



CakePHP:

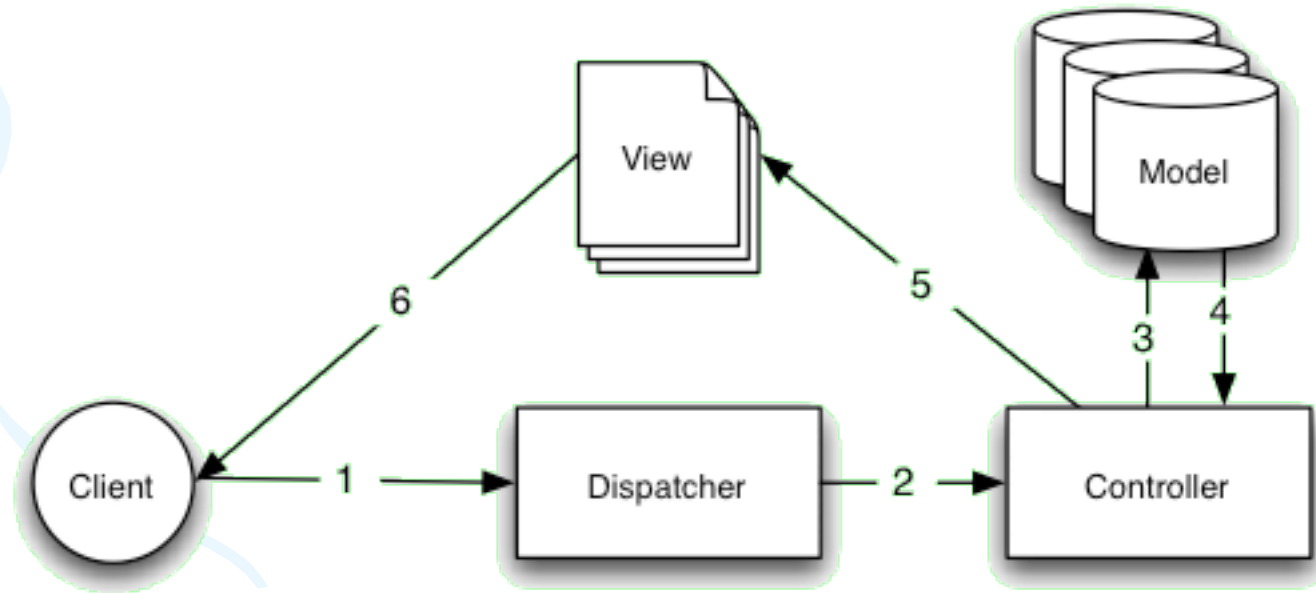
A Primer



MVC Design Pattern

- Model – Domain Data & Logic
 - Usually represents a data source
- Controller – Business Logic
 - Processes requests between the user (view) and the model
- View – Presentation Logic
 - Provides a user-friendly interpretation of the model

MVC Design Pattern



Cake!

- Loosely Coupled
- Convention Over Configuration
- Fat models! Skinny controllers! No pie!

Bad cake!



Good cake!





Models

- Data Sources
 - Database tables
 - Files
 - CSV
 - XML
 - Web Services
 - Google Calendar
 - Amazon products
 - Other Cake apps
 - CouchDB (REST)
- Magic
 - Created/Modified fields
 - `findBy<fieldName>/findAllBy<fieldName>`



Models - Associations

- **hasOne**
 - Speech hasOne Timeslot
- **hasMany**
 - Post hasMany Comments
- **belongsTo**
 - Tells Cake that this association goes both ways
- **hasAndBelongsToMany**
 - tags
- We can change associations at runtime with `bindModel/unbindModel`
- Not necessary to define all associations if we aren't going to use them often (but it's healthy)



Models – Data Retrieval

- Retrieving Data

- `$model->recursive = [-1|0|1|2|3];`

- Conditions

- `$params = array('conditions' => array('Post.id <>' => 4, 'Post.body LIKE' => 'is awesome because', 'Post.created' > date('Y-m-d, strtotime('-2 weeks'))));`

- `$model->find('list', $params);`

- `$model->find('all', $params);`

- `$model->find('first', $params);`

- `$model->find('count', $params);`

- `$model->find('neighbors', $params);`

- `$model->find('threaded', $params);`

- `$model->query('SELECT * FROM posts');`



Models - Saving

- `$model->create();`
- `$model->save(array('Post' => array('body' => 'But I like pie.')));`
- `$model->saveField('title', 'Frist Psot');`
- `$model->updateAll();`
- `$model->saveAll();`
- `$model->id;`



Models - Validation

- `$validates = array('email' => 'email');`
- Many built-in validation rules
 - `alphaNumeric`, `email`, `date`, `cc`, `ssn`, `url`, etc.
- Custom validation rules
 - Write a custom function
 - Assign it as the rule for that field
 - `$validates = array('birth_date' => array('rule' => array('checkOldEnough', 21), 'message' => 'You are not 21.'));`
- `$model->validates()` is called implicitly before every `$model->save()`



Models - Callbacks

- Implementing 'real' domain logic
 - beforeFind/afterFind
 - beforeSave/afterSave
 - beforeDelete/afterDelete
 - beforeValidate
 - onError



Behaviors

- Abstract functionality which may apply to many models, or does not directly apply to the domain logic of this model.
- Behaviors work by adding and overriding existing Model methods.
- Built-in Behaviors
 - Tree
 - Acl
 - Containable
- `$actsAs = array('Tree');`



Controllers

- List (and implement) every business action
 - `view($id)`, `admin_view($id)`, `staff_view($id)`
- By default, PostsController uses the Post model.
- Controllers should only setup the views and process input from the user on its way to the model.



Controllers - Attributes

- Ties everything together
 - `$uses = array('Post');`
 - `$helpers = array('Html', 'Form', 'JavaScript');`
 - `$components = array('Email');`
 - `$layout = 'print';`
 - `$paginate = array('limit' => 25, 'order' => array('Post.created' => 'asc'));`



Controllers - Methods

- `$controller->set('first_name' => 'Vikram');`
- `$controller->render('summary');`
- `$controller->flash('Sorry, but your blerg post was too verbose.');`
- `$controller->paginate();`
- `$controller->redirect(array('controller' => 'posts', 'action' => 'index'));`



Controllers - Callbacks

- beforeFilter/afterFilter
 - Useful for checking SSL, permissions, other pre-conditions
- beforeRender/afterRender
 - Useful for setting common view variables, layouts



Components

- Abstracting logic and functionality which can apply to many controllers
- Usage is similar to that of models
- Components should not use models
- Built-in Components
 - Email
 - Security
 - Session
 - Auth
 - RequestHandler



Views

- Presentation logic, which should be layout-agnostic
- Layouts
 - Wrap around a view
 - Not always text/html
 - text/csv
 - text/xml
 - application/pdf
 - video/ogg



Elements

- Reusable view code
 - Advertisements
 - Menus
 - Login forms
 - Receipts (on-screen, PDF and email)
- Elements can be cached individually



Helpers

- Abstracting logic which can apply to many views
- Commonly used to reduce the number of languages and APIs required
 - JavascriptHelper
 - HtmlHelper/FormHelper
 - AjaxHelper



App[Model|Controller|Helper]

- Behaviors are to Models as
- Components are to Controllers as
- Helpers are to Views
- Each piece inherits from App[Model|Controller|Helper]
- `$this->requestAction(array('controller' => 'comments', 'action' => 'index', 7));`
- Useful for defining methods which need to be available in (almost) every class of that type
 - `beforeFilter`, `beforeSave`, `beforeDelete`



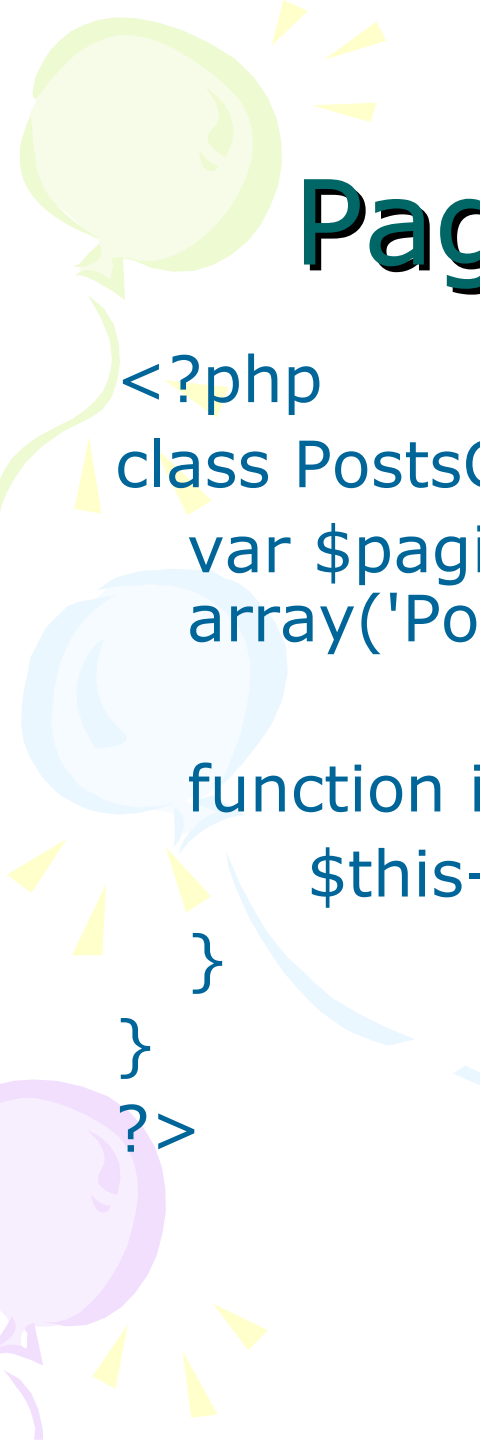
Plugins

- Packaged sets of models, controllers and views
- Only shares a database with the main app
- `/pizzaboy/delivery/takeout/4/myhouse`
- Plugins may use their own layouts
- `requestAction` works here too



Deployment

- Multiple applications per installation
- Requirements
 - mod_rewrite provides clean URLs, but not required
 - Plenty of DB support
 - PHP4/5



Pagination - Controller

```
<?php
```

```
class PostsController extends AppController {  
    var $paginate = array('limit' => 25, 'order' =>  
        array('Post.created' => 'asc'));
```

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

```
}  
?>
```



Pagination - View

```
<?php echo $paginator->counter(); ?>
<div class="header"><?php echo $paginator->sort('title'); ?>
| <?php echo $paginator->sort('body'); ?></div>
<ul>
  <?php foreach($posts as $post): ?>
  <li><?php echo $post['Post']['title']; ?> - <?php echo
  $post['Post']['body']; ?></li>
  <?php endforeach; ?>
</ul>

<?php echo $paginator->prev(); ?>
|<?php echo $paginator->numbers();?>
<?php echo $paginator->next(); ?>
```


Three balloons (green, blue, and purple) are positioned vertically on the left side of the slide. Each balloon has a string and several small yellow triangular flags attached to it. The green balloon is at the top, the blue one in the middle, and the purple one at the bottom.

Pagination

But wait, there's a demo...



Bake

script/generate? What's that?

- Create a project
- Setup database config
- Generate models, controllers and views
- Migrations



Scaffolding

- Automagic, almost no code generation
- Create your models
- Controller:
 - `<?php class PostsController extends ApplicationController { var $scaffold; } ?>`
- We can extend scaffolds, but not much
- Downside: You finished the app, but your SLOC = 6



Routing

- Named parameters are supported
 - `/posts/index/page:3/sort:Post.body/direction:asc`
- Prefix Routing
 - `/appname/admin/posts/view/12 => PostsController::admin_view(12)`
 - We can make as many prefixes as we want
- Create custom routes specific to your application
 - `/appname/support`
`=> /appname/customers/support_request/from:customer/to:level1_tech`



Caching

- Models
- Full page
- Element-only
- Pre-request query caching
- Cache anything with global function `cache()`
- Multiple engine options
 - File
 - APC
 - Xcache
 - Memcache



Console

- Bake!
- Shells
 - CLI-based Cake apps
 - Use the same MVC classes as webapps
- Tasks
 - Extend Shells



Single Table Inheritance

- Models make this work beautifully through `beforeFilter`

```
class Post {}  
class BlogPost {  
  function beforeFilter() {  
    $conditions['BlogPost']['type'] = 'blog';  
  }}  
class NewsPost {  
  function beforeFilter() {  
    $conditions['NewsPost']['type'] = 'news';  
  }}  
}
```



Vikram Dighe
Swift Signal, Inc.

610-246-6551

vikram@swiftsignal.com