BihongoPHP User Guide

**Installation**

* [Install manually](#installwithmanually)

**Tutorial: Create a simple blog site**

* [Create controller](#Createacontroller)
* [Set up database](#Setupdatabase)
* [Create model](#Createmodel)
* [Setup route](#Setuproute)
* [Conclusion](#Conclusion)

**General Tropics**

* Controllers
* Views
* Models
* Helpers
* Library
* Creating Library
* Security

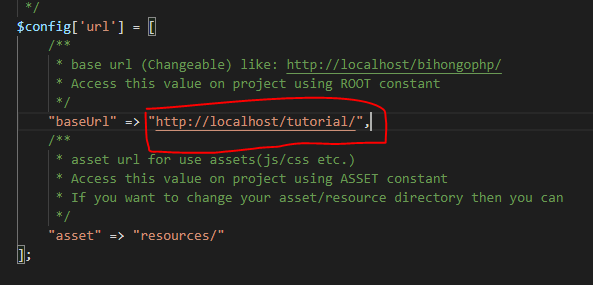
**Install manually**

To install manually go to bihongoPHP github page and download zip file from following address:

<https://github.com/SojebSikder/bihongophp/releases>

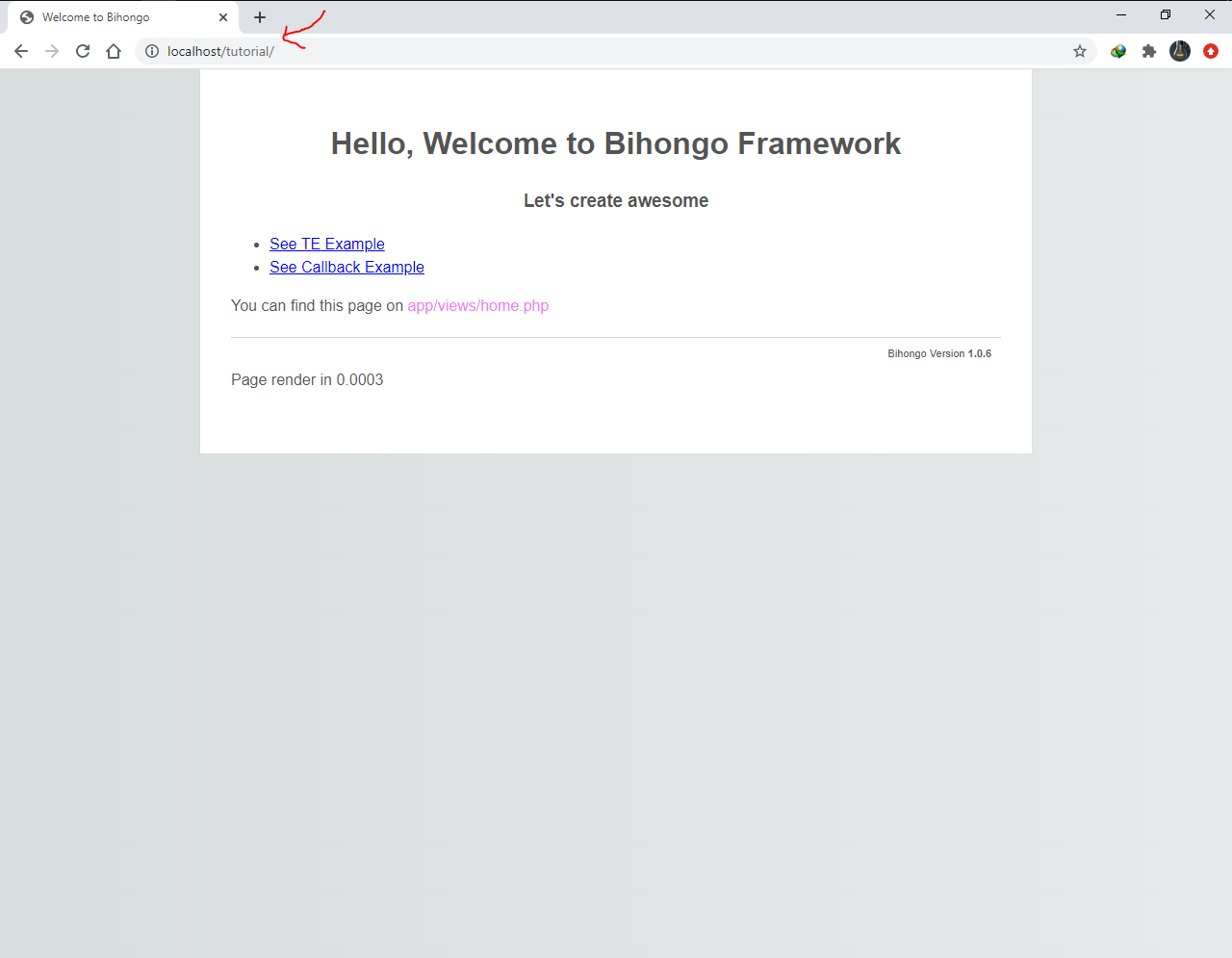
[DO NOT CLONE REPOSITORY. USE **RELEASE LATEST VERSION**]

* Extract files and put all files where you wish.
* After install bihongoPHP we have to configure first. Go to config->config.php.
* First change Base Url. Change baseUrl to your project url.



Now type your project on browser and hit enter. Mine is <http://localhost/tutorial>

We will see following page open



Now our BihongoPHP ready for creating project.

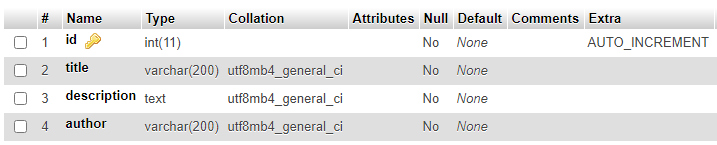
**Tutorial: Create a simple blog site**

If you are a ‘Learn by doing’ type then we will create a simple project using bihongoPHP.

We will create a simple blog app.

Set up database:

We need database for storing data. We will make simple database for our tutorial. Let’s use mysql database.

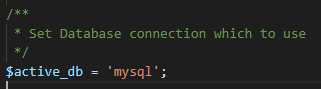


Our database have 4 column id, title, description and author. Now add some values:



After creating our database we have to need setup dbname on our config. Go to config/database.php.

Make sure $active\_db set to mysql.



Now change host, username, password, dbname as your requirements.



Create model:

After set up database now we will create model. Go to this folder app->models and create a file BlogModel.php.

Class name and file name should be same.

Syntax: NameModel

After creating file add following code:

<?php

class BlogModel extends Model{

public function \_\_construct(){

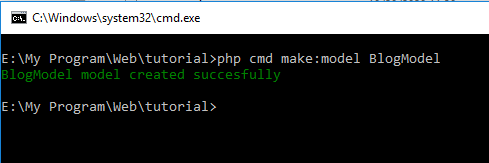
parent::\_\_construct();

}

}

Again you can create model using cmd console. Look at this command:

php cmd make:model BlogModel



**So you created your Model**. Now what we do, we have to fetch blog from our database. So add following code on BlogModel.php:

<?php

class BlogModel extends Model{

public function \_\_construct(){

parent::\_\_construct();

}

public function getBlog(){

return $this->db->select("SELECT \* FROM blog");

}

}

Here we see a method named getBlog. We $this->db property used to database crud operation.

Create a controller:

We will create controller. Go to this folder app->controllers. And create a file and name it whatever you want. Class name and file name should be same.

Syntax: NameController.php

After creating file and add following code:

<?php

class BlogController extends Controller{

public function \_\_construct(){

parent::\_\_construct();

}

public function home(){

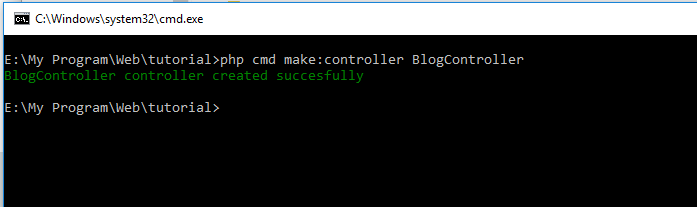
$this->load->view("home");

}

}

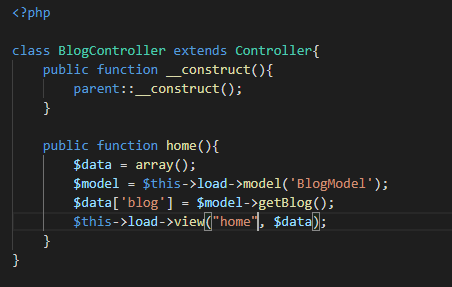
You can create controller easily with cmd console. Open Command prompt/Terminal from your project. And type this command:

php cmd make:controller BlogController

This will create BlogController.php file on app->controllers folder.

[**Note: Remember that Class name start with Capital Letter**.]

Now add some code into this



First we defined a array() inside $data variable.

$model variable for model. For loading model we have to use

$this->load->model(‘BlogModel’);

This create a object for BlogModel and store in $model variable.

In $data[‘blog’] we call getBlog() method.

And by this code

$this->load->view(‘home’, $data);

will render home page on browser. Second parameter is for transfer data, so we place $data variable for transfer $data on home page. Now create a home.php file on this folder app->views. Add following code (You can design your page as your requirements) :

<!DOCTYPE html>

<html>

<head>

    <base href="<?php echo ROOT ?>">

    <meta charset="UTF-8">

    <link rel="icon" href="<?php echo ICON ?>" type="image/png" sizes="16x16">

    <meta name="description" content="">

    <meta name="keywords" content="">

    <meta name="author" content="">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Welcome to <?php echo TITLE ?></title>

</head>

<body>

<?php foreach($blog as $getBlog): ?>

    <h3><?php echo $getBlog['title'] ?></h3>

    <p><?php echo $getBlog['description'] ?></p>

    <hr>

    <?php endforeach ?>

</body>

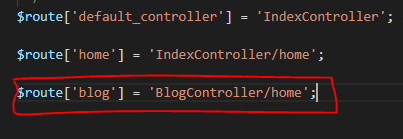
</html>

Here you can see many constant. These are predefined constant by BihongoPHP. Such as ROOT constant return app base url. Icon, TITLE were return icon url and title name.

Setup route:

Now we will setup route. Go to this folder config->routes.php

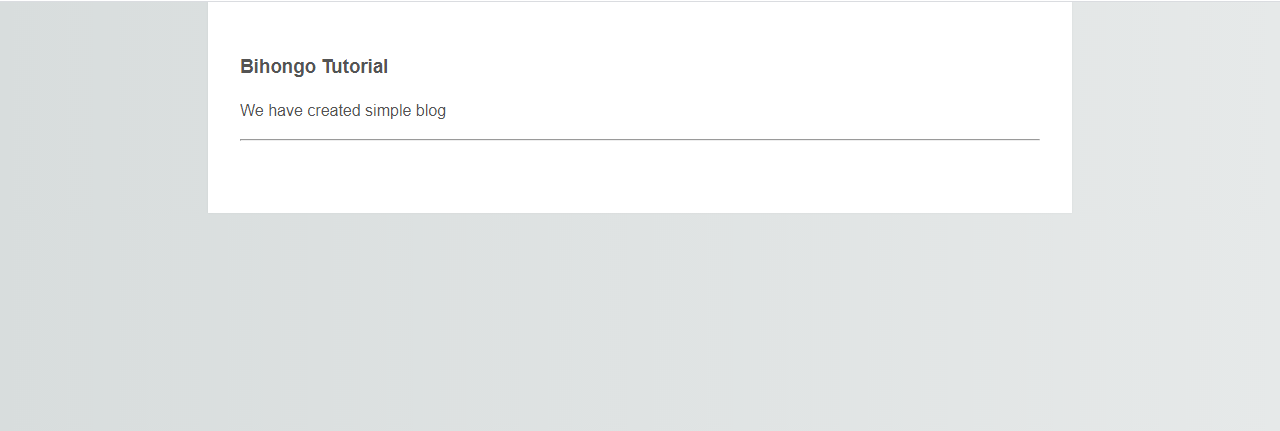
And add route name as your requirements. I will add blog.



We can see BlogController/home. BlogController is Controller name and home is method name which we want to execute. If we not put method name then home() method invoke automatically.

Now run your app on this address : <http://localhost/tutorial/blog>

You will see this result:



Conclusion:

We done a very simple project for learning purpose. Now you have a simple knowledge about BihongoPHP Framework. Now the question is what is next?

To know more about BihongoPHP go to general tropics page. Happy Coding ☺

**General Tropics**

Controllers:

Now we will learn about controller.

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

Controller is class file and used in url. BihongoPHP controller look like this:

<?php

class IndexController extends Controller{

    public function \_\_construct(){

        parent::\_\_construct();

    }

    public function home(){

        $this->load->view("home", $data);

    }

}

?>

Note: Class names must start with an uppercase letter.