Modules Library

revIgniter provides a modules library which needs to be loaded in your application controller if you intend to use modules as widgets or view partials, respectively, or as libraries. Please read more about modules in section HMVC – Extending MVC and in section Modules.

Initializing the Library

Like most other libraries in revIgniter, the modules library is initialized in your controller using the **rigLoadLoadLibrary** handler:

```
rigLoaderLoadLibrary "Modules"
```

Note: There is no need to load the modules library if a module controller is used as application controller and no further modules are incorporated as widgets.

Handler Reference

rigLoadModule pModuleSegments

After loading the modules library use this handler to load module controllers (as many as you like) within an application controller or within a module. Please read about module controller specifications in the "Modules" section. The parameter is the path to a module controller relative to **application/modules/** excluding the **controllers** directory. Do not include the file extension (.lc).

In the following example this handler loads module controller **myCoolModule.lc** located in **application/modules/myCoolModule/controllers** and runs the main handler **myCoolModule**. Note: In case the controller in question is named after your module the path specified in the parameter doesn't need to include the name of the controller.

```
rigLoadModule "myCoolModule"
```

The next example loads module controller **myModuleController.lc** located in **application/modules/myCoolModule/controllers** and runs the main handler **myModuleController**.

```
rigLoadModule "myCoolModule/myModuleController"
```

In the following example this handler loads module controller **someModuleController.lc** located in **application/modules/myCoolModule/controllers/mySubDirectory** and runs the main handler **someModuleController**.

```
rigLoadModule "myCoolModule/mySubDirectory/someModuleController"
```

rigRunModule(pModuleSegments, pParamsA)

Returns output (a view) or a return value from a module controller. The first parameter is the path to a module controller optionally including additional segments used to pass parameter values. Please see module paths in the "Modules" section. The second parameter is optional and is used to pass parameters as an array.

In the first example this function calls the default handler **myCoolModuleIndex** of controller **myCoolModule.lc** which is located in **application/modules/myCoolModule/controllers**. The view or data returned is stored in **gData**.

```
put rigRunModule("myCoolModule") into gData["myCoolModuleOutput"]
```

In the next example this function calls handler **someHandler** of controller **myCoolModule.lc** which is located in **application/modules/myCoolModule/controllers**.

```
put rigRunModule("myCoolModule/someHandler") into gData["myCoolModuleOutput"]
```

In the next example this function calls handler **someHandler** of controller **someModuleController.lc** which is located in **application/modules/myCoolModule/controllers**

```
put rigRunModule("myCoolModule/someModuleController/someHandler") into gData["myCoolModuleOutput"]
```

In this last example this function calls handler **someHandler** of controller **someModuleController.lc** which is located in **application/modules/myCoolModule/controllers/mySubDirectory**.

```
put rigRunModule("myCoolModule/mySubDirectory/someModuleController/someHandler") into gData["myCoolModuleOutput"]
```

Running a Module Within a View

If your module handler returns a view or data which should be displayed in the browser you can call it within views like:

```
<? get rigRunModule("myCoolModule") ?>
```

Here is another example:

```
<? get rigRunModule("myCoolModule/someModuleController/someHandler/name/joe/location/UK/gender/male") ?>
```

Note: Before calling rigRunModule() it is required to load the appropriate module controller.

rigFetchModuleSegment(pN, pNoResult)

This function returns the URI segment, or URI value based on the number provided. Where **pN** is the segment number you wish to retrieve. Segments are numbered from left to right. For example, if your module URI is this:

```
myCoolModule/someModuleController/someHandler/name/joe/location/UK/gender/male
```

The segment numbers would be this:

1. myCoolModule
2. someModuleController
3. someHandler
4. name
5. joe
6. location
7. UK
8. gender
9. male

So, you could retrieve for example "name" and "joe" like:

```
put rigFetchModuleSegment(4) && "=" && rigFetchModuleSegment(5) into tKeyValue
```

By default the function returns FALSE if the segment does not exist. The optional second parameter permits you to set your own default value if the segment is missing. For example, this would tell the function to return "n/a" in the event of failure:

```
put rigFetchModuleSegment(5, "n/a") into tName
```

rigModuleUriToAssoc(pN, pDefaultA)

This function generates an associative array of module URI data starting at the supplied segment. For example, if this is your URI: **myCoolModule/someHandler/name/joe/location/UK/gender/male**, you can use this function to generate an array with this prototype:

```
tArray["name"] =>value = "joe"
tArray["location"] =>value = "UK"
tArray["gender"] =>value = "male"
```

The first parameter of the function lets you set an offset. By default it is set to 3 assuming your module URI contains a myModule/handler in the first and second segments. Example:

```
put rigModuleUriToAssoc(3) into tArray

put tArray["name"] into gData["name"]
```

The second parameter lets you set default key names, so that the array returned by the function will always contain expected indexes, even if missing from the URI. Example:

```
put "name,location,gender" into tDefault
split tDefault using comma

put rigModuleUriToAssoc(3, tDefault) into tArray
```

If the module URI does not contain a value in your default array, an array index will be set to that name, with a value of FALSE.

Lastly, if a corresponding value is not found for a given key (if there is an odd number of URI segments) the value will be set to FALSE.