Framing the Frameworks

What Are They and Do I Need One?

Ben Ramsey

What Is a Framework?

- Unified library of tools/functions/classes
- Usually with a unified API
- Generally try to separate style, content, and logic (sometimes through MVC)
- A lazy programmer's dream

Frameworks provide:

- Database abstraction
- Authentication
- Templating
- Internationalization
- Caching
- What else?

Why Use a Framework?

- You don’t have to “reinvent the wheel”
- Often held to higher scrutiny than your own code, so it may be more secure
- Speeds up development time, thus saving hours and money
- Why not?

To Build Or Not...

- What are you looking for in a framework?
- Does something else already meet my needs (chances are, it does)?
- Can I easily extend an existing framework to suit my needs?
- What do you need that's so unique it warrants your own framework?
- Why waste your time?

Selected Frameworks

- php.MVC
- WACT
- Prado
- Cake
- PEAR

php.MVC

- Implements the Model-View-Controller (MVC) design pattern
- PHP port of Jakarta Struts
- Offers many of the features of Struts, including configuration through XML
- Still in beta stages
- Last release April ’04

php.MVC

- Framework provides a single entry point Controller that is responsible for sending HTTP request to the appropriate “Action handler” (Model) after which it is forwarded to the appropriate View component
- Appears to make use of some PEAR packages

php.MVC Code Example

- Go to it on the Web

WACT

- Stands for: Web Application Component Toolkit
- A theory- and pattern-based approach
- Implements “Enterprise Patterns” in an effort to mimic (but not port) J2EE and .NET patterns/practices
- Intended to facilitate the practices of “Refactoring” and “Test Driven Design”

WACT

- Implements the following Enterprise Patterns:
  - Model-View-Controller (MVC)
  - Template View
  - Page Controller
  - Front Controller
  - Application Controller
  - Transaction Script
  - Record Set

WACT

- Still considered “alpha”
- Last release in December ’04
- Future support planned for:
  - Domain Model
  - Intercepting Filter

WACT Code Example

```
<?php
/* hello.php */

require_once '../wact/framework/common.inc.php';
require_once WACT_ROOT . 'template/template.inc.php';

$hello =& new Template('/hello-world.html');
$hello->display();
?>
```

WACT Code Example

```
<!-- templates/source/hello-world.html -->

<html>
<body>
<core:import file="hello.ini"/>
<h1>Hello, {$location}!</h1>
<h2>From &ldquo;{$name}&rdquo;.</h2>
</body>
</html>
```

```
# templates/source/hello.ini

location = wherever here is
name = Your name here
```

Prado

- Stands for: PHP Rapid Application Development Object-oriented
- Inspired by Apache Jakarta Tapestry
- Ideas borrowed from Borland Delphi and Microsoft ASP.NET
- Originally in PHP 4 but rewritten in PHP 5 for Zend's Coding Contest
- Winner of Zend's PHP 5 contest

Prado

- A component-based and event-driven Web programming framework for PHP5
- A component combines an XML specification file, an HTML template, and a PHP class
- Components are combined together to form larger components or complete pages

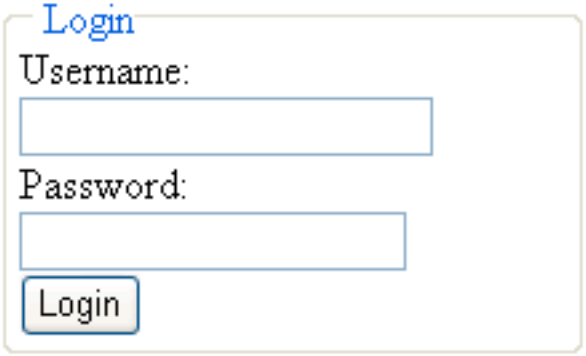
page class

```php
class LoginPage extends TPage
{
  public function loginClicked($sender,$param)
  {
    $username=$this->Username->Text;
    $password=$this->Password->Text;
    // do login stuff here
  }
}
```

page template

```
<com:TForm>
<fieldset><legend>Login</legend>
<label>Username:</label><br/>
<com:TTextBox ID="Username"/><br/>
<label>Password:</label><br/>
<com:TTextBox ID="Password" TextMode="Password"/><br/>
<com:TButton Text="Login" OnClick="loginClicked"/>
</fieldset>
</com:TForm>
```

page display



Component-based and Event-driven Programming

If an end user clicks on the login button, the loginClicked method will be invoked automatically, and the input field data can be retrieved in an object-oriented way.

Cake

- Designed as a Ruby-on-Rails “rip-off”
- Aims to bring the power, flexibility, and ease-of-use of Ruby-on-Rails to PHP applications
- Still in alpha/beta stages (0.2.9)
- Last release on 28 April 2005

Cake

- Compatible with PHP 4 and PHP 5
- Implements CRUD (Create, Read, Update, Delete) support for simplified querying of databases (boasts no need to write SQL for basic operations)
- Request dispatcher with clean URLs
- Templates use PHP syntax
- Very little Apache configuration; just needs .htaccess and mod_rewrite

Cake

- What the future holds:
  - Model/controller factories
  - Auto-validating of data in models
  - Database table relationships
  - Cache management
  - Ajax integration

Cake Code Example

```
<?php
/* app/models/post.php */

class Post extends AppModel {
}

?>
```

Cake Code Example

```
<?php
/* app/controllers/posts_controller.php */

class PostsController extends AppController {

    function index() {
    }

    function view($id) {
        $this->post->set_id($id);
        $this->set('data', $this->post->read());
    }

}

?>
```

Cake Code Example

```
<!-- app/views/posts/index.thtml -->
<table>
<tr>
    <th>ID</th>
    <th>Title</th>
    <th>Created</th>
</tr>
<?php foreach ($this->post->find_all() as $post): ?>
<tr>
    <td><?php echo $post['id']; ?></td>
    <td><a href="<?php echo $BASE; ?>/posts/view/
        <?php echo $post['id']; ?>"><?php echo $post['title']; ?>
        </a></td>
    <td><?php echo $post['created']; ?></td>
</tr>
<?php endforeach; ?>
</table>
```

Cake Code Example

```
<!-- app/views/posts/view.thtml -->

<h2><?php echo $data['title']; ?></h2>
<p><small>Created: <?php echo $data['created']; ?></small></p>
<p><?php echo $data['body']; ?></p>
```

PEAR

- The PHP Extension and Application Repository
- Is it a framework?
- Calls itself a “framework”
- Provides all the functionality of the frameworks mentioned: database abstraction, templates, caching, and more

PEAR

- So, why isn't it a "framework"?
- It doesn't have the mentality of a framework
  - It is a flexible framework
  - It is an extensible framework
  - It provides choice and alternatives
- Doesn't adhere to any design patterns

PEAR Code Example

```
require_once 'DB.php';

$db =& DB::connect('mysql://user:pass@localhost/dbname');
if (PEAR::isError($db)) {
    die($db->getMessage());
}

$res =& $db->query('SELECT * FROM mytable');
if (PEAR::isError($res)) {
    die($res->getMessage());
}

while ($res->fetchInto($row, DB_FETCHMODE_ASSOC)) {
    echo $row['id'] . "\n";
}
```

Special Mentions

- Solar
  http://solarphp.com
- Horde
  http://horde.org
- Midgard Framework
  http://midgard-project.org

Final Thoughts

- Too many Frameworks to name
- How can a Framework be more than just YAPF (Yet Another PHP Framework)?
- Simplicity, ease-of-use, documentation, and facilitation of programming are key
- In short, if it speeds up your development without fuss, it's a Good Thing(TM)

For more information...

- **php.MVC:** http://phpmvc.net
- **WACT:** http://wact.sourceforge.net
- **PRADO:** http://xisc.com
- **Cake:** http://sputnik.pl/cake
- **PEAR:** http://pear.php.net

- **My Web site:** http://benramsey.com

Questions?