* [Welcome to CodeIgniter](http://docs.google.com/welcome.html)
* [Installation Instructions](http://docs.google.com/installation/index.html)
  + [Downloading CodeIgniter](http://docs.google.com/installation/downloads.html)
  + [Installation Instructions](http://docs.google.com/installation/index.html)
  + [Upgrading From a Previous Version](http://docs.google.com/installation/upgrading.html)
  + [Troubleshooting](http://docs.google.com/installation/troubleshooting.html)
* [CodeIgniter Overview](http://docs.google.com/overview/index.html)
  + [Getting Started](http://docs.google.com/overview/getting_started.html)
  + [CodeIgniter at a Glance](http://docs.google.com/overview/at_a_glance.html)
  + [Supported Features](http://docs.google.com/overview/features.html)
  + [Application Flow Chart](http://docs.google.com/overview/appflow.html)
  + [Model-View-Controller](http://docs.google.com/overview/mvc.html)
  + [Architectural Goals](http://docs.google.com/overview/goals.html)
* [Tutorial](http://docs.google.com/tutorial/index.html)
  + [Static pages](http://docs.google.com/tutorial/static_pages.html)
  + [News section](http://docs.google.com/tutorial/news_section.html)
  + [Create news items](http://docs.google.com/tutorial/create_news_items.html)
  + [Conclusion](http://docs.google.com/tutorial/conclusion.html)
* [Contributing to CodeIgniter](http://docs.google.com/contributing/index.html)
  + [Writing CodeIgniter Documentation](http://docs.google.com/documentation/index.html)
  + [Developer’s Certificate of Origin 1.1](http://docs.google.com/DCO.html)
* [General Topics](http://docs.google.com/index.html)
  + [CodeIgniter URLs](http://docs.google.com/urls.html)
  + Controllers
  + [Reserved Names](http://docs.google.com/reserved_names.html)
  + [Views](http://docs.google.com/views.html)
  + [Models](http://docs.google.com/models.html)
  + [Helpers](http://docs.google.com/helpers.html)
  + [Using CodeIgniter Libraries](http://docs.google.com/libraries.html)
  + [Creating Libraries](http://docs.google.com/creating_libraries.html)
  + [Using CodeIgniter Drivers](http://docs.google.com/drivers.html)
  + [Creating Drivers](http://docs.google.com/creating_drivers.html)
  + [Creating Core System Classes](http://docs.google.com/core_classes.html)
  + [Creating Ancillary Classes](http://docs.google.com/ancillary_classes.html)
  + [Hooks - Extending the Framework Core](http://docs.google.com/hooks.html)
  + [Auto-loading Resources](http://docs.google.com/autoloader.html)
  + [Common Functions](http://docs.google.com/common_functions.html)
  + [Compatibility Functions](http://docs.google.com/compatibility_functions.html)
  + [URI Routing](http://docs.google.com/routing.html)
  + [Error Handling](http://docs.google.com/errors.html)
  + [Caching](http://docs.google.com/caching.html)
  + [Profiling Your Application](http://docs.google.com/profiling.html)
  + [Running via the CLI](http://docs.google.com/cli.html)
  + [Managing your Applications](http://docs.google.com/managing_apps.html)
  + [Handling Multiple Environments](http://docs.google.com/environments.html)
  + [Alternate PHP Syntax for View Files](http://docs.google.com/alternative_php.html)
  + [Security](http://docs.google.com/security.html)
  + [PHP Style Guide](http://docs.google.com/styleguide.html)
* [Libraries](http://docs.google.com/libraries/index.html)
  + [Benchmarking Class](http://docs.google.com/libraries/benchmark.html)
  + [Caching Driver](http://docs.google.com/libraries/caching.html)
  + [Calendaring Class](http://docs.google.com/libraries/calendar.html)
  + [Shopping Cart Class](http://docs.google.com/libraries/cart.html)
  + [Config Class](http://docs.google.com/libraries/config.html)
  + [Email Class](http://docs.google.com/libraries/email.html)
  + [Encrypt Class](http://docs.google.com/libraries/encrypt.html)
  + [Encryption Library](http://docs.google.com/libraries/encryption.html)
  + [File Uploading Class](http://docs.google.com/libraries/file_uploading.html)
  + [Form Validation](http://docs.google.com/libraries/form_validation.html)
  + [FTP Class](http://docs.google.com/libraries/ftp.html)
  + [Image Manipulation Class](http://docs.google.com/libraries/image_lib.html)
  + [Input Class](http://docs.google.com/libraries/input.html)
  + [Javascript Class](http://docs.google.com/libraries/javascript.html)
  + [Language Class](http://docs.google.com/libraries/language.html)
  + [Loader Class](http://docs.google.com/libraries/loader.html)
  + [Migrations Class](http://docs.google.com/libraries/migration.html)
  + [Output Class](http://docs.google.com/libraries/output.html)
  + [Pagination Class](http://docs.google.com/libraries/pagination.html)
  + [Template Parser Class](http://docs.google.com/libraries/parser.html)
  + [Security Class](http://docs.google.com/libraries/security.html)
  + [Session Library](http://docs.google.com/libraries/sessions.html)
  + [HTML Table Class](http://docs.google.com/libraries/table.html)
  + [Trackback Class](http://docs.google.com/libraries/trackback.html)
  + [Typography Class](http://docs.google.com/libraries/typography.html)
  + [Unit Testing Class](http://docs.google.com/libraries/unit_testing.html)
  + [URI Class](http://docs.google.com/libraries/uri.html)
  + [User Agent Class](http://docs.google.com/libraries/user_agent.html)
  + [XML-RPC and XML-RPC Server Classes](http://docs.google.com/libraries/xmlrpc.html)
  + [Zip Encoding Class](http://docs.google.com/libraries/zip.html)
* [Database Reference](http://docs.google.com/database/index.html)
  + [Quick Start: Usage Examples](http://docs.google.com/database/examples.html)
  + [Database Configuration](http://docs.google.com/database/configuration.html)
  + [Connecting to a Database](http://docs.google.com/database/connecting.html)
  + [Running Queries](http://docs.google.com/database/queries.html)
  + [Generating Query Results](http://docs.google.com/database/results.html)
  + [Query Helper Functions](http://docs.google.com/database/helpers.html)
  + [Query Builder Class](http://docs.google.com/database/query_builder.html)
  + [Transactions](http://docs.google.com/database/transactions.html)
  + [Getting MetaData](http://docs.google.com/database/metadata.html)
  + [Custom Function Calls](http://docs.google.com/database/call_function.html)
  + [Query Caching](http://docs.google.com/database/caching.html)
  + [Database Manipulation with Database Forge](http://docs.google.com/database/forge.html)
  + [Database Utilities Class](http://docs.google.com/database/utilities.html)
  + [Database Driver Reference](http://docs.google.com/database/db_driver_reference.html)
* [Helpers](http://docs.google.com/helpers/index.html)
  + [Array Helper](http://docs.google.com/helpers/array_helper.html)
  + [CAPTCHA Helper](http://docs.google.com/helpers/captcha_helper.html)
  + [Cookie Helper](http://docs.google.com/helpers/cookie_helper.html)
  + [Date Helper](http://docs.google.com/helpers/date_helper.html)
  + [Directory Helper](http://docs.google.com/helpers/directory_helper.html)
  + [Download Helper](http://docs.google.com/helpers/download_helper.html)
  + [Email Helper](http://docs.google.com/helpers/email_helper.html)
  + [File Helper](http://docs.google.com/helpers/file_helper.html)
  + [Form Helper](http://docs.google.com/helpers/form_helper.html)
  + [HTML Helper](http://docs.google.com/helpers/html_helper.html)
  + [Inflector Helper](http://docs.google.com/helpers/inflector_helper.html)
  + [Language Helper](http://docs.google.com/helpers/language_helper.html)
  + [Number Helper](http://docs.google.com/helpers/number_helper.html)
  + [Path Helper](http://docs.google.com/helpers/path_helper.html)
  + [Security Helper](http://docs.google.com/helpers/security_helper.html)
  + [Smiley Helper](http://docs.google.com/helpers/smiley_helper.html)
  + [String Helper](http://docs.google.com/helpers/string_helper.html)
  + [Text Helper](http://docs.google.com/helpers/text_helper.html)
  + [Typography Helper](http://docs.google.com/helpers/typography_helper.html)
  + [URL Helper](http://docs.google.com/helpers/url_helper.html)
  + [XML Helper](http://docs.google.com/helpers/xml_helper.html)

Toggle Table of Contents

[CodeIgniter](http://docs.google.com/index.html)

* [Welcome to CodeIgniter](http://docs.google.com/welcome.html)
* [Installation Instructions](http://docs.google.com/installation/index.html)
  + [Downloading CodeIgniter](http://docs.google.com/installation/downloads.html)
  + [Installation Instructions](http://docs.google.com/installation/index.html)
  + [Upgrading From a Previous Version](http://docs.google.com/installation/upgrading.html)
  + [Troubleshooting](http://docs.google.com/installation/troubleshooting.html)
* [CodeIgniter Overview](http://docs.google.com/overview/index.html)
  + [Getting Started](http://docs.google.com/overview/getting_started.html)
  + [CodeIgniter at a Glance](http://docs.google.com/overview/at_a_glance.html)
  + [Supported Features](http://docs.google.com/overview/features.html)
  + [Application Flow Chart](http://docs.google.com/overview/appflow.html)
  + [Model-View-Controller](http://docs.google.com/overview/mvc.html)
  + [Architectural Goals](http://docs.google.com/overview/goals.html)
* [Tutorial](http://docs.google.com/tutorial/index.html)
  + [Static pages](http://docs.google.com/tutorial/static_pages.html)
  + [News section](http://docs.google.com/tutorial/news_section.html)
  + [Create news items](http://docs.google.com/tutorial/create_news_items.html)
  + [Conclusion](http://docs.google.com/tutorial/conclusion.html)
* [Contributing to CodeIgniter](http://docs.google.com/contributing/index.html)
  + [Writing CodeIgniter Documentation](http://docs.google.com/documentation/index.html)
  + [Developer’s Certificate of Origin 1.1](http://docs.google.com/DCO.html)
* [General Topics](http://docs.google.com/index.html)
  + [CodeIgniter URLs](http://docs.google.com/urls.html)
  + Controllers
  + [Reserved Names](http://docs.google.com/reserved_names.html)
  + [Views](http://docs.google.com/views.html)
  + [Models](http://docs.google.com/models.html)
  + [Helpers](http://docs.google.com/helpers.html)
  + [Using CodeIgniter Libraries](http://docs.google.com/libraries.html)
  + [Creating Libraries](http://docs.google.com/creating_libraries.html)
  + [Using CodeIgniter Drivers](http://docs.google.com/drivers.html)
  + [Creating Drivers](http://docs.google.com/creating_drivers.html)
  + [Creating Core System Classes](http://docs.google.com/core_classes.html)
  + [Creating Ancillary Classes](http://docs.google.com/ancillary_classes.html)
  + [Hooks - Extending the Framework Core](http://docs.google.com/hooks.html)
  + [Auto-loading Resources](http://docs.google.com/autoloader.html)
  + [Common Functions](http://docs.google.com/common_functions.html)
  + [Compatibility Functions](http://docs.google.com/compatibility_functions.html)
  + [URI Routing](http://docs.google.com/routing.html)
  + [Error Handling](http://docs.google.com/errors.html)
  + [Caching](http://docs.google.com/caching.html)
  + [Profiling Your Application](http://docs.google.com/profiling.html)
  + [Running via the CLI](http://docs.google.com/cli.html)
  + [Managing your Applications](http://docs.google.com/managing_apps.html)
  + [Handling Multiple Environments](http://docs.google.com/environments.html)
  + [Alternate PHP Syntax for View Files](http://docs.google.com/alternative_php.html)
  + [Security](http://docs.google.com/security.html)
  + [PHP Style Guide](http://docs.google.com/styleguide.html)
* [Libraries](http://docs.google.com/libraries/index.html)
  + [Benchmarking Class](http://docs.google.com/libraries/benchmark.html)
  + [Caching Driver](http://docs.google.com/libraries/caching.html)
  + [Calendaring Class](http://docs.google.com/libraries/calendar.html)
  + [Shopping Cart Class](http://docs.google.com/libraries/cart.html)
  + [Config Class](http://docs.google.com/libraries/config.html)
  + [Email Class](http://docs.google.com/libraries/email.html)
  + [Encrypt Class](http://docs.google.com/libraries/encrypt.html)
  + [Encryption Library](http://docs.google.com/libraries/encryption.html)
  + [File Uploading Class](http://docs.google.com/libraries/file_uploading.html)
  + [Form Validation](http://docs.google.com/libraries/form_validation.html)
  + [FTP Class](http://docs.google.com/libraries/ftp.html)
  + [Image Manipulation Class](http://docs.google.com/libraries/image_lib.html)
  + [Input Class](http://docs.google.com/libraries/input.html)
  + [Javascript Class](http://docs.google.com/libraries/javascript.html)
  + [Language Class](http://docs.google.com/libraries/language.html)
  + [Loader Class](http://docs.google.com/libraries/loader.html)
  + [Migrations Class](http://docs.google.com/libraries/migration.html)
  + [Output Class](http://docs.google.com/libraries/output.html)
  + [Pagination Class](http://docs.google.com/libraries/pagination.html)
  + [Template Parser Class](http://docs.google.com/libraries/parser.html)
  + [Security Class](http://docs.google.com/libraries/security.html)
  + [Session Library](http://docs.google.com/libraries/sessions.html)
  + [HTML Table Class](http://docs.google.com/libraries/table.html)
  + [Trackback Class](http://docs.google.com/libraries/trackback.html)
  + [Typography Class](http://docs.google.com/libraries/typography.html)
  + [Unit Testing Class](http://docs.google.com/libraries/unit_testing.html)
  + [URI Class](http://docs.google.com/libraries/uri.html)
  + [User Agent Class](http://docs.google.com/libraries/user_agent.html)
  + [XML-RPC and XML-RPC Server Classes](http://docs.google.com/libraries/xmlrpc.html)
  + [Zip Encoding Class](http://docs.google.com/libraries/zip.html)
* [Database Reference](http://docs.google.com/database/index.html)
  + [Quick Start: Usage Examples](http://docs.google.com/database/examples.html)
  + [Database Configuration](http://docs.google.com/database/configuration.html)
  + [Connecting to a Database](http://docs.google.com/database/connecting.html)
  + [Running Queries](http://docs.google.com/database/queries.html)
  + [Generating Query Results](http://docs.google.com/database/results.html)
  + [Query Helper Functions](http://docs.google.com/database/helpers.html)
  + [Query Builder Class](http://docs.google.com/database/query_builder.html)
  + [Transactions](http://docs.google.com/database/transactions.html)
  + [Getting MetaData](http://docs.google.com/database/metadata.html)
  + [Custom Function Calls](http://docs.google.com/database/call_function.html)
  + [Query Caching](http://docs.google.com/database/caching.html)
  + [Database Manipulation with Database Forge](http://docs.google.com/database/forge.html)
  + [Database Utilities Class](http://docs.google.com/database/utilities.html)
  + [Database Driver Reference](http://docs.google.com/database/db_driver_reference.html)
* [Helpers](http://docs.google.com/helpers/index.html)
  + [Array Helper](http://docs.google.com/helpers/array_helper.html)
  + [CAPTCHA Helper](http://docs.google.com/helpers/captcha_helper.html)
  + [Cookie Helper](http://docs.google.com/helpers/cookie_helper.html)
  + [Date Helper](http://docs.google.com/helpers/date_helper.html)
  + [Directory Helper](http://docs.google.com/helpers/directory_helper.html)
  + [Download Helper](http://docs.google.com/helpers/download_helper.html)
  + [Email Helper](http://docs.google.com/helpers/email_helper.html)
  + [File Helper](http://docs.google.com/helpers/file_helper.html)
  + [Form Helper](http://docs.google.com/helpers/form_helper.html)
  + [HTML Helper](http://docs.google.com/helpers/html_helper.html)
  + [Inflector Helper](http://docs.google.com/helpers/inflector_helper.html)
  + [Language Helper](http://docs.google.com/helpers/language_helper.html)
  + [Number Helper](http://docs.google.com/helpers/number_helper.html)
  + [Path Helper](http://docs.google.com/helpers/path_helper.html)
  + [Security Helper](http://docs.google.com/helpers/security_helper.html)
  + [Smiley Helper](http://docs.google.com/helpers/smiley_helper.html)
  + [String Helper](http://docs.google.com/helpers/string_helper.html)
  + [Text Helper](http://docs.google.com/helpers/text_helper.html)
  + [Typography Helper](http://docs.google.com/helpers/typography_helper.html)
  + [URL Helper](http://docs.google.com/helpers/url_helper.html)
  + [XML Helper](http://docs.google.com/helpers/xml_helper.html)

[CodeIgniter](http://docs.google.com/index.html)

* [Docs](http://docs.google.com/index.html) »
* [General Topics](http://docs.google.com/index.html) »
* Controllers

classic layout

[**Controllers**](#gjdgxs)[**¶**](#30j0zll)

Controllers are the heart of your application, as they determine how HTTP requests should be handled.

Page Contents

* [Controllers](#30j0zll)
  + [What is a Controller?](#1fob9te)
  + [Let’s try it: Hello World!](#3znysh7)
  + [Methods](#2et92p0)
  + [Passing URI Segments to your methods](#tyjcwt)
  + [Defining a Default Controller](#3dy6vkm)
  + [Remapping Method Calls](#1t3h5sf)
  + [Processing Output](#4d34og8)
  + [Private methods](#2s8eyo1)
  + [Organizing Your Controllers into Sub-directories](#17dp8vu)
  + [Class Constructors](#3rdcrjn)
  + [Reserved method names](#26in1rg)
  + [That’s it!](#lnxbz9)

[**What is a Controller?**](#35nkun2)[**¶**](#1fob9te)

**A Controller is simply a class file that is named in a way that can be associated with a URI.**

Consider this URI:

example.com/index.php/blog/

In the above example, CodeIgniter would attempt to find a controller named Blog.php and load it.

**When a controller’s name matches the first segment of a URI, it will be loaded.**

[**Let’s try it: Hello World!**](#1ksv4uv)[**¶**](#3znysh7)

Let’s create a simple controller so you can see it in action. Using your text editor, create a file called Blog.php, and put the following code in it:

<?php  
class Blog extends CI\_Controller {  
  
 public function index()  
 {  
 echo 'Hello World!';  
 }  
}

Then save the file to your *application/controllers/* directory.

Important

The file must be called ‘Blog.php’, with a capital ‘B’.

Now visit the your site using a URL similar to this:

example.com/index.php/blog/

If you did it right, you should see:

Hello World!

Important

Class names must start with an uppercase letter.

This is valid:

<?php  
class Blog extends CI\_Controller {  
  
}

This is **not** valid:

<?php  
class blog extends CI\_Controller {  
  
}

Also, always make sure your controller extends the parent controller class so that it can inherit all its methods.

[**Methods**](#44sinio)[**¶**](#2et92p0)

In the above example the method name is index(). The “index” method is always loaded by default if the **second segment** of the URI is empty. Another way to show your “Hello World” message would be this:

example.com/index.php/blog/index/

**The second segment of the URI determines which method in the controller gets called.**

Let’s try it. Add a new method to your controller:

<?php  
class Blog extends CI\_Controller {  
  
 public function index()  
 {  
 echo 'Hello World!';  
 }  
  
 public function comments()  
 {  
 echo 'Look at this!';  
 }  
}

Now load the following URL to see the comment method:

example.com/index.php/blog/comments/

You should see your new message.

[**Passing URI Segments to your methods**](#2jxsxqh)[**¶**](#tyjcwt)

If your URI contains more than two segments they will be passed to your method as parameters.

For example, let’s say you have a URI like this:

example.com/index.php/products/shoes/sandals/123

Your method will be passed URI segments 3 and 4 (“sandals” and “123”):

<?php  
class Products extends CI\_Controller {  
  
 public function shoes($sandals, $id)  
 {  
 echo $sandals;  
 echo $id;  
 }  
}

Important

If you are using the [*URI Routing*](http://docs.google.com/routing.html) feature, the segments passed to your method will be the re-routed ones.

[**Defining a Default Controller**](#z337ya)[**¶**](#3dy6vkm)

CodeIgniter can be told to load a default controller when a URI is not present, as will be the case when only your site root URL is requested. To specify a default controller, open your **application/config/routes.php** file and set this variable:

$route['default\_controller'] = 'blog';

Where ‘blog’ is the name of the controller class you want used. If you now load your main index.php file without specifying any URI segments you’ll see your “Hello World” message by default.

For more information, please refer to the “Reserved Routes” section of the [*URI Routing*](http://docs.google.com/routing.html) documentation.

[**Remapping Method Calls**](#3j2qqm3)[**¶**](#1t3h5sf)

As noted above, the second segment of the URI typically determines which method in the controller gets called. CodeIgniter permits you to override this behavior through the use of the \_remap() method:

public function \_remap()  
{  
 // Some code here...  
}

Important

If your controller contains a method named \_remap(), it will **always** get called regardless of what your URI contains. It overrides the normal behavior in which the URI determines which method is called, allowing you to define your own method routing rules.

The overridden method call (typically the second segment of the URI) will be passed as a parameter to the \_remap() method:

public function \_remap($method)  
{  
 if ($method === 'some\_method')  
 {  
 $this->$method();  
 }  
 else  
 {  
 $this->default\_method();  
 }  
}

Any extra segments after the method name are passed into \_remap() as an optional second parameter. This array can be used in combination with PHP’s [call\_user\_func\_array()](http://php.net/call_user_func_array) to emulate CodeIgniter’s default behavior.

Example:

public function \_remap($method, $params = array())  
{  
 $method = 'process\_'.$method;  
 if (method\_exists($this, $method))  
 {  
 return call\_user\_func\_array(array($this, $method), $params);  
 }  
 show\_404();  
}

[**Processing Output**](#1y810tw)[**¶**](#4d34og8)

CodeIgniter has an output class that takes care of sending your final rendered data to the web browser automatically. More information on this can be found in the [*Views*](http://docs.google.com/views.html) and [*Output Class*](http://docs.google.com/libraries/output.html) pages. In some cases, however, you might want to post-process the finalized data in some way and send it to the browser yourself. CodeIgniter permits you to add a method named \_output() to your controller that will receive the finalized output data.

Important

If your controller contains a method named \_output(), it will **always** be called by the output class instead of echoing the finalized data directly. The first parameter of the method will contain the finalized output.

Here is an example:

public function \_output($output)  
{  
 echo $output;  
}

Note

Please note that your \_output() method will receive the data in its finalized state. Benchmark and memory usage data will be rendered, cache files written (if you have caching enabled), and headers will be sent (if you use that [*feature*](http://docs.google.com/libraries/output.html)) before it is handed off to the \_output() method. To have your controller’s output cached properly, its \_output() method can use:

if ($this->output->cache\_expiration > 0)  
{  
 $this->output->\_write\_cache($output);  
}

If you are using this feature the page execution timer and memory usage stats might not be perfectly accurate since they will not take into account any further processing you do. For an alternate way to control output *before* any of the final processing is done, please see the available methods in the [*Output Library*](http://docs.google.com/libraries/output.html).

[**Private methods**](#4i7ojhp)[**¶**](#2s8eyo1)

In some cases you may want certain methods hidden from public access. In order to achieve this, simply declare the method as being private or protected and it will not be served via a URL request. For example, if you were to have a method like this:

private function \_utility()  
{  
 // some code  
}

Trying to access it via the URL, like this, will not work:

example.com/index.php/blog/\_utility/

Note

Prefixing method names with an underscore will also prevent them from being called. This is a legacy feature that is left for backwards-compatibility.

[**Organizing Your Controllers into Sub-directories**](#2xcytpi)[**¶**](#17dp8vu)

If you are building a large application you might want to hierarchically organize or structure your controllers into sub-directories. CodeIgniter permits you to do this.

Simply create sub-directories under the main *application/controllers/* one and place your controller classes within them.

Note

When using this feature the first segment of your URI must specify the folder. For example, let’s say you have a controller located here:

application/controllers/products/Shoes.php

To call the above controller your URI will look something like this:

example.com/index.php/products/shoes/show/123

Each of your sub-directories may contain a default controller which will be called if the URL contains *only* the sub-directory. Simply put a controller in there that matches the name of your ‘default\_controller’ as specified in your *application/config/routes.php* file.

CodeIgniter also permits you to remap your URIs using its [*URI Routing*](http://docs.google.com/routing.html) feature.

[**Class Constructors**](#1ci93xb)[**¶**](#3rdcrjn)

If you intend to use a constructor in any of your Controllers, you **MUST** place the following line of code in it:

parent::\_\_construct();

The reason this line is necessary is because your local constructor will be overriding the one in the parent controller class so we need to manually call it.

Example:

<?php  
class Blog extends CI\_Controller {  
  
 public function \_\_construct()  
 {  
 parent::\_\_construct();  
 // Your own constructor code  
 }  
}

Constructors are useful if you need to set some default values, or run a default process when your class is instantiated. Constructors can’t return a value, but they can do some default work.

[**Reserved method names**](#3whwml4)[**¶**](#26in1rg)

Since your controller classes will extend the main application controller you must be careful not to name your methods identically to the ones used by that class, otherwise your local functions will override them. See [*Reserved Names*](http://docs.google.com/reserved_names.html) for a full list.

Important

You should also never have a method named identically to its class name. If you do, and there is no \_\_construct() method in the same class, then your e.g. Index::index() method will be executed as a class constructor! This is a PHP4 backwards-compatibility feature.

[**That’s it!**](#2bn6wsx)[**¶**](#lnxbz9)

That, in a nutshell, is all there is to know about controllers.

[Next](http://docs.google.com/reserved_names.html)   [Previous](http://docs.google.com/urls.html)

© Copyright 2014 - 2016, British Columbia Institute of Technology. Last updated on Oct 28, 2016.

Built with [Sphinx](http://sphinx-doc.org/) using a [theme](https://github.com/snide/sphinx_rtd_theme) provided by [Read the Docs](https://readthedocs.org).