

# jQuery Tutorial

jQuery is a JavaScript Library.

jQuery greatly simplifies JavaScript programming.

jQuery is easy to learn.

## "Try it Yourself" Examples in Each Chapter

With our online editor, you can edit the code, and click on a button to view the result.

### Example

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

**Click on the "Try it Yourself" button to see how it works.**

[Start learning jQuery now!](#)

## jQuery Examples

Learn by examples! At W3Schools you will find a lot of jQuery examples to edit and test yourself.

[jQuery Examples](#)

## jQuery Quiz Test

Test your jQuery skills at W3Schools!

[jQuery Quiz](#)

## jQuery References

At W3Schools you will find a complete reference of all jQuery selectors, methods, properties and events.

[jQuery Reference](#)

# jQuery Introduction

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

## What You Should Already Know

Before you start studying jQuery, you should have a basic knowledge of:

- HTML
- CSS
- JavaScript

If you want to study these subjects first, find the tutorials on our [Home page](#).

## What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library.

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

jQuery 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.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

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

**Tip:** In addition, jQuery has plugins for almost any task out there.

## Why jQuery?

There are lots of other JavaScript frameworks out there, but jQuery seems to be the most popular, and also the most extendable.

Many of the biggest companies on the Web use jQuery, such as:

- Google
- Microsoft
- IBM
- Netflix

**Will jQuery work in all browsers?** The jQuery team knows all about cross-browser issues, and they have written this knowledge into the jQuery library. jQuery will run exactly the same in all major browsers.

# jQuery Get Started

## 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

## Downloading jQuery

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

**Tip:** Place the downloaded file in the same directory as the pages where you wish to use it.

**Do you wonder why we do not have type="text/javascript" inside the <script> tag?** This is not required in HTML5. JavaScript is the default scripting language in HTML5 and in all modern browsers!

# jQuery CDN

If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).

Both Google and Microsoft host jQuery.

To use jQuery from Google or Microsoft, use one of the following:

## 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>
```

**One big advantage of using the hosted jQuery from Google or Microsoft:** Many users already have downloaded jQuery from Google or Microsoft 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

With jQuery you select (query) HTML elements and perform "actions" on them.

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"`.

**Are you familiar with CSS selectors?** jQuery uses CSS syntax to select elements. You will learn more about the selector syntax in the next chapter of this tutorial.

## The Document Ready Event

You might have noticed that all jQuery methods in our examples, are inside a document ready event:

```
$(document).ready(function(){    // jQuery methods go here... });
```

This is to prevent any jQuery code from running before the document is finished loading (is ready).

It is good practice to wait for the document to be fully loaded and ready before working with it. This also allows you to have your JavaScript code before the body of your document, in the head section.

Here are some examples of actions that can fail if methods are run before the document is fully loaded:

- Trying to hide an element that is not created yet
- Trying to get the size of an image that is not loaded yet

**Tip:** The jQuery team has also created an even shorter method for the document ready event:

```
$(function(){    // jQuery methods go here... });
```

Use the syntax you prefer. We think that the document ready event is easier to understand when reading the code.

# jQuery Selectors

jQuery selectors are one of the most important parts of the jQuery library.

## jQuery Selectors

jQuery selectors allow you to select and manipulate HTML element(s).

jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more. It's based on the existing [CSS Selectors](#), and in addition, it has some own custom selectors.

All selectors in jQuery start with the dollar sign and parentheses: `$(())`.

### 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();
    });
});
```

# The #id Selector

The jQuery #id selector uses the id attribute of an HTML tag to find the specific element. An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element.

To find an element with a specific id, write a hash character, followed by the id of the HTML element:

```
$("#test")
```

## Example

When a user clicks on a button, the element with id="test" will be hidden:

## Example

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

# The .class Selector

The jQuery class selector finds elements with a specific class.

To find elements with a specific class, write a period character, followed by the name of the class:

```
$(".test")
```

## Example

When a user clicks on a button, the elements with class="test" will be hidden:

## Example

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

# More Examples of jQuery Selectors

Syntax	Description	Example
<code>\$("*")</code>	Selects all elements	<a href="#">Try it</a>
<code>\$(this)</code>	Selects the current HTML element	<a href="#">Try it</a>
<code>\$(".p.intro")</code>	Selects all <code>&lt;p&gt;</code> elements with class="intro"	<a href="#">Try it</a>
<code>\$(".p:first")</code>	Selects the first <code>&lt;p&gt;</code> element	<a href="#">Try it</a>
<code>\$(".ul li:first")</code>	Selects the first <code>&lt;li&gt;</code> element of the first <code>&lt;ul&gt;</code>	<a href="#">Try it</a>
<code>\$(".ul li:first-child")</code>	Selects the first <code>&lt;li&gt;</code> element of every <code>&lt;ul&gt;</code>	<a href="#">Try it</a>
<code>\$("[href]")</code>	Selects all elements with an href attribute	<a href="#">Try it</a>
<code>\$(".a[target='_blank']")</code>	Selects all <code>&lt;a&gt;</code> elements with a target attribute value equal to " <code>_blank</code> "	<a href="#">Try it</a>
<code>\$(".a[target!='_blank'])</code>	Selects all <code>&lt;a&gt;</code> elements with a target attribute value NOT equal to " <code>_blank</code> "	<a href="#">Try it</a>
<code>\$(":button")</code>	Selects all <code>&lt;button&gt;</code> elements and <code>&lt;input&gt;</code> elements of type="button"	<a href="#">Try it</a>
<code>\$("tr:even")</code>	Selects all even <code>&lt;tr&gt;</code> elements	<a href="#">Try it</a>
<code>\$("tr:odd")</code>	Selects all odd <code>&lt;tr&gt;</code> elements	<a href="#">Try it</a>

Use our [jQuery Selector Tester](#) to demonstrate the different selectors.

For a complete reference of all the jQuery selectors, please go to our [jQuery Selectors Reference](#).

# Functions In a Separate File

If your website contains a lot of pages, and you want your jQuery functions to be easy to maintain, you can put your jQuery functions in a separate .js file.

When we demonstrate jQuery in this tutorial, the functions are added directly into the <head> section. However, sometimes it is preferable to place them in a separate file, like this (use the src attribute to refer to the .js file):

## Example

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

# jQuery Event Methods

jQuery is tailor-made to respond to events in an HTML page.

## 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".

Here are some common DOM events:

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

## jQuery Syntax For Event Methods

In jQuery, most DOM events have an equivalent jQuery method.

To assign a click event to all paragraphs on a page, you can do this:

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

The next step is to define what should happen when the event fires. You must pass a function to the event:

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

# Commonly Used jQuery Event Methods

## **\$(document).ready()**

The \$(document).ready() method allows us to execute a function when the document is fully loaded. This event is already explained in the [jQuery Syntax](#) chapter.

## **click()**

The click() method attaches an event handler function to an HTML element.

The function is executed when the user clicks on the HTML element.

The following example says: When a click event fires on a <p> element; hide the current <p> element:

### Example

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

## **dblclick()**

The dblclick() method attaches an event handler function to an HTML element.

The function is executed when the user double-clicks on the HTML element:

### Example

```
$( "p" ).dblclick(function(){
    $(this).hide();
});
```

## **mouseenter()**

The mouseenter() method attaches an event handler function to an HTML element.

The function is executed when the mouse pointer enters the HTML element:

### Example

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

## **mouseleave()**

The mouseleave() method attaches an event handler function to an HTML element. The function is executed when the mouse pointer leaves the HTML element:

### Example

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

## **mousedown()**

The mousedown() method attaches an event handler function to an HTML element. The function is executed, when the left, middle or right mouse button is pressed down, while the mouse is over the HTML element:

### Example

```
$("#p1").mousedown(function(){
    alert("Mouse down over p1!");
});
```

## **mouseup()**

The mouseup() method attaches an event handler function to an HTML element. The function is executed, when the left, middle or right mouse button is released, while the mouse is over the HTML element:

### Example

```
$("#p1").mouseup(function(){
    alert("Mouse up over 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.

The function is executed when the form field gets focus:

### Example

```
$(input).focus(function(){
    $(this).css("background-color", "#cccccc");
});
```

## blur()

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

The function is executed when the form field loses focus:

### Example

```
$(input).blur(function(){
    $(this).css("background-color", "#ffffff");
});
```

# The 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();
});
```

Attach multiple event handlers to a <p> element:

## Example

```
$( "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 and Show

Hide, Show, Toggle, Slide, Fade, and Animate. WOW!

Click to show/hide panel

## Examples

[jQuery hide\(\)](#). Demonstrates a simple jQuery hide() method.

[jQuery hide\(\)](#). Another hide() demonstration. How to hide parts of text.

## jQuery hide() and show()

With jQuery, you can hide and show HTML elements with the hide() and show() methods:

### Example

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

$("#show").click(function(){
    $("p").show();
});
```

### Syntax:

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

The optional speed parameter specifies the speed of the hiding/showing, and can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the hide() or show() method completes (you will learn more about callback functions in a later chapter).

The following example demonstrates the speed parameter with hide():

### Example

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

# jQuery 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();
});
```

## Syntax:

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

The optional speed parameter can take the following values: "slow", "fast", or milliseconds.  
The optional callback parameter is a function to be executed after toggle() completes.

# jQuery Effects - Fading

With jQuery you can fade elements in and out of visibility.

Click to fade in/out panel

## Examples

[jQuery fadeIn\(\)](#) Demonstrates the jQuery fadeIn() method.

[jQuery fadeOut\(\)](#) Demonstrates the jQuery fadeOut() method.

[jQuery fadeToggle\(\)](#) Demonstrates the jQuery fadeToggle() method.

[jQuery fadeTo\(\)](#) Demonstrates the jQuery fadeTo() method.

## jQuery Fading Methods

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

jQuery has the following fade methods:

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

## jQuery fadeIn() Method

The jQuery fadeIn() method is used to fade in a hidden element.

### Syntax:

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the fading completes.

The following example demonstrates the fadeIn() method with different parameters:

### Example

```
$("button").click(function(){
    $("#div1").fadeIn();
    $("#div2").fadeIn("slow");
    $("#div3").fadeIn(3000);
});
```

# jQuery fadeOut() Method

The jQuery fadeOut() method is used to fade out a visible element.

## Syntax:

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the fading completes.

The following example demonstrates the fadeOut() method with different parameters:

## Example

```
$("#button").click(function(){
    $("#div1").fadeOut();
    $("#div2").fadeOut("slow");
    $("#div3").fadeOut(3000);
});
```

# jQuery fadeToggle() Method

The jQuery fadeToggle() method toggles between the fadeIn() and fadeOut() methods.

If the elements are faded out, fadeToggle() will fade them in.

If the elements are faded in, fadeToggle() will fade them out.

## Syntax:

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the fading completes.

The following example demonstrates the fadeToggle() method with different parameters:

## Example

```
 $("button").click(function(){
    $("#div1").fadeToggle();
    $("#div2").fadeToggle("slow");
    $("#div3").fadeToggle(3000);
});
```

# jQuery fadeTo() Method

The jQuery fadeTo() method allows fading to a given opacity (value between 0 and 1).

## Syntax:

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

The required speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The required opacity parameter in the fadeTo() method specifies fading to a given opacity (value between 0 and 1).

The optional callback parameter is a function to be executed after the function completes.

The following example demonstrates the fadeTo() method with different parameters:

## Example

```
$("#button").click(function(){
    $("#div1").fadeTo("slow", 0.15);
    $("#div2").fadeTo("slow", 0.4);
    $("#div3").fadeTo("slow", 0.7);
});
```

# jQuery Effects - Sliding

The jQuery slide methods slide elements up and down.

Click to slide down/up the panel

## Examples

[jQuery slideDown\(\)](#) Demonstrates the jQuery slideDown() method.

[jQuery slideUp\(\)](#) Demonstrates the jQuery slideUp() method.

[jQuery slideToggle\(\)](#) Demonstrates the jQuery slideToggle() method.

## jQuery Sliding Methods

With jQuery you can create a sliding effect on elements.

jQuery has the following slide methods:

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

## jQuery slideDown() Method

The jQuery slideDown() method is used to slide down an element.

### Syntax:

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the sliding completes.

The following example demonstrates the slideDown() method:

### Example

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

# jQuery slideUp() Method

The jQuery slideUp() method is used to slide up an element.

## Syntax:

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the sliding completes.

The following example demonstrates the slideUp() method:

## Example

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

# jQuery slideToggle() Method

The jQuery slideToggle() method toggles between the slideDown() and slideUp() methods.

If the elements have been slid down, slideToggle() will slide them up.

If the elements have been slid up, slideToggle() will slide them down.

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

The optional speed parameter can take the following values: "slow", "fast", milliseconds.

The optional callback parameter is a function to be executed after the sliding completes.

The following example demonstrates the slideToggle() method:

## Example

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

# jQuery Effects - Animation

The jQuery animate() method lets you create custom animations.

Start Animation

jQuery

## jQuery Animations - The animate() Method

The jQuery animate() method is used to create custom animations.

### Syntax:

```
$(selector).animate({params}, speed, callback);
```

The required params parameter defines the CSS properties to be animated.

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the animation completes.

The following example demonstrates a simple use of the animate() method; it moves a <div> element to the right, until it has reached a left property of 250px:

### Example

```
 $("button").click(function(){
     $("div").animate({left: '250px'});
});
```

By default, all HTML elements have a static position, and cannot be moved. To manipulate the position, remember to first set the CSS position property of the element to relative, fixed, or absolute!

# jQuery animate() - Manipulate Multiple Properties

Notice that multiple properties can be animated at the same time:

## Example

```
$( "button" ).click( function() {
    $( "div" ).animate({
        left: '250px',
        opacity: '0.5',
        height: '150px',
        width: '150px'
    });
});
```

**Is it possible to manipulate ALL CSS properties with the animate() method?** Yes, almost! However, there is one important thing to remember: all property names must be camel-cased when used with the animate() method: You will need to write paddingLeft instead of padding-left, marginRight instead of margin-right, and so on. Also, color animation is not included in the core jQuery library. If you want to animate color, you need to download the [Color Animations plugin](#) from jQuery.com.

# jQuery animate() - Using Relative Values

It is also possible to define relative values (the value is then relative to the element's current value). This is done by putting += or -= in front of the value:

## Example

```
$( "button" ).click( function() {
    $( "div" ).animate({
        left: '250px',
        height: '+=150px',
        width: '+=150px'
    });
});
```

# jQuery animate() - Using Pre-defined Values

You can even specify a property's animation value as "show", "hide", or "toggle":

## Example

```
$( "button" ).click( function() {
    $( "div" ).animate({
        height: 'toggle'
    });
});
```

# jQuery animate() - Uses Queue Functionality

By default, jQuery comes with queue functionality for animations.

This means that if you write multiple animate() calls after each other, jQuery creates an "internal" queue with these method calls. Then it runs the animate calls ONE by ONE.

So, if you want to perform different animations after each other, we take advantage of the queue functionality:

## Example 1

```
$( "button" ).click( function() {
    var div = $( "div" );
    div.animate({height: '300px', opacity: '0.4'}, "slow");
    div.animate({width: '300px', opacity: '0.8'}, "slow");
    div.animate({height: '100px', opacity: '0.4'}, "slow");
    div.animate({width: '100px', opacity: '0.8'}, "slow");
});
```

The example below first moves the <div> element to the right, and then increases the font size of the text:

## Example 2

```
$( "button" ).click( function() {
    var div = $( "div" );
    div.animate({left: '100px'}, "slow");
    div.animate({fontSize: '3em'}, "slow");
});
```

# jQuery Stop Animations

The jQuery stop() method is used to stop animations or effects before it is finished.

[Start sliding](#)

[Stop sliding](#)

Click to slide down/up the panel

## Examples

[jQuery\\_stop\(\)\\_sliding](#) Demonstrates the jQuery stop() method.

[jQuery\\_stop\(\)\\_animation\\_\(with\\_parameters\)](#) Demonstrates the jQuery stop() method.

## jQuery stop() Method

The jQuery stop() method is used to stop an animation or effect before it is finished.

The stop() method works for all jQuery effect functions, including sliding, fading and custom animations.

### Syntax:

```
$(selector).stop(stopAll,goToEnd);
```

The optional stopAll parameter specifies whether also the animation queue should be cleared or not. Default is false, which means that only the active animation will be stopped, allowing any queued animations to be performed afterwards.

The optional goToEnd parameter specifies whether or not to complete the current animation immediately. Default is false.

So, by default, the stop() method kills the current animation being performed on the selected element.

The following example demonstrates the stop() method, with no parameters:

### Example

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

## jQuery Effects Reference

For a complete overview of all jQuery effects, please go to our [jQuery Effect Reference](#).

# jQuery Callback Functions

A callback function is executed after the current effect is 100% finished.

## jQuery Callback Functions

JavaScript statements are executed line by line. However, with effects, the next line of code can be run even though the effect is not finished. This can create errors.

To prevent this, you can create a callback function.

A callback function is executed after the current effect is finished.

Typical syntax: **`$(selector).hide(speed,callback);`**

### Examples

The example below has a callback parameter that is a function that will be executed after the hide effect is completed:

#### Example with Callback

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

The example below has no callback parameter, and the alert box will be displayed before the hide effect is completed:

#### Example without Callback

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

# jQuery - Chaining

With jQuery, you can chain together actions/methods.

Chaining allows us to run multiple jQuery methods (on the same element) within a single statement.

## jQuery Method Chaining

Until now we have been writing jQuery statements one at a time (one after the other). However, there is a technique called chaining, that allows us to run multiple jQuery commands, one after the other, on the same element(s).

**Tip:** This way, browsers do not have to find the same element(s) more than once.

To chain an action, you simply append the action to the previous action.

The following example chains together the `css()`, `slideUp()`, and `slideDown()` methods. The "p1" element first changes to red, then it slides up, and then it slides down:

### Example

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

We could also have added more method calls if needed.

**Tip:** When chaining, the line of code could become quite long. However, jQuery is not very strict on the syntax; you can format it like you want, including line breaks and indentations. This also works just fine:

### Example

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

jQuery throws away extra whitespace and executes the lines above as one long line of code.

# jQuery - Get Content and Attributes

jQuery contains powerful methods for changing and manipulating HTML elements and attributes.

## 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.

**DOM = Document Object Model** The DOM defines a standard for accessing HTML and XML documents: *"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*

## 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

The following example demonstrates how to get content with the jQuery `text()` and `html()` methods:

### Example

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

The following example demonstrates how to get the value of an input field with the jQuery val() method:

## Example

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

## Get Attributes - attr()

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"));
});
```

The next chapter explains how to set (change) content and attribute values.

# jQuery - Set Content and Attributes

## Set Content - text(), html(), and val()

We will use the same three methods from the previous page to **set content**:

- `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

The following example demonstrates how to set content with the jQuery `text()`, `html()`, and `val()` methods:

### Example

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

# A Callback Function for text(), html(), and val()

All of the three jQuery methods above: text(), html(), and val(), also come with a callback function. The callback function has two parameters: the index of the current element in the list of elements selected and the original (old) value. You then return the string you wish to use as the new value from the function.

The following example demonstrates text() and html() with a callback function:

## Example

```
$("#btn1").click(function(){
    $("#test1").text(function(i, origText){
        return "Old text: " + origText + " New text: Hello world!
        (index: " + i + ")";
    });
});

$("#btn2").click(function(){
    $("#test2").html(function(i, origText){
        return "Old html: " + origText + " New html: Hello <b>world!</b>
        (index: " + i + ")";
    });
});
```

## Set Attributes - attr()

The jQuery attr() method is also used to set/change attribute values.

The following example demonstrates how to change (set) the value of the href attribute in a link:

## Example

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

The attr() method also allows you to set multiple attributes at the same time.

The following example demonstrates how to set both the href and title attributes at the same time:

## Example

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

## A Callback Function for attr()

The jQuery method attr(), also comes with a callback function. The callback function has two parameters: the index of the current element in the list of elements selected and the original (old) attribute value. You then return the string you wish to use as the new attribute value from the function.

The following example demonstrates attr() with a callback function:

## Example

```
$( "button" ).click(function(){
  $( "#w3s" ).attr( "href", function(i, origValue){
    return origValue + "/jquery/";
  });
});
```

# jQuery - Add Elements

With jQuery, it is easy to add new elements/content.

## Add New HTML Content

We will look at four jQuery methods that are used to add new content:

- `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

### jQuery append() Method

The jQuery `append()` method inserts content AT THE END of the selected HTML elements.

#### Example

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

### jQuery prepend() Method

The jQuery `prepend()` method inserts content AT THE BEGINNING of the selected HTML elements.

#### Example

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

# Add Several New Elements With append() and prepend()

In both examples above, we have only inserted some text/HTML at the beginning/end of the selected HTML elements.

However, both the append() and prepend() methods can take an infinite number of new elements as parameters. The new elements can be generated with text/HTML (like we have done in the examples above), with jQuery, or with JavaScript code and DOM elements.

In the following example, we create several new elements. The elements are created with text/HTML, jQuery, and JavaScript/DOM. Then we append the new elements to the text with the append() method (this would have worked for prepend() too) :

## Example

```
function appendText() {  
    var txt1 = "<p>Text.</p>";           // Create element with HTML  
    var txt2 = $("<p></p>").text("Text."); // Create with jQuery  
    var txt3 = document.createElement("p"); // Create with DOM  
    txt3.innerHTML = "Text.";  
    $("body").append(txt1, txt2, txt3);     // Append the new elements  
}
```

# jQuery after() and before() Methods

The jQuery after() method inserts content AFTER the selected HTML elements.

The jQuery before() method inserts content BEFORE the selected HTML elements.

## Example

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

# Add Several New Elements With after() and before()

Also, both the after() and before() methods can take an infinite number of new elements as parameters. The new elements can be generated with text/HTML (like we have done in the example above), with jQuery, or with JavaScript code and DOM elements.

In the following example, we create several new elements. The elements are created with text/HTML, jQuery, and JavaScript/DOM. Then we insert the new elements to the text with the after() method (this would have worked for before() too) :

## Example

```
function afterText() {
    var txt1 = "<b>I </b>";                                // Create element with HTML
    var txt2 = $("<i></i>").text("love ");                // Create with jQuery
    var txt3 = document.createElement("b");                  // Create with DOM
    txt3.innerHTML = "jQuery!";
    $("img").after(txt1, txt2, txt3);                      // Insert new elements after
<img>
}
```

# jQuery - Remove Elements

With jQuery, it is easy to remove existing HTML elements.

## Remove Elements/Content

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

### jQuery `remove()` Method

The jQuery `remove()` method removes the selected element(s) and its child elements.

#### Example

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

### jQuery `empty()` Method

The jQuery `empty()` method removes the child elements of the selected element(s).

#### Example

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

# Filter the Elements to be Removed

The jQuery remove() method also accepts one parameter, which allows you to filter the elements to be removed.

The parameter can be any of the jQuery selector syntaxes.

The following example removes all <p> elements with class="test":

## Example

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

This example removes all <p> elements with class="test" or class="demo":

## Example

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

# jQuery - Get and Set CSS Classes

With jQuery, it is easy to manipulate the CSS of elements.

[Toggle class](#)

## jQuery Manipulating CSS

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

## Example Stylesheet

The following stylesheet will be used for all the examples on this page:

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

## jQuery addClass() Method

The following example shows how to add class attributes to different elements. Of course you can select multiple elements, when adding classes:

### Example

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

You can also specify multiple classes within the addClass() method:

## Example

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

## jQuery removeClass() Method

The following example shows how to remove a specific class attribute from different elements:

## Example

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

## jQuery toggleClass() Method

The following example will show how to use the jQuery toggleClass() method. This method toggles between adding/removing classes from the selected elements:

## Example

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

## jQuery css() Method

The jQuery css() method will be explained in the next chapter.

# jQuery - css() Method

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

## Return a CSS 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" );
```

## Set a CSS 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" );
```

# Set Multiple CSS Properties

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 - Dimensions

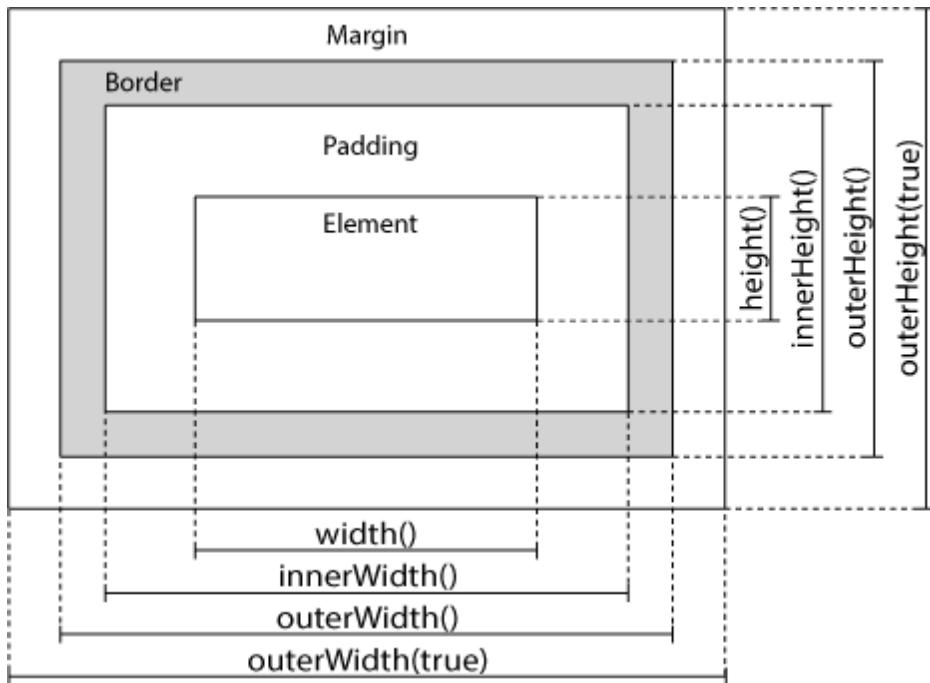
With jQuery, it is easy to work with the dimensions of elements and browser window.

## jQuery Dimension Methods

jQuery has several important methods for working with dimensions:

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

## jQuery Dimensions



# jQuery width() and height() Methods

The width() method sets or returns the width of an element (excludes padding, border and margin).

The height() method sets or returns the height of an element (excludes padding, border and margin).

The following example returns the width and height of a specified <div> element:

## Example

```
$( "button" ).click( function() {
    var txt = "";
    txt += "Width: " + $( "#div1" ).width() + "<br>";
    txt += "Height: " + $( "#div1" ).height();
    $( "#div1" ).html( txt );
});
```

# jQuery innerWidth() and innerHeight() Methods

The innerWidth() method returns the width of an element (includes padding).

The innerHeight() method returns the height of an element (includes padding).

The following example returns the inner-width/height of a specified <div> element:

## Example

```
$( "button" ).click( function() {
    var txt = "";
    txt += "Inner width: " + $( "#div1" ).innerWidth() + "<br>";
    txt += "Inner height: " + $( "#div1" ).innerHeight();
    $( "#div1" ).html( txt );
});
```

# jQuery outerWidth() and outerHeight() Methods

The outerWidth() method returns the width of an element (includes padding and border).  
The outerHeight() method returns the height of an element (includes padding and border).  
The following example returns the outer-width/height of a specified <div> element:

## Example

```
$( "button" ).click( function() {
    var txt = "";
    txt += "Outer width: " + $( "#div1" ).outerWidth() + "<br>";
    txt += "Outer height: " + $( "#div1" ).outerHeight();
    $( "#div1" ).html( txt );
});
```

The outerWidth(true) method returns the width of an element (includes padding, border, and margin).

The outerHeight(true) method returns the height of an element (includes padding, border, and margin).

## Example

```
$( "button" ).click( function() {
    var txt = "";
    txt += "Outer width (+margin): " + $( "#div1" ).outerWidth( true ) + "
<br>";
    txt += "Outer height (+margin): " + $( "#div1" ).outerHeight( true );
    $( "#div1" ).html( txt );
});
```

# jQuery More width() and height()

The following example returns the width and height of the document (the HTML document) and window (the browser viewport):

## Example

```
$(“button”).click(function(){
    var txt = “”;
    txt += “Document width/height: ” + $(document).width();
    txt += “x” + $(document).height() + “\n”;
    txt += “Window width/height: ” + $(window).width();
    txt += “x” + $(window).height();
    alert(txt);
});
```

The following example sets the width and height of a specified <div> element:

## Example

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

# jQuery Traversing

## What is 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.

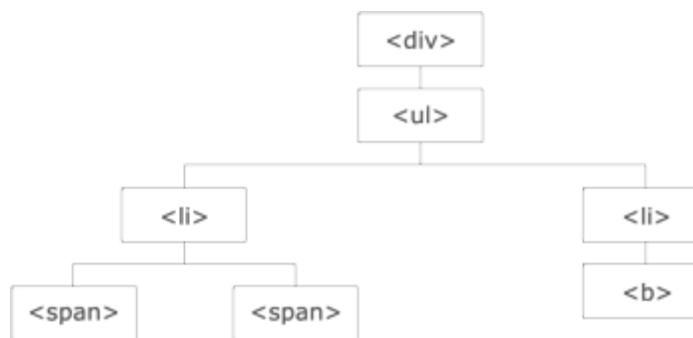


Illustration explained:

- 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>`

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

## Traversing the DOM

jQuery provides a variety of methods that allow us to traverse the DOM.

The largest category of traversal methods are tree-traversal.

The next chapters will show us how to travel up, down and sideways in the DOM tree.

## jQuery Traversing Reference

For a complete overview of all jQuery Traversing methods, please go to our [jQuery Traversing Reference](#).

# jQuery Traversing - Ancestors

An ancestor is a parent, grandparent, great-grandparent, and so on.

With jQuery you can traverse up the DOM tree to find ancestors of an element.

## Traversing Up the DOM Tree

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

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

### jQuery `parent()` Method

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:

#### Example

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

### jQuery `parents()` Method

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 `<span>` elements:

#### Example

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

You can also use an optional parameter to filter the search for ancestors.

The following example returns all ancestors of all `<span>` elements that are `<ul>` elements:

## Example

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

## jQuery parentsUntil() Method

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

The following example returns all ancestor elements between a `<span>` and a `<div>` element:

## Example

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

# jQuery Traversing - Descendants

A descendant is a child, grandchild, great-grandchild, and so on.

With jQuery you can traverse down the DOM tree to find descendants of an element.

## Traversing Down the DOM Tree

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

- `children()`
- `find()`

### jQuery `children()` Method

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:

#### Example

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

You can also use an optional parameter to filter the search for children.

The following example returns all `<p>` elements with the class name "first", that are direct children of `<div>`:

#### Example

```
$(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 `<span>` elements that are descendants of `<div>`:

## Example

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

The following example returns all descendants of `<div>`:

## Example

```
$(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()`

### jQuery `siblings()` Method

The `siblings()` method returns all sibling elements of the selected element.

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

#### Example

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

You can also use an optional parameter to filter the search for siblings.

The following example returns all sibling elements of `<h2>` that are `<p>` elements:

#### Example

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

# jQuery next() Method

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();
});
```

# jQuery nextUntil() Method

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");
});
```

# jQuery prev(), prevAll() & prevUntil() Methods

The `prev()`, `prevAll()` and `prevUntil()` methods work just like the methods above but with reverse functionality: they return previous sibling elements (traverse backwards along sibling elements in the DOM tree, instead of forward).

# jQuery Traversing - Filtering

## The first(), last(), eq(), filter() and not() Methods

The most basic filtering methods are first(), last() and eq(), which allow you to select a specific element based on its position in a group of elements.

Other filtering methods, like filter() and not() allow you to select elements that match, or do not match, a certain criteria.

### jQuery first() Method

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

The following example selects the first <div> element:

#### Example

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

### jQuery last() Method

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

The following example selects the last <div> element:

#### Example

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

# jQuery eq() method

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);
});
```

# jQuery filter() Method

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");
});
```

# jQuery not() Method

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");
});
```

# jQuery - AJAX Introduction

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

## jQuery AJAX Example

### Let jQuery AJAX Change This Text

[Get External Content](#)

## 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.

Examples of applications using AJAX: Gmail, Google Maps, Youtube, and Facebook tabs.

You can learn more about AJAX in our [AJAX tutorial](#).

## What About jQuery and AJAX?

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 Methods

In the next chapters we will look at the most important jQuery AJAX methods.

# 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.

## Syntax:

```
$(selector).load(URL,data,callback);
```

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.

**Here is the content of our example file: "demo\_test.txt":**

```
<h2>jQuery and AJAX is FUN!!!</h2> <p id="p1">This is some text in a paragraph.</p>
```

The following example loads the content of the file "demo\_test.txt" into a specific <div> element:

## Example

```
$("#div1").load("demo_test.txt");
```

It is also possible to add a jQuery selector to the URL parameter.

The following example loads the content of the element with id="p1", inside the file "demo\_test.txt", into a specific <div> element:

## Example

```
$("#div1").load("demo_test.txt #p1");
```

The optional callback parameter specifies a callback function to run when 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:

## Example

```
$( "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 Reference

For a complete overview of all jQuery AJAX methods, please go to our [jQuery AJAX Reference](#).

# 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.

## HTTP Request: GET vs. POST

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.

To learn more about GET and POST, and the differences between the two methods, please read our [HTTP Methods GET vs POST](#) chapter.

## jQuery \$.get() Method

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

### Syntax:

```
$.get(URL, callback);
```

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

The optional callback parameter is the name of a function to be executed if the request succeeds.

The following example uses the \$.get() method to retrieve data from a file on the server:

### Example

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

The first parameter of \$.get() is the URL we wish to request ("demo\_test.asp").

The second parameter is a callback function. The first callback parameter holds the content of the page requested, and the second callback parameter holds the status of the request.

**Tip:** Here is how the ASP file looks like ("demo\_test.asp"):

```
<% response.write("This is some text from an external ASP file.") %>
```

## jQuery \$.post() Method

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

### Syntax:

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

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

The optional data parameter specifies some data to send along with the request.

The optional callback parameter is the name of a function to be executed if the request succeeds.

The following example uses the \$.post() method to send some data along with the request:

## Example

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

The first parameter of \$.post() is the URL we wish to request ("demo\_test\_post.asp").

Then we pass in some data to send along with the request (name and city).

The ASP script in "demo\_test\_post.asp" reads the parameters, processes them, and returns a result.

The third parameter is a callback function. The first callback parameter holds the content of the page requested, and the second callback parameter holds the status of the request.

**Tip:** Here is how the ASP file looks like ("demo\_test\_post.asp"):

```
<% dim fname,city
fname=Request.Form("name") city=Request.Form("city")
Response.Write("Dear " & fname & ". ") Response.Write("Hope you live well in "
& city & ".") %>
```

# jQuery - The noConflict() Method

What if you wish to use other frameworks on your pages, while still using jQuery?

## jQuery and Other JavaScript Frameworks

As you already know; jQuery uses the **\$** sign as a shortcut for jQuery.

There are many other popular JavaScript frameworks like: Angular, Backbone, Ember, Knockout, and more.

### **What if other JavaScript frameworks also use the \$ sign as a shortcut?**

If two different frameworks are using the same shortcut, one of them might stop working.

The jQuery team have already thought about this, and implemented the noConflict() method.

## The jQuery noConflict() Method

The noConflict() method releases the hold on the **\$** shortcut identifier, so that other scripts can use it.

You can of course still use jQuery, simply by writing the full name instead of the shortcut:

### Example

```
$.noConflict();
jQuery(document).ready(function(){
    jQuery("button").click(function(){
        jQuery("p").text("jQuery is still working!");
    });
});
```

You can also create your own shortcut very easily. The noConflict() method returns a reference to jQuery, that you can save in a variable, for later use. Here is an example:

### Example

```
var jq = $.noConflict();
jq(document).ready(function(){
    jq("button").click(function(){
        jq("p").text("jQuery is still working!");
    });
});
```

If you have a block of jQuery code which uses the \$ shortcut and you do not want to change it all, you can pass the \$ sign in as a parameter to the ready method. This allows you to access jQuery using \$, inside this function - outside of it, you will have to use "jQuery":

## Example

```
$ .noConflict();  
jQuery(document).ready(function($){  
    $("button").click(function(){  
        $("p").text("jQuery is still working!");  
    });  
});
```

## jQuery Misc Reference

For a complete overview of all jQuery Misc methods, please go to our [jQuery Misc Reference](#).

# jQuery - Filters

## jQuery Filters

Use jQuery to filter/search for specific elements.

### Filter Tables

Perform a case-insensitive search for items in a table:

#### Example

Type something in the input field to search the table for first names, last names or emails:

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@mail.com
July	Dooley	july@greatstuff.com
Anja	Ravendale	a_r@test.com

#### jQuery

```
<script>
$(document).ready(function(){
  $("#myInput").on("keyup", function() {
    var value = $(this).val().toLowerCase();
    $("#myTable tr").filter(function() {
      $(this).toggle($(this).text().toLowerCase().indexOf(value) > -1)
    });
  });
</script>
```

**Example explained:** We use jQuery to loop through each table rows to check if there are any text values that matches the value of the input field. The `toggle` method hides the row (`display:none`) that does not match the search. We use the `toLowerCase()` DOM method to convert the text to lower case, which makes the search case insensitive (allows "john", "John", and even "JOHN" on search).

## Filter Lists

Perform a case-insensitive search for items in a list:

### Example

Type something in the input field to search the list for items:

- First item
- Second item
- Third item
- Fourth

## Filter Anything

Perform a case-insensitive search for text inside a div element:

### Example

I am a paragraph.

I am a div element inside div.

Another paragraph.

# jQuery Selectors

## jQuery Selectors

Use our [jQuery Selector Tester](#) to demonstrate the different selectors.

Selector	Example	Selects
<u>*</u>	<code>\$('*')</code>	All elements
<u>#id</u>	<code>\$("#lastname")</code>	The element with id="lastname"
<u>.class</u>	<code>\$(".intro")</code>	All elements with class="intro"
<u>.class,.class</u>	<code>(".intro,.demo")</code>	All elements with the class "intro" or "demo"
<u>element</u>	<code>("p")</code>	All <p> elements
<u>e1,e2,e3</u>	<code>("h1,div,p")</code>	All <h1>, <div> and <p> elements
<u>:first</u>	<code>("p:first")</code>	The first <p> element
<u>:last</u>	<code>("p:last")</code>	The last <p> element
<u>:even</u>	<code>("tr:even")</code>	All even <tr> elements
<u>:odd</u>	<code>("tr:odd")</code>	All odd <tr> elements
<u>:first-child</u>	<code>("p:first-child")</code>	All <p> elements that are the first child of their parent
<u>:first-of-type</u>	<code>("p:first-of-type")</code>	All <p> elements that are the first <p> element of their parent
<u>:last-child</u>	<code>("p:last-child")</code>	All <p> elements that are the last child of their parent
<u>:last-of-type</u>	<code>("p:last-of-type")</code>	All <p> elements that are the last <p> element of their parent
<u>:nth-child(n)</u>	<code>("p:nth-child(2)")</code>	All <p> elements that are the 2nd child of their parent
<u>:nth-last-child(n)</u>	<code>("p:nth-last-child(2)")</code>	All <p> elements that are the 2nd child of their parent, counting from the last child

Selector	Example	Selects
<u>:nth-of-type(n)</u>	<code>\$(“p:nth-of-type(2)”) </code>	All <code>&lt;p&gt;</code> elements that are the 2nd <code>&lt;p&gt;</code> element of their parent
<u>:nth-last-of-type(n)</u>	<code>\$(“p:nth-last-of-type(2)”) </code>	All <code>&lt;p&gt;</code> elements that are the 2nd <code>&lt;p&gt;</code> element of their parent, counting from the last child
<u>:only-child</u>	<code>\$(“p:only-child”) </code>	All <code>&lt;p&gt;</code> elements that are the only child of their parent
<u>:only-of-type</u>	<code>\$(“p:only-of-type”) </code>	All <code>&lt;p&gt;</code> elements that are the only child, of its type, of their parent
<hr/>		
<u>parent &gt; child</u>	<code>\$(“div &gt; p”) </code>	All <code>&lt;p&gt;</code> elements that are a direct child of a <code>&lt;div&gt;</code> element
<u>parent descendant</u>	<code>\$(“div p”) </code>	All <code>&lt;p&gt;</code> elements that are descendants of a <code>&lt;div&gt;</code> element
<u>element + next</u>	<code>\$(“div + p”) </code>	The <code>&lt;p&gt;</code> element that are next to each <code>&lt;div&gt;</code> elements
<u>element ~ siblings</u>	<code>\$(“div ~ p”) </code>	All <code>&lt;p&gt;</code> elements that are siblings of a <code>&lt;div&gt;</code> element
<hr/>		
<u>:eq(index)</u>	<code>\$(“ul li:eq(3)”) </code>	The fourth element in a list (index starts at 0)
<u>:gt(no)</u>	<code>\$(“ul li:gt(3)”) </code>	List elements with an index greater than 3
<u>:lt(no)</u>	<code>\$(“ul li:lt(3)”) </code>	List elements with an index less than 3
<u>:not(selector)</u>	<code>\$(“input:not(:empty)”) </code>	All input elements that are not empty
<hr/>		
<u>:header</u>	<code>\$(“:header”) </code>	All header elements <code>&lt;h1&gt;</code> , <code>&lt;h2&gt;</code> ...
<u>:animated</u>	<code>\$(“:animated”) </code>	All animated elements
<u>:focus</u>	<code>\$(“:focus”) </code>	The element that currently has focus
<u>:contains(text)</u>	<code>\$(“:contains('Hello')”) </code>	All elements which contains the text "Hello"
<u>:has(selector)</u>	<code>\$(“div:has(p)”) </code>	All <code>&lt;div&gt;</code> elements that have a <code>&lt;p&gt;</code> element

Selector	Example	Selects
<u>:empty</u>	<code>\$(":empty")</code>	All elements that are empty
<u>:parent</u>	<code">(":parent")</code">	All elements that are a parent of another element
<u>:hidden</u>	<code>("p:hidden")</code>	All hidden <p> elements
<u>:visible</u>	<code>("table:visible")</code>	All visible tables
<u>:root</u>	<code>(":root")</code>	The document's root element
<u>:lang(<i>language</i>)</u>	<code>("p:lang(de)")</code>	All <p> elements with a lang attribute value starting with "de"
<hr/>		
<u>[attribute]</u>	<code>("[href]")</code>	All elements with a href attribute
<u>[attribute=value]</u>	<code>("[href='default.htm'])</code>	All elements with a href attribute value equal to "default.htm"
<u>[attribute!=value]</u>	<code>("[href!='default.htm'])</code>	All elements with a href attribute value not equal to "default.htm"
<u>[attribute\$=value]</u>	<code>("[href\$='.jpg'])</code>	All elements with a href attribute value ending with ".jpg"
<u>[attribute =value]</u>	<code>("[title ='Tomorrow'])</code>	All elements with a title attribute value equal to 'Tomorrow', or starting with 'Tomorrow' followed by a hyphen
<u>[attribute^=value]</u>	<code>("[title^='Tom'])</code>	All elements with a title attribute value starting with "Tom"
<u>[attribute~价值]</u>	<code>("[title~='hello'])</code>	All elements with a title attribute value containing the specific word "hello"
<u>[attribute*=value]</u>	<code>("[title*='hello'])</code>	All elements with a title attribute value containing the word "hello"
<hr/>		
<u>:input</u>	<code">(":input")</code">	All input elements
<u>:text</u>	<code">(":text")</code">	All input elements with type="text"
<u>:password</u>	<code">(":password")</code">	All input elements with type="password"
<u>:radio</u>	<code">(":radio")</code">	All input elements with type="radio"

<b>Selector</b>	<b>Example</b>	<b>Selects</b>
<u>:checkbox</u>	<code>\$("#:checkbox")</code>	All input elements with type="checkbox"
<u>:submit</u>	<code>\$("#:submit")</code>	All input elements with type="submit"
<u>:reset</u>	<code>\$("#:reset")</code>	All input elements with type="reset"
<u>:button</u>	<code>\$("#:button")</code>	All input elements with type="button"
<u>:image</u>	<code>\$("#:image")</code>	All input elements with type="image"
<u>:file</u>	<code>\$("#:file")</code>	All input elements with type="file"
<u>:enabled</u>	<code>\$("#:enabled")</code>	All enabled input elements
<u>:disabled</u>	<code>\$("#:disabled")</code>	All disabled input elements
<u>:selected</u>	<code>\$("#:selected")</code>	All selected input elements
<u>:checked</u>	<code>\$("#:checked")</code>	All checked input elements

# jQuery \* Selector

## Example

Select all elements inside the document:

```
$( "*" )
```

## Definition and Usage

The \* selector selects all elements in the document, including html, head and body.

If the \* selector is used together with another element, it selects all child elements within the specified element.

**Tip:** The \* selector can be heavy to process for some browsers.

## Syntax

```
$( "*" )
```

## Try it Yourself - Examples

Select all elements inside <body> Use the \* selector to select all elements inside the <body> element.

# jQuery #id Selector

## Example

Select the element with the id "intro":

```
$("#intro")
```

## Definition and Usage

The #id selector selects the element with the specific id.

The id refers to the id attribute of an HTML element.

**Note:** The id attribute must be unique within a document.

**Note:** Do not start an id attribute with a number. It may cause problems in some browsers.

## Syntax

```
$("#id")
```

Parameter	Description
<i>id</i>	Required. Specifies the id of the element to select

# jQuery .class Selector

## Example

Select all elements with class "intro":

```
$(".intro")
```

## Definition and Usage

The .class selector selects all elements with the specific class.

The class refers to the class attribute of an HTML element.

The class attribute is used to set a particular style for several HTML elements.

**Note:** Do not start a class attribute with a number. It may cause problems in some browsers.

## Syntax

```
$(".class")
```

Parameter	Description
<i>class</i>	Required. Specifies the class of the elements to select

## Try it Yourself - Examples

Select all p elements with class "intro" How to select all p elements with class "intro".

# jQuery multiple classes Selector

## Example

Select all elements with class "intro", "demo" or "end":

```
$(".intro, .demo, .end")
```

## Definition and Usage

The .class selector can also be used to select multiple classes.

**Note:** Separate each class with a comma.

**Note:** Do not start a class attribute with a number. It may cause problems in some browsers.

## Syntax

```
$(".class1,.class2,.class3,...")
```

Parameter	Description
<i>class</i>	Required. Specifies the class of the elements to select

# jQuery element Selector

## Example

Select all <p> elements:

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

## Definition and Usage

The element selector selects all elements with the specific element name.

## Syntax

```
$( "element" )
```

Parameter	Description
<i>element</i>	Required. Specifies the element to select

## Try it Yourself - Examples

Select all p elements with class "intro" How to select all p elements with class "intro".

# jQuery multiple elements Selector

## Example

Select all `<h2>`, `<div>` and `<span>` elements:

```
$( "h2, div, span" )
```

## Definition and Usage

The element selector can also be used to select multiple elements.

**Note:** Separate each element with a comma.

## Syntax

```
$( "element1,element2,element3,... " )
```

Parameter	Description
<code>element</code>	Required. Specifies the elements to be selected

## Try it Yourself - Examples

Select all elements Use the `*` selector to select all elements in a document.

# jQuery :first Selector

## Example

Select the first <p> element:

```
$(“p:first”)
```

## Definition and Usage

The :first selector selects the first element.

**Note:** This selector can only select one single element. Use the :first-child selector to select more than one element (one for each parent).

This is mostly used together with another selector to select the first element in a group (like in the example above).

**Tip:** To select the last element in a group, use the :last selector.

## Syntax

```
$(“:first”)
```

## Try it Yourself - Examples

Difference between :first and :first-child Show the difference between the :first and :first-child selectors.

# jQuery :last Selector

## Example

Select the last <p> element:

```
$( "p:last" )
```

## Definition and Usage

The :last selector selects the last element.

**Note:** This selector can only select one single element. Use the [:last-child](#) selector to select more than one element (one for each parent).

This is mostly used together with another selector to select the last element in a group (like in the example above).

**Tip:** To select the first element in a group, use the [:first](#) selector.

## Syntax

```
$( ":last" )
```

## Try it Yourself - Examples

[Difference between :last and :last-child](#) Show the difference between the :last and :last-child selectors.

# jQuery :even Selector

## Example

Select every other (even) <tr> element:

```
$(“tr:even”)
```

## Definition and Usage

The :even selector selects each element with an even index number (like: 0, 2, 4, etc.).

The index numbers start at 0.

This is mostly used together with another selector to select every even indexed element in a group (like in the example above).

**Tip:** Use the :odd selector to select elements with odd index numbers.

## Syntax

```
$(“:even”)
```

# jQuery :odd Selector

## Example

Select every other (odd) <tr> element:

```
$( "tr:odd" )
```

## Definition and Usage

The :odd selector selects each element with an odd index number (like: 1, 3, 5, etc.).

The index numbers start at 0.

This is mostly used together with another selector to select every odd indexed element in a group (like in the example above).

**Tip:** Use the :even selector to select elements with even index numbers.

## Syntax

```
$( ":odd" )
```

# jQuery :first-child Selector

## Example

Select every `<p>` element that is the first child of its parent:

```
$(“p:first-child”)
```

## Definition and Usage

The `:first-child` selector selects all elements that are the first child of their parent.

**Tip:** Use the `:last-child` selector to select elements that are the last child of their parent.

## Syntax

```
$(“:first-child”)
```

## Try it Yourself - Examples

[Select the first `<p>` element of all `<div>` elements](#) How to select the first `<p>` element of all `<div>` elements.

[Difference between `:first` and `:first-child`](#) Show the difference between the `:first` and `:first-child` selectors.

[Difference between `:first`, `:first-child` and `:first-of-type`](#) Show the difference between the `:first`, `:first-child` and `:first-of-type` selectors.

# jQuery :first-of-type Selector

## Example

Select every `<p>` element that is the first `<p>` element of its parent:

```
$(“p:first-of-type”)
```

## Definition and Usage

The `:first-of-type` selector selects all elements that are the first child, of a particular type, of their parent.

**Tip:** This is the same as `:nth-of-type(1)`.

**Tip:** Use the `:last-of-type` selector to select all elements that are the last child, of a particular type, of their parent.

## Syntax

```
$(“:first-of-type”)
```

## Try it Yourself - Examples

Select the first `<p>` element of all `<div>` elements How to select the first `<p>` element of all `<div>` elements.

Difference between `:first`, `:first-child` and `:first-of-type` Show the difference between the `:first`, `:first-child` and `:first-of-type` selectors.

# jQuery :last-child Selector

## Example

Select every `<p>` element that is the last child of its parent:

```
$(“p:last-child”)
```

## Definition and Usage

The `:last-child` selector selects all elements that are the last child of their parent.

**Tip:** Use the `:first-child` selector to select elements that are the first child of their parent.

## Syntax

```
$(“:last-child”)
```

## Try it Yourself - Examples

[Select the last `<p>` element of all `<div>` elements](#) How to select the last `<p>` element of all `<div>` elements.

[Difference between `:last` and `:last-child`](#) Show the difference between the `:last` and `:last-child` selectors.

[Difference between `:last`, `:last-child` and `:last-of-type`](#) Show the difference between the `:last`, `:last-child` and `:last-of-type` selectors.

# jQuery :last-of-type Selector

## Example

Select every `<p>` element that is the last `<p>` element of its parent:

```
$( "p:last-of-type" )
```

## Definition and Usage

The `:last-of-type` selector selects all elements that are the last child, of a particular type, of their parent.

**Tip:** This is the same as `:nth-last-of-type(1)`.

**Tip:** Use the `:first-of-type` selector to select all elements that are the first child, of a particular type, of their parent.

## Syntax

```
$( ":last-of-type" )
```

## Try it Yourself - Examples

[Select the last `<p>` element of all `<div>` elements](#) How to select the last `<p>` element of all `<div>` elements.

[Difference between `:last`, `:last-child` and `:last-of-type`](#) Show the difference between the `:last`, `:last-child` and `:last-of-type` selectors.

# jQuery :nth-child() Selector

## Example

Select each `<p>` element that is the third child of its parent:

```
$("p:nth-child(3)")
```

## Definition and Usage

The `:nth-child(n)` selector selects all elements that are the *n*th child, regardless of type, of their parent.

**Tip:** Use the `:nth-of-type()` selector to select all elements that are the *n*th child, **of a particular type**, of their parent.

## Syntax

```
:nth-child(n|even|odd|formula)
```

Parameter	Description
<i>n</i>	The index of each child to match. Must be a number. The first element has the index number 1.
even	Selects each even child element
odd	Selects each odd child element
<i>formula</i>	Specifies which child element(s) to be selected with a formula ( <i>an + b</i> ). Example: <code>p:nth-child(3n+2)</code> selects each 3rd paragraph, starting at the 2nd child element

## Try it Yourself - Example

Select each `<p>` element that is the second child of all `<div>` elements How to select each `<p>` element that is the second child of all `<div>` elements.

Using a formula (*an + b*) How to use a formula (*an + b*) to select different child elements.

Using "even" and "odd" How to use even and odd to select different child elements.

# jQuery :nth-last-child() Selector

## Example

Select each `<p>` element that is the third child of its parent, counting from the last child:

```
$("p:nth-last-child(3)")
```

## Definition and Usage

The `:nth-last-child(n)` selector selects all elements that are the *n*th child, regardless of type, of their parent, counting from the last child.

**Tip:** Use the `:nth-last-of-type()` selector to select all elements that are the *n*th child, **of a particular type**, of their parent, counting from the last child.

## Syntax

```
:nth-last-child(n|even|odd|formula)
```

Parameter	Description
<i>n</i>	The index of each child to match. Must be a number. The first element has the index number 1.
even	Selects each even child element
odd	Selects each odd child element
<i>formula</i>	Specifies which child element(s) to be selected with a formula ( <i>an + b</i> ). Example: <code>p:nth-last-child(3n+2)</code> selects each 3rd paragraph, starting at the last 2nd child

## Try it Yourself - Example

Select each `<p>` element that is the first child of all `<div>` element, counting from the last child How to select a `<p>` element that is the first child of all `<div>` elements, counting from the last child.

Using a formula (*an + b*) How to use a formula (*an + b*) to select different child elements, counting from the last child.

# jQuery :nth-of-type() Selector

## Example

Select each `<p>` element that is the third `<p>` element of its parent:

```
$("p:nth-of-type(3)")
```

## Definition and Usage

The `:nth-of-type(n)` selector selects all elements that are the *n*th child, of a particular type, of their parent.

**Tip:** Use the `:nth-child()` selector to select all elements that are the *n*th child, **regardless of type**, of their parent.

## Syntax

```
:nth-of-type(n|even|odd|formula)
```

Parameter	Description
<i>n</i>	The index of each child to match. Must be a number. The first element has the index number 1.
even	Selects each even child element
odd	Selects each odd child element
<i>formula</i>	Specifies which child element(s) to be selected with a formula ( <i>an + b</i> ). Example: <code>p:nth-of-type(3n+2)</code> selects each 3rd paragraph, starting at the 2nd paragraph

## Try it Yourself - Example

Select each `<p>` element that is the second `<p>` element of all `<div>` elements How to select each `<p>` element that is the second `<p>` element of all `<div>` elements.

Using a formula (*an + b*) How to use a formula (*an + b*) to select different child elements.

Using "even" and "odd" How to use even and odd to select different child elements.

# jQuery :nth-last-of-type() Selector

## Example

Select each `<p>` element that is the third `<p>` element of its parent, counting from the last child:

```
$("p:nth-last-of-type(3)")
```

## Definition and Usage

The `:nth-last-of-type(n)` selector selects all elements that are the *n*th child, of a particular type, of their parent, counting from the last child.

**Tip:** Use the `:nth-last-child()` selector to select all elements that are the *n*th child, **regardless of type**, of their parent, counting from the last child.

## Syntax

```
:nth-last-of-type(n|even|odd|formula)
```

Parameter	Description
<i>n</i>	The index of each child to match. Must be a number. The first element has the index number 1.
even	Selects each even child element
odd	Selects each odd child element
<i>formula</i>	Specifies which child element(s) to be selected with a formula ( <i>an + b</i> ). Example: <code>p:nth-last-child(3n+2)</code> selects each 3rd paragraph, starting at the last 2nd paragraph

## Try it Yourself - Example

Select each `<p>` element that is the first `<p>` element of all `<div>` element, counting from the last child How to select each `<p>` element that is the first `<p>` element of all `<div>` elements, counting from the last child.

Using a formula (an + b) How to use a formula ( $an + b$ ) to select different `<p>` elements, counting from the last child.

Using "even" and "odd" How to use even and odd to select different `<p>` elements, counting from the last child.

Difference between :nth-child(), :nth-last-child(), :nth-of-type() and :nth-of-last-type() The difference between `p:nth-child(2)`, `p:nth-last-child(2)`, `p:nth-of-type(2)` and `p:nth-of-last-type(2)`.

# jQuery :only-child Selector

## Example

Select each <p> element that is the only child of its parent:

```
$("p:only-child")
```

## Definition and Usage

The :only-child selector selects every element that is the only child of its parent.

## Syntax

```
$(":only-child")
```

# jQuery :only-of-type Selector

## Example

Select each <p> element that is the only <p> element of its parent:

```
$( "p:only-of-type" )
```

## Definition and Usage

The :only-of-type selector selects every element that is the only child of its type, of its parent.

## Syntax

```
$( ":only-of-type" )
```

# jQuery parent > child Selector

## Example

Select all `<span>` elements that are a direct child of a `<div>` element:

```
$( "div > span" )
```

## Definition and Usage

The ("parent > child") selector selects all elements that are a direct child of the specified element.

## Syntax

```
("parent > child")
```

Parameter	Description
<i>parent</i>	Required. Specifies the parent element to be selected
<i>child</i>	Required. Specifies the direct child element (of the specified parent element) to be selected

## Try it Yourself - Example

Select all `<li>` elements that are a direct child of an `<ul>` element How to select all `<li>` elements that are a direct child of an `<ul>` element with a classname.

# jQuery parent descendant Selector

## Example

Select all `<span>` elements that are descendants of a `<div>` element:

```
$( "div span" )
```

## Definition and Usage

The ("parent descendant") selector selects all elements that are descendants of a specified element.

A descendant of an element could be a child, grandchild, great-grandchild, etc, of that element.

## Syntax

```
("parent descendant")
```

Parameter	Description
<code>parent</code>	Required. Specifies the parent element to be selected
<code>descendant</code>	Required. Specifies the descendant element (of the specified parent element) to be selected

## Try it Yourself - Example

Select all `<li>` elements that are descendants of an `<ul>` element How to select all `<li>` elements that are descendants of an `<ul>` element.

# jQuery element + next Selector

## Example

Select the `<p>` element that are next to each `<div>` element:

```
$("div + p")
```

## Definition and Usage

The ("element + next") selector selects the "next" element of the specified "element". The "next" element must be placed right after the specified "element" to be selected.

Example: If you have two `<p>` elements right after a `<div>` element, this syntax:

- `$("div + p")` - will only select the first `<p>` element, because it is the next element of the `<div>` element (the other `<p>` element will be ignored)

Example: If you have an `<h2>` element right after a `<div>` element, and then a `<p>` element, this syntax:

- `$("div + p")` - will not select the `<p>` element, because the next element of the `<div>` element is the `<h2>` element

**Note:** Both of the specified elements must share the same parent.

## Syntax

```
("element + next")
```

Parameter	Description
<code>element</code>	Required. Any valid jQuery selector
<code>next</code>	Required. Specifies the element that should be the <i>next</i> element of the <code>element</code> parameter

## Try it Yourself - Example

Select the `<div>` element that are next to each `<ul>` element How to select the `<div>` element that are next to each `<ul>` element.

# jQuery element ~ siblings Selector

## Example

Select all `<p>` elements that are siblings of the `<div>` element:

```
$("div ~ p")
```

## Definition and Usage

The ("element ~ siblings") selector selects all elements that are *siblings* of the specified "element".

For example:

- `$("div ~ p")` - selects all `<p>` elements that are siblings of the `<div>` element.

**Note:** Both of the specified elements must share the same parent.

## Syntax

```
("element ~ siblings")
```

Parameter	Description
<code>element</code>	Required. Any valid jQuery selector
<code>siblings</code>	Required. Specifies the <i>siblings</i> of the <code>element</code> parameter

# jQuery :eq() Selector

## Example

Select the second <p> element:

```
$("p:eq(1)")
```

## Definition and Usage

The :eq() selector selects an element with a specific index number.

The index numbers start at 0, so the first element will have the index number 0 (not 1).

This is mostly used together with another selector to select a specifically indexed element in a group (like in the example above).

## Syntax

```
$(":eq(index)")
```

Parameter	Description
<i>index</i>	Required. Specifies the index of the element

# jQuery :gt() Selector

## Example

Select all <tr> elements after the 4 first:

```
$(“tr:gt(3)”)
```

## Definition and Usage

The :gt() selector selects elements with an index number higher than a specified number.

The index numbers start at 0.

This is mostly used together with another selector to select the last elements in a group (like in the example above).

**Tip:** Use the [:lt selector](#) to select elements index numbers lesser than the specified number.

## Syntax

```
$(“:gt(index)”)
```

Parameter	Description
<i>index</i>	Required. Specifies which element to select. Elements with an index higher than the specified number is selected.

# jQuery :lt() Selector

## Example

Select the 4 first <tr> elements:

```
$(“tr:lt(4)”)
```

## Definition and Usage

The :lt() selector selects elements with an index number less than a specified number.

The index numbers start at 0.

This is mostly used together with another selector to select the first elements in a group (like in the example above).

**Tip:** Use the :gt selector to select elements index numbers greater than the specified number.

## Syntax

```
$(“:lt(index)”)
```

Parameter	Description
<i>index</i>	Required. Specifies which element to select. Elements with an index lesser than the specified number is selected.

# jQuery :not() Selector

## Example

Select all <p> elements except those with class="intro":

```
$("p:not(.intro)")
```

## Definition and Usage

The :not() selector selects all elements except the specified element.

This is mostly used together with another selector to select everything except the specified element in a group (like in the example above).

## Syntax

```
$(":not(selector)")
```

Parameter	Description
<i>selector</i>	Required. Specifies the element to not select. This parameter accepts any kind of selector

# jQuery :header Selector

## Example

Select all header elements (h1 to h6):

```
$(":header")
```

## Definition and Usage

The :header selector selects all header elements (<h1> to <h6>).

## Syntax

```
$(":header")
```

# jQuery :animated Selector

## Example

Select any element that is currently animated:

```
$(":animated")
```

## Definition and Usage

The :animated selector selects all elements that are currently animated.

## Syntax

```
$(":animated")
```

# jQuery :focus Selector

## Example

Select the element that currently has focus:

```
$(":focus")
```

## Definition and Usage

The :focus selector selects the element that currently has focus.

**Tip:** This selector is often used with a tag name or another selector, if not, this selector will be the same as ("\*:focus").

## Syntax

```
$(":focus")
```

# jQuery :contains() Selector

## Example

Select all <p> elements containing "is":

```
$("p:contains(is)")
```

## Definition and Usage

The :contains() selector selects elements containing the specified string.

The string can be contained directly in the element as text, or in a child element.

This is mostly used together with another selector to select the elements containing the text in a group (like in the example above).

**Note:** The text is case sensitive.

## Syntax

```
$(":contains(text)")
```

Parameter	Description
<i>text</i>	Required. Specifies the text to find

# jQuery :has() Selector

## Example

Select all <p> elements that have a <span> element inside of them:

```
$(“p:has(span)”)
```

## Definition and Usage

The :has() selector selects all elements that have one or more elements inside of them, that matches the specified selector.

**Tip:** To select an element that have multiple elements inside of it, use comma (see example below).

## Syntax

```
$(“:has(selector)”)
```

Parameter	Description
<i>selector</i>	Required. Specifies the element to select. This parameter accepts any kind of selector

## Try it Yourself - Examples

Select an element with multiple elements inside How to select an element that has multiple elements inside of it.

Select elements that does NOT have a specified element inside Using the :not selector together with :has to select elements that does NOT have an element inside of it.

# jQuery :empty Selector

## Example

Select all empty elements:

```
$(":empty")
```

## Definition and Usage

The :empty selector selects empty elements.

An empty element is an element without child elements or text.

## Syntax

```
$(":empty")
```

# jQuery :parent Selector

## Example

Select all <td> elements with children, including text:

```
$("td:parent")
```

## Definition and Usage

The :parent selector selects all elements that are the parent of another element, including text nodes.

## Syntax

```
$(":parent")
```

# jQuery :hidden Selector

## Example

Show hidden <p> elements:

```
$(“p:hidden”).show();
```

## Definition and Usage

The :hidden selector selects hidden elements.

Hidden elements are elements that are:

- Set to display:none
- Form elements with type="hidden"
- Width and height set to 0
- A hidden parent element (this also hides child elements)

**Note:** This selector will not work on elements with visibility:hidden.

## Syntax

```
$(“:hidden”)
```

# jQuery :visible Selector

## Example

Select all visible <p> elements:

```
$(“p:visible”)
```

## Definition and Usage

The :visible selector selects every element that is currently visible.

Visible elements are elements that are not:

- Set to display:none
- Form elements with type="hidden"
- Width and height set to 0
- A hidden parent element (this also hides child elements)

## Syntax

```
$(“:visible”)
```

# jQuery :root Selector

## Example

Set the background color for the HTML document to yellow:

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

## Definition and Usage

The :root selector selects the document's root element.

In HTML, the root element is always the <html> element.

## Syntax

```
$(":root")
```

# jQuery :lang() Selector

## Example

Select all <p> elements with a lang attribute value that starts with "it":

```
$(“p:lang(it)”)
```

## Definition and Usage

The :lang() selector selects all elements with the language attribute starting with a specified value.

**Note:** The value has to be a whole word, either alone, like lang="en", or followed by a hyphen( - ), like lang="en-us".

## Syntax

```
$(“:lang(Language)”)
```

# jQuery [attribute] Selector

## Example

Select every element with an id attribute:

```
$("[id]")
```

## Definition and Usage

The [attribute] selector selects each element with the specified attribute.

## Syntax

```
$("[attribute]")
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find

# jQuery [attribute=value] Selector

## Example

Select every element containing an id attribute with the value "choose":

```
$("[id=choose]")
```

## Definition and Usage

The [attribute=value] selector selects each element with the specified attribute and value.

## Syntax

```
$("[attribute=value]")
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the value to find

# jQuery [attribute!=value] Selector

## Example

Select all <p> elements **NOT** containing a class attribute with the value "intro":

```
$("p[class!='intro']")
```

## Definition and Usage

The [attribute!=value] selector selects each element that does **NOT** have the specified attribute and value.

Elements with the selected attribute, but with a different value, will be selected.

## Syntax

```
$("[attribute!='value']")
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the value to find

# jQuery [attribute\$=value] Selector

## Example

Select all <a> elements with a href attribute that ends with ".org":

```
$("a[href$='.org']")
```

## Definition and Usage

The [attribute\$=value] selector selects each element with a specific attribute, with a value ending in a specific string.

## Syntax

```
$("[attribute$='value'])")
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the string the value should end with

# jQuery [attribute]=value Selector

## Example

Select all <p> elements with a title attribute that starts with the value "Tomorrow":

```
$(“p[title|=‘Tomorrow’]”)
```

## Definition and Usage

The [attribute]=value selector selects each element with a specified attribute, with a value equal to a specified string (like "en") or starting with that string followed by a hyphen (like "en-us").

**Tip:** This selector is often used to handle language attributes.

## Syntax

```
$(“[attribute|=‘value’]”)
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the string the attribute value should start with

## More Examples

Handle language attributes This example selects all <a> elements with a hreflang attribute that starts with the value "en".

# jQuery [attribute^=value] Selector

## Example

Select all <input> elements with a name attribute that starts with "nation":

```
$(“input[name^='nation’ ]”)
```

## Definition and Usage

The [attribute^=value] selector selects each element with a specific attribute, with a value beginning in a specific string.

## Syntax

```
$(“[attribute^='value’ ]”)
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the string the value should begin with

# jQuery [attribute $\sim$ =value] Selector

## Example

Select all <input> elements with a name attribute that contains the specific word "nation":

```
$(“input[name~=‘nation’]”)
```

## Definition and Usage

The [attribute $\sim$ =value] selector selects each element with a specific attribute, with a value containing a specific string.

**Tip:** The string can contain whitespace.

## Syntax

```
$(“[attribute $\sim$ =‘value’]”)
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the string value

# jQuery [attribute\*=value] Selector

## Example

Select all <input> elements with a name attribute that contains the word "nation":

```
$(“input[name*=‘nation’]”)
```

## Definition and Usage

The [attribute\*=value] selector selects each element with a specific attribute, with a value containing a string.

## Syntax

```
$(“[attribute*=‘value’]”)
```

Parameter	Description
<i>attribute</i>	Required. Specifies the attribute to find
<i>value</i>	Required. Specifies the string value

# jQuery :input Selector

## Example

Select all input elements:

```
$(":input")
```

## Definition and Usage

The :input selector selects form elements.

This selector also works with the button element.

## Syntax

```
$(":input")
```

# jQuery :text Selector

## Example

Select <input> elements with type="text":

```
$(":text")
```

## Definition and Usage

The :text selector selects input elements with type=text.

## Syntax

```
$(":text")
```

# jQuery :password Selector

## Example

Select <input> elements with type="password":

```
$(":password")
```

## Definition and Usage

The :password selector selects input elements with type=password.

## Syntax

```
$(":password")
```

# jQuery :radio Selector

## Example

Select all <input> elements with type="radio":

```
$(":radio")
```

## Definition and Usage

The :radio selector selects input elements with type=radio.

## Syntax

```
$(":radio")
```

# jQuery :checkbox Selector

## Example

Select all <input> elements with type="checkbox":

```
$( ":checkbox" )
```

## Definition and Usage

The :checkbox selector selects input elements with type=checkbox.

## Syntax

```
$( ":checkbox" )
```

# jQuery :submit Selector

## Example

Select <input> and <button> elements with type="submit":

```
$(":submit")
```

## Definition and Usage

The :submit selector selects button and input elements with type=submit.

If a button element has no defined type, most browsers will use it as a button with type=submit.

**Tip:** Using input:submit as a selector will not select the button element.

## Syntax

```
$(":submit")
```

# jQuery :reset Selector

## Example

Select <input> and <button> elements with type="reset":

```
$(":reset")
```

## Definition and Usage

The :reset selector selects button and input elements with type=reset.

**Tip:** Using input:reset as a selector will not select the button element.

## Syntax

```
$(":reset")
```

# jQuery :button Selector

## Example

Select <button> elements and <input> elements with type="button":

```
$(":button")
```

## Definition and Usage

The :button selector selects button elements, and input elements with type=button.

**Tip:** Using input:button as a selector will not select the button element.

# jQuery :image Selector

## Example

Select <input> elements with type="image":

```
$(":image")
```

## Definition and Usage

The :image selector selects input elements with type=image.

## Syntax

```
$(":image")
```

# jQuery :file Selector

## Example

Select <input> elements with type="file":

```
$(":file")
```

## Definition and Usage

The :file selector selects input elements with type=file.

## Syntax

```
$(":file")
```

# jQuery :enabled Selector

## Example

Select all the enabled form elements:

```
$(":enabled")
```

## Definition and Usage

The :enabled selector selects all enabled form elements.

## Syntax

```
$(":enabled")
```

# jQuery :disabled Selector

## Example

Select all the disabled form elements:

```
$(":disabled")
```

## Definition and Usage

The :disabled selector selects all disabled form elements.

## Syntax

```
$(":disabled")
```

# jQuery :selected Selector

## Example

Select the pre-selected item(s) in a drop-down list:

```
$(":selected")
```

## Definition and Usage

The :selected selector selects option elements that are pre-selected.

**Note:** This selector will not work on checkboxes or radio buttons. Use the :checked selector instead.

## Syntax

```
$(":selected")
```

# jQuery :checked Selector

## Example

Select all checked elements (checkboxes or radio buttons):

```
$(":checked")
```

## Definition and Usage

The :checked selector selects all checked checkboxes or radio buttons.

## Syntax

```
$(":checked")
```

# jQuery Event Methods

## jQuery Event Methods

Event methods trigger or attach a function to an event handler for the selected elements. The following table lists all the jQuery methods used to handle events.

Method / Property	Description
<u>bind()</u>	Deprecated in version 3.0. Use the <a href="#">on()</a> method instead. Attaches event handlers to elements
<u>blur()</u>	Attaches/Triggers the blur event
<u>change()</u>	Attaches/Triggers the change event
<u>click()</u>	Attaches/Triggers the click event
<u>dblclick()</u>	Attaches/Triggers the double click event
<u>delegate()</u>	Deprecated in version 3.0. Use the <a href="#">on()</a> method instead. Attaches a handler to current, or future, specified child elements of the matching elements
<u>die()</u>	Removed in version 1.9. Removes all event handlers added with the live() method
<u>error()</u>	Removed in version 3.0. Attaches/Triggers the error event
<u>event.currentTarget</u>	The current DOM element within the event bubbling phase
<u>event.data</u>	Contains the optional data passed to an event method when the current executing handler is bound
<u>event.delegateTarget</u>	Returns the element where the currently-called jQuery event handler was attached
<u>event.isDefaultPrevented()</u>	Returns whether event.preventDefault() was called for the event object
<u>event.isImmediatePropagationStopped()</u>	Returns whether event.stopImmediatePropagation() was called for the event object

<b>Method / Property</b>	<b>Description</b>
<u>event.isPropagationStopped()</u>	Returns whether event.stopPropagation() was called for the event object
<u>event.namespace</u>	Returns the namespace specified when the event was triggered
<u>event.pageX</u>	Returns the mouse position relative to the left edge of the document
<u>event.pageY</u>	Returns the mouse position relative to the top edge of the document
<u>event.preventDefault()</u>	Prevents the default action of the event
<u>event.relatedTarget</u>	Returns which element being entered or exited on mouse movement.
<u>event.result</u>	Contains the last/previous value returned by an event handler triggered by the specified event
<u>event.stopImmediatePropagation()</u>	Prevents other event handlers from being called
<u>event.stopPropagation()</u>	Prevents the event from bubbling up the DOM tree, preventing any parent handlers from being notified of the event
<u>event.target</u>	Returns which DOM element triggered the event
<u>event.timeStamp</u>	Returns the number of milliseconds since January 1, 1970, when the event is triggered
<u>event.type</u>	Returns which event type was triggered
<u>event.which</u>	Returns which keyboard key or mouse button was pressed for the event
<u>focus()</u>	Attaches/Triggers the focus event
<u>focusin()</u>	Attaches an event handler to the focusin event
<u>focusout()</u>	Attaches an event handler to the focusout event
<u>hover()</u>	Attaches two event handlers to the hover event
<u>keydown()</u>	Attaches/Triggers the keydown event
<u>keypress()</u>	Attaches/Triggers the keypress event
<u>keyup()</u>	Attaches/Triggers the keyup event

<b>Method / Property</b>	<b>Description</b>
<u>live()</u>	Removed in version 1.9. Adds one or more event handlers to current, or future, selected elements
<u>load()</u>	Removed in version 3.0. Attaches an event handler to the load event
<u>mousedown()</u>	Attaches/Triggers the mousedown event
<u>mouseenter()</u>	Attaches/Triggers the mouseenter event
<u>mouseleave()</u>	Attaches/Triggers the mouseleave event
<u>mousemove()</u>	Attaches/Triggers the mousemove event
<u>mouseout()</u>	Attaches/Triggers the mouseout event
<u>mouseover()</u>	Attaches/Triggers the mouseover event
<u>mouseup()</u>	Attaches/Triggers the mouseup event
<u>off()</u>	Removes event handlers attached with the on() method
<u>on()</u>	Attaches event handlers to elements
<u>one()</u>	Adds one or more event handlers to selected elements. This handler can only be triggered once per element
<u>\$.proxy()</u>	Takes an existing function and returns a new one with a particular context
<u>ready()</u>	Specifies a function to execute when the DOM is fully loaded
<u>resize()</u>	Attaches/Triggers the resize event
<u>scroll()</u>	Attaches/Triggers the scroll event
<u>select()</u>	Attaches/Triggers the select event
<u>submit()</u>	Attaches/Triggers the submit event
<u>toggle()</u>	Removed in version 1.9. Attaches two or more functions to toggle between for the click event
<u>trigger()</u>	Triggers all events bound to the selected elements

Method / Property	Description
<u>triggerHandler()</u>	Triggers all functions bound to a specified event for the selected elements
<u>unbind()</u>	Deprecated in version 3.0. Use the <a href="#"><u>off()</u></a> method instead. Removes an added event handler from selected elements
<u>undelegate()</u>	Deprecated in version 3.0. Use the <a href="#"><u>off()</u></a> method instead. Removes an event handler to selected elements, now or in the future
<u>unload()</u>	Removed in version 3.0. Attaches an event handler to the unload event

# jQuery bind() Method

## Example

Attach a click event to the <p> element:

```
$(“p”).bind(“click”, function(){  
    alert(“The paragraph was clicked.”);  
});
```

## Definition and Usage

The **bind()** method was deprecated in version 3.0. Use the [on\(\)](#) method instead.

The bind() method attaches one or more event handlers for selected elements, and specifies a function to run when the event occurs.

## Syntax

```
$(selector).bind(event,data,function,map)
```

Parameter	Description
<i>event</i>	Required. Specifies one or more events to attach to the elements. Multiple event values are separated by space. Must be a valid event.
<i>data</i>	Optional. Specifies additional data to pass along to the function
<i>function</i>	Required. Specifies the function to run when the event occurs
<i>map</i>	Specifies an event map ( <code>{event:function, event:function, ...}</code> ) containing one or more events to attach to the elements, and functions to run when the event occurs

## Try it Yourself - Examples

[Attach multiple events](#) How to attach multiple events to an element.

[Using an event map](#) How to use an event map to attach several events/functions to the selected elements.

[Pass along data to the function](#) How to pass along data to a custom named event handler.

# jQuery blur() Method

## Example

Attach a function to the blur event. The blur event occurs when the <input> field loses focus:

```
$( "input" ).blur(function(){
    alert("This input field has lost its focus.");
});
```

## Definition and Usage

The blur event occurs when an element loses focus.

The blur() method triggers the blur event, or attaches a function to run when a blur event occurs.

**Tip:** This method is often used together with the [focus\(\)](#) method.

## Syntax

Trigger the blur event for the selected elements:

```
$(selector).blur()
```

[Try it](#)

Attach a function to the blur event:

```
$(selector).blur(function)
```

[Try it](#)

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the blur event occurs

# jQuery change() Method

## Example

Alert a text when an <input> field is changed:

```
$( "input" ).change(function(){
    alert("The text has been changed.");
});
```

## Definition and Usage

The change event occurs when the value of an element has been changed (only works on <input>, <textarea> and <select> elements).

The change() method triggers the change event, or attaches a function to run when a change event occurs.

**Note:** For select menus, the change event occurs when an option is selected. For text fields or text areas, the change event occurs when the field loses focus, after the content has been changed.

## Syntax

Trigger the change event for the selected elements:

```
$(selector).change()
```

Try it

Attach a function to the change event:

```
$(selector).change(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the change event occurs for the selected elements

# jQuery click() Method

## Example

Click on a <p> element to alert a text:

```
$( "p" ).click(function(){
    alert("The paragraph was clicked.");
});
```

## Definition and Usage

The click event occurs when an element is clicked.

The click() method triggers the click event, or attaches a function to run when a click event occurs.

## Syntax

Trigger the click event for the selected elements:

```
$(selector).click()
```

Try it

Attach a function to the click event:

```
$(selector).click(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the click event occurs

# jQuery dblclick() Event

## Example

Double-click on a <p> element to alert a text:

```
$( "p" ).dblclick(function(){
    alert("The paragraph was double-clicked");
});
```

## Definition and Usage

The dblclick event occurs when an element is double-clicked.

The dblclick() method triggers the dblclick event, or attaches a function to run when a dblclick event occurs.

**Tip:** The dblclick event also generates a click event. This can cause problems if both events are applied to the same element.

## Syntax

Trigger the dblclick event for the selected elements:

```
$(selector).dblclick()
```

Try it

Attach a function to the dblclick event:

```
$(selector).dblclick(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the dblclick event occurs

# jQuery delegate() Method

## Example

When a <p> element inside a <div> element is clicked, change the background color of all <p> elements:

```
$( "div" ).delegate( "p", "click", function(){
    $( "p" ).css( "background-color", "pink" );
});
```

## Definition and Usage

The `delegate()` method was deprecated in version 3.0. Use the [on\(\)](#) method instead.

The `delegate()` method attaches one or more event handlers for specified elements that are children of selected elements, and specifies a function to run when the events occur.

Event handlers attached using the `delegate()` method will work for both current and FUTURE elements (like a new element created by a script).

## Syntax

```
$(selector).delegate(childSelector, event, data, function)
```

Parameter	Description
<code>childSelector</code>	Required. Specifies one or more child elements to attach the event handler to
<code>event</code>	Required. Specifies one or more events to attach to the elements. Multiple event values are separated by space. Must be a valid event
<code>data</code>	Optional. Specifies additional data to pass along to the function
<code>function</code>	Required. Specifies the function to run when the event occurs

## Try it Yourself - Examples

[Add event handlers for future elements](#) How to use the `delegate()` method to add event handlers for elements not yet created.

[Pass along data to the function](#) How to pass along data to a custom named event handler.

# jQuery die() Method

## Example

Remove all event handlers added with the `live()` method for all `<p>` elements:

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

## Definition and Usage

The `die()` method was [deprecated](#) in jQuery version 1.7, and removed in version 1.9. Use the [off\(\)](#) method instead.

The `die()` method removes one or more event handlers, added with the [live\(\)](#) method, for the selected elements.

## Syntax

```
$(selector).die(event,function)
```

Parameter	Description
<code>event</code>	Required. Specifies one or more event handlers to remove. Multiple event values are separated by space. Must be a valid event
<code>function</code>	Optional. Specifies a specific function to remove

## Try it Yourself - Examples

[Remove one specific event handler](#) How to use the `die()` method to remove only one specific event handler from the selected elements.

# jQuery error() Method

## Example

If the image element encounters an error, replace it with a text:

```
$( "img" ).error(function(){
    $( "img" ).replaceWith("<p>Error loading image!</p>");
});
```

## Definition and Usage

The `error()` method was deprecated in jQuery version 1.8, and **removed** in version 3.0.

The error event occurs when an element encounters an error (if the element is not loaded correctly).

The `error()` method triggers the error event, or attaches a function to run when an error event occurs.

**Tip:** This method is a shortcut for `bind('error', handler)`.

## Syntax

Trigger the error event for the selected elements:

```
$(selector).error()
```

Try it

Attach a function to the error event:

```
$(selector).error(function)
```

Try it

Parameter	Description
<code>function</code>	Optional. Specifies the function to run when the error event occurs

# jQuery event.currentTarget Property

## Example

`event.currentTarget` is typically equal to `this`:

```
$( "h1, h2, p" ).click( function(event){  
    alert( event.currentTarget === this );  
});
```

## Definition and Usage

The `event.currentTarget` property is the current DOM element within the event bubbling phase, and is typically equal to `this`.

## Syntax

```
event.currentTarget
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

## Try it Yourself - Examples

Return the content of elements Using `event.currentTarget.innerHTML` to return content of elements.

# jQuery event.data Property

## Example

Return the data passed with the on() method for each <p> element:

```
$( "p" ).each(function(i){
    $(this).on("click", {x:i}, function(event){
        alert("The " + $(this).index() + ". paragraph has data: " +
event.data.x);
    });
});
```

## Definition and Usage

The event.data property contains the optional data passed to an event method when the current executing handler is bound.

## Syntax

*event.data*

Parameter	Description
<i>event</i>	Required. The <i>event</i> parameter comes from the event binding function

# jQuery event.delegateTarget Property

## Example

Change the background color of the <div> element (an ancestor of the <button> element):

```
$("div").on("click", "button", function(event){
    $(event.delegateTarget).css("background-color", "pink");
});
```

## Definition and Usage

The `event.delegateTarget` property returns the element where the currently-called jQuery event handler was attached.

This property is useful for delegated events attached by the `on()` method, where the event handler is attached at an ancestor of the element being processed.

**Tip:** `event.delegateTarget` is equal to `event.currentTarget`, if the event is directly-bound to an element and no delegation occurs (see example below).

## Syntax

```
event.delegateTarget
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

## Try it Yourself - Examples

[Difference between `delegateTarget` and `currentTarget` with delegation](#) Show the difference between `delegateTarget` and `currentTarget` when delegation occurs.

[The `delegateTarget` and `currentTarget` are equal for directly bound events](#) Show that `delegateTarget` and `currentTarget` are equal for directly bound events.

# jQuery event.isDefaultPrevented() Method

## Example

Prevent a link from opening the URL, and check if preventDefault() was called:

```
$( "a" ).click(function(event){  
    event.preventDefault();  
    alert("Was preventDefault() called: " + event.isDefaultPrevented());  
});
```

## Definition and Usage

The event.isDefaultPrevented() method checks whether the [preventDefault\(\)](#) method was called for the event.

## Syntax

```
event.isDefaultPrevented()
```

Parameter	Description
<i>event</i>	Required. The <i>event</i> parameter comes from the event binding function

# jQuery

## event.isImmediatePropagationStopped()

### Method

#### Example

Check if event.stopImmediatePropagation() was called:

```
$("div").click(function(event){  
    event.stopImmediatePropagation();  
    alert(event.isImmediatePropagationStopped());  
});
```

#### Definition and Usage

This method checks whether the [event.stopImmediatePropagation\(\)](#) was called for the event. This method returns true if event.stopImmediatePropagation() is called, and false if not.

#### Syntax

```
event.isImmediatePropagationStopped()
```

Parameter	Description
event	Required. The <i>event</i> parameter comes from the event binding function

# jQuery event.isPropagationStopped() Method

## Example

Check if event.stopPropagation() was called:

```
$("div").click(function(event){  
    event.stopPropagation();  
    alert(event.isPropagationStopped());  
});
```

## Definition and Usage

The event.isPropagationStopped() method checks whether event.stopPropagation() was called for the event.

This method returns true if event.stopPropagation() is called, and false if not.

## Syntax

```
event.isPropagationStopped()
```

Parameter	Description
event	Required. The <i>event</i> parameter comes from the event binding function

# jQuery event.namespace Property

## Example

Add and remove a custom namespace:

```
$( "p" ).on( "custom.someNamespace", function(event){  
    alert(event.namespace);  
});  
$( "p" ).click(function(event){  
    $(this).trigger("custom.someNamespace");  
});  
$( "button" ).click(function(){  
    $("p").off("custom.someNamespace");  
});
```

## Definition and Usage

The `event.namespace` property returns the custom namespace when the event was triggered. This property can be used by plugin authors to handle tasks differently depending on the namespace used.

**Tip:** Namespaces beginning with an underscore are reserved for jQuery.

## Syntax

```
event.namespace
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

## Try it Yourself - Examples

[Remove a custom namespace for a particular click event](#) Remove a custom namespace for a particular click event, without removing any other click event handlers.

# jQuery event.pageX Property

## Example

Return the position of the mouse pointer:

```
$(document).mousemove(function(event){  
    $("span").text("X: " + event.pageX + ", Y: " + event.pageY);  
});
```

## Definition and Usage

The `event.pageX` property returns the position of the mouse pointer, relative to the left edge of the document.

**Tip:** This event property is often used together with the `event.pageY` property.

## Syntax

```
event.pageX
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.pageY Property

## Example

Return the position of the mouse pointer:

```
$(document).mousemove(function(event){  
    $("span").text("X: " + event.pageX + ", Y: " + event.pageY);  
});
```

## Definition and Usage

The `event.pageY` property returns the position of the mouse pointer, relative to the top edge of the document.

**Tip:** This event property is often used together with the `event.pageX` property.

## Syntax

```
event.pageY
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.preventDefault() Method

## Example

Prevent a link from opening the URL:

```
$("a").click(function(event){  
    event.preventDefault();  
});
```

## Definition and Usage

The `event.preventDefault()` method stops the default action of an element from happening.  
For example:

- Prevent a submit button from submitting a form
- Prevent a link from following the URL

**Tip:** Use the `event.isDefaultPrevented()` method to check whether the `preventDefault()` method was called for the event.

## Syntax

```
event.preventDefault()
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.relatedTarget Property

## Example

Return the related target of an element:

```
$("div").mouseenter(function(event){  
    alert("relatedTarget is: " + event.relatedTarget);  
});
```

## Definition and Usage

The `event.relatedTarget` property returns which element being entered or exited on mouse movement.

## Syntax

```
event.relatedTarget
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.result Property

## Example

Return the value of the previous click event:

```
$("button").click(function(){
    return "Hello world!";
});
$("button").click(function(event){
    $("p").html(event.result);
});
```

## Definition and Usage

The `event.result` property contains the last/previous value returned by an event handler triggered by the specified event.

## Syntax

```
event.result
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.stopImmediatePropagation() Method

## Example

Execute the first event handler, and stop the rest of the event handlers from being executed:

```
$("div").click(function(event){
    alert("Event handler 1 executed");
    event.stopImmediatePropagation();
});
$("div").click(function(event){
    alert("Event handler 2 executed");
});
$("div").click(function(event){
    alert("Event handler 3 executed");
});
```

## Definition and Usage

The `event.stopImmediatePropagation()` method stops the rest of the event handlers from being executed.

This method also stops the event from bubbling up the DOM tree.

**Tip:** Use the `event.isImmediatePropagationStopped()` method to check whether this method was called for the event.

## Syntax

```
event.stopImmediatePropagation()
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.stopPropagation() Method

## Example

Stop the click event from bubbling to parent elements:

```
$("span").click(function(event){
    event.stopPropagation();
    alert("The span element was clicked.");
});
$("p").click(function(event){
    alert("The p element was clicked.");
});
$("div").click(function(){
    alert("The div element was clicked.");
});
```

## Definition and Usage

The `event.stopPropagation()` method stops the bubbling of an event to parent elements, preventing any parent event handlers from being executed.

**Tip:** Use the `event.isPropagationStopped()` method to check whether this method was called for the event.

## Syntax

```
event.stopPropagation()
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.target Property

## Example

Return which DOM element triggered the event:

```
$("p, button, h1, h2").click(function(event){  
    $("div").html("Triggered by a " + event.target.nodeName + " element.");  
});
```

## Definition and Usage

The `event.target` property returns which DOM element triggered the event.

It is often useful to compare `event.target` to `this` in order to determine if the event is being handled due to event bubbling.

## Syntax

```
event.target
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.timeStamp Property

## Example

Return the number of milliseconds since Jan 1, 1970, when the click event occurs for a button:

```
$( "button" ).click(function(event){  
    $( "span" ).text(event.timeStamp);  
});
```

## Definition and Usage

The `event.timeStamp` property returns the number of milliseconds since January 1, 1970, when the event is triggered.

## Syntax

```
event.timeStamp
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

## More Examples

[Display the number of milliseconds since last click event](#) Show how to use `event.timeStamp` to display the number of milliseconds since last click event.

# jQuery event.type Property

## Example

Return which event type was triggered:

```
$("p").on("click dblclick mouseover mouseout", function(event){  
    $("div").html("Event: " + event.type);  
});
```

## Definition and Usage

The `event.type` property returns which event type was triggered.

## Syntax

```
event.type
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

# jQuery event.which Property

## Example

Return which keyboard key was pressed:

```
$("input").keydown(function(event){  
    $("div").html("Key: " + event.which);  
});
```

## Definition and Usage

The `event.which` property returns which keyboard key or mouse button was pressed for the event.

## Syntax

```
event.which
```

Parameter	Description
<code>event</code>	Required. The <code>event</code> parameter comes from the event binding function

## More Examples

[Return which mouse button was pressed](#) Show which mouse button was pressed for the event.

# jQuery focus() Method

## Example

Attach a function to the focus event. The focus event occurs when the <input> field gets focus:

```
$( "input" ).focus(function(){
  $("span").css("display", "inline").fadeOut(2000);
});
```

## Definition and Usage

The focus event occurs when an element gets focus (when selected by a mouse click or by "tab-navigating" to it).

The focus() method triggers the focus event, or attaches a function to run when a focus event occurs.

**Tip:** This method is often used together with the [blur\(\)](#) method.

## Syntax

Trigger the focus event for selected elements:

```
$(selector).focus()
```

[Try it](#)

Attach a function to the focus event:

```
$(selector).focus(function)
```

[Try it](#)

### Parameter

*function*

### Description

Optional. Specifies the function to run when the focus event occurs

# jQuery focusin() Method

## Example

Set background color of a <div> element when the <div> element or any child elements get focus:

```
$( "div" ).focusin(function(){
    $(this).css("background-color", "#FFFFCC");
});
```

## Definition and Usage

The focusin event occurs when an element (or any elements inside it) gets focus.

The focusin() method attaches a function to run when a focus event occurs on the element, or any elements inside it.

Unlike the focus() method, the focusin() method also triggers if any child elements get focus.

**Tip:** An element gets focus when selected by a mouse click or by "tab-navigating" to it.

**Tip:** This method is often used together with the focusout() method.

## Syntax

```
$(selector).focusin(function)
```

Parameter	Description
<i>function</i>	Required. Specifies the function to run when the focusin event occurs.

# jQuery focusout() Method

## Example

Set background color of a <div> element when the <div> element or any child elements loses focus:

```
$( "div" ).focusout(function(){
    $(this).css("background-color", "#FFFFFF");
});
```

## Definition and Usage

The focusout event occurs when an element (or any elements inside it) loses focus.

The focusout() method attaches a function to run when a focusout event occurs on the element, or any elements inside it.

Unlike the [blur\(\)](#) method, the focusout() method also triggers if any child elements lose focus.

**Tip:** This method is often used together with the [focusin\(\)](#) method.

## Syntax

```
$(selector).focusout(function)
```

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the focusout event occurs

# jQuery hover() Method

## Example

Change the background color of a <p> element when the mouse pointer hovers over it:

```
$("p").hover(function(){
    $(this).css("background-color", "yellow");
}, function(){
    $(this).css("background-color", "pink");
});
```

## Definition and Usage

The hover() method specifies two functions to run when the mouse pointer hovers over the selected elements.

This method triggers both the mouseenter and mouseleave events.

**Note:** If only one function is specified, it will be run for both the mouseenter and mouseleave events.

## Syntax

```
$(selector).hover(inFunction,outFunction)
```

Parameter	Description
<i>inFunction</i>	Required. Specifies the function to run when the mouseenter event occurs
<i>outFunction</i>	Optional. Specifies the function to run when the mouseleave event occurs

# jQuery keydown() Method

## Example

Set the background color of an <input> field when a keyboard key is pressed down:

```
$("input").keydown(function(){
    $("input").css("background-color", "yellow");
});
```

## Definition and Usage

The order of events related to the keydown event:

1. keydown - The key is on its way down
2. keypress - The key is pressed down
3. keyup - The key is released

The keydown event occurs when a keyboard key is pressed down.

The keydown() method triggers the keydown event, or attaches a function to run when a keydown event occurs.

**Tip:** Use the event.which property to return which keyboard key was pressed.

## Syntax

Trigger the keydown event for the selected elements:

```
$(selector).keydown()
```

Try it

Attach a function to the keydown event:

```
$(selector).keydown(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the keydown event is triggered

## Try it Yourself - Examples

# jQuery keypress() Method

## Example

Count the number of key presses in an <input> field:

```
$( "input" ).keypress(function(){
    $( "span" ).text(i += 1);
});
```

## Definition and Usage

The order of events related to the keypress event:

1. keydown - The key is on its way down
2. keypress - The key is pressed down
3. keyup - The key is released

The keypress() method triggers the keypress event, or attaches a function to run when a keypress event occurs.

The keypress event is similar to the keydown event. The event occurs when a button is pressed down.

However, the keypress event is not fired for all keys (e.g. ALT, CTRL, SHIFT, ESC). Use the keydown() method to also check these keys.

## Syntax

Trigger the keypress event for the selected elements:

```
$(selector).keypress()
```

Try it

Attach a function to the keypress event:

```
$(selector).keypress(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the keypress event is triggered

# jQuery keyup() Method

## Example

Set the background color of an <input> field when a keyboard key is released:

```
$( "input" ).keyup(function(){
    $( "input" ).css( "background-color", "pink" );
});
```

## Definition and Usage

The order of events related to the keyup event:

1. keydown - The key is on its way down
2. keypress - The key is pressed down
3. keyup - The key is released

The keyup event occurs when a keyboard key is released.

The keyup() method triggers the keyup event, or attaches a function to run when a keyup event occurs.

**Tip:** Use the event.which property to return which key was pressed.

## Syntax

Trigger the keyup event for the selected elements:

```
$(selector).keyup()
```

Try it

Attach a function to the keyup event:

```
$(selector).keyup(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the keyup event is triggered.

## Try it Yourself - Examples

Determine which key was pressed. How use the event.which property to determine which key was pressed.

# jQuery live() Method

## Example

Hide or show a <p> element when a button is clicked:

```
$("button").live("click", function(){
    $("p").slideToggle();
});
```

## Definition and Usage

The `live()` method was deprecated in jQuery version 1.7, and removed in version 1.9. Use the `on()` method instead.

The `live()` method attaches one or more event handlers for selected elements, and specifies a function to run when the events occur.

Event handlers attached using the `live()` method will work for both current and FUTURE elements matching the selector (like a new element created by a script).

**Tip:** To remove event handlers, use the `die()` method.

## Syntax

```
$(selector).live(event,data,function)
```

Parameter	Description
<code>event</code>	Required. Specifies one or more events to attach to the elements. Multiple event values are separated by space. Must be a valid event.
<code>data</code>	Optional. Specifies additional data to pass along to the function
<code>function</code>	Required. Specifies the function to run when the event occurs

## Try it Yourself - Examples

[Add event handlers for future elements](#) How to use the `live()` method to add event handlers for elements not yet created.

# jQuery load() Method

## Example

Alert a text when an image is fully loaded:

```
$("img").load(function(){
    alert("Image loaded.");
});
```

## Definition and Usage

The `load()` method was deprecated in jQuery version 1.8 and **removed** in version 3.0.

The `load()` method attaches an event handler to the `load` event.

The `load` event occurs when a specified element has been loaded.

This event works with elements associated with a URL (image, script, frame, iframe), and the window object.

Depending on the browser, the `load` event may not trigger if the image is cached (Firefox and IE).

**Note:** There is also a jQuery AJAX method called `load()`. Which one is called, depends on the parameters.

## Syntax

```
$(selector).load(function)
```

Parameter	Description
<code>function</code>	Required. Specifies the function to run when the specified element is done loading

## Try it Yourself - Examples

Show some text when an image is fully loaded How to change the text of a `<div>` element when an image is fully loaded.

Alert a text when the page is fully loaded How to alert a text when the window object is fully loaded including images.

# jQuery mousedown() Method

## Example

Press down the left mouse button over a <div> element to insert some text:

```
$("div").mousedown(function(){
    $(this).after("Mouse button pressed down.");
});
```

## Definition and Usage

The mousedown event occurs when the left mouse button is pressed down over the selected element.

The mousedown() method triggers the mousedown event, or attaches a function to run when a mousedown event occurs.

**Tip:** This method is often used together with the [mouseup\(\)](#) method.

## Syntax

Trigger the mousedown event for the selected elements:

```
$(selector).mousedown()
```

Try it

Attach a function to the mousedown event:

```
$(selector).mousedown(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the mousedown event is triggered

# jQuery mouseenter() Method

## Example

Set the background color to yellow, when the mouse pointer enters a <p> element:

```
$("p").mouseenter(function(){
    $("p").css("background-color", "yellow");
});
```

## Definition and Usage

The mouseenter event occurs when the mouse pointer is over (enters) the selected element. The mouseenter() method triggers the mouseenter event, or attaches a function to run when a mouseenter event occurs..

**Note:** Unlike the mouseover event, the mouseenter event only triggers when the mouse pointer enters the selected element. The mouseover event is triggered if a mouse pointer enters any child elements as well. See the example at the end of the page for a demonstration.

**Tip:** This event is often used together with the mouseleave event.

## Syntax

Trigger the mouseenter event for the selected elements:

```
$(selector).mouseenter()
```

Try it

Attach a function to the mouseenter event:

```
$(selector).mouseenter(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the mouseenter event is triggered

## Try it Yourself - Examples

[The difference between mouseenter\(\), mouseover\(\) and mousemove\(\)](#). Demonstrates the difference between mouseenter(), mouseover() and mousemove().

# jQuery mouseleave() Method

## Example

Set the background color to gray, when the mouse pointer leaves a <p> element:

```
$( "p" ).mouseleave(function(){
    $( "p" ).css("background-color", "gray");
});
```

## Definition and Usage

The mouseleave event occurs when the mouse pointer leaves the selected element.

The mouseleave() method triggers the mouseleave event, or attaches a function to run when a mouseleave event occurs.

**Note:** Unlike the mouseout event, the mouseleave event only triggers when the mouse pointer leaves the selected elements. The mouseout event is triggered if a mouse pointer leaves any child elements as well as the selected element. See the example at the end of the page for a demonstration.

**Tip:** This event is often used together with the mouseenter event.

## Syntax

Trigger the mouseleave event for the selected elements:

```
$(selector).mouseleave()
```

Try it

Attach a function to the mouseleave event:

```
$(selector).mouseleave(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the mouseleave event is triggered

## Try it Yourself - Examples

[The difference between mouseleave\(\) and mouseout\(\)](#). Demonstrates the difference between mouseleave() and mouseout().

# jQuery mousemove() Method

## Example

Get the position of the mouse pointer within a page:

```
$(document).mousemove(function(event){  
    $("span").text(event.pageX + ", " + event.pageY);  
});
```

## Definition and Usage

The mousemove event occurs whenever the mouse pointer moves within the selected element.

The mousemove() method triggers the mousemove event, or attaches a function to run when a mousemove event occurs.

**Note:** Each time a user moves the mouse one pixel, a mousemove event occurs. It takes system resources to process all mousemove events. Use this event carefully.

## Syntax

Trigger the mousemove event for the selected elements:

```
$(selector).mousemove()
```

Try it

Attach a function to the mousemove event:

```
$(selector).mousemove(function)
```

Try it

### Parameter Description

<i>function</i>	Optional. Specifies the function to run when the mousemove event is triggered
-----------------	---

## Try it Yourself - Examples

The difference between mouseover(), mouseenter() and mousemove(). Demonstrates the difference between mouseover(), mouseenter() and mousemove().

# jQuery mouseout() Method

## Example

Set the background color to gray, when the mouse pointer leaves a <p> element:

```
$( "p" ).mouseout(function(){
    $( "p" ).css("background-color", "gray");
});
```

## Definition and Usage

The mouseout event occurs when the mouse pointer leaves the selected element.

The mouseout() method triggers the mouseout event, or attaches a function to run when a mouseout event occurs.

**Note:** Unlike the mouseleave event, the mouseout event is triggered if a mouse pointer leaves any child elements as well as the selected element. The mouseleave event only triggers when the mouse pointer leaves the selected element. See the example at the end of the page for a demonstration.

**Tip:** This event is often used together with the mouseover event.

## Syntax

Trigger the mouseout event for the selected elements:

```
$(selector).mouseout()
```

Try it

Attach a function to the mouseout event:

```
$(selector).mouseout(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the mouseout event is triggered

## Try it Yourself - Examples

[The difference between mouseout\(\) and mouseleave\(\)](#). Demonstrates the difference between mouseout() and mouseleave().

# jQuery mouseover() Method

## Example

Set the background color to yellow, when the mouse pointer is over a <p> element:

```
$("p").mouseover(function(){
    $("p").css("background-color", "yellow");
});
```

## Definition and Usage

The mouseover event occurs when the mouse pointer is over the selected element.

The mouseover() method triggers the mouseover event, or attaches a function to run when a mouseover event occurs.

**Note:** Unlike the mouseenter event, the mouseover event triggers if a mouse pointer enters any child elements as well as the selected element. The mouseenter event is only triggered when the mouse pointer enters the selected element. See the example at the end of the page for a demonstration.

**Tip:** This event is often used together with the mouseout event.

## Syntax

Trigger the mouseover event for the selected elements:

```
$(selector).mouseover()
```

Try it

Attach a function to the mouseover event:

```
$(selector).mouseover(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the mouseover event is triggered

## Try it Yourself - Examples

[The difference between mouseover\(\), mouseenter\(\) and mousemove\(\)](#). Demonstrates the difference between mouseover(), mouseenter() and mousemove().

# jQuery mouseup() Method

## Example

Release the left mouse button over a <div> element to insert some text:

```
$( "div" ).mouseup(function(){
  $(this).after("Mouse button released.");
});
```

## Definition and Usage

The mouseup event occurs when the left mouse button is released over the selected element. The mouseup() method triggers the mouseup event, or attaches a function to run when a mouseup event occurs.

**Tip:** This method is often used together with the [mousedown\(\)](#) method.

## Syntax

Trigger the mouseup event for the selected elements:

```
$(selector).mouseup()
```

[Try it](#)

Attach a function to the mouseup event:

```
$(selector).mouseup(function)
```

[Try it](#)

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the mouseup event is triggered

# jQuery off() Method

## Example

Remove the click event for all <p> elements:

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

## Definition and Usage

The off() method is most often used to remove event handlers attached with the [on\(\)](#) method. As of jQuery version 1.7, the off() method is the new replacement for the unbind(), die() and undelegate() methods. This method brings a lot of consistency to the API, and we recommend that you use this method, as it simplifies the jQuery code base.

**Note:** To remove specific event handlers, the selector string must match the one passed to the on() method, when the event handler was attached.

**Tip:** To attach an event that only runs once and then removes itself, use the [one\(\)](#) method.

## Syntax

```
$(selector).off(event, selector, function(eventObj), map)
```

Parameter	Description
<code>event</code>	Required. Specifies one or more events or namespaces to remove from the selected element(s). Multiple event values are separated by a space. Must be a valid event
<code>selector</code>	Optional. A selector which should match the one originally passed to the on() method when attaching event handlers
<code>function(eventObj)</code>	Optional. Specifies the function to run when the event occurs
<code>map</code>	Specifies an event map ( <code>{event:function, event:function, ...}</code> ) containing one or more event to attach to the elements, and functions to run when the events occur

## Try it Yourself - Examples

Remove all click event handlers, added with on(). How to remove all click event handlers for all <p> elements added with the on() method.

Remove one specific event function added with on(). How to remove a specific function added with the on() method.

Remove an event handler using an event object How to remove an event handler after the event has been triggered a certain number of times.

# jQuery on() Method

## Example

Attach a click event to the <p> element:

```
$("p").on("click", function(){
    alert("The paragraph was clicked.");
});
```

## Definition and Usage

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

As of jQuery version 1.7, the on() method is the new replacement for the bind(), live() and delegate() methods. This method brings a lot of consistency to the API, and we recommend that you use this method, as it simplifies the jQuery code base.

**Note:** Event handlers attached using the on() method will work for both current and FUTURE elements (like a new element created by a script).

**Tip:** To remove event handlers, use the off() method.

**Tip:** To attach an event that only runs once and then removes itself, use the one() method.

# Syntax

```
$(selector).on(event,childSelector,data,function,map)
```

Parameter	Description
<i>event</i>	Required. Specifies one or more event(s) or namespaces to attach to the selected elements. Multiple event values are separated by space. Must be a valid event
<i>childSelector</i>	Optional. Specifies that the event handler should only be attached to the specified child elements (and not the selector itself, like the deprecated <code>delegate()</code> method).
<i>data</i>	Optional. Specifies additional data to pass along to the function
<i>function</i>	Required. Specifies the function to run when the event occurs
<i>map</i>	Specifies an event map ( <code>{event:function, event:function, ...}</code> ) containing one or more event to attach to the selected elements, and functions to run when the events occur

## Try it Yourself - Examples

[Attach multiple events](#) How to attach multiple events to an element.

[Attach multiple event handlers using the map parameter](#) How to attach multiple event handlers to the selected elements using the map parameter.

[Attach a custom event on an element](#) How to attach a customized namespace event on an element.

[Pass along data to the function](#) How to pass along data to the function.

[Add event handlers for future elements](#) Show that the `on()` method also works for elements not yet created.

[Remove an event handler](#) How to remove an event handler using the `off()` method.

# jQuery one() Method

## Example

Increase the text size of a <p> element when it is clicked (the event will only trigger once for each <p> element):

```
$( "p" ).one( "click", function(){
    $(this).animate({fontSize: "+=6px"});
});
```

## Definition and Usage

The one() method attaches one or more event handlers for the selected elements, and specifies a function to run when the event occurs.

When using the one() method, the event handler function is only run **ONCE** for each element.

## Syntax

```
$(selector).one(event,data,function)
```

Parameter	Description
event	Required. Specifies one or more events to attach to the elements. Multiple event values are separated by space. Must be a valid event.
data	Optional. Specifies additional data to pass along to the function
function	Required. Specifies the function to run when the event occurs

## Try it Yourself - Examples

[Attach two events](#) How to attach two event handlers ("click" and "dblclick") to a <p> element.

# jQuery \$.proxy() Method

## Example

Enforce the context of the "test" function, inside objPerson:

```
$("button").click($.proxy(objPerson, "test"));
```

## Definition and Usage

The \$.proxy method takes an existing function and returns a new one with a particular context.

This method is often used for attaching events to an element where the context is pointing back to a different object.

**Tip:** If you bind the function returned from \$.proxy, jQuery will still unbind the correct function if passed the original.

## Syntax 1

```
$(selector).proxy(function,context)
```

Try it

## Syntax 2

```
$(selector).proxy(context,name)
```

Try it

Parameter	Description
<i>function</i>	The existing function to be called
<i>context</i>	The name of the object where the function lies
<i>name</i>	The existing function whose context will be changed (should be a property of the context object).

# jQuery ready() Method

## Example

Use ready() to make a function available after the document is loaded:

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

## Definition and Usage

The ready event occurs when the DOM (document object model) has been loaded. Because this event occurs after the document is ready, it is a good place to have all other jQuery events and functions. Like in the example above.

The ready() method specifies what happens when a ready event occurs.

**Tip:** The ready() method should not be used together with <body onload="">.

## Syntax

Two syntaxes can be used:

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

The ready() method can only be used on the current document, so no selector is required:

```
$(function)
```

Parameter	Description
<i>function</i>	Required. Specifies the function to run after the document is loaded

# jQuery resize() Method

## Example

Count the number of times the browser window is resized:

```
$(window).resize(function(){
    $('span').text(x += 1);
});
```

## Definition and Usage

The resize event occurs when the browser window changes size.

The resize() method triggers the resize event, or attaches a function to run when a resize event occurs.

## Syntax

Trigger the resize event for the selected elements:

```
$(selector).resize()
```

Try it

Attach a function to the resize event:

```
$(selector).resize(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the resize event is triggered

# jQuery scroll() Method

## Example

Count the number of times the scroll is used for an element:

```
$( "div" ).scroll(function(){
    $( "span" ).text(x += 1);
});
```

## Definition and Usage

The scroll event occurs when the user scrolls in the specified element.

The scroll event works for all scrollable elements and the window object (browser window).

The scroll() method triggers the scroll event, or attaches a function to run when a scroll event occurs.

## Syntax

Trigger the scroll event for the selected elements:

```
$(selector).scroll()
```

Try it

Attach a function to the scroll event:

```
$(selector).scroll(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the scroll event is triggered

## Try it Yourself - Examples

[Toggle between classes on different scroll positions](#) How to toggle between classes on different scroll positions.

# jQuery select() Method

## Example

Alert a message when a text is selected in a text field:

```
$("input").select(function(){
    alert("Text marked!");
});
```

## Definition and Usage

The select event occurs when a text is selected (marked) in a text area or a text field. The select() method triggers the select event, or attaches a function to run when a select event occurs.

## Syntax

Trigger the select event for the selected elements:

```
$(selector).select()
```

Try it

Attach a function to the select event:

```
$(selector).select(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the select event is triggered

# jQuery submit() Method

## Example

Display an alert when a form is submitted:

```
$("form").submit(function(){
    alert("Submitted");
});
```

## Definition and Usage

The submit event occurs when a form is submitted.

This event can only be used on <form> elements.

The submit() method triggers the submit event, or attaches a function to run when a submit event occurs.

## Syntax

Trigger the submit event for the selected elements:

```
$(selector).submit()
```

Try it

Attach a function to the submit event:

```
$(selector).submit(function)
```

Try it

Parameter	Description
<i>function</i>	Optional. Specifies the function to run when the submit event is triggered

## Try it Yourself - Examples

[Prevent the default action of a submit button](#) How to prevent the form being submitted using the event.preventDefault() method.

# jQuery toggle() Method

## Example

Toggle between colors when clicking on a <p> element:

```
$("p").toggle(  
    function(){ $("p").css({"color": "red"}); },  
    function(){ $("p").css({"color": "blue"}); },  
    function(){ $("p").css({"color": "green"}); }  
);
```

## Definition and Usage

The **toggle()** method was deprecated in jQuery version 1.8, and **removed** in version 1.9.

The **toggle()** method attaches two or more functions to toggle between for the click event for the selected elements.

When clicking on an element, the first specified function fires, when clicking again, the second function fires, and so on.

**Note:** There is also a jQuery Effects method called **toggle()**. Which one is called, depends on the parameters.

## Syntax

```
$(selector).toggle(function)
```

Parameter	Description
<i>function</i>	Required. A function to run every time the selected element is clicked

## Try it Yourself - Examples

[Toggle between multiple functions](#) How to toggle between multiple functions.

# jQuery trigger() Method

## Example

Trigger the select event of an <input> element:

```
$( "button" ).click(function(){
    $( "input" ).trigger( "select" );
});
```

## Definition and Usage

The trigger() method triggers the specified event and the default behavior of an event (like form submission) for the selected elements.

This method is similar to the [triggerHandler\(\)](#) method, except that triggerHandler() does not trigger the default behavior of the event.

## Syntax

```
$(selector).trigger(event, eventObj, param1, param2, ...)
```

Parameter	Description
<i>event</i>	Required. Specifies the event to trigger for the specified element. Can be a custom event, or any of the standard events
<i>param1, param2, ...</i>	Optional. Additional parameters to pass on to the event handler. Additional parameters are especially useful with custom events

## Try it Yourself - Examples

[Pass additional parameters on to a custom event](#) How to pass along additional parameters to a custom event handler.

[The difference between trigger\(\) and triggerHandler\(\)](#) Demonstrates the difference between trigger() and triggerHandler().

# jQuery triggerHandler() Method

## Example

Trigger the select event of an <input> element:

```
$( "button" ).click(function(){
    $( "input" ).triggerHandler( "select" );
});
```

## Definition and Usage

The triggerHandler() method triggers the specified event for the selected element.

This method is similar to the [trigger\(\)](#) method, except that trigger() also triggers the default behavior of an event (like form submission).

## Syntax

```
$(selector).triggerHandler(event,param1,param2,...)
```

Parameter	Description
event	Required. Specifies the event to trigger for the specified element
param1,param2,...	Optional. Additional parameters to pass on to the event handler. Additional parameters are especially useful with custom events

## Try it Yourself - Examples

[Pass additional parameters on to a custom event](#) How to pass along additional parameters to a custom event handler.

[The difference between trigger\(\) and triggerHandler\(\)](#) Demonstrates the difference between trigger() and triggerHandler().

# jQuery unbind() Method

## Example

Remove all event handlers for all <p> elements:

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

## Definition and Usage

The `unbind()` method was deprecated in version 3.0. Use the `off()` method instead.

The `unbind()` method removes event handlers from selected elements.

This method can remove all or selected event handlers, or stop specified functions from running when the event occurs.

This method can also unbind event handlers using an event object. This is used to unbind an event from within itself (like removing an event handler after the event has been triggered a certain number of times).

**Note:** If no parameters are specified, the `unbind()` method will remove ALL event handlers from the specified element.

**Note:** The `unbind()` method works on any event handler attached with jQuery.

## Syntax

```
$(selector).unbind(event,function,eventObj)
```

Parameter	Description
<code>event</code>	Optional. Specifies one or more events to remove from the elements. Multiple event values are separated by space. If this is the only parameter specified, all functions bound to the specified event will be removed.
<code>function</code>	Optional. Specifies the name of the function to unbind from the specified event for the element
<code>eventObj</code>	Optional. Specifies the event object to remove to use. The <code>eventObj</code> parameter comes from the event binding function

# jQuery undelegate() Method

## Example

Remove all event handlers added with the delegate() method from all elements:

```
$("body").undelegate();
```

## Definition and Usage

The undelegate() method was deprecated in version 3.0. Use the [on\(\)](#) method instead.

The undelegate() method removes one or more event handlers, added with the [delegate\(\)](#) method.

## Syntax

```
$(selector).undelegate(selector, event, function)
```

Parameter	Description
<i>selector</i>	Optional. Specifies the selector to remove event handlers from
<i>event</i>	Optional. Specifies one or more event types to remove handler functions from
<i>function</i>	Optional. Specifies a specific event handler function to remove

## Try it Yourself - Examples

[Remove all click event handlers](#) How to use the undelegate() method to remove all click event handlers for all <p> elements.

[Remove a specific function added with delegate\(\)](#) How to use the undelegate() method to remove only a specific function.

# jQuery unload() Method

## Example

Alert a message when navigating away from the page:

```
$(window).unload(function(){
    alert("Goodbye!");
});
```

## Definition and Usage

The `unload()` method was deprecated in jQuery version 1.8 and **removed** in version 3.0.

The `unload` event occurs when the user navigates away from the page.

The `unload` event is triggered when:

- a link to leave the page is clicked
- a new URL is typed in the address bar
- the forward or back buttons are used
- the browser window is closed
- the page is reloaded

The `unload()` method specifies what happens when a `unload` event occurs.

The `unload()` method should only be used on the `window` object.

**Note:** The `unload` event might work differently in different browsers. Be sure to test this method in all browsers, before use.

## Syntax

```
$(selector).unload(function)
```

Parameter	Description
<code>function</code>	Required. Specifies the function to run when the <code>unload</code> event is triggered

# jQuery Effect Methods

## jQuery Effect Methods

The following table lists all the jQuery methods for creating animation effects.

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

# jQuery animate() Method

## Example

"Animate" an element, by changing its height:

```
$("button").click(function(){
    $("#box").animate({height: "300px"});
});
```

## Definition and Usage

The animate() method performs a custom animation of a set of CSS properties.

This method changes an element from one state to another with CSS styles. The CSS property value is changed gradually, to create an animated effect.

Only numeric values can be animated (like "margin:30px"). String values cannot be animated (like "background-color:red"), except for the strings "show", "hide" and "toggle". These values allow hiding and showing the animated element.

**Tip:** Use "+=" or "-=" for relative animations.

## Syntax

```
(selector).animate({styles}, speed, easing, callback)
```

Parameter	Description
styles	<p>Required. Specifies one or more CSS properties/values to animate.</p> <p><b>Note:</b> The property names must be camel-cased when used with the animate() method: You will need to write paddingLeft instead of padding-left, marginRight instead of margin-right, and so on.</p> <p>Properties that can be animated:</p> <ul style="list-style-type: none"><li>• <u>backgroundPositionX</u></li><li>• <u>backgroundPositionY</u></li><li>• <u>borderWidth</u></li><li>• <u>borderBottomWidth</u></li><li>• <u>borderLeftWidth</u></li><li>• <u>borderRightWidth</u></li><li>• <u>borderTopWidth</u></li><li>• <u>borderSpacing</u></li><li>• <u>margin</u></li><li>• <u>marginBottom</u></li></ul>

Parameter	Description
	<ul style="list-style-type: none"> <li>• <u>marginLeft</u></li> <li>• <u>marginRight</u></li> <li>• <u>marginTop</u></li> <li>• <u>opacity</u></li> <li>• <u>outlineWidth</u></li> <li>• <u>padding</u></li> <li>• <u>paddingBottom</u></li> <li>• <u>paddingLeft</u></li> <li>• <u>paddingRight</u></li> <li>• <u>paddingTop</u></li> <li>• <u>height</u></li> <li>• <u>width</u></li> <li>• <u>maxHeight</u></li> <li>• <u>maxWidth</u></li> <li>• <u>minHeight</u></li> <li>• <u>minWidth</u></li> <li>• <u>fontSize</u></li> <li>• <u>bottom</u></li> <li>• <u>left</u></li> <li>• <u>right</u></li> <li>• <u>top</u></li> <li>• <u>letterSpacing</u></li> <li>• <u>wordSpacing</u></li> <li>• <u>lineHeight</u></li> <li>• <u>textIndent</u></li> </ul> <p>Only numeric values can be animated (like "margin:30px"). String values cannot be animated (like "background-color:red"), except for the strings "show", "hide" and "toggle". These values allow hiding and showing the animated element.</p>
<i>speed</i>	<p>Optional. Specifies the speed of the animation. Default value is 400 milliseconds</p> <p>Possible values:</p> <ul style="list-style-type: none"> <li>• milliseconds (like 100, 1000, 5000, etc)</li> <li>• "slow"</li> <li>• "fast"</li> </ul>

Parameter	Description
<code>easing</code>	<p>Optional. Specifies the speed of the element in different points of the animation. Default value is "swing". Possible values:</p> <ul style="list-style-type: none"> <li>"swing" - moves slower at the beginning/end, but faster in the middle</li> <li>"linear" - moves in a constant speed</li> </ul> <p><b>Tip:</b> More easing functions are available in external plugins.</p>
<code>callback</code>	Optional. A function to be executed after the animation completes. To learn more about callback, please read our <a href="#">jQuery Callback</a> chapter

## Alternate Syntax

```
(selector).animate({styles}, {options})
```

Parameter	Description
<code>styles</code>	Required. Specifies one or more CSS properties/values to animate (See possible values above)
<code>options</code>	<p>Optional. Specifies additional options for the animation. Possible values:</p> <ul style="list-style-type: none"> <li>duration - sets the speed of the animation</li> <li>easing - specifies the easing function to use</li> <li>complete - specifies a function to be executed after the animation completes</li> <li>step - specifies a function to be executed for each step in the animation</li> <li>progress - specifies a function to be executed after each step in the animation</li> <li>queue - a Boolean value specifying whether or not to place the animation in the effects queue</li> <li>specialEasing - a map of one or more CSS properties from the <code>styles</code> parameter, and their corresponding easing functions</li> <li>start - specifies a function to be executed when the animation begins</li> <li>done - specifies a function to be executed when the animation ends</li> <li>fail - specifies a function to be executed if the animation fails to complete</li> <li>always - specifies a function to be executed if the animation stops without completing</li> </ul>

# jQuery clearQueue() Method

## Example

Stop the remaining functions in the queue:

```
$("button").click(function(){
    $("div").clearQueue();
});
```

## Definition and Usage

The clearQueue() method removes all items from the queue that have not yet been run. Note that when a function has started to run, it runs until it is completed.

Related methods:

- [queue\(\)](#) - Shows the queued functions
- [dequeue\(\)](#) - Removes the next function from the queue, and then executes the function

**Tip:** Unlike the [stop\(\)](#) method (that only works with animation), the clearQueue() method can remove any queued functions.

## Syntax

```
$(selector).clearQueue(queueName)
```

Parameter	Description
queueName	Optional. Specifies the name of the queue. The default is "fx", the standard effects queue

## Try it Yourself - Example

[Using queue-related methods together](#) How to use queue(), dequeue() and clearQueue() together.

# jQuery delay() Method

## Example

Delay different <div> elements:

```
$("button").click(function(){
    $("#div1").delay("slow").fadeIn();
    $("#div2").delay("fast").fadeIn();
});
```

## Definition and Usage

The delay() method sets a timer to delay the execution of the next item in the queue.

## Syntax

```
$(selector).delay(speed,queueName)
```

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the delay Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>queueName</i>	Optional. Specifies the name of the queue Default is "fx", the standard effects queue

## Try it Yourself - Example

[delay\(\) and animate\(\)](#) How to delay an animation.

# jQuery dequeue() Method

## Example

Remove the next function from the queue, and then execute the function:

```
$("div").queue(function(){
    $("div").css("background-color", "red");
    $("div").dequeue();
});
```

## Definition and Usage

The dequeue() method removes the next function from the queue, and then executes the function.

A queue is one or more function(s) waiting to run.

The dequeue() method is used together with the [queue\(\)](#) method.

**Note:** You should ensure that the dequeue() method is called after adding a function with queue(), to allow the process to continue.

## Syntax

```
$(selector).dequeue(queueName)
```

Parameter	Description
<i>queueName</i>	Optional. Specifies the name of the queue Default is "fx", the standard effects queue

## Try it Yourself - Example

[Using queue-related methods together](#) How to use queue(), dequeue() and clearQueue() together.

# jQuery fadeIn() Method

## Example

Fade in all <p> elements:

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

## Definition and Usage

The fadeIn() method gradually changes the opacity, for selected elements, from hidden to visible (fading effect).

**Tip:** This method is often used together with the [fadeOut\(\)](#) method.

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the fading effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the fadeIn() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[fadeIn\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the fading effect.

[fadeIn\(\) - Using the callback parameter](#) How to use the callback parameter when fading in and out an element.

# jQuery Effect fadeOut() Method

## Example

Fade out all <p> elements:

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

## Definition and Usage

The fadeOut() method gradually changes the opacity, for selected elements, from visible to hidden (fading effect).

**Tip:** This method is often used together with the [fadeIn\(\)](#) method.

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the fading effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the fadeOut() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[fadeOut\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the fading effect.

[fadeIn\(\) - Using the callback parameter](#) How to use the callback parameter when gradually changing the opacity for selected elements.

# jQuery Effect fadeTo() Method

## Example

Gradually change the opacity of all <p> elements:

```
$("button").click(function(){
    $("p").fadeTo(1000, 0.4);
});
```

## Definition and Usage

The fadeTo() method gradually changes the opacity, for selected elements, to a specified opacity (fading effect).

# Syntax

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

Parameter	Description
<i>speed</i>	Required. Specifies the speed of the fading effect. Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>opacity</i>	Required. Specifies the opacity to fade to. Must be a number between 0.00 and 1.00
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul>
<i>callback</i>	Optional. A function to be executed after the fadeTo() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[fadeTo\(\) - Using the callback parameter](#) How to use the callback parameter when fading in and out an element.

# jQuery fadeToggle() Method

## Example

Toggle between fading in and fading out different boxes:

```
$("button").click(function(){
    $("#div1").fadeToggle();
});
```

## Definition and Usage

The fadeToggle() method toggles between the [fadeIn\(\)](#) and [fadeOut\(\)](#) methods.

If the elements are faded out, fadeToggle() will fade them in.

If the elements are faded in, fadeToggle() will fade them out.

**Note:** Hidden elements will not be displayed at all (no longer affects the layout of the page).

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the fading effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the effect. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the fadeToggle() method is completed To learn more about callback, please read our <a href="#">jQuery Callback</a> chapter

## Try it Yourself - Examples

[fadeToggle\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the fading effect.

[fadeToggle\(\) - Using the callback parameter](#) How to use the callback parameter to execute a function after the fadeToggle() method is completed.

# jQuery finish() Method

## Example

Finish the currently running animation:

```
$("#complete").click(function(){
    $("div").finish();
});
```

## Definition and Usage

The `finish()` method stops the currently-running animations, removes all queued animations, and completes all animations for the selected elements.

This method is similar to the `.stop(true,true)` method, except that `finish()` also causes the CSS property of all queued animations to stop.

## Syntax

```
$(selector).finish(queueName)
```

Parameter	Description
<code>queueName</code>	Optional. Specifies the name of the queue to stop animations

# jQuery hide() Method

## Example

Hide all <p> elements:

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

## Definition and Usage

The hide() method hides the selected elements.

**Tip:** This is similar to the CSS property display:none.

**Note:** Hidden elements will not be displayed at all (no longer affects the layout of the page).

**Tip:** To show hidden elements, look at the [show\(\)](#) method.

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the hide effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the hide() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[hide\(\) - Using the speed parameter](#) How to use the speed parameter when hiding/showing an element.

[hide\(\) - Using the callback parameter](#) How to use the callback parameter when hiding/showing an element.

# jQuery queue() Method

## Example

Display the length of a queue in a <span> element:

```
$("span").text(div.queue().length);
```

## Definition and Usage

The queue() method shows the queue of functions to be executed on the selected elements.

A queue is one or more function(s) waiting to run.

The queue() method can be used together with the [dequeue\(\)](#) method.

## Syntax

```
$(selector).queue(queueName)
```

Parameter	Description
queueName	Optional. Specifies the name of the queue Default is "fx", the standard effects queue

## Try it Yourself - Example

[Using queue-related methods together](#) How to use queue(), dequeue() and clearQueue() together.

[Count the length of the queue + loop the queue](#) How to count the length of the queue + loop the queue.

# jQuery Effect show() Method

## Example

Show all hidden <p> elements:

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

## Definition and Usage

The show() method shows the hidden, selected elements.

**Note:** show() works on elements hidden with jQuery methods and display:none in CSS (but not visibility:hidden).

**Tip:** To hide elements, look at the [hide\(\)](#) method.

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the show effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the show() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[show\(\) - Using the speed parameter](#) How to use the speed parameter when hiding/showing an element.

[show\(\) - Using the callback parameter](#) How to use the callback parameter when hiding/showing an element.

# jQuery slideDown() Method

## Example

Slide-down (show) all hidden <p> elements:

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

## Definition and Usage

The slideDown() method slides-down (shows) the selected elements.

**Note:** slideDown() works on elements hidden with jQuery methods and display:none in CSS (but not visibility:hidden).

**Tip:** To slide-up (hide) elements, look at the [slideUp\(\)](#) method.

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the slide effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins.</p>
<i>callback</i>	Optional. A function to be executed after the slideDown() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[slideDown\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the slide effect.

[slideDown\(\) - Using the callback parameter](#) How to use the callback parameter when sliding up and down the selected elements.

# jQuery slideToggle() Method

## Example

Toggle between slideUp() and slideDown() for all <p> elements:

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

## Definition and Usage

The slideToggle() method toggles between slideUp() and slideDown() for the selected elements.

This method checks the selected elements for visibility. slideDown() is run if an element is hidden. slideUp() is run if an element is visible - This creates a toggle effect.

# Syntax

```
$(selector).slideToggle(speed,easing,callback)
```

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the slide effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the slideToggle() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[slideToggle\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the slide effect.

[slideToggle\(\) - Using the callback parameter](#) How to use the callback parameter when toggling between slideUp() and slideDown().

# jQuery slideUp() Method

## Example

Slide-up (hide) all <p> elements:

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

## Definition and Usage

The slideUp() method slides-up (hides) the selected elements.

**Note:** Hidden elements will not be displayed at all (no longer affects the layout of the page).

**Tip:** To slide-down (show) elements, look at the [slideDown\(\)](#) method.

# Syntax

```
$(selector).slideUp(speed,easing,callback)
```

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the slide effect. Default value is 400 milliseconds Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the slideUp() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[slideUp\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the slide effect.

[slideUp\(\) - Using the callback parameter](#) How to use the callback parameter when sliding up and down the selected elements.

# jQuery stop() Method

## Example

Stop the currently running animation:

```
$("#stop").click(function(){
    $("div").stop();
});
```

## Definition and Usage

The stop() method stops the currently running animation for the selected elements.

## Syntax

```
$(selector).stop(stopAll,goToEnd)
```

Parameter	Description
<i>stopAll</i>	Optional. A Boolean value specifying whether or not to stop the queued animations as well. Default is false
<i>goToEnd</i>	Optional. A Boolean value specifying whether or not to complete all animations immediately. Default is false

## Try it Yourself - Examples

[Stop queued animations as well](#) How to stop the queued animations.

[Complete all animations immediately](#) How to complete all animations immediately.

# jQuery toggle() Method

## Example

Toggle between hide and show for all <p> elements:

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

## Definition and Usage

The toggle() method toggles between hide() and show() for the selected elements.

This method checks the selected elements for visibility. show() is run if an element is hidden. hide() is run if an element is visible - This creates a toggle effect.

**Note:** Hidden elements will not be displayed at all (no longer affects the layout of the page).

**Tip:** This method can also be used to toggle between custom functions.

# Syntax

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

Parameter	Description
<i>speed</i>	Optional. Specifies the speed of the hide/show effect Possible values: <ul style="list-style-type: none"><li>• milliseconds</li><li>• "slow"</li><li>• "fast"</li></ul>
<i>easing</i>	Optional. Specifies the speed of the element in different points of the animation. Default value is "swing" Possible values: <ul style="list-style-type: none"><li>• "swing" - moves slower at the beginning/end, but faster in the middle</li><li>• "linear" - moves in a constant speed</li></ul> <p><b>Tip:</b> More easing functions are available in external plugins</p>
<i>callback</i>	Optional. A function to be executed after the toggle() method is completed To learn more about callback, visit our <a href="#">jQuery Callback chapter</a>

## Try it Yourself - Examples

[toggle\(\) - Using the speed parameter](#) How to use the speed parameter to specify the speed of the hide/show effect.

[toggle\(\) - Using the callback parameter](#) How to use the callback parameter when toggling between the hide/show effect.

# jQuery HTML / CSS Methods

## jQuery HTML / CSS Methods

The following table lists all the methods used to manipulate the HTML and CSS.

The methods below work for both HTML and XML documents. Exception: the `html()` method.

<b>Method</b>	<b>Description</b>
<code>addClass()</code>	Adds one or more class names to selected elements
<code>after()</code>	Inserts content after selected elements
<code>append()</code>	Inserts content at the end of selected elements
<code>appendTo()</code>	Inserts HTML elements at the end of selected elements
<code>attr()</code>	Sets or returns attributes/values of selected elements
<code>before()</code>	Inserts content before selected elements
<code>clone()</code>	Makes a copy of selected elements
<code>css()</code>	Sets or returns one or more style properties for selected elements
<code>detach()</code>	Removes selected elements (keeps data and events)
<code>empty()</code>	Removes all child nodes and content from selected elements
<code>hasClass()</code>	Checks if any of the selected elements have a specified class name
<code>height()</code>	Sets or returns the height of selected elements
<code>html()</code>	Sets or returns the content of selected elements
<code>innerHeight()</code>	Returns the height of an element (includes padding, but not border)
<code>innerWidth()</code>	Returns the width of an element (includes padding, but not border)
<code>insertAfter()</code>	Inserts HTML elements after selected elements
<code>insertBefore()</code>	Inserts HTML elements before selected elements
<code>offset()</code>	Sets or returns the offset coordinates for selected elements (relative to the document)
<code>offsetParent()</code>	Returns the first positioned parent element
<code>outerHeight()</code>	Returns the height of an element (includes padding and border)

<b>Method</b>	<b>Description</b>
<u>outerWidth()</u>	Returns the width of an element (includes padding and border)
<u>position()</u>	Returns the position (relative to the parent element) of an element
<u>prepend()</u>	Inserts content at the beginning of selected elements
<u>prependTo()</u>	Inserts HTML elements at the beginning of selected elements
<u>prop()</u>	Sets or returns properties/values of selected elements
<u>remove()</u>	Removes the selected elements (including data and events)
<u>removeAttr()</u>	Removes one or more attributes from selected elements
<u>removeClass()</u>	Removes one or more classes from selected elements
<u>removeProp()</u>	Removes a property set by the prop() method
<u>replaceAll()</u>	Replaces selected elements with new HTML elements
<u>replaceWith()</u>	Replaces selected elements with new content
<u>scrollLeft()</u>	Sets or returns the horizontal scrollbar position of selected elements
<u>scrollTop()</u>	Sets or returns the vertical scrollbar position of selected elements
<u>text()</u>	Sets or returns the text content of selected elements
<u>toggleClass()</u>	Toggles between adding/removing one or more classes from selected elements
<u>unwrap()</u>	Removes the parent element of the selected elements
<u>val()</u>	Sets or returns the value attribute of the selected elements (for form elements)
<u>width()</u>	Sets or returns the width of selected elements
<u>wrap()</u>	Wraps HTML element(s) around each selected element
<u>wrapAll()</u>	Wraps HTML element(s) around all selected elements
<u>wrapInner()</u>	Wraps HTML element(s) around the content of each selected element

# jQuery addClass() Method

## Example

Add a class name to the first <p> element:

```
$("button").click(function(){
    $("p:first").addClass("intro");
});
```

## Definition and Usage

The addClass() method adds one or more class names to the selected elements.

This method does not remove existing class attributes, it only adds one or more class names to the class attribute.

**Tip:** To add more than one class, separate the class names with spaces.

## Syntax

```
$(selector).addClass(classname,function(index,currentclass))
```

Parameter	Description
classname	Required. Specifies one or more class names to be added
function(index,currentclass)	Optional. Specifies a function that returns one or more class names to be added <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentclass</i> - Returns the current class name of the selected element</li></ul>

## Try it Yourself - Examples

[Add two classes to an element](#) How to add two class names to a specified element.

[Add classes using a function](#) How to add classes to selected elements using a function.

[Change the class name of an element](#) How to use addClass() and removeClass() to remove one class name, and add a new class name.

# jQuery after() Method

## Example

Insert content after each <p> element:

```
$("button").click(function(){
    $("p").after("<p>Hello world!</p>");
});
```

## Definition and Usage

The after() method inserts specified content after the selected elements.

**Tip:** To insert content before selected elements, use the before() method.

## Syntax

```
$(selector).after(content,function(index))
```

Parameter	Description
<i>content</i>	Required. Specifies the content to insert (can contain HTML tags) Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<i>function(index)</i>	Specifies a function that returns the content to insert <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li></ul>

## Try it Yourself - Examples

[after\(\) - Create content with HTML, jQuery and DOM](#) Insert content with the after() method.

[Insert content using a function](#) How to use a function to insert content after selected elements.

# jQuery append() Method

## Example

Insert content at the end of all <p> elements:

```
$("button").click(function(){
    $("p").append("<b>Appended text</b>");
});
```

## Definition and Usage

The append() method inserts specified content at the end of the selected elements.

**Tip:** To insert content at the beginning of the selected elements, use the [prepend\(\)](#) method.

## Syntax

```
$(selector).append(content,function(index,html))
```

Parameter	Description
<i>content</i>	Required. Specifies the content to insert (can contain HTML tags) Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<i>function(index,html)</i>	Optional. Specifies a function that returns the content to insert <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>html</i> - Returns the current HTML of the selected element</li></ul>

## Try it Yourself - Examples

[append\(\) - Create content with HTML, jQuery and DOM](#) Insert content with the append() method.

[Append content using a function](#) Using a function to insert content at the end of the selected elements.

# jQuery appendTo() Method

## Example

Insert a `<span>` element at the end of each `<p>` element:

```
$("button").click(function(){
    $("<span>Hello World!</span>").appendTo("p");
});
```

## Definition and Usage

The `appendTo()` method inserts HTML elements at the end of the selected elements.

**Tip:** To insert HTML elements at the beginning of the selected elements, use the `prependTo()` method.

## Syntax

```
$(content).appendTo(selector)
```

Parameter	Description
<code>content</code>	Required. Specifies the content to insert (must contain HTML tags). <b>Note:</b> If <code>content</code> is an existing element, it will be moved from its current position, and inserted at the end of the selected elements.
<code>selector</code>	Required. Specifies on which elements to append the content to

## Try it Yourself - Examples

[Insert an existing element](#) How to use the `appendTo()` method to insert an existing element at the end of each selected element.

# jQuery attr() Method

## Example

Set the width attribute of an image:

```
$("button").click(function(){
    $("img").attr("width", "500");
});
```

## Definition and Usage

The attr() method sets or returns attributes and values of the selected elements.

When this method is used to **return** the attribute value, it returns the value of the FIRST matched element.

When this method is used to **set** attribute values, it sets one or more attribute/value pairs for the set of matched elements.

# Syntax

Return the value of an attribute:

```
$(selector).attr(attribute)
```

Set the attribute and value:

```
$(selector).attr(attribute,value)
```

Set attribute and value using a function:

```
$(selector).attr(attribute,function(index,currentvalue))
```

Set multiple attributes and values:

```
$(selector).attr({attribute:value, attribute:value,...})
```

Parameter	Description
<i>attribute</i>	Specifies the name of the attribute
<i>value</i>	Specifies the value of the attribute
<i>function(index,currentvalue)</i>	Specifies a function that returns the attribute value to set <ul style="list-style-type: none"><li>• <i>index</i> - Receives the index position of the element in the set</li><li>• <i>currentvalue</i> - Receives the current attribute value of selected elements</li></ul>

## Try it Yourself - Examples

[Return attribute value](#) How to return the value of an attribute for an element.

[Set attribute and value using a function](#) How to use a function to set the attribute value for an element.

[Set multiple attribute and value pairs](#) How to set multiple attributes/values for an element.

# jQuery before() Method

## Example

Insert content before each <p> element:

```
$("button").click(function(){
    $("p").before("<p>Hello world!</p>");
});
```

## Definition and Usage

The before() method inserts specified content in front of (before) the selected elements.

**Tip:** To insert content after selected elements, use the [after\(\)](#) method.

## Syntax

```
$(selector).before(content,function(index))
```

Parameter	Description
<i>content</i>	Specifies the content to insert (can contain HTML tags) Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<i>function(index)</i>	Optional. Specifies a function that returns the content to insert <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li></ul>

## Try it Yourself - Examples

[before\(\) - Create content with HTML, jQuery and DOM](#) Insert content with the before() method.

[Insert content using a function](#) How to use a function to insert content before selected elements.

# jQuery clone() Method

## Example

Clone all <p> elements and insert them at the end of the <body> element:

```
$("button").click(function(){
    $("p").clone().appendTo("body");
});
```

## Definition and Usage

The clone() method makes a copy of selected elements, including child nodes, text and attributes.

## Syntax

```
$(selector).clone(true|false)
```

Parameter	Description
true	Specifies that event handlers also should be copied
false	Default. Specifies that event handlers should not be copied

## Try it Yourself - Examples

[Copy an element including event handlers](#) How to use the clone() method to copy an element, including event handlers.

# jQuery css() Method

## Example

Set the color property of all <p> elements:

```
$("button").click(function(){
    $("p").css("color", "red");
});
```

## Definition and Usage

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

**When used to return properties:** This method returns the specified CSS property value of the FIRST matched element. However, shorthand CSS properties (like "background" and "border") are not fully supported and may give different results in different browsers.

**When used to set properties:** This method sets the specified CSS property for ALL matched elements.

# Syntax

Return the CSS property value:

```
$(selector).css(property)
```

Set the CSS property and value:

```
$(selector).css(property,value)
```

Set CSS property and value using a function:

```
$(selector).css(property,function(index,currentvalue))
```

Set multiple properties and values:

```
$(selector).css({property:value, property:value, ...})
```

Parameter	Description
<i>property</i>	Specifies the CSS property name, like "color", "font-weight", etc.
<i>value</i>	Specifies the value of the CSS property, like "red", "bold", etc.
<i>function(index,currentvalue)</i>	Specifies a function that returns the new value for the CSS property <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentvalue</i> - Returns the current value of the CSS property</li></ul>

## Try it Yourself - Examples

Return CSS property value Return the specified CSS property value of the FIRST matched element.

Set CSS property and value using a function Using a function to change a CSS property for the selected elements.

Set multiple CSS properties and value pairs How to set multiple CSS properties and values for the selected elements.

# jQuery detach() Method

## Example

Remove all <p> elements:

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

## Definition and Usage

The detach() method removes the selected elements, including all text and child nodes. However, it keeps data and events.

This method also keeps a copy of the removed elements, which allows them to be reinserted at a later time.

**Tip:** To remove the elements and its data and events, use the [remove\(\)](#) method instead.

**Tip:** To remove only the content from the selected elements, use the [empty\(\)](#) method.

## Syntax

```
$(selector).detach()
```

## Try it Yourself - Examples

[Remove and restore an element](#) How to use the detach() method to remove and restore an element.

[Difference between detach\(\) and remove\(\)](#) Show the difference between the detach() and remove() methods.

# jQuery empty() Method

## Example

Remove the content of all <div> elements:

```
$("button").click(function(){
    $("div").empty();
});
```

## Definition and Usage

The empty() method removes all child nodes and content from the selected elements.

**Note:** This method does not remove the element itself, or its attributes.

**Tip:** To remove the elements without removing data and events, use the [detach\(\)](#) method.

**Tip:** To remove the elements and its data and events, use the [remove\(\)](#) method.

## Syntax

```
$(selector).empty()
```

# jQuery hasClass() Method

## Example

Check if any <p> element has a class named "intro":

```
$("button").click(function(){
    alert($(".p").hasClass("intro"));
});
```

## Definition and Usage

The hasClass() method checks if any of the selected elements have a specified class name. If ANY of the selected elements has the specified class name, this method will return "true".

## Syntax

```
$(selector).hasClass(classname)
```

Parameter	Description
<i>classname</i>	Required. Specifies the class name to search for in the selected elements

# jQuery height() Method

## Example

Return the height of a <div> element:

```
$( "button" ).click(function(){
    alert($("div").height());
});
```

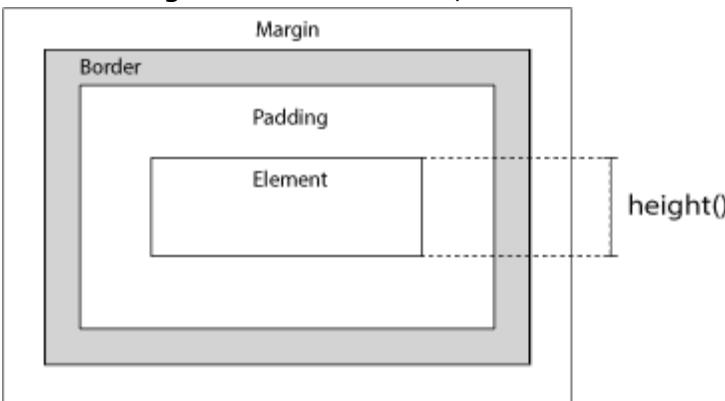
## Definition and Usage

The height() method sets or returns the height of the selected elements.

When this method is used to **return** height, it returns the height of the FIRST matched element.

When this method is used to **set** height, it sets the height of ALL matched elements.

As the image below illustrates, this method does not include padding, border, or margin.



Related methods:

- [width\(\)](#) - Sets or returns the width of an element
- [innerWidth\(\)](#) - Returns the width of an element (includes padding)
- [innerHeight\(\)](#) - Returns the height of an element (includes padding)
- [outerWidth\(\)](#) - Returns the width of an element (includes padding and border)
- [outerHeight\(\)](#) - Returns the height of an element (includes padding and border)

# Syntax

Return the height:

```
$(selector).height()
```

Set the height:

```
$(selector).height(value)
```

Set the height using a function:

```
$(selector).height(function(index, currentheight))
```

Parameter	Description
<code>value</code>	Required for setting height. Specifies the height in px, em, pt, etc. Default unit is px
<code>function(index, currentheight)</code>	Optional. Specifies a function that returns the new height of selected elements <ul style="list-style-type: none"><li>• <code>index</code> - Returns the index position of the element in the set</li><li>• <code>currentheight</code> - Returns the current height of the selected element</li></ul>

## Try it Yourself - Examples

[Set the height of an element](#) How to set the height of an element using different units.

[Increase the height using a function](#) How to use a function to increase the height of an element.

[Return the height of the document and window elements](#) How to return the current height of the document and window elements.

[Display dimensions with related methods](#) How to use width(), height(), innerHeight(), innerWidth(), outerWidth() and outerHeight().

# jQuery html() Method

## Example

Change the content of all <p> elements:

```
$("button").click(function(){
    $("p").html("Hello <b>world</b>!");
});
```

## Definition and Usage

The html() method sets or returns the content (innerHTML) of the selected elements.

When this method is used to **return** content, it returns the content of the FIRST matched element.

When this method is used to **set** content, it overwrites the content of ALL matched elements.

**Tip:** To set or return only the text content of the selected elements, use the text() method.

# Syntax

Return content:

```
$(selector).html()
```

Set content:

```
$(selector).html(content)
```

Set content using a function:

```
$(selector).html(function(index, currentcontent))
```

Parameter	Description
<i>content</i>	Required. Specifies the new content for the selected elements (can contain HTML tags)
<i>function(index, currentcontent)</i>	Optional. Specifies a function that returns the new content for the selected elements <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentcontent</i> - Returns the current HTML content of the selected element</li></ul>

## Try it Yourself - Examples

[Return element content](#) How to return the content of an element.

[Set element content using a function](#) Using a function to set the content of all selected elements.

# jQuery innerHeight() Method

## Example

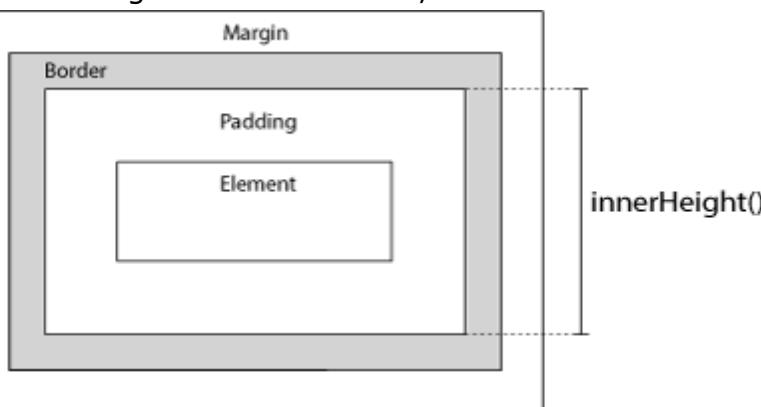
Return the inner height of a <div> element:

```
$( "button" ).click(function(){
    alert($("div").innerHeight());
});
```

## Definition and Usage

The innerHeight() method returns the inner height of the FIRST matched element.

As the image below illustrates, this method includes padding, but not border and margin.



Related methods:

- [width\(\)](#) - Sets or returns the width of an element
- [height\(\)](#) - Sets or returns the height of an element
- [innerWidth\(\)](#) - Returns the width of an element (includes padding)
- [outerWidth\(\)](#) - Returns the width of an element (includes padding and border).
- [outerHeight\(\)](#) - Returns the height of an element (includes padding and border).

## Syntax

```
$(selector).innerHeight()
```

## Try it Yourself - Examples

[Display dimensions with related methods](#) How to use width(), height(), innerHeight(), innerWidth(), outerWidth() and outerHeight().

# jQuery innerWidth() Method

## Example

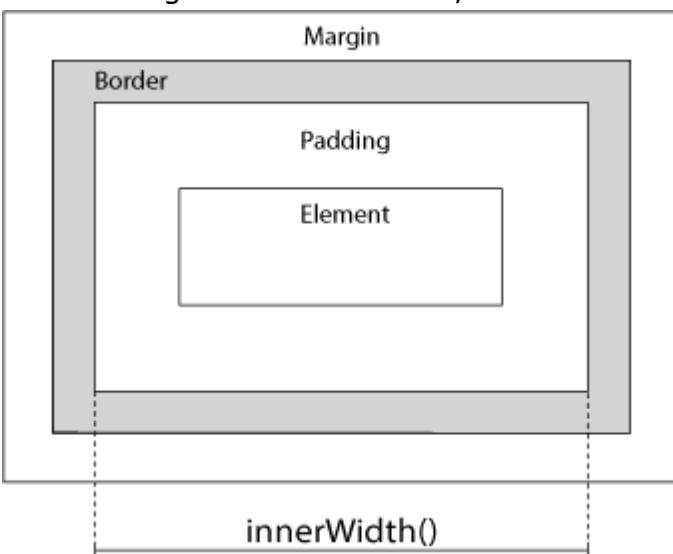
Return the inner width of a <div> element:

```
$( "button" ).click(function(){
    alert($("div").innerWidth());
});
```

## Definition and Usage

The innerWidth() method returns the inner width of the FIRST matched element.

As the image below illustrates, this method includes padding, but not border and margin.



Related methods:

- [width\(\)](#) - Sets or returns the width of an element
- [height\(\)](#) - Sets or returns the height of an element
- [innerHeight\(\)](#) - Returns the height of an element (includes padding)
- [outerWidth\(\)](#) - Returns the width of an element (includes padding and border).
- [outerHeight\(\)](#) - Returns the height of an element (includes padding and border).

## Syntax

```
$(selector).innerWidth()
```

## Try it Yourself - Examples

# jQuery insertAfter() Method

## Example

Insert a `<span>` element after each `<p>` element:

```
$("button").click(function(){
    $("<span>Hello world!</span>").insertAfter("p");
});
```

## Definition and Usage

The `insertAfter()` method inserts HTML elements after the selected elements.

**Tip:** To insert HTML elements before the selected elements, use the [insertBefore\(\)](#) method.

## Syntax

```
$(content).insertAfter(selector)
```

Parameter	Description
<code>content</code>	Required. Specifies the content to insert (must contain HTML tags). <b>Note:</b> If <code>content</code> is an existing element, it will be moved from its current position, and inserted after the selected elements.
<code>selector</code>	Required. Specifies where to insert the content

## Try it Yourself - Examples

[Insert an existing element](#) How to use the `insertAfter()` method to insert an existing element after each selected element.

# jQuery insertBefore() Method

## Example

Insert a `<span>` element before each `<p>` element:

```
$( "button" ).click(function(){
  $("<span>Hello world!</span>").insertBefore( "p" );
});
```

## Definition and Usage

The `insertBefore()` method inserts HTML elements before the selected elements.

**Tip:** To insert HTML elements after the selected elements, use the [insertAfter\(\)](#) method.

## Syntax

```
$(content).insertBefore(selector)
```

Parameter	Description
<code>content</code>	Required. Specifies the content to insert (must contain HTML tags). <b>Note:</b> If <code>content</code> is an existing element, it will be moved from its current position, and inserted before the selected elements.
<code>selector</code>	Required. Specifies where to insert the content

## Try it Yourself - Examples

[Insert an existing element](#) How to use the `insertBefore()` method to insert an existing element before each selected element.

# jQuery offset() Method

## Example

Return the offset coordinates of a <p> element:

```
$("button").click(function(){
    var x = $("p").offset();
    alert("Top: " + x.top + " Left: " + x.left);
});
```

## Definition and Usage

The offset() method set or returns the offset coordinates for the selected elements, relative to the document.

**When used to return the offset:** This method returns the offset coordinates of the FIRST matched element. It returns an object with 2 properties; the top and left positions in pixels.

**When used to set the offset:** This method sets the offset coordinates of ALL matched elements.

# Syntax

Return the offset coordinates:

```
$(selector).offset()
```

Set the offset coordinates:

```
$(selector).offset({top:value, left:value})
```

Set offset coordinates using a function:

```
$(selector).offset(function(index, currentOffset))
```

Parameter	Description
{top:value, left:value}	Required when setting the offset. Specifies the top and left coordinates in pixels. Possible values: <ul style="list-style-type: none"><li>• Name/Value pairs, like {top:100, left:100}</li><li>• An object with top and left properties (<a href="#">example</a>)</li></ul>
function(index, currentOffset)	Optional. Specifies a function that returns an object containing the top and left coordinates <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentOffset</i> - Returns the current coordinates of the selected element</li></ul>

## Try it Yourself - Examples

[Set the offset coordinates](#) How to set the offset coordinates of an element.

[Set offset coordinates using a function](#) Using a function to set the offset coordinates of an element.

[Set the offset coordinates for an element using an object](#) How to set the offset coordinates for an element using a new object.

[Set the offset coordinates for an element using the offset coordinates of another element](#) How to set the offset coords for an element using the offset coords of an existing element.

# jQuery offsetParent() Method

## Example

Set the background color of the closest positioned parent element of the <p> element:

```
$("button").click(function(){
    $("p").offsetParent().css("background-color", "red");
});
```

## Definition and Usage

The offsetParent() method returns the first positioned parent element.

**Tip:** An element can be positioned with jQuery, or with the CSS position property (relative, absolute, or fixed).

## Syntax

```
$(selector).offsetParent()
```

# jQuery outerHeight() Method

## Example

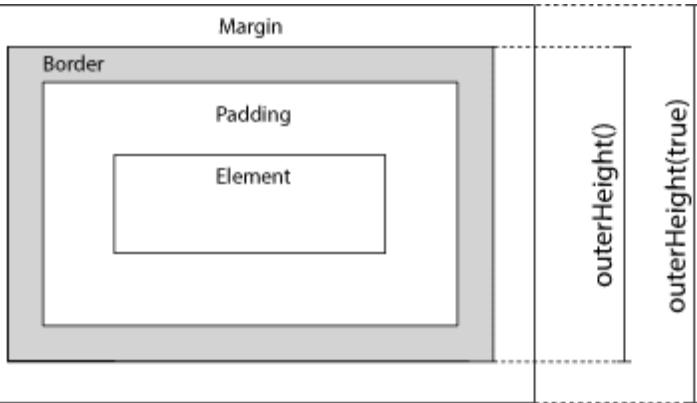
Return the outer height of a <div> element:

```
$( "button" ).click(function(){
    alert($("div").outerHeight());
});
```

## Definition and Usage

The outerHeight() method returns the outer height of the FIRST matched element. As the image below illustrates, this method includes padding and border.

**Tip:** To include the margin, use outerHeight(true).



Related methods:

- [width\(\)](#) - Sets or returns the width of an element
- [height\(\)](#) - Sets or returns the height of an element
- [innerWidth\(\)](#) - Returns the width of an element (includes padding)
- [innerHeight\(\)](#) - Returns the height of an element (includes padding)
- [outerWidth\(\)](#) - Returns the width of an element (includes padding and border).

# Syntax

```
$(selector).outerHeight(includeMargin)
```

Parameter	Description
<i>includeMargin</i>	Optional. A Boolean value specifying whether or not to include the margin <ul style="list-style-type: none"><li>• <i>false</i> - Default. Does not include the margin</li><li>• <i>true</i> - Includes the margin</li></ul>

## Try it Yourself - Examples

[Include the margin](#) Specify whether or not to include the margin.

[Display dimensions with related methods](#) How to use width(), height(), innerHeight(), innerWidth(), outerWidth() and outerHeight().

# jQuery outerWidth() Method

## Example

Return the outer width of a <div> element:

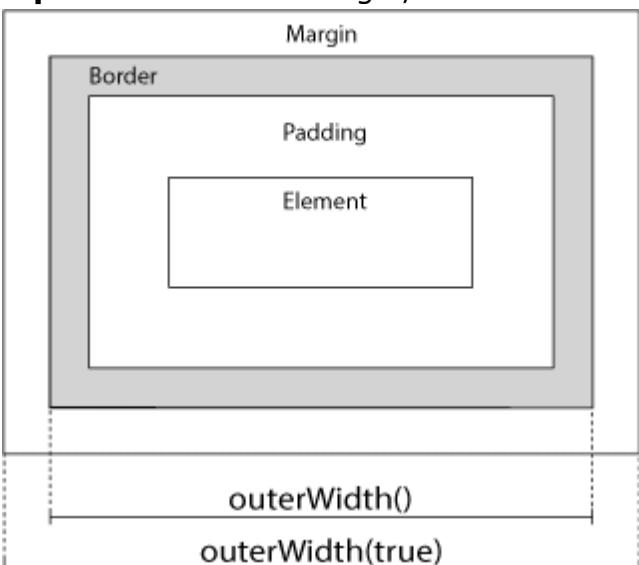
```
$( "button" ).click(function(){
    alert($("div").outerWidth());
});
```

## Definition and Usage

The outerWidth() method returns the outer width of the FIRST matched element.

As the image below illustrates, this method includes padding and border.

**Tip:** To include the margin, use outerWidth(true).



Related methods:

- [width\(\)](#) - Sets or returns the width of an element
- [height\(\)](#) - Sets or returns the height of an element
- [innerWidth\(\)](#) - Returns the width of an element (includes padding)
- [innerHeight\(\)](#) - Returns the height of an element (includes padding)
- [outerHeight\(\)](#) - Returns the height of an element (includes padding and border).

# Syntax

```
$(selector).outerWidth(includeMargin)
```

Parameter	Description
<i>includeMargin</i>	Optional. A Boolean value specifying whether or not to include the margin <ul style="list-style-type: none"><li>• <i>false</i> - Default. Does not include the margin</li><li>• <i>true</i> - Includes the margin</li></ul>

## Try it Yourself - Examples

[Include the margin](#) Specify whether or not to include the margin.

[Display dimensions with related methods](#) How to use width(), height(), innerHeight(), innerWidth(), outerWidth() and outerHeight().

# jQuery position() Method

## Example

Return the top and left position of a <p> element:

```
$( "button" ).click( function(){
    var x = $( "p" ).position();
    alert("Top: " + x.top + " Left: " + x.left);
});
```

## Definition and Usage

The position() method returns the position (relative to its parent element) of the first matched element.

This method returns an object with 2 properties; the top and left positions in pixels.

## Syntax

```
$(selector).position()
```

## Try it Yourself - Examples

Return the position of an element relative to its parent element How to get the position of a <p> element inside a <div> element.

# jQuery prepend() Method

## Example

Insert content at the beginning of all <p> elements:

```
$( "button" ).click( function(){
    $( "p" ).prepend( "<b>Prepended text</b>" );
});
```

## Definition and Usage

The prepend() method inserts specified content at the beginning of the selected elements.

**Tip:** To insert content at the end of the selected elements, use the [append\(\)](#) method.

## Syntax

```
$(selector).prepend(content,function(index,html))
```

Parameter	Description
<i>content</i>	Required. Specifies the content to insert (can contain HTML tags) Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<i>function(index,html)</i>	Optional. Specifies a function that returns the content to insert <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>html</i> - Returns the current HTML of the selected element</li></ul>

## Try it Yourself - Examples

[prepend\(\) - Create content with HTML, jQuery and DOM](#) Insert content with the prepend() method.

[Prepend content using a function](#) Using a function to insert content at the beginning of the selected elements.

# jQuery prependTo() Method

## Example

Insert a `<span>` element at the beginning of each `<p>` element:

```
$( "button" ).click( function(){
    $( "<span>Hello World!</span>" ).prependTo( "p" );
});
```

## Definition and Usage

The `prependTo()` method inserts HTML elements at the beginning of the selected elements.

**Tip:** To insert HTML elements at the end of the selected elements, use the `appendTo()` method.

## Syntax

```
$(content).prependTo(selector)
```

Parameter	Description
<code>content</code>	Required. Specifies the content to insert (must contain HTML tags). <b>Note:</b> If <code>content</code> is an existing element, it will be moved from its current position, and inserted at the beginning of the selected elements.
<code>selector</code>	Required. Specifies on which elements to prepend the content to

## Try it Yourself - Examples

[Insert an existing element](#) How to use the `prependTo()` method to insert an existing element at the beginning of each selected element.

# jQuery prop() Method

## Example

Add and remove a property named "color":

```
$( "button" ).click( function(){
    var $x = $( "div" );
    $x.prop( "color" , "FF0000" );
    $x.append( "The color property: " + $x.prop( "color" ) );
    $x.removeProp( "color" );
});
```

## Definition and Usage

The prop() method sets or returns properties and values of the selected elements.

When this method is used to **return** the property value, it returns the value of the FIRST matched element.

When this method is used to **set** property values, it sets one or more property/value pairs for the set of matched elements.

**Note:** The prop() method should be used to retrieve property values, e.g. DOM properties (like tagName, nodeName, defaultChecked) or your own custom made properties.

**Tip:** To retrieve HTML attributes, use the attr() method instead.

**Tip:** To remove a property use the removeProp() method.

# Syntax

Return the value of a property:

```
$(selector).prop(property)
```

Set the property and value:

```
$(selector).prop(property,value)
```

Set property and value using a function:

```
$(selector).prop(property,function(index,currentvalue))
```

Set multiple properties and values:

```
$(selector).prop({property:value, property:value,...})
```

Parameter	Description
<i>property</i>	Specifies the name of the property
<i>value</i>	Specifies the value of the property
<i>function(index,currentvalue)</i>	Specifies a function that returns the property value to set <ul style="list-style-type: none"><li>• <i>index</i> - Receives the index position of the element in the set</li><li>• <i>currentvalue</i> - Receives the current property value of selected elements</li></ul>

## Try it Yourself - Examples

Difference between prop() and attr() prop() and attr() might return different values. This example shows the differences when used to return the "checked" status of a checkbox.

# jQuery remove() Method

## Example

Remove all <p> elements:

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

## Definition and Usage

The remove() method removes the selected elements, including all text and child nodes. This method also removes data and events of the selected elements.

**Tip:** To remove the elements without removing data and events, use the [detach\(\)](#) method instead.

**Tip:** To remove only the content from the selected elements, use the [empty\(\)](#) method.

## Syntax

```
$(selector).remove(selector)
```

Parameter	Description
<i>selector</i>	Optional. Specifies one or more elements to be removed. To remove multiple elements, separate them with a comma (,)

## Try it Yourself - Examples

[Difference between detach\(\) and remove\(\)](#) Show the difference between the detach() and remove() methods.

[Remove all <p> elements with class="test"](#) Using the optional parameter to filter the elements to be removed.

[Remove all <p> elements with class="test" and "demo"](#) Using the optional parameter to filter multiple elements to be removed.

# jQuery removeAttr() Method

## Example

Remove the style attribute from all <p> elements:

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

## Definition and Usage

The removeAttr() method removes one or more attributes from the selected elements.

## Syntax

```
$(selector).removeAttr(attribute)
```

Parameter	Description
<i>attribute</i>	Required. Specifies one or more attributes to remove. To remove several attributes, separate the attribute names with a space

## Try it Yourself - Examples

[Remove several attributes from the selected elements](#) How to remove the id and class attribute from the selected elements.

# jQuery removeClass() Method

## Example

Remove the class name "intro" from all <p> elements:

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

## Definition and Usage

The removeClass() method removes one or more class names from the selected elements.

**Note:** If no parameter is specified, this method will remove ALL class names from the selected elements.

## Syntax

```
$(selector).removeClass(classname,function(index,currentclass))
```

Parameter	Description
<i>classname</i>	Optional. Specifies one or more class names to remove. To remove several classes, separate the class names with space <b>Note:</b> If this parameter is empty, all class names will be removed
<i>function(index,currentclass)</i>	Optional. A function that returns one or more class names to remove <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentclass</i> - Returns the current class name of selected elements</li></ul>

## Try it Yourself - Examples

[Change the class name of an element](#) How to use addClass() and removeClass() to remove one class name, and add a new class name.

[Remove class using a function](#) Using a function to remove a class from the selected elements.

# jQuery removeProp() Method

## Example

Add and remove a property named "color":

```
$(“button”).click(function(){
    var $x = $(“div”);
    $x.prop(“color”, “FF0000”);
    $x.append(“The color property: ” + $x.prop(“color”));
    $x.removeProp(“color”);
});
```

## Definition and Usage

The removeProp() method removes a property set by the prop() method.

**Note:** Do not use this method to remove HTML attributes like style, id, or checked. Use the removeAttr() method instead.

## Syntax

```
$(selector).removeProp(property)
```

Parameter	Description
<i>property</i>	Specifies the name of the property to remove

# jQuery replaceAll() Method

## Example

Replace all `<p>` elements with `<h2>` elements:

```
$( "button" ).click( function(){
    $( "<h2>Hello world!</h2>" ).replaceAll( "p" );
});
```

## Definition and Usage

The `replaceAll()` method replaces selected elements with new HTML elements.

## Syntax

```
$(content).replaceAll(selector)
```

Parameter	Description
<code>content</code>	Required. Specifies the content to insert (must contain HTML tags)
<code>selector</code>	Required. Specifies which elements to be replaced

## Try it Yourself - Examples

Replace every `<p>` element that is the last child How replace every `<p>` element that is the last child of its parent, with a `<span>` element.

# jQuery replaceWith() Method

## Example

Replace the first <p> element with new text:

```
$( "button" ).click( function(){
    $( "p:first" ).replaceWith( "Hello world!" );
});
```

## Definition and Usage

The replaceWith() method replaces selected elements with new content.

## Syntax

```
$(selector).replaceWith(content, function(index))
```

Parameter	Description
<i>content</i>	Required. Specifies the content to insert (can contain HTML tags) Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<i>function(index)</i>	Optional. Specifies a function that returns the content to replace <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li></ul>

## Try it Yourself - Examples

[Replace content using a function](#) How to use a function to replace selected elements with new content.

# jQuery scrollLeft() Method

## Example

Return the horizontal scrollbar position for a <div> element:

```
$(“button”).click(function(){
    alert($(“div”).scrollLeft());
});
```

## Definition and Usage

The scrollLeft() method sets or returns the horizontal scrollbar position for the selected elements.

**Tip:** When the scrollbar is on the far left side, the position is 0.

**When used to return the position:** This method returns the horizontal position of the scrollbar for the FIRST matched element.

**When used to set the position:** This method sets the horizontal position of the scrollbar for ALL matched elements.

## Syntax

Return horizontal scrollbar position:

```
$(selector).scrollLeft()
```

Set horizontal scrollbar position:

```
$(selector).scrollLeft(position)
```

Parameter	Description
<i>position</i>	Specifies the horizontal scrollbar position in pixels

## Try it Yourself - Examples

[Set the horizontal scrollbar position](#) How to set the horizontal scrollbar position for an element.

# jQuery scrollTop() Method

## Example

Return the vertical scrollbar position for a <div> element:

```
$( "button" ).click(function(){
    alert($("div").scrollTop());
});
```

## Definition and Usage

The scrollTop() method sets or returns the vertical scrollbar position for the selected elements.

**Tip:** When the scrollbar is on the top, the position is 0.

**When used to return the position:** This method returns the vertical position of the scrollbar for the FIRST matched element.

**When used to set the position:** This method sets the vertical position of the scrollbar for ALL matched elements.

## Syntax

Return vertical scrollbar position:

```
$(selector).scrollTop()
```

Set vertical scrollbar position:

```
$(selector).scrollTop(position)
```

Parameter	Description
<i>position</i>	Specifies the vertical scrollbar position in pixels

## Try it Yourself - Examples

[Set the vertical scrollbar position](#) How to set the vertical scrollbar position for an element.

[Toggle between classes on different scroll positions](#) How to toggle between classes on different scroll positions.

[Add smooth scrolling to page anchors](#) Using animate() together with scrollTop() to add smooth scrolling to links.

# jQuery text() Method

## Example

Set text content for all <p> elements:

```
$( "button" ).click( function(){
    $( "p" ).text( "Hello world!" );
});
```

## Definition and Usage

The text() method sets or returns the text content of the selected elements.

When this method is used to **return** content, it returns the text content of all matched elements (HTML markup will be removed).

When this method is used to **set** content, it overwrites the content of ALL matched elements.

**Tip:** To set or return the innerHTML (text + HTML markup) of the selected elements, use the [html\(\)](#) method.

# Syntax

Return text content:

```
$(selector).text()
```

Set text content:

```
$(selector).text(content)
```

Set text content using a function:

```
$(selector).text(function(index, currentcontent))
```

Parameter	Description
<i>content</i>	Required. Specifies the new text content for the selected elements <b>Note:</b> Special characters will be encoded
<i>function(index, currentcontent)</i>	Optional. Specifies a function that returns the new text content for the selected elements <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentcontent</i> - Returns current content of selected elements</li></ul>

## Try it Yourself - Examples

[Return text content](#) How to return text content of the selected elements.

[Set text content using a function](#) Using a function to set the text content of the selected elements.

# jQuery toggleClass() Method

## Example

Toggle between adding and removing the "main" class name for all <p> elements:

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

## Definition and Usage

The toggleClass() method toggles between adding and removing one or more class names from the selected elements.

This method checks each element for the specified class names. The class names are added if missing, and removed if already set - This creates a toggle effect.

However, by using the "switch" parameter, you can specify to only remove, or only add a class name.

## Syntax

```
$(selector).toggleClass(classname,function(index,currentclass),switch)
```

Parameter	Description
classname	Required. Specifies one or more class names to add or remove. To specify several classes, separate the class names with a space
function(index,currentclass)	Optional. Specifies a function that returns one or more class names to add/remove <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentclass</i> - Returns current class name of selected elements</li></ul>
switch	Optional. A Boolean value specifying if the class should only be added (true), or only be removed (false)

# jQuery unwrap() Method

## Example

Remove the parent element of all <p> elements:

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

## Definition and Usage

The unwrap() method removes the parent element of the selected elements.

## Syntax

```
$(selector).unwrap()
```

## Try it Yourself - Examples

[Wrap and unwrap an element](#) How to toggle between wrapping and unwrapping an element.

# jQuery val() Method

## Example

Set the value of the <input> field:

```
$( "button" ).click( function(){
    $( "input:text" ).val( "Glenn Quagmire" );
});
```

## Definition and Usage

The val() method returns or sets the value attribute of the selected elements.

**When used to return value:** This method returns the value of the value attribute of the FIRST matched element.

**When used to set value:** This method sets the value of the value attribute for ALL matched elements.

**Note:** The val() method is mostly used with HTML form elements.

# Syntax

Return the value attribute:

```
$(selector).val()
```

Set the value attribute:

```
$(selector).val(value)
```

Set the value attribute using a function:

```
$(selector).val(function(index, currentValue))
```

Parameter	Description
<i>value</i>	Required. Specifies the value of the value attribute
<i>function(index, currentValue)</i>	Optional. Specifies a function that returns the value to set. <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentValue</i> - Returns the current value attribute of selected elements</li></ul>

## Try it Yourself - Examples

[Return the value attribute](#) How to return the value of the value attribute of the FIRST matched element.

[Set the value attribute using a function](#) Using a function to set the value of the value attribute.

# jQuery width() Method

## Example

Return the width of a <div> element:

```
$( "button" ).click( function(){
    alert( $( "div" ).width() );
});
```

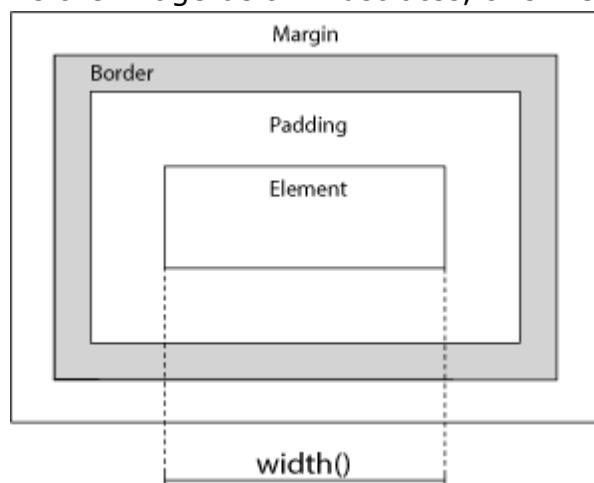
## Definition and Usage

The width() method sets or returns the width of the selected elements.

When this method is used to **return** width, it returns the width of the FIRST matched element.

When this method is used to **set** width, it sets the width of ALL matched elements.

As the image below illustrates, this method does not include padding, border, or margin.



Related methods:

- [height\(\)](#) - Sets or returns the height of an element
- [innerWidth\(\)](#) - Returns the width of an element (includes padding)
- [innerHeight\(\)](#) - Returns the height of an element (includes padding)
- [outerWidth\(\)](#) - Returns the width of an element (includes padding and border).
- [outerHeight\(\)](#) - Returns the height of an element (includes padding and border).

# Syntax

Return the width:

```
$(selector).width()
```

Set the width:

```
$(selector).width(value)
```

Set the width using a function:

```
$(selector).width(function(index, currentwidth))
```

Parameter	Description
<i>value</i>	Required for setting width. Specifies the width in px, em, pt, etc. Default unit is px
<i>function(index, currentwidth)</i>	Optional. Specifies a function that returns the new width of selected elements <ul style="list-style-type: none"><li>• <i>index</i> - Returns the index position of the element in the set</li><li>• <i>currentwidth</i> - Returns the current width of the selected element</li></ul>

## Try it Yourself - Examples

[Set the width of an element](#) How to set the width of an element using different units.

[Decrease the width using a function](#) How to use a function to decrease the width of an element.

[Return the width of the document and window elements](#) How to return the current width of the document and window elements.

[Display dimensions with related methods](#) How to use width(), height(), innerHeight(), innerWidth(), outerWidth() and outerHeight().

[Responsive Design with width\(\)](#) How to change the background color of the page on smaller screens.

# jQuery wrap() Method

## Example

Wrap a `<div>` element around each `<p>` element:

```
$(“button”).click(function(){
    $("p").wrap(“<div></div>”);
});
```

## Definition and Usage

The `wrap()` method wraps specified HTML element(s) around each selected element.

## Syntax

```
$(selector).wrap(wrappingElement,function(index))
```

Parameter	Description
<code>wrappingElement</code>	Required. Specifies what HTML element(s) to wrap around each selected element  Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<code>function(index)</code>	Optional. Specifies a function that returns the wrapping element <ul style="list-style-type: none"><li>• <code>index</code> - Returns the index position of the element in the set</li></ul>

## Try it Yourself - Examples

[Create a new element with the DOM](#) How to use `document.createElement()` to create an element, and wrap it around each selected element.

[Wrap elements using a function](#) Using a function to specify what to wrap around each selected element.

[Wrap and unwrap an element](#) How to toggle between wrapping and unwrapping an element.

# jQuery wrapAll() Method

## Example

Wrap a <div> element around all <p> elements:

```
$( "button" ).click( function(){
    $( "p" ).wrapAll( "<div></div>" );
});
```

## Definition and Usage

The wrapAll() method wraps specified HTML element(s) around all selected elements.

## Syntax

```
$(selector).wrapAll(wrappingElement)
```

Parameter	Description
<i>wrappingElement</i>	Required. Specifies what HTML element(s) to wrap around the selected elements Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>

## Try it Yourself - Examples

[Create a new element with the DOM](#) How to use document.createElement() to create an element, and wrap it around all selected elements.

# jQuery wrapInner() Method

## Example

Wrap a **<b>** element around the content of each **<p>** element:

```
$( "button" ).click( function(){
    $( "p" ).wrapInner( "<b></b>" );
});
```

## Definition and Usage

The `wrapInner()` method wraps specified HTML element(s) around the content (`innerHTML`) of each selected element.

## Syntax

```
$(selector).wrapInner(wrappingElement, function(index))
```

Parameter	Description
<code>wrappingElement</code>	Required. Specifies what HTML element(s) to wrap around the content of each selected element  Possible values: <ul style="list-style-type: none"><li>• HTML elements</li><li>• jQuery objects</li><li>• DOM elements</li></ul>
<code>function(index)</code>	Optional. Specifies a function that returns the wrapping element <ul style="list-style-type: none"><li>• <code>index</code> - Returns the index position of the element in the set</li></ul>

## Try it Yourself - Examples

[Create a new element with the DOM](#) How to use `document.createElement()` to create an element, and wrap it around the content of each selected element.

[Wrap content using a function](#) Using a function to specify what to wrap around the content of each selected element.

# jQuery Traversing Methods

## jQuery Traversing Methods

Method	Description
<u>add()</u>	Adds elements to the set of matched elements
<u>addBack()</u>	Adds the previous set of elements to the current set
<u>andSelf()</u>	<b>Deprecated in version 1.8.</b> An alias for addBack()
<u>children()</u>	Returns all direct children of the selected element
<u>closest()</u>	Returns the first ancestor of the selected element
<u>contents()</u>	Returns all direct children of the selected element (including text and comment nodes)
<u>each()</u>	Executes a function for each matched element
<u>end()</u>	Ends the most recent filtering operation in the current chain, and return the set of matched elements to its previous state
<u>eq()</u>	Returns an element with a specific index number of the selected elements
<u>filter()</u>	Reduce the set of matched elements to those that match the selector or pass the function's test
<u>find()</u>	Returns descendant elements of the selected element
<u>first()</u>	Returns the first element of the selected elements
<u>has()</u>	Returns all elements that have one or more elements inside of them
<u>is()</u>	Checks the set of matched elements against a selector/element/jQuery object, and return true if at least one of these elements matches the given arguments
<u>last()</u>	Returns the last element of the selected elements
<u>map()</u>	Passes each element in the matched set through a function, producing a new jQuery object containing the return values
<u>next()</u>	Returns the next sibling element of the selected element
<u>nextAll()</u>	Returns all next sibling elements of the selected element

<b>Method</b>	<b>Description</b>
<u>nextUntil()</u>	Returns all next sibling elements between two given arguments
<u>not()</u>	Returns elements that do not match a certain criteria
<u>offsetParent()</u>	Returns the first positioned parent element
<u>parent()</u>	Returns the direct parent element of the selected element
<u>parents()</u>	Returns all ancestor elements of the selected element
<u>parentsUntil()</u>	Returns all ancestor elements between two given arguments
<u>prev()</u>	Returns the previous sibling element of the selected element
<u>prevAll()</u>	Returns all previous sibling elements of the selected element
<u>prevUntil()</u>	Returns all previous sibling elements between two given arguments
<u>siblings()</u>	Returns all sibling elements of the selected element
<u>slice()</u>	Reduces the set of matched elements to a subset specified by a range of indices

# jQuery add() Method

## Example

Add `<p>` and `<span>` elements to an existing group of elements (`<h1>`):

```
$( "h1" ).add( "p" ).add( "span" )
```

## Definition and Usage

The `add()` method adds elements to an existing group of elements.

This method adds elements on the whole document, or just inside context elements if the `context` parameter is specified.

## Syntax

```
$(selector).add(element, context)
```

Parameter	Description
<code>element</code>	Required. Specifies a selector expression, a jQuery object, one or more elements or an HTML snippet to be added to an existing group of elements
<code>context</code>	Optional. Specifies the point in the document at which the selector expression should begin matching

## Try it Yourself - Examples

[DOM](#) Add a `<span>` element to an existing group of `<p>` elements, by using a reference to a DOM element.

[HTML snippet](#) Using an HTML snippet to add a `<span>` element to an existing `<p>` element.

# jQuery children() Method

## Example

Return elements that are direct children of <ul>:

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

Result:

```
body (great-grandparent)
div (grandparent)

    ul (parent)
        • li (child) span (grandchild)
```

## Definition and Usage

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

**The DOM tree:** This method only traverse a single level down the DOM tree. To traverse down multiple levels (to return grandchildren or other descendants), use the [find\(\)](#) method.

**Tip:** To traverse a single level up the DOM tree, or all the way up to the document's root element (to return parents or other ancestors), use the [parent\(\)](#) or [parents\(\)](#) method.

**Note:** This method does not return text nodes. To return all children including text nodes, use the [contents\(\)](#) method.

## Syntax

```
$(selector).children(filter)
```

Parameter	Description
<i>filter</i>	Optional. Specifies a selector expression to narrow down the search for children

# jQuery closest() Method

## Example

Return the first ancestor of <span>, that is an <ul> element:

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

Result:

```
body (great-great-grandparent)
div (great-grandparent)

ul (second ancestor - second grandparent)
    ul (first ancestor - first grandparent)
        li (direct parent) span
```

## Definition and Usage

The closest() method returns the first ancestor of the selected element.

An ancestor is a parent, grandparent, great-grandparent, and so on.

**The DOM tree:** This method traverse upwards from the current element, all the way up to the document's root element (<html>), to find the first ancestor of DOM elements.

This method is similar to [parents\(\)](#), in that they both traverse up the DOM tree. The differences are as follows:

### **closest()**

- Begins with the current element
- Travels up the DOM tree and returns the first (single) ancestor that matches the passed expression
- The returned jQuery object contains zero or one element

### **parents()**

- Begins with the parent element
- Travels up the DOM tree and returns all ancestors that matches the passed expression
- The returned jQuery object contains zero or more than one element

## Other related methods:

- [parent\(\)](#) - returns the direct parent element of the selected element
- [parentsUntil\(\)](#) - returns all ancestor elements between two given arguments

# Syntax

Return the first ancestor of the selected element:

```
$(selector).closest(filter)
```

Return the first ancestor using a DOM context to look up the DOM tree within:

```
$(selector).closest(filter, context)
```

Parameter	Description
<i>filter</i>	Required. Specifies a selector expression, element or jQuery object to narrow down the ancestor search
<i>context</i>	Optional. A DOM element within which a matching element may be found

## Try it Yourself - Examples

[Return the first ancestor of <span>, that is a <span> element](#) Because this method begins with the current element, a search for the first <span> of <span>, will return <span>.

[Pass in a DOM element as the context within which to search for the first ancestor element](#)

Using both parameters to pass in a DOM element as the context within which to search for the first <ul> element.

# jQuery contents() Method

## Example

Find all the text nodes inside a <div> element and wrap them with a <b> element:

```
$( "div" ).contents().filter( "text" ).wrap( "<b/>" );
```

## Definition and Usage

The contents() method returns all direct children, including text and comment nodes, of the selected element.

A text node is the actual text displayed by an element.

This method is similar to the [children\(\)](#) method, except that it returns text and comment nodes as well. The contents() method can also access the HTML of an iframe, if it is in the same domain.

## Syntax

```
$(selector).contents()
```

# jQuery each() Method

## Example

Alert the text of each <li> element:

```
$( "button" ).click( function(){
    $( "li" ).each( function(){
        alert( $( this ).text() )
    });
});
```

## Definition and Usage

The each() method specifies a function to run for each matched element.

**Tip:** return false can be used to stop the loop early.

## Syntax

```
$(selector).each(function(index,element))
```

Parameter	Description
function( <i>index,element</i> )	Required. A function to run for each matched element. <ul style="list-style-type: none"><li>• <i>index</i> - The index position of the selector</li><li>• <i>element</i> - The current element (the "this" selector can also be used)</li></ul>

# jQuery eq() Method

## Example

Select the second <p> element (index number 1):

```
$( "p" ).eq(1).css("background-color", "yellow");
```

## Definition and Usage

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 (not 1).

## Syntax

```
$(selector).eq(index)
```

Parameter	Description
<i>index</i>	Required. Specifies the index of the element. Can either be a positive or negative number. <b>Note:</b> Using a negative number will start the index count from the end of the selected elements, instead of the beginning.

## Try it Yourself - Examples

[Using a negative number](#) Using a negative number to return the second <p> element from the end of the selected elements.

# jQuery filter() Method

## Example

Return all `<p>` elements with class name "intro":

```
$(“p”).filter(“.intro”)
```

## Definition and Usage

The `filter()` method returns elements that match a certain criteria.

This 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.

This method is often used to narrow down the search for an element in a group of selected elements.

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

## Syntax

```
$(selector).filter(criteria,function(index))
```

Parameter	Description
<code>criteria</code>	Optional. Specifies a selector expression, a jQuery object or one or more elements to be returned from a group of selected elements. <b>Tip:</b> To specify multiple criteria, use comma.
<code>function(index)</code>	Optional. Specifies a function to run for each element in the set. If it returns true, the element is kept. Otherwise, the element is removed. <ul style="list-style-type: none"><li>• <code>index</code> - The index position of the element in the set</li></ul> <p><b>Note:</b> <code>this</code> is the current DOM element.</p>

## Try it Yourself - Examples

[Return all `<p>` elements that are even](#) Using the `:even` selector together with `filter()` to return all `<p>` elements that are even.

[Multiple criteria](#) How to return all `<p>` elements with class "intro" and id "outro".

[Using a jQuery object](#) How to return all `<p>` elements with class "intro" inside of a `<div>` element, with a jQuery object.

# jQuery find() Method

## Example

Return all <span> elements that are descendants of <ul>:

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

Result:

```
body (great-grandparent)
class="descendants">
div (grandparent)

    ul (parent) • li (child) span (grandchild)
```

## Definition and Usage

The `find()` method returns descendant elements of the selected element.

A descendant is a child, grandchild, great-grandchild, and so on.

**The DOM tree:** This method traverse downwards along descendants of DOM elements, all the way down to the last descendant. To only traverse a single level down the DOM tree (to return direct children), use the `children()` method.

**Note:** The `filter` parameter is required for the `find()` method, unlike the rest of the tree traversal methods.

**Tip:** To return all of the descendant elements, use the "\*" selector.

# Syntax

```
$(selector).find(filter)
```

Parameter	Description
<i>filter</i>	Required. A selector expression, element or jQuery object to filter the search for descendants <b>Note:</b> To return multiple descendants, separate each expression with a comma.

## Try it Yourself - Examples

[Return all descendant elements of <html>](#) Using the "\*" selector to return all elements that are descendants of <html>.

[Return all <span> elements that are descendants of <ul>](#) How to return all <span> elements that are descendants of an <ul> element.

[Only select descendants with a given class name](#) How to return descendant elements with class name "first".

[Return multiple descendants](#) How to return multiple descendant elements.

[Filter the descendant search with a jQuery collection of all <ul> elements](#) How to return all <span> elements that are descendants of an <ul> element with a jQuery object.

[Show the descendants of an element by tag names](#) A demonstration which shows who the descendants of a <div> element actually are.

# jQuery first() Method

## Example

Select the first `<p>` element inside the first `<div>` element:

```
$( "div p" ).first()
```

## Definition and Usage

The `first()` method returns the first element of the selected elements.

**Tip:** To return the last element, use the `last()` method.

## Syntax

```
$(selector).first()
```

## Try it Yourself - Examples

Select the first `<span>` element in a set of `<p>` elements How to select the first `<span>` element in a set of `<p>` elements.

# jQuery has() Method

## Example

Return all `<p>` elements that have a `<span>` element inside of them:

```
$( "p" ).has( "span" )
```

## Definition and Usage

The `has()` method returns all elements that have one or more elements inside of them, that matches the specified selector.

**Tip:** To select elements that have multiple elements inside of them, use comma (see example below).

## Syntax

```
$(selector).has(element)
```

Parameter	Description
<i>element</i>	Required. Specifies a selector expression or an element to match elements against

## Try it Yourself - Examples

[Return an element with multiple elements inside](#) How to return an element that has multiple elements inside of it.

[Return all <p>, <h3> and <div> elements that have a <span> element inside of them](#) How to return all `<p>`, `<h3>` and `<div>` elements that have a `<span>` element inside of them.

[Return elements containing links](#) How to return a `<span>` element that has a hyperlink.

# jQuery is() Method

## Example

If the parent of <p> is a <div> element, alert some text:

```
if ($("#p").parent().is("div")) {  
    alert("Parent of p is div");  
}
```

## Definition and Usage

The is() method checks if one of the selected elements matches the *selectorElement*.

## Syntax

```
$(selector).is(selectorElement, function(index,element))
```

Parameter	Description
<i>selectorElement</i>	Required. Specifies a selector expression, element or a jQuery object to match the current set of elements against. Returns true if there is at least one match from the given argument, and false if not.
<i>function(index,element)</i>	Optional. Specifies a function to run for the group of selected elements. <ul style="list-style-type: none"><li>• index - the index position of the element</li><li>• element - the current element (the "this" selector can also be used)</li></ul>

# jQuery last() Method

## Example

Select the last <p> element inside the last <div> element:

```
$( "div p" ).last()
```

## Definition and Usage

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

**Tip:** To return the first element, use the [first\(\)](#) method.

## Syntax

```
$(selector).last()
```

## Try it Yourself - Examples

[Select the last <span> element in a set of <p> elements](#) How to select the last <span> element in a set of <p> elements.

# jQuery next() Method

## Example

Return the next sibling element of each <li> element with class name "start":

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

Result:

- ul (parent)
- li (sibling)
- li (sibling)
- li (sibling with class name "start")
- li (sibling) li (sibling)
- li (sibling)

## Definition and Usage

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

Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse forward along the next sibling of DOM elements.

Related methods:

- [nextAll\(\)](#) - returns all next sibling elements of the selected element
- [nextUntil\(\)](#) - returns all next sibling elements between two given arguments

## Syntax

```
$(selector).next(filter)
```

Parameter	Description
<i>filter</i>	Optional. Specifies a selector expression to narrow down the next sibling search

# jQuery nextAll() Method

## Example

Return all next sibling elements of each <li> element with class name "start":

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

Result:

- ul (parent)
  - li (sibling)
  - li (sibling)
  - li (sibling with class name "start")
  - li (sibling)
  - li (sibling)
  - li (sibling)

## Definition and Usage

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

Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse forward along siblings of DOM elements.

Related methods:

- [next\(\)](#) - returns the next sibling element of the selected element
- [nextUntil\(\)](#) - returns all next sibling elements between two given arguments

# Syntax

```
$(selector).nextAll(filter)
```

Parameter	Description
<i>filter</i>	Optional. Specifies a selector expression to narrow down the search for next siblings <b>Note:</b> To return multiple siblings, separate each expression with a comma.

## Try it Yourself - Examples

[Narrow down the search](#) How to filter the search for next sibling elements.

[Return multiple siblings](#) Using the filter parameter to get all siblings of a `<h2>` element that have class names "first", "second" and "third".

[Select all next sibling elements of a <p> element](#) How to select all next sibling elements of a `<p>` element.

[Select all next sibling <p> elements of <div>](#) How to select all next sibling `<p>` elements of each `<div>` element.

# jQuery nextUntil() Method

## Example

Return all sibling elements between two <li> elements with class name "start" and "stop":

```
$(document).ready(function(){
    $("li.start").nextUntil("li.stop").css({"color": "red", "border": "2px
solid red"});
});
```

Result:

- ul (parent)
- li (sibling)
- li (sibling)
- li (sibling with class name "start")
- li (sibling)
- li (sibling)
- li (sibling)
- li (sibling with class name "stop")

## Definition and Usage

The nextUntil() method returns all next sibling elements between the *selector* and *stop*.  
Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse forward along siblings of DOM elements.

**Note:** If both parameters are empty, this method will return all next sibling elements (same as the [nextAll\(\)](#) method).

Related methods:

- [next\(\)](#) - returns the next sibling element of the selected element
- [nextAll\(\)](#) - returns all next sibling elements of the selected element

# Syntax

```
$(selector).nextUntil(stop,filter)
```

Parameter	Description
<i>stop</i>	Optional. A selector expression, element or jQuery object indicating where to stop the search for next matching siblings elements
<i>filter</i>	Optional. Specifies a selector expression to narrow down the search for sibling elements between the <i>selector</i> and <i>stop</i> <b>Note:</b> To return multiple siblings, separate each expression with a comma.

## Try it Yourself - Examples

[Narrow down the search](#) Using both parameters to filter the search for next sibling elements between two arguments.

[Return multiple siblings](#) How to return multiple sibling elements between two arguments.

[DOM](#) Using a DOM element instead of a selector to return all sibling elements between two given arguments.

[Using the DOM with both parameters](#) Using a DOM element instead of a selector and both parameters to filter the next sibling search between two arguments.

# jQuery not() Method

## Example

Return all <p> elements that do **not** have the class name "intro":

```
$( "p" ).not( ".intro" )
```

## Definition and Usage

The not() method returns elements that do not match a certain criteria.

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

This method is often used to remove one or more elements from a group of selected elements.

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

## Syntax

```
$(selector).not(criteria,function(index))
```

Parameter	Description
<i>criteria</i>	Optional. Specifies a selector expression, a jQuery object or one or more elements to be removed from a group of selected elements. <b>Tip:</b> To specify multiple criteria, use comma.
<i>function(index)</i>	Optional. Specifies a function to run for each element in a group. If it returns true, the element is removed. Otherwise, the element is kept. <ul style="list-style-type: none"><li>• <i>index</i> - The index position of the element in the set</li></ul> <p><b>Note:</b> <i>this</i> is the current DOM element.</p>

## Try it Yourself - Examples

[Return all <p> elements that are not even](#) Using the :even selector together with not() to return all <p> elements that are not even.

[Multiple criteria](#) How to return all <p> elements that do not have the class "intro" and id "outro".

# jQuery offsetParent() Method

## Example

Set the background color of the closest positioned parent element of the <p> element:

```
$( "button" ).click( function(){
    $( "p" ).offsetParent().css( "background-color", "red" );
});
```

## Definition and Usage

The offsetParent() method returns the first positioned parent element.

**Tip:** An element can be positioned with jQuery, or with the CSS position property (relative, absolute, or fixed).

## Syntax

```
$(selector).offsetParent()
```

# jQuery parent() Method

## Example

Return the direct parent element of <span>:

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

Result:

```
body (great-great-grandparent)
div (great-grandparent)

    ul (grandparent)
    • li (direct parent) span
```

## Definition and Usage

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

**The DOM tree:** This method only traverse a single level up the DOM tree. To traverse all the way up to the document's root element (to return grandparents or other ancestors), use the parents() or the parentsUntil() method.

**Tip:** To traverse a single level down the DOM tree, or all the way down to the last descendant (to return children or other descendants), use the children() or find() method.

## Syntax

```
$(selector).parent(filter)
```

Parameter	Description
<i>filter</i>	Optional. Specifies a selector expression to narrow down the parent search

# jQuery parents() Method

## Example

Return all ancestor elements of <span>:

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

Result:

body (great-great-grandparent)

div (great-grandparent)

ul (grandparent)

• li (direct parent) span

## Definition and Usage

The parents() method returns all ancestor elements of the selected element.

An ancestor is a parent, grandparent, great-grandparent, and so on.

**The DOM tree:** This method traverse upwards from the parent element along ancestors of DOM elements, all the way up to the document's root element (<html>).

**Note:** If the filter parameter is empty, this function will select all ancestors of a set of elements, from the direct parent and all the way up to <body> and <html>. It is therefore often useful to pass a selector expression to narrow down the search result.

This method is similar to [closest\(\)](#), in that they both traverse up the DOM tree. The differences are as follows:

### **parents()**

- Begins with the parent element
- Travels up the DOM tree and returns all ancestors that matches the passed expression
- The returned jQuery object contains zero or more than one element

### **closest()**

- Begins with the current element
- Travels up the DOM tree and returns the first ancestor that matches the passed expression
- The returned jQuery object contains zero or one element

## Other related methods:

- `parent()` - returns the direct parent element of the selected element
- `parentsUntil()` - returns all ancestor elements between two given arguments

# Syntax

```
$(selector).parents(filter)
```

Parameter	Description
<code>filter</code>	Optional. Specifies a selector expression to narrow down the search for ancestors <b>Note:</b> To return multiple ancestors, separate each expression with a comma.

## Try it Yourself - Examples

[Narrow down the search](#) How to use the filter parameter to return all ancestors of `<span>` that are `<ul>` elements.

[Return multiple ancestors](#) How to use the filter parameter to return all ancestors of `<span>` that are `<li>` and `<div>` elements.

[Show the ancestors of an element by tag names](#) A demonstration which shows who the ancestors of a `<span>` element actually are.

# jQuery parentsUntil() Method

## Example

Return all ancestor elements between <span> and <div>:

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

Result:

```
body (great-great-grandparent)
div (great-grandparent)

ul (grandparent)
• li (direct parent) span
```

## Definition and Usage

The parentsUntil() method returns all ancestor elements between the *selector* and *stop*.

An ancestor is a parent, grandparent, great-grandparent, and so on.

**The DOM tree:** This method traverse upwards from the parent element along ancestors of DOM elements, all the way up to the document's root element, until it reaches a specific element.

**Note:** If both parameters are empty, this method will return all ancestor elements (same as the parents() method).

Related methods:

- parent() - returns the direct parent element of the selected element
- parents() - returns all ancestor elements of the selected element
- closest() - returns the first ancestor of the selected element

# Syntax

```
$(selector).parentsUntil(stop,filter)
```

Parameter	Description
<i>stop</i>	Optional. A selector expression, element or jQuery object indicating where to stop the search for matching ancestor elements
<i>filter</i>	Optional. Specifies a selector expression to narrow down the search for ancestors between <i>selector</i> and <i>stop</i> <b>Note:</b> To return multiple ancestors, separate each expression with a comma.

## Try it Yourself - Examples

[Narrow down the search](#) How to use both parameters to filter the search for a specific element between <span> and <div>.

[Return multiple ancestors](#) How to return multiple ancestors between <span> and <body>.

[DOM](#) Return all ancestors between <span> and <div> using a DOM element.

[Using a DOM element and a selector expression to filter the search](#) Using a DOM element to narrow down the ancestor search for <ul> elements between <span> and <div>.

# jQuery prev() Method

## Example

Return the previous sibling element of each <li> element with class name "start":

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

Result:

- ul (parent)
- li (sibling)
- li (sibling) li (sibling)
- li (sibling with class name "start")
- li (sibling)
- li (sibling)

## Definition and Usage

The prev() method returns the previous sibling element of the selected element.

Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse backwards along the previous sibling of DOM elements.

Related methods:

- [prevAll\(\)](#) - returns all previous sibling elements of the selected element
- [prevUntil\(\)](#) - returns all previous sibling elements between two given arguments

## Syntax

```
$(selector).prev(filter)
```

Parameter	Description
<i>filter</i>	Optional. Specifies a selector expression to narrow down the previous sibling search

# jQuery prevAll() Method

## Example

Return all previous sibling elements of each <li> element with class name "start":

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

Result:

- ul (parent)
  - li (sibling)
  - li (sibling)
  - li (sibling)
  - li (sibling with class name "start")
  - li (sibling)
  - li (sibling)

## Definition and Usage

The prevAll() method returns all previous sibling elements of the selected element. Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse backwards along siblings of DOM elements.

Related methods:

- [prev\(\)](#) - returns the next sibling element of the selected element
- [prevUntil\(\)](#) - returns all next sibling elements between two given arguments

# Syntax

```
$(selector).prevAll(filter)
```

Parameter	Description
<i>filter</i>	Optional. Specifies a selector expression to narrow down the search for previous siblings <b>Note:</b> To return multiple siblings, separate each expression with a comma.

## Try it Yourself - Examples

[Narrow down the search](#) How to filter the search for previous sibling elements.

[Return multiple siblings](#) How to use the filter parameter to return all siblings of a <h2> element that have class names "first", "second" and "third".

[Select all previous sibling elements of a <p> element](#) How to select all previous sibling elements of a <p> element.

[Select all previous sibling <p> elements of <div>](#) How to select all previous sibling <p> elements of each <div> element.

# jQuery prevUntil() Method

## Example

Return all sibling elements between two <li> elements with class name "start" and "stop":

```
$(document).ready(function(){
    $("li.start").prevUntil("li.stop").css({"color": "red", "border": "2px
solid red"});
});
```

Result:

- ul (parent)
- li (sibling with class name "stop")
- li (sibling)
- li (sibling)
- li (sibling)
- li (sibling with class name "start")
- li (sibling)
- li (sibling)

## Definition and Usage

The `prevUntil()` method returns all previous sibling elements between the *selector* and *stop*. Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse backwards along siblings of DOM elements.

**Note:** If both parameters are empty, this method will return all previous sibling elements (same as the [prevAll\(\)](#) method).

Related methods:

- [prev\(\)](#) - returns the previous sibling element of the selected element
- [prevAll\(\)](#) - returns all previous sibling elements of the selected element

# Syntax

```
$(selector).prevUntil(stop,filter)
```

Parameter	Description
<i>stop</i>	Optional. A selector expression, element or jQuery object indicating where to stop the search for previous matching siblings elements
<i>filter</i>	Optional. Specifies a selector expression to narrow down the search for sibling elements between the <i>selector</i> and <i>stop</i> <b>Note:</b> To return multiple siblings, separate each expression with a comma.

## Try it Yourself - Examples

[Narrow down the search](#) Using both parameters to filter the search for previous sibling elements between two arguments.

[Return multiple siblings](#) How to return multiple sibling elements between two arguments.

[DOM](#) Using a DOM element instead of a selector to return sibling elements between two arguments.

[Using the DOM with both parameters](#) Using a DOM element instead of a selector and both parameters to filter the previous sibling search between two arguments.

# jQuery siblings() Method

## Example

Return all sibling elements of each <li> element with class name "start":

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

Result:

- ul (parent)
  - li (sibling)
  - li (sibling)
  - li (sibling with class name "start")
  - li (sibling)
  - li (sibling)

## Definition and Usage

The `siblings()` method returns all sibling elements of the selected element.

Sibling elements are elements that share the same parent.

**The DOM tree:** This method traverse forward and backwards along siblings of DOM elements.

**Tip:** Use the `prev()` or `next()` method to narrow down the search for only previous or next sibling elements.

## Syntax

```
$(selector).siblings(filter)
```

Parameter	Description
<code>filter</code>	Optional. Specifies a selector expression to narrow down the search for sibling elements

# jQuery slice() Method

## Example

Start the selection of <p> elements from the <p> element with index number 2:

```
$("p").slice(2)
```

## Definition and Usage

The slice() method selects a subset of elements based on its index.

A subset is a set that is a part of a larger set.

This method is used to limit the selection of elements in a group, by a start and end point: the *start* parameter is a starting index (starts at 0) from which to create the subset, and the *stop* parameter is an optional ending point.

## Syntax

```
$(selector).slice(start,stop)
```

Parameter	Description
<i>start</i>	Required. Specifies where to start the selection of elements. The index numbers start at 0. <b>Note:</b> Using a negative number will select elements from the end of the selected elements, instead of the beginning.
<i>stop</i>	Optional. Specifies where to end the selection of elements. If omitted, the selection continues until the end of the set. The index numbers start at 0. <b>Note:</b> Using a negative number will select elements from the end of the selected elements, instead of the beginning.

## Try it Yourself - Examples

Start and stop How to use both parameters to select <p> elements from a start point and an end point.

Using a negative number Using a negative number to start the selection of <p> elements counting from the end, instead of the beginning.

Using both parameters with a negative number Using start and stop with a negative number to start the selection of <p> elements counting from the end.

# jQuery AJAX Methods

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

The following table lists all the jQuery AJAX methods:

Method	Description
<code>\$.ajax()</code>	Performs an async AJAX request
<code>\$.ajaxPrefilter()</code>	Handle custom Ajax options or modify existing options before each request is sent and before they are processed by <code>\$.ajax()</code>
<code>\$.ajaxSetup()</code>	Sets the default values for future AJAX requests
<code>\$.ajaxTransport()</code>	Creates an object that handles the actual transmission of Ajax data
<code>\$.get()</code>	Loads data from a server using an AJAX HTTP GET request
<code>\$.getJSON()</code>	Loads JSON-encoded data from a server using a HTTP GET request
<code>\$.parseJSON()</code>	Deprecated in version 3.0, use <a href="#">JSON.parse()</a> instead. Takes a well-formed JSON string and returns the resulting JavaScript value
<code>\$.getScript()</code>	Loads (and executes) a JavaScript from a server using an AJAX HTTP GET request
<code>\$.param()</code>	Creates a serialized representation of an array or object (can be used as URL query string for AJAX requests)
<code>\$.post()</code>	Loads data from a server using an AJAX HTTP POST request
<code>ajaxComplete()</code>	Specifies a function to run when the AJAX request completes
<code>ajaxError()</code>	Specifies a function to run when the AJAX request completes with an error
<code>ajaxSend()</code>	Specifies a function to run before the AJAX request is sent
<code>ajaxStart()</code>	Specifies a function to run when the first AJAX request begins
<code>ajaxStop()</code>	Specifies a function to run when all AJAX requests have completed
<code>ajaxSuccess()</code>	Specifies a function to run when an AJAX request completes successfully
<code>load()</code>	Loads data from a server and puts the returned data into the selected element

<b>Method</b>	<b>Description</b>
<u>serialize()</u>	Encodes a set of form elements as a string for submission
<u>serializeArray()</u>	Encodes a set of form elements as an array of names and values

# jQuery ajax() Method

## Example

Change the text of a <div> element using an AJAX request:

```
$( "button" ).click( function(){
    $.ajax({url: "demo_test.txt", success: function(result){
        $( "#div1" ).html(result);
    }});
});
```

## Definition and Usage

The ajax() method is used to perform an AJAX (asynchronous HTTP) request.

All jQuery AJAX methods use the ajax() method. This method is mostly used for requests where the other methods cannot be used.

## Syntax

```
$.ajax({name:value, name:value, ... })
```

The parameters specifies one or more name/value pairs for the AJAX request.

Possible names/values in the table below:

Name	Value/Description
async	A Boolean value indicating whether the request should be handled asynchronous or not. Default is true
beforeSend(xhr)	A function to run before the request is sent
cache	A Boolean value indicating whether the browser should cache the requested pages. Default is true
complete(xhr,status)	A function to run when the request is finished (after success and error functions)
contentType	The content type used when sending data to the server. Default is: "application/x-www-form-urlencoded"
context	Specifies the "this" value for all AJAX related callback functions

Name	Value/Description
data	Specifies data to be sent to the server
dataFilter( <i>data,type</i> )	A function used to handle the raw response data of the XMLHttpRequest
dataType	The data type expected of the server response.
error( <i>xhr,status,error</i> )	A function to run if the request fails.
global	A Boolean value specifying whether or not to trigger global AJAX event handles for the request. Default is true
ifModified	A Boolean value specifying whether a request is only successful if the response has changed since the last request. Default is: false.
jsonp	A string overriding the callback function in a jsonp request
jsonpCallback	Specifies a name for the callback function in a jsonp request
password	Specifies a password to be used in an HTTP access authentication request.
processData	A Boolean value specifying whether or not data sent with the request should be transformed into a query string. Default is true
scriptCharset	Specifies the charset for the request
success( <i>result,status,xhr</i> )	A function to be run when the request succeeds
timeout	The local timeout (in milliseconds) for the request
traditional	A Boolean value specifying whether or not to use the traditional style of param serialization
type	Specifies the type of request. (GET or POST)
url	Specifies the URL to send the request to. Default is the current page
username	Specifies a username to be used in an HTTP access authentication request
xhr	A function used for creating the XMLHttpRequest object

## Try it Yourself - Examples

# jQuery ajaxSetup() Method

## Example

Set the default URL and success function for all AJAX requests:

```
$( "button" ).click( function(){
    $.ajaxSetup({url: "demo_ajax_load.txt", success: function(result){
        $( "div" ).html(result);}});
    $.ajax();
});
```

## Definition and Usage

The ajaxSetup() method sets default values for future AJAX requests.

## Syntax

```
$.ajaxSetup({name:value, name:value, ... })
```

The parameters specifies the settings for AJAX requests with one or more name/value pairs. Possible names/values in the table below:

Name	Value/Description
async	A Boolean value indicating whether the request should be handled asynchronous or not. Default is true
beforeSend(xhr)	A function to run before the request is sent
cache	A Boolean value indicating whether the browser should cache the requested pages. Default is true
complete(xhr,status)	A function to run when the request is finished (after success and error functions)
contentType	The content type used when sending data to the server. Default is: "application/x-www-form-urlencoded"
context	Specifies the "this" value for all AJAX related callback functions
data	Specifies data to be sent to the server

Name	Value/Description
dataFilter( <i>data,type</i> )	A function used to handle the raw response data of the XMLHttpRequest
dataType	The data type expected of the server response.
error( <i>xhr,status,error</i> )	A function to run if the request fails.
global	A Boolean value specifying whether or not to trigger global AJAX event handles for the request. Default is true
ifModified	A Boolean value specifying whether a request is only successful if the response has changed since the last request. Default is: false.
jsonp	A string overriding the callback function in a jsonp request
jsonpCallback	Specifies a name for the callback function in a jsonp request
password	Specifies a password to be used in an HTTP access authentication request.
processData	A Boolean value specifying whether or not data sent with the request should be transformed into a query string. Default is true
scriptCharset	Specifies the charset for the request
success( <i>result,status,xhr</i> )	A function to be run when the request succeeds
timeout	The local timeout (in milliseconds) for the request
traditional	A Boolean value specifying whether or not to use the traditional style of param serialization
type	Specifies the type of request. (GET or POST)
url	Specifies the URL to send the request to. Default is the current page
username	Specifies a username to be used in an HTTP access authentication request
xhr	A function used for creating the XMLHttpRequest object

## Try it Yourself - Examples

# jQuery get() Method

## Example

Send an HTTP GET request to a page and get a result back:

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

## Definition and Usage

The `$.get()` method loads data from the server using a HTTP GET request.

## Examples

Request "test.php", but ignore return results:

```
$.get("test.php");
```

Request "test.php" and send some additional data along with the request (ignore return results):

```
$.get("test.php", { name:"Donald", town:"Ducktown" });
```

Request "test.php" and pass arrays of data to the server (ignore return results):

```
$.get("test.php", { 'colors[]' : ["Red","Green","Blue"] });
```

Request "test.php" and alert the result of the request:

```
$.get("test.php", function(data){ alert("Data: " + data); });
```

# Syntax

```
$.get(URL, data, function(data, status, xhr), dataType)
```

Parameter	Description
<i>URL</i>	Required. Specifies the URL you wish to request
<i>data</i>	Optional. Specifies data to send to the server along with the request
<i>function(data,status,xhr)</i>	Optional. Specifies a function to run if the request succeeds Additional parameters: <ul style="list-style-type: none"><li>• <i>data</i> - contains the resulting data from the request</li><li>• <i>status</i> - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror")</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li></ul>
<i>dataType</i>	Optional. Specifies the data type expected of the server response. By default jQuery performs an automatic guess. Possible types: <ul style="list-style-type: none"><li>• "xml" - An XML document</li><li>• "html" - HTML as plain text</li><li>• "text" - A plain text string</li><li>• "script" - Runs the response as JavaScript, and returns it as plain text</li><li>• "json" - Runs the response as JSON, and returns a JavaScript object</li><li>• "jsonp" - Loads in a JSON block using JSONP. Will add an "?callback=?" to the URL to specify the callback</li></ul>

# jQuery getJSON() Method

## Example

Get JSON data using an AJAX request, and output the result:

```
$( "button" ).click( function() {
    $.getJSON( "demo_ajax_json.js", function(result){
        $.each(result, function(i, field){
            $( "div" ).append(field + " ");
        });
    });
});
```

## Definition and Usage

The getJSON() method is used to get JSON data using an AJAX HTTP GET request.

## Syntax

```
$(selector).getJSON(url,data,success(data,status,xhr))
```

Parameter	Description
<i>url</i>	Required. Specifies the url to send the request to
<i>data</i>	Optional. Specifies data to be sent to the server
<i>success(data,status,xhr)</i>	Optional. Specifies the function to run if the request succeeds Additional parameters: <ul style="list-style-type: none"><li>• <i>data</i> - contains the data returned from the server.</li><li>• <i>status</i> - contains a string containing request status ("success", "notmodified", "error", "timeout", or "parsererror").</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li></ul>

# jQuery getScript() Method

## Example

Get and run a JavaScript using an AJAX request:

```
$( "button" ).click( function(){
    $.getScript( "demo_ajax_script.js" );
});
```

## Definition and Usage

The `getScript()` method is used to get and execute a JavaScript using an AJAX HTTP GET request.

## Syntax

```
$(selector).getScript(url,success(response,status))
```

Parameter	Description
<code>url</code>	Required. Specifies the url to send the request to
<code>success(response,status)</code>	Optional. Specifies the function to run if the request succeeds Additional parameters: <ul style="list-style-type: none"><li>• <code>response</code> - contains the result data from the request</li><li>• <code>status</code> - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror")</li></ul>

# jQuery param() Method

## Example

Output the result of a serialized object:

```
$( "button" ).click( function(){
    $( "div" ).text($.param(personObj));
});
```

## Definition and Usage

The param() method creates a serialized representation of an array or an object.

The serialized values can be used in the URL query string when making an AJAX request.

## Syntax

```
$.param(object, trad)
```

Parameter	Description
<i>object</i>	Required. Specifies an array or object to serialize
<i>trad</i>	Optional. A Boolean value specifying whether or not to use the traditional style of param serialization

# jQuery post() Method

## Example 1

Load data from the server using a HTTP POST request:

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

## Example 2

Change the text of a <div> element using an AJAX POST request:

```
$( "input" ).keyup( function(){
    var txt = $( "input" ).val();
    $.post( "demo_ajax_gethint.asp" , {suggest: txt}, function(result){
        $( "span" ).html(result);
    });
});
```

## Definition and Usage

The `$.post()` method loads data from the server using a HTTP POST request.

# Syntax

```
$(selector).post(URL,data,function(data,status,xhr),dataType)
```

Parameter	Description
<i>URL</i>	Required. Specifies the url to send the request to
<i>data</i>	Optional. Specifies data to send to the server along with the request
<i>function(data,status,xhr)</i>	Optional. Specifies a function to run if the request succeeds Additional parameters: <ul style="list-style-type: none"><li>• <i>data</i> - contains the resulting data from the request</li><li>• <i>status</i> - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror")</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li></ul>
<i>dataType</i>	Optional. Specifies the data type expected of the server response. By default jQuery performs an automatic guess. Possible types: <ul style="list-style-type: none"><li>• "xml" - An XML document</li><li>• "html" - HTML as plain text</li><li>• "text" - A plain text string</li><li>• "script" - Runs the response as JavaScript, and returns it as plain text</li><li>• "json" - Runs the response as JSON, and returns a JavaScript object</li><li>• "jsonp" - Loads in a JSON block using JSONP. Will add an "?callback=?" to the URL to specify the callback</li></ul>

# jQuery ajaxComplete() Method

## Example

Show a "loading" indicator image while an AJAX request is going on:

```
$(document).ajaxStart(function(){
    $("#wait").css("display", "block");
});

$(document).ajaxComplete(function(){
    $("#wait").css("display", "none");
});
```

## Definition and Usage

The ajaxComplete() method specifies a function to be run when an AJAX request completes.

**Note:** As of jQuery version 1.8, this method should only be attached to document.

Unlike ajaxSuccess(), functions specified with the ajaxComplete() method will run when the request is completed, even it is not successful.

## Syntax

```
$(document).ajaxComplete(function(event,xhr,options))
```

Parameter	Description
<i>function(event,xhr,options)</i>	Required. Specifies the function to run when the request completes Additional parameters: <ul style="list-style-type: none"><li>• <i>event</i> - contains the event object</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li><li>• <i>options</i> - contains the options used in the AJAX request</li></ul>

# jQuery ajaxError() Method

## Example

Trigger an alert box when an AJAX request fails:

```
$(document).ajaxError(function(){
    alert("An error occurred!");
});
```

## Definition and Usage

The ajaxError() method specifies a function to be run when an AJAX request fails.

**Note:** As of jQuery version 1.8, this method should only be attached to document.

## Syntax

```
$(document).ajaxError(function(event,xhr,options,exc))
```

Parameter	Description
<i>function(event,xhr,options,exc)</i>	Required. Specifies the function to run if the request fails Additional parameters: <ul style="list-style-type: none"><li>• <i>event</i> - contains the event object</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li><li>• <i>options</i> - contains the options used in the AJAX request</li><li>• <i>exc</i> - contains the JavaScript exception, if one occurred</li></ul>

## Try it Yourself - Examples

[Use the xhr and options parameter](#) How to use the options parameter to get a more useful error message.

# jQuery ajaxSend() Method

## Example

Change the content of a <div> element when an AJAX requests is about to be sent:

```
$(document).ajaxSend(function(e, xhr, opt){  
    $("div").append("<p>Requesting: " + opt.url + "</p>");  
});
```

## Definition and Usage

The ajaxSend() method specifies a function to run when an AJAX requests is about to be sent.

**Note:** As of jQuery version 1.8, this method should only be attached to document.

## Syntax

```
$(document).ajaxSend(function(event, xhr, options))
```

# jQuery ajaxStart() Method

## Example

Show a "loading" indicator image when an AJAX request starts:

```
$(document).ajaxStart(function(){
    $(this).html("<img src='demo_wait.gif'>");
});
```

## Definition and Usage

The ajaxStart() method specifies a function to be run when an AJAX request starts.

**Note:** As of jQuery version 1.8, this method should only be attached to document.

## Syntax

```
$(document).ajaxStart(function())
```

Parameter	Description
<i>function()</i>	Required. Specifies the function to run when the AJAX request starts

# jQuery ajaxStop() Method

## Example

Trigger an alert box when all AJAX requests have completed:

```
$(document).ajaxStop(function(){
    alert("All AJAX requests completed");
});
```

## Definition and Usage

The ajaxStop() method specifies a function to run when ALL AJAX requests have completed.

When an AJAX request completes, jQuery checks if there are any more AJAX requests. The function specified with the ajaxStop() method will run if no other requests are pending.

**Note:** As of jQuery version 1.8, this method should only be attached to document.

## Syntax

```
$(document).ajaxStop(function())
```

Parameter	Description
<i>function()</i>	Required. Specifies the function to run when all AJAX requests have completed

# jQuery ajaxSuccess() Method

## Example

Trigger an alert box when an AJAX request completes successfully:

```
$(document).ajaxSuccess(function(){
    alert("AJAX request successfully completed");
});
```

## Definition and Usage

The ajaxSuccess() method specifies a function to be run when an AJAX request is successfully completed.

**Note:** As of jQuery version 1.8, this method should only be attached to document.

## Syntax

```
$(document).ajaxSuccess(function(event,xhr,options))
```

Parameter	Description
<i>function(event,xhr,options)</i>	Required. Specifies the function to run if the request succeeds Additional parameters: <ul style="list-style-type: none"><li>• <i>event</i> - contains the event object</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li><li>• <i>options</i> - contains the options used in the AJAX request</li></ul>

## Try it Yourself - Examples

[Use the options parameter](#) How to use the options parameter to restrict the function to AJAX requests for a specific file.

# jQuery load() Method

## Example

Load the content of the file "demo\_test.txt" into a specific <div> element:

```
$(“button”).click(function(){
    $(“#div1”).load(“demo_test.txt”);
});
```

## Definition and Usage

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

**Note:** There is also a jQuery Event method called load. Which one is called, depends on the parameters.

## Syntax

```
$(selector).load(url,data,function(response,status,xhr))
```

Parameter	Description
<i>url</i>	Required. Specifies the URL you wish to load
<i>data</i>	Optional. Specifies data to send to the server along with the request
<i>function(response,status,xhr)</i>	Optional. Specifies a callback function to run when the load() method is completed. Additional parameters: <ul style="list-style-type: none"><li>• <i>response</i> - contains the result data from the request</li><li>• <i>status</i> - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror")</li><li>• <i>xhr</i> - contains the XMLHttpRequest object</li></ul>

## Try it Yourself - Examples

[Make an AJAX request, and send data with the request](#) How to use the data parameter to send data with the AJAX request (this example uses the an example explained in our [AJAX Tutorial](#))

# jQuery serialize() Method

## Example

Output the result of serialized form values:

```
$( "button" ).click( function(){
    $( "div" ).text( $( "form" ).serialize() );
});
```

## Definition and Usage

The `serialize()` method creates a URL encoded text string by serializing form values. You can select one or more form elements (like input and/or text area), or the form element itself.

The serialized values can be used in the URL query string when making an AJAX request.

## Syntax

```
$(selector).serialize()
```

# jQuery serializeArray() Method

## Example

Output the result of form values serialized as arrays:

```
$( "button" ).click( function(){
    var x = $( "form" ).serializeArray();
    $.each(x, function(i, field){
        $( "#results" ).append(field.name + ":" + field.value + " ");
    });
});
```

## Definition and Usage

The serializeArray() method creates an array of objects (name and value) by serializing form values.

You can select one or more form elements (like input and/or text area), or the form element itself.

## Syntax

```
$(selector).serializeArray()
```

# jQuery Miscellaneous Methods

## jQuery Misc Methods

Method	Description
<u>data()</u>	Attaches data to, or gets data from, selected elements
<u>each()</u>	Execute a function for each matched element
<u>get()</u>	Get the DOM elements matched by the selector
<u>index()</u>	Search for a given element from among the matched elements
<u>\$.noConflict()</u>	Release jQuery's control of the \$ variable
<u>\$.param()</u>	Create a serialized representation of an array or object (can be used as URL query string for AJAX requests)
<u>removeData()</u>	Removes a previously-stored piece of data
<u>size()</u>	Removed in version 3.0. Use the <u>length</u> property instead.
<u>toArray()</u>	Retrieve all the DOM elements contained in the jQuery set, as an array

# jQuery Misc data() Method

## Example

Attach data to a <div> element, then retrieve the data:

```
$("#btn1").click(function(){
    $("div").data("greeting", "Hello World");
});
$("#btn2").click(function(){
    alert($("#div").data("greeting"));
});
```

## Definition and Usage

The data() method attaches data to, or gets data from, selected elements.

**Tip:** To remove data, use the [removeData\(\)](#) method.

## Return Data from an Element

Returns attached data from a selected element.

## Syntax

```
$(selector).data(name)
```

Parameter	Description
<i>name</i>	Optional. Specifies the name of data to retrieve. If no name is specified, this method will return all stored data for the element as an object

## Attach Data to an Element

Attach data to selected elements.

# Syntax

```
$(selector).data(name,value)
```

Parameter	Description
<i>name</i>	Required. Specifies the name of data to set
<i>value</i>	Required. Specifies the value of data to set

## Attach Data to an Element Using an Object

Attach data to selected elements using an object with name/value pairs.

# Syntax

```
$(selector).data(object)
```

Parameter	Description
<i>object</i>	Required. Specifies an object containing name/value pairs

# jQuery Misc each() Method

## Example

Alert the text of each <li> element:

```
$( "button" ).click( function(){
    $( "li" ).each( function(){
        alert( $( this ).text() )
    });
});
```

## Definition and Usage

The each() method specifies a function to run for each matched element.

**Tip:** return false can be used to stop the loop early.

## Syntax

```
$(selector).each(function(index,element))
```

Parameter	Description
function( <i>index,element</i> )	Required. A function to run for each matched element. <ul style="list-style-type: none"><li>• <i>index</i> - The index position of the selector</li><li>• <i>element</i> - The current element (the "this" selector can also be used)</li></ul>

# jQuery Misc get() Method

## Example

Get the name and value of the first <p> element:

```
$( "button" ).click( function(){
    var x = $( "p" ).get(0);
    $( "div" ).text( x.nodeName + ": " + x.innerHTML );
});
```

## Definition and Usage

The get() method gets the DOM elements specified by the selector.

## Syntax

```
$(selector).get(index)
```

Parameter	Description
<i>index</i>	Optional. Specifies which of the matching elements to get (by index number)

# jQuery Misc index() Method

## Example

Get the index of the clicked <li> element, relative to its siblings:

```
$( "li" ).click( function(){
    alert$(this).index());
});
```

## Definition and Usage

The index() method returns the index position of specified elements relative to other specified elements.

The elements can be specified by jQuery selectors, or a DOM element.

**Note:** If the element is not found, index() will return -1.

## Index of First Matched Element, Relative to Sibling elements.

Get the index position of the first matched selected element relative to its sibling elements.

## Syntax

```
$(selector).index()
```

## Index of an Element, Relative to the Selector.

Get the index position of an element, relative to the selector.

The element can be specified using a DOM element, or a jQuery selector.

## Syntax

```
$(selector).index(element)
```

<b>Parameter</b>	<b>Description</b>
<i>element</i>	Optional. Specifies the element to get the index position of. Can be a DOM element or a jQuery selector

# jQuery Misc noConflict() Method

## Example

Use the noConflict() method to specify a new name for the jQuery variable:

```
var jq = $.noConflict();
```

## Definition and Usage

The noConflict() method releases jQuery's control of the \$ variable.

This method can also be used to specify a new custom name for the jQuery variable.

**Tip:** This method is useful when other JavaScript libraries use the \$ for their functions.

## Syntax

```
$.noConflict(removeAll)
```

Parameter	Description
<i>removeAll</i>	Optional. A Boolean value that specifies whether or not to release jQuery's control of ALL jQuery variables (including "jQuery")

# jQuery Misc param() Method

## Example

Output the result of a serialized object:

```
$( "button" ).click( function(){
    $( "div" ).text($.param(personObj));
});
```

## Definition and Usage

The param() method creates a serialized representation of an array or an object.

The serialized values can be used in the URL query string when making an AJAX request.

## Syntax

```
$.param(object, trad)
```

Parameter	Description
<i>object</i>	Required. Specifies an array or object to serialize
<i>trad</i>	Optional. A Boolean value specifying whether or not to use the traditional style of param serialization

# jQuery Misc removeData() Method

## Example

Remove previously attached data from a <div> element:

```
$("#btn2").click(function(){
    $("div").removeData("greeting");
    alert("Greeting is: " + $("div").data("greeting"));
});
```

## Definition and Usage

The removeData() method removes data previously set with the [data\(\)](#) method.

## Syntax

```
$(selector).removeData(name)
```

Parameter	Description
<i>name</i>	Optional. Specifies the name of data to remove. If no name is specified, this method will remove all stored data from the selected elements

# jQuery Misc size() Method

## Example

Alert the number of elements matched by the jQuery selector:

```
$( "button" ).click( function(){
    alert( $( "li" ).size() );
});
```

## Definition and Usage

The `size()` method was deprecated in version 1.8 and removed in jQuery version 3.0. Use the [length](#) property instead.

The `size()` method returns the number of elements matched by the jQuery selector.

## Syntax

```
$(selector).size()
```

# jQuery Misc toArray() Method

## Example

Convert the <li> elements to an array, then alert the innerHTML of the array elements:

```
$("button").click(function(){
    var x = $("li").toArray()
    for (i = 0; i < x.length; i++) {
        alert(x[i].innerHTML);
    }
});
```

## Definition and Usage

The toArray() method returns the elements matched by the jQuery selector as an array.

## Syntax

```
$(selector).toArray()
```

# jQuery Properties

## jQuery Properties

Property	Description
<u>context</u>	Removed in version 3.0. Contains the original context passed to <code>jQuery()</code>
<u>jquery</u>	Contains the jQuery version number
<u>jQuery.fx.interval</u>	Change the animation firing rate in milliseconds
<u>jQuery.fx.off</u>	Globally disable/enable all animations
<u>jQuery.support</u>	A collection of properties representing different browser features or bugs (Intended for jQuery's internal use)
<u>length</u>	Contains the number of elements in the jQuery object

# jQuery context Property

## Example

Determine the context:

```
$( "div" ).append( "<p>" + $( "div" ).context + "</p>" )
.append( "<p>" + $( "div" , document.body ).context.nodeName + "</p>" );
```

## Definition and Usage

The **context** property was deprecated in version 1.8 and removed in jQuery version 3.0.

The context property contains the original context passed to jQuery, which could be a DOM node context, or, if no node is passed, the document context.

## Syntax

```
context
```

# jQuery jquery Property

## Example

Alert the current running jQuery version:

```
$("button").on("click",function(){
  var version = $.jquery;
  alert("You are running jQuery version: " + version);
});
```

## Definition and Usage

The jquery property returns a string containing the jQuery version number.

## Syntax

```
$.jquery
```

# jQuery jQuery.fx.interval Property

## Example

Cause the animation of a <div> element to run with less frames:

```
$("#toggle").on("click", function(){
    $("div").toggle(5000);
});
$("#interval").on("click", function(){
    jQuery.fx.interval = 500;
});
```

## Definition and Usage

The `jQuery.fx.interval` property is used to change the animation firing rate in milliseconds. The default value is 13 milliseconds. This property is often used to modify the number of frames per second at which animations will run. Lowering the firing rate can make animations to run smoother. However, it may cause performance and CPU implications.

**Note:** For any changes to this property to take effect, no animation should be running or all animations should be stopped first.

**Note:** This property has no effect in browsers that support the `requestAnimationFrame` property.

## Syntax

```
jQuery.fx.interval = milliseconds;
```

Parameter	Description
<code>milliseconds</code>	Required. Specifies the animation firing rate in milliseconds. Default is 13 milliseconds

# jQuery jQuery.fx.off Property

## Example

Toggle animation on and off:

```
$("#true").click(function(){
    jQuery.fx.off = true;
});
$("#false").click(function(){
    jQuery.fx.off = false;
});
$("#toggle").click(function(){
    $("div").toggle("slow");
});
```

## Definition and Usage

The jQuery.fx.off property is used to globally disable/enable all animations.

Default value is false, which allows animations to run normally. When set to true, all animation methods will be disabled, which will immediately set elements to their final state, instead of displaying an effect.

**Tip:** To shorten the code, it is possible to use \$.fx.off instead of jQuery.fx.off.

## Syntax

```
jQuery.fx.off = true|false;
```

Parameter	Description
true	Specifies that animations should be disabled
false	Default. Specifies that animations should be enabled

# jQuery jQuery.support Property

## Example

Test whether the browser can create an XMLHttpRequest object:

```
$(document).ready(function(){
    $("p").html("This browser can create XMLHttpRequest object: " +
    jQuery.support.ajax);
});
```

## Definition and Usage

The `jQuery.support` property contains a collection of properties representing different browser features or bugs.

This property was primarily intended for jQuery's internal use.

# Syntax

`jQuery.support.propvalue`

Parameter	Description
<i>propvalue</i>	Required. Specifies the function to test for. The tests included are: <ul style="list-style-type: none"><li>• ajax</li><li>• boxModel</li><li>• changeBubbles</li><li>• checkClone</li><li>• checkOn</li><li>• cors</li><li>• cssFloat</li><li>• hrefNormalized</li><li>• htmlSerialize</li><li>• leadingWhitespace</li><li>• noCloneChecked</li><li>• noCloneEvent</li><li>• opacity</li><li>• optDisabled</li><li>• optSelected</li><li>• scriptEval()</li><li>• style</li><li>• submitBubbles</li><li>• tbody</li></ul>

# jQuery length Property

## Example

Alert the number of <li> elements:

```
$( "button" ).click( function() {
    alert( $( "li" ).length );
});
```

## Definition and Usage

The length property contains the number of elements in the jQuery object.

## Syntax

```
$(selector).length
```