

jQuery fundamentals

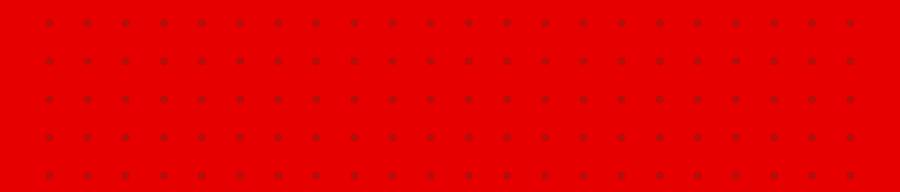


Pregatit de: Daniel Vișoiu Data: 04.05.2015

Agenda

- 1. Introduction
- 2. Basics
- 3. Selectors
- 4. Interacting with the DOM
- 5. Handling events
- 6. Ajax

Introduction



What is jQuery?

- Lightweight, "write less, do more", JavaScript library
- The purpose is to make it much easier to use JavaScript on your website
- Takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code
- Simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation

jQuery features

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

Why use jQuery?

- The price is right. The jQuery library is free
- It's light
- It works anywhere
- There's a low learning curve
- CSS3 compliant
- Highly documented: don't hesitate to use http://api.jquery.com/

Basics

How to use ·

- Simply include the jQuery script in your HTML page: <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
- The complete list of jQuery versions can be found at http://jquery.com/download/

The ready function

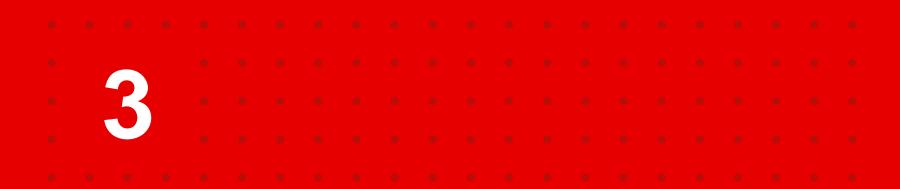
- Specifies a function to execute when the DOM is fully loaded
- Almost every HTML page that has jQuery uses it
- It has two forms:

```
$(document).ready(function () {
    // Handler for .ready() called.
});

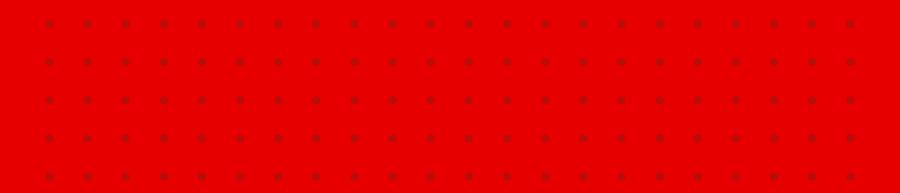
$(function () {
    // Handler for .ready() called.
});
```

jQuery API

- Don't hesitate to use https://api.jquery.com/
- Explains anything you need to know about jQuery and offers great examples
- Works like the namespace pattern: every functionality is grouped in the \$ global variable



jQuery Selectors



Selectors

- Allow page elements to be selected
- Single or multiple elements are supported
- Syntax:
 - \$(selectorExpression)
 - jQuery(selectorExpression)

• You can test selectors online; for ex http://selectortester.ru/4nny1d66

Teamnet

Tag selectors

\$('p') selects all elements

\$('a') selects all <a> elements

Selecting multiple tags

\$('p, a, span') selects all paragraphs, anchors, and span elements

Selection descendants

\$('ancestor descendant') selects all descendants of the ancestor

\$('table tr') selects all tr elements that are descendants of the table element

Element id selector

Use the # character to select elements by their id:

\$('#myId')

selects element

Element class selector

Use the . character to select elements by their class name:

```
$('.myClass')
selects  element
```

• To reference multiple tags use the , character to separate the class names:

```
$('.blueDiv, .redDiv')
selects all elements containing the class .blueDiv and .redDiv
```

You can combine this with element tag names as well:

```
$('a.myClass')
selects only <a> tags with the class="myClass"
```

Teamnet

Attribute value selector •

Use brackets [attribute] to select based on attribute name and/or value:\$('a[title]')

selects all <a> elements that have a title attribute

\$('a[title="Programming"]);

selects all anchor elements that have a "Programming" title attribute value

Input element selector

```
$('input')
selects all input elements
```

```
$('input[type="radio"])
targets all radio buttons on the page
```



CSS selectors

• Any valid CSS selector is a valid jQuery selector

Syntax	Description
\$("*")	Selects all elements
\$(this)	Selects the current HTML element
\$("p.intro")	Selects all elements with class="intro"
\$("p:first")	Selects the first element
\$("ul li:first")	Selects the first element of the first
\$("ul li:first-child")	Selects the first element of every
\$("[href]")	Selects all elements with an href attribute



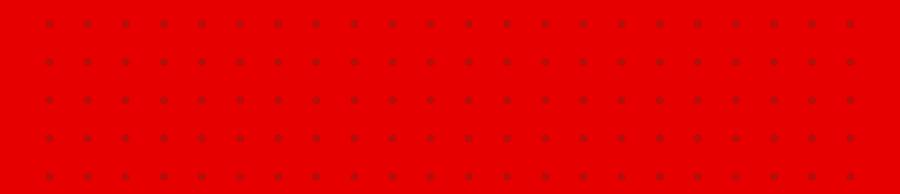
Exercise 1

Go to http://jsfiddle.net/jvbheqkp/

Select the following (using only jQuery ©):

- The node having the id "hotelsContainer"
- All the span tags that are children of the node with the id "third"
- All the nodes having the class "right"

Interacting with the DOM



Iterating through nodes

To iterate through jQuery objects use.each(function(index, element){})

```
$('div').each(function(index, element) {
    console.log(index + '=' + $(element).text());
});
```

You can also use the this object to reference the element



Accesing attributes

Object attributes can be accessed using .attr(attributeName)

var val = \$('#CustomerDiv').attr(title); // retrives the title attribute

Modifying attributes

• Object attributes can be modified using .attr(attributeName, value)

```
$('img').attr('title', 'My Image Title');
changes the title attribute to a value of "My Image Title"
```

 To modify multitple attributes, pass a JSON object containing name/value pairs

```
$('img').attr({
    title: 'My Image Title",
    style: 'border: 2px solid black"
});
```

Teamnet

Creating nodes ·

- Two ways of creating nodes using jQuery:
 - \$(htmlString), where htmlString is a valid HTML, in string format
 ex: \$("<div class='bar'>bla</div>
 //ul>");
 - \$(tag [, options]), where tag is the tag of the node you want to create, and optionally, an object containing the attributes to add to the newly created node (as of jQuery 1.8)

```
ex (all are equivalent):
$('<div>', { 'class': 'main' });
$('<div/>', { 'class': 'main' });
$('<div></div>', { 'class': 'main' });
```

Adding and removing nodes

- Four key methods handle inserting nodes into elements:
 - container.append(nodeToAppend)
 - nodeToAppend.appendTo(container)
 - container.prepend(nodeToPrepend)
 - nodeToPrepend.prependTo(container)
- To remove nodes from a element use .remove()

Modifying styles

• The .css() function can be used to modify an object's style: \$('div').css("color", "red");

Multiple styles can be modified by passing a JSON object

```
$('div').css({
        "color": "red",
        "font-weight": "bord"
});
```

Working with css classes

- The four methods for working with CSS Class attributes are:
 - addClass()
 - .hasClass()
 - .removeClass()
 - .toggleClass()

Exercise 2

Create a table containing hotel information, **dynamically**, using jQuery:

- the hotel list will be stored in an array
- this is the **content** of the HTML body:

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

- there will be a function (for example generateHotelView) that will accept two arguments: a **container** (jQuery object that will containing all the content) and an **array of hotels**; all of the logic will be inside this **main function** to keep the namespace pollution to the minimum (similar to a IIFE)
- when this function is called, the entire table will be created and populated

теαтпет