

# jQuery

*intro*

**PHP web development 2019/2020**

Milena Tomova  
Vratsa Software

<https://vratsasoftware.com/>

1. Introducing jQuery
2. Ways of adding jQuery to a project
3. Useful resources
4. Selecting elements and modifying the DOM
5. Traversing the DOM
6. Add/Remove elements
7. jQuery objects
8. jQuery events, preventDefault()

The image features a dark blue circle in the center, containing the word "jQuery" in white. The background is a light gray illustration of a mountain range with a dense forest of evergreen trees in the foreground. The mountains are depicted with sharp peaks and ridges, while the trees are represented by simple, stylized outlines.

**jQuery**



# Introducing jQuery

jQuery

- an **open-source** JavaScript library
- not the only one JS library
- but the most popular and widely used
- provides
  - **cross-browser compatibility**
  - **simpler syntax**
    - DOM elements are easier to select and traverse
    - easier to create elements
    - easier to manage events
    - easier to handle Ajax requests

# Introducing jQuery

Why is jQuery so popular?

Why is jQuery so popular?

- jQuery is easy to study
- jQuery easy to expand
- jQuery is light
- A lot of developers contribute to jQuery features development and maintenance

Using jQuery won't be enough -

**you will still need to use vanilla JavaScript!**



## Ways of adding jQuery to a project



# Ways of adding jQuery to a project

1. [Download jQuery files and link them to the project](#)
2. [Use a CDN /preferable/](#)





## Useful resources



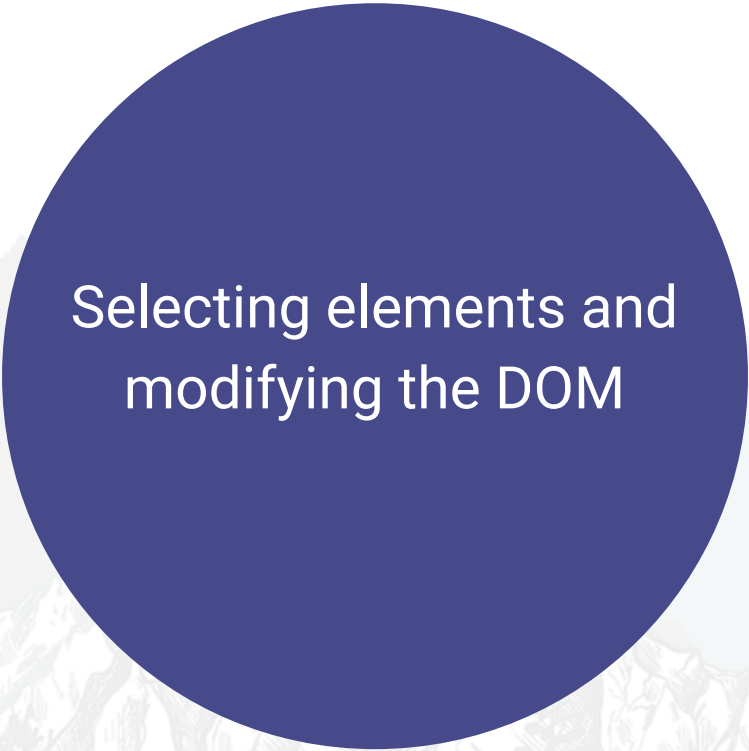


<https://jquery.com/>

Zak Ruvalcaba, Anne Boehm - **Murach's jQuery, Nth edition**  
(Training & Reference)

**Ray Nicholus, Beyond jQuery**

**Jason Lengstorf, Keith Wald - Pro PHP and jQuery (2nd Edition)**  
(The Expert's Voice in Web Development) - 2016



Selecting elements and  
modifying the DOM



Selecting elements from the DOM resembles the JS DOM element selection

`jQuery(selector)` = `$(selector)`  
full version                      short version

select element

**//by tag**

`$("div")`

**//by id**

`$("#navigation")`

**//by class  
selectors**

`$(".menu-item")`

**//by combination of**

`$("ul.menu li")`



## *Calling methods*

- select elements to call a method on

```
let element = $('selector');
```

- call a method -

```
element.methodName(parameter);
```

Almost always the selection returns **a collection** of elements!

## The syntax for calling a jQuery method

```
$ ("selector").methodName(parameters)
```

## Some common jQuery methods

Method	Description
<code>val()</code>	Get the value of a text box or other form control.
<code>val(value)</code>	Set the value of a text box or other form control.
<code>text()</code>	Get the text of an element.
<code>text(value)</code>	Set the text of an element.
<code>next([type])</code>	Get the next sibling of an element or the next sibling of a specified type if the parameter is coded.
<code>submit()</code>	Submit the selected form.
<code>focus()</code>	Move the focus to the selected form control or link.



# Traversing the DOM



- select element /get a jQuery object to apply the jQuery methods on/  
let element = \$(selector);

Then use the jQuery object properties for  
Next and previous siblings, Parents and children

```
let nextSibling = element.next([selector])  
//next sibling
```

```
let prevSibling = element.prev([selector])  
//previous sibling
```

**nextSibling, prevSibling** - new jQuery object are returned

**nextSibling, prevSibling - new jQuery object are returned**

If a jQuery object - you can apply jQuery object`s methods!

Else - you will get an error!



element.**parent([selector])**

//returns the immediate parent of each of the element the method is applied to

element.**parents(selector)**

//returns the parent that is selected with the given selector



# Traversing the DOM

```
<div id="wrapper">  
  <ul id="items-list">  
    <li>Item 1</li>  
    <li>Item 2</li>  
    <li class="special">Item 3</li>  
    <li>Item 4</li>  
  </ul>  
</div>
```

```
<script type="text/javascript">  
  var $node = $(".special");  
  console.log($node.parent().attr("id"));  
  console.log($node.parents("div").attr("id"));  
  console.log($node.parents("#wrapper").attr("id"));  
</script>
```

items-list

wrapper

wrapper





Add/Remove  
elements



**jQuery**.appendTo()/prependTo()

**jQuery** stands for **the jQuery object** -

the element from the DOM selected or created using jQuery

**Using JS** document.getElementById...

document.querySelector...

document.createElement....

**methods** won't create (a) jQuery object/s!



# Add/remove elements

create an element - a jQuery object

```
let hElement = $('<h1>header</h1>')
```

create elements - jQuery objects

```
let multipleElements = $('<ul><li>Hello</li></ul>')
```

# Add/remove elements

```
jQuery.appendTo()/prependTo()
```

```
$("<ul><li>Hello</li></ul>").appendTo("body");
```

```
jQuery.append()/prepend()
```

```
$("body").prepend("<h1>header</h1>");
```



# Add/remove elements

```
<div class="container">  
  <div class="inner">  
    <p>First  
Paragraph</p>  
    <p>Second  
Paragraph</p>  
  </div>  
</div>
```

```
$( 'p' ).remove();  
    //will remove all p-elements  
from the document
```

```
$( '.inner' ).remove();
```





jQuery objects



**The jQuery objects** have more methods and properties than JavaScript DOM objects.

`addClass()`, `removeClass()`,  
`toggleClass()`

`on(event, callback)` //for events listening  
and handling

`animate()`, `fadeIn()`, etc...

**jQuery methods** for changing the DOM elements

**jQuery.css("color", "#f3f")**

**jQuery.html()**

*returns the innerHTML*

**jQuery.html(content)**

*sets the innerHTML*

**jQuery.text(content)**

*sets and escapes the innerHTML*





# jQuery events



jQuery has methods

for adding **on()**

and removing events **off()**

```
function onButtonClick(){  
    $(".selected").removeClass("selected");  
    $(this).addClass("selected");  
}
```

```
$("#a.button").on("click", onButtonClick);
```

**\$(this)** is the **event target**



## jQuery.**preventDefault()**

used to stop the default behaviour of an element when an event is triggered.

***Clicking** submit button, **submits** the form.*

***Clicking** a tag, leads to its **href** or **reloads** the page when href is missing.*

To be able to validate form data with JS before submission -

## submitButton.**preventDefault()**

To use an a-element for a slider control button -

## aElement.**preventDefault()**

in the event handler methods.



## Two common jQuery event methods

Event method	Description
<code>ready(handler)</code>	The event handler runs when the DOM is ready.
<code>click(handler)</code>	The event handler runs when the selected element is clicked.

## Two ways to code an event handler for the jQuery ready event

### The long way

```
$(document).ready(function() {  
    alert("The DOM is ready");  
});
```

### The short way

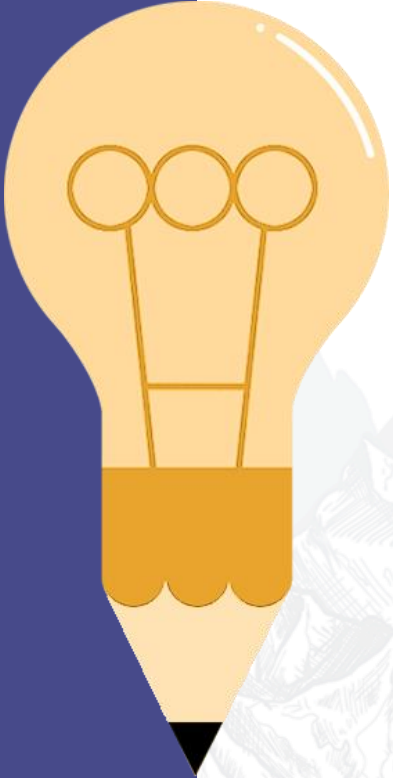
```
$(function() {  
    alert("The DOM is ready");  
}); // (document).ready is assumed
```

## An event handler for the click event of all h2 elements

```
$("#h2").click(function() {  
    alert("This heading has been clicked");  
});
```

## The click event handler within the ready event handler

```
$(document).ready(function() {  
    $("#h2").click(function() {  
        alert("This heading has been clicked");  
    });  
});  
// end of click event handler  
// end of ready event handler
```



**click()** won't work on elements that **do not exist** in the DOM when the document is loaded in the browser /elements created and added to the DOM afterwards/.

Always use **on()** instead on element(s) that **are part of the DOM** when the document is **initially loaded in the browser**.




# A summary of the most useful jQuery methods

Method	Description
<b>next</b> ( [ <i>selector</i> ] )	Get the next sibling of each selected element or the first sibling of a specified type if the parameter is coded.
<b>prev</b> ( [ <i>selector</i> ] )	Get the previous sibling of each selected element or the previous sibling of a specified type if the parameter is coded.
<b>attr</b> ( <i>attributeName</i> )	Get the value of the specified attribute from the first selected element.
<b>attr</b> ( <i>attributeName</i> , <i>value</i> )	Set the value of the specified attribute for each selected element.
<b>css</b> ( <i>propertyName</i> )	Get the value of the specified property from the first selected element.
<b>css</b> ( <i>propertyName</i> , <i>value</i> )	Set the value of the specified property for each selected element.
<b>addClass</b> ( <i>className</i> )	Add one or more classes to the selected elements and, if necessary, create the class. If you use more than one class as the parameter, separate them with spaces.
<b>removeClass</b> ( [ <i>className</i> ] )	Remove one or more classes. If you use more than one class as the parameter, separate them with spaces.
<b>toggleClass</b> ( <i>className</i> )	If the class is present, remove it. Otherwise, add it.
<b>hide</b> ( [ <i>duration</i> ] )	Hide the selected elements. The duration parameter can be “slow”, “fast”, or a number giving the time in milliseconds. By default, the duration is 400 milliseconds, “slow” is 600 milliseconds, and “fast” is 200 milliseconds.
<b>show</b> ( [ <i>duration</i> ] )	Show the selected elements. The duration parameter is the same as for the hide method.
<b>each</b> ( <i>function</i> )	Run the function for each element in an array.

# A summary of the most useful jQuery event methods

Event method	Description
<code>ready(handler)</code>	The handler runs when the DOM is ready.
<code>unload(handler)</code>	The handler runs when the user closes the browser window.
<code>error(handler)</code>	The handler runs when a JavaScript error occurs.
<code>click(handler)</code>	The handler runs when the selected element is clicked.
<code>dblclick(handler)</code>	The handler runs when the selected element is double-clicked.
<code>mouseenter(handler)</code>	The handler runs when the mouse pointer enters the selected element.
<code>mouseover(handler)</code>	The handler runs when the mouse pointer moves over the selected element.
<code>mouseout(handler)</code>	The handler runs when the mouse pointer moves out of the selected element.
<code>hover(handlerIn, handlerOut)</code>	The first event handler runs when the mouse pointer moves into an element. The second event handler runs when the mouse pointer moves out.
<code>event.preventDefault()</code>	Stops the default action of an event from happening.





## jQuery chaining methods



# jQuery chaining methods

jQuery methods always return a result or this  
So you can chain jQuery methods -  
the **next method** is applied on **the result of the previous method**.

```
$('#<button>')  
  .addClass('btn-success')  
  .html('Click me for success')  
  .on('click', onSuccessButtonClick)  
  .appendTo(document.body);
```



# Questions?



Гнездото  
Coworking

Цялостен  
курс по  
програми  
ране

Дизайн  
курс

Курс по  
дигит.  
маркетинг

MindHub



# Partners



**Telerik  
Academy**



**MindHub**

**ПРОМЯНАТА**



# Trainings @ Vratsa Software



- Vratsa Software – High-Quality Education, Profession and Jobs
  - [www.vratsasoftware.com](http://www.vratsasoftware.com)
- The Nest Coworking
  - [www.nest.bg](http://www.nest.bg)
- Vratsa Software @ Facebook
  - [www.fb.com/VratsaSoftware](http://www.fb.com/VratsaSoftware)
- Slack Channel
  - [www.vso.slack.com](http://www.vso.slack.com)

