jQuery

Topics

- Intro to jQuery
- Adding Jquery to web pages
- Syntax
- jQuery Selectors
- Events
- Effects
- Chaining
- DOM Manipulation
- Traversing

What is jQuery?

- jQuery is a fast, small and feature-rich JavaScript library included in a single .js file
- It provides many built-in functions using which you can accomplish various tasks easily and quickly

Adding jQuery to web pages – Way 1

- Downloading jQuery
  - Production version – For live website because it has been minified and compressed
  - Development version - For testing and development (uncompressed and readable code)

```
<head>
<script src="jquery-3.3.1.min.js"></script>
</head>
```

Adding jQuery to web pages – Way 2

- jQuery CDN (Content Delivery Network)

Google CDN:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
</head>
```

Microsoft CDN:

```
<head>
<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js"></script>
</head>
```

jQuery Syntax

- With jQuery HTML elements can be selected and "actions" can be performed

Basic syntax is: **$(*selector*).*action*()**

- A $ sign to define/access jQuery
- A (*selector*) to "query (or find)" HTML elements
- A jQuery *action*() to be performed on the element(s)

Examples:

$(this).hide() - hides the current element.

$("p").hide() - hides all <p> elements.

$(".test").hide() - hides all elements with class="test".

$("#test").hide() - hides the element with id="test".

Document Ready Event

- It is good practice to wait for the document to be fully loaded and ready before working with it

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

    // jQuery methods go here...

});
```

```
$(function(){

    // jQuery methods go here...

});
```

This is to prevent any jQuery code from running before the document is finished loading

jQuery Selectors

- JQuery Selectors doc

- jQuery selectors allow to select and manipulate HTML element(s)
- All selectors in jQuery start with the dollar sign and parentheses: $()

jQuery Selectors

The element Selector

The jQuery element selector selects elements based on the element name.

You can select all <p> elements on a page like this:

```
$("p")
```

Example

When a user clicks on a button, all <p> elements will be hidden:

Example

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

jQuery Selectors

Syntax	Description
$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"
$("p:first")	Selects the first <p> 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

jQuery Selectors

$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

jQuery Events

What are Events?

All the different visitor's actions that a web page can respond to are called events.

An event represents the precise moment when something happens.

Examples:

- moving a mouse over an element
- selecting a radio button
- clicking on an element

The term **"fires/fired"** is often used with events. Example: "The keypress event is fired, the moment you press a key".

- JQuery Events doc

jQuery Events

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

jQuery Syntax For Event Methods

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

```
$("p").click(function(){
  // action goes here!!
});
```

```
$("#p1").mouseenter(function(){
    alert("You entered p1!");
});
```

```
$("#p1").mouseleave(function(){
    alert("Bye! You now leave p1!");
});
```

hover( )

The hover() method takes two functions and is a combination of the mouseenter() and mouseleave() methods.

The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML element:

Example

```
$("#p1").hover(function(){
    alert("You entered p1!");
},
function(){
    alert("Bye! You now leave p1!");
});
```

focus( )

The focus() method attaches an event handler function to an HTML form field.

```
$(document).ready(function(){
    $("input").focus(function(){
        $(this).css("background-color", "#cccccc");
    });
    $("input").blur(function(){
        $(this).css("background-color", "#ffffff");
    });
});
</script>
</head>
<body>

Name: <input type="text" name="fullname"><br>
Email: <input type="text" name="email">

</body>
</html>
```

on( ) Method

The on() method attaches one or more event handlers for the selected elements.

Attach a click event to a <p> element:

Example

```
$("p").on("click", function(){
    $(this).hide();
});
```

on( ) Method

Attach multiple event handlers to an element

```
$("p").on({
    mouseenter: function(){
        $(this).css("background-color", "lightgray");
    },
    mouseleave: function(){
        $(this).css("background-color", "lightblue");
    },
    click: function(){
        $(this).css("background-color", "yellow");
    }
});
```

jQuery Effects

Hide, Show, Toggle, Slide, Fade

```
$(selector).hide(speed,callback);

$(selector).show(speed,callback);
```

Hide , show

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
    $("#hide").click(function(){
        $("p").hide();
    });
    $("#show").click(function(){
        $("p").show();
    });
});
</script>
</head>
<body>

<p>If you click on the "Hide" button, I will disappear.</p>

<button id="hide">Hide</button>
<button id="show">Show</button>

</body>
</html>
```

jQuery Effects

```
$("button").click(function(){
    $("p").hide(1000);
});
```

toggle()

With jQuery, you can toggle between the hide() and show() methods with the toggle() method.

Shown elements are hidden and hidden elements are shown:

Example

```
$("button").click(function(){
    $("p").toggle();
});
```

```
$(selector).toggle(speed,callback);
```

Fade()

With jQuery you can fade an element in and out of visibility.

jQuery has the following fade methods:

- fadeIn()
- fadeOut()
- fadeToggle()
- fadeTo()

Fading

```
$(selector).fadeIn(speed,callback);
```

```
$(selector).fadeOut(speed,callback);
```

```
$(selector).fadeToggle(speed,callback);
```

```
$(selector).fadeTo(speed,opacity,callback);
```

slide()

With jQuery you can create a sliding effect on elements.

jQuery has the following slide methods:

- slideDown()
- slideUp()
- slideToggle()

```
$(selector).slideDown(speed,callback);
```

jQuery without call back

$(*selector*).hide(*speed,callback*);

```
$("button").click(function(){
    $("p").hide(1000);
    alert("The paragraph is now hidden");
});
```

jQuery Callback Functions

$(*selector*).hide(*speed,callback*);

```
$("button").click(function(){
    $("p").hide("slow", function(){
        alert("The paragraph is now hidden");
    });
});
```

jQuery Chaining

- Can chain together actions/methods
- Chaining allows us to run multiple jQuery methods (on the same element) within a single statement

```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

```
$("#p1").css("color", "red")
  .slideUp(2000)
  .slideDown(2000);
```

jQuery DOM Manipulation

- One very important part of jQuery is the possibility to manipulate the DOM
- jQuery comes with a bunch of DOM related methods that make it easy to access and manipulate elements and attributes

Get & Set Content

Get Content - text(), html(), and val()

Three simple, but useful, jQuery methods for DOM manipulation are:

- text() - Sets or returns the text content of selected elements
- html() - Sets or returns the content of selected elements (including HTML markup)
- val() - Sets or returns the value of form fields

```
$("#btn1").click(function(){
    alert("Text: " + $("#test").text());
});
$("#btn2").click(function(){
    alert("HTML: " + $("#test").html());
});
```

```
$("#btn1").click(function(){
    alert("Value: " + $("#test").val());
});
```

Get attributes

The jQuery attr() method is used to get attribute values.

The following example demonstrates how to get the value of the href attribute in a link:

Example

```
$("button").click(function(){
    alert($("#w3s").attr("href"));
});
```

Set Content and Attributes

- text() - Sets or returns the text content of selected elements
- html() - Sets or returns the content of selected elements (including HTML markup)
- val() - Sets or returns the value of form fields

```
$("#btn1").click(function(){
    $("#test1").text("Hello world!");
});
$("#btn2").click(function(){
    $("#test2").html("<b>Hello world!</b>");
});
$("#btn3").click(function(){
    $("#test3").val("Dolly Duck");
});
```

Set Attributes - attr()

```
$("button").click(function(){
    $("#w3s").attr("href", "https://www.w3schools.com/jquery/");
});
```

```
$("button").click(function(){
    $("#w3s").attr({
        "href" : "https://www.w3schools.com/jquery/",
        "title" : "W3Schools jQuery Tutorial"
    });
});
```

Add new HTML Elements

- append() - Inserts content at the end of the selected elements
- prepend() - Inserts content at the beginning of the selected elements
- after() - Inserts content after the selected elements
- before() - Inserts content before the selected elements

```
$("p").append("Some appended text.");

$("p").prepend("Some prepended text.");
```

```
$("img").after("Some text after");

$("img").before("Some text before");
```

Add multiple new HTML Elements

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>
<script>
function appendText() {
    var txt1 = "<p>Text.</p>";               // Create text with HTML
    var txt2 = $("<p></p>").text("Text.");  // Create text with jQuery
    var txt3 = document.createElement("p");
    txt3.innerHTML = "Text.";                // Create text with DOM
    $("body").append(txt1, txt2, txt3);      // Append new elements
}
</script>
</head>
<body>

<p>This is a paragraph.</p>
<button onclick="appendText()">Append text</button>

</body>
</html>
```

Remove HTML Elements

To remove elements and content, there are mainly two jQuery methods:

- remove() - Removes the selected element (and its child elements)
- empty() - Removes the child elements from the selected element

```
$("#div1").empty();

$("p").remove(".test, .demo");
```

```
$("#div1").remove();

$("p").remove(".test");
```

jQuery Manipulating – get & set CSS class

- addClass() - Adds one or more classes to the selected elements
- removeClass() - Removes one or more classes from the selected elements
- toggleClass() - Toggles between adding/removing classes from the selected elements
- css() - Sets or returns the style attribute

jQuery Manipulating CSS class – addClass()

```css
.important {
    font-weight: bold;
    font-size: xx-large;
}

.blue {
    color: blue;
}
```

```javascript
$("button").click(function(){
    $("h1, h2, p").addClass("blue");
    $("div").addClass("important");
});
```

jQuery Manipulating CSS class – addClass()

```
.important {
    font-weight: bold;
    font-size: xx-large;
}

.blue {
    color: blue;
}
```

```
$("button").click(function(){
    $("h1, h2, p").addClass("blue");
    $("div").addClass("important");
});
```

Multiple classes within the addClass() method

```
$("button").click(function(){
    $("#div1").addClass("important blue");
});
```

jQuery Manipulating CSS class – removeClass()

```css
.important {
    font-weight: bold;
    font-size: xx-large;
}

.blue {
    color: blue;
}
```

```
$("button").click(function(){
    $("h1, h2, p").removeClass("blue");
});
```

jQuery Manipulating CSS class – toggleClass()

```css
.important {
    font-weight: bold;
    font-size: xx-large;
}

.blue {
    color: blue;
}
```

```javascript
$("button").click(function(){
    $("h1, h2, p").toggleClass("blue");
});
```

jQuery - css() Method

The css() method sets or returns one or more style properties for the selected elements

Get property

To return the value of a specified CSS property, use the following syntax:

```
css("propertyname");
```

The following example will return the background-color value of the FIRST matched element:

Example

```
$("p").css("background-color");
```

jQuery - css() Method

Set property

To set a specified CSS property, use the following syntax:

```
css("propertyname","value");
```

The following example will set the background-color value for ALL matched elements:

Example

```
$("p").css("background-color", "yellow");
```

jQuery - css() Method

Set Multiple Property

To set multiple CSS properties, use the following syntax:

```
css({"propertyname":"value","propertyname":"value",...});
```

The following example will set a background-color and a font-size for ALL matched elements:

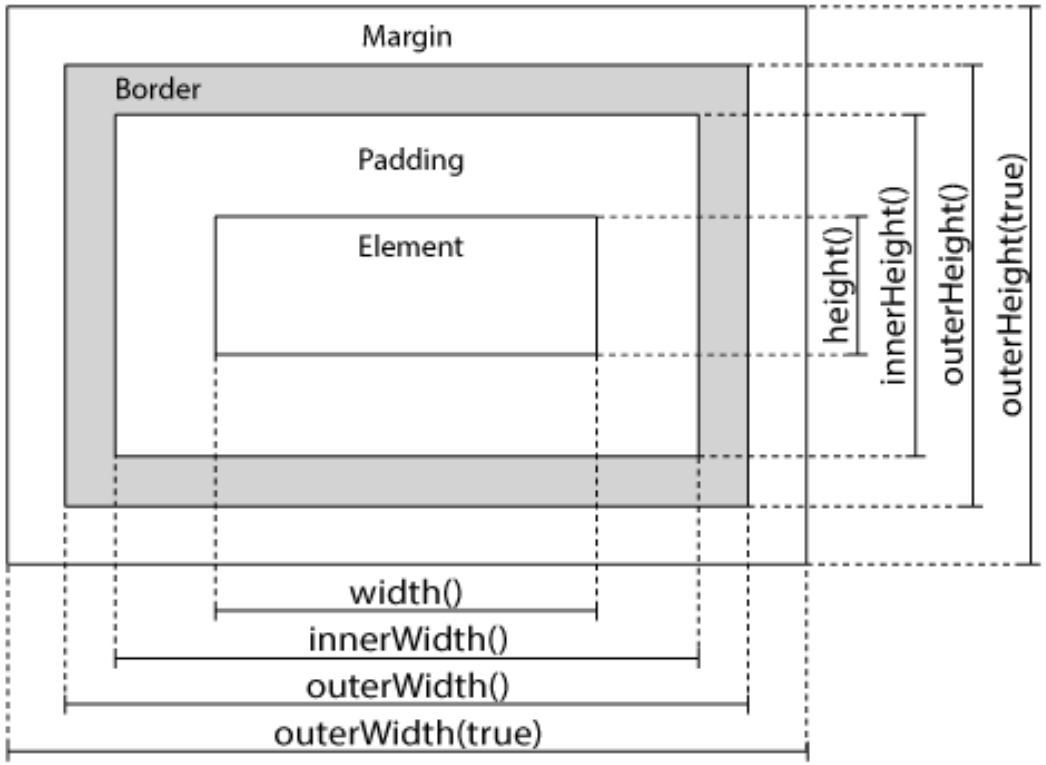
Example

```
$("p").css({"background-color": "yellow", "font-size": "200%"});
```

jQuery Dimension Methods

jQuery has several important methods for working with dimensions:

- width()
- height()
- innerWidth()
- innerHeight()
- outerWidth()
- outerHeight()

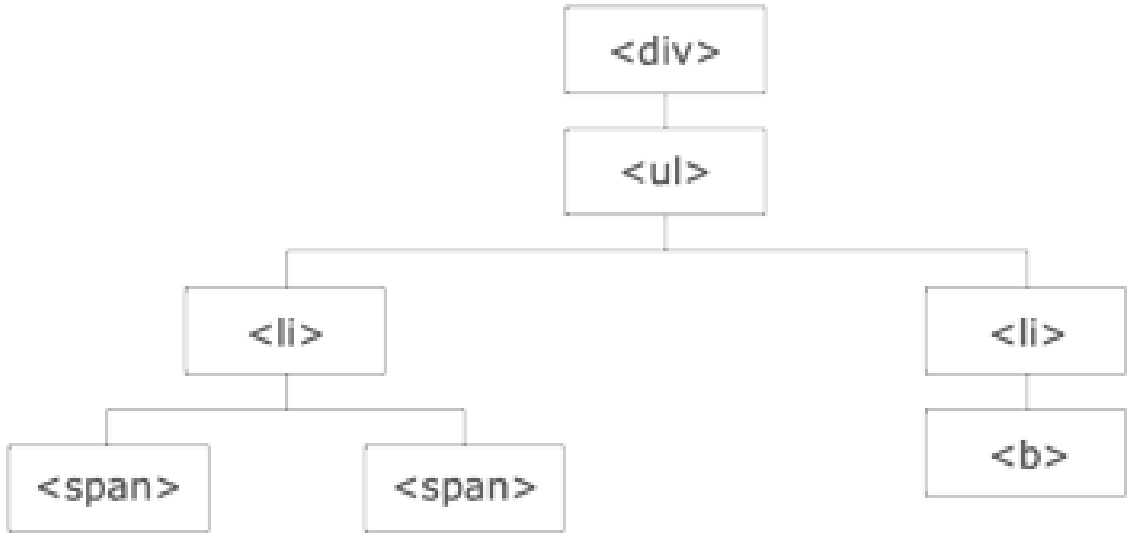



```
$("button").click(function(){
    $("#div1").width(500).height(500);
});
```

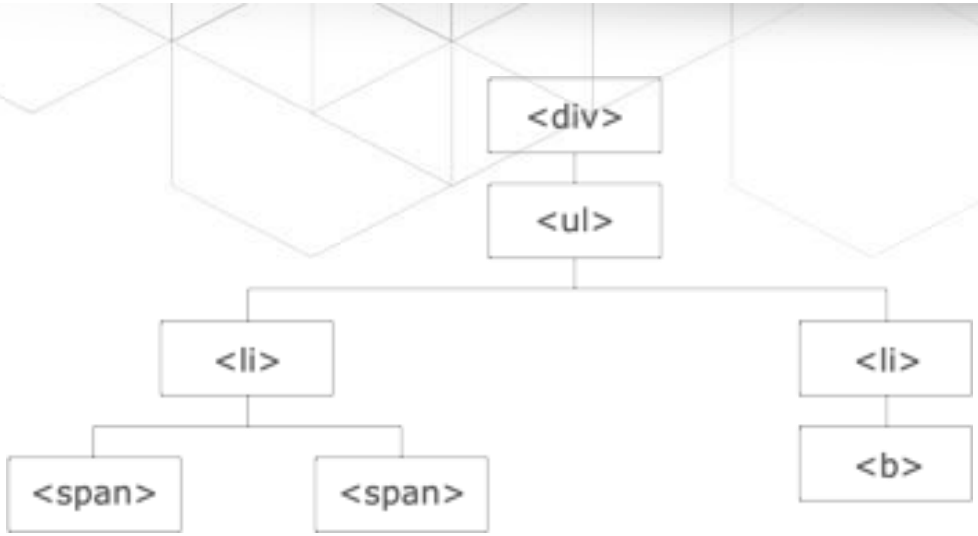
jQuery Traversing

- jQuery traversing means "move through“ to "find" (or select) HTML elements based on their relation to other elements

- Start with one selection and move through that selection until you reach the elements you desire



jQuery Traversing



- The element is the **parent** of <ul>, and an **ancestor** of everything inside of it
- The <ul> element is the **parent** of both <li> elements, and a **child** of
- The left <li> element is the **parent** of , **child** of <ul> and a **descendant** of
- The element is a **child** of the left <li> and a **descendant** of <ul> and
- The two <li> elements are **siblings** (they share the same parent)
- The right <li> element is the **parent** of <b>, child of <ul> and a **descendant** of
- The <b> element is a **child** of the right <li> and a **descendant** of <ul> and

An ancestor : parent, grandparent, great-grandparent, and so on.
A descendant : child, grandchild, great-grandchild, and so on.
Siblings share the same parent.

Traversing the DOM - Ancestors

Three useful jQuery methods for traversing up the DOM tree are:

- parent()
- parents()
- parentsUntil()

Parent()

The parent() method returns the direct parent element of the selected element.

This method only traverse a single level up the DOM tree.

The following example returns the direct parent element of each elements:

Example

```
$(document).ready(function(){
    $("span").parent();
});
```

```
$(document).ready(function(){
    $("span").parent().css({"color": "red", "border": "2px solid red"});
});
```

Parents()

The parents() method returns all ancestor elements of the selected element, all the way up to the document's root element (<html>).

The following example returns all ancestors of all elements:

Example

```
$(document).ready(function(){
    $("span").parents();
});
```

```
$(document).ready(function(){
    $("span").parents("ul");
});
```

ParentsUntil()

The parentsUntil() method returns all ancestor elements between two given arguments.

The following example returns all ancestor elements between a and a element:

Example

```
$(document).ready(function(){
    $("span").parentsUntil("div");
});
```

jQuery Traversing - Descendants

Two useful jQuery methods for traversing down the DOM tree are:

- children()
- find()

Children()

The children() method returns all direct children of the selected element.

This method only traverses a single level down the DOM tree.

The following example returns all elements that are direct children of each elements:

Example

```
$(document).ready(function(){
    $("div").children();
});
```

```
$(document).ready(function(){
    $("div").children("p.first");
});
```

jQuery find() Method

The find() method returns descendant elements of the selected element, all the way down to the last descendant.

The following example returns all elements that are descendants of :

Example

```
$(document).ready(function(){
    $("div").find("span");
});
```

```
$(document).ready(function(){
    $("div").find("*");
});
```

jQuery Traversing - Siblings

- With jQuery you can traverse sideways in the DOM tree to find siblings of an element.
- Siblings share the same parent

Traversing Sideways in The DOM Tree

There are many useful jQuery methods for traversing sideways in the DOM tree:

- siblings()
- next()
- nextAll()
- nextUntil()
- prev()
- prevAll()
- prevUntil()

Siblings()

```
$(document).ready(function(){
    $("h2").siblings();
});
```

```
$(document).ready(function(){
    $("h2").siblings("p");
});
```

next()

The next() method returns the next sibling element of the selected element.

The following example returns the next sibling of <h2>:

Example

```
$(document).ready(function(){
    $("h2").next();
});
```

jQuery nextAll() Method

The nextAll() method returns all next sibling elements of the selected element.

The following example returns all next sibling elements of <h2>:

Example

```
$(document).ready(function(){
    $("h2").nextAll();
});
```

nextUntil()

The nextUntil() method returns all next sibling elements between two given arguments.

The following example returns all sibling elements between a <h2> and a <h6> element:

Example

```
$(document).ready(function(){
    $("h2").nextUntil("h6");
});
```

Similarly jQuery prev(), prevAll() & prevUntil() methods work

jQuery Traversing - Filtering

- first(), last() and eq()
  - Allows to select a specific element based on its position in a group of elements

- filter() and not()
  - allows to select elements that match, or do not match, a certain criteria

First()

The first() method returns the first element of the specified elements.

The following example selects the first element:

Example

```
$(document).ready(function(){
    $("div").first();
});
```

Last()

The last() method returns the last element of the specified elements.

The following example selects the last element:

Example

```
$(document).ready(function(){
    $("div").last();
});
```

eq()

The eq() method returns an element with a specific index number of the selected elements.

The index numbers start at 0, so the first element will have the index number 0 and not 1. The following example selects the second <p> element (index number 1):

Example

```
$(document).ready(function(){
    $("p").eq(1);
});
```

filter()

The filter() method lets you specify a criteria. Elements that do not match the criteria are removed from the selection, and those that match will be returned.

The following example returns all <p> elements with class name "intro":

Example

```
$(document).ready(function(){
    $("p").filter(".intro");
});
```

not()

The not() method returns all elements that do not match the criteria.

Tip: The not() method is the opposite of filter().

The following example returns all <p> elements that do not have class name "intro":

Example

```
$(document).ready(function(){
    $("p").not(".intro");
});
```

Thank You

Our Locations: China | Costa Rica | India | Mauritius | Philippines | Poland | Singapore | U.K. | U.S.A.