

# jQuery

Escreva Menos, Faça Mais

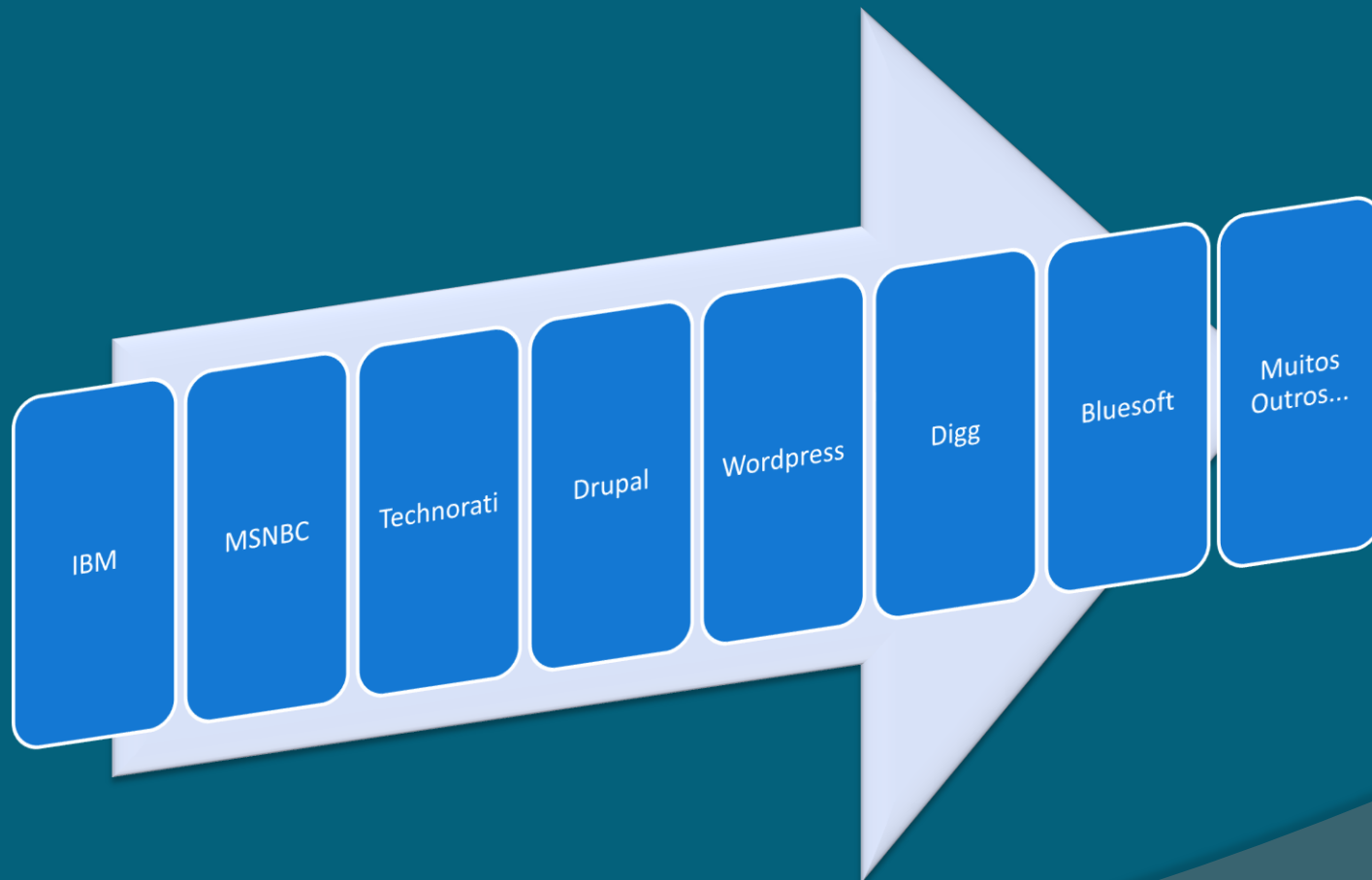


André Faria Gomes

# O Que é jQuery?

- ⦿ Biblioteca JavaScript OpenSource.
- ⦿ Simplifica a interação entre HTML e JavaScript.
- ⦿ Excelente comunidade.
- ⦿ Centenas de posts diários em fóruns .
- ⦿ Plugins e extensões.
- ⦿ Funciona nos principais browsers.
- ⦿ Possui apenas 20Kb.

# Quem usa jQuery?



# \$('jQuery')

- Possui uma sintaxe extremamente simples com a finalidade de encontrar elementos em uma página HTML e permitir que se faça algo com eles.
- Sua Sintaxe Básica é \$('elemento')
- Fluent Interfaces

# jQuery Core



# Selecionar Elementos por Tipo

- Todos os elementos div
- `$('div')`
- Todas as tabelas
- `$('table')`
- Todos os elementos div, h1 e h2
- `$('h1,h2,div');`
- Todos os Elementos
- `$('*')`

# Por Id ou Class

- ⊙ `<div id="menu">`
  - `<div id="home"> Home </div>`
  - `<div class="red"> Links </div>`
- ⊙ `</div>`
  
- ⊙ `$('#menu')` //obtem o item menu
- ⊙ `$('#home')`
- ⊙ `$('.red')` //obtem o div com class='red'

# Por Estado

- ⦿ `$('div:empty');` //Sem sub-elementos
- ⦿ `$('input:enabled');` //Habilitados
- ⦿ `$('input:disabled');`
- ⦿ `$('input:checked');`
- ⦿ `$('option:selected');`
- ⦿ `$('div:visible');` // Visíveis
- ⦿ `$('div:hidden');` // Invisíveis



# Por Atributos

- Elementos que possuam um atributo checked.
- \$(':checkbox[checked]);
- \$("a[@name]);
- \$('div[name=item]); //igual a item
- \$('[name^=item]); //começe com item
- \$('[name\$=item]); //termine com item
- \$('[name\*=item]); //contenha item

# Por Pais e Filhos

- ◉ `$('p strong');` //Obtem um elemento strong que possua um parágrafo.
- ◉ Obtem um elemento li que esteja dentro de um elemento ul.
- ◉ `$('li > ul');`
- ◉ Elementos h3 com filhos (inclusive textos)
- ◉ `$('h3:parent');`
- ◉ Elementos li com o texto “Bluesoft”
- ◉ `$('li:contains(“Bluesoft”');`

# Formulários

- `$(':input')`: //Todos os elementos input
- `$(':text')`: //Todos os elementos input type="text"
- `$(':password')`
- `$(':radio')`
- `$(':checkbox')`
- `$(':file')`
- `$(':submit')`
- `$(':image')`
- `$(':reset')`
- `$(':button')`: //type="button" ou `<button/>`

# Por Índice?

```
<select id="numeroFuncionarios">  
  <option>até 50</option>  
  <option>até 200</option>  
  <option>até 500</option>  
  <option>500 ou mais</option>  
</select>
```

- `$('option:even');` //par
- `$('option:odd');` //impar
- `$('option:eq(0)');` //obtem o elemento de índice 0
- `$('option:nth(1)');` //obtem o elemento de índice 1
- `$('option:gt(1)');` // todos com índices maiores que 1
- `$('option:lt(4)');` // todos com índices menores que 4
- `$('option:first');` //o primeiro
- `$('option:last');` //o último
- `$('option').slice(1,3);` //Do Índice 1 até o 3
- Observe que o índice inicia no Zero

# Encontrando o Índice

- Pesquisa em todos os elementos selecionados e retorna o índice do elemento se encontrado, inicia em 0.
- Retorna -1 se o elemento não for encontrado.
- `<div id="foobar"><b></b><span id="foo"></span></div>`
- `$("*").index( $('#foobar')[0]);`
  - Retornaria 0
- `$("*").index( $('#foo')[0]);`
  - Retornaria 1
- `$("*").index( $('#bar')[0]);`
  - Retornaria -1

# Chaining

- ⦿ `$("#el").hide();`
- ⦿ `$("# el").hide().color("red");`
- ⦿ `$("# el").hide().color("red").slideDown();`

# Modificando Seletores

- `add()`
- `children()`
- `eq()`
- `filter()`
- `find()`
- `gt()`
- `lt(),`
- `next()`
- `not()`
- `parent()`
- `parents()`
- `siblings()`
- `end()`

# CSS

- ⦿ `<p id="xx" style="color:red">a</p>`
- ⦿ `css("")` para obter
- ⦿ `$("#xx").css('color') //retorna red`
- ⦿ `css(",")` para alterar
- ⦿ `$("#xx").css('color','blue');`
- ⦿ `$("#xx").css({'color':'cyan', 'font-size':'90'});`



# Offset

- Obtem o posicionamento do Elemento
- `var top = $("#menu").offset().top;`
- `var left = $("#menu").offset().left;`

# Altura e Comprimento

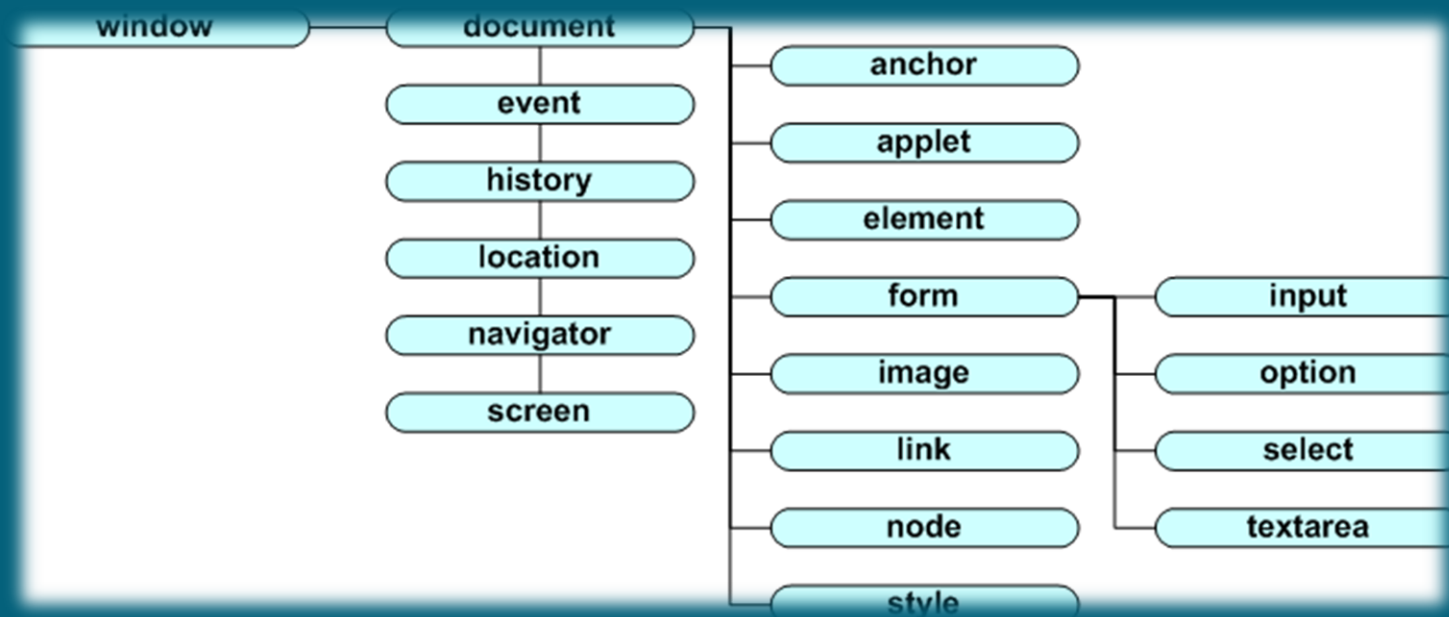
- ⦿ Obter a altura
  - `$( 'el' ).height();`
- ⦿ Alterar a altura
  - `$( 'el' ). height(20);`
- ⦿ Obter o comprimento
  - `$( 'el' ). width();`
- ⦿ Alterar o comprimento
  - `$( 'el' ). Width(100);`



# Length / Size

- ⦿ Retorna o número de elementos selecionados.
- ⦿ ` `
- ⦿ `$("img").length;`
  - Retornaria 2
- ⦿ `$("img").size;`
  - Retornaria 2

# DOM



# attr

- ⦿ Altera um atributo de um elemento.
- ⦿ ``
- ⦿ Acessar um atributo
  - `$('#imagem').attr('src')`
    - ⦿ Retornaria "scrum.jpg"
- ⦿ Alterar um atributo
  - `$("#imagem").attr("src","openup.jpg");`
    - ⦿ Altera o src do elemento img.

# attr

- ⦿ Altera um atributo de um elemento, através de uma função personalizada.
- ⦿ `$("#img").attr("title", function(index) {`
  - `return this.src;`
- ⦿ `});`
- ⦿ Antes
  - ``
- ⦿ Depois
  - ``

# removeAttr

- ⦿ Remove um atributo de um elemento.
- ⦿ `$("input").removeAttr("disabled")`
- ⦿ Antes
  - `<input type="text" name="idade" disabled="disabled"/>`
- ⦿ Depois
  - `<input type="text" name="idade"/>`

# val

- ⦿ `<input id="a" type="text" value="444">`
- ⦿ Obter o valor de um elemento
  - `$('#a').val();`
    - Retornaria "444"
- ⦿ Alterar o valor de um elemento
  - `$('#a').val("222")`



# class

- ⦿ `addClass();`
- ⦿ `hasClass();`
- ⦿ `toggleClass();`
- ⦿ `removeClass();`
  
- ⦿ `$('#div').addClass('vermelho');`

# html() / text()

- ⦿ `<div id="story">`
  - Era uma vez....
- ⦿ `</div>`
- ⦿ Retorna o conteúdo do div story
- ⦿ `$('#story').html();`
- ⦿ Altera o conteúdo do div story
- ⦿ `$('#story').html('Novo Conteúdo!!!!');`
- ⦿ *O mesmo conceito é aplicado para `text()` e `text('...')`.*

# Inserindo Dentro do Elemento

- ⦿ `<div id="item"> Compras </div>`

- ⦿ `$('#item').append(' OK ')`

- Compras OK

- ⦿ `$('#item').prepend(' - ')`

- - Compras

- ⦿ `$("p").appendTo("#foo");`

- ⦿ `$("p").prependTo("#foo");`

# Inserindo Fora do Elemento

- ⦿ `<div id="item"> Compras </div>`
- ⦿ Insere depois do elemento
- ⦿ `$('#item').after('<div>fim</div>');`
- ⦿ Insere antes do elemento
- ⦿ `$('#item').before('<div>inicio</div>');`
- ⦿ `$("#p").insertAfter("#foo");`
- ⦿ `$("#p").insertBefore("#foo");`

# Clone

- ⦿ Clonar um elemento
  - `clone();`
- ⦿ Clonar um elemento incluindo eventos
  - `clone(true);`
- ⦿ `$("b").clone().prependTo("p");`

# Empty

- ⦿ Remove todos os elementos filhos (child nodes) de um elemento.
- ⦿ `$("p").empty();`
- ⦿ Antes
  - `<p>Hello, <span>Person</span> <a href="#">and person</a></p>`
- ⦿ Depois
  - `<p></p>`

# E tem mais....

- `replaceWith();`

# remove()

- ⦿ Remove todos os objetos selecionados

- ⦿ `$('remove')`

- Antes
  - `<p>Hello</p> how are <p>you?</p>`
- Depois
  - `how are`

- ⦿ `$("p").remove(".hello");`

- Antes
  - `<p class="hello">Hello</p> how are <p>you?</p>`
- Depois
  - `how are <p>you?</p>`



# wrap();

## ⦿ `<p>Test Paragraph.</p>`

- Antes

- `<p>Test Paragraph.</p>`

- Depois

- `<div class='wrap'><p>Test Paragraph.</p></div>`

# add(expression)

- Adiciona mais elementos a seleção atual.
- `$("p").add("span");`

# add(HTML)

- Adiciona mais elementos, criados “on the fly”, para o conjunto de elementos selecionados.
- `$("p").add("<span>Again</span>")`

# add(element)

- Adiciona um ou mais elementos para o conjunto de elementos selecionados.
- `$("p").add(document.getElementById("a"));`
- `$("p").add(document.forms[0].elements );`

# children(expr)

- Obtem o conjunto de elementos que contenha todos os filhos de cada elemento do conjunto de selecionados. Pode-se utilizar filtros.

`<a>x</a>`

- `<p>Hello</p><div><span>Hello Again</span><a>x</a></div><p>And Again</p>`
- `$("div").children();`
  - `<span>Hello Again</span><a>x</a>`
- `$("div").children('a');`
  - `<a>x</a>`

# contains(str)

- ⦿ Filtra o conjunto de elementos que contanha o texto especificado.
- ⦿ `<p>This is just a test.</p><p>So is this</p>`
- ⦿ `$("p").contains("test")`
  - `<p>This is just a test.</p>`

# filter(expression)

- Remove elementos que não correspondam a expressão.
- `<p class="selected">Hello</p><p>How are you?</p>`
- `$("p").filter(".selected")`
  - `<p class="selected">Hello</p>`
- `<p>Hello</p><p>Hello Again</p><p class="selected">And Again</p>`
- `$("p").filter(".selected, :first")`
  - `<p>Hello</p>, <p class="selected">And Again</p>`

# filter(function)

- ⦿ Permite que se escreva uma função com os critérios para filtragem de elementos.
- ⦿ `<p><ol><li>Hello</li></ol></p><p>How are you?</p>`
- ⦿ `$("p").filter(function(index) {`
  - `return $("ol", this).length == 0;`
- ⦿ `})`
- ⦿ `<p>How are you?</p>`



# find(expression)

- Procura por todos os elementos que combinem com a expressão. Esse método é ideal para encontrar elementos descendentes.
- `<p><span>Hello</span>, how are you?</p>`
- `$("p").find("span");`
  - `<span>Hello</span>`

# is(expression)

- ⦿ Verifica a seleção atual contra a expressão e retorna true caso ao menos um elemento da seleção combine com a expressão.
- ⦿ `<form><input type="checkbox" /></form>`
- ⦿ `$("input[@type='checkbox']").parent().is("form");`
  - true

# next(exp) / previous(exp)

- Obtem os próximos elementos. A utilização do filtro é opcional.
- `<p>Hello</p><p>Hello Again</p><div><span>And Again</span></div>`
- `$("p").next()`
  - `<p>Hello Again</p>, <div><span>And Again</span></div>`

# not(el)

- ⦿ Remove o elemento especificado do conjunto de elementos.
- ⦿ `<p>Hello</p><p id="selected">Hello Again</p>`
- ⦿ `$("p").not( $("#selected")[0] );`
  - `<p>Hello</p>`

# not(expression)

- ⦿ Remove os elementos que combinem com determinada expressão.
- ⦿ `<p>Hello</p><p id="selected">Hello Again</p>`
- ⦿ `$("p").not("#selected")`
  - `<p>Hello</p>`

# parent();

- ⦿ `<div><p>Hello</p><p>Hello</p></div>`
- ⦿ `$("p").parent()`
  - `<div><p>Hello</p><p>Hello</p></div>`
- ⦿ `<div><p>Hello</p></div><div class="selected"><p>Hello Again</p></div>`
- ⦿ `$("p").parent(".selected")`
  - `<div class="selected"><p>Hello Again</p></div>`

# parents(expression);

- Obtem todos os objetos pai menos o root (<html>). O filtro é opcional.

# Obter elementos DOM

- Utiliza-se a função `get()`;
- Obter todos os elementos IMG
  - `$("#img").get();`

Seleciona todas os elementos IMG e retorna o primeiro

- `$("#img").get(0);`

Seleciona todas os elementos IMG e retorna todos menos os três primeiros. (Obs: gt = Greater Than = Maior que)

- `$("#img").gt(2).get();`

Seleciona todas os elementos IMG e retorna os dois primeiros. (Obs: lt = Less Than = Menor que)

- `$("#img").lt(2).get();`



# Criar Elementos DOM

- Cria um elemento DOM “on-the-fly” através de uma string com tags HTML.
- `$("<div><p>Olá</p></div>").appendTo("body")`
  - O exemplo acima cria um elemento div com um paragrafo e o insere dentro do elemento body.

# Extendendo o jQuery

- ⦿ Estende o próprio objeto jQuery.
- ⦿ Pode ser usado para adicionar funções ao namespace do jQuery ou funções de plugin.

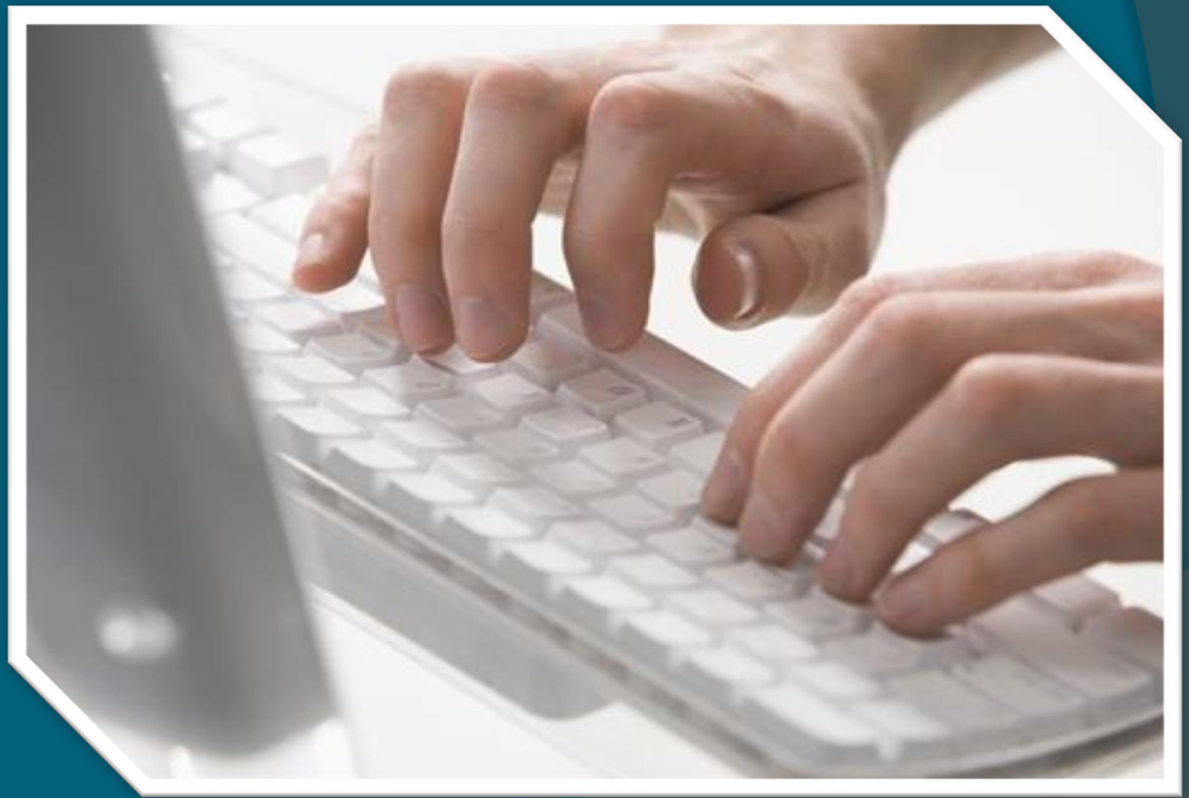
# Adicionando Funções ao Namespace

- ⦿ `jQuery.extend({`
  - `min: function(a, b) { return a < b ? a : b; },`
  - `max: function(a, b) { return a > b ? a : b; }`
- ⦿ `});`
  
- ⦿ `$.min(10,20);`
- ⦿ `$.min(10,20);`

# Adicionando Funções de Plugin

- ⦿ `jQuery.fn.extend({`
- ⦿ `check: function() {`
- ⦿ `return this.each(function() { this.checked = true; });`
- ⦿ `},`
- ⦿ `unchecked: function() {`
- ⦿ `return this.each(function() { this.checked = false; });`
- ⦿ `}`
- ⦿ `});`
  
- ⦿ `$("input[@type=checkbox]").check();`
- ⦿ `$("input[@type=radio]").unchecked();`

# Eventos e Manipulação



# Ready

- ⦿ O código inserido dentro da função “ready” é executado somente quando os objetos DOM estiverem prontos para serem manipulados.
- ⦿ Sintaxe
  - `$(document).ready(function(){//seu código failsafe});`
  - `$(function(){ //seu código failsafe });`
  - `jQuery(function($) { //seu código failsafe });`

# Event Handling

- ⦿ Bind
- ⦿ `$("p").bind("click", function(e){`
- ⦿ `alert('cliqueu no parágrafo!')`
- ⦿ `});`
  
- ⦿ Unbind
- ⦿ `$("p").unbind("click");`
  
- ⦿ One
- ⦿ `$("p").one("click", function(e){`
- ⦿ `alert('cliqueu no parágrafo!')`
- ⦿ `});`
- ⦿ `//Só exibe o alerta a primeira vez que clicar!`

# Triggers

- ⦿ `$("#button:first").click(function () {`
  - `alert('a');`
- ⦿ `});`
- ⦿ `$("#button:last").click(function () {`
  - `$("#button:first").trigger('click');`
  - `alert('b');`
- ⦿ `});`
- ⦿ Ao Clicar no último botão serão exibidos dois alerts.



# Hover / Toggle

- ⦿ `$(“a”).hover(function(){...}, function(){...});`
- ⦿ Executa f1 quando ganha foco e f2 quando perde.
  
- ⦿ `$(“a”).toggle(function(){...}, function(){...});`
- ⦿ Alterna entre a execução f1() e f2() a cada clique.

# Event Helpers

- ⦿ Existem alguns facilitadores para fazer bind de eventos comuns...
- ⦿ `click(f); // $('p').click(function(){...})`
- ⦿ Equivale a `$('p').bind('click',... function(){...})`
- ⦿ `$('p').click();`
- ⦿ Executa o evento click dos elementos 'p'.

# Event Helpers

- `click( )`
- `dblclick( )`
- `error( )`
- `focus( )`
- `keydown( )`
- `keypress( )`
- `keyup( )`
- `load( )`
- `mousedown( )`
- `mousemove( )`
- `mouseout( )`
- `mouseover( )`
- `mouseup( )`
- `resize( )`
- `scroll( )`
- `select( )`
- `submit( )`
- `unload( )`

# Efeitos Especiais



# Show / Hide

- Exibe um elemento
- `$("#p").show()`
- `$("#p").show('slow')` //slow, normal, fast
- Esconde um elemento
- `$("#p").hide()`
- `$("#p").hide('slow')` //slow, normal, fast
- Se estiver escondido exibe senão esconde
- `$("#p").toggle()`

# Sliding / Fading

- `slideDown`
  - `slideOut`
  - `slideToggle`
  - `fadeIn`
  - `fadeOut`
- 
- Você também pode criar seus próprios efeitos e animações.

# AJAX



# Load

- ⦿ Carrega a página napa.jsp no elemento e
- ⦿ `$("#e").load("napa.jsp ");`
- ⦿ Com parametrôos e evento
- ⦿ `$("#feeds").load("napa.jsp",`
  - `{limit: 25}, function(){`
    - ⦿ `alert("The last 25 entries in the feed have been loaded");`
- ⦿ `});`



# Get

- ⦿ \$.get("pedido.action",
  - { codigo: "1234", fornecedor: "44" }
- ⦿ );
  
- ⦿ \$.get("test.cgi",
  - { name: "John", time: "2pm" },
  - function(data){ alert("Data Loaded:" + data); }
- ⦿ );

# JSON

- ⦿ \$.getJSON(
  - "test.js",
  - function(json){ alert("JSON Data: " + json.users[3].name); }
- ⦿ );

# Executando Scripts

- ⦿ \$.ajax(
  - type: "GET",
  - url: "test.js",
  - dataType: "script" });
- \$.getScript("test.js",
  - function(){ alert("Script loaded and executed."); }
- );

# Enviando dados

- ⦿ \$.ajax({
  - type: "POST",
  - url: "wendola.jsp",
  - data: "name=John&location=Boston", success:  
function(msg){
    - alert( "Data Saved: " + msg ); }
  - });

# Última Versão

- ⦿ \$.ajax({
  - url: "test.html",
  - **cache: false,**
  - timeout: 20,
  - Data: {devedor:'bolha', pagador:'wendola'},
  - type: 'POST',
  - username: 'juniao'
  - password: 'cocazero',
  - error: function(){alert('error')},
  - success: function(html){
    - \$("#results").append(html); }
  - });

# Sincronicidade

- ⦿ `var html = $.ajax({`
  - `url: "some.php",`
  - `async: false }`
- `).responseText;`

# Serialização

- ⦿ function showValues() {
- ⦿     var str = \$("form").serialize();
- ⦿     \$("#results").text(str);
- ⦿     }
  
- ⦿ single=Single&multiple=Multiple&multiple=Multiple3&check=check2&radio=radio1

# Utilidades





# Each

- `var arr = [ "one", "two", "three", "four", "five" ];`
- `var obj = { one:1, two:2, three:3, four:4, five:5 };`
- `jQuery.each(arr, function() {`
- `$("#" + this).text("My id is " + this + ".");`
- `return (this != "four"); // will stop running to skip "five"`
- `});`
- `jQuery.each(obj, function(i, val) {`
- `$("#" + i).append(document.createTextNode(" - " + val));`
- `});`
- *Percorre um objeto ou array executando a função para cada elemento encontrado.*

# Grep

- `var arr = [ 1, 9, 3, 8, 6, 1, 5, 9, 4, 7, 3, 8, 6, 9, 1 ];`
- `$("div").text(arr.join(", "));`
- `arr = jQuery.grep(arr, function(n, i){`
- `return (n != 5 && i > 4);`
- `});`
- `$("p").text(arr.join(", "));`
- `arr = jQuery.grep(arr, function (a) { return a != 9; });`
- `$("span").text(arr.join(", "));`
- `//Age como um filtro de elementos`

# Mais

- ◉ Retira espaços em branco
- ◉ `jQuery.trim( str )`
  
- ◉ Verifica se o objeto é uma função
- ◉ `jQuery.isFunction( obj )`
  
- ◉ Remove elementos duplicados
- ◉ `jQuery.unique( array )`
  
- ◉ Une dois arrays sem repetir os elementos.
- ◉ `jQuery.merge(array, array)`

# Browser

- ⦿ `if($.browser.safari || $.browser.msie) {`
  - `$( function() {`
    - `alert("this is safari!");`
  - `});`
- ⦿ `}`

# jQuery User Interface



# jQuery UI

- ⦿ Drag and dropping
- ⦿ Sorting
- ⦿ Selecting
- ⦿ Resizing
- ⦿ Accordions
- ⦿ Date pickers
- ⦿ Dialogs
- ⦿ Sliders
- ⦿ Tabs.

# jQuery UI Tabs

[search](#) | [results](#) | [details](#) | [compare](#) | [customize](#) | [cart](#) | [checkout: 1, 2, 3, 4, 4b](#) | [confirmation](#) | [login](#)

---

## Checkout

1. Personal Info

2. Shipping Details

3. Billing Details

4. Order Review

5. Confirmation

check out as guest

log in

[long, boring playback of order details]

---

[search](#) | [results](#) | [details](#) | [compare](#) | [customize](#) | [cart](#) | [checkout: 1, 2, 3, 4, 4b](#) | [confirmation](#) | [login](#)

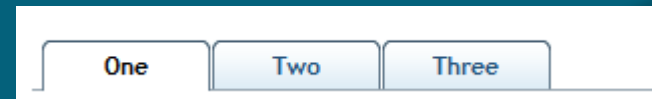
Copyright © 2007-2008 [Brian J. Dillard](#) | [Pathfinder Development](#) | [Agile Ajax](#)

Copyright © 2007-2008 [Brian J. Dillard](#) | [Pathfinder Development](#) | [Agile Ajax](#)

[search](#) | [results](#) | [details](#) | [compare](#) | [customize](#) | [cart](#) | [checkout: 1, 2, 3, 4, 4b](#) | [confirmation](#) | [login](#)

# tabs()

- `<div id="example" class="flora">`
  - `<ul>`
  - `<li><a href="#fragment-1"><span>One</span></a></li>`
  - `<li><a href="#fragment-2"><span>Two</span></a></li>`
  - `<li><a href="#fragment-3"><span>Three</span></a></li>`
  - `</ul>`
  - `<div id="fragment-1">conteúdo da aba 1</div>`
  - `<div id="fragment-2">conteúdo da aba 2</div>`
  - `<div id="fragment-3">conteúdo da aba 3</div>`
  - `</div>`
- 
- `$("#example > ul").tabs();`





# jQuery Tabs - Eventos

- ⦿ tabsselect – dispara ao selecionar uma aba
- ⦿ tabsload – dispara ao carregar uma aba
- ⦿ tabsshow – dispara ao exibir uma aba
- ⦿ tabsadd – dispara ao adicionar uma aba
- ⦿ tabsremove – dispara ao remover uma aba
- ⦿ tabsenable – dispara ao habilitar uma aba.
- ⦿ tabsdisable – dispara ao desabilitar uma aba.

# jQuery Validator

# Validação

- Evitar que o usuário mude de aba caso a aba atual não passe na validação.

- `$('#example').tabs({`
- `select: function(ui) {`
- `var isValid = ... // form validation returning true or false`
- `return isValid;`
- `}`
- `});`

# Métodos

- `tabs( options )`
  - Cria uma novo componente jQuery UI Tabs
- `tabs( "add", url, label, index )`
  - Adiciona uma nova aba
- `tabs( "remove", index )`
  - Remove uma aba
- `tabs( "enable", index )` / `tabs( "disable", index )`
  - Habilita / Desabilita uma aba
- `tabs( "select", index )`
  - Seleciona uma aba
- `tabs( "load", index )`
  - Carrega uma aba com ajax
- `tabs( "url", index, url )`
  - Carrega uma url em uma aba com ajax
- `tabs( "destroy" )`
  - Destroi o componente
- `tabs( "length" )`
  - Retorna o número de abas
- `tabs( "rotate", ms, continuing )`
  - Realiza um rodízio automático entre as abas

# jQuery Form Plugin

- O jQuery Form Plugin permite que você facilmente transforme formulários HTML comuns em formulários AJAX.

# jQuery Form Plugin

- `<form id="myForm" action="comment.php" method="post">`  
    Name: `<input type="text" name="name" />`  
    Comment: `<textarea name="comment"></textarea>`  
    `</form>`
- `$('#myForm').ajaxForm(function() {`  
    `alert("Thank you for your comment!");`  
    `});`

# Opções

- ```
$(document).ready(function() {  
  var options = {  
    target:    '#output1', // target element(s) to be updated with server response  
    beforeSend: showRequest, // pre-submit callback (return false = abort)  
    success:    showResponse // post-submit callback  
  
    url:    url    // override for form's 'action' attribute  
    type:    type    // 'get' or 'post', override for form's 'method' attribute  
    dataType: null    // 'xml', 'script', or 'json' (expected server response type)  
    clearForm: true    // clear all form fields after successful submit  
    resetForm: true    // reset the form after successful submit  
  
    // the $.ajax options can be used here too, for example:  
    //timeout: 3000  
  };  
  
  // bind form using 'ajaxForm'  
  $('#myForm1').ajaxForm(options);  
});
```

# jQuery Validator

- ⦿ Simplicidade
  - ⦿ Maturidade (Julho 2006)
  - ⦿ Padronização
  - ⦿ Mensagens de Erro
- 
- ⦿ <http://docs.jquery.com/Plugins/Validation>



# Seletores Especiais

:blank

- Traz todos os elementos em branco

:filled

- Traz todos os elementos preenchidos

:unchecked

- Traz todos os elementos não 'checados'

# Formatador de Mensagens

- ◉ Semelhante ao `Message.format()` do Java!
- ◉ 1. Cria-se um template.
- ◉ 2. Cria-se um formatador.
- ◉ 3. Aplica-se o formatador.
- ◉ `var template = "{0}, {1}, esta é sua {2} visita. Grande {1}!! ";`
- ◉ `var formatter = jQuery.format(msg);`
- ◉ `var msg = template('Bom Dia', 'Sr. Ricardex', 'quinta');`
- ◉ O valor da variável `msg` seria:
  - Bom Dia, Sr. Ricardex, esta é sua quinta visita. Grande Sr. Ricardex!!

# Métodos de Validação

## ⦿ form();

- Valida um formulário, retorna **true** se for válido, caso contrário **false**.

```
$("#meuFormulario").validate().form()
```

# Métodos de Validação

## ⦿ element(x);

- Valida um único elemento, retorna **true** se for válido caso contrário **false**.
- \$("#meuFormulario").validate().**element**("#idade");

# Métodos de Validação

## ⦿ resetForm();

- Redefine os campos com seus valores originais, remove as classes CSS que indicam elementos inválidos e esconde mensagens de erro.

## ⦿ var validator = \$("#meuForm").validate(); validator.resetForm();

# Métodos de Validação

## ⦿ showErrors

- Adiciona e exibe uma mensagem de erro programaticamente.

⦿ `var validator = $("#myform").validate();`  
`validator.showErrors({"primeiroNome": "Eu sei que o seu primeiro nome é Bob, Bob!"});`

# Métodos de Validação

- ⦿ `addMethod()`

- Adiciona um novo tipo de validação.

- ⦿ Ex: Validator que verifica se uma url é de um domínio.

- `jQuery.validator.addMethod("domain",  
function(value) {
  - ⦿ return this.optional(element) ||  
/^http://bluesoft.com.br/.test(value); }, "Por favor,  
Especifique o domínio correto");`

# Métodode de Validação

- ⊙ `addClassRules(name, rules) / addClassRules(rules)`
  - Associa regras de validação a uma Classe CSS.
- ⊙ `jQuery.validator.addClassRules({`
  - `cep: {`
    - `required: true,`
    - `minlength: 8,`
    - `maxlength: 8`
  - `}`
- ⊙ `});`
- ⊙ `<input type='text' name='cep' class='cep'/>`

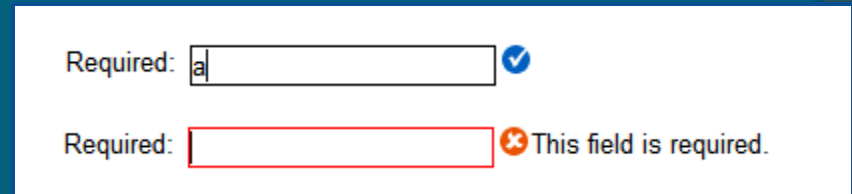


# required()

- ⦿ `$("#formulario").validate(`
  - `{ rules: { nomeDoCampo: "required" } }`
- ⦿ `});`

- ⦿ CSS

- `label.error {`
  - `background: url('unchecked.gif') no-repeat; padding-left: 16px; margin-left: .3em; }`
- `label.valid {`
  - `background: url('checked.gif') no-repeat; display: block; width: 16px; height: 16px; }`



Required:  ✓

Required:  ✗ This field is required.

# required( dependency-expression )

- ⦿ `$("#formulario").validate({`
  - `rules: {`
    - `campoTexto: {`
      - `required: "#campoCheckBox:checked"`
      - `}`
    - `}`
  - ⦿ `});`
  - ⦿ `$("#campoCheckBox").click(function() {`
    - `$("#campoTexto").valid();`
  - ⦿ `});`

Se o 'CheckBox' for selecionado o campo de texto torna-se obrigatório (required).

The image shows two examples of a form field. In the first example, the checkbox is unchecked, and the text 'Check to make next field required' is displayed. In the second example, the checkbox is checked, and the text 'Check to make next field required' is displayed, followed by a red error message 'This field is required.' with a red 'x' icon.

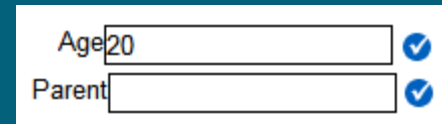
☐ Check to make next field required

☒ Check to make next field required  
 This field is required.

# required( dependency-callback )

- Torna o elemento 'required', dependendo do resultado da chamada de um 'callback'.

- `$("#formulario").validate({`
  - `rules: {`
    - `idade: { required: true, min: 3 },`
    - `pai: {`
      - `required: function(element) {`
        - `- return $("#idade").val() < 13;`
        - `}`
      - `}`
    - `}`
  - `});`
  - `$("#idade").blur(function() {`
    - `$("#pai").valid();`
  - `});`



Form showing valid state: Age 20 (checked) and Parent (checked).



Form showing invalid state: Age 12 (checked) and Parent (required, error message: This field is required).

Se a idade (age) for menor que 13 o preenchimento do campo pai (parent) torna-se obrigatório!

# remote(url)

- ⦿ Realiza um 'request' em uma 'URL' para verificar se um elemento é válido.
- ⦿ Ex: Verifica se o nome de usuário já existe.
- ⦿ `$("#formulario").validate({`
  - `rules: {`
    - ⦿ `usuario: {`
      - `required: true,`
      - `remote: "verificaUsuario.jsp"`
    - ⦿ `}`
  - `}`
- ⦿ `});`

# minlength(x) / maxlength(x)

```
⊙ $("#formulario").validate({  
  • rules: {  
    ○ field: {  
      • required: true,  
      • minlength: 3,  
      • maxlength: 4,  
      • // rangelength: [3, 4]  
    ○ }  
  • }  
⊙ });
```

# min() / max()

```
⊙ $("#formulario").validate({  
  • rules: {  
    ○ field: {  
      • required: true,  
      • min: 13,  
      • max: 23,  
      • // range: [13, 23]  
    ○ }  
  • }  
⊙ });
```

# Números

- ⦿ min:4
  - Verifica se o número é maior que 4
- ⦿ max: 100
  - Verifica se o número é menor que 100
- ⦿ range: [10,20]
  - Verifica se o número está entre 10 e 20.
- ⦿ number:true
  - Valida números no padrão americano (100.00)
- ⦿ numberDE:true
  - Valida números no padrão alemão (100,00)
- ⦿ creditcard:true
  - Valida números de cartão de crédito
- ⦿ digits:true
  - Permite apenas números inteiros

# Datas

- ⦿ date: true
  - Valida Datas
- ⦿ dateISO: true
  - Valida datas no padrão ISO
- ⦿ dataDE: true
  - Valida datas no Padrão Alemão (1.1.2008)



# equalsTo

- ⦿ Verifica se o valor de campo é igual ao valor de outro.
- ⦿ `$("#myform").validate({`
  - `rules: {`
    - ⦿ `password: "required",`
    - ⦿ `passwordConfirmation: {`
      - `equalTo: "#password"`
    - ⦿ `}`
  - `}`
- ⦿ `});`

# Mais validadores

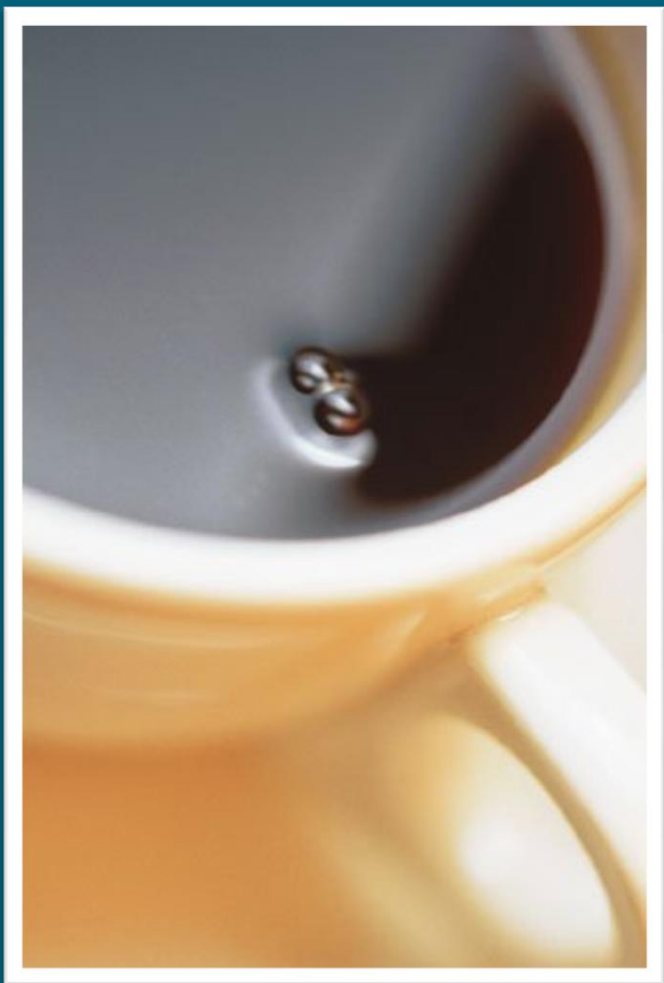
- ⦿ email: true
  - Valida endereços de e-mail.
- ⦿ url: true
  - Valida URL's
- ⦿ accept: "xls|csv"
  - Permite apenas Strings que terminem com .xls ou .cvs, usado para validar extensões de arquivos.

# Dúvidas



# Referências Bibliográficas

- [http://docs.jquery.com/Main\\_Page](http://docs.jquery.com/Main_Page)
- <http://dev.jquery.com/view/trunk/tools/api-browser/>
- <http://docs.jquery.com/UI/Tabs>
- <http://blogs.pathf.com/agileajax/2008/03/jquery-form-and.html>
- <http://docs.jquery.com/Plugins/Validation>
- <http://www.malsup.com/jquery/form/>
- <http://simplesideias.com.br>



Obrigado!

André Faria

[andrefaria@bluesoft.com.br](mailto:andrefaria@bluesoft.com.br)