MANUAL PARA DESENVOLVEDORES
MODULOS DE PAGAMENTO

Neste material você encontra todas as informações necessárias ao desenvolvimento de módulos de pagamento compatíveis com o gerenciador HOSTMGR.

O primeiro passo para desenvolver é compreender como o modulo é organizado dentro do sistema. Para tanto, tenha em mente que cada modulo deve ser representado por um arquivo com nome exclusivo, escrito em minúsculo e armazenado no diretório **/modulos/pagamentos** sendo seu respectivo arquivo de retorno (caso aplicável) armazenado, com o mesmo nome, no diretório **/módulos/pagamentos/retorno.**

Junto desse material você recebeu os arquivos de exemplo para o desenvolvimento do primeiro modulo, que desmembramos abaixo.

Variável $moduloPagamento

```
1  <?
2  # MODULO DE PAGAMENTO DE EXEMPLO
3
4  $moduloPagamento=array('nome'=>'exemplo','nomeVisivel'=>'Modulo de exemplo');
```

É uma variável do tipo array que recebe informações básicas sobre o modulo, onde "**nome**" deve ser o mesmo nome dado ao arquivo que contem as informações do modulo sem a extensão "**.php"**, nesse caso, **"exemplo"**.

No mesmo sentido, a chave "**nomeVisivel**" é o nome da forma de pagamento mostrada ao usuário quando da utilização do modulo. Esse valor é apenas uma referencia, pois pode ser alterado pelo administrador após ativar o modulo no sistema.

Variável $dadosModuloPagamento

```php
$dadosModuloPagamento=array(

    /*ex: array('a',        'b',        'c',        300,    d,      e,  f,  g),

    a = tipo de campo (textfield, select, textarea, radio, checkbox, password)
    b = nome/Rotulo do campo, legenda a ser exibida ao usuario
    c = nome do campo, parametro a ser utilizado
    300 = largura do campo em pixels
    d = altura do campo (aplicavel ao textarea)
    e = campo obrigatorio (true = sim, false = nao)
    f = array com valores das opções, usado em select e radio ex: array('opcao1' => 1, 'opcao2' => 2)
    g = rotulo adicional ao usuario, aplicado ao final do campo

    */

    array('textfield', 'Token', 'token', 300, '', true, '', 'seu token'),
);
```

Contem informações sobre os campos de configuração do modulo. Neste array são gerados os campos que o administrador deverá preencher ao ativar o modulo.

Os campos comumente cadastrados são: Email, Conta, token, taxa, dentre outros dados configuráveis que você possa precisar para criar o seu modulo de pagamento.

Os campos são gerados na ordem cadastrada seguindo a regra de um array/campo por linha como no arquivo de exemplo.

Função moduloFatura_exemplo()

```php
function ModuloFatura_exemplo(){

    global $parametros, $config;

    /*$parametros['fatura']
      $parametros['cliente']
      $parametros['moduloPagamento']*/

    $codigo = 'Código html/javascript visivel na fatura.<br />Token: ';
    $codigo.= $parametros['moduloPagamento']['token'];

    return $codigo;

}
```

O nome dessa função é composto pelo prefixo “ModuloFatura_” sucedido do nome de seu modulo. No nosso caso, “exemplo”.

A função é executada toda vez que o cliente final abre uma fatura que utiliza esse modulo/forma de pagamento, ou quando a forma de pagamento em questão é selecionada.

As principais informações estão armazenadas na variável **$parametros** onde estão organizados os dados distribuídos em arrays.

$parametros[‘fatura’] – Array que contem os dados referentes a fatura como (id_fatura, valor, valor_pago, data_vencimento)

$parametros[‘cliente’] – Array que contem os dados do cliente como (nome, empresa, telefone, email, endereço, numero, bairro, cidade, estado, cpf_cnpj)

$parametros[‘moduloPagamento’] – Array que contem os dados referentes ao seu modulo. São os valores configurados para os campos que você criou. No nosso exemplo, **token** e **campo2**

Caso haja alguma dúvida quanto ao correto nome do campo a ser utilizado você pode listar todos os campos por meio da função print_r.

Exemplo:

```
<?php
echo’<pre>’;
print_r($parametros[‘cliente’]); //lista todas as variáveis e valores do array cliente
echo’</pre>’; ?>
```

RETORNO AUTOMÁTICO DE DADOS

O retorno automático é parte crucial para quem pretende ter um sistema totalmente automatizado. Sua função é realizar a baixa automática da fatura sempre que receber uma notificação de pagamento.

Em regra, quando se integra um gateway de pagamento você poderá cadastrar uma URL de Retorno, que receberá os dados referente a transação que a deu origem, ou enviá-la na primeira etapa da integração.

Junto desde material você também recebeu um outro arquivo de exemplo que fica localizado no diretório **/modulos/pagamentos/retorno.** Esse arquivo é o responsável por tratar o retorno de dados e realizar a baixa da fatura de forma automática.

Normalmente a URL de retorno cadastrada em seu gateway ficará parecida como a URL abaixo.

http://www.suaempresacomhostmgr.com.br/modulos/pagamentos/retorno/exemplo.php

Entendendo o funcionamento do arquivo de retorno

```php
<?php
# MODULO DE PAGAMENTO DE EXEMPLO -> RETORNO AUTOMÁTICO

include"../../../includes/conexao.php";
include"../../../includes/loadConfig.php";
include"../../../admin/includes/comandos.php";

$moduloPagamento=array();

$sql=$banco->consultar("select * from modulospagamento where modulo='exemplo'");

foreach($banco->lista($sql) as $linha){
    $moduloPagamento[$linha['parametro']]=$linha['valor'];
}

# ------------ Se condigo começa abaixo --------------------------
```

O código acima é a primeira etapa do módulo de retorno, seu conteúdo não deve ser removido.

Note que de acordo com o código a variável **$moduloPagamento** é um array que contem as informações dos campos que você criou ao desenvolver o modulo.

Assim como no exemplo anterior, seus valores seriam **$moduloPagamento['token']** e **$moduloPagamento['campo2']**

```
17 if (count($_POST) > 0) { // Verifica se algum dado foi recebido
18
19     $id_fatura=(int)$_POST['id_fatura'];
20     $status=$_POST['status'];
21
22     if($status=='pago'){
23
24         $fatura=carregaParametrosFatura($id_fatura);
25
26         $res=adicionarPagamento($fatura['id_fatura'], $transacao, $fatura['valor'],"exemplo");
27
28         if ($res) {
29             echo"Sucesso";
30         } else {
31             echo"Erro ao confirmar pagamento";
32         }
33
34     }else{
35         echo"A fatura não foi paga";
36     }
37
38 } else {
39     // POST não recebido, indica que a requisição é o retorno do Checkout exemplo.
40     // No término do checkout o usuário é redirecionado para este bloco.
41     echo'<h3>Obrigado por efetuar a compra.</h3>';
42 }
```

A partir da linha 16 você deve trabalhar o seu código de acordo com as necessidades do modulo escolhido

Função carregaParametrosFatura($id)

Essa função é responsável por recuperar as informações da fatura. Ao chama-la você deve indicar o numero da fatura de que precisa obter os dados, normalmente recebido via POST ou XML quando recebe o retorno.

Com as informações retornadas você poderá verificar a consistência dos dados recebidos, como por exemplo verificar se o valor pago é suficiente para quitação da fatura.

Função adicionarPagamento(fatura, transação, valor, modulo)

A função é o coração do retorno automático, ela é responsável pela confirmação do pagamento e consequente liberação de serviços caso configurado para o produto.

Você deve carrega-la com quatro parâmetros:

- Fatura => Número/id da fatura que está sendo baixada/paga
- Transação => Código da transação retornado por seu gateway de pagamento
- Valor => Valor que está sendo pago no formato 9999.99
- Modulo => Nome do seu modulo de pagamento. De acordo com nosso exemplo deve-se utilizar “exemplo”

Com as informações acima você será capaz de desenvolver módulos de pagamentos compatíveis com o HOSTMGR.

Opcionalmente caso queira distribuí-lo você poderá contar com nosso sistema de licenças para disponibilizar e gerenciar suas chaves e clientes.

Caso haja qualquer dúvida contate nosso suporte por meio de ticket em nossa central do cliente (http://central.hostmgr.com.br). Criamos um departamento específico para desenvolvedores.