

MVC na WEB um Exemplo

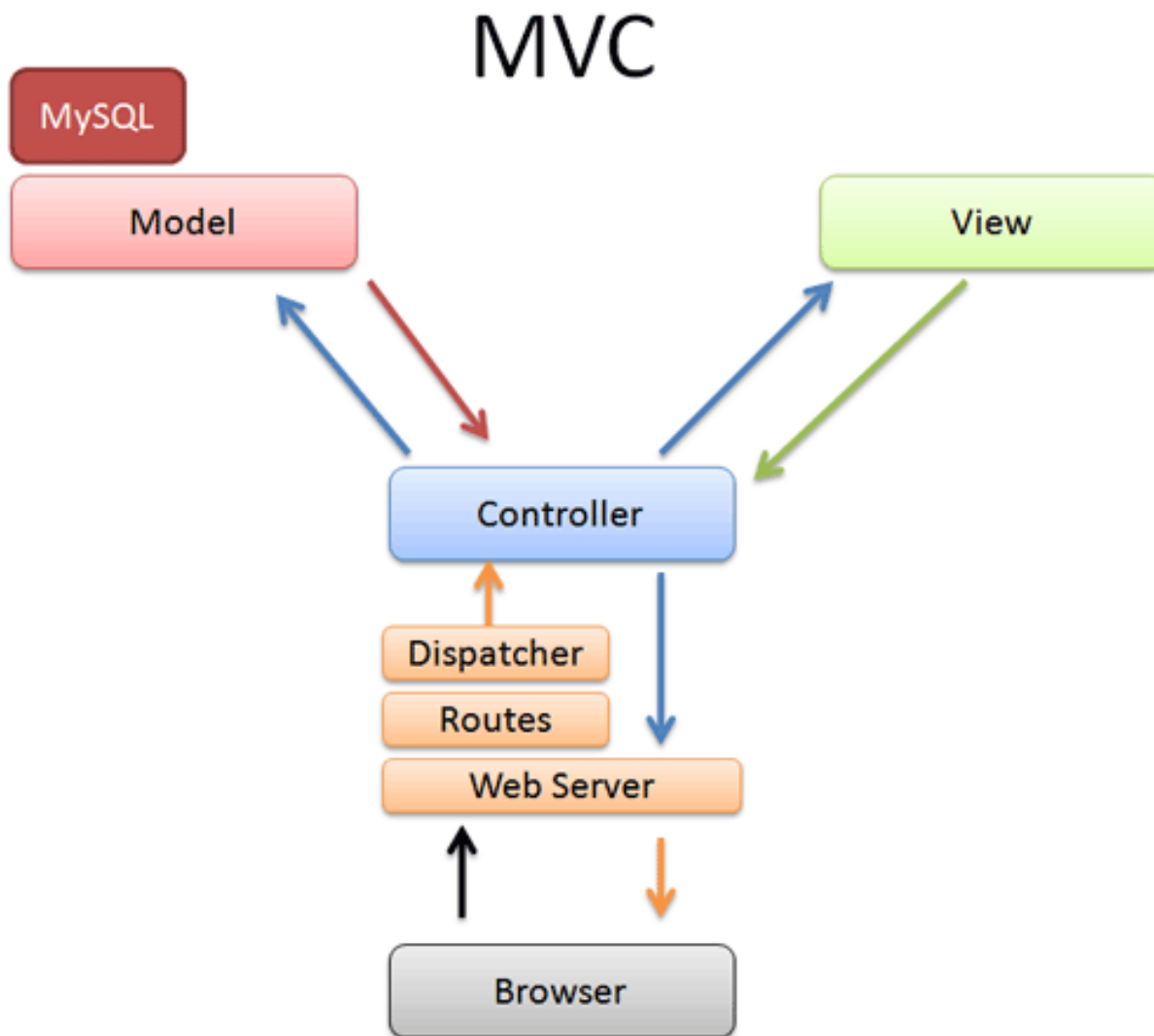


Padrão M.V.C.

Model-view-controller (MVC) é um padrão de software que divide a aplicação em três camadas:

Model, View e Controller.

Padrão M.V.C.



Rota / Roteamento

Rotear é apontar uma direção correta.

Imagine você pedindo informações em um posto de gasolina ou loja de conveniência. Você pergunta ao atendente e ele aponta a direção correta, ou pelo menos você espera que as instruções estejam corretas.

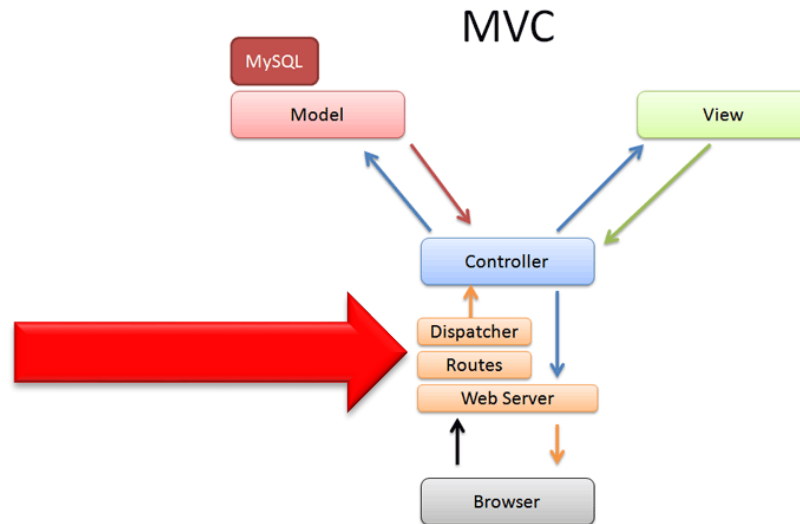
O roteamento é exatamente isso.

Uma solicitação é enviada para um recurso, o roteador fornece as instruções necessárias para que a solicitação atinja o recurso correto.

Dispatcher / Expedidor

A expedição (despachar) usa as informações da etapa de roteamento para executar um recurso.

Despachar sabe exatamente o que criar e as etapas necessárias para gerar/executar o recurso, mas somente depois de obter as instruções do roteador.



Router e Dispatcher

Uma implementação simples e ingênua de router e dispatch para entendimento do conceito (com 3 classes e um programa principal).

```
class Requisicao {  
  
    private $metodo;  
    private $caminho;  
  
    function __construct($metodo, $caminho) {  
        $this->metodo = $metodo;  
        $this->caminho = $caminho;  
    }  
  
    function getMetodo() {  
        return $this->metodo;  
    }  
  
    function getCaminho() {  
        return $this->caminho;  
    }  
  
}
```

```
<?php
```

```
class Rota {
    private $rotas = [
        'get' => [],
        'post' => []
    ];

    function get($recurso, callable $conteudo) {
        $this->rotas['get'][$recurso] = $conteudo;
        return $this;
    }

    function post($recurso, callable $conteudo) {
        $this->rotas['post'][$recurso] = $conteudo;
        return $this;
    }

    function match(Requisicao $requisicao) {
        $method = strtolower($requisicao->getMetodo());
        if (!isset($this->rotas[$method])) {
            return false;
        }
        $path = $requisicao->getCaminho();
        foreach ($this->rotas[$method] as $recurso => $conteudo) {
            if ($recurso === $path) {
                return $conteudo;
            }
        }
        return false;
    }
}
```

Router e Dispatcher

```
class Dispatcher {  
  
    private $rota;  
  
    function __construct(Rota $rota) {  
        $this->rota = $rota;  
    }  
  
    function recurso(Requisicao $request) {  
        $recurso = $this->rota->match($request);  
        if (!$recurso) {  
            echo ">>>Não consegui achar o recurso!\n";  
            return;  
        }  
  
        $recurso();  
    }  
  
}
```


Router e Dispatcher

```
<?php
require_once("rota.class.php");
require_once("requisicao.class.php");
require_once("dispatcher.class.php");

$rota = new Rota();
$rota->get('comida', function() { echo "Você enviou: GET comida <br>"; });
$rota->post('bebida', function() { echo "Você enviou: POST bebida <br>"; });

$dispatcher = new Dispatcher($rota);

$dispatcher->recurso(new Requisicao('GET', 'comida'));
$dispatcher->recurso(new Requisicao('POST', 'bebida'));
$dispatcher->recurso(new Requisicao('GET', 'qualquerCoisa'));
?>
```

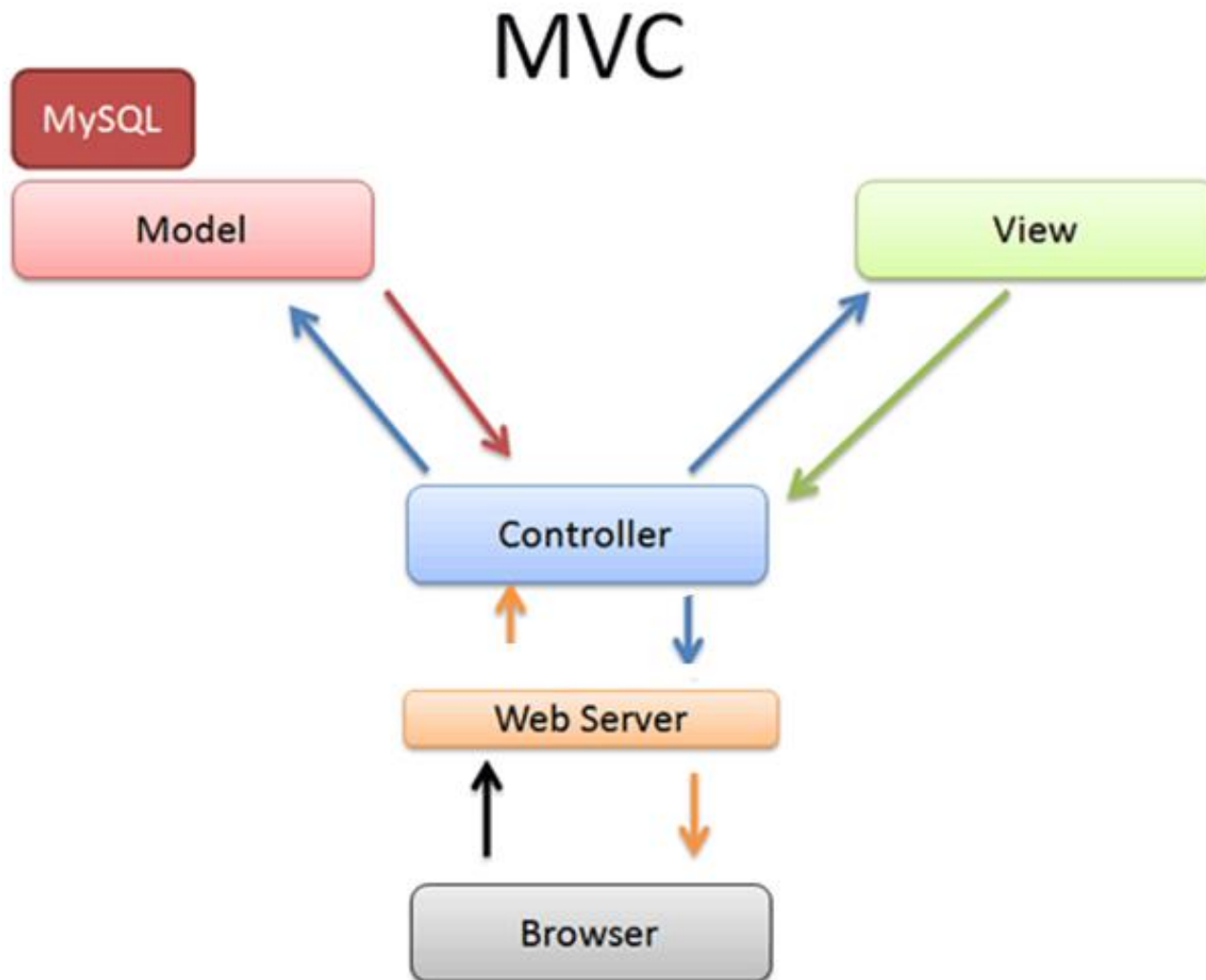


Você enviou: GET comida

Você enviou: POST bebida

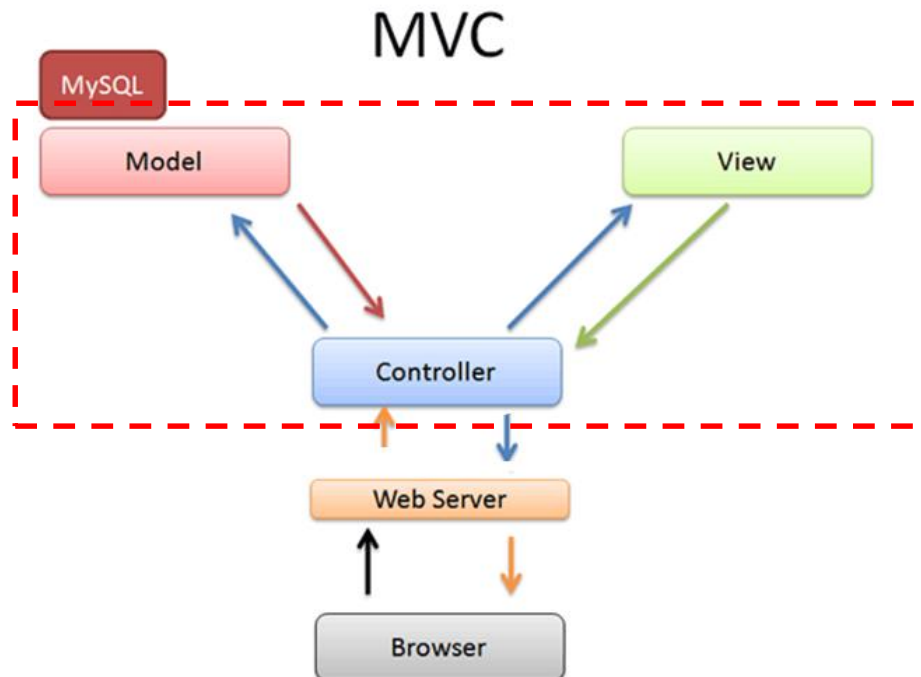
>>> Não consegui achar o recurso!

Padrão M.V.C.



Padrão M.V.C.

Acrônimo de Model, View e Controller.
É um padrão de desenvolvimento de software
que divide a aplicação em três camadas.



Padrão M.V.C.

Model é onde deve ficar a parte **lógica da aplicação**, ou seja, **todos os recursos** da sua aplicação (**consultas ao banco de dados, validações, lógica de disparo de email...**).

A camada **model** apenas tem o necessário para que tudo aconteça, mas não executa nada de fato, porque não sabe quando cada recurso deve ser executado.

Padrão M.V.C.

View é onde estão os recursos necessários para **exibir dados** e, isso pode ser muito amplo: código html, componentes visuais, menus multiníveis, etc.

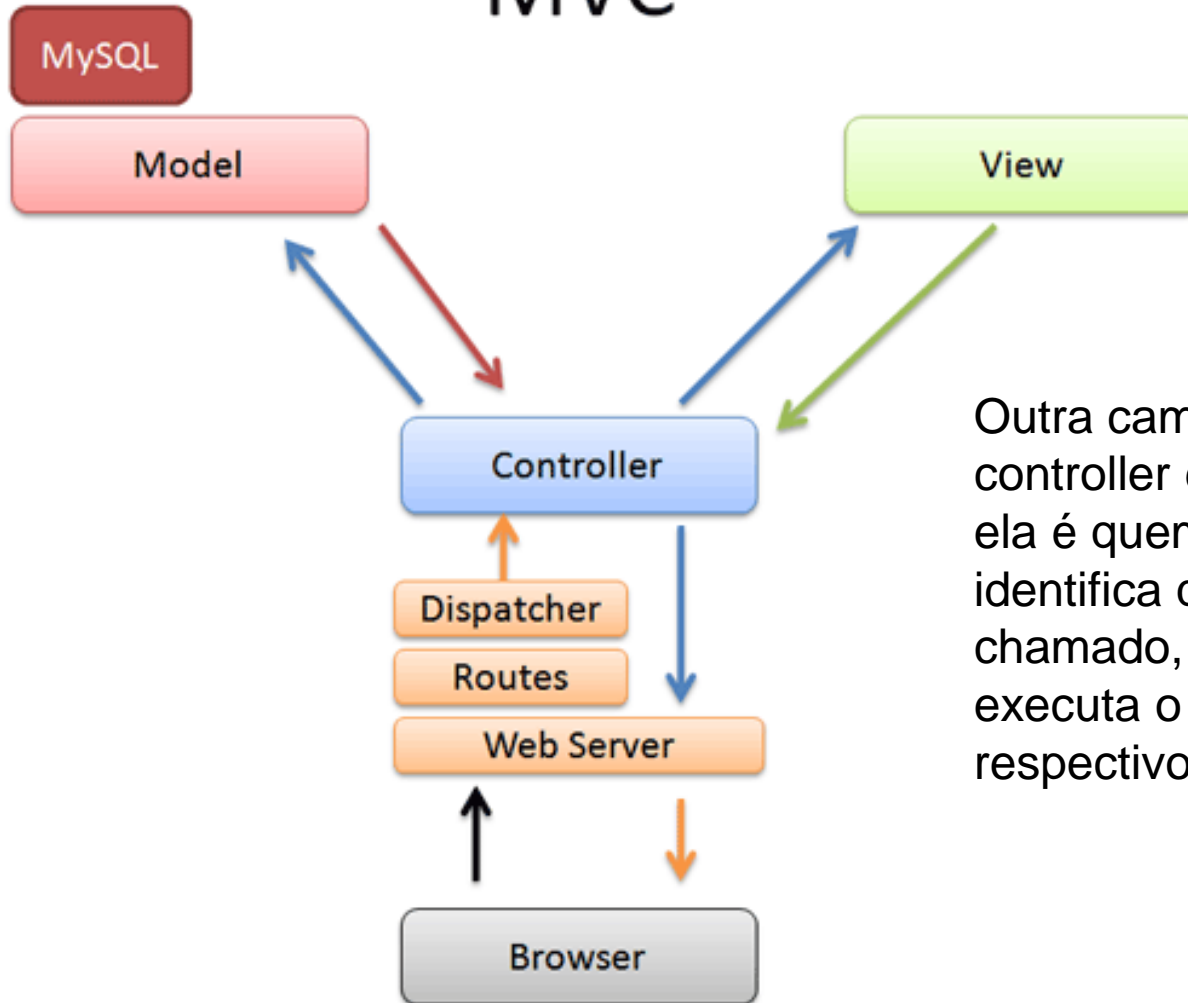
Assim como a Model, a View não sabe quando deve executar os recursos, apenas sabe como fazer, não quando.

Padrão M.V.C.

Controller é quem comanda tudo.
Ele deve saber quem está acionando o sistema (um formulário ou outro recurso), o que está solicitando e, onde o recurso se encontra para ser acionado.
Ou seja, o Controller **tem a obrigação de dizer o que deve acontecer e quando**; tão logo ele seja acionado.

Padrão M.V.C.

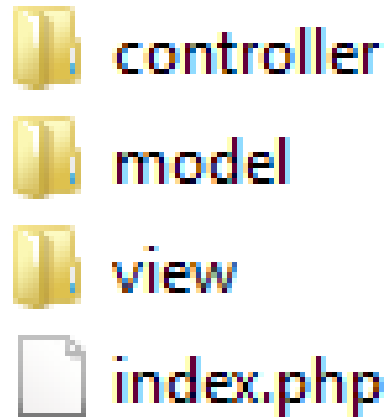
MVC



Outra camada que trabalha com o controller é o **router**, ou roteamento, ela é quem pega a URL digitada e identifica qual controller deve ser chamado, o controller por sua vez executa o model e o view respectivos.

Padrão M.V.C.

Uma implementação simples e ingênua do padrão M.V.C. para entendimento do conceito.



Padrão M.V.C.


```
<?php
require_once("model/Model.php");

class Controller {
    public $model;

    public function __construct()
    {
        $this->model = new Model();
    }


    public function executar()
    {
        // mostra uma lista de livros
        if (!isset($_GET['book']))
        {
            $books = $this->model->getBookList();
            include 'view/booklist.php';
        }
        else
        {
            // mostra o livro solicitado
            $book = $this->model->getBook($_GET['book']);
            include 'view/viewbook.php';
        }
    }
}

?>
```

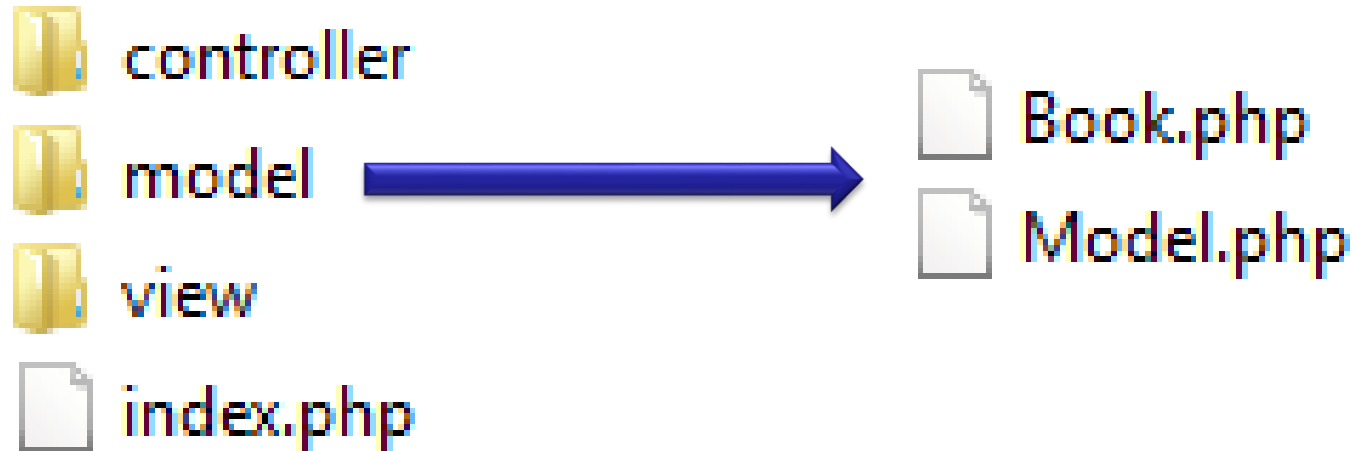
 controller

 model

 view

 index.php

Padrão M.V.C.



Padrão M.V.C.

```
<?php

class Book {
    public $title;
    public $author;
    public $description;

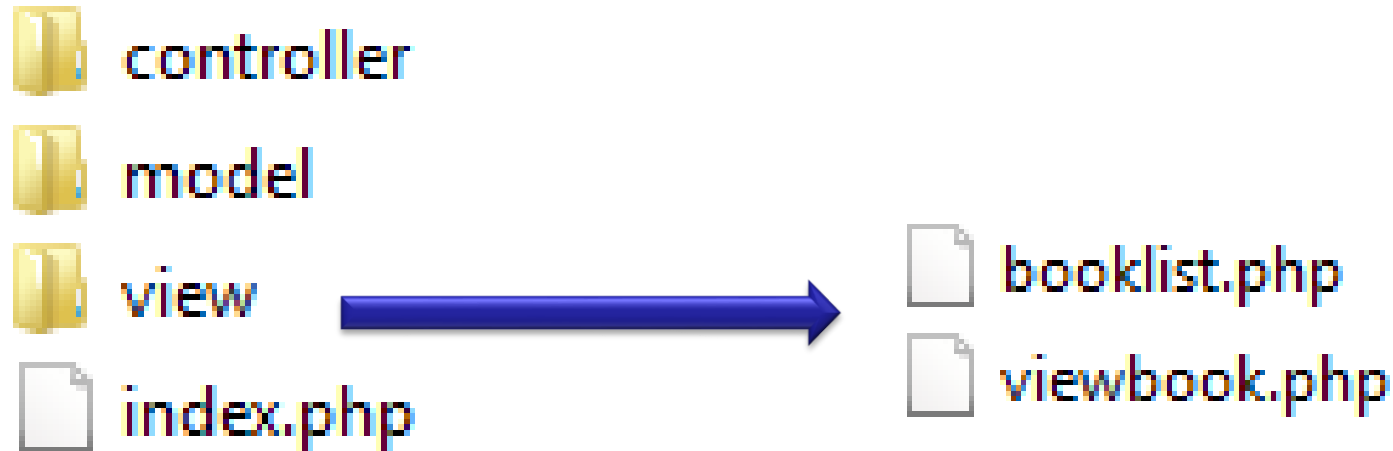
    public function __construct($title, $author, $description)
    {
        $this->title = $title;
        $this->author = $author;
        $this->description = $description;
    }
}

?>
```

Padrão M.V.C.

```
<?php
require_once("model/Book.php");
class Model {
    public function getBookList()
    {
        return array(
            "Engenharia de Software" => new Book("Engenharia de Software", "Tonsig, S.L.", "Sobre análise."),
            "Moonwalker" => new Book("Moonwalker", "J. Walker", "Não faço a menor ideia do que seja."),
            "PHP for Dummies" => new Book("PHP for Dummies", "Some Smart Guy", "Precisa ler inglês.")
        );
    }
    public function getBook($title)
    {
        $allBooks = $this->getBookList();
        return $allBooks[$title];
    }
}
?>
```

Padrão M.V.C.



Padrão M.V.C.

```
<?php
echo "
<html>
<head></head>
<body>
<h1> Lista de Livros </h1>
<table>
    <tr><td>Titulo</td><td>Autor</td><td>Descrição</td></tr>";

foreach ($books as $title => $book)
{
    echo '<tr><td><a href="index.php?book=' . $book->title . '">' .
        $book->title . '</a></td><td>' .
        $book->author . '</td><td>' .
        $book->description . '</td></tr>';
}

echo "
</table>
</body>
</html>";
?>
```

Padrão M.V.C.

```
<?php
echo "
<html>
<head></head>
<body>" ;
    echo '<h1>Livro Solicitado</h1>';
    echo 'Titulo:' . $book->title . '<br/>';
    echo 'Autor:' . $book->author . '<br/>';
    echo 'Descrição:' . $book->description . '<br/>';

echo "
<hr>
<a href='index.php'>Retornar</a>
</body>
</html>" ;
?>
```

Padrão M.V.C.

Index.php

```
<?php
    require_once("controller/Controller.php");
    $controller = new Controller();
    $controller->executar();
?>
```