smartest™

user's guide

smartest™
user's guide

Creating, editing and publishing web content with the Smartest Web Content Management Platform

Smartest: User's Guide

Written by Marcus Gilroy-Ware

With thanks to: Chris Brauer, Eddie Tejeda, Sereen Joseph, Nancy Arnold.

For use with Smartest version 0.3.7 beta

Printed in the UK

Published by VSC Books, a part of VSC Creative Ltd.

Smartest is a project of VSC Labs, a part of VSC Creative Ltd.

Please visit: http://www.vsclabs.com/

VSC Creative is a Limited Liability Company registered in England & Wales, company number 05746683, registered address: 57B Vale Rd, London N4 1PP, United Kingdom.

Table of Contents

Introduction

This manual is written to help all users of the Smartest Web Content Management Platform with a variety of aspects of its use. From installation and setup, to building your website, to everyday maintenance and content adding, deleting and editing.

This manual is not designed to be read in order. If there are specific aspects of the system that you want to learn more about, you can skip freeliy to those pages without needing necessarily to read everything before them.

Intelligent Formatting

Learning a new system can be daunting, and names of features, screens, and even sections and chapters in the manual can be confusing at first. In writing this manual, we've used text formatting to help you keep track of what we're talking about.

- Every time we mention anything important, we'll format it as **bold.** You can find most of these terms in the glossary.
- If the term we are using refers to a specific page or user-interface feature of Smartest, we'll additionally use **underline**
- If the term we are using refers to a more abstract feature of Smartest - including those that exist on or over several screens, we'll use ***italics***.

Roles

To make it easier for the different users of Smartest to find the bits of this manual that will be most relevant to them, we have labeled every section with one or more of three different roles.

These roles may all be the same person, or they may each be distinct members of a web team. We do encourage all users to read and understand as much of this manual as they can. We will use the colored icons you see next to the roles throughout this manual to clarify which content is most important for whom.

See the following page for an explanation of each of the three roles.

Content editor

You are responsible for the content of the website. You are adding and removing pages, files, and ensuring that the site is kept fresh and up to date. You will need to understand the basic concepts within Smartest, such as ***Page Elements Structure***, ***Placeholders***, ***Containers***, etc.

Site developer

It is your job to get the Smartest website ready once Smartest has been installed. Your responsibilities include coding, **HTML** creation and editing, and/or some or all of the design work. You may well also oversee the maintenance as a technical support and authority. You hopefully have experience programming in **PHP**, though you should read sections adressed to this role even if this isn't the case.

Site administrator

You are comfortable with many of the advanced aspects of using a web server, and are likely a server administrator or web technician. You know what **PHP** is, you know how to use a command line and are familiar with basic computer science concepts like system resources and **caching**. You also have responsibility for security, including permissions, users and groups.

Important concepts and paradigms

Everytime we are introducing or summarising any of the most important concepts in Smartest, we will give the text a pale blue background like this box to help you identify it.

Tutorials and 'how-tos'

Throughout this manual, in addition to explaining the abtract features of Smartest and giving you specific examples, we'll show you how to actually build a brand new site, from start to finish. We'll color these sections with an orange background like this paragraph, so that you can find them easily.

The case for dynamic HTML

Viewing an **HTML** page simply involves your computer downloading a file from a web server, before following the instructions contained in it and displaying the result.

The file you download is either stored on the server as a pre-made file, which would be an example of **static** content, or it can actually be created automatically and spontaneously by the server, based on instructions given to it, which is known as **dynamic** content. Your web browser neither knows nor cares if the page was generated or already existed.

Dynamic pages are usually generated in response to input from users. For example, if you go to Amazon.com and search for books by George Orwell, the server will search a huge database of books, and compile and send a list of any information it finds in the form of HTML. HTML, no less, that exactly matches the style, layout, and accessibility of the page you'd see if you had searched for any other author.

No Amazon employee needed to sit there for hours and create author-specific search result pages for every single author, because the web server is *configured* to do it automatically, and send it to your web browser. This configuration of the web server involves giving it instructions about how any of the pages should look without referring to qualitative (ie which author), or quantitative (ie how many books) aspects of the final result.

The developer simply gives basic stylistic (ie "put the jacket of the book on the left hand side") and mathematical (ie "if there are more than 10 books, separate the results into two pages") instructions about the page and lets the server do the rest, using these ingredients.

This way, when Amazon update their design or layout from time to time, they most likely do not have to go through hundreds of pre-existing search result pages making the same change over and over. They make the change once, to one set of instructions, and it is immediately reflected on all of their search results.

When maintaining any other type of website that consists of more than a few pages, the principle is the same. Purely static HTML cannot ever have this degree of efficiency. In order to allow for a wide variety of different things to be published on our site, instantly and at will, we have to break the design of our pages into the special types of instructions that are described above.

These instructions are called templates, and they are a key feature of Smartest, and every other content management system. In short, they are inherent to the very idea of content management.

1. Getting Started

Planning Your Website

Smartest comes more or less without a shred of **HTML**. The point of this is that site designers and developers bring their own designs and ideas into Smartest and bring them to life, rather than dressing up their ideas with "themes".

Know what you want

Obviously, it's important before building a site that you have a clear idea of what the site will be about. What will the pages say? How will they be arranged out in the **site's hierarchy** (See **site hierarchy, chapter 3**). Will they all look the same?

Make a Site Map

Plan which information should be stored on which pages. A sensible site structure is one of the most important usability factors for any website. If in doubt, always split pages up into smaller, more specific pages rather than trying to cram everything onto a single page.

Know How Smartest Works

In order to do build your site in the most efficient way, it's good to already have an understanding of how Smartest works, and what it has to offer. How, for instance, can you make placeholders work for you?

Although Smartest is intuitive and we strive to make it do things your way, learning any new system can take time, and one of the best ways to learn is to experiment.

Design for Optimal Flexibility

When creating a visual mockup for your site, if you haven't already done so, try to visualize how you will make best usage of Smartest's content management elements such as ***Placeholders*** and ***Containers***. If this is your first website, why not flick to that page now to learn more about these features.

Multi-Site Functionality

Smartest is built to manage several websites in one single installation.

Files, such as images and text, as well as the inputs such as Containers and Placeholders (see chapter 3) can optionally be re-used across all of your sites, and any user can be selectively granted permissions to work with each and every site.

This capability also means that migrating all of your websites from one server to another is much more straightforward.

Installing Smartest

Obtaining Smartest

First, download Smartest from http://sma.rte.st/.

What you will get is a **zip archive**, either in ZIP or TAR/GZIP format. Either way, make sure that this archive is expanded and uploaded to the server. If you have SSH access to your server then you may find it easier to upload the archive, then uncompress it there, in which case the TAR/GZIP format is preferable.

Alternatively, if you like to live dangerously, but want to benefit from bug fixes and new features as they are implemented, you can obtain the current development version of Smartest straight from its version control. For this you need **SSH acccess** to your server, and you need to have **SVN** installed on it. Then run:

```
$ svn co svn://svn.vsclabs.com/smartest
```

Running the installer

At this point, if you haven't already done so, you need to create a **MySQL** database, and a user that has permission to create tables there.

If you will be using Smartest with its own **hostname** (recommended), then you will need to make sure that hostname points to Smartest's `./Public/` directory. If not, then you will need to use a **symbolic link** to point to the `./Public/` directory.

Once you have made the `./Public/` directory accessible, you should now access Smartest in a web browser at whatever that URL is. The installer will start automatically.

Technical Notes

Important: We advise that you do not allow the **Smartest root directory** to be directly web accessible, as this will allow site visitors to browse around your configuration files which contain sensitive information.

The first thing the installer will do is ask you to do is set up the file and directory permissions so that Smartest can function.

Setting up file & directory permissions

Smartest's core requires the ability to be able to save files onto specific places the hard disk of the server, within the Smartest root directory, and come back to them later.

The first thing the installer will ask you to do is to make this possible, since it probably won't be initially because of how web servers work.

If you have SSH access, you can do this very quickly using a shell script that Smartest creates automatically for this purpose. Instructions for this are provided on-screen as part of the installer.

MySQL access

To create a new Smartest installation, Smartest needs an empty database, and an account that has `SELECT`, `INSERT`, `UPDATE`, `DELETE`, `CREATE`, `ALTER` and `DROP` permissions on that database.

Once you've set this up, Smartest will be ready to use. It's easy to mis-type sometimes, so be sure to be careful of typos and the username and password must be typed exactly as they will be used.

If you still have trouble, try connecting to the MySQL client manually (or via **phpMyAdmin** if you have it), to verify that this is possible.

Tutorial: Creating additional sites

As Smartest is aimed as much at professional web agencies as at any other users, it is expected that users will have already completed their design and converted it to one or more flat HTML files. This file should be put in the Presentation/Masters/ directory, and given a .tpl (for *template*) file suffix.

To create a new site is extremely easy. To start, click "Create New Site" underneath the sites list that you see on the first screen after logging in.

If this doesn't appear, it's because your user account doesn't have permission to create new sites. If you have permission to change your own **permission tokens**, make sure you have the "permission to create new sites" token or ask the system administrator grant it to you, and then log out and back in again.

Once on the new site screen, you'll need to enter the title, domain name, and master template of your site. The title can be anything you like. The domain name must be a **fully qualified domain** name that points at your server. The master template should be the master template file mentioned in the first paragraph of this box.

Finding Your Way Around

Smartest is designed to be as intuitive as possible, and we're always looking for ways to further improve this aspect of the system.

Logging In

To log into Smartest, first go to the log-in screen. To find the log-in screen, add `/smartest` to the end of your site's homepage URL. For example, If your site was `http://www.example.com/mywebsite/`, the URL of the log-in screen would be `http://www.example.com/mywebsite/smartest`.

Your system administrator should have given you a username and password to use when logging in. If you are the system administrator, you will have entered a primary username and password when installing Smartest.

Once you have logged in, you will see a list of available sites for you to edit. This list doesn't necessarily show all of the sites hosted in your Smartest installation, since permissions to edit each site can be administered separately.

To continue editing, click the site you wish to edit. This will take you to the Site screen for that site, which you should think of as a 'home' screen. Every section of Smartest also has its own initial screen, which you can always go back to by clicking the relevant part of the main menu.

The Site Screen

The site screen, which is always at the top of the main menu, will show you an array of icons - each one leading to control over a different aspect of the site you have chosen to edit:

- **Modify Presentation**: This option will take you to the site tree screen, where you will see the hierarchy of all the pages in your website.
- Manage Data: This option will bring up a menu of the different kinds of data that can be managed in this section, including everything in the Data Manager.
- **Browse and Upload Files**: This option will take you to the Files Repository.
- Todo List: This is a dynamically generated list of all the tasks and responsibilities that await your attention as a result of your and other users' prior actions.
- **Browse & Add Templates**: Similar to the Files Repository, this option will take you to the section where you can create, edit, upload and delete Master and Container Templates.
- **Administer User Accounts**: This option, if you have been granted sufficient privileges, will allow you to create and edit user accounts, and to control their permissions. See chapter 4 on Users and Permissions for more.

- **Modify Settings**: This option allows you to change some parameters of the site you are editing. It is only recommended for advanced users.

Navigating Around Smartest

The Main Menu

The main menu is for navigating between the principal parts of Smartest, such as between the Data Manager and the Pages Manager. This menu appears on every screen, and never changes its appearance. If you are ever lost within Smartest, you can always use this menu to take you back to the home screen, or to the first screen of whichever section you are in.

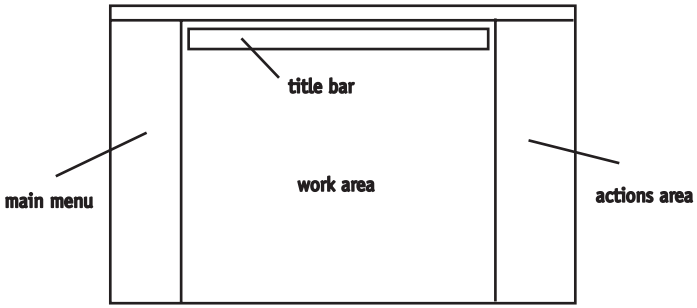
The Work Area

The work area is the primary area of user activity. All editing of pages, data, templates and other system content is done here.

The Actions Area

The actions area is a screen-specific menu of actions relating to the activity you are engaged in. This area is not present on all screens, as some screens only require activity in the Work Area. On many screens where the Actions Area *is* present, it is divided into two sections an upper and a lower. The upper section will only become visible once the user has made a selection inside the Work Area, since it contains actions that are only applicable once you have made a selection. However the lower section will always be visible if the actions area is present.

To demonstrate by example, on the **Site Hierarchy** screen, the actions in the upper section of the Actions Area are all things that relate to the selected page, like editing, deleting and adding a child page to the selected page. Conversely, the actions in the lower area are things like clearing the pages cache, releasing all pages, or leaving the site - none of which are applied specifically to any page.



A word about *caching*

It's important that all users of Smartest (and indeed of the internet) understand the concept of **caching**. A **Cache** is a place where a computer system remembers the outcome of a specific calculation that is being done very regularly, so that it doesn't have to re-do the calculation every time. This significantly lowers the about of work the computer (in this case your web server) has to do. See the glossary at the back of this manual or http://en.wikipedia.org/wiki/Cache for more.

Smartest makes heavy use of caching at various levels, including:

- Pages are cached, so that the Smartest Engine doesn't have to build pages every time they are requested by site visitors. You can control how long cached pages last for, ranging from permanent to expiring every second.
- Configuration and system data is cached so that more complex operations need only to happen once every time configuration is changed.

If you are ever expecting a page to update and it does not, or you make a change such as updating the host name of a site within Smartest, it's a good policy to empty the pages cache. Pages stored in the cache have already undergone all their processing and are pure static HTML, so they won't be up to date unless they are expire or are deleted.

2. Introducing Templates

Why Use Templates?

If you haven't yet read it, "The Case for Dynamic HTML" on page 3 is a dedicated explanation for beginners on what templates and dynamic HTML are, and why they are essential. If you aren't clear on these points, now is a good time to read it.

Trying to run a website of more than even a few pages written *purely* in HTML would be neither easy nor efficient. For example, if you were to only use static HTML, recurring elements such as navigation bars and headers could not be physically re-used on each page, so if they were changed, the same identical change would have to be made on every page of the whole website.

The most popular solution to this problem is to use a system of templates, which can be included on every page where they are needed, like using ingredients in a cake.

Types of Template in Smartest

Every page in Smartest uses one **Master Template**, which contains the most basic, universal aspects of the page such as the `<head>` and as many smaller template fragments - called **Layout Templates** in Smartest - as required.

The "Image Goes Here" Paradigm

Templates are essentially whole or partial HTML pages that are *abstracted* so that they can be re-used. The reason they can be thought of as abstracted is that not all of the details need be specified in the template file itself.

For example, rather than actually writing the HTML instruction that would produce a particular image, users can include an instruction that says "image goes here", and Smartest will then:

1) Give the user a much more user-friendly way to choose the image, and to change the image as often as they please.

2) Create the subsequent HTML instruction and send it to your browser, producing the correct image.

The template containing this instruction can then be used on various different pages, each with the potential to show a different image.

This capability will be referred to as the **"image goes here" paradigm**, and is a cornerstone of Smartest.

You may have as many Master Templates as you want in your website, although each page can only use one at a time. Generally most of what makes up the `<body>` of the page is held in Layout Templates, which are smaller fragments of dynamic HTML that can inserted as needed on different pages.

Smartest makes heavy use of the sort of templating instructions outlined above. These types of instructions are known as ***Elements*** in Smartest, and there are different, specialized kinds.

See the sections in Chapter 5 on ***Placeholders*** and ***Containers*** for more information.

Where to put your templates.

Templates often start off as plain HTML files with no dynamic elements in them at all. When you import a master template to a new site, it is highly likely that this is what you will do. To do this, place your HTML file in `Presentation/Masters/`, which is where all master templates are kept.

Container templates, used to define containers, are kept in Presentation/Layouts/. To add new container templates to the system from outside Smartest, first put them here, then go to the files repository and click 'Detect newly uploaded files'.

The Smartest Engine

Smartest's templating system is built as an extension in functionality to the widely popular **Smarty** presentation layer for PHP, using Smarty's open plugins API. The version of Smarty bundled with Smartest is completely standard and can be upgraded or replaced at any time without affecting the Smartest Engine. This means that the Smartest Engine compiles and caches in exactly the same way that Smarty would do.

Smarty has full documentation, which you should read if you are going to be writing or editing templates in Smartest. However, we will outline here the aspects of Smarty that are relevant to use in Smartest.

The template tags for the Smartest Engine look like this:

`<?sm:` and `:?>`

(opening and closing respectively).

These tags tell Smartest to pay attention to what is between them.

Template Reference

Once again, the template tags for the Smartest Engine look like this:

`<?sm:` and `:?>` (opening and closing respectively).

Template tags are a hugely important aspect of the template markup. They are to the Smartest Engine what < and > are to HTML, in that they tell the parser to pay attention to whatever is between them.

Smarty, which is the underlying technology that the Smartest Engine is built on, uses basis control structures such as looping and conditionals and other principles such as variables, that are borrowed from computer science.

These aren't at all difficult once you've taken a moment to familiarize yourself with them but if you are unsure of what variables, conditionals or looping are, please see the appendix on page 41.

Basic Syntax

Variables

To display the value of a variable in Smarty, simply write it enlosed in template tags:

```
<?sm:$variable:?>
```

Arrays can be accessed either with the dot(.) operator for non-numeric keys:

```
<?sm:$myarray.value:?>
```

```
<?sm:$myarray.subarray.value:?>
```

Or using square brackets for numeric keys:

```
<?sm:$myarray[0]:?>
```

```
<?sm:$myarray[0][5]:?>
```

Conditionals

In situations where you want the template to display something only if a certain variable has the right value, you can use a conditional `<?sm:if:?>` tag:

```
<?sm:if $myvariable == "value":?>
```

```
This text won't always display.
```

```
<?sm:/if:?>
```

Looping

If you need to do something for every member of an array, such as print a table row or bullet point, you would use seomthing like:

```
<ul>
```

```
<?sm:foreach from=$myarray key="key" item="foo":?>
```

```
<li><?sm:$key:?>=><?sm:$foo:?></li>

<?sm:/foreach:?>

</ul>
```

This would print the key and value of the array for every member of that array.

Functions

Unlike many other template languages, Smarty also has the ability to support primitive functions.

To call a **function**, enclose its name in Smartest tags like this:

```
<?sm:myfunction:?>
```

If the function requires arguments to be passed to it, you pass them as quoted attributes, similar to HTML attributes.

```
<?sm:myfunction argument="foo" second_argument="bar":?>
```

Smarty functions don't return their results as in a normal language, but simply display them. This is because Smarty is only for presentation, not for any complex type of logic.

3. Files and the Files Repository

This chapter explains to use your writing, images and other content on Smartest web pages. This covers creating, editing and deleting, as well as. It also gives an explanation of how Smartest stores and organizes these files.

The Files Repository

What is it?

The ***Files Repository*** is where you will store and keep track of all images, text, scripts, stylesheets, downloadable files, and multimedia elements that you use in your site.

It follows the same file organization paradigms that are hopefully already familiar from your PC, such as folders and icons.

Using a repository to store files means that they are centralized. This is always more efficient, but is especially more useful if you are using the same file on more than one page of your site.

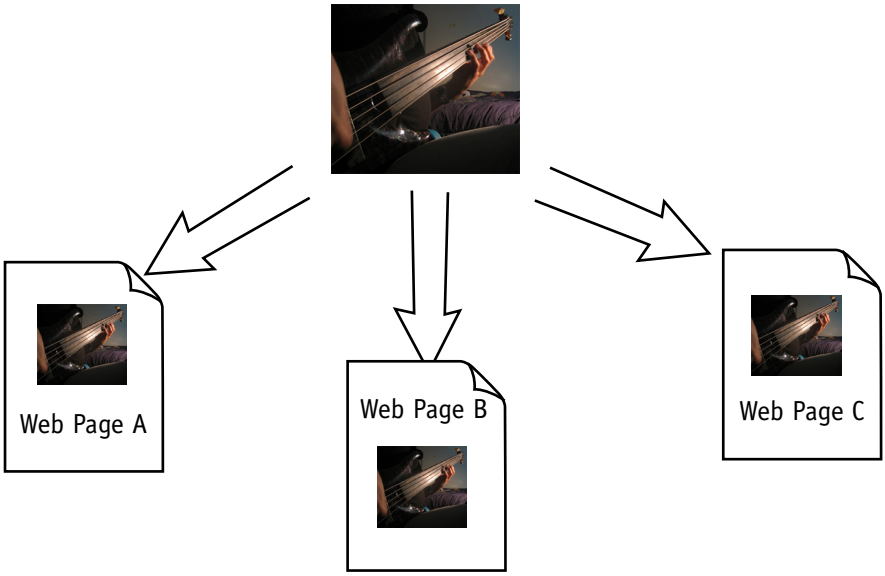


Figure 3.1 - Centralized file storage

Also, bear in mind Smartest will not 'know about' files that are not stored in the repository, so they won't appear as options in the dropdown list when you are choosing what to put on your web pages. This means that you won't easily be able to use them on your site, so it's good to get into the habit of putting things in the repository as a

starting point.

First things first: Uploading files into the repository

First, go to the start screen of the ***Files Repository*** by clicking **Files** in the **Main Menu**. To upload a file, click the file type you would like to add to, so as to highlight it, and then choose **Add another one of this type** in the **Actions Area**. Some files, such as text files, stylesheets and scripts, can be input directly, as well as uploaded.

When uploading a file, you must make sure that the file you are uploading is the same type that you have selected in Smartest's interface. For instance, if you are creating a text file, you can only upload plain text or HTML files. A PDF or an image that displays text will not work because the text in these files only makes sense to human beings, not to computers that are expecting something different.

Uploading Batches of Files

If you are uploading a large number of files at once, you can also tranfer them by FTP and click **Detect Newly Uploaded Files** in the actions area of the main screen of the repository. This will bring up a list, organized by type, of any files that have been found on the server that are not yet in the repository. Next to each file is a checkbox. Check the boxes next to the files you wish to import into the repository and click **Next**.

Now, each file whose box you checked will be displayed with an empty name field and a 'share this file with other sites' checkbox.

Get to work

How to create a new text file in three clicks!

1. From the respository main screen, select "Formatted Text".
2. Click the "Add another one of this type" option that appears.
3. Enter your text into the editor.
4. Click "Save".

How your files are stored

For each file, Smartest has a normal filename as you might see on your PC - for example, *biography.txt*. This is almost always the same as the name it originally had before you imported it into Smartest.

In addition to this, Smartest gives you the opportunity to use a more straightforward, easy to remember name for each file. This is displayed on the

Navigating the files repository

The first screen of the repository is an array of the different file types that Smartest supports. You can return to this screen whenever you need to by clicking **Files** in the **Main Menu**. Clicking a file type will highlight it, and reveal a list in the **Actions Area** of options that can be carried out with that type of file.

Browsing files

- Go to the start screen of the **Files Repository** by clicking **Files** in the **Main Menu**.
- Choose the folder icon for the type of the file type you would like to browse
- Click **Show me all of this type** in the **Actions Area**, or double-click the folder icon.
- You now see an array of all the files of that type. If you are looking for a specific file, you will find it here, and clicking it once will expand the **Actions Area** to reveal a few actions that may be carried out on it.

Editing files

Follow the instructions above to browse to the file, and select **Edit This File** in the **Actions Area**.

Though most file types have default parameters govening their display that can be edited, not all file types can be directly edited. Images, for example require external software to be edited.

The editing of those files that can be directly edited within smartest is divided into different screens. The text editor screen alllows for easy formatting (for those file types that support it, such as “Formatted Text”)

Working with text

Smartest provides some extended functionality when working with formatted text files. Images can be embedded and captioned within the text, and hyperlinks can be easily added without the need to use any HTML or other markup.

Easy Links

Links can be thought of as roughly divided into two main types: extermal links, which point to addresses of other websites, and internal links, which point to parts of the same website where you are using Smartest.

Both types can be easily created in Smartest formatted text files, using the square

brackets, `[` and `]`.

External links

External links are created using a single set of square brackets, containing a URL, followed by any text that should be embedded in the link:

```
[http://www.smartestproject.com Smartest]
```

This example will create a link that displays the word 'Smartest' and when clicked, takes the user to the address `http://www.smartestproject.com`.

If you would just like to display the URL as the link's text, simply enclose it in square brackets:

```
[http://www.smartestproject.com]
```

Internal links

Internal links are created using a double set of square brackets, containing the type of entity that is being linked to, a colon, plus the short name of the entity.

For example, to link to a page that has the short name 'home', this would be the link:

```
[[page:home]]
```

Say you had a model called Article, you could link to an article that had the short name 'world-peace-declared' like so:

```
[[article:world-peace-declared]]
```

Further, If you want to embed text in the link that is not the title of the page or item you're linking to, use the 'pipe' or 'bar' character, `|`. This example will display the text 'this article' as a link to the same article as above.

```
[[article:world-peace-declared|this article]]
```

Text attachments

In some cases it's necessary to incorporate images and other elements into your text file, for instance as a way to illustrate the ideas of the text.

In Smartest, these supporting materials are called ***Attachments***. Much like with e-mail, these files accompany the text they are attached to, wherever it appears in the end.

Adding attachments

To add an attachment, you will have to enter the **Source Editor**, since the **Rich Text Editor** doesn't display attachments.

Code

```
<p>Paragraph A</p>

<?sm:attachment name=”map_1” :?>

<p>Paragraph B</p>
```

Resulting Layout

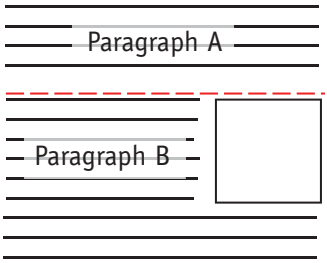


Figure 3.2 - placing floated attachments

There, you'll see the bare **markup** that gives the text its formatting. In most cases, markup consists largely of **HTML Tags.**

For those that are unfamiliar with HTML, paragraph tags are a way of grouping bits of text together that would grammatically be considered to be the same paragraph. They look like this:

```
<p>This is some text</p>
```

In the case of text that has been pasted from Microsoft Word, you may see paragraph tags that look like this:

```
<p class="MsoNormal">This is some text</p>
```

The class=”MsoNormal” tag can be deleted or ignored as it has no effect on how the text is displayed, and thus is only taking up space. Thanks, Microsoft!

Attachments are added by inserting a tag similar to those used in templates (for which the complete syntax and reference is included at the end of this chapter). Attachment tags look like this:

```
<?sm:attachment name="attachmentname" :?>
```

The value *attachmentname* can be anything consisting of lowercase letters, numbers and underscores, and must be unique for each attachment.

Captions and Floating

All attachments have the possibility of being accompanied by a caption. They can also be situated differently within the text they illustrate, depending on their purpose. They can either be **Floated**, which will cause the text to surround them, or free-standing.

Any images that are centered cannot be simultaneously floated.

Finding the Spot

What you'll need to do next is find the spot where the attachment is going to be inserted. This should always be outside a paragraph tag. In other words, it should always be before the or after the . Attaching files within paragraph tags won't break the system or cause any serious problems, but the resulting output won't be valid HTML, and you may have unpredictable results.

Hopefully, you'll recognize the text that you've entered and be able to identify the place where you'd like the attachment to be. Remember that you can move it later without affecting it in any other ways.

If the attachment is floated, the top of it will be aligned with the paragraph that comes after it. (see figure 2.1 on the opposite page).

Defining The Attachments

Once you've successfully added the attachment tag, it's time to choose which image you want to attach. Ensuring that you've saved the text file you added the attachment tags to, click on **Edit File Attachments** in the **Actions Area**.

This screen will show a list of the tags found while scanning the text of the file you are editing. Tags that were incorrectly entered may not show up here, so it's important to make sure you've got the beginning and end of the tag right. Again, they look like this:

`<?sm:` and `:?>`

(It's also easy to miss-spell the word 'attachment' if you are typing quickly).

Each attachment will have an **Edit Button** directly under it. Click it to be taken to the **Define Attachment** screen. This screen displays a form where you'll enter the information as to which file you are going to attach, and how you want it to look.

The information needed is:

- **Attached file**: A dropdown menu of all the files in the repository that are the right type to be attached.
- **Position**: Should the image be aligned to the left, to the right, or centered?
- **Caption**: What, if any, should the caption be for the attachment?
- **Caption Alignment**: How should the caption text be aligned in relation to the image?
- **Float:** Checking this box will cause the text to surround the attachment, rather than forming a complete break. Centered attachments cannot be floated.
- **Show Border**: Checking this box will activate a grey border which can help in some situations to separate the attached image from the text that surrounds it.

Enter values for these fields, and click Save. Your Attachment will now be visible when you preview any pages where the text appears.

File groups

The file repository allows you to organize your files together into smaller, meaningful groups. These groups can be limited to specific file type, placeholder type (see chapter 5), or can accept any file type.

Create a file group

Groups can be created easily, by clicking ‘Create a file group’ in the actions area of the main files repository screen.

Smartest will ask you to name the group and choose (optionally) what types of files the group should be limited to. Click ‘Save’ and a new empty group will be created.

Adding and removing files from groups

Files can be added to and removed from groups from both the ‘File info’ screen and whilst directly editing files. These screens show a list of any groups the file already belongs to, followed by a dropdown menu of any groups that accept files of the appropriate type, but that do not currently contain the file.

Alternatively, if you would like to quickly populate a group with many files this can be done by directly editing the group.

Deleting files

Follow the instructions above to browse to the file, and select **Delete This File** in the **Actions Area**.

4. Items (the Data Manager)

What is it?

The ***Data Manager*** is a data storage and modelling tool that allows for hugely flexible, context-specific information storage. However, while the ***data manager*** system is comparable to a normal, table-based database in terms of basic function, it is not intended for exactly the same usage, nor does it store the data in the same way.

What is it for?

The data manager is for storing structured, two-dimentional information about alike objects. Whether its cars, people, news articles, videos or any other type of objects, the data manager is where you'll store and edit them.

The best way to quickly understand how the Data Manager is useful is to follow the initial workflow that a user may follow.

Creating a data model

The structures that you will give your data are called ***models***. A model can be thought of as a group of similar objects.

For instance, let's say we have created a "Person" model for storing data about people at a company.

We can then add ***properties*** to the model (which are analogous to database columns). For instance, our 'person' model may have properties such as first name, surname, phone number, and e-mail address. If you choose, some properties can be subject to validation, which means that they can be mandatory, or required to conform to a specific format.

We can now add as many ***items*** to the model as we want, adding first name, surname etc for each person. Items are analogous to database table rows, so each item has a corresponding value for every property.

Every item in all models will already have a 'name' property.

You can create as many models as you wish, and each one can have as many properties as they wish. Properties can be added and deleted as easily as the items themselves.

Model properties themselves are **typed**. This allows better input validation, and makes the data stored in the Data Manager much more useful.

The types that the data manager currently uses are:

- Number
- Single-Line Text
- Multi-line Plain Text
- **Boolean** (true/false)
- File
- Foreign Key
- Date
- Date/Time stamp

Foreign keys

The **foreign key** type, as in a real database, allows a property of one model to be semantically linked to another entire model, so that setting a value for that property involves choosing one item from whichever model the property is linked to.

How does it work?

The data manager stores all item and model data without having to make new databases or tables every time you create a new model or update it every time you add a new property or remove an old one.

This makes it highly flexible, and ideal for storing certain types of data, such as articles in an online magazine. However, while the Data Manager is conceptually comparable with a normal database, it is not designed as a substitute for pure table-driven systems.

Technical Notes

If you are familiar with ordinary relational databases, a **model** is analogous to a database table, an **item** is similar to a record, a **property** is analogous to a column. Hence an item has a its own value for each property.

The data manager uses four tables to store all data: one for models, one for properties, one for items, and one for the property values for each item.

An actual PHP class is generated for each model, and is stored in `Library/ObjectModel/`.

Working With Items and Models

Creating Models

Before your create your model, it's always a good idea to plan it out first. What will it

be used for? there may be more than one answer to this question.

To start creating your model, go to the ***Data Manager*** by clicking **Data** in the **Main Menu**. Click on the **Create and edit items or models** option. Choose the **Build a New Model** action under **Model Options** in the **Actions Area**.

You wil be presented with two fields, Model Name and Model Plural Name. Enter in these two fields the words that you wish to use to refer to your models. The upper field will be used whenever you are talking about a particular item, for instance “Person”. The lower field will be used whenever you are referring to the data in the model collectively, for instance “People”.

As you type in the upper field, you’ll notice that the lower, plural field is filled in with a guessed plural version of your singular model name. As this process of guessing simply involves adding an S to the end of the singular, there will be cases - as with Person/People where it guesses incorrectly. You can override this guessing feature and change the plural to anything you like once you’ve entered the singular.

Once you’ve chosen your model’s singular and plural names, click **Next >>** and since a model without any properties is near useless, you will be taken to adding the first property to your model.

Adding and removing properties

As explained above, the properties are the most important part of the model since they give the shape that your data will have to fit inside. This shape or structure of the data is called a schema.

If you are not already on the **Add Property** screen, go there:

- Navigating to your models via the Data Manager Start Screen,
- Selecting the model you want to add a property to and click Edit Model Properties in the Actions Area
- Clicking Add Property in the Actions Area of the following screen.

Enter the name and the type of your property. The name should be simple, since derivatives of it will be used in a variety of different ways. Don’t be too concise, but similarly, names that are too long will be cumbersome when using the Data Manager API. See the Developer documentation on using the Data Manager API.

Under the name and type of your property, you have the opportunity to make the property required. Checking this box will make it impossible to publish any items that leave this property blank.

If you would like to add another property after this one, change the value of the **Continue To** option to “Add another property to model *MODELNAME*”

(MODELNAME being the whatever the name of your model is), an keep repeating this process until your model has the schema that you would like.

Adding items

Once you've arrived at your desired schema, you can start entering data into the system. Remember that if you plan on using files such as images and text, those iles have to be uploaded to the files repository before they can be used as an item property.

Navigate to the models screen:

- Click on **Data** in the **Main Menu**
- Click **Create and edit items or models**.

Once you are on the Models screen, highlight the model you want to add an item to, and click **Add a new member item** in the **Actions Area**.

The Add Item screen lists each property in your model's schema, plus the built-in name property. Enter values in this form and click **Save Changes**. You will be taken to a list of all the items of that model. You can then continue by editing the item you just created, or any others listed on that screen.

Editing items

The editing of items is split up into two screens - the Item Properties screen. Various different parts of the Smartest interface will bring you to the main Edit Item (item properties) screen whenever you need to edit an item, but the quickest route to start editing an item if you are not necessarily on one of the other screens that has it conveniently linked is to follow these steps:

- Click on **Data** in the **Main Menu**
- Click **Create and edit items or models**.
- Select the model that the item you want to edit belongs to, and click **Browse items** in the **Actions Area**.
- Select the item you wish to edit and click **Edit Properties** in the **Actions Area**.

Draft vs. Live

Almost all data in the Data Manager has two definitions - one "live" which will be used every time the information is sent to your site's visitors, and one "draft" which will only ever be used to show you how things will look on the edit preview screens.

Importantly, some information does not make use of this ability to preview, and will be available to build pages as soon as it has been saved.

- The built-in "name" property of all items.
- Any tags that have been linked to the items.
- Meta-page property.

- Search Terms field.

Item properties screen

The item properties screen is where you will make all changes to the actual property data of your items. On this screen, you'll see form elements already displaying any data that you might have entered when the item was created or in prior edits. Use these form elements to change the values for the item, and use the **Save Changes** button at the bottom to make sure your work is saved.

The **Save Changes** button will usually keep you on the **Edit Item** screen unless Smartest judges that you have come from the middle of another very specific task, like previewing an ***Object Meta-Page*** or clearing tasks in your workflow tasklist. This is so that you can continue to save your changes *while* editing, without having to re-open the item. When you have finished editing the item, click on Back to MODELNAME in the lower part of the Actions Area.

Most of the data you edit on this screen has draft status, and will not become available to the visitors of your website until you publish the item. See the section below on Publishing Items for more information on this.

In addition to the properties that make up the schema of your model, and the built-in name property, there are several special properties that give you additional control over your items. Remember that only the properties that have been added by users make use of the draft/live distinction.

Tags screen

This screen is where you can control how the item you are editing is tagged. You can also create new tags from this screen by clicking **Add a New Tag** at the top of the **Actions Area**.

Related content screen

A common and powerful navigational tool on many websites is the semantic attachment of one entity, such as a page, to other, "related" entities - possibly other pages, or records from a database.

Smartest features built in support for this type of semantic mapping, and any item in the data manager can be related to any other, or to any page in the CMS. On this screen, which you can access by clicking the **Related Content tab** that is visible at the top of the **work area** while editing an item, you'll see a list of any items of the same model that are related to the current item, a list of static pages below that, followed by lists of items from other models.

To update these relationships, click **Edit...** beneath the appropriate list, and use the

checkboxes on the resulting screen to indicate which items or pages are related.

Author attribution

The authors screen allows you to attribute one or more users with authorship of the item you are editing. To edit this information, choose the **Authors tab** while editing and item, and check or uncheck the checkboxes next to each user as required. This information can then be accessed using the `<?sm:byline:?>` tag in your templates.

Publishing items

To publish an item is to update all its live properties with whatever data has been saved since the item was last published. This is the only way that live data manager content can be updated.

Validation of your data occurs when you attempt to publish the data. In other words, you can create a new item, stop half-way through and save, and come back to it later, but when the item is published, it must contain valid data.

In the Data Manager, unlike the ***Page Manager***, publishing only means the process of updating live data with draft data, and no cached pages will be refreshed. If you have caching turned on (and this is the default behavior), pages will not be rebuilt until the cache is cleared or expires.

Deleting items

To delete an item, follow these steps:

- Click on **Data** in the **Main Menu**
- Click **Create and edit items or models**.
- Select the model that the item you want to edit belongs to, and click **Browse items** in the **Actions Area**.
- Select the item you wish to edit and click **Edit Properties** in the **Actions Area**.

Data sets

The Sets feature is an important feature of the data manager. It provides the ability for you to create subsets of your items around a particular set of criteria or theme of your choosing. There are two types of sets, dynamic and static.

Dynamic sets

Dynamic sets can be thought of as saved queries. You define specific criteria and Smartest will retrieve any items that match all of those criteria.

Static sets

Static sets are completely free subsets of items that can be included or excluded as you

prefer, and in any order you like. These sets work similarly to folders on your PC.

Building sets with your data

To build data sets, go to the first screen of the Data manager and click **Organize items into sets**. Here you'll see a list of any existing sets that may have been added earlier or by other users. To create a new set, click **Create A New Data Set** in the **Actions Area**. This will bring up the New Set screen, which requires four pieces of information.

- Set label: Largely self-explanatory, this field will be how your set is referred to within the Smartest interface. An all lowercase, letters, number and underscores derivative of what you enter here will be created to be used in templates and other code.
- With items from model: Choose which **model** this set will apply take its items from.
- Set Type: choose whether the set you are creating will be dynamic or static.
- Share this Set?: If the set is dynamic, ticking this checkbox allows the set and its criteria to be used in other sites. Note that the items themselves are site specific and will not be transferred or copied from one site to another.

Technical note

The retrieval of dynamic data sets uses the same data querying engine internally that the Data Query API uses.

Retrieving data from sets

The contents of any set, dynamic or static, can be listed on any page at any time. See the section on displaying data for more information.

5. Pages

The Pages Manager - What is it?

The ***Pages Manager*** is the portion of Smartest that deals with the publishing of arbitrary content. In other words, whatever you like. It is a cornerstone of the Smartest content management system.

Smartest does not come with any pre-made templates or "themes," so it's up to you or your designers, developers and copywriters if you have them, to produce the material that makes up the site.

The pages manager is designed to with two clear goals in mind:

- To help you and your team to create a website that communicated *exactly* what you or your clients want it to, and in exactly the right way.
- Help you to maintain the pages and even create new ones without having to rely continually on designers and developers once they are no longer working on the project.

How does it work?

Site hierarchy

Virtually all websites are *hierarchical* in their structure.

By hierarchical, we mean that the semantic relationships between the pages that make up the website can be understood similarly to a family tree – the difference being that each page has only one parent!

To understand hierarchical site architecture, imagine a relationship where page A is the "parent" of page B. Page B is said to be under page A, and generally is about a more specific theme than page B. Page B may also have a child page, perhaps called page C, that goes into even more specific detail on one aspect of page B.

Only one page has no parent – the home page. This page is said to be the *root* of the site it belongs to. The home page is usually the main entry point regardless of which specific content you'd like to access, and usually doesn't have a specific theme other than that of the entire site.

In Smartest, the site hierarchy can be viewed at any time by clicking **Pages** in the **left-hand toolbar**.

The Site Hierarchy screen is the starting point of almost every page-related task. Whether you are creating, editing or deleting pages, you will need to start here.

Explaining elements

Going back to the **Image Goes Here Paradigm** again (see page 10), the user is asked to choose a file or template to fill a space on the page, known as an element. The file or information that the user inserts when filling that space is called a **definition.**

In other words, the definition of an element is a three way link between a page, the element that is being defined on that page, and the file or data that has been chosen by the user to define that element.

One more time: The *page*, the *space* on that page that needs filling (an **'element'**), and the *file or data* that fills that space (the **'definition'**).

The end result is "a thing on a page".

Elements are invoked by including a special **tag** in your templates.

These tags will be parsed and understood by the Pages Manager when you go to edit your pages, so that you are shown the appropriate user interface controls to define them with.

Pages are then **rendered** by the Smartest Engine following the definitions that you, the user, create for each element on each page. This means that on the page that the public sees, the template tags that say where the elements actually go have been sneakily replaced by the *actual HTML* (or other markup) that causes the image, text or whatever it is to display.

Element types

For building flat pages, there are four main types of element that you can place in the template - placeholders, fields, containers and itemspaces.

Placeholders

Placeholders do exactly as their name suggests. They are a sign to the Smartest Engine that a particular type of element, for example an image, should be rendered at that location. They are probably the most literally "Image Goes Here" of the elements because they are actually used to include images and other easily visible and identifiable aspects of the page.

To use a placeholder, include the following in your template:

```
<?sm:placeholder name="PLACEHOLDERNAME":?>
```

Placeholders are typed, which is to say that they are grouped into different types, just as the files in the file repository are. There are, for example, placeholders for text, placeholders for images, as well as for other types.

However, placeholders are more powerful than a literal one-to-one mapping of file types to placeholder types. You may have a space on a webpage that can be filled by more than one type of file. For instance, image files are separated into JPEG, GIF, and PNG files, but when it comes to using the images on your web pages, there is the option to use a generic image placeholder that can accept any of the three image types.

To use a file in a placeholder, you must have already uploaded it to the **Files Repository**.

For information on how to add your own placeholder types, see the developer documentation.

Fields

Fields are pieces of text up to 255 characters in length that can be edited by the user via either text input boxes or dropdown menus created in the dropdowns editor.

In short, fields are for **metadata** about pages**.** They are designed to be able to carry context- and site-specific instructions and simple data *about the page they are on* to the templates so that they can be displayed or used to build that page correctly.

Although you can use fields for either visible text display (like a very primitive placeholder) or informational control, they were designed with the latter in mind, and placeholders make the former usage unnecessary.

A good working example might be the following: Imagine your website uses color schemes to theme pages, so that pages pertaining to one theme are blue, while pages pertaining to a second are red, and so on. If the "theme" field is referred to in the template, the Pages Manager will recognise the tag and give the user a chance to change its value. Fields are a useful way that the user can easily control and change which theme the page should use.

To both display a field's value and cause it to register itself in the Pages Manager for editing, use the following tag in your template:

```
<?sm:field name="FIELDNAME":?>
```

If you're using the field for page building purposes only, and you'd prefer not to display the actual value of the field, then use this:

```
<?sm:field name="FIELDNAME" display="false":?>
```

If you're using the data from a single field more than once, you can use the `$this` variable to access it without creating an edit button each time it is referred to:

```
<?sm:$this.fields.FIELDNAME:?>
```

Note that if you *only* refer to the field this way, it will not show up in the **Page**

Elements Tree when the page is being edited.

Fields are site specific in that they need to be re-created for each website you host within your Smartest installation.

Containers

Containers are for providing easy control over the layout of your pages. They give you the option of choosing on every page between layout **Container Templates**. Container Templates are "incomplete" pieces of **Dynamic HTML** that support the full capabilities of the Smarty-based Smartest template language. Any **Container Template** can be inserted into any container, although users must be very careful to avoid infinite loops. See the glossary definition of **Page Hierarchy** for more information.

To include a container on your page, use this tag:

```
<?sm:container name="CONTAINERNAME":?>
```

For example, let's say you have the option of using a two or a three column layout on any page. For the sake of simplicity, let's say that all pages use the same master template. In this situation you will create one **Layout Template** for each layout, and then choose which template should be inserted in place of the tag above.

Containers are conceptually similar to placeholders, except that they insert templates - which can have anything in them including further containers - rather than actual items like images or text.

Template tags

If you always wish to make use of the same template, without offering the user the choice of which template to use, then you can use a template tag. Template tags won't show up in the page elements tree, and always will be rendered the same way.

```
<?sm:template name="images_grid":?>
```

or

```
<?sm:template name="images_grid.tpl":?>
```

Will both render Presentation/Layouts/images_grid.tpl.

Creating New Page Elements

To create a new Placeholder, Container or Field is easy. Include the tag that refers to it (like those shown on previous pages) in your templates. It will be detected, and since it has a `name=""` parameter that isn't recognized, it will show up red instead of black, and you'll be given options to add it from the Page Elements tree.

Quick Recap

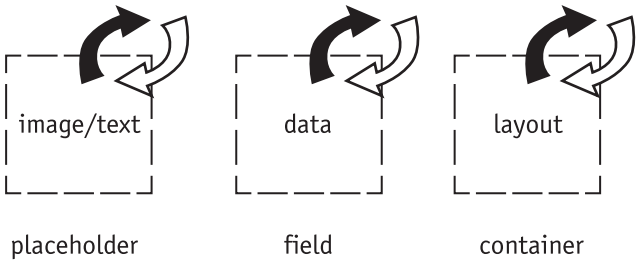


figure 5.1 Page element functions

Understanding contexts

The rendering of Smartest markup is not universally treated the same. In Chapter 2 on files, you will have seen how **Attachment** tags, similar to **Placeholders** or **Containers** can be included in the text and are treated in a similar way.

It would be easy, though mistaken, to assume that **Placeholder** tags can be used in text files alongside **Attachments**, and vice versa. The difference is a matter of **context**.

When a text file is being rendered, that process is not actually specific to the page where it is occurring. The text fragment uses the same attachments wherever it is included, and changing it on one page will change it on any other pages where it appears. Information about the page where it is happening is still available, but the process is occurring in **File Context**.

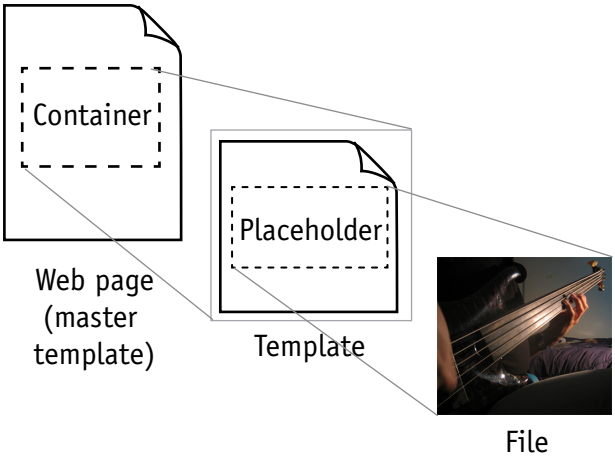
Similarly, Attachments can only be made between yext files and images. Attempts at using attachments within your layout templates will not be recognized, and will produce an error in the preview screen to remind that it is not possible to use attachments in **Page Context**.

Websites and their structure

Page Elements Tree

If you've read the above section on containers, you'll know that it's possible to use any of the templating instructions described in the previous sections, including containers, within your Container Templates. In other words, this makes it possible to have a container inside a container inside a container and so on infinitely.

This recursive paradigm allows the elements on a page to be arranged in a hierarchical



manner. For instance, if you have a container within a single page that allows you to choose a column layout, and you choose a three column layout, you can then have three more containers - one in each column - to help you control how each of those columns is laid out.

The draft vs. live distinction

Almost everything that you can change whilst editing pages in the Pages Manager has two definitions - one “live” which will be used every time a page is rendered and sent to your site’s visitors, and one “draft” which will only ever be used to show you how things will look on the page preview screen.

It is important to note that **Templates** do not share this quality. Changes to any templates will be visible as soon as the pages that use them are next built.

Publishing pages

To publish a page is to update all its live placeholder, field and container definitions with whatever data has been saved since the last time the page was published. Live definitions can only ever be updated by publishing a page, and never edited directly.

You are allowed to publish undefined page elements, though before doing so you will see a warning, listing all undefined elements.

When you publish a page, the Pages Manager will destroy any copies of that page that it finds in the cache.

Page presets

If there are certain definitions of Placeholders, Fields, and Containers that are going to be common to a large number of pages, you can create a preset from any page,

which will save all of the draft definitions on that page into a group, which can then be loaded and applied to any new page. Changing the page that the preset was created from will not affect any pages created with that preset before the change was made.

Page types

Flat pages

Flat pages are so-called because they are "just" web pages. In other words, they are simply a space for you to combine your templates and other resources such as text and images with Smartest's flat-page content management elements like placeholders and containers. Flat Pages will probablby form the vast majority of your website.

They are based around the simple idea that every page may look different and say something different. The way you design one page has no bearing or implications for how other pages of the same site will look (although if they share a template, editing that template will change both pages).

Object meta pages

Object meta pages are used to represent items (records) from the ***data manager***. They are designed so that one "page" in the site hierarchy tree can represent any and all items that belong to the model chosen for that page. For instance, if there is a model called Car, we can build a page that can be used to describe any of the cars that are stored in the data manager. When site visitors view information about a specific car, they will load that same page, regardless of which car they are viewing. For Object Meta-Pages, the URL contains variables that are filled in differently so that Smartest knows which car is being requested.

Special page types

Special pages are just like normal pages, except that instead of being about one particular topic, as in a normal page, they are used only in specific situations, like searches, viewing of tagged content, and errors. There are three of these 'special' types of page.

Search Page

The search page is loaded every time a site visitor makes use of Smartest's built-in search functionality. The contents of the page will list any results returned in the search, but all other parts of the page will behave like a normal page, and you can use placeholders to give the search page its own identity.

Tag page

Similarly to the Search Page, the tag page is loaded every time tagged content is

viewed. This page relies on having the right templates inserted in order to display results. See Chapter 6 for more help with tagging.

Error (404) page.

This page is loaded every time an unrecognised URL is requested.

Using elements on special pages

All three types of special page still support the same **page elements** and other templating instructions as other pages, though their definitions will always be the same regardless of the specifics of the request that the user has made. When you first Set up Smartest, you will see these three pages listed in the site hierarchy as children of the home page.

Working with pages

Workflow

Smartest uses a system of workflow to help content creators and editors keep their tasks organized.

When a user opens a page to begin editing it, this page is then **locked** to prevent other users from opening it. The page will remain locked until the user **releases** the page.

Once the page has been released, the page will be ready for approval if the approval feature is turned on, otherwise the page will be ready to publish. In the case that the user making the changes does not require approval, he or she can publish the page without needing to release it first, though publishing the page will also release it.

To release all the pages that are locked by your user account, click **Release All Pages** in the **actions area** on the **Site Hierarchy** page.

Page data

All pages are stored in the same way, and have the same basic set of parameters that are used to identify them and control how they behave. Most of these parameters do not affect how the page actually looks.

See overleaf for a list of these parameters and what they do.

figure 5.3 - Page parameters

Name	Purpose
Title	This will be used every time the page is linked to or displayed in breadcrumbs. It is the primary name of the page. It can be accessed in the templates as `<?sm:$this.page.title:?>` (and as `<?sm:$this.page.title_formatted:?>` with the site name appended)
Name	The ***name*** of the page is a derivative of the title that is created when the page is created. It is all lower-case and contains no spaces or special characters. it is used in dynamic linking to pages: e.g. `<?sm:link to="page:my-page" with="Read More":?>`
URL	This is the address of the page at which it is accessed over the world wide web. Every page in Smartest can have unlimited URLs, provided that they are unique. The primary URL is entered when the page is created, but can be modified at any time.
Parent	All pages have a parent except the home page. The 'parent' of a page is the information that governs where in the site's page hierarchy the page is stored. See **Site Hierarchy** on page 21 for more about site hierarchy.
Icon	Often it's necessary, for instance when linking to a page from another page, to summarise the linked page with a small graphic representation. The page's Icon field allows you to choose one image for each page from the file repository to use for this purpose.
Search Terms	Enter single words here which will be used in finding pages that match user searches. Note that the text entered in this field is never visible to outside users - it only controls the behavior of the search.
Description	Pages in Smartest don't have any implicit link to their body text, since all body text is stored as text files in the files repository and displayed using placeholders. The description field can be used when pages are being listed automatically, for instance in search results or RSS feeds derived from tags, or when you use the built-in navigation data from the $this varable to list child, sibling or parent pages of the current page.

Meta Keywords	This field is for storing keywords if you are using a `<meta name="keywords" />` tag on your master template.
Meta Description	This field is for storing a description other than the primary page description if you are using a `<meta name="description" />` tag on your master template.
Cache As HTML	This parameter switches the page caching on or off for specific pages. By default, caching is turned on, and it's not recommended to turn it off unless you have a very specific need for a page to always change. Turning this off just causes your web server to have to do a little bit more work.
Cache Frequency	This parameter controls how long the cached copy of the page should be used before it is regenerated, and varies from permanent (i.e. cached page doesn't expire until deleted), to regenerating every second (provided that the page is requested - pages are only generated when they are requested and no valid cached copy exists).

Creating new pages

To create a page, you first need to choose where it will sit in the site hierarchy. As explained in the **Site Hierarchy** section (page 21), every page except the homepage must be the child of another page. To start creating your page highlight a page in the **Site Hierarchy**, and choose **Add a New Page** to add your new page as the child of the page you have selected. You will then be taken through the three steps for creating your page.

Step 1. Choose which type of page you are going to make (see previous page or glossary for explanation of page types).

Step 2. Input the page's basic identifying information, such as title, url, master template, and ***Page Preset*** (if any).

Step 3. Confirm your choices and decide where to be taken next. Think about whether you are going to continue editing this page (and the chances are you will), and if so which of the screens listed is right for the kinds of changes you are going to make.

Editing pages

To edit any page, highlight it in the site hierarchy and click **Edit This Page** in the **Actions Area**.

If no other user has locked the page, the page edit screen will appear, and at that point the page is locked so that other users will be unable to start editing it. This is a normal

part of Smartest's workflow, and when you are finished, the page will be flagged for approval.

Editing pages is divided up into four distinct screens: **Overview**, **Elements Tree**, **Tags**, and **Preview**.

Page properties screen

This screen is for changing the basic meta-information of a page, such as its title, url, position within the site hierarchy, and caching frequency. A full description of all these parameters can be found above in *Figure 5.2 - Page parameters*.

Page elements tree screen

This screen shows the page's internal hierarchy of elements. All elements can be defined and re-defined here, but note that ***containers*** can only be modified from this screen, and not in the **preview screen** like ***fields*** and ***placeholders***. To redefine any element, simply highlight it, and click the corresponding option that appears in the **actions area**.

Preview screen

This screen allows you to see how the changes you have made will look before a page is published. What you see in the preview screen is built entirely using the draft definitions of your page elements, and completely bypasses the page cache.

You can also edit placeholders and fields directly from the preview window, using the icons that are automatically inserted beside your placeholders and fields. Because these icons are inserted automatically, they may slightly alter the appearance or spacing of your page. These small changes will not be present when your page is published.

Tags screen

This screen allows you to control how your page is tagged. You may attach the page to as none or all of the tags, or any combination. You can also add any new tags from this screen. Note that tags are not site specific, so tags created on one site will be available to all others within the same Smartest install, although when you request all items and pages tagged with a particular word or phrase, only the documents from the current site will be retrieved. For more information, see Chapter 5 - Tagging and Searching.

Related Content Screen

A common and powerful navigational tool on many websites is the semantic attachment of one entity, such as a page, to other, "related" entities - possibly other pages, or records from a database.

Smartest features built in support for this type of semantic mapping, and any page in the website manager can be related to any other page, or to any item in the ***Data Manager***.

On this screen, which you can access by clicking the **Related Content tab** while editing a page, you'll see a list of any pages that are related to the current page, followed by lists of related items from the **models** in the Data Manager. To update these relationships, click **Edit...** beneath the appropriate list, and use the checkboxes on the resulting screen to indicate which items or pages are related.

Author Attribution

The authors screen allows you to attribute one or more users with authorship of the item you are editing. To edit this information, choose the **Authors tab** while editing and item, and check or uncheck the checkboxes next to each user as required. This information can then be accessed using the `<?sm:byline:?>` tag in your templates.

Deleting Pages

Deleting a page is a very easy process. From the **Site Hierarchy** page, select the page you would like to delete, and choose the **Delete Page** option in the **Actions Area**.

6. Tagging & Searching

Studies show that most visitors to your site will be looking for something specific, and may leave if the home page doesn't display an obvious way to find that special something. In addition to clear, intuitive navigation and good optimization for **Search Engines** like Google, many sites employ a site-specific search to help visitors find exactly the page they're looking for on your site.

Searching

To make your search effective in Smartest, you need to make sure to use the Search Terms field on Pages and Items. Think about what aspects of your site might be searched for, and what visitors might enter in order to find them.

They may miss-spell words, or use misnomers that are commonly accepted despite being different to the words you've chosen for your web content. Since the search terms on Pages and Items are never actually seen by site visitors, you can enter any terms you think might be relevant.

Tagging

In the mid 2000s, a movement in web development called **Web 2.0** began to look for alternatives to a normal text search. One of the more popular innovations to come out of this was a system called "tagging".

With tagging, single words can be attached to entities which are relevant to that word. Then, the visitor to the site can request to see all content that is attached to a given word.

Smartest features built-in tagging not only for web pages, but for individual items in the ***Data Manager*** (provided that they can be displayed on an **Object Meta-Page**).

Tagging pages

To tag a page, open it up, and click on the **Tags tab**. You'll see any tags you've already entered listed on that screen, with a checkbox next to each one. Any tags that are attached to the page will already be checked, and you can attach or un-attach tags by checking and un-checking them respectively and clicking **Save**.

To add a new tag, click Add a New Tag in the Actions Area, and enter the prase you'd like to use as a tag in the form that follows. Click Save, and you''ll now see that tag in the list of tags and checkboxes, ready to use.

Remember that if the web page you want to tag is actually representation of an item

from the Data Manager, via an Object Meta-Page, then you'll need to tag the item, rather than the meta-page.

Tagging Items

Navigate to the item, and open it up for editing. You'll see two tabs at the top of the edit screen. Click the **Tags tab**. You'll see any tags you've already entered listed on that screen, with a checkbox next to each one. Any tags that are attached to the page will already be checked, and you can attach or un-attach tags by checking and un-checking them respectively and clicking **Save**.

How Tags Work

Each tag has both a label that you give it, which can be both upper and lower case and contain spaces, and a short name containing only lowercase letters and hyphens, which is derived from the label you enter.

Tag Feeds

Every tag is also available as an RSS feed. All published items that are attached to a given tag will be retrieved in the order they were last published.

The URL of these feeds is very simple:

```
http://[yoursite]/tag/[tag short name]/feed
```

Of course, replace `[yoursite]` with the actual base URL of your site, and `[tag short name]` with the short name of the tag you wish to retrieve (See "*how tags work*" above).

Appendices

Appendix 1: Computer Science 101

Variables

A variable is an extremely abstract, but deceptively simple concept. If you're having trouble getting your head around it, maybe you're making it too complex. Think of a variable as a *symbol*, which like any symbol has a *name* and a *value*.

In computer sceince, the name of a variable is always the same, but its value can be modified at any time.

In Smartest, and in PHP (the language used to create Smartest), variables are identifiable because they begin with a dollar sign, like so:

$myvariable

Arrays

An array is a variable that instead of having just one value, contains a list of values. These can include other arrays.

Conditionals

A conditional statement is one that checks if a specified condition is being met. For example, a conditional can be used to check if a variable has a certain value (ie "if the value of `$myvariable` is equal to 2"), or if its value meets a certain condition (ie "if the value of `$myvariable` is equal to 2 *or more*").

Looping

Looping is when you carry out an action a set number of times. Either you can specify the number (ie "do [said action] 5 times"), or you can autmatically set the number based on a list (**array**) of values (ie "do [said action] for every value stored in `$myvariable`").

Functions

When programmers have to carry out the same or a similar process many times, it's optimal to write that procedure just once, abstracting out the bits that are different each time (using variables). This type of small but often used procedure is called a function.

Using a function means that you only have to improve of fix that operation in one place, and the benefit is immediately available everywhere.

Glossary

Boolean

A boolean value is a value that can only be true or false. These values are sometimes represented as 1 and 0 respectively.

Cache

A cache is a place where software systems store information that they are likely to need often. Rather than reload or re-calculate that information every time, it is more efficient for the system to calculate it the first time only, and store it in a way that makes it quick and easy to access in the future.

Container

A container is a space into which a template that is used for layout can be inserted into a page.

Content Management System

A content management system is any system (of which Smatest is an example) that allows information, pictures and other content to be displayed, manipulated and updated without requiring a human being to directly edit HTML, code, or any other computer parsed instructions.

Date/Time stamp

A value that records, to the second, an exact moment in time.

DNS (Domain Name System)

The Domain Name System, or DNS, is the worldwide system that translates Fully Qualified Domains into IP addrdesses

Draft Status

When the definition of a placeholder, container or field, or the value of an item property, has draft status, it means it can only be seen by logged in site editors and administrators who are working on the site.

Dynamic HTML

Dynamic HTML is HTML that contains elements that are replaced when it is processed by the server.

Field

A unit of data that can be used to pass intructions to the indiviual pages, which can be either displayed or used to control the page's appearance in other ways.

Flat Page

A page that does not require you to follow any data structure. Flat pages contain only the information you want them to contain.

File Repository

The section of Smartest where files such as images, text, adobe flash, quicktime movies, and other types are stored and organized. These centralized files can then be summoned in a variety of different contexts.

Fully Qualified Domain Name

A fully qualified domain name, is a name for a **server** that can be referred to from anywhere and resolved by **DNS**.

Function

A portion of code within a larger program, which performs a specific task and can usually be called on from most other parts of the program when that task is needed. The point of functions is to reduce duplication of functionality in code: it is better to wrote good code once and re-use it, than to re-write the same functionality everywhere it is needed. (based on: `http://en.wikipedia.org/wiki/Function_(programming)`)

Host name

See **Fully Qualified Domain Name**.

HTML

Stands for Hypertext Markup Language. HTML Is the standard for creating and displaying content on the world wide web. It is a system for description of web content. For more information, see http://en.wikipedia.org/wiki/HTML

Item

Item is a generic name for any of the *instances* of a model (*see below for a description of "model"*). In other words, where "Car" is the model, "Dark Red 1997 Honda Civic automatic transmission with Wisconsin plates" might be an item - an instance of that model.

Live Status

Any definition or item property value with live status will be used next time a page containing that information is built by the CMS. This is in contrast to draft status which is never used to build live pages.

Markup

Markup is a type of instruction given to computers by people that refers to layout or presentation. The best known example of a markup is HTML (*see above*).

Model

A Model is a set structure of properties in which you can store data. For instance, a "Car" model would be likely to have properties to describe things like the make, model, year, color, transmission type, etc, of the cars you had on your website, and you would be able to store information about lots of different cars in it by entering information differently for each of the properties for each car.

MySQL

MySQL is a highly popular and trusted piece of software for creating, storing and querying (relational) databases.

Open Source

"Open source" can be used to mean a wide variety of different things. In its loosest definition, it means software that allows end users to see how it was coded. Officially, it refers to software that not only allows this, but is licensed in such a way as to allow end users to modify the software and/or re-distribute it, with or without modification. Open source software is usually free because, while open-source licenses do not prohibit the selling of such software, its inherent availability and possibilities for redistribution effectively drive the price down to 0.

Object Meta Page

This is a page that is mapped onto a model, so that by changing the url, you can make the object meta page about any of the items of that model.

Placeholder

A space on a Smartest web page into which files in the repository can be inserted. See Chapter 4 for more information. See chapter 2 for more information on the file repository.

Page

When you enter a specific address in your web browser and hit return the resulting content (assuming the address you entered was correct) is commonly called a web-

page or simply page. When dealing with most pages in Smartest this definition holds true. However, when using an ***Object Meta-Page*** (see above), many similar but differing pages in this traditional sense can come from what is represented as a single page in the Smartest **site hierarchy**.

Page Hierarchy

When using **Nested Containers** on a page (see chapter 4 and figure 4.1), the elements on that page can be thought of as a tree structure. If for instance, you have a container - lets call it container 1 - in a template (let's call it template 1), and the template inserted into container 1 (which we'll call template 2) also contains page elements, then those elements are one step below any elements that appear on template 1.

PHP

PHP is a scripting language, originally created by Rasmus Lerdorf in 1994, for programming web servers. Smartest is written in PHP version 5.

phpMyAdmin

phpMyAdmin is an **open source** software utility written in **PHP** that is often installed on web **servers** to aid in the administration or **MySQL** databases.

Preset

In Smartest, a preset is a group of pre-defined containers, placeholders, item spaces and lists that can be used to quickly create a new page that has those definitions.

Property (Data Manager)

A property is a specific way of describing something stored in the data manager. For instance, if the **Model** is "Person", then one property, which describes something specific, might be that person's gender. It doesn't say what their name is, or their age - just their gender. Multiple properties are usually used in a single **Model.**

Recursion

A recursive process is one made up of a particular procedure causing itself to be re-run, so that it must continue running until it cannot go any further.

See: http://en.wikipedia.org/wiki/Recursion

Rich Text

Text that contains formatting information, such as bold and italic, as well as actual words and sentences. This type of text usually has to be **parsed**.

Server

The word ‘server’ can be one of two things, depending on context.

(1) Sometimes a server is a physical computer, on which resources are stored that can be accessed from outside.

(2) A server can also be a program that is always running, usually on a physical server (see first meaning, above), and that responds to requests for it to act.

Site Administrator (Role)

The Site Administrator is somebody who is reponsible for making sure that Smartest is installed and set up correctly, and that it is up to date.

Site Hierarchy

Websites are collections of web pages, organized into hierarchical structures.

These structures are based around a home page with more sematically specific pages leading off from there, yet more specific pages lead off from them, and so on. The overall map of the site is thought to resemble a tree for this reason.

the Smartest root directory

This is the folder/directory that contains all other files and folders that make up Smartest, save for the database files. This folder should not be web accessible. When you first download and uncompress Smartest, the resulting folder is the Smartest root directory.

SSH, SSH Access

SSH is a way of controlling servers remotely using text commands from anywhere in the world in a way that means the information transferred to and from the server is encrypted and cannot be intercepted.

“SSH Access” is the availability of this method, which is the de-facto standard for server administrators who work with remote servers (many of the wold’s web servers are stored in different buildings, cities, or even countries from the people who use them).

SVN

SVN, or SubVersioN, is a source-code version control software package that is used to aid the development and distribution of Smartest. It is managed and maintained by a company called CollabNet.

Template

A template is a piece of markup that contains instructions that can be created and understood by people and processed by a web server. Templates usually specify some aspects of the page, for example such as basic layout or style, without actually saying *what specifically* goes on the page. A template might say for example, how all articles shoould look, without making reference to any specific article(s). This is useful because although articles may be topical, the visual appearance of the articles may be more lasting. This is an example of separating form from content.

Web Accessible

If a folder of file is web accessible, then it can be viewed/downloaded using a standard web browser by anybody connected to the internet, if its address is known.

ZIP Archive

In computing, an "archive" is often an entire folder containing other files and folders that has been reduced, reversibly, into a single file that is usually far smaller in size than the combined total of all the files that were in the folder from which it was derived. This is useful for quickly and accurately distributing files that belong together.

According to Wikipedia, the term "ZIP" was first used by archiving software first released by Phil Katz in 1989, but Katz has stated that he would like to allow the term "ZIP" to apply to any file archive type. (http://tinyurl.com/zipinfo).


VSC BOOKS