Loading

Loading in Templates

Modules may be loaded in a template by including a specific jdoc:include tag. This tag includes two attributes: type, which must be specified as module in this case and name, which specifies the position that you wish to load. Any modules assigned to the specified position (set via the administrative Module Manager) declared in the name attribute will have their output placed in the template (the jdoc:include is removed by the CMS afterwards).

```
<jdoc:include type="modules" name="footer" />
```

Advanced Template Loading

The countModules method can be used within a template to determine the number of modules enabled in a given module position. This is commonly used to include HTML around modules in a certain position only if at least one module is enabled for that position. This prevents empty regions from being defined in the template output and is a technique sometimes referred to as "collapsing columns".

For example, the following code includes modules in the 'user1' position only if at least one module is enabled for that position.

```
<?php if ($this->countModules( 'user1' )) : ?>
 <div class="user1">
  <jdoc:include type="modules" name="user1" />
 </div>
<?php endif; ?>
```

The countModules method can be used to determine the number of Modules in more than one Module position. More advanced calculations can also be performed.

The argument to the countModules function is normally just the name of a single Module position. The function will return the number of Modules currently enabled for that Module position. But you can also do simple logical and arithmetic operations on two or more Module positions.

```
$this->countModules('user1 + user2');
```

Although the usual arithmetic operators, +. -. *, / will work as expected, these are not as useful as the logical operators 'and' and 'or'. For example, to determine if the 'user1' position and the 'user2' position both have at least one Module enabled, you can use the function call:

```
$this->countModules('user1 and user2');
```

Careful: A common mistake is to try something like this:

```
$this->countModules('user1' and 'user2');
```

This will return false regardless of the number of Modules enabled in either position, so check what you are passing to countModules carefully.

You must have exactly one space character separating each item in the string. For example, 'user1+user2' will not produce the desired result as there must be a space character either side of the '+' sign. Also, 'user1 &nbp;+ user2' will produce an error message as there is more than one space separating each element.

Example using the or operator: The user1 and user2 Module positions are to be displayed in the region, but you want the region to not appear at all if no Modules are enabled in either position.

```
<?php if ($this->countModules('user1 or user2')) : ?>
 <div class="container">
  <jdoc:include type="modules" name="user1" />
  <jdoc:include type="modules" name="user2" />
 </div>
<?php endif; ?>
```

Advanced example: The user1 and user2 Module positions are to be displayed side-by-side with a separator between them. However, if only one of the Module positions has any Modules enabled then the separator is not needed. Furthermore, if neither user1 or user2 has any Modules enabled then nothing is output.

```
<?php if ($this->countModules('user1 or user2')) : ?>
 <div class="user1user2">

 <?php if ($this->countModules('user1')) : ?>
```

```
  <jdoc:include type="modules" name="user1" />
 <?php endif; ?>

 <?php if ($this->countModules('user1 and user2')) : ?>
  <div class="greyline"></div>
 <?php endif; ?>

 <?php if ($this->countModules('user2')) : ?>
  <jdoc:include type="modules" name="user2" />
 <?php endif; ?>

 </div>
<?php endif; ?>
```

Notice how the first countModules call determines if there any Modules to display at all. The second determines if there are any in the 'user1' position and if there are it displays them. The third call determines if both 'user1' and 'user2' positions have any Modules enabled and if they do then if provides a separator between them. Finally, the fourth call determines if there are any enabled Modules in the 'user2' position and displays them if there are any.

Loading in Components

Sometimes it is necessary to render a module within a component. This can be done with the HubzeroModuleHelper class provided by HUBzero.

HubzeroModuleHelper::renderModules($position)

> Used for loading potentially multiple modules assigned to a position. This will capture the rendered output of all modules assigned to the $position parameter passed to it and return the compiled output.
>
> ```
> $output = HubzeroModuleHelper::renderModules('footer');
> ```

HubzeroModuleHelper::renderModule($name)

> Used for loading a single module of a specific name. This will capture the rendered output of the module with the $name parameter passed to it and return the compiled output.

```
$output = HubzeroModuleHelper::renderModule('mod_footer');
```

HubzeroModuleHelper::displayModules($position)

Used for loading a single module of a specific name. This will echo rendered output of the module with the $name parameter passed to it.

```
HubzeroModuleHelper::displayModules('footer');
```

HubzeroModuleHelper::renderModule($name)

Used for loading a single module of a specific name. This will output the module with the $name parameter passed to it.

```
HubzeroModuleHelper::displayModule('mod_footer');
```

Loading in Articles

Modules may be loaded in an article by including a specific {xhub:module} tag. This tag includes one required attribute: position, which specifies the position that you wish to load. Any modules assigned to the specified position (set via the administrative Module Manager) declared in the position attribute will have their output placed in the article in the location of the {xhub:module} tag.

```
{xhub:module position="footer"}
```

Note: To use this feature, the xhub Tags plugin for content must be installed and active.