Codeigniter Farsi

Producer by Hamidreza Pahlavan
Email: pahlavan@day.ir
www.DAY.ir

هدف از ایجاد این کتاب ارتقاء سطح علمی دوستان است و هیچ تضمینی راجع به مناسب بودن محتویات آن وجود ندارد.

از خوانندگان محترم تقاضا می شود با ارسال نظرات و پیشنهادات ، ما را از اشکالات احتمالی این کتاب آگاه سازند

Email: Pahlavan@day.ir

Website: www.day.ir

با تشکر: حمیدرضا پهلوان

15 فروردین 1389

Codeigniter چیست؟

Codeigniter فریم ورکی تحت زبان سرور ساید پی اچ پی است که به صورت مجانی در اختیار کاربرانش قرار گرفته است.

چگونه Codeigniter را دریافت کنیم؟

شما می توانید با نوشتن آدرس زیر در مرورگر خود و از طریق اینترنت آن را دریافت کنید.

Http://www.codeigniter.com

به طور پیش فرض آدرس های URL در Codeigniter برای انسان ها و موتورهای جستجو گر بهینه شده است.

و اساس آن بر اساس رویکرد مستقیم است.

مثل:

Example.com/news/article/my_article

Codeigniter یک فریم ورک بر اساس معماری MVC و یا (Model – View - Controller) است در ادامه با این مفهوم بیشتر آشنا خواهید شد.

بخش URL :

بخش هایی در URL وجود دارند که به ترتیب هر یک را توضیح خواهیم داد اما ابتدا به مثال فوق توجه کنید.

Example.com/class/function/id

- قسمت اول نشان دهنده کلاس کنترلر می باشد.
- قسمت دوم نشان دهنده تابع می باشد که درون کلاس وجود دارد.
- قسمت سوم و یا هر گونه قطعات دیگر نشان دهنده متغیر تابع و یا همان پارامتر تابع است.

URL Class:

این تابع به شما امکان می دهد که اطلاعات خود را از طریق این آدرس بازیابی کنید.

توجه: این کلاس به طور خودکار ازطرف برنامه پشتیبانی می شود.

پاک کردن فایل index.php :

به طور پیش فرض صفحه index.php URL اصلی شما قرار گرفته است مانند مثال ریز:

example.com/**index.php**/news/article/my_article

شما می توانید به وسیله سند htaccess. بر روی فضای وب خود این فایل را به صورت بسیار ساده ای حذف کنید.

```
RewriteEngine on
RewriteCond $1 !^(index\.php|images|robots\.txt)
RewriteRule ^(.*)$ /index.php/$1 [L]
```

اضافه کردن پسوند برای URL :

سند config.php در مسیر config/config.php را باز کنید ، شما می توانید پسوند ویژه ای برای صفحاتی که توسط Codeigniter تولید می شود بر گزینید.

برای مثال اگر آدرس URL مورد نظر این مسیر باشد:

example.com/index.php/products/view/shoes

به این صورت تبدیل می شود.

example.com/index.php/products/view/shoes.html

مانند زمانی که صفحه ای با فرمت HTML ساخته اید.

فعال کردن دنباله پرس وجو:

در اغلب موارد شاید شما بخواهید از دنباله های پرس و جو در برنامه های خود استفاده کنید. Codeigniter این امر را به صورت اختیاری برای شما قرار داده است. شما می توانید سند Config.php را باز کنید سپس عبارت

```
$config['enable_query_strings'] = FALSE;
$config['controller_trigger'] = 'c';
$config['function_trigger'] = 'm';
```

را جستجو کنید سپس مقدار False را به True تبدیل نمایید و سند را ذخیره کنید.

اگر شما دنباله پرس وجو را تغییر دهید این پرسه فعال می شود. از حالا به بعد آدرس شما به این شکل خواهد بود

index.php?c=controller&m=method

Controller (کنترلر)

کنترلر چیست؟

کنترلر یک کلاس از نوع ساده است که به این شکل نام گزاری شده است که می تواند ارتباط دهنده با URL باشد.

به این URL توجه کنید:

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

در مثال بالا Codeigniter سعی می کند تا کنترلر مورد نظر را با نام blog.php پیدا کنید و آن را فراخانی کند.

هنگامی مه نام کنترلر مشابه با اولین قسمت از URL بود عکل فراخانی انجام می شود.

اجازه دهید با هم اولین برنامه را بنویسیم:

اجازه دهید با هم یک نمونه از کنترلر را با هم بسازیم تا بتوانیم در عمل این کار را ببینیم. قطعه کد زیر را کپی کنید

```
<?php
class Blog extends Controller {
        function index(){
                echo 'Hello World!';
        }
}
?>
```

و صفحه ای با نام blog.php بسازید سپس کد را درون این صفحه انتقال دهید و آن را با همان نام درون مسیر

application/controllers/

ذخیره کنید. حال از طریق آدرس مرورگر خود آدرس زیر را وارد نمایید و دکمه Enter را بفشارید:

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

توجه: نام کلاس باید با نام بزرگ شروع شود ، اگر شما تمامی مراحل را به درستی سپری کرده باشید باید عبارت Hello world! را درون صفحه مرورگر مشاهده کنید.

نکته قابل توجه دیگر این است که تمامی کلاس ها باید از کلاس Controller ارث بری کنند.

توابع (Function):

در مثال بالا تابع را index نام گزاری کردیم نام index باعث می شود در هنگامی که کلاس فراخانی شد به صورت خودکار تابعی که با نام index ساخته شده است نیز فراخانی شود.

این موضوع در بردارنده این مفهوم می باشد که دیگر نیازی به مقداردهی برای قسمت دوم آدرس URL نیست.

اجازه دهید تابع جدید به کلاس اضافه کنیم:

```
<?php
class Blog extends Controller {
	function index(){
		echo 'Hello World!';
	}
	function comments(){
		echo 'Look at this!';
	}
}
?>
```

حال طریقه آدرس دهی را به این شکل تغییر دهید.

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

اگر تمامی مراحل را به درستی انجام داده باشید با عبارت Look at this! را مشاهده کنید.

پاس دادن آرگومان به تابع (Passing URL segment to your function) :

اگر آدرس URL شما بیشتر از 2 قسمت داشته باشد به طور خودکار بقیه قسمت ها به عنوان آرگومان تابع در نظر گرفته می شود.

مانند مثال فوق:

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

قطعه کد:

```
<?php
class Products extends Controller {
   function shoes($sandals, $id)
   {
      echo $sandals;
      echo $id;
   }
}
?>
```

تعریف کنترلر پیش فرض:

Codeigniter این قابلیت را دارد که می تواند کنترلر پیش فرض در مواقعی که URL حاضر نیست بارگزاری نمایید.

مثل زمانی که URL شما نام Root آدرس سایت می باشد.

برای تعیین کنترلر پیش فرض سند routes.php را از مسیر

application/config/routes.php

باز نماییدو متغییر را تعیین کنید:

```
$route['default_controller'] = '**Blog**';
```

اگر شما در حال حاضر index.php را load کنید بدون تعیین قسمتی از URL واژه !Hello world را مشاهده خواهید کرد.

صدا زدن تابع های (Remapping) :

در نوشته های بالا دیدیم که دومین قسمت در URL صدا کننده این است که ما از کدام تابع می خواهیم استفاده کنیم.

اما Codeigniter این اجازه را به شما می دهد که بتوانید تابعی را که قصد استفاده از آن را دارید به کنترل خود در آورید و این کار توسط توابع Remapping انجام می شود.

```
function _remap()
{
```

```
    // Some code here...
}
```

مثال مربوط به این قسمت:

```
function _remap($method)
{
    if ($method == 'some_method')
    {
        $this->$method();
    }
    else
    {
        $this->default_method();
    }
}
```

پردازش خروجی:

Codeigniter دارای یک کلاس خروجی است که به نام _out که می تواند خروجی کلاس ها و توابع شما را برای نشان دادن در مرورگر بهینه کند.

شبه کد آن بدین شکل است.

```
function _output($output)
{
    echo $output;
}
```

گاهی اوقات شما بدین شکل می پسندید که تابعی به صورت جداگانه برای نمایش بر روی مرورگر ایجاد کنید و امنیت بیشتری را درون کد های خود اعمال کنید.

توجه: اگر شما برای نشان دادن خروجی تابع خود از این تابع استفاده می کنید باید همیشه از این تابع بهره ببرید.

این تابع یک آرگومان ورودی دریافت می کند و آن را بر روی صفحه مرورگر چاپ می کند.

توابع خصوصی (Private):

در برخی از موارد شما می خواهید تعدادی از توابع را از جلوی چشم عموم دور نمایید. راه حل این امر تابع خصوصی می باشد.

برای ساخت یک تابع خصوصی ابتدا یک آندرلاین به نام مورد نظر اضافه کنید البته باید به این نکته توجه داشته باشید که نام مورد نظر شما کلمه رزرو شده نباشد.

```
function _utility()
{
  // some code
}
```

حال با تایپ کردن نام این تابع دیگر خروجی آن را مشاهده نخواهید کرد.

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

کنترل و سازمان دهی کلاس ها درون یک زیر پوشه:

اگر شما می خواهید یک برنامه بزرگ و گسترده بنویسید بهتر این است که کلاس های خود را درون یک زیر پوشه جمع آوری کنید.

Codeigniter برای شما این امکان را فراهم آورده است.

به سادگی یک پوشه درون مسیر

application/controllers

ایجاد کنید و تمامی کلاس های خود را درون آن قرار دهید. زمانی که شما از این ویژگی استفاده می کنید اولین قسمت از آدرس URL شما باید نام این پوشه باشد. برای مثال بگزارید به شما بگوییم کلاس های خود را درون این پوشه بگزارید در این مسیر:

application/controllers/**products**/shoes.php

برای فراخانی آن بدین صورت لینک می کنیم:

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

کلاس های سازنده (Constructors):

اگر شما می خواهید از توابع سازنده درون کلاس های خود استفاده کنید باید از قطعه کد زیر درون کلاس خود به این شکل استفاده کنید:

```
parent::Controller();
```

نمونه کد در پی اچ پی ورژن 4:

```
<?php
class Blog extends Controller {

    function Blog()
```

```
    {
        parent::Controller();
    }
}
?>
```

نمونه کد در پی اچ پی ورژن 5:

```
<?php
class Blog extends Controller {

    function __construct()
    {
        parent::Controller();
    }
}
?>
```

مشاهدات (View) :

View و یا مشاهدات یک نمونه صفحه ساده وب است و یا حتی می تواند یک صفحه کد باشد که شامل , header footer . غیره باشد.

در حقیقت مشاهدات می توانند به طور انعطاف پذیر درون هم دیگر قرار گیرند.

توجه به این نکته ضروری است که شما هرگز نباید view ها رو به طور مستقیم فراخانی کنید بلکه فراخانی آنها باید از طریق کلاس صورت گیرد.

به یاد داشته باشید Codeigniter یک فریم ورک MVC است پس کلاس ها مانند پلیس کنترل ترافیک در آن عمل می کنند.

درست کردن صفحه مشاهده:

از یک ویرایش گر استفاده کنید و قطعه کد زیر را درون آن جایگزین نمایید و نام سند را blogview.php بگزارید و آن را درون پوشه view ذخیره کنید.

```
<html>
<head>
<title>My Blog</title>
</head>
<body>
```

```
        <h1>Welcome to my Blog!</h1>
</body>
</html>
```

application/views/

بارگزاری صفحه مشاهدات :

اگر شما می خواهید صفحه viewخود را به صورت صحیح فراخانی کنید باید این کار را از درون کلاس کنترلر به کمک قطعه کد زیر انجام دهید.

```
$this->load->view('name');
```

به جای name نام صفحه مورد نظر را درج کنید نیازی به گزاشتن پسوند php. نیست البته اگر از فرمتی به غیر از php استفاده می کنید باید پسوند آن را نیز ذکر کنید.

حال صفحه ای را که قبلا به نام blog.php در مسیر کنترلر ذخیره کرده بودیم باز کنید سپس قطعه کد زیر را در آن قرار دهید.

```
<?php
class Blog extends Controller {

        function index()
        {
                $this->load->view('blogview');
        }
}
?>
```

حال با تایپ آدرس زیر در آدرس مرورگر خود باید صفحه view را مشاهده کنید.

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

بارگزاری چندین صفحه View :

Codeigniter به صورت بسیار هوشمندانه ای به وسیله تابع

```
$this->load->view('name');
```

می تواند چندین صفحه view را نشان دهد. اگر شما نیاز دارید که هم زمان چندین صفحه view را با هم باز کنید می توانید از این سیستم استفاده کنید.

شاید شما بخواهید که صفحه مشاهده دارای یک هدر و یا یک فوتر و یا حتی یک منو باشد.

این عمل با قطعه کد زیر انجام می شود.

```
<?php

class Page extends Controller {

  function index()
  {
    $data['page_title'] = 'Your title';
    $this->load->view('header');
    $this->load->view('menu');
    $this->load->view('content', $data);
    $this->load->view('footer');
  }

}
?>
```

شما در مثال قبل توانستید به این نکته پی ببرید که چگونه اطلاعات پویا به صفحات اضافه کنید.

ذخیره کردن مشاهدات در یک پوشه:

صفحه مشاهدات شما برای سادگی بیشتر می توانند درون یک پوشه قرار گیرند ، اگر شما ترجیح می دهید این عمل را انجام دهید باید در هنگام فراخانی صفحات مشاهده نام پوشه را نیز ذکر کنید.

```
$this->load->view('folder_name/file_name');
```

اضافه کردن اطلاعات پویا به وسیله آرایه:

نمونه قطعه کد ساخته شده توسط آرایه:

```
$data = array(
            'title' => 'My Title',
            'heading' => 'My Heading',
            'message' => 'My Message'
        );

$this->load->view('blogview', $data);
```

نمونه قطعه کد ساخته شده توسط کلاس آبجکت:

```
$data = new Someclass();
$this->load->view('blogview', $data);
```

توجه : اگر شما از شی استفاده کنید ، متغییر کلاس های شما به عناصر آرایه تبدیل می شوند

اجازه دهید که با یک کلاس کنترلر کار کنیم ، ابتدا قطعه کد زیر را به فایل قبل اضافه کنید.

```
<?php
class Blog extends Controller {

	function index()
	{
		$data['title'] = "My Real Title";
		$data['heading'] = "My Real Heading";

		$this->load->view('blogview', $data);
	}
}
?>
```

کار تمام نشده است حال فایل blogview را در پوشه view باز کنید و آن را به صورت زیر تغییر دهید.

```
<html>
<head>
<title><?php echo $title;?></title>
</head>
<body>
	<h1><?php echo $heading;?></h1>
</body>
</html>
```

حال صفحه را بار گزاری کنید تغییرات را باید ببینید.

درست کردن چرخه (Creating loops) :

پاس دادن اطلاعات آرایه به متغییر های محدودی در اسناد view محدود نمی شود ، شما می توانید متغییر های بیشتری به آن پاس بدهید. که می تواند تولید چندین سطر نمایید

در هنگامی که شما اطلاعاتی را از بانک اطلاعاتی خود بیرون می کشید بهترین راه استفاده از آرایه ها ی چرخشی است.

به عنوان مثال:

```
<?php

class Blog extends Controller {


	function index()

	{

		$data['todo_list'] = array('Clean House', 'Call Mom', 'Run Errands');


		$data['title'] = "My Real Title";

		$data['heading'] = "My Real Heading";


		$this->load->view('blogview', $data);

	}

}

?>
```

حالا برای ساخت روش چرخشی فایل view خود را باز کنید و آن را به شکل زیر تغییر دهید:

```
<html>
<head>
<title><?php echo $title;?></title>
</head>
<body>
<h1><?php echo $heading;?></h1>
<h3>My Todo List</h3>
<ul>
<?php foreach($todo_list as $item):?>
<li><?php echo $item;?></li>
<?php endforeach;?>
</ul>
</body>
</html>
```

برگشت اطلاعات:

اگر سومین پارامتر تابع

```
$this->load->view('blogview');
```

را که مقدار boolean دارد را به TRUE تغییر دهید آن گاه دیگر خروجی بر روی صفحه مرورگر نخواهد بود بلکه تابع خروجی را در قالب یک رشته تنظیم می کند ، در صورت تنظیم نشدن پارامتر سوم مقدار FALSE به صورت پیش فرض قرار گرفته است.

```
$string = $this->load->view('myfile', '', true);
```

مدل یک رویکرد اختیاری است که برای کسانی در نظر گرفته شده است که می خواهند از مدل سنتی MVC فراتر روند.

مدل چیست؟

مدل کلاس های PHP ای هستند که برای کارکردن با بانک اطلاعاتی طراحی شده اند.

ممکن است یک مدل کلاس شامل توابعی برای کار با insert و یا update و یا حتی بازیابی اطلاعات وب سایت باشد.

```
class Blogmodel extends Model {

    var $title   = '';
    var $content = '';
    var $date    = '';

    function Blogmodel()
    {
        // Call the Model constructor
        parent::Model();
    }

    function get_last_ten_entries()
    {
        $query = $this->db->get('entries', 10);
        return $query->result();
    }

    function insert_entry()
    {
        $this->title   = $_POST['title']; // please read the below note
        $this->content = $_POST['content'];
        $this->date    = time();

        $this->db->insert('entries', $this);
    }

    function update_entry()
    {
        $this->title   = $_POST['title'];
        $this->content = $_POST['content'];
        $this->date    = time();
```

```
        $this->db->update('entries', $this, array('id' => $_POST['id']));
    }

}
```

تشریح مدل:

کلاس های مدل در مسیر :

application/models/

ذخیره می شوند ، شما حتی اگر مایل هستید می توانید از سازمان دهی پوشه های تو در تو نیز بهره ببرید.

قطعه کد زیر یک نمونه اولیه برای کلاس مدل است:

```
class Model_name extends Model {

    function Model_name()
    {
        parent::Model();
    }
}
```

توجه: به جای Model_name نام کلاس قرار می گیرد که باید با نام صفحه یکسان باشد ضمنا باید به خاطر داشته باشید که حرف اول نام کلاس با حروف بزرگ نوشته شود.

```
class User_model extends Model {

    function User_model()
    {
        parent::Model();
    }
}
```

مسیر فایل قرار داده شده :

application/models/**user_model.php**

بارگزاری مدل (Loading a Model) :

مدل های شما به طور معمول از طریق کلاس های کنترل فراخانی می شوند. برای بارگزاری مدل باید از تابع زیر استفاده کنید.

```
$this->load->model('Model_name');
```

اگر مدل شما داخل پوشه ای قرار داده شده است برای آدرس دهی آن باید آدرس پوشه را نیز ذکر کنید مانند:

application/models/blog/queries.php

```
$this->load->model('blog/queries');
```

پس از بارگزاری شما می توانید از نام شی به عنوان نام کلاس خود استفاده کنید:

```
$this->load->model('Model_name');

$this->Model_name->function();
```

اگر شما مایل هستید می توانید مدل خود را به نام شی های مختلف اختصاص بدهید و در موقع بارگزاری مدل ، مشخصه دوم (آرگومان دوم) تابع را تنظیم و یک نام برای آن انتخاب نمایید:

```
$this->load->model('Model_name', 'fubar');

$this->fubar->function();
```

ارتباط با بانک اطلاعاتی:

هنگامی که مدل شما بارگزاری شد به طور خودکار به پایگاه داده ارتباط داده نمی شوید ، گزینه های زیر برای ارتباط پیدا کردن هر چه بهتر به این موضوع است.

شما می توانید از مدل های استاندارد پایگاه داده در کلاس های کنترلر خود استفاده کنید.

شما می توانید در هنگام بارگزاری مدل خودبا تنظیم سومین پارامتر تابع به TRUE امکان اتصال خودکار به بانک اطلاعاتی را فراهم کنید که توسط تنظیمات پیش فرض Config پایگاه داده کار می کند.

شما می توانید این عمل را هم به شیوه دستی داخل کد برنامه خود و هم از طریق اسناد Config انجام دهید.

```
$config['hostname'] = "localhost";
$config['username'] = "myusername";
$config['password'] = "mypassword";
$config['database'] = "mydatabase";
$config['dbdriver'] = "mysql";
$config['dbprefix'] = "";
$config['pconnect'] = FALSE;
$config['db_debug'] = TRUE;
```

```
$this->load->model('Model_name', '', $config);
```

توابع کمکی:

همیاران (کمک) این نام بدین دلیل پیشنهاد شده است که به شما در اجرای وظایف کمک می کند.

هر سند کمک کنندهای یک سری مجموعه ساده از توابع و دسته بندی خاص است.

URL Helper : به شما در ایجاد پیوند های کمکی و یا کمک در ایجاد فرم کمک می کنند.

کمک کننده متن: قالب بندی متن های مختلف

کمک کننده کوکی: تنظیم و خواندن کوکی ها

کمک کننده پوشه: که به شما در زمینه تبادل بین پوشه های کمک می کند

بر خلاف اکثر سیستم های دیگر در Codeigniter ، کمک کننده ها به فرمت آبجکت ارینتد (شی گرا) نوشته نشده اند.

این توابع بسیار ساده اند و مانند توابع کار می کنند که هر تابع کمک کننده بدون وابستگی به دیگر توابع وظایف خاصی به عهده دارد.

Codeigniter به طور پیش فرض توابع کمک کننده را فراخانی نمی کند ، پس قدم اول برای کار با توابع کمک کننده فراخانی آنهاست.

پس از فراخانی ، آنها را می توانید در مقیاس های وسیع در کنترلرهای و یا حتی در view ها مورد استفاده قرار دهید.

مسیر قرارگیری کمک کننده ها درون مسیر System/helpers قرار دارند.

بارگزاری کمک کننده ها (Loading a Helper) :

بارگزاری کمک کننده ها در نهایت سادگی امکان پذیر است ، برای این منظور از قطعه کد زیر استفاده می کنیم.

```
$this->load->helper('name');
```

شما می توانید به جای name نام کمک کننده مورد نظر خود را بنویسید.

توجه : برای ذکر نام کمک کننده نیازی به نوشتن پسوند فایل نیست.

مثلا: برای بارگزاری کمک کننده URL بدین شیوه عمل می نماییم :

```
$this->load->helper('url');
```

توابع کمک کننده می توانند در هر جایی از درون کلاس یا تابع کنترلر و یا حتی از درون view ها فراخانی شوند. اما فراخانی توابع کمک کننده درون صفحات view کار جالبی نیست و ما آن را توصیه نمی کنیم.

شما می توانی برای کمک کنده های یک تابع سازنده بسازید تا در صورتی که کلاس فراخانی شد توابع کمک کننده نیز فراخانی شوند.

توجه: توابع کمک کننده هیچ مقداری را باز نمی گرداند پس سعی نکنید به آن متغیری اختصاص دهید.

بارگزاری چندین کمک کننده (**Loading Multiple Helpers**) :

اگر شما به بیشتر از یک کمک کننده نیاز دارید می توانید به روش زیر این کار را انجام دهید:

```
$this->load->helper( array('helper1', 'helper2', 'helper3') );
```

بارگزاری خودکار کمک کننده ها (**Auto-loading Helpers**) :

شما می توانید با تنظیم کردن مقدار در مسیر :

application/config/autoload.php

این امکان را به وجود آورید که کمک کننده ها به طور خودکار بارگزاری شوند.

استفاده کردن از کمک کننده ها (**Using a Helper**) :

پس از آنکه شما کمک کننده ای را بارگزاری کردید فایل کمک کننده حاوی توابعی است که قصد کمک به شما را دارند ، شما باید آنها را مثل توابع استاندارد PHP صدا بزنید

به عنوان مثال : برای ساخت یک پیوند از تابع anchor() استفاده می کنیم و آن را داخل یک سند view ذخیره می کنیم.

```
<?php echo anchor('blog/comments', 'Click Here');?>
```

توجه: حتما باید داخل سند config از مسیر config/config.php در قسمت base_url() آدرس خود را جایگزین کنید.

مثل نمونه:

```
$config['base_url'] = "http://127.0.0.1/CodeIgniter_1.7.1/";
```

پلاگین ها (Plugins) :

پلاگین ها تقریبا برای کمک به شما به وجود آمده اند تفاوت اصلی پلاگین ها با کمک کننده ها در این است که پلاگین ها معمولا تابع های کوچک و تک را توسعه می دهند در صورتی که کمک کننده ها معمولا توابع را جمع آوری می کنند.

از طرف دیگر کمک کننده ها شامل قسمت اصلی و هسته کد هستند اما پلاگین ها معمولا توسط جوامع توسعه پیدا می کنند.

پلاگین ها درون شاخه system/plugins قرار دارند ، شما حتی می توانیدیک پوشه به نام Plugins ساخته و آن را درون پوشه application قرار دهید.

Codeigniter ابتدا درون پوشه system/application/plugins را نگاه می کند اگر پلاگین مورد نظر را پیدا نکرد جستجو برای پیدا کردن فایل مربوطه را درون این شاخه به انجام می رساند system/plugins

بارگزاری پلاگین ها (**Loading a Plugin**) :

بارگزاری پلاگین ها در نهایت سادگی امکان پذیر است. به کمک قطعه کد زیر این کار انجام می شود.

```
$this->load->plugin('name');
```

هر کجا که name قرار دارد باید نام پلاگین قرار گیرد و این امکان نیز برای شما در نظر گرفته شده است که می توانید نام کامل پلاگین را ذکر نکنید و یا حتی از به کار بردن پسوند php. خودداری کنید.

برای مثال ما می خواهیم پلاگین تصاویر امنیتی (Captcha) را بارگزاری کنیم ، نام صفحه پلاگین captcha_pi.php است. با قطعه کد زیر این کار را انجام می دهیم.

$this->load->plugin('**captcha**');

پلاگین ها نیز مانند توابع کمک کننده در هر جایی می توانند فراخانی شوند.

بارگزاری چندین پلاگین (**Loading Multiple Plugins**) :

گاهی اوقات شما احتیاج دارید که چندین پلاگین را با هم فراخانی کنید می توانید این کار را به روش زیر انجام دهید.

$this->load->plugin(**array('plugin1**', '**plugin2**', '**plugin3')**);

بارگزاری به صورت خودکار (**Auto-loading Plugins**) :

شما می توانید این پروسه را با باز کردن سند autoload.php از شاخه زیر و تنظیم آن انجام دهید:

application/config/autoload.php

استفاده از پلاگین ها (**Using a Plugin**) :

مواردی که شما برای فراخوانی پلاگین ها احتیاج دارید 2 مورد است:

1- نام پلاگین:
2- مسیر پلاگین:

استفاده کردن از توابع کتابخانه (**Using CodeIgniter Libraries**) :

تمامی توابع موجود در codeigniter درون شاخه system/libraries قرار دارند. در اغلب موارد برای استفاده از یکی از این کلاس ها که دارای تعدادی تابع هستندبه روش زیر عمل می کنیم.

$this->load->library('**class name**');

هر جا که class name نوشته شده باشد نام کتابخانه مورد نظر قرار می گیرد.

برای مثال یک کلاس معتبر ساز را فراخوانی کی کنیم:

$this->load->library('**validation**');

پس از فراخانی شما می توانید به همان روشی که در ادامه توضیح داده خواهد شد عمل نمایید.

مسیر یابی URL:

به طور معمول یک رابطه یک به یک بین رشته URL و کلاس متناظر آن وجود دارد که از الگوی زیر بهره می برد.

example.com/**class**/**function**/**id**/

با این حال شاید شما بخواهید کلاس های مختلفی را با این شیوه فراخانی کنید به جای آدرس دهی متناظر برای هر کدام از آدرس ها ، اجازه هید به شما بگوییم که می خواهیم با استفاده از چنین آدرس دهی کلاس ها را فراخانی کنیم.

به طور معمول قسمت دوم URL بازگرداننده نام تابع است اما در این مثال به جای Codeigniter Productid به شما اجازه می دهد بر این امر غلبه کنید و URL ها رو Remap کنید.

example.com/product/1/
example.com/product/2/
example.com/product/3/
example.com/product/4/

تنظیم کردن قوانین مسیر یابی به دست خود (**Setting your own routing rules**) :

روش های مسیر یابی در مسیر زیر قرار دارند :

application/config/routes.php

درون این فایل تعدادی آرایه که با $route مشاهده می کنید تعریف شده اند که به شما اجازه می دهند که بر اساس ضوابط مشخص خودتان مسیریابی را تعریف کنید.

مسیر یابی شامل 2 قسمت می شود:

- wildcards

- Regular expressions

: Wildcards

Wildcards معمولی به شکل مثال زیر معمولا مسیریابی می شوند.

```
$route['product/:num'] = "catalog/product_lookup";
```

در این مسیریابی ، مقدار کلید آرایه باید با آدرس URL یکی باشد ، در حالی که مقدار آرایه با محتوای مقصد آن را دوباره مسیر یابی کند.

در مثال بالا اگر کلمه product و catalog در قسمت اول و دوم URL پیدا شود از این شیوه استفاده خواهد کرد.

شما می توانید مقدار دقیق را یکی کنید و یا از 2 نوع Wildcards استفاده کنید.

```
:num  //only number

:any // any character
```

در زیر تعدادی مثال برای شما می آوریم:

```
$route['journals'] = "blogs";
```

اگر آدرس حاوی کلمه " journals " در اولین قسمت آدرس باشد به کلاس remapp blog می شوید.

```
$route['blog/joe'] = "blogs/users/34";
```

با نوشتن blog/joe در مقدار کلید آرایه بازدن چنین آدرسی controller به آرس blogs/users/34 انتقال داده می شود.

```
$route['product/:any'] = "catalog/product_lookup";
```

بعد از کلاس catalog هر مقداری می تواند قرار گیرد

```
$route['product/(:num)'] = "catalog/product_lookup_by_id/$1";
```

عبارات منظم (**Regular Expressions**) :

اگر شما ترجیح می دهید از عبارت منظم برای تعریف کردن آدرس URL استفاده کنید.

اجازه دارید از هر عبارت منظم درستی برای این امر استفاده کنید.

```
$route['products/([a-z]+)/(\d+)'] = "$1/id_$2";
```

مثل مثال بالا:

products/shirts/123

مسیر های رزرو شده (**Reserved Routes**) :

از عبارت زیر می توانید برای هنگامی که صفحه ای را load کی کنید و آن صفحه شامل اطلاعاتی نمی باشید استفاده کنید مانند کلاس کنترلر welcome

```
$route['default_controller'] = 'welcome';
```

هندل کردن خطا ها (**Error Handling**) :

Codeigniter به شما اجازه می دهد که یک پیام خطا برای برنامه خود بسازید به کمک توابعی که در پایین توضیح داده می شود.

علاوه بر این codeigniter به شما اجازه می دهد که پیام های خطا و اشکال زدایی را به عنوان فایل متنی ذخیره کنید و یا به شما نشان دهد.

:Show_error(` message ')

این تابع پیغام های خطایی را که تولید می شوند را در قالب یک الگوی خاص نشان میدهد.

show_error('message' [, int $status_code= 500]);

application/errors/error_general.php

:show_404('page')

این تابع خطای صفحه 404 را در قالب یک الگوی خاص نشان می دهد. این تابع انتظار دارد رشته پاس داده شده به آن آدرس فایلی باشد که پیدا نشده است.

توجه: codeigniter به طور خودکار اگر صفحه ای را پیدا نکند خطای 404 را پیدا می کند.

:log_message('level', 'message')

این تابع به شما اجازه می دهد درون فایل log پیامی بنویسید.

که دارای 2 آرگومان ورودی می باشد:

1- نوع پیام که شامل (اشکال زدایی ، خطا ، اطلاعات) هست را نشان می دهد
2- خود متن پیام

```
if ($some_var == "")
{
   log_message('error', 'Some variable did not contain a value.');
}
else
```

```
{
   log_message('debug', 'Some variable was correctly set');
}

log_message('info', 'The purpose of some variable is to provide some value.');
```

3 نوع پیام وجود دارد:

1- پیام خطا: خطا های واقعی ، که شامل خطا های php هستند.
2- پیام های اشکال زدایی: این پیام ، پیامی است که به شما در اشکال زدایی کمک می کند. برای مثال اگر کلاس خود را ارزیابی کنید ، می توانید به اطلاعات اشکال زدایی دسترسی پیدا کنید.
3- پیغام های اطلاعاتی : پیام های با الویت پایین تر هستند. مثل دادن اطلاعات مربوط به چندید فرآیند.

Codeigniter به طور پیش فرض تعدادی پیام خطا برای شما می سازد اما شاید شما بخواهید خطا های دست ساز به وسیله خود را در برنامه داشته باشید به کمک توابع گفته شده در بالا می توانید این کار را انجام دهید.

کش کردن صفحه (Web Page Caching) :

Codeigniter به شما اجازه می دهد که صفحه مورد نظر خود را در رسیدن به بیشترین عملکرد کش کنید

کش شدن می تواند برای هر صفحه و یا بر اساس هر صفحه فعال شود. و شما می توانید برای تجدید شدن صفحه زمان کش را تنظیم کنید.

هر زمانی که صفحه برای بار بعدی بارگزاری شد ، کش فایل بازیابی می شود و بنابر درخواست به مرورگر ارسال می شود اگر آن صفحه منقضی شده باشد ، قبل از اینکه به مرورگر ارسال شود کش حذف شده و دوباره لود می شود.

توجه: مبنا بطور پیش فرض برای کش نشدن گذاشته شده است و دلیل آن نیز بدین منظور است که شما تغییرات را در صفحه خود ببینید.

فعال کردن کش کردن (**Enabling Caching**):

برای فعال کردن کش شدن ، هر جا که خواستید در کد کلاس خود در کنترلرها این تگ را قرار دهید.

```
$this->output->cache(n);
```

هر کجا که n قرار دارد باید عددی برای میزان دقیقه کش شدن تنظیم شود.

محل قرار گیری آن محدودیتی ندارد و در هر جایی که از لحاظ منطقی می پندارید درست است می توانید این قطعه کد را قرار دهید.

توجه: قبل از گذاشتن قطعه کد باید به سطح دسترسی پوشه system/cache توجه داشته باشید.

مدیریت برنامه های خود (**Managing your Applications**):

به صورت پیش فرض ، فرض شده است ، که اگر شما قصد دارید از Codeigniter برای مدیریت کردن برنامه های خود استفاده کنید.

شما باید یکپوشه درون مسیر system/application بسازید ، اگر چه می توانید مجموعه های متعددی از برنامه ها را روی یک codeigniter تقسیم کنید یا حتی مکان پوشه application را تغییر دهید.

تعویض کردن نام پوشه (**Relocating your Application Folder**) :

اگر شما این چنین می پسندید که نام پوشه application را تعویض نمایید باید صفحه index.php را در root اصلی بازکرده و در مکان $application_folder نام پوشه مورد نظر را درج کنید:

```
$application_folder = "/Path/to/your/application";
```

جابجایی پوشه application (**Relocating your Application Folder**) :

Codeigniter برای شما این امر را که بتوانید پوشه application را به مکان دیگری در پوشه system منتقل کنید میسر ساخته است.

برای انجام این عمل سند index.php را مسیر اصلی (root) باز کرده و مقدار :

```
$application_folder = "/Path/to/your/application";
```

را تنظیم نمایید.

راه اندازی برنامه های متعدد (**Running Multiple Applications with one CodeIgniter Installation**) :

اگر شما دوست دارید دستورات codeigniter را برای برنامه های متعدد خود به اشتراک بگزارید و فقط از یک پوشه application بهره ببرید ، می توانید در پوشه application زیر پوشه ای با نام دلخواه ایجاد کنید.

برای مثال : اجازه بدهید با یکدیگر 2 زیر پوشه داخل فولدر application به نام های foo و bar درست کنیم ، که ساختار آن به شکل زیر است.

```
system/application/foo/
system/application/foo/config/
system/application/foo/controllers/
```

system/application/**foo**/errors/
system/application/**foo**/libraries/
system/application/**foo**/models/
system/application/**foo**/views/
system/application/**bar**/
system/application/**bar**/config/
system/application/**bar**/controllers/
system/application/**bar**/errors/
system/application/**bar**/libraries/
system/application/**bar**/models/
system/application/**bar**/views/

برای اینکه بتوانید به طور منظم از برنامه مورد نظر خود استفاده کنید فایل index.php را باز کرده و در متغیر

$application_folder مقدار مورد نظر را جایگزین کنید به عنوان مثال:

```
$application_folder = "application/foo";
```

هر برنامه شما باید شامل یک index.php باشد برای اینکه فایل مورد نظر خود را لود کنید باید مقدار index.php را تغییر دهید.

متناسب کردن کدهای PHPبرای دیدن سند (**Alternate PHP Syntax for View Files**) :

اگر شما نمی پسندید که از template engin ، codeigniter استفاده کنید می توانید از کد های خالص PHP برای نشان دادن اطلاعات خود استفاده کنید.

برای کوچک کردن کدهای PHP تان و یا قابل فهم تر کردن آنها می توانید از alternative php و یا از کد های کوچک استفاده نمایید.

اگر شما با این شیوه کد نویسی آشنایی ندارید در ادامه آن را فرا می گیرید. نمونه هایی از آن مانند این است که شما در هنگام کد نویسی از پرانتز درون کد خود استفاده نمی کنید و یا حتی برای نوشتن از کلمه echo چشم پوشی می نمایید.

پشتیبانی خودکار از تگ کوچک (**Automatic Short Tag Support**) :

توجه: اگر شما توصیفی را بر روی سرور یافتید که عمل نکرد باید short_tag را از درون سند phpini روشن کنید ، برای این منظور می توانید از شاخه زیر این کار را انجام دهید.

config/config.php

:Alternative Echos

به طور معمول در این روش از echo و print به این شکل استفاده می نماییم.

```
<?php echo $variable; ?>
```

به صورت کد کوچک:

```
<?=$variable?>
```

ساختمان کنترل (**Alternative Control Structures**) :

سازه های کنترلی if ، for ، foreach و while ساده شده آن به شکل مثال زیر است.

```
<ul>
<?php foreach($todo as $item): ?>
<li><?=$item?></li>
<?php endforeach; ?>
</ul>
```

توجه: در این کد از پرانتز باز کردن حلقه و انتهای حلقه استفاده نشده است و به جای آن از عبارت endforeach برای پرانتز آخر حلقه استفاده شده است.

ساختار اطلاعات مشابه کوچک شده syntax زیر است.

endif, **endfor**, **endforeach**, and **endwhile**

همچنین توجه کنید که به جای سیمی کالون بعد از هر ساختار (به جز آخرین) از علامت : استفاده شده است.

اینجا یک مثال دیگر داریم که از if , else if , else و دونقطه استفاده شده است.

```
<?php if ($username == 'sally'): ?>
  <h3>Hi Sally</h3>
<?php elseif ($username == 'joe'): ?>
  <h3>Hi Joe</h3>
<?php else: ?>
  <h3>Hi unknown user</h3>
```

<?php endif; ?>

امنیت (Security) :

اطلاعات این بخش بهترین بهترین تمرین در خصوص امنیت وب و جزئیات و ویژگی های امنیت داخلی codeigniter است.

امنیت در URL ها (**URI Security**) :

Codeigniter تا حدودی اجازه دسترسی به رشته های URL را می دهد البته تا جایی که موجب تخریب داده های برنامه نشود.

داده هایی که شامل عبارات زیر هستند.

- آلفا – متن عددی
- علامت مد : ~
- نقطه : .
- دونقطه : :
- آندرسکور : _
- دش : -

GET, POST, and COOKIE Data :

چون codeigniter از قسمت های مختلف URL به جای دنباله های پرس و جو برای گرفتن اطلاعات استفاده می کند ، (در صورتی که شما این عمل را در سند config فعال نکرده باشید) متغیر GET را نمی پذیرد.

Register_globals :

در مدتی که سیستم آغاز به کار می کند به جز متغیر های $_post ، و متغیر $_cookie بقیه متغیر ها تنظیم نشده اند.

اگر می خواهید تمامی تنظیمات را در مسیر یابی باطل کنید باید متغیر را غیر فعال کنید:

register_globals = off.

متغیر magic_quotes_runtime به طور پیش فرز در هنگام شروع به کار سیستم غیر فعال می باشد. به صورتی که به شما اجازه حذف کردن اسلش زمانی که اطلاعاتی را از پایگاه داده بازیابی می کنید نمی دهد.

تمرین بسیار عالی:

قبل از پذیرش تعدادی اطلاعات در برنامه خود ، که آیا اطلاعات از submit شدن اطلاعات post داده می شود یا اطلاعات کوکی یا اطلاعات URL و یا حتی اطلاعات XML_RPC یا هر اطلاعات دیگر از سرور که برای شما اجرا می شود

باید به سه نکته توجه داشته باشید.

1- اگر ممکن است اطلاعات آلوده باشند آن ها را فیلتر کنید.
2- در صورتی که اطمینان از درستی اطلاعات ندارید آنها را معتبر کنید.
3- قبل از ارسال اطلاعات به پایگاه داده آنها را Escape کنید.

Codeigniter برای کمک به پردازش این مسئله در ادامه توابعی را معرفی می کند.

XSS Filtering :

Codeigniter به منزله روبرو شدن با فیلترکردن اسکریپت بنا شده است. این عمل فیلترینگ برای انجام ، معمولا از فنون خاصی در جهت جلوگیری از جاسازی و حفاظت از کد شما در برابر کدهای مخرب جاوا اسکریپت یا کدهای دیگر از قبیل دزدیدن کوکی یا چیز های مخرب دیگر بنا شده است.

معتبر کردن اطلاعات (**Validate the data**) :

Codeigniter دارای کلاس های خاصی برای معتبر کردن فرم می باشد ، که به شما کمک می کند اطلاعات فرم ها را معتبر کنید ، از قبیل فیلتر کردن و درست کردن اطلاعات شما

درست کردن اطلاعات قبل از ارسال به پایگاه داده:

هرگز اطلاعاتی را که هنوز escape نکرده اید به پایگاه داده وارد نکنید.

شیوه و چهار چوب اصلی (**General Style and Syntax**) :

- فرمت سندها
- طریقه بستن تگ ها
- نامگذاری کلاس ها و متد ها
- نامگذاری متغیرها
- نام متغیرها
- کامنت

- ثابت ها

- درست ، غلط و پوچ

- عملگر های منطقی

- مقایسه مقدارهای بازگشتی

- غلط یابی کد

- فضای خالی در سند ها

- سازگاری

- کلاس و نام های فایل ها با استفاده از کلمات عمومی

- نام جداول پایگاه داده

- یک فایل در هر کلاس

- فضای خالی

- شکستن خط

- تورفتگی کدها

- براکت و فاصله

- ترجمه متن ها در کنترل پنل

- متغیر ها و متد های خصوصی

- خطا ها در php

- بازکردن تگ های کوچک

- یک دستورالعمل در هر خط

- رشته

- کوری اس کی یو ال

- پیش فرض آرگومان توابع

- تداخل برچسب پارامتر ها

فرمت سند ها:

سند ها حتما باید با فرمت UTF-8 ذخیره شوند.

طریقه بستن تگ ها:

تگ بستن کد در PHP <? است که برای جدا کردن کدها از یکدیگر استفاده می شود.

با این حال ، اگر شما از این شیوه استفاده می کنید مقدار فضای خالی در زیر تگ بسته شده خواهیم داشت که اصلا کار درستی نیست .

قسمت های خالی در فضا های PHP صفحات را بعدا دچار مشکل می کند ، به همین دلیل تمام اسناد PHP باید از بستن تگ خودداری کنند و در عوض به جای آن از پیغام اتمام فایل استفاده کنند و مسیر واقعی سند ذخیره را در انتهای صفحه درج نمایند.

این شیوه به شما اجازه می دهد که فایل های ناقص و تکمیل شده را شناسایی کنید.

INCORRECT:

```
<?php echo "Here's my code!"; ?>
```

CORRECT:

```
<?php echo "Here's my code!";
/* End of file myfile.php */
/* Location: ./system/modules/mymodule/myfile.php */
```

نامگذاری کلاس ها و متد ها:

حرف اول کلاس باید به شکل بزرگ نوشته شود و کلمات چندگانه باید به وسیله_ از یکدیگر جدا شوند ، نباید کلمات به یکدیگر بچسبند.

تمام مدل های دیگر باید به طور کامل با حروف کوچک نامگذاری شوند و باید به خوبی و وضوح نشان دهنده کار آن باشد ، ترجیحا از یک فعل تشکیل شده باشد .

توصیه ما این است که نام انتخابی نه طولانی باشد و نه کوتاه

INCORRECT: class superclass class SuperClass

CORRECT: class Super_class

توابع سازنده باید با نام کلاس یکی باشند:

```
class Super_class {
        function Super_class()
        {
        }
}
```

مثالی از نام های نا مناسب و روشهای نامگذاری غلط:

آندراسکور ندارد و توضیح نامناسب:

```
function fileproperties()          // not descriptive and needs underscore separator
```

کلمات به یکدیگر چسبیده اند و توضیح مناسب ندارد:

```
function fileProperties()          // not descriptive and uses CamelCase
```

تا حدی مناسب است اما آندراسکور ندارد:

```
function getfileproperties()               // Better!  But still missing underscore separator
```

از کلمات چسبیده به هم استفاده شده است:

```
function getFileProperties()               // uses CamelCase
```

بیش از حد طولانی است:

```
function get_the_file_properties_from_the_file() // wordy
```

درست:

```
function get_file_properties()     // descriptive, underscore separator, and all lowercase letters
```

نامگذاری متغیرها:

دستورالعمل ها برای نامگذاری متغیر ها بسیار مشابه نامگذاری کلاس ها است. یعنی متغیر های باید شامل فقط کارکترهای با حروف کوچک باشند .

از آندراسکور برای جدا کردن نام ها استفاده کنند و نام آنها قابل فهم و در حد مطلوب باشد.

از متغیر های بسیار کوچک و یا بدون تفهیم فقط برای حلقه ها استفاده کنید.

INCORRECT:

$j = 'foo'; // single letter variables should only be used in for() loops

$Str // contains uppercase letters

$bufferedText // uses CamelCasing, and could be shortened without losing semantic meaning

$groupid // multiple words, needs underscore separator

$name_of_last_city_used // too long

CORRECT:

for ($j = 0; $j < 10; $j++)

$str

$buffer

$group_id

$last_city

کامنت:

در حالت عمومی ، کد ها باید به حالت بارخیزی (یعنی کم حجم) کامنت شوند.

این نه تنها کمک می کند تا توصیف جریان و قصد از ساخت کد آسانتر شود بلکه این امر مختص برنامه نویسان با تجربه است.

می توانید تصور کنید اگر بعد از مدت زمانی دوباره به کد خود نگاه انداختید می تواند چه مقدار گرانبها باشد.

توجه: هیچ فرمت خواصی برای نوشتن کامنت احتیاج نیست.

```
/**
 * Super Class
 *
 * @package     Package Name
 * @subpackage          Subpackage
 * @category    Category
 * @author      Author Name
 * @linkhttp://example.com
```

```
 */
class Super_class {
/**
 * Encodes string for use in XML
 *
 * @access      public
 * @param       string
 * @return      string
 */
function xml_encode($str)
```

از // برای کامنت کردن استفاده کنید.

از خالی گزاشتن خط های خالی بین کامنت های زیاد خودداری کنید.

```
// break up the string by newlines
$parts = explode("\n", $str);
// A longer comment that needs to give greater detail on what is
// occurring and why can use multiple single-line comments.  Try to
// keep the width reasonable, around 70 characters is the easiest to
// read.  Don't hesitate to link to permanent external resources
// that may provide greater detail:
//
// http://example.com/information_about_something/in_particular/
$parts = $this->foo($parts);
```

ثابت ها:

ثابت ها از تمام قواعدی که برای نامگذاری متغیر ها نام بردیم استفاده می کنند البته به غیر از ثابت هایی که حتما باید با نام بزرگ نوشته شوند.

INCORRECT:

```
myConstant       // missing underscore separator and not fully uppercase

N                // no single-letter constants

S_C_VER                  // not descriptive

$str = str_replace('{foo}', 'bar', $str);     // should use LD and RD constants
```

CORRECT:

```
MY_CONSTANT

NEWLINE

SUPER_CLASS_VERSION

$str = str_replace(LD.'foo'.RD, 'bar', $str);
```

درست ، غلط و پوچ:

کلمات null و false و true همیشه با حروف بزرگ نوشته می شوند.

INCORRECT:

```
if ($foo == true)

$bar = false;

function foo($bar = null)
```

CORRECT:

```
if ($foo == TRUE)

$bar = FALSE;
```

function foo($bar = NULL)

عملگر های منطقی:

به جای استفاده از || برای یا از OR استفاده کنید.

به جای استفاده از AND از && استفاده کنید.

بعد از علامت نقیض همیشه یک space فاصله قرار دهید.

INCORRECT:

if ($foo || $bar)

if ($foo AND $bar) // okay but not recommended for common syntax highlighting applications

if (!$foo)

if (! is_array($foo))

CORRECT:

if ($foo OR $bar)

if ($foo && $bar) // recommended

if (! $foo)

if (! is_array($foo))

مقایسه مقدارهای بازگشتی:

اغلب توابع PHP در هنگام مواجه شدن با شکست مقدار False را باز می گردانند.

اما ممکن است مقدار بازگشتی در هنگام ناموفق بودن مقدار صفر باشد پس شاید بخواهیم ارزیابی و مقایسه کنید

برای سنجش مقدار بولین باید به جای == از === استفاده کنید و برای نقیض از ==!

INCORRECT:

```
// If 'foo' is at the beginning of the string, strpos will return a 0,

// resulting in this conditional evaluating as TRUE

if (strpos($str, 'foo') == FALSE)
```

CORRECT:

```
if (strpos($str, 'foo') === FALSE)
```

INCORRECT:

```
function build_string($str = "")

{

        if ($str == "")    // uh-oh!  What if FALSE or the integer 0 is passed as an argument?

        {

        }

}
```

CORRECT:

```
function build_string($str = "")

{

        if ($str === "")

        {

        }

}
```

تعیین کردن نوع رشته برای متغیر $str

```
$str = (string) $str;      // cast $str as a string
```

غلط یابی کد:

برای اشکال زدایی و رفع عیب کد خود می توانید از توابع زیر استفاده کنید:

var_dump(), print_r(), die(), and exit()

```
// print_r($foo);
```

کلاس و نام های فایل ها با استفاده از کلمات عمومی:

زمانی که کلاس یا فایل شما کلمه مشترکی داشته باشد یا ممکن است کاملا مانند نام یکی از توابع PHP باشد از پسوند های منحصر به فرد استفاده کیدتا مشکلی پیش نیاید.

INCORRECT:

class Email	pi.email.php
class Xml	ext.xml.php
class Import	mod.import.php

CORRECT:

class Pre_email	pi.pre_email.php
class Pre_xml	ext.pre_xml.php
class Pre_import	mod.pre_import.php

نام های جداول بانک اطلاعاتی:

هر جدولی که شما اضافه می کنید.

حتما باید اضافه exp_ را اضافه کنید بعد از این کارکتر هر نامی که نمودار و توصیف کننده موضوع و یا کمپانی است می تواند بیایید ، و یا حتی توضیح مختصری راجع به جدول می تواند قرار گیرد.

INCORRECT:

```
email_addresses          // missing both prefixes
pre_email_addresses      // missing exp_ prefix
exp_email_addresses      // missing unique prefix
```

CORRECT:

exp_pre_email_addresses

کلاس محک زنی (**Benchmarking Class**) :

Codeigniter دارای یک کلاس محک زنی می باشد که همیشه فعال می باشد.

که کاربر را قادر می سازد که فاصله زمانی بین دو نقطه را محاسبه کند.

استفاده کردن از کلاس محک زدن (**Using the Benchmark Class**) :

کلاس های محک زن می تواند داخل کنترلرها(کلاس ها) و یا داخل مدل ها و یا حتی داخل سند view ها نوشته شود.

- نقطه شروع را علامت گزاری کنید
- نقطه پایانی را علامت گزاری کنید
- برای مشاهده نتیجه تابع elapsed time را فراخانی کنید.

یک مثال از قطعه کد:

```
$this->benchmark->mark('code_start');

// Some code happens here

$this->benchmark->mark('code_end');

echo $this->benchmark->elapsed_time('code_start', 'code_end');
```

توجه: کلمات code_start و code_end کلمات فرضی هستند ، شما می توانید هر نوع کلمه ای مشابه این عبارت استفاده کنید.

حتی می توانید چندین کد را با یکدیگر استفاده کنید.

```
$this->benchmark->mark('dog');
// Some code happens here
$this->benchmark->mark('cat');
// More code happens here
$this->benchmark->mark('bird');
```

```
echo $this->benchmark->elapsed_time('dog', 'cat');
echo $this->benchmark->elapsed_time('cat', 'bird');
echo $this->benchmark->elapsed_time('dog', 'bird');
```

نمایش میزان زمان اجرا (**Displaying Total Execution Time**) :

با استفاده از دستور زیر می توانید میزان زمان اجرا را تخمین بزنید:

```
<?php echo $this->benchmark->elapsed_time();?>
```

کوچک شده کد بالا:

```
{elapsed_time}
```

نمایش میزان مصرف حافظه (**Displaying Memory Consumption**) :

با استفاده از دستور زیر می توانید میزان مصرف حافظه را تعیین کنید.

```
<?php echo $this->benchmark->memory_usage();?>
```

کوچک شده کد بالا:

```
{memory_usage}
```

کلاس تقویم (**Calendaring Class**) :

کلاس های تقویم به شما امکان ایجاد تقویم پویا می دهد ، از طریق فرمت ارائه شده به وسیله تقویم این امکان برای شما فراهم شده است که کنترل 100% روی تقویم داشته باشید.

شما همچنین می توانید اطلاعاتی را به سلولهای جدول پاس دهید.

همانند اکثر کلاسها در codeigniter ، شما می توانید توابع و کلاس تقویم را به وسیله کد زیر فراخانی کنید:

```
$this->load->library('calendar');
```

پس از بارگزاری ، تقویم به وسیله قطعه کد $this->clalender در دسترس شماست.

نمایش تقویم (**Displaying a Calendar**) :

یک مثال ساده از چگونگی نمایش تقویم

```
$this->load->library('calendar');

echo $this->calendar->generate();
```

در مثال بالا این قطعه کد زمان هم اکنون را روی سرور برای شما می سازد ، برای نشان دادن ماه و سال خاص باید مقادیر مورد نظر را به تابع پاس داد.

```
$this->load->library('calendar');

echo $this->calendar->generate(2006, 6);
```

در کد بالا شما تقویم ماه June و سال 2006 را می سازد ، اولین پارامتر مربوط به سال و دومین آرگومان مربوط به ماه است.

کلاس های پایگاه داده (The Database Class) :

شروع سریع:

پردازش کلاس پایگاه داده (**Initializing the Database Class**) :

در صورتی که قبلا در فایل config تنظیمات پایگاه داده را انجام داده باشید این قطعه کد پایگاه داده شما را فراخانی می کند.

```
$this->load->database();
```

بعد از فراخانی پایگاه داد به توضیح سایر قسمت ها می پردازیم:

توجه: در صورتی که تمامی صفحات نیاز به پایگاه داده دارند دیگر نیازی به فراخانی به این شکل نیست می توانید به صورت خودکار این کار را از طریق پوشه config انجام دهید.

قطعه کد استاندارد برای نمایش سطرهای زیاد به روش آبجکت:

```
$query = $this->db->query('SELECT name, title, email FROM my_table');

foreach ($query->result() as $row)
{
   echo $row->title;
```

```
    echo $row->name;
    echo $row->email;
}

echo 'Total Results: ' . $query->num_rows();
```

تابع ()Result در کد بالا اطلاعات را باز می گرداند.

قطعه کد استاندارد برای نمایش سطرهای زیاد به روش آرایه:

```
$query = $this->db->query('SELECT name, title, email FROM my_table');

foreach ($query->result_array() as $row)
{
    echo $row['title'];
    echo $row['name'];
    echo $row['email'];
}
```

آزمایش خروجی (**Testing for Results**) :

زمانی که قطعه خود را اجرا کردید ولی خروجی حاصل نشد ، می توانید به وسیله تابع num_rows() خروجی کد خویش را آزمایش کنید.

```
$query = $this->db->query("YOUR QUERY");

if ($query->num_rows() > 0)
{
   foreach ($query->result() as $row)
   {
      echo $row->title;
      echo $row->name;
      echo $row->body;
   }
}
```

قطعه کد استاندارد برای خروجی تک سطر (**Standard Query With Single Result**) :

```
$query = $this->db->query('SELECT name FROM my_table LIMIT 1');

$row = $query->row();
echo $row->name;
```

به یاد داشته باشید تمامی دستورات اس کی یو ال باید با حروف بزرگ نوشته شوند.

قطعه کد استاندارد برای خروجی تک سطر (**Standard Query With Single Result (Array version)**) :

```
$query = $this->db->query('SELECT name FROM my_table LIMIT 1');

$row = $query->row_array();
echo $row['name'];
```

ورود استاندارد (**Standard Insert**) :

```
$sql = "INSERT INTO mytable (title, name)
        VALUES (".$this->db->escape($title).", ".$this->db->escape($name).")";

$this->db->query($sql);

echo $this->db->affected_rows();
```

Active Record Query :

ساده شده دستورات بالا به این شکل است نمونه عملیات بالا توسط توابع زیر انجام می شود

```
$query = $this->db->get('table_name');

foreach ($query->result() as $row)
{
   echo $row->title;
}
```

Active Record Insert:

```
$data = array(
            'title' => $title,
            'name' => $name,
```

```
    'date' => $date
);

$this->db->insert('mytable', $data);

// Produces: INSERT INTO mytable (title, name, date) VALUES ('{$title}', '{$name}', '{$date}')
```

Codeigniter از الگویی به نام Active Record پایگاه داده استفاده می کند. که این الگو به شما اجازه می دهد

خواسته های پایگاه داده ای خود را که در قالب insert،update و ... هستند را به صورت بهینه تر و کوچک تر پیاده پیاده سازی نمایید.

در بسیاری از مواقع نیازی به نوشتن بیش از یک و یا 2 خط ندارید.

فراتر از این به سادگی می توانید از سود ها و امکانات Active Record استفاده کنید مانند درست کردن پایگاه داده مستق و درون برنامه.

از زمانی که قطعه کد برای هر پایگاه داده ساخته شد به شما اجازه داده می شود که کد امن تری بسازید ، بنابر این مقادیر به صورت خودکار توسط سیستم Escape می شوند.

:Selecting Data

این تابع به شما اجازه می دهد تا دستورات select ، SQL ، را پیاده سازی کنید.

$this->db->get();

به وسیله این قطعه کد کلیه اطلاعات ستون ها و جداول در خروجی ظاهر می شود.

```
$query = $this->db->get('mytable');
// Produces: SELECT * FROM mytable
```

بخش دوم و سوم آرگومان تابع به ترتیب نشان دهنده محدودیت از سطر چندم و تعداد آن می باشد.

```
$query = $this->db->get('mytable', 10, 20);

// Produces: SELECT * FROM mytable LIMIT 20, 10 (in MySQL. Other databases have slightly different syntax)
```

حال برای نشان دادن خروجی:

```
$query = $this->db->get('mytable');

foreach ($query->result() as $row)
{
   echo $row->title;
}
```

در ادامه بحث به توابع نتیجه (Result Function) می پردازیم.

$this->db->get_where();

Where اضافه شده در تابع بالا به شما این امکان را می دهد که محدودیتی روی Select ایجاد کنید

```
$query = $this->db->get_where('mytable', array('id' => $id), $limit, $offset);
```

$this->db->select();

به شما اجازه می دهد که Select را در دستورات درج کنید.

```
$this->db->select('title, content, date');

$query = $this->db->get('mytable');

// Produces: SELECT title, content, date FROM mytable
```

توجه : اگر شما می خواهید به کمک * کلیه ستون ها را در خروجی چاپ کنید دیگر نیازی به استفاده از این تابع ندارید

اگر مقدار پارامتر دوم تابع Select() را به False تبدیل کنید.

Codeigniter دیگر هیچ کنترل و محافظتی روی نام جدول و فیلد شما ندارد ، در صورت درج نکردن مقدار True ، False پیش فرض محسوب می شود.

```
$this->db->select('(SELECT SUM(payments.amount) FROM payments WHERE payments.invoice_id=4')
AS amount_paid', FALSE);
$query = $this->db->get('mytable');
```

$this->db->select_max();

به کمک این تابع می توانید بیشترین مقدار را پیدا کنید و در صورت علاقه می توانید با تعیین نام مناسب در بخش دوم تابع به عنوان آرگومان ، نام مجازی برای آن ستون ایجاد کنید.

```
$this->db->select_max('age');
$query = $this->db->get('members');
// Produces: SELECT MAX(age) as age FROM members

$this->db->select_max('age', 'member_age');
$query = $this->db->get('members');
// Produces: SELECT MAX(age) as member_age FROM members
```

$this->db->select_min();

```
$this->db->select_min('age');
$query = $this->db->get('members');
// Produces: SELECT MIN(age) as age FROM members
```

$this->db->select_avg();

```
$this->db->select_avg('age');
$query = $this->db->get('members');
// Produces: SELECT AVG(age) as age FROM members
```

$this->db->select_sum();

```
$this->db->select_sum('age');
$query = $this->db->get('members');
// Produces: SELECT SUM(age) as age FROM members
```

$this->db->from();

به شما اجازه داده می شود که بخشی از کد را بنویسید:

```
$this->db->select('title, content, date');
$this->db->from('mytable');

$query = $this->db->get();

// Produces: SELECT title, content, date FROM mytable
```

$this->db->join();

با نوشتن کد زیر به شما اجازه داده می شود دو جدول را به یکدیگر متصل کنید:

```
$this->db->select('*');
$this->db->from('blogs');
$this->db->join('comments', 'comments.id = blogs.id');

$query = $this->db->get();

// Produces:
```

```
// SELECT * FROM blogs
// JOIN comments ON comments.id = blogs.id
```

شما همچنین می توانید پیوند های دیگری نظیر left, right, outer, inner, left outer, and right outer را پیاده سازی کنید.

مثال:

```
$this->db->join('comments', 'comments.id = blogs.id', 'left');

// Produces: LEFT JOIN comments ON comments.id = blogs.id
```

$this->db->where();

این تابع به صورت 4 الگو پیاده سازی می شود.

1- Simple key/value method:

```
$this->db->where('name', $name);

// Produces: WHERE name = 'Joe'
```

چند خطی:

```
$this->db->where('name', $name);
$this->db->where('title', $title);
$this->db->where('status', $status);

// WHERE name 'Joe' AND title = 'boss' AND status = 'active'
```

2- **Custom key/value method:**

```
$this->db->where('name !=', $name);
$this->db->where('id <', $id);

// Produces: WHERE name != 'Joe' AND id < 45
```

3- **Associative array method:**

```
$array = array('name' => $name, 'title' => $title, 'status' => $status);

$this->db->where($array);

// Produces: WHERE name = 'Joe' AND title = 'boss' AND status = 'active'
```

و یا اگر از عملگر مساوی استفاده نمی کنید می توانید از کد زیر استفاده کنید:

```
$array = array('name !=' => $name, 'id <' => $id, 'date >' => $date);

$this->db->where($array);
```

4- Custom string:

```
$where = "name='Joe' AND status='boss' OR status='active'";

$this->db->where($where);
```

$this->db->or_where();

```
$this->db->where('name !=', $name);
$this->db->or_where('id >', $id);

// Produces: WHERE name != 'Joe' OR id > 50
```

توجه: توصیه نمی شود از do_where() به جای or_where() استفاده کنید.

$this->db->where_in();

```
$names = array('Frank', 'Todd', 'James');
$this->db->where_in('username', $names);
// Produces: WHERE username IN ('Frank', 'Todd', 'James')
```

$this->db->or_where_in();

```
$names = array('Frank', 'Todd', 'James');
$this->db->or_where_in('username', $names);
// Produces: OR username IN ('Frank', 'Todd', 'James')
```

$this->db->where_not_in();

```
$names = array('Frank', 'Todd', 'James');
$this->db->where_not_in('username', $names);
// Produces: WHERE username NOT IN ('Frank', 'Todd', 'James')
```

$this->db->or_where_not_in();

```
$names = array('Frank', 'Todd', 'James');
$this->db->or_where_not_in('username', $names);
// Produces: OR username NOT IN ('Frank', 'Todd', 'James')
```

$this->db->like();

این تابع تمام مقدار های a پاس داده شده به این تابع را با b به شکل خودکار Escape می کند.

1- Simple key/value method:

```
$this->db->like('title', 'match');
// Produces: WHERE title LIKE '%match%'
```

می توانید چندین خط از این تابع را در کنار یکدیگر استفاده کنید مانند دستور AND در دستورات SQL عمل خواهد کرد.

```
$this->db->like('title', 'match');
$this->db->like('body', 'match');

// WHERE title LIKE '%match%' AND body LIKE '%match%
```

```
$this->db->like('title', 'match', 'before');
// Produces: WHERE title LIKE '%match'

$this->db->like('title', 'match', 'after');
// Produces: WHERE title LIKE 'match%'

$this->db->like('title', 'match', 'both');
// Produces: WHERE title LIKE '%match%'
```

2- **Associative array method:(** پاس دادن چندین مقدار به وسیله آرایه **)**

```
$array = array('title' => $match, 'page1' => $match, 'page2' => $match);

$this->db->like($array);

// WHERE title LIKE '%match%' AND page1 LIKE '%match%' AND page2 LIKE '%match%'
```

$this->db->or_like();

```
$this->db->like('title', 'match');
$this->db->or_like('body', $match);

// WHERE title LIKE '%match%' OR body LIKE '%match%'
```

$this->db->or_not_like();

```
$this->db->like('title', 'match');
$this->db->or_not_like('body', 'match');

// WHERE title LIKE '%match% OR body NOT LIKE '%match%'
```

$this->db->group_by();

پیاده سازی دستور group by

```
$this->db->group_by("title");
// Produces: GROUP BY title
```

پاس دادن 2 ورودی:

```
$this->db->group_by(array("title", "date"));
// Produces: GROUP BY title, date
```

$this->db->distinct();

محدود کردن خروجی:

```
$this->db->distinct();
$this->db->get('table');

// Produces: SELECT DISTINCT * FROM table
```

$this->db->having();

دستور having با یک آرگومان:

```
$this->db->having('user_id = 45');
// Produces: HAVING user_id = 45
```

دستور having با 2 آرگومان:

```
$this->db->having('user_id', 45);
// Produces: HAVING user_id = 45
```

```
$this->db->having(array('title =' => 'My Title', 'id <' => $id));
// Produces: HAVING title = 'My Title', id < 45


$this->db->having('user_id', 45);
// Produces: HAVING `user_id` = 45 in some databases such as MySQL
$this->db->having('user_id', 45, FALSE);
// Produces: HAVING user_id = 45
```

$this->db->or_having();

مانند دستور Having عمل می کند

$this->db->order_by();

نزولی:

```
$this->db->order_by("title", "desc");
// Produces: ORDER BY title DESC
```

صعودی:

```
$this->db->order_by('title desc, name asc');
// Produces: ORDER BY title DESC, name ASC

$this->db->order_by("title", "desc");
$this->db->order_by("name", "asc");

// Produces: ORDER BY title DESC, name ASC
```

$this->db->limit();

```
$this->db->limit(10);
// Produces: LIMIT 10
```

```
$this->db->limit(10, 20);
// Produces: LIMIT 20, 10 (in MySQL. Other databases have slightly different syntax)
```

$this->db->count_all_results();

```
echo $this->db->count_all_results('my_table');
// Produces an integer, like 25

$this->db->like('title', 'match');
$this->db->from('my_table');
echo $this->db->count_all_results();
// Produces an integer, like 17
```

$this->db->count_all();

```
echo $this->db->count_all('my_table');
// Produces an integer, like 25
```

Inserting Data:

$this->db->insert();

وارد کردن اطلاعات:

وارد کردن اطلاعات به روش آزایه:

```
$data = array(
            'title' => 'My title' ,
            'name' => 'My Name' ,
            'date' => 'My date'
          );

$this->db->insert('mytable', $data);
// Produces: INSERT INTO mytable (title, name, date) VALUES ('My title', 'My name', 'My date')
```

اولین آرگومان دربردارنده نام جدول می باشد و دومین نام ستون های جدول پایگاه داده

وارد کردن اطلاعات به روش آبجکت:

```
/*
   class Myclass {
      var $title = 'My Title';
      var $content = 'My Content';
      var $date = 'My Date';
   }
*/
$object = new Myclass;
$this->db->insert('mytable', $object);

// Produces: INSERT INTO mytable (title, content, date) VALUES ('My Title', 'My Content', 'My Date')
```

$this->db->set();

این تابع هم برای عملیات Insert پیاده سازی می شود و هم برای عملیات Update

```
$this->db->set('name', $name);
$this->db->insert('mytable');

// Produces: INSERT INTO mytable (name) VALUES ('{$name}')
```

چندین دستور کنار یکدیگر:

```
$this->db->set('name', $name);
$this->db->set('title', $title);
$this->db->set('status', $status);
$this->db->insert('mytable');
```

با اضافه کردن False برای آرگومان سوم به تابع می گوییم عملیات Escape کردن را انجام ندهد

```
$this->db->set('field', 'field+1', FALSE);
$this->db->insert('mytable');
// gives INSERT INTO mytable (field) VALUES (field+1)
```

```
$this->db->set('field', 'field+1');
$this->db->insert('mytable');
// gives INSERT INTO mytable (field) VALUES ('field+1')
```

پاس دادن اطلاعات به تابع Set() توسط آرایه:

```
$array = array('name' => $name, 'title' => $title, 'status' => $status);

$this->db->set($array);
$this->db->insert('mytable');
```

به روش آبجکت:

```
/*
   class Myclass {
      var $title = 'My Title';
      var $content = 'My Content';
      var $date = 'My Date';
   }
*/

$object = new Myclass;

$this->db->set($object);
$this->db->insert('mytable');
```

به روز کردن اطلاعات (**Updating Data**) :

```
$this->db->update();
```

```
$data = array(
            'title' => $title,
            'name' => $name,
            'date' => $date
          );

$this->db->where('id', $id);
$this->db->update('mytable', $data);

// Produces:
// UPDATE mytable
// SET title = '{$title}', name = '{$name}', date = '{$date}'
// WHERE id = $id
```

به روش آبجکت:

```
/*
   class Myclass {
      var $title = 'My Title';
      var $content = 'My Content';
      var $date = 'My Date';
   }
*/

$object = new Myclass;

$this->db->where('id', $id);
$this->db->update('mytable', $object);
```

```
// Produces:
// UPDATE mytable
// SET title = '{$title}', name = '{$name}', date = '{$date}'
// WHERE id = $id
```

یا:

```
$this->db->update('mytable', $data, "id = 4");

$this->db->update('mytable', $data, array('id' => $id));
```

حذف اطلاعات (**Deleting Data**) :

```
$this->db->delete();
```

به وسیله این تابع می توانید اطلاعات را حذف کنید:

```
$this->db->delete('mytable', array('id' => $id));

// Produces:
// DELETE FROM mytable
// WHERE id = $id
```

به روش دیگر:

```
$this->db->where('id', $id);
$this->db->delete('mytable');

// Produces:
// DELETE FROM mytable
// WHERE id = $id
```

شما در صورت امکان می توانید اطلاعات را هم زمان بیشتر از یک جدول حذف کنید.

```
$tables = array('table1', 'table2', 'table3');
$this->db->where('id', '5');
$this->db->delete($tables);
```

```
$this->db->empty_table();
```

شما می توانید به وسیله دستور زیر:

```
$this->db->from('mytable');
$this->db->truncate();
// or
$this->db->truncate('mytable');

// Produce:
// TRUNCATE mytable
```

اطلاعات یک جدول را تخلیه کنید.

متد زنجیره ای (Method Chaining) :

متد زنجیره ای به شما اجازه می دهد با چندین تابع اطتباط داشته باشید:

```
$this->db->select('title')->from('mytable')->where('id', $id)->limit(10, 20);

$query = $this->db->get();
```

Active Record Caching:

```
$this->db->start_cache();
```

صدا زدن این تابع باعث کش شدن اطلاعات می شود.

```
$this->db->stop_cache();
```

صدا زدن این تابع باعث خاتمه کش شدن می شود.

```
$this->db->flush_cache();
```

این تابع باعث می شود کلیه عملیات کشی که انجام شده است حذف شود:

```
$this->db->start_cache();
$this->db->select('field1');
$this->db->stop_cache();

$this->db->get('tablename');

//Generates: SELECT `field1` FROM (`tablename`)

$this->db->select('field2');
$this->db->get('tablename');

//Generates: SELECT `field1`, `field2` FROM (`tablename`)

$this->db->flush_cache();

$this->db->select('field2');
$this->db->get('tablename');

//Generates: SELECT `field2` FROM (`tablename`)
```

کلاس ایمیل (Email Class) :

Codeigniter دارای یک کلاس قوی برای فرستادن ایمیل می باشد که امکانات زیر را پشتیبانی می کند.

- پشتیبانی از انواع پروتکل های مانند : mail، sendmail، smtp
- چندین دریافت کننده
- BCC ، CC
- ایمیل کردن اسناد HTML و یا فقط متن ساده
- پیوست کردن اسناد
- شکست کلمه
- اولویت
- شکستن ایمیل های بزرگ به ایمیل های کوچک

- ابزاری برای کشف خطا در ایمیل

فرستادن ایمیل (Sending Email) :

فرستادن ایمیل در Codeigniter فقط ساده نیست اما شما می توانید بسته به احتیاج خود آن را از درون Config فایل تنظیم کنید.

در مثال زیر پایه کد برای فرستادن ایمیل توضیح داده شده است که می توانید این کد ها را درون اسناد کلاس های Controller قرار دهید.

```
$this->load->library('email');

$this->email->from('your@example.com', 'Your Name');
$this->email->to('someone@example.com');
$this->email->cc('another@another-example.com');
$this->email->bcc('them@their-example.com');

$this->email->subject('Email Test');
$this->email->message('Testing the email class.');

$this->email->send();

echo $this->email->print_debugger();
```

تنظیم کردن ایمیل (Setting Email Preferences) :

در تنظیمات ایمیل که ارائه شده است 17 امکان مختلف برای مناسب کردن ایمیل ارسالی گنجانده شده است.

شما می توانید یا به طور دستی به وسیله توضیحات در زیر آن را تنظیم کنید یا به طور خودکار از داخل سند Config آن را تنظیم کنید.

```
$config['protocol'] = 'sendmail';
$config['mailpath'] = '/usr/sbin/sendmail';
$config['charset'] = 'iso-8859-1';
$config['wordwrap'] = TRUE;
```

```
$this->email->initialize($config);
```

برای تنظیم کردن خودکار به صفحه Config/email.php مراجعه کنید.

منابع کلاس های ایمیل (Email Function Reference) :

```
$this->email->from();
```

ارسال کننده ایمیل

```
$this->email->from('you@example.com', 'Your Name');
```

```
$this->email->reply_to();
```

اگر اطلاعاتی به وسیله تابع form() ارسال نشد از تابع فوق استفاده کنید.

```
$this->email->reply_to('you@example.com', 'Your Name');
```

```
$this->email->to();
```

دریافت کننده ایمیل

```
$this->email->to('someone@example.com');
```

```
$this->email->to('one@example.com, two@example.com, three@example.com');
```

ارسال برای چندین ایمیل به کمک آرایه:

```
$list = array('one@example.com', 'two@example.com', 'three@example.com');
$this->email->to($list);
```

```
$this->email->cc();
```

دریافت کننده ایمیل با شرایط خاص:

```
$this->email->bcc();
```

دریافت کننده ایمیل با شرایط خاص:

```
$this->email->subject();
```

موضوع نامه

```
$this->email->subject('This is my subject');

$this->email->message();
```

متن نامه

```
$this->email->message('This is my message');

$this->email->set_alt_message();
```

ایمیل جایگزین:

```
$this->email->set_alt_message('This is the alternative message');

$this->email->clear();
```

اگر مقدار آرگومان تابع را به True تبدیل کنید تمام ضمیمه های پیوسته شده به نامه حذف خواهند شد.

```
foreach ($list as $name => $address)
{
   $this->email->clear();

   $this->email->to($address);
   $this->email->from('your@example.com');
   $this->email->subject('Here is your info '.$name);
   $this->email->message('Hi '.$name.' Here is the info you requested.');
   $this->email->send();
}

$this->email->clear(TRUE);

$this->email->send();
```

چگونه از ارسال ایمیل مطمئن شویم

```
if ( ! $this->email->send())
{
   // Generate error
}

$this->email->attach();
```

ضمیمه کردن تصاویر و غیره ...

```
$this->email->attach('/path/to/photo1.jpg');
$this->email->attach('/path/to/photo2.jpg');
$this->email->attach('/path/to/photo3.jpg');

$this->email->send();
```

کلاس های رمز گزاری (Encryption Class) :

کلاس های رمز گزاری 2 راه به شما توصیه می کنند .

تنظیم کردن کلید (Setting your Key) :

کلید یک قطعه اطلاعات است که پردازش رمز نگاری را کنترل می کنند و اجازه می دهند که رشته رمز نگاری و یا رمز گشایی شود.

در حقیقت کلیدی که شما انتخاب می کنید هم در رمز نگاری و هم در رمز گشایی استفاده می شود پس برای انتخاب آن دقت کنید.

می توانید کلید را در سند Config تغییر دهید

```
$config['encryption_key'] = "YOUR KEY";
```

پردازش کلاس (Initializing the Class) :

مانند تمامی کلاس ها در Codeigniter باید ابتدا کتابخانه مربوطه را بارگزاری کرد.

```
$this->load->library('encrypt');
```

بعد از بارگزاری می توانید از آن استفاده کنید.

```
$this->encrypt->encode()
```

برای انجام عمل Encryption می توانید از تابع فوق استفاده کنید.

```
$msg = 'My secret message';

$encrypted_string = $this->encrypt->encode($msg);
```

اگر نمی خواهید از کلید تعریف شده در صفحه Config استفاده کنید می توانید مانند نمونه زیر استفاده کنید.

```
$msg = 'My secret message';
$key = 'super-secret-key';

$encrypted_string = $this->encrypt->encode($msg, $key);

$this->encrypt->decode();
```

رمز گشایی رشته رمزنگاری شده:

```
$encrypted_string = 'APANtByIGI1BpVXZTJgcsAG8GZl8pdwwa84';
$plaintext_string = $this->encrypt->decode($encrypted_string);
```

کلاس آپلود کردن (File Uploading Class) :

Codeigniter کلاس هایی دارا می باشد که به شما اجازه می دهد که Uploading را انجام دهید.

شما می توانید امکانات مختلف آن را ست کنید.

مثل ، محدود کردن فرمت و یا حجم ارسال شده.

پردازش (The Process) :

عمل آپلود کردن شامل انجام عملیاتی است که فایلی را به شکل غیر مستقیم بر روی فضای وب سایت منتقل می کنیم.

- فرم ارسال نمایش دیده می شود
- به کاربر اجازه داده می شود اسناد خود را انتخاب کند سپس آن را آپلود کند.
- بعد از فشردن دکمه Submit سند مشخص شده به مقصد فرستاده می شود.
- در امتداد راه ، فایل معتبر سازی می شود برا اساس آن تنظیماتی که شما ست کرده اید.
- پس از آپلود ، کاربر پیغام موفقیت آمیز دریافت می کند.

درست کردن فرم آپلود (Creating the Upload Form) :

از یک ویرایشگر متن استفاده کنید ، یک فرم به نام Upload_form.php ساخته و آن را درون مسیر applications/Views ذخیره کنید.

```
<html>

<head>
```

```
<title>Upload Form</title>

</head>

<body>

<?php echo $error;?>

<?php echo form_open_multipart('upload/do_upload');?>

<input type="file" name="userfile" size="20" />

<br /><br />

<input type="submit" value="upload" />

</form>

</body>

</html>
```

شما باید به این نکته توجه کنید که ما در کد خود از یک باز کننده فرم کمکی استفاده می کنیم.

شما باید به این نکته نیز توجه کنید که متغیری به نام $error ایجاد کرده ایم که خطا های احتمالی را نشان دهیم.

The Success Page:

از یک ویرایشگر متن استفاده کنید و سندی به نام Upload_sucess.php بسازید و کد را داخل آن درج نمایید و /ان را درون مسیر Applications/views ذخیره کنید.

```
<html>

<head>

<title>Upload Form</title>

</head>

<body>

<h3>Your file was successfully uploaded!</h3>
```

```
<ul>

<?php foreach($upload_data as $item => $value):?>

<li><?php echo $item;?>: <?php echo $value;?></li>

<?php endforeach; ?>

</ul>

<p><?php echo anchor('upload', 'Upload Another File!'); ?></p>

</body>

</html>
```

The Controller**:**

از یک ویرایشگر متن استفاده کنید و فایلی با کد زیر با نام Upload.php بسازید.و آن را درون شاخه Applications/Controllers ذخیره کنید.

```
<?php

class Upload extends Controller {

	function Upload(){

		parent::Controller();

		$this->load->helper(array('form', 'url'));

	}

	function index(){

		$this->load->view('upload_form', array('error' => ' ' ));

	}

	function do_upload(){

		$config['upload_path'] = './uploads/';

		$config['allowed_types'] = 'gif|jpg|png';
```

```
                $config['max_size']   = '100';

                $config['max_width']  = '1024';

                $config['max_height']  = '768';


                $this->load->library('upload', $config);


                if ( ! $this->upload->do_upload()){

                        $error = array('error' => $this->upload->display_errors());

                        $this->load->view('upload_form', $error);

                }

                else{

                        $data = array('upload_data' => $this->upload->data());

                        $this->load->view('upload_success', $data);

                }

        }

}

?>
```

ساخت پوشه آپلود (The Upload Folder) :

شما احتیاج دارید که یک پوشه که مقصد تصویر آپلود شده است ایجاد کنید و همچنین باید دسترسی 777 برای آن ایجاد کنید(محل قرار گیری این پوشه در کنار فایل اصلی index.php ، Root برنامه می باشد.)

برای انجام کار آدرس زیر را داخل محل آدرس مرورگر خود وارد کنید

example.com/index.php/upload/

اگر تمامی مراحل را به درستی انجام داده باشید تصویر باید به درستی آپلود شود.

(ضمیمه این کتاب قطعه کدی برای آپلود کردن تصویر و تغییر حجم آن و حذف عکس می باشد.)

تحلیل کلاس های آپلود (Initializing the Upload Class) :

مانند اکثر کلاس های دیگر در Codeigniter ، کلاس های Upload را نیز باید اول فراخانی شوند.

```
$this->load->library('upload');

$this->upload->do_upload();
```

انجام عملیات آپلود به کمک تابع بالا امکان پذیر می باشد.

توجه : در تگ فورم حتما باید از این مشخصه برای بارگزاری تصویر استفاده کنید:

```
<form method="post" action="some_action" enctype="multipart/form-data" />
```

اگر شما می پسندید که نام ساده شخصی به مقدار تابع do_upload() پاس دهید می توانید به کمک کد زیر این کار را انجام دهید.

```
$field_name = "some_field_name";
$this->upload->do_upload($field_name);


$this->upload->display_errors();
```

بازیابی هر پیغام خطا:

اگر مقدار تابع do_upload() ، False بود یعنی عملیات با شکست مواجه شده است اما این تابع به صورت خودکار ، این امر را اطلاع نمی دهد.

پس از تابع زیر استفاده می کنیم.

```
$this->upload->display_errors();
```

شما همچنین می توانید به خطا فرمت بدهید.

```
$this->upload->display_errors('<p>', '</p>');

$this->upload->data();
```

این یک تابع کمک کننده است زیرا اطلاعات ارزشمندی راجع به سند آپلود شده در اختیار شما قرار می دهد.

Item	Description
file_name	The name of the file that was uploaded including the file extension.

```
Array
(
    [file_name]    => mypic.jpg
    [file_type]    => image/jpeg
    [file_path]    => /path/to/your/upload/
    [full_path]    => /path/to/your/upload/jpg.jpg
    [raw_name]     => mypic
    [orig_name]    => mypic.jpg
    [file_ext]     => .jpg
    [file_size]    => 22.2
    [is_image]     => 1
    [image_width]  => 800
    [image_height] => 600
    [image_type]   => jpeg
    [image_size_str] => width="800" height="200"
)
```

file_type	The file's Mime type
file_path	The absolute server path to the file
full_path	The absolute server path including the file name
raw_name	The file name without the extension
orig_name	The original file name. This is only useful if you use the encrypted name option.
file_ext	The file extension with period
file_size	The file size in kilobytes
is_image	Whether the file is an image or not. 1 = image. 0 = not.
image_width	Image width.
image_heigth	Image height
image_type	Image type. Typically the file extension without the period.
image_size_str	A string containing the width and height. Useful to put into an image tag.

جدول زیر اطلاعات دقیق تری در اختیار شما قرار می دهد.

ولید کردن اطلاعات فرم (Form Validation) :

Overview:

قبل از توضیح راجع به ولید کردن اطلاعات فرم اجازه دهید توضیح دهیم هدف از ولید کردن اطلاعات چیست:

- فرم نمایش پیدا می کند
- شما باید فرم را پر کنید و ارسال کنید
- اگر شما هر درخواستی را که تعیین نشده است و یا به اندازه کافی گویا نیست ارسال کنید دچار مشکل می شوید.
- این پردازش ها ادامه پیدا می کند تا اطلاعات شما داخل فرم ولید باشد
- در پایان تمام اطلاعات لازم چک می شوند
- تمامی اطلاعات بر اساس نوع عملکرد و ضوابط خاص بازبینی می شوند

مثال: کلمع عبور نمی تواند خالی ارسال شود

- مطابق بودن اطلاعات با اصول امنیتی
- در صورت نیاز تغییر اطلاعات
- ساده سازی اطلاعات برای ورود به پایگاه داده

بر عکس تصورات رایج معتبر سازی اصلا عملیات پیچیده ای نیست بلکه اگر درست پیاده سازی نشود برنامه نویس را دچار آشفتگی می کند.

تمرین معتبر کردن فرم (Form Validation Tutorial) :

از یک ویرایشگر متن استفاد کنید و صفحه ای به نام Myform.php داخل فولدر Application/views ایجاد کنید.

```
<html>

<head>

<title>My Form</title>

</head>

<body>

<?php echo validation_errors(); ?>

<?php echo form_open('form'); ?>

<h5>Username</h5>

<input type="text" name="username" value="" size="50" />

<h5>Password</h5>
```

```
<input type="text" name="password" value="" size="50" />

<h5>Password Confirm</h5>

<input type="text" name="passconf" value="" size="50" />

<h5>Email Address</h5>

<input type="text" name="email" value="" size="50" />

<div><input type="submit" value="Submit" /></div>

</form>

</body>

</html>
```

The Success Page:

از یک ویرایشگر متن استفاده کنید و صفحه ای به نام formsuccess.php در داخل مسیر Application/views ایجاد کنید:

```
<html>

<head>

<title>My Form</title>

</head>

<body>

<h3>Your form was successfully submitted!</h3>

<p><?php echo anchor('form', 'Try it again!'); ?></p>

</body>

</html>
```

The Controller:

از یک ویرایشگر متن استفاده کنید و صفحه ای به نام form.php در مسیر Application/controllers ایجاد کنید.

```
<?php

class Form extends Controller {

        function index(){

                $this->load->helper(array('form', 'url'));

                $this->load->library('form_validation');

                if ($this->form_validation->run() == FALSE){

                        $this->load->view('myform');

                }

                else{

                        $this->load->view('formsuccess');

                }

        }

}

?>
```

در صورتی که کلیه مراحل بالا را با موفقیت طی کرده باشید با وارد کردن آدرس زیر مرورگر وب می توانید فرم را مشاهده کنید :

example.com/index.php/form/

توجه: کد زیر در اسناد بالا تمامی خطاهای به وقوع پیوسته در سیستم معتبر سازی را نمایان می سازد.

```
<?php echo validation_errors(); ?>
```

شیوه برپایی معتبر سازی (Setting Validation Rules) :

Codeigniter به شما اجازه می دهد تا تعداد زیادی معتبرساز را که برای گرفتن فیلد ، نیاز دارید بنا کنید.

ساختار اصلی آن بدین شکل می باشد:

```
$this->form_validation->set_rules();
```

تابع بالا 3 آرگومان ورودی دریافت می کند:

1- نام فیلد مورد نظر : نام دقیق فیلد در فرم
2- نامی که کاربران در هنگام بروز خطا آن را مشاهده می کنند.
3- عملیاتی که سیستم بر روی فیلد انجام می دهد.

یک مثال ساده:

```
$this->form_validation->set_rules('username', 'Username', 'required');
$this->form_validation->set_rules('password', 'Password', 'required');
$this->form_validation->set_rules('passconf', 'Password Confirmation', 'required');
$this->form_validation->set_rules('email', 'Email', 'required');
```

حال عبارت بالا را در کلاس Controller جایگزین می کنیم:

```
<?php
class Form extends Controller {
	function index(){
		$this->load->helper(array('form', 'url'));
		$this->load->library('form_validation');
		$this->form_validation->set_rules('username', 'Username', 'required');
		$this->form_validation->set_rules('password', 'Password', 'required');
		$this->form_validation->set_rules('passconf', 'Password Confirmation',
'required');
		$this->form_validation->set_rules('email', 'Email', 'required');
		if ($this->form_validation->run() == FALSE){
			$this->load->view('myform');
		}
		else{
			$this->load->view('formsuccess');
		}
```

```
        }
}
?>
```

به یاد داشته باشید مانند اکثر کلاس ها در Codeigniter ابتدا باید کلاس های مربوط به سیستم معتبر ساز را فراخانی کرد:

```
$this->load->library('form_validation');
```

شیوه درست دریافت اطلاعات (Cascading Rules) :

Codeigniter به شما اجازه میدهد که از چندین شیوه مانند نمونه پایین برای معتبر کردن اطلاعات خود استفاده کنید.

```
$this->form_validation->set_rules('username', 'Username',
'required|min_length[5]|max_length[12]');
$this->form_validation->set_rules('password', 'Password', 'required|matches[passconf]');
$this->form_validation->set_rules('passconf', 'Password Confirmation', 'required');
$this->form_validation->set_rules('email', 'Email', 'required|valid_email');
```

- تعداد کارکترهای کلمه عبور برای کمترین حالت 5 کارکتر و برای بیشترین حالت 12 کارکتر می باشد.
- رمز عبور باید با Password Confirm مشابه باشد.
- آدرس ایمیل باید معتبر باشد.

ساده سازی اطلاعات (Prepping Data) :

علاوه بر امکانات ذکر شده شما می توانید به روشهای متعددی اطلاعات ارسال شده را ساده کنید:

```
$this->form_validation->set_rules('username', 'Username',
'trim|required|min_length[5]|max_length[12]|xss_clean');


$this->form_validation->set_rules('password', 'Password',
'trim|required|matches[passconf]|md5');


$this->form_validation->set_rules('passconf', 'Password Confirmation', 'trim|required');


$this->form_validation->set_rules('email', 'Email', 'trim|required|valid_email');
```

نگه داشتن مقادیر داخل فرم (Re-populating the form) :

سند Myform.php را باز کنید و ۀن را مطابق با کد زیرویرایش کنید هرگز فراموش نکنید تابع Set_value() را جایگزین کنید.

```
set_value('field name')

--------------------------------


<html>

<head>

<title>My Form</title>

</head>

<body>

<?php echo validation_errors(); ?>

<?php echo form_open('form'); ?>

<h5>Username</h5>
```

```
<input type="text" name="username" value="<?php echo set_value('username'); ?>"
size="50" />

<h5>Password</h5>

<input type="text" name="password" value="<?php echo set_value('password'); ?>"
size="50" />

<h5>Password Confirm</h5>

<input type="text" name="passconf" value="<?php echo set_value('passconf'); ?>"
size="50" />

<h5>Email Address</h5>

<input type="text" name="email" value="<?php echo set_value('email'); ?>" size="50" />

<div><input type="submit" value="Submit" /></div>

</form>

</body>

</html>
```

صدا زدن تابع معتبر ساز خود (Callbacks: Your own Validation Functions) :

در سیستم معتبر ساز این امکان برای شما فراهم شده است که تابع خود را صدا بزنید و آن طور که نیاز دارید آن را توسعه دهید.

برای مثال:

در داخل کلاس قطعه کد Username را به روش زیر بازنویسی کنید.

```
$this->form_validation->set_rules('username', 'Username', 'callback_username_check');
```

```
<?php
```

```
class Form extends Controller {

	function index(){

		$this->load->helper(array('form', 'url'));

		$this->load->library('form_validation');

		$this->form_validation->set_rules('username', 'Username',
'callback_username_check');

		$this->form_validation->set_rules('password', 'Password', 'required');

		$this->form_validation->set_rules('passconf', 'Password Confirmation',
'required');

		$this->form_validation->set_rules('email', 'Email', 'required');

		if ($this->form_validation->run() == FALSE){

			$this->load->view('myform');

		}

		else{

			$this->load->view('formsuccess');

		}

	}

	function username_check($str){

		if ($str == 'test'){

			$this->form_validation->set_message('username_check', 'The %s
field can not be the word "test"');

			return FALSE;

		}

		else{

			return TRUE;

		}

	}

}

?>
```

Rule	Parameter	Description	Example
required	No	Returns FALSE if the form element is empty.	
matches	Yes	Returns FALSE if the form element does not match the one in the parameter.	matches[form_item]
min_length	Yes	Returns FALSE if the form element is shorter then the parameter value.	min_length[6]
max_length	Yes	Returns FALSE if the form element is longer then the parameter value.	max_length[12]
exact_length	Yes	Returns FALSE if the form element is not exactly the parameter value.	exact_length[8]
alpha	No	Returns FALSE if the form element contains anything other than alphabetical characters.	
alpha_numeric	No	Returns FALSE if the form element contains anything other than alpha-numeric characters.	
alpha_dash	No	Returns FALSE if the form element contains anything other than alpha-numeric characters, underscores or dashes.	
numeric	No	Returns FALSE if the form element contains anything other than numeric characters.	
integer	No	Returns FALSE if the form element contains anything other than an integer.	
is_natural	No	Returns FALSE if the form element contains anything other than a natural number: 0, 1, 2, 3, etc.	
is_natural_no_zero	No	Returns FALSE if the form element contains anything other than a natural number, but not zero: 1, 2, 3, etc.	
valid_email	No	Returns FALSE if the form element does not contain a valid email address.	
valid_emails	No	Returns FALSE if any value provided in a comma separated list is not a valid email.	
valid_ip	No	Returns FALSE if the supplied IP is not valid.	
valid_base64	No	Returns FALSE if the supplied string contains anything other than valid Base64 characters.	

Name	Parameter	Description
xss_clean	No	Runs the data through the XSS filtering function, described in the Input Class page.
prep_for_form	No	Converts special characters so that HTML data can be shown in a form field without breaking it.

prep_url	No	Adds "http://" to URLs if missing.
strip_image_tags	No	Strips the HTML from image tags leaving the raw URL.
encode_php_tags	No	Converts PHP tags to entities.

تنظیم کردن پیام های خطا (Setting Error Messages) :

تمامی خطاهای پیش فرض در داخل مسیر

language/english/form_validation_lang.php

قرار دارند ، برای اینکه پیام خطای دلخواه خود را ایجاد کنید باید به مراحل زیر توجه کنید:

```
$this->form_validation->set_message('rule', 'Error Message');
```

هر کجا که Rule مشابه قانون خاص بود متن نوشته شده را تغییر دهید.

اگر رشته خطا شامل %s باشد هر جا نام فیلدی انتخاب کرده باشید جایگزین آن خواهد شد.

در این مثال ، پیغام خطا تنظیم می شود هر کجا که نام تابع به آن پاس داده شود.

```
$this->form_validation->set_message('username_check')
```

شما همچنین می توانید هر پیغام خطایی در این باره را غیر فعال کنید.

```
$this->form_validation->set_message('required', 'Your custom message here');
```

تغییر دادن (Changing the Error Delimiters) :

به طور پیش فض کلاس نشان دادن خطاها برای فرم از تگ <P> استفاده می کند شما می توانید به 2 روش آن را تغییر دهید:

1- به طور عام (Changing delimiters Globally) :

برای این منظور در قسمت Controller ،تابع مقدار نمونه را جایگزین کنید.

```
$this->form_validation->set_error_delimiters('<div class="error">', '</div>');
```

در این مثال ما از تگ P به تگ Div منتقل شدیم.

2- به طور جداگانه (Changing delimiters Individually) :

در این قسمت 2 مدل وجود دارد:

```
<?php echo form_error('field name', '<div class="error">', '</div>'); ?>
```

یا

```
<?php echo validation_errors('<div class="error">', '</div>'); ?>
```

کمک کننده برای نمایش خطاها (Helper Reference) :

```
form_error();
```

نشان دهنده پیغام خطا در ارتباط با منبع تغذیه شده نام فیلد در فرم می باشد.

```
<?php echo form_error('username'); ?>
```

```
set_value();
```

به شما اجازه می دهد که مقدار ورودی برای مکان های ورودی لنتخاب کنید ، شما باید نام مکان مورد نظر را به عنوان آرگومان اول تابع به آن پاس بدهید ، دومین آرگومان مقدار آن می باشد.

```
<input type="text" name="quantity" value="<?php echo set_value('quantity', '0'); ?>" size="50" />
```

در تابع بالا به هنگام لود شدن مقدار صفر در داخل Textarea نمایان می شود.

set_select();

اگر شما از یک تگ <menu> استفاده کردید این تابع به شما اجازه می دهد که آیتم های منو را انتخاب کنید.

اولین آرگومان تابع در برگیرنده نام تگ <select> می باشد و دومین آرگومان دربردارنده مقدار هر آیتم است و آرگومان سوم نیز نشان دهنده با مقدار بولین True و یا False می باشد که آیا مقدار پیش فرض انتخاب شود یا خیر.

```
<select name="myselect">
<option value="one" <?php echo set_select('myselect', 'one', TRUE); ?> >One</option>
<option value="two" <?php echo set_select('myselect', 'two'); ?> >Two</option>
<option value="three" <?php echo set_select('myselect', 'three'); ?> >Three</option>
</select>
```

set_checkbox();

به شما اجازه می دهد تا یک چک باکس ایجاد کنید:

اولین آرگومان نام Checkbox می باشد ، دومین آرگومان مقدار Checkbox است و سومین آرگومان مقدار پیش فرض با قرار دادن True ویا False

```
<input type="checkbox" name="mycheck[]" value="1" <?php echo set_checkbox('mycheck[]', '1'); ?> />
<input type="checkbox" name="mycheck[]" value="2" <?php echo set_checkbox('mycheck[]', '2'); ?> />
```

set_radio();

برای ایجاد radio box به کار می رود:

```
<input type="radio" name="myradio" value="1" <?php echo set_radio('myradio', '1', TRUE); ?> />
<input type="radio" name="myradio" value="2" <?php echo set_radio('myradio', '2'); ?> />
```

کلاس های اف تی پی (**FTP Class**) :

کلاس های FTP ، Coideigniter به شما اجازه می دهند تا فایل های خود را از طریق اف تی پی انتقال دهید.

اف تی پی می تواند ایندا و پوشه ها را حرکت دهد ، مجدد نام گذاری کند و یا حتی حذف کند.

کلاس های اف تی پی همچنین شامل تابع های میرور هستند که به شما اجازه می دهند یک Directory از مسیر فایل های خود داخل اف تی پی ایجاد کنید.

توجه: کلاس های اف تی پی قادر به پشتیبانی از SSLFTP و یا SFTP نمی باشند.

پردازش کلاس (**Initializing the Class**) :

مانند اغلب کلاس ها در Codeigniter ، کلاس های FTP را باید بدین شیوه فراخانی کرد:

```
$this->load->library('ftp');
```

استفاده از یک مثال:

در این مثال ارتباط دهنده به سرور FTP باز است و متصل و در روی Local اسناد با فرمت ASCII خوانده می شوند و بارگزاری می شوند.

این اسناد باید دارای سطح دسترسی 755 باشند

توجه: سطح دسترسی فقط در PHP5 مورد استفاده است.

```
$this->load->library('ftp');

$config['hostname'] = 'ftp.example.com';
$config['username'] = 'your-username';
$config['password'] = 'your-password';
$config['debug'] = TRUE;

$this->ftp->connect($config);

$this->ftp->upload('/local/path/to/myfile.html', '/public_html/myfile.html', 'ascii', 0775);

$this->ftp->close();
```

این قطعه کد اسناد را از روی سرور بازیابی می کند:

```
$this->load->library('ftp');

$config['hostname'] = 'ftp.example.com';
$config['username'] = 'your-username';
$config['password'] = 'your-password';
$config['debug'] = TRUE;

$this->ftp->connect($config);

$list = $this->ftp->list_files('/public_html/');

print_r($list);

$this->ftp->close();
```

در این مثال مسیر لوکال میرور روی مسیر سرور است:

```
$this->load->library('ftp');

$config['hostname'] = 'ftp.example.com';
$config['username'] = 'your-username';
$config['password'] = 'your-password';
$config['debug'] = TRUE;

$this->ftp->connect($config);

$this->ftp->mirror('/path/to/myfolder/', '/public_html/myfolder/');

$this->ftp->close();
```

مرجع توابع (**Function Reference**) :

$this->ftp->connect();

این تابع به سرور FTP متصل می شود و Login می کند ، ابزارات اتصال توسط آرایه تنظیم می شوند و یا شما می توانید آنها را از داخل اسناد Config تغییر دهید.

```
$this->load->library('ftp');

$config['hostname'] = 'ftp.example.com';
$config['username'] = 'your-username';
$config['password'] = 'your-password';
$config['port']     = 21;
$config['passive']  = FALSE;
$config['debug']    = TRUE;

$this->ftp->connect($config);
```

برای تنظیم کردن مشخصات مسیر زیر را دنبال کنید:

Config/ftp.php

this->ftp->upload();

به کمک این تابع شما اطلاعات را روی سرور منتقل می کنید اما 2 نکته راباید رعایت کنید:

1- مسیر را به صورت صحیح وارد کنید
2- سطح دسترسی را به 0775 تنظیم کنید

$this->ftp->upload('/local/path/to/myfile.html', '/public_html/myfile.html', 'ascii', 0775);

$this->ftp->rename();

این امکان برای شما فراهم شده است ، این تابع 2 آرگمان را به عنوان ورودی دریافت می کند

1- مسیر فایل مورد نظر
2- نام تعویضی برای فایل مورد نظر

$this->ftp->move();

این تابع به شما اجازه می دهد که فایل ها را جابه جا کنید

مانند مثال:

```
// Moves blog.html from "joe" to "fred"
$this->ftp->move('/public_html/joe/blog.html', '/public_html/fred/blog.html');
```

$this->ftp->delete_file();

$this->ftp->delete_file('/public_html/joe/blog.html');

این تابع به شما اجازه می دهد مسیری را که به عنوان آرگومان دریافتی می پذیرد حذف کند.

$this->ftp->delete_dir();

$this->ftp->delete_dir('/public_html/path/to/folder/');

$this->ftp->list_files();

```
$list = $this->ftp->list_files('/public_html/');
print_r($list);
```

به شما اجازه می دهد که لیستی از فایل های داخل FTP را در قالب یک Array بر گردانید.

$this->ftp->mirror();

$this->ftp->mirror('/path/to/myfolder/', '/public_html/myfolder/');

$this->ftp->mkdir();

به شما اجازه می دهد یک پوشه را روی سرور ایجاد کنید.

// Creates a folder named "bar"
$this->ftp->mkdir('/public_html/foo/bar/', DIR_WRITE_MODE);

$this->ftp->chmod();

به شما اجازه می دهد سطح دسترسی ایجاد کنید.

// Chmod "bar" to 777
$this->ftp->chmod('/public_html/foo/bar/', DIR_WRITE_MODE);

$this->ftp->close();

این تابع اتصال به FTP را قطع می کند.

کلاس جدول (**HTML Table Class**) :

کلاس های جدول امکان ساخت جداول اچ تی ام ال را برای شما به وجود آورده اند

مانند اغلب کلاس ها در Codeigniter ابتدا کلاس مربوطه را فراخانی کنید.

$this->load->library('table');

مثال:

این یک مثال ساده است که به شما نشان می دهدکه چگونه جدول را به کمک آرایه های چند بعدی بسازید .

توجه: اولین آرایه که index است به عنوان Header در نظر گرفته می شود و شما می توانید به کمک تابع Set_header() ، Header را تعیین کنید.

```
$this->load->library('table');

$data = array(
        array('Name', 'Color', 'Size'),
        array('Fred', 'Blue', 'Small'),
        array('Mary', 'Red', 'Large'),
        array('John', 'Green', 'Medium')
        );

echo $this->table->generate($data);
```

مثال: تنظیم کردن Header و ستون های جدول:

```
$this->load->library('table');

$query = $this->db->query("SELECT * FROM my_table");

echo $this->table->generate($query);
```

مثال دیگر:

```
$this->load->library('table');

$this->table->set_heading('Name', 'Color', 'Size');

$this->table->add_row('Fred', 'Blue', 'Small');
$this->table->add_row('Mary', 'Red', 'Large');
$this->table->add_row('John', 'Green', 'Medium');

echo $this->table->generate();
```

مثالی دیگر:

```
$this->load->library('table');

$this->table->set_heading(array('Name', 'Color', 'Size'));

$this->table->add_row(array('Fred', 'Blue', 'Small'));
$this->table->add_row(array('Mary', 'Red', 'Large'));
$this->table->add_row(array('John', 'Green', 'Medium'));

echo $this->table->generate();
```

تغییر ظاهر جدول به کمک تابع (**Changing the Look of Your Table**) :

این تابع به شما اجازه می دهد تا ظاهر جدول را تغییر دهید:

```
$tmpl = array (
                'table_open'          => '<table border="0" cellpadding="4" cellspacing="0">',

                'heading_row_start'   => '<tr>',
                'heading_row_end'     => '</tr>',
                'heading_cell_start'  => '<th>',
                'heading_cell_end'    => '</th>',

                'row_start'           => '<tr>',
                'row_end'             => '</tr>',
                'cell_start'          => '<td>',
                'cell_end'            => '</td>',

                'row_alt_start'       => '<tr>',
                'row_alt_end'         => '</tr>',
                'cell_alt_start'      => '<td>',
                'cell_alt_end'        => '</td>',

                'table_close'         => '</table>'
          );
```

```
$this->table->set_template($tmpl);
```

$this->table->generate();

اطلاعات ساخت جدول به عنوان آرگومان به تابع پاس داده می شود و جدول ساخته می شود.

$this->table->set_caption();

به شما اجازه داده می شود به جدول خود عنوان اضافه کنید.

```
$this->table->set_caption('Colors');
```

$this->table->set_heading();

می توانید به جدول خود هدر اضافه کنید:

```
$this->table->set_heading('Name', 'Color', 'Size');

$this->table->set_heading(array('Name', 'Color', 'Size'));
```

$this->table->add_row();

می توانید به جدول یک سطر جدید اضافه کنید.

```
$this->table->add_row('Blue', 'Red', 'Green');

$this->table->add_row(array('Blue', 'Red', 'Green'));
```

$this->table->make_columns();

```
$list = array('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', 'ten', 'eleven', 'twelve');
$new_list = $this->table->make_columns($list, 3);
$this->table->generate($new_list);
// Generates a table with this prototype
<table border="0" cellpadding="4" cellspacing="0">
<tr>
<td>one</td><td>two</td><td>three</td>
</tr><tr>
```

```
<td>four</td><td>five</td><td>six</td>
</tr><tr>
<td>seven</td><td>eight</td><td>nine</td>
</tr><tr>
<td>ten</td><td>eleven</td><td>twelve</td></tr>
</table>
```

$this->table->set_template();

```
$tmpl = array ( 'table_open'  => '<table border="1" cellpadding="2" cellspacing="1" class="mytable">'
);
$this->table->set_template($tmpl);
```

$this->table->set_empty();

```
$this->table->set_empty(" ");
```

$this->table->clear();

```
$this->load->library('table');

$this->table->set_heading('Name', 'Color', 'Size');
$this->table->add_row('Fred', 'Blue', 'Small');
$this->table->add_row('Mary', 'Red', 'Large');
$this->table->add_row('John', 'Green', 'Medium');
echo $this->table->generate();
```

$this->table->clear();

```
$this->table->set_heading('Name', 'Day', 'Delivery');
$this->table->add_row('Fred', 'Wednesday', 'Express');
$this->table->add_row('Mary', 'Monday', 'Air');
```

```
$this->table->add_row('John', 'Saturday', 'Overnight');
echo $this->table->generate();
```

کلاس های پردازش تصویر (**Image Manipulation Class**) :

Codeigniter دارای کلاس هایی برای انجام عملیات زیر بر روی تصویر می باشد:

- تغییر اندازه تصویر
- ساخت تصویر کوچک
- بریدن عکس
- چرخاندن عکس
- افکت بر روی عکس

توجه: تمامی امکانات بالا از سوی کتابخانه های PHP از قبیل GD/GD2, NetPBM, and ImageMagick پشتیبانی می شوند.

مانند اغلب کلاس ها در Codeigniter ابتدا کلاس مربوطه را بدین شیوه فراخانی می کنیم:

```
$this->load->library('image_lib');
```

مثالی از پردازش عکس:

```
$config['image_library'] = 'gd2';
$config['source_image'] = '/path/to/image/mypic.jpg';
$config['create_thumb'] = TRUE;
$config['maintain_ratio'] = TRUE;
$config['width'] = 75;
$config['height'] = 50;

$this->load->library('image_lib', $config);

$this->image_lib->resize();
```

پردازش توابع (**Processing Functions**) :

- $this->image_lib->resize()
- $this->image_lib->crop()
- $this->image_lib->rotate()

- $this->image_lib->watermark()
- $this->image_lib->clear()

در این توابع اگر فرآیندی با شکست مواجه شود مقدار False را بر می گرداند که برای نشان دادن Error در برنامه ها به کار می رود.

```
echo $this->image_lib->display_errors();
```

به عنوان مثال:

```
if ( ! $this->image_lib->resize()){
    echo $this->image_lib->display_errors();
}
```

شما همچنین می توانید یک ا را بر اساس تگ های HTML بسازید

مانند:

```
$this->image_lib->display_errors('<p>', '</p>');
```

$this->image_lib->resize();

تابعی برای تغییر حجم و اندازه تصویر

```
$config['create_thumb'] = TRUE;
```

ساخت تصویر کوچک با نام thumb ، در صورت False بودن عمل ساختن تصویر کوچک با نام مورد نظر امکان پذیر نخواهد بود.

```
$config['new_image'] = '/path/to/new_image.jpg';
```

ساخت کپی از تصویر با اندازه کوچک تر در مسیری که توسط این تابع تعیین می شود.

$this->image_lib->crop();

```
$config['image_library'] = 'imagemagick';
$config['library_path'] = '/usr/X11R6/bin/';
$config['source_image'] = '/path/to/image/mypic.jpg';
$config['x_axis'] = '100';
$config['y_axis'] = '60';
$this->image_lib->initialize($config);
```

```
if ( ! $this->image_lib->crop()){
    echo $this->image_lib->display_errors();
}
```

$this->image_lib->rotate();

```
$config['image_library'] = 'netpbm';
$config['library_path'] = '/usr/bin/';
$config['source_image'] = '/path/to/image/mypic.jpg';
$config['rotation_angle'] = 'hor';

$this->image_lib->initialize($config);

if ( ! $this->image_lib->rotate())
{
    echo $this->image_lib->display_errors();
}
```

$this->image_lib->clear();

کلیه اعمال انجام گرفته بر روی تصویر حذف می شوند.

Image Watermarking;

مثالی از افکت بر روی تصویر:

```
$config['source_image'] = '/path/to/image/mypic.jpg';
$config['wm_text'] = 'Copyright 2006 - John Doe';
$config['wm_type'] = 'text';
$config['wm_font_path'] = './system/fonts/texb.ttf';
$config['wm_font_size'] = '16';
$config['wm_font_color'] = 'ffffff';
$config['wm_vrt_alignment'] = 'bottom';
$config['wm_hor_alignment'] = 'center';
$config['wm_padding'] = '20';

$this->image_lib->initialize($config);
```

```
$this->image_lib->watermark();
```

کلاس های ورودی (**Input Class**) :

کلاس های ورودی 2 هدف را دنبال می کنند

- پردازش اطلاعات ورودی برای حفظ امنیت
- فراهم آوردن تعدادی تابع های آماده و کمک کننده برای واکشی اطلاعات خروجی و پردازش آن ها

فیلتر کردن (**Security Filtering**) :

تابع های امنیتی عمل فیلترینگ را به صورت خودکار هنگامی که یک کلاس کنترلر فراخانی می شوند انجام می دهند.

- Codeigniter هیچ دلیل نمی بیند که اجازه دهد متغیر GET اجرا شود
- هنگامی که Register_globals روشن باشد تمام متغیر های Global خراب می شوند
- مقادیر متغیر های POST و Cookie فیلتر می شوند فقط عدد و حروف اجازه ورود دارند
- محیط فیلترینگ XSS را فراهم می کند

XSS Filtering;

$this->input->xss_clean();

```
$data = $this->input->xss_clean($data);
```

در صورتی که تمایل دارید این امر به شکل خودکار انجام شود می توانید از طریق مسیر

application/config/config.php

مقدار متغیر را به True تبدیل کنید.

```
$config['global_xss_filtering'] = TRUE;
```

این قطعه کد بسیار به شما در امنیت آپلود تصویر کمک می کندبه طوری که اگر مقدار True برگردانده شود تصویر سالم است.

```
if ($this->input->xss_clean($file, TRUE) === FALSE)
{
```

```
    // file failed the XSS test
}
```

Using POST, COOKIE, or SERVER Data;

قدیمی:

```
if ( ! isset($_POST['something'])){
    $something = FALSE;
}
else
{
    $something = $_POST['something'];
}
```

جدید:

```
$something = $this->input->post('something');
```

$this->input->post();

```
$this->input->post('some_data');
```

در صورت تنظیم شدن دومین آرگومان تابع به True وروردی XSS Filter می شود.

```
$this->input->post('some_data', TRUE);
```

$this->input->get();

```
$this->input->get('some_data', TRUE);
```

$this->input->get_post();

$this->input->get_post('some_data', TRUE);

$this->input->cookie()

$this->input->cookie('some_data', TRUE);

$this->input->server();

$this->input->server('some_data');

$this->input->ip_address();

echo $this->input->ip_address();

$this->input->valid_ip($ip);

```
if ( ! $this->input->valid_ip($ip))
{
    echo 'Not Valid';
}
else
{
    echo 'Valid';
}
```

$this->input->user_agent();

echo $this->input->user_agent();

کلاس های ایجاد صفحه بندی (**Pagination Class**) :

Codeigniter دارای یک کلاس می باشد که به راحتی می توانید صفحه گذاری را ایجاد کنید و می توانید 100% آن را اختصاص سازی کنید و از امکاناتش بهره برداری کنید.

کلاس های صفحه گذاری به این شکل می باشد:

« First < 1 2 **3** 4 5 > Last »

مثال: این جا یک مثال است که به شما چگونگی استفاده از این کلاس را می آموزید:

```
$this->load->library('pagination');

$config['base_url'] = 'http://example.com/index.php/test/page/';
$config['total_rows'] = '200';
$config['per_page'] = '20';

$this->pagination->initialize($config);

echo $this->pagination->create_links();
```

اختصاص دادن توضیحات بیشتر از حوصله این مقاله خارج است برای اطلاعات بیشتر می توانید به آدرس زیر مراجعه کنید:

http:// CodeIgniter.com/user_guide/libraries/pagination.html

کلاس های سشن (**Session Class**) :

کلاس های سشن این امکان را برای شما فراهم می کنند تا از موقعیت کاربران اطلاع حاصل کنید.

این کلاس ها اطلاعات کاربران را درون خود ذخیره می کنند ، شما می توانید اطلاعات جمع آوری شده از طریق جلسات کاربری و یا همان سشن را برای امنیت بیشتر درون پایگاه داده ذخیره کنید.

ابتدا کلاس مربوطه را فراخانی می کنیم:

```
$this->load->library('session');
```

زمانی که صفحه لود می شود کلاس سشن ، چک می شود که آیا اطلاعات جلسات کاربری معتبر وجود دارد یا خیر ، در صورتی که اطلاعاتی وجود نداشت و یا تاریخ جلسات کاربری انقضا یافته بود یک متغیر جلسه کاربری جدید ایجاد می شود و آن را داخل Cookie ذخیره می کند ، اما اگر جلسات کاربری وجود داشت اطلاعات آن به روز رسانی می شود.

با هر به روز رسانی session_id مجدد ساخته می شود.

این مطلب بسیار با ارزش است که متوجه شوید کلاسات جلسات کاربری به صورت خودکار لود می شوند و نیاز به هیچ فراخانی ندارند.

اطلاعات جلسات کاربری چیست (**What is Session Data?**) :

- Session_id واحد برای هر کاربر
- Ip address هر کاربر
- 50 کلمه اول اطلاعات مرورگر کاربر
- تاریخ و زمان آخرین فعالیت در سایت

```
[array]
(
    'session_id'    => random hash,
    'ip_address'    => 'string - user IP address',
    'user_agent'    => 'string - user agent data',
```

```
    'last_activity' => timestamp
)
```

بازیابی اطلاعات سشن (**Retrieving Session Data**) :

شما به وسیله این اطلاعات می توانید اطلاعات را بازیابی کنید.

```
$this->session->userdata('item');
```

هر جا که item قرار دارد می توانید تابع مورد نظر را جا سازی کنید.

```
$session_id = $this->session->userdata('session_id');
```

در صورتی که مقدار مورد نظر اضافه نشده باشد مقدار False برگردانده می شود.

حذف متغیر جلسه ای (**Removing Session Data**) :

```
$this->session->unset_userdata('some_name');
```

می توانید برای حذف جلسات کاربری به جای some_name نام قسمت مورد نظر را جاگزاری کنید مثل: session_id

```
$array_items = array('username' => '', 'email' => '');
$this->session->unset_userdata($array_items);
```

خراب کردن جلسات کاربری (**Destroying a Session**) :

```
$this->session->sess_destroy();
```

کمک کننده ها (**Helper**) :

کمک کننده زمان (**Date Helper**) :

ابتدا باید کلاس مربوطه را فراخانی کنید.

```
$this->load->helper('date');
```

now();

برچسب زمانی هم اکنون بر روی سرور را بر اساس ساعت گرینویچ بر می گرداند.

در صورتی که خواستار نمایش ساعت در مکان جغرافیایی خود می باشید از این تابع بهره نخواهید برد.

mdate();

این تابع یکی از توابع زمان PHP است کهبه شما اجازه می دهد از آرایش زمان Mysql استفاده کنید هر جا مشابه کد روبه رو وجود داست %Y %m %d

مزایای استفاده از این کد این است که دیگر هیچ نگرانی از Escape نشدن کارکترهای ندارید :

```
$datestring = "Year: %Y Month: %m Day: %d - %h:%i %a";
$time = time();
echo mdate($datestring, $time);
```

توابع استاندارد:

standard_date();

به شما اجازه می دهد تا تاریخ را با فرمت مورد نظر خود بسازید.

```
$format = 'DATE_RFC822';
$time = time();
echo standard_date($format, $time);
```

Constant	Description	Example
DATE_ATOM	Atom	2005-08-15T16:13:03+0000
DATE_COOKIE	HTTP Cookies	Sun, 14 Aug 2005 16:13:03 UTC
DATE_ISO8601	ISO-8601	2005-08-14T16:13:03+0000
DATE_RFC822	RFC 822	Sun, 14 Aug 2005 16:13:03 UTC
DATE_RFC850	RFC 850	Sunday, 14-Aug-05 16:13:03 UTC
DATE_RFC1036	RFC 1036	Sunday, 14-Aug-05 16:13:03 UTC
DATE_RFC1123	RFC 1123	Sun, 14 Aug 2005 16:13:03 UTC
DATE_RFC2822	RFC 2822	Sun, 14 Aug 2005 16:13:03 +0000
DATE_RSS	RSS	Sun, 14 Aug 2005 16:13:03 UTC
DATE_W3C	World Wide Web Consortium	2005-08-14T16:13:03+0000

local_to_gmt();

بر اساس این تابع برچسب زمانی بر اساس unix به دست می آورید.

که نحوه استفاده از آن مثل زیر است:

```
$now = time();
$gmt = local_to_gmt($now);
```

gmt_to_local();

```
$timestamp = '1140153693';
$timezone = 'UM8';
$daylight_saving = TRUE;
echo gmt_to_local($timestamp, $timezone, $daylight_saving);
```

Time Zone	Location
UM12	(UTC - 12:00) Enitwetok, Kwajalien
UM11	(UTC - 11:00) Nome, Midway Island, Samoa
UM10	(UTC - 10:00) Hawaii
UM9	(UTC - 9:00) Alaska
UM8	(UTC - 8:00) Pacific Time
UM7	(UTC - 7:00) Mountain Time
UM6	(UTC - 6:00) Central Time, Mexico City
UM5	(UTC - 5:00) Eastern Time, Bogota, Lima, Quito
UM4	(UTC - 4:00) Atlantic Time, Caracas, La Paz
UM25	(UTC - 3:30) Newfoundland
UM3	(UTC - 3:00) Brazil, Buenos Aires, Georgetown, Falkland Is.
UM2	(UTC - 2:00) Mid-Atlantic, Ascention Is., St Helena
UM1	(UTC - 1:00) Azores, Cape Verde Islands
UTC	(UTC) Casablanca, Dublin, Edinburgh, London, Lisbon, Monrovia
UP1	(UTC + 1:00) Berlin, Brussels, Copenhagen, Madrid, Paris, Rome
UP2	(UTC + 2:00) Kaliningrad, South Africa, Warsaw
UP3	(UTC + 3:00) Baghdad, Riyadh, Moscow, Nairobi
UP25	(UTC + 3:30) Tehran
UP4	(UTC + 4:00) Adu Dhabi, Baku, Muscat, Tbilisi
UP35	(UTC + 4:30) Kabul
UP5	(UTC + 5:00) Islamabad, Karachi, Tashkent
UP45	(UTC + 5:30) Bombay, Calcutta, Madras, New Delhi
UP6	(UTC + 6:00) Almaty, Colomba, Dhaka
UP7	(UTC + 7:00) Bangkok, Hanoi, Jakarta
UP8	(UTC + 8:00) Beijing, Hong Kong, Perth, Singapore, Taipei

UP9	(UTC + 9:00) Osaka, Sapporo, Seoul, Tokyo, Yakutsk
UP85	(UTC + 9:30) Adelaide, Darwin
UP10	(UTC + 10:00) Melbourne, Papua New Guinea, Sydney, Vladivostok
UP11	(UTC + 11:00) Magadan, New Caledonia, Solomon Islands
UP12	(UTC + 12:00) Auckland, Wellington, Fiji, Marshall Island

mysql_to_unix();

برچسب زمان mysql را می گیرد به عنوان ورودی و بر اساس unix تحویل می دهد.

```
$mysql = '20061124092345';
$unix = mysql_to_unix($mysql);
```

unix_to_human();

این تابع ورودی بر اساس برچسب زمانی بر اساس unix میگیرد و بر اساس تاریخ قابل فهم برای انسان تحویل می دهد.

اگر شما می خواهید در هنگام ارسال اطلاعات فرم ، از زمان استفاده کنید ارسال به این روش بسیار سود اور خواهد بود.

YYYY-MM-DD HH:MM:SS AM/PM

```
$now = time();
echo unix_to_human($now); // U.S. time, no seconds
echo unix_to_human($now, TRUE, 'us'); // U.S. time with seconds
echo unix_to_human($now, TRUE, 'eu'); // Euro time with seconds
```

human_to_unix();

تبدیل تاریخ قابل فهم برای انسان به unix

```
$now = time();
$human = unix_to_human($now);
$unix = human_to_unix($human);
```

timespan();

زمان unix را مانند مثال زیر فرمت می دهد.

```
$post_date = '1079621429';
$now = time();

echo timespan($post_date, $now);
```

days_in_month();

عدد ، روز تاریخ ورودی را بر می گرداند.

```
echo days_in_month(06, 2005);
```

اگر پارامتر دوم تابع را خالی بگزارید سال فعلی را به عنوان منبع تغزیه استفاده می کند.

timezones();

زمان محلی را بر اساس جدول زمان بندی بر می گزداند.

```
echo timezones('UM5');
```

timezone_menu();

این تابع برای وب سایت هایی که عمل ثبت نام انجام می دهند بسیار مفید است زیرا زمان user local را بر می گرداند.

- پارامتر اول نشان دهنده زمان
- پارامتر دوم که برای تنظیم استایل به کار می رود می توانید به آن کلاس CSS متصل کنید

کلاس کمک کننده ایمیل (Email Helper) :

فراخانی کلاس:

```
$this->load->helper('email');
```

مثال:

```
$this->load->helper('email');
if (valid_email('email@somesite.com')){
```

```
    echo 'email is valid';
}
else{
    echo 'email is not valid';
}
```

send_email('recipient', 'subject', 'message')

مقصد ، موضوع ، متن

کمک کننده فرم (**Form Helper**) :

فراخانی تابع:

```
$this->load->helper('form');
```

form_open();

به جای نوشتن تگ فرم استفاده می شود

```
echo  form_open('email/send');
```

اطلاعات فرم را به کلاس email و تابع send ارسال کن

```
<form method="post" action="http:/example.com/index.php/email/send" />
```

اضافه کردن مشخصات بیشتر (**Adding Attributes**) :

```
$attributes = array('class' => 'email', 'id' => 'myform');
echo form_open('email/send', $attributes);
```

مانند:

```
<form method="post" action="http:/example.com/index.php/email/send"  class="email"  id="myform" />
```

اضافه کردن فیلد پنهان (**Adding Hidden Input Fields**) :

```
$hidden = array('username' => 'Joe', 'member_id' => '234');
echo form_open('email/send', '', $hidden);
```

مانند:

```
<form method="post" action="http:/example.com/index.php/email/send">
<input type="hidden" name="username" value="Joe" />
<input type="hidden" name="member_id" value="234" />
```

form_open_multipart();

تگ فرم مناسب برای انتقال اسناد

form_hidden();

```
form_hidden('username', 'johndoe');
// Would produce:
<input type="hidden" name="username" value="johndoe" />
```

اضافه کردن چندین فیلد پنهان:

```
$data = array(
              'name'  => 'John Doe',
              'email' => 'john@example.com',
              'url'   => 'http://example.com'
            );
echo form_hidden($data);
// Would produce:
<input type="hidden" name="name" value="John Doe" />
<input type="hidden" name="email" value="john@example.com" />
<input type="hidden" name="url" value="http://example.com" />
```

form_input();

فیلد ورودی:

```
echo form_input('username', 'johndoe');
```

چندین فرم ورودی:

```
$data = array(
              'name'        => 'username',
              'id'          => 'username',
              'value'       => 'johndoe',
```

```
'maxlength'   => '100',
'size'        => '50',
'style'       => 'width:50%',
);
```

```
echo form_input($data);
// Would produce:
<input type="text" name="username" id="username" value="johndoe" maxlength="100" size="50" style="width:50%" />
```

اضافه کردن رخداد های جاوا اسکریپت:

```
$js = 'onClick="some_function()"';
echo form_input('username', 'johndoe', $js);
```

form_password();

مشابه فرم های ورودی عمل می کنند:

form_upload();

مشابه فرم های ورودی عمل می کنند:

form_textarea();

مشابه فرم های ورودی عمل می کنند:

form_dropdown();

```
$options = array(
            'small'  => 'Small Shirt',
            'med'    => 'Medium Shirt',
            'large'  => 'Large Shirt',
            'xlarge' => 'Extra Large Shirt',
          );

$shirts_on_sale = array('small', 'large');

echo form_dropdown('shirts', $options, 'large');
```

```
// Would produce:

<select name="shirts">
<option value="small">Small Shirt</option>
<option value="med">Medium Shirt</option>
<option value="large" selected="selected">Large Shirt</option>
<option value="xlarge">Extra Large Shirt</option>
</select>

echo form_dropdown('shirts', $options, $shirts_on_sale);

// Would produce:
<select name="shirts" multiple="multiple">
<option value="small" selected="selected">Small Shirt</option>
<option value="med">Medium Shirt</option>
<option value="large" selected="selected">Large Shirt</option>
<option value="xlarge">Extra Large Shirt</option>
</select>
```

اطلاعات بیشتر:

```
$js = 'id="shirts" onChange="some_function();"';
echo form_dropdown('shirts', $options, 'large', $js);
```

form_fieldset();

```
echo form_fieldset('Address Information');
echo "<p>fieldset content here</p>\n";
echo form_fieldset_close();

// Produces
<fieldset>
<legend>Address Information</legend>
```

```
<p>form content here</p>
</fieldset>
```

دومین اطلاعات شامل اطلاعات اضافه است:

```
$attributes = array('id' => 'address_info', 'class' => 'address_info');
echo form_fieldset('Address Information', $attributes);
echo "<p>fieldset content here</p>\n";
echo form_fieldset_close();



// Produces
<fieldset id="address_info" class="address_info">
<legend>Address Information</legend>
<p>form content here</p>
</fieldset>
```

form_fieldset_close();

برای بستن تک فیلدست استفاده می شود.

```
$string = "</div></div>";
echo fieldset_close($string);
// Would produce:
</fieldset>
</div></div>;
```

form_checkbox();

```
echo form_checkbox('newsletter', 'accept', TRUE);
// Would produce:
<input type="checkbox" name="newsletter" value="accept" checked="checked" />
```

در صورت تنظیم مقدار True مقدار مورد نظر پیش فرض انتخاب می شود.

دادن امکانات بیشتر به وسیله آرایه امکان پذیر است.

```
$data = array(
    'name'        => 'newsletter',
    'id'          => 'newsletter',
    'value'       => 'accept',
    'checked'     => TRUE,
    'style'       => 'margin:10px',
    );

echo form_checkbox($data);

// Would produce:
<input type="checkbox" name="newsletter" id="newsletter" value="accept" checked="checked" style="margin:10px" />


$js = 'onClick="some_function()"';
echo form_checkbox('newsletter', 'accept', TRUE, $js)
```

form_radio();

مانند تگ چک باکس عمل می نماید.

form_submit();

```
echo form_submit('mysubmit', 'Submit Post!');
// Would produce:
<input type="submit" name="mysubmit" value="Submit Post!" />
```

form_label();

مانند تگ submit عمل می نماید.

```
echo form_label('What is your Name', 'username');
// Would produce:
<label for="username">What is your Name</label>
```

چندین تگ لیبل:

```
$attributes = array(
    'class' => 'mycustomclass',
    'style' => 'color: #000;',
);
echo form_label('What is your Name', 'username', $attributes);

// Would produce:
<label for="username" class="mycustomclass" style="color: #000;">What is your Name</label>
```

form_reset();

مانند تگ submit عمل می نماید.

form_button();

```
echo form_button('name','content');
// Would produce
<button name="name" type="button">Content</button>
```

مثالی دیگر:

```
$data = array(
    'name' => 'button',
    'id' => 'button',
    'value' => 'true',
    'type' => 'reset',
    'content' => 'Reset'
);
echo form_button($data);

// Would produce:
<button name="button" id="button" value="true" type="reset">Reset</button>
```

مثالی دیگر:

```
$js = 'onClick="some_function()"';
echo form_button('mybutton', 'Click Me', $js);
```

form_close();

```
$string = "</div></div>";
echo form_close($string);

// Would produce:

</form>
</div></div>
```

form_prep();

این تابع در صورتی که بخواهید عبارت را از کسی نقل کنید بسیار مفید خواهد بود.

```
$string = 'Here is a string containing "quoted" text.';
<input type="text" name="myform" value="$string" />
```

روش دیگر:

```
<input type="text" name="myform" value="<?php echo form_prep($string); ?>" />;
```

set_value();

به شما امکان می دهد که مقداری برای اسناد ورودی برگزینید در صورتی که مقدار پارامتر دوم تابع تنظیم شود مقدار پیش فرض خواهد بود.

```
<input type="text" name="quantity" value="<?php echo set_value('quantity', '0'); ?>" size="50" />
```

set_select();

اگر شما از منو select استفاده می کنید این تابع به شما اجازه می دهد که آیتم های منو را نشان دهید.

اولین پارامتر نام menu است و دومین پارامتر باید دارای ارزش مورد نظر باشد. اگر مقدار پارامتر سوم True تنظیم شود مقدار پیش فرض برای آن تعیین می شود.

```
<select name="myselect">
<option value="one" <?php echo set_select('myselect', 'one', TRUE); ?> >One</option>
<option value="two" <?php echo set_select('myselect', 'two'); ?> >Two</option>
<option value="three" <?php echo set_select('myselect', 'three'); ?> >Three</option>
</select>
```

set_checkbox();

```
<input type="checkbox" name="mycheck" value="1" <?php echo set_checkbox('mycheck', '1'); ?> />
<input type="checkbox" name="mycheck" value="2" <?php echo set_checkbox('mycheck', '2'); ?> />
```

set_radio();

```
<input type="radio" name="myradio" value="1" <?php echo set_radio('myradio', '1', TRUE); ?> />
<input type="radio" name="myradio" value="2" <?php echo set_radio('myradio', '2'); ?> />
```

کمک کننده تگ های اچ تی ام ال (**HTML Helper**) :

فراخانی کلاس مورد نظر:

```
$this->load->helper('html');
```

br();

ایجاد break در صفحات

```
echo br(3);
```

Result:

heading();

ایجاد تگ هدینگ

```
echo heading('Welcome!', 3);
```

پارامتر دوم درجه بزرگی و یا کوچکی تگ هدر می باشد.

img();

```
echo img('images/picture.jpg');
// gives <img src="http://site.com/images/picture.jpg" />
```

در صورت تنظیم True برای آرگومان دوم تابع مسیر مورد نظر برای تصویر شبیه سازی می شود.

```
echo img('images/picture.jpg', TRUE);
// gives <img src="http://site.com/index.php/images/picture.jpg" />
```

می توانید به وسیله آرایه اطلاعات تکمیلی به تگ تصویر اضافه کنید:

```
$image_properties = array(
        'src' => 'images/picture.jpg',
        'alt' => 'Me, demonstrating how to eat 4 slices of pizza at one time',
        'class' => 'post_images',
        'width' => '200',
        'height' => '200',
        'title' => 'That was quite a night',
        'rel' => 'lightbox',
);

img($image_properties);
// <img src="http://site.com/index.php/images/picture.jpg" alt="Me, demonstrating how to eat 4 slices
of pizza at one time" class="post_images" width="200" height="200" title="That was quite a night"
rel="lightbox" />
```

link_tag();

```
echo link_tag('css/mystyles.css');
// gives <link href="http://site.com/css/mystyles.css" rel="stylesheet" type="text/css" />
```

مثال های بیشتر:

```
echo link_tag('favicon.ico', 'shortcut icon', 'image/ico');
// <link href="http://site.com/favicon.ico" rel="shortcut icon" type="image/ico" />

echo link_tag('feed', 'alternate', 'application/rss+xml', 'My RSS Feed');
// <link href="http://site.com/feed" rel="alternate" type="application/rss+xml" title="My RSS Feed" />
```

اضافه کردن اطلاعات تکمیلی به کمک آرایه:

```
$link = array(
        'href' => 'css/printer.css',
        'rel' => 'stylesheet',
        'type' => 'text/css',
        'media' => 'print'
);

echo link_tag($link);
// <link href="http://site.com/css/printer.css" rel="stylesheet" type="text/css" media="print" />
```

nbs();

پیاده سازی ( ) به کمک nbs() صورت می گیرد.

```
echo nbs(3);
```

ورودی تابه تعداد تگ مورد نظر می باشد.

ol() and ul();

پیاده سازی لیست

```
$this->load->helper('html');

$list = array(
```

```
        'red',
        'blue',
        'green',
        'yellow'
        );

$attributes = array(
              'class' => 'boldlist',
              'id'    => 'mylist'
              );

echo ul($list, $attributes);
```

مشابه:

```
<ul class="boldlist" id="mylist">
  <li>red</li>
  <li>blue</li>
  <li>green</li>
  <li>yellow</li>
</ul>
```

meta();

پدید آورنده تگ متا

```
echo meta('description', 'My Great site');
// Generates: <meta name="description" content="My Great Site" />

echo meta('Content-type', 'text/html; charset=utf-8', 'equiv'); // Note the third parameter. Can be "equiv"
or "name"
// Generates: <meta http-equiv="Content-type" content="text/html; charset=utf-8" />

echo meta(array('name' => 'robots', 'content' => 'no-cache'));
// Generates: <meta name="robots" content="no-cache" />

$meta = array(
      array('name' => 'robots', 'content' => 'no-cache'),
      array('name' => 'description', 'content' => 'My Great Site'),
```

```
        array('name' => 'keywords', 'content' => 'love, passion, intrigue, deception'),
        array('name' => 'robots', 'content' => 'no-cache'),
        array('name' => 'Content-type', 'content' => 'text/html; charset=utf-8', 'type' => 'equiv')
    );

echo meta($meta);
// Generates:
// <meta name="robots" content="no-cache" />
// <meta name="description" content="My Great Site" />
// <meta name="keywords" content="love, passion, intrigue, deception" />
// <meta name="robots" content="no-cache" />
// <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
```

doctype();

```
echo doctype();
// <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

echo doctype('html4-trans');
// <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

Doctype	Option	Result
XHTML 1.1	doctype('xhtml11')	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
XHTML 1.0 Strict	doctype('xhtml1-strict')	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
XHTML 1.0 Transitional	doctype('xhtml1-trans')	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
XHTML 1.0 Frameset	doctype('xhtml1-frame')	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
HTML 5	doctype('html5')	<!DOCTYPE html>
HTML 4 Strict	doctype('html4-strict')	<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
HTML 4 Transitional	doctype('html4-trans')	<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
HTML 4 Frameset	doctype('html4-frame')	<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

کمک کننده لینک (**URL Helper**) :

فراخانی کلاس مربوطه:

```
$this->load->helper('url');
```

site_url();

آدرس کامل کلاس ورودی را بر می گرداند.

```
echo site_url("news/local/123");
```

روشی دیگر:

```
$segments = array('news', 'local', '123');
echo site_url($segments);
```

base_url();

```
echo base_url();
```

current_url();

```
echo current_url();
```

uri_string();

این تابع در صورتی که ورودی آن یک آدرس URL باشد فقط قسمت مربوط کلاس ها و توابع را بر می گرداند.

http://some-site.com/blog/comments/123;

خروجی:

```
/blog/comments/123
```

index_page();

صفحه پیش فرض را که در سند Config تنظیم شده است بر می گرداند.

```
echo index_page();
```

anchor();

ساخت تگ <a>

```
<a href="http://example.com">Click Here</a>
```

anchor(**uri segments**, **text**, **attributes**)

مثال:

```
echo anchor('news/local/123', 'title="My News"');
```

مثال:

```
echo anchor('news/local/123', 'My News', array('title' => 'The best news!'));
```

anchor_popup();

ساخت صفحات popup

```
$atts = array(
              'width'      => '800',
              'height'     => '600',
              'scrollbars' => 'yes',
              'status'     => 'yes',
              'resizable'  => 'yes',
              'screenx'    => '0',
              'screeny'    => '0'
            );

echo anchor_popup('news/local/123', 'Click Me!', $atts);
```

mailto();

```
echo mailto('me@my-site.com', 'Click Here to Contact Me');
```

safe_mailto();

این تابع از spam شدن ایمیل شما جلوگیری می کند.

auto_link();

این تابع به صورت خودکار صفحه را به آدرس رشته ورودی می فرستد.

```
$string = auto_link($string);
```

url_title();

این تابع در صورتی که به آن رشته ای را به عنوان خروجی به آن ارسال کنید رشته را به صورت رشته امن تبدیل می کند

```
$title = "What's wrong with CSS?";

$url_title = url_title($title);

// Produces: Whats-wrong-with-CSS
```

می توانید با تنظیم کردن پارامتر دوم تابع مثل قرار دادن underscore به تابع بگویید جای خالی را با underscore پر کند.

```
$title = "What's wrong with CSS?";

$url_title = url_title($title, 'underscore');

// Produces: Whats_wrong_with_CSS
```

مثالی دیگر:

```
$title = "What's wrong with CSS?";

$url_title = url_title($title, 'underscore', TRUE);

// Produces: whats_wrong_with_css
```

prep_url();

این تابع به ابتدای آدرس ورودی تابع http:// اضافه می نماید.

```
$url = "example.com";
$url = prep_url($url);
```

redirect();

این تابع به شما اجازه می دهد تا عمل انتقال را انجام دهید

پارامتر دوم تابع می تواند به refresh هم تغییر پیدا کند اما درصورتی که تنظیم نشود مقدار پیش فرض location قرار می گیرد.

```
if ($logged_in == FALSE)
{
    redirect('/login/form/', 'refresh');
}

// with 301 redirect
redirect('/article/13', 'location', 301);
```

(موفق و پیروز باشید)