JQuery In 8 Hours

For Beginners
Learn JQuery Fast!

Ray Yao

JQuery
In 8 Hours

By Ray Yao **For Beginners Learn JQuery Fast!**

Ray Yao

About the Author Ray Yao: Certified PHP engineer by Zend, USA Certified JAVA programmer by Sun, USA Certified SCWCD developer by Oracle, USA Certified A+ professional by CompTIA, USA Certified ASP. NET expert by Microsoft, USA Certified MCP professional by Microsoft, USA Certified TECHNOLOGY specialist by Microsoft, USA Certified NETWORK+ professional by CompTIA, USA

To know more Ray Yao's books in Amazon: www.websprogram.com/books.php

www.amazon.com/author/rayyao

Preface

This book is a useful book about JQuery programming. You can learn complete primary knowledge of JQuery fast and easily. The straightforward definitions, the plain and simple examples, the elaborate explanations and the neat and beautiful layout feature this helpful and educative book. You will be impressed by the new distinctive composing style. Reading this book is a great enjoyment! You can master all essential JQuery skill quickly.

Source Code for Download

This book provides source code for download; you can download the source code for better study, or copy the source code to your favorite editor to test the programs. The download link of the source code is at the last page of this book.

Start Coding Today!

Table of Contents

Hour 1
JQuery Basic

What is jQuery?

JQuery is a client side-scripting library of JavaScript. You can access any element, make animation, and validate input by using the library. No extra code can achieve the result by writing one or few lines of code instead of writing dozen lines of codes. You can handle the events easily in the html document; get fast results from server using Ajax, and so on……

Advantages:

It helps to run with all kind of browsers and is compatible various browsers.

It helps to implement critical functionality without writing hundreds of line of codes.

It is fast to implement customized action.

Know more JQuery:

http://learn.jquery.com/

http://api.jquery.com/

http://forum.jquery.com/

http://www.websprogram.com/

Download JQuery:

http://jquery.com/download/

Install jQuery

We can use jQuery in two ways: (1) If you want to use jQuery file locally, then download it. Download jQuery from http://jquery.com/download/, put the downloaded file in **the same folder** with your jQuery files, and reference it in <head> section of html document as following:

```
<head> <script src="jquery-1.11.3.js"> </script> </head>
```

(2) If you do not want to download jQuery, then you can include it in html document as following:

```
<head> <script src="http://code.jquery.com/jquery-latest.js"></script>

</head>
```

First jQuery script

Now we write the first jQuery script. Create a new html document named “Test.html”.

Example 1.1

```
<html>
<head>
<title>jQuery Hello World</title> <script src = "jquery-1.11.3.js"> </script>
<script type="text/javascript"> $(document).ready(function () {
$("#divID").html("Hello World !"); }); </script> </head>
<body>
<div id="divID"> </div>
</body>
</html>
```

Save file with named Test.html and run it by any browser, you will see the result:

Output:

```
Hello World!
```

Explanation:

“<script src = "jquery-1.11.3.js"</script>” means to use jQuery code in current document.

“$(document).ready (function ( ){ })” means to run the function automatically when the web page is completely loaded.

“$("#divID")” accesses the tag whose id is “divID”.

“html( )” displays the contents.

“$("#divID").html("Hello World !")” displays “Hello World!” in a tag whose id is “divID”.

“ ” defines a tag named “divID” where some contents will be shown.

Congratulation! You have achieved your first target.

Run When Page Loaded

“$(document).ready(function (){ })” means to execute the function when the web page is loaded completely.

Example 1.2

<html>

<head>

<title>JQuery path Testing</title> <script src = "jquery-1.11.3.js"> </script>
</head>

<body>

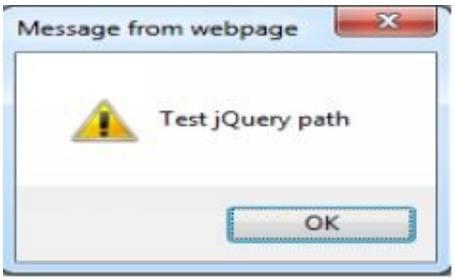
<script type="text/javascript"> $(document).ready(function ( ) {

alert("Test jQuery path"); }); </script> </body>

</html>

Save file with testjquery.html and run it any browser, you will see the alert box with message “Test JQuery path”.

Output:



Explanation:

“<script src = "jquery-1.11.3.js"> </script>” uses jQuery code in current document.

“$(document).ready(function ( ){ })” executes the function when the web page is loaded completely.

What is $(“ ”) meaning?

“$” is a symbol of jQuery.

“$( )” accesses an element in current html document.

For example:

$(“span”) accesses a tag, its tag name is “span”.

$ (“#3”) accesses a tag, its id is “3”.

$(“.clss5”) accesses a tag, its class name is “clss5”.

Show Content (1)

```
$(“tag”).html( )
```

$(“tag”) accesses the “tag”.

.html( ) shows the contents without html symbol.

Example 1.3

<html>

<head>

<title>How to use html()</title> <script src="jquery-1.11.3.js"></script>
<script> $(document).ready(function(){

$("div").html("<h1> Hallo world! </h1>"); }); </script> </head>

<body>

</body>

</html>

Please run the browse, you will see the result.

Output:

Hello World!

Explanation:

“<script src="jquery-1.11.3.js"></script>” uses jQuery code in current document.

“$(document).ready(function(){ }” executes the function when the web page is loaded completely.

“$("div").html("<h1> Hallo world! </h1>")” displays “Hallo world!” in tag “div”.

Note: You cannot see<h1>…</h1>, because **.**html( ) shows contents without html symbol.

Show Content (2)

$(“tag”).text( )

“$(“tag”)” accesses the “tag”

.text( ) shows contents with html symbol.

Example 1.4

```
<html>
<head>
<title>How to use text()</title> <script src="jquery-1.11.3.js"></script>
<script> $(document).ready(function(){
$("div").text("<h1>Hallo world!</h1>"); }); </script> </head>
<body>
<div></div>
</body>
</html>
```

Please run the browse, you will see the result.

Output:

<h1>Hallo world!</h1>

Explanation:

“<script src="jquery-1.11.3.js"> </script>” uses jQuery code in current document.

“$(document).ready(function(){ }” executes the function when the web page is loaded completely.

“$("div").text ("<h1> Hallo world! </h1>")” displays “Hallo world!” in tag “div”.

Note: You can see<h1>…</h1>, because **.**text( ) shows contents with html symbol.

Set CSS Style

$(“selector”).css (“style”)

What is selector?

All HTML elements based on their id, classes, types (text, radio etc.), attributes (id, title, src etc), tag name (div, p, form, table, tr, th, td etc.) etc are jQuery selectors.

$(“selector”) means to access one specified element.

What is css( )?

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

css(“style”) means to set a style for a specified element.

“$(“selector”).css (“style”)” accesses a specified element, and sets a css style for it.

Set CSS by ID

“$(“#id”).css( )” accesses a tag by its id, and sets a css style.

If we want to access any element by its “id”, then we have to use “#” to select that element. css( ) method can set or change any property.

Example 1.5

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/JavaScript">
function changeColor ( ) {
```

```
$ ("#divTest").css ("background-color", "red");
}
```

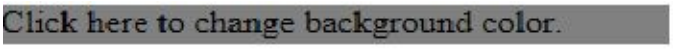
```
</script>
</head>
```

<body>

Click here to change background color.

```
</body>
</html>
```

Original:



Click on text “Click here to change background color”, and then you will see the div background color changed.

Output:

Click here to change background color

Explanation:

“$ (“#divTest”).css (“background-color”, “red”)” accesses a tag whose id is “divTest”, and sets its background color as “red”.

“<div id=”divTest” onclick=”changeColor( )” executes the function “changeColor( )” when clicking a tag whose id is “divTest”.

In above example, you can see I am changing “div” background color by accessing its id ‘**#divTest**”.

Set CSS by Tag

“$(“tag”).css( )” accesses a tag by its name, and sets a css style.

If we want to access any element by its “tag” name, then we have to use “tag” name to select that element, css( ) method can set or change any property.

Example 1.6

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/JavaScript">
function changeColor ( )  {
```

```
$ ("div").css ("background-color", "red");
}
```

```
</script>
</head>
```

<body>

<div onclick="changeColor()" width: 300px; style="cursor: pointer; width: 300px; Height:20px; background-color: #cccccc;">

Click here to change background color.

```
</body>
</html>
```

Original:

Click on text “Click here to change background color”, and then you will see the div background color will be changed.

Output:

Click here to change background color.

Explanation:

“$ (“div”).css (“background-color”, “red”)” accesses a tag whose tag name is “divTest”, and sets its background color as “red”.

“<div onclick=”changeColor()”” executes the function “changeColor( )” when clicking a tag whose tag name is “div”.

In above example, you can see I am changing “div” background color by using tag “**div”**.

Set CSS by Class

“$(“**.**class”)**.** css( )” accesses an element by its class name, and sets a css style.

If we want to access any element by “class” name then we have to use “class name” to select that element, css( ) method can set or change any property.

Example 1.7

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/JavaScript">
function changeColor ( ){
```

```
$(".divClick").css ("background-color", "red");
}
```

</script>
</head>

<body>

Click here to change background color.

```
</body>
</html>
```

Original:

Click on text “Click here to change background color”, and then you will see the div background color will be changed.

Output:

Explanation:

“$ (“.divClick”).css (“background-color”, “red”)” accesses a tag whose class name is “divClick”, and sets its background color as “red”.

“<div class=”divClick” onclick=”changeColor()”” executes the function “changeColor( )” when clicking a tag whose class name is “divClick”.

In above example, you can see I am changing “div” background color by using class “**.divClick”**.

Set CSS by Specified Tag

“$ (“'p:nth-child(n)”). css( )” accesses the “n” tag, and sets a css style.

“$ (“'p:nth-child(1)”). css( )” accesses the first tag, and sets a css style.

“$ (“'p:nth-child(4)”). css( )” accesses the fourth tag, and sets a css style.

Example 1.8

<html>

```
<head>
<script src="jquery-1.11.3.js">  </script>
```

```
<script type="text/javascript">
function setCSS() {
```

```
$('p:nth-child(1)').css("font-style", "italic");
$('p:nth-child(4)').css("font-style", "italic");
```

```
}
</script>
```

</head>
<body>

<h1>Selecting the first and last tag</h1>

I love jQuery 1 time
I love jQuery 2 time

I love jQuery 3 time
I love jQuery 4 time

```
</div>
<input type = "button"  value="Click Me"
```

```
   onclick="setCSS()"> </input>
</body>
```

</html>

Original:

I love jQuery 1 time
I love jQuery 2 time
I love jQuery 3 time
I love jQuery 4 time

After clicking the button “Click me”, you will see the result.

Output:

I love jQuery 1 time I love jQuery 2 time
I love jQuery 3 time
I love jQuery 4 time

Explanation:

“onclick="setCSS()"” executes the function when clicking the button.

“$ (“'p:nth-child(1)”). css( )” accesses the first tag, and sets a css style.

“$ (“'p:nth-child(4)”). css( )” accesses the fourth tag, and sets a css style.

“onclick="setCSS()"” runs the setCSS() when clicking the button.

Set CSS by Index

$(“div”).eq(n).css( ) accesses an element whose index number is “n”, and sets a css style.

The .eq() selector is used to select an element with a specific index number.

Note: The index numbers starts from 0, so the first element will have the index number 0.

Example 1.9

<html>
<head>

<title>JQuery tag.3rd</title>

<script src = "jquery-1.11.3.js"></script>

```
<script type="text/javascript">
function changeColor() {
```

```
$("div").eq(2).css("background-color", "black");
}
```

```
</script>
</head>
```

<body>

<input type="button" value="Click Me" onclick="changeColor()" />

```
</body>
</html>
```

Original:

Click on text “Click Me”, and then you will see the **third div** background color will be changed.

Output:

Explanation:

“$("div").eq(2).css("background-color", "black")” accesses the 3th tag “div”, and sets its background color as black.

“<input type="button" value="Click Me" onclick = "changeColor( )" />” runs “changeColor( )” when clicking the button.

The color of **3st location** div has been changed from green to black.

Set CSS by First Tag

$(“tag**:**first”). css( ) accesses the first element, and sets a css style.

“**:**first” selector is used to select the first element from matched *elements*.

Example 1.10

<html>

<head>
<title>JQuery tag:first</title>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function changeColor() {
$("div:first").css("background-color", "black");
```

```
}
</script>
```

</head>
<body>

```
<div style="float: left; background-color: aqua; width: 150px; height: 120px;">
</div>

<div style="float: left; background-color: maroon; width: 150px; height:
120px;"> </div>
```


<input type="button" value="Click Me" onclick="changeColor()" />
</body>

</html>

Original:

Click Me

Click the button "Click Me", and then you will see the **first div** background color will be changed.

Output:

Explanation:

“$("div**:**first").css("background-color", "black")” accesses the first tag “div”, and sets its background color as black.

“<input type="button" value="Click Me" onclick = "changeColor( )" />” runs “changeColor( )” when clicking the button.

The color of **1st location** div has been changed from aqua to black.

Set CSS by Last Tag

$(“tag**:**last”).css( ) accesses the last element, and sets a css style.

“**:**last” selector is used to select the last tag from matched elements.

Example 1.11

<html>
<head>

<title>JQuery tag:last</title>

<script src="jquery-1.11.3.js"></script>

```
<script type="text/javascript">
function changeColor() {
```

```
$("div:last").css("background-color", "black");
}
```

</script>
</head>

<body>

<input type="button" value="Click Me" onclick="changeColor()" />

```
</body>
</html>
```

Original:

will see the **last div** background color will be changed.

Output:

Explanation:

“$("div**:**last").css("background-color", "black")” accesses the last tag “div”, and sets its background color as black.

“<input type="button" value="Click Me" onclick = "changeColor( )" />” runs “changeColor( )” when clicking the button.

The color of **last location** div has been changed from green to black.

Set CSS by Embed Tag

$("body") selector will get the content inside the "body" tag. $("body div") will get the all div elements inside the “body” tag.

$(“body div tag”). css( ) accesses a “tag” under the “div” inside the “body”, and sets a css style.

Example 1.12

<html>
<head>

```
<title>JQuery tag.3rd</title>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function changeColor() {
```

```
$("body div p").css("background-color", "gray");
}
```

```
</script>
</head>
```

<body>

```
<p align="center" style="float: left; background-color: aqua; width: 150px; height: 120px;"> </p>

<p align="center" style="float: left; background-color: maroon; width: 150px; height: 120px;"></p>
```


<input name="button" type="button" onClick="changeColor()" value="Click Me" />

</html>

Original:

(. see that text color of all “**p**” tag of ‘ d to “**Gray**”.

Output:

Explanation:

“$("body div p”).css("background-color", "gray")” accesses all “p” taps inside the “div” tag which locates inside the “body” tag, and sets the background color as gray.

“<input type="button" value="Click Me" onclick = "changeColor( )" />” runs “changeColor( )” when clicking the button.

Exercises

Access tag, id, class

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>Access tag, id, class</title> </head>

<body>

This id is idName, so become red
 Here has no definition, so no change in color
 This class is className, so become red This tag is p, so become red </body>

<script src="jquery-1.11.3.js" type="text/javascript"></script> **<script type="text/javascript"> $(function(){**

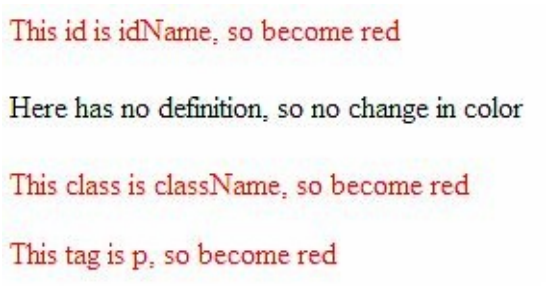
$("p, #idName, .className").css("color","red"); }); </script> </html>

```
<html>
<head>
<title>Access tag, id, class</title>
</head>
<body>
<div id="idName">This id is idName, so become red</div><br>
<div>Here has no definition, so no change in color</div><br>
<div class="className">This class is className, so become red</div>
<p >This tag is p, so become red</p>
</body>
<script src="jquery-1.11.3.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
$("p, #idName, .className").css("color","red");
});
</script>
</html>
```

(Figure 1)

Please save the file with name “Access.html” in the folder with **jquey-1.11.3.js**. Note**:** make sure to use **“.html”** extension name. Double click “Access.html” file, the “Access.html” will be run by a browser, and see the output. (Figure 2)

Output: (Figure 2 as following)



Hour 2
jQuery Function

Add a Class

The addClass() method is used to adds class to every matching element.

Example 2.1

<html>

<head>

<script type="text/javascript"

src="jquery-1.11.3.js">
</script>

```
<script type="text/javascript">
function myFunction(){
```

```
$('p').addClass("redClass");
}
```

</script>

<style>

```
.redClass {
color: red; font-style: italic;
```

```
}
</style>
```

</head>
<body>

PHP in 8 Hours
JQuery in 8 Hours

JavaScript in 8 Hours

```
<form>
<input type = "button" value="Add"
```

```
onclick="myFunction()"></input>
</form>
```

```
</body>
</html>
```

Original:

PHP in 8 Hours

JQuery in 8 Hours

JavaScript in 8 Hours

Click on button “Add” and see the result

Output:

PHP in 8 Hours JQuery in 8 Hours JavaScript in 8 Hours

Explanation

$('p').addClass("redClass"); accesses “p” tag, adds a class “redClass” style to specified text.

“onclick="myFunction()"” executes the “myFunction( )” when clicking the button.

Before clicking the link there is no color in the text, but when you click “Add”, the text will appear red.

Add & Remove Class

The toggleClass( ) method is used to add or remove class for every element in the matching selected elements.

If the specified class is missing, the class will be added and if class is added, the class will be removed.

Example 2.2

<html>
<head>

```
<script type="text/javascript"
src="jquery-1.11.3.js">
```

</script>

<script type="text/javascript">

```
function myFunction(){
$('p').toggleClass("redClass");
```

```
}
</script>
```

<style>
.redClass {

```
color: red; font-style: italic;
}
```

</style>
</head>

<body>
PHP in 8 Hours

JQuery in 8 Hours
JavaScript in 8 Hours

<form>

```
<input type = "button" value="Toggle"
onclick="myFunction()"></input>
```

```
</form>
</body>
```

</html>

Original:

PHP in 8 Hours

JQuery in 8 Hours

JavaScript in 8 Hours

Click on button “Toggle” and see the result

Output:

PHP in 8 Hours JQuery in 8 Hours JavaScript in 8 Hours

Explanation:

$('p').toggleClass("redClass"); accesses “p” tag, adds a class “redClass” when the class “redClass” is missing, or removes the class “redClass” when the class “redClass” is existing.

“onclick="myFunction()"” executes the “myFunction( )” when clicking the button.

Before clicking the link there is no color in the text, but when you click “Toggle”, the text will appear red. If you click again, the color of text will disappear again.

The Size of Element

The size( ) method returns the number of elements matched.

Example 2.3

```
<html>
<head>
<title>jQuery Size Method </title> <script src="jquery-1.11.3.js"></script>
<script type="text/javascript"> function getSize() {
alert("the size of li is: "+$("li").size()); }
</script> </head>
<body>
<input type="button" onclick="getSize()" value="Click here  to check li" />
<ul>
<li>Home</li>
<li>Product</li> <li>About Us</li> </ul>
</body>
</html>
```

Original:

› check li" button and see the result.

Output:

Explanation:

“$("li").size());” accesses the tag “li”, and counts how many tags “li”.

onclick="getSize()" executes the “getSize( )” when clicking the “Click here to check li”.

Show Hidden Elements

The show( ) method is used to show the hidden elements which are selected.

Example 2.4

<html>
<head>

<title>jQuery Show Method </title>

<script src="jquery-1.11.3.js"></script>

```
<script type="text/javascript">
function showText() {
```

```
$("#divMsg").show();
}
```

```
</script>
</head>
```

<body>

Click here.

This is Testing.....

</body>

</html>

Original:

Click here

Click on “Click here” link and see the result.

Output:

Click here
This is Testing……

Explanation:

$("#divMsg").show() accesses the tag whose id is “divMsg”, and shows its hidden contents.

onclick="showText()" executes the “showtext( )” when clicking “Click here”.

Hide Selected Elements

The hide( ) is used to hide the elements which are selected.

Example 2.5

<html>

<head>
<title>jQuery Hide Method </title>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function hideText() {
$("#divMsg").hide();
```

```
}
</script>
```

</head>
<body>

Click here

This is Testing...

```
</body>
</html>
```

Original:

Click here This is Testing……

Click on “Click here” link and see the result.

Output:

Click here

Explanation:

“$("#divMsg").hide()” accesses the tag whose id is “divMsg” and hides its selected text.

“onclick="hideText()"” executes the “hideText” when clicking “Click here”.

After clicking “Click here”, the div has been disappeared.

Hide & Show Elements

“toggle( )” switches between hide( ) and show( ) for the matching elements.

Example 2.6

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function toggleEffect() {
```

```
$("#divMsg").toggle();
}
```

```
</script>
</head>
```

<body>
 Click here

This is Testing...

```
</body>
</html>
```

Original:

Click here
This is Testing……

Click on “Click here” link and see the result. If selected element is visible then it will be hidden, and if it is hidden then it will be shown.

Output:

Click here

Note: If you click “Click here” again, it will show This is Testing again.

Explanation:

“$("#divMsg").toggle()” accesses the tag whose id is “divMsg”, shows the hidden text or hides the shown text.

“onclick="toggleEffect()"” executes the “toggleEffect( )” when clicking “Click here”.

Slide Element Up

The slideUp() method is used to slide the matched element up and hide it.

Example 2.7

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function slideUp() {
$("#divMsg").slideUp();
```

```
}
</script>
```

</head>
<body>

```
<div onclick="slideUp()" style="cursor: pointer;">
Click here</div>
```

<div id="divMsg" style="background-color: green; color: white; margin-top: 10px;

width: 120px; height: 150px">

This is Tesing...
</body>

</html>

Original:

Click here
This is Testing……

Click on “Click here” link and see the result.

Output:

Click here

Note: when you click on "Click here", the This is Testing…… smoothly will slide from down to up and be hidden.

Explanation:

“$("#divMsg").slideUp()” accesses the tag whose id is “divMsg”, and slowly hides the text from down to up.

“onclick="slideUp()"” executes the “slideUp( )” when clicking “Click here”.

Slide Element Down

The slideDown() method is used to slide the matched element down and show it.

Example 2.8

```
<html>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function slideDown() {
```

```
$("#divMsg").slideDown();
}
```

</script>
</head>

<body>

Click here

This is Tesing...
</body>

</html>

Original:

Click here

Click on “Click here” link and see the result.

Output:

N click on “Click here”, the div smoothly will slide from up to down and be shown.

Explanation:

“$("#divMsg").slideDown()” accesses the tag whose id is “divMsg”, and slowly shows the text from up to down.

“onclick="slideDown()"” executes the “slideDown( )” when clicking “Click here”.

Insert Before Element

insertBefore() and before() methods are used for same task, insert a text or html content before the matching elements. Their main difference is only in the syntax.

Example 2.9

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function insertBeforeHtml() {
$("<div style='color:green;'>This text is inserted by insertBefore() method.
</div>").insertBefore($("#divInsertBefore"));
```

```
$("#divInsertBefore").before("<div style='color:gray;'>This text is inserted by before() method.</div>");
}
```

```
</script>
</head>
```

<body>

Click here

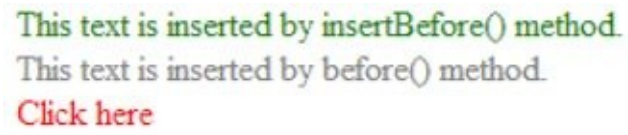
</body>

</html>

Original:

Click here Click on “Click here” link and see the result.

Output:


This text is inserted by insertBefore() method.
This text is inserted by before() method.
Click here

Explanation:

“.**insertBefore**($("#divInsertBefore"))” inserts a text before the tag whose id is “divInsertBefore”.

“$("#divInsertBefore").**before”** inserts a text before the tag whose id is “divInsertBefore”.

Their main difference is only in the syntax.

Insert After Element

The insertAfter() and after() methods is used for the same task, insert a text or html content after the matching elements. Their major difference is in the syntax.

Example 2.10

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function insertAfterHtml() {
```

```
$("<div style='color:green;'>This text is inserted by insertAfter() method.
</div>").insertAfter($("#divInsertAfter"));

$("#divInsertAfter").after("<div style='color:gray;'>This text is inserted by
after() method.</div>");
```

```
}
</script>
```

</head>
<body>

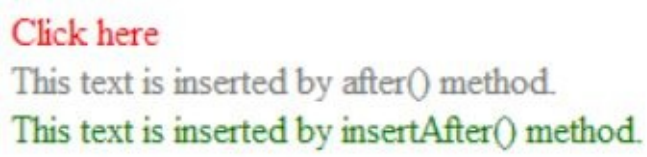
Click here

```
</body>
</html>
```

Original:

Click here Click on “Click here” link and see the result.

Output:


Click here
This text is inserted by after() method.
This text is inserted by insertAfter() method.

Explanation:

“**.insertAfter**($("#divInsertAfter"));” inserts a text after the tag whose id is “divInsertAfter”.

“$("#divInsertAfter")**.after**” inserts a text after the tag whose id is “divInsertAfter”.

Their main difference is only in the syntax.

Check Element Condition

“is( )” tests an elements to see whether it matches given condition. It returns true or false.

Example 2.11

```
<html>
<head>
<script src = "jquery-1.11.3.js"> </script> <script type="text/javascript">
function checkGender(){
if($("#chkMale").is(":checked")) {
alert("Male is selected"); }
else if(
$("#chkFemale").is(":checked")) {
alert("Female is  selected"); }}
</script> </head>
<body>
<div id="chkGender" onclick="checkGender()" style="cursor: pointer; color:
red;"> Click here</div>
<input type="radio" id="chkMale" name="sex" value="male">Male<br> <input
type="radio" id="chkFemale" name="sex" value="female">Female </body>
</html>
```

Original:

Click here ○Male ○Female Click on “Click here” link and see the result.

Output:


Click here
Male
Female
Message from webpage
Female is selected
OK

Explanation:

$("#chkMale").is(":checked") tests “#chkMale” tag to see if it is checked.

$("#chkFemale").is(":checked") tests “#chkFemale” tag to see if it is checked.

“onclick="checkGender()” executes “checkGender” function when clicking “Click here”.

How Many Elements?

$(“”).length counts the number of specified elements.

The length property contains the number of elements in the jQuery object.

Example 2.12

```
<html>
```

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function checkLength() {
```

```
alert("The Number of divs are: " + $("div").length);
}
```

```
</script>
</head>
```

<body>

 Click here

This is first div

This is 2nd div

This is 3rd div

This is 4th div

```
</body>
</html>
```

Original:

Click here This is first div
This is 2nd div
This is 3rd div
This is 4th div

Click on “Click here” link and see the result.

Output:


Click here
This is first div
This is 2nd div
This is 3rd div
This is 4th div
Message from webpage
The Number of divs are: 4
OK

Explanation:

“$("div").length” accesses the “div” tag, and counts how many elements inside the “div”.

Exercises

replaceWith( )

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>replaceWith</title> </head>

<script src="jquery-1.11.3.js" type="text/javascript"></script> **<script type="text/javascript"> function replaceWith(){**

$("#replaceWith").replaceWith("Hello, this is a new text!"); }; </script>
<body>

 This text will be replaced with a new text

<input type="button" value="Relace" onclick="replaceWith()">

</body>

</html>

```
<html>
<head>
<title>replaceWith</title>
</head>
<script src="jquery-1.11.3.js" type="text/javascript"></script>
<script type="text/javascript">
function replaceWith(){
$("#replaceWith").replaceWith("Hello, this is a new text!");
};
</script>
<body>
<p id="replaceWith">
This text will be replaced with a new text</p>
<p>
<input type="button" value="Relace" onclick="replaceWith()">
</p>
</body>
</html>
```

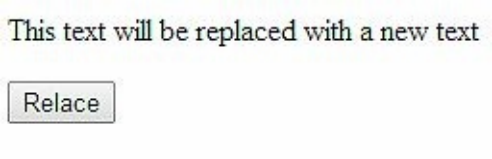
(Figure 1)

Please save the file with name “ReplaceWith.html”.

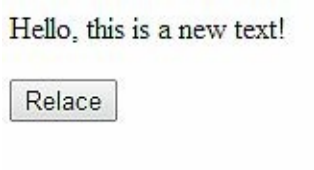
Note**:** make sure to use **“.html”** extension name.

Double click “ReplaceWith.html” file, the “ReplaceWith.html” will be run by a browser, click the button, and see the output. (Figure 2)

Original:



Output:



(Figure 2)

Explanation:

replaceWith( ) is used to replace an existing text with a new text.

Hour 3

Elements Selection

$(document).ready( )

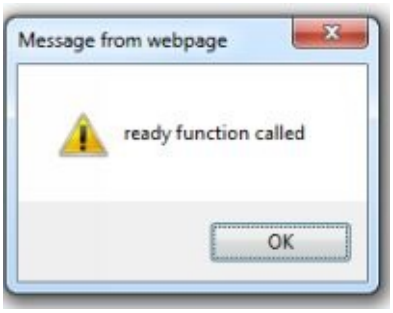
“$(document).ready( )” is used to specify a function to execute after loading web page.

The event occurs after web page is loaded.

Example 3.1

```
<html>
<head>
<title>This is Test Page </title> <script src="jquery-1.11.3.js"></script> <script type="text/javascript"> $(document).ready(function( ){
alert("ready function called"); }); </script> </head>
<body>
<p>Example:</p> <p>$(document).ready(function( ){</p> <p> .....
                                                    });
</p>
</body>
</html>
```

Output:



Explanation:

“$(document).ready(function( ))” executes the function automatically when the web page is loaded.

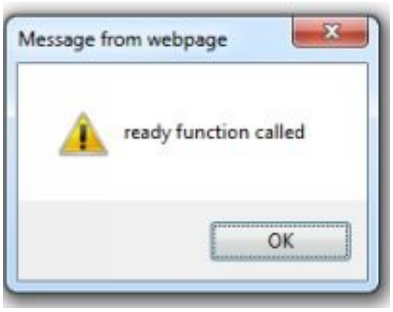
$ (function ( ) { })

$ (function ( ) { }) does the same task as $ (document).ready ( ) function does.

Example 3.2

```
<html>
<head>
<title>This is Test Page </title> <script src="jquery-1.11.3.js"></script> <script type="text/javascript"> $(function ( ) {
alert("ready function called"); }); </script> </head>
<body>
   ......
</body>
</html>
```

Output:



Explanation:

“$(function ( ) { }” runs like “$(document).ready( )”. After the web page is loaded, the “$(function ( ) { }” will run automatically.

$(“div tag”)

“$(‘div tag’) can access all top level child elements and all sub level child elements inside div tag by tag name.

Example 3.3

```
<Html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function getchild() {
alert($("p span").length); }
</script> </head>
<body>
<div onclick="getchild()" style="cursor: pointer; width: 300px; height: 20px;
color: red;"> Click here
</div>
<p>
<span style="float: left; width: 100%; padding: 5px;">This is top level child
</span> <span style="float: left; width: 100%; padding: 5px;">This is top level
child </span> <span style="float: left; width: 100%; padding: 5px;">This is top
level child <span style="float: left; width: 100%; padding: 5px;">This is sub
level child </span> </span>
</p>
</body>
</html>
```

Original:

Click here This is top level child

This is top level child

This is top level child

  This is sub level child

Click on “Click here” link and see the result.

Output:


Click here
This is top level child
This is top level child
This is top level child
This is sub level child
Message from webp...
4
OK

Explanation:

“**($("p span").length**” accesses all top level child elements “span” and sub level child elements “span” inside the “p” tag, and counts the number of all “span” elements.

$(“element”).length counts the number of elements.

In above example, you can see the result is 4. (length of all span elements).

$(“div>tag”)

“$(‘div>tag’) can access all top level child elements inside div tag. But not including the sub level child elements.

Example 3.4

<Html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function getchild() {
alert($("p>span").length);
```

```
}
</script>
```

</head>
<body>

Click here

<p>

This is top level child

This is top level child

This is top level child

This is sub level child

</p>

```
</body>
</html>
```

Original:

Click here This is top level child

This is top level child

This is top level child

This is sub level child

Click on “Click here” link and see the result.

Output:

Explanation:

“**$("p>span").length**” accesses all top level child elements “span” inside the “p” tag, but not including all sub level child elements “span”, and counts the number of all top level child elements.

$(“element”).length counts the number of elements.

In above example, you can see the result is 3 (length of all top level child “span” element).

$(“div tag”) & $(“div>tag”)

$(“div tag”) will get the **all top & sub-level child elements** inside the “div” tag, while $(“div>tag”) will only get the **all top-level elements**, not including the **sub-level elements.** Let’s see this with example.

Example 3.5

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function getchild() {
alert( $("p  span").length );
```

```
alert( $("p>span").length );
}
```

</script>

</head>
<body>

Click here

<p>

This is top level child

This is top level child

This is top level child

This is sub level child

</p>

```
</body>
</html>
```

Original:

Click here

This is top level child

This is top level child

This is top level child

  This is sub level child

Click on “Click here” link and see the result.

Output:

Explanation:

“$(“div tag”)” accesses both top level and sub level child elements.

“$(“div>tag”)” accesses only top level child elements, not includes the sub level child elements.

In the above example, you can see the first count is 4 for all top and sub level child elements, and second count is 3 for all top-level child elements only.

$(“tag:contains(text)”)

“$(“tag:contains(text)”)” accesses a tag that contains a specified “text”.

Example 3.6

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function checkContains() {
alert( $("span:contains('second')").html( ) ); }
</script> </head>
<body>
<div onclick="checkContains()" style="cursor: pointer; width: 300px; height:
20px; color: red;"> Click here
</div>
<span style="float: left; width: 100%; padding: 5px;">This is first span </span>
<span style="float: left; width: 100%; padding: 5px;">This is second span
</span> </body>
</html>
```

Original:

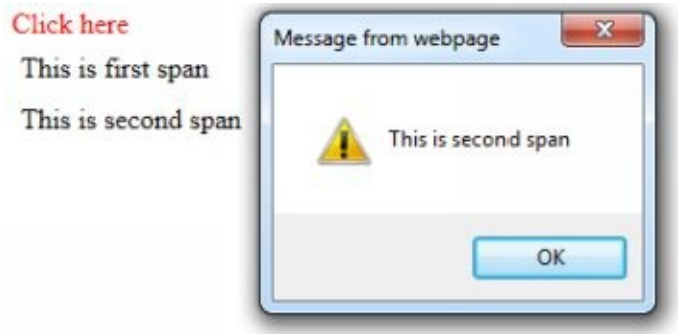
Click here

This is first span

This is second span

Click on “Click here” link and see the result.

Output:


Click here
This is first span
This is second span
Message from webpage
This is second span
OK

Explanation:

$("span:contains('second')") accesses the “span” that contains a text “second”.

“onclick="checkContains()"” runs the checkContains( ) when clicking the “Click here”.

$(“tag [attribute]”)

$(“tag [attribute]”) can access a tag with a specified attribute.

Example 3.7

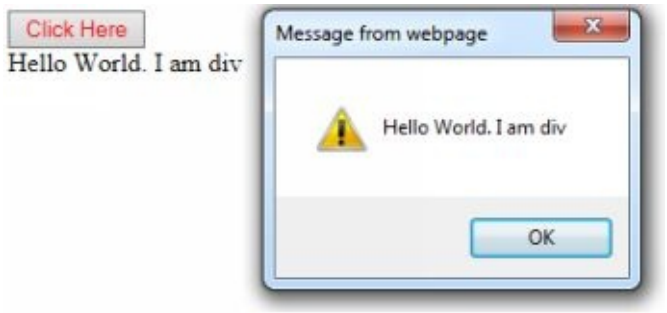
```
<html>
<head>
<title>jQuery tag attribute</title> <script src="jquery-1.11.3.js"></script>
<script type="text/javascript"> function getTitle() {
alert( $("div[ title ]").html() ); }
</script> </head>
<body>
<input onclick="getTitle( )" type="button" value="Click Here" style=" color: red;" /> <div title="This is title" style="float:left; width:100%">Hello World. I am div</div> </body>
</html>
```

Original:

Click Here

Hello World. I am div Click on “Click here” link and see the result.

Output:


Click Here
Hello World. I am div
Message from webpage
Hello World. I am div
OK

Explanation:

“**$("div[title]").**html( )” accesses the tag “div” with an attribute “title”, and displays its contents.

“onclick="getTitle( )” calls the function “getTitle( )” when clicking the button.

$(“tag [attribute=value]”)

“$(“tag [attribute=value]”)” accesses a tag by “attribute=value”.

Example 3.8

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function getHtmlByTitle () {
alert( $("div[title='2nd']" ).html() );
```

```
}
</script>
```

</head>
<body>

```
<input onclick="getHtmlByTitle()" type="button" value="Click Here"
style="color: red;" />

<div title="first" style="float: left; width: 100%">
```

This is first div

This is 2nd div

</body>

</html>

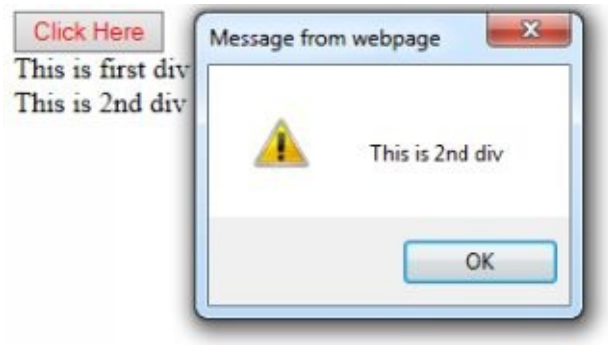
Original:

Click Here

This is first div This is 2nd div

Click on “Click here” link and see the result.

Output:


Click Here
This is first div
This is 2nd div
Message from webpage
This is 2nd div
OK

Explanation:

“$("div[title='2nd']").html()” accesses the tag “div” by “title=2nd” and displays its contents.

“onclick="getHtmlByTitle()"” calls the function “getHtmlByTitle( )” when clicking the button.

$(“#id”).is(‘tag’)

$(“#id”).is(‘tag’) tests an element with a specified id to see if it is a “tag” element. It will return true if the element is a “tag” element. Otherwise, it will return false.

Example 3.9

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function checkElementByTag() {
alert($("#divHello").is("div"));
```

```
}
</script>
```

</head>
<body>

<input onclick="checkElementByTag( )" type="button" value="Click Here" style="color: red;" />

Hello world

</body>

</html>

Original:

Click Here

Hello world Click on “Click here” link and see the result.

Output:

Explanation:

“$("#divHello").is("div");” returns true if the element with ID “#divHello” is a element.

“onclick="checkElementByTag( )” calls the function “checkElementByTag( )” when clicking the button.

$(“tag:eq(3)”)

$(“tag:eq(3)”) accesses a tag with an index number “3”.

:eq() is used to select an element with a specific index number.

Note: index number begins from 0.

Example 3.10

```
<html>
<head>
<title>Jquery tag:eq(3)</title> <script src="jquery-1.11.3.js"></script> <script type="text/javascript"> function getElementByIndex() {
alert($("div:eq(3)").html()); }
</script>
</head>
<body>
<input onclick="getElementByIndex()" type="button" value="Click Here" style="color: red;" /> <div style="float: left; width: 100%"> This is first div</div> <div style="float: left; width: 100%"> This is 2nd div</div> <div style="float: left; width: 100%"> This is 3rd div</div> <div style="float: left; width: 100%"> This is 4th div</div> <div style="float: left; width: 100%"> This is 5th div</div> </body>
</html>
```

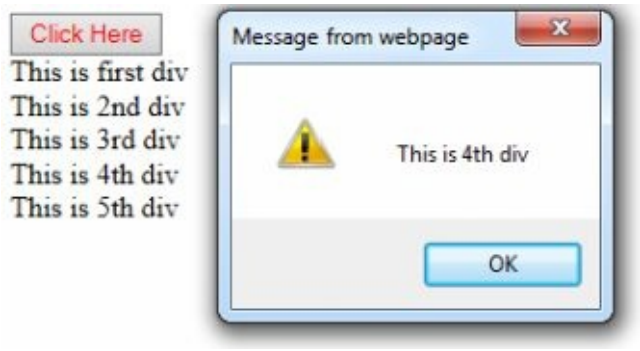
Original:

Click Here

This is first div
This is 2nd div This is 3rd div
This is 4th div
This is 5th div

Click on “Click here” link and see the result.

Output:


Click Here
This is first div
This is 2nd div
This is 3rd div
This is 4th div
This is 5th div
Message from webpage
This is 4th div
OK

Explanation:

“$”(div:eq(3)").html()” accesses a tag “div” at the position of index number “3”, and shows its contents.

Note: index number begins from 0.

$(“input:checked”)

“$(“input:checked”)” accesses an “input” tag that has been checked.

The “**:**checked” selector usually works for radio buttons, checkboxes, and select elements.

Example 3.11

<html>
<head>

<title>This is Test Page </title>

<script src="jquery-1.11.3.js"></script>

```
<script type="text/JavaScript">
function getCheckedCount() {
```

```
alert($("input:checked").length);
}
```

</script>
</head>

<body>

<input onclick="getCheckedCount( )" type="button" value="Click Here" style="color: red;" />

```
<input type="checkbox" value="Apple" checked="checked" />Apple
<input type="checkbox" value="Banana" />Banana
```

<input type="checkbox" value="Peach" checked="checked" />Peach

<input type="checkbox" value="Mango" />Mango

```
</body>
</html>
```

Original:

Click on “Click here” link and see the result.

Output:

Explanation:

“($("input:checked").length)” accesses an input tag which has been checked, and count how many input tag has been checked.

“onclick="getCheckedCount( )"” calls the function “getCheckedCount( )” when clicking the button.

$(“select option:selected ”)

“$(“select option**:**selected ”)” accesses a “select” tag with a selected option.

Example 3.12

```
<html>
<head>
<title>Jquery selected option</title> <script src="jquery-1.11.3.js"></script>
<script type="text/javascript"> function getSelectedValue() {
alert($("select option:selected").text()); }
</script> </head>
<body>
<input onclick="getSelectedValue()" type="button" value="Click Here"
style="color: red;" /> <select>
<option>First</option> <option selected>second</option>
<option>third</option> </select>
</body>
</html>
```

Original:

Click on Click Here and see the result. text( ) shows the contents.

Output:

Explanation:

“$("select option:selected").text( )” accesses a “select” tag whose option has been selected, and displays its contents.

onclick="getSelectedValue()" calls the function “getSelectedValue( )” when clicking the button.

Exercises

filter( )

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>filter</title> </head>

<body>

<h3>

0

1 2 3 4 5 </h3> </body>

<script src="jquery-1.11.3.js"></script> **<script type="text/javascript">**
$(function(){

$("p").filter(".study").css("color","red"); }); </script> </html>

```
</head>
<body>
<h3><center>
<p>0</p>
<p id="study">1</p>
<p class="study">2</p>
<p class="study">3</p>
<p class="study">4</p>
<p id="study">5</p>
</h3></center>
</body>
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
$(function(){
$("p").filter(".study").css("color","red");
});
</script>
</html>
```

(Figure 1)

Please save the file with name “Filter.html”.

Note**:** make sure to use **“.html”** extension name.

Double click “Filter.html” file, the “Filter.html” will be run by a browser, and see the output. (Figure 2)

Original:	Output:
0	0
1	1
2	2
3	3
4	4
5	5

(Figure 2)

Explanation:

filter( ) is used to get specified element.

Hour 4

More Function

attr( )

The attr() method is used to set the attributes. For example: If you want to set an attributes value by using the attr( ) function, you can write the code like this: $(“img”).attr(“alt” , ”This is an image.”) .

If you want to disabled an element, then you can write the code like this: $(“#elementID”).attr(“disabled” , ”disabled”).

Example 4.1

```
<html>
<head>
<title>Jquery selected option</title> <script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript"> function setAttr() {
$("input:eq(1)").attr("disabled", "disabled"); }
</script> </head>
<body>
<input type="button" onclick="setAttr( )" value="Click Here" style="color:
red;" /> <input type="text" value="Hello World" /> </body>
</html>
```

Original:

Click on “Click here” link and see the result.

Output:

In the above example, the Hello World text has been disabled.

Explanation:

“$("input:eq(1)") accesses the “input” tag whose index number is 1.

“.attr("disabled", "disabled");” sets the “disabled” property’s value is disable.

“onclick="setAttr( )"” calls the function “setAttr( )” when clicking the button.

Note: The index number begins from zero.

html( )

The html( ) method is used to get the HTML contents of the element or set the HTML contents of every matched element.

Example 4.2

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
function setGetHtml() {
alert("Before Changing content is: " + $("#divContent").html());
```

```
$("#divContent").html("Html is changed");
alert("After Changing content is: " + $("#divContent").html());
```

```
}
</script>
```

</head>
<body>

```
<input onclick="setGetHtml()" type="button" value="Click Here" style="color: red;" />

<div id="divContent">Hello World</div>
```

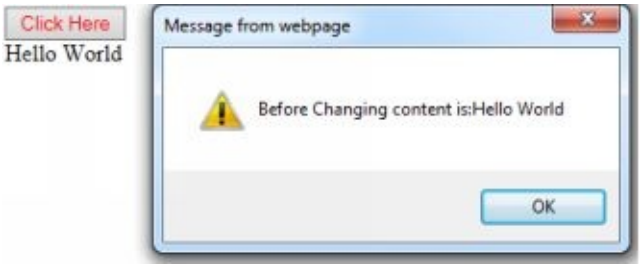
```
</body>
</html>
```

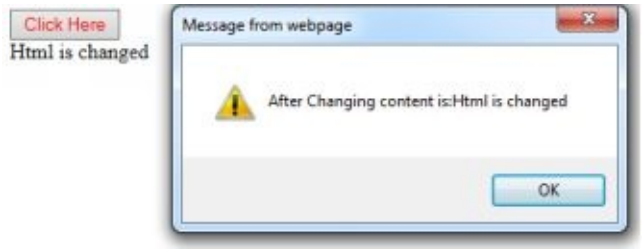
Original:

Hello World

Click on “Click here” button and see the result.

Output:


Click Here
Hello World
Message from webpage
Before Changing content is:Hello World
OK


Click Here
Html is changed
Message from webpage
After Changing content is:Html is changed
OK

Explanation:

When calling the function “setGetHtml( )”, three following commands will be executed.

(1) “$("#divContent").html( ));” gets the contents from “#divContent”.

(2) “$("#divContent").html("Html is changed");” sets the contents “Html is changed” to “divContent”.

(3) “$("#divContent").html( ));” gets the contents from “#divContent”.

In the above example, you can see the “Hello World” text at first, but when you click “Click Here”, the text “**Hello World**” has been changed into “**Html is changed**”, and two alert messages are shown respectively.

text( )

The text() method is used to get the text of the element or set the text of every matched element.

Example 4.3

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
function setGetText() {
alert("Before Changing text is: " + $("#divContent").text());
```

```
$("#divContent").text("Hello World text is changed");
alert("After Changing text is: " + $("#divContent").text());
```

```
}
</script>
```

</head>
<body>

```
<input onclick="setGetText()" type="button" value="Click Here" style="color: red;" />

<div id="divContent">Hello World I am text</div>
```

```
</body>
</html>
```

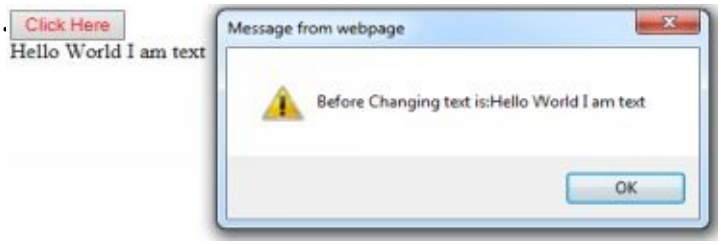
Original:

Click Here

Hello World I am text

Click on “Click here” link and see the result

Output:


Click Here
Hello World I am text
Message from webpage
Before Changing text is:Hello World I am text
OK


Click Here
Hello World text is changed
Message from webpage
After Changing text is:Hello World text is changed
OK

Explanation:

When calling the function “setGetText( )”, three following commands will be executed.

(1) “$("#divContent").text( ));” gets the contents from “#divContent”.

(2) “$("#divContent").text("Hello World text is changed ");” sets the contents “Hello World text is changed” to “divContent”.

(3) “$("#divContent").text( ));” gets the contents from “#divContent”.

In the above example you can see the “**Hello World I am text**” at first, but when you click “Click Here”, the text “Hello World I am text” has been changed into “**Hello World text is changed**”, and two alert messages are shown respectively.

append( )

append( ) method is used to add the value at the end of the specified element.

Example 4.4

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
function appendHtml() {
$("#divContent").append("</br> This is new content");
```

```
}
</script>
```

</head>
<body>

```
<input onclick="appendHtml()" type="button" value="Click Here" style="color: red;" />

<div id="divContent">Hello World</div>
```

```
</body>
</html>
```

Original:

Click on “Click here” button and see the result

Output:

Explanation:

“$("#divContent").append("</br> This is new content");” accesses the tag “#divContent”, and appends “This is new content” text to the end of the original text.

In the above example, you can see the “**Hello World**” text at first, but when you click “Click Here”, the “**This is new content**” text has been appended at the end of the original text “Hello World”.

width( )

The width() method is used to get the width of an element.

Example 4.5

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
function getWidth() {
alert("The width for the div is " + $("#divContent").width() + "px.");
```

```
}
</script>
```

<body>

<input onclick="getWidth()" type="button" value="Get Width" style="color: red;" />

Hello World

</body>

</html>

Original:

Get Width

Hello World

Click on “Get Width” link and see the result.

Output:


Get Width
Hello World
Microsoft Internet Explorer
The width for the div is 200px.
OK

Explanation:

“$("#divContent").width()” accesses the “#divContent” tag, and get its width( ).

height( )

The height() function is used to get the height of an element.

Example 4.6

```
<html>
```

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/JavaScript">
function getHeight() {
```

```
alert("The height for the div is " + $("#divContent").height() + "px.");
}
```

```
</script>
</head>
```

<body>

<input onclick="getHeight()" type="button" value="Get Height" style="color: red;" />

Hello World

</body>

</html>

Original:

Get Height

Hello World

Click on “Get Height” link and see the result.

Output:


Get Height
Hello World
Microsoft Internet Explorer
The height for the div is 60px.
OK

Explanation:

“$("#divContent").height()” accesses the tag “#divContent”, and returns the height of the element.

before( )

The before( ) method inserts specified content before the selected elements.

Example 4.7

```
<html>
```

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/JavaScript">
function insertContentBefore() {
```

```
$("#divContent").before(" <div>This is new text</div>");
}
```

```
</script>
</head>
```

<body>

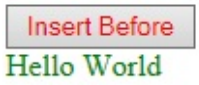
<input onclick="insertContentBefore()" type="button" value="Insert Before" style="color: red;" />

```
<div id="divContent" style="width: 200px; height: 20px; color: Green;">
Hello World
```

</body>

</html>

Original:



Hello World

Output:

Explanation:

$("#divContent").before(" This is new text ") accesses the tag “#divConent”, and inserts the text “This is new text” before the element.

after( )

The after( ) method inserts specified content after the selected elements.

Example 4.8

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/JavaScript">
function insertContentAfter() {
$("#divContent").after(" <div>This is new text</div>"); }
</script> </head>
<body>
<input onclick="insertContentAfter()" type="button" value="Insert After"
style="color: red;" /> <div id="divContent" style="width: 200px; height: 20px;
color: Green;"> Hello World
</div>
</body>
</html>
```

Original:

Output:

Explanation:

“$("#divContent").after(" This is new text ");” accesses the tag “#divContent”, and inserts the text “This is new text” after the element.

val( )

The val( ) method gets or sets the attribute value of the selected elements.

Example 4.9

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
function getVal() {
alert($("#ddlGender").val());
```

```
}
</script>
```

</head>
<body>

```
<input onclick="getVal()" type="button" value="Get Val" style="color: red;" />
<select id="ddlGender">
```

<option value="Male">Male</option>
<option value="Female">Female</option>

</select>
</body>

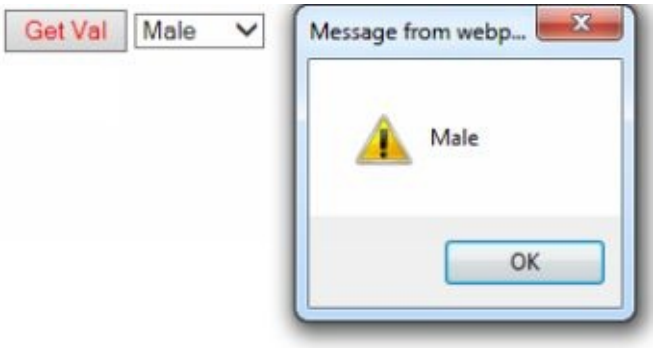
</html>

Original:

Get Val	Male v

Select “Male v”, and click on “Get Val” button.

Output:



Explanation:

Select “Male v”, and click on “Get Val” button.

“$("#ddlGender").val( );” accesses the element “#ddlGender”, and return its option value.

Exercises

slice(start, end-1)

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>Slice()</title> </head>

<body><h3> 0

1

2

3

4

5

</h3> </body>

<script src="jquery-1.11.3.js"></script> **<script type="text/javascript">**
$(function(){

$("p").slice(2,5).css("color","red"); }); </script> </html>

```
<html>
<head>
<title>Slice()</title>
</head>
<body><h3><center>
<p>0</p>
<p>1</p>
<p>2</p>
<p>3</p>
<p>4</p>
<p>5</p>
</center></h3>
</body>
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
$(function(){
$("p").slice(2,5).css("color","red");
});
</script>
</html>
```

(Figure 1)

Please save the file with name “Slice.html”.

Note**:** make sure to use **“.html”** extension name.

Double click “Slice.html” file, the “Slice.html” will be run by a browser, and see the output. (Figure 2)

Original:	Output:
0	0
1	1
2	2
3	3
4	4
5	5

(Figure 2)

Explanation:

slice(start, end-1) is used to get specified element from start index to end-1 index.

Note: index starts with zero.

Hour 5

Event

bind(event, function(event){ })

The bind( ) method is used to bind the event and runs the function.

An event represents the precise action when something happens. *E.g.* click, key up, mouse over represent three events. Moving a mouse over an element, selecting a radio button and clicking on an element are example of event.

Example 5.1

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
$(document).ready(function () {
$("#btn").bind("click", function () {
```

```
alert("User clicked on button.");
});
```

```
});
</script>
```

<body>
<input id="btn" type="button" value="Click Here" style="color: red;" />

```
</body>
</html>
```

Original:

Click Here

Click on “Get Here” button and see the result.

Output:

Explanation:

“$("#btn").bind("click", function () { }” runs the function when clicking the button “#btn”.

one(event, function(event){ })

The one( ) method is used to bind an event and runs the function only once.

Example 5.2

```
<html>
```

```
<head>
<script src="jquery-1.11.3.js"></script>
```

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

```
$("#btnBind").one("click", function () {
alert("This will be displayed only once.");
```

```
});
});
```

```
</script>
</head>
```

<body>

<input id="btnBind" type="button" value="One Time Bind Event" style="color: red;" />

```
</body>
</html>
```

Original:

One Time Bind Event

Click on “One Time Bind Event” button and see the result.

Output:


One Time Bind Event
Message from webpage
This will be displayed only once.
OK

Explanation:

“$("#btnBind").one("click", function () { }” runs the function only once when clicking the button “#btnBind”.

event(function( ){ })

“event(function( ){ })” executes the function when an event occurs. “event” means click, mouseOver, keyDown……

Example 5.3

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
$(document).ready(function () {
$("#btnClick").click(function () {
```

```
alert("I am Click Event");
});
```

```
$("#btnMouseOver").mouseover(function () {
alert("I am Mouse Over Event");
```

```
});
$("#btnMouseLeave").mouseleave(function () {
```

```
alert("I am Mouse Leave Event");
});
```

```
});
</script>
```

</head>
<body>

```
<input id="btnClick" type="button" value="Click Event" style="color: gray;" />
<input id="btnMouseOver" type="button" value="Mouse Over Event"
style="color: gray;" />
```

<input id="btnMouseLeave" type="button" value="Mouse Leave Event" style="color: gray;" />

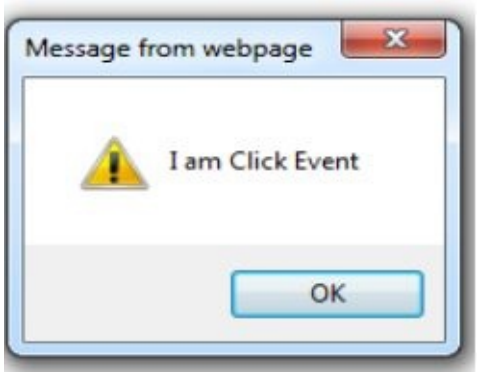
</body>

</html>

Original:

Take action on “Click Event”, “Mouse Over Event” and “Mouse Leave Event” buttons respectively and see the result.

Output:



Explanation:

“$("#btnClick").click(function ()” runs the function when clicking the button “#btnClick”.

“$("#btnMouseOver").mouseover(function ()” runs the function when mouse over the button.

“$("#btnMouseLeave").mouseleave(function ()” runs the function when mouse leave the button.

event.screenX & event.screenY

We can use “event.screenX” and “event.screenY” to get the coordinates of the mouse pointer, relative to the screen, when the mouse is clicked on an element.

Example 5.4

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/JavaScript">
function getCordinates() {
var x = event.screenX; var y = event.screenY; alert("X coords: " + x + ", Y
coords: " + y); }
</script> </head>
<body onmousedown="getCordinates()"><br><br> <center><h3>Please click
anywhere in web page.</h3></center> </body>
</html>
```

Original:

Please click anywhere in web page.

Output:


Please click anywhere in web page.
Microsoft Internet Explorer
X coords: 473, Y coords: 299
OK

Explanation:

“event.screenX;” returns the x coordinate relative to the screen.

“event.screenY;” returns the y coordinate relative to the screen.

event.pageX & event.pageY

The “event.pageX” and “event.pageY” properties used to get the position of the mouse pointer, relative to the left and top edge of the document respectively.

Example 5.5

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/JavaScript">
$(document).ready(function () {
$(document).mousemove(function (event) {
var x = event.pageX; var y = event.pageY; $("span").text("pageX: " + x +
", pageY: " + y); }); }); </script> </head>
<body>
<br><br><center> <p>The Position of Mouse is at:  <span></span></p>
</center>
</body>
</html>
```

Original:

The Position of Mouse is at: Please move mouse on the page you will see the magic.

Output:

Explanation:

“event.pageX;” returns the x coordinate relative to the document.

“event.pageY;” returns the y coordinate relative to the document.

event.keyCode

“event**.**keyCode” returns the Unicode value of a non-character key in a keyPress event.

Example 5.6

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
function getChar(event) {
alert("key code is: " + event.keyCode);
```

```
}
</script>
```

</head>
<body>

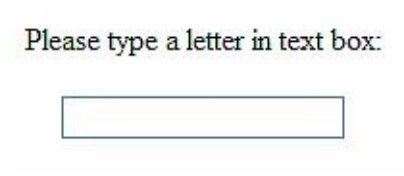
Please type a letter in text box:

<input type=”text” onkeydown="getChar(event);" />

</body>

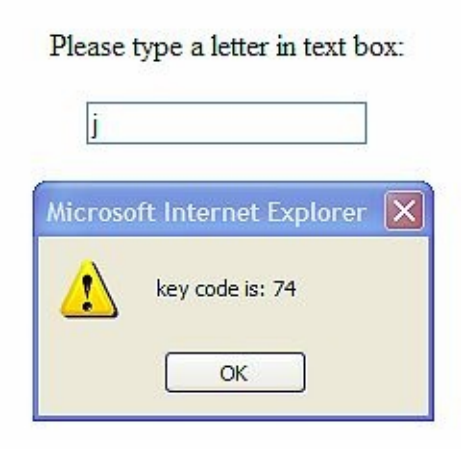
</html>

Original:



Input a letter “j” in the text field and see the result.

Output:


Please type a letter in text box:
j
Microsoft Internet Explorer
key code is: 74
OK

Explanation:

When input a letter “j” into text field, an event occurs, and returns a message “key code is: 74”

“event.keyCode” returns the Unicode value of a non-character key in a keyPress event.

hover(over, out)

The “hover(over, out)” binds handlers for both mouseEnter and mouseLeave events.

Example 5.7

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/JavaScript">
```

```
$(document).ready(function () {
$('div').hover(
```

```
function () {
$(this).css({ "background-color": "red" });
```

```
}
);
```

```
});
</script>
```

</head>
<body>

Mouse Move Here

</body>

</html>

Original:

Move the Mouse on “Mouser Move Here” Div You will see the color of div would be changed from **green** to **red**.

Output:

Explanation:

“$('div').hover(function ( )” runs the function when mouse hovers above the tag “div”.

“$(this).css({ "background-color": "red" }” accesses the “div”, and sets a css style. “this” represents “div”.

Exercises

toggle(function());

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>Toggle</title> </head>

<body>

<input type="button" value="Toggle"

onclick="myFunction()"/> Please click the button.

 This text will be shown and hidden. </body>

<script src="jquery-1.11.3.js"></script> **<script type="text/javascript">**
function myFunction(){

$("#content").toggle(function(){

}); }; </script> </html>

```
<html>
<head>
<title>Toggle</title>
</head>
<body><br>
<input type="button" value="Toggle"
onclick="myFunction()"/>
Please click the button.
<p id="content" >
This text will be shown and hidden.</p>
</body>
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
function myFunction(){
$("#content").toggle(function(){
});
};
</script>
</html>
```

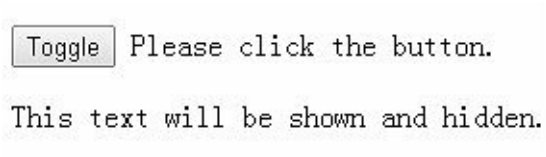
(Figure 1)

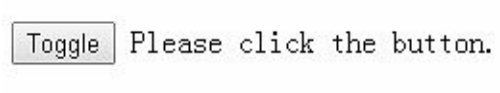
Please save the file with name “Toggle.html”.

Note**:** make sure to use **“.html”** extension name.

Double click “Toggle.html” file, the “Toggle.html” will be run by a browser, click the button, and see the output. (Figure 2)

Output:





(Figure 2)

Explanation:

toggle() function is used to switch different actions.

Hour 6

Effects & Animation

show(duration, callback)

The “show(duration, callback)” shows element with duration. It has two optional parameters (duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function, which is executed after the show() method is completed.

slow	600 milliseconds
normal	400 milliseconds
fast	200 milliseconds

Example 6.1

```
<html>
```

<head>
<title>jQuery show Method with duration and callback parameters</title>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
$(document).ready(function(){
$("#btnShow").click(function(){
```

```
$("p").show("slow",function(){
alert("I am call after paragraph fully shown");
```

```
});
});
```

```
});
</script>
```

</head>
<body>

```
<input type="button" id="btnShow" value="Show Me" />
<p style="display: none">
```

This is a paragraph</p>
</body>

</html>

Original:

Click on “Show Me” buttons and see the result. “This is a paragraph” will be shown.

Output:

Explanation:

“$("#btnShow").click(function()” runs the function when clicking the button whose id is “#btnShow”.

“$("p").show("slow",function()” accesses the tag “p”, shows its contents “This is a paragraph” slowly, and runs the function, returns an alert message “I am call after paragraph fully shown”.

hide(duration, callback)

The “hide(duration, callback)” hides element with duration. It has two optional parameters(duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function which is executed after completing the hide() method.

slow	600 milliseconds
normal	400 milliseconds
fast	200 milliseconds

Example 6.2

```
<html>
<head>
<title>jQuery hide Method with duration and callback parameters</title> <script src="jquery-1.11.3.js"></script> <script type="text/javascript">
$(document).ready(function(){
$("#btnHide").click(function(){
$("p").hide("slow",function(){
alert("I am call after paragraph fully hidden"); }); }); }); </script> </head>
<body>
<input type="button" id="btnHide" value="Hide Me" /> <p> This is a paragraph</p> </body>
</html>
```

Original:

Click on “Hide Me” button and see the result. “This is a paragraph” will be hidden.

Output:

Hide Me

You cannot see “This is a paragraph” now.

Explanation:

“$("#btnHide").click(function()” runs the function when clicking the button whose id is “#btnHide”.

“$("p").hide("slow",function()” accesses the tag “p”, hides its contents “This is a paragraph” slowly, and runs the function, returns an alert message “I am call after paragraph fully hidden”.

toggle(duration, callback)

The “toggle(duration, callback)” toggles between hide( ) and show( ) elements. It has two optional parameters (duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function, which is executed after the toggle() method is completed.

Example 6.3

<html>
<head>

<title>jQuery hide Method with duration and callback parameters</title>
<script src="jquery-1.11.3.js"></script>

```
<script type="text/javascript">
function toggleEffect() {
```

```
$("#divMsg").toggle("fast",function(){
alert("The toggle method is completed!");});
```

```
}
</script>
```

</head>
<body>

<input type="button" onclick="toggleEffect();" value="Toggle between hide and show" >

I am div

```
</body>
</html>
```

Original:

Click on “Toggle between hide and show” button and see the result.

Output:

When you keep clicking the button, message “I am div” will show and hide alternately.

Explanation:

“$("#divMsg").toggle("fast",function()” accesses the tag “#divMsg”, shows and hides its contents “I am div” fast, and runs the function, returns an alert message “The toggle method( ) is completed!”.

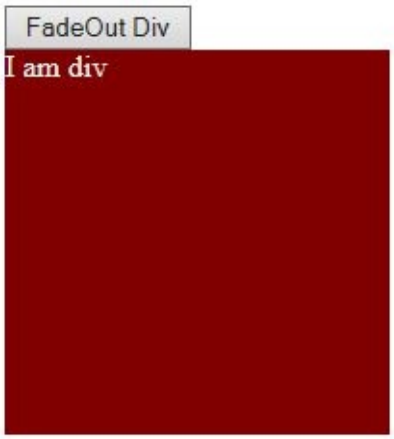
fadeOut(duration, callback)

The fadeOut(duration, callback) changes the opacity for selected elements, from visible to hidden. It has two optional parameters(duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function, which is executed after the fadeOut() method is completed.

Example 6.4

```
<html>
<head>
<title>jQuery hide Method with duration and callback parameters</title> <script src="jquery-1.11.3.js"></script> <script type="text/javascript"> function fadeOutDiv() {
$("#divMsg").fadeOut(500,function(){
alert("The fadeOut method is completed!");}); }
</script> </head>
<body>
<div>
<input type="button" onclick="fadeOutDiv();" value="FadeOut Div" ><div>
<div id="divMsg" style="background-color: maroon; color: white; width: 200px; height: 200px;"> I am div</div>
</body>
</html>
```

Original:



] ck), 500 milliseconds is a duration.

(ıtton and see the result. “I am div” will fade out.

Output:

Explanation:

“$("#divMsg").fadeOut(500,function( )” accesses the tag “#divMsg”, fades out its contents “I am div” in 500 milliseconds, and runs the function, returns an alert message “The fadeout method is completed!”.

fadeIn(duration, callback)

The fadeIn(duration, callback) changes the opacity for selecting elements, from hidden to visible. It has two optional parameters(duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function, which is executed after the fadeIn() method is completed.

Example 6.5

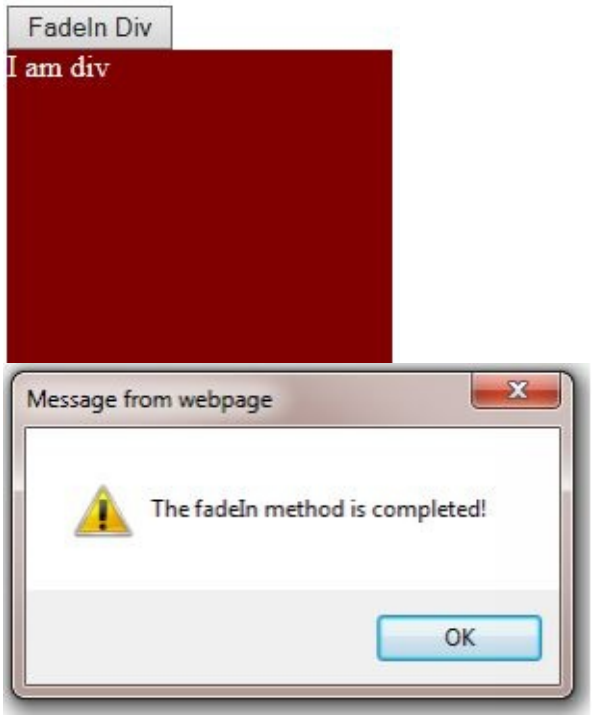
```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function fadeInDiv() {
$("#divMsg").fadeIn(500,function(){
alert("The fadeIn method is completed!");}); }
</script> </head>
<body>
<div>
<input type="button" onclick="fadeInDiv();" value="FadeIn Div" ><div> <div
id="divMsg" style="background-color: maroon; color: white; width: 200px;
height: 200px; display: none;">
I am div</div>
</body>
</html>
```

Note: fadeIn (500, callback), 500 milliseconds is a duration.

Original:

Click on "FadeIn Div" button and see the result. "I am div " will fade in.

Output:


FadeIn Div
I am div
Message from webpage
The fadeIn method is completed!
OK

Explanation:

“$("#divMsg").fadeIn(500,function( )” accesses the tag “#divMsg”, fades in its contents “I am div” in 500 milliseconds, and runs the function, returns an alert message “The fadeIn method is completed!”.

slideUp(duration, callback)

The slideUp(duration, callback) is used to slide the matched element up by hiding. It has two optional parameters (duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function, which is executed after the slideUp() method is completed.

Example 6.6

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function slideUpDiv() {
$("#divMsg").slideUp(500,function(){
```

```
alert("The slideUp method is completed!");});
}
```

```
</script>
</head>
```

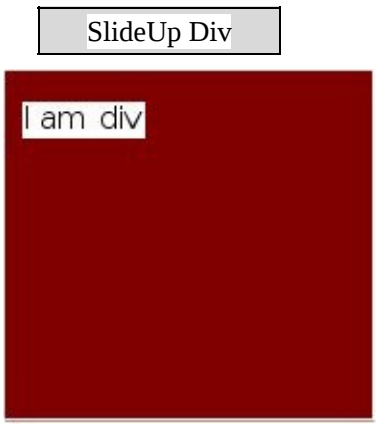
<body>

<input type="button" onclick="slideUpDiv();" value="slideUp Div" >

I am div
</body>

</html>

Original:



Click on “slideUp Div” button and see the result. “I am div” will slide up.

Output:

SlideUp Div

Explanation:

“$("#divMsg").slideUp(500,function( )” accesses the tag “#divMsg”, slides up its contents “I am div” in 500 milliseconds, and runs the function, returns an alert message “The slideUp method is completed!”.

slideDown(duration, callback)

The slideDown(duration, callback) is used to slide the matched element down by showing. It has two optional parameters(duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is function, which is executed after the slideDown() method is completed.

Example 6.7

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function slideDownDiv() {
$("#divMsg").slideDown(500,function(){
```

```
alert("The slideDown method is completed!");});
}
```

```
</script>
</head>
```

<body>

<input type="button" onclick="slideDownDiv();" value="slideDown Div" >

<div id="divMsg" style="background-color: maroon; color: white; width: 200px; height: 200px;

display: none;">
I am div

```
</body>
</html>
```

Original:

Click on “slideDown Div” button and see the result.

Output:


slideDown Div
I am div
Message from webpage
The slideDown method is completed!
OK

Explanation:

“$("#divMsg").slideDown(500,function( )” accesses the tag “#divMsg”, slides down its contents “I am div” in 500 milliseconds, and runs the function, returns an alert message “The slideDown method is completed!”.

slideToggle(duration, callback)

The slideToggle(duration, callback) works between slideUp( ) and slideDown( ) for the selected elements. It has two optional parameters(duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set time in milliseconds. The second parameter “callback” is a function, which is executed after the slideToggle() method is completed.

Example 6.8

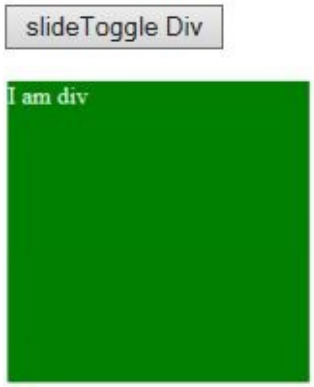
```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function slideToggleDiv() {
$("#divMsg").slideToggle(500,function(){
alert("The slideToggle method is completed!");}); }
</script> </head>
<body>
<div>
<input type="button" onclick="slideToggleDiv();" value="slideToggle Div" >
<div> <div id="divMsg" style="background-color: green; color: white; width:
200px; height: 200px;"> I am div</div>
</body>
</html>
```

Original:

Click on “slideToggle Div” button and see the result.

Output:

Click on “slideToggle Div” button and see the result.



Explanation:

“$("#divMsg").slideToggle(500,function( )” accesses the tag “#divMsg”, slides up and slides down its contents “I am div” in 500 milliseconds, and runs the function, returns an alert message “The slideToggle method is completed!”

In the above example, when you keep clicking the button “slideToggleDiv”, the text “I am Div” will slide up and slide down alternately.

fadeTo(duration, opacity)

The fadeTo(duration, opacity) method gradually changes the opacity for selected elements to a specified opacity (fading effect). It has two optional parameters (duration, callback). The first parameter “duration” is a speed (slow, normal, fast) or you can set a time in milliseconds. The second parameter “callback” is a function, which is executed after the fadeTo() method is completed.

Opacity value: from 1 (opaque) to 0 (transparent).

Example 6.9

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function fadeToDiv() {
$("#divMsg").fadeTo(2000,0.3); }
</script> </head>
<body>
<div>
<input type="button" onclick="fadeToDiv();" value="fadeTo Div" ><div><br
/> <div id="divMsg" style="background-color: green; color: white; width:
200px; height: 200px;"> I am div</div>
</body>
</html>
```

Original:

Click on “fadeTo Div” button and see the result.

Output:

Explanation:

“$("#divMsg").fadeTo(2000,0.3)” accesses the tag “divMsg”, fades its contents “I am div” to the opacity value 0.3 in 2000 milliseconds.

Opacity value: from 1 (opaque) to 0 (transparent).

animate( )

```
animate (“width, height, opacity, fontSize, opposition”,
duration, callback)
```

The animate (arguments) method is used to create customized animations. The “width, height, opacity, fontSize, opposition” parameters define the CSS properties to be animated. The parameter “duration” has three possible values: Slow, Normal, Fast, or you can set the time in milliseconds. The parameter “callback” is a function, which is executed after the animation completes.

Example 6.10

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function animateDiv() {
$("#divMsg").animate({
```

width:'350px', height:'350px', opacity:'0.5', fontSize:'3em', left:'450px'}, 'slow', function( ){

alert("The animate method is completed!")}

```
); }
</script>
```

</head>
<body>

```
<div>
<input type="button" onclick="animateDiv();" value="animate to Div" ><div>
```


I am div with some custom animation

```
</body>
</html>
```

Original:

Click on “animate to Div” button and see the result.

Output:

animate to Div

Explanation:

“$("#divMsg").animate({ width:'350px', height:'350px', opacity:'0.5', fontSize:'3em', left:'450px'}, 'slow', function( )” accesses the tag “#divMsg”, shows a customized animation with specified size, opacity, font, opposition and duration. At last, runs the function and displays an alert message “The animation method is completed!”.

Exercises

animate();

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>Animation</title> <style type="text/css"> #panel {position: relative; width: 100px; height:100px; border: 1px solid #000000;background: yellow; cursor: pointer}

</style>

<script src="jquery-1.11.3.js" type="text/javascript"></script> **<script type="text/javascript"> $(function(){**

$("#panel").css("opacity", "0.8"); $("#panel").click(function(){

$(this).animate({left: "400px", height:"200px" ,opacity: "1"}, 3000) .animate({top: "200px" , width :"200px"}, 3000) .animate({left: "1px", top: "1px", width:"200", height:"200"}, 3000) .fadeOut("slow"); }); });
</script> </head>

<body>

 Click Me! </body>

</html>

```
<html>
<head>
<title>Animation</title>
<style type="text/css">
#panel {position: relative; width: 100px; height:100px; border:
1px solid #000000;background: yellow; cursor: pointer}
</style>
<script src="jquery-1.11.3.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
$("#panel").css("opacity", "0.8");
$("#panel").click(function(){
$(this).animate({left: "400px", height:"200px" ,opacity: "1"}, 3000)
.animate({top: "200px" , width :"200px"}, 3000 )
.animate({left: "1px", top: "1px", width:"200", height:"200"}, 3000)
.fadeOut("slow");
});
});
</script>
</head>
<body>
<div id="panel" ><br><br><center>Click Me!</center></div>
</body>
</html>
```

(Figure 1)

Please save the file with name “Animate.html”.

Note**:** make sure to use **“.html”** extension name.

Double click “Animate.html” file, the “Animate.html” will be run by a browser, click the panel, and see the output.

(Figure 2)

Output:

(Figure 2)

Explanation:

animate() function is used to create a customized animation with jQuery.

Hour 7

Utility Functions

$.each ( )

$(“tag”).each(function()) accesses a specified tag, and repeatedly executes the function.

Example 7.1

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function loopWithEach() {
```

```
$("ul li").each(function () {
$(this).css("color", "red");
```

```
});
}
```

```
</script>
</head>
```

<body>

<input type="button" onclick="loopWithEach();" value="Click Here" />

```
<ul>
<li>First</li>
```

Second
Third

</body>

</html>

Original:

Click on “Click Here” button and see the result.

Output:

Explanation:

“$("ul li").each(function ()” accesses the tag “li” inside the “ul”, and executes the function repeatedly.

“$(this).css("color", "red")” accesses the “this” object which represents tag “li”, and sets a css style.

When you click on “Click Here” button, the color of all three elements are changed to red because each( ) method accesses all elements one by one. each( ) method works like a loop.

$.makeArray ( )

$.makeArray(object) creates an array.

$.makeArray(object) is jQuery utility method which returns JavaScript array created by an array-like object.

Example 7.2

<html>

```
<head>
<script src="jquery-1.11.3.js"></script>
```

```
<script type="text/javascript">
function makeArray() {
```

```
var obj = $("li");
var myArray = $.makeArray(obj);
```

```
myArray.reverse();
$("ul").html(myArray);
```

```
}
</script>
```

</head>
<body>

```
<div>
<input type="button" onclick="makeArray();" value="Click Here" />
```

<ul style="color: black; font-weight: bold;">

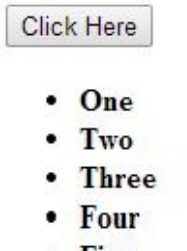
One
Two

Three
Four

Five

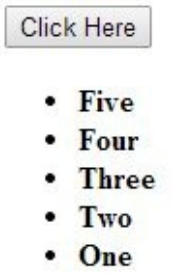

```
</body>
</html>
```

Original:



ck Here" button and see the result. reverse( ) will make its object reversed. Output will show 5,4,3,2,1.

Output:



Explanation:

“var obj = $("li");” creates an object for tag “li”.

“var myArray = $.makeArray(obj)” creates an array for the elements of “li”.

“$("ul").html(myArray);” accesses the tag “ul”, and displays element of “myArray”.

“myArray.reverse();” reverses all elements in “myArray”.

When you clicked on “**Click Here**” button the “li” elements values will be reversed.

$.isArray ( )

$.isArray(array) used to check whether the argument is an array or not. It returns Boolean value “True” or “False”. Let`s take an example for clarification.

Example 7.3

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function checkArray( ) {
var a = [ ]; var b = [1, 2, 3]; var c="I am string"; alert('a is an array?  ' + $.isArray(a)); alert('b is an array?  ' + $.isArray(b)); alert('c is an array?  ' + $.isArray(c)); }
</script> </head>
<body>
<div>
<input type="button" onclick="checkArray();" value="Check Array" /> </div>
</body>
</html>
```

Original:

Click on “Check Array” button and see the result.

Output:


Message from webpage
a is an array: true
OK


Message from webpage
b is an array: true
OK


Message from webpage
c is an array: false
OK

Explanation:

“$.isArray(a)” checks the array “a” to see if it is an array.

When you click on “Check Array” button, the first and second alert box returns true values while third alert box returns false value.

$.inArray ( )

$.inArray() searches for a specified value within an array and returns its index number. If the specified value does not exist in the array, it returns -1.

Example 7.4

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function checkArray() {
var myArray = ["Pete", "John","Andy"];
```

```
$("#spnPosition").html($.inArray("John", myArray));
}
```

```
</script>
</head>
```

<body>

<input type="button" onclick="checkArray();" value="check Array Position" />

John is at position:

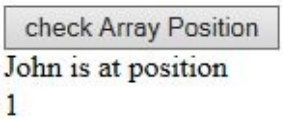
</body>

</html>

Original:

Click on “Check Array Position” button and see the result.

Output:



Explanation:

You can see the result is “1”.

“$("#spnPosition")” accesses the element “#spnPosition”.

“.html(text);” displays the text at specified positon.

“$.inArray("John", myArray)” returns the index number of “John” in myArray.

$.grep ( )

```
$.grep( array, function( values, index ))
```

$.grep(…) method creates a new array by filtering existing array.

Parameter “array”**:** existing array name.

Parameter “values”**:** the elements’ value in existing array.

Parameter “index”**:** the index number in existing array Note**:** The existing array is not affected.

Example 7.5

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
function grepFunc() {
var oldArray = [1, 5, 3, 8, 6, 7, 9,11,15,13];
```

```
newArray = $.grep(oldArray, function ( n, i ) {
return ( n !== 15 && i > 3 );
```

```
// oldArray represents the existing array name
// n represents the elements’ value of oldArray
```

```
// i represents the index number of oldArray
});
```

```
$("#grepValues").html(newArray.join(", "));
}
```

```
</script>
</head>
```

<body>

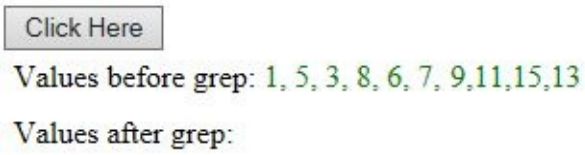
<input type="button" onclick="grepFunc();" value="Click Here" />

Values before grep: 1, 5, 3, 8, 6, 7, 9,11,15,13

Values after grep:

```
</body>
</html>
```

Original:



Click on “Click Here” button and see the result.

Output:

Explanation:

“newArray = $.grep(…)” creates a new array by filtering an existing array.

“n !== 15 && i > 3” filters an existing array by a logical expression**:** elements’ value cannot be 15 and index number must be greater than 3.

“$("#grepValues").html( )” accesses the tag “#grepValues”, and shows the elements’ value of new array in here.

“newArray.join(", ")” joins the elements’ value with “,”.

$.unique ( )

The $.unique( ) function removes duplicate values in an array. It returns unique element values in the array.

Example 7.6

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
function getUniqueValues() {
var myArr = [2, 5, 5, 7, 7, 8]; alert("Before using unique Method :
"+myArr); myArr = $.unique(myArr); //remove duplicate values var val =
""; for (var i = 0; i < myArr.length; i++) {
val += myArr[i] + ","; //get all unique values }
val = val.slice(0, -1) //remove last comma alert("After using unique Method
: "+val); }
</script> </head>
<body>
<div>
<input type="button" onclick="getUniqueValues();" value="Click Here" />
</div>
</body>
</html>
```

Original:

Click on “Click Here” button and see the result.

Output:

Explanation:

“myArr = $.unique(myArr);” removes duplicate element values in myArr.

“array.slice(m, n)” returns elements value from m to n-1. But if “n” is -1, removes the last element value.

$.trim ( )

The trim( ) method removes whitespace at the beginning and at the end of a string.

Example 7.7

<html>
<head>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
$(document).ready(function () {
var str = "        Hello        World!        ";
```

```
$("#afterTrim").html("After trim: " + $.trim(str));
});
```

```
</script>
</head>
```

<body>

Original String: Hello   World!

</body>

</html>

Original:

Original String: Hello World!

Run the browser, you will see the result.

Output:

```
Original String:    Hello    World!
After trim: Hello World!
```

Explanation:

“$.trim(str)” trims the whitespace at the beginning and at the end of the string “str”.

Exercises

replaceAll();

Open Notepad, write jQuery codes (Figure 1): <html>

<head>

<title>test</title> </head>

<body>

<input type="button" value="Replace All"

onclick="myFunction()"/>
  This is an original text,
which will be replaced by a new text. </body>

<script src="jquery-1.11.3.js" type="text/javascript"></script> **<script type="text/javascript"> function myFunction(){**

**$("
 This is new text. Updated! ") .replaceAll("#replaceAll"); }; </script>** </html>

```
<html>
<head>
<title>test</title>
</head>
<body>
<br>
<input type="button" value="Replace All"
onclick="myFunction()"/><br>
<p id="replaceAll">
This is an original text,</p>
<p id="replaceAll">which will be replaced by a new text.</p>
</body>
<script src="jquery-1.11.3.js" type="text/javascript"></script>
<script type="text/javascript">
function myFunction(){
$("<br><div><strong>This is new text. Updated!</strong></div>")
.replaceAll("#replaceAll");
};
</script>
</html>
```

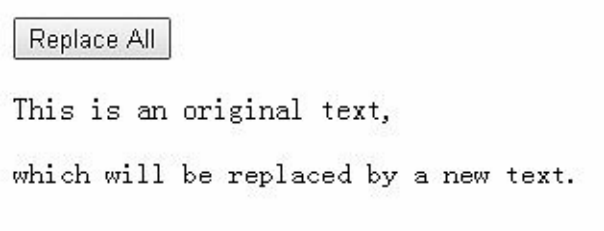
(Figure 1)

Please save the file with name “ReplaceAll.html”.

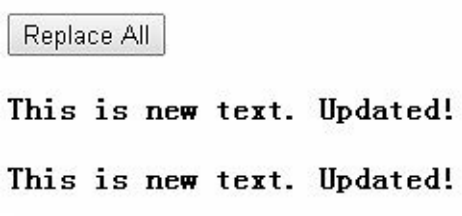
Note**:** make sure to use **“.html”** extension name.

Double click “ReplaceAll.html” file, the “ReplaceAll.html” will be run by a browser, click the button for two times, and see the output. (Figure 2)

Original:



Output:



(Figure 2)

Explanation:

replaceAll() function is used to replace an existing whole text with a new text.

Hour 8

Ajax by jQuery

What is Ajax?

AJAX stands for "**A**synchronous **J**avaScript **A**nd **X**ML". AJAX is a new technique for creating better, faster, and more interactive web scripts with XML, HTML, CSS, PHP, ASP and Java Script by the server.

AJAX is used to update parts of a web page, without reloading the whole page.

That means with Ajax, you can communicate with the server to renew the information, without refreshing the web page.

By jQuery, you can easily implement Ajax in your web pages.

To run Ajax, you need a server.

Set up Server

If you want to run Ajax, you need to set up a server in your own computer.

"**AppServ**" is free software to setup a server; it can run Apache, PHP, MySQL, PhpMyAdmin, Ajax, JavaScript, JQuery……

Step 1:

Download “**AppServ**” form link: (Figure 1)
http://www.appservnetwork.com/

(Figure 1)

Step 2:

Install AppServ to local computer.

C:\AppServ

(Figure 2)

(Figure 2)

Step 3:

Please check all box to install anything.(Figure 3)

(Figure 3)

Step 4:

Server Name: localhost

Email Address: xxxxxxxx@xxxxx.xxx Apache HTTP Port: 80 (Figure 4)

(Figure 4)

Step 5:

Enter root password: **12345** (Figure 5) (Default user name: **root**)

(Figure 5)

Step 6:

Check the box, Start Apache, Start MySQL

Click Finish button. (Figure 6)

(Figure 6)

Step 7:

Test to see if AppServ installation is successful, please run a browser, enter the address: **http://localhost** If you can see the page like this: (Figure 7)

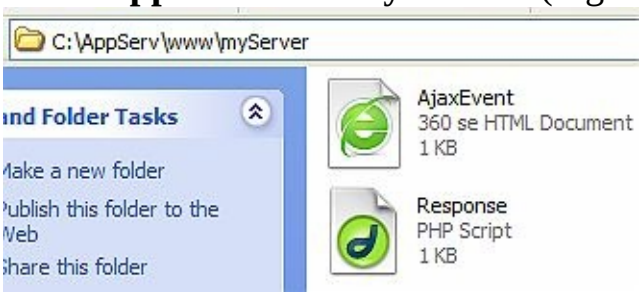
(Figure 7)

Congratulation! Your own server runs successfully.

Step 8:

How to use server?

Open folder **C:\AppServ\www**, please create a folder named “myServer”, which can work as a workplace. You can put all you program files to “myServer” folder. From now on, you are able to run Ajax files, PHP files and HTML files on **C:\AppServ\www\myServer.** (Figure 8)



(Figure 8)

Note:

Please use "**http://localhost/.....**"to run any files on the server, for example: If you want to run AjaxEvent**.**html, enter address: **http://localhost/myServer/**AjaxEven**.**html If you want to run Response**.**php, enter address: **http://localhost/myServer/**Response**.**php

load( )

load( ) can remotely load the resource file from the server, and displays the downloaded data.

```
load( url,  data,  function )
```

Argument "url" is a resource file on the server.

Argument "data" sends data {key**:** value} to server. (optional) Argument "function" will run when the Ajax operation is complete. (optional)

Example 8.1

(1) A resource file is on the server: (myFile.txt)
(C:\AppServ\www\myServer\myFile.txt)

Hello! Ajax is testing!

(2) jQuery file is on the server: (loadText.html)
(C:\AppServ\www\myServer\loadText.html)

```
<html>
<head>
<body>
<script src="jquery-1.11.3.js"></script> <script type="text/javascript">
$(function(){
$("div").load("myFile.txt"); }); </script> </head>
<br>Respond: <br><br><div> </div> </body>
</html>
```

(3) jquery-1.11.3.js is on the server.

(C:\AppServ\www\myServer\jquery-1.11.3.js) Run the loadText.html by address: **http://localhost/myServer/loadText.html**

Output:

Respond:

Hello! Ajax is testing!

Explanation:

“$(function(){ }” executes the function when loadTest.html is loaded.

“$(“div”).load(“myFile.txt”);” loads the “myFile.txt” and display its contents in tag “div” of loadTest.html.

Note:

“jquery-1.11.3.js”, “myFirst.txt” and “loadText.html” should be put in the same folder.

If you want to know more about Ajax, you‘d better learn one of the languages such as PHP, ASP and JSP first.

$.post( )

$.post( ) can send data to the server.

```
$.post( url,  data,  function )
```

Argument “url” is a resource file on the server.

Argument “data” sends data {key**:** value} to server.

Argument “function” will run when the Ajax operation is complete.

Example 8.2

(1) A resource file on the server: (postFile.php)
(C:\AppServ\www\myServer\postFile.php) <html>

<body>

<?php

if ($_POST["data"] == "100") { echo ("100"); }

?>

</body>

</html>

(2) jQuery file: (postTest.html) (C:\AppServ\www\myServer\postTest.html)

<html>
<body>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
$(function(){
$.post ("postFile.php", {data:100}, function(data){
```

```
$("div").html(data);
}); });
```

</script>

Respond:

</body>

</html>

(3) jquery-1.11.3.js is on the server.

(C:\AppServ\www\myServer\jquery-1.11.3.js) Run the postTest.html by address: **http://localhost/myServer/postTest.html**

Output:

Respond:

100

Explanation:

“if ($_POST[“data”] == “100”) tests the received data to see if it equals 100.

“echo “100”;” displays 100.

“{data**:** 100}” sends the “data**:**100” to the server.

“function(data)” is a callback function which will run when the ajax operation is complete.

“$(“div”).html(data);” displays the “data” in the tag “div” of postTest.html.

Note:

postFile.php and postTest.html should be put in the same folder.

$.get( )

$.get( ) can send data to the server.

```
$.get( url,  data,  function )
```

Argument “url” is a resource file on the server.

Argument “data” sends data {key**:** value} to server.

Argument “function” will run when the Ajax operation is complete.

Example 8.3

(1) A resource file on the server: (getFile.php)

(C:\AppServ\www\myServer\getFile.php) <html>

<body>

<?php

if ($_GET["data"] == "100") { echo ("100"); }

?>

</body>

</html>

(2) jQuery file: (getTest.html) (C:\AppServ\www\myServer\getTest.html)
<html>

<body>

<script src="jquery-1.11.3.js"></script> **<script type="text/javascript">**
$(function(){

$.get ("getFile.php", {data:100}, function(data){

$("div").html(data); }); }); </script> Respond:

</body>

</html>

(3) jquery-1.11.3.js is on the server.

(C:\AppServ\www\myServer\jquery-1.11.3.js) Run the getTest.html by address:
http://localhost/myServer/getTest.html

Output:

Respond:

100

Explanation:

“if ($_GET[“data”] == “100”) tests the received data to see if it equals 100.

“echo “100”;” displays 100.

“{data**:** 100}” sends the “data**:**100” to the server.

“function(data)” is a callback function which will run when the ajax operation is complete.

“$(“div”).html(data);” displays the “data” in the tag “div” of getTest.html.

$.ajax( ) with post method

$.ajax( ) can communicates with the server, and replaces the load( ), $.post( ), $.get( ).

```
$.ajax({
type: “GET” or “POST”,
url: “source file”,
data: { key: value }
success: function(data){ }

                                    });
```

Argument “GET/POST” is the http request type.

Argument “url” is a resource file on the server.

Argument “data” sends data {key**:** value} to server.

Argument “function” will run when the Ajax operation is successful.

Example 8.4

(1) A resource file on the server: (postFile.php)
(C:\AppServ\www\myServer\postFile.php) <html>

<body>

<?php

if ($_POST["data"] == "100") { echo ("100"); }

?>

</body>

</html>

(2) jQuery file: (postAjax.html) (C:\AppServ\www\myServer\postAjax.html)
<html>

<body>

<script src="jquery-1.11.3.js"></script> **<script type="text/javascript">**
$(function(){

$.ajax({

type: "POST", url: "postFile.php", // load postFile.php data: {data: 100},
success: function(data){

$("div").html(data); // show data in "div"

}

});}); </script> Respond:

</body>

</html>

(3) jquery-1.11.3.js is on the server.

(C:\AppServ\www\myServer\jquery-1.11.3.js) Run the postAjax.html by address:

http://localhost/myServer/postAjax.html

Output:

Respond:

100

Explanation:

$.ajax( ) communicates with the server using POST method.

$.ajax( ) with get method

$.ajax( ) can communicates with the server, and replaces the load( ), $.post( ), $.get( ).

```
$.ajax({
type: “GET” or “POST”,
url: “source file”,
data: { key: value }
success: function(data){ }

                                    });
```

Argument “GET/POST” is the http request type.

Argument “url” is a resource file on the server.

Argument “data” sends data {key: value} to server.

Argument “function” will run when the Ajax operation is successful.

Example 8.5

(1) A resource file on the server: (getFile.php)\
(C:\AppServ\www\myServer\getFile.php) <html>

```
<body>
<?php
if ($_GET["data"] == "100") { echo ("100"); }
?>
</body>
</html>
```

(2) jQuery file: (getAjax.html) (C:\AppServ\www\myServer\getAjax.html)

<html>
<body>

```
<script src="jquery-1.11.3.js"></script>
<script type="text/javascript">
```

```
$(function(){
$.ajax({
```

```
type: "get",
url: "getFile.php",   // load getFile.php
```

data: {data: 100},

success: function(data){

```
$("div").html(data);  // show data in "div"
}
```

```
});});
</script>
```

Respond:

```
</body>
</html>
```

(3) jquery-1.11.3.js is on the server.

(C:\AppServ\www\myServer\jquery-1.11.3.js) Run the getAjax.html by address:

http://localhost/myServer/getAjax.html

Output:

Respond:

100

Explanation:

$.ajax( ) communicates with the server using GET method.

Ajax Error

$.ajax( ) can handle errors with a callback function.

```
$.ajax({
type: “GET” or “POST”,
url: “source file”,
data: { key: value }
error: function(xhr, message, e){ }
                                    });
```

“xhr” represents a XMLHttpRequest Object.

“message” represents a error message.

“e” is an exception object.

Example 8.6

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script>
$(document).ready(function(){
$.ajax({
type: "GET", url: "noFile.txt",   // wrong file success: suss, error: err }); });
function suss(data) {
$("div").text(data); }
function err(xhr, message, e) {
$("div").text(message); }
</script> </head>
<body>
<h3>Handling Ajax errors</h3> Success or error message: <div></div>
</body>
</html>
```

Output:

Sussess or error message:

error

Explanation:

“url: "noFile.txt", ” specifies a file that is not existing, so an error occurred.

“error: err” catches the error and call the function err( ){ }.

“$("div").text(message);” shows the error message.

Ajax Success

$.ajax( ) will execute the callback function if Ajax runs successfully.

```
$.ajax({
type: “GET” or “POST”,
url: “source file”,
data: { key: value }
success: function(data){ }
                                    });
```

“success: function(data){ }” will execute the function if Ajax runs successfully.

Example 8.7

```
<html>
<head>
<script src="jquery-1.11.3.js"></script> <script>
$(document).ready(function(){
$.ajax({
type: "GET", url: "myFile.txt",    // right file success: suss,
error: err }); }); function suss(data) {
$("div").text(data); }
function err(xhr, message, e) {
$("div").text(message); }
</script> </head>
<body>
<h3>If run successfully</h3> Success or error message: <div></div> </body>
</html>
```

Output:

Success or error message:
Hello! Ajax is testing!

Explanation:

“url: "myFile.txt",” specifies a file that is existing, and there is no any incorrect codes in the whole program, therefore, the Ajax runs successfully.

“success: suss,” will call the function suss( ){ } if the Ajax program runs successfully.

“$("div").text(data);” will show “myFile.txt”s content:”Hello! Ajax is testing!”

Exercises

Ajax Event

Open Notepad, write two files as following, and put them to the same folder "myServer" with **jquery-1.11.3.js** in the server.

(1) Response.php (C:\AppServ\www\myServer\Response.php) <?php

if ($_POST["data"] == "1") {

echo 'Ajax Runs Successfully!'; }

if ($_POST["data"] == "2") {

echo 'Ajax Error Occurs!'; }

?>

(2) AjaxEvent.html (C:\AppServ\www\myServer\AjaxEvent.html) <html>

<head>

<script src="jquery-1.11.3.js"></script> **<script> function myFun(){**

$.ajax({

type: "POST", url: "Response.php", data: {data: 1}, // set "data" as 1

success: callback, }); }; function callback(data){

$("#results").text(data); }

</script> </head>

<body>

<input type="button" value="Run Ajax"

onclick="myFun()">

Handling Ajax events

 Click the button, Ajax responses:

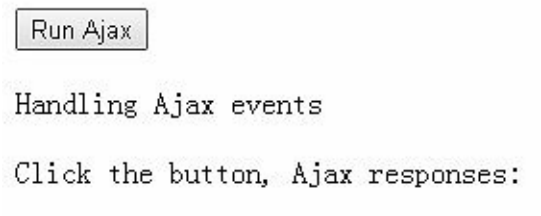
<h3> </h3> </body>

</html>

(3) jquery-1.11.3.js is on the server.

(C:\AppServ\www\myServer\jquery-1.11.3.js) Run the AjaxEvent.html by address: **http://localhost/myServer/AjaxEven.html** And click the button. (Figure)

Original:


Run Ajax
Handling Ajax events
Click the button, Ajax responses:

Output:

(Figure)

How time flies! It is time to say good-bye.

www.websprogram.com/books.php

www.amazon.com/author/rayyao

Source Code for Download

Please download the source code of this book.

Source Code of JQuery for Download

Dear My Friend, If you are not satisfied with this eBook, could you please return the eBook for a refund within seven days of the date of purchase?
If you found any errors in this book, could you please send an email to me?
yaowining@yahoo.com
I will appreciate your email. Thank you very much!
Sincerely
Ray Yao

My friend, See you!