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.