Maintenance Document

Contents

File Structure

This section will explain the File Structure for the site to show the most important files, what they do and which files they interact with.

index.php

The main index.php in the root folder.

This file begins by including config.php to gain access to specific functions defined in config, such as getDirectory which returns the path of the page, e.g. the path of Breaking News would be /news/breaking_news. The function getSelectedPage is also defined in config and used by index, getSelectedPage has one argument, the directory of the current page, and returns the id from the database of the current directory, if it exists.

Index.php then includes header.php to display the header of the site.

It then uses getSelectedPage to get the id of the current page.

If the id returned is greater than 0 then the page exists in the database.

> Check to see if an actual directory exists for the path, if it does it is a Static page and so displays the Static page by including the index.php file in the directory for the current page.
>
> If an actual directory does not exist then it is a Dynamic Page and so we include breadcrumbs.php to generate the breadcrumbs for the dynamic page, include menu.php to generate the menu with the current page selected, and include content.php to gain access to functions to displayPageTitle, and displayDynamicContent.

If the id returned was less than 0 then the page does not exist in the database and so we check to see if it is an admin page or a partners_zone page and if it is then we include the index.php for those respective sections.

If the id returned was less than 0 and it was not an admin page or partners_zone page then we include pagenotfound.php to display to the user that the page does not exist.

We finally include footer.php to display the footer of the site.

header.php

header.php in the root directory.

This file produces the html for the header part of the site. This includes meta tags, title, links to style sheets, links to javascript files and the site logo.

breadcrumbs.php

breadcrumbs.php in the root directory.

This file generates the breadcrumbs with URLS for the selected Dyanmic page.

menu.php

menu.php in the root directory.

This file generates the menu based upon the pages in the database and gives each menu item specific CSS classes and id's. The current page is also marked as "selected" with a CSS id, so that the CSS can style the selected menu item with it's sub menus dropped down.

The CSS files responsible for styling the menu are located in /css/ and the important files are flyout.css, flyout_ie.css (used for ie6 fixes), colour_scheme.php and colour_scheme_ie.php (used for ie6 fixes). flyout.php generates the generic CSS to deal with all the menu items, colour_scheme.php generates the CSS for each specific menu item to display their colours.

content.php

content.php in the root directory

This file is used to display the content for Dynamic Pages by retrieving the content from the database, using two functions displayPageTitle and displayDynamicContent.

displayPageTitle takes one argument, the id of the current page, and displays the title of the current page that is stored in the database.

displayDynamicContent takes one argument, the id of the current page, and displays the content of the current page that is stored in the database.

footer.php

footer.php in the root directory.

This file produces the html for the footer part of the site, this includes the logos for the project partners.

Database Structure

colour_schemes

This table holds the data for all the colour schemes. Each Colour Scheme has a `name` and a `colour`.

events

This table holds the data for all the events. Each Event has an automatically generated `id`, a `title` for the event, `content` representing the data about the event, `author` holding the name of the person who added the event and `date` giving the date of the event.

files

This table holds data for all of the files/folders, and distinguishes between whether they are viewable by partners and whether they are inner files. Each File has an automatically generated `id`, the `name` of the file, the `parent` for the file so we know which other file it is a sub folder of, `path1` storing the path to the folder/file, `author` representing who submitted the file, `visibile` which is either 0 or 1 to denote whether the file/folder is viewable by partners or not and `description representing a description of the file.

images

This table holds data for all the images belonging to the Research pages and the rotating image on the Home page. Each Image has an automatically generated `id`, the `name` of the file, the `category` the image belongs to, a `description` of the image and the `date` that it was added.

news

This table holds the data for all the news items. Each News item has an automatically generated `id`, a `headline` representing the news headline, a `content` representing the body of the news item, an `author` representing who wrote the News item and a `date` representing the date of the News item.

newsletters

This table holds data for all the newsletters that have been sent out. Each Newsletter has an automatically generated `id`, a `subject` representing the Subject of the newsletter email, a `body` attribute representing the content that was sent for the newsletter email, a `date` representing when the newsletter was sent and a `file` representing what file, if any, was attached.

pages

This table holds the data for all of the pages of the site. Each Page has an automatically generated `id`, a `menu_name` field representing the name of this page to be displayed in the menu, a `page_name` field representing the name of the page to be displayed as a title on the page. A Page also has a `content` field which holds the html content for the page, a `sort` field which holds the sort number for this page which determines what position it is displayed in the menu and a `parent` field which holds the id of the parent this page belongs to if it is a sub menu, it is 0 if it is a root page. Each Page also has an `editable` field which is 1 if the page can be edited, 0 if not, a `deleteable` field which is 1 if the page can be deleted and 0 if not, a `moveable` field which is 1 if the page can be moved in the menu and 0 if not. Pages also have a `url_name` which is the name of the page that is typed into the URL bar, a `colour_scheme` field which represents the colour scheme for the page and a `menu_display` field which is set to 1 if the page should appear in the menu and 0 if it should not.

subscribers

This table holds data for all the subscribers of the newsletter. Each subscriber has an automatically generated `id`, a `first_name` field representing the first name of the subscriber, a `second_name` field representing the second name of the subscriber, an `email` field representing the email address for the subscriber, this is unique for each subscriber, and a `category` field representing the Subscriber Category that this subscriber belongs to.

subscriber_categories

This table holds data for all the Subscriber Categories that Subscribers can belong to. Each Subscriber Category has a `name` representing the name of the Subscriber Category.

users

This table holds the data for all the users of the system. Each User has a `name` representing the username for the user, a `password` representing an MD5 encrypted password for the user, a `manager` field which is either Y or N, if it is Y then the user is a manager if it is an N the user is a partner and `email` representing the email address of the user.

Adding a Static Page

This section will explain the process of adding a new Static Page such as /news/.

Create a Sub Directory

In the root directory create a folder with the name that you want for the url, such as “example”. This will mean that to get to the page you would type in http://www.examplesite.com/example.

Add a .htaccess

In the newly created directory we need to create a .htaccess, we will copy an existing one from one of the other Static Pages such as /news/. This .htaccess tells the server that whenever anyone navigates to http://www.examplesite.com/example that the server should look at the index.php in the root directory to decide what is displayed, explained previously was that the index.php in the root directory would check to see if the current page was a static page and if it was to include the index.php for the current page. So the next step is to create and index.php for the current page.

Add an index.php

In the newly created directory we need to create an index.php. This will be the file that is included to display the breadcrumbs, menu, page title and content for the static page as well as the content which is up to you to define later on.

Display Breadcrumbs in your index.php

In your new index.php you will need to display the breadcrumbs for this Static Page. Take a look at an example such as the /news/ Static Page to get an example:

```
<div id="breadcrumbs">
	<p><a href="<?php echo(getRoot()); ?>">Home</a> > <a href="<?php echo(getRoot()); ?>news">News &amp; Events</a></p>
</div>
```

So for our new example Static Page it would look like this:

```
<div id="breadcrumbs">
	<p><a href="<?php echo(getRoot()); ?>">Home</a> > <a href="<?php echo(getRoot()); ?>example">Example</a></p>
</div>
```

Display Menu in your index.php

To display the menu we need to just include the menu.php in the root directory, this will do all the work for us. To include the menu you put this into your index.php below your breadcrumbs:

```
<?php displayMenu(); ?>
```

Display Page Title in your index.php

Below the include of the menu in your index.php you will need to place some html display the title for the page. To display the title you use this html:

```
<h3>Example</h3>
```

Display Content in your index.php

Below the html for the page title you can display the content for the page and perform any php that you need to.

Remember when writing any php to make sure that is contained within a php start tag: <?php and an end tag: ?>.

Add a personal_style.css

In the sub directory with your index.php you can create a personal_style.css file which will automatically be included by header.php. This will allow you to define your own custom css for your new Static Page.

Add new Static Page to the Database

For your new page to be accessible and for it to be displayed in the menu we need to add it to the database. To add it to the database we need to add it to the \`pages\` table. To do this I recommend using phpMyAdmin which can be accessed through cPanel.

We first select the table we want to edit, this is called \`pages\`.

When we are shown the structure of \`pages\` we need to select SQL from the choices at the top.

On the SQL page we need to input this query and click go:

```
SELECT MAX(`sort`) FROM `pages` WHERE `parent`=0;
```

This query will give us the current highest sort value of all the pages with a parent of 0. If we are adding a Static Page that has a different parent than the root then we need to find the id of the parent and put that into the query above instead of the 0.

Select Insert from the choices at the top of the page.

On the Insert Row page we now need to fill in some details:

Id: leave the Value field blank

menu_name: in the Value field put in the name that will be displayed in the menu for this page.

page_name: in the Value field put in the title for this page.

content: leave the Value field blank.

sort: Add 1 to the current highest sort value we found before and put this into the Value field. (note: you can add a different sort value so that it appears in a different place in the menu but you must make sure you change all other page's sort values that are the same or higher to your new page and with the same parent so that the sort values are increased by 1).

parent: Put the id of the parent for this static page into the Value field.

editable: This sets whether the page can be edited, because it is a static page we will put 0 in the Value field so that the page can not be edited.

deleteable: This sets whether the page can be deleted, because it is a static page we will put 0 in the Value field so that the page can not be deleted.

moveable: This sets whether the page can be moved around in the menu structure, because it is a static page we will put a 0 in the Value field so that it does not get moved about.

url_name: In the Value field we need to put the name we used for the sub directory in the first step.

colour_scheme: In the value field we need to put the name of a Colour Scheme to use for the page, an example is "HomeScheme".

menu_display: This sets whether the page is displayed in the menu, set this Value to 0 if you do not want the page to show up on the menu, set it to 1 if you do.