



|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Change Status |
| 1.0 |  |  |  |

**TABLE OF CONTENTS**

* ABOUT THIS GUIDE 5
* LEGEND 6
* SESSION SCHEDULE 7
* TOPIC 1 – Session Introduction 8
* SUBTOPIC 1: Participant’s Introduction 8
* SUBTOPIC 2: Ice Breaker Activity 9
* SUBTOPIC 3: Session Objectives 10
* TOPIC 2 –Getting Started with jQuery 12
* SUBTOPIC 1: jQuery Introduction 12
* SUBTOPIC 2: jQuery Syntax 13
* SUBTOPIC 3: jQuery Selectors 15
* SUBTOPIC 4: jQuery Events 18
* TOPIC 3 –jQuery Effects 22
* SUBTOPIC 1: jQuery Hide/Show 22
* SUBTOPIC 2: jQuery Fade 24
* SUBTOPIC 3: jQuery slide 29
* SUBTOPIC 4: jQuery Animate 32
* SUBTOPIC 5: jQuery Callback 36
* SUBTOPIC 6: jQuery Chaining 38
* TOPIC 4 –jQuery HTML 40
* SUBTOPIC 1: jQuery Get 40
* SUBTOPIC 2: jQuery Set 44
* SUBTOPIC 3: jQuery Add 47
* SUBTOPIC 4: jQuery Remove 50
* SUBTOPIC 5: jQuery CSS classes 55
* SUBTOPIC 6: jQuery css() 61
* TOPIC 5 –jQuery Traversing 65
* SUBTOPIC 1: jQuery Traversing 65
* SUBTOPIC 2: jQuery Ancestors 68
* SUBTOPIC 3: jQuery Descendants 73
* SUBTOPIC 4: jQuery Siblings 78
* SUBTOPIC 5: jQuery Filtering 82
* TOPIC 6 –Summary 89

**ABOUT THIS GUIDE**

This Participant’s Guide addresses an ILT program of 100 hours duration based on the **ITES Functional Skills Training** course. The program can be divided into 10 sessions.

This Participant’s Guide will help the participants gain a thorough understanding of the basic functional skills required in the ITES industry.

The features of this guide are as follows:

* It provides hands on experience of the basic Web Design functional skills that are required in the ITES industry.
* It provides references to the page numbers of the ILT version of this program. The instructor will discuss the points as mentioned in this guide when displaying the various screens in the course.
* It provides spaces after each screen content for the participants to note down any important points being explained by the instructor or write their own comments related to the same content.
* It includes activities that you will perform in the class and instructions to carry out the activities.
* It includes a ‘References’ section that provides a list of books and other sources that can help you enhance your knowledge on the concepts that are taught in the class.

**LEGEND**

The following legend explains the various icons used in this guide.

|  |  |
| --- | --- |
| Icon | Description |
|  | Approximate duration of each section |
|  | Project course screen |
|  | Discussion |
|  | Group Activity |
|  | Activity |
|  | Important |

180 minutesThe facilitator will present the program schedule for the session in detail.



# 

**SUBTOPIC 1: Participant’s Introduction**

**TOPIC 1 – Introduction To jQuery**

 **The ILT course will be projected.**

## 

## E:\Projects\Equate-IES_VSS\Production Assets\Graphics\Used Assets\global\icons\EQUATE_icon0007.png

**SUBTOPIC 2: Icebreaker activity 1**

Take part in the activities as guided by the facilitator, and make notes as said by the facilitator.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

## E:\Projects\Equate-IES_VSS\Production Assets\Graphics\Used Assets\global\icons\EQUATE_icon0007.png

**SUBTOPIC 3: Icebreaker activity 2**

* Write down as many keywords as you can related to jQuery in your notepads. You can discuss that within the group and write them within 5 minutes.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

Now, the facilitator will present the objectives of the course. This would include various jQuery topics that would be covered in this course. S/he will also outline the tasks that you would learn by attending this course.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

## 

**Topic 2: Getting started with jQuery**

## 

**SUBTOPIC 1: jQuery Introduction**

jQuery, in reality, is JavaScript itself. The only difference is that JavaScript is a programming language while jQuery is a framework built with JavaScript to aid web developers in their tasks.

jQuery is a lightweight JavaScript library that was designed to make the JavaScript programming simpler and better. It converts a lot of multi-line JavaScript codes into simple methods, which can be called with a single line of code.

jQuery seems to be the most popular JavaScript framework and also the most extendable. Most organizations on the Web use jQuery, such as Google, Microsoft, IBM, Netflix, etc.

**Adding jQuery to a Page**

You can add jQuery to your web page in two ways. The first way is to download the jQuery from the official Website, jquery.com. The second way is to include it from a Content Delivery Network (CDN), such as Google and Microsoft.

**Downloading jQuery**

There are two downloadable versions of jQuery:

* Production version – this version is best for your live Website because it has been minified and compressed
* Development version – this version is best suited for testing and development. It comprises uncompressed and readable codes

The jQuery library is referenced with HTML ***<script>***tag. Make sure that you keep the tag within the ***<head>*** section.

**jQuery from CDN**

If you do not wish to download and host jQuery yourself, you can include it from a Content Delivery Network, such as Google & Microsoft.To use jQuery from Google or Microsoft, you can use one of the following:

**Google CDN:**

<head>  
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>  
</head>

**Microsoft CDN:**

<head>  
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.11.3.min.js"></script>  
</head>

Notes: The jQuery library is referenced with HTML ***<script>*** tag. Make sure that you keep the tag within the ***<head>*** section.

**SUBTOPIC 2: jQuery Syntax**

jQuery syntax comprises three elements: a **$** sign to access jQuery, **SELECTORS** to query the HTML elements, and **ACTION,** which is to be performed on the HTML elements.

**Syntax:**

$(selection).action()

**Example:**

$(“p”).show() - shows all <p> elements.

**The Document Ready Event**

Every jQuery code runs within a Document Ready Event. This prevents jQuery code to run before the document is fully loaded. There are some actions that can fail if the document is not ready before it is executed.

You can’t hide an element not created yet or you can’t get the size of an image not loaded yet if the JavaScript runs before the document is loaded, which is why Document Ready Event is used.

**Syntax:**

$(document).ready(function(){

// jQuerycode

});

There is an alternate method available for Document ready event

**Syntax:**

$(function(){

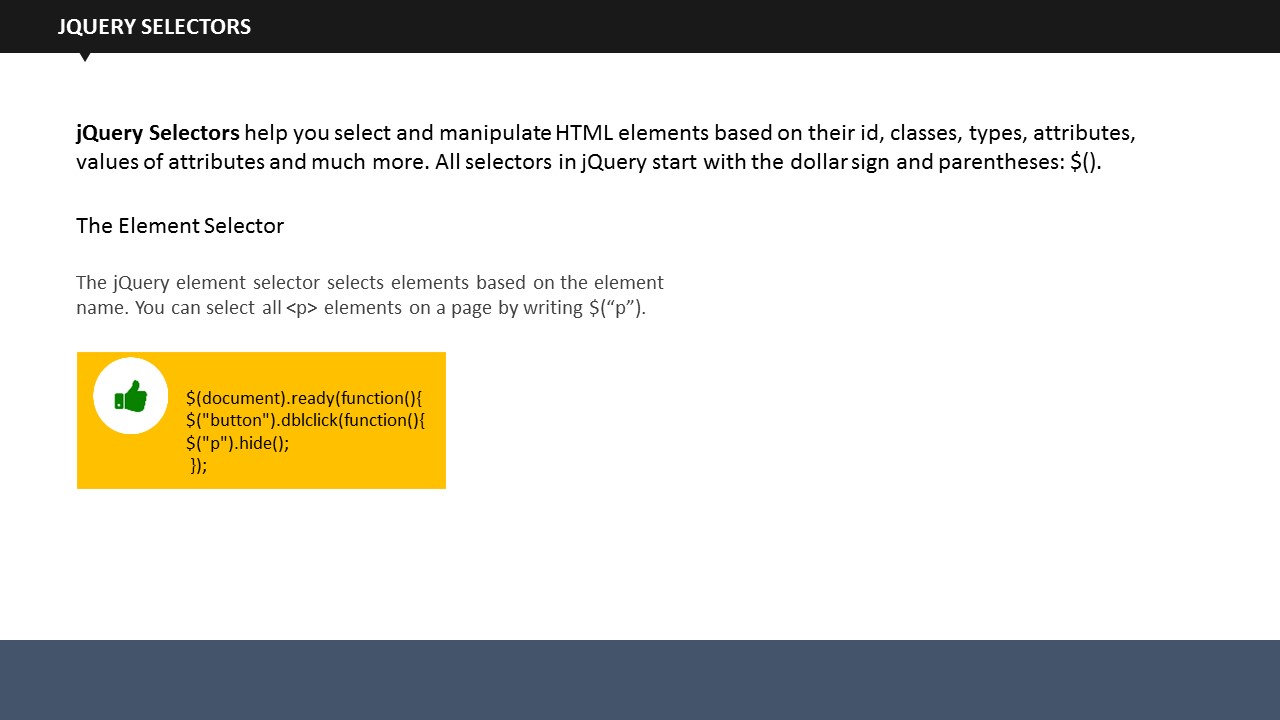
// jQuery code

});

However, for ease of understanding, the former method is easier than the latter.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 3: jQuery Selectors**

jQuery selectors are an integral part of jQuery library. It allows web developers to find and perform action(s) on HTML elements.

HTML elements are find on basis of their classes, id, attributes, value of attributes, types and much more.

All selectors are written within the brackets and are preceded with a $ sign. See different jQuery selectors below.

**The Element Selector**

The Element selector selects an element based on its name.

**Example:**

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

In this example, when a user double clicks a button, all the ***<p>*** elements will be hidden.

**The #id Selector**

The #id selector uses the id attribute of HTML tags. It is used when you want to find a specific, unique element, which is why id should remain unique.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("#check").hide();

});

});

</script>

</head>

<body>

<h2>Heading</h2>

<p> Paragraph.</p>

<p id="check"> Another paragraph.</p>

<button> Double click me</button>

</body>

</html>

When the user double clicks on the button, element with id=”check” will be hidden.

**The .class Selector**

The jQuery .class Selector selects elements corresponding to a particular class. You will use class name preceded by a period to find elements from that particular class.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$(".check").hide();

});

});

</script>

</head>

<body>

<h2 class="check">Heading text</h2>

<p class="check">Paragraph text.</p>

<p>another paragraph.</p>

<button> double click me</button>

</body>

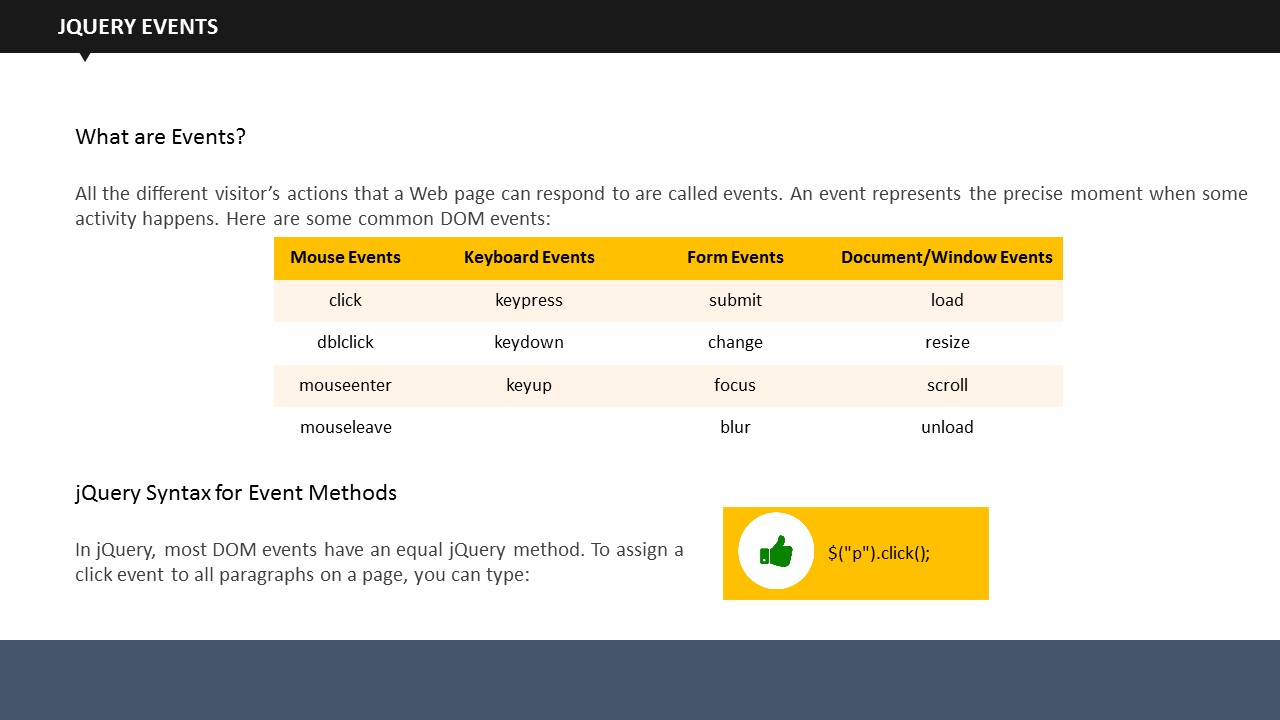
</html>

When a user double clicks the button, ***<h2>*** element and ***<p>*** element with class=”check” will be hidden.

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

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**SUBTOPIC 4: jQuery Events**

An Event is an action performed by the visitor on a web page. All the different visitor’s actions that a Web page can respond to are called events. An event represents the precise moment when some activity happens

**Example:**

* Moving a mouse
* Checking a checkbox

On the screen, you can see different mouse, keyboard, form, and windows related events.

**Syntax:**

$("p").click();

This syntax is to assign a click event to all paragraphs on a page. Similarly, we use other events as well.

jQuery has multiple methods to respond to events. You can see them listed below.

**$(document).ready()**

$(document).ready() is an example of jQuery event method where a function will execute only after the document is fully loaded.

**click()**

The click() method is executed when the user clicks the HTML element.

**Syntax:**

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

* After assigning, you have to define what should happen when that event fires.

**dblclick()**

The dblclick() method is executed when the user double-clicks the HTML element.

**Syntax:**

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

**mouseup()**

The mouseup() method is executed when the left mouse button is released, while the mouse is over the HTML element.

**Syntax:**

$("#p1").mouseup(function(){  
 alert("Mouse up over p1!");  
});

**hover()**

The hover() method combines the mouseenter() and mouseleave() methods. The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML element.

**Syntax:**

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

**focus()**

The focus() method is executed when the cursor is seen on the form field.

**Syntax:**

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

**blur()**

The blur() method is executed when the cursor moves away from the form field.

**Syntax:**

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

**on()**

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

* Attaching a click event to a ***<p>*** element

**Syntax:**

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

* Attaching multiple event handlers to a <p> element

**Syntax:**

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

**The on() Method**

This method allows one or more event handlers for the selected elements.

**Example**

$("p").on({  
    mouseenter: function(){  
        $(this).css("background-color", "gray");  
    },   
    mouseleave: function(){  
        $(this).css("background-color", “red");  
    },   
    click: function(){  
        $(this).css("background-color", "green");  
    }   
});

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**TOPIC 2: jQuery Effects**

## 



**SUBTOPIC 1: jQuery Hide/Show**

There are a lot of effects that can be added to a web page taking help of jQuery. The hide and show are two of the most basic effects used to hide and show HTML elements respectively.

**Example:**

$(“button”).click(function(){

$(“p”).hide();

On clicking the button, the hide function will hide the entire paragraph. Similarly, the show function will show the hidden paragraph.

**Syntax:**

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

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

The speed parameter (optional) determines how long it will take for the effect to take place. It takes values such as “slow, fast, or milliseconds”. The callback parameter is also optional. What it does is it gets executed after hiding or showing is completed. You will learn more about the callback later.

**Example:**

$(“button”).click(function(){

$(“p”).hide(1000);

});

On clicking the button, the entire text in the paragraph will be hidden within 1 second (1000 milliseconds).

**jQuery toggle()**

You can toggle between the hide() and show() using the toggle() method. With help of toggle(),the visible elements get hidden and hidden elements become visible..

**Syntax:**

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

**Example:**

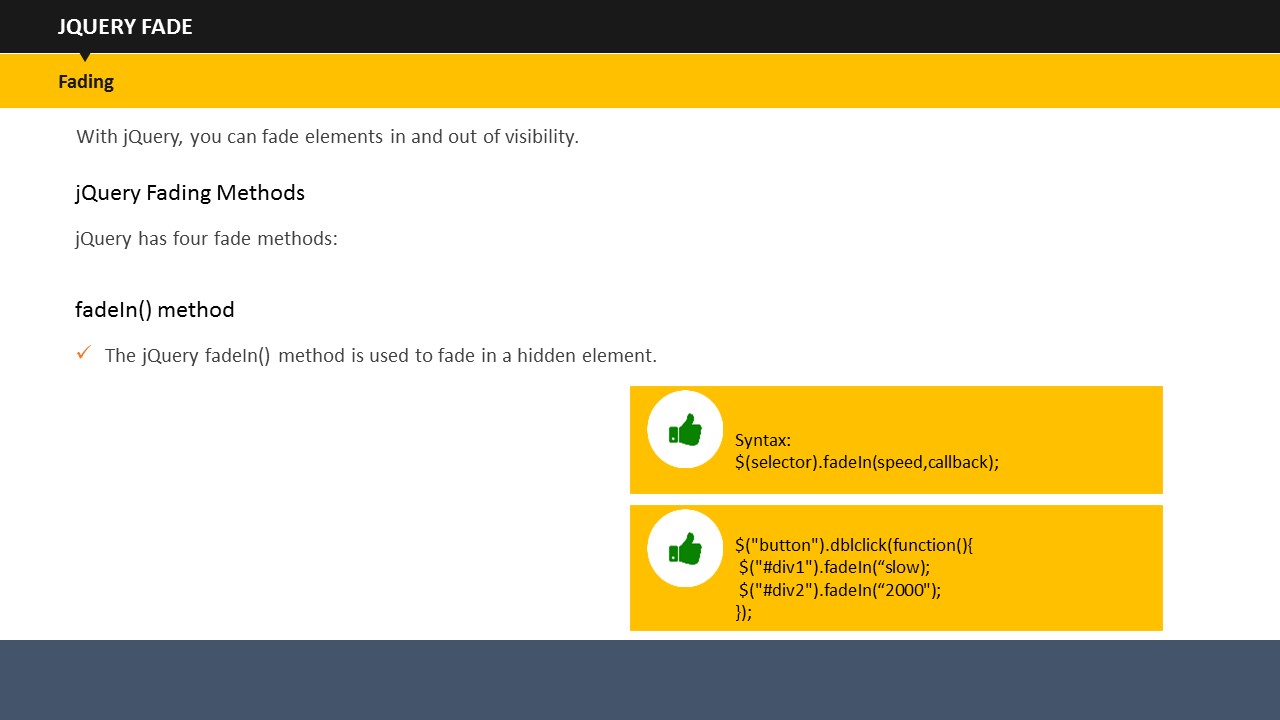
$(“button”).click(function(){

$(“p”).toggle();

});

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**SUBTOPIC 2: jQuery Fade**

jQuery provides an option to fade the text, images or graphics present on the web page.

You can fade in, fade out, fade toggle, and fade to effects to give your web page a lovely look.

For this, there are four methods you can use, which are fadeIn(), fadeOut(), fadeToggle() and fadeTo() methods.

See how these methods work.

**jQueryfadeIn() Method**

**Syntax:**

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

This method fades in a hidden element on page.

Coming on to the syntax, the speed here reflects how much time it would take for elements to show that effect. The callback function will be studied later.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$(“button”).dblclick(function(){

$(“#div1”).fadeIn(“slow”);

$(“#div2”).fadeIn(“2000”);

});

});

</script>

</head>

<body>

<p>Show fadeIn() with various parameters.</p>

<button> Double Click to fade in boxes</button><br><br>

<div id="div1" style="width:80px;height:80px;display:none;background-color:red;"></div><br>

<div id="div2" style="width:80px;height:80px;display:none;background-color:green;"></div><br>

</body>

</html>

In this example, there is a button on whose double click event will fade in two boxes. The first box will appear slowly while the second box will appear after 1 second of the double click event.

**jQuery fadeOut() Method**

**Syntax:**

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

This methods fades out the already visible elements.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$(“button”).dblclick(function(){

$(“#div1”).fadeOut(“slow”);

$(“#div2”).fadeOut(“2000”);

});

});

</script>

</head>

<body>

<p>Show fadeOut() with various parameters.</p>

<button> Double Click to fade in boxes</button><br><br>

<div id="div1" style="width:80px;height:80px;display:none;background-color:red;"></div><br>

<div id="div2" style="width:80px;height:80px;display:none;background-color:green;"></div><br>

</body>

</html>

In this example, there is a button on whose double click event will fade out two boxes. The first box will fade out slowly while the second box will disappear after 2 seconds of the double click event.

**jQuery fadeToggle() Method**

**Syntax:**

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

The fadeToggle() method, as the name suggests, toggles between fadeout() and fadeIn() methods. If the elements are already faded in, it would fade out them, and vice versa.

**Example:**

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("#div1").fadeToggle(“slow);

$("#div2").fadeToggle("4000");

});

});

</script>

</head>

<body>

<p>Show fadeToggle() with different speed parameters.</p>

<button> Double Click to fade in/fade out boxes</button><br><br>

<div id="div1" style="width:80px;height:80px;background-color:red;"></div>

<br>

<div id="div2" style="width:80px;height:80px;background-color:green;"></div>

<br>

</body>

</html>

In this example, on double clicking the button, the boxes will fade in and fade out as per their initial position.

**jQuery fadeTo() Method**

**Syntax:**

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

****The fadeTo() method helps in fading the elements to the given opacity.

* Opacity level is the transparency level of an element. You can set it according to you. It can be set between 0 to 1 where 0 is fully transparent, 0.5 is half transparent, while 1 signifies no transparency.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("#div1").fadeTo("slow", 0.15);

$("#div2").fadeTo("slow", 0.65);

});

});

</script>

</head>

<body>

<p> Show fadeTo() with different parameters.</p>

<button> Double Click to fade boxes</button><br><br>

<div id="div1" style="width:80px;height:80px;background-color:red;"></div><br>

<div id="div2" style="width:80px;height:80px;background-color:green;"></div><br>

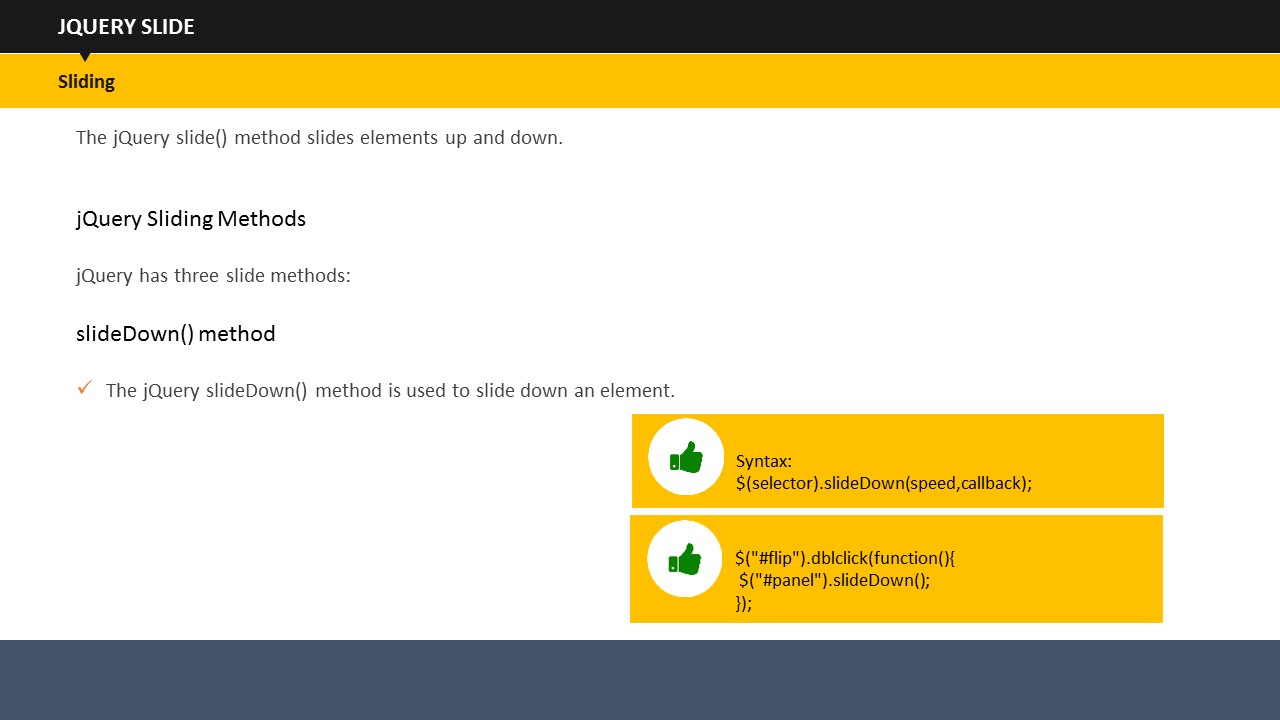
</body>

</html>

This example will fade the two given boxes to their respective opacity values. The red box will fade to 0.15 while the green box will fade to 0.65 opacity. Both the boxes will fade to their respective opacity very slowly.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**SUBTOPIC 3: jQuery Slide**

You must have seen sliders on several Websites and must have been greatly impressed by the same.

Now you can create that yourself as jQuery makes use of Slide effect which is used to slide up and down the elements.

slideUp(), slideDown(), and slideToggle() are three methods provided in jQueryto achieve the sliding effect.

See how these methods work.

**jQuery slideUp() Method**

**Syntax:**

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

The slideUp() Method slides up an element.

The say and callback parameter works exactly as we have just studied in previous topics.

See the example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("#flip").dblclick(function(){

$("#panel").slideUp("2000");

});

});

</script>

<style>

#panel, #flip {

padding: 4px;

text-align: center;

background-color: #e5eecc;

border: solid 2px #c3c3c3;

}

#panel {

padding: 60px;

}

</style>

</head>

<body>

<div id="flip">double click to slide up this panel</div>

<div id="panel">Hello User!</div>

</body>

</html>

This example will slide up the panel upon double clicking the flip box. The panel will take two seconds to completely slide up because the speed parameter is set at 2000 milliseconds.

**jQuery slideDown() Method**

**Syntax:**

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

The slideDown() Method slides down an element.

See an example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("#flip").dblclick(function(){

$("#panel").slideDown("2000");

});

});

</script>

<style>

#panel, #flip {

padding: 4px;

text-align: center;

background-color: #e5eecc;

border: solid 2px #c3c3c3;

}

#panel {

padding: 60px;

}

</style>

</head>

<body>

<div id="flip">double click to slide down this panel</div>

<div id="panel">Hello User!</div>

</body>

</html>

This example will slide down the panel when the flip box is double clicked. This panel will again take two seconds to completely slide up because the speed parameter is set at 2000 milliseconds.

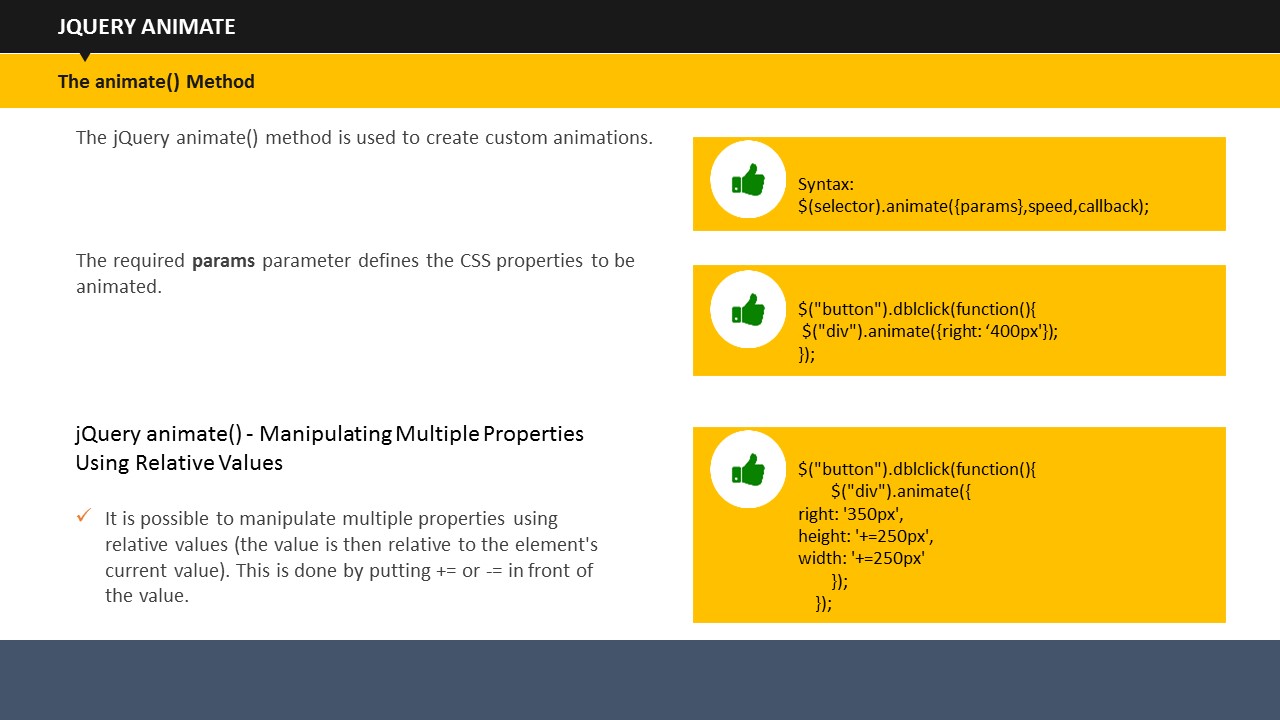
**jQuery slideToggle()**

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

The slideToggle() method, as the name suggests, toggles between slideDown() and slideUp() methods. If the element is already slid down, it would slid them up, and vice versa.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 4: jQuery Animate**

jQuery lets you create custom animations with the help of jQuery animate() method.

**Syntax:**

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

* The paramsparameter defineswhich CSS elements to animate.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("div").animate({right: '400px'});

});

});

</script>

</head>

<body>

<button>Double click to begin animation</button>

<div style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>

</body>

</html>

This example moves the***<div>***element to its right till it reaches its right property of 250 px.

* The default position of HTML elements is static, which means they cannot be moved. Therefore, before animating your page, make sure that you change the CSS position property to fixed, absolute, or relative.

You can use a lot of properties with animate() method, which are listed below:

**jQuery animate()- manipulate multiple properties using relative values**

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("div").animate({

right: '350px',

height: '+=250px',

width: '+=250px'

});

});

});

</script>

</head>

<body>

<button>double click here to begin the animation</button>

<div style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>

</body>

</html>

This example animates a ***<div>***element by manipulating multiple properties using relative values. On double clicking the button, the box will start to move to its right till it reaches its right property of 350 px, while its height and width will correspondingly grow to 250 px each.

The “+=” means that the ***<div>***element will continue to increase in its width and height by a factor of 250 px each, every time you double click the button.

**jQuery animate() – using pre-defined values**

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("div").animate({

height: 'toggle'

});

});

});

</script>

</head>

<body>

<button>double click here for animation to begin</button>

<div style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>

</body>

</html>

This example shows a ***<div>*** element. On double clicking this element, the element will disappear. When you click it again, it will reappear again. This is because of the toggle condition which is associated with its height.

**jQuery Stop**

jQuery stop() method stops all the effects before its completion. It stops the slide, fade and animation effects.

**Syntax:**

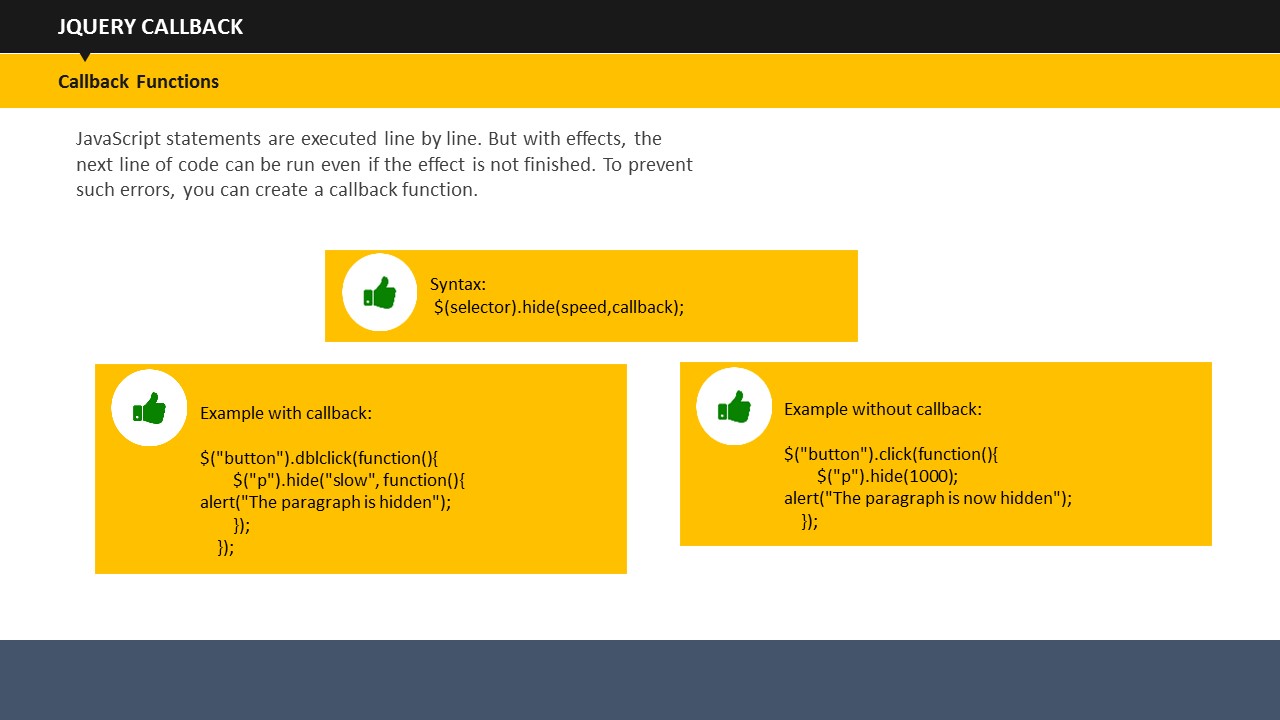
$(selector).stop(stopAll, goToEnd);

The stopAll parameter is optional. It specifies whether the animation queue should be cleared or not. Its default value is false.

The other optional goToEnd parameter determines whether to complete the ongoing animation immediately or not. Its default value is also false.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 5: jQuery Callback**

Like many other languages, JavaScript executes its code by following line by line execution. But there can be instances when one effect can run longer than the effect that is to follow it. In such cases, you may not get the desired output or get errors. callback() method is used to resolve this very problem. It is executed only after the current effect is completed. See the following example to understand it.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("p").hide("slow", function(){

alert("The paragraph is hidden");

});

});

});

</script>

</head>

<body>

<button>Hide</button>

<p>This is a test paragraph.</p>

</body>

</html>

In this example, there is a paragraph which gets hidden upon double clicking the button. The hiding effect will take one second. When the effect finishes, an alert message pops up which reads “the paragraph is hidden”.

Suppose, if you have not used the callback() method, your code would look like this:

$("button").click(function(){

$("p").hide(1000);

alert("The paragraph is now hidden");

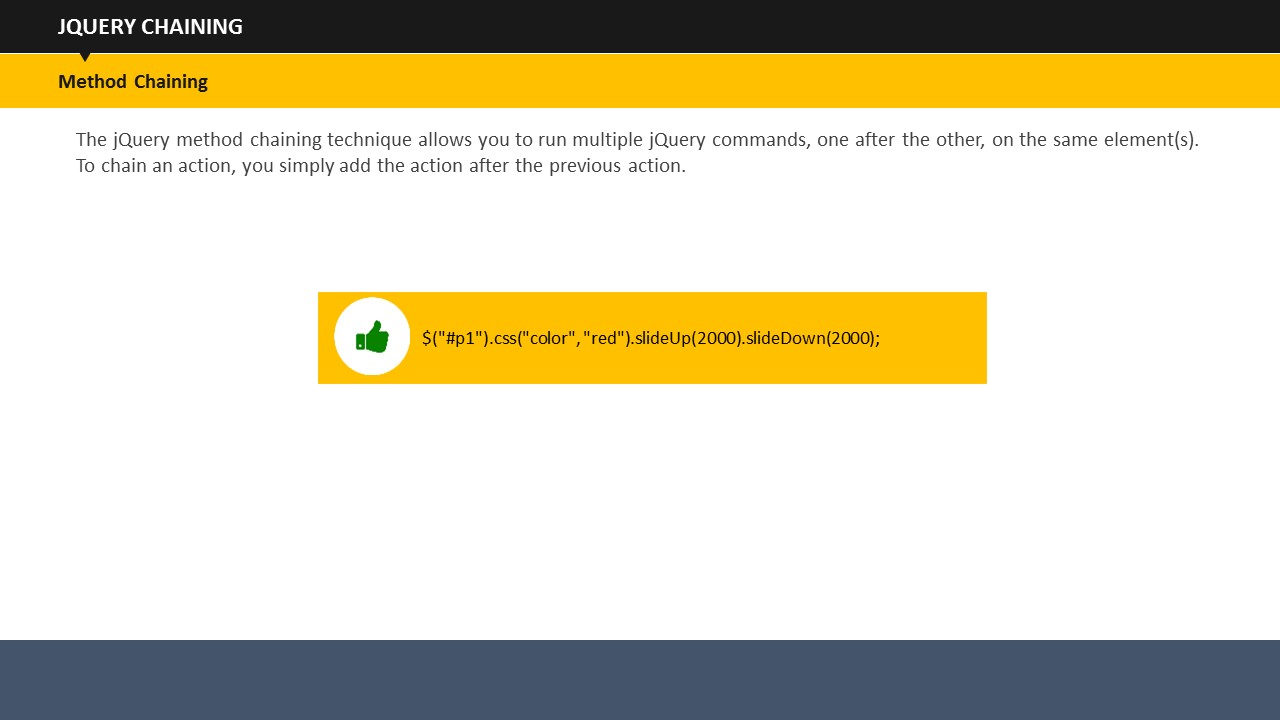
});

In this example, the alert message would appear straight after the double click, which is not the desired outcome.

We use callback() method to make sure that one effect will come in to play only after its preceding effect has fully taken place.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**SUBTOPIC 6: jQuery Chaining**

Up until now, we have been using jQuery statements one after the other. This gets a little too long. Resolving this, jQuery provides an option of Chaining in which we can run more than one jQuery methods on the same element.

All you need to do is to append a method to the previous method. jQuery provides an option to write chained codes either in a single line or you can spread them through multiple lines.

See the following example to understand it.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("#p1"). slideUp(2000).slideDown(2000);

});

});

</script>

</head>

<body>

<p id="p1">Chaining example</p>

<button>Click me</button>

</body>

</html>

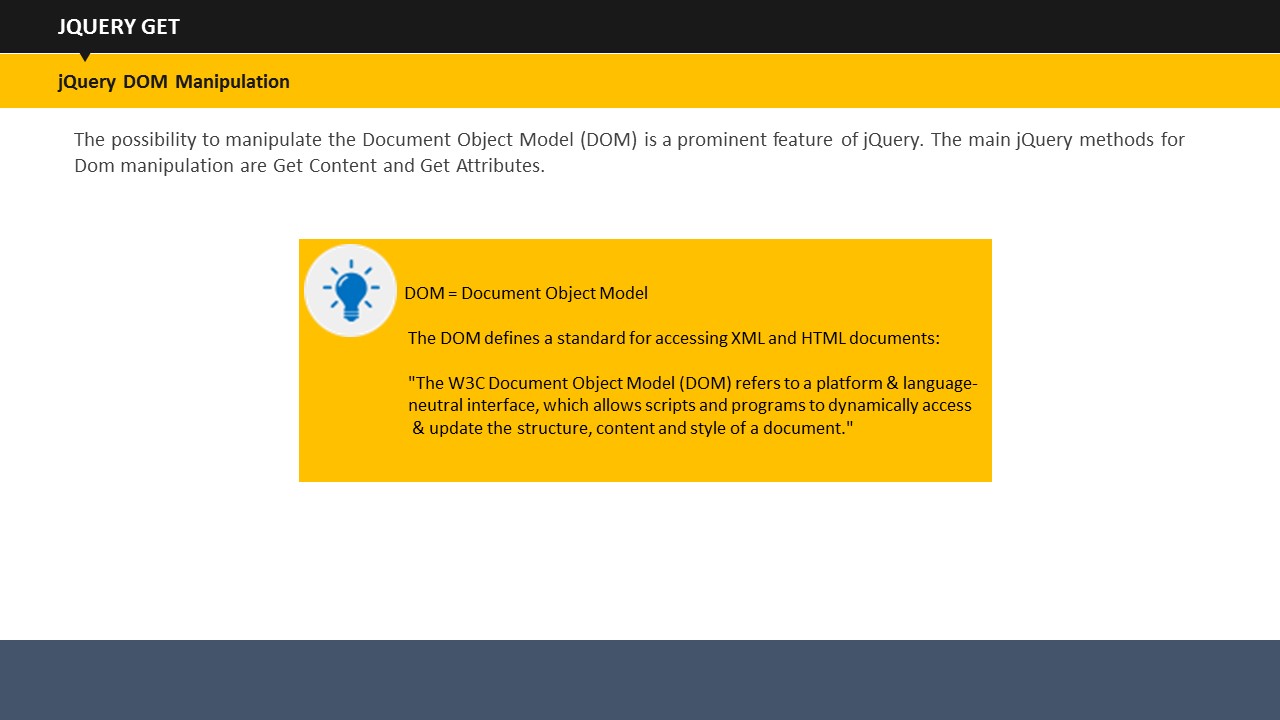
The example shows a paragraph with text “chaining example” with a button underneath. When you double click the button, the paragraph will slide up for two seconds hiding the paragraph and then it will automatically slid down revealing the paragraph.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

## 

**TOPIC 3: jQuery with HTML/CSS**



**SUBTOPIC 1: jQuery Get**

One very prominent feature of jQuery is the possibility to manipulate the Document Object Model (DOM). DOM defines a standard to access XML and HTML documents. Let us see important jQuery methods for DOM manipulation.

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

* Text() – it returns the text content of the selected elements
* Html() – it returns the content of the selected elements
* Val() – it returns the value of form fields

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("#bttn1").dblclick(function(){

alert("Text: " + $("#check").text());

});

$("#bttn2").dblclick(function(){

alert("HTML: " + $("#check").html());

});

});

</script>

</head>

<body>

<p id="check">there is some <b>bold</b> text here.</p>

<button id="bttn1">Display Text</button>

<button id="bttn2">Display HTML</button>

</body>

</html>

This example displays two buttons and a paragraph. When you double click on the “Display Text” button, the script would display the entire text within the ***<p>*** element sans ***<b>***and***</b>***tags. When you double click “Display HTML button, the same text will be shown including the tags.

Now see the example of val() method.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

alert("Value: " + $("#tcheck").val());

});

});

</script>

</head>

<body>

<p>Name: <input type="text" id="check" value="MohdShafi"></p>

<button>Display Value</button>

</body>

</html>

This example fetches the value from the form field. In this case, the value that will be shown is MohdShafi.

**Get attributes- attr()**

The attr() method returns the attribute values.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

alert($("#abc").attr("href"));

});

});

</script>

</head>

<body>

<p><a href="http://www.abc123.com" id="abc">abc123.com</a></p>

<button>Display href Value</button>

</body>

</html>

This example fetches the href value of the Website.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

## 

**SUBTOPIC 2: jQuery Set**

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

* Text() – it sets the text content of the selected elements
* Html() – it sets the content of the selected elements
* Val() – it sets the value of form fields

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("#btn1").dblclick(function(){

$("#check1").text("Hey users");

});

$("#btn2").dblclick(function(){

$("#tcheck2").html("<b>Hey users!</b>");

});

$("#btn3").dblclick(function(){

$("#tcheck3").val("Dolly Dew");

});

});

</script>

</head>

<body>

<p id="check1">This is a test paragraph.</p>

<p id="check2">This is another test paragraph.</p>

<p>Input field: <input type="text" id="check3" value="Mickey up"></p>

<button id="btn1">Set Text</button>

<button id="btn2">Set HTML</button>

<button id="btn3">Set Value</button>

</body>

</html>

In this example, here are three buttons – Set Text, Set HTML and Set Value- along with two paragraphs and one form field.

On double clicking the “Set Text” button, paragraph 1 will be replaced with the given text.

On double clicking the “set HTML”, paragraph 2 will be replaced with the given text.

On double clicking the “Set Value” button, form value will get changed as well.

**Set attributes- attr()**

Similarly, you can change the href as well as the title using attr() method.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("#abc").attr({

"href" : "http://www.abc123.com/456",

"title" : "ABC 123 456"

});

});

});

</script>

</head>

<body>

<p><a href="http://www.abc123.com" title="abc 123" id="abc">abc123.com</a></p>

<button> sethref and title</button>

<p>Point to the link to see the changed href attribute and newly set title value.</p>

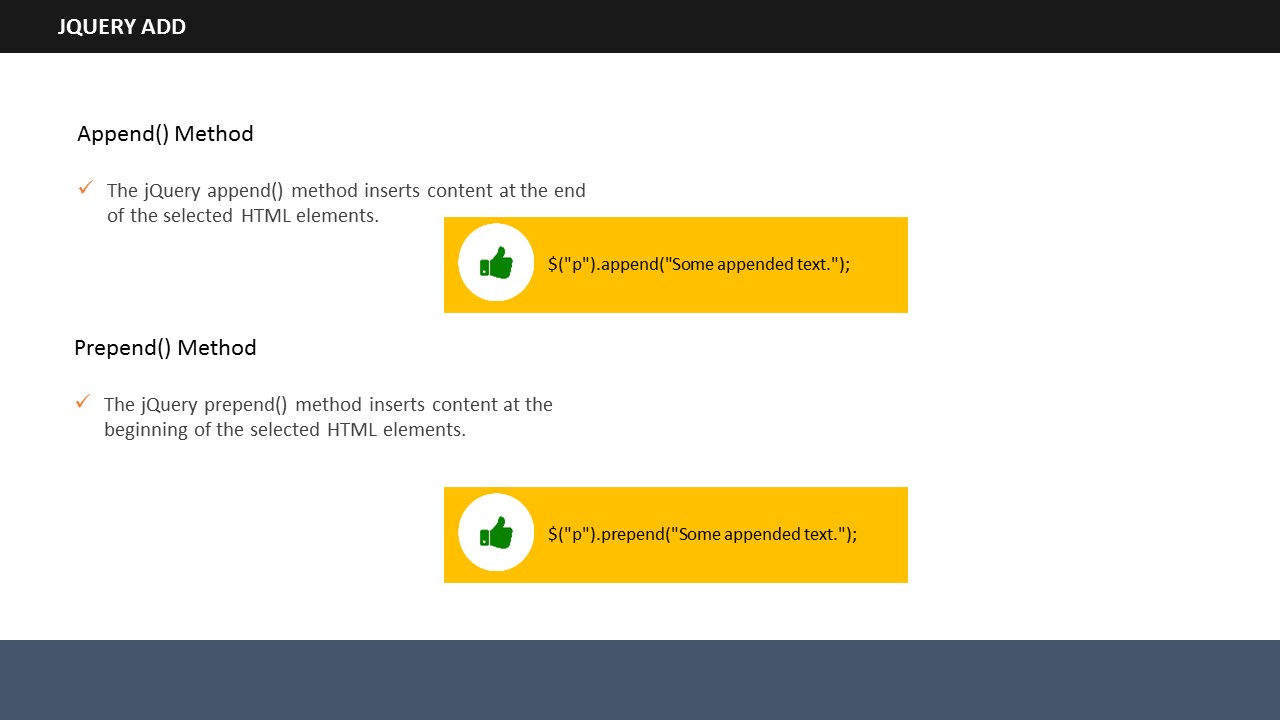
</body>

</html>

With the help of attr(), we have changed the href attribute and also reset the title of the URL.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**SUBTOPIC 3: jQuery- Add**

With jQuery, it is very easy to add new elements anywhere on the page. There are methods to do that, which are as follow:

* append()
* prepend()
* after()
* before()

Have a more detailed look at them now.

**append**()

Append() method is used to add a new element at the end of selected HTML elements.

**prepend**():

Prepend() method is used to add a new element at the end of beginning of selected HTML elements.

**before**()

Before() method is used to add a new element before selected HTML elements.

**after**()

After () method is used to add a new element after selected HTML elements.

You can add several elements using these methods. You can generate new elements with any one of the HTML, jQuery, or DOM.

**append() and prepend()**

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

functionappendText() {

var text1 = "<p>Hi.</p>";

var text2 = $("<p></p>").text("Hi.");

var text3 = document.createElement("p");

text3.innerHTML = "Hi";

$("body").append(text1, text2, text3);

}

</script>

</head>

<body>

<p>Hello users.</p>

<button ondblclick="appendText()">Append text</button>

</body>

</html>

In this example, three variable with text “Hi” are created using each of HTML, jQuery, and DOM. These variables are then appended after the paragraph “Hello users” after double clicking the button.

This works same for prepend() as well.

**before() and after()**

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

functionafterText() {

var text1 = "<b>I </b>";

var text2 = $("<i></i>").text("like ");

var text3 = document.createElement("b");

text3.innerHTML = "programming!";

$("img").after(text1, text2, text3);

}

</script>

</head>

<body>

<imgsrc="/images/w3jquery.gif" alt="jQuery" width="100" height="140">

<p>Double click the button to insert text after the image.</p>

<button ondblclick="afterText()">Insert after</button>

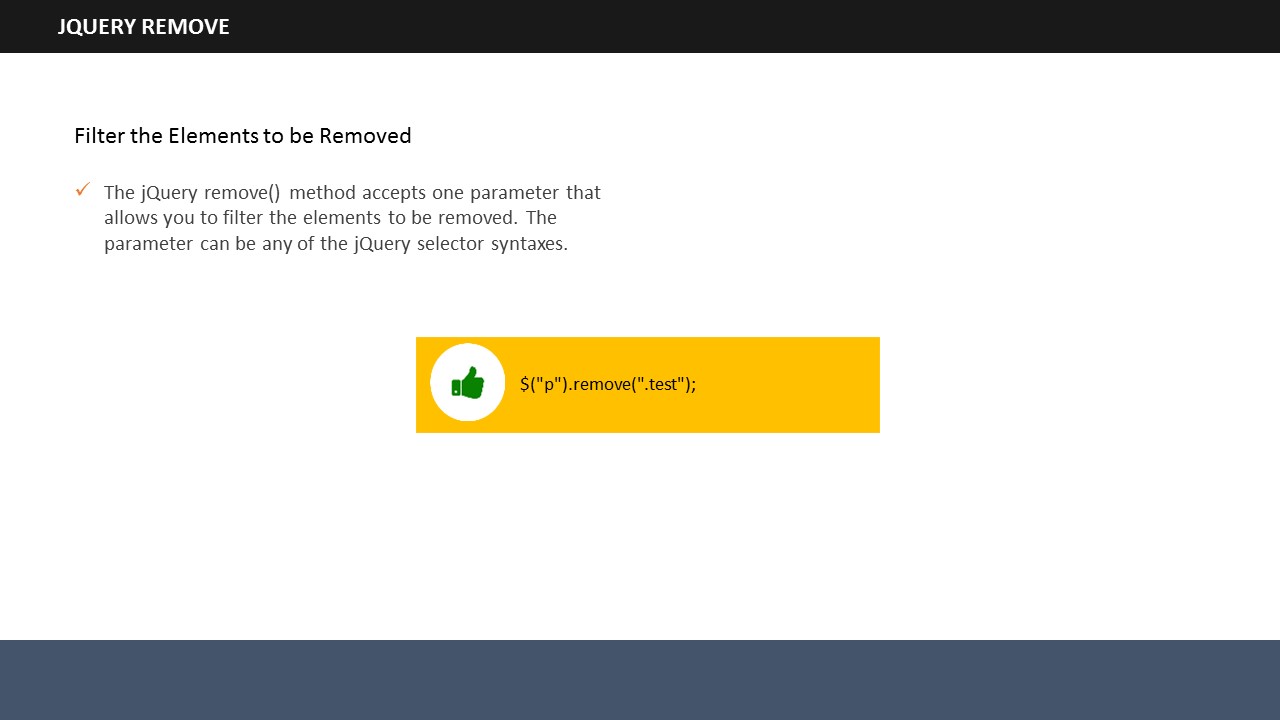
</body>

</html>

This example inserts text after an image. The same procedure applies for before() method as well.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 4: jQuery- Remove**

After learning how to add elements, it is time to know about how to remove them. jQuery provides two methods to remove elements – remove() and empty(). Take a look at each of them.

**jQuery remove() method**

The remove() method, as its name suggests, removes selected elements along with their child elements.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

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

});

});

</script>

</head>

<body>

<div id="div1" style="height:200px;width:400px;border:2px solid black;background-color:red;">

This is demo text in the div.

<p>This is a line in the div.</p>

<p>This is another line in the div.</p>

</div>

<br>

<button>Remove div element</button>

</body>

</html>

In this example, you can see there is a div comprising of three lines. When you double click on the “remove div element”, the entire div will be removed. Here, the selected element was div and the paragraphs were its child elements, all of which are removed due to the remove() method.

**jQuery empty() method**

The empty() method removes child elements of the selected elements.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

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

});

});

</script>

</head>

<body>

<div id="div1" style="height:200px;width:400px;border:1px solid black;background-color:red;">

This is demo text in the div.

<p>This is a line in the div.</p>

<p>This is another line in the div.</p>

</div>

<br>

<button>Empty the div element</button>

</body>

</html>

In this example, everything is same, except the method, which is empty() there. When you double click on the “Empty the div element”, only the child elements of div will be removed, and not the div.

**Removing the elements after filtering them**

You can remove elements by filtering as well. remove() method accepts parameter through which you can do the same. See the example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("p").remove(".check, .example");

});

});

</script>

<style>

.check {

color: yellow;

font-size: 25px;

}

.example {

color: red;

font-size: 30px;

}

</style>

</head>

<body>

<p>This is some text.</p>

<p class="check">This is class="check".</p>

<p class="check">This is class="test".</p>

<p class="example">This is class="example".</p>

<button>Remove elements with class="check" and class="example"</button>

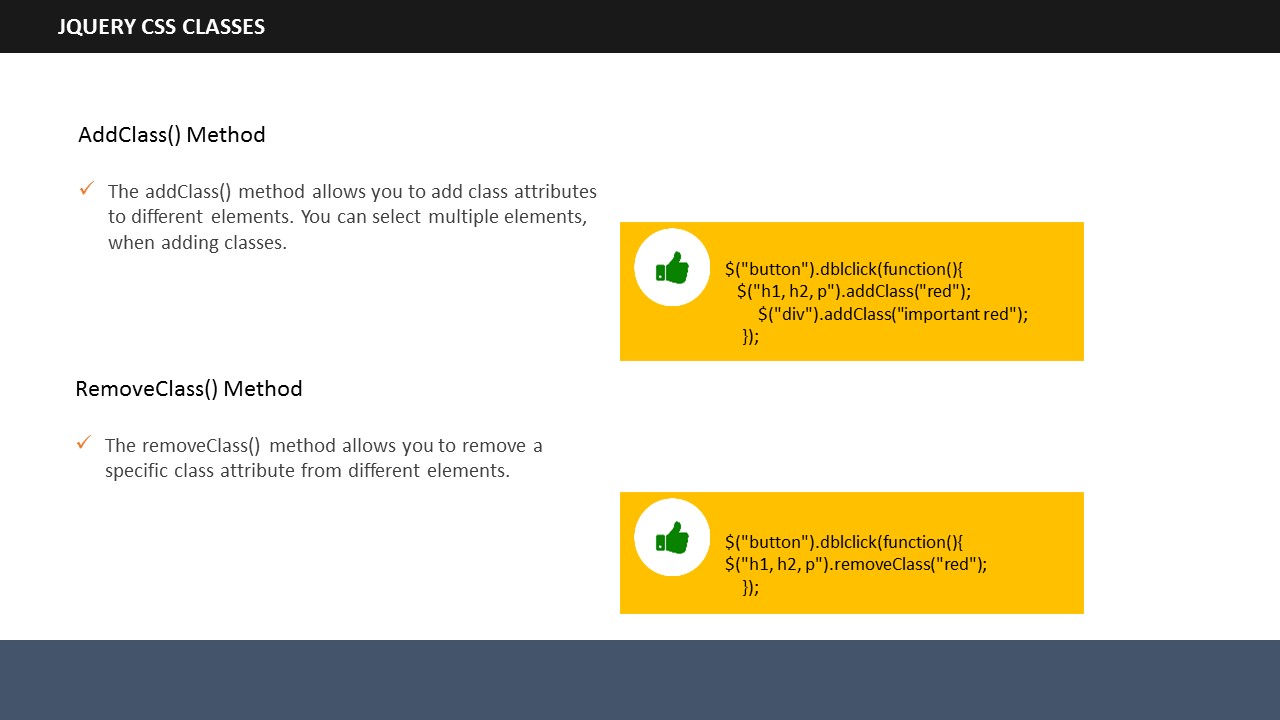
</body>

</html>

In this example, you can see all elements containing class= “check and class= “example” got removed.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 5: jQuery- CSS classes**

jQuery has simplified manipulating the CSS of elements. It has several methods through which we can manipulate the CSS. Let’s have a look.

* addClass()
* removeClass()
* toggleClass()

**Example:**

.important {

font-weight : bold;

font-size : large;

}

.red {

color : red;

}

This is a stylesheet that we are going to use to understand addClass(), removeClass(), and toggleClass().

**addClass()**

See the following example.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("h1, h2, p").addClass("red");

$("div").addClass("important red");

});

});

</script>

<style>

.important {

font-weight: bold;

font-size: large;

}

.red{

color: red;

}

</style>

</head>

<body>

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<p>This is a demo paragraph.</p>

<p>This is another demo paragraph.</p>

<div>This is demo text!</div><br>

<button>Add classes </button>

</body>

</html>

This example adds “important” as well as “red CSS” classes to the elements. The class important is added to heading 1, heading 2 and paragraph while both the classes are added to the ***<div>*** elements.

**removeClass()**

removeClass() method removes class attributes from the elements. See the following example.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("h1, h2, p").removeClass("red");

});

});

</script>

<style>

.important {

font-weight : bold;

font-size : large;

}

.red {

color : red;

}

</style>

</head>

<body>

<h1 class="red">Heading 1</h1>

<h2 class="red">Heading 2</h2>

<p class="red">This is a demo paragraph.</p>

<p>This is another demo paragraph.</p>

<button>Remove class from elements</button>

</body>

</html>

This example displays two headings and two paragraphs. Both headings and the first paragraph use the class red. removeClass() method removes the red class attribute from the headings and the first paragraph.

**toggleClass()**

You can toggle between adding and removing CSS classes with the help of toggleClass() method. Let us see an example.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("h1, h2, p").toggleClass("red");

});

});

</script>

<style>

.red {

color: red;

}

</style>

</head>

<body>

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<p>This is a demo paragraph.</p>

<p>This is another demo paragraph.</p>

<button>Toggle</button>

</body>

</html>

This example illustrates the toggleClass(). You add and remove the class red on the double click event of the mouse.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 6: jQuery- CSS()**

jQuery allows you to set or get multiple style properties of selected elements. jQuery css() method helps you do that.

**Return a CSS property**

**Syntax:**

Css(“property name”);

Let us see an example which will return the background color value. Remember that it will return the value of only the first matched element.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

alert("Background color = " + $("p").css("background-color"));

});

});

</script>

</head>

<body>

<h2>Heading Text</h2>

<p style="background-color:#00FF00">sample paragraph.</p>

<p style="background-color:#3366FF">sample paragraph.</p>

<p style="background-color:#CC99FF">sample paragraph.</p>

<button>Return background color of p</button>

</body>

</html>

The example returns the background color of the first paragraph, which was green. It returns the background color value of only green because it was the first matched element.

**Set multiple CSS properties**

You can set one or more CSS properties of elements in jQuery.

Syntax:

For setting one CSS property:

**css(“property name”, “value”);**

For setting multiple CSS properties:

**css({“property name” : “value”, “property name”: “value”,….});**

Following example will make you understand it better.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").dblclick(function(){

$("p").css({"background-color": "green", "font-size": "150%"});

});

});

</script>

</head>

<body>

<h2> Heading Text</h2>

<p style="background-color:#00FF00">sample paragraph.</p>

<p style="background-color:#3366FF">sample paragraph.</p>

<p style="background-color:#CC99FF">sample paragraph.</p>

<p>Demo paragraph.</p>

<button>Set multiple styles for p</button>

</body>

</html>

In this example, the background color and the font size of all the ***<p>*** elements are changed. We have changed two values, you can change many more and all at the same time.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



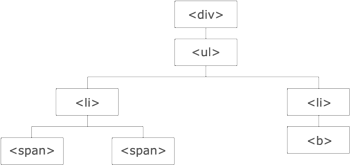
**TOPIC 3: jQuery Traversing**

## 

**SUBTOPIC 1: jQuery Traversing**

jQuery traversing is used to select or find HTML elements depending on their relation to other elements. It begins with one selection & move through until it reaches the desired elements.

See the following image and paragraph to understand it better.



The ***<div>*** element is the parent of ***<ul>***& an ancestor of all elements inside it.

The ***<ul>*** element is the parent of both ***<li>*** elements, child of ***<div>,*** and descendant of***<div>.***

The left ***<li>*** element is the parent of both ***<span>***, child of ***<ul>*** and a descendant of ***<div>.***

The two ***<span>*** elements are children of the left ***<li>*** and descendants of ***<ul>*** and ***<div>.*** They both are siblings with each other as they share the same parent.

The two ***<li>*** elements are siblings as they share the same parent.

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

The ***<b>*** element is a child of the right ***<li>*** and a descendant of ***<ul>*** and ***<div>.***

* Siblings share the same parents
* An ancestor can be a parent, grand-parent, great-grand-parent and so on
* A descendant can be a child, grand-child, great grand-child,and so on

**Traversing the DOM**

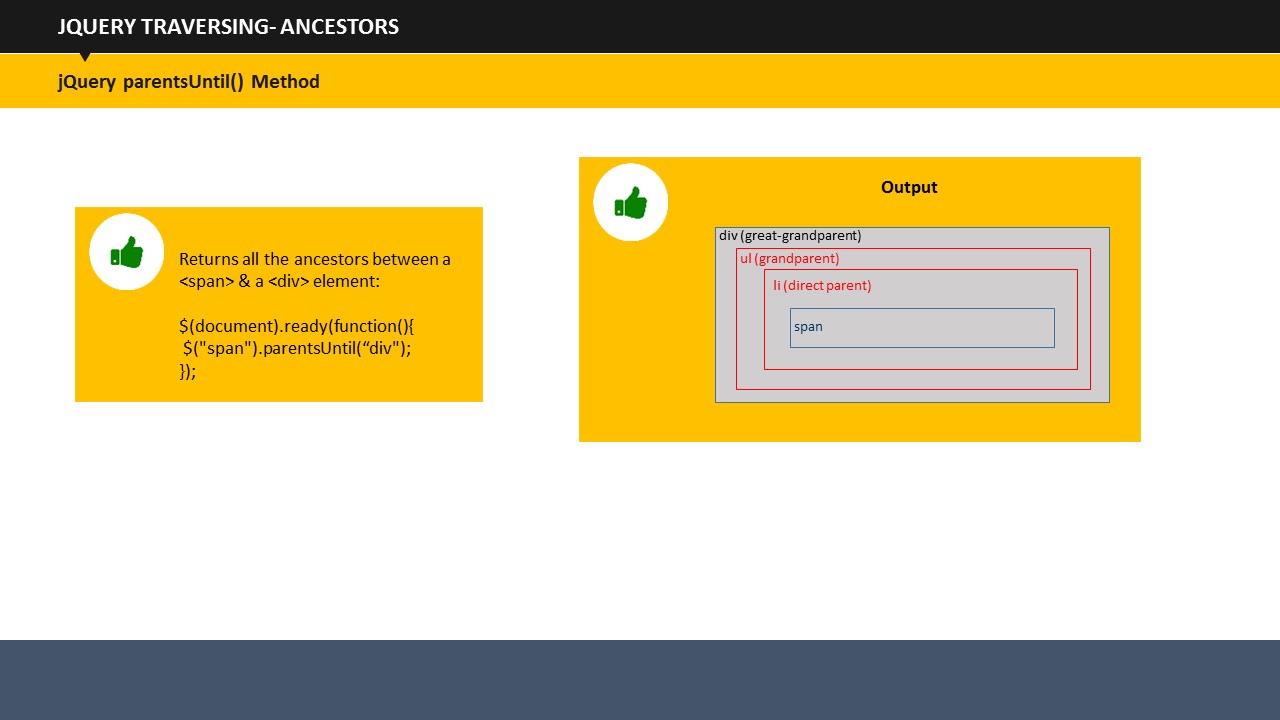
jQuery provides numerous methods to traverse the Document Object Model (DOM). You can easily traverse up, down and sideways in DOM through tree traversal, which is the largest category of traversal methods.

**Some useful jQuery traversing methods**

|  |  |
| --- | --- |
| **add() :** Add elements to the set of matched elements**:**  **addBack():** Adds the previous set of elements to the current set  **andSelf():** It is an alias for an addBack()  **children():** Returns all direct children  **closest() :** Returns the first ancestor  **contents() :** Returns all direct children  **each() :** Executes a function for each matched element  **end() :** Ends the most recent filtering operation in the current chain, and return the set of matched elements to previous state | **has() :** Returns all elements that have on element inside them  **is() :** Checks the set of matched elements against a selector/element/jQuery object, and return true if at least one of these elements matches the given arguments  **map()** : Passes each element in the matched set through a function, producing a new jQuery object containing the return values  **offsetParent() :** Returns the first positioned parent element  **slice() :** Reduces the set of matched elements to a subset specified by a range of indices |

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 2: jQuery Ancestors**

As you know by now, an ancestor can be a parent, grand-parent, great grand-parent and so on. You can make use of parent(), parents(), and parentsUntil() method to traverse up the DOM in order to find ancestors of an element. Let us have a look at each of them.

**jQuery parent() Method**

This method returns the immediate parent of the selected element. Since it deals only with direct parent, it only goes one level up of the selected element, and don’t go beyond it. See the example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.ancestors \* {

display: block;

border: 1.5px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

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

});

</script>

</head>

<body>

<div class="ancestors">

<div style="width:600px;">div (great-grandparent)

<ul>ul (grandparent)

<li>li (direct parent)

<span>span</span>

</li>

</ul>

</div>

<div style="width:600px;">div (grandparent)

<p>p (direct parent)

<span>span</span>

</p>

</div>

</div>

</body>

</html>

Here, we wanted to find the parent of span. So, what we did was we changed some CSS properties of the parent of ***<span>***, so that it would show up. Its color and border color is changed from normal to red.

**jQuery parents() Method**

jQuery parents() method returns all the ancestor elements of the selected element. It traverses all through the root of the tree to find and return all ancestors. An example will help you understand it better.

**Example:**

<html>

<head>

<style>

.ancestors \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("span").parents().css({"color": "red", "border": "1.5px solid red"});

});

</script>

</head>

<body class="ancestors">body (great-great-grandparent)

<div style="width:600px;">div (great-grandparent)

<ul>ul (grandparent)

<li>li (direct parent)

<span>span</span>

</li>

</ul>

</div>

</body>

</html>

The following example would return ancestors of***<span>***elements. The code changed the CSS properties of all parents of ***<span>*** elements to make them stand out.

**jQueryparentsUntil() Method**

We have traversed direct parent; we have traversed all the ancestor elements. But what if we want to look for a parent only two levels up of the selected elements. For that, jQuery has a parentsUntil() Method. It returns ancestor elements between two arguments.

Check the example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.ancestors \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("span").parentsUntil("body").css({"color": "red", "border": "1.5px solid red"});

});

</script>

</head>

<body class="ancestors"> body (great-great-grandparent)

<div style="width:600px;">div (great-grandparent)

<ul>ul (grandparent)

<li>li (direct parent)

<span>span</span>

</li>

</ul>

</div>

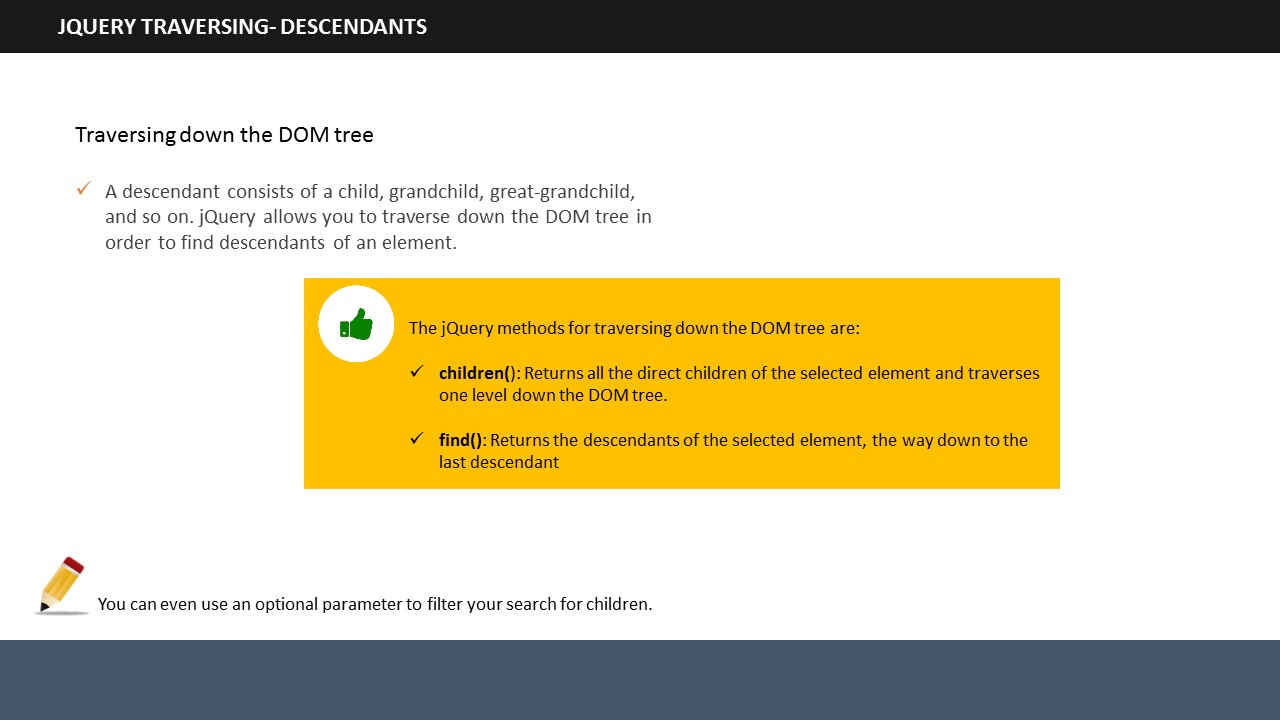
</body>

</html>

This example searches for all the parents of ***<span>*** from its immediate parent till the body. In this case, li, ul and, and div are the results of the code.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 3: jQuery Descendants**

As you know by now, a descendant can be a child, grand-child, great-grandchild, and so on. You can make use of children() and find() Method to traverse down the DOM in order to find descendants of an element. Have a look at each of them to understand it more.

**jQuery children() method**

Similar to parent() Method, children() Method also searches down only one level, and return the immediate children of the selected element. See its example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.descendants \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("div").children().css({"color": "red", "border": "1.5px solid red"});

});

</script>

</head>

<body>

<div class="descendants" style="width:500px;">div (current element)

<p>p (child)

<span>span (grandchild)</span>

</p>

<p>p (child)

<span>span (grandchild)</span>

</p>

</div>

</body>

</html>

This example returns the direct children of each of the ***<div>*** elements.

You can also filter the method to get a particular child.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.descendants \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("div").children("p.2").css({"color": "red", "border": "1.5px solid red"});

});

</script>

</head>

<body>

<div class="descendants" style="width:600px;">div (current element)

<p class="1">p (child)

<span>span (grandchild)</span>

</p>

<p class="2">p (child)

<span>span (grandchild)</span>

</p>

</div>

</body>

</html>

In this example, the child search is filtered to get exactly what we wanted, and not anything else, as we got in the previous example.

**jQuery find() Method**

The find() method returns all the descendants of the selected element. It searches till the last descendant. See the example below.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.descendants \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("div").find("\*").css({"color": "red", "border": "1.5px solid red"});

});

</script>

</head>

<body>

<div class="descendants" style="width:600px;">div (current element)

<p>p (child)

<span>span (grandchild)</span>

</p>

<p>p (child)

<span>span (grandchild)</span>

</p>

</div>

</body>

</html>

This example returns all the descendants of ***<div>***elements.

One can look for a particular descendant by changing the syntax a little. Let us see ho**Example:**

$(document).ready(function(){

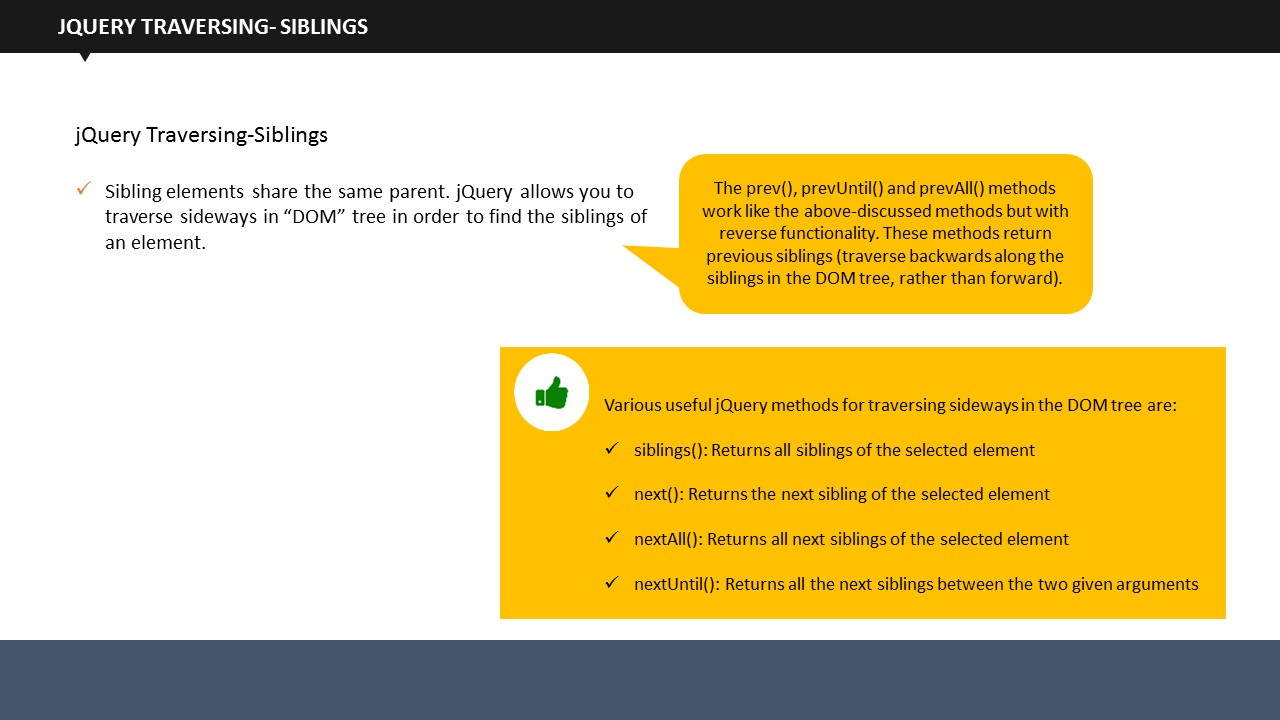
$("div").find("span").css({"color": "red", "border": "1.5px solid red"});

});

Instead of \*, I wrote span. Initially the code was looking for all descendants of the ***<div>*** element, but with this modification, the program will return only **<span>** elements.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |



**SUBTOPIC 4: jQuery Siblings**

As I have said, siblings share the same parent.

So far, we have traversed up and down to find ancestors and descendants. Now we are going to traverse sideways to search for siblings.

jQuery has provided us a variety of methods to traverse for siblings. You can use siblings(), next(), nextAll(), nextUntil(), prev(), prevAll(), and prevUntil() Methods to traverse sideways in a DOM tree. They are described below.

**jQuery siblings() Method**

Simplest of them all, jQuerysiblings() return all siblings of selected element.

An example would clarify it better. Have a look.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.siblings \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("h2").siblings().css({"color": "red", "border": "1.5px solid red"});

});

</script>

</head>

<body class="siblings">

<div>div (parent)

<p>p</p>

<span>span</span>

<h2>h2</h2>

<h3>h3</h3>

<p>p</p>

</div>

</body>

</html>

This example returns all siblings of <h2>.

**jQuery next()**

jQuery next() Method, as the name suggests, returns the next sibling of the selected element. Let us see an example.

**Example:**

$(document).ready(function(){

$("h2").next().css({"color": "red", "border": "1.5px solid red"});

});

Rest of the code is similar to the previous example. The only change is the usage of next() Method in place of siblings() Method. This code will return only the next sibling of <h2>.

**jQuery nextAll()**

jQuerynextAll() Method returns all next siblings of the selected element.

**Example:**

$(document).ready(function(){

$("h2").nextAll().css({"color": "red", "border": "1.5px solid red"});

});

Rest of the code remains the same, except the method. The nextAll() method here returns all the next elements of ***<h2>***.

**jQuery nextUntil()**

The jQuery nextUntil() Method returns all the sibling elements between two given arguments.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.siblings \* {

display: block;

border: 1px solid lightgrey;

color: grey;

padding: 4px;

margin: 12px;

}

</style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("h2").nextUntil("h6").css({"color": "red", "border": "2px solid red"});

});

</script>

</head>

<body class="siblings">

<div>div (parent)

<p>p</p>

<span>span</span>

<h2>h2</h2>

<h3>h3</h3>

<h4>h4</h4>

<h5>h5</h5>

<h6>h6</h6>

<p>p</p>

</div>

</body>

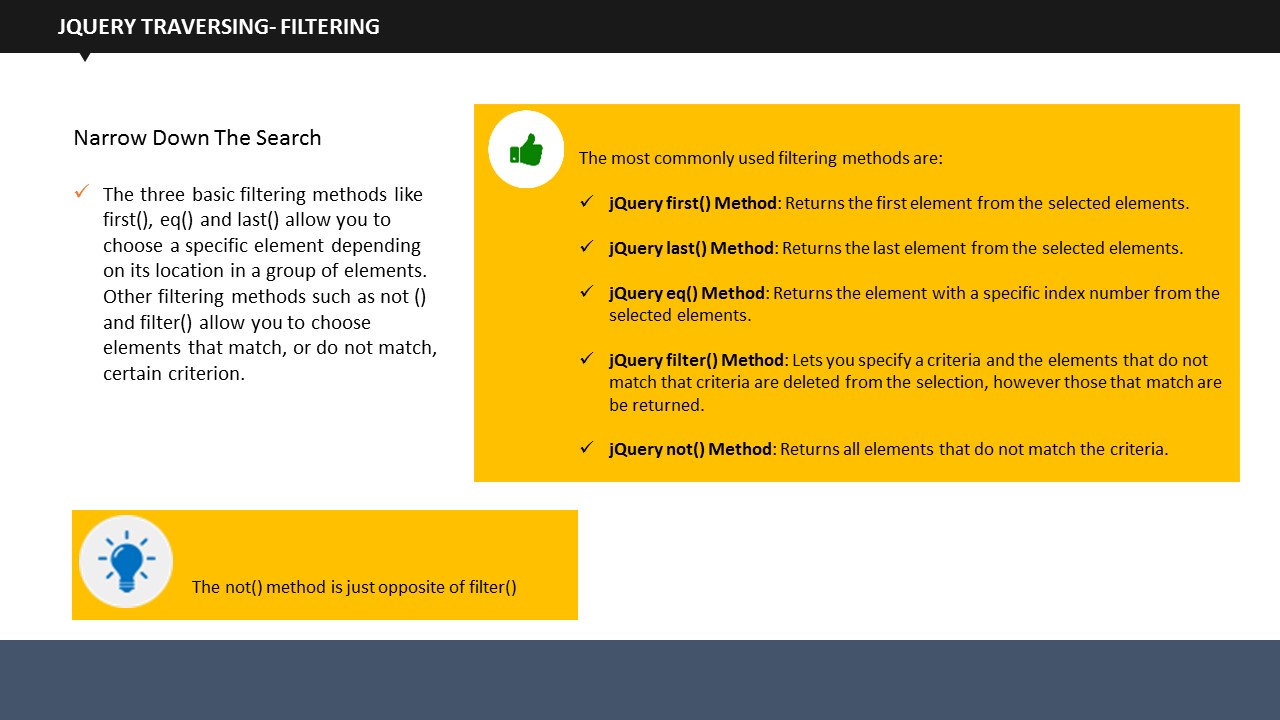
</html>

The nextUntil() Method will display all the sibling elements between ***<h2>*** and ***<h6>***. So, the final output which you can see on screen is ***<h3>***, ***<h4>*** and ***<h5>*** being colored in red.

The prev(), prevAll(), and prevUntil() Methods work the same way as next(), nextAll(), and nextUntil(), except that they work in reverse functionality. In short, they traverse backwards in the Dom tree rather than forward.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |



**SUBTOPIC 5: jQuery Traversing – Filtering**

jQuery provides three basic filtering methods, which are first(), last(), and eq(). They allow you to select particular elements basis on their positions within the group of elements.

Other methods namely filter() and not() are used to select elements which either match or do not match a specific criteria.

**jQuery first() Method**

The jQuery first() method selects the first element among the selected elements. Check the example.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

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

});

</script>

</head>

<body>

<h1>This is my homepage</h1>

<p>This is a demo paragraph in body.</p>

<div style="border: 1px solid black;">

<p>This is a demo paragraph in a div.</p>

<p>This is another demo paragraph in a div.</p>

</div><br>

<div style="border: 2px solid black;">

<p>This is the a demo paragraph in another div.</p>

<p>This is another demo paragraph in another div.</p>

</div>

<p>This is the final paragraph in body.</p>

</body>

</html>

The program takes two ***<div>***elements within a body. Both contain some paragraphs. Our code searches for the first ***<p>*** element of the first ***<div>***, and display the same in the output.

**jQuery last()**

Similar to first() Method, last() Method returns the last elements among selected elements. Let us take the same example which we just taken for the first() Method.

**Example:**

$(document).ready(function(){

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

});

Rest of the code is same, only the Method is changed. The output of this program returns the last ***<p>*** element of the last ***<div>***.

**jQuery eq() Method**

jQuery eq() Method returns the index number of the selected element.

* First element holds an index value of 0, and not 1

An example would clarify it better.

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("p").eq(2).css("background-color", "yellow");

});

</script>

</head>

<body>

<h1>This is my homepage</h1>

<p>I am Donald (index 0).</p>

<p>Donald Darwin= (index 1).</p>

<p>I live in Dhaka (index 2).</p>

<p>I love Cricket a lot (index 3).</p>

</body>

</html>

This example filters the given text for index value 2. When the code runs, the ***<p>***element with an index value 2 will be highlighted. In this case, it is “I live in Dhaka”, which is being highlighted, as you can see.

**jQuery filter() Method**

The jQuery filter() Method allows web developers to filter elements based on a criteria. If the elements match the criteria, they are returned, and those that don’t, they are removed from selection. Check the example..

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

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

});

</script>

</head>

<body>

<h1>This is Homepage</h1>

<p>My name is Darwin.</p>

<p class="introduce">I live in Dhaka.</p>

<p>I love Dhaka.</p>

<p class="introduce">I love cricket a lot.</p>

</body>

</html>

</body>

</html>

This example filters the ***<h>*** and the ***<p>*** elements for class “introduce”, and changed the background- color of those that matched the criteria. Only two ***<p>*** elements matched and they are displayed, while rest of them were removed from selection.

**jQuery not() Method**

In contrast to the filter() Method(), the not() Method returns those values that do not match the criteria while removing those that do. See the following example.

**Example:**

$(document).ready(function(){

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

});

If we look at the previous example, if we write not() instead of filter() in there, we get “My name is Darwin” and “I love Dhaka” as output.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |

 **Objectives Mapping Activity:**

 **Duration:** 5 Minutes

You can refer to the checklist given here to map the objectives outlined at the beginning of the session with the learnings from this session. This will help you analyze whether the learning objectives have actually been met or not. Check the boxes if the objectives are met.

 What is jQuery, its syntax, selectors and various events

Different jQuery effects

How to use jQuery with HTML and CSS

jQuery traversing

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**TOPIC 5: Summary**

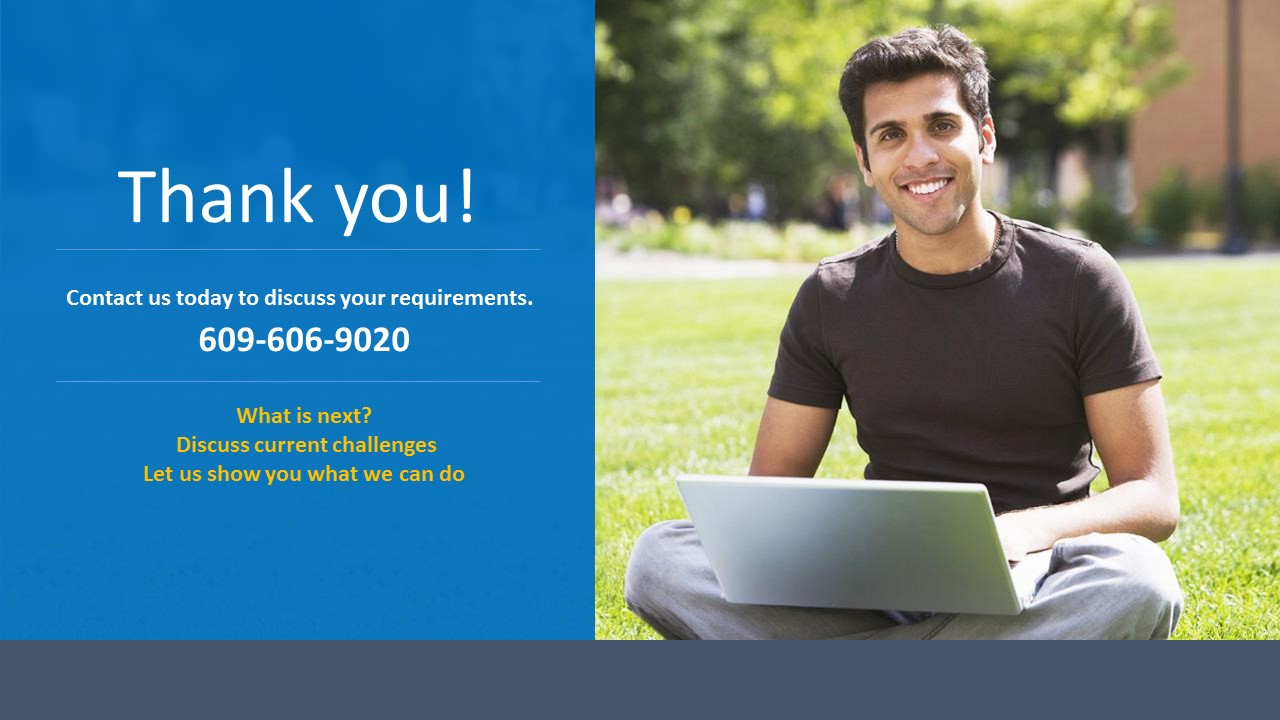
 **Summarization activity**

**Duration:** 10 Minutes

It’s time to review what we have learned so far. Write in the space given in this guide, what you have learned today.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |



This is the end of this session on jQuery.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |



**ITES Functional Skills Training**

# LAB EXERCISES

**Exercise 1:Toggle between hiding & showing the <p> element when you click on the "Toggle" button.**

**Edit this code:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("button").click(function(){

$("p").method();

});

});

</script>

</head>

<body>

<button>Toggle</button>

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

</body>

</html>

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**Exercise 2: When the mouse pointer enters the <span> element, it should be hidden. Use the correct event to do so.**

**Edit this code:**

<!DOCTYPE html>

<html>

<head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

$("selector").event(function(){

$(this).hide();

});

});

</script>

</head>

<body>

<span>If you mouse over me, I will disappear.</span><br>

<span>If you mouse over me, I will disappear.</span>

</body>

</html>

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**Exercise 3: Write a program to add a blue color to all sibling elements between <h2> and <h6>.**

**Hint:** Use the nextUntil() method.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

**Exercise 4: Write a program to set the font-size of <div> to 100 pixels using the animate() method.The duration of the effect should be "slow".**

**Hint:**  $("element").animate({property:"value"},speed);

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**Exercise 5: Write a program to add a red color to third <p> element.**

**Hint:** Use the eq() method, and remember that index numbers start at 0.

**Your space:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |