

# MVC

## Model-View-Controller

### Design Patterns

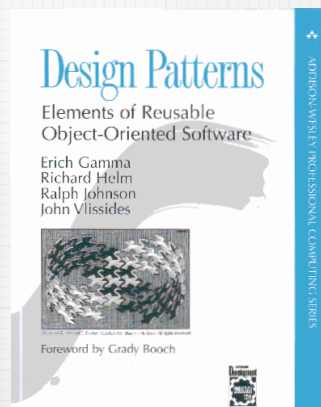
- Certain programs reuse the same basic structure or set of ideas
- These regularly occurring structures have been called **Design Patterns**

### Design Patterns

#### Design Patterns: Elements of Reusable Object-Oriented Software

Addison-Wesley

Erich Gamma, Richard Helm , Ralph Johnson, John Vlissides



## Patterns

### Behavioral Patterns

Chain of responsibility

Command

Interpreter

Iterator

Mediator

Memento

Observer

State

Strategy

Template method

Visitor

### Structural Patterns

Adapter

Bridge

Composite

Decorator

Facade

Flyweight

Proxy

Proxy Clone pattern

### Creational patterns

Abstract Factory

Builder

Factory Method

Prototype

Singleton

## MVC - design pattern

A way of structuring an application into different component parts

First described by Trygve Reenskaug working at Xerox research labs (1979)

Influential paper: Applications Programming in Smalltalk-80 - How to use Model-View-Controller

## Using MVC

### Model

Code that is core to the application

person  
invoice  
car  
booking

## Using MVC

### Model

Code that is  
core to the  
application

person  
invoice  
car  
booking

### View

Code that is key to  
the presentation /  
interaction with the  
user

HTML, CSS  
forms

## Using MVC

### Model

Code that is  
core to the  
application

person  
invoice  
car  
booking

### View

Code that is key to  
the presentation /  
interaction with the  
user

HTML, CSS  
forms

**Controller** Code that uses the Model and View

## Using MVC in PHP

### Model

PHP Classes  
SQL  
No HTML

### View

PHP for display  
HTML  
No SQL

**Controller** No SQL or HTML

## MVC choices

- Build it all yourself
- Use pre existing MVC frameworks (typically class and object based)

## Many different frameworks

PHP Framework	PHP4	PHP5	MVC	Multiple DB's	ORM	DB Objects	Templates	Caching	Validation	Ajax	Auth Module	Modules	EDP
Akelos	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-
ash.MVC	-	✓	✓	-	-	✓	✓	-	✓	-	✓	✓	-
CakePHP	✓	✓	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	-
CodeIgniter	✓	✓	✓	✓	-	✓	✓	✓	✓	-	-	-	-
DIY	-	✓	✓	-	✓	✓	✓	✓	-	✓	-	-	-
eZ Components	-	✓	-	✓	-	✓	✓	✓	✓	-	-	-	-
Fusebox	✓	✓	✓	✓	-	-	-	✓	-	✓	-	✓	-
PHP on TRAX	-	✓	✓	✓	✓	✓	-	-	✓	✓	-	✓	-
PHPDevShell	-	✓	-	-	-	-	✓	-	-	✓	✓	✓	-
PhpOpenbiz	-	✓	✓	✓	✓	✓	✓	-	✓	✓	✓	-	-
Prado	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
QPHP	✓	✓	✓	✓	-	✓	✓	-	✓	✓	✓	✓	✓
Seagull	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-
Symfony Project	-	✓	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	-
WACT	✓	✓	✓	✓	-	✓	✓	-	✓	-	-	✓	-
WASP	-	✓	✓	-	-	✓	✓	-	✓	✓	✓	✓	-
Yii	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Zend	-	✓	✓	✓	✓	✓	-	✓	✓	-	✓	✓	-
ZooP	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	✓	-	-

- <http://www.phpframeworks.com/>

## Using MVC in PHP

### Two different approaches

Mixed imperative and class/object style

Smarty Templating

Class/object style throughout

Cake framework



## Using MVC in PHP - Smarty

### Model

Class style

### View

HTML templates

### Controller

Imperative style

## Using MVC in PHP - Smarty

### Model

PHP Classes  
SQL  
No HTML

### View

Smarty templates  
(HTML and smarty  
variables)  
No SQL

### Controller

Smarty controller code

## MVC strategy - Smarty

### Model

Construct model  
classes for the  
application

### View

### Controller

## MVC strategy - Smarty

### Model

Construct model  
classes for the  
application

### View

Construct Views

HTML templates

### Controller

## MVC strategy - Smarty

### Model

Construct model  
classes for the  
application

### View

Construct Views

HTML templates

### Controller

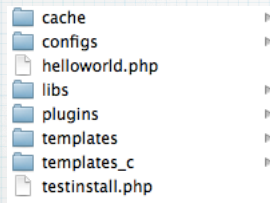
Create controllers that pass  
data to views

## Simple Smarty Example - 1

Hello World in HTML

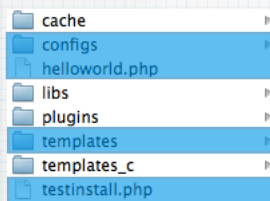
## Smarty Directory Structure

Smarty expects certain folders and files



## Smarty Directory Structure

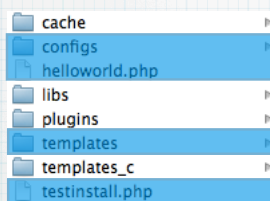
Smarty expects certain folders and files



Don't save files in these  
or alter the contents

## Smarty Directory Structure

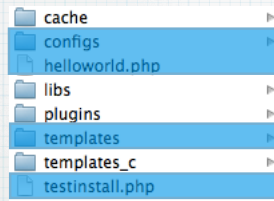
Smarty expects certain folders and files



Used to store temporary versions

## Smarty Directory Structure

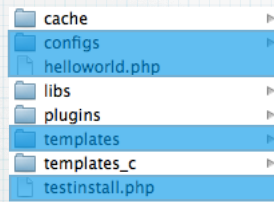
Smarty expects certain folders and files



Core Smarty Files

## Smarty Directory Structure

Smarty expects certain folders and files



"Compiled" versions of the web pages

## Simple Smarty Example - 1

View

Construct View

HTML template

Controller

Create controller to pass object to view



## Simple Smarty Example - 1

### Controller

Create controller to pass object to view

```
<?php
//Place controller code here

// These lines initialise the smarty View object
// Don't change these unless the files are in different directories
require('libs/Smarty.class.php');
$smarty = new Smarty();
$smarty->setTemplateDir('templates');
$smarty->setCompileDir('templates_c');
$smarty->setCacheDir('cache');
$smarty->setConfigDir('configs');

//These are the lines to change
//Add one assign line for each named piece of data with the data

//This line should be changed to match the template filename in /templates
$smarty->display('');
?>
```

## Simple Smarty Example - 1

### Controller

Create controller to pass object to view

```
//These are the lines to change
//Add one assign line for each named piece of data with the data
$smarty->assign('hello', 'Hello World!!');
```

Add one line for every piece of data that we pass through to the template

In the template, the smarty variable `hello` will have the value `Hello World!!`

Note - no `hello` in front of `hello`

## Simple Smarty Example - 1

### Controller

Create controller to pass object to view

```
//This line should be changed to match the template filename in /templates
$smarty->display('helloworld.tpl');
```

Name the template file that will be used

Smarty expects `.tpl` extension

## Simple Smarty Example - 1

### Controller

Create controller to pass object to view

```
<?php
//Place controller code here

// These lines initialise the smarty View object
// Don't change these unless the files are in different directories
require('libs/Smarty.class.php');
$smarty = new Smarty();
$smarty->setTemplateDir('templates');
$smarty->setCompileDir('templates_c');
$smarty->setCacheDir('cache');
$smarty->setConfigDir('configs');

//These are the lines to change
//Add one assign line for each named piece of data with the data
$smarty->assign('hello', 'Hello World!!');

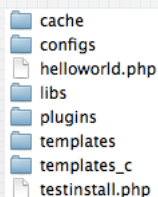
//This line should be changed to match the template filename in /templates
$smarty->display('helloworld.tpl');
?>
```

Save this as  
helloworld.php

## Simple Smarty Example - 1

### Controller

Create controller to pass object to view



```
cache
configs
helloworld.php
libs
plugins
templates
templates_c
testinstall.php
```

Save the controller in the root

Save the template in the  
templates folder

Best practice - give controller  
and template same name

helloworld.php

helloworld.tpl

## Simple Smarty Example - 1

### View

Construct View

HTML template

helloworld.tpl

```
<html>
<head>
<title>Smarty</title>
</head>
<body>
<p>                </p>
</body>
</html>
```

A plain HTML file with all the  
variable bits of data removed

Smarty inserts the data for us  
from the controller

## Simple Smarty Example - 1

View

Construct View

HTML template

helloworld.tpl

```
<html>
<head>
<title>Smarty</title>
</head>
<body>
<p>{$hello}</p>
</body>
</html>
```

Use this syntax to insert a smarty value

{ \$smartyname }

Save this template in the templates folder

## Simple Smarty Example - 1

Bring up helloworld.php in browser

Hello World!!

## Simple Smarty Example - 2

Create an Employee using the employee object and print some values out

## Simple Smarty Example - 2

### Model

Construct model classes for the application

### View

Construct Views

HTML templates

### Controller

Create controllers that pass data to views

## Simple Smarty Example - 2

### Model

emp class

### View

Print an Emp objects details

### Controller

controller to create an emp object and pass to View

## Simple Smarty Example - 2

### Model

### EMP class

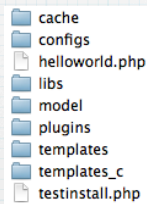
```
1 |<?php
2 | class Emp {
3 |     protected $empno;
4 |     protected $ename;
5 |     protected $job;
6 |     protected $mgr;
7 |     protected $hiredate;
8 |     protected $sal;
9 |     protected $comm;
10 |    protected $deptno;
11 |
12 |    function __construct($new_empno="", $new_ename="", $new_job="", $new_mgr="", $new_hired
13 |        $this->empno=$new_empno;
14 |        $this->ename=$new_ename;
15 |        $this->job=$new_job;
16 |        $this->hiredate=$new_hired;
17 |        $this->sal=$new_sal;
18 |        $this->comm=$new_comm;
19 |        $this->deptno=$new_deptno;
20 |    }
21 |
22 |    function get_empno(){
23 |        return $this->empno;
```



## Simple Smarty Example - 2

### Model

### EMP class



```
cache
configs
helloworld.php
libs
model
plugins
templates
templates_c
testinstall.php
```

Create a folder called **model** to save **model classes**

Save **class.Emp.php** in here

## Simple Smarty Example - 2

### Controller

Create controller to pass data to view

```
<?php
//Place controller code here

// These lines initialise the smarty View object
// Don't change these unless the files are in different directories
require('libs/Smarty.class.php');
$smarty = new Smarty();
$smarty->setTemplateDir('templates');
$smarty->setCompileDir('templates_c');
$smarty->setCacheDir('cache');
$smarty->setConfigDir('configs');

//These are the lines to change
//Add one assign line for each named piece of data with the data

//This line should be changed to match the template filename in /templates
$smarty->display('');
?>
```

## Simple Smarty Example - 2

### Controller

Create controller to pass data to view

```
<?php

//Place controller code here
require ('model/class.Emp.php');
$newEmp = new Emp(123, "Homer", "Manager", 456, "02/02/98", 34000, 0, 2);

// These lines initialise the smarty View object
// Don't change these unless the files are in different directories
require('libs/Smarty.class.php');
$smarty = new Smarty();
$smarty->setTemplateDir('templates');
$smarty->setCompileDir('templates_c');
$smarty->setCacheDir('cache');
$smarty->setConfigDir('configs');

//These are the lines to change
//Add one assign line for each named piece of data with the data
$smarty->assign('ename', $newEmp->get_ename());
$smarty->assign('job', $newEmp->get_job());
```

## Simple Smarty Example - 2

### Controller

Create controller to pass data to view

```
1 <?php
2
3 //Place controller code here
4 require ('model/class.Emp.php');
5 $newEmp = new Emp(123, "Homer", "Manager", 456, "02/02/98", 34000, 0, 2);
6
7
8 // These lines initialise the smarty View object
9 // Don't change these unless the files are in different directories
10 require('libs/Smarty.class.php');
11 $smarty = new Smarty();
12 $smarty->setTemplateDir('templates');
13 $smarty->setCompileDir('templates_c');
14 $smarty->setCacheDir('cache');
15 $smarty->setConfigDir('configs');
16
17 //These are the lines to change
18 //Add one assign line for each named piece of data with the data
19 $smarty->assign('ename', $newEmp->get_ename());
20
21 //This line should be changed to match the template filename in /templates
22 $smarty->display('empTest1.tpl');
23 ?>
```

Save as empTest1.php

## Simple Smarty Example - 2

### View

Create view empTest1.tpl

```
1 <html>
2 <head>
3 <title>Display an Emp</title>
4 </head>
5 <body>
6 <h1>Details for an Emp</h1>
7 <p>Employees name is { $ename }</p>
8 </body>
9 </html>
```

## Simple Smarty Example - 2

### Details for an Emp

Employees name is Homer

## Simple Smarty Example - 2

In fact a better approach

Pass the object into the View

Use the -> notation to access the methods in the View

## Simple Smarty Example - 2

```
17 //These are the lines to change
18 //Add one assign line for each named piece of data with the data
19 $smarty->assign('passedEmp', $newEmp);
20
```

Pass in the whole `$newEmp` object with the name `passedEmp`

```
1 <html>
2 <head>
3 <title>Display an Emp</title>
4 </head>
5 <body>
6 <h1>Details for an Emp</h1>
7 <p>Employees name is {$passedEmp->get_ename()} and they are a {$passedEmp->get_job()}</p>
8 <p>{$passedEmp->get_ename()} earns {$passedEmp->get_sal()}</p>
9 </body>
10 </html>
```

Now use the methods inside the object `$passedEmp`

## Simple Smarty Example - 3

In online exercise

Use the `empDB` class to get the emp details from a mysql table



## MVC frameworks

- Have very strict rules for use
- Names must match certain dictated patterns
- Files must be placed in certain places
- Classes in your application inherit from the MVC classes
- Very steep learning curve, but RAD