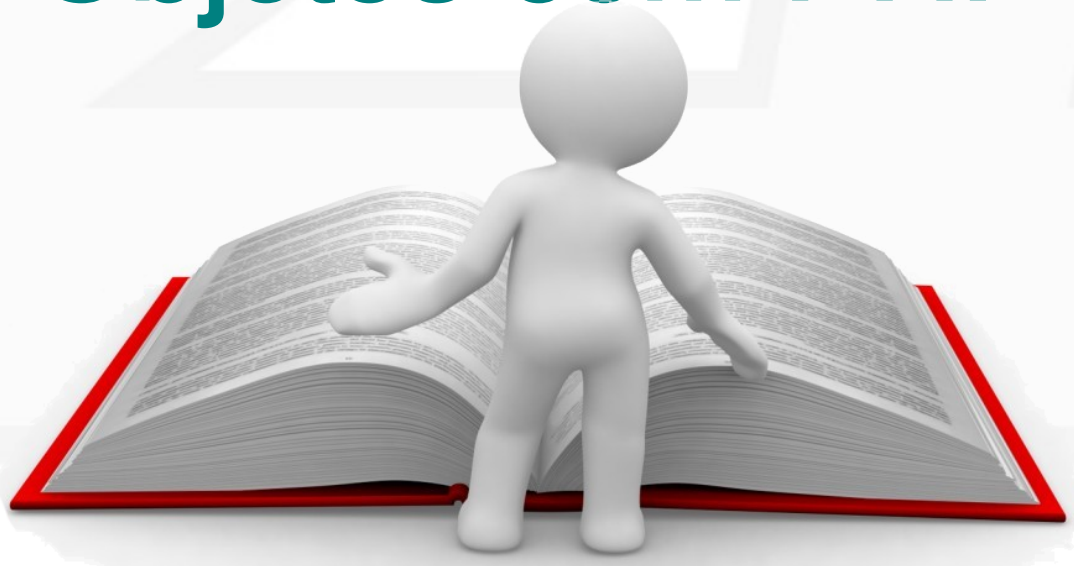
Cursos, soluções e serviços baseados em software livres e padrões abertos para ambientes de missão crítica

4LINUX
FREE SOFTWARE SOLUTIONS

Bem-vindo ao curso

Desenvolvimento Orientado a Objetos com PHP



Projeto Loja Virtual

/var/www/loja

index.php
bootstrap.php
config.php
banco.sql

app
- Controller.php
- View.php
- Modules
  - **Compras.php**
  - Livros.php

Arquivo que interpreta o que foi enviado via POST por carrinho.tpl.php e, de acordo com o que foi recebido via POST, chama métodos de Carrinho.php como esvaziarCarrinho, listarProdutos, etc

lib
- Banco.php
- **Carrinho.php**

Arquivo que contém os métodos esvaziarCarrinho, listarProdutos, etc

views
- footer.tpl.php
- header.tpl.php
- home.tpl.php
- Compras
  - **carrinho.tpl.php**
- Livros
  - alterar.tpl.php
  - inserir.tpl.php
  - livros.tlp.php
  - ver.tlp.php

Formulário que envia informações via POST para Compras.php

Entendendo o Modelo

Lib/Carrinho.php

```
<?php

class Carrinho {
    protected $banco;

    public function __construct() {
        $this->banco = Banco::instanciar();
    }

    public function adicionarProduto($produto) {
        if(isset($_SESSION['compras'][$produto])) {
            $_SESSION['compras'][$produto] += 1;
        } else {
            $_SESSION['compras'][$produto] = 1;
        }
    }

    public function removerProduto($produto) {
        unset($_SESSION['compras'][$produto]);

        if(empty($_SESSION['compras'])) {
            unset($_SESSION['compras']);
        }
    }
```

Neste método, iremos adicionar produtos ao nosso carrinho, guardado na sessão

Caso o produto já esteja no carrinho, aumentamos a quantidade, senão, adicionamos ele

Neste método, iremos remover produtos do nosso carrinho, guardado na sessão

Removemos o produto especificado da nossa sessão

Caso este seja o último produto do carrinho, isto é, o array estiver vazio depois de remover o produto, removemos o array das compras

Entendendo o Modelo

Lib/Carrinho.php

```
public function esvaziarCarrinho() {
          unset($_SESSION['compras']);
     }



     public function alterarQuantidade($produto, $quantidade) {
          if($quantidade == 0) {
               unset($_SESSION['compras'][$produto]);
          } else {
               $_SESSION['compras'][$produto] = $quantidade;
          }
     }



     public function listarProdutos() {
          if(isset($_SESSION['compras'])) {
               foreach($_SESSION['compras'] as $key => $value) {
                    $livro = $this->banco->ver("livros", "*", "id = $key");
                    $lista[$key] = $livro[0];
                    $lista[$key]['quantidade'] = $value;
               }

               return $lista;
          }
     }
}
```

Neste método, iremos esvaziar o carrinho

Neste método, iremos alterar a quantidade de um produto do nosso carrinho

Neste método, iremos listar os produtos do nosso carrinho

Entendendo o Modelo

Views/Compras/carrinho.tpl.php

Quem irá receber os dados deste formulário via POST?

Depende do que foi passado via URL via GET! Quem define isso é o controller!

Ex: index.php?module=**compras**&action=**carrinho**
Onde carrinho indentifica carrinho.tpl.php
E compras identifica Compras.php

```
<h2>Carrinho de compras</h2>
<?php
if(isset($data)) {
     $total = 0;
?>
<form action="" method="POST">
     <table cellspacing="2" cellpadding="5" border="1">
<?php

     foreach($data as $produto) {
          $total += $produto['preco'] * $produto['quantidade'];
          ?>
          <tr>
          <td><strong><?php echo $produto['titulo'] ?></strong>
          <input type="hidden" name="produto[]" value="<?php echo $produto['id'] ?>" /></td>
          <td><input type="text" name="quantidade[]" value="<?php echo $produto['quantidade'] ?>" /></td>
          <td><input type="checkbox" name="remover[]" value="<?php echo $produto['id'] ?>" /></td>
          </tr>
          <?php
     }

     ?>
```

Entendendo o Modelo

Views/Compras/carrinho.tpl.php

```
     <tr>
          <td colspan="3" align="right"><strong>Total:</strong> R$ <?php echo $total ?></td>
          <tr>
               <td colspan="3" align="right">
                    <input type="submit" name="compra" value="Finalizar compra" />
                    <input type="submit" name="esvaziar" value="Esvaziar carrinho" />
                    <input type="submit" name="atualizar" value="Atualizar carrinho" />
               </td>
          </tr>
     </tr>
     </table>
</form>
<?php
} else {
     echo "<p>Não há nada no carrinho.</p>";
}
?>
```

Sendo assim, dependendo do que for recebido via post, determinada ação será realizada. Quem realiza essa ação?
Métodos de Compras.php que por sua vez chamam métodos de Carrinho.php

E como Compras.php entra na história? Graças ao Controller!

Entendendo o Modelo

app/Modules/Compras.php

```
<?php

class Compras {
     protected $banco;
     protected $carrinho;

     public function __construct() {
          $this->banco = Banco::instanciar();
          $this->carrinho = new Carrinho;
     }

     public function adicionar() {
          $this->carrinho->adicionarProduto($_GET['id']);
          header("Location: index.php?module=compras&action=carrinho");
     }

     public function carrinho() {
          if(isset($_POST['esvaziar'])) {
               $this->carrinho->esvaziarCarrinho();
          }

          if(isset($_POST['atualizar'])) {
               foreach($_POST['produto'] as $chave => $produto) {
                    $this->carrinho->alterarQuantidade($produto, $_POST['quantidade'][$chave]);
               }
```

Para cada produto em nosso carrinho, chamaremos o método de alteração de quantidade

Entendendo o Modelo

app/Modules/Compras.php

Sendo assim, Compras.php é o responsável por interpretar o que foi enviado pelo formulário da view e a chamar os métodos de Carrinho.php dependendo do caso.

```
     if(isset($_POST['remover'])) {
          foreach($_POST['remover'] as $produto) {
               $this->carrinho->removerProduto($produto);
          }
     }
}

if(isset($_POST['compra'])) {
     foreach($_POST['produto'] as $key => $value) {
          $quantidade = $_POST['quantidade'][$key];
          $produto = $this->banco->ver("livros", "*", "id = $value");
          $compras[] = "[{$quantidade}-{$produto[0]['id']}-{
               $produto[0]['titulo']}-{$produto[0]['preco']}]";
     }

     $compras = implode(', ', $compras);
     $this->carrinho->esvaziarCarrinho();
     die("Compramos os itens: $compras");
}

return $this->carrinho->listarProdutos();

}}
```

Caso o checkbox de remoção tenha sido marcado, itere entre os valores marcados e chame o método de remoção com o value do checkbox

Criamos um dado pré-formatado com informações da compra. A partir daqui, podemos chamar um método de uma classe de boleto, cartão ou PagSeguro

Unimos nossos dados pré-formatados, separados por vírgulas, para cada produto comprado

Mostramos na tela os itens comprados

Projeto Loja Virtual

/var/www/loja

index.php
bootstrap.php
config.php
banco.sql

app
- Controller.php
- View.php
- Modules
  - Compras.php
  - **Livros.php**

Arquivo que interpreta o que foi enviado via POST pelas views alterar.tpl.php, inserir.tpl.php, etc e, de acordo com o que foi recebido via POST, chama métodos de Banco.php como remover, ver, alterar, inserir, etc

lib
- **Banco.php**
- Carrinho.php

Arquivo que contém os métodos de acessoao banco como remover, ver, alterar, etc

views
- footer.tpl.php
- header.tpl.php
- home.tpl.php
- Compras
  - carrinho.tpl.php
- Livros
  - **alterar.tpl.php**
  - **inserir.tpl.php**
  - **livros.tlp.php**
  - **ver.tlp.php**

Formulários que enviam informações via POST para Livros.php

Entendendo o Modelo

app/Modules/Livros.php

```
<?php

class Livros {
     protected $banco;
     protected $tabela = 'livros';

     public function __construct() {
          $this->banco = Banco::instanciar();
     }

     public function livros() {
          return $this->banco->listar($this->tabela);
     }

     public function inserir() {
          if($_POST) {
               $this->banco->inserir($this->tabela, $_POST);
               header("Location: index.php?module=livros&action=livros");
          }
     }

     public function alterar() {
          if($_POST) {
               $this->banco->inserir($this->tabela, $_POST['id'], $_POST);
               header("Location: index.php?module=livros&action=livros");
          }

          $dados = $this->banco->ver($this->tabela, '*', $_GET['id']);
          return $dados[0];
     }
```

Entendendo o Modelo

app/Modules/Livros.php

```
    public function remover($tabela, $id) {
        $query = "DELETE FROM `$tabela`";

        if(!empty($id)) {
            $query .= " WHERE id = ?";
        }
            $this->executar($query, array($id));
    }

    public function listar($tabela, $campos = '*', $onde = null,
        $filtro = null, $ordem = null, $limite = null) {
        $query = "SELECT $campos FROM `$tabela`";

        if(!empty($onde)) {
            $query .= " WHERE $onde";
        }

        if(!empty($filtro)) {
            $query .= " LIKE $filtro";
        }

        if(!empty($ordem)) {
            $query .= " ORDER BY $ordem";
        }

        if(!empty($limite)) {
            $query .= " LIMIT $limite";
        }
        return $this->consultar($query);
    }
```

```
    public function ver($tabela, $campos, $onde) {
        $query = "SELECT $campos FROM `$tabela`";

        if(!empty($onde)) {
            $query .= " WHERE $onde";
        }

        return $this->consultar($query);
    }

}
```

Entendendo o Modelo

views/livros.tpl.php

```
 <ul>
<?php foreach($data as $livro) { ?>
        <li><?php echo $livro['titulo'] ?> (<a href="index.php?module=compras&action=adicionar&id=
               <?php echo $livro['id'] ?>">Comprar</a>)</li>
<?php } ?>
    </ul>
```

views/ver.tpl.php

```
<?php print_r($data); ?>
```

Livros.php precisa estar pronto para entender que método de Banco.php chamar, de acordo com o que for passado à ele via POST

views/inserir.tpl.php

```
<form action="" method="POST">
     <p><input type="text" name="titulo" /></p>
     <p><input type="text" name="conteudo" /></p>
     <p><input type="submit" name="insert" value="Enviar" /></p>
</form>
```

views/alterar.tpl.php

```
<form action="" method="POST">
     <p><input type="text" name="titulo" value="<?php echo $data['titulo'] ?>" /></p>
     <p><input type="text" name="conteudo" value="<?php echo $data['conteudo'] ?>" /></p>
     <p><input type="submit" name alterar value="Enviar" /></p>
</form>
```

Prática

Hoje vimos o Carrinho de Compras e a Lista de Produtos.

A idéia é que vocês criem métodos para colocar as compras em Banco de Dados, afinal não faremos integração com boleto ou pagseguro – pelo menos por enquanto!

Como já foi dito anteriormente, fique a vontade para criar um MVC mais simples se preferir.

Lembre que foram passados outros dois esqueletos de MVC: o MVC-Simples e o MVC-Final. Eles poderão servir como base para que você crie sua loja from scratch – do zero.