



A Guide to CMS Functions

2017-02-13

Contents

1	INTRODUCTION	3
1.1	Who Should Read This Guide	3
1.2	What You Will Learn	3
2	WHAT IS A CMS FUNCTION?	4
2.1	Purposes of Using Functions	4
2.2	Where Can Functions Be Used?	4
2.3	Function Markup	5
3	TYPES OF FUNCTIONS	6
3.1	Visual Functions	6
3.2	XSLT Functions	9
3.3	Razor Functions	9
3.4	User Control Functions	10
3.5	C# Functions	11
3.6	SQL Functions	12
3.7	MVC functions	13
4	HOW TO USE FUNCTIONS.....	14
4.1	Inserting Functions	14
4.2	Editing Function Properties	15
4.3	Errors in Functions	16
4.4	Testing functions	17
5	HOW TO SET FUNCTION PARAMETERS	20
5.1	Default Parameters	22
5.2	Constant Parameters	22
5.3	Function Calls	22
6	FUNCTION DOCUMENTATION	24
6.1	Help Texts	24
6.2	Function Information	24
6.3	Generated Documentation	25
7	TEST YOUR KNOWLEDGE	27
7.1	Task: Insert a function without parameters	27
7.2	Task: Insert a function and set its required parameter	27
7.3	Task: Override the default value of a parameter	27
7.4	Task: Insert a function's markup and modify a value	27
7.5	Task: View information on functions	27

1 Introduction

Functions are one of the central concepts in C1 CMS. They allow you not only to render content kept internally on a website but also integrate external content, functionality or logic within the website.

Being convenient for presenting web content in a highly customizable way, the functions themselves are highly customizable. Besides, being uniform in use, the functions are normally based on various technologies such as Razor, XSLT, .NET or SQL, which are transparent to the end users.

1.1 Who Should Read This Guide

This guide is intended for content editors who would like to learn how to use CMS functions that help reuse content, use dynamic content and enhance content by customizing its presentation and behavior.

Since this guide is solely focused on using rather than creating functions, it might be sufficient to have proper permissions in the Content perspective where you will insert functions on pages and edit their properties.

If you do not mind editing XHTML directly, you might need proper permissions in Layout and Functions.

In addition, if you are going test your knowledge by completing the tasks at the end of this guide, you will need access to the System perspective with permissions required to install add-ons.

1.2 What You Will Learn

When you finish reading this guide, you will know:

- What CMS functions are
- Where and what you can use functions for
- What types of functions there are
- How to insert functions in content and markup
- How to edit function properties
- How to set function parameters
- What types of parameters you can set
- Where to look up the description of functions and their parameters

2 What is a CMS Function?

A function in C1 CMS encapsulates static or dynamic content as well as behavior that can be further reused in multiple places such as XHTML or XSLT content on the website. Each function is a logical unit that you can insert on a page, in markup of a page template or another function etc.

To use functions, you do not necessarily need to know how to create functions or what types of functions there are in C1 CMS. You can thus skip the following chapter and go on to read about using functions.

However, a quick overview of this aspect of CMS functions might help you use functions more effectively. Plus, if you are eventually going to learn how to create your own functions, the following few sections and the following chapter might serve as a good introduction to the CMS function's internals.

2.1 Purposes of Using Functions

The functions in C1 CMS enjoy three major uses:

- **Content Reuse:** Functions make parts of content or functionality reusable within a single website and across multiple websites.
- **Dynamic Content Rendering:** Functions present dynamic content and allow its customization.
- **Integration:** Functions integrate external content or functionality by wrapping it up in format usable in C1 CMS.

Please note that combination of the above uses in one function is not uncommon.

2.2 Where Can Functions Be Used?

In C1 CMS, you can use functions in several places.

As a special graphic representation, you can insert functions on the Content tab of the Content Editor in the Visual mode.

If the function outputs content, it will show this content in the editor as function preview.

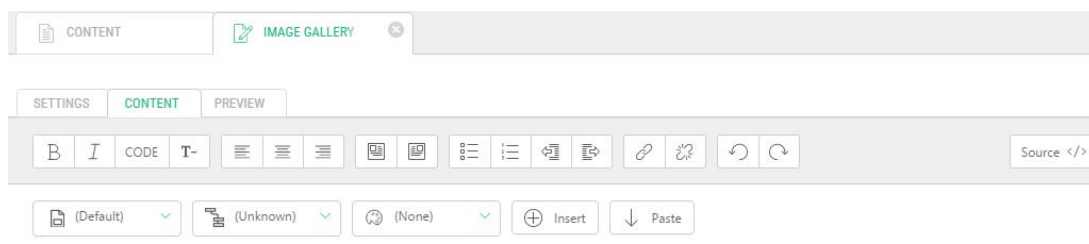


Image Gallery - Slimbox2

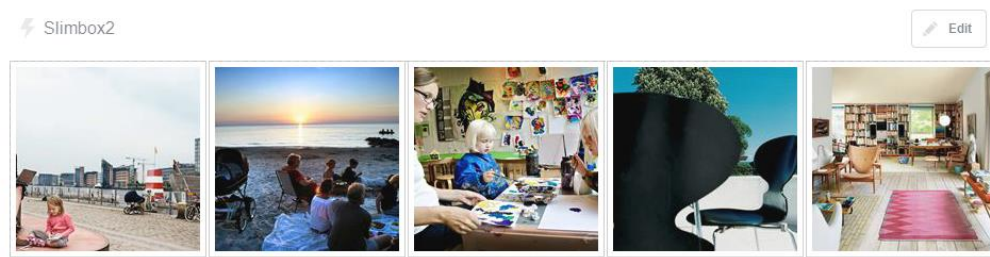


Figure 1: Preview of a function inserted in Visual Content Editor

As markup, you can insert functions on:

- the Content tab of the content editor in the Source mode
- the Markup Code tab of the page template editor
- the Template tab of the XSLT and other function editors



Figure 2: A function markup inserted in the Source mode

In general, as markup, you can insert functions wherever you can edit XML directly.

Also, you can use functions as [parameters to other functions](#).

Please note that in the Visual mode of the content editor, you can only insert functions that return XHTML. To have access to all the functions, you'll need to switch to the Source mode.

2.3 Function Markup

When inserted as markup, functions should follow a specific syntax.

```
<f:function name="Composite.Forms.NestedHtmlForm"
xmlns:f="http://www.composite.net/ns/function/1.0">
  <f:param name="FormMarkup">
    <f:function name="Composite.Search.SimplePageSearch.SearchResults" />
  </f:param>
</f:function>
```

Listing 1: A sample of function markup

The syntax is described as function definition schema in
~\Composite\schemas\Functions\Function.xsd.

The function elements are defined in this namespace:

```
xmlns:f="http://www.composite.net/ns/function/1.0"
```

The f:function element defines a CMS function. It must contain the name attribute (required), which specifies the function's name.

The function may contain no, one or more f:param elements, which define parameters of the function. The f:param element must contain the name attribute (required), which specifies the parameter's name, and, optionally, the value attribute, which specifies the value of this parameter.

Besides, the f:param element can contain one f:function element that embeds another CMS function.

3 Types of Functions

When creating functions, developers and designers are not limited to one technology in C1 CMS and have a choice of several types:

- [Visual functions](#)
- [XSLT functions](#)
- [Razor functions](#)
- [User Control functions](#)
- [C# functions](#)
- [SQL functions](#)
- [MVC functions](#) (available via an addon)

To the end user such as a content author, the functions look and behave uniformly in most cases. It means that when the content editor adds a function on a page, he or she does not really know what type of function it actually is.

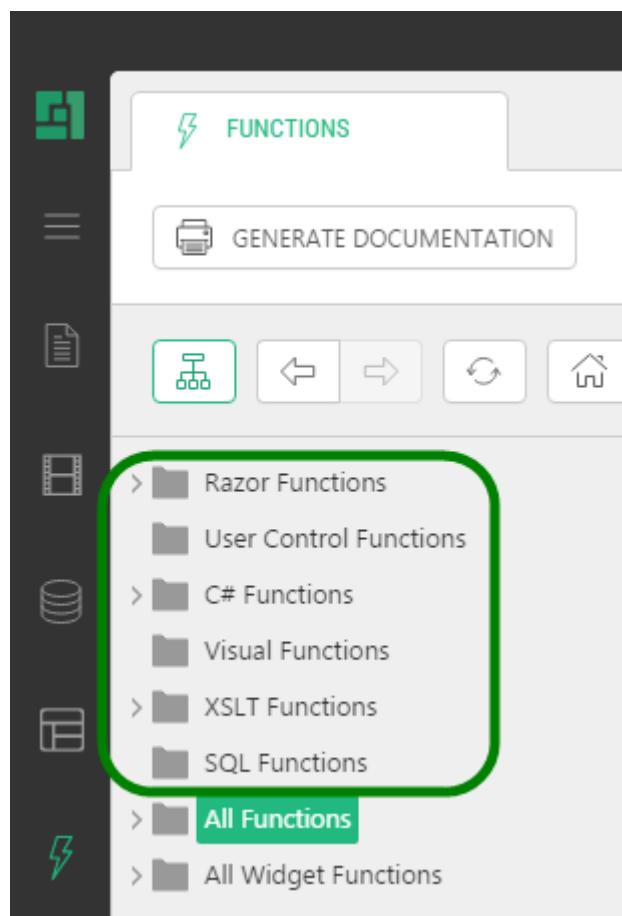


Figure 3: Six types of function

Out of these 6, Razor, XSLT and C# functions are most commonly used in C1 CMS. (This statistics might help you prioritize topics to further study if you are going to learn more about CMS functions.)

In the following few sections, most function types will be discussed in detail.

3.1 Visual Functions

Visual functions are the simplest and most user-friendly way of rendering dynamic content in C1 CMS. That, of course, comes with a price. Being the simplest means being less flexible if

compared to other types. However, if you are just starting with functions as well as data types, Visual functions might be quite helpful.

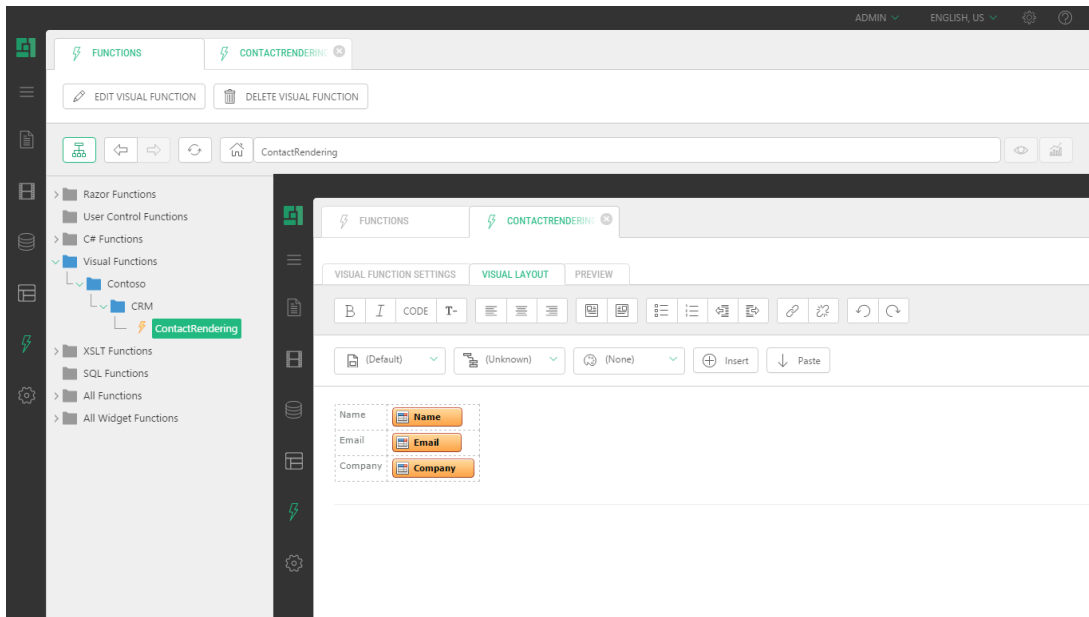


Figure 4: A Visual function

As dynamic content is based on use of data types in C1 CMS, the Visual functions are a quick and convenient way of presenting data these data types contain.

One visual function always stands for one data type. You cannot use a visual function to present data from two or more datatypes.

Visual functions present data as a list of data items. Each data item displays values in the data type fields.

Name	Jane Smith
Email	jane.smith@fabrikam.net
Company	Fabrikam

Name	John Doe
Email	john.doe@fabrikam.net
Company	Fabrikam

Figure 5: Content rendered by a visual function

The creator of a visual function can select which fields to use for rendering as well as sort items by one field and limit the number of items to display.

For more information, please see the "[Guide to Visual Functions](#)".

3.1.1 Parameters of Visual Functions

You do not define parameters for Visual functions. All visual functions come with the same set of input parameters (function properties when inserted on a page):

- **List Filter:** The filter applied to the data to select items to show. No filtering by default. (Available in the Advanced view only.)
- **Item list length:** The maximum number of items to show. The default value is 10.
- **Item sorting:** The field to use when sorting items. The default value is the field in the data type set as a title. You can also use '(random)' to show items randomly.
- **Sort ascending:** When checked ("True"), items are sorted in ascending order (alphabetically or chronologically). The default value is "True". This field is ignored when '(random)' sorting is active.

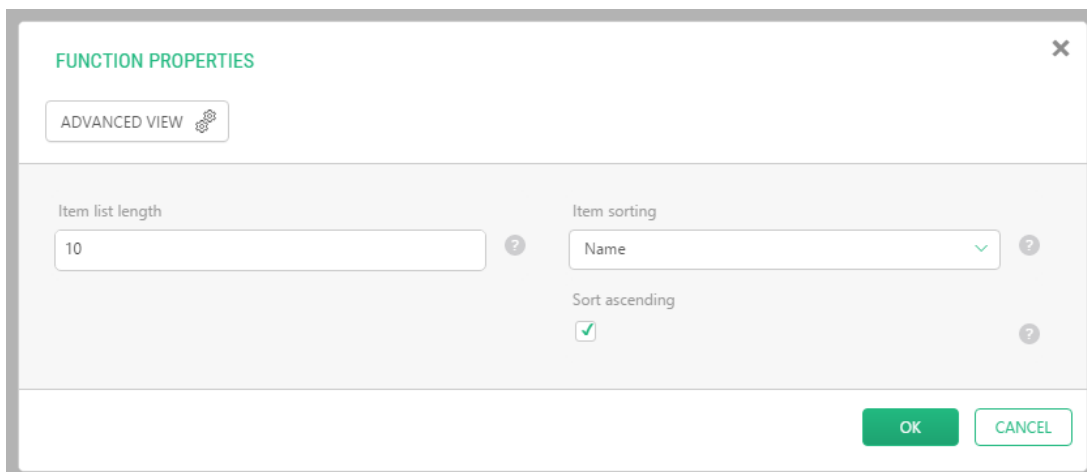


Figure 6: Parameters of a visual function (when inserted on a page)

All these parameters come with default values. However, you can override them by switching to the Advanced view and specifying constant values for, or executing function calls on, them.

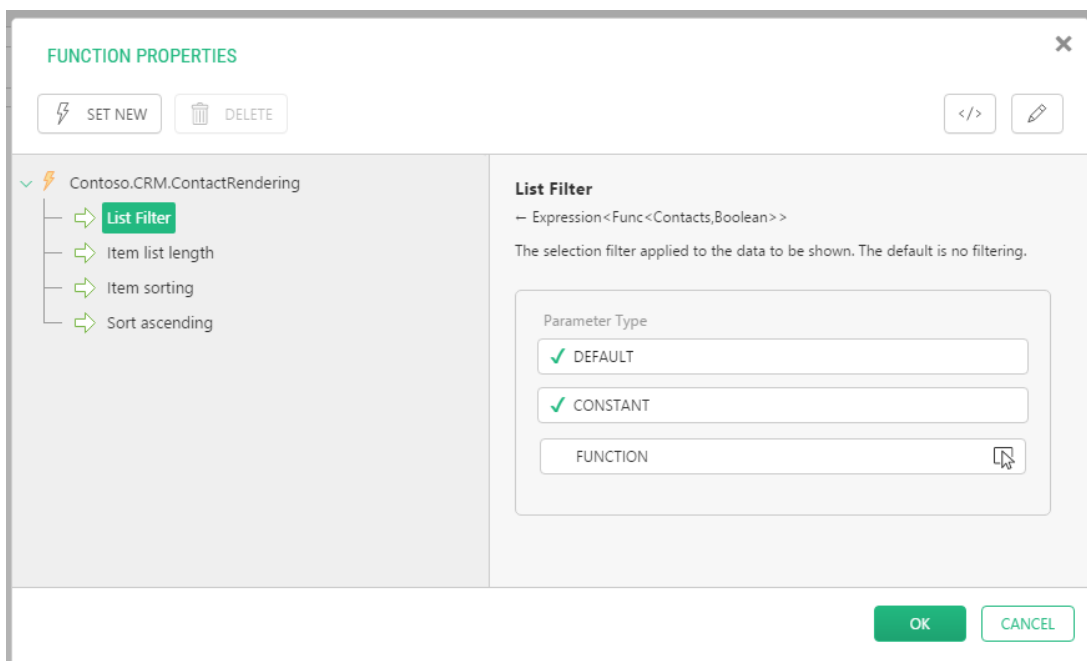


Figure 7: Parameters of a visual function in the Advanced view (when inserted on a page)

(For information on parameters in CMS functions, please refer to [How to Set Function Parameters](#) further in this guide.)

3.2 XSLT Functions

XSLT functions are key functions in C1 CMS. As XSLT is all about transforming XML, XSLT functions transform anything that can be passed to it in XML format. In terms of C1 CMS, “anything” means “everything”. All the power of XSLT is available in XSLT functions.

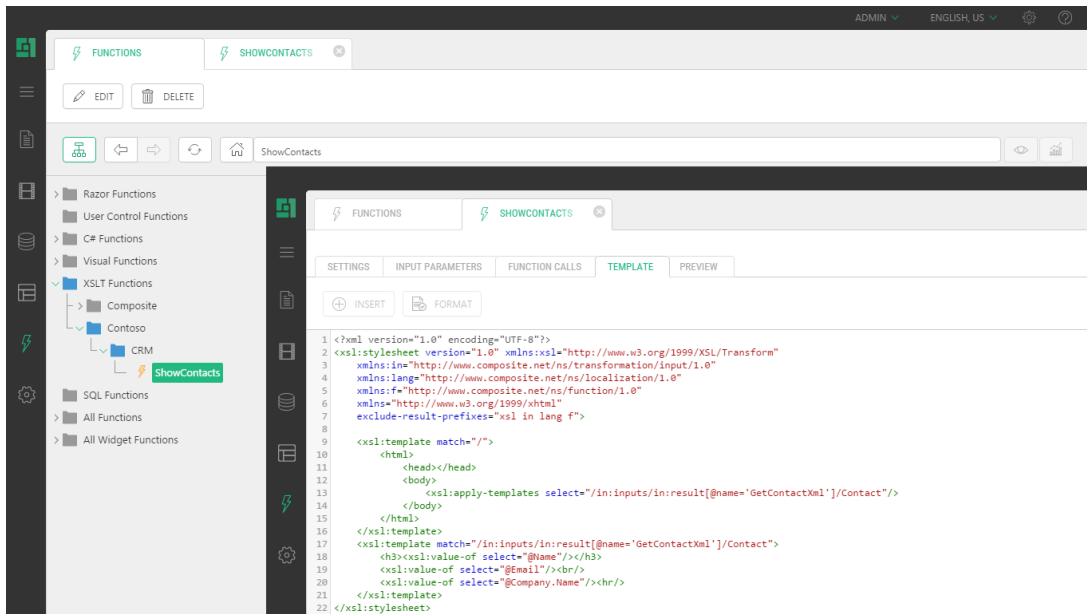


Figure 8: An XSLT function

XSLT functions have all the uses outlined in the section [Purposes of Using Functions](#).

XSLT functions can be used to insert portions of shared content in multiple destinations, for example, templates or pages. Unlike Visual functions, XSLT functions can present data from data types in a flexible and highly customizable way, not only as one-to-one field-by-field representation but also as a result of processing and transforming values in those fields and more.

Internally, a XSLT function can use outputs of other CMS functions available in the system.

For more information, please see the [“Guide to XSLT Functions”](#).

3.2.1 Parameters of XSLT Functions

An XSLT function allows its end users to customize the appearance and behavior of its output by setting its input parameters. Each XSLT function might have no, one or more parameters that differ from function to function.

(For information on parameters in CMS functions, please refer to [How to Set Function Parameters](#) further in this guide.)

3.3 Razor Functions

Razor Functions allow you to create CMS functions using the Razor syntax. Basically, you can mix in and use C# code within your XHTML.

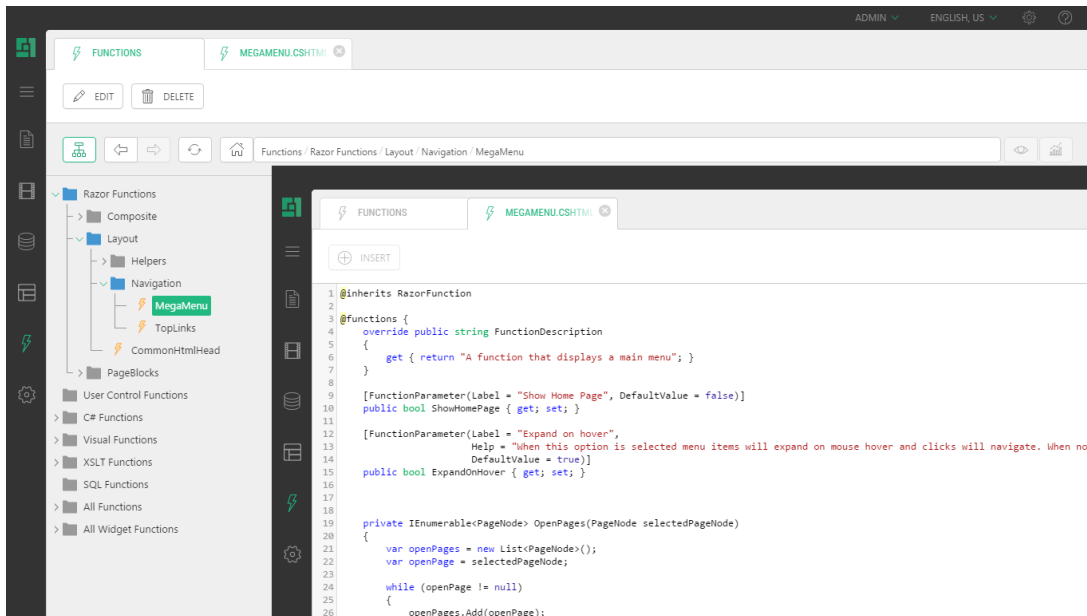


Figure 9: A Razor function

Razor Functions can make a good alternative to using either XSLT or C# functions or even both.

For more information, please see the [“Guide to Razor Functions”](#).

3.3.1 Parameters of Razor Functions

Defining parameters are different from defining them in XSLT functions.

Any public property declared within @function directive serves as a Razor Function’s parameter. And you can fine-tune these parameters with the FunctionParameter attribute.

For more information, please see [“Defining Razor Function Parameters”](#).

3.4 User Control Functions

User Control Functions allow you to turn standard ASP.NET User Controls into regular CMS functions.

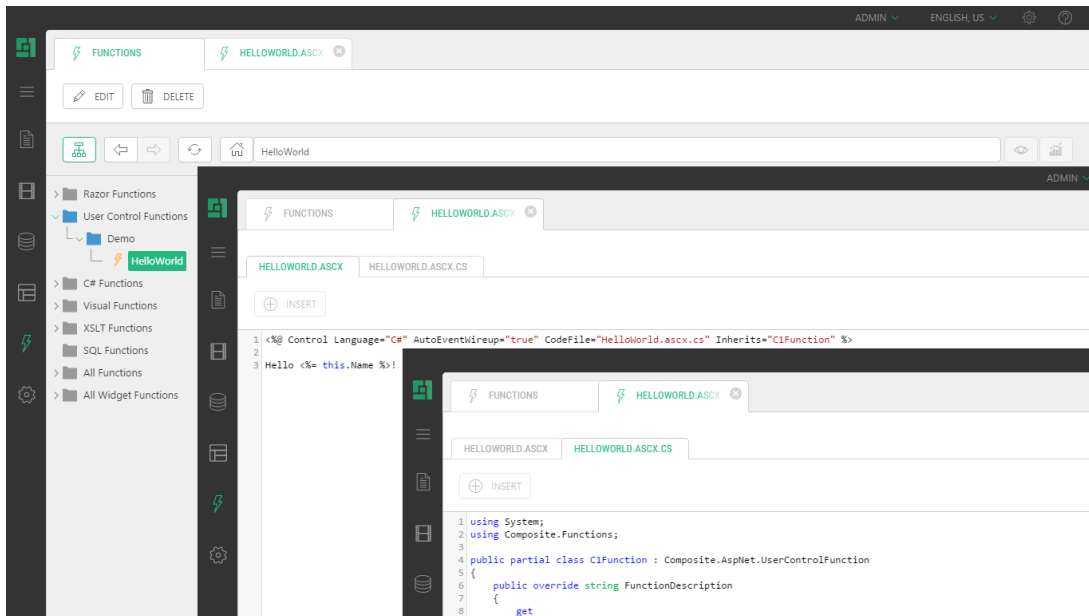


Figure 10: A User Control function

These functions are a quick and easy way of embedding your custom controls within pages, page templates and other CMS functions.

For more information, please see the [“Guide to User Control Functions”](#).

3.4.1 Parameters of User Control Functions

Defining parameters are different from defining them in XSLT functions but similar to that in Razor functions.

Any public property declared within the control class will make a User Control Function’s parameter. And you can extend these parameters with the `FunctionParameter` attribute.

For more information, please see [“Defining User Control Function Parameters”](#).

3.5 C# Functions

In C1 CMS, you can use external C# code, which, when being exposed in a specific manner, can be made available in C1 CMS as a C# function.

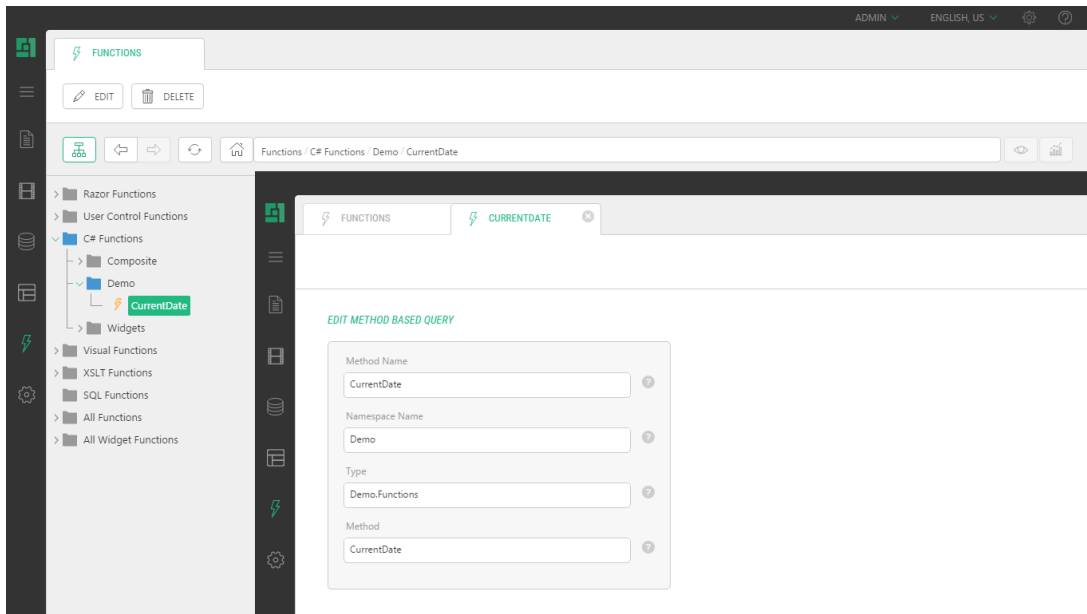


Figure 11: A C# function

With all the power of .NET Framework and C# on the one hand and programmatic access to a C1 CMS website and all its constituents via C1 CMS APIs on the other hand, C# functions proves to be highly effective tools in web development and web design in C1 CMS.

To the end user, they look nothing more than regular “CMS functions” and can be used in the same manner as other types.

Unlike Visual and XSLT functions, C# functions are more developer-oriented. You will more than often come across C# functions in function calls from within XSLT functions than in the source code of C1 CMS web page. They might be used internally in XSLT functions to perform tasks that XSLT code cannot perform.

Like XSLT functions, they can serve multiple purposes or combination thereof. They can help reuse shared content, render dynamic content or integrate external functionality within C1 CMS.

For more information, please see the [“Guide to C# Functions”](#).

3.5.1 Parameters of C# Functions

A C# function in C1 CMS represents an “actual” exposed method in a specific class (type) in C# code added to the website. Therefore, all the method’s parameters are available as parameters of such a CMS function.

Users of C# functions work with its parameters as with those of other types of functions.

(For information on parameters in CMS functions, please refer to [How to Set Function Parameters](#) further in this guide.)

3.6 SQL Functions

SQL functions in C1 CMS allow you to make queries in databases using SQL syntax.

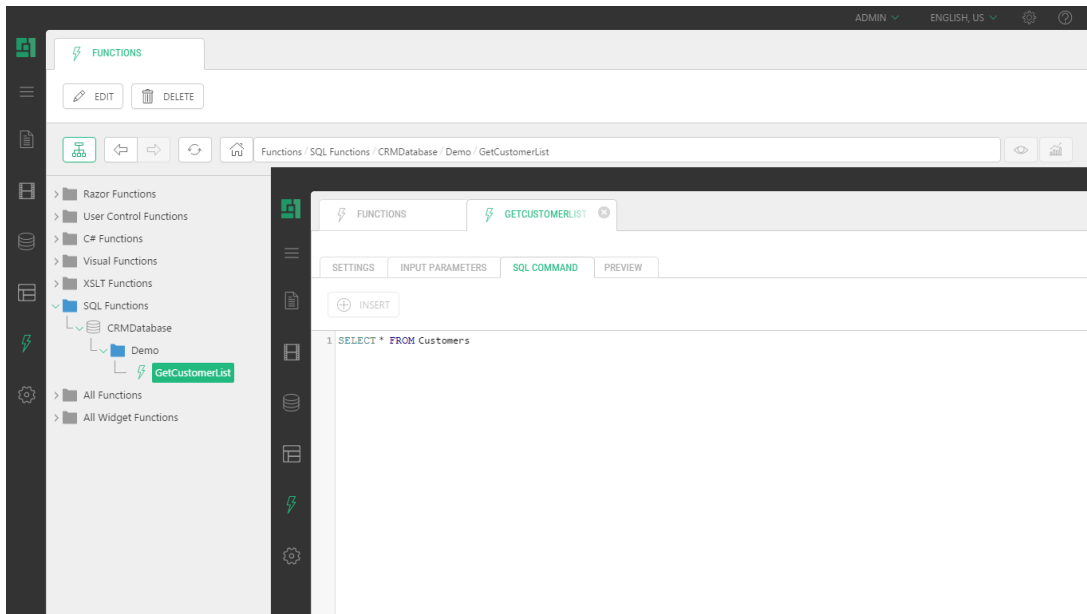


Figure 12: An SQL function

As some queries (SELECT) return data, SQL functions with these queries can return data, too. The returned data is in XML format and can be further used directly in other functions being processed and rendered or even directly on a page.

The functions can be used on any databases that support SQL queries.

You can also use SQL functions to execute stored procedures in databases.

For more information, please see the "[Guide to SQL Functions](#)".

3.6.1 Parameters of SQL Functions

An SQL function allows its users to customize its logic by setting its input parameters. Each SQL function might have no, one or more parameters that differ from function to function. The input parameters are further used in its SQL query as regular SQL variable.

Users of SQL functions work with its parameters as with those of other types of functions.

(For information on parameters in CMS functions, please refer to [How to Set Function Parameters](#) further in this guide.)

3.7 MVC functions

MVC functions are the way of integrating ASP.NET MVC applications into C1 CMS. You can register your MVC controllers and actions as CMS functions without changing your MVC applications, or you can make small changes to the latter and have C1 CMS auto-discover them.

MVC functions are not built-in in the system as the other types of CMS functions are. The MVC functions are available in C1 CMS via a special add-on. In the GUI, they are only visible in the Insert Function window and not available from the Functions perspectives.

For more information, please see the "[MVC Functions](#)".

4 How to Use Functions

Based on what type of editor in C1 CMS your work in - visual or markup, the way you [insert functions](#) and [edit their properties](#) will differ.

In Visual Content Editor, you work with functions via GUI. The functions are represented with a special graphic object in the content and you edit its properties in ad-hoc property windows.

In Markup Editor, you normally insert and edit the [function's markup](#).

4.1 Inserting Functions

To insert a function:

1. Place a cursor where you want the function to appear.
2. On the Insert menu, click Function (or Function Markup).
3. In the Select Function window, expand the proper nodes and select the function.

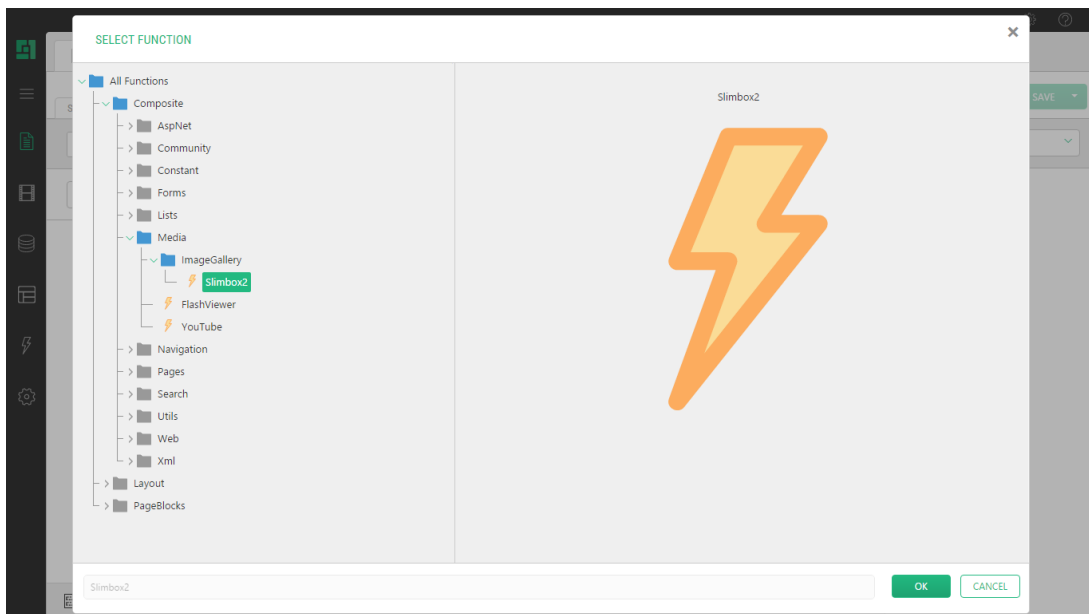


Figure 13: Selecting a function to insert

4. Click OK.
5. In the Function Properties window, specify values in the parameters one by one.

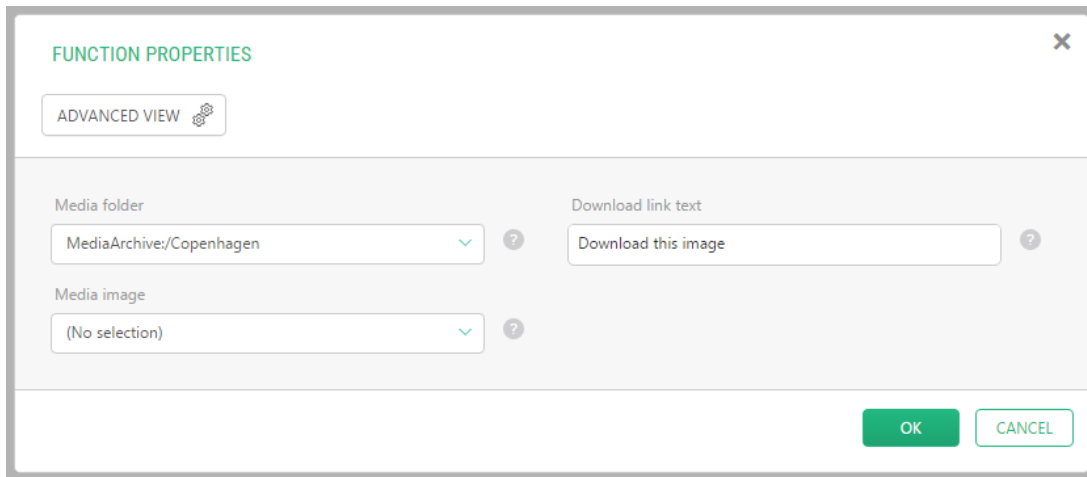


Figure 14: Parameters of a function to insert

6. Click OK.

Please note that if the function has no parameters, it appears on the page or in the markup immediately after Step 4.

You might as well add the function to the source code manually as markup.



Figure 15: Inserting the function's markup

Please note that a lot of functions in C1 CMS come with add-ons and you first have to install them to use their functions (such as the Composite.Media.FlashViewer function in the figure above).

4.2 Editing Function Properties

Editing function properties implies modifying its parameters. In Visual Content Editor, you can use the Function Properties window to make necessary changes. However, you should edit the markup directly in the markup editor.

To edit function properties in the Visual Content Editor:

1. Select the function's graphic representation on a page.
2. Click Function Properties on the tab's toolbar, or Edit on the function preview.

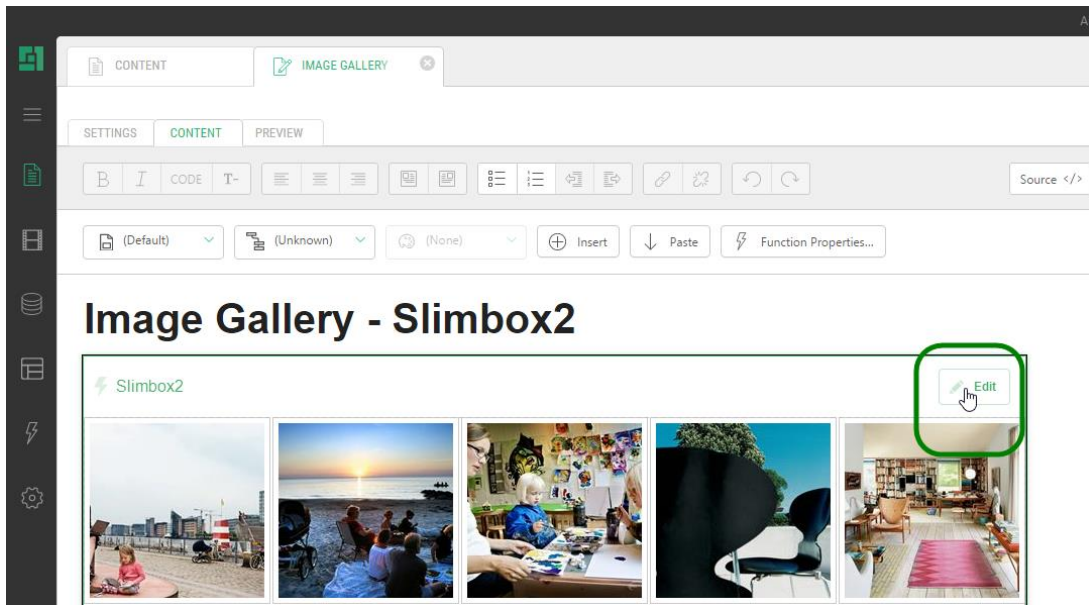


Figure 16: Editing the function's properties

3. In the Function Properties set or change the values of parameters where necessary.
4. Click OK.

To edit function properties in the markup editor:

1. Locate the function in the markup.
2. If necessary, change the value of a parameter in the value attribute of the f:param element.
3. If necessary, add or remove one or more f:param elements that stand for function parameters.
4. Save the markup.

Please note that you can learn about a function and its parameters by referring to the function's [documentation](#).

4.3 Errors in Functions

When something in a function is set incorrectly, you might experience the so called "error in a CMS function" when viewing it in a web browser.

[Error]

Figure 17: Error in a function on a page

To be able to see detailed information about the error, you need to be logged in to the CMS Console.

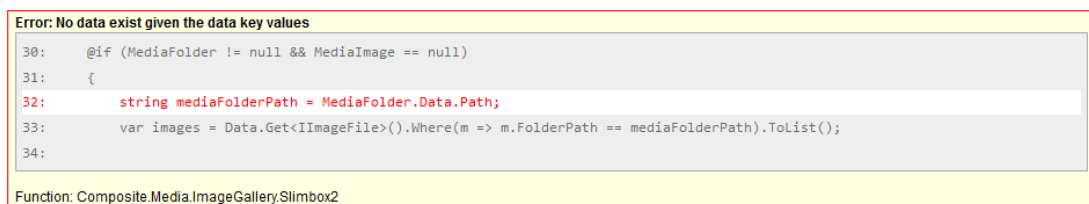


Figure 18: Error in Function message

In this case, you should, first of all, check if you have set all the required parameters, the values are valid, and the function is inserted correctly.

4.4 Testing functions

From the “Functions” perspective, you can also test certain aspects of a CMS functions. You can preview its result - if any - with:

- predefined test values
- your own values
- a different runtime:
 - a different page
 - a different scope
 - a different language

To test a CMS function:

1. From the “Functions” perspective, expand “All Functions”.
2. Locate and select the function you need by expanding its respective namespace.
3. Click “Test Function” on the toolbar.

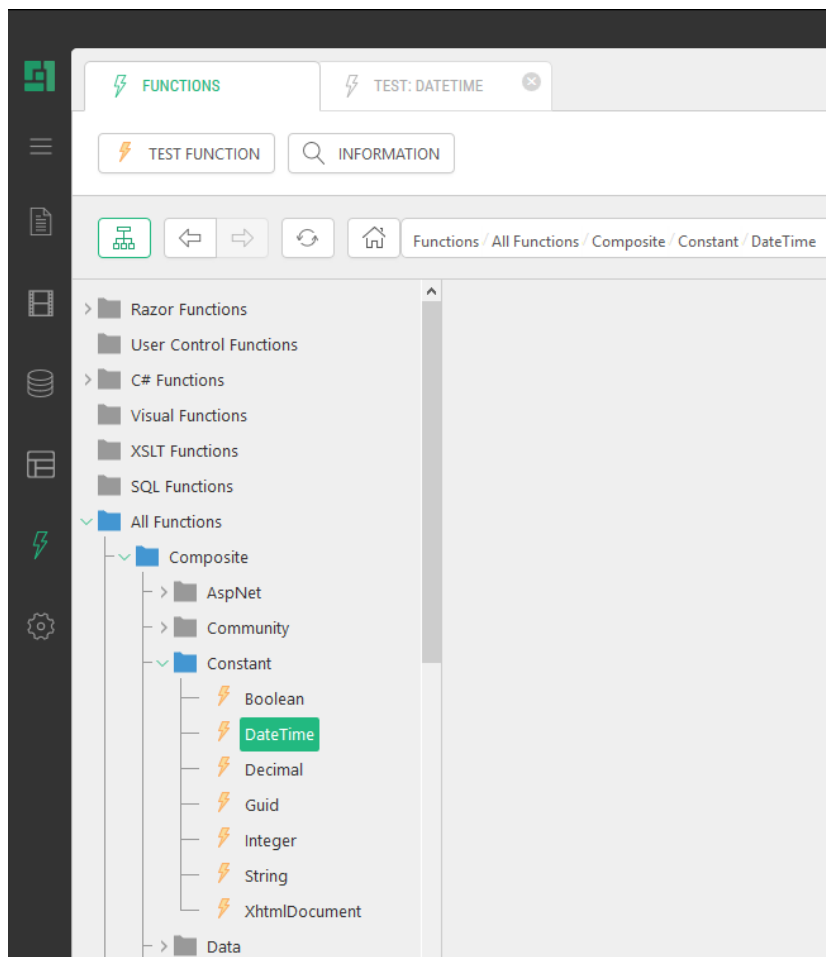


Figure 19: Testing a function

The test view of the function will open.

On the “Runtime” tab, keep default test values in the Page, Scope and Language fields or override them with your own ones.

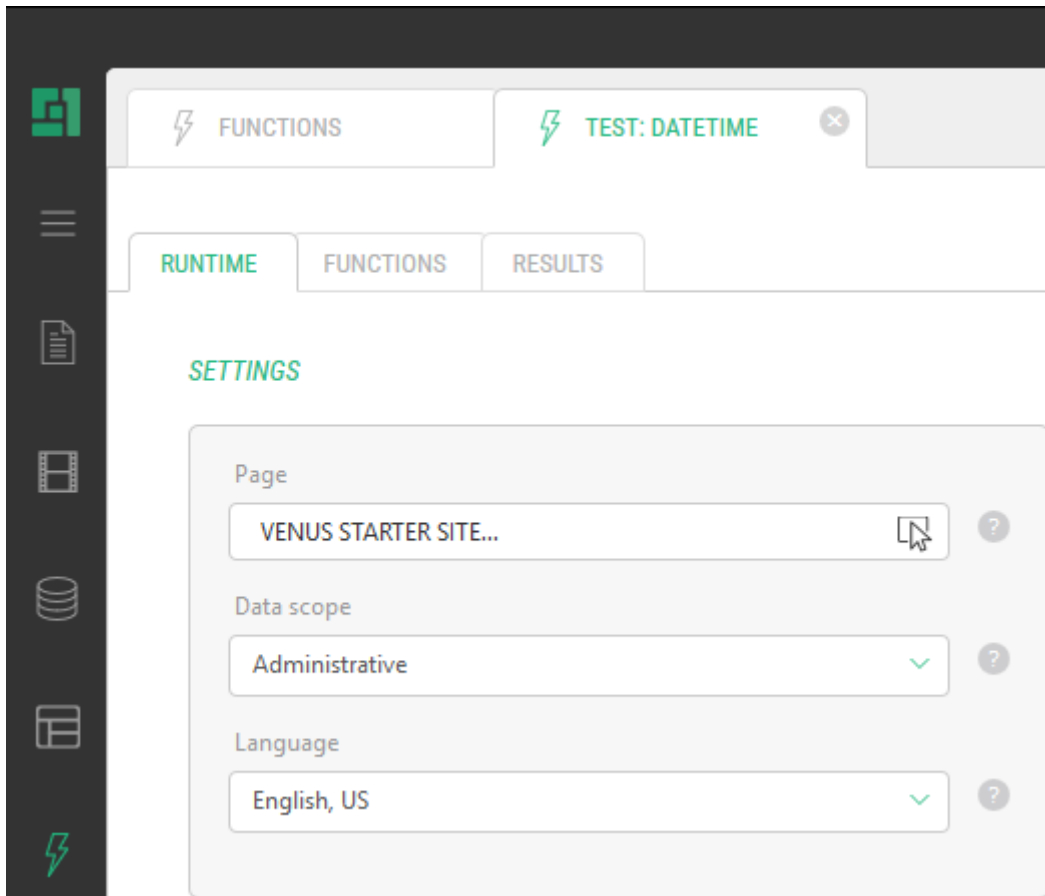


Figure 20: Test Function - Runtime

On the “Functions” tab, keep the default test values of the function parameters or override them with your own. Here you can also add one or more functions if necessary, for example, to see how they interact.

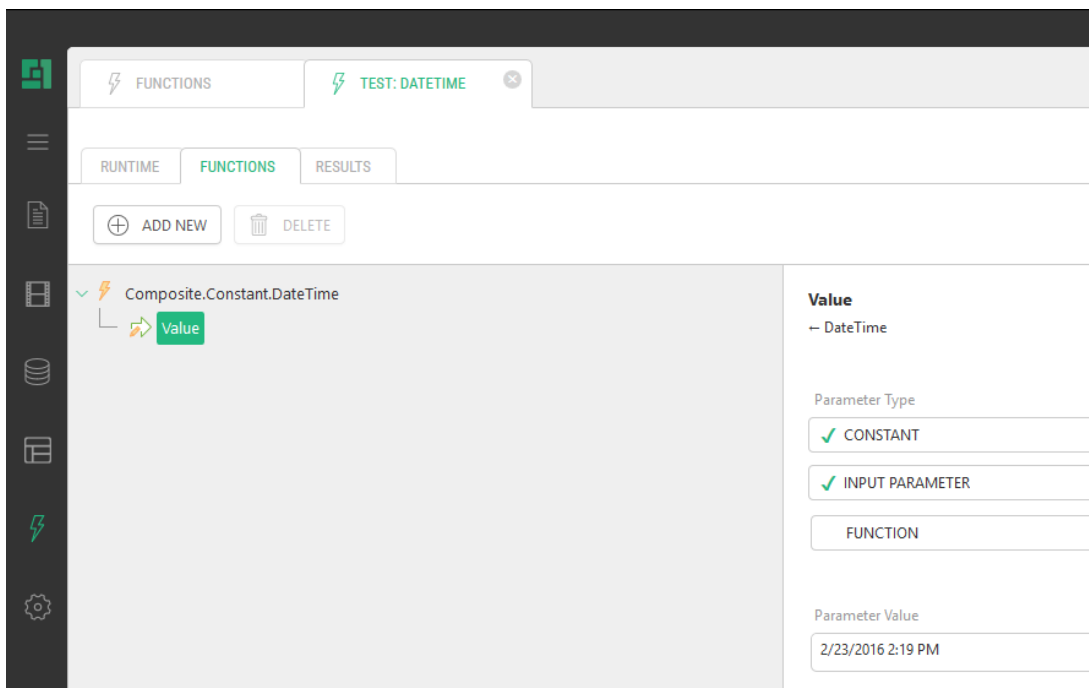


Figure 21: Test Function - Functions

On the “Results” tab, you can see how the function’s output with the specified runtime and values.

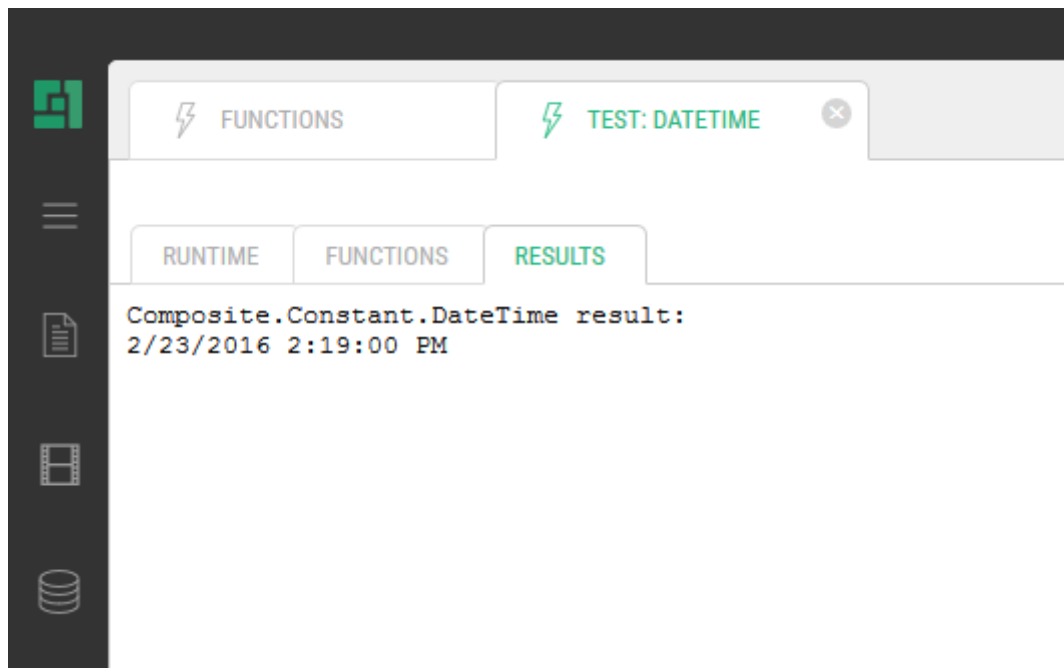


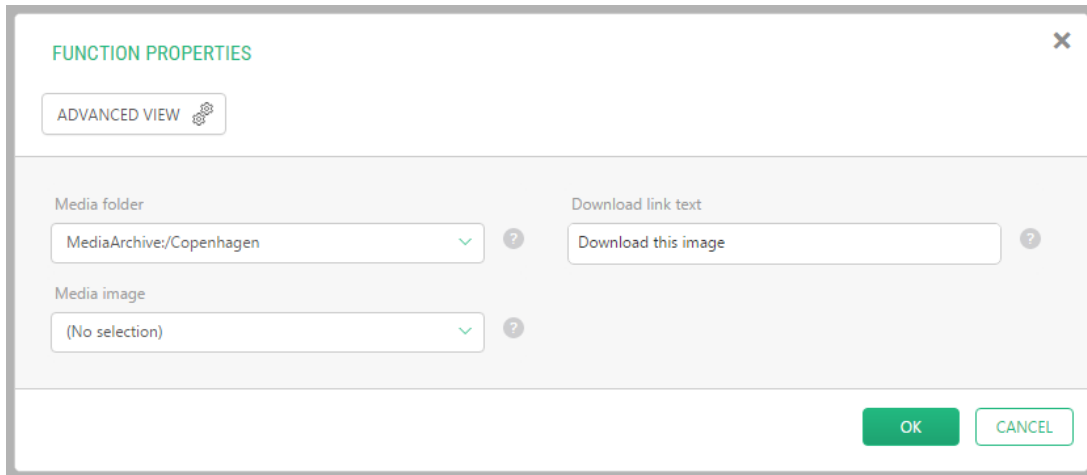
Figure 22: Test Function - Results

Here on this tab you can also see errors in the function - if any.

5 How to Set Function Parameters

Many functions require some information from the user before executing their logic. The user passes the required information to the functions via function parameters.

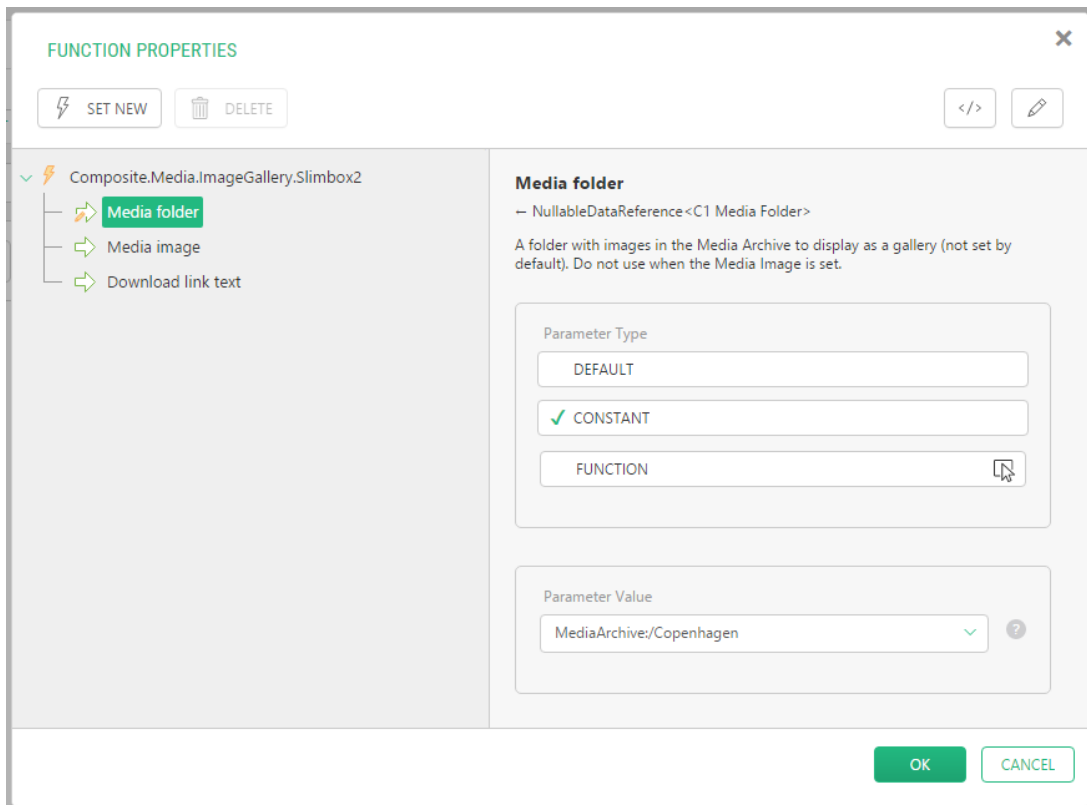
By default, the function parameters are presented in the Basic view where you can quickly set the parameters in the corresponding fields and override the default values.



The screenshot shows a dialog window titled "FUNCTION PROPERTIES" with a close button (X) in the top right corner. Below the title bar, there is a button labeled "ADVANCED VIEW" with a gear icon. The main area contains three dropdown menus: "Media folder" with the value "MediaArchive/Copenhagen", "Media image" with the value "(No selection)", and "Download link text" with the value "Download this image". Each dropdown menu has a question mark icon to its right. At the bottom right, there are two buttons: "OK" and "CANCEL".

Figure 23: Parameters of a function to insert (Basic view)

For more control over values in the parameters, you may want to switch to the Advanced view.



The screenshot shows the "FUNCTION PROPERTIES" dialog in its advanced view. At the top, there are buttons for "SET NEW" (with a lightning bolt icon) and "DELETE" (with a trash icon), and icons for code editing (</>) and editing (pencil). On the left, a tree view shows the function "Composite.Media.ImageGallery.Slimbox2" with three parameters: "Media folder" (highlighted in green), "Media image", and "Download link text". The right pane is titled "Media folder" and shows the parameter type as "NullableDataReference<C1 Media Folder>". Below this, there is a description: "A folder with images in the Media Archive to display as a gallery (not set by default). Do not use when the Media Image is set." The "Parameter Type" section has three radio buttons: "DEFAULT", "CONSTANT" (which is checked), and "FUNCTION" (with a copy icon). The "Parameter Value" section has a dropdown menu with the value "MediaArchive/Copenhagen" and a question mark icon. At the bottom right, there are "OK" and "CANCEL" buttons.

Figure 24: Parameters of a function to insert (Advanced view)

When you set parameters in the Function Properties window, you normally have a choice of three types of parameters:

- [Default](#)
- [Constant](#)
- [Function](#)

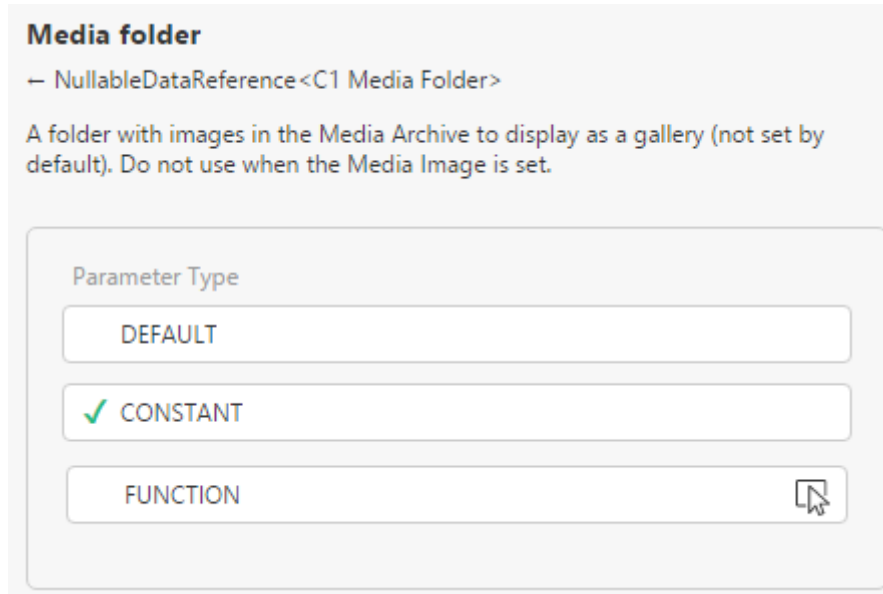


Figure 25: Parameter types in the Function Properties window

If any of those are not available for a particular function, they usually do not appear in the GUI.

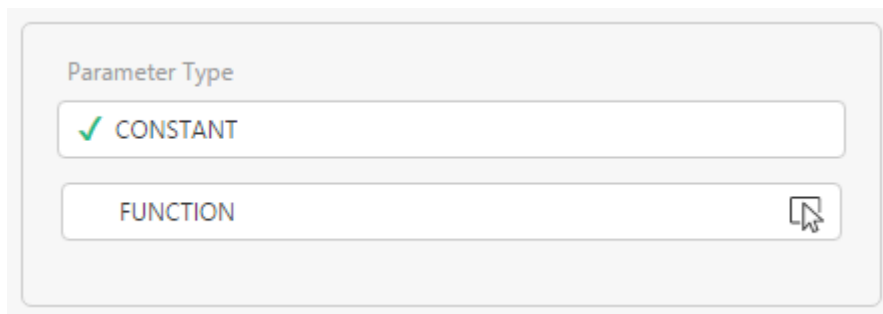





Figure 26: Default parameter type not used

Besides, the parameters might be required or optional. If at least one of the required parameter is not set, the function will not work correctly, and normally, you will not be able to insert it via GUI.

Other parameters are optional, which means that you can skip setting these parameters. They are either set to some smart defaults or not set because they extend the function's possibilities and thus can be skipped in the most basic scenario.

Required parameters are normally marked as  in the GUI, while optional parameters are indicated as . When the value of the required parameter is set or the default value of an optional parameter is modified, the parameter is marked as .

To set a parameter of a function:

1. Insert or edit a function.
2. Select a parameter.

3. If necessary, change the parameter type by clicking the corresponding Parameter Type button.
4. Specify the value.
5. Click save the changes.

Please note that the currently used parameter type is normally disabled in the Function Properties window.

5.1 Default Parameters

Some functions come with predefined values for their parameters, normally acceptable for the first run or the most common usage scenario. When you insert such a function, it sets its parameters to these defaults.

However, you can always override the default values:

1. [Edit the function properties](#).
2. Change the parameter type from the default value to a [constant](#) value or a [function call](#).
3. Set the new value by replacing the existing one if necessary.
4. Click OK.

Tip: If you want to view what value is used by default, you can change the default parameter type to the constant one. Normally, the constant value contains the default value on optional parameters.

5.2 Constant Parameters

Constant parameters require that a value of specific type should be specified for the function to work properly.

There are several types of constant parameters, each represented with a corresponding widget in the GUI. They include but are not limited to plain text strings, numerals, options, lists and C1 CMS entities such as pages, media files or data types as well as XML or XHTML formatted texts.

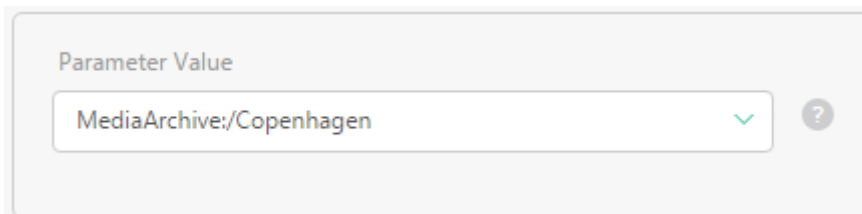


Figure 27: A constant parameter (a media file)

(For information about widgets in general, please refer to [Widgets](#) in “A Guide to Creating Data Types”.)

5.3 Function Calls

Not only can default or user-defined constants be values of the parameters. In some cases, the value of a parameter must be calculated based on the current context at the moment the function renders content.

That is why you can use other functions to set values of the parameters. The function calculates the value and returns it to the parameter.

It is not uncommon that the function used to set a value can in turn has its parameters set by other functions.

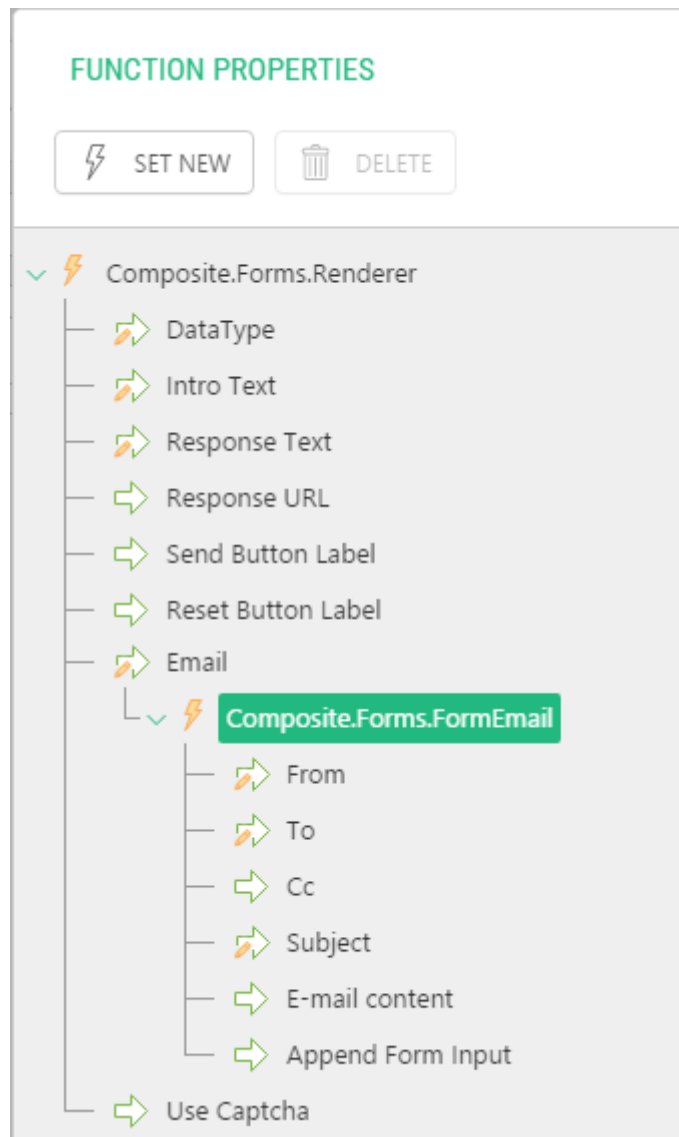


Figure 28: A function call as a parameter value

To use a function call on a parameter:

1. Select the parameter in the Function Properties window.
2. Click Function to change the value type of the parameter.
3. In the “Value for Parameter...” window select the proper function (as you do when [selecting a function to insert](#)).
4. Click OK. The function appears as a child element to the parameter element.
5. If required, select the parameters of the function that set the value, and sets their values, too.
6. Click OK.

6 Function Documentation

The functions and their parameters are well documented in C1 CMS.

6.1 Help Texts

When you insert functions or edit their properties, you can always click the help button next to the parameter's field and read its description.

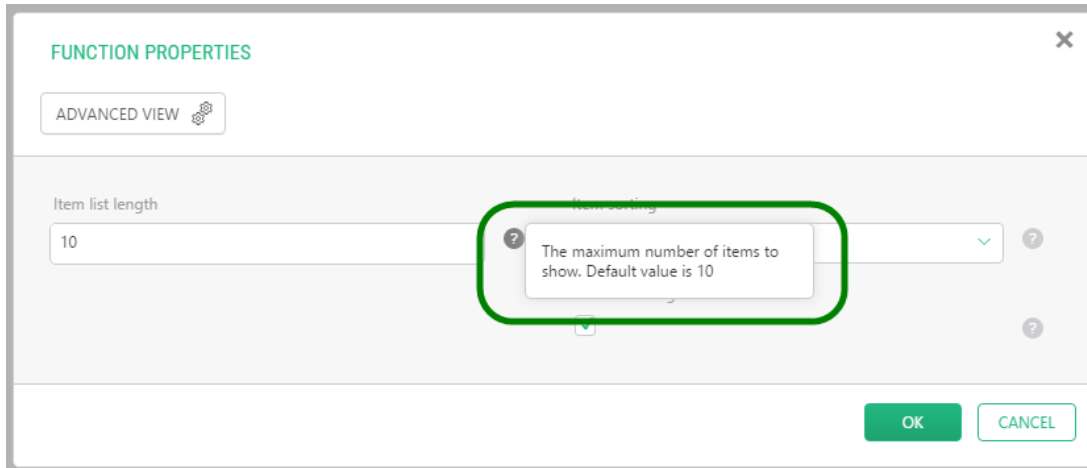


Figure 29: Description of a parameter in the Function Properties window (Basic view)

In the Advanced view, you can also read the description of the function and its parameters in the Function Properties window.

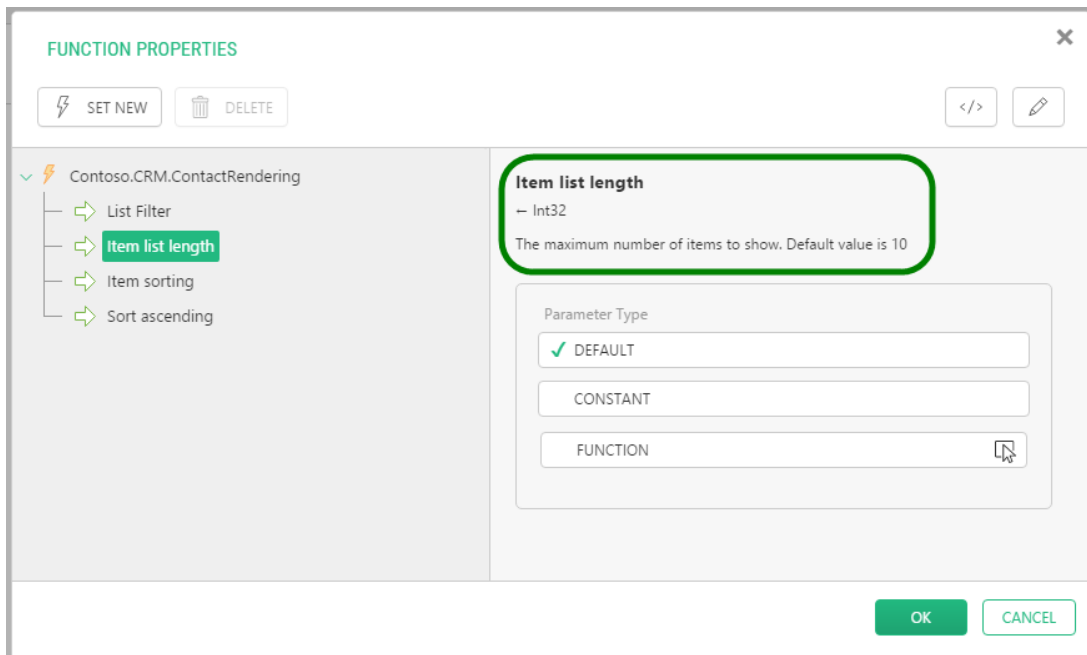


Figure 30: Description of a parameter in the Function Properties window (Advanced view)

6.2 Function Information

In the Function perspective, you can read information about a specific function and its parameters:

1. In the Functions perspective, expand “All Functions”.
2. Locate and select a function you like.
3. Click “Information” on the toolbar.

The information will appear in the right-pane view.

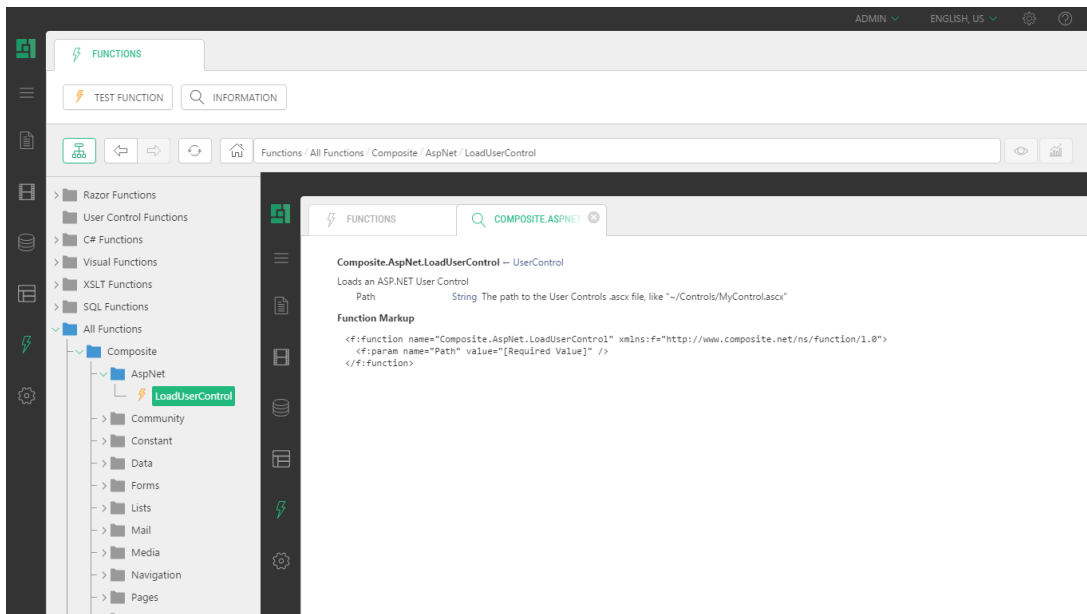


Figure 31: Information about a function and its parameters

6.3 Generated Documentation

When you insert or edit a function’s markup, you can look up information on the function in “Generated Documentation”.

To generate the documentation:

1. In the Functions perspective, select “All Functions”.
2. On the toolbar, click “Generate Documentation”.

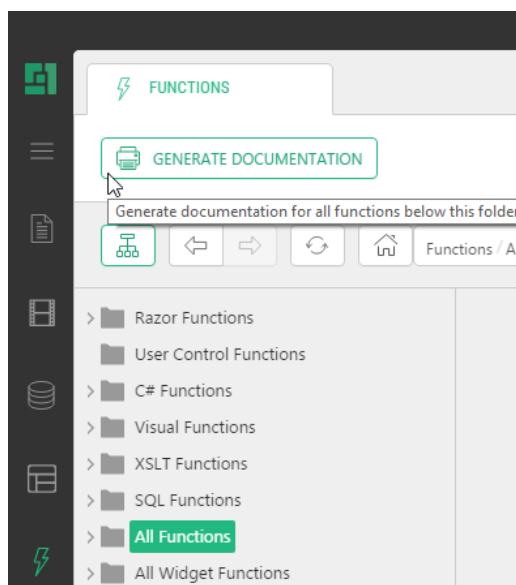


Figure 32: Generating documentation for all the functions

The documentation will appear in the right-pane view.

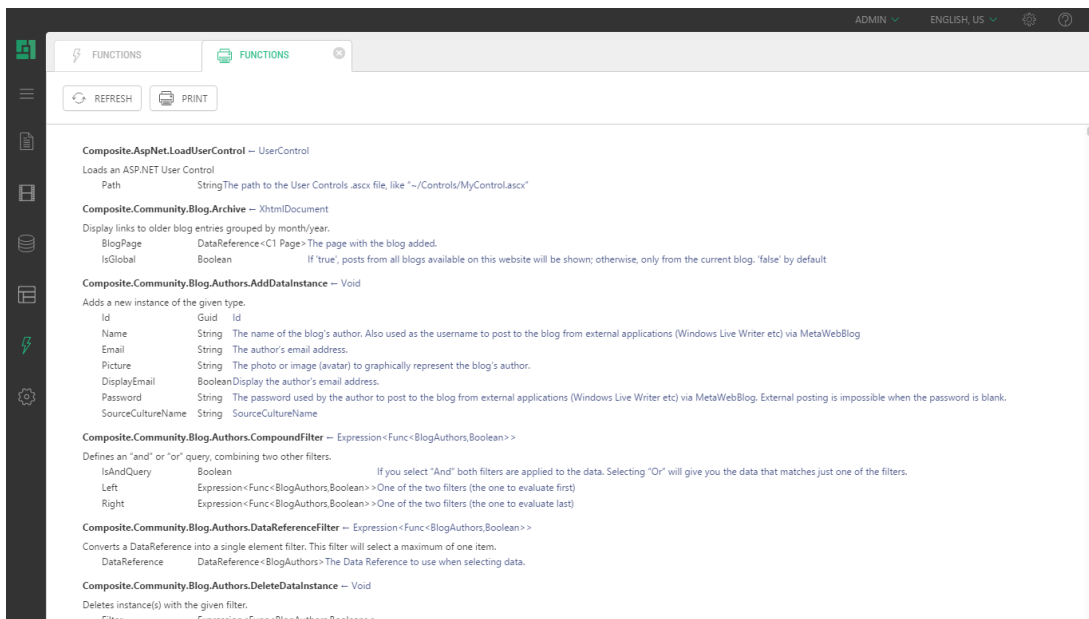


Figure 33: Function documentation

Normally, you do not need information on all functions, so you can generate the documentation on a specific namespace in the same manner as described above.

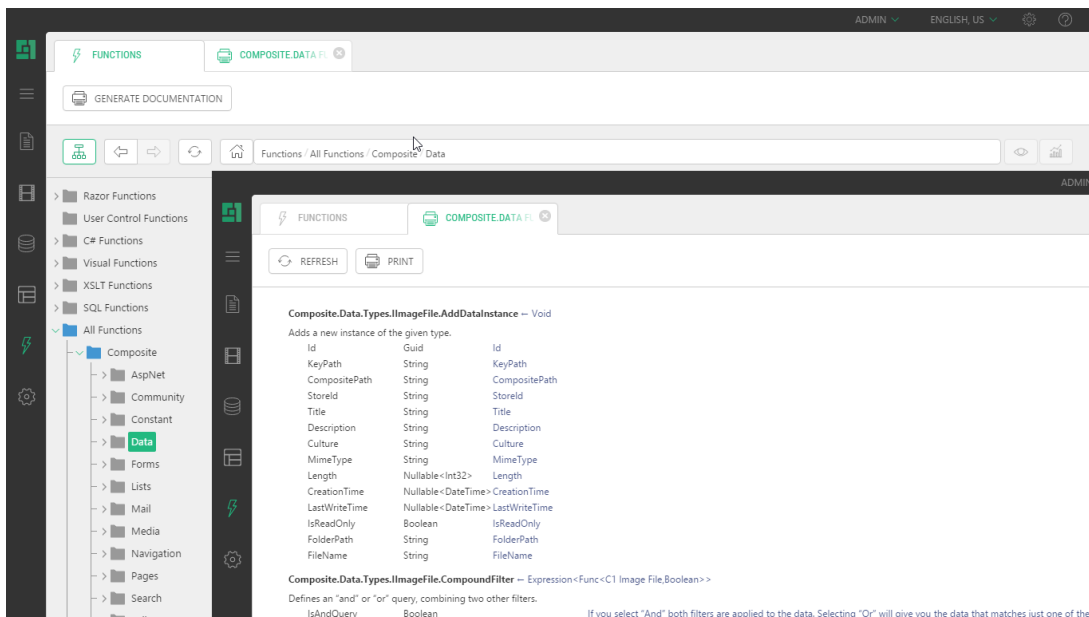


Figure 34: Generating documentation for selected functions

7 Test Your Knowledge

7.1 Task: Insert a function without parameters

1. Create a page in the Content perspective.
2. Insert the function `Composite.Pages.QuickSitemap` on the page.
3. Preview the page.

7.2 Task: Insert a function and set its required parameter

1. Install the RSS Reader add-on from the System perspective.
2. Create a page in the Content perspective.
3. Insert the function `Composite.Feeds.RssReader` on the page.
4. Set its required RSS Source parameter to some RSS feed.
5. Preview the page.

7.3 Task: Override the default value of a parameter

1. Edit the page where you have inserted `Composite.Feeds.RssReader`.
2. Edit the function's properties.
3. Override the default value of the RSS List Length parameter setting it to a greater number, e.g. 10.
4. Preview the page and check for the changes.

7.4 Task: Insert a function's markup and modify a value

1. Edit the page where you have inserted `Composite.Feeds.RssReader` and switch to its Source.
2. Copy the function's markup.
3. Create another page and switch to its Source.
4. Insert the copied function's markup on the page between the `<body>` and `</body>` tags.
5. Replace the values of the `RSSSource` and `ListLength` parameters with some other values.
6. Preview the page and check for the changes.

7.5 Task: View information on functions

1. In the Functions perspective, generate documentation for `Composite.Xslt.Extensions` functions.
2. View the descriptions of the functions.
3. View information on `Composite.Feeds.RssReader`.