Sitecore supports the `$name` token in standard values. Sitecore replaces the `$name` token with the name of the item during creation. This replacement, however, only takes place during item creation. The item name then becomes the field contents, which replaces the token `$name` in standard values.

Note
If a user renames an item, the `$name` token will *not* be re-evaluated. Thus, fields that had been assigned a value using the `$name` will continue to show the original item name even after the item has been renamed. If the field value that has been set using the `$name` token is reset, the field will then show the value `$name` and the user must set the field value to something more appropriate.

2.2.2 Blank vs. NULL Field Values

When a content author clears a field, such as a text field, this will often leave the field with blank contents, which is not the same as NULL contents. By default, such a field will display a blank value, not the standard value defined for the field.

Field definition items, however, contain a checkbox named Reset Blank. If this is selected, Sitecore replaces blank values with NULL. Thus, when Reset Blank is enabled for a field, and the user blanks out the contents of that field for an individual item, the field value in that item will be reset to NULL and the Content Editor will display the standard value for the field.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Chapter 3

The Standard Template

This chapter contains details about the Sitecore Standard template.

This chapter contains the following sections:

- Standard Template Overview
- The Structure of the Standard Template

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

3.1 Standard Template Overview

The Standard template is the Sitecore default base template shared by most other data templates. This is done either explicitly through inheritance, or implicitly if not specifically excluded from inheritance (by specifying the null GUID as a base template). If a template does not specify a base template, it inherits directly from the Sitecore standard template by default.

The Standard template defines sections such as security and workflow relevant to all types of items. It defines how Sitecore should manage an item, such as when it should be published, which workflow it is in, which users should be allowed to access it, and so on.

Most templates should be based “eventually” on the Standard template, either directly or indirectly.

The Quick Info section, which is always the first section of a template shown to Administrator and Developer users, is a field section in the Standard template. This field section contains item location details, the templates location in the content structure and various GUID’s related to storage.

Most fields in the Standard template represent system values shared with all versions in all languages.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

3.2 The Structure of the Standard Template

The following sections contain details of all the fields in the Standard template.

Important
All fields in the standard template begin with double underscores. Access these fields through the properties of the Sitecore.FieldIDs class.

3.2.1 Advanced Section

The **Advanced** section associates a data template with its standard values item.

- **Standard values** — this field contains a link to the Standard Values item.
- **Tracking** — this field contains analytics profile information.
- **Source** — this field designates the cloned item.

3.2.2 Appearance Section

The **Appearance** section controls how the Content Editor displays the item.

- **Context Menu** — this field contains a link to menu displayed when a user right clicks on an item in the Content Editor's content tree.
- **Display Name** — this field overrides the item name in the Content Editor.
- **Editor** — this field defines a custom editor for an item.
- **Editors** — this field controls the custom editor tabs that appear in the Content Editor.
- **Hidden** — this field controls whether or not the item is visible in the Content Editor. You can manage the visibility of hidden items by selecting the **Hidden Items** checkbox on the **View** tab of the main ribbon in the Content Editor.
- **Icon** — this field controls the icon shown in the content tree and in the item header in the Content Editor.
- **Read Only** — this field controls whether the item appears read-only.
- **Ribbon** — this field controls provides a technique to customize the Content Editor ribbon.
- **Skin** — this field associates formatting information with the item.
- **Sortorder** — this field is used by the system to specify the order in which the fields are displayed.
- **Style** — this field associates formatting information with the item.
- **Subitems Sorting** — this field controls the rule used to sort the children of the item.
- **Thumbnail** — this field associates a thumbnail image with an item.
- **Originator** — this field indicates the branch template used to create the item.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

3.2.3 Help Section

The **Help** section stores helpful information about the item.

- **Help link** — this field contains the link to the detailed help for this item.
- **Long description** — this field contains the fly-over help shown when hovering the mouse over an item in the content tree.
- **Short description** — this field contains the description shown in the item header of the Content Editor.

3.2.4 Layout Section

The **Layout** section is used primarily to associate the data template with its renderings.

- **Renderings** — this field contains the renderings used to display this item.
- **Renderers** — this field contains the renderers used to display this item.
- **Presets** — this field contains layout presets available for an item

Note
Managed Web sites use the Renderings field. Some Sitecore user interface components use the Renderers field.

3.2.5 Lifetime Section

The **Lifetime** section is used to hold the publication restrictions for individual versions of this item.

- **Valid from** — this field contains the date from which this version is valid for publication.
- **Valid to** — this field contains the date to which this version is valid for publication.
- **Hide version** — this field controls whether to hide this version from publication.

3.2.6 Indexing Section

The **Indexing** section is used to configure facets and boost settings.

- **Boost Value** — this field contains a float boost value, such as *1.0*, *2.3* or *0.1*. A value higher than zero pushes the results nearer the top of the results. A value lower than zero drags the results down nearer to the bottom.
- **Boosting Rules** — this field enables you to set rules for boosting. Boosting allows you to push an item higher up in the results. For more information on boosting, see the *Sitecore Search and Indexing Guide*.
- **Facets** — this field shows the facets that are displayed when you run a search from this item. Use this field to add or remove the facets that are associated with the item.

3.2.7 Insert Options Section

The **Insert Options** section is used to hold a set of Insert Rules and Options.

- **Insert Rules** — this field contains the insert rules.
- **Insert Options** — this field contains the insert options.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

3.2.8 Item Buckets Section

The **Item Buckets** section contains settings for items that have been converted to item buckets or made bucketable.

- **Bucket Parent Reference** — this field displays the path to the parent item. If you revert an item bucket or bucketable item this path is used to restore the connection to the parent item.
- **Is Bucket** — if this check box is selected it means that the item is a bucket. This is a system field and should not be set manually.
- **Bucketable** — select this field to make an item bucketable.
- **Lock Child Relationship** — if an item has children, this field enables you to decide whether the children should be added to the bucket or remain as children.
- **Default Bucket Query** — this field allows you to specify a default query that is run when you open the search tab and displays a list of items that match the results of the query.
- **Persistent Bucket Filter** — this field allows you to add a query to the search field that you cannot remove so it is always part of the search.
- **Enabled Views** — this field shows which views are enabled for displaying search results. Use this field to add or remove views. There are three views that are globally enabled by default.
- **Default View** — this field contains a list of views from which you can choose one to set as the default.
- **Quick Actions** — this field contains actions that you can associate with an item when it appears in the search results. For example, download, preview or publish. For more information on quick actions, see the manual *Developer's Guide to Item Buckets and Search*.

3.2.9 Publishing Section

The **Publish** section is used to hold the publishing restriction information.

- **Publish** — this field contains the date from which this item is valid for publication.
- **Unpublish** — this field contains the date to which this version is valid for publication.
- **Publishing groups** — this field contains the valid Publishing targets for the item.
- **Never publish** — this field controls whether or not an item should be published. Never publish overrides Publish and Unpublish.

3.2.10 Security Section

The **Security** section is used to hold the security settings.

- **Owner** — this field contains the name of the user who is the current owner of the item. For use with the virtual Creator-Owner role.
- **Security** — this field contains the security access rights for the item.

3.2.11 Statistics Section

The **Statistics** section is used to hold the basic creation, revision and update statistics.

- **Created** — this field contains the date and time the item was created.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

- **Created by** — this field contains the name of the user who created the item.
- **Revision** — this field contains the revision number, stored as a GUID text string.
- **Updated** — this field contains the last date and time that the item was updated.
- **Updated by** — this field contains the name of the user who performed the last update.

3.2.12 Tagging Section

The **Tagging** section contains words or phrases used to make associations between items making them easier for Sitecore search to find.

- **Semantics** — this field enables you to enter a word or phrase to use as a tag.

3.2.13 Tasks Section

The **Tasks** section is used to hold reminder information for associated tasks and to store an archive date.

- **Archive date** — this field contains the date the item will be archived.
- **Reminder date** — this field contains the date when a reminder e-mail will be sent to the reminder recipients.
- **Reminder recipients** — this field contains the e-mail address of the recipients of the reminder. This can be one or more e-mail addresses separated by a semi colon (“;”).
- **Reminder text** — this field contains the text of the reminder e-mail.

3.2.14 Validators Section

The **Validators** section is used to hold the validation rules.

- **Quick Action Bar Validation Rules** — this field contains the item validation rules displayed in the Quick Action Bar.
- **Validation Button Validation Rules** — this field contains the item validation rules used by the Validation button in the Proofing group of the **Review** tab in the Content Tree.
- **Validation Bar Validation Rules** — this field contains the item validation rules displayed in the Validation Bar on the right of the Content Editor.
- **Workflow Validation Rules** — this field contains the item validation rules that are used in workflow validation.
- **Suppressed Validation Rules** — this field disables global item validation rules.

3.2.15 Workflow Section

The **Workflow** section is used to hold the workflow status information.

- **Workflow** — this field contains the workflow state the item is in.
- **Workflow State** — this field contains the current workflow state the item is in.
- **Lock** — this field contains information about whether or not the item is locked, who locked it and what date/time it was locked.
- **Default workflow** — this field contains information about the default workflow for items created from this template.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Chapter 4

The Template Field

This chapter describes the details of the data template field definition items and the various field types associated with data template fields. This chapter also describes the query syntax that should be used in the *Source* property of certain data template field definition items.

This chapter contains the following sections:

- Understanding the Template Field Definition Item
- The Data Template Field Types
- Sitecore Query

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

4.1 Understanding the Template Field Definition Item

Data template fields define the user interface controls and other properties that influence how the field behaves in the Content Editor and Page Editor. It is primarily used to control the structure of a field and can provide automatic error checking when you enter data in the field.

The template field is equivalent to a property in object-oriented programming, or a column in a relational database.

Every field in a template must exist within a section, which organizes fields into logical and reusable groups. For more information about Template Sections, see section 2.1.2, Data Template Sections.

4.1.1 The Template Field Template

Field definition items are based on the *Template field* template. Each template field definition item defines many field properties.

Field Name

This is the name assigned to the field when it is created, used as the label for the field unless the **Title** field is filled in.

Tip
To make code easier to read when referencing fields by name, construct field names like variable names, without special characters.

Note
Field names *should* be unique. If fields are defined on a single data template with the same name then a validation error will occur when the data template is saved. But fields may also be inherited from base templates, thus, an item could potential contain multiple fields with the same name. If this occurs, Sitecore will display both fields in the Content Editor, but programmers, who use the field name to retrieve contents when using the API and XSLT Renderings, may get unexpected results.

These field properties in the Data section of this template include:

Type

The field type specifies which user interface control the Content Editor will display to accept input for this field and also controls the storage format for that field. For more information about field types, see the section, *The Data Template Field Types*.

Title

The title is displayed above the field in the Content Editor, unless the title is blank, in which case the Content Editor displays the field name.

Source

The source property provides information that influences the user interface control associated with the field in the Content Editor. The behavior of the source field depends on the field's type.

Some examples include:

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

- For list field types, such as Droplink, the source property indicates a location in the content tree that includes the items included in the list displayed by the field.
- For image and file types, the source property indicates the start folder displayed in the media library dialog.

Note
If the path begins with a tilde (“~”) character, the dialog opens with the desired folder selected, but allows the user to access the entire tree.

- For rich text field definitions, the item specified in the source property specifies a HTML editor profile controlling the features provided by the editing interface.
- For selection fields, the field source property may specify a Sitecore query using the `query:` prefix before the actual query statement as shown in the following example. For more information about Sitecore query syntax, see the section *Sitecore Query*.

```
query:/sitecore/content/Home/Employees/*[ startswith(@EmployeeName, 'A')]
```

Source Parameters

Various fields, such as Treelist, and TreelistEx, support the following parameters that are used in the Source property as an alternative to SitecoreQuery:

- **DataSource** — the field data source item, equivalent to using a path as the field source property as described previously.
- **DatabaseName** — the name of the database containing the data source item.
- **IncludeTemplatesForSelection** — the user can select items associated with this comma-separated list of template names.
- **ExcludeTemplatesForSelection** — the user cannot select items associated with this comma-separated list of template names.
- **IncludeTemplatesForDisplay** — the user can navigate items associated with this comma-separated list of template names.
- **ExcludeTemplatesForDisplay** — the user cannot see items associated with a comma-separated list of template names.
- **IncludeItemsForDisplay** — the user can see items with this comma-separated list of IDs.
- **ExcludeItemsForDisplay** — the user cannot see items with this comma-separated list of IDs.
- **AllowMultipleSelection** — the user can select more than one item.

You can separate multiple parameters in the source property with the ampersand (“&”) character as shown in the following example.

```
DataSource=/sitecore/content/home&IncludeTemplatesForSelection=section,sitemap
```

Blob

This field is for internal Sitecore use only and should not be used.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Shared

When this checkbox is selected, the field has the same value for every numbered version in all supported languages. When the *Shared* property is set, changes to the field value in any language or numbered version of the item will be reflected in all the other language versions and numbered versions.

Shared values should only be considered in the following cases:

- Old values of the field are irrelevant.
- Workflow restrictions do not apply to the field value.
- Values are very large (versioning consumes storage).

Note
If you select both the Shared checkbox and the Unversioned checkbox, then the field is shared. All versions in all languages of the item share a single value for this field. For more information about Unversioned fields, see the section Unversioned.

Note
Workflow and publishing restrictions do not apply to the values of shared fields.

Unversioned

When this checkbox is selected, the field has the same value for every numbered version within a language, but may have different values between languages. Unversioned fields are similar to shared fields, but the system can maintain different field values for different languages.

Note
If you select both the Shared checkbox and the Unversioned checkbox, then the field is shared. All versions in all languages of the item share a single value for this field. For more information about shared fields, see the section Shared.

Note
Workflow and publishing restrictions do not apply to the values of shared fields.

Default Value

This field is for internal Sitecore use only and should not be used.

Validation

This field can contain a regular expression against which the content of the field is validated when the field is saved. In order to save the field, its value must match this regular expression.

Validation Text

The system displays this message when you try to save the field and its contents do not match the expression in the Validation field.

Field Security

This field opens the Assign Security dialog for the Field Definition item. The Field Read and Field Write access rights apply.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Reset Blank

If the Reset Blank checkbox is selected, it resets the field value to NULL when the item is saved with this field set to blank. NULL fields reflect the standard value for the field. For more details about blank and NULL fields, see the section Blank vs. NULL Field Values.

Exclude from Text Search

If this checkbox is selected, this field is not included in the search indexes.

Page Editor Buttons

This field allows you to select controls that will be associated with this field in Page Editor.

Note
Additional values for template fields are also found in the Validation Rules, Appearance, Help, Security, Statistics and Workflow sections.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

4.2 The Data Template Field Types

Each field has an associated field type. The type of each field controls:

- The user interface component Sitecore presents for that field in user interfaces such as the Content Editor and the Page Editor.
- The format of the value stored for the field.
- The .NET classes and programming techniques developers use to access the field value, such as the classes in the `Sitecore.Data.Fields` namespace, the `renderField` pipeline and the FieldRenderer Web control.

Excluding the binary components of database media, Sitecore stores all field values as text. Some field types, such as single-line text, store a simple text value. Other field types, such as Multilist, store the GUIDs of the selected Sitecore items separated by pipe characters ("|"). Complex field types generally store an XML element. An Image field containing a single XML element with attributes representing different image properties. The field used to store layout details contains a more complex XML structure.

You can implement custom user interfaces for data templates by implementing custom field types, using the IFrame field type, or by implementing custom editors. For more information about the IFrame field type, see the section The IFrame Field Type.

To assist developers in selecting appropriate field types, Sitecore separates field types into the categories described in the following sections.

4.2.1 The Analytics Field Types

The Analytics field types are for internal use by Sitecore.

Warning
Do not use the Analytics field types.

4.2.2 The Simple Field Types

The Simple field types represent and individual value. An individual value may have multiple properties, such as the attributes in an Image field or the images and links in a Rich Text field. Sitecore provides the Simple field types described in the following sections.

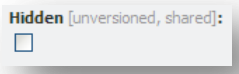
The Checkbox Field Type

This field type displays a toggle button. If the user selects the checkbox, Sitecore stores the value `1` (the number one). If the user does not select the checkbox, Sitecore stores a blank value.

Tip
Check for the literal value 1 in a checkbox field, or use a property such as `Sitecore.Data.Fields.CheckboxField.Checked`. Do not compare the value of a Checkbox field against NULL, `String.Empty`, `0` (the number zero), or any value other than `1`.

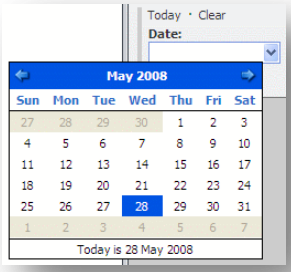
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The following image shows checkbox named **Hidden** in the **Appearance** section of the **Standard** template.

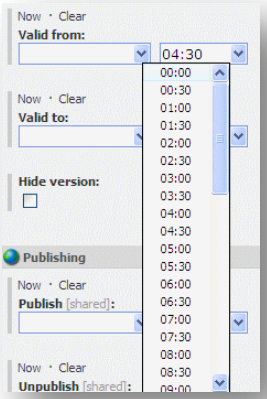


The Date and Datetime Field Types

The Date field type providers a user interface allowing a user interface to select or enter a date using manually. The Datetime field type adds a user interface to select or enter a time manually. Sitecore stores the content as a text string with the format `yyyymmddThhmmss`. If the user does not enter a time, Sitecore stores the value `000000` (midnight). The value stored represents local time for the Web server.



The following image shows the **Valid from** Datetime field in the **Lifetime** section of the standard template, including the drop-down list of times.



The File Field Type

The File field type allows the user to select an item in the media library. Users can click Open file to open the **Media Browser** and select the desired file from the media library.

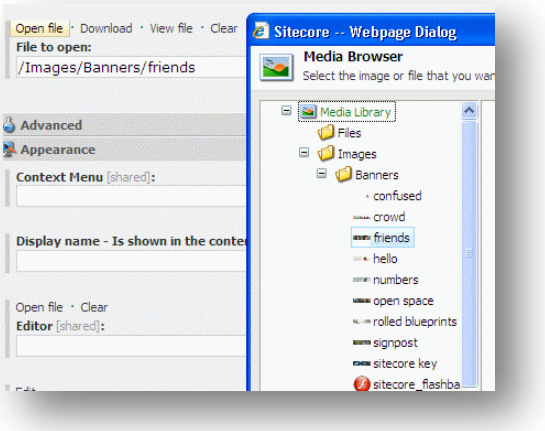
You can use the Source property of a File field to control the media folder selected when the user opens the Media Browser, or to specify the root item for this selection interface.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

If the Source property of a File field specifies an item, the Media Browser cannot navigate above that item. The user cannot navigate above the specified media folder. The default Source for File field is the root item in the Media Library (`/Sitecore/Media Library`).

If the Source property of a File field starts with a tilde character (`~/Sitecore/Media Library/Files`), the Media Browser will show the entire Media Library content tree with the specified media folder selected, allowing the user to navigate above that media folder.

The following image shows a file field with the path populated. On the right, you can also see the **Media Browser** opened with the item highlighted.

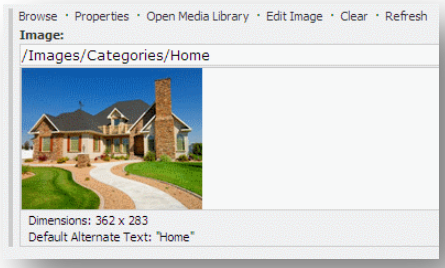


The Image Field Type

The Image field type allows the user to select an image from the media library and specify image properties. Users can click Open file to open the **Media Browser** and select the desired file from the media library.

The Source property of an Image field controls the selected item in the Media Browser. For more information about the Source property, see the section *The File Field Type*.

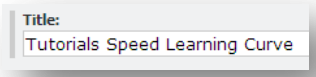
The following image shows the **Image** field from the **Home-Products** item in the Starter Kit.



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The Integer, Number, Password, and Single-Line Text Field Types

These field types store a single line of text.



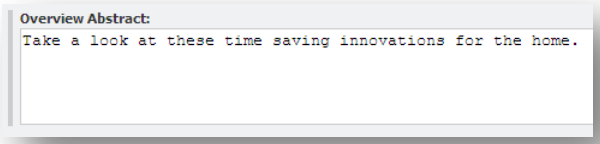
The Password field type stores plain text, but masks input in the Content Editor. Sitecore does not hash or otherwise mask the provided text in the database.



The Multi-Line Text Field Type

Use the Multi-line field type for simple text values spanning multiple lines. This field has no validation and there is no Rich Text support for fields of this type.

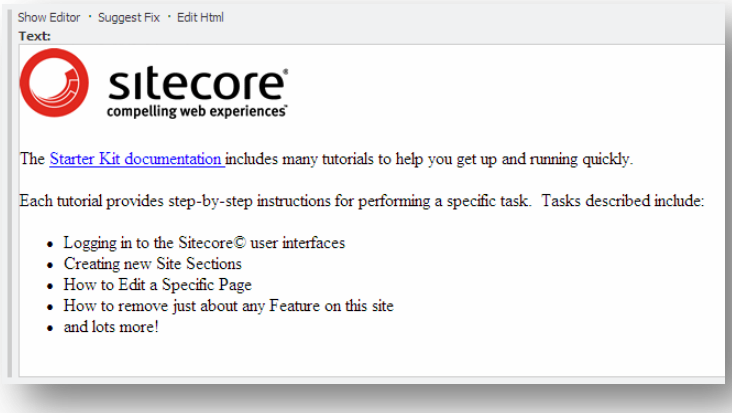
The following image shows the **Overview Abstract** multi-line text field from the **Home Products** item in the Starter Kit.



The Rich Text Field Type

This field type stores HTML text. The field displays the contents as a browser would display the source HTML, although the field actually stores character encoded HTML. The Show Editor button provides access to a Rich Text Editor, while the Edit Html button provides access to the stored HTML. For further information about Rich Text fields, see the Content Reference manual.

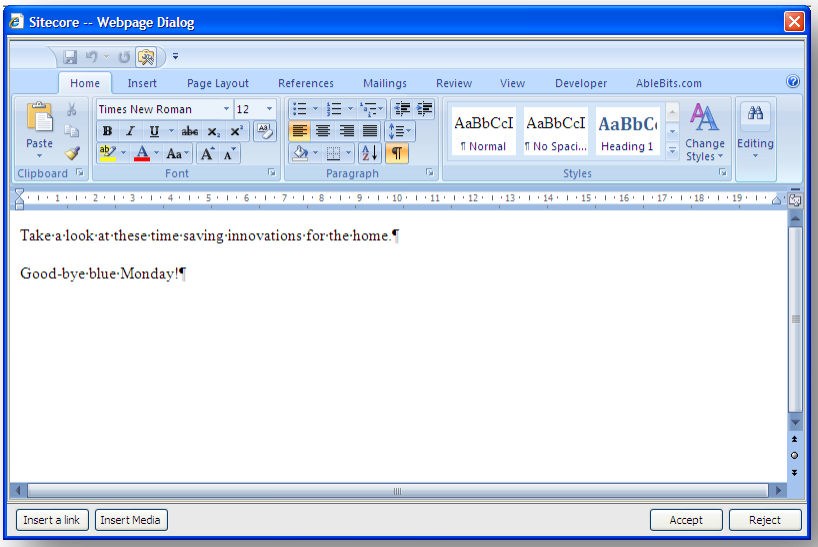
The following image shows a rich text field which includes an internal link and an image:



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The Word Document Field Type

The Word Document field type allows the user to edit HTML using Microsoft Word embedded in the browser.



Note
The Word Document field type requires Microsoft Internet Explorer 6 or higher. For instructions to configure Internet Explorer, see the *Internet Explorer Configuration Guide.*

You can set the `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with `name` `WordField.InlineEditing.Width` to control the width of the editor. You can set the `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with name `WordField.InlineEditing.Height` to control the height of the editor. You can set the `value` attribute of the `/configuration/sitecore/settings/setting` element in `web.config` with name `WordField.InlineEditing.Padding` to control the padding of the editor.

4.2.3 The List Field Types

Each list field types allow the user to select zero or more items using different user interfaces providing different features. The Source property of the field specifies the options in the list.

Note
If the Source property specifies an item, the Checklist, Droplist, and Multlist fields display the children of the specified item, and the Droptree, Internal Link, Treelist, and Treelist field types root the tree at the specified item.

Tip
The source property of the Checklist, Droplist, Droptree, and Multilist field types support Sitecore query. For more information about Sitecore query, see the section *Sitecore Query*.

Tip
The Checklist, Droptree, Multilist, Treelist, and TreelistEx field types store the GUIDs of the items selected by the user separated by a pipe character (“|”). If the field stores a reference to a single item,

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

you can access it using the `Sitecore.Data.Fields.ReferenceField` class or the `sc:item()` XSL extension method. If the field stores more than one reference, you can access the selected items using the `Sitecore.Data.Fields.MultilistField` class or the `sc:Split()` XSL extension method.

The Multilist with Search Type

A mulitilist with search field has no limitation and can scale to thousands of items. We recommend that you use this field type when referring to very large lists of items not just items that are in item buckets.

Use this field to attach a search query to a multi-list field and display the search results as selectable items. For example, if you want a multi-list of all product items, you can set the source field of the field to `TemplateFilter="Product ID"` and this returns the items in the list.

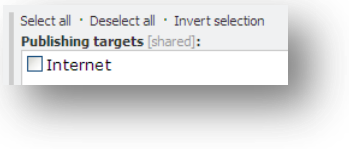
The Treelist with Search Type

The treelist with search field allows you to reference items in much the same way as a multi-list field. However, you can also enter a GUID to determine the start location of the search. This overrides the default filter that has been set on the template. You can use the search box to find the items in this field and search adheres to any filters applied.

The Checklist Field Type

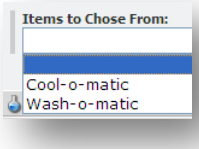
The Checklist field type allows the user to select one or more checkboxes representing the children of the item specified by the Source property of the field. The user cannot order their selection.

The following image shows the Publishing targets checklist field in the Publishing section of the Standard Template:



The Droplist Field

The Droplist field type allows the user to select a single item from the list specified by the Source property of the field.



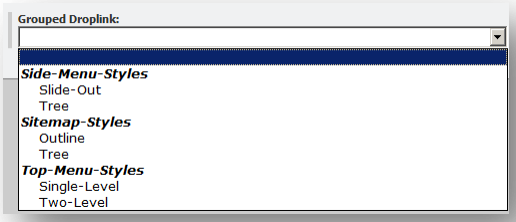
Important

The Droplist field type stores the name of the selected item. The Droplink field type stores its ID. For more information about the Droplink field type, see the section Use this field to search for and create a link to an item that is stored anywhere in the content tree.
The Droplink Field Type.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

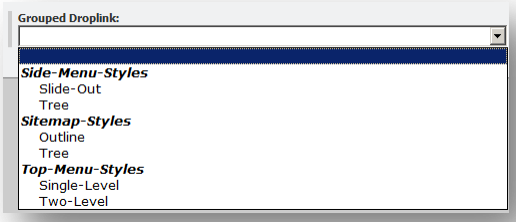
The Grouped Droplink Field Type

The Grouped Droplink field type allows the user to select a single grandchild of the item specified by the Source property of the field. The Grouped Droplink field type stores the GUID of the selected item.



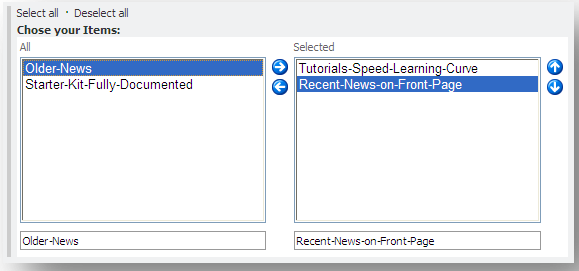
The Grouped Droplist Field Type

The Grouped Droplist field type allows the user to select a single grandchild of the item specified by the Source property of the field. The Grouped Droplist field type stores the name of the selected item.



The Multilist Field Type

The Multilist field type allows the user to select zero or more items from the list specified by the Source property of the field. The user can sort the selected items.



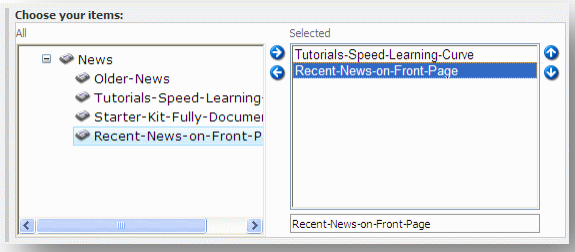
The Name Value List Field Type

The Name Value List field type allows the user to enter values for zero or more arbitrary keys. Sitecore stores the values entered as a list of key=value pairs separated by ampersands ("&"). Sitecore uses the Name Value List field type for rendering parameters. For more information about rendering parameters, see the *Presentation Component Reference.*

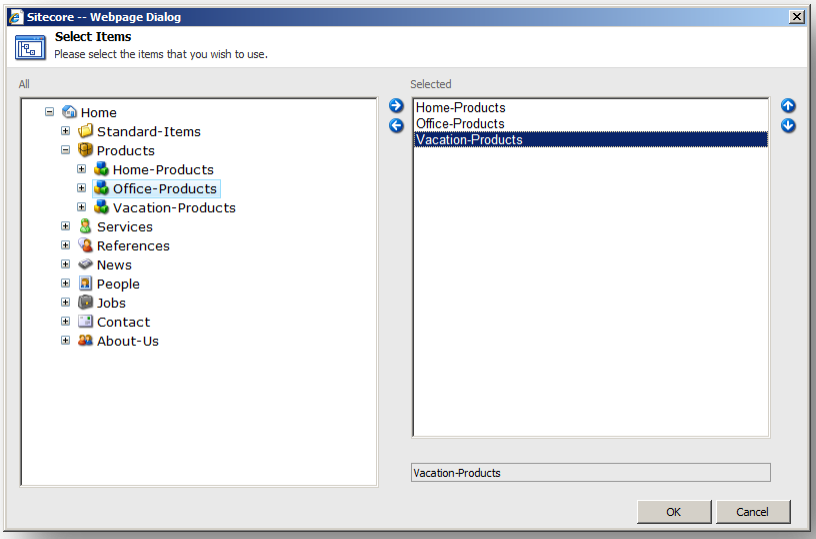
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The Treelist and TreelistEx Field Types

The Treelist field type allows the user to select zero or more descendants of the item specified by the Source property of the field. The user can sort the selected items.



The TreelistEx field provides the same functionality as the Treelist field, but only displays the selected items until the user elects to edit the list, which opens a new browser window. The Content Editor loads TreelistEx fields more quickly than Treelist fields.



Tip
Use the Treelist field type for fields that users edit frequently; use the TeelistEx field type for fields that users edit less frequently.

4.2.4 The Link Field Types

Link field types allow the user to enter links to items, external URLs, anchors, email addresses, and JavaScript functions.

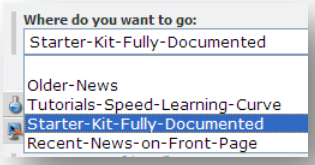
The General Link with Search Type

Use this field to search for and create a link to an item that is stored anywhere in the content tree.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

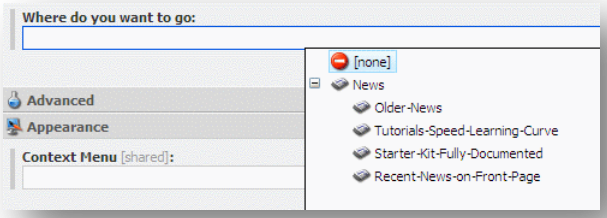
The Droplink Field Type

The Droplink field type allows the user to select a single item from the list specified by the Source property of the field using a drop-down list.



The Droptree Field Type

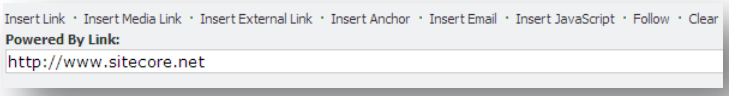
The Droptree field type allows the user to select a descendant of the item specified by the Source property of the field.



The General Link Field Type

The General Link field type allows the user to link to an item, an URL, an anchor, an email address, or a JavaScript function.

The following image shows a General Link field that populated with an external link to a URL:



The Version Link Field Type

This field type is for Sitecore internal use only.

4.2.5 The Developer Field Types

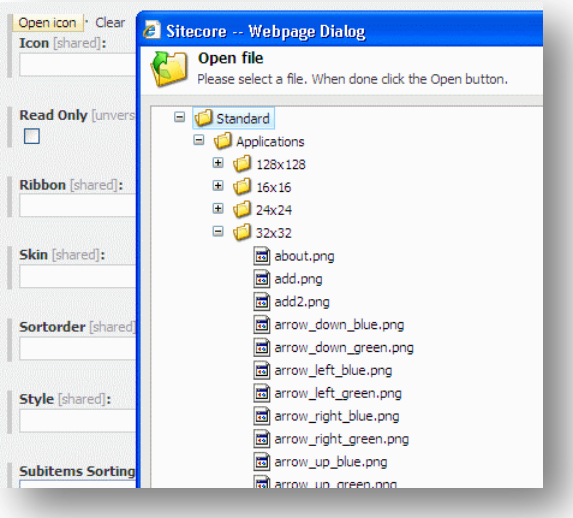
The developer field types primarily control the appearance of items in the Content Editor. Typical Web site developers do not use the Developer field types for end user data.

The Icon Field Types

The Icon field type associates a themed icon with each item.

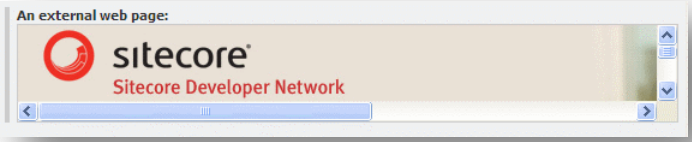
Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The following image shows an Icon field with the Open file dialog open.



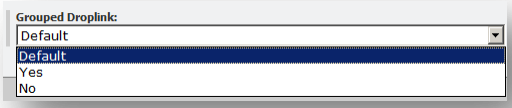
The IFrame Field Type

The IFrame field type display a Web page within a field. The Source property of the field species the URL to display. The following image shows an example of an external Web page.



The Tristate Field Type

The Tristate field type allows the user to select one of three options: Default, Yes, and No. If the user selects Default, the value of the field is an empty string. If the user selects Yes, the value of the field is the number `1`. If the user selects No, the value of the field is the number `0`.



4.2.6 The System Field Types

The system field types provide advanced features to meet specific requirements. Typical Web site developers do not use the System field types for end user data.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The Query Datasource Type

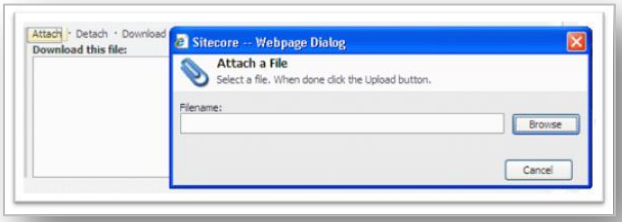
You can set the datasource for a presentation component using the Sitecore presentation layer. Setting a datasource enables you to create an internal link to specific content. You can select content to use as a datasource from the content tree or by specifying a search query. The search query is then passed to the code in your solution. For example you could search for items based off a particular template and have its results act as the datasource for a presentation component.

The Attachment Field Type

Database media items use the Attachment field type to store binary data.

Warning
Do not use the Attachment field type in your data templates.

The following image shows a field of type Attachment with the Attach a File dialog open.



The Custom Field Type

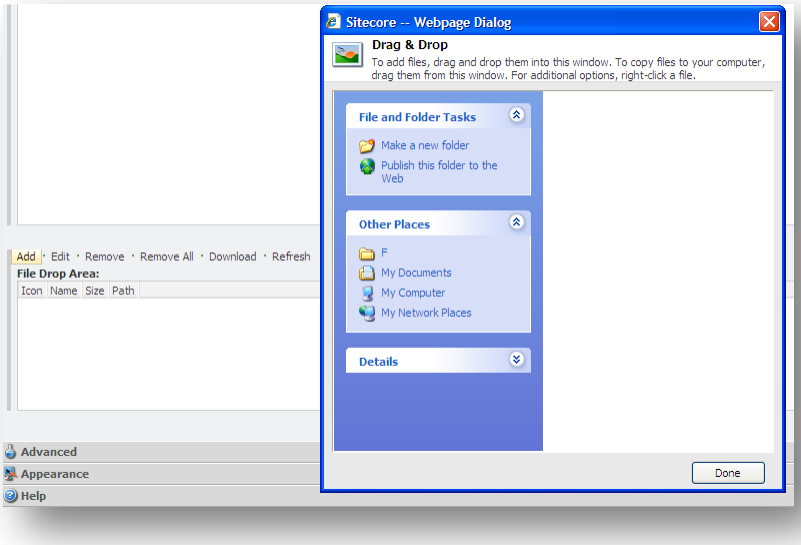
To prevent including irrelevant entries in the list of field types, Sitecore provides the Custom field type for infrequently used custom fields. For example, Sitecore uses the Custom field type for the CachingField, ExperienceTunerField, and TestLaboratoryField field types used in layout details.

The File Drop Area (FDA) Field Type

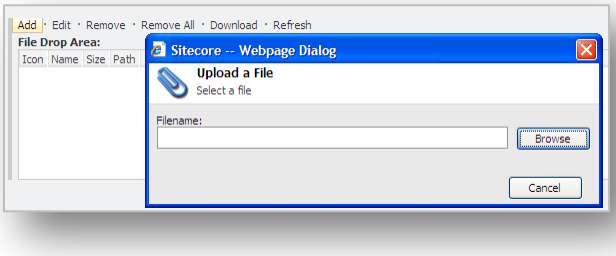
The File Drop Area (FDA) field type allows the user to manage files using the WebDAV protocol. For more information see the *WebDAV Configuration* reference. The value of a File Drop Area field references a media library folder containing corresponding media items. The Source property has no impact on File Drop Area fields.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

If the **File Drop Area** field is enabled, a **Drag & Drop** dialog box appears when you click Add:



If the **File Drop Area** field is disabled, an **Upload File** dialog box appears when you click Add:



When Sitecore publishes an item, it automatically publishes media items referenced in any File Drop Area fields of that item, including changes and deletions to those media.

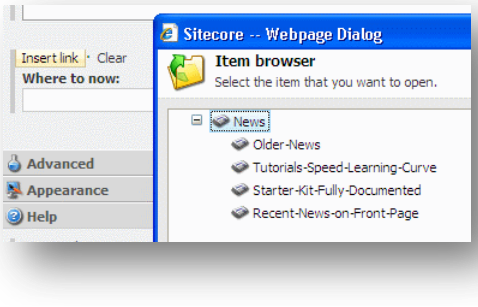
The Internal Link Field Type

The Internal Link field type allows the user to select a descendant of the item specified by the Source property of the field.

Warning
Do not use the Internal Link field type in your data templates. It is a Sitecore System field type which is appropriate only for Sitecore internal use and customizations. It is not suited for end user data because it points to a path and item name. This link can easily be broken by end users if they rename an item. Use Droplink, Droptree or General Link instead.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The following image shows the item selection dialog activated by the Insert Link command above an Internal Link field.

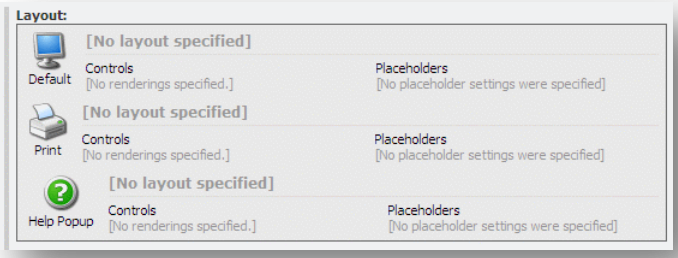


The Layout Field Type

The Layout field type allows the user to manipulate layout details for each item.

Warning
Do not use the Layout field type in your data templates.

The following image shows a Layout field.



The Rules Field Type

The Rules field type allows configuration of conditional renderings.

Warning
Do not use the Rules field type in your data templates.

The Security Field Type

The Security field type allows manipulation of security settings for each item.

Warning
Do not use the Security field type in your data templates.

The Template Field Source Field Type

The Template Field Source field type allows the user to select the Source property in a template field definition item.

Warning
Do not use the Template Field Source field type in your data templates.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

The Thumbnail Field Type

The Thumbnail field type allows the user to select a thumbnail image from the Media Library.

Warning
Do not use the Thumbnail Field field type in your data templates.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

4.3 Sitecore Query

Sitecore query provides query string syntax that filters and retrieves items from the Sitecore content tree using a simplified version of XPath syntax. Developers use Sitecore query strings in .NET code and in the Source field of field definition items (requires the “query:” prefix at the start of the Source field).

Note
Sitecore query is not scalable if you are using item buckets to store a large number of items in the content tree.

4.3.1 General Syntax

The Sitecore query syntax leverages the concept of a context item and uses the following symbols to reference related items:

Symbol	Meaning
`/`	the root of the content tree or a parent-child relationship
`text`	match by item name
`#`	escape text containing dashes (-) . For example: #meta-data#. It can also be used to escape special words, for example: #and#, #or#.
`*`	wildcard, match items of any name
`..`	the parent of the context item
`[ ]`	search criteria related to fields and XML element attributes
`@`	a field defined in the item’s base template
`@@`	an XML element attribute, all Sitecore items are treated as “item” elements which contain the following attributes: • name The item’s name • key The item’s name in all lower-case characters • id The item’s GUID • tid The item’s base template’s GUID • mid The branch template used to create the item, if any • sortorder The item’s sort order • template The item’s base template’s name • parentid The item’s parent’s GUID

Combining these symbols references specific items or groups of items. For example:

Sitecore Query String	Result set
`/*`	The root of the content tree.
`/sitecore/content/home`	The Sitecore Home item.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Sitecore Query String	Result set
`/sitecore/content/home/*[startswith(@title,'P')]`	Immediate subitems under the home item that contain a Title field that starts with “P”.
`*[@__hidden='1']`	All hidden subitems underneath the context item.
`query: /*/content/#meta-data#/colors/*[@show='1']`	A Source field which selects all subitems under the Color item that have the “Show” checkbox field checked.
`./*[@@tid="{A87A00B1-E6DB-45AB-8B54-636FEC3B5523}"]`	Subitems under the context item based on the Folder template.

4.3.2 Axes

The axis component of a query determines the direction of the node selection in relation to the context node. An axis can be thought of as a directional query.

Note
The axis component of a query is not scalable when using item buckets and a large number of content items.

The following table lists some common axes:

Axes	Description
`ancestor`	Returns all the ancestors of the context item (same as XPath).
`ancestor-or-self`	Returns the context item and all the ancestors of the context item (same as XPath).
`child`	(`/*`) returns all the children of the context item (same as XPath).
`descendant`	(`//*`) returns all the descendants of the context item; a descendant is a child or a child of a child and so on (same as XPath).
`descendant-or-self`	Returns the context item and all the descendants of the context item (same as XPath).
`following`	Returns all the following siblings of the context node (same as following-sibling in XPath).

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Axes	Description
`parent`	(`..`) returns the parent item of the context item (same as XPath).
`preceding`	Returns all the preceding siblings of the context item (same as preceding-sibling in XPath).
`self`	(`.`) returns the context item (same as XPath).
`[int]`	Returns the child item with the specified index.

4.3.3 Operators

You can use the following list of operators in Sitecore query expressions:

Operator	Description	Example	Return Value
`|`	Combination.	`//Products|//Shapes`	All items named Products or Shapes.
`+`	Addition.	`6 + 4`	10.
`-`	Subtraction	`6 - 4`	2.
`*`	Multiplication.	`6 * 4`	24.
`div`	Division.	`8 div 4`	2.
`=`	Equality.	`position() = 3`	True if `position()` is 3, otherwise False.
`!=`	Not equal	`position() != 3`	False if `position()` is 3, otherwise True.
`<`	Less than.	`position() < 4`	True if `position()` is less than 4, otherwise False.
`<=`	Less than or equal to.	`position() <= 4`	True if `position()` is less than or equal to 4, otherwise False.
`>`	Greater than.	`position() > 4`	True if `position()` is greater than 4, otherwise False.
`>=`	Greater than or equal to.	`position() >= 4`	True if `position()` is greater than or equal to 4, otherwise False.
`or`	Logical OR.	`position() = 3 or position() = 4`	True if `position()` is 3 or 4, otherwise False.
`and`	Logical AND.	`position() > 3 and position() < 7`	True if `position()` is greater than 3 and less than 7, otherwise False.
`mod`	Modulus (remainder of division).	`5 mod 2`	1.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

4.3.4 Internal Architecture

Sitecore processes queries using the fastest technology possible. This could be either the SQL database, if the data provider supports the requested query, or in the Sitecore data manager. Note that the SQL database provides the best performance, but, unfortunately does not provide support for all queries.

The SQL database supports queries such as "`/sitecore/content/home`" which resolves an item by path, and "`//home`" which locates a set of items by name.

How does this work in practice? Imagine that you have a large site that uses an SQL database and you're trying to find all the content items named *needle*. The following API code will retrieve this result set:

```
Item content = Sitecore.Context.Database.GetItem("/sitecore/content");
Item[] needles = content.Axes.SelectItems("//needle");
```

The SQL Server data provider supports this kind of query, so the query is resolved directly in the database and runs fairly quickly even though the database contains a large number of content items.

Adding additional search criteria, however, can change how the Sitecore performs the search. In the previous example, imagine that the items include a checkbox field named IsHidden. To find needles that are not hidden:

```
Sitecore.Data.Items.Item content =
  Sitecore.Context.Database.Items["/sitecore/content"];
Sitecore.Data.Items.Item[] needles =
  content.Axes.SelectItems("//needle[@IsHidden != '1']");
```

The SQL Server data provider does not support predicates (the portion of the search string enclosed in square brackets: `[@IsHidden != '1'])`. Therefore, Sitecore resolves this query in the data manager using the query API. To do this, the all the items in the query scope (all descendants of `/sitecore/content` in this example) are loaded so that the predicate can be evaluated against each item and the matching items returned. Unfortunately, the performance penalty associated with loading this many items can be quite dramatic for large sets of items.

Based on this understanding, we can see that a more optimal approach would be to first find all items named needle, then searching through the result set in memory to find the items that are not hidden.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Chapter 5

Field and Item Validation

This chapter provides details of the various ways that validation can be used on fields and items to control the authenticity of entered data. For more information about validation, see the *Client Configuration Cookbook.*

This chapter contains the following sections:

- Understanding Validation Rules and Options
- Default Validation Rules
- Validation Options

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

5.1 Understanding Validation Rules and Options

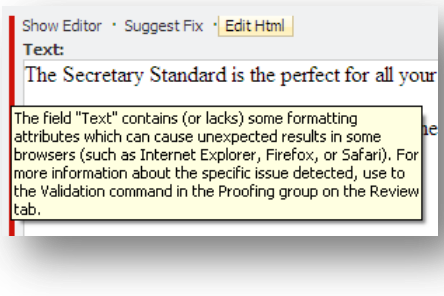
Validation rules and options validate field values using field validation rules resolved on-the-fly based on settings specified in the `web.config` file. The sections holding validation settings including direct expression settings, pipelines and event handlers as well as simpler settings like the ability to turn off the **Validation** bar in the Content Editor.

These validation rules validate a single field value in a single language version of an item. They can also validate each item using any number of item validation rules, which validate global values such as the item name, as well as conditions involving multiple field values.

When validating items and fields the validation rules use the same basic technology, and the main difference between the item and field validation rules are:

- Field validation rules validate values in one or more fields in a single language version. Whereas item validation rules validate item data shared to all versions, such as the item name, as well as conditions involving multiple field values.
- The user interface displays its field validation rules from `/sitecore/system/Settings/Validation Rules/Field Rules`, and its item validation rules from `/sitecore/system/Settings/Validation Rules/Item Rules`.
- In templates, the template field definitions define the field validation rules, whereas the template standard values and individual items define the item validation rules.

Any field validation issues appear to the left of the field value in the Content Editor as colored bars. The colors displayed are gray for no errors, yellow for warnings and red for errors. In the following image, you can see a field validation warning (the red bar); with a tooltip explanation displayed that contains more details about the error.



Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

5.2 Default Validation Rules

The following tables list examples of the preinstalled item and field validation rules and a brief description of the validation that they perform. Later versions may include additional preinstalled rules.

5.2.1 Item Validation Rules

Item Validation Rule	Validates
Broken Links	Checks all languages and all versions for broken links in one or more fields.
Duplicate Name	Checks that the item name is unique among siblings.
Full Page XHtml	Validates the XHTML generated when a visitor requests an item.
Media Size Too Big	Checks whether a Media Library item exceeds a specific size.
Url Characters	Checks if an item name contains characters that must be escaped when rendering URLs, which negatively impacts search engine indexing.

5.2.2 Field Validation Rules

Field Validation Rule	Validates
Broken Links	Checks if a field contains broken links.
Is Email	Checks if a field contains an email address.
Is Integer	Checks if a field contains an integer.
Is XHtml	Checks if a field contains XHTML.
Max Length 40	Checks if a field contains a value of 40 or less characters.
Rating 1 to 9	Checks if a field contains a value between 1 and 9.
Required	Checks if a field contains a value.
Spellcheck	Checks spelling using the RAD Editor spell check validation, also used in the Rich Text editor.
Type and Assembly	Checks if the value properly references a class in an accessible assembly.
W3C XHtml Validation	Validates the field HTML using the W3C validation service (the CMS must have Internet access).

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

5.2.3 System Field Validation Rules

System Field Validation Rule	Validates
Alt Required	Checks that the alt text is filled in.
Extension May Not Start with a Dot	Checks that the media file extension does not start with a dot.
Extern Link Target	Checks for external links, i.e. the links to other sites.
Image Has Alt Text	Checks the alt text on an image.
Image Has Alt Text from Media Library	Checks if the media item has default alt text. The default alt text (from the media library) will be used.
Image Size	Checks the size for the images referenced through image fields.
Rich Text Image Size	Checks the image dimensions for the images included in the rich text fields, i.e. if the image is too big to look good in the site design.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

5.3 Validation Options

Validation options can be chosen in the **Validation Rules** section of the data template standard values and template field definitions. Sitecore comes with a set of predefined validation options which are listed in the following table with a brief description of the validation that they perform.

Validation Option	Controls
Quick Action Bar	Validation issues appear in the **Quick Action** bar on the left in Content Editor
Validation Button	Validation issues appear when the user chooses the Validation command from the **Proofing** group on the **Review** tab, and when the user invokes a transition to a workflow state which includes the workflow validation action.
Validation Bar	Validation issues appear in the **Validation** bar on the right in Content Editor
Workflow Validation Rules	Validation issues appear in the user interface when a user chooses a workflow command associated with the workflow validation action. The user cannot complete the workflow action without resolving all validation errors.
Suppressed Validation Rules	Validation issues no longer appear in the Quick Action Bar if you suppress global validation rules on individual content items.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

Chapter 6

Insert Options

This chapter describes insert options, which control the types of items that users can insert beneath existing items. Insert options can include data templates, branch templates, and command templates.

This chapter contains the following sections:

- Insert Options
- Branch Templates
- Command Templates

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

6.1 Insert Options

Sitecore administrators and developers configure insert options to control the types of items that users can insert beneath existing items. Insert options can include:

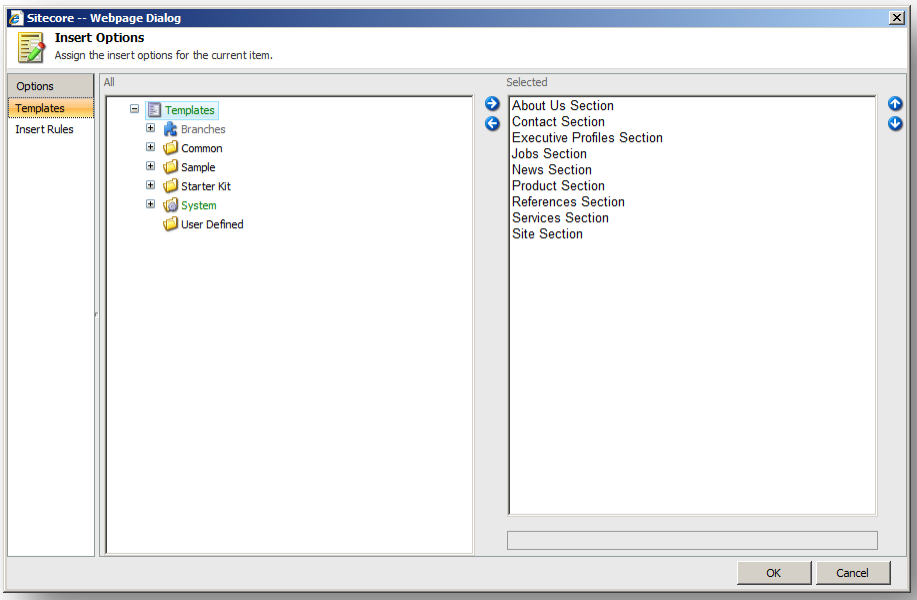
- Data Templates.
- Branch Templates.
- Command Templates.

6.1.1 Effective Insert Options

Sitecore uses the resources described in the following sections to determine the effective insert options for each user (the data templates, branch templates, and command templates that the user is allowed to invoke to insert a new item) for each item.

6.1.2 Assigned Insert Options

You can assign insert options in the standard values of a data template. Assigned insert options can include data templates, branch templates, command templates, and insert rules.



Important
Whenever possible, assign insert options in standard values instead of assigning insert options to individual items.

6.1.3 Insert Rules

Administrators can assign insert rules in insert options to dynamically redefine effective insert options for the user at runtime. Developers can implement custom insert rules for administrators to select.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

6.1.4 Insert Options Rules

Administrators can implement insert options rules to define effective insert options, providing user interfaces to select rule parameters. Unlike insert rules that apply to specific items, insert options rules apply to all items, though conditions typically limit their application. For more information about insert rules, see the section Insert Rules. For information about the rules engine, which provides an additional mechanism to control effective insert options, see the *Rules Engine Cookbook.*

6.1.5 The uiGetMasters Pipeline

Sitecore constructs the list of effective insert options by invoking the `uiGetMasters` pipeline defined in `web.config`. The `uiGetMasters` pipeline defines the list of data templates, command templates, and branch templates available to the user for inserting new items.

The `uiGetMasters` pipeline uses the following process to determine effective insert options for an item:

1. Add each of the item's assigned insert options to which the context user has read access to the list of effective insert options.
2. Apply any insert rules associated with the item to the list of effective insert options. For more information about insert rules, see the section Insert Rules.
3. Apply insert options rules to the list of effective insert options. For more information about insert options rules, see the section Insert Options Rules.
4. Remove any items to which the context user does not have the `insert:show` access right.

Developers can add custom processors to the `uiGetMasters` pipeline containing logic to determine effective insert options for the user at runtime. Unlike insert rules, which apply only to the items that reference those rules, the `uiGetMasters` pipeline applies to all items.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

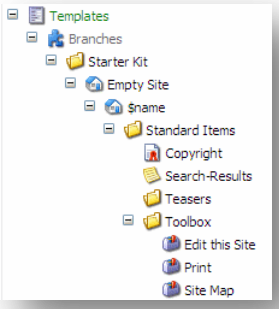
6.2 Branch Templates

Branch templates allow users to insert items using structures of items predefined by Sitecore administrators. A branch template consists of a branch template definition item, including all of its descendants.

When a user chooses a branch template from the insert options, Sitecore creates a copy of the entire collection of descendants defined underneath the branch template definition item, and places that collection as a group of subitems underneath the currently selected item.

Sitecore copies all field contents defined in the branch template descendants, but replaces the `$name` token assigned to fields and item names with the name provided by the user during the insert operation.

The following image shows the Empty Site branch template from the Sitecore Starter Kit. The only immediate subitem of the branch template definition item is called `$name`. When you create an item based on this branch template, Sitecore requests a name and will replace the `$name` token with the given name for the created item. Below 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.



6.2.1 Using Branch Templates

You can use branch templates to help users to create multiple items using a reusable, predefined structure. Branch templates can also be used to copy initial field values into the created item (rather than inherit them from the standard values). Note that this includes both field values for the Standard template fields (which are assigned via the ribbon controls, for example, the item's icon or access right assignments) and fields defined in custom data templates (such as a product number).

Branch templates can also be useful when you want content authors to be able to create multiple items at once (either siblings or descendants). Multiple content sub-trees can be created using branch templates.

What happens?

When the user invokes the branch template, the system carries out several actions:

- It copies the descendants of the branch template definition item, including all field values, to create new items.
- It performs token substitution on the new items, replacing `$name` and other tokens in both item names and field values with the name entered by the user when invoking the branch template.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

- In branch templates consisting of one child item with zero or more descendants, the name of that child is typically `$name` so the user can specify the name of the root item to create. For item names in branch templates, only `$name` is supported
- For any field not overriding its standard value in the branch template, the corresponding field in the new items contains its standard value, unless the standard value contains a token such as `$name`, which Sitecore also replaces with the given item name.

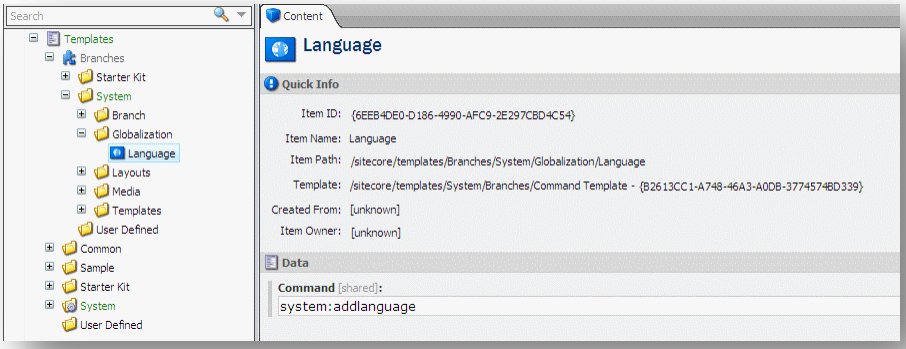
Note
Field values in branch templates could easily result in data duplication. Therefore template standard values are generally preferable to field values in branch templates. You should also remember that field values in branch templates are not inherited like template standard values, but are copied. Therefore, changes made to a branch after creating items using that branch are not reflected in the items previously created.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.

6.3 Command Templates

Command templates define a class and method to be called during an insert operation. Unlike data templates and branch templates, which consist of predefined structures, command templates reference Sitecore UI commands to invoke wizards or other logic used to create new items.

In the following image, you can see the *Language* command template that is used when you create 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.



6.3.1 Using Command Templates

You can use command templates to insert items according to logic rather than predefined structures. Command templates can also be used to insert multiple items when a user invokes an insert option, but command templates provide more flexibility than branch templates.

A command template can be assigned as an insert option to items and standard values. The command template insert option will appear identical to data template and branch template insert options. The only difference is that command template insert option will trigger a Sitecore UI command.

Command templates will typically invoke a wizard application that collects information from a user and then programmatically creates an appropriate set of items.

One example of a Sitecore defined command template involves creating a template:

- The insert options for `/Sitecore/System/Languages` include the `/Sitecore/Templates/Branches/System/Globalization/Language` command template.
- This invokes the same command used by the control panel to create a new language.
- The value of the **Command** field in each command template corresponds to an entry in the `/App_Config/Commands.config` file and makes the system to invoke methods in the specified class when the user invokes the command template.

Sitecore® is a registered trademark. All other brand and product names are the property of their respective holders. The contents of this document are the property of Sitecore. Copyright © 2001-2013 Sitecore. All rights reserved.