Introduction

Getting Started

As a developer you are tasked with altering or extending the functionality of a HUBzero install or one of its extensions. You will need to be proficient in PHP and have some familiarity with such things as JavaScript or CSS. If you are new to HUBzero, this reference should help guide you through the creation of extensions such as modules and widgets (more on those later). Thankfully, the requirements for getting started creating HUBzero extensions are minimal: knowledge of programming in PHP and a good text editor. While those are the only *requirements* we do, however, recommend you have working knowledge of the following:

- HTML
- Cascading StyleSheets (CSS)
- JavaScript (familiarity with the jQuery framework is a plus)
- XML
- Model-View-Controller (MVC) design pattern
- Object-Oriented Programming

Release Notes

Changes

The Hubzero library underwent several significant changes.

Namespaced

One of the biggest changes was the namespacing of the Hubzero library. In most cases, this means a simple find & replace for Hubzero class names. Underscores "_" become back-slashes "\\". Example:

```
// old
Hubzero_User_Profile

// new
Hubzero\\User\\Profile
```

Helper Methods Renamed and Moved

Old	New
Hubzero_View_Helper_Html::niceidformat()	Hubzero\\Utility\\S
Hubzero_View_Helper_Html::formatSize()	Hubzero\\Utility\\N
Hubzero_View_Helper_Html::shortenText()	Hubzero\\Utility\\S
Hubzero_View_Helper_Html::purifyText()	Hubzero\\Utility\\S
Hubzero_View_Helper_Html::str_highlight()	Hubzero\\Utility\\S
Hubzero_View_Helper_Html::timeAgo()	JHTML::_('date.re

Portions of the Hubzero library were reorganized and, consequently, some class names changed.

Classes Moved and Renamed

Old	New
Hubzero_Group	Hubzero\\User\\Group
Hubzero_Group_Helper	Hubzero\\User\\Group\\He
Hubzero_Group_InviteEmail	Hubzero\\User\\Group\\Inv
Hubzero_Geo	Hubzero\\Geocode\\Geoc
Hubzero\\Object	Hubzero\\Base\\Object
Hubzero\\ItemList	Hubzero\\Base\\ItemList
Hubzero\\Model	Hubzero\\Base\\Model
Hubzero_Document	Hubzero\\Document\\Asse
Hubzero_Component	Hubzero\\Component\\{Sit
Hubzero_Api_Controller	Hubzero\\Component\\Api
Hubzero_Browser	Hubzero\\Browser\\Detect
Hubzero_Ldap	Hubzero\\Utility\\Ldap

The Hubzero\\Browser\\Detector class also had some methods named.

Renamed Methods Old	New
getBrowser()	name()
getBrowserVersion()	version()
getBrowserMajorVersion()	major()
getBrowserMinorVersion()	minor()
getOs()	platform()
getOsVersion()	platformVersion()
getUserAgent()	agent()

Deprecated

ximport()

> Namespaced Hubzero classes are now autoloaded and ximport() calls are now deprecated and should be removed where used.

Additions

New Classes

Along with the renaming and moving of several classes in the core Hubzero library, a handful of new classes were incorporated.

Class	Notes
HubzeroUtilityNumber	Various methods for manipulating and formatting numbers
HubzeroViewView	Base view class
HubzeroComponentView	Component view
HubzeroPluginView	Plugin view

Sub-view Helpers

Loading a sub-view (view within a view) can now be done via the view() method. This method accepts two arguments: 1) the view name and 2) the parent folder name [option]. If the second argument is not passed, the parent folder is inherited from the view the method is called from (i.e., $this).

```
<?php

$this->view('layout')
       ->set('foo', $bar)
```

```
        ->display();

?>
```

View Asset Helpers

Component and plugin views now have helpers for pushing Cascading StyleSheets and JavaScript assets to the document.

The css() method provides a quick and convenient way to attach stylesheets. For components, 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 or plugin will be used. For instance, if called within a view of the component com_tags, the system will look for a stylesheet named tags.css.
2. The name of the extension to look for the stylesheet. For components, this will be the component name (e.g., com_tags). For plugins, this is the name of the plugin folder and requires the third argument be passed to the method.
3. **Plugin views only.** The name of the plugin.

Method chaining is also allowed.

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

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 ...
```

Geocode Library & Plugins

The Hubzero library now comes with a helper class for various geocoding utilities. The class provides helpers for getting a list of countries, geocoding an address (i.e., getting longitude and latitude for a street address or IP address), and reverse geocoding an address (i.e., getting a street address for longitude and latitude).

When a method of the class is called (e.g. locate()), a plugin event is fired and any number of services may respond. A plugin for each available service resides in the newly created geocode plugins group.

Note: Some services may require registration.

Installation

Directories & File Structure

The initial directory structure of a HUBzero install.

```
/hubzero
  /administrator
  /api
  /bin
  /cache
  /cli
  /components
  /images
  /includes
  /language
  /libraries
  /logs
  /media
  /migrations
  /modules
  /plugins
  /site
  /templates
  /tmp
  /unittest
  /vendor
  configuration.php
  index.php
  htaccess.txt
  robots.txt
```

While this looks very much like past hub installs, there are some noticeable exceptions. Some directories vital to HUBzero functionality have been added. A quick explanation of the additional directories:

/api
: HUBzero comes with an API for accessing data from the various components and extensions in a light-weight, speedy manner. This directory contains the entry point to the API and can be accessed by visiting http://yourhub.org/api

/migrations
: This is where database migrations are stored and is vital to keeping an install up-to-

date.

/vendor

HUBzero uses Composer to manage several libraries that the framework employs. The vendor directory is the repository used and managed by Composer for those libraries and should not be directly altered.

Accessing Files

Accessing via SSH

The following tutorial should help you in using SSH to connect to and from your HUBzero server(s). You should be relatively comfortable with using a terminal (also referred to as a "command-line tool") to navigate directories and manipulate files.

Warning: Most accounts do **not** have SSH/sFTP access initially. Your system administrator must grant your account access before you will be able to connect.

From a terminal type ssh <user>@<host>. You will then be prompted for a password. Both the username and password will typically be the same as the account you registered on <host>.

```
yourmachine:~ you$ ssh username@host
yourmachine:~ you$ username@host password:

host ~
```

Windows Clients

- PuTTY (a Telnet and SSH client)

Mac OSX

All versions of Mac OSX come with Terminal.app which may be found in the /Utilities directory of your /Applications directory.

Accessing via sFTP

sFTP, or secure FTP, is a program that uses SSH to transfer files. Unlike standard FTP, it encrypts both commands and data, preventing passwords and sensitive information from being transmitted in the clear over the network. It is functionally similar to FTP, but because it uses a different protocol, you can't use a standard FTP client to talk to an sFTP server, nor can you connect to an FTP server with a client that supports only sFTP.

The following tutorial should help you in using sFTP to connect to and from your HUBzero server(s).

Warning: Most accounts do **not** have SSH/sFTP access initially. Your system administrator must grant your account access before you will be able to connect.

Graphical Clients

Using graphical SFTP clients simplifies file transfers by allowing you to transmit files simply by dragging and dropping icons between windows. When you open the program, you will have to enter the name of the host (e.g., yourhub.org) and your HUB username and password.

Windows Clients

- WinSCP
- BitKinex
- FileZilla
- PuTTY

Mac OSX Clients

- Transmit
- Fetch
- Cyberduck
- Flow
- Fugu

Command-line

You can use command line SFTP from your Unix account, or from your Mac OS X or Unix workstation. To start an SFTP session, at the command prompt, enter:

```
yourmachine:~ you$ sftp username@host
yourmachine:~ you$ username@host password:
```

host ~

Some standard commands for command-line sFTP

Command	Description
cd	Char
chmod	Char comp
chown	Char comp
dir (or ls)	List t remo
exit (or quit)	Close and e
get	Copy local
help (or ?)	Get h
lcd	Char

Command	Description
lls	See a
lmkdir	the lo Crea
ln (or symlink)	Crea comp
lpwd	Show direc
lumask	Char
mkdir	Crea
put	Copy remo
pwd	Show direc
rename	Rena
rm	Delet
rmdir	Remo direc
version	Displ
!	In Ur enter SFTP !pwd) dropp

Finding Files

Once connected to a server, by either sFTP or directly with SSH, you will need to find the web root which contains the HUB install. The web root for the production version of a HUB can be found at /www/yourhub. Typically, HUBs will also have a development version of a HUB, which can be found at /www/dev.

Once in the desired directory, file layout and directory structure follows the conventions detailed in Installation unless otherwise noted.

See the Installation overview for details on a typical HUBzero install's directory structure.

Direct Database Access

Accessing via command-line

The following tutorial should help you in using SSH to connect to and from your HUBzero server(s) and access the database. You should be relatively comfortable with using a terminal (also referred to as a "command-line tool") to navigate directories and manipulate files.

Warning: Most accounts do **not** have SSH/sFTP access initially. Your system administrator must grant your account access before you will be able to connect.

See Accessing Files for further details on how to use SSH.

Libraries

Hubzero

Location:

```
/libraries/Hubzero
```

The Hubzero library contains code that is essential for a hub to run properly and altering or adding to the library without Hubzero approval is *strongly* discouraged.

File Formatting

For files that contain only PHP code, the closing tag ("?>") is omitted. It is not required by PHP, and omitting it prevents the accidental injection of trailing white space into the response.

Class Names

Class names may only contain alphanumeric characters. Numbers are permitted in class names but are discouraged in most cases. Underscores are only permitted in place of the path separator; the filename "/libraries/Hubzero/User/Helper.php" must map to the class name "HubzeroUserHelper".

If a class name is comprised of more than one word, the first letter of each new word must be capitalized. Successive capitalized letters are not allowed, e.g. a class "HubzeroPDF" is not allowed while "HubzeroPdf" is acceptable.

Note: Code deployed alongside Hubzero libraries must never start with "Hubzero".

Filenames

Hubzero standardizes on a class naming convention whereby the names of the classes directly map to the directories in which they are stored. The root level directory of Hubzero's standard library is the "/libraries/Hubzero" directory. All Hubzero classes are stored hierarchically under this root directory.

For all other files, only alphanumeric characters, underscores, and the dash character ("-") are permitted. Spaces are strictly prohibited.

File names must map to class names as described above.

Debugging

Debug Mode

To turn on Debug mode:

- Login to the administration area e.g. http:/YOURSITE/administrator/
- At the top under the **Site** menu click **Global Configuration**.
- Click the **System** tab.
- Under the **Debug Settings** section change **Debug System** to *Yes*.
- Click the **Save** button.

Debug mode will output a list of all queries that were executed in order to generate the page. This will also turn on a stack trace output for error and warning pages. Hubzero components will also have PHP error reporting turned on, allowing one to see any PHP errors that may be present.

Note: Turning on debugging mode for production (live) sites is strongly discouraged and it is recommended to be avoided if at all possible.

Restricting who sees debug output

Since debug mode can contain potentially sensitive, it is **strongly** recommended that access to debug output is restricted to the administrator or super administrator user access levels and/or a defined list of users.

To restrict:

- Login to the administration area e.g. http:/YOURSITE/administrator/
- At the top under the **Extensions** menu click **Plugin Manager**.
- Select **System** from the "Select Type" drop-down.
- Find the debug plugin, typically titled "System - Debug", and click to edit.
- Under the **Parameters** section select the **Allowed Groups** and/or enter a comma-seprated list of usernames into the Allows Users box.
- Click the **Save** button.

Inspecting Variables

Hubzero provides the utility class HubzeroUtilityDebug for dumping variables.

dump()
: This will perform a print_r on the variable passed, wrapping the output in HTML <pre> tags.

stop()

This will perform a print_r on the variable passed, wrapping the output in HTML <pre> tags and die();.

log()

This method allows developers to dump variables to the debug toolbar, allowing data to be inspected without interrupting the flow or process of the code or output. **Note:** This feature requires the global Debug mode and system debug plugin to be enabled.

Example

```
$myvar = array(
    'one' => 'foo',
    'two' => 'bar',
);

HubzeroUtilityDebug::dump($myvar);
```

Illegal variable ... passed to script.

One encounters the following error:

Illegal variable _files or _env or _get or _post or _cookie or _server or _session or globals passed to script.

This error is generated when the key of a key-value pair is numeric in one of the following variables: _files or _env or _get or _post or _cookie or _server or _session or globals. An example of this would be $_POST[5] = 'value'. This is most often generated by having form elements with numeric values as names. For example:

```
<input type="text" name="5" />
```

As the error indicates, this is not allowed. Element names must include at least one non-numeric character. Examples:

```
<input type="text" name="n5" />

<input type="text" name="n_5" />
```