**Default controller:**

When we set up CI framework by default one controller will be provided as default controller i.e. Welcome.

If we make any controller as default controller it will execute automatically when we will type in browser url upto project name.

To change the default controller we have to follow below steps:

1. Go to routes.php
2. For $route[‘default\_controller’] we have to assign our controller name instead of Welcome.
3. Example: $route[‘default\_controller’]=’home\_controller’;

**Note:**

Default controller will execute automatically when we will type url upto project name.

Example: <http://localhost/project_name>

**404\_override:**

In CI wheneve will type invalid url, CI framework will display error code as 404.

If we want to override the design with other output we can.

Following steps we have to use override 404 page.

1. Create a new file under 404\_view.php and write code what we need to display.
2. Inside any controller create a method and load this view i.e 404\_view page

Example: public function page\_not\_found(){

$this->load->view(‘404\_view’);

}

1. Go to routes.php
2. Inside routes.php we have to assign value for $route[‘404\_override’] that value is controller name & method name

**Example:**

**$route[‘404\_override’] = ‘home\_controller/page\_not\_found’;**

**base\_url ( ) :**

1. Using this function we can get base url name which we configured inside config.php
2. This function return type is string
3. To access base url we have to load url helper then only we can access the function otherwise it will give fatal error.

**Example**:

**public function test(){**

**$this->load->helper(‘url’)**

**echo base\_url();**

**}**

**Autoloading:**

Whenever we are loading either helper or libraries into controllers or views and models there are two types of loading available in CI

1. Manual loading
2. Auto loading

**Manual loading:**

It is a process of loading libraries or helpers whenever we required. If we load with manual loading it will applicable to that area when we are doing manual loading.

**Auto loading:**

If we load any library or helper with auto loading we can access its features throughout the project and it will load one time.

**Note:**

When we have a requirement to use same code again and again better to go with autoloading.

**How to auto load helpers or libraries:**

In CI framework to develop auto loading mechanism CI provided a separate page with the name called autoload.php which is available inside config folders.

**Steps to auto load for helpers:**

1. Go to config folder
2. Open autoload.php
3. Look for the variable ie $autoload[‘helper’]
4. For this variable inside array we have to specify list of helpers with comma(,) separator

**Example:**

$autoload[‘helper’] = array(‘url’); //$autoload[‘helper’] is a configuration variable

**How to auto load libraries:**

Inside autoload.php there is a configuration with the name called $autoload[‘libraries’]. For this variable we have to specify list of library name with (,) separated.

**Constants:**

Constant is nothing but a fixed value and we cannot override constant variable values. Constants are similar to variable but variable allows overriding the value but variable allows overriding the value but constant doesn’t allow.

To declare constants variable we have to use a function called define().

In CI framework all constant variables we have to declare or define in constants.php

**Example:**

define(‘CSS\_PATH’,’assets/css/’);

**How to create a model:**

Model file creation is similar to controller creation only. When extending we have to use CI\_model

**Steps to execute model method:**

1. Load a model into required controller by using

$this->load->model() //assignment file name

1. Once model is loaded we can start model model methods by using following syntax i.e.

$this->model\_class\_name->method\_name();

**Example:**

$this->test\_method->f1();

Test\_model.php

<?php

Class Test\_model extends CI\_model{

Public function f1(){

echo “Welcome to model”;

}

}

?>

Test\_controller.php

<?php

Class Test\_controller extends CI\_controller{

Public function test(){

$this->load->model(‘test\_model’);

$this->test\_model->f1();

}

}

?>

**Requesting url :** <http://localhost/newfk/index.php/test_controller/test>

**Note:**

1.If we want to send data from controller to model we have to use arguments & parameters

2. if we want to send data from model data to controller we have to use return keyword.