Assets

Overview

Frequently, components will make use of their styles, images, and scripts to further enhance the interface and user experience. There are a number of helpers to make adding CSS and Javascript to the document a quick and easy process.

Directory Structure & Files

Assets are stored in the same directory as the entry point, views, and controllers for each client type of a component. This means, for example, the administrative side and front-end of a component may make use of completely different assets.

While there are no hard rules on the placement and organization of the files, it is highly recommended to follow the structure detailed below as it helps keep both small and large projects clean, organized, and allows for several helper methods (detailed in the "Helpers" section) to function, eliminating the tedious need for path building and file existence checking before attaching to the document.

All assets are stored within an assets folder, which is further sub-divided by asset type. The most common types being js (javascript), css (cascading stylesheets), and img (images) but may also contain any other asset such as fonts, less, and so on.

```
/app
.. /components
.. .. /{ComponentName}
.. .. .. /{ClientName}
.. .. .. .. /assets
.. .. .. .. .. /css
.. .. .. .. .. /img
.. .. .. .. .. /js
```

Helpers

The HubzeroComponentSiteController and HubzeroComponentView classes bring with them some useful methods for pushing StyleSheets and JavaScript assets to the document and building paths to images. These methods can be called from within a controller or a component view.

Cascading Stylesheets

The css() method provides a quick and convenient way to attach stylesheets. It accepts two arguments:

1. The name of the stylesheet to be pushed to the document (file extension is optional). If no name is provided, the name of the component (without the com_ prefix) will be used. For instance, if called within a view of the members component com_members, the system will look for a stylesheet named members.css.
2. The name of the extension to look for the stylesheet. This accepts either module, component or plugin name and will follow the same naming conventions used for extension directories (e.g. "com_tags", "mod_login", etc). Passing an extension name of "system" will retrieve assets from the core system assets (/core/assets).

For the defined stylesheet to be found, the assets **must** be organized as described in the "Directory Structure & Files" section.

Method chaining is also allowed.

```
<?php
// Push a stylesheet to the document
$this->css()
      ->css('another');
?>
... view HTML ...
```

Javascript

Similarly, a js() method is available for pushing javascript assets to the document. The arguments accepted are the same as the css() method described above.

```
<?php
// Push some javascript to the document
$this->js()
      ->js('another');
?>
... view HTML ...
```

Images

Finally, a img() method is available for building paths to images within the component's assets directory. Unlike the css() and js() methods, this helper does not add anything to the global document object and, instead, simply returns an absolute file path.

ASSETS

Given the following directory structure:

```
/app
.. /components
.. .. /{ComponentName}
.. .. .. /{ClientName}
.. .. .. .. /assets
.. .. .. .. .. /img
.. .. .. .. .. .. picture.png
```

From a component view:

```
<!-- Generate the path to the image -->
<img src="<?php echo $this->img('picture.png'); ?>" alt="My picture" /
>
```