Manual de jQuery

Autores do manual

Este manual foi criado pelos seguintes colaboradores de Criarweb.com:

Miguel Angel Alvarez - Tradução de JML
(6 capítulos)

Miguel Angel Alvarez - Tradução de Celeste Veiga
(3 capítulos)

Introdução a jQuery

Bem-vindos ao Manual sobre jQuery que vamos publicar em CriarWeb.com, com o qual pretendemos esclarecer aos usuários o método de trabalho e programação de aplicações do lado do cliente, compatíveis com todos os navegadores mais comuns.

O que é jQuery

Para simplificar, poderíamos dizer que jQuery é um framework Javascript, porém talvez muitos dos leitores se perguntarão o que é um framework. Pois é um produto que serve como base para a programação avançada de aplicações, que oferece uma série de funções ou códigos para realizar tarefas habituais. Em outras palavras, framework são umas bibliotecas de código que contém processos ou rotinas já prontas para usar. Os programadores utilizam os frameworks para que eles mesmos não tenham que desenvolver as tarefas mais básicas, visto que no próprio framework já há implementações que estão provadas, funcionam e não se necessitam voltar a programar.

> **Nota:**Caso não saiba o que é Javascript certamente não lhe interessará este artigo, porém pode aprendê-lo também em CriarWeb.com: O que é Javascript

Por exemplo, no caso que nos ocupa, jQuery é um framework para a linguagem Javascript, logo será um produto que nos simplificará a vida para programar nesta linguagem. Como provavelmente saberemos, quando um desenvolvedor tem que utilizar Javascript, geralmente tem que se preocupar por fazer scripts compatíveis com vários navegadores e para isso tem que incorporar muito código que o único que faz é detectar o browser do usuário, para fazer uma coisa ou outra dependendo de se é Internet Explorer, Firefox, Opera, etc. jQuery é onde mais nos pode ajudar, visto que implementa uma série de classes (de programação orientada a objetos) que nos permitem programar sem nos preocuparmos do navegador com o qual nos está visitando o usuário, já que funcionam de exata forma em todas as plataformas mais habituais.

Sendo assim, este framework Javascript, nos oferece uma infra-estrutura com a qual teremos muito maior facilidade para a criação de aplicações complexas do lado do cliente. Por exemplo, com jQuery obteremos ajuda na criação de interfaces de usuário, efeitos dinâmicos, aplicações que fazem uso de Ajax, etc. Quando programemos Javascript com jQuery teremos a nossa disposição uma interface para programação que nos permitirá fazer coisas com o navegador que estamos certos de que funcionarão para todos nossos visitantes. Simplesmente devemos conhecer as bibliotecas do framework e programar utilizando as classes, suas propriedades e métodos para a execução de nossos objetivos.

Ademais, todas estas vantagens que sem dúvida são muito de agradecer, com jQuery as obtemos de maneira gratuita, já que o framework tem licença para uso em qualquer tipo de plataforma, pessoal ou comercial. Para isso, simplesmente teremos que incluir em nossas páginas um script Javascript que contém o código de jQuery, que podemos baixar da própria página web do produto e começar a utilizar o framework.

O arquivo do framework ocupa uns 56 KB, o que é bastante razoável e não atrasará o download de nossa página (se nosso servidor envia os dados comprimidos, o que é bastante normal, o peso de jQuery será de uns 19 KB). Além disso, nosso servidor o enviará ao cliente a primeira vez que visite uma página do site. Nas seguintes páginas o cliente já terá o arquivo do framework, por isso não necessitará transferi-lo e o tomará do cache. Côo o qual o carregamento da página só se verá afetada pelo peso deste framework uma vez por usuário.

As vantagens na hora de desenvolvimento das aplicações, assim como as portas que nos abre jQuery compensam extraordinariamente o peso do pacote.

Vantagens de jQuery com respeito a outras alternativas

É importante comentar que jQuery não é o único framework que existe no mercado. Existem várias soluções similares que também funcionam muito bem, que basicamente nos servem para fazer o mesmo. Como é normal, cada um dos frameworks tem suas vantagens e desvantagens, porém jQuery é um produto com uma aceitação por parte dos programadores muito boa e um grau de penetração no mercado muito amplo, o que faz supor que é uma das melhores opções. Ademais, é um produto sério, estável, bem documentado e com uma grande equipe de desenvolvedores encarregados da melhora e atualização do framework. Outra coisa muito interessante é a dilatada comunidade de criadores de plugins ou componentes, o que torna fácil encontrar soluções já criadas em jQuery para implementar assuntos como interfaces de usuário, galerias, votações, efeitos diversos, etc.

Um dos competidores de jQuery, do qual já publicamos em CriarWeb.com um amplo manual para programadores, é Mootools, que também possui vantagens similares. Este é o link ao Manual de Mootools, que também pode ser interessante, porque certamente temos explicado com mais detalhes que jQuery.

jQuery, é para mim?

Se estiver interessado em enriquecer sua página web com componentes da chamada Web 2.0, como efeitos dinâmicos, Ajax, interação, interfaces de usuário avançadas, etc., jQuery é uma ferramenta imprescindível para desenvolver todas estas coisas sem ter que se complicar com os níveis mais baixos do desenvolvimento, já que muitas funcionalidades já estão implementadas, ou ainda as bibliotecas do framework lhe permitirão realizar a programação muito mais rápida e livre de erros.

Agora também, todas estas melhorias da web 2.0, que à princípio pode ser muito atrativas, também tem têm um custo em tempo de desenvolvimento dos projetos. Sem um framework como jQuery, o tempo de criação e depuração de todos esses componentes dinâmicos seria muito maior, porém ainda assim ninguém diz que seja somente instalar o sistema e iniciar. Entretanto, o mais complicado de jQuery é aprender a usá-lo, igual ao que ocorre com qualquer outro framework Javascript. Requererá do desenvolvedor habilidades avançadas de programação, assim como o conhecimento, ao menos básico, da programação orientada a objetos. Uma vez aprendido as vantagens de utilizá-lo compensarão mais que de sobra o esforço. Esperamos que com este Manual de jQuery, que estamos publicando em CriarWeb.com possa aprender o necessário para desenvolver seus próprios componentes dinâmicos em Javascript com os que enriquecer suas aplicações.

Podemos conhecer jQuery acessando à página de início do framework Javascript: http://jquery.com/

*Artigo por **Miguel Angel Alvarez - Tradução de JML***

Demo muito simples de uso de jQuery

Com o objetivo de que os leitores possam ter uma rápida idéia das possibilidades de jQuery, escrevendo umas brevíssimas linhas de código Javascript, vamos publicar um par de exemplos bem simples que nos ilustrem, porém sem complicar muito. Servirão para a introdução a jQuery que estamos publicando no Manual de jQuery.

A idéia deste artigo não é explicar as funcionalidades que vamos demonstrar, e sim ver o pouco código que tivemos que escrever para realizar uns scripts com dinamismos simples. Talvez os scripts em si não digam muito a um leitor pouco experiente, mas aos que já tiveram contato com os pormenores que há que seguir para fazer estes efeitos, de maneira que sejam compatíveis com todos os navegadores, saberão que jQuery nos simplificou muito nossa tarefa.

Sendo assim, não se preocupe demais com os detalhes destes códigos, que explicaremos em CriarWeb.com mais adiante com mais detalhes.

Demo 1 de jQuery

Para começar veremos este exemplo, onde temos dois botões e um texto. Ao clicar um botão, mudaremos o texto e ao clicar o outro colocaremos outro texto distinto.

Podemos ver o exemplo em funcionamento em uma página à parte.

Neste exemplo, temos uma camada que tem este código:

```
<div id="camada" style="padding: 10px; background-color: #ff8800">Clique em um botão</div>
```

Logo, temos dois botões com estes códigos:

```
<input type="button" value="Botão A" onclick="$('#camada').html('Clicou no botão <b>A</b>')">
<input type="button" value="Botão B" onclick="$('#camada').html('Recebido um clique no botão <b>B</b>')">
```

Como se pode ver, nos botões há definido um par de eventos onclick (um em cada um) que executam uma instrução Javascript quando se clica sobre eles. A instrução está em Javascript, porém faz uso de algumas ferramentas disponíveis em jQuery para trabalho com os elementos da página. Neste caso, explicando rapidamente, se faz uma seleção do elemento DIV da camada e logo se executa um método sobre ele para mudar o conteúdo HTML do elemento.

Demo 2 de jQuery

Neste outro exemplo, veremos algo um pouquinho mais complexo. Bom, realmente não tem maior complexidade, porém estamos utilizando jQuery de uma maneira um pouco distinta, que requer algo mais de código por nossa parte. Agora vamos ter duas camadas em nossa página. Uma camada estará sempre visível e a outra camada aparecerá inicialmente oculta e vamos mostrá-la ou ocultá-la dependendo de se o usuário coloca o mouse em cima da camada que está sempre visível.

Para se ter uma idéia exata de nosso exemplo se pode ver em uma janela à parte.

Neste segundo exemplo, temos este código HTML, com as duas camadas.

```
<div id="camada" style="padding: 10px; background-color: #ff8800;">Coloque o mouse em cima desta
camada</div>
<br>
<div id="mensagem" style="display: none;">Colocou o mouse em cima!! <br>(Agora tire-o da camada)</div>
```

Agora vamos ter um código Javascript com jQuery que defina os eventos do usuário, para quando situa o mouse dentro ou fora da primeira camada.

```
$("#camada").mouseenter(function(evento){
  $("#mensagem").css("display", "block");
});
$("#capa").mouseleave(function(evento){
  $("#mensagem").css("display", "none");
});
```

Desta simples maneira, criamos dois eventos no DIV com id="camada". Ademais, definimos o código dos eventos por meio de funções, que se encarregarão de mostrar ou ocultar a segunda camada com id="mensagem".

```
$("#mensagem").css("display", "block");
```

Isto nos seleciona a camada com id "mensagem" e com o método css() indicamos que queremos mudar o atributo "display" ao valor "block" desse elemento.

```
$("#mensagem").css("display", "none");
```

Esta outra linha muito similar, simplesmente muda o "display" a "none" para ocultar o elemento.

Esperamos que estes dois exemplos sirva para ver rapidamente algumas coisas que se podem fazer com jQuery com muito pouco esforço e que funcionam em todos os navegadores.

*Artigo por **Miguel Angel Alvarez - Tradução de JML***

Passos para utilizar jQuery em sua página web

Neste artigo vamos explicar como começar a utilizar jQuery em suas páginas web, passo a passo, para que possa fazer seu primeiro script jQuery e assim compreender os fundamentos de uso deste framework Javascript. Neste ponto seria bom que soubesse o que é jQuery, o que foi explicado já no Manual de jQuery que viemos publicando em CriarWeb.com.

A idéia é que você mesmo possa ir fazendo os passos que vamos relatar, para que comprove o simples que é começar a utilizar jQuery. Este artigo segue o esquema com o que os próprios criadores de jQuery ensinam a utilizar seu produto, portanto está diretamente inspirado na documentação de jQuery.

1.- Download da última versão do framework

Acesse à página de jQuery para baixar a última versão do framework. No momento de escreber este artigo a release atual é a 1.3.2, e com a qual realizamos estes exemplos. Entretanto, pode ser que tenha sido publicada uma nova versão que melhore a atual.

Dão duas possibilidades para baixar, uma que se chama PRODUCTION, que é a adequada para páginas web em produção, visto que está minimizada e ocupa menos espaço, com o qual o carregamento em nosso site será más rápido. A outra possibilidade é baixar a versão que se chama DEVELPOMENT, que está com o código sem comprimir, com o qual ocupa mais espaço, mas se poderá ler a implementação das funções do framework, que pode ser interessante em etapa de desenvolvimento, porque poderemos mergulhar no código de jQuery caso tenhamos que entender algum assunto do trabalho com o framework.

Você verá que o download é um arquivo js que contem o código completo do framework. Coloque o arquivo em uma pasta em seu computador para fazer as provas.

2.- Crie uma página HTML simples

Agora, no mesmo diretório onde você colocou o arquivo js, coloque um arquivo html com o seguinte código.

```
<html>
  <head>
    <script src="jquery-1.3.2.min.js" type="text/javascript"></script>
    <script>

    </script>
  </head>
<body>
<a href="http://www.criarweb.com/">CriarWeb.com</a>
</body>
</html>
```

Como poderá ver, é uma página bem simples, na qual temos uma chamada a um script externo chamado jquery-1.3.2.min.js. Este é o código de jQuery que baixamos da página do framework. Se tiver baixado uma versão distinta, talvez o arquivo se chame de outra maneira, assim que é possível que tenha que editar esse nome de arquivo para colocar o que tiver no diretório.

Com esse script já incluímos todas as funções de jQuery dentro de nossa página. Só tem que prestar atenção a que tanto o arquivo .html desta página, como o arquivo .js do framework estejam no mesmo diretório. E, como dizia, que o arquivo que incluímos com a etiqueta SCRIPT seja o .js que nós baixamos.

Ademais, como se pode ver, deixamos dentro do HEAD uma etiqueta SCRIPT de abertura e fechamento que está vazia. Todo o código que vamos explicar a seguir você terá que colocá-lo dentro dessa etiqueta.

3.- Executar código quando a página tiver sido carregada

O passo que vamos explicar agora é importante que se entenda, embora sem lugar a dúvidas ao longo do manual publicado em CriarWeb.com, o veremos detalhadamente. Trata-se de detectar o momento em que a página está pronta para receber comandos Javascript que fazem uso do DOM.

Quando fazemos certas ações complexas com Javascript temos que estar seguros que a página tenha terminado de carregar e esteja pronta para receber comandos Javascript que utilizem a estrutura do documento com o objetivo de mudar coisas, como criar elementos, tirá-los, aletrar suas propriedades, etc. Se não esperamos que a página esteja pronta para receber instruções corremos um alto risco de obter erros de Javascript na execução.

Geralmente, quando se deseja executar Javascript depois do download da página, se não utilizamos nenhum framework, o mais normal será utilizar um código como este:

```
window.onload = function () {
  alert("carregado...");
}
```

Porém esta instrução, que carrega uma funcionalidade no evento onload do objeto window, só se executará quando o navegador tiver baixado completamente TODOS os elementos da

página, o que inclui imagens, iframes, banners, etc. o que pode demorar bastante, dependendo dos elementos que tiver essa página e seu peso.

Porém, na verdade não faz falta esperar todo esse tempo de carregamento dos elementos da página para poder executar instruções Javascript que alterem o DOM da página. Só teria que fazê-lo quando o navegador tiver recebido o código HTML completo e tiver processado ao renderizar a página. Para isso, jQuery inclui uma maneira de fazer ações justo quando já está pronta a página, embora haja elementos que não tenham sido carregados completamente. Isto se faz com a seguinte instrução:

```
$(document).ready(function(){
  //código a executar quando o DOM estiver pronto para receber instruções.
});
```

Por dar uma explicação a este código, digamos que com $(document) se obtém uma referência ao documento (a página web) que se está carregando. Logo, com o método ready() se define um evento, que se desata ao ficar pronto o documento para realizar ações sobre o DOM da página.

4.- Inserir um manejador de evento à etiqueta A (link) que há na página

Agora que já sabemos como e quando devemos executar as ações de jQuery que alterem a funcionalidade, conteúdos ou aspecto da página, podemos inserir um pouco de código para demonstrar o método de trabalho com jQuery.

Para este primeiro exemplo vamos criar um evento click no link, para mostrar uma mensagem quando se clique sobre o link. Para criar um evento click sobre um elemento temos que invocar ao método click sobre esse elemento e passar-lhe como parâmetro uma função com o código que queremos que se execute quando se clica.

```
$("a").click(function(evento){
  //aqui o código que se deve executar ao clicar
});
```

Como vemos no código anterior, com $("a") obtemos uma referência ao link. Bom, na verdade com isso estamos selecionando todas as etiquetas A (links) do documento, porém como só há um link, em realidade nos serve. Logo, o método click() sobre $("a") estamos definindo um evento, que se executará quando se clique sobre o link. Como se pode ver, ao método click se passa uma função onde se deve colocar o código Javascript que queremos que se execute quando se clique sobre o link.

Agora vejamos a definição do evento clique completa, colocada dentro do evento ready do document, para que se atribua quando a página estiver pronta.

```
$(document).ready(function(){
  $("a").click(function(evento){
    alert("Clicou o link...Agora será enviado a CriarWeb.com");
  });
});
```

Por linhas, isto é uma recapitulação do que fizemos:

```
$(document).ready(function(){
```

Esta linha serve para fazer coisas quando a página estiver pronta para receber instruções jQuery que modifiquem o DOM.

```
$("a").click(function(evento){
```

Com esta linha estamos selecionando todas as etiquetas A do documento e definindo um evento click sobre esses elementos.

```
alert("Clicou o link...Agora será enviado a CriarWeb.com");
```

Com esta linha simplesmente mostramos uma mensagem de alerta informando ao usuário que se clicou sobre o link.

5.- Salve o arquivo html e abra-o em um navegador

Uma vez que tivermos feito nossa primeira página com jQuery, pode salvá-la e executá-la em um navegador. Prove clicar no link e deveria sair a janela de alerta.

Pode-se ver este script em funcionamento em uma página à parte.

Já está feito e funcionando seu primeiro script com jQuery!

Caso tenham ficado dúvidas, deixamos aqui o código completo que deveria ter:

```
<html>
<head>
  <title>Primeiro script com jQuery</title>
<script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<script>
$(document).ready(function(){
  $("a").click(function(evento){
    alert("Clicou o link...Agora será enviado a CriarWeb.com");
  });
});
</script>
</head>

<body>

<a href="http://www.criarweb.com">Ir a CriarWeb.com</a>

</body>
</html>
```

6.- Extra: Bloquear o comportamento normal de um link

Agora vejamos uma pequena modificação para alterar o comportamento padrão dos links. Como sabemos, quando se clica um link nos leva a uma URL. Logo, ao clicar, primeiro se executa o que tivermos colocado no evento click do link e logo o link leva ao navegador a uma nova URL.

Este comportamento se pode bloquear também desde jQuery, adicionando uma chamada ao método preventDefault() sobre o evento. Se observar, a função definida para marcar o comportamento do evento click sobre o link recebia um parâmetro. Esse parâmetro é um manejador de evento. E tem seus próprios métodos e propriedades, como este preventDefault() que comentávamos. Seu uso é o seguinte:

```
$(document).ready(function(){
  $("a").click(function(evento){
    alert("Clicou o link, mas vamos cancelar o evento...Portanto, não levaremos a CriarWeb.com");
    evento.preventDefault();
  });
});
```

Como pudemos ver invocando a evento.preventDefault() o que conseguimos neste caso é que

o link não leve a nenhum lugar, simplesmente se executará o código Javascript contido para o evento click.

*Artigo por **Miguel Angel Alvarez - Tradução de JML***

Básicos jQuery: adicionar e tirar classes CSS sobre elementos

Para nos acostumarmos ao trabalho com o framework Javascript jQuery vamos continuar fazendo exemplos simples de funcionamento, que vamos explicar na medida das possibilidades com o que conhecemos até agora no Manual de jQuery.

Claro que estes exercícios são um pouco especiais, dado que servem para ilustrar o modo de trabalho com jQuery, porém sem explicar todos os detalhes relacionados com o uso do framework. Por enquanto que queremos é mostrar uma introdução ao sistema e mostrar por alto algumas de suas possibilidades. No futuro publicaremos artigos que irão pouco a pouco explicando todos os detalhes de trabalho com jQuery.

No caso que nos ocupa, queremos demonstrar o uso de jQuery para alterar elementos de uma página web, adicionando e tirando classes CSS. Isto é bem simples, porque em jQuery os elementos têm duas classes chamadas addClass() e removeClass(), que servem justamente para que o elemento que recebe o método se aplique uma classe CSS ou se elimine. Por isso o que vamos aprender, com respeito ao artigo anterior -Passos para utilizar jQuery-, é utilizar esses novos métodos dos elementos.

Para complicá-lo só um pouco mais, vamos adicionar e tirar classes do elemento com resposta a ações do usuário, para aprender também novos eventos de usuário.

Em nosso exemplo vamos ter dois elementos. Primeiro, uma camada DIV com um texto. Depois teremos um link que estará fora da camada capa DIV. Ao situar o usuário o mouse sobre um link adicionaremos uma classe CSS à camada DIV e ao retirar o mouse do link eliminaremos a classe CSS que havíamos adicionado antes. Se desejar, para esclarecer o exemplo, podemos ver o exercício em funcionamento em uma página à parte.

> **Nota:** Temos em conta que foi lido o artigo anterior, no qual explicamos passo por passo como fazer sua primeira página com jQuery, pois necessitaremos conhecer algumas coisas que já se comentaram ali.

1.- Criar a página com uma camada, um link e a definição de uma classe CSS

O primeiro passo seria construir uma página com todos os elementos que queremos que façam parte deste primeiro exemplo: a camada DIV, o link e a definição da classe CSS.

Ademais, agora também vamos incluir o script de jQuery, que necessitaremos para acessar às funções do framework Javascript.

```
<html>
<head>
   <title>Adicionar e tirar classes CSS a elementos</title>
   <script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
.clasecss{
   background-color: #ff8800;
```

```
  font-weight: bold;
}
</style>
</head>

<body>
<div id="camada">
Esta camada é independente e vou adicionar e eliminar classes css sobre ela
</div>
<br>
<br>
<a href="http://www.criarweb.com">Adicionar e tirar classe na camada de cima</a>
</body>
</html>
```

Perfeito, agora já temos a infra-estrutura necessária para nosso exemplo, com todos os integrantes do mesmo. Só nos faltaria fazer o seguinte passo, que é adicionar os comportamentos dinâmicos com jQuery.

2.- Recordar: adicionar sempre os scripts jQuery quando o documento está "ready"

Agora vamos começar a colocar o código Javascript, porém iniciaremos lembrando aos leitores que qualquer funcionalidade que quisermos agregar à página por meio de jQuery, deve ser incluída quando o documento estiver pronto para receber ações que modifiquem o DOM da página.

Para isto temos que incluir sempre o código:

```
$(document).ready(function(){
  //aqui colocaremos as instruções que modifiquem o DOM
});
```

3.- Acrescentar os eventos mouseover e mouseout para adicionar e tirar uma classe CSS

Neste passo, vamos criar um par de eventos que associaremos aos links. Este par de eventos serão lançados quando o usuário colocar o ponteiro do mouse sobre o link (mouseover) e quando o usuário retirar o mouse do link (mouseout).

Por exemplo, para definir um evento mouseover se tem que chamar ao método mouseover() sobre o elemento que queremos associar o evento. Ademais, ao método mouseover() se envia por parâmetro uma função com o código que se quer executar como resposta a esse evento.

No caso de adicionar uma classe temos que utilizar o método addClass(), que se tem que invocar sobre o elemento ao que queremos adicionar a classe. A addClass() temos que passar uma cadeia com o nome da classe CSS que queremos adicionar.

Para selecionar o elemento que queremos adicionar a classe fazemos $("#idElemento"), ou seja, utilizamos a função dólar passando o identificador do elemento que queremos selecionar, precedida do caractere "#". Por exemplo, com $("#capa") estamos selecionando um elemento da página cujo id="camada".

```
$("a").mouseover(function(event){
  $("#camada").addClass("classecss");
});
```

De maneira análoga, porém com o método mouseout(), definimos o evento para quando o usuário tira o mouse do link.

```
$("a").mouseout(function(event){
  $("#capa").removeClass("classecss");
});
```

4.- Código completo do exemplo jQuery

Agora vejamos o código completo deste exercício:

```
<html>
<head>
  <title>Adionar e tirar classes CSS a elementos</title>
  <script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
.clasecss{
  background-color: #ff8800;
  font-weight: bold;
}
</style>
<script>
$(document).ready(function(){
  $("a").mouseover(function(event){
    $("#camada").addClass("classecss");
  });
  $("a").mouseout(function(event){
    $("#camada").removeClass("classecss");
  });
});
</script>
</head>

<body>

<div id="camada">
Esta camada é independente e vou adicionar e eliminar classes css sobre ela
</div>

<br>
<br>

<a href="http://www.criarweb.com">Adicionar e tirar classe na camada de cima</a>
</body>
</html>
```

Podemos ver o exemplo desenvolvido no artigo em uma página à parte.

*Artigo por **Miguel Angel Alvarez - Tradução de JML***

Mostrar e ocultar elementos da página com jQuery

Para aprender rapidamente como fazer coisas típicas com jQuery, vamos explicar como mostrar ou ocultar elementos da página, mudando propriedades das folhas de estilo em cascata. Para que este artigo tenha um pouco mais de praticidade, vamos realizar um exemplo de formulário onde alguns dos elementos estão escondidos. Nesse formulário, ao marcar uma opção de um campo checkbox, se mostrarão esses elementos escondidos e ao desmarcar essa opção, se ocultarão.

O método que se dispõem em jQuery para alterar as folhas de estilo se chama css() e podemos invocá-lo sobre elementos da página, aos que queremos mudar suas propriedades CSS. À princípio, para mostrar e ocultar elementos, é bom alterar o atributo "display", colocando o valor "none" para que não apareça e "block" para que sim o faça.

O que deveríamos saber para poder entender este exemplo se pode aprender no Manual de jQuery que viemos publicando em CriarWeb.com. Embora, como este é um exemplo básico, será suficiente com estudar o artigo Passos fundamentais para usar jQuery.

Ocultar e mostrar uma camada com jQuery

Vamos ver brevemente como usar o mencionado método css(). Primeiro, teríamos que ter um elemento na página, que é o que vamos mostrar ou ocultar.

```
<div id="meuelemento">
Conteudo do elemento...
</div>
```

Para ocultar este elemento, teríamos que invocar o método css() da seguinte maneira:

```
$("#meuelemento").css("display", "none");
```

Como se pode ver, se acessa ao elemento com a função dólar $() e o seletor "#meuelemento". Logo, ao método css() lhe passamos o valor "display" e "none", porque queremos alterar a propriedade display e atribuir-lhe o valor "none", para ocultar o elemento.

Para mostrá-lo faríamos algo parecido, porém, colocando o valor "block" no atributo CSS "display".

```
$("#meuelemento").css("display", "block");
```

Nota: o método css() admite outros parâmetros. Se só recebe um parâmetro, de tipo string, devolve o valor CSS atribuído a esse parâmetro. Também poderia receber um só parâmetro, neste caso de com uma notação de objeto com pares chave/valor, e então atribuiria todos esses estilos CSS, especificados pelos pares chave/valor no objeto, ao elemento da página onde tiver invocado o método.

Mostrar ou ocultar elementos como resposta a um evento

Agora que já sabemos como realizar uma mudança no atributo display, vamos ver como colocar esta instrução em andamento quando o usuário realizar ações na página. Lembrar que no início do artigo comentava que íamos criar um formulário com um campo de seleção (campo checkbox) e que ao ativar esse campo se mostrariam outros conteúdos no formulário. Ao desativá-lo, se ocultariam esses conteúdos do formulário.

Para entender bem o que desejamos fazer, podemos ver o exercício em funcionamento em uma página à parte.

O primeiro que podemos apresentar é o formulário com o qual vamos trabalhar.

```
<form>
Nome: <input type="text" name="nome">
<br>
<input type="checkbox" name="maior_idade" value="1" id="demaior_idade"> Sou maior de idade
<br>
<div id="formulariomaiores" style="display: none;">
Dado para maiores de idade: <input type="text" name="maiores_idade">
</div>
</form>
```

Como se poderá ver, é um formulário como outro qualquer. Só que tem uma camada com id="formulariomaiores", que contem os elementos do formulário que queremos mostrar ou ocultar ao marcar ou desmarcar o checkbox. Essa camada estará inicialmente oculta, e para isso colocamos o atributo style="display: none;". Podemos observar também no checkbox com id="demaior_idade", que é o que vai servir para marcar se se deseja ou não ver a parte final do formulário.

Agora há que fazer coisas quando se clica no checkbox com id="demaior_idade". Para isso, deveríamos criar um código Javascript e jQuery como este:

```
$(document).ready(function(){
  $("#demaior_idade").click(function(){
    //o que se deseja fazer ao receber um clique o checkbox
  });
});
```

Como já comentamos, estas linhas servem para especificar eventos. $(document).ready() se faz para lançar instruções quando o navegador está preparado para recebê-las e $("#demaior_idade").click() serve para realizar ações quando se clicou sobre o elemento com id "demaior_idade", que é o checkbox do formulário. (Leia o artigo Passos para trabalhar com jQuery para mais informação sobre este ponto).

Agora temos que ver as instruções que devemos executar como resposta a esse evento.

```
if ($("#demaior_idade").attr("checked")){
  $("#formulariomaiores").css("display", "block");
}else{
  $("#formulariomaiores").css("display", "none");
}
```

Basicamente, o que fazemos é comprovar o estado do atributo "checked" do elemento "#demaior_idade". Isto se faz com o método attr() indicando como parâmetro o valor do atributo HTML que queremos comprovar. Então, se estava "checked", se tem que mostrar o elemento e se não estava marcado o checkbox, haveria que ocultá-lo.

Espero que se entenda bem. Agora deixo aqui o código completo deste exemplo:

```
<html>
<head>
  <title>Mostrar Ocultar</title>
<script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<script>
$(document).ready(function(){
  $("#demaior_idade").click(function(evento){
    if ($("#demaiora_idade").attr("checked")){
      $("#formulariomaiores").css("display", "block");
    }else{
      $("#formulariomaiores").css("display", "none");
    }
  });
});
</script>
</head>

<body>

<form>
Nome: <input type="text" name="nome">
<br>
<input type="checkbox" name="maior_idade" value="1" id="demaior_idade"> Sou maior de idade
<br>
<div id="formulariomaiores" style="display: none;">
Dado para maiores de idade: <input type="text" name="maiores_idade">
</div>
```

```
</form>

</body>
</html>
```

Finalmente, podemos vê-lo em funcionamento em uma página à parte.

*Artigo por **Miguel Angel Alvarez - Tradução de JML***

Efeitos rápidos com jQuery

Uma das vantagens mais destacadas de jQuery é a realização de efeitos especiais para páginas web, que se desenvolvem rápido e com pouco código fonte. Estes efeitos servem para aplicar dinamismo a uma página web e uma resposta atrativa frente a interação com o usuário, o que faz com que as páginas programadas com jQuery ofereçam uma imagem desejada.

Os efeitos com jQuery, pelo menos muitos deles, podem se realizar sem muitas complicações, já que existem umas funções que simplificam a tarefa dos desenvolvedores (Ver a biblioteca Effects). Em muitos casos conseguir um efeito nos levará a uma linha de código em nosso programa, como esta:

```
$("#camadaefeitos").hide("slow");
```

Com isto conseguimos que o elemento com id="camadaefeitos" desapareça da página. Porém ademais, o efeito não é uma simples fusão do elemento na página (tornar-se transparente), e sim, que também vai acompanhado de uma redução de tamanho progressiva até desaparecer.

Combinando os efeitos com a interação de usuário, por meio de eventos, podemos conseguir que os efeitos respondam às ações do visitante, o que multiplica as possibilidades, mantendo a simplicidade, elegância e facilidade de manutenção do código Javascript. Veremos em um exemplo a seguir.

Exemplo de efeitos e interação em jQuery

No seguinte exemplo vamos mostrar um uso simples das funções de efeitos de jQuery. Vamos implementar um simples efeito de ocultar e mostrar um elemento da página web.

Podemos ver o exemplo em funcionamento em uma página à parte.

Como podemos ver, vamos ter uma camada e um par de links. Com jQuery faremos que ao clicar os links se oculte e se mostre a camada, com as funções da biblioteca Effects.

Para começar, este é o código HTML do exemplo, que compreende tanto a camada como os links.

```
<div id="camadaefeitos" style="background-color: #cc7700; color:fff; padding:10px;">
Isto é uma camada que nos servirá para fazer efeitos!
</div>

<p>
<a href="#" id="ocultar">Ocultar a camada</a> |
<a href="#" id="mostrar">Mostrar a camada</a>
</p>
```

Agora vem a parte interessante, que é na que associamos eventos a estes dois links e codificamos as chamadas às funções de Effects, que farão com que se mostre e se oculte a camada.

O código Javascript, que faz uso de jQuery seria o seguinte:

```
$(document).ready(function(){
  $("#ocultar").click(function(event){
   event.preventDefault();
   $("#camadaefeitos").hide("slow");
  });

  $("#mostrar").click(function(event){
   event.preventDefault();
   $("#camadaefeitos").show(3000);
  });
});
```

Como se pode ver, primeiro temos que definir o evento ready do objeto $(document), para fazer coisas quando o documento está preparado para receber ações.

Logo, se define o evento click sobre cada um dos dois links. Para isso, invoco o método click sobre o link, que selecionamos com jQuery através do identificador da etiqueta A.

```
$("#ocultar").click(function(event){
```

Com isto estou definindo o evento click sobre o elemento com id="ocultar".

> **Nota:** ler o artigo anterior Passos para utilizar jQuery em sua página web, no qual falávamos sobre eventos e outras generalidades deste framework Javascript. Poderemos encontrar explicações mais detalhadas sobre como definir eventos Javascript com jQuery.

Dentro da função a executar quando se clica, se coloca a chamada à função dos efeitos.

```
$("#camadaefeitos").hide("slow");
```

Isto faz com que nossa camada, a que havíamos visto o identificador (atributo id) "camadaefeitos", se oculte. Passamos o parâmetro "slow" porque queremos que o efeito seja lento.

Agora vejamos a função dos efeitos com outra chamada:

```
$("#camadaefeitos").show(3000);
```

Isto faz com que se mostre o elemento com id "camadaefeitos", e que o processo de mostrar-se dure 3000 milésimos de segundos.

Não há mais complicações, portanto se tiverem entendido isto, já saberão fazer efeitos simples, porém atrativos com jQuery em sua página web. Agora poderão ver o código completo deste exemplo criado por CriarWeb.com para demonstrar o uso de efeitos.

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html>
<head>
  <title>Efeitos com jQuery</title>
  <script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<script>
$(document).ready(function(){
  $("#ocultar").click(function(event){
   event.preventDefault();
   $("#camadaefeitos").hide("slow");
  });

  $("#mostrar").click(function(event){
```

```
    event.preventDefault();
    $("#camadaefeitos").show(3000);
   });
});
</script>
</head>

<body>

<div id="camadaefeitos" style="background-color: #cc7700; color:fff; padding:10px;">
Isto é uma camada que nos servirá para fazer efeitos!
<br>
<br>
Coloco este texto simplesmente de prova
</div>

<p>
<a href="#" id="ocultar">Ocultar a camada</a> |
<a href="#" id="mostrar">Mostrar a camada</a>
</p>

</body>
</html>
```

Por último, coloco o link de novo ao exemplo em funcionamento.

Como se pode comprovar, fazer efeitos com jQuery é bastante simples. Claro que há outros detalhes importantes e outros tipos de efeitos e funcionalidades de personalização dos mesmos, porém isto nos serviu para demonstrar o simples que é trabalhar com este framework Javascript. Nos próximos capítulos continuaremos explorando casos de uso típicos de jQuery.

*Artigo por **Miguel Angel Alvarez - Tradução de JML***

Callback de funções jQuery

Freqüentemente, quando fazemos aplicações enriquecidas do lado do cliente com jQuery nos vemos na necessidade de encadear várias chamadas a funções, para que uma se execute atrás da outra, criando um efeito mais elaborado. Neste artigo veremos o simples que é realizar aquilo que em inglês se chama "callback", ou seja, uma função que se executa depois de outra.

Acumular funções, para que se executem uma atrás da outra, nos servirá para fazer muitas coisas. No nosso dia -a -dia com jQuery iremos encontrando a utilidade, porém, no momento, para explicar um caso em que nos resultará imprescindível, me ocorre que desejemos fazer uma sequência de efeitos e mudanças dinâmicos em um elemento.

Por exemplo, imaginemos que se deseja ocultar uma camada com um efeito de esmaecer (de opaco a transparente), depois mudá-la para outra posição e voltar a mostrá-la (já na nova posição) com outro efeito de esmaecer (neste caso de transparente a opaco). Em principio, poderíamos pensar em fazer um código como este:

```
$("#minhacamada").fadeOut(2000);
$("#minhacamada").css({top: 300, left:200});
$("#minhacamada").fadeIn(2000);
```

Neste caso estamos alterando as propriedades de uma camada com id="minhacamada". Primeiro chamamos a fadeOut() para ocultá-la com um fundido, que durará 2 segundos (veja-

se o parâmetro 2000, que são os milésimos de segundo que durará o efeito). Depois, alteramos a posição da camada, trocando seus atributos CSS. Para terminar, tornamos a mostrá-la com um esmaecimento de outros 2000 milésimos de segundo.

> **Nota:** para poder entender melhor estas chamadas a efeitos, por favor, consulte o artigo Efeitos Rápidos com jQuery.

Se lançamos a execução destas sentenças, tal como aparece no código, será como se se executassem todas ao mesmo tempo. Como os fadeOut e fadeIn levarão 2 segundos a serem executados e as mudanças das propriedades CSS top e left são imediatos, o que ocorrerá será que primeiro veremos a camada se mover para a nova posição e depois veremos os dois efeitos de esmaecer.

O melhor para entender este caso é vê-lo em funcionamento.

Como realizar uma pilha de execução de funções

Já vimos um dos casos em que necessitaríamos executar funções em uma pilha, uma depois da outra, esperando que termine completamente a execução de qualquer efeito ou ação antes de começar com a seguinte. Agora, vamos ver como fazê-lo com jQuery.

Simplesmente temos que saber que todas as funções ou métodos de jQuery podem receber um parâmetro adicional com o nome da função que se tem que executar depois que termine o processamento da primeira. Essa segunda função que se executa depois da primeira é a que se conhece em inglês por callback. Um exemplo simples para entendê-lo.

```
minhaFuncao ("parametros da funcao", funcionCallback);
```

Nesse esquema de chamada à minhaFunção(), se estão passando dois parâmetros. O primeiro seria um suposto parâmetro que necesitasse minhaFunção() e o segundo, que é o que nos interessa neste caso, o nome da função que se tem que executar depois que acabe. Com este código, primeiro se executa minhaFunção() e quando acaba completamente, se executa funçãoCallback(). Contudo, atenção que este exemplo foi simplificado para que se possa entender facilmente e esta sintaxe só valerá se funçãoCallback não receber parâmetros, porque não os podemos indicar com o nome da função. Vejamos então uma forma de fazer com que este callback funcione sempre:

```
minhaFuncao ("parametros da funcao", function(){
  funcionCallback();
});
```

Com este código, que funcionaria igual que o anterior, o bom é que sim podemos indicar os parâmetros que se necessitem para a chamada a funçãoCallback().

Exemplo real de callback com jQuery

Agora que aprendemos toda a teoria, vejamos um exemplo prático que solucionaria o problema comentado anteriormente sobre o processamento de diversos efeitos e mudanças nas propriedades dos objetos, para que se façam sempre na sequência que desejamos. Trata-se simplesmente de aplicar as chamadas com callback que vimos.

```
$("#minhacamada").fadeOut(1000, function(){
  $("#minhacamada").css({'top': 300, 'left':200});
  $("#minhacamada").fadeIn(1000);
});
```

Como se pode ver, na chamada a fadeOut() estamos passando como parâmetros o valor 1000, que são os milésimos de segundo que tem que durar o efeito fade out(esmaecido a transparente), e a seguir a função callback, que se executará depois que fadeOut() tenha terminado.

Como o método css() (encontra-se como primeira instrução da função callback) é instantâneo, não se necessita fazer um callback para executar o fadeIn(), pode-se escrever diretamente na linha seguinte de código. Assim , percebe- se que o callback, pelo menos neste exemplo, só é necessário ser feito quando se executam funções que realizarão um processamento prolongado.

Podemos ver este exemplo de callback em uma página à parte.

Até aqui, ao longo dos capítulos deste manual de jQuery, lemos a introdução deste popular framework Javascript, que pode ser visto na seção "How to use jQuery" publicada na página de documentação. Em CriarWeb.com enriquecemos este tutorial de jQuery trazendo novos exemplos e explicações adicionais, de modo que qualquer pessoa, com alguns conhecimentos básicos de Javascript, possa entender e aprender a usar estas bibliotecas de programação cross-browser. Agora, fazemos uma pausa neste manual y voltaremos breve com novos artigos para explicar outros assuntos sobre a programação com jQuery.

*Artigo por **Miguel Angel Alvarez - Tradução de Celeste Veiga***

Uso de Ajax muito simples com jQuery

Chegou o momento de fazer uma primeira aproximação a Ajax, na série de artigos que estamos publicando em CriarWeb.com para mostrar o uso deste framework (leiam o Manual de jQuery).

Uma das vantagens dos frameworks Javascript é que nos permitem desenvolver scripts que utilizam Ajax de uma maneira muito fácil , além de não nos complicar a vida com a compatibilidade entre diferentes navegadores. Sem dúvida, qualquer pessoa que saiba um pouco de Javascript, pode em pouco tempo começar a utilizar Ajax com algum destes frameworks. Nós vamos demonstrar como é simples com jQuery.

A primeira impressão que tive sobre o uso de Ajax com jQuery foi realmente grata, pela facilidade com que se pode realizar um primeiro exemplo. Trata-se de um exemplo extremamente simples, mas que serve para abrir as portas para muitas aplicações interessantes. Temos que tirar o chapéu ante a extrema simplicidade com que se pode fazer um exemplo similar com jQuery. Basta dizer que neste exemplo de Ajax utilizaremos apenas 4 linhas de código, das quais apenas 1 é para executar a própria chamada ao servidor por Ajax.

Trazer um conteúdo com Ajax ao clicar em um link

Neste simples exemplo faremos chamada a Ajax, para trazer um conteúdo, quando se clique em um link. Isto foi colocado em funcionamento no servidor de CriarWeb.com e pode ser visto em uma página à parte.

Aqui podemos ver o link, no qual colocamos um identificador (atributo id) para selecioná-lo desde jQuery.

```
<a href="#" id="linkajax">Clique aqui!</a>
```

Se lemos até aqui o Manual de jQuery saberemos como atribuir um evento a um link. No entanto, recordemos como atribuir uma função para quando se clica em um link:

```
$(document).ready(function(){
  $("#linkajax").click(function(evento){
    //elimino o comportamento padrão do link
    evento.preventDefault();
    //Aqui colocaria o código da chamada a Ajax
  });
})
```

Só já falta colocar em funcionamento Ajax dentro da função do evento "click" sobre o link. Porém antes, vou necessitar de uma camada onde mostrar o conteúdo que vamos receber do servidor com a chamada Ajax. Colocaremos id="destino" para nos poder referir a ela desde jQuery:

E agora a parte mais interessante, onde podemos ver quão fáceis são as coisas com este framework Javascript. Uma única linha de código será suficiente:

```
$("#destino").load("conteudo-ajax.html");
```

Com esta simples sentença estamos realizando uma chamada a Ajax com jQuery. É uma simples invocação ao método load() de um elemento da página, em concreto o que havíamos colocado com id="destino". Ao método load(), passamos como parâmetro a rota da página que queremos carregar dentro do elemento.

O arquivo que carregamos com load() neste exemplo é "conteudo-ajax.html" e simplesmente lhe colocamos um texto qualquer e o guardamos neste mesmo diretório da página web onde está o script jQuery.

> **Nota:** para que este exemplo funcione deve ser colocado em um servidor web, já que a chamada por Ajax é feita por http e o arquivo que se carrega então tem que ser recebido por um servidor web, que o mande com esse protocolo ao navegador. Se você coloca os arquivos em seu disco duro e os executa da mesma forma, não funcionará. Você pode utilizar qualquer espaço de hosting que você tenha ou então um servidor web que possa ter instalado em seu computador.

Simples assim! Podemos ver o código completo deste exemplo:

```
<html>
<head>
  <title>Ajax Simples</title>
<script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<script>
$(document).ready(function(){
  $("#enlaceajax").click(function(evento){
    evento.preventDefault();
    $("#destino").load("conteudo-ajax.html");
  });
})
</script>
</head>
<body>

<a href="#" id="linkajax">Clique aqui!</a>
<br>
<div id="destino"></div>

</body>
</html>
```

Podemos ver o exemplo em funcionamento em uma página à parte.

Caberia comentar que jQuery tem muitos outros métodos para realizar conexões por Ajax, que podem servir para muitos outros casos de trabalho que possamos encontrar. Nós escolhemos o mais simples para dar uma primeira demonstração das possibilidades.

Passar parâmetros e executar ações depois da chamada a Ajax

O método load() que vimos no exemplo anterior tem outros dois parâmetros opcionais que podemos utilizar, caso necessário:

Parâmetros a passar à página: a página que carreguemos com Ajax pode receber parâmetros pela URL, que se especificam com a típica notação de propriedades e valores de jQuery.

```
{nome: "Pepe", idade: 45}
```

Por exemplo, com este código estaríamos enviando à página os dados nome e idade, com os valores "pepe" e 45. Esses dados viajariam na URL, pelo método "POST".

> **Nota:** Desde jQuery 1.3 também se podem enviar os parâmetros à página a ser carregada com Ajax por meio de uma variável de tipo string, no lugar de uma notação de objetos como havíamos comentado. Quando se usa uma string para especificar os parâmetros a serem enviados no request http, estes viajam pelo método GET. Quando se utiliza una notação de objetos como a que vimos, os dados viajam pelo método POST.

Função callback: como outros métodos de jQuery, podemos especificar opcionalmente uma função a ser executada quando se termine de executar o método. Neste caso, quando se termine a chamada Ajax, se podem fazer ações, como apagar uma mensagem típica de "carregando...".

> **Nota:** Em um artigo anterior já comentamos o habitual uso de funções callback em jQuery.

Agora vejamos um código onde fazemos uso destes dois parâmetros:

```
$(document).ready(function(){
  $("#linkajax").click(function(evento){
    evento.preventDefault();
    $("#destino").load("recebe-parametros.php", {nome: "Pepe", idade: 45}, function(){
      alert("recebidos os dados por ajax");
    });
  });
})
```

Neste caso estamos carregando com load() uma página PHP chamada "recebe-parametros.php". Estamos passando os parâmetros "nome" e "idade" a uma página, que podemos obter por GET. Além disso, colocamos uma função callback na que simplesmente fazemos um alert(), que se executará quando a chamada a Ajax tenha terminado.

Este seria o código fonte de "recebe-parametros.php":

```
Recebido o seguinte dado:
<br>
Nome: <?php echo $_POST["nome"];?>
<br>
```

```
Idade: <?php echo $_POST["idade"];?>
```

Podemos ver este exemplo em uma página à parte.

Com isto podemos comprovar quão simples é realizar com jQuery um carregamento de conteúdos recebidos por Ajax.

*Artigo por **Miguel Angel Alvarez - Tradução de Celeste Veiga***

Ajax jQuery com mensagem de carregamento

Vamos ver algumas coisas típicas que podemos desejar fazer com Ajax em uma página web, facilitando o processo de desenvolvimento com o framework Javascript jQuery. Nesta linha de artigos publicamos há pouco tempo um sobre o uso de Ajax em jQuery simplificado. No mencionado artigo vimos como fazer uma chamada Ajax com 1 só linha de código (o próprio Ajax era uma linha de código, embora sejam necessárias mais instruções para associar as ações Ajax como resposta a eventos na página).

Uma das coisas que habitualmente podemos querer fazer ao realizar uma chamada Ajax é a criação de uma mensagem de carregamento, que podemos colocar com um simples texto "carregando..." ou com a típica imagem de Ajax Loader. Neste artigo veremos como criar essa mensagem de carregamento ao realizar uma solicitação Ajax com jQuery.

Aos interessados, recomenda-se ver este exemplo de Ajax em uma página à parte.

Por que uma mensagem de carregamento ao fazer Ajax

Quando fazemos uma solicitação por Ajax, os resultados de tal chamada, às vezes, demoram a chegar. Durante esse tempo o usuário pode não ver nenhuma reação por parte do navegador, o que pode confundi-lo e levá-lo a pensar que não clicou corretamente no link ou no botão. É normal, nesse caso, que o usuário clique repetidas vezes um link ou em um botão de envio de formulário, gerando repetidas e desnecessárias chamadas ao servidor, o que pode acarretar diversos problemas.

Assim, é conveniente mostrar uma mensagem de carregamento para indicar que sua solicitação foi realizada e o processo está em andamento e esperando resposta do servidor.

Vamos explicar a implementação desta mensagem de carregamento, mas sempre tendo em conta que nossa intenção neste exemplo é manter um código pequeno que possa ser entendido facilmente. Queremos assinalar, no entanto, que as coisas podem ser feitas de outra maneira, algo melhoradas, quando saibamos mais coisas sobre a framework Javascript jQuery e coloquemos em funcionamento algumas práticas aconselháveis de programação orientada a objetos.

Preparando o código HTML

Como um primeiro passo, vamos mostrar o código HTML que teremos na página que fará a solicitação Ajax.

```
<a href="#" id="linkajax">Clique aqui!</a>
<div id="carregando" style="display:none; color: green;">Carregando...</div>
<br>
```

```
<div id="destino"></div>
```

Como se pode ver, temos três elementos: 1) o link, que ativará a chamada Ajax ao clicar sobre ele. 2) uma camada com id="carregando" que é onde está a mensagem de carregamento (nós colocamos um texto, mas se pode colocar qualquer coisa, como o típico gif animado que mostra que a solicitação está sendo processada (convém perceber que também essa camada carregando está oculta no momento, graças ao atributo de estilo CSS display:none). 3) a camada "destino", onde mostraremos a resposta recebida depois da solicitação Ajax.

Chamada Ajax com jQuery e a mensagem de carregamento

Neste ponto vamos descrever como se teria que fazer a chamada ao servidor com Ajax, porém o certo é que este processo já foi explicado com detalhes anteriormente, motivo pelo qual remeto ao artigo Uso de Ajax muito simples com jQuery, onde vocês encontrarão explicações que vou dar por sabidas neste caso. O que explicarei neste artigo é como se deve mostrar a mensagem de carregamento quando se inicia a chamada e como eliminá-la uma vez que tenhamos recebido a resposta por Ajax.

Outra coisa que o leitor deverá conhecer para poder entender este exemplo é Mostrar e ocultar elementos da página com jQuery.

```
$(document).ready(function(){
  $("#linkajax").click(function(evento){
    evento.preventDefault();
    $("#carregando").css("display", "inline");
    $("#destino").load("pagina-lenta.php", function(){
      $("#carregando").css("display", "none");
    });
  });
})
```

Vou comentando linha a linha o código anterior.

Na primeira linha se está especificando um método ready() sobre o objeto document, que serve para gerar um código a ser executado quando a página estiver preparada para receber instruções Javascript que trabalhem com o DOM.

Na segunda linha se acessa o elemento com identificador "linkajax" (ou seja, o link que ativará o Ajax) para definir um evento "click".

Na linha seguinte se executa o método preventDefault() sobre o evento produzido ao clicar sobre o link. Isto se faz para anular o comportamento típico do link.

Agora vem a parte na que se mostrará a mensagem de carregamento:

```
$("#carregando").css("display", "inline");
```

Simplesmente se mostra a camada com id="carregando", com uma simples troca no atributo CSS display da camada. Essa troca de atributo é feita com o método css() sobre o elemento que queremos alterar, tal como se viu no artigo Mostrar e ocultar elementos da página com jQuery.

Agora, com a seguinte linha de código:

```
$("#destino").load("pagina-lenta.php", function(){
```

Faz-se a chamada Ajax, com o método load() sobre a camada que queremos atualizar com o conteúdo trazido por Ajax, que é a camada com id="destino". Este método recebe a URL da página a ser carregada e uma função callback, que será executada depois que o método load() tenha terminado de ser processado, isto é, depois que a solicitação Ajax tenha sido recebida e

tenha mostrado seu conteúdo na camada que recebe o método.

Então, nessa função callback, temos que ocultar a camada com a mensagem de carregamento, posto que quando se execute esta função já se terá realizado todo o processamento Ajax. Isso é conseguido com o método css(), alterando a propriedade display, de maneira similar à que havíamos realizado para mostrar a camada carregando.

```
$("#carregando").css("display", "none");
```

Isto é tudo. Realmente não há nenhuma complicação especial. Embora, como disse, estas coisas possam ser feitas de uma maneira mais elegante quando aprendamos um pouco mais sobre jQuery.

Caso sirva para esclarecer as coisas, deixo a seguir o código completo da página que faz a solicitação com jQuery:

```
<html>
<head>
  <title>Ajax Simples</title>
<script src="jquery-1.3.2.min.js" type="text/javascript"></script>
<script>
$(document).ready(function(){
  $("#linkajax").click(function(evento){
    evento.preventDefault();
    $("#carregando").css("display", "inline");
    $("#destino").load("pagina-lenta.php", function(){
      $("#carregando").css("display", "none");
    });
  });
})
</script>
</head>

<body>
Isto é um Ajax com uma mensagem de carregando...
<br>
<br>

<a href="#" id="linkajax">Clique aqui!</a>
<div id="carregando" style="display:none; color: green;">Carregando...</div>
<br>
<div id="destino"></div>

</body>
</html>
```

Código da página PHP invocado por Ajax

O código da página PHP que trazemos por ajax "pagina-lenta.php" é o seguinte:

```
<?php
sleep(3);
echo ("demorou 3 segundos para executar esta pagina...");
?>
```

Na realidade não tem nada de especial. Simplesmente fazemos um sleep(3) para ordenar a PHP que espere 3 segundos antes de terminar de processar a página e enviá-la ao cliente. Desse modo, consigo que a solicitação http demore um pouco em ser respondida e possamos ver a mensagem de carregamento durante um pouco mais de tempo na página.

Finalmente, ponho de novo o link para ver este exercício em funcionamento.

*Artigo por **Miguel Angel Alvarez - Tradução de Celeste Veiga***