

Nesta seção você encontra artigos sobre a linguagem PHP e a ferramenta Delphi for PHP

Relatórios no PHP com arquivos PDF

Aprenda a gerar arquivos PDF no PHP para emissão de relatórios



Resumo DevMan

Todos nós sabemos que grande parte dos sistemas de como um todo acabam não tendo tanta utilidade quando não há relatórios. É assim com aplicações Win32, mais cedo ou mais tarde o cliente pede para desenvolver algum tipo de relatório, tais como: comissão de vendas, vendas no mês, controle de estoque, gráfico de vendas, fluxo de caixa etc.

É possível desenvolver relatórios em Delphi for Win32 numa facilidade incrível, já que temos diversos editores de relatórios para todo gosto e bolso.

Na web não é diferente, os relatórios serão solicitados e teremos que estar preparados para desenvolver a solução para o cliente.

A diferença maior entre aplicações Win32 e Web é que em Web a gama de opções para o desenvolvimento é bastante pequena. Por isso, nesse artigo quero mostrar uma boa solução para

desenvolver relatórios na Web utilizando a biblioteca FPDF que é específica para programação PHP.

Nesse artigo veremos

- Como utilizar a classe FPDF para criar arquivos PDF;
- Como gerar relatórios em PDF para Web Sites PHP.

Qual a finalidade

- Implementar uma solução para relatórios, em formato profissional, no PHP.

Quais situações utilizam esses recursos?

- Desenvolvimento de sistemas e sites com PHP onde se necessita de relatórios para obter-se informações.



Fabrício Desbessel

(fabricao@fabricao.pro.br)

é professor de Linguagem de Programação do Curso Técnico em Informática do Centro Tecnológico Frederico Jorge Logemann de Horizontina/RS e da FAHOR Faculdade Horizontina. Delphiano de coração está sempre disposto a provar que com o Delphi sempre teremos a melhor solução, até mesmo com o PHP. Site www.fabricao.pro.br.

Quem tem uma aplicação WEB sabe que haverá solicitação de relatórios que deverão ter uma aparência profissional, que não percam a formatação de uma impressora para outra e, o mais importante, possam ser salvos para arquivo ou enviados via e-mail.

O formato ideal para relatórios na Web atualmente é o PDF, pois além de imprimir nas mais variadas impressoras sem perder formatação, podemos guardar

os arquivos e enviá-los por e-mail ou qualquer outra mídia.

Portanto, nesse artigo criaremos arquivos PDF utilizando a biblioteca *PDFLib* do PHP. Vamos aprender na prática!

Preparando a utilização

Como o *FPDF* é uma classe, basta fazer o *download* do mesmo e colocar em um diretório dentro da sua aplicação. Para baixar acesse o site www.fpdf.org e o link *Downloads*. Escolha a versão mais atual

até o fechamento dessa edição. Depois de fazer o *download* descompacte o arquivo. Como resultado, tem-se o arquivo da classe (*fpdf.php*) que deverá ser copiada para o diretório da sua aplicação, bem como a pasta *font* que contém as fontes *TrueType* que poderão ser utilizadas.

Os demais arquivos são a documentação, um tutorial e informações. No *site* também existe um tutorial traduzido para o português do Brasil que pode ser baixado.

FPDF é liberado sob uma licença permissiva: não há qualquer restrição de utilização. Você pode inseri-lo livremente na sua aplicação (comercial ou não), com ou sem modificações.

Conhecendo alguns recursos disponíveis

Para geração de arquivos *PDF*, a *FPDF* oferece suporte para inserir imagens nos formatos *JPEG*, *PNG* e *GIF*. Para a edição do texto, qualquer fonte *TrueType* pode ser utilizada, desde que seja distribuída juntamente com a aplicação, podendo-se também definir a cor das fontes. Quanto à formatação do texto, é possível criar cabeçalhos e rodapés, criar *links* hipertexto, sendo também possível a quebra automática de linhas e páginas. Além de também definir as medidas das margens.

Nota: Para inserir nos relatórios imagens do tipo *Gif*, será necessário ter carregada no *PHP* a biblioteca GD. Em instalações do *Delphi for PHP* essa biblioteca já vem carregada. Se não seu caso, verifique o arquivo *PHP.ini*.

E o melhor de tudo é que o arquivo *PDF* resultante estará devidamente compactado, o que torna sua exibição mais rápida, em um tempo aceitável para aplicações *Web*.

Construindo os primeiros relatórios

Vamos iniciar nosso primeiro teste criando uma aplicação bastante conhecida de todos, a famosa e tradicional mensagem: *Ola Mundo!* Abra seu editor preferido e digite o código da **Listagem 1**. Minha preferência é pelo *Delphi For PHP*, para agilizar a execução dos testes e ter a possibilidade de *debug* do código *PHP*,

Listagem 1. Relatório Olá Mundo!

```
<?php
define('FPDF_FONTPATH','C:/CD/PDF/fpdf16/font/');
require('fpdf16/fpdf.php');
$pdf=new FPDF();
$pdf->AddPage();
$pdf->SetFont('Arial','B',50);
$pdf->Cell(40,10,'Olá Mundo!');
$pdf->Output();
?>
```

Listagem 2. Relatório com várias linhas seguidas

```
<?php
define('FPDF_FONTPATH','C:/CD/PDF/fpdf16/font/');
require('fpdf16/fpdf.php');
$pdf=new FPDF('P','mm','A4');
$pdf->AddPage();
$pdf->SetFont('Arial','',10);
$pdf->Cell(190,4,'Olá Mundo!',1,1);
$pdf->Cell(190,4,'Proxima Linha 1',1,1);
$pdf->Cell(190,4,'Proxima Linha 2',1,1,'C');
$pdf->Cell(190,4,'Proxima Linha 3 - Texto que será alinhado à direita',1,1,'R');
$pdf->Output();
?>
```

mas sinta-se a vontade para utilizar seu software de preferência, como o *Bloco de Notas* do *Windows*, por exemplo.

A primeira linha de comando no código da **Listagem 1** serve para definir o valor de uma constante, utilizada pela classe para saber o diretório onde estão as fontes. Perceba que no meu caso aponte para *C:/CD/PDF/fpdf16/font/*, ou seja, local. Obviamente quando distribuir essa aplicação terei que modificar o caminho para que se adapte as *urls* de meu sistema *on-line*. Mais abaixo vemos uma chamada ao método *SetFont*, que serve para escolher a fonte, sua formatação e tamanho. O comando *require* do *PHP* inclui o arquivo passado como parâmetro no *script* atual, exigindo sua existência. Essa inclusão é necessária para adicionar o código da classe *FPDF* ao nosso *script PHP*.

Para iniciar a criação de um relatório declara-se uma variável que irá receber um objeto da classe *FPDF*. Esse objeto representa o arquivo *PDF* que vamos criar. E ele adicionamos uma página através do método *AddPage* e, para escrever utiliza-se o método *Cell* que tem como parâmetro os tamanhos *width*, *height* e o texto a ser impresso.

O *Cell* na verdade define uma célula com largura (*width*) e altura (*height*) onde pode-se, inclusive, centralizar ou alinhar o texto na direita e na esquerda. Para exibir o relatório criado, utiliza-se o método *Output* que faz uma saída redirecionando a página para um arquivo *PDF* que

deverá ser carregado no *browser*.

Agora vamos evoluir um pouco o relatório codificando conforme a **Listagem 2**, onde acrescentamos alguns parâmetros nos métodos.

Na **Listagem 2** criamos o relatório e já configuramos a orientação e formato da folha. Isso é feito na linha

```
$pdf=new FPDF('P','mm','A4');
```

O primeiro parâmetro (P) está definindo a orientação *Portrait* (retrato). Se você deseja fazer um relatório no formato horizontal, defina o parâmetro como (L) que significa *Landscape* (paisagem). Também pode-se definir a unidade de medida do relatório, esse é o segundo parâmetro. No caso, foi definido em milímetros (mm), mas pode-se utilizar: (cm) *centímetros*, (pt) *pontos* e (in) *polegadas*. Essa definição de unidade de medida será aplicada no documento todo e será utilizada para qualquer posicionamento dentro do mesmo. Por isso é muito importante definir a unidade padrão antes de organizar os relatórios, pois mudar depois poderá desorganizar completamente a formatação. Finalizando os parâmetros do relatório pode-se definir o formato da folha e os valores que podem ser utilizados são: *A3*, *A4*, *A5*, *Letter* e *Legal*. Lembrando que no Brasil o formato padrão é o *A4*.

O método *AddPage* também possui o parâmetro de orientação e formato da página, o que possibilita que você tenha páginas diferenciadas dentro do mesmo relatório.

Listagem 3. Relatório com cabeçalho e rodapé

```
<?php
define('FPDF_FONTPATH','C:/CD/PDF/fpdf16/font/');
require('fpdf16/fpdf.php');

class RELATORIO extends FPDF{
    function Header(){
        $this->Image('logo_CD.gif',10,8);
        $this->SetFont('Arial','B',15);
        $this->Cell(80);
        $this->Cell(100,10,'Relatório de Demonstração do FPDF',0,0,'C');
        $this->Ln(20);
    }
    function Footer(){
        $this->SetY(-15);
        $this->SetFont('Arial','I',8);
        $this->Cell(0,10,'Page '.$this->PageNo().'/{nb}',0,0,'C');
    }
}
$pdf=new RELATORIO();
$pdf->AliasNbPages();
$pdf->AddPage();
$pdf->SetFont('Times','',12);
for($i=1;$i<=40;$i++){
    $pdf->Cell(0,10,'Texto da linha número '.$i,0,1);
    $pdf->Output();
}
?>
```

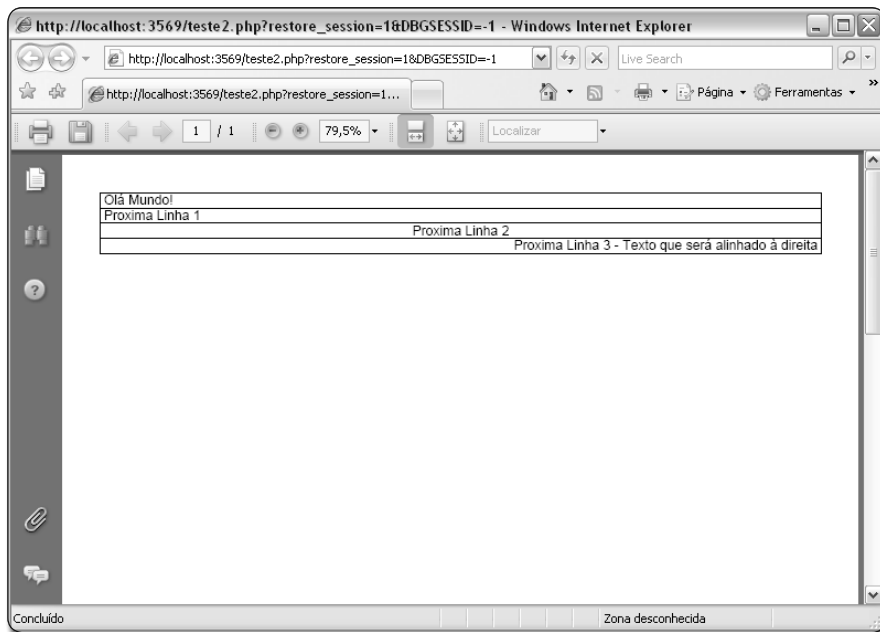


Figura 1. Relatório criado através do script da Listagem 2

Para definir a fonte, utilizou-se o método *SetFont* com o parâmetro de fonte como *Arial*. O parâmetro seguinte está em branco, ou seja, sem nenhuma formatação original, lembrando que é possível formatar como negrito (B), itálico (I) e sublinhado (U). Inclusive pode-se formatar a fonte com todas as opções passando no segundo parâmetro *BIU*. No final do *SetFont* tem-se o tamanho da fonte, em pontos, que, se não for informado será 12 (doze). Ao definir um *SetFont*, essa definição valerá para todo o relatório até que exista outra definição, ou seja, outro *SetFont*.

Nas linhas que chamam o método *Cell* na **Listagem 2**, definiu-se a largura (*width*) em 190, sendo que uma folha do tipo A4 tem 210 mm, deixando 20 mm de margem, ou seja, 2 centímetros. Como altura utilizou-se a medida de 4 mm o que é suficiente para um texto com fonte *Arial* tamanho 10. Ainda no método *Cell*, após o texto que deverá ser escrito colocou-se no próximo parâmetro o número '1' que significa que uma borda será aplicada em toda a célula, o que é bem interessante em relatórios que precisam de linhas a cada registro. Depois

da borda, tem-se mais um parâmetro com '1' para definir uma quebra de linha após o texto. Com isso a próxima chamada ao método *Cell* estará posicionado na linha seguinte.

Na chamada escreve o texto *Próxima Linha 2* tem-se, como último parâmetro a letra *C* que define que o texto deve ser centralizado dentro da célula, levando em conta seu tamanho (largura). Pode-se ainda definir a posição do texto com *R* (direita) e *L* (esquerda), como demonstrase na última chamada ao método *Cell* que alinha o texto na margem direita. Colocando esse *script* para executar tem-se como resultado o exibido na **Figura 1**.

Adicionando cabeçalho e rodapé ao relatório

A classe *FPDF* possui dois métodos que não foram implementados, deixando essa possibilidade de extensão dos mesmos. Esses métodos são o *Header* (cabeçalho) e o *Footer* (rodapé). Portanto, para criar um cabeçalho e rodapé faz-se uma extensão da classe definindo esses dois métodos. É importante salientar a necessidade de utilizar o *\$this* para chamar os métodos da própria classe. Veja na **Listagem 3** um relatório com cabeçalho e rodapé, bem como recursos de imagem e contador de páginas.

No código da **Listagem 3** estende-se (herança) a classe *FPDF* criando a classe *Relatorio*. Com essa extensão podemos definir os métodos *Header* e *Footer*.

No *Header*, invocamos o método *Image* para adicionar uma imagem contida no mesmo diretório do *script PHP*, que será iniciada na posição horizontal 10 e vertical 8. Tem-se a opção ainda de passar mais um parâmetro, o tamanho que a imagem deve ter de largura, assim ela será redimensionada proporcionalmente. No caso da **Listagem 3** não declarou-se o tamanho da imagem e a mesma será exibida no tamanho original. Na seqüência tem-se uma chamada ao *Cell* passando como parâmetro 80 (oitenta), com o objetivo de criar uma célula de espaço entre a figura e o título do relatório que é impresso através da próxima chamada. No final do *Header* tem-se uma chamada ao método *Ln* que cria uma linha em branco com uma

quebra, onde seu parâmetro é a altura que se não for informada será a mesma altura da última chamada ao *Cell*.

No *Footer* vemos uma chamada a *SetY* que serve para posicionar a escrita em relação vertical. O valor negativo refere-se à base do relatório, ou seja, tem-se 15 mm de baixo para cima. Depois imprime-se o número da página atual, através do método *PageNo*, bem como a quantidade de páginas através do *alias {nb}* que é resultado da função *AliasNbPages*. Assim fecha-se a extensão da classe *FPDF*.

Continuando, tem-se a criação do relatório que agora é uma variável do tipo *Relatorio* que é a classe estendida e que contém o cabeçalho e rodapé. Faz-se necessário chamar o método *AliasNbPages* para definir o alias que conterá o número final de páginas do relatório. Define-se uma fonte do tipo *Times* e tem-se um *looping* para imprimir o mesmo texto várias vezes, ocasionando quebras de páginas. Na **Figura 2** apresenta-se o resultado do relatório da **Listagem 3**.

Colorindo o relatório

Para se ter uma idéia do quão interessante é a classe *FPDF*, temos um recurso interessante. Mesmo o relatório sendo *PDF* é interessante utilizar cores para destacar informações importantes e que devem ser rapidamente identificadas. A classe possui isso e veremos que é bem fácil colorir um relatório. Indo direto ao ponto tem-se na **Listagem 4** a criação de um relatório com cores.

Na **Listagem 4**, criamos uma caixa personalizada e nela escrevemos um texto. Utilizamos o método *GetStringWidth* para descobrir o tamanho que o texto contido na variável terá no relatório, levando em conta a unidade de medida do mesmo. A esse resultado soma-se mais 6 (seis) para ter um tamanho ideal da caixa que será criada. Utilizou-se o método *SetX* para posicionar o texto em relação horizontal, fazendo uma conta para posicionar o resultado bem ao centro. Essa conta é simples, pegou-se o tamanho do relatório (210), diminuiu-se o tamanho do texto (\$w) e dividiu-se por dois (2).

O *SetDrawColor* é utilizado para definir a cor das linhas (bordas) que compõem a célula, utilizando como parâmetro o

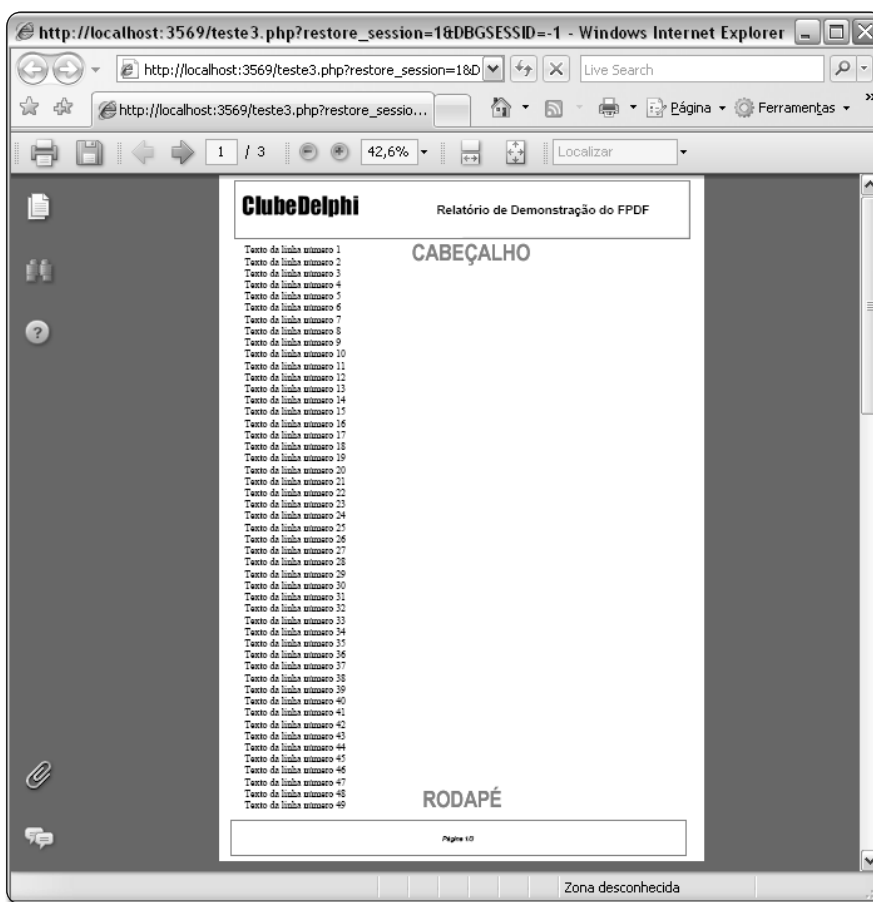


Figura 2. Relatório com cabeçalho e rodapé

Listagem 4. Relatório com cores

```
<?php
define('FPDF_FONTPATH','C:/CD/PDF/fpdf16/font/');
require('fpdf16/fpdf.php');
$pdf=new FPDF();
$texto='Teste com Cores';
$pdf->SetFont('Arial','',14);
$pdf->AddPage();
$w=$pdf->GetStringWidth($texto)+6;
$pdf->SetX((210-$w)/2);
$pdf->SetDrawColor(0,80,180);
$pdf->SetFillColor(230,230,0);
$pdf->SetTextColor(220,50,50);
$pdf->SetLineWidth(1);
$pdf->Cell($w,$texto,1,1,'C',true);
$pdf->Ln(10);
$pdf->Output();
?>
```

padrão de cores *RGB* (*Red*, *Green*, *Blue*), onde define-se a quantidade de vermelho (0 a 255), a quantidade de verde (0 a 255) e a quantidade de azul (0 a 255). Para mudar a cor interna da célula utiliza-se o *SetFillColor* que também utiliza o padrão *RGB*, bem como o método *SetTextColor* que define a cor do texto propriamente. Ainda pode-se definir o tamanho da linha da célula através do método *SetLineWidth*.



Nota do DevMan

No padrão *RGB* todas as cores são formadas por misturas de vermelho, verde e azul, sendo assim cada cor possui uma representação numérica que informa a quantidade de cada mistura. Por exemplo, a cor branca é representada como *RGB*(1,1,1). Para obter esses valores você pode utilizar programas de edição gráfica, como o *Photoshop*.

Listagem 5. Listagem de dados em formato de tabela

```
<?php
define('FPDF_FONTPATH','C:/CD/PDF/fpdf16/font/');
require('fpdf16/fpdf.php');
$MYSQL = mysql_pconnect("localhost","root","root");
mysql_select_db('direto',$MYSQL);
$SQL = "select * from clientes order by nome";
$MResultado = mysql_query($SQL, $MYSQL);
class RELATORIO extends FPDF{
    function Header(){
        $this->Image('logo_CD.gif',10,8);
        $this->SetFont('Arial','B',15);
        $this->Cell(80);
        $this->Cell(100,10,'Listagem de Clientes',0,0,'C');
        $this->Ln(20);
    }
    function Footer(){
        $this->SetY(-15);
        $this->SetFont('Arial','I',8);
        $this->Cell(0,10,'Página '.$this->PageNo().'/{nb}',0,0,'C');
    }
}
$pdf=new RELATORIO();
$pdf->AliasNbPages();
$pdf->AddPage();
$pdf->SetFont('Times','B',12);
//Colunas
$pdf->Cell(20,4,'Código',1,0,'C');
$pdf->Cell(60,4,'Nome',1,0,'C');
$pdf->Cell(30,4,'D.Nasc',1,0,'C');
$pdf->Cell(30,4,'Fone',1,0,'C');
$pdf->Cell(50,4,'E-mail',1,1,'C');
$pdf->SetFont('Times','',12);
//Registros
while ($MColuna = mysql_fetch_assoc($MResultado)){
    $pdf->Cell(20,4,$MColuna["id"],1,0,'C');
    $pdf->Cell(60,4,$MColuna["nome"],1,0,'C');
    $pdf->Cell(30,4,$MColuna["datanasc"],1,0,'C');
    $pdf->Cell(30,4,$MColuna["fone"],1,0,'C');
    $pdf->Cell(50,4,$MColuna["email"],1,1,'C');
}
mysql_free_result($MResultado);
$pdf->Output();
?>
```

Listando dados no formato de tabelas

Até agora, vimos como fazer a impressão de relatórios simples, sem acesso a dados. Agora vamos à parte que mais interessa: uma listagem de dados. Com tudo que já foi escrito nesse artigo fica fácil pensar e fazer uma listagem que coloca, no formato de tabela, o resultado de uma consulta ao banco de dados. Mas, para facilitar vamos ver um exemplo na **Listagem 5**, que busca dados de uma tabela de *Clientes* em um banco *MySQL*.

Na **Listagem 5**, tem-se uma consulta ao banco de dados *MySQL*, em uma tabela de clientes. Montou-se o relatório com cabeçalho e rodapé, como visto anteriormente. Na questão da tabela criaram-se os títulos da coluna de forma centralizada e definindo um tamanho. Na última coluna (*E-mail*) colocou-se 1 no parâmetro que quebra a linha para que o registro seja iniciado abaixo dos títulos.

Com o *while* criou-se um *loop* até o final de registros da consulta imprimindo os campos com as mesmas definições da coluna. Perceba que algumas colunas estão centralizadas e outras não. Na **Figura 3** vemos o relatório finalizado.

Conhecendo todas as possibilidades da classe FPDF

Bem, até agora viu-se os principais métodos da classe *FPDF* que, ao meu ver, são suficientes para criar a maioria dos relatórios. Mas ainda existe mais, veja a **Tabela 1** todos os métodos e a descrição das suas funcionalidades.

Conclusão

Nesse artigo apresentou-se a classe *FPDF* que possibilita, facilmente, criar

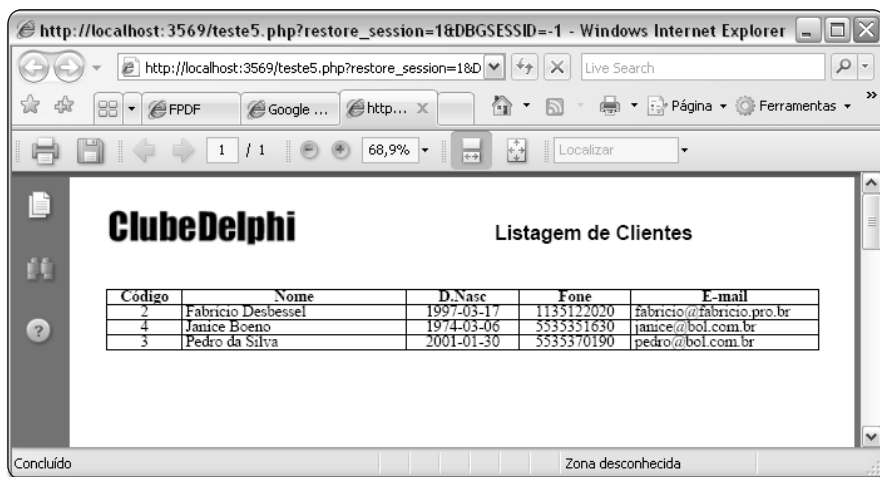
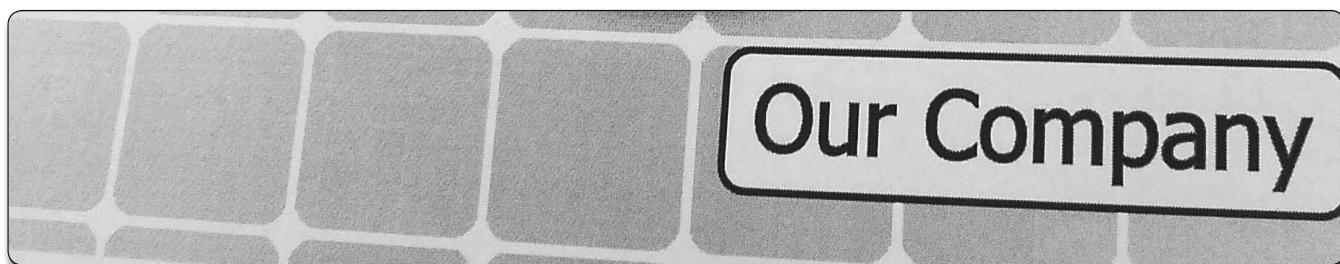


Figura 3. Relatório com dados no formato de tabela



Método	Descrição
AcceptPageBreak	Orienta sobre aceitação ou não de quebra de página automática
AddFont	Possibilita a adição de nova fonte ao relatório
AddLink	Cria um link externo (http)
AddPage	Adiciona uma nova página ao relatório
AliasNbPages	Cria um apelido para variável que controla o número de páginas do relatório
Cell	Formata e imprime uma célula
Close	Fecha (termina) o relatório
Error	Cria um exceção do tipo fatal que aborta a execução do código restante
Footer	Método do rodapé que pode ser estendido na classe filha
FPDF	Nome da classe construtora (pai)
GetStringWidth	Retorna o tamanho do texto levando em conta a unidade de medida
GetX	Retorna a posição horizontal atual do cursor (relatório)
GetY	Retorna a posição vertical atual do cursor (relatório)
Header	Método do cabeçalho que pode ser estendido na classe filha
Image	Imprime uma imagem do tipo PNG, GIF e JPG
Line	Cria uma linha
Link	Insere um link criado pelo método <i>AddLink</i>
Ln	Força uma quebra de linha
MultiCell	Formata e imprime uma célula com várias linhas fazendo quebra automática
Output	Salva e/ou retorna o relatório. Em outras palavras apresenta o mesmo no browser
PageNo	Retorna o número da página
Rect	Cria um retângulo (desenho)
SetAuthor	Define o autor do documento
SetAutoPageBreak	Define o modo de quebra automática
SetCompression	Define se existe ou não compressão do relatório
SetCreator	Define o criador do documento
SetDisplayMode	Define o modo de visualização
SetDrawColor	Define a cor das linhas
SetFillColor	Define a cor de fundo (preenchimento)
SetFont	Define a fonte para escrita
SetFontSize	Define o tamanho da fonte
SetKeywords	Associa palavras chaves ao documento (utilizado para buscas)
SetLeftMargin	Define a margem esquerda do relatório
SetLineWidth	Define a largura da linha
SetLink	Define o destino de um link interno
SetMargins	Define as margens do relatório
SetRightMargin	Define a margem direita do relatório
SetSubject	Define o assunto do documento
SetTextColor	Define a cor do texto que será escrito
SetTitle	Define o título do documento
SetTopMargin	Define a margem superior do relatório
SetX	Seta a posição horizontal para a próxima escrita
SetXY	Seta a posição horizontal e vertical para a próxima escrita
SetY	Seta a posição vertical para a próxima escrita
Text	Imprime um texto (escreve)
Write	Imprime o texto na sequência

Tabela 1. Métodos disponíveis na classe FPDF

relatórios no formato *PDF* para apresentação na *WEB*. Você pode aprender mais sobre essa classe vendo os tutoriais disponíveis no site do projeto: www.fpdf.org. Agora temos uma boa forma de criar relatórios no PHP sendo com *Delphi For PHP* ou não. Coloque a mão na massa e crie vários relatórios em seus sistemas ●

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

