

What is jQuery

jQuery is a fast, lightweight, and feature-rich JavaScript library that is based on the principle "*write less, do more*". Its easy-to-use APIs makes the things like HTML document traversal and manipulation, event handling, adding animation effects to a web page much simpler that works seamlessly across all the major browsers. jQuery also gives you the ability to create an Ajax based application in a quick and simple way.

Companies like Google, Microsoft and IBM are using the jQuery for their applications. So you can easily understand how popular the jQuery is?

Advantages of jQuery

The biggest advantage of jQuery comes from its selectors that allow you to traverse the DOM tree of an HTML document's structure in an efficient manner.

Additionally, using the jQuery inbuilt methods you can create animations and effects like sliding transition, showing or hiding an element, etc. with a single line of code.

Here are some more advantages, why one should opt for jQuery:

- **Save lots of time** — You can save lots of time and efforts by using the jQuery inbuilt effects and selectors and concentrate on other development work.
- **Simplify common JavaScript tasks** — jQuery considerably simplifies the common JavaScript tasks. Now you can easily create feature rich and interactive web pages with fewer lines of codes, a typical example is retrieving the information from a server and updates the page without refreshing.
- **Easy to use** — jQuery is very easy to use. Anybody with the basic working knowledge of HTML, CSS and JavaScript can start development with jQuery.
- **Compatible with browsers** — jQuery is created with modern browsers in mind and it is compatible with all major modern browsers such as Mozilla Firefox, Google Chrome, Safari, Internet Explorer, and Opera.
- **Absolutely Free** — And the best part is, it is completely free to download and use.

Standard jQuery Syntax

A jQuery statement typically starts with the dollar sign (\$) and ends with a semicolon (;). In jQuery, the dollar sign (\$) is just an alias for jQuery. Let's consider the following example code which demonstrates the most basic statement of the jQuery.

Example

[Try this code »](#)

- ```
• <script type="text/javascript">
• $(document).ready(function(){
• // Some code to be executed...
• alert("Hello World!");
• });
```

- `</script>`

The above example simply displays an alert message "Hello World!" to the user.

## Explanation of code

If you are completely new to the jQuery, you might think what that code was all about. OK, let's go through each of the parts of this script one by one.

- The `<script>` element — Since jQuery is just a JavaScript library, so the jQuery code can be placed inside the `<script>` element. However, if you want to place it in an [external JavaScript file](#), which is preferred, you just remove this part.
- The `$(document).ready(handler);` — This statement is typically known as ready event. Where the *handler* is basically a function that is passed to the `ready()` method to be executed safely as soon as the document is ready to be manipulated i.e. when the DOM hierarchy has been fully constructed. The jQuery `ready()` method is typically used with an anonymous function. So, the above example can also be written in a shorthand notation like this:

### Example

[Try this code »](#)

- `<script type="text/javascript">`
- `$(function(){`
- `// Some code to be executed...`
- `alert("Hello World!");`
- `});`
- `</script>`

**Tip:** You can use any syntax you like as both the syntax are equivalent. However, the document ready event is easier to understand when reading the code.

Further, inside an event handler function you can write the jQuery statements to perform any action following the basic syntax, like: `$(selector).action();`

Where, the `$(selector)` basically selects the HTML elements from the DOM tree so that it can be manipulated and the `action()` applies some action on the selected elements such as changes the CSS property value, or sets the element's contents, etc. Let's consider another example that sets the paragraph text after the DOM is ready:

### Example

[Try this code »](#)

- `<!DOCTYPE html>`
- `<html>`
- `<head>`
- `<meta charset="utf-8">`
- `<title>jQuery Document Ready Demo</title>`
- `<link rel="stylesheet" type="text/css" href="css/style.css">`
- `<script src="js/jquery-1.11.3.min.js"></script>`
- `<script type="text/javascript">`

- `$(document).ready(function){`
- `$("p").text("Hello World!");`
- `});`
- `</script>`
- `</head>`
- `<body>`
- `<p>Not loaded yet.</p>`
- `</body>`
- `</html>`

In the jQuery statement of the example above (*line no-10*) the `p` is a jQuery selector which select all the paragraphs i.e. the `<p>` elements in the document, later the `text()` method set the paragraph's text content to "Hello World!" text. The paragraph text in the example above is replaced automatically when the document is ready. But what if we want the user to perform some action before executing the jQuery code to replace the paragraph text. Let's consider one last example:

### Example

[Try this code »](#)

- `<!DOCTYPE html>`
- `<html>`
- `<head>`
- `<meta charset="utf-8">`
- `<title>jQuery Click Handler Demo</title>`
- `<link rel="stylesheet" type="text/css" href="css/style.css">`
- `<script src="js/jquery-1.11.3.min.js"></script>`
- `<script type="text/javascript">`
- `$(document).ready(function){`
- `$("button").click(function(){`
- `$("p").text("Hello World!");`
- `});`
- `});`
- `</script>`
- `</head>`
- `<body>`
- `<p>Not loaded yet.</p>`
- `<button type="button">Replace Text</button>`
- `</body>`
- `</html>`

## jQuery Selectors

*In this tutorial you will learn how to select HTML elements using jQuery.*

### Selecting Elements with jQuery

JavaScript is most commonly used to get or modify the content or value of the HTML elements on the page, as well as to apply some effects like show, hide, animations etc. But, before you can perform any action you need to find or select the target HTML element.

Selecting the elements through a typical JavaScript approach could be very painful, but the jQuery works like a magic here. The ability of making the DOM elements selection simple and easy is one of the most powerful feature of the jQuery.

**Tip:** The jQuery supports almost all the selectors defined in the latest CSS3 specifications, as well as it has its own custom selectors. These custom selectors greatly enhance the capabilities selecting the HTML elements on a page.

In the following sections, you will see some of the common ways of selecting the elements on a page and do something with them using the jQuery.

## Selecting Elements by ID

You can use the ID selector to select a single element with the unique ID on the page. For example, the following jQuery code will select and highlight an element having the ID attribute `id="mark"`, when the document is ready to be manipulated.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Highlight element with id mark
4. $("#mark").css("background", "yellow");
5. });
6. </script>
```

In the example above, the `$(document).ready()` is an event that is used to manipulate a page safely with the jQuery. Code included inside this event will only run once the page DOM is ready. We'll learn more about the events in next chapter.

---

## Selecting Elements by Class Name

The class selector can be used to select the elements with a specific class.

For example, the following jQuery code will select and highlight the elements having the class attribute `class="mark"`, when the document is ready.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Highlight elements with class mark
4. $(".mark").css("background", "yellow");
5. });
6. </script>
```

---

## Selecting Elements by Name

The element selector can be used to select elements based on the element name. For example, the following jQuery code will select and highlight all the paragraph i.e. the `<p>` elements of the document when it is ready.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Highlight paragraph elements
4. $("p").css("background", "yellow");
5. });
6. </script>
```

---

## Selecting Elements by Attribute

You can use the attribute selector to select an element by one of its HTML attributes, such as a link's `target` attribute or an input's `type` attribute, etc.

For example, the following jQuery code will select and highlight all the text inputs i.e. `<input>` elements with the `type="text"`, when the document is ready.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Highlight paragraph elements
4. $('input[type="text"]').css("background", "yellow");
5. });
6. </script>
```

---

## Selecting Elements by Compound CSS Selector

You can also combine the CSS selectors to make your selection even more precise. For instance, you can combine the class selector with an element selector to find the elements in a document that has certain type and class.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Highlight only paragraph elements with class mark
• $("p.mark").css("background", "yellow");
•
• // Highlight only span elements inside the element with ID mark
• $("#mark span").css("background", "yellow");
•
• // Highlight li elements inside the ul elements
• $("ul li").css("background", "red");
•
• // Highlight li elements only inside the ul element with id mark
• $("ul#mark li").css("background", "yellow");
•
• // Highlight li elements inside all the ul element with class mark
• $("ul.mark li").css("background", "green");
•
• // Highlight all anchor elements with target blank
• $('a[target="_blank"]').css("background", "yellow");
• });
• </script>
```

---

## jQuery Custom Selector

In addition to the [CSS defined selectors](#), jQuery provides its own custom selector to further enhancing the capabilities of selecting elements on a page.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Highlight table rows appearing at odd places
• $("tr:odd").css("background", "yellow");
```

- 
- // Highlight table rows appearing at even places
- \$("tr:even").css("background", "orange");
- 
- // Highlight first paragraph element
- \$("p:first").css("background", "red");
- 
- // Highlight last paragraph element
- \$("p:last").css("background", "green");
- 
- // Highlight all input elements with type text inside a form
- \$("form :text").css("background", "purple");
- 
- // Highlight all input elements with type password inside a form
- \$("form :password").css("background", "blue");
- 
- // Highlight all input elements with type submit inside a form
- \$("form :submit").css("background", "violet");
- });
- </script>

## jQuery Events

*In this tutorial you will learn how to handle events with jQuery.*

### What are Events

Events are often triggered by the user's interaction with the web page, such as when a link or button is clicked, text is entered into an input box or textarea, selection is made in a select box, key is pressed on the keyboard, the mouse pointer is moved etc. In some cases, the Browser itself can trigger the events, such as the page load and unload events.

jQuery enhances the basic event-handling mechanisms by offering the events methods for most native browser events, some of these methods are `ready()`, `click()`, `keypress()`, `focus()`, `blur()`, `change()`, etc. For example, to execute some JavaScript code when the DOM is ready, you can use the jQuery `ready()` method, like this:

#### Example

Try this code »

1. `<script type="text/javascript">`
2. `$(document).ready(function(){`
3.     // Code to be executed

```
4. alert("Hello World!");
5. });
6. </script>
```

**Note:** The `$(document).ready()` is an event that is used to manipulate a page safely with the jQuery. Code included inside this event will only run once the page DOM is ready i.e. when the document is ready to be manipulated.

In general, the events can be categorized into four main groups — [mouse events](#), [keyboard events](#), [form events](#) and [document/window events](#). The following section will give you the brief overview of all these events as well as related jQuery methods one by one.

## Mouse Events

A mouse event is fired when the user click some element, move the mouse pointer etc. Here're some commonly used jQuery methods to handle the mouse events.

### The `click()` Method

The jQuery `click()` method attach an event handler function to the selected elements for "click" event. The attached function is executed when the user clicks on that element. The following example will hide the `<p>` elements on a page when they are clicked.

#### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("p").click(function(){
4. $(this).slideUp();
5. });
6. });
7. </script>
```

**Note:** The `this` keyword inside the jQuery event handler function is a reference to the element where the event is currently being delivered.

### The `dblclick()` Method

The jQuery `dblclick()` method attach an event handler function to the selected elements for "dblclick" event. The attached function is executed when the user double-clicks on that element. The following example will hide the `<p>` elements when they are double-clicked.

#### Example

Try this code »



```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("p").dblclick(function(){
4. $(this).slideUp();
5. });
6. });
7. </script>
```

## The `hover()` Method

The jQuery `hover()` method attach one or two event handler functions to the selected elements that is executed when the mouse pointer enters and leaves the elements. The first function is executed when the user place the mouse pointer over an element, whereas the second function is executed when the user removes the mouse pointer from that element.

The following example will highlight `<p>` elements when you place the cursor on it, the highlighting will be removed when you remove the cursor.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("p").hover(function(){
4. $(this).addClass("highlight");
5. }, function(){
6. $(this).removeClass("highlight");
7. });
8. });
9. </script>
```

**Tip:** You can consider the `hover()` method is a combination of the jQuery `mouseenter()` and `mouseleave()` methods.

## The `mouseenter()` Method

The jQuery `mouseenter()` method attach an event handler function to the selected elements that is executed when the mouse enters an element. The following example will add the class `highlight` to the `<p>` element when you place the cursor on it.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("p").mouseenter(function(){
4. $(this).addClass("highlight");
5. });
6. });
```

7. `</script>`

## The `mouseleave()` Method

The jQuery `mouseleave()` method attach an event handler function to the selected elements that is executed when the mouse leaves an element. The following example will remove the class `highlight` from the `<p>` element when you remove the cursor from it.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("p").mouseleave(function(){
4. $(this).removeClass("highlight");
5. });
6. });
7. </script>
```

For more mouse event methods, please check out the [jQuery Events Reference](#) »

---

## Keyboard Events

A keyboard event is fired when the user press or release a key on the keyboard. Here're some commonly used jQuery methods to handle the keyboard events.

## The `keypress()` Method

The jQuery `keypress()` method attach an event handler function to the selected elements (typically form controls) that is executed when the browser receives keyboard input from the user. The following example will display a message when the keypress event is fired and how many times it is fired when you press the key on the keyboard.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. var i = 0;
4. $('input[type="text"]').keypress(function(){
5. $("span").text(i += 1);
6. $("p").show().fadeOut();
7. });
8. });
9. </script>
```

**Note:**The keypress event is similar to the keydown event, except that modifier and non-printing keys such as Shift, Esc, Backspace or Delete, Arrow keys etc. trigger keydown events but not keypress events.

## The `keydown()` Method

The jQuery `keydown()` method attach an event handler function to the selected elements (typically form controls) that is executed when the user first presses a key on the keyboard. The following example will display a message when the keydown event is fired and how many times it is fired when you press the key on the keyboard.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. var i = 0;
4. $('input[type="text"]').keydown(function(){
5. $("span").text(i += 1);
6. $("p").show().fadeOut();
7. });
8. });
9. </script>
```

## The `keyup()` Method

The jQuery `keyup()` method attach an event handler function to the selected elements (typically form controls) that is executed when the user releases a key on the keyboard. The following example will display a message when the keyup event is fired and how many times it is fired when you press and release a key on the keyboard.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. var i = 0;
4. $('input[type="text"]').keyup(function(){
5. $("span").text(i += 1);
6. $("p").show().fadeOut();
7. });
8. });
9. </script>
```

**Tip:**The keyboard events can be attached to any element, but the event is only sent to the element that has the focus. That's why the keyboard events generally attached to the form controls such as text input box or textarea.

---

## Form Events

A form event is fired when a form control receive or loses focus or when the user modify a form control value such as by typing text in a text input, select any option in a select box etc. Here're some commonly used jQuery methods to handle the form events.

### The `change()` Method

The jQuery `change()` method attach an event handler function to the `<input>`, `<textarea>` and `<select>` elements that is executed when its value changes. The following example will display an alert message when you select any option in dropdown select box.

#### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("select").change(function(){
4. var selectedOption = $(this).find(":selected").val();
5. alert("You have selected - " + selectedOption);
6. });
7. });
8. </script>
```

**Note:**For select boxes, checkboxes, and radio buttons, the event is fired immediately when the user makes a selection with the mouse, but for the text input and textarea the event is fired after the element loses focus.

### The `focus()` Method

The jQuery `focus()` method attach an event handler function to the selected elements (typically form controls and links) that is executed when it gains focus. The following example will display a message when the text input receive focus.

#### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("input").focus(function(){
4. $(this).next("span").show().fadeOut("slow");
5. });
6. });
7. </script>
```

## The `blur()` Method

The jQuery `blur()` method attach an event handler function to the form elements such as `<input>`, `<textarea>`, `<select>` that is executed when it loses focus. The following example will display a message when the text input loses focus.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("input").blur(function(){
4. $(this).next("span").show().fadeOut("slow");
5. });
6. });
7. </script>
```

## The `submit()` Method

The jQuery `submit()` method attach an event handler function to the `<form>` elements that is executed when the user is attempting to submit a form. The following example will display a message depending on the value entered when you try to submit the form.

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("form").submit(function(event){
• var mob = /^[1-9]{1}[0-9]{9}$/;
• var enteredValue = $("#inputMobile").val();
• if(mob.test(enteredValue) == false && enteredValue != 10){
• $("p").html("Invalid mobile number").show().fadeOut(1000);
• } else{
• $("p").html("Valid mobile number").show().fadeOut(1000);
• }
• event.preventDefault();
• });
• });
• </script>
```

**Tip:** A form can be submitted either by clicking a submit button, or by pressing Enter when certain form elements have focus.

---

## Document/Window Events

Events are also triggered in a situation when the page DOM (Document Object Model) is ready or when the user resize or scrolls the browser window, etc. Here're some commonly used jQuery methods to handle such kind of events.

## The `ready()` Method

The jQuery `ready()` method specify a function to execute when the DOM is fully loaded. The following example will replace the paragraphs text as soon as the DOM hierarchy has been fully constructed and ready to be manipulated.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("p").text("The DOM is now loaded and can be manipulated.");
4. });
5. </script>
```

## The `resize()` Method

The jQuery `resize()` method attach an event handler function to the window element that is executed when the size of the browser window changes.

The following example will display the current width and height of the browser window when you try to resize it by dragging its corners.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $(window).resize(function() {
4. $(window).bind("resize", function(){
5. $("p").text("Window width: " + $(window).width() + ", " + "Window height: " +
6. $(window).height());
7. });
8. });
9. </script>
```

## The `scroll()` Method

The jQuery `scroll()` method attach an event handler function to the window or scrollable iframes and elements that is executed whenever the element's scroll position changes.

The following example will display a message when you scroll the browser window.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $(window).scroll(function() {
4. $("p").show().fadeOut("slow");
5. });
6. });
7. </script>
```

## jQuery Show and Hide Effects

*In this tutorial you will learn how to show hide HTML elements using jQuery.*

### jQuery `show()` and `hide()` Methods

You can show and hide HTML elements using the jQuery `show()` and `hide()` methods. The `hide()` method simply sets the `inline style display: none` for the selected elements. Conversely, the `show()` method restores the `display properties` of the matched set of elements to whatever they initially were—typically block, inline, or inline-block—before the inline style `display: none` was applied to them. Here's is an example.

#### **Example**

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Hide displayed paragraphs
• $(".hide-btn").click(function(){
• $("p").hide();
• });
•
• // Show hidden paragraphs
• $(".show-btn").click(function(){
• $("p").show();
• });
• });
• </script>
```

You can optionally specify the duration (also referred as speed) parameter for making the jQuery show hide effect animated over a specified period of time.

Durations can be specified either using one of the predefined string `'slow'` or `'fast'`, or in a number of milliseconds, for greater precision; higher values indicate slower animations.

## Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Hide displayed paragraphs with different speeds
• $(".hide-btn").click(function(){
• $(".p.normal").hide();
• $(".p.fast").hide("fast");
• $(".p.slow").hide("slow");
• $(".p.very-fast").hide(50);
• $(".p.very-slow").hide(2000);
• });
•
• // Show hidden paragraphs with different speeds
• $(".show-btn").click(function(){
• $(".p.normal").show();
• $(".p.fast").show("fast");
• $(".p.slow").show("slow");
• $(".p.very-fast").show(50);
• $(".p.very-slow").show(2000);
• });
• });
• </script>
```

**Note:** The speed or duration string 'fast' indicates the durations of 200 milliseconds, while the string 'slow' indicates the durations of 600 milliseconds.

You can also specify a [callback function](#) to be executed after the `show()` or `hide()` method completes. We'll learn more about the callback function in upcoming chapters.

## Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Display alert message after hiding paragraphs
• $(".hide-btn").click(function(){
• $(".p").hide("slow", function(){
• // Code to be executed
• alert("The hide effect is completed.");
• });
• });
•
• // Display alert message after showing paragraphs
• $(".show-btn").click(function(){
• $(".p").show("slow", function(){
• // Code to be executed
```



- alert("The show effect is completed.");
- });
- });
- });
- </script>

## jQuery toggle() Method

The jQuery `toggle()` method show or hide the elements in such a way that if the element is initially displayed, it will be hidden; if hidden, it will be displayed (i.e. toggles the visibility).

### Example

[Try this code »](#)

1. `<script type="text/javascript">`
2. `$(document).ready(function(){`
3.     `// Toggles paragraphs display`
4.     `$(".toggle-btn").click(function(){`
5.         `$( "p" ).toggle();`
6.     `});`
7. `});`
8. `</script>`

Similarly, you can specify the duration parameter for the `toggle()` method to make it animated like the `show()` and `hide()` methods.

### Example

[Try this code »](#)

- `<script type="text/javascript">`
- `$(document).ready(function(){`
- `// Toggles paragraphs with different speeds`
- `$(".toggle-btn").click(function(){`
- `$( "p.normal" ).toggle();`
- `$( "p.fast" ).toggle("fast");`
- `$( "p.slow" ).toggle("slow");`
- `$( "p.very-fast" ).toggle(50);`
- `$( "p.very-slow" ).toggle(2000);`
- `});`
- `});`
- `</script>`

Similarly, you can also specify a [callback function](#) for the `toggle()` method.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Display alert message after toggling paragraphs
• $(".toggle-btn").click(function(){
• $("p").toggle(1000, function(){
• // Code to be executed
• alert("The toggle effect is completed.");
• });
• });
• });
• </script>
```

## jQuery Fading Effects

*In this tutorial you will learn how to fade in and out elements using jQuery.*

### jQuery `fadeIn()` and `fadeOut()` Methods

You can use the jQuery `fadeIn()` and `fadeOut()` methods to display or hide the HTML elements by gradually increasing or decreasing their opacity.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Fading out displayed paragraphs
• $(".out-btn").click(function(){
• $("p").fadeOut();
• });
•
• // Fading in hidden paragraphs
• $(".in-btn").click(function(){
• $("p").fadeIn();
• });
• });
• </script>
```

Like other jQuery effects methods, you can optionally specify the duration or speed parameter for the `fadeIn()` and `fadeOut()` methods to control how long the fading animation will run. Durations can be specified either using one of the predefined

string `'slow'` or `'fast'`, or in a number of milliseconds; higher values indicate slower animations.

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Fading out displayed paragraphs with different speeds
• $(".out-btn").click(function(){
• $(".p.normal").fadeOut();
• $(".p.fast").fadeOut("fast");
• $(".p.slow").fadeOut("slow");
• $(".p.very-fast").fadeOut(50);
• $(".p.very-slow").fadeOut(2000);
• });
•
• // Fading in hidden paragraphs with different speeds
• $(".in-btn").click(function(){
• $(".p.normal").fadeIn();
• $(".p.fast").fadeIn("fast");
• $(".p.slow").fadeIn("slow");
• $(".p.very-fast").fadeIn(50);
• $(".p.very-slow").fadeIn(2000);
• });
• });
• </script>
```

**Note:** The effect of `fadeIn()/fadeOut()` method looks similar to `show()/hide()`, but unlike `show()/hide()` method the `fadeIn()/fadeOut()` method only animates the opacity of the target elements and does not animates their dimensions.

You can also specify a [callback function](#) to be executed after the `fadeIn()` or `fadeOut()` method completes. We'll learn more about the callback function in upcoming chapters.

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Display alert message after fading out paragraphs
• $(".out-btn").click(function(){
• $(".p").fadeOut("slow", function(){
• // Code to be executed
• alert("The fade-out effect is completed.");
• });
• });
• });
```

- 
- `// Display alert message after fading in paragraphs`
- `$(".in-btn").click(function(){`
- `$(".p").fadeIn("slow", function(){`
- `// Code to be executed`
- `alert("The fade-in effect is completed.");`
- `});`
- `});`
- `});`
- `</script>`

## jQuery `fadeToggle()` Method

The jQuery `fadeToggle()` method display or hide the selected elements by animating their opacity in such a way that if the element is initially displayed, it will be fade out; if hidden, it will be fade in (i.e. toggles the fading effect).

### Example

Try this code »

1. `<script type="text/javascript">`
2. `$(document).ready(function(){`
3.  `// Toggles paragraphs display with fading`
4.  `$(".toggle-btn").click(function(){`
5.  `$(".p").fadeToggle();`
6.  `});`
7. `});`
8. `</script>`

Similarly, you can specify the duration parameter for the `fadeToggle()` method like `fadeIn()`/`fadeOut()` method to control the duration or speed of the fade toggle animation.

### Example

Try this code »

- `<script type="text/javascript">`
- `$(document).ready(function(){`
- `// Fade Toggles paragraphs with different speeds`
- `$(".toggle-btn").click(function(){`
- `$(".p.normal").fadeToggle();`
- `$(".p.fast").fadeToggle("fast");`
- `$(".p.slow").fadeToggle("slow");`
- `$(".p.very-fast").fadeToggle(50);`
- `$(".p.very-slow").fadeToggle(2000);`
- `});`

- });
- </script>

Similarly, you can also specify a [callback function](#) for the `fadeToggle()` method.

### Example

Try this code »

- <script type="text/javascript">
- \$(document).ready(function(){
- // Display alert message after fade toggling paragraphs
- \$(".toggle-btn").click(function(){
- \$("p").fadeToggle(1000, function(){
- // Code to be executed
- alert("The fade-toggle effect is completed.");
- });
- });
- });
- </script>

## jQuery `fadeTo()` Method

The jQuery `fadeTo()` method is similar to the `.fadeIn()` method, but unlike `.fadeIn()` the `fadeTo()` method lets you fade in the elements to a certain opacity level.

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

The required opacity parameter specifies the final opacity of the target elements that can be a number between 0 and 1. The duration or speed parameter is also required for this method that specifies the duration of the fade to animation.

### Example

Try this code »

- <script type="text/javascript">
- \$(document).ready(function(){
- // Fade to paragraphs with different opacity
- \$(".to-btn").click(function(){
- \$(".p.none").fadeTo("fast", 0);
- \$(".p.partial").fadeTo("slow", 0.5);
- \$(".p.complete").fadeTo(2000, 1);
- });
- });
- </script>

# Query Sliding Effects

*In this tutorial you will learn how to create slide motion effect using jQuery.*

## jQuery `slideUp()` and `slideDown()` Methods

The jQuery `slideUp()` and `slideDown()` methods are used to hide or show the HTML elements by gradually decreasing or increasing their height (i.e. by sliding them up or down).

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Slide up displayed paragraphs
• $(".up-btn").click(function(){
• $("p").slideUp();
• });
•
• // Slide down hidden paragraphs
• $(".down-btn").click(function(){
• $("p").slideDown();
• });
• });
• </script>
```

Like other jQuery effects methods, you can optionally specify the duration or speed parameter for the `slideUp()` and `slideDown()` methods to control how long the slide animation will run. Durations can be specified either using one of the predefined string `'slow'` or `'fast'`, or in a number of milliseconds; higher values indicate slower animations.

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Sliding up displayed paragraphs with different speeds
• $(".up-btn").click(function(){
• $("p.normal").slideUp();
• $("p.fast").slideUp("fast");
• $("p.slow").slideUp("slow");
• $("p.very-fast").slideUp(50);
• $("p.very-slow").slideUp(2000);
• });
• });
```

- 
- // Sliding down hidden paragraphs with different speeds
- \$(".down-btn").click(function(){
- \$(".p.normal").slideDown();
- \$(".p.fast").slideDown("fast");
- \$(".p.slow").slideDown("slow");
- \$(".p.very-fast").slideDown(50);
- \$(".p.very-slow").slideDown(2000);
- });
- });
- </script>

You can also specify a [callback function](#) to be executed after the `slideUp()` or `slideDown()` method completes. We'll learn more about the callback function in upcoming chapters.

### Example

Try this code »

- <script type="text/javascript">
- \$(document).ready(function(){
- // Display alert message after sliding up paragraphs
- \$(".up-btn").click(function(){
- \$(".p").slideUp("slow", function(){
- // Code to be executed
- alert("The slide-up effect is completed.");
- });
- });
- // Display alert message after sliding down paragraphs
- \$(".down-btn").click(function(){
- \$(".p").slideDown("slow", function(){
- // Code to be executed
- alert("The slide-down effect is completed.");
- });
- });
- });
- </script>

---

## jQuery `slideToggle()` Method

The jQuery `slideToggle()` method show or hide the selected elements by animating their height in such a way that if the element is initially displayed, it will be slide up; if hidden, it will be slide down i.e. toggles between the `slideUp()` and `slideDown()` methods.

## Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Toggles paragraphs display with sliding
4. $(".toggle-btn").click(function(){
5. $("p").slideToggle();
6. });
7. });
8. </script>
```

Similarly, you can specify the duration parameter for the `slideToggle()` method like `slideUp()` and `slideDown()` methods to control the speed of the slide toggle animation.

## Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Slide Toggles paragraphs with different speeds
• $(".toggle-btn").click(function(){
• $("p.normal").slideToggle();
• $("p.fast").slideToggle("fast");
• $("p.slow").slideToggle("slow");
• $("p.very-fast").slideToggle(50);
• $("p.very-slow").slideToggle(2000);
• });
• });
• </script>
```

Similarly, you can also specify a [callback function](#) for the `slideToggle()` method.

## Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Display alert message after slide toggling paragraphs
• $(".toggle-btn").click(function(){
• $("p").slideToggle(1000, function(){
• // Code to be executed
• alert("The slide-toggle effect is completed.");
• });
• });
• });
• </script>
```



# jQuery Animation Effects

*In this tutorial you will learn how to animate CSS properties using jQuery.*

## jQuery `animate()` Method

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

The `animate()` method is typically used to animate numeric CSS properties, for example, [width](#), [height](#), [margin](#), [padding](#), [opacity](#), [top](#), [left](#), etc. but the non-numeric properties such as [color](#) or [background-color](#) cannot be animated using the basic jQuery functionality.

**Note:** Not all CSS properties are animatable. In general, any CSS property that accepts values that are numbers, lengths, percentages, or colors is animatable. However, the color animation is not supported by the core jQuery library. To manipulate and animate the color use the [jQuery color plugin](#).

## Syntax

The basic syntax of the jQuery `animate()` method can be given with:

```
$(selector).animate({ properties }, duration, callback);
```

The parameters of the `animate()` method have the following meanings:

- The required properties parameter defines the [CSS properties](#) to be animated.
- The optional duration parameter specifies how long the animation will run. Durations can be specified either using one of the predefined string `'slow'` or `'fast'`, or in a number of milliseconds; higher values indicate slower animations.
- The optional [callback](#) parameter is a function to call once the animation is complete.

Here's a simple example of the jQuery `animate()` method that animates an image from its original position to the right by 300 pixels on click of the button.

## Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. $("img").animate({
5. left: 300
6. });
7. });
8. });
9. </script>
```

**Note:** All HTML elements have static position by default. Since the static element cannot be moved, so you must set the CSS [position](#) property for the element to `relative`, `fixed`, or `absolute` to manipulate or animate its position.

---

## Animate Multiple Properties At Once

You can also animate multiple properties of an element together at the same time using the `animate()` method. All the properties animated simultaneously without any delay.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $(".box").animate({
• width: "300px",
• height: "300px",
• marginLeft: "150px",
• borderWidth: "10px",
• opacity: 0.5
• });
• });
• });
• </script>
```

**Note:** The [CSS properties](#) names must be camel-cased when using with the `animate()` method, e.g. if you want to animate the font size you need to write `fontSize` rather than `font-size`. Similarly, write `marginLeft` instead of `margin-left`, `borderWidth` instead of `border-width`, and so on.

**Tip:** You must set the [border-style](#) property for the element before animating its [border-width](#) property. An element must have borders before you can animate the border width, because the default value of the `border-style` property is `none`.

---

## Animate Multiple Properties One by One or Queued Animations

You can also animate the multiple properties of an element one by one individually in a queue using the jQuery's chaining feature. We'll learn more about chaining in next chapter.

The following example demonstrates a jQuery queued or chained animation, where each animation will start once the previous animation on the element has completed.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $(".box")
• .animate({width: "300px"})
• .animate({height: "300px"})
• .animate({marginLeft: "150px"})
• .animate({borderWidth: "10px"})
• .animate({opacity: 0.5});
• });
• });
• </script>
```

---

## Animate Properties with Relative Values

You can also define the relative values for the animated properties. If a value is specified with a leading `+=` or `-=` prefix, then the target value is calculated by adding or subtracting the given number from the current value of the property.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $(".box").animate({
• top: "+=50px",
• left: "+=50px",
• width: "+=50px",
• height: "+=50px"
• });
• });
• });
• </script>
```

---

## Animate Properties with Pre-defined Values

In addition to the numeric values, each property can take the strings 'show', 'hide', and 'toggle'. It will be very helpful in a situation when you simply want to animate the property from its current value to the initial value and vice versa.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $(".box").animate({
• width: 'toggle'
• });
• });
• });
• </script>
```

## jQuery Stop Animations

*In this tutorial you will learn how to stop running animations using jQuery.*

### jQuery `stop()` Method

The jQuery `stop()` method is used to stop the jQuery animations or effects currently running on the selected elements before it completes.

The basic syntax of the jQuery `stop()` method can be given with:

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

The parameters in the above syntax have the following meanings:

- The optional `stopAll` Boolean parameter specifies whether to remove queued animation or not. Default value is `false`, that means only the current animation will be stopped, rest of the animations in the queue will run afterwards.
- The optional `goToEnd` Boolean parameter specifies whether to complete the current animation immediately. Default value is `false`.

Here's a simple example that demonstrates the jQuery `stop()` method in real action in which you can start and stop the animation on click of the button.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
```

```

• $(document).ready(function(){
• // Start animation
• $(".start-btn").click(function(){
• $(".img").animate({left: "+=150px"}, 2000);
• });
•
• // Stop running animation
• $(".stop-btn").click(function(){
• $(".img").stop();
• });
•
• // Start animation in the opposite direction
• $(".back-btn").click(function(){
• $(".img").animate({left: "-=150px"}, 2000);
• });
•
• // Reset to default
• $(".reset-btn").click(function(){
• $(".img").animate({left: "0"}, "fast");
• });
• });
• </script>

```

**Note:** The jQuery `stop()` method works for all jQuery effects such as [fading](#), [sliding](#), [animated show and hide](#) effects as well as [custom animations](#).

Here's one more example of this method in which, if you click the "Slide Toggle" button again after starting the animation but before it is completed, the animation will begin in the opposite direction immediately from the saved starting point.

### Example

Try this code »

```

1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Kill and toggle the current sliding animation
4. $(".toggle-btn").on("click", function(){
5. $(".box").stop().slideToggle(1000);
6. });
7. });
8. </script>

```

---

## Creating Smooth Hover Effect

While creating the animated hover effect one of the common problem you may face is multiple queued animations, when you place and remove the mouse cursor rapidly. Because, in this situation [mouseenter](#) or [mouseleave](#) events are triggered quickly before the animation complete. To avoid this problem and create a nice and smooth hover effect you can add the `stop(true, true)` to the method chain, like this:

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $(".box").hover(function(){
4. $(this).find("img").stop(true, true).fadeOut();
5. }, function(){
6. $(this).find("img").stop(true, true).fadeIn();
7. });
8. });
9. </script>
```

## jQuery Chaining

*In this tutorial you will learn how chain multiple methods in jQuery.*

### jQuery Method Chaining

The jQuery provides another robust feature called method chaining that allows us to perform multiple action on the same set of elements, all within a single line of code. This is possible because most of the jQuery methods return a jQuery object that can be further used to call another method. Here's an example.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. $("p").animate({width: "100%"}).animate({fontSize: "46px"}).animate({borderWidth:
5. 30});
6. });
7. </script>
```

The above example demonstrate the chaining of three `animate()` method. When a user click the trigger button, it expands the `<p>` to 100% width. Once the [width](#) change is complete the [font-size](#) is start animating and after its completion, the [border](#) animation will begin.

**Tip:** The method chaining not only helps you to keep your jQuery code concise, but it also can improve your script's performance since browser doesn't have to find the same elements multiple times to do something with them.

You can also break a single line of code into multiple lines for greater readability. For example, the sequence of methods in the above example could also be written as:

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("p")
• .animate({width: "100%"})
• .animate({fontSize: "46px"})
• .animate({borderWidth: 30});
• });
• });
• </script>
```

Some jQuery methods doesn't return the jQuery object. In general, [setters](#) i.e. methods that assign some value on a selection return a jQuery object, that allows you to continue calling jQuery methods on your selection. Whereas, [getters](#) return the requested value, so you can't continue to call jQuery methods on the value returned by the getter.

A typical example of this scenario is the `html()` method. If no parameters are passed to it, the HTML contents of the selected element is returned instead of a jQuery object.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• // This will work
• $("h1").html("Hello World!").addClass("test");
•
• // This will NOT work
• $("p").html().addClass("test");
• });
• });
• </script>
```

## jQuery Callback

*In this tutorial you will learn how define a callback function for the jQuery effect.*

## jQuery Callback Functions

JavaScript statements are executed line by line. But, since jQuery effect takes some time to finish the next line code may execute while the previous effect is still running. To prevent this from happening jQuery provides a callback function for each effect method. A callback function is a function that is executed once the effect is complete. The callback function is passed as an argument to the effect methods and they typically appear as the last argument of the method. For example, the basic syntax of the jQuery `slideToggle()` effect method with a callback function can be given with:

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

Consider the following example in which we've placed the `slideToggle()` and `alert()` statements next to each other. If you try this code the alert will be displayed immediately once you click the trigger button without waiting for slide toggle effect to complete.

### Example

Try this code »

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. $("p").slideToggle("slow");
5. alert("The slide toggle effect has completed.");
6. });
7. });
8. </script>
```

And, here's the modified version of the previous example in which we've placed the `alert()` statement inside a callback function for the `slideToggle()` method. If you try this code the alert message will be displayed once the slide toggle effect has completed.

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("p").slideToggle("slow", function(){
• // Code to be executed once effect is complete
• alert("The slide toggle effect has completed.");
• });
• });
• });
• </script>
```

Similarly, you can define the callback functions for the other jQuery effect methods, like `show()`, `hide()`, `fadeIn()`, `fadeOut()`, `animate()`, etc.



**Note:** If the effect method is applied to multiple elements, then the callback function is executed once for each selected element, not once for all.

### Example

Try this code »

```
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("h1, p").slideToggle("slow", function(){
• // Code to be executed once effect is complete
• alert("The slide toggle effect has completed.");
• });
• });
• });
• </script>
```

If you try the above example code, it will display the same alert message two times once per `<h1>` and `<p>` element, upon clicking the trigger button.

## jQuery Getters & Setter

*In this tutorial you will learn how to get or set the element's content and attribute value as well as the from control value using jQuery.*

## jQuery Get or Set Contents and Values

Some jQuery methods can be used to either assign or read some value on a selection. A few of these methods are `text()`, `html()`, `attr()`, and `val()`.

When these methods are called with no argument, it is referred to as a *getter*, because it gets (or reads) the value of the element. When these methods are called with a value as an argument, it's referred to as a *setter* because it sets (or assigns) that value.

## jQuery `text()` Method

The jQuery `text()` method is either used to get the combined text contents of the selected elements, including their descendants, or set the text contents of the selected elements.

**Get Contents with `text()` Method**

The following example will show you how to get the text contents of paragraphs:

### Example

Try this code »

```

• <script type="text/javascript">
• $(document).ready(function(){
• // Get combined text contents of all paragraphs
• $(".btn-one").click(function(){
• var str = $("p").text();
• alert(str);
• });
•
• // Get text contents of the first paragraph
• $(".btn-two").click(function(){
• var str = $("p:first").text();
• alert(str);
• });
• });
• </script>

```

**Note:** The jQuery `text()` retrieves the values of all the selected elements (i.e. combined text), whereas the other getters such as `html()`, `attr()`, and `val()` returns the value only from the first element in the selection.

Set Contents with `text()` Method

The following example will show you how to set the text contents of a paragraph:

### Example

[Try this code »](#)

```

• <script type="text/javascript">
• $(document).ready(function(){
• // Set text contents of all paragraphs
• $(".btn-one").click(function(){
• $("p").text("This is demo text.");
• });
•
• // Set text contents of the first paragraph
• $(".btn-two").click(function(){
• $("p:first").text("This is another demo text.");
• });
• });
• </script>

```

**Note:** When the jQuery `text()`, `html()`, `attr()`, and `val()` methods are called with a value as an argument it sets that value to every matched element.

---

## jQuery `html()` Method

The jQuery `html()` method is used to get or set the HTML contents of the elements.

### Get HTML Contents with `html()` Method

The following example will show you how to get the HTML contents of the paragraph elements as well as a `<div>` element container:

#### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Get HTML contents of first selected paragraph
• $(".btn-one").click(function(){
• var str = $("p").html();
• alert(str);
• });
•
• // Get HTML contents of an element with ID container
• $(".btn-two").click(function(){
• var str = $("#container").html();
• alert(str);
• });
• });
• </script>
```

**Note:** If multiple elements are selected, the `html()` method only returns the HTML contents of the first element from the set of matched elements.

### Set HTML Contents with `html()` Method

The following example will show you how to set the HTML contents of the `<body>` element:

#### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Set HTML contents for document's body
4. $(".button").click(function(){
5. $("body").html("<p>Hello World!</p>");
6. });
7. });
8. </script>
```

---

## jQuery `attr()` Method

You can use the jQuery `attr()` method to either get the value of an element's attribute or set one or more attributes for the selected element.

#### Get Attribute Value with `attr()` Method

The following example will show you how get the `href` attribute of the hyperlink i.e. the `<a>` element as well as the `alt` attribute of an `<img>` element:

#### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Get href attribute value of first selected hyperlink
4. $(".btn-one").click(function(){
5. var str = $("a").attr("href");
6. alert(str);
7. });
8.
9. // Get alt attribute value of an image with ID sky
10. $(".btn-two").click(function(){
11. var str = $("img#sky").attr("alt");
12. alert(str);
13. });
14. });
15. </script>
```

**Note:** If multiple elements are selected, the `attr()` method only returns the attribute value of the first element from the set of matched elements.

#### Set Attributes with `attr()` Method

The following example will show you how to set the `checked` attribute of the checkbox.

#### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. // Check all the checkboxes
4. $(".button").click(function(){
5. $('input[type="checkbox"]').attr("checked", "checked");
6. });
7. });
8. </script>
```

The `attr()` method also allows you to set multiple attributes at a time. The following example will show you how to set the `class` and `title` attribute for the `<img>` elements.

#### Example

[Try this code »](#)

```

• <script type="text/javascript">
• $(document).ready(function(){
• // Add a class and title attribute to all the images
• $("button").click(function(){
• $("img").attr({
• "class" : "frame",
• "title" : "Hot Air Balloons"
• });
• });
• });
• </script>

```

## jQuery val() Method

The jQuery `val()` method is mainly used to get or set the current value of the [HTML form elements](#) such as `<input>`, `<select>` and `<textarea>`.

Get the Values of Form Fields with `val()` Method

The following example will show you how to get the values of form controls:

### Example

[Try this code »](#)

```

• <script type="text/javascript">
• $(document).ready(function(){
• // Get value of a text input with ID name
• $("button.get-name").click(function(){
• var name = $('input[type="text"]#name').val();
• alert(name);
• });
•
• // Get value of a textarea with ID comment
• $("button.get-comment").click(function(){
• var comment = $("textarea#comment").val();
• alert(comment);
• });
•
• // Get value of a select box with ID city
• $("button.get-city").click(function(){
• var city = $("select#city").val();
• alert(city);
• });
• });
• </script>

```

**Note:** If multiple form elements are selected, the `val()` method only returns the value of the first element from the set of matched elements.

Set the Values of Form Fields with `val()` Method

The following example will show you how to set the values of the form controls:

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Set value of all the text inputs
• $("button").click(function(){
• var text = $(this).text();
• $('input[type="text"]').val(text);
• });
• });
• </script>
```

## jQuery Insert Content

*In this tutorial you will learn how to insert new elements or contents to the document using jQuery.*

### jQuery Insert New Content

jQuery provides several methods,

like `append()`, `prepend()`, `html()`, `text()`, `before()`, `after()`, `wrap()` etc. that allows us to insert new content inside an existing element.

The jQuery `html()` and `text()` methods have already covered in the previous chapter, so in this chapter, we will discuss about the rest of them.

### jQuery `append()` Method

The jQuery `append()` method is used to insert content to the end of the selected elements.

The following example will append some HTML to all the paragraphs on document ready, whereas append some text to the container element on button click.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
```

- `// Append all paragraphs`
- `$("#p").append(' <a href="#">read more...</a>');`
- 
- `// Append an element with ID container`
- `$("#button").click(function(){`
- `$("#container").append("This is demo text.");`
- `});`
- `});`
- `</script>`

**Note:** The contents or elements inserted using the jQuery `append()` and `prepend()` methods is added inside of the selected elements.

## jQuery `prepend()` Method

The `prepend()` method is used to insert content to the beginning of the selected elements.

The following example will prepend some HTML to all the paragraphs on document ready, whereas prepend some text to the container element on button click.

### Example

[Try this code »](#)

- `<script type="text/javascript">`
- `$(document).ready(function(){`
- `// Prepend all paragraphs`
- `$("#p").prepend("<strong>Note:</strong> ");`
- 
- `// Prepend an element with ID container`
- `$("#button").click(function(){`
- `$("#container").prepend("This is demo text.");`
- `});`
- `});`
- `</script>`

## Insert Multiple Elements with `append()` and `prepend()` Method

The jQuery `append()` and `prepend()` also supports passing in multiple arguments as input.

The jQuery code in the following example will insert a `<h1>`, `<p>` and an `<img>` element inside the `<body>` element as a last three child nodes.

## Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• var newHeading = "<h1>Important Note:</h1>";
• var newParagraph = document.createElement("p");
• newParagraph.innerHTML = "Lorem Ipsum is dummy text...";
• var newImage = $('');
• $("body").append(newHeading, newParagraph, newImage);
• });
• </script>
```

---

## jQuery `before()` Method

The jQuery `before()` method is used to insert content before the selected elements. The following example will insert a paragraph before the container element on document ready, whereas insert an image before the `<h1>` element on button click.

## Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Add content before an element with ID container
• $("#container").before("<p>— The Beginning —</p>");
•
• // Add content before headings
• $("button").click(function(){
• $("h1").before('');
• });
• });
• </script>
```

**Note:** The contents or elements inserted using the jQuery `before()` and `after()` methods is added outside of the selected elements.

---

## jQuery `after()` Method

The jQuery `after()` method is used to insert content after the selected elements.



The following example will insert a paragraph after the container element on document ready, whereas insert an image after the `<h1>` element on button click.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Add content after an element with ID container
• $("#container").after("<p>— The End —</p>");
•
• // Add content after headings
• $("button").click(function(){
• $("h1").after('');
• });
• });
• </script>
```

---

## Insert Multiple Elements with `before()` and `after()` Method

The jQuery `before()` and `after()` also supports passing in multiple arguments as input. The following example will insert a `<h1>`, `<p>` and an `<img>` element before the `<p>` elements.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• var newHeading = "<h2>Important Note:</h2>";
• var newParagraph = document.createElement("p");
• newParagraph.innerHTML = "Lorem Ipsum is dummy text...";
• var newImage = $('');
• $("p").before(newHeading, newParagraph, newImage);
• });
• </script>
```

---

## jQuery `wrap()` Method

The jQuery `wrap()` method is used to wrap an HTML structure around the selected elements.

The following example will wrap the container elements with a `<div>` element with the class `.wrapper` on document ready, whereas wrap all the inner content of the paragraph elements first with the `<b>` and again with `<em>` element.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Wrap elements with class container with HTML
• $(".container").wrap('<div class="wrapper"></div>');
•
• // Wrap paragraph's content with HTML
• $("button").click(function(){
• $("p").contents().wrap("");
• });
• });
• </script>
```

## jQuery Remove Elements & Attribute

*In this tutorial you will learn how to remove the HTML elements or its contents as well as its attribute from the document using jQuery.*

### jQuery Remove Elements or Contents

jQuery provides handful of methods, such as `empty()`, `remove()`, `unwrap()` etc. to remove existing HTML elements or contents from the document.

#### jQuery `empty()` Method

The jQuery `empty()` method removes all child elements as well as other descendant elements and the text content within the selected elements from the DOM.

The following example will remove all the content inside of the elements with the class `.container` on click of the button.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Empty container element
• $("button").click(function(){
• $(".container").empty();
• });
• });
• </script>
```

- });
- });
- </script>

**Note:** According to the W3C (World Wide Web Consortium) DOM specification, any string of text within an element is considered a child node of that element.

## jQuery `remove()` Method

The jQuery `remove()` method removes the selected elements from the DOM as well as everything inside it. In addition to the elements themselves, all bound events and jQuery data associated with the elements are removed.

The following example will remove all the `<p>` elements with the class `.hint` from the DOM on button click. Nested elements inside these paragraphs will be removed, too.

### Example

[Try this code »](#)

- <script type="text/javascript">
- \$(document).ready(function(){
- // Removes paragraphs with class "hint" from DOM
- \$("#button").click(function(){
- \$("p.hint").remove();
- });
- });
- </script>

The jQuery `remove()` method can also include a selector as an optional parameter, that allows you to filter the elements to be removed. For instance, the previous example's jQuery DOM removal code could be rewritten as follows:

### Example

[Try this code »](#)

- <script type="text/javascript">
- \$(document).ready(function(){
- // Removes paragraphs with class "hint" from DOM
- \$("#button").click(function(){
- \$("p").remove(".hint");
- });
- });
- </script>

**Note:** You can also include selector expression as a parameter within the jQuery `remove()` method, like `remove(".hint, .demo")` to filter multiple elements.

---

## jQuery `unwrap()` Method

The jQuery `unwrap()` method removes the parent elements of the selected elements from the DOM. This is typically the inverse of the `wrap()` method.

The following example will remove the parent element of `<p>` elements on button click.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Removes the paragraph's parent element
• $("button").click(function(){
• $("p").unwrap();
• });
• });
• </script>
```

---

## jQuery `removeAttr()` Method

The jQuery `removeAttr()` method removes an [attribute](#) from the selected elements.

The example below will remove the `href` attribute from the `<a>` elements on button click.

### Example

[Try this code »](#)

```
• <script type="text/javascript">
• $(document).ready(function(){
• // Removes the hyperlink's href attribute
• $("button").click(function(){
• $("a").removeAttr("href");
• });
• });
• </script>
```

# jQuery Add and Remove CSS Classes

*In this tutorial you will learn how to add or remove CSS classes using jQuery.*

## jQuery CSS Classes Manipulation

jQuery provides several methods, such as `addClass()`, `removeClass()`, `toggleClass()`, etc. to manipulate the CSS classes assigned to HTML elements.

### jQuery `addClass()` Method

The jQuery `addClass()` method adds one or more classes to the selected elements. The following example will add the class `.page-header` to the `<h1>` and the class `.highlight` to the `<p>` elements with class `.hint` on button click.

#### Example

Try this code »

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery addClass() Demo</title>
• <style type="text/css">
• .page-header{
• color: red;
• text-transform: uppercase;
• }
• .highlight{
• background: yellow;
• }
• </style>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("h1").addClass("page-header");
• $("p.hint").addClass("highlight");
• });
• });
• </script>
• </head>
• <body>
• <h1>Demo Text</h1>
• <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
• <p class="hint">Tip: Lorem Ipsum is dummy text.</p>
```

- `<button type="button">Add Class</button>`
- `</body>`
- `</html>`

You can also add multiple classes to the elements at a time. Just specify the space separated list of classes within the `addClass()` method, like this:

### Example

Try this code »

- `<!DOCTYPE html>`
- `<html lang="en">`
- `<head>`
- `<meta charset="utf-8">`
- `<title>jQuery addClass() Demo</title>`
- `<style type="text/css">`
- `.page-header{`
- `color: red;`
- `text-transform: uppercase;`
- `}`
- `.highlight{`
- `background: yellow;`
- `}`
- `</style>`
- `<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>`
- `<script type="text/javascript">`
- `$(document).ready(function(){`
- `$("button").click(function(){`
- `$("h1").addClass("page-header highlight");`
- `});`
- `});`
- `</script>`
- `</head>`
- `<body>`
- `<h1>Hello World</h1>`
- `<p>The quick brown fox jumps over the lazy dog.</p>`
- `<button type="button">Add Class</button>`
- `</body>`
- `</html>`

## jQuery `removeClass()` Method

Similarly, you can remove the classes from the elements using the jQuery `removeClass()` method. The `removeClass()` method can remove a single class, multiple classes, or all classes at once from the selected elements.

The following example will remove the class `.page-header` from the `<h1>` and the class `.hint` and `.highlight` from the `<p>` elements on button click.

### Example

Try this code »

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery removeClass() Demo</title>
• <style type="text/css">
• .page-header{
• color: red;
• text-transform: uppercase;
• }
• .highlight{
• background: yellow;
• }
• </style>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("h1").removeClass("page-header");
• $("p").removeClass("hint highlight");
• });
• });
• </script>
• </head>
• <body>
• <h1 class="page-header">Demo Text</h1>
• <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
• <p class="hint highlight">Tip: Lorem Ipsum is dummy text.</p>
• <button type="button">Remove Class</button>
• </body>
• </html>
```

When the `removeClass()` method is called without an argument it will remove all the classes from the selected elements. Here's an example:

### Example

Try this code »

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery removeClass() Demo</title>
• <style type="text/css">
```

```

• .page-header{
• color: red;
• text-transform: uppercase;
• }
• .highlight{
• background: yellow;
• }
• </style>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("h1").removeClass();
• $("p").removeClass();
• });
• });
• </script>
• </head>
• <body>
• <h1 class="page-header">Demo Text</h1>
• <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
• <p class="hint highlight">Tip: Lorem Ipsum is dummy text.</p>
• <button type="button">Remove Class</button>
• </body>
• </html>

```

---

## jQuery toggleClass() Method

The jQuery `toggleClass()` add or remove one or more classes from the selected elements in such a way that if the selected element already has the class, then it is removed; if an element does not have the class, then it is added i.e. toggle classes.

### Example

Try this code »

```

• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery toggleClass() Demo</title>
• <style type="text/css">
• p{
• padding: 10px;
• cursor: pointer;
• font: bold 16px sans-serif;
• }

```



- `.highlight{`
- `background: yellow;`
- `}`
- `</style>`
- `<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>`
- `<script type="text/javascript">`
- `$(document).ready(function(){`
- `$("p").click(function(){`
- `$(this).toggleClass("highlight");`
- `});`
- `});`
- `</script>`
- `</head>`
- `<body>`
- `<p>Click on me to toggle highlighting.</p>`
- `<p class="highlight">Click on me to toggle highlighting.</p>`
- `<p>Click on me to toggle highlighting.</p>`
- `</body>`
- `</html>`

## jQuery Get and Set CSS Properties

*In this tutorial you will learn how to get or set style properties using jQuery.*

### jQuery `css()` Method

The jQuery `css()` method is used to get the [computed value](#) of a CSS property or set one or more CSS properties for the selected elements.

This method provides a quick way to apply the styles directly to the HTML elements (i.e. [inline styles](#)) that haven't been or can't easily be defined in a stylesheet.

### Get a CSS Property Value

You can get the computed value of an element's CSS property by simply passing the property name as a parameter to the `css()` method. Here's the basic syntax:

```
$(selector).css("propertyName");
```

The following example will retrieve and display the computed value of the CSS [background-color](#) property of a `<div>` element, when it is clicked.

#### Example

[Try this code »](#)

- `<!DOCTYPE html>`
- `<html lang="en">`
- `<head>`
- `<meta charset="utf-8">`

```

• <title>jQuery css() Demo</title>
• <style type="text/css">
• div{
• width: 100px;
• height: 100px;
• margin: 10px;
• cursor: pointer;
• display: inline-block;
• }
• </style>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("div").click(function(){
• var color = $(this).css("background-color");
• $("#result").html(color);
• });
• });
• </script>
• </head>
• <body>
• <div style="background-color:orange;"></div>
• <div style="background-color:#ee82ee;"></div>
• <div style="background-color:rgb(139,205,50);"></div>
• <div style="background-color:#f00;"></div>
• <p>The computed background-color property value of this DIV element is: <b
id="result"></p>
• </body>
• </html>

```

## Set a Single CSS Property and Value

The `css()` method can take a property name and value as separate parameters for setting a single CSS property for the elements. The basic syntax can be given with: `$(selector).css("propertyName", "value");`

The following example will set the CSS `background-color` property of the `<div>` elements, to the `color value` `red`, when it is clicked.

### Example

[Try this code »](#)

```

• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery css() Demo</title>
• <style type="text/css">
• .box{
• width: 100px;
• height: 100px;

```

```

• margin: 10px;
• cursor: pointer;
• border: 1px solid #cdcdcd;
• display: inline-block;
• }
• </style>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $(".box").click(function(){
• $(this).css("background-color", "red");
• });
• });
• </script>
• </head>
• <body>
• <div class="box"></div>
• <div class="box"></div>
• <div class="box"></div>
• <div class="box"></div>
• </body>
• </html>

```

## Set Multiple CSS Properties and Values

You can also set multiple CSS properties with the `css()` method. The basic syntax for setting the more than one property for the elements can be given with:

```
$(selector).css({"propertyName":"value", "propertyName":"value", ...});
```

The following example will set the `background-color` as well as the `padding` CSS property for the selected elements at the same time.

### Example

[Try this code »](#)

```

• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery css() Demo</title>
• <style type="text/css">
• p{
• font-size: 18px;
• font-family: Arial, sans-serif;
• }
• </style>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("p").css({"background-color": "yellow", "padding": "20px"});

```

- `});`
- `});`
- `</script>`
- `</head>`
- `<body>`
- `<h1>This is a heading</h1>`
- `<p style="background-color:orange;">This a paragraph.</p>`
- `<p style="background-color:#ee82ee;">This is another paragraph.</p>`
- `<p style="background-color:rgb(139,205,50);">This is none more paragraph.</p>`
- `<p>This is one last paragraph.</p>`
- `<button type="button">Add CSS Styles</button>`
- `</body>`

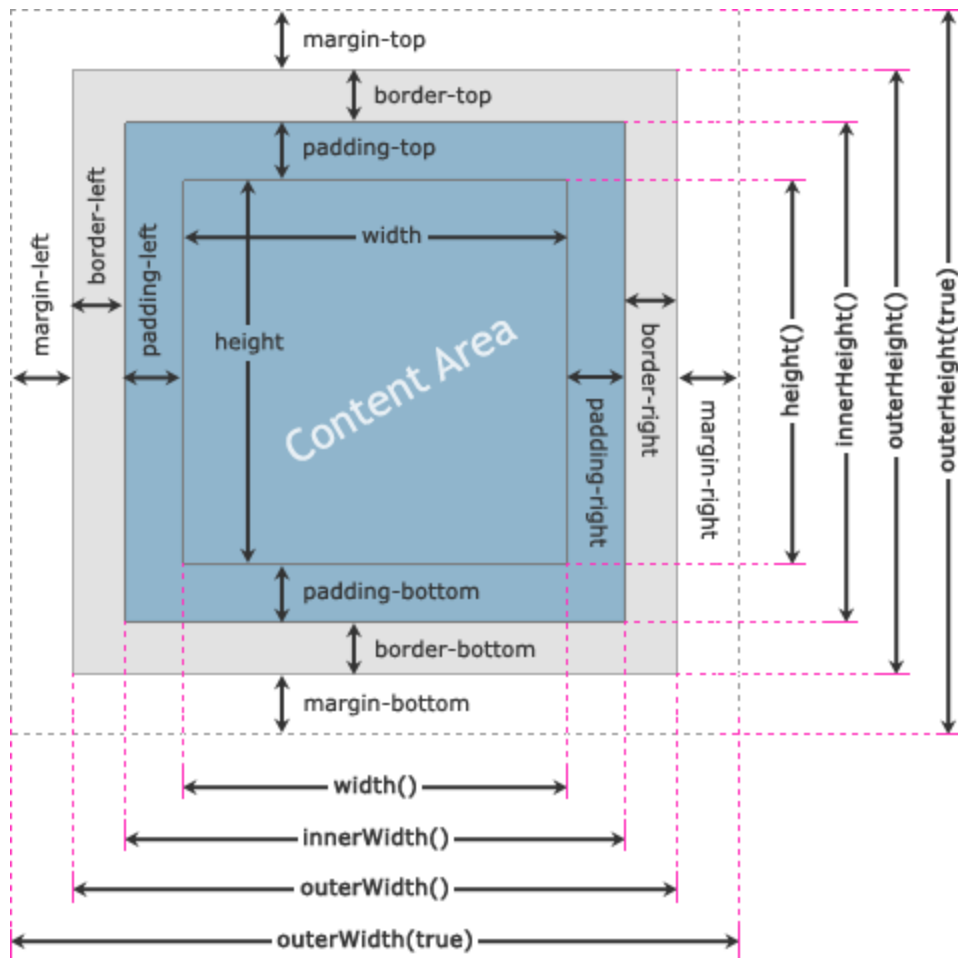
## jQuery Dimensions

*In this tutorial you will learn how to get or set dimensions of an element's box such as width and height using jQuery.*

### Understanding the jQuery Dimensions

jQuery provides several methods, such

as `height()`, `innerHeight()`, `outerHeight()`, `width()`, `innerWidth()` and `outerWidth()` to get and set the CSS dimensions for the elements. Check out the following illustration to understand how these methods are calculating the dimensions of an element's box.



## jQuery `width()` and `height()` Methods

The jQuery `width()` and `height()` methods get or set the [width](#) and the [height](#) of the element respectively. This width and height doesn't include [padding](#), [border](#) and [margin](#) on the element. The following example will return the width and height of a `<div>` element.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. var divWidth = $("#box").width();
5. var divHeight = $("#box").height();
6. $("#result").html("Width: " + divWidth + ", " + "Height: " + divHeight);
7. });
```

```
8. });
9. </script>
```

Similarly, you can set the width and height of the element by including the value as a parameter within the `width()` and `height()` method. The value can be either a string (number and unit e.g. 100px, 20em, etc.) or a number. The following example will set the width of a `<div>` element to 400 pixels and height to 300 pixels respectively.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. $("#box").width(400).height(300);
5. });
6. });
7. </script>
```

**Note:** Use the jQuery `width()` or `height()` method if you want to use an element's width or height in a mathematical calculation, since it returns the `width` and `height` property value as a unit-less pixel value (e.g. 400). Whereas, the `css("width")` or `css("height")` methods returns value with units (e.g. 400px).

---

## jQuery `innerWidth()` and `innerHeight()` Methods

The jQuery `innerWidth()` and `innerHeight()` methods get or set the *inner width* and the *inner height* of the element respectively. This inner width and height includes the `padding` but excludes `border` and `margin` on the element. The following example will return the inner width and height of a `<div>` element on the click of a button.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. var divWidth = $("#box").innerWidth();
5. var divHeight = $("#box").innerHeight();
6. $("#result").html("Inner Width: " + divWidth + ", " + "Inner Height: " +
divHeight);
7. });
8. });
9. </script>
```

Similarly, you can set the element's inner width and height by passing the value as a parameter to the `innerWidth()` and `innerHeight()` method. These methods only alter the width or height of the element's content area to match the specified value.

For example, if the current width of the element is 300 pixels and the sum of the left and right padding is equal to 50 pixels then the new width of the element after setting the inner width to 400 pixels is 350 pixels i.e.  $\text{New Width} = \text{Inner Width} - \text{Horizontal Padding}$ . Similarly, you can estimate the change in height while setting the inner height.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("#button").click(function(){
4. $("#box").innerWidth(400).innerHeight(300);
5. });
6. });
7. </script>
```

---

## jQuery `outerWidth()` and `outerHeight()` Methods

The jQuery `outerWidth()` and `outerHeight()` methods get or set the *outer width* and the *outer height* of the element respectively. This outer width and height includes `padding` and `border` but excludes the `margin` on the element. The following example will return the outer width and height of a `<div>` element on the click of a button.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("#button").click(function(){
4. var divWidth = $("#box").outerWidth();
5. var divHeight = $("#box").outerHeight();
6. $("#result").html("Outer Width: " + divWidth + ", " + "Outer Height: " +
7. divHeight);
8. });
9. });
10. </script>
```

You can also get the outer width and height that includes `padding` and `border` as well as the `margin` of the element. For that just specify the `true` parameter for the outer width methods, like `outerWidth(true)` and `outerHeight(true)`.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. var divWidth = $("#box").outerWidth(true);
5. var divHeight = $("#box").outerHeight(true);
6. $("#result").html("Outer Width: " + divWidth + ", " + "Outer Height: " +
divHeight);
7. });
8. });
9. </script>
```

Similarly, you can set the element's outer width and height by passing the value as a parameter to the `outerWidth()` and `outerHeight()` methods. These methods only alter the width or height of the element's content area to match the specified value, like the `innerWidth()` and `innerHeight()` methods.

For example, if the current width of the element is 300 pixels, and the sum of the left and right padding is equal to 50 pixels, and the sum of the width of the left and right border is 20 pixels then the new width of the element after setting the outer width to 400 pixels is 330 pixels i.e.  $\text{New Width} = \text{Outer Width} - (\text{Horizontal Padding} + \text{Horizontal Border})$ . Similarly, you can estimate the change in height while setting the outer height.

### Example

[Try this code »](#)

```
1. <script type="text/javascript">
2. $(document).ready(function(){
3. $("button").click(function(){
4. $("#box").outerWidth(400).outerHeight(300);
5. });
6. });
7. </script>
```

## jQuery Ajax Load

*In this tutorial you will learn how to load data from server using jQuery.*

### jQuery `load()` Method

The jQuery `load()` method loads data from the server and place the returned HTML into the selected element. This method provides a simple way to load data asynchronous from a web server. The basic syntax of this method can be given with:

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



The parameters of the `load()` method has the following meaning:

- The required *URL* parameter specifies the URL of the file you want to load.
- The optional *data* parameter specifies a set of query string (i.e. key/value pairs) that is sent to the web server along with the request.
- The optional *complete* parameter is basically a callback function that is executed when the request completes. The callback is fired once for each selected element.

Let's put this method into real use. Create a blank HTML file "test-content.html" and save it somewhere in your [web server](#). Now place the following HTML code inside of this file:

### Example

[Try this code »](#)

1. `<h1>Simple Ajax Demo</h1>`
2. `<p id="hint">This is a simple example of Ajax loading.</p>`
3. `<p></p>`

Now, create one more HTML file say "load-demo.html", and save it at the same location where you've saved the previous file. Now put the following HTML code inside of it:

### Example

[Try this code »](#)

- `<!DOCTYPE html>`
- `<html lang="en">`
- `<head>`
- `<meta charset="utf-8">`
- `<title>jQuery load() Demo</title>`
- `<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>`
- `<script type="text/javascript">`
- `$(document).ready(function(){`
- `$("#button").click(function(){`
- `$("#box").load("test-content.html");`
- `});`
- `});`
- `</script>`
- `</head>`
- `<body>`
- `<div id="box">`
- `<h2>Click button to load new content inside DIV box</h2>`
- `</div>`
- `<button type="button">Load Content</button>`
- `</body>`
- `</html>`

Finally, open this page in your browser and click the "Load Content" button. You'll see the content of DIV box is replaced by the HTML content of the "test-content.html" file.

**Tip:** To test this Ajax example you need to place the HTML files on a web server. You can [set up a local web server](#) on your PC by installing WampServer or XAMPP. You must open the demo file using "http://" since Ajax makes HTTP requests.

**Note:** Ajax request can be made only to the files that exist on the same web server that serves the page from which the Ajax request is sent, not to external or remote servers for security reasons. This is called same-origin policy.

Further, the callback function can have three different parameters:

- **responseTxt** — Contains the resulting content if the request succeeds.
- **statusTxt** — Contains the status of the request such as success or error.
- **jqXHR** — Contains the XMLHttpRequest object.

Here's the modified version of the previous example that will display either the success or error message to the user depending on the status of the request.

### Example

[Try this code »](#)

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery load() Demo</title>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("button").click(function(){
• $("#box").load("test-content.html", function(responseTxt, statusTxt, jqXHR){
• if(statusTxt == "success"){
• alert("New content loaded successfully!");
• }
• if(statusTxt == "error"){
• alert("Error: " + jqXHR.status + " " + jqXHR.statusText);
• }
• });
• });
• });
• </script>
• </head>
• <body>
• <div id="box">
• <h2>Click button to load new content inside DIV box</h2>
• </div>
• <button type="button">Load Content</button>
• </body>
• </html>
```

---

## Loading Page Fragments

The jQuery `load()` also allows us to fetch only a portion of the document. This is simply achieved by appending the `url` parameter with a space followed by a [jQuery selector](#), let's check out the following example to make it more clear.

### Example

[Try this code »](#)

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery load() Demo</title>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("#button").click(function(){
• $("#box").load("test-content.html #hint");
• });
• });
• </script>
• </head>
• <body>
• <div id="box">
• <h2>Click button to load new content inside DIV box</h2>
• </div>
• <button type="button">Load Content</button>
• </body>
• </html>
```

## jQuery Ajax GET and POST Requests

*In this tutorial you will learn how to send and receive data from a web server through Ajax via HTTP GET or POST methods using jQuery.*

### jQuery `$.get()` and `$.post()` Methods

The jQuery's `$.get()` and `$.post()` methods provide simple tools to [send and retrieve data asynchronously](#) from a web server. Both the methods are pretty much identical, apart from one major difference — the `$.get()` makes Ajax requests using the [HTTP GET method](#), whereas the `$.post()` makes Ajax requests using the [HTTP POST method](#).

The basic syntax of these methods can be given with:

```
$.get(URL, data, success); -Or- $.post(URL, data, success);
```

The parameters in the above syntax have the following meaning:

- The required *URL* parameter specifies the URL to which the request is sent.
- The optional *data* parameter specifies a set of query string (i.e. key/value pairs) that is sent to the web server along with the request.
- The optional *success* parameter is basically a callback function that is executed if the request succeeds. It is typically used to retrieve the returned data.

**Note:** The HTTP GET and POST methods are used to send request from a browser to a server. The main difference between these methods is the way in which the data is passed to the server. Check out the tutorial on [GET and POST methods](#) for the detailed explanation and comparison between these two methods.

## Performing GET Request with AJAX using jQuery

The following example uses the jQuery `$.get()` method to make an Ajax request to the "date-time.php" file using HTTP GET method. It simply retrieves the date and time returned from the server and displays it in the browser without refreshing the page.

### Example

[Try this code »](#)

```

• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery get() Demo</title>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("#button").click(function(){
• $.get("date-time.php", function(data){
• // Display the returned data in browser
• $("#result").html(data);
• });
• });
• });
• </script>
• </head>
• <body>
• <div id="result">
• <h2>Content of the result DIV box will be replaced by the server date and
time</h2>
• </div>
• <button type="button">Load Date and Time</button>
• </body>
• </html>

```

Here's our "date-time.php" file that simply output the current date and time of the server.

### **Example** **Download**

```
1. <?php
2. // Return current date and time from the server
3. echo date("F d, Y h:i:s A");
4. ?>
```

**Tip:** If you face any difficulty while running these examples locally on your PC, please check out the tutorial on [jQuery Ajax load](#) for the solution.

You can also send some data to the server with the request. In the following example the jQuery code makes an Ajax request to the "create-table.php" as well as sends some additional data to the server along with the request.

### **Example** **Try this code »**

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery get() Demo</title>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("#button").click(function(){
• // Get value from input element on the page
• var numValue = $("#num").val();
•
• // Send the input data to the server using get
• $.get("create-table.php", {number: numValue} , function(data){
• // Display the returned data in browser
• $("#result").html(data);
• });
• });
• });
• </script>
• </head>
• <body>
• <label>Enter a Number: <input type="text" id="num"></label>
• <button type="button">Show Multiplication Table</button>
• <div id="result"></div>
• </body>
• </html>
```

Here's the PHP script of our "create-table.php" file that simply output the multiplication table for the number entered by the user on button click.

### **Example** **Download**

```
• <?php
• $number = htmlspecialchars($_GET["number"]);
• if(is_numeric($number) && $number > 0){
• echo "<table>";
• for($i=0; $i<11; $i++){
• echo "<tr>";
• echo "<td>$number x $i</td>";
• echo "<td>=</td>";
• echo "<td>" . $number * $i . "</td>";
• echo "</tr>";
• }
• echo "</table>";
• }
• }
• ?>
```

---

## Performing POST Request with AJAX using jQuery

POST requests are identical to GET requests in jQuery. So, generally which method you should use either `$.get()` or `$.post()` is basically depends on the requirements of your server-side code. If you have large amount of data to be transmitted (e.g. form data) you need to use POST, because GET has a stringent limit on the data transfer.

### **Example** **Try this code »**

```
• <!DOCTYPE html>
• <html lang="en">
• <head>
• <meta charset="utf-8">
• <title>jQuery post() Demo</title>
• <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
• <script type="text/javascript">
• $(document).ready(function(){
• $("form").submit(function(event){
• // Stop form from submitting normally
• event.preventDefault();
•
• /* Serialize the submitted form control values to be sent to the web server with
the request */
```

```

• var formValues = $(this).serialize();
•
• // Send the form data using post
• $.post("display-comment.php", formValues, function(data){
• // Display the returned data in browser
• $("#result").html(data);
•
• });
• });
• });
• </script>
• </head>
• <body>
• <form>
• <label>Name: <input type="text" name="name"></label>
• <label>Comment: <textarea cols="50" name="comment"></textarea></label>
• <input type="submit" value="Send">
• </form>
• <div id="result"></div>
• </body>
• </html>

```

Here's our "display-comment.php" file that simply output the data entered by the user.

### ***Example*** **Download**

```

1. <?php
2. $name = htmlspecialchars($_POST["name"]);
3. $comment = htmlspecialchars($_POST["comment"]);
4. echo "Hi, $name. Your comment has been received successfully." . "
";
5. echo "Here's the comment what you've entered: $comment";
6. ?>

```