

# CakePHP

A PHP Framework

A series of horizontal lines in teal and light blue colors, located on the right side of the slide, extending from the left edge of the slide.

# CakePHP

- A framework for developing applications in PHP
- Inspired by Ruby on Rails
- Follows MVC design pattern
- Convention over configuration
  - No wheel reinventing required!

# MVC

- Model
  - Data layer
- View
  - Presentation layer
- Controller
  - Logic layer

# CakePHP Framework

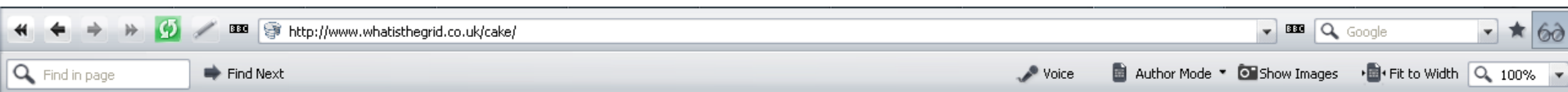
- app/
  - config/
  - controllers/
  - models/
  - plugins/
  - tmp/
  - vendors/
  - views/
  - webroot/
- cake/
  - config/
  - docs/
  - libs/
- vendors/

# Naming conventions

- <http://book.cakephp.org/view/328/Cake-Conventions>
- Table names: “notes”, “my\_notes”
- Model: “mynote.php”->“MyNote”
- Controller: “my\_notes\_controller.php”->“MyNotesController”
- Views named after actions, organised in folders according to the related controller:
  - `views/my_notes/index.html`
  - `views/my_notes/add.html`

# Paths + parameters

- Cake uses url to pass parameters
- Apache mod\_rewrite converts url into scriptname and parameters
- `http://www.example.com`  
`/controllername/action/param1/param2/...`
- Uses paths to figure out views
- Views stored in “controllername” folder



## CakePHP Rapid Development

Your database configuration file is not present.

### CakePHP release information is on CakeForge

[Read the release notes and get the latest version](#)

### Editing this Page

To change the content of this page, create: /app/Views/pages/home.html.

To change its layout, create: /app/Views/layouts/default.html.

[See the views section of the manual for more info](#)

You can also add some CSS styles for your pages at: app/webroot/css/.

### More about Cake

CakePHP is a rapid development framework for PHP which uses commonly known design patterns like Active Record, Association Data Mapping, Front Controller and MVC. Our primary goal is to provide a structured framework that enables PHP users at all levels to rapidly develop robust web applications, without any loss to flexibility.

- [Cake Software Foundation](#)
  - Promoting development related to CakePHP
- [The Bakery](#)
  - Everything CakePHP
- [Book Store](#)
  - Recommended Software Books
- [CakeSchwag](#)
  - Get your own CakePHP gear - Doughnate to Cake
- [CakePHP](#)
  - The Rapid Development Framework
- [CakePHP Manual](#)
  - Your Rapid Development Cookbook
- [CakePHP API](#)
  - Docblock Your Best Friend
- [CakeForge](#)

# OOP in PHP

- Limited support in PHP <5
- Much better support in PHP >=5
- Simpler than Java OOP

```
class SomeClass {  
    function func() {  
        ....  
    }  
}  
  
SomeClass s = new someClass();  
s->func();
```



# Hello world... again

- Remember application is separated into model / view / controller
- Model:

```
<?php
/* /app/model/hello.php */
class Hello extends AppModel {
    var $name      = 'Hello';
    var $useTable = false;
}
?>
```

# Hello world... again

- View:

```
<!--
```

```
/* /app/views/index.html */
```

```
-->
```

```
<hr size=1/>
```

```
<h1><?php echo $data ?></h1>
```

```
<hr size=1/>
```

- Controller:

```
<?php
/* app/controller/hello_controller.php */
class HelloController extends ApplicationController {
    var $name = "Hello";
    var $uses = 'Hello';
    function index() {
        $data = 'Hello world!';
        $this->set('data', $data);
    }
}
?>
```

Navigation bar with back, forward, and search buttons. The address bar shows <http://www.whatisthegrid.co.uk/cake/hello>. The search bar contains the text "Google".

## CakePHP Rapid Development

Hello world!

# Simple DB table app

- An online contact list
- We want to add, edit, view and delete names and phone numbers
- Uses a single table

# Model

- Add table to DB:

```
CREATE TABLE cake_contacts (  
    id INT UNSIGNED AUTO_INCREMENT  
    PRIMARY KEY,  
    name VARCHAR(50),  
    number VARCHAR(50),  
    created DATETIME DEFAULT NULL,  
    modified DATETIME DEFAULT NULL  
);
```

# Model

- Add a script called contact.php to models/

```
<?php
```

```
class Contact extends AppModel
```

```
{
```

```
    var $name = 'Contact';
```

```
}
```

```
?>
```

# View

- views/contacts/index.html

```
<h1>Contact list</h1>
```

```
<p>
```

```
<?php echo $html->link('Add Contact',  
    'contacts/add') ?>
```

```
</p>
```

```
<table>
```

```
    <tr>
```

```
        <th>Id</th>
```

```
        <th>Name</th>
```

```
        <th>Number</th>
```

```
    </tr>
```



# View

- views/contacts/index.html cntd...

```
<?php foreach ($contacts as $contact): ?>
  <tr>
    <td><?php echo $contact['Contact']['id']; ?></td>
    <td>
      <?php
        echo $html->link($contact['Contact']['name'],
          "contacts/view/{$_contact['Contact']['id']}")?>
        [<?php echo $html->link('Edit',
          "contacts/edit/{$_contact['Contact']['id']}")?>,
        <?php echo $html->link('Delete',
          "contacts/delete/{$_contact['Contact']['id']}",
          null, 'Sure?')?>]
      </td>
    <td><?php echo $contact['Contact']['created']; ?>
    </td>
  </tr>
<?php endforeach; ?>
</table>
```

# View

- **views/contacts/view.thtml**

```
<h1><?php echo  
    $data['Contact']['name']?></h1>  
<p><small>  
Created: <?php echo  
    $data['Contact']['created']?>  
</small></p>  
<p><?php echo  
    $data['Contact']['number']?></p>
```

# View

- views/contacts/add.html

```
<h1>Add Contact</h1>
<form action="<?php echo $html->url("contacts/add");
?>" method="post">
  <p>Name:
    <?php echo $html->input('Contact/name',
array('size' => '40')) ?>
  </p>
  <p>Number:
    <?php echo $html->input('Contact/number',
array('size' => '40')) ?>
  </p>
  <p><?php echo $html->submit('Save') ?>
  </p>
</form>
```

# View

- views/contacts/edit.html

```
<h1>Edit Contact</h1>
<form action="<?php echo $html-
    >url('/contacts/edit')?>" method="post">
    <?php echo $html->hidden('Contact/id'); ?>
    <p>Name:
        <?php echo $html->input('Contact/name',
            array('size' => '40')) ?>
    </p>
    <p>Number:
        <?php echo $html->input('Contact/number',
            array('size' => '40')) ?>
    </p>
    <p>
        <?php echo $html->submit('Save') ?>
    </p>
</form>
```

# Controller

- /app/controllers/contacts\_controller.php:

```
<?php
class ContactsController extends ApplicationController
{
    var $name = 'Contacts';

    function index() {
        $this->set('contacts', $this->Contact-
>findAll());
    }

    function view($id) {
        $this->Contact->id = $id;
        $this->set('data', $this->Contact->read());
    }
}
```

# Controller

- /app/controllers/notes\_controller.php:

```
function add() {
    if (!empty($this->data['Contact'])) {
        if($this->Contact->save($this->data['Contact'])) {
            $this->flash('Your contact has been added.',
                '/contacts/');
        }
    }
}

function delete($id) {
    if ($this->Contact->del($id)) {
        $this->flash('The contact with id: '.$id.' has been
deleted.', '/contacts/');
    }
}
```

# Controller

- /app/controllers/notes\_controller.php:

```
function edit($id = null) {  
    if (empty($this->data['Contact'])) {  
        $this->Contact->id = $id;  
        $this->data = $this->Contact->read();  
    } else {  
        if($this->Contact->save($this->data['Contact'])) {  
            $this->flash('Your contact has been  
updated.', '/contacts/');  
        }  
    }  
}  
?  
>
```



## CakePHP Rapid Development

### Contact list

[Add Contact](#)

Id	Name	Number
1	<a href="#">Michael Fish</a> <a href="#">[Edit]</a> <a href="#">[Delete]</a>	2008-11-11 11:08:11
5	<a href="#">Billy Giles</a> <a href="#">[Edit]</a> <a href="#">[Delete]</a>	2008-11-11 11:13:08
4	<a href="#">John Kettley</a> <a href="#">[Edit]</a> <a href="#">[Delete]</a>	2008-11-11 11:12:53

...../cake/contacts/add

...../cake/contacts/edit/1

...../cake/contacts/view/4

### Add Contact

Name:

Number:

### Edit Contact

Name:

Number:

### John Kettley

Created: 2008-11-11 11:12:53  
01333 444777



# Other benefits

- Bake script – command line script generator
- Uses LAMP common web platform
  - (Linux, Apache, MySQL and PHP)
- Helpers for HTML, Forms, Pagination, AJAX, Javascript, XML, RSS
- Scaffolding (no need for views)
  - Create controller with `var $scaffold;`

## List Contacts

Id	Name	Number	Notes	Created	Modified	Actions
1	Michael Fish	01222 333444		2008-11-11 11:08:11	2008-11-11 11:08:11	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
4	John Kettley	01333 444777		2008-11-11 11:12:53	2008-11-11 11:12:53	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
6	Ian McGaskill	01999 444333		2008-11-11 11:15:04	2008-11-11 11:15:04	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>

- [New Contact](#)

# Disadvantages

- Mainly due to the limitations of PHP
  - Clumsy OOP
  - Access data through arrays not classes (which RoR does) – more code in view
- Create tables in separate SQL
- Not well documented yet

# Baking

- Right click on "My Computer" and select "properties" Select the "Advanced Tab"
- Click the "Environment Variables" button at the bottom Under the "System Variables" list find the "Path" variable and click "edit"
- Now add the full path of your Php directory in your server e.g. ";C:\server\php" separated by a comma.

# Baking...

- Open the windows command prompt (Start -> Run -> type "cmd" then click 'ok')
- Navigate to the "Bake" script location by typing "cd C:\server\htdocs\cakephp\cake\scripts"
- Run the "Bake" Script by typing "php bake.php"

# Baking...

```
Baking...
```

```
-----  
Name: app
```

```
Path: C:\server\htdocs\cakephp\app  
-----
```

```
[M]odel
```

```
[C]ontroller
```

```
[V]iew
```

```
What would you like to Bake? <M/U/C>
```

```
>
```

# Baking...

```
-----  
Controller Bake:  
-----
```

```
Possible Controllers based on your current database:
```

```
1. Posts
```

```
Enter a number from the list above, or type in the name of another controller
```

```
> 1
```

```
Would you like bake to build your controller interactively?
```

```
Warning: Choosing no will overwrite controller if it exist. (y/n)
```

```
[y] > n
```

```
Would you like to include some basic class methods (index(), add(), view(),  
<>)? (y/n)
```

```
[y] > y
```

```
Would you like to create the methods for admin routing? (y/n)
```

```
[y] > _
```

# Baking...

```
<?php
class PostsController extends ApplicationController {

    var $name = 'Posts';
    var $helpers = array('Html', 'Form' );

    function index() {
        $this->Post->recursive = 0;
        $this->set('posts', $this->Post->findAll());
    }

    function view($id = null) {
        if (!$id) {
            $this->Session->setFlash('Invalid id for Post.');
```

# Baking...

```
What would you like to Bake? <M/U/C>
> u
-----
View Bake:
-----
Possible Controllers based on your current database:
1. Posts

Enter a number from the list above, or type in the name of another controller
> 1

Would you like bake to build your views interactively?
Warning: Choosing no will overwrite views if it exist. <y/n>
[yl] > n

Would you like to create the views for admin routing? <y/n>
[ln] > n

Creating file C:\server\htdocs\cakephp\app\views\posts\index.html
WroteC:\server\htdocs\cakephp\app\views\posts\index.html

Creating file C:\server\htdocs\cakephp\app\views\posts\view.html
WroteC:\server\htdocs\cakephp\app\views\posts\view.html

Creating file C:\server\htdocs\cakephp\app\views\posts\add.html
WroteC:\server\htdocs\cakephp\app\views\posts\add.html

Creating file C:\server\htdocs\cakephp\app\views\posts\edit.html
WroteC:\server\htdocs\cakephp\app\views\posts\edit.html
-----

View Scaffolding Complete.
```



# Cheat Sheet

- **CakePHP Naming Conventions**
  - **CakePHP Models**
    - class names are singular
    - class names UpperCamelCased
    - filenames use a lower-case underscored syntax
    - database tables are plural underscored
    - set var \$name in your model definition (PHP4)

# Cheat Sheet...

- **CakePHP Naming Conventions**

- **CakePHP Controllers**

- class names are plural
    - class names are UpperCamelCased for multi-word controllers
    - class names also end with 'Controller'
    - file names use a lower-case underscored syntax
    - file names also end with '\_controller.php'.

# Cheat Sheet...

- **CakePHP Naming Conventions**

- **CakePHP Views**

- views are in folders that match controller
    - view folders are plural underscored
    - views are named after actions they display.
    - name the view file after action name, in lowercase.

# Cheat Sheet...

- **CakePHP naming conventions – Examples**
  - Assuming we have a database table named **orders**, the following standard CakePHP naming conventions should be used:
- **Model**
  - filename = order.php
  - classname = Order
  - directory = app/models
- **View**
  - filename = (same as the action name in the controller)
  - extension = .ctp (the filename extension)
  - directory = app/views/orders
- **Controller**
  - filename = orders\_controller.php
  - classname = OrdersController
  - directory = app/controllers

# Cheat Sheet...

- **CakePHP naming conventions – Examples...**
  - Assuming we have a database table named **order\_items**, the following standard CakePHP naming conventions should be used:
- **Model**
  - filename = order\_item.php
  - classname = OrderItem
  - directory = app/models
- **View**
  - filename = (same as the action name in the controller)
  - extension = .ctp (the filename extension)
  - directory = app/views/order\_items
- **Controller**
  - filename = order\_items\_controller.php
  - classname = OrderItemsController
  - directory = app/controllers

# Cheat Sheet...

- Bake Commands
  - cake bake
  - cake bake controller
  - cake bake model
  - cake bake view
  - cake bake project
  - cake bake controller orders
  - cake bake model order

# Cheat Sheet...

- **CakePHP Foreign Key Examples and Relationship Types**

Relationship	Association	Type Example
one to one	hasOne	A user has one profile.
one to many	hasMany	A user can have multiple recipes.
many to one	belongsTo	Many recipes belong to a user.
many to many	hasAndBelongsToMany	Recipes have, and belong to many tags.

# Cheat Sheet...

- relationship type examples
  - # in a Post model class:
  - # each Post belongs to a User
  - var \$belongsTo = array('User');
  - # TODO
  - var \$hasOne ...
  - # in the User model
  - var \$hasMany = array('Post');
  - # TODO
  - var \$hasAndBelongsToMany



# Cheat Sheet...

- **The CakePHP recursive attribute**

Value	Meaning
-1	returns only the current model, and ignores all associations.
0	returns the current model, plus its owner(s).
1	returns the current model, its owner(s), plus their associated models.
2	returns the current model, its owner(s), their associated models, and the associated models of any associations.

```
function index() {  
    $this->Post->recursive = 0;  
    $this->set('posts', $this->paginate); }
```

# Cheat Sheet...

- find query parameters
  - Type
    - 'first'
    - can be 'all', 'first', or 'list'. determines what type of find operation to perform. (TODO - more info here)
  - Conditions
    - array containing the find (select) conditions as key/value pairs
  - Fields
    - array specifying which fields should be retrieved in the resulting select query
  - order
    - sql 'order by conditions. field name must be followed by ASC or DESC
  - page
    - page number, used for paged data

# Cheat Sheet...

- find query parameters...
  - **limit**
    - a limit on the number of results returned, like 'select \* from orders limit 20'.
  - **offset**
    - sql offset value (i haven't used this query much myself, but i think it refers to skipping X number of rows returned in a query)
  - **recursive**
    - the cakephp recursive value, relating to associated model data

# Cheat Sheet...

- find query Examples...
  - `$this->Post->find('all');`
    - Simple Enough?
  - `$this->Post->find('all', array('conditions'=>array('User.id'=>5)));`
    - A CakePHP find query with one condition:
  - `$this->Post->find('all', array('conditions'=>array('User.id'=>'<> 5')));`
    - A CakePHP find query with one "not equal to" condition:

# Cheat Sheet...

- find query Examples...
  - `$this->Post->find('all', array('conditions'=>array('User.id'=>1, 'Post.id'=>'> 50')));`
    - A CakePHP find query with multiple conditions:
  - `$this->Post->find('all', array('conditions'=>array('User.id'=>5), 'fields'=>'Post.name', 'order'=>'Post.id ASC', 'limit'=>20, 'recursive'=>0));`
    - A CakePHP find query that uses all the find function parameters:

# Cheat Sheet...

- Order by Examples
  - `array('order'=>'date ASC')`
  - `array('order'=>'date DESC')`
  - `array('order'=>'User.id DESC')`

# Cheat Sheet...

- other CakePHP find query examples:
  - `$this->Order->find('all');`
  - `$this->Order->find(null, null, 'date DESC');`
  - `$this->Order->find('all',  
array('conditions'=>array('User.id'=>1)));`
  - `$this->Order->find('all',  
array('conditions'=>array('User.id'=>array(1,2,3,4))));`
  - `$this->Order->find('all',  
array('conditions'=>array('User.id'=>'<> 1')));`
  - `$this->Order->find('all',  
array('conditions'=>array('User.id'=>1,  
'DATE(Post.date)'=>'CURDATE()')));`
  - `$this->Order->find('all', array('order'=>'date ASC',  
'limit'=>20, 'recursive'=>0);`

# Cheat Sheet...

- Many More Scenarios from <http://book.cakephp.org/view/1017/Retrieving-Your-Data>
  - find threaded
  - find neighbors
  - findAllBy
  - findBy
  - query
  - field
  - read



# Cheat Sheet...

- **CakePHP logging**

- `CakeLog::write('debug', 'Something did not work');`
- `$this->log("Something did not work!", 'debug');`
- `Configure::write('log', E_WARNING);`

# Cheat Sheet...

- CakePHP controller properties:
  - `$name = null`
  - `$action = null`
  - `$autoLayout = true`
  - `$autoRender = true`
  - `$base = null`
  - `$beforeFilter = null`
  - `$cacheAction = false`
  - `$components = array()`
  - `$data = array()`
  - `$helpers = array('Html')`

# Cheat Sheet...

- `$here = null`
- `$layout = 'default'`
- `$output = null`
- `$pageTitle = false`
- `$params = array()`
- `$persistModel = false`
- `$plugin = null`
- `$uses = false`
- `$view = 'View'`
- `$viewPath = null`
- `$webroot = null`
- `$_viewClass = null`
- `$_viewVars = array()`

# Cheat Sheet...

- **CakePHP Data Validation**

```
<?php
    class User extends AppModel {
        var $name = 'User';
        var $validate = array( 'login' => 'alphaNumeric',
            'email' => 'email', 'born' => 'date' );
    } ?>
```

# Cheat Sheet...

- CakePHP controller callbacks
  - `afterFilter ()`
  - `beforeFilter ()`
  - `beforeRender ()`

# Cheat Sheet...

- CakePHP Model callbacks
  - `afterDelete ()`
  - `afterFind ($results)`
  - `afterSave ()`
  - `beforeDelete ()`
  - `beforeFind (&$queryData)`
  - `beforeSave ()`
  - `beforeValidate ()`

# Cheat Sheet...

- **CakePHP Helpers**
  - **Html Helper**
  - **Form Helper**
  - **Ajax Helper**
  - **Text Helper**
  - **Time Helper**
  - **Number Helper**

# Cheat Sheet...

- **CakePHP Components**
  - **Session Component**
  - **RequestHandler Component**
  - **Security Component**
  - **ACL Component**



# Cheat Sheet...

# Cheat Sheet...

# Cheat Sheet...

# Cheat Sheet...

# Cheat Sheet...

# Cheat Sheet...

# Cheat Sheet...

# Cheat Sheet...



# Cheat Sheet...