Entry Point

Overview

Joomla! is always accessed through a single point of entry: index.php for the Site Application or administrator/index.php for the Administrator Application. The application will then load the required component, based on the value of 'option' in the URL or in the POST data. For our component, the URL would be:

```
index.php?option=com_hello&view=hello
```

This will load our main file, which can be seen as the single point of entry for our component: components/com_hello/hello.php.

Implementation

(preferred) HUBzero Methodology

HUBzero components differ in subtle, but key ways from standard Joomla! components. This is, in part, due to legacy issues. Some changes are made to aid in development while others may simply be a difference in philosophy. Note, however, that **no** differences require hacking or altering Joomla! in any way and HUBzero methodologies will run on any stock Joomla! install.

```
<?php
// No direct access
defined( '_JEXEC' ) or die( 'Restricted access' );

// Check if debugging is turned on
// If it is, we'll turn on PHP error reporting so we can more clearly
see our PHP bugs
if (JFactory::getConfig()->getValue('config.debug'))
{
 error_reporting(E_ALL);
 @ini_set('display_errors','1');
}

if (version_compare(JVERSION, '1.6', 'lt'))
{
 $jacl = JFactory::getACL();
 $jacl->addACL($option, 'manage', 'users', 'super administrator');
 $jacl->addACL($option, 'manage', 'users', 'administrator');
 $jacl->addACL($option, 'manage', 'users', 'manager');
```

```
}

jimport('joomla.application.component.helper');

// Get the requested controller
$controllerName = JRequest::getCmd('controller', 'one');
// Ensure the controller exists
if (!file_exists(JPATH_COMPONENT . DS . 'controllers' . DS . $controll
erName . '.php'))
{
 $controllerName = 'one';
}
require_once(JPATH_COMPONENT . DS . 'controllers' . DS . $controllerNa
me . '.php');
$controllerName = 'ExampleController' . ucfirst(strtolower($controller
Name));

// Instantiate controller
$controller = new $controllerName();
// Execute whatever task(s)
$controller->execute();
// Redirect as needed
$controller->redirect();
```

Here, you can see we've added a few things and made one subtle change in calling the execute() method. First, we added some lines that check if site debugging is turned on. If so, we turn on PHP error reporting. This can aid greatly in development.

Next, we added the jimport call to include some Joomla component helpers. This is done specifically because HUBzero controllers do **not** extend JController. JController does some autoloading and initial setup for Joomla! components and since we're not employing it, we need to do some class loading and setup of our own.

Then we look for a requested controller name. There is a default set in case none has been passed or if the requested controller is not found. With the controller name, we build the class name for the controller following the standard camel-cased pattern of {Component name}Controller{Controller name}

Finally, we removed the JRequest::getWord('task') being passed to the execute() method. HUBzero controllers handle the task request within the execute() method, rather than passing the task to it.

Joomla! 1.5 Methodology

ENTRY POINT

The code for this file is fairly typical across components.

```
<?php
// No direct access
defined( '_JEXEC' ) or die( 'Restricted access' );

// Require the base controller
require_once( JPATH_COMPONENT.DS.'controller.php' );

// Require specific controller if requested
if ($controller = JRequest::getWord('controller')) {
 $path = JPATH_COMPONENT.DS.'controllers'.DS.$controller.'.php';
 if (file_exists($path)) {
  require_once $path;
 } else {
  $controller = '';
 }
}

// Create the controller
$classname = 'HelloController'.$controller;
$controller = new $classname( );

// Perform the Request task
$controller->execute( JRequest::getWord( 'task' ) );

// Redirect if set by the controller
$controller->redirect();
```

The first statement is a security check.

JPATH_COMPONENT is the absolute path to the current component, in our case components/com_hello. If you specifically need either the Site component or the Administrator component, you can use JPATH_COMPONENT_SITE or JPATH_COMPONENT_ADMINISTRATOR.

DS is the directory separator of your system: either '/' or '\'. This is automatically set by the framework so the developer doesn't have to worry about developing different versions for different server OSs. The 'DS' constant should always be used when referring to files on the local server.

After loading the base controller, we check if a specific controller is needed. In this component,

the base controller is the only controller, but we will leave this conditional check "in place" for future use.

JRequest:getWord() finds a word variable in the URL or the POST data. So if our URL is index.php?option=com_hello&controller=controller_name, then we can retrieve our controller name in our component using: echo JRequest::getWord('controller');

Now we have our base controller 'HelloController' in com_hello/controller.php, and, if needed, additional controllers like 'HelloControllerController1' in com_hello/controllers/controller1.php. Using this standard naming scheme will make things easy later on: '{Componentname}{Controller}{Controllername}'

After the controller is created, we instruct the controller to execute the task, as defined in the URL: index.php?option=com_hello&task=sometask. If no task is set, the default task 'display' will be assumed. When display is used, the 'view' variable will decide what will be displayed. Other common tasks are save, edit, new...

The controller might decide to redirect the page, usually after a task like 'save' has been completed. This last statement takes care of the actual redirection.

The main entry point (hello.php) essentially passes control to the controller, which handles performing the task that was specified in the request.

Note that we don't use a closing php tag in this file: ?>. The reason for this is that we will not have any unwanted whitespace in the output code. This is default practice since Joomla! 1.5, and will be used for all php-only files.