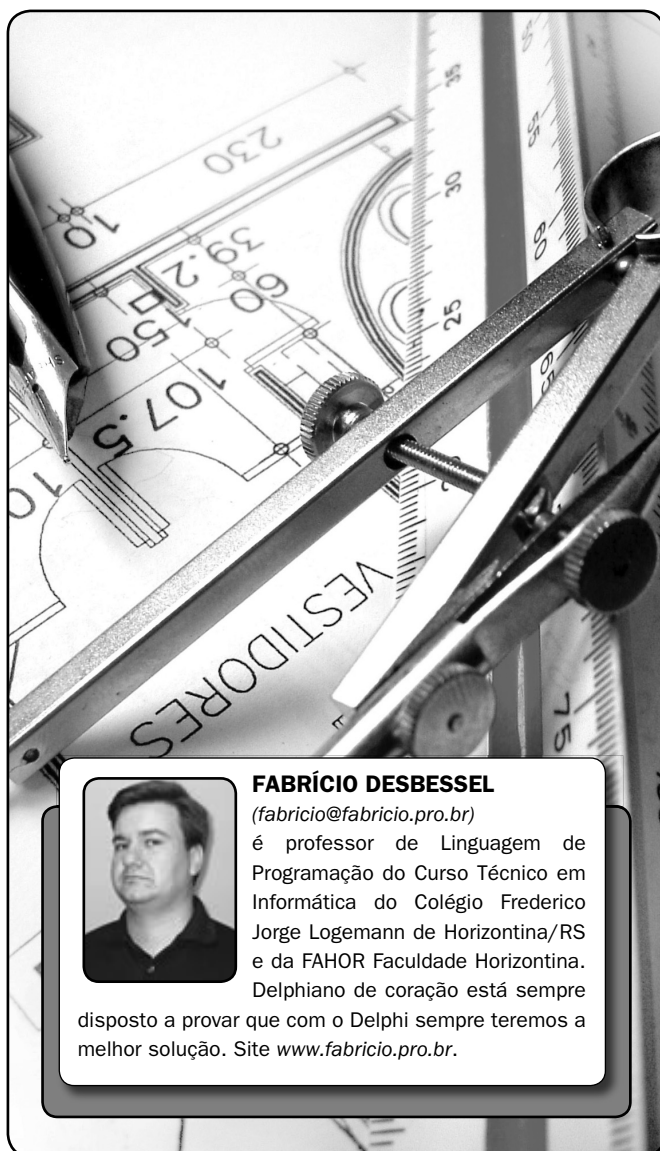


# Como trabalhar com imagens em aplicações ASP.NET



**FABRÍCIO DESBESSEL**

([fabricao@fabricao.pro.br](mailto:fabricao@fabricao.pro.br))

é professor de Linguagem de Programação do Curso Técnico em Informática do Colégio 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. Site [www.fabricio.pro.br](http://www.fabricio.pro.br).

**E**m aplicações Web, é comum a necessidade de trabalhar com imagens. Especialmente com fotos e com serviços de redimensionamento e aplicações de marcas, juntando diferentes imagens. Isso tudo pode ser feito no Delphi for .NET através do namespace *System.Drawing.Imaging* que fornece funcionalidades avançadas de imagem (GDI+). A funcionalidade básica dos gráficos é fornecida pelo namespace de *System.Drawing*.

Neste artigo, veremos como trabalhar com imagens em aplicações ASP.NET, usando os namespaces acima descritos. Veremos com redimensionar uma imagem, além de inserir uma marca d'água.

## Respondendo uma imagem

Inicialmente vamos criar uma aplicação ASP.NET que responderá uma imagem, conforme um campo de consulta (*QueryString*), informado pelo usuário, através do link. Crie uma nova aplicação ASP.NET através do menu *File>ASP.NET Web Application - Delphi For .NET*. Defina um nome para a aplicação e lembre-se de estar com o IIS rodando para que o Delphi crie a pasta virtual.

Para este exemplo nomeie a aplicação como "Foto01". Acesse o *Page\_Load* e digite o código da **Listagem 1**. O código está comentado, de forma que fica fácil o entendimento.

### Listagem 1. Código do Page\_Load

```
var
  Imagem: System.Drawing.Image;
begin
  if System.IO.File.Exists(Server.MapPath(
    '\Foto01\fotos\' + Request.QueryString['f'])) then
    begin
      Imagem := System.Drawing.Image.FromFile(
        Server.MapPath('\Foto01\fotos\' +
          Request.QueryString['f']));
      Response.ContentType := 'image/jpeg';
      Imagem.Save(Response.OutputStream,
        System.Drawing.Imaging.ImageFormat.Jpeg);
      Imagem.Dispose();
    end;
end;
```

Antes de testar crie uma pasta chamada “fotos” dentro da pasta da aplicação (em *C:\Inetpub\wwwroot\Foto01*). Copie algumas fotos para essa pasta e acrescente os namespaces *System.Drawing.Imaging* e *System.IO* na cláusula *uses*.

Compile a aplicação e altere o link informado no *browser*, acrescentando o campo de consulta “f” que informará o arquivo da foto que será respondido (Ex.: *http://localhost/Foto01/WebForm1.aspx?f=nomefoto.jpg*). Veja na **Figura 1** um exemplo (veja o nome do arquivo na URL).

## Explicando o código

O primeiro comando verifica se o arquivo de imagem passado pelo campo de consulta realmente existe. Se existir, a variável *Imagem* será carregada com o arquivo. Define-se o conteúdo de resposta (*image/jpeg*) e responde-se a imagem para a aplicação.

## Redimensionando uma imagem antes de responder

Geralmente precisamos limitar o tamanho da imagem antes de exibi-la, para que o carregamento das mesmas não fique muito longo. O interessante é que utilizando o namespace *System.Drawing.Imaging* pode-se fazer isso de forma rápida e com um bom resultado na diminuição do tamanho do arquivo, agilizando a apresentação da foto no *browser*.

Crie uma nova aplicação ASP.NET dando o nome de “Foto02”. Acesse o *Page\_Load* e digite o código da **Listagem 2**.

**Listagem 2.** Código para redimensionar a imagem

```
var
  imgBase, imgFinal: System.Drawing.Image;
  largura, altura: Double;
  dimensoes: Size;
  grafico: System.Drawing.Graphics;
  retangulo: Rectangle;
begin
  if System.IO.File.Exists(Server.MapPath(
    '\Foto02\fotos\' + Request.QueryString['f'])) then
  begin
    imgBase := System.Drawing.Image.FromFile(
      Server.MapPath('\Foto02\fotos\' +
        Request.QueryString['f']));
    largura := Convert.ToInt16(
      Request.QueryString['t']);
    if largura = 0 then
    begin
      largura := imgBase.Width;
      altura := imgBase.Height;
    end
    else
      altura := imgBase.Height /
        (imgBase.Width / largura);
    dimensoes := Size.Create(Convert.ToInt32(largura),
      Convert.ToInt32(altura));
    imgFinal := System.Drawing.Bitmap.Create(
      dimensoes.Width, dimensoes.Height,
      imgBase.PixelFormat);
    grafico := System.Drawing.Graphics.FromImage(
      imgFinal);
    retangulo := Rectangle.Create(0, 0,
      dimensoes.Width, dimensoes.Height);
    grafico.DrawImage(imgBase, retangulo);

    Response.ContentType := 'image/jpeg';
    imgFinal.Save(Response.OutputStream,
      System.Drawing.Imaging.ImageFormat.Jpeg);
    imgFinal.Dispose;
    imgBase.Dispose;
  end;
end;
```

Acrescente os namespaces *System.Drawing.Imaging* e *System.IO* na cláusula *uses*. Crie a pasta “fotos” e copie algumas fotos.

Compile a aplicação e modifique o link para: *http://localhost/Foto02/WebForm1.aspx?f=nomefoto.jpg&t=300*, para que a imagem apareça redimensionada no *browser* (**Figura 2**).

## Explicando o código

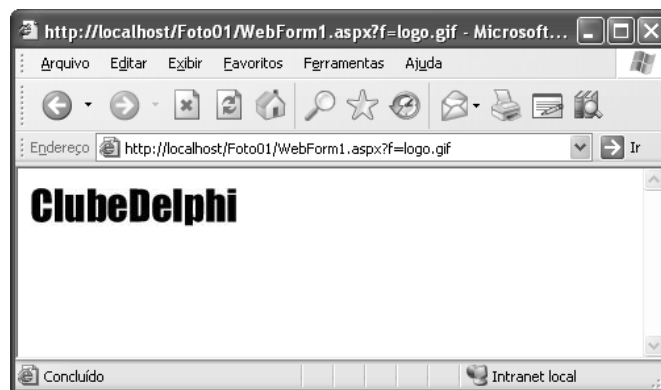
Passamos também, através de um campo de consulta (*t*), a largura da foto. Assim, foi necessário fazer um cálculo proporcional para que a imagem não perca a relação entre largura e altura. Utilizamos duas imagens: a *base* e a *final* e foi necessário utilizar um gráfico e um retângulo para fazer o redimensionamento da imagem.

Para facilitar também utilizamos uma variável do tipo *Size* que possui as dimensões *Width* e *Height*. Essa opção é interessante porque não é feita somente uma alteração no tamanho apresentado, mas sim, no tamanho do arquivo que será carregado.

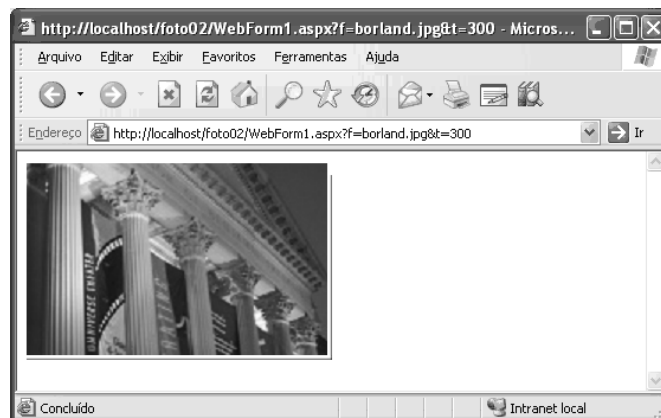
Realizou-se alguns testes onde uma imagem de 101 Kb respondida pela aplicação, ainda sem alteração de tamanho, foi respondida com um tamanho de 73,9 Kb sem perder qualidade. Em outro teste solicitou-se a resposta com um tamanho de 100 *pixels* e o arquivo exibido na resposta teve seu tamanho reduzido para 3,46 Kb, apresentando uma boa taxa de compressão.

## Colocando uma marca na imagem

Muitos Web Designers perdem um bom tempo modificando fotos para, simplesmente, colocar uma marca da empresa, endereço do site ou *copyright*. No Delphi, podemos sobrepor



**Figura 1.** Exemplo de como exibir imagens no *browser*



**Figura 2.** Redimensionando imagens no *browser*

uma imagem à outra, criando assim um efeito de marca d'água (dependendo do estilo da imagem que será marca).

Basicamente, esse efeito é criado através da sobreposição de imagens e/ou utilização de transparências, portanto, adicione um arquivo chamado *marca.gif* que será sobreposto sobre outra imagem. Crie uma nova aplicação dando o nome de “Foto03”. Acesse o *Page\_Load* e digite o código contido na **Listagem 3**.

**Listagem 3.** Código para colocar uma marca na imagem

```
var
imgBase, imgFinal, imgMarca: System.Drawing.Image;
largura, larguraMarca, altura, alturaMarca: Double;
dimensoes: Size;
grafico: System.Drawing.Graphics;
retangulo, retanguloMarca: Rectangle;
inicioX, inicioY: Integer;
begin
if System.IO.File.Exists(Server.MapPath(
'\Foto03\fotos\')+Request.QueryString['f']) then
begin
imgBase := System.Drawing.Image.FromFile(
Server.MapPath('\Foto03\fotos\')+
Request.QueryString['f']);
largura := imgBase.Width;
altura := imgBase.Height;
dimensoes := Size.Create(Convert.ToInt32(largura),
Convert.ToInt32(altura));
imgFinal := System.Drawing.Bitmap.Create(
dimensoes.Width, dimensoes.Height,
imgBase.PixelFormat);
grafico := System.Drawing.Graphics.FromImage(
imgFinal);
retangulo := Rectangle.Create(0, 0,
dimensoes.Width, dimensoes.Height);
grafico.DrawImage(imgBase, retangulo);
if System.IO.File.Exists(Server.MapPath(
'\Foto03\fotos\')+ 'marca.gif') then
begin
imgMarca := System.Drawing.Image.FromFile(
Server.MapPath('\Foto03\fotos\')+ 'marca.gif');
larguraMarca := imgMarca.Width;
alturaMarca := imgMarca.Height;

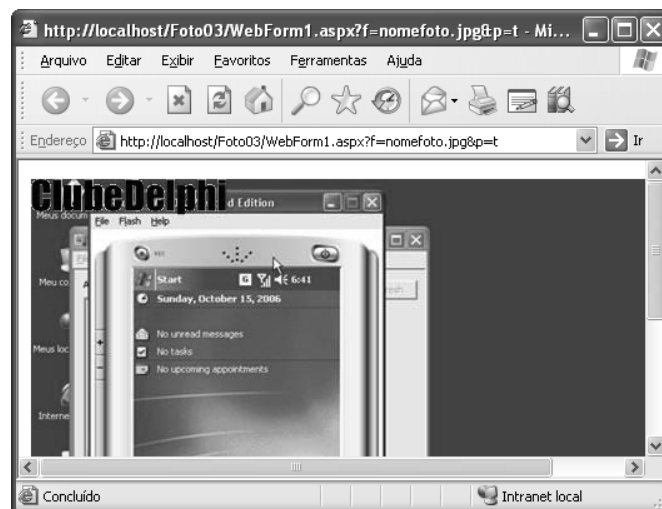
if Request.QueryString['p'] = 't' then
begin
inicioX := 0;
inicioY := 0;
end;
if Request.QueryString['p'] = 'b' then
begin
inicioX := Convert.ToInt32(dimensoes.Width) -
Convert.ToInt32(larguraMarca);
inicioY := Convert.ToInt32(dimensoes.Height) -
Convert.ToInt32(alturaMarca);
end;
dimensoes.Width := Convert.ToInt32(larguraMarca);
dimensoes.Height := Convert.ToInt32(alturaMarca);
retanguloMarca := Rectangle.Create(inicioX,
inicioY, dimensoes.Width, dimensoes.Height);
grafico.DrawImage(imgMarca, retanguloMarca);
end;
Response.ContentType := 'image/jpeg';
imgFinal.Save(Response.OutputStream,
System.Drawing.Imaging.ImageFormat.Jpeg);
imgFinal.Dispose;
imgBase.Dispose;
end;
end;
```

Não esqueça de declarar no *uses* os namespaces *System.Drawing.Imaging* e *System.IO*. Também crie a pasta “fotos” e copie algumas fotos antes de compilar e testar.

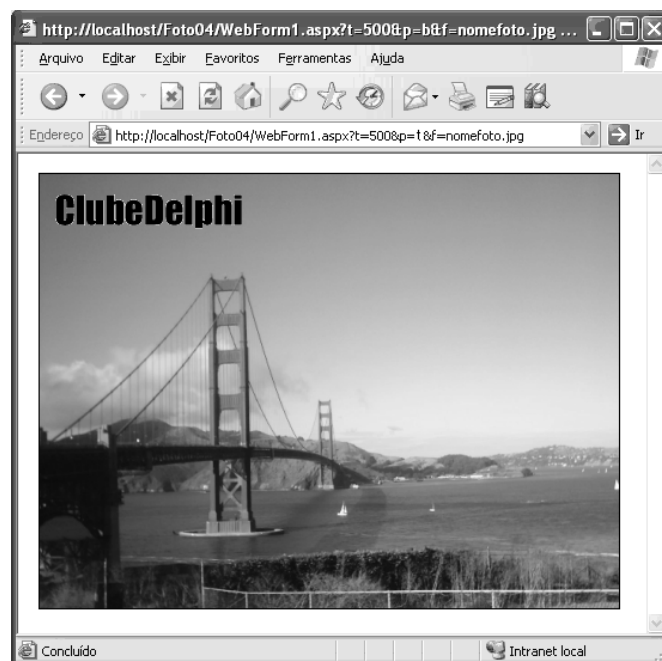
## Explicando o código

Para sobrepor uma imagem à outra foi necessária a utilização de dois retângulos. Criou-se um campo de consulta (*QueryString* chamada *p*) para definir a posição da marca, podendo ser no topo (*t*) ou em baixo (*b*). Se for no topo, a marca será exibida no canto superior esquerdo. Se for embaixo será exibida no canto inferior direito. Para colocar a figura embaixo foi necessário fazer um cálculo para definir a posição inicial de altura e largura (X e Y).

Compile a aplicação e modifique o link para: <http://localhost/>



**Figura 3.** Imagem com uma marca



**Figura 4.** Browser exibindo uma imagem redimensionada e com uma marca

**Listagem 4.** Imagem com marca e tamanho definidos

```

const
    Proporcão = 2;
var
    imgBase, imgFinal, imgMarca: System.Drawing.Image;
    largura, larguraMarca, altura, alturaMarca: Double;
    dimensoes: Size;
    grafico: System.Drawing.Graphics;
    retangulo, retanguloMarca: Rectangle;
    inicioX, inicioY, tamanho: Integer;
begin
    if System.IO.File.Exists(Server.MapPath(
        '\Foto04\fotos\' + Request.QueryString['f']) then
        imgBase := System.Drawing.Image.FromFile(
            Server.MapPath('\Foto04\fotos\' + Request.QueryString['f']))
    else
        imgBase := System.Drawing.Image.FromFile(
            Server.MapPath('\Foto04\fotos\indisponivel.jpg'));
    tamanho := Convert.ToInt32(Request.QueryString['t']);
    if tamanho = 0 then
        begin
            largura := imgBase.Width;
            altura := imgBase.Height;
        end
    else
        begin
            largura := tamanho;
            altura := imgBase.Height / (imgBase.Width / tamanho);
        end
    end;
    dimensoes := Size.Create(Convert.ToInt32(largura),
        Convert.ToInt32(altura));
    imgFinal := System.Drawing.Bitmap.Create(
        dimensoes.Width, dimensoes.Height, imgBase.PixelFormat);
    grafico := System.Drawing.Graphics.FromImage(imgFinal);
    retangulo := Rectangle.Create(0, 0, dimensoes.Width,
        dimensoes.Height);
    grafico.DrawImage(imgBase, retangulo);
    if (largura > 100) and (System.IO.File.Exists(
        Server.MapPath('\Foto04\fotos\' + 'marca.gif'))) then
        begin
            imgMarca := System.Drawing.Image.FromFile(
                Server.MapPath('\Foto04\fotos\' + 'marca.gif'));
            larguraMarca := dimensoes.Width / proporcão;
            alturaMarca := (imgMarca.Height / imgMarca.Width) *
                (dimensoes.Width / proporcão);
            if Request.QueryString['p'] = 't' then
                begin
                    inicioX := 0;
                    inicioY := 0;
                end
            else
                if Request.QueryString['p'] = 'b' then
                    begin
                        inicioX := Convert.ToInt32(dimensoes.Width) -
                            Convert.ToInt32(larguraMarca) - 4;
                        inicioY := Convert.ToInt32(dimensoes.Height) -
                            Convert.ToInt32(alturaMarca) - 4;
                    end
                end;
            dimensoes.Width := Convert.ToInt32(larguraMarca);
            dimensoes.Height := Convert.ToInt32(alturaMarca);
            retanguloMarca := Rectangle.Create(inicioX,
                inicioY, dimensoes.Width, dimensoes.Height);
            grafico.DrawImage(imgMarca, retanguloMarca);
        end
    end;
    Response.ContentType := 'image/jpeg';
    imgFinal.Save(Response.OutputStream,
        System.Drawing.Imaging.ImageFormat.Jpeg);
    imgFinal.Dispose;
    imgBase.Dispose;
end;

```

Foto03/WebForm1.aspx?f=nomefoto.jpg&p=t, para que a imagem apareça com uma marca no browser na parte superior (**Figura 3**).

### Incluindo uma marca com tamanho proporcional ao redimensionamento

Agora juntaremos todos os exemplos vistos até então, possibilitando ao usuário a definição do tamanho da imagem que será exibida, adicionando uma marca (texto) que terá um tamanho proporcional ao tamanho final da imagem e ainda escolhendo a posição dessa marca.

Se a imagem não for encontrada, será respondida uma imagem (o arquivo está para download). Para o arquivo de marca,

criou-se uma imagem contendo o texto “ClubeDelphi”, que foi salva no formato GIF, com fundo transparente (chamada *marca.gif*).

Vamos criar uma nova aplicação dando o nome de “Foto04”. Acesse o *Page\_Load* e digite o código contido na **Listagem 4**.

Novamente, não esqueça de declarar no *uses* os namespaces *System.Drawing.Imaging* e *System.IO* e também criar a pasta “fotos” e copiar algumas fotos antes de compilar e testar.

### Explicando o código

Definiu-se uma constante para estabelecer a proporção do tamanho da marca levando em conta o tamanho da imagem definido pelo campo de consulta (*t*) do link. Com isso foi necessário fazer o cálculo do tamanho da marca levando em conta a proporção definida na constante. A marca criada para o exemplo na **Figura 4** teve seu tamanho definido em 1157 x 44 pixels, salva no formato GIF, com fundo transparente.





### Conclusão

Apresentamos mais uma facilidade do ASP.NET, onde poderemos criar, muito rapidamente, uma aplicação que trabalhe com imagens para a internet. Exemplos como álbuns de fotos, sistemas de vendas de veículos, imobiliárias etc., são ótimos candidatos a usufruir das técnicas aqui demonstradas. ■

# PENSE...

QUANTO TEMPO  
VOCÊ GASTARIA  
PARA DESENVOLVER  
COBRANÇA COM BOLETOS  
BANCÁRIOS PARA  
APENAS UM BANCO  
NO SEU SOFTWARE

## COBREBEMX

-  56 BANCOS E MAIS DE 430 CARTEIRAS DE COBRANÇA PARA IMPRESSÃO E/OU ENVIO DE BOLETO BANCÁRIO POR EMAIL;
-  GERAÇÃO DE BOLETOS ON LINE;
-  GERAÇÃO E LEITURA DE ARQUIVOS (REMESSA/RETORNO) NOS PADRÕES FEBRABAN E CNAB;
-  MAIS DE 40 EXEMPLOS EM DIVERSAS LINGUAGENS DE PROGRAMAÇÃO



DOWNLOADS E INFORMAÇÕES EM [WWW.COBBREBEM.COM](http://WWW.COBBREBEM.COM)