Helpers

Overview

Unlike components, which potentially can have multiple controllers, modules do not require a controller class. As such, the module directory structure doesn't include a /controllers subdirectory or controller.php. Instead, the setting of parameters, inclusion of any necessary files, and the instantiation of the module's view are done within the helper.php file.

The helper.php file contains that helper class that is used to retrieve the data to be displayed in the module output. Most modules will have at least one helper but it is possible to have a module with more or none.

Directory Structure & Files

The directory structure used for MVC oriented modules includes the helper.php file in the top directory for that module. While there is no rule stating that we must name our helper class as we have, but it is helpful to do this so that it is easily identifiable and locateable.

```
/app
.. /modules
.. .. /mod_{ModuleName}
.. .. .. helper.php
```

Implementation

In our mod_helloworld example, the helper class will have one method: display(). This method will output the contents of the module.

Here is the code for the mod_helloworld helper.php file:

```
<?php
namespace ModulesHelloWorld;

use HubzeroModuleModule;

class Helper extends Module
{
    public function display()
    {
        echo 'Hello, World!';
    }
```

```
}
```

More advanced modules might include multiple database requests or other functionality in the helper class method, passing data to a view and rendering the view.

```
<?php
namespace ModulesHelloWorld;

use HubzeroModuleModule;

class Helper extends Module
{
    public function display()
    {
        // Retrieve rows from the database
        $this->rows = $this->getItems();

        // Render the view
        require $this->getLayoutPath();
    }

    public function getItems()
    {
        $db = App::get('db');
        $db->setQuery(" ... ");
        return $db->loadObjectList();
    }
}
```