Magento

Alterador de Parcelas com produtos configuráveis + Módulo com layout.

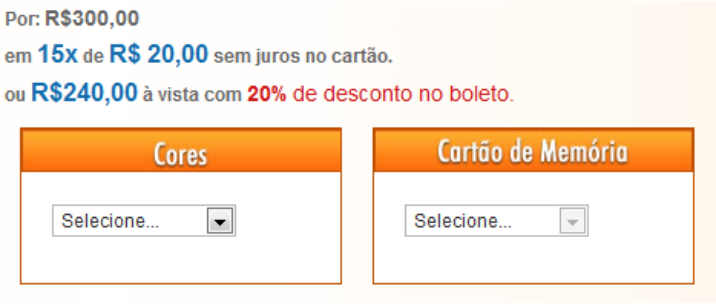
Esse tutorial foi corrigido por causa de um problema de delay no FIREFOX. Seguindo-o todos os passos, entenderá como criar um módulo com layout + ajax + bloco estático.
Seguindo somente os passos que não deixei uma mensagem:

Obs: O código abaixo serve de exemplo de como criar um módulo com layout, porém para a parte de parcelas ocorre um problema de lentidão (por ter que carregar o layout inteiro) no FIR EFOX. Aconselho a desconsiderar o passo abaixo.Para o nosso módulo, não necessitamos de layout.

Você criará um módulo que realizará o AJAX e retornará as parcelas.

Resultado Final:

Antes de selecionar as opções:



Depois da seleção de cor e cartão de memória:

Repare: Os campos cor e cartão de memória foram alterados, e seu preço também, fazendo a parcela aumentar.
Os passos para se criar a parte funcional da parcela são:
1º Criar um bloco chamado parcela e chamá-lo na página de visão do produto.
2º Criar um módulo para qual o AJAX fará a requisição da alteração do produto.

Criando o bloco

Para criar o bloco simples que irá ser chamado na página de visão do produto, crie o seguinte arquivo:
App/design/frontend/**default/default**/catalog/product/view/**parcelas.phtml**

Parcelas.phtml

```
<div id="parcelado" >

  <?php

  $_product = $this->getProduct();
  $capital = $_product->getPrice();
  $_max_parcelas = 15; // Máximo de parcelas no cartão
  $cartao =number_format($capital /$_max_parcelas,2,',','.');
  $boleto = $capital * 0.8; //Desconto a vista no boleto
  echo '<font style="color:#656565"><b>Por:</b></font> <font style="font-size:14px; font-weight:bold;
color:#656565;">R$'.number_format($capital,2,',','.').'</font><br/> <font style="color:#656565"><b>em</b></font>
<font style="font-size:16px; font-weight:bold; color:#156EAF;">'.$_max_parcelas.'x</font> <font
style="color:#656565"><b>de</b></font> <strong><font style="font-size:16px; font-weight:bold; color:#156EAF;">R$
'.$cartao.'</font></strong> <font style="color:#656565"><b>sem juros no cart&atilde;o.</b></font> </font><br/>
<font style="color:#656565"><b>ou</b></font> <font style="font-size:16px; font-weight:bold;
color:#156EAF;">R$'.number_format($boleto,2,',','.').'</font> <font style="color:#656565"><b>à vista com</b></font>
<font style="color:#CF1919; font-size:14px;"><b>20%</b> de desconto no boleto.';
  ?>
</div>
```

Detalhes que devem ser percebidos: A div possui um id=”**parcelado**”. Utilizaremos para poder realizar o AJAX.

Continuando...

Agora para que nosso bloco apareça na página de visão do produto, precisamos adiciona-lo ao arquivo de layout da página de visão.
O arquivo é **catalog.xml,** e encontra-se em: App/design/frontend/**default/default**/layout/**catalog.xml**.
Procure por : <catalog_product_view translate=”label”>
Cole dentro desse tag:

Catalog.xml

```
<block type="catalog/product_view" name="product.parcelas" as="parcelas"
template="catalog/product/view/parcelas.phtml"/>
```

Referenciando no layout, conseguimos chamá-lo no template.
Abra o arquivo: App/design/frontend/default/default/catalog/product/**view.phtml**

View.phtml

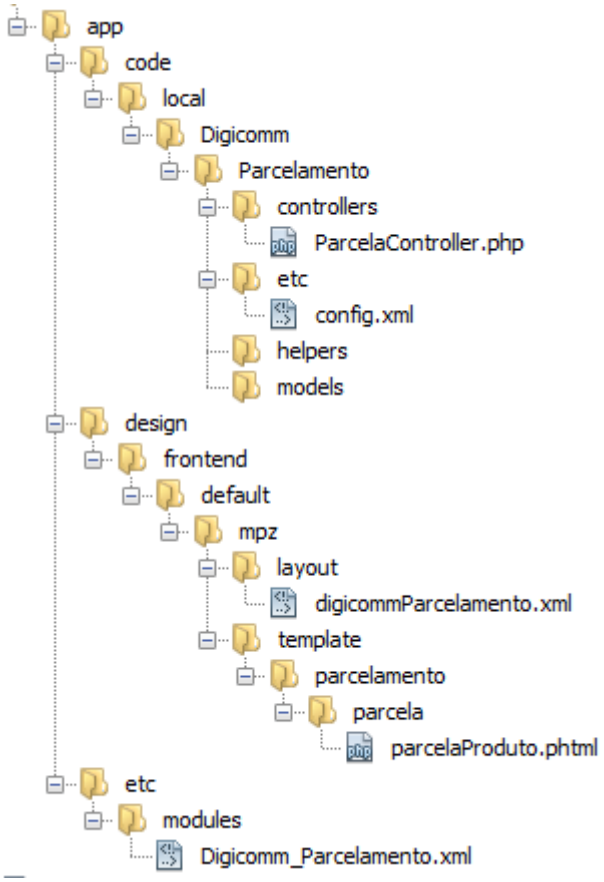
```
<?php echo $this->getChildHtml("parcelas"); ?>
```

A função getChildHtml busca no layout o bloco que possui o atribute **as** igual ao pedido (as ="parcelas").
Feito isso você já deverá ver a parte de parcelado.

Agora, e se eu criar um produto configurável, seu preço é alterado via AJAX, como o preço das parcelas serão alterados?

Para corrigir esse problema, precisaremos de uma página para realizarmos essa requisição, e aproveitando isso, vamos criar um módulo, que poderá ser usado em outras ocasiões, sendo mais fácil de ser recordado, do que se você colocar em pastas diferentes.

Vamos criar a estrutura do módulo:



Breve explicação:

ParcelaController.php : Será utilizada para controle em relação a URL, junto com o arquivo **config**.xml.
O parcelaController contribui com "**Controller/Action**" e o config com o "**router**".

No final, sua URL ficará: www.seusite.com/index.php/**router/Controller/Action**

digicommParcelamento.xml : Onde se definirá o layout para cada página.
Ex: Caso a URL seja: index.php/**SuperMercado/Corredor/Produto** , será buscado no xml (que o nome será definido no config.xml, neste caso, digicommParcelamento.xml), o tag:
<SuperMercado_Corredor_Produto translate="label">

parcelaProduto.phtml : Arquivo de visão, onde você poderá mostrar os resultados.(Caso use layout)
Digicomm_Parcelamento.xml : Ativador do módulo.

Editando os arquivos

Abra o arquivo Digicomm_Parcelamento.xml, vamos ativar o módulo.

Digicomm_Parcelamento.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
   <modules>
      <Digicomm_Parcelamento>
         <active>true</active>
         <codePool>local</codePool>
      </Digicomm_Parcelamento>
   </modules>
</config>
```

Definimos que o módulo Digicomm/Parcelamento está localizado na pasta local, sendo o destino final:
app/code/**local/Digicomm/Parcelamento**.

No arquivo config.xml, vamos definir o módulo também:

Config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<config>

   <modules>
      <Digicomm_Parcelamento>
         <version>0.0.1</version>
      </Digicomm_Parcelamento>
   </modules>
</config>
```

Definimos que o módulo está na versão 0.0.1 (Usado para Updates no banco de dados).

No arquivo ParcelaController.php, vamos criar nossa primeira rota.

ParcelaController.php

```
class Digicomm_Parcelamento_ParcelaController extends Mage_Core_Controller_Front_Action {
   public function indexAction() {
         echo  “Funcionou!”;
     }
}
```

Resumindo, na URL, ficará:

index.php/router/**Parcela/index**

Agora que criamos nossa ParcelaController.php, vamos setar nossa router no **config**.xml

Dentro do tag <config> , coloque:
<frontend>

Config.xml

```
<frontend>
   <routers>
      <Parcelamento>
        <use>standard</use>
        <args>
          <module>Digicomm_Parcelamento</module>
          <frontName>Parcelamento</frontName>
        </args>
      </Parcelamento>
    </routers>
</frontend>
```

Vejamos a definição dos tags:

<frontend> Indica que o que roteador será usada na parte da frente do website.
<routers> É onde você declara todas as rotas.
<Parcelamento> É o identificador desta rota.
<use>Standart</use> Pode ter o valor standart (para a visão frontal) ou admin (para a parte administrativa)
<module>Digicomm_Parcelamento</module> Indica qual módulo contém o controller que manuseia essa rota
<frontName> Parcelamento </frontend> É o nome da rota que será usado na URL.

Feito isso, o módulo está quase pronto, já possibilitando que o vejamos.
Teste a URL:
Seusite.com.br/index.php/**Parcelamento/Parcela/index**
Se tudo estiver ok, ele deverá mostrar a mensagem de "Funcionou!"

Continuando...
Agora precisamos criar um layout, para que possamos trabalhar com o arquivo **parcelaProduto**.pthml.
Para que possamos mexer com o arquivo de layout, precisamos referencia-lo no config.xml.
Adicione dentro do tag frontend:

Obs: O código abaixo serve de exemplo de como criar um módulo com layout, porém para a parte de parcelas ocorre um problema de lentidão (por ter que carregar o layout inteiro) no FIR EFOX. Aconselho a desconsiderar o passo abaixo.Para o nosso módulo, não necessitamos de layout.

Config.xml

```
<frontend>
      ... (<routers> ...</routers>)
  <layout>
        <updates>
          <Parcelamento>
            <file>digicommParcelamento.xml</file>
          </Parcelamento>
        </updates>
    </layout>
</frontend>
```

Definimos que o módulo Parcelamento, terá como arquivo de layout, o arquivo **digicommParcelamento**.xml

Vamos criar o layout para o: index.php/**Parcelamento/Parcela/index**

Obs: O código abaixo serve de exemplo de como criar um módulo com layout, porém para a parte de parcelas ocorre um problema de lentidão (por ter que carregar o layout inteiro) no FIR EFOX. Aconselho a desconsiderar o passo abaixo.Para o nosso módulo, não necessitamos de layout.

digicommParcelamento.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<layout version="0.1.0">
  <parcelamento_parcela_index>
    <reference name="root">
      <action method="setTemplate">
        <template>page/3columns.phtml</template>
      </action>
    </reference>
    <reference name="content">
      <block type="core/template" name="parcelamento.parcelas"
template="parcelamento/parcela/parcelaProduto.phtml"></block>
    </reference>
  </parcelamento_parcela_index>
</layout>
```

Resumindo: Ele terá o template de 3 colunas utilizará o arquivo **parcelaProduto**.pthml.

Para que você possa ver o layout funcionando, no ParcelaController.php, adicione as seguintes linhas dentro da function IndexAction (Caso queira aprender a usar o layout, coloque **somente** as linhas em **laranja**) caso queira seguir o tutorial para o módulo de Parcela, somente as linhas em **verde**.

ParcelaController.php

```
Public function indexAction(){
    $valor = $this->getRequest()->get('valor');
    $valorFormatado = explode("R$", $valor);
    $valorFormatado = explode("</b>",$valorFormatado[1]);
    $valorFormatado= $valorFormatado[0];
    $valor = str_replace(".","",$valorFormatado);

    $max_parcelas = 15;
    $desconto_boleto = 0.8; //20% de desconto
    $desconto_texto = "20%";

    $cartao = $valor/$max_parcelas;
    $boleto = $valor* $desconto_boleto;

    ?>

<font style="color: rgb(101, 101, 101);">
  <b>Por:</b>
```

```
</font>
<font style="font-size: 14px; font-weight: bold; color: rgb(101, 101, 101);">
   R$ <?php echo number_format($valor,2,',','.');?>     <img title="Aguarde..." alt="Aguarde..."
src="http://www.mpzshop.com/skin/frontend/enterprise/mpzee/images/loading/loader1.gif"
id="parcela_atualiza" class="hidden" style="vertical-align: middle;">   </font>
<font style="color: rgb(101, 101, 101);">
   <b>em</b>
</font>
<font style="font-size: 16px; font-weight: bold; color: rgb(21, 110, 175);">
   15x
</font>
<font style="color: rgb(101, 101, 101);">
   <b>de</b>
</font>
<strong>
   <font style="font-size: 16px; font-weight: bold; color: rgb(21, 110, 175);">
      R$ <?php echo number_format($cartao,2,',','.')?>
   </font>
</strong>
<font style="color: rgb(101, 101, 101);">
   <b>sem juros no cartão.</b>
</font>
<br>
<font style="color: rgb(101, 101, 101);">
   <b>ou</b>
</font>
<font style="font-size: 16px; font-weight: bold; color: rgb(21, 110, 175);">
   R$ <?php echo number_format($boleto,2,',','.'); ?>   </font>
<font style="color: rgb(101, 101, 101);">
   <b>à vista com</b>
</font>
<font style="color: rgb(207, 25, 25); font-size: 14px;">
   <b>20%</b> de desconto no boleto
</font>
      <?php
/*
Caso queira mostrar o layout, use somente essa parte.
Novamente, não apaguei para servir de exemplo de como adicionar um módulo com layout.
$this->loadLayout();
 $this->renderLayout();
*/
   }
```

Usando layout: Feito isso, deverá carregar o layout de 3 colunas sem nenhum conteúdo no centro.
Seguindo Tutorial : Feito isso, o módulo estará completo.
Agora vamos editar o conteúdo da página **parcelaProduto**.pthml

Obs: O código abaixo serve de exemplo de como criar um módulo com layout, porém para a parte de parcelas ocorre um problema de lentidão (por ter que carregar o layout inteiro) no FIR EFOX. Aconselho a desconsiderar o passo abaixo.Para o nosso módulo, não necessitamos de layout.

parcelaProduto.pthml

```
<?php
      $valor = $this->getRequest()->get('valor');
```

```
	$valorFormatado = explode("R$", $valor);
	$valorFormatado = explode("</b>",$valorFormatado[1]);
	$valorFormatado= $valorFormatado[0];
	$valor = str_replace(".","",$valorFormatado);

	$max_parcelas = 15;
	$desconto_boleto = 0.8; //20% de desconto
	$desconto_texto = "20%";

	$cartao = $valor/$max_parcelas;
	$boleto = $valor* $desconto_boleto;

	?>

<font style="color: rgb(101, 101, 101);">
  <b>Por:</b>
</font>
<font style="font-size: 14px; font-weight: bold; color: rgb(101, 101, 101);">
  R$ <?php echo number_format($valor,2,',','.');?>     <img title="Aguarde..." alt="Aguarde..."
src="http://www.mpzshop.com/skin/frontend/enterprise/mpzee/images/loading/loader1.gif"
id="parcela_atualiza" class="hidden" style="vertical-align: middle;">   </font>
<font style="color: rgb(101, 101, 101);">
  <b>em</b>
</font>
<font style="font-size: 16px; font-weight: bold; color: rgb(21, 110, 175);">
  15x
</font>
<font style="color: rgb(101, 101, 101);">
  <b>de</b>
</font>
<strong>
  <font style="font-size: 16px; font-weight: bold; color: rgb(21, 110, 175);">
    R$ <?php echo number_format($cartao,2,',','.')?>
  </font>
</strong>
<font style="color: rgb(101, 101, 101);">
  <b>sem juros no cartão.</b>
</font>
<br>
<font style="color: rgb(101, 101, 101);">
  <b>ou</b>
</font>
<font style="font-size: 16px; font-weight: bold; color: rgb(21, 110, 175);">
  R$ <?php echo number_format($boleto,2,',','.'); ?>   </font>
<font style="color: rgb(101, 101, 101);">
  <b>à vista com</b>
</font>
<font style="color: rgb(207, 25, 25); font-size: 14px;">
  <b>20%</b> de desconto no boleto
</font>
```

Detalhe: O preço do produto será enviado via metodo **post**, e terá o nome = “valor”.
Lembre-se disso porque será utilizado para fazer a requisição AJAX.
(Agora, caso você queira ver a página, dará erro, pelo motivo de estar tentando quebrar um array que não existe)
Lembre-se que esta página será acessada somente via AJAX, por isso mesmo que tenha esse erro, **não** edite!

Recordando...

Criamos um bloco e um módulo, que serão usados para o parcelamento do produto.
O primeiro (somente bloco) será usado como "Inicial", já o segundo (módulo) será usado via AJAX. Esse módulo só será utilizado por **produtos configuráveis,** não se esqueça!
Produtos simples não chegarão a ver esse módulo porque não possuem alteração no preço.

Ok, agora criei tudo o que você disse, e já possuo um produto configurável.

Abra o arquivo view.pthml, que se encontra em: app/design/frontend/default/default/catalog/product/**view.pthml**

No final do arquivo, adicionaremos nossa função AJAX para chamar nosso módulo.

PS: Essa função foi criada em Jquery, por ser mais fácil compreender do que prototype.
Caso ainda não tenha o JQUERY no seu Magento:

Abra o **Admin > Sistema > Magento Connect > Abrir Magento Connect**, faça login e cole no campo que diz **Paste extension key to install**:

```
magento-community/Mxperts_Jquery_Tools
```

Clique em **install.**
Tendo instalado, faça o logout do admin e entre novamente (**Necessário**).
Vá em **Admin>Sistema>Configurações>Jquery** (Barra lateral a esquerda)
Ative o Jquery e pronto!.

Agora que você está com o jQuery, vamos continuar.

view.pthml (No final do arquivo)

```
<script type="text/javascript">
   //<![CDATA[
   jQuery(document).ready(function(){
      jQuery("select").change(function(){

         jQuery.ajax({
            url: '<?php echo $this->getUrl("Parcelamento/Parcela/index")?>',
            type:"POST",
            data:({"valor": jQuery('.price').html()}),
            success: function(data) {
               jQuery('#parcelado').html(data);
            }
         });
      })
   })
   //]]>
</script>
```

Resumindo: Quando o cliente clicar em qualquer select da página, o jquery fará uma requisição via Ajax, para o nosso módulo. Os dados serão enviados via POST, e a variável enviada terá o nome "valor" e enviará um o que estiver escrito num com a class="price".

Feito isso, a parte de parcelamento estará completa!

Qualquer dúvida, problema me envie uma mensagem em: gabriel.sc310@hotmail.com