

# Introdução às tecnologias Web - ITW

## Aula 9 – jQuery /

jQueryUI / Bootstrap Javascript / AJAX / JSON



# Sumário

A biblioteca jQuery



AJAX, JSON



A biblioteca jQueryUI



Os componentes javascript da biblioteca Bootstrap



# O que o jQuery

jQuery é uma biblioteca JavaScript multi-plataforma projetada para simplificar a programação (*scripting*) do lado do cliente de HTML.

A sintaxe do jQuery foi projetada para tornar mais fácil a navegação nos elementos de um documento. Exemplos:

- \* selecionar elementos DOM
- \* criar animações,
- \* manipular eventos e
- \* desenvolver aplicações AJAX.

# Vantagens da utilização de jQuery

## Separação entre o Javascript e o HTML

Ao invés de usar atributos HTML para identificar as funções para manipulação de eventos, o jQuery lida com eventos puramente em JavaScript.

Deste modo, ps marcadores/elementos HTML e o código Javascript são completamente separados.

Em Javascript:

```
<form id="target" action="http://192.168.160.36/FormEcho.aspx">
  <input type="submit" value="Enviar" class="btn btn-default" onclick="return validateForm()" />
</form>
```

```
function validateForm() {
  // TODO ...
}
```

Em jQuery

```
<form id="target" action="http://192.168.160.36/FormEcho.aspx">
  <input type="submit" value="Enviar" class="btn btn-default" />
</form>
```

```
$("#target").submit(function (event) {
  // TODO ...
});
```

# Vantagens da utilização de jQuery

## Elimina incompatibilidades entre navegadores:

Os motores de Javascript dos diferentes navegadores diferem ligeiramente, de modo que o código Javascript que funciona para um navegador pode não funcionar em outro.

O jQuery lida com todas essas inconsistências entre browsers e fornece uma interface consistente que funciona nos diferentes navegadores.

## Extensível:

O jQuery é muito extensível – através a adição de novas livrarias ao projeto.

Novos eventos, elementos e métodos podem ser facilmente adicionados e depois reutilizados como um plugin.

# Desvantagens da utilização de jQuery

A desvantagem mais crítica do jQuery é ser uma biblioteca grande (~88k a versão min; ~281Kb, a versão normal) para importar e, dessa biblioteca, muitas vezes utilizamos apenas uma pequena parte das funcionalidades disponibilizadas.

Outra desvantagem é que a abstração do jQuery esconde as partes complexas do JavaScript, dificultando a aprendizagem do JavaScript.

Performance - o javascript puro é mais rápido a aceder ao DOM

Testing in Chrome 32.0.1700.107 32-bit on Windows Server 2008 R2 / 7 64-bit		
Test		Ops/sec
jQuery ID Selector	<code>var \$el = \$('#hello');</code>	1,813,016 ±1.04% 91% slower
JavaScript ID Selector	<code>var \$el = document.querySelector('#hello');</code>	10,126,325 ±0.40% 48% slower
jQuery Class Selector	<code>var \$el = \$('.bye');</code>	527,960 ±3.48% 97% slower
JavaScript Class Selector	<code>var \$el = document.querySelector('.bye');</code>	1,623,869 ±0.29% 92% slower
GetElementById	<code>var \$el = document.getElementById('hello');</code>	19,624,531 ±0.31% fastest

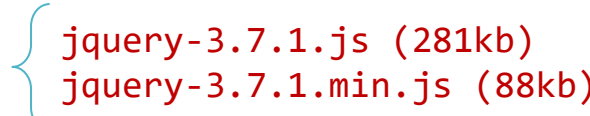
# Utilização da biblioteca jQuery

A biblioteca jQuery é um único ficheiro JavaScript, contendo todas as suas funcionalidades: acesso aos elementos DOM, eventos, efeitos e funções comuns do Ajax.

Esse ficheiro pode ser incluído numa página web através da ligação a uma cópia local, desde que previamente descarregada, ou interligando-o a uma das muitas cópias disponíveis a partir de servidores públicos.

## Ficheiro local:

```
<script src="Scripts/jquery-3.7.1.min.js"></script>
```



jquery-3.7.1.js (281kb)  
jquery-3.7.1.min.js (88kb)

## Ficheiro remoto (CDN):

```
<script src="https://code.jquery.com/jquery-3.7.1.min.js"></script>  
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
```

Lista de livrarias suportadas pela google:

- <https://developers.google.com/speed/libraries/>

# Sintaxe jQuery

A sintaxe jQuery foi feita a pensar especialmente na seleção de elemento(s) HTML e na execução de alguma ação sobre o(s) mesmo(s).

A sintaxe básica é: `$("selector").action()`

Um sinal **\$** para definir/aceder à biblioteca jQuery

Um (**seletor**) para "consultar/encontrar" elementos HTML no documento

Uma **ação** jQuery () a ser executada no(s) elemento(s)



# Seletores jQuery (1)

Os seletores jQuery são usados para “encontrar” (ou selecionar) elementos HTML baseados no nome, id, classes, tipos, atributos, valores de atributos e muito mais.

Exemplo:

```
<p>Este é um parágrafo.</p>  
<button type="button"><i class="fa fa-edit"></i></button>
```

```
<script>  
$(document).ready(function(){  
    $("button").click(function(){  
        $("p").hide();  
    });  
});  
</script>
```

Neste exemplo, o seletor usado é apenas o nome do marcador html (button ou p). Quando qualquer button for carregado, todos os elementos html do tipo <p>...</p> serão escondidos .hide().

## Seletores jQuery (2)

O seletor jQuery **#id** usa o atributo id de um elemento HTML para o localizar no documento.

Como sabemos, um id deve ser único dentro de uma página; assim, este seletor é utilizado para encontrar um elemento único.

Exemplo:

```
<div id="errorMessage">O nome deve ter pelo menos três letras.</div>  
<button type="button"><i class="fa fa-edit"></i></button>
```

```
<script>  
  $(document).ready(function () {  
    $("button").click(function () {  
      $("#errorMessage").show();  
    });  
  });  
</script>
```

## Seletores jQuery (3)

O seletor de classe jQuery localiza ocorrências dessa classe específica.

Para encontrar elementos com uma classe específica, escreva `."` + nome da classe.

Exemplo:

```
<div class="bannerTop"></div>
```

```
<script>
  $(document).ready(function () {
    $(".bannerTop").css({"background-color": "yellow", "font-size": "200%"});
  });
</script>
```

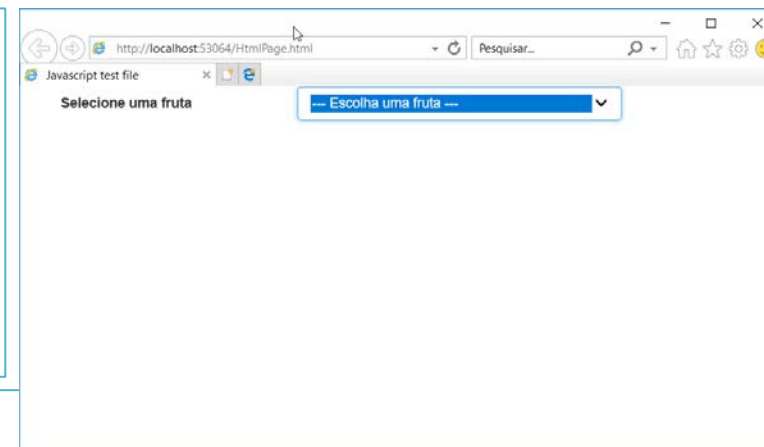
Neste caso, depois do documento estar carregado no browser, são alteradas propriedades CSS da classe bannerTop. Isso é feito utilizando o método/ação `.css()`

## Seletores jQuery (4)

O seletor `option:selected` permite saber qual a opção selecionada numa caixa de seleção

```
<label for="fruta" class="col-md-4 form-control-static">Selecione uma fruta</label>
<div class="col-md-6">
  <select id="fruta" class="form-control">
    <option>--- Escolha uma fruta ---</option>
    <option value="1">Banana</option>
    <option value="2">Maçã</option>
    <option value="3">Pera</option>
  </select>
</div>
```

```
$(document).ready(function () {
  $("#fruta").change(function () {
    var retVal = $("#fruta option:selected").val() + " - " + $("#fruta option:selected").text();
    alert(retVal)
  });
});
```



Este evento (`.change()`) é ativado sempre que o utilizador altera a seleção.

- O valor selecionado é recolhido através do método `.val()`; (equivalente ao `.value` em javascript)
- O texto da opção é recolhido através do método `.text()` ou `html()` (equivalente aos `.innerText` e `.innerHTML` em javascript)

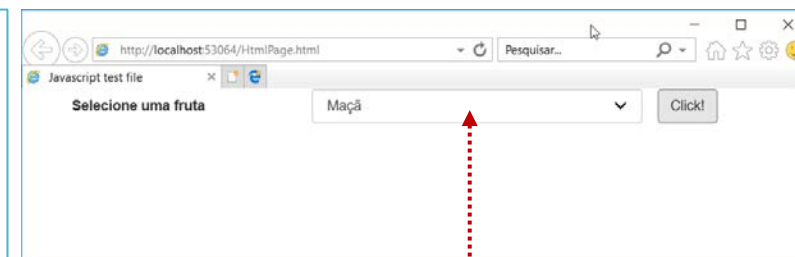
## Seletores jQuery (5)

Nos exemplos anteriores os seletores foram quase sempre utilizados para verificar os valores dos diversos elementos html (get).

O jQuery pode também ser usado para atuar/alterar os elementos (set).

Exemplo:

```
<label for="fruta" class="col-sm-4 form-control-static">Selecione uma fruta</label>
<div class="col-sm-6">
  <select id="fruta" class="form-control">
    <option>--- Escolha uma fruta ---</option>
    <option value="1">Banana</option>
    <option value="2">Maçã</option>
    <option value="3">Pera</option>
  </select>
</div>
<div class="col-sm-2">
  <button type="button" id="myButton" class="btn btn-default">Click!</button>
</div>
```



```
$(document).ready(function () {
  $("#myButton").click(function () {
    $("#fruta").val(2);
  });
});
```

# Outros seletores jQuery

Sintaxe	Descrição
<code>\$("*")</code>	Seleciona todos os elementos do documento
<code>\$(this)</code>	Seleciona o elemento html que está a ser manipulado. Nota: this sem aspas(!!!)
<code>\$("p.intro")</code>	Seleciona todos os elementos <code>&lt;p&gt;</code> com <code>class="intro"</code>
<code>\$("p:first")</code>	Seleciona o primeiro elemento <code>&lt;p&gt;</code> do documento
<code>\$("ul li:first")</code>	Seleciona o primeiro element <code>&lt;li&gt;</code> do primeiro elemento <code>&lt;ul&gt;</code>
<code>\$("ul li:first-child")</code>	Seleciona o primeiro elemento <code>&lt;li&gt;</code> de todos os elementos <code>&lt;ul&gt;</code>
<code>\$("[href]")</code>	Seleciona todos os elementos que possuam o atributo <code>"href"</code>
<code>\$("a[target]='_blank']")</code>	Seleciona todos os elementos <code>&lt;a&gt;</code> com o atributo <code>target</code> igual a <code>"_blank"</code>
<code>\$("a[target!='_blank']")</code>	Seleciona todos os elementos <code>&lt;a&gt;</code> com o atributo <code>target</code> diferente de <code>"_blank"</code>
<code>\$(":button")</code>	Seleciona todos os elementos <code>&lt;button&gt;</code> e/ou <code>&lt;input&gt;</code> com <code>type="button"</code>
<code>\$("tr:even")</code>	Seleciona todas as linhas pares dos elementos <code>&lt;tr&gt;</code>
<code>\$("tr:odd")</code>	Seleciona todas as linhas ímpares dos elementos <code>&lt;tr&gt;</code>

## O evento `$(document).ready()` (1)

Como pudémos observar nos exemplos apresentados nos slides anteriores, todos os métodos de jQuery estão sempre dentro de um evento

`$(document).ready()`

Isso evita que qualquer código jQuery seja executado antes do documento carregar completamente (ou seja, só depois do documento estar pronto - **is ready**).

Também pode ser utilizada outra notação equivalente:

`$.ready()` ou `$("document").ready()`

Isso evita que o script possa ser executado, por exemplo, antes de a biblioteca jQuery estar também ela carregada

## O evento `$(document).ready()` (2)

É uma boa prática esperar que o documento seja totalmente carregado e esteja pronto antes de atuar nele.

Exemplos de ações que podem falhar se os métodos forem executados antes que o documento seja totalmente carregado:

*Tentar ocultar um elemento que ainda não foi criado;*

*Tentar obter o tamanho de uma imagem que ainda não está carregada; etc..*

Se ainda se lembram, em Javascript, para evitar estes problemas, carregava-se o `<script>` apenas no final do documento html, evitando que qualquer elemento fosse referido antes de ser criado.

Utilizando a livreria jQuery, escrevendo o código dentro da sequência:

```
$(document).ready(function () {  
    /* Código aqui... */  
});
```

o `<script>` tanto pode ficar no início do documento html como no final.



# Noções de programação por objetos (outra vez...)

Um objeto, qualquer que seja – um carro, um telemóvel ou um elemento html – possui um conjunto de:

Propriedades

Métodos,

Eventos

- Quais as propriedades de um carro?
- Quais os métodos de um carro?
- Quais os eventos de um carro?

- Quais as propriedades de um elemento html?
- Quais os métodos de um elemento html?
- Quais os eventos de um elemento html?

# Métodos para manipulação do DOM com jQuery - GET

Os três métodos para recolha de informação de elementos DOM são:

**text()** - Retorna o conteúdo de texto dos elementos selecionados;

**html()** - Retorna o conteúdo dos elementos selecionados;

**val()** - Retorna o valor dos campos de um formulário (<input>).

.oOo.

O método **attr()** é usado para obter valores de atributos. O exemplo a seguir demonstra como obter o valor do atributo href em um link:

```
$(document).ready(function(){
    $("button").click(function () {
        alert($("#nextPage").attr("href"));
    });
});
```

# Método para manipulação do DOM com jQuery - SET

Os métodos para atribuição de valores a elementos DOM são os mesmos: text(), html(), val() e attr() mas a sintaxe é distinta.

Exemplos:

```
$(document).ready(function(){  
    $("button").click(function () {  
        $("title").text("This is a text");  
        $("#pageTitle").html("This is a <strong>text</strong>");  
        $("#name").val("Dolly Duck");  
        $("#nextPage").attr("href", "http://www.ua.pt");  
    });  
});
```

# JQuery – Obter<sub>(get)</sub> e definir<sub>(set)</sub> classes CSS

jQuery tem vários métodos para manipulação CSS. Examinaremos os seguintes métodos:

**addClass()** - Adiciona uma ou mais classes aos elementos selecionados;

**removeClass()** - Remove uma ou mais classes dos elementos selecionados;

**toggleClass()** - Alterna entre adicionar / remover classes dos elementos selecionados;

.oOo.

**css()** - Define ou retorna o atributo com todos os estilos.

```
<script>
  $(document).ready(function () {
    $(".bannerTop").css({"background-color": "yellow", "font-size": "200%"});
  });
</script>
```

<----- Isto é notação JSON ----->

# Eventos jQuery

O jQuery para além de permitir a alteração das propriedades dos elementos html – por tipo, id ou por className – consegue também capturar eventos numa página HTML.

Todas as ações que um visitante realiza numa página da web geram **eventos** que sinalizam essas ações.

Exemplos de eventos numa página web:

- \* Mover o rato sobre um elemento;
- \* Selecionar um botão de opção;
- \* Clicar num elemento.

# Eventos jQuery

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

# Sintaxe jQuery para métodos associados com eventos

Em jQuery, a maioria dos eventos DOM têm um método jQuery equivalente. Para atribuir um evento de clique associado a todos os parágrafos de uma página, por exemplo, pode fazer o seguinte:

```
$("#p").click();
```

O passo seguinte deve definir “o que acontece quando o evento é disparado”. Assim, deve ser indicada uma **function** para o evento.

```
<script>
$(document).ready(function(){
    $("#p").click(function(){
        // o que fazer quando o rato for carregado
        // em todos os elementos do tipo “p” deve ser
        // programado aqui!!
    });
});
</script>
```

Neste exemplo gerimos de dois eventos:

- o **document.ready()**; e ainda
- o **click()** dos elementos do tipo `<p>...</p>`

Teste do Modelo de dados - Livro em papel / Descrição

Número de inscrição

Folhas

Descrição da propriedade

Declarante

Preencha com o nome da pessoa

Natureza do prédio

--- Selecione uma das opções ---

Área Rústica

Área Urbana

Valor

Confrontação (Norte)

Confrontação (Sul)

Confrontação (Este)

Confrontação (Oeste)

Causa

Estado do registo

--- Selecione uma das opções ---

Data de registo

dd/mm/aaaa

Data do direito

dd/mm/aaaa

Documentação apresentada

Notas

Teste do Modelo de dados - Livro em papel / Descrição

Número de inscrição

→ O campo Número de inscrição é de preenchimento obrigatório.

Folhas

→ O campo Folhas é de preenchimento obrigatório.

Descrição da propriedade

→ O campo Descrição da propriedade é de preenchimento obrigatório.

Declarante

Preencha com o nome da pessoa

Natureza do prédio

--- Selecione uma das opções ---

→ O campo Natureza do prédio é de preenchimento obrigatório.

Área Rústica

Área Urbana

Valor

→ O campo Valor é de preenchimento obrigatório.

Confrontação (Norte)

→ O campo Confrontação (Norte) é de preenchimento obrigatório.

Confrontação (Sul)

→ O campo Confrontação (Sul) é de preenchimento obrigatório.

Confrontação (Este)

→ O campo Confrontação (Este) é de preenchimento obrigatório.

Confrontação (Oeste)

→ O campo Confrontação (Oeste) é de preenchimento obrigatório.

Causa

Estado do registo

--- Selecione uma das opções ---

→ O campo Estado do registo é de preenchimento obrigatório.

Data de registo

dd/mm/aaaa

→ O campo Data de registo é de preenchimento obrigatório.

Data do direito

dd/mm/aaaa

→ O campo Data do direito é de preenchimento obrigatório.

Documentação apresentada

Notas

→ O campo Notas é de preenchimento obrigatório.





```
var errors = $("span.field-validation-error").length;
console.log("errors =", errors);
if (errors > 0) {
    $("#submit_form").html("<i class='fa fa-save'></i> <span class='position-absolute top-0 start-100 translate-middle badge rounded-pill bg-danger' title='O Formulário tem " + errors + " Erros'>" + errors + "</span>");
}
else {
    $("#submit_form").html("<i class='fa fa-save'></i>");
    if (buttonId.indexOf("submit_form") == 0) {
        var JSONdata = generateJSON();
        //-- Envia dados para o servidor...
    }
}
```

# JSON

JavaScript Object Notation

# JSON - JavaScript Object Notation.

JSON é um formato de troca/intercâmbio de dados simples que é independente da linguagem de programação utilizada.

É uma linguagem auto-descritiva e, por isso, fácil de entender.

Usa a sintaxe JavaScript, mas o formato JSON é somente texto.

O texto pode ser lido e usado como formato de dados por qualquer linguagem de programação.

Exemplo:

```
<html>
<body>
  <h2>Manipulação JSON em JavaScript</h2>
  <p id="demo"></p>
  <script>
    var text = '{"name":"Zé Maria Pincel","address":"Rua 8 de Maio, 5","phone":"351 123456789"}';
    var obj = JSON.parse(text);
    document.getElementById("demo").innerHTML =
      obj.name + "<br>" +
      obj.address + "<br>" +
      obj.phone;
  </script>
</body>
</html>
```

## JSON Netflix (num ano passado) / Para este ano: será a Paris 2024

/api/Titles

```
{
  "TotalTitles": 6234,
  "TotalPages": 125,
  "CurrentPage": 125,
  "PageSize": 50,
  "HasPrevious": true,
  "HasNext": false,
  "Titles": [
    {
      "Id": 80987906,
      "Name": "Yu-Gi-Oh! Arc-V"
    },
    {
      "Id": 80988834,
      "Name": "Yucatán"
    },
    {
      "Id": 80217769,
      "Name": "Yummy Mummies"
    },
    {
      "Id": 80126991,
      "Name": "Yunus Emre"
    },
    {
      "Id": 70001564,
      "Name": "Yuva"
    },
    {
      "Id": 80008434,
      "Name": "Z Nation"
    }
  ]
}
```

/api/Actors

```
{
  "TotalActors": 27391,
  "TotalPages": 548,
  "CurrentPage": 1,
  "PageSize": 50,
  "HasPrevious": false,
  "HasNext": true,
  "Actors": [
    {
      "Id": 14604,
      "Name": "2 Chainz",
      "Titles": 1
    },
    {
      "Id": 15728,
      "Name": "4Minute",
      "Titles": 1
    },
    {
      "Id": 3581,
      "Name": "50 Cent",
      "Titles": 3
    },
    {
      "Id": 14607,
      "Name": "A Boogie Wit tha Hoodie",
      "Titles": 1
    },
    {
      "Id": 19243,
      "Name": "A-ra Go",
      "Titles": 1
    }
  ]
}
```

/api/Categories

```
{
  "TotalCategories": 42,
  "TotalPages": 1,
  "CurrentPage": 1,
  "PageSize": 50,
  "HasPrevious": false,
  "HasNext": false,
  "Categories": [
    {
      "Id": 1,
      "Name": "Action & Adventure",
      "Titles": 597
    },
    {
      "Id": 19,
      "Name": "Anime Features",
      "Titles": 45
    },
    {
      "Id": 20,
      "Name": "Anime Series",
      "Titles": 117
    },
    {
      "Id": 31,
      "Name": "British TV Shows",
      "Titles": 210
    },
    {
      "Id": 8,
      "Name": "Children & Family Movies",
      "Titles": 1
    }
  ]
}
```

# AJAX

Asynchronous JavaScript and XML



# AJAX (Asynchronous JavaScript and XML)

AJAX é uma funcionalidade que permite que páginas HTML troquem dados com um servidor para atualizar apenas partes dessa página mas sem ser necessário recarregar toda a página.

Através da utilização de AJAX, o carregamento de dados é feito em segundo plano e o resultado exibido na página da Web, sem recarregar a página.

Exemplos de alguma das aplicações que utilizam AJAX: Gmail, Google Maps, Facebook, Instagram, ...

Como sabem, os dados não são todos carregados ao mesmo tempo. Sempre que vamos deslizando na página, novos dados vão sendo carregados dinamicamente...

# Exemplo de uma chamada AJAX em jQuery

```
var data = "abc";
$.ajax({
  type: "GET",
  url: "http://somewhere/somepage/somedetails",
  data: {
    "data": data
  },
  dataType: "json",

  success: function (datas, textStatus, jqXHR) {
    //if received a response from the server
  },

  error: function (jqXHR, textStatus, errorThrown) {
    //if there was no response from the server
  },

  beforeSend: function (jqXHR, settings) {
    //capture the request before it was sent to server (in send calls)
  },

  complete: function (jqXHR, textStatus) {
    //this is called after the response or error functions are finished
    //so that we can take some action
  }
});
```

# Exemplo

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
  <button>Get External Content</button>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        $.ajax({
          url: "demo_test.txt",
          success: function (result) {
            $("#div1").html(result);
          }
        });
      });
    });
  </script>
</body>
</html>
```

**Let jQuery AJAX Change This Text**

Get External Content

**jQuery and AJAX is FUN!**

This is some text in a paragraph.

Get External Content



# Métodos AJAX em jQuery

Os métodos `$.get()` e `$.post()`

`$.get(URL, callback);`

```
$("#button").click(function () {  
    $.get("getPageAddress", function (data, status) {  
        alert("Data: " + data + "\nStatus: " + status);  
    });  
});
```

`$.post(URL, data, callback);`

```
$("#button").click(function () {  
    $.post("postPageAddress",  
        {  
            name: "Biden, Joe",  
            city: "Washington"  
        },  
        function (data, status) {  
            alert("Data: " + data + "\nStatus: " + status);  
        });  
});
```

# Manipulação do DOM + AJAX

(jQuery + Asynchronous JavaScript And XML + JSON)

Exemplo: Elaborar um formulário onde o utilizador pode ir buscar, remotamente e sem voltar a carregar o documento as condições meteorológicas de uma qualquer cidade no mundo!

City Name:	Aveiro / PT
Coordinates:	Lon (°): -8.65 / Lat (°): 40.64
weather:	Sky is Clear
temp:	290.334°K / 17.184000000000026°C
pressure:	1007.32
Dados recolhidos	<pre>{   "coord": {     "lon": -8.65,     "lat": 40.64   },   "weather": [     {       "id": 800,       "main": "Clear",       "description": "Sky is Clear",       "icon": "01n"     }   ],   "base": "cmc stations",   "main": {     "temp": 290.334,     "pressure": 1007.32,     "humidity": 77,     "temp_min": 290.334,     "temp_max": 290.334,     "sea_level": 1040.01,     "grnd_level": 1007.32,     "speed": 0.92,     "deg": 204.501   },   "clouds": {     "all": 0   },   "dt": 1447010744,   "sys": {     "message": 0.0051,     "country": "PT",     "sunrise": 1446966804,     "sunset": 1447003369   },   "id": 2742611,   "name": "Aveiro",   "cod": 200 }</pre>



# Quem fornece a informação do tempo?

<http://api.openweathermap.org/>

Mensagem de sucesso “beautified”

Erro

```
{"cod":"404","message":"Error: Not found city"}
```

Sucesso

```
{"coord":{"lon":-8.61,"lat":41.15},"weather":[{"id":802,"main":"Clouds","description":"scattered clouds","icon":"03n"}],"base":"stations","main":{"temp":301.15,"pressure":1018,"humidity":58,"temp_min":301.15,"temp_max":301.15},"visibility":16093,"wind":{"speed":7.2,"deg":50},"clouds":{"all":40},"dt":1447008780,"sys":{"type":1,"id":819,"message":0.03,"country":"PT","sunrise":1446966858,"sunset":1447003295},"id":2735943,"name":"Porto","cod":200}
```

```
{
  "coord": {
    "lon": -8.61,
    "lat": 41.15
  },
  "weather": [
    {
      "id": 803,
      "main": "Clouds",
      "description": "broken clouds",
      "icon": "04d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 284.62,
    "pressure": 1020,
    "humidity": 81,
    "temp_min": 283.71,
    "temp_max": 285.15
  },
  "visibility": 10000,
  "wind": {
    "speed": 1.5,
    "deg": 160
  },
  "clouds": {
    "all": 75
  },
  "dt": 1573556719,
  "sys": {
    "type": 1,
    "id": 6900,
    "country": "PT",
    "sunrise": 1573543096,
    "sunset": 1573579138
  },
  "timezone": 0,
  "id": 2735943,
  "name": "Porto",
  "cod": 200
}
```

**{JSON}**  
JavaScript Object Notation

<http://api.openweathermap.org/data/2.5/weather?q=porto,pt&appid=b2b1df463182c3cca5276e9d3267cc95> (#valid@12nov19)

# O código da interface

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>jQuery Weather Test</title>
  <link href="https://fonts.googleapis.com/css?family=Roboto:300" rel="stylesheet" type="text/css">
  <link href="../../Content/bootstrap.min.css" rel="stylesheet" />
</head>
<body>
  <div class="container">
    <select id="citySelector" class="form-select">
      <option value="">Select a city name</option>
      <option value="Aveiro, PT">Aveiro</option>
      <option value="Porto, PT">Porto</option>
      <option value="Paris, FR">Paris</option>
      <option value="London, UK">Londres</option>
      <option value="New York, USA">Nova Iorque</option>
    </select>
    <table class="table table-striped d-none">
      <tr><td class="col-xs-2">City Name:</td><td class="col-xs-10" id="cityName"></td></tr>
      <tr><td class="col-xs-2">Coordinates:</td><td class="col-xs-10" id="coordinates"></td></tr>
      <tr><td class="col-xs-2">weather:</td><td class="col-xs-10" id="weather"></td></tr>
      <tr><td class="col-xs-2">temp:</td><td class="col-xs-10" id="temp"></td></tr>
      <tr><td class="col-xs-2">pressure:</td><td class="col-xs-10" id="pressure"></td></tr>
      <tr><td class="col-xs-2">Dados recolhidos</td>
        <td class="col-xs-10" ><pre id="allData"></pre></td>
      </tr>
    </table>
  </div>
```

# O código de manipulação da informação

```
<script src="../../Scripts/jquery-3.7.1.min.js"></script>
<script>
$(document).ready(function () {
    $("#citySelector").change(function () {
        $.ajax({
            url: "http://api.openweathermap.org/data/2.5/weather",
            data: {
                q: $("#citySelector").val(),
                APPID: 'b2b1df463182c3cca5276e9d3267cc95'
            },
            success: function (data) {
                if (data.name){
                    $('table').removeClass('d-none');
                    $("#cityName").html(data.name + ' / ' + data.sys.country);
                    $("#coordinates").html('Lon (°): ' + data.coord.lon + ' / Lat (°):' + data.coord.lat);
                    $("#weather").html(data.weather[0].description);
                    $("#temp").html(data.main.temp.toString() + '°K / ' + (data.main.temp - 273.15).toString() + '°C');
                    $("#pressure").html(data.main.pressure);
                    $("#allData").html(JSON.stringify(data, null, 4).replace(/\n/g, "<br>"));
                }
                else {
                    $('table').addClass('d-none');
                    alert(data.message);
                }
            },
            error: function () {
                $('table').addClass('d-none');
                alert('Erro!');
            }
        });
    });
});
</script>
</body>
```

# jQueryUI

# jQueryUI – jQuery User Interface

jQuery UI é uma coleção de widgets de interface gráfica, efeitos visuais animados e temas implementados com jQuery, CSS's e HTML

*- um widget é um pequeno aplicativo com funcionalidade limitada que pode ser instalado e executado dentro de uma página web*

Esta livraria assenta sobre a livraria jQuery e possui muitas funcionalidades que são também cobertas pelo Bootstrap.

Assim, esta livraria será abordada de modo genérico e identificadas apenas algumas funcionalidades que não cobertas pelo Bootstrap.

# Utilização da biblioteca jQuery UI

## Utilização:

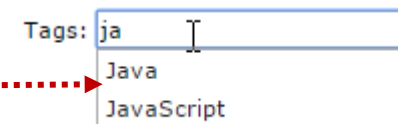
```
<link rel="stylesheet" href=" https://code.jquery.com/ui/1.12.1/themes/smoothness/jquery-ui.css">  
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
```



# jQuery UI Widgets

**Acordeão** – grupo de contentores organizados na forma de um acordeão

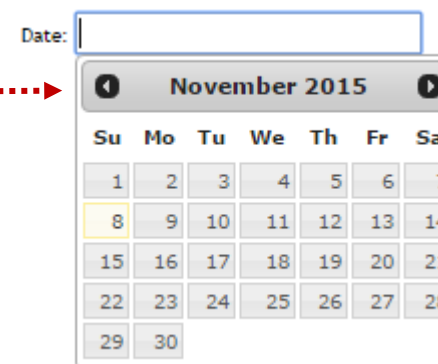
**Autocomplete** - caixas que permitem o preenchimento automático com base no que o utilizador digita



**Button** - botão com apresentação melhorada.

Permite que botões rádio e caixas de seleção sejam convertidos em botões

**Datepicker** – componente com calendário para recolha de campos com datas



**Dialog** - caixas de diálogo colocadas em cima de outros conteúdos

**Menu** – componente que permite mostrar e gerir os elementos de um menu

**Progressbar** - barras de progresso – animandas, ou não

**Slider** – barras de arrastamento totalmente personalizáveis

**Spinner** – gere o valor de um número com setas

**Tabs** - manipulação interface com tabuladores

**Tooltip** - Mostrar uma dica sobre um determinado conteúdo ou operação

# Exemplos DOM

Para uma lista completa de widgets jQuery UI, ver <http://jqueryui.com/widget/>

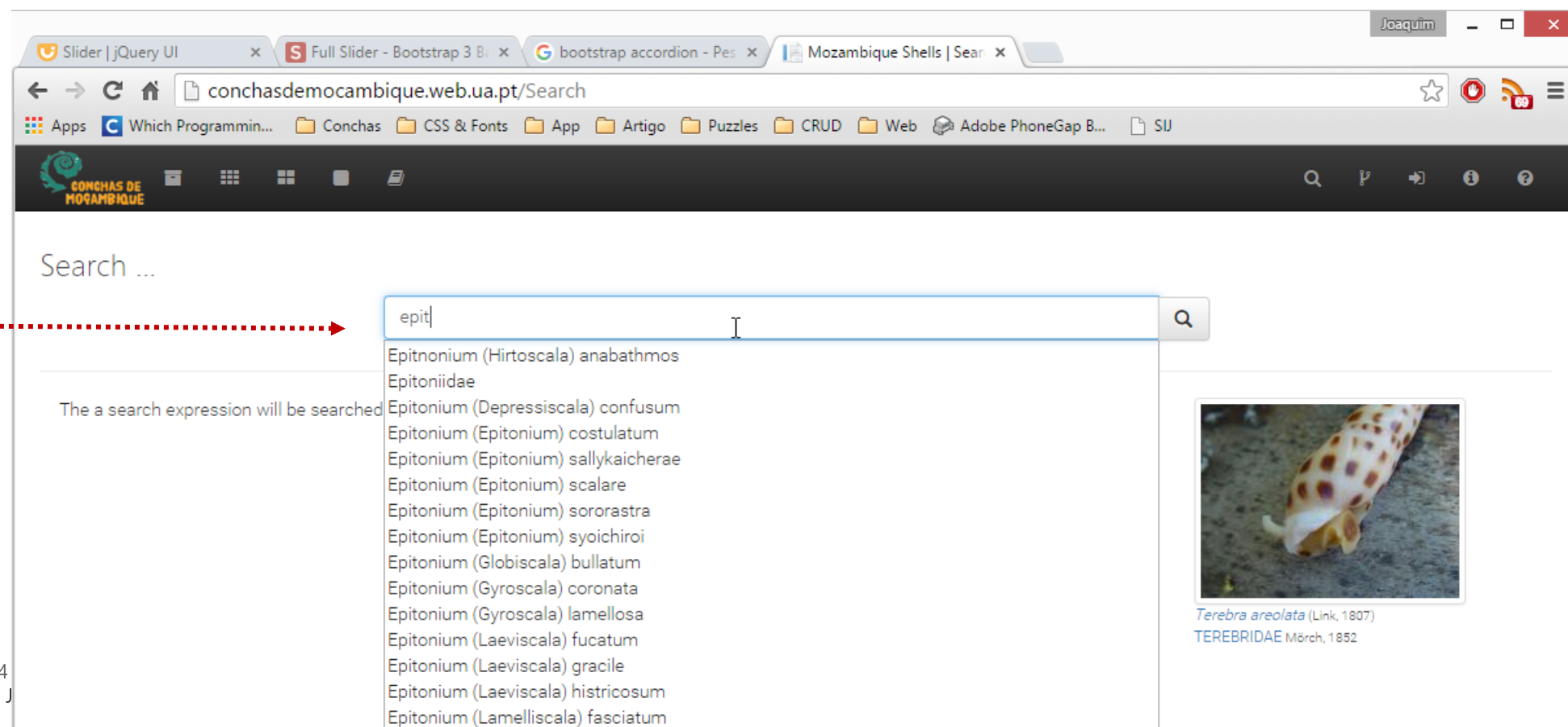
```
<div id="draggable" class="ui-widget-content">  
  <p>Drag me around</p>  
</div>
```

Esta propriedade pode ser "interessante" para aplicar às peças de xadrez da aula 2; isso faz com que as peças se possam mover dentro da página...

```
<script type="text/javascript">  
// Make #draggable draggable  
$(function () {  
  $("#draggable").draggable();  
});  
</script>
```

# Exemplo de autocomplete

Este exemplo foi retirado do site <http://conchasdemocambique.web.ua.pt> e permite ao utilizador procurar pelo nome de uma classe, família, subfamília ou espécie de conchas.



# O código

Nota:

Este exemplo não é repetível fora do contexto porque o webService que serve a pesquisa ([DynamicShellSearch.asmx/SearchData](#)) está, intencionalmente, protegido de modo a só poder ser utilizado pelo próprio site.



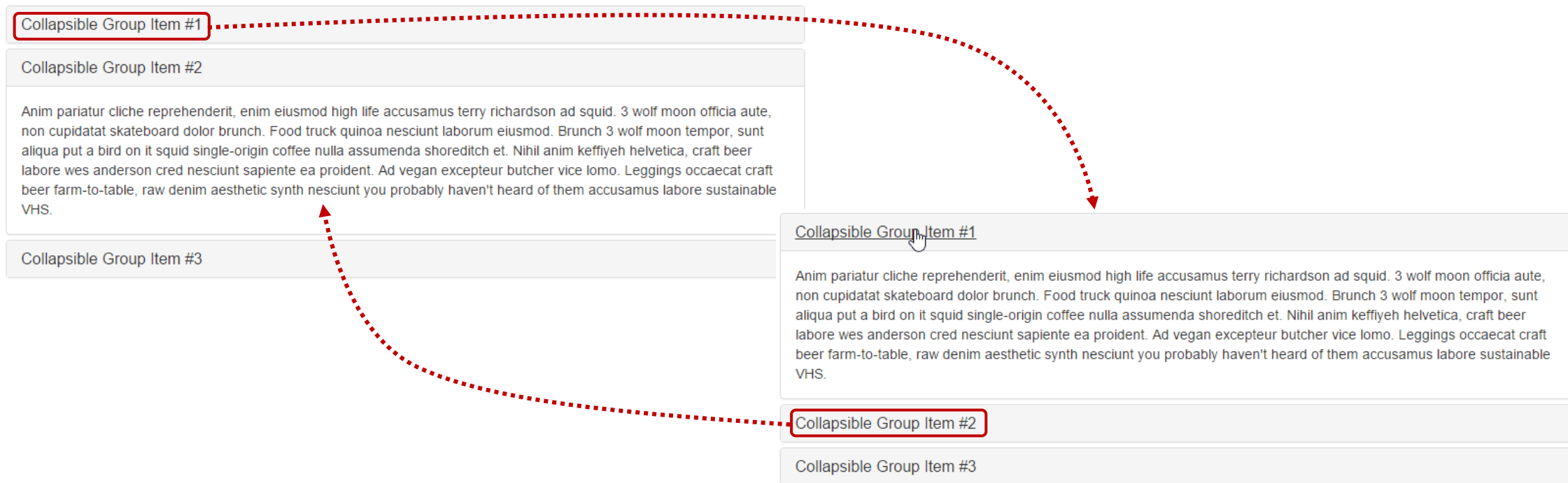
```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title></title>
  <link href="Content/bootstrap.min.css" rel="stylesheet" />
  <link href="Content/themes/ui-darkness/jquery.ui.base.css" rel="stylesheet" />
</head>
<body>
  <div class="container">
    <input class="form-control" ID="SearchText" placeholder="Search expression" />
  </div>
  <script src="Scripts/jquery-2.1.4.min.js"></script>
  <script src="Scripts/jquery-ui-1.11.4.min.js"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $("#SearchText").autocomplete({
        minLength: 4,
        source: function (request, response) {
          $.ajax({
            type: "POST",
            contentType: "application/json; charset=utf-8",
            url: "DynamicShellSearch.asmx/SearchData",
            data: "{ 'DName': '" + $('#SearchText').val() + '",",
            dataType: "json",
            success: function (data) {
              response(data.d);
            },
            error: function (result) {
              alert(result.statusText);
            }
          });
        }
      });
    });
  </script>
</body>
</html>
```

# Bootstrap vs jQueryUI

Muitas das funcionalidades jQueryUI são comuns à biblioteca Bootstrap

# Bootstrap Collapse $\approx$ jQueryUI Accordion

<https://getbootstrap.com/docs/5.0/components/collapse/>



# Bootstrap Buttons ≈ jQueryUI Button

<https://getbootstrap.com/docs/5.0/components/buttons/>

## EXAMPLE

Loading state

Copy

```
<button type="button" id="myButton" data-loading-text="Loading..." class="btn btn-primary"
autocomplete="off">
  Loading state
</button>

<script>
  $('#myButton').on('click', function () {
    var $btn = $(this).button('loading')
    // business logic...
    $btn.button('reset')
  })
</script>
```

Checkbox 1 (pre-checked)

Checkbox 2

Checkbox 3

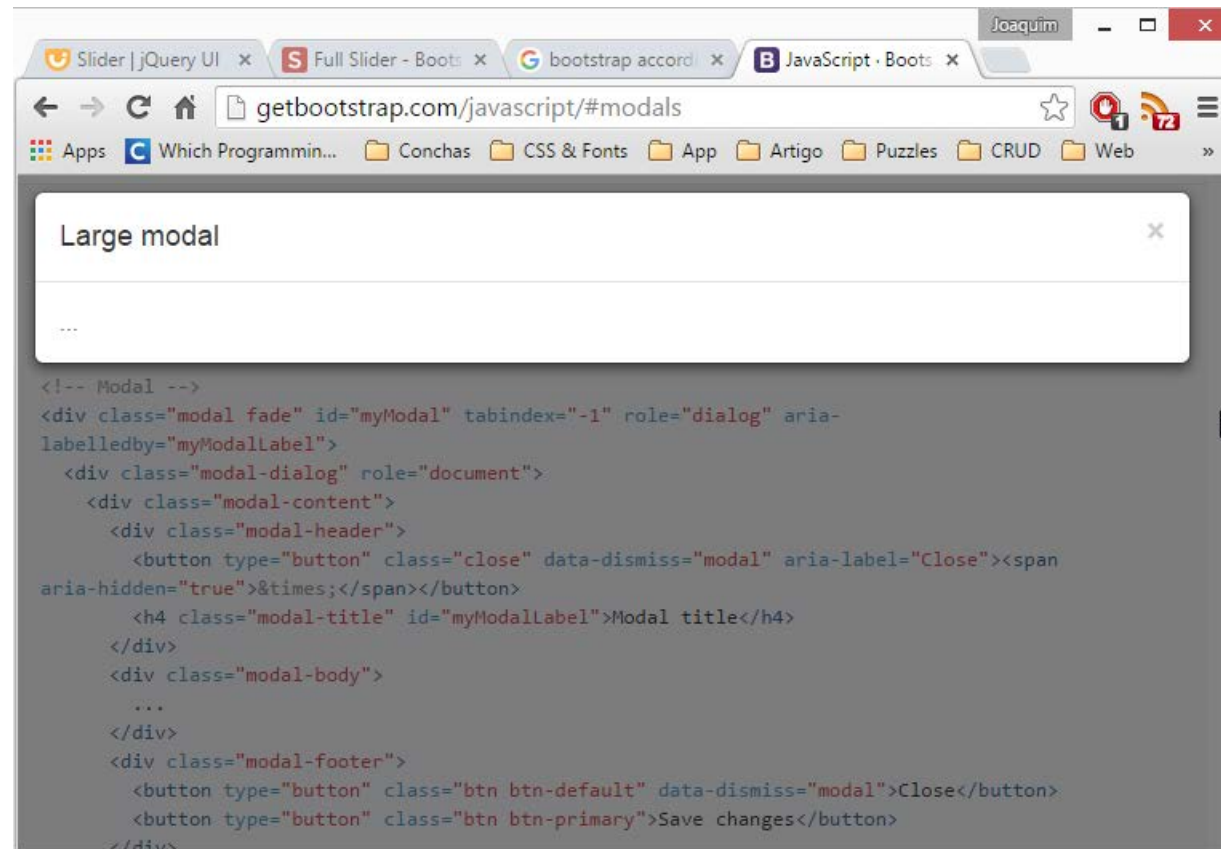
Checkbox 1 (pre-checked)

Checkbox 2

Checkbox 3

# Bootstrap Modal $\approx$ jQueryUI Dialog

<https://getbootstrap.com/docs/5.0/components/modal/>





# Bootstrap Tabs ≈ jQueryUI Tabs

<https://getbootstrap.com/docs/5.0/components/navs-tabs/>



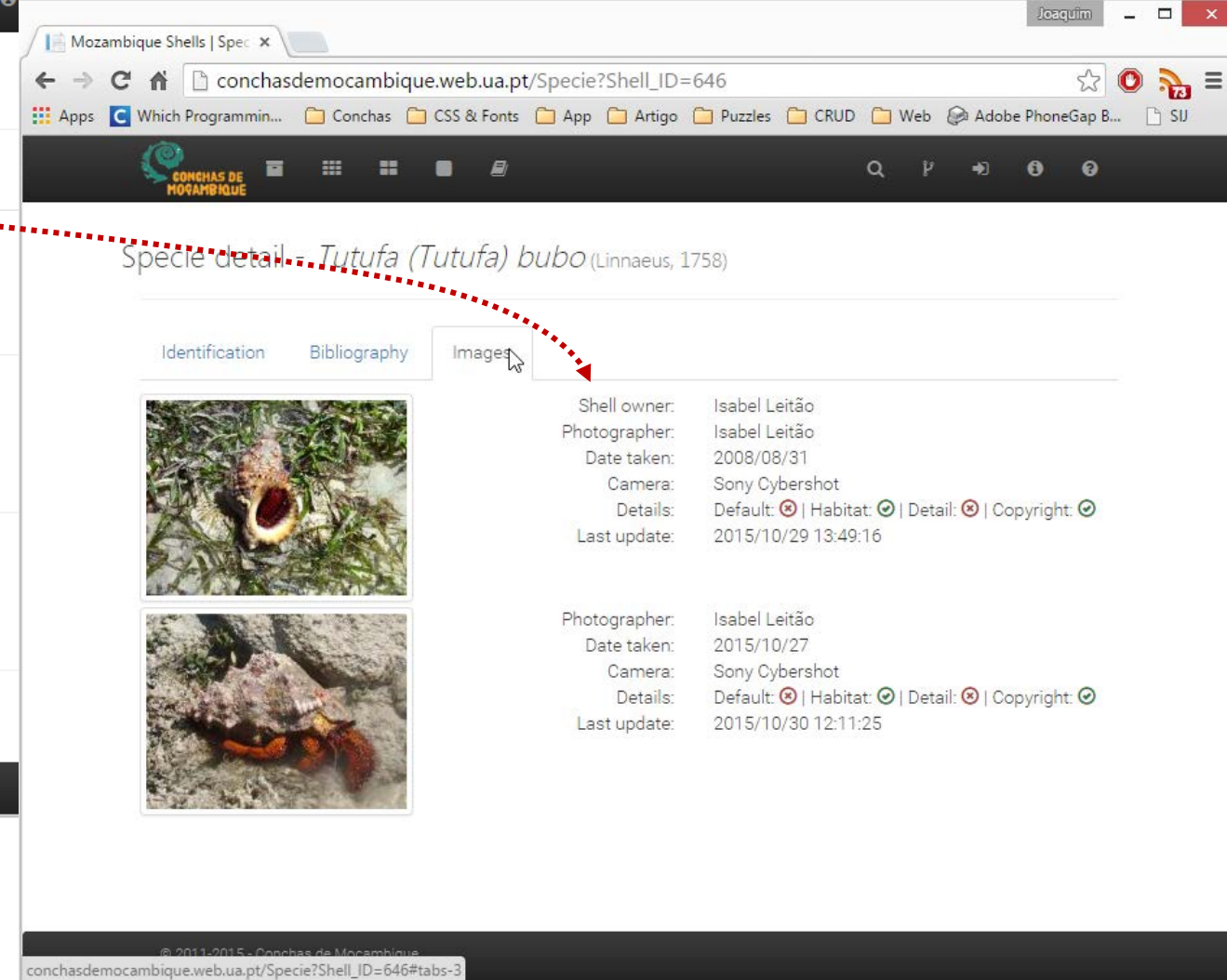
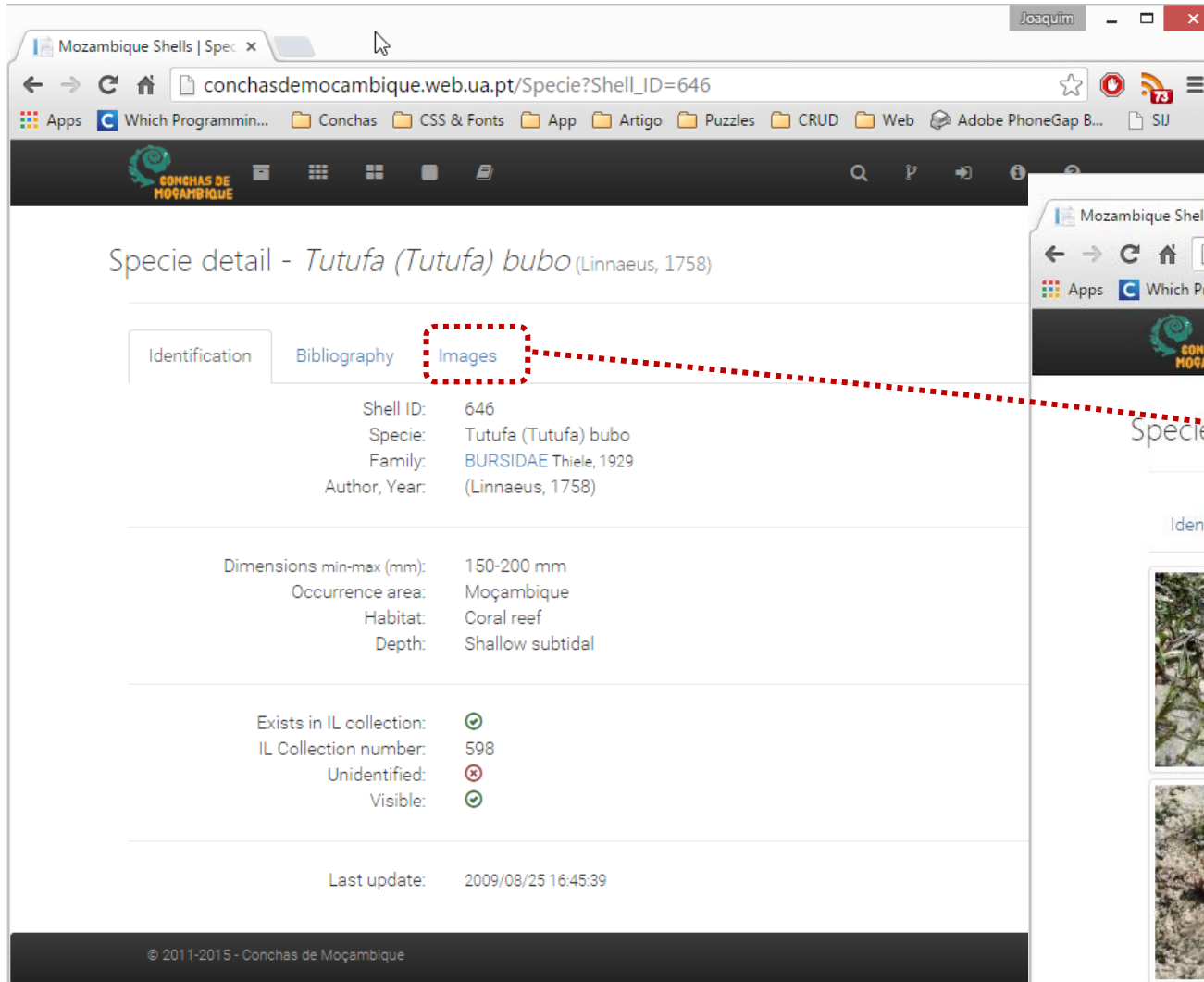
Raw denim you probably haven't heard of them jean shorts Austin. Nesciunt tofu stumptown aliqua, retro synth master cleanse. Mustache cliche tempor, williamsburg carles vegan helvetica. Reprehenderit butcher retro keffiyeh dreamcatcher synth. Cosby sweater eu banh mi, qui irure terry richardson ex squid. Aliquip placeat salvia cillum iphone. Seitan aliquip quis cardigan american apparel, butcher voluptate nisi qui.



Food truck fixie locavore, accusamus mcsweeney's marfa nulla single-origin coffee squid. Exercitation +1 labore velit, blog sartorial PBR leggings next level wes anderson artisan four loko farm-to-table craft beer twee. Qui photo booth letterpress, commodo enim craft beer mlkshk aliquip jean shorts ullamco ad vinyl cillum PBR. Homo nostrud organic, assumenda labore aesthetic magna delectus mollit. Keytar helvetica VHS salvia yr, vero magna velit sapiente labore stumptown. Vegan fanny pack odio cillum wes anderson 8-bit, sustainable jean shorts beard ut DIY ethical culpa terry richardson biodiesel. Art party scenester stumptown, tumblr butcher vero sint qui sapiente accusamus tattooed echo park.

A funcionalidade dos Tabs é muito semelhante à do Accordeon, ou seja, seccionar a quantidade de informação apresentada mas uma faz o seccionamento na vertical (accordeon) enquanto a outra faz na horizontal (Tabs)

# Exemplos:



# Bootstrap Tooltip/Popover ≈ jQueryUI Tooltips

<https://getbootstrap.com/docs/5.0/components/tooltips/>

<https://getbootstrap.com/docs/5.0/components/popovers/>

Tooltip on the left

Tooltip on the top

Tooltip on the bottom

Tooltip on the right

## EXAMPLE

Click to toggle popover

Popover title

And here's some amazing content. It's very engaging. Right?

## EXAMPLE

Dismissible popover

Dismissible popover

And here's some amazing content. It's very engaging. Right?

# Bootstrap Carousel ≈ jQueryUI Slider

<https://getbootstrap.com/docs/5.0/components/carousel/>

Funcionalidade parcial.

O Bootstrap Carousel já foi introduzido em aula anterior. Estamos agora em condições de perceber “como o programar”.

Exemplo:

```
<div id="carousel-example-generic" class="carousel slide" data-ride="carousel">
  <!-- Indicators -->
  <ol class="carousel-indicators">
    <li data-target="#carousel-example-generic" data-slide-to="0" class="active"></li>
    <li data-target="#carousel-example-generic" data-slide-to="1"></li>
    <li data-target="#carousel-example-generic" data-slide-to="2"></li>
    <li data-target="#carousel-example-generic" data-slide-to="3"></li>
  </ol>
  <!-- Carousel -->
  <div id="myCarousel" class="carousel slide">...</div>
</div>
<script>
  $( '.carousel' ).carousel({
    interval: 10000
  })
</script>
```

O elemento com a classe carousel, deve mudar de 10.000 em 10.000 milissegundos, ou seja de 10 em 10 segundos.

# Exemplo

