jQuery

December 2022

Agenda

JQUERY INTRO

What is jQuery?

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

The jQuery library contains the following features:

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

Adding jQuery to Your Web Pages

There are several ways to start using jQuery on your web site. You can:

- Download the jQuery library from jQuery.com
- Include jQuery from a CDN, like Google

There are two versions of jQuery available for downloading:

- Production version - this is for your live website because it has been minified and compressed
- Development version - this is for testing and development (uncompressed and readable code)
- Both versions can be downloaded from jQuery.com.

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):

```
<head>
  <script src="jquery-3.5.1.min.js"></script>
</head>
```

If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network). Google is an example of someone who host jQuery:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.
5.1/jquery.min.js"></script>
</head>
```

One big advantage of using the hosted jQuery from Google:

Many users already have downloaded jQuery from Google when visiting another site. As a result, it will be loaded from cache when they visit your site, which leads to faster loading time. Also, most CDN's will make sure that once a user requests a file from it, it will be served from the server closest to them, which also leads to faster loading time.

jQuery Syntax

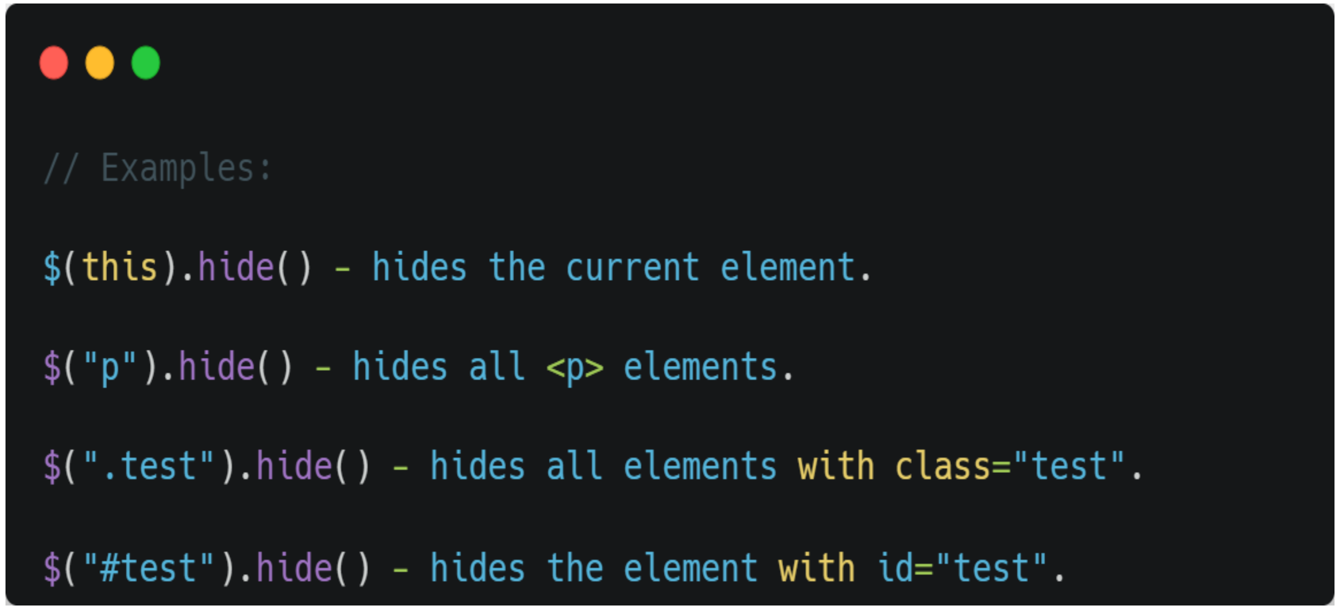
The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

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".
```

jQuery uses CSS syntax to select elements. You will learn more about the selector syntax in the next chapter of this tutorial.

jQuery Event Methods

What are Events?

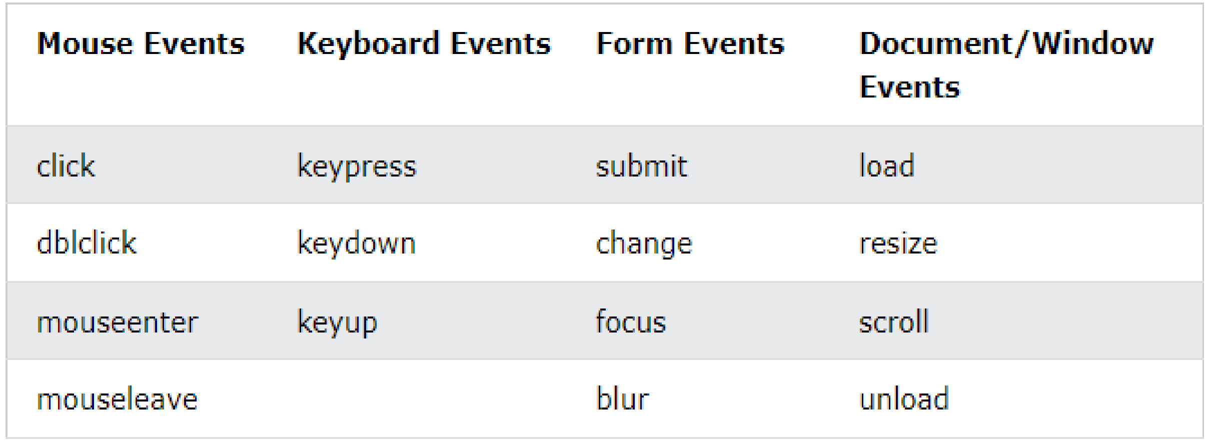
All the different visitors' actions that a web page can respond to are called events.

An event represents the precise moment when something happens.

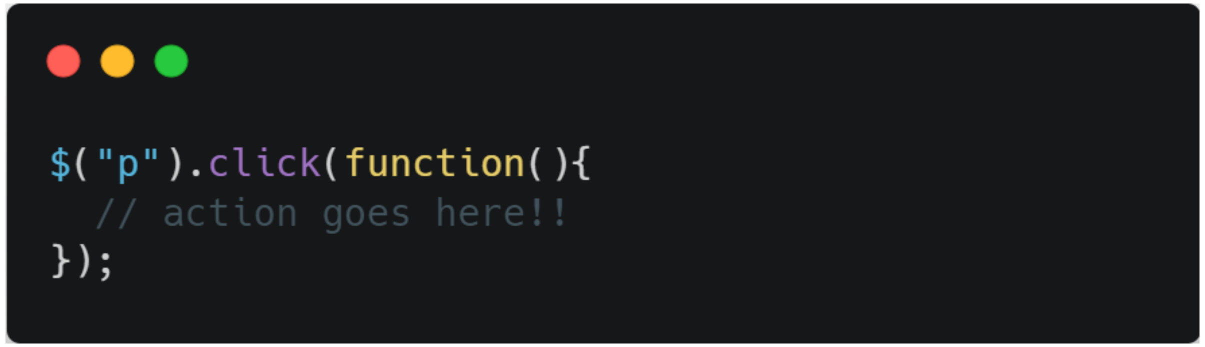
Examples:

- 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".



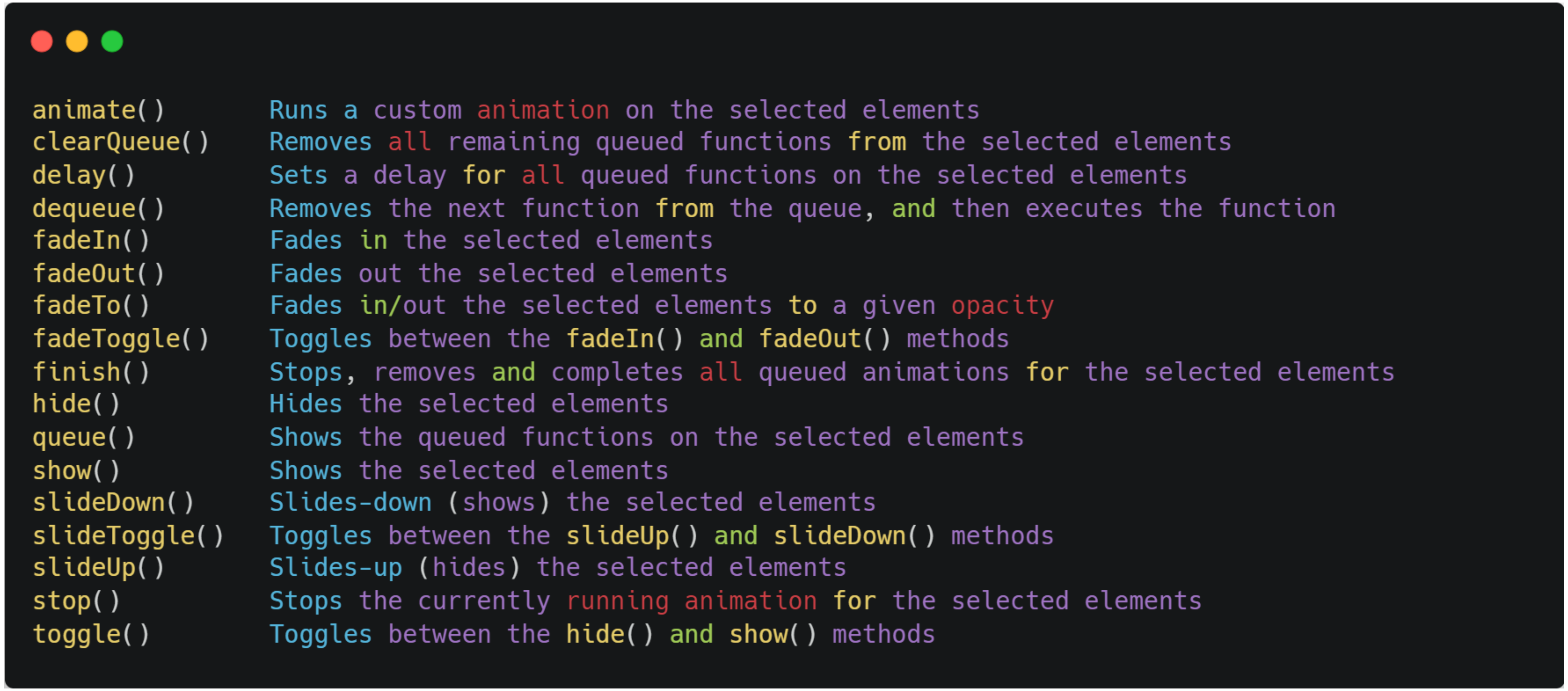
Mouse Events	**Keyboard Events**	**Form Events**	**Document/Window Events**
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload



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

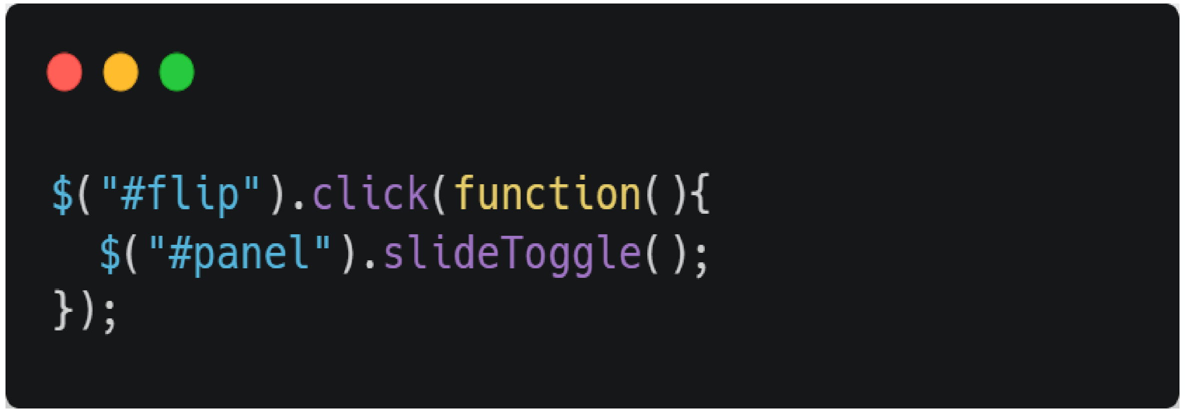
JQUERY EFFECTS

jQuery Effects

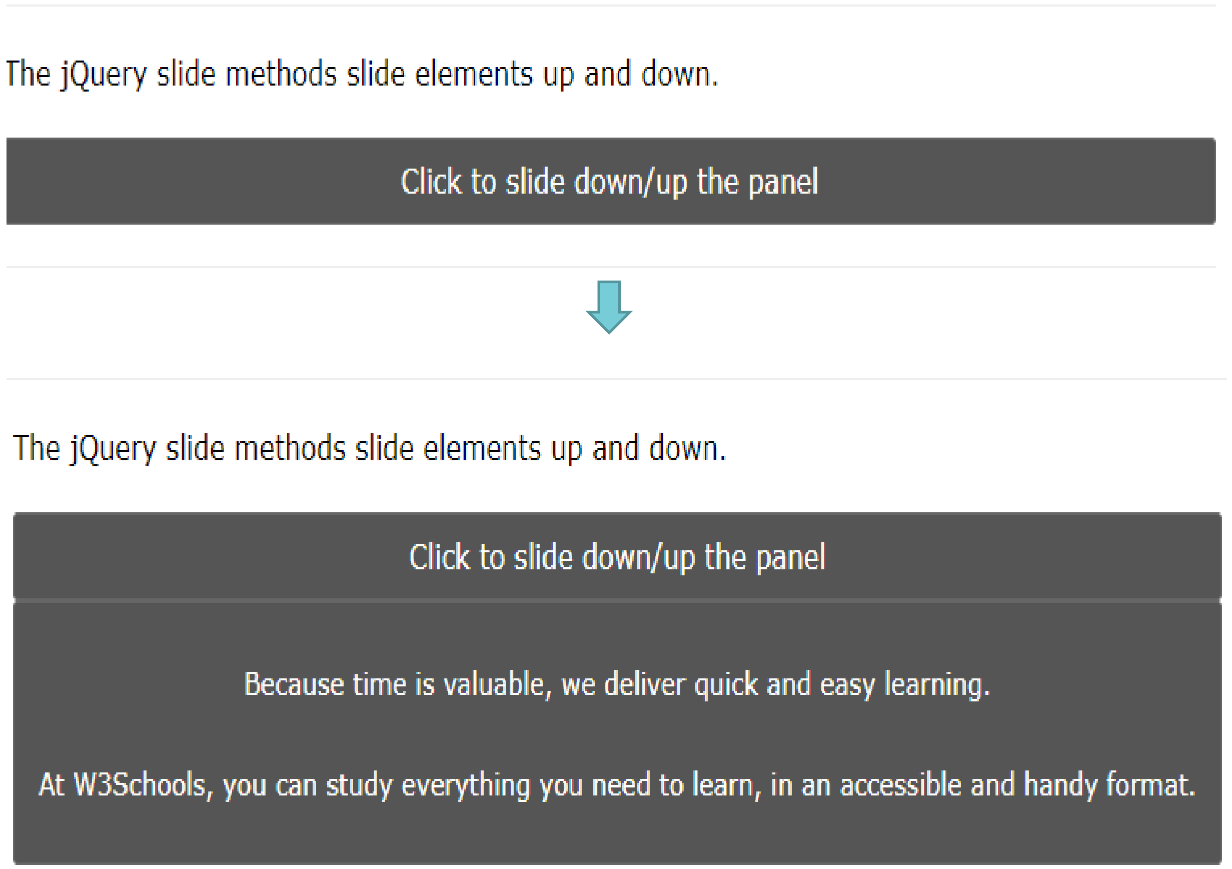


```
animate()       Runs a custom animation on the selected elements
clearQueue()    Removes all remaining queued functions from the selected elements
delay()         Sets a delay for all queued functions on the selected elements
dequeue()       Removes the next function from the queue, and then executes the function
fadeIn()        Fades in the selected elements
fadeOut()       Fades out the selected elements
fadeTo()        Fades in/out the selected elements to a given opacity
fadeToggle()    Toggles between the fadeIn() and fadeOut() methods
finish()        Stops, removes and completes all queued animations for the selected elements
hide()          Hides the selected elements
queue()         Shows the queued functions on the selected elements
show()          Shows the selected elements
slideDown()     Slides-down (shows) the selected elements
slideToggle()   Toggles between the slideUp() and slideDown() methods
slideUp()       Slides-up (hides) the selected elements
stop()          Stops the currently running animation for the selected elements
toggle()        Toggles between the hide() and show() methods
```

jQuery Effect example



```
$("#flip").click(function(){
  $("#panel").slideToggle();
});
```



The jQuery slide methods slide elements up and down.

Click to slide down/up the panel

The jQuery slide methods slide elements up and down.

Click to slide down/up the panel

Because time is valuable, we deliver quick and easy learning.

At W3Schools, you can study everything you need to learn, in an accessible and handy format.

JQUERY HTML + CSS

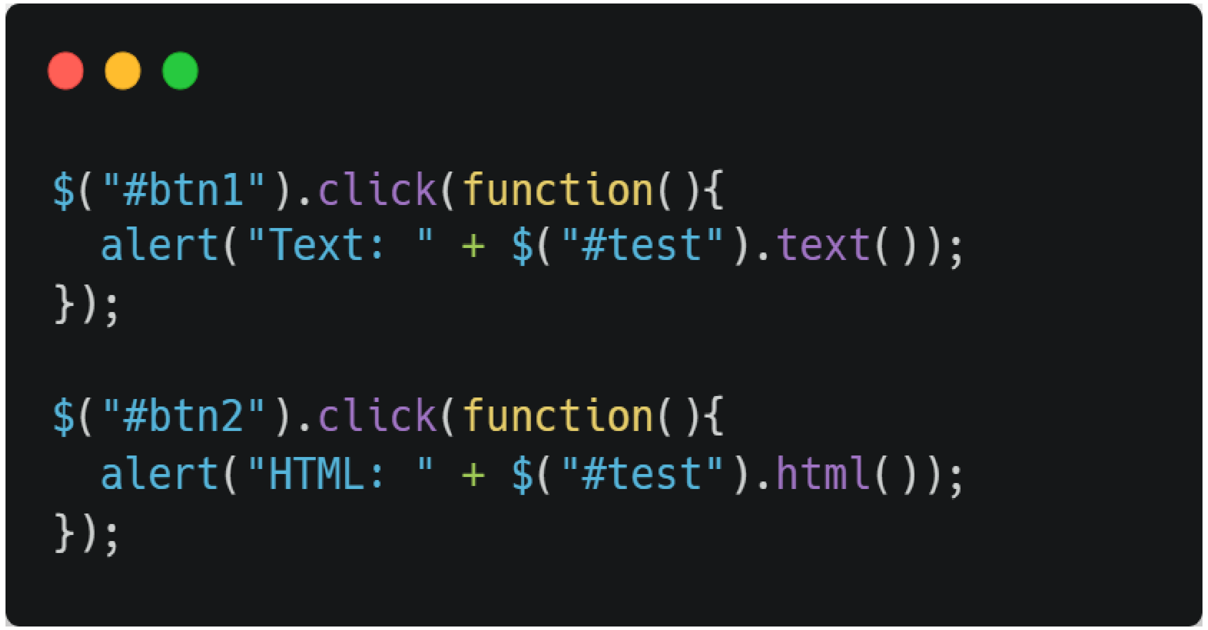
jQuery DOM Manipulation

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.

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



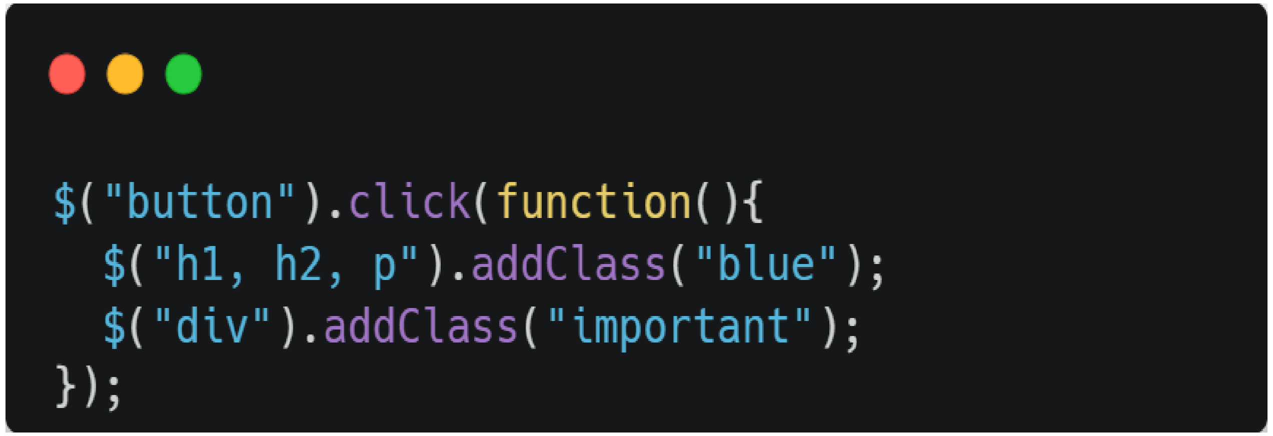
```
$("#btn1").click(function(){
  alert("Text: " + $("#test").text());
});

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

jQuery - Get and Set CSS Classes

jQuery has several methods for CSS manipulation. We will look at the following methods:

- 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



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

jQuery HTML / CSS Methods

addClass()	Adds one or more class names to selected elements
after()	Inserts content after selected elements
append()	Inserts content at the end of selected elements
appendTo()	Inserts HTML elements at the end of selected elements
attr()	Sets or returns attributes/values of selected elements
before()	Inserts content before selected elements
clone()	Makes a copy of selected elements
css()	Sets or returns one or more style properties for selected elements

jQuery HTML / CSS Methods

detach()	Removes selected elements (keeps data and events)
empty()	Removes all child nodes and content from selected elements
hasClass()	Checks if any of the selected elements have a specified class name
height()	Sets or returns the height of selected elements
html()	Sets or returns the content of selected elements
innerHeight()	Returns the height of an element (includes padding, but not border)
innerWidth()	Returns the width of an element (includes padding, but not border)
insertAfter()	Inserts HTML elements after selected elements

jQuery HTML / CSS Methods

insertBefore()	Inserts HTML elements before selected elements
offset()	Sets or returns the offset coordinates for selected elements
offsetParent()	Returns the first positioned parent element
outerHeight()	Returns the height of an element (includes padding and border)
outerWidth()	Returns the width of an element (includes padding and border)
position()	Returns the position (relative to the parent element) of an element
prepend()	Inserts content at the beginning of selected elements
prependTo()	Inserts HTML elements at the beginning of selected elements

jQuery HTML / CSS Methods

prop()	Sets or returns properties/values of selected elements
remove()	Removes the selected elements (including data and events)
removeAttr()	Removes one or more attributes from selected elements
removeClass()	Removes one or more classes from selected elements
removeProp()	Removes a property set by the prop() method
replaceAll()	Replaces selected elements with new HTML elements
replaceWith()	Replaces selected elements with new content
scrollLeft()	Sets or returns the horizontal scrollbar position of selected elementst

jQuery HTML / CSS Methods

scrollTop()	Sets or returns the vertical scrollbar position of selected elements
text()	Sets or returns the text content of selected elements
toggleClass()	Toggles between adding/removing one or more classes from selected elements
unwrap()	Removes the parent element of the selected elements
val()	Sets or returns the value attribute of the selected elements (for form elements)
width()	Sets or returns the width of selected elements
wrap()	Wraps HTML element(s) around each selected element
wrapAll()	Wraps HTML element(s) around all selected elements
wrapInner()	Wraps HTML element(s) around the content of each selected element

JQUERY TRAVERSING

jQuery Traversing

jQuery traversing, which means "move through", are used 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.

The image below illustrates an HTML page as a tree (DOM tree). With jQuery traversing, you can easily move up (ancestors), down (descendants) and sideways (siblings) in the tree, starting from the selected (current) element. This movement is called traversing - or moving through - the DOM tree.

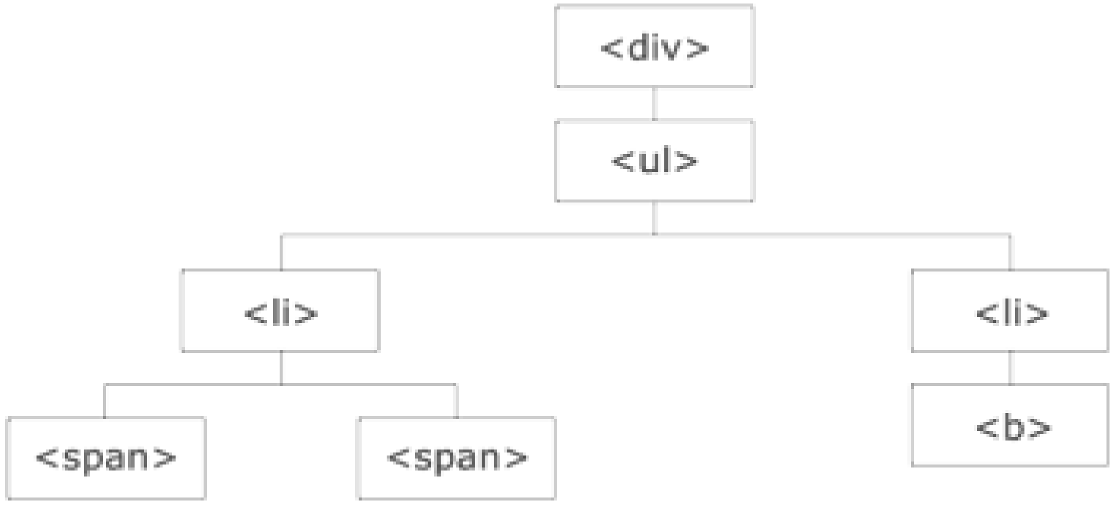


Illustration explained:

- The <div> 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 <div>
- The left <li> element is the **parent** of <span>, **child** of <ul> and a **descendant** of <div>
- The <span> element is a **child** of the left <li> and a **descendant** of <ul> and <div>
- 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 <div>
- The <b> element is a **child** of the right <li> and a **descendant** of <ul> and <div>

jQuery Traversing

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

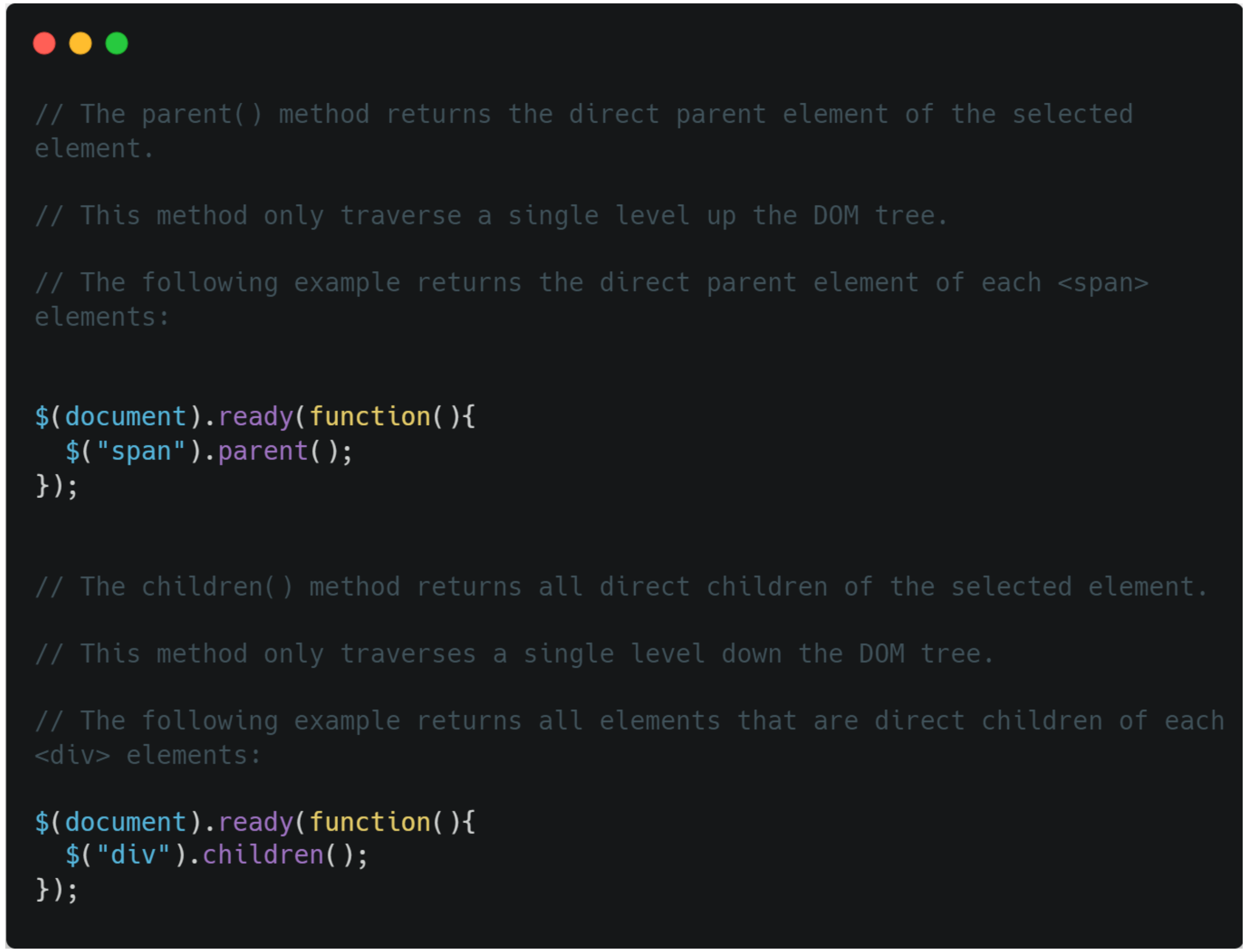
- parent()
- parents()
- parentsUntil()

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

- children()
- find()

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

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



```
// 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 <span>
elements:


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


// 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
<div> elements:

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

JQUERY AJAX

jQuery - AJAX Introduction

What is AJAX?

AJAX = Asynchronous JavaScript and XML.

In short, AJAX is about loading data in the background and display it on the webpage, without reloading the whole page.

AJAX is the art of exchanging data with a server and updating parts of a web page - without reloading the whole page.

jQuery provides several methods for AJAX functionality.

With the jQuery AJAX methods, you can request text, HTML, XML, or JSON from a remote server using both HTTP Get and HTTP Post - And you can load the external data directly into the selected HTML elements of your web page!

Without jQuery, AJAX coding can be a bit tricky!

Writing regular AJAX code can be a bit tricky, because different browsers have different syntax for AJAX implementation. This means that you will have to write extra code to test for different browsers. However, the jQuery team has taken care of this for us, so that we can write AJAX functionality with only one single line of code.

jQuery - AJAX load() Method

The jQuery load() method is a simple, but powerful AJAX method.

The load() method loads data from a server and puts the returned data into the selected element.

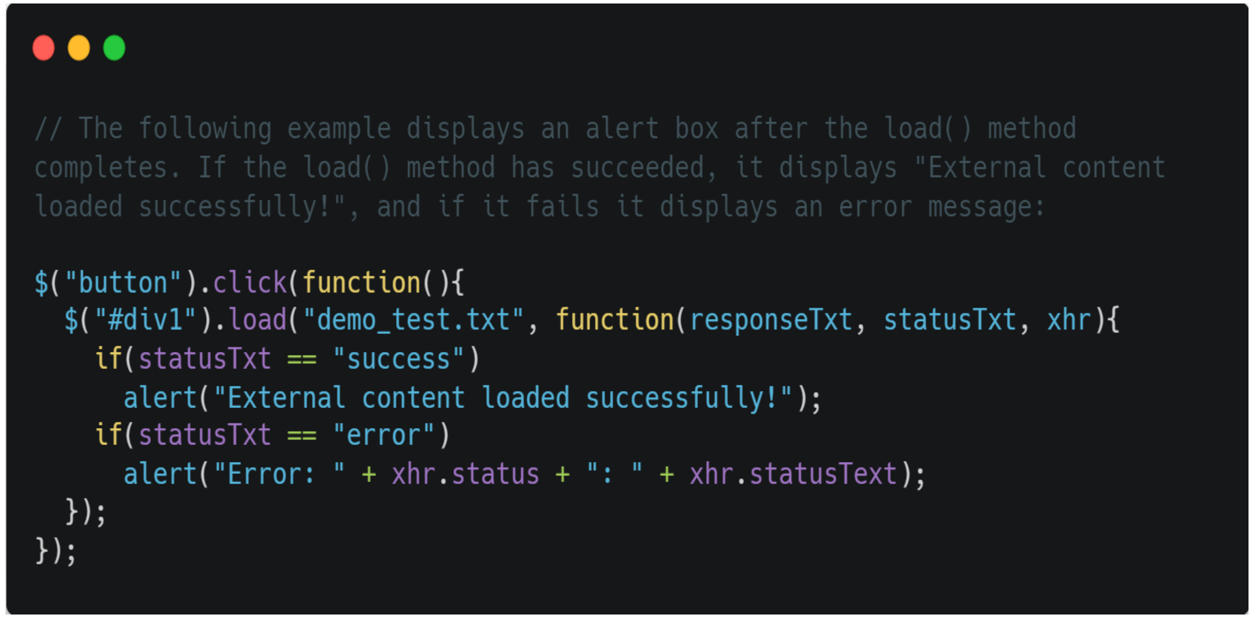
The required URL parameter specifies the URL you wish to load.

The optional data parameter specifies a set of querystring key/value pairs to send along with the request.

The optional callback parameter is the name of a function to be executed after the load() method is completed.

The callback function can have different parameters:

- responseTxt - contains the resulting content if the call succeeds
- statusTxt - contains the status of the call
- xhr - contains the XMLHttpRequest object



```
// The following example displays an alert box after the load() method
completes. If the load() method has succeeded, it displays "External content
loaded successfully!", and if it fails it displays an error message:

$("button").click(function(){
  $("#div1").load("demo_test.txt", function(responseTxt, statusTxt, xhr){
    if(statusTxt == "success")
      alert("External content loaded successfully!");
    if(statusTxt == "error")
      alert("Error: " + xhr.status + ": " + xhr.statusText);
  });
});
```

jQuery - AJAX get() and post() Methods

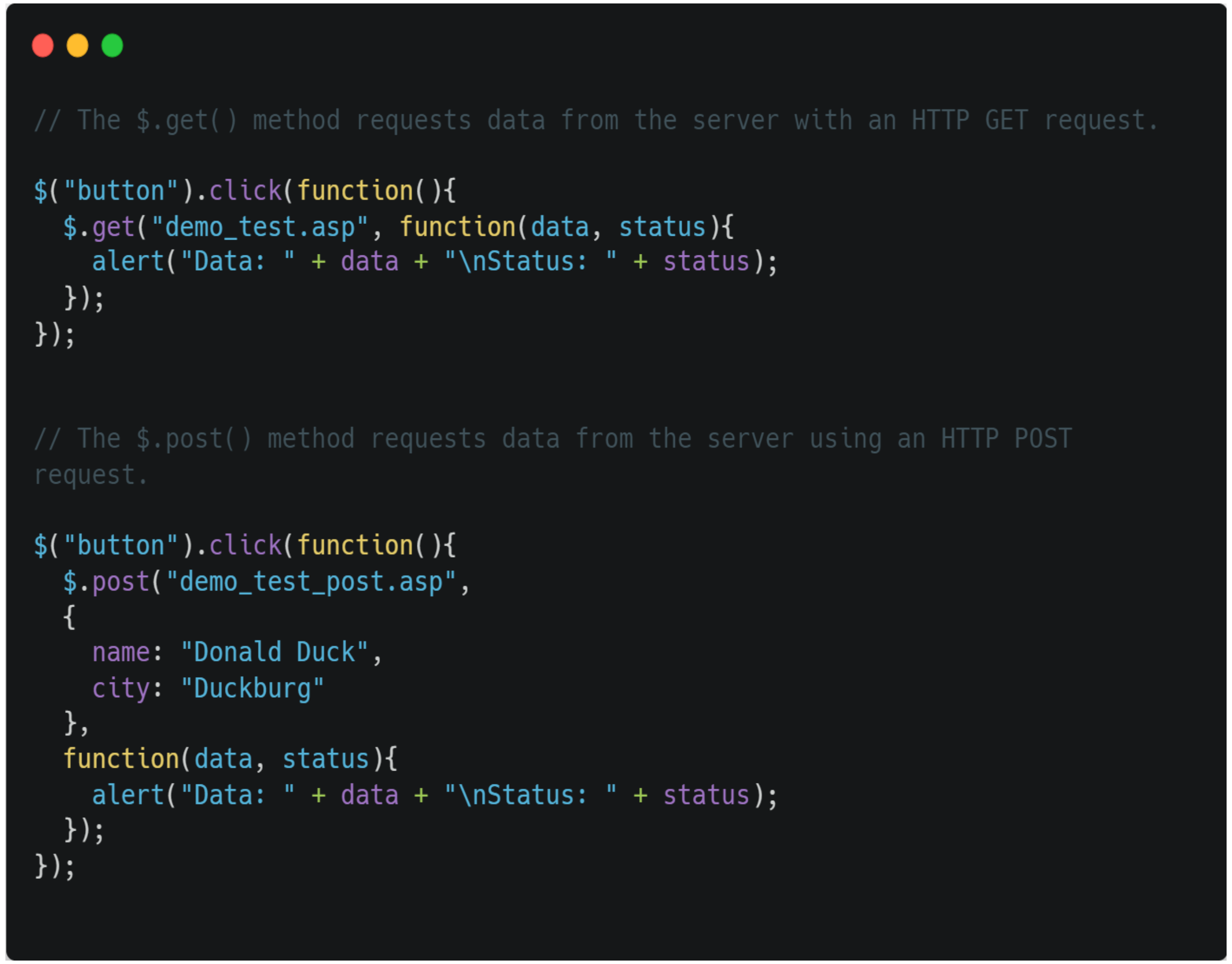
The jQuery get() and post() methods are used to request data from the server with an HTTP **GET** or **POST** request.

Two commonly used methods for a request-response between a client and server are: GET and POST.

- **GET** - Requests data from a specified resource
- **POST** - Submits data to be processed to a specified resource

GET is basically used for just getting (retrieving) some data from the server. Note: The GET method may return cached data.

POST can also be used to get some data from the server. However, the POST method NEVER caches data, and is often used to send data along with the request.



```
// The $.get() method requests data from the server with an HTTP GET request.

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


// The $.post() method requests data from the server using an HTTP POST
request.

$("button").click(function(){
  $.post("demo_test_post.asp",
  {
    name: "Donald Duck",
    city: "Duckburg"
  },
  function(data, status){
    alert("Data: " + data + "\nStatus: " + status);
  });
});
```

THANK YOU!