Unit III
jQuery

- JQuery: Introduction
- Loading JQuery,
- Selecting elements,
- changing styles,
- creating elements,
- appending elements,
- removing elements,
- handling events.

jQuery Introduction

- jQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: Write less, do more.
- It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler
- Prerequisite:
  - HTML
  - CSS
  - JavaScript
- jQuery is a lightweight, "write less, do more", JavaScript library.
- Many of the biggest companies on the Web use jQuery, such as:
  - Google
  - Microsoft
  - IBM
  - Netflix

core features supported by jQuery:

- HTML manipulation
- CSS Manipulation
- DOM Selection: jQuery provides Selectors to retrieve DOM element based on different criteria like tag name, id, css class name, attribute name, value, nth child in hierarchy etc.
- DOM Manipulation: You can manipulate DOM elements using various built-in jQuery functions. For example, adding or removing elements, modifying html content, css class etc.
- Special Effects: You can apply special effects to DOM elements like show or hide elements, fade-in or fade-out of visibility, sliding effect, animation etc
- Events: jQuery library includes functions which are equivalent to DOM events like click, dblclick, mouseenter, mouseleave, blur, keyup, keydown etc.
- AJAX Support: The jQuery helps you a lot to develop a responsive and feature rich site using AJAX technology.
- Animations: The jQuery comes with plenty of built-in animation effects which you can use in your websites.
- Lightweight: The jQuery is very lightweight library - about 19KB in size (Minified and gzipped).
- Cross Browser Support: The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+

Advantages of jQuery

- Easy to learn: jQuery is easy to learn because it supports same JavaScript style coding.
- Write less do more: jQuery provides a rich set of features that increase developers' productivity by writing less and readable code.
- Excellent API Documentation: jQuery provides excellent online API documentation.
- Cross-browser support: jQuery provides excellent cross-browser support without writing extra code.

Adding jQuery to Your Web Pages

- There are two ways to use jQuery.

1) Local Installation – You can download jQuery library on your local machine and include it in your HTML code.

- The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):
- <head>
  <script src="jquery-3.3.1.min.js"></script>
  </head>

jQuery CDN

2) CDN Based Version – You can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

- Both Google and Microsoft host jQuery.
- Google CDN:
- <head>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  </head>
- Microsoft CDN:
- <head>
  <script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js"></script>
  </head

jQuery Syntax

- With jQuery you select (query) HTML elements and perform "actions" on them.
- jQuery Syntax
  - The jQuery syntax is tailor-made for **selecting** HTML elements and performing some **action** on the element(s).
  - All jQuery selectors start with a dollor sign and parenthesis e.g. $(). It is known as the factory function.
- Basic syntax is: **$(*selector*).*action*()**
  - A $ sign to define/access jQuery
  - A (*selector*) to "query (or find)" HTML elements
  - A jQuery *action*() to be performed on the element(s)
- Examples: jQuery uses CSS syntax to select elements
  - $(this).hide() - hides the current element.
  - $("p").hide() - hides all <p> elements.
  - $(".test").hide() - hides all elements with class="test".
  - $("#test").hide() - hides the element with id="test".

S.No.	Selector	Description
1)	Tag Name:	It represents a tag name available in the DOM. For example: $('p') selects all paragraphs 'p' in the document.
2)	Tag ID:	It represents a tag available with a specific ID in the DOM. For example: $('#real-id') selects a specific element in the document that has an ID of real-id.
3)	Tag Class:	It represents a tag available with a specific class in the DOM. For example: $('.real-class') selects all elements in the document that have a class of real-class.

The Document Ready Event

- All jQuery methods are inside a document ready event:
- $(document).ready(function(){

  // jQuery methods go here...

  });
- This is to prevent any jQuery code from running before the document is finished loading (is ready).
- $(function(){

  // jQuery methods go here...

  });

```
$(document)  // finds global document object
```

```
$(document).ready()  // determines whether document object is ready or not
```

```
// specify callback function which will be execute once document is ready(loaded)
$(document).ready(function(){

// write jQuery code here to interact with DOM
})
```

- The $(document).ready() function determines when the full DOM hierarchy is loaded whereas window.onload event is raised when entire window is loaded including DOM, images, css and other required resources.
- The DOM loads before entire window loads.
- $(document).ready() doesn't wait for images to be loaded.

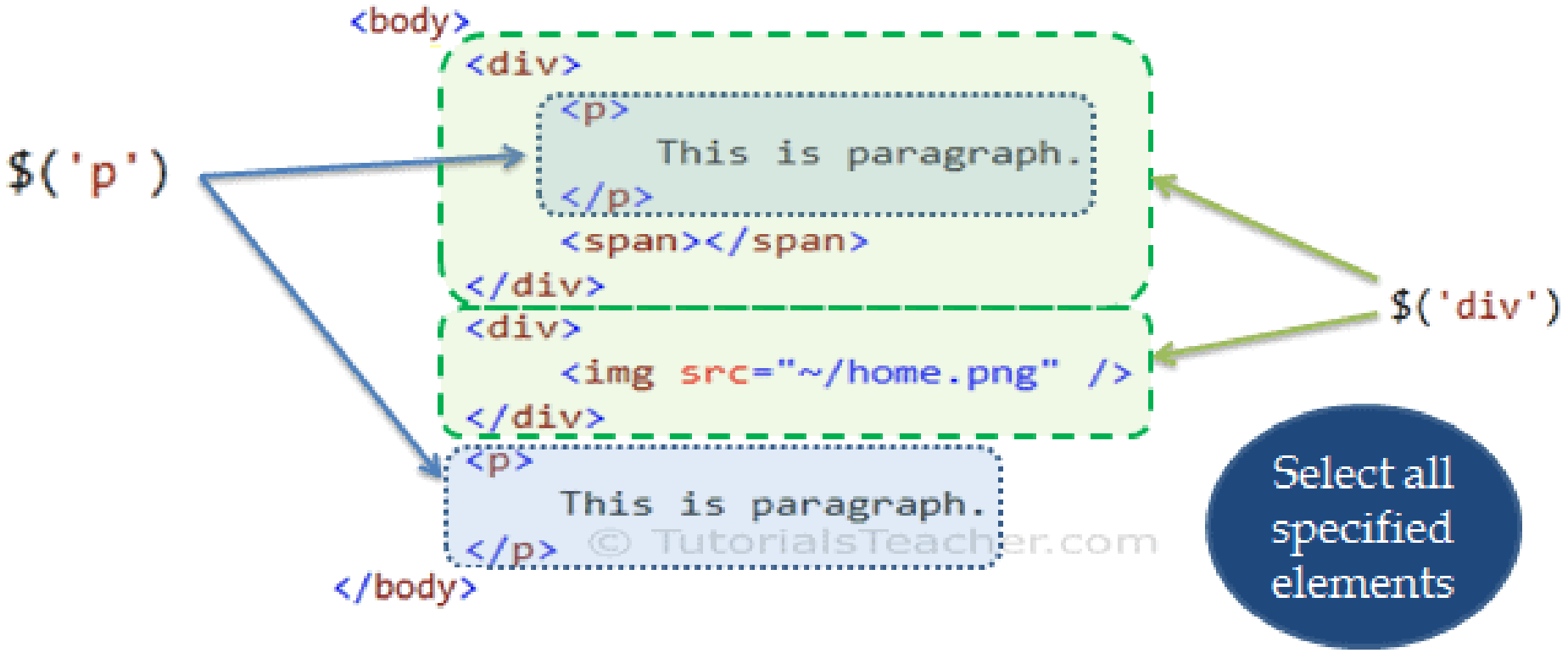
```
<!DOCTYPE html>
<html>
<head>
   <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"
>
   </script>
   <script>
      window.onload = function () {
         alert('window loaded');
      };

      $(document).ready(function () {
         alert('document loaded');
      });
   </script>
</head>
<body>
            <h1>Demo: window.onload() vs $(document).ready()</h1>
</body>
</html>
```

Then this text

First it print this

When you open the above html file in a browser,
first alert 'document loaded' is displayed then second alert 'windows loaded' is displayed.
Thus, DOM loads first followed by the entire window.

- jQuery Selectors
  - jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes.
  - All selectors in jQuery start with the dollar sign and parentheses: $()
- The element Selector
  - The jQuery element selector selects elements based on the element name.
  - You can select all <p> elements on a page like this:
    - $("p")
- $(document).ready(function(){
  $("button").click(function(){
  $("p").hide();
  });
  });
-

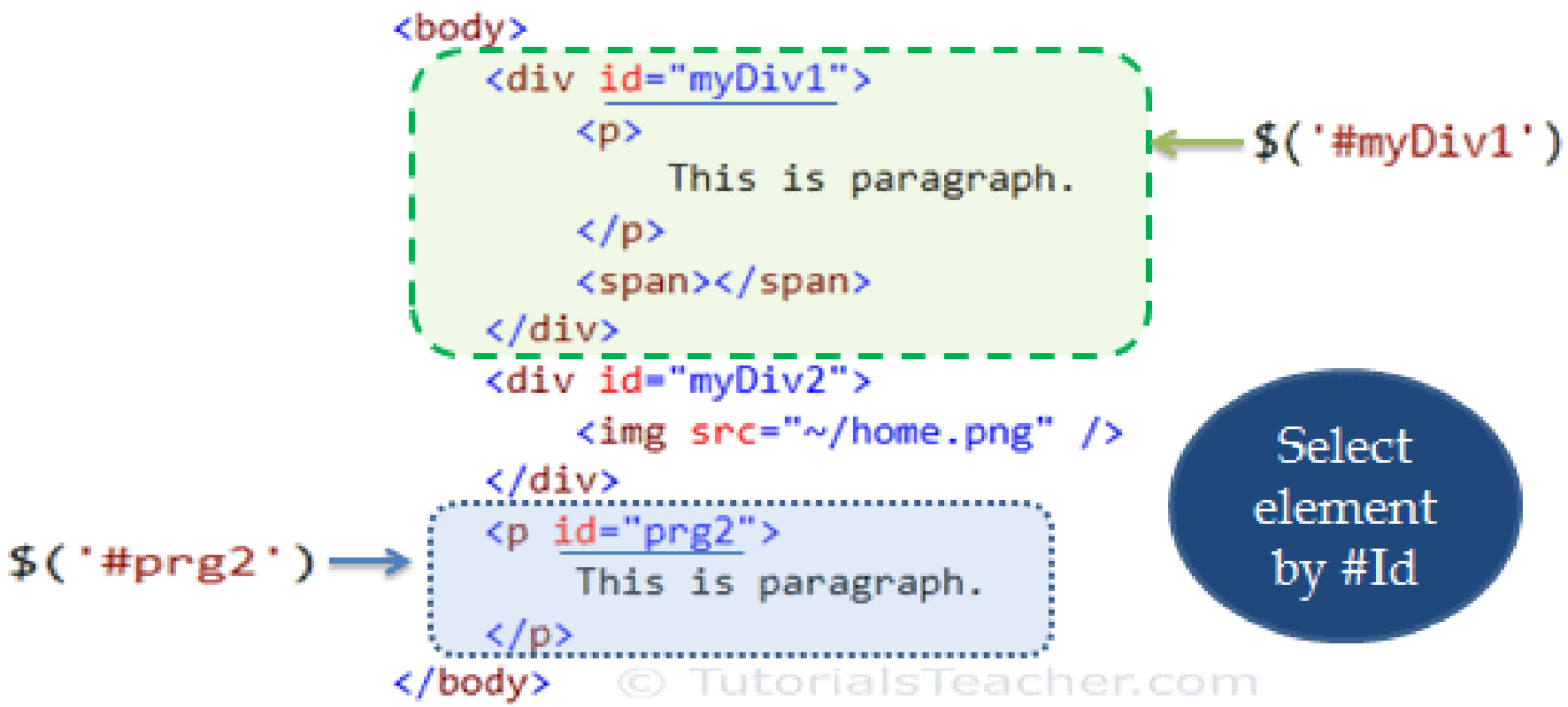

<body>

This is paragraph.

This is paragraph.

</body>
$('p')
$('div')
Select all specified elements
© TutorialsTeacher.com

- The #id Selector
  - The jQuery #id selector uses the id attribute of an HTML tag to find the specific element.
  - To find an element with a specific id, write a hash character, followed by the id of the HTML element: $("#test")
  - $(document).ready(function(){
    $("button").click(function(){
    $("#test").hide();
    });
    });


<body>

This is paragraph.

This is paragraph.

</body>
$('#myDiv1')
$('#prg2')
Select element by #Id
© TutorialsTeacher.com

- The .class Selector : The jQuery class selector finds elements with a specific class.
- To find elements with a specific class, write a period character, followed by the name of the class: $(".test")
- Ex:

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

Category	Selector	Description
Find element	$('div')	Find all elements
	$('p, div, code')	Find <p>, and <code> elements
Find descendant elements	$('div p')	Find all <p> elements which are descendants of
	$('div > p')	Find <p> which is child of
	$(*)	Find all elements
Find by Id	$('#myDiv')	Find element whose id is *myDiv*
	$('div#myDiv')	Find element whose Id is *myDiv*
	$('#myDiv1, #myDiv2')	Find multiple elements by id separated by comma.
Find by CSS class	$('.myCSSClass')	Find all the elements with *class=myCSSClass*.

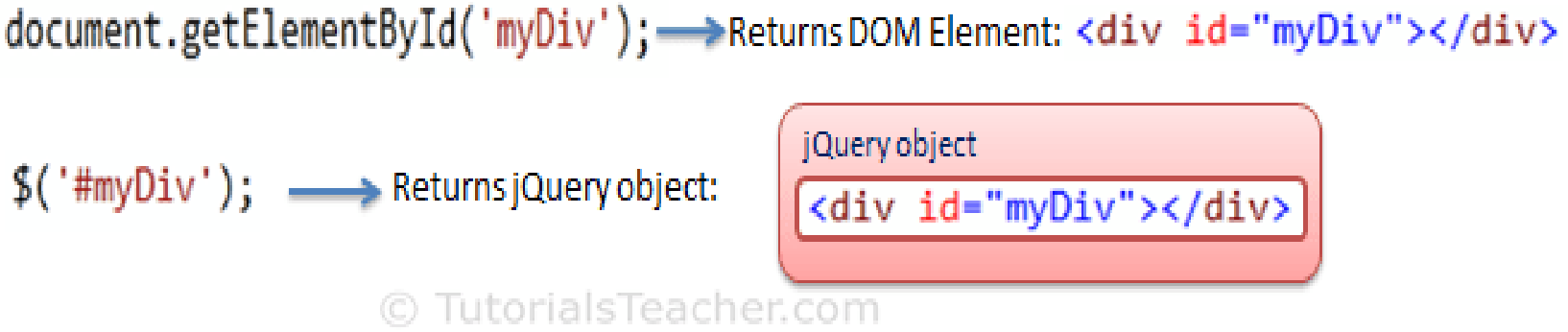
	$('.myCSSClass1, .myCSSClass2')	Finds all elements whose class attribute is set to *myCSSClass1* or *myCSSClass2*
	$('div.myCSSClass')	Finds all elements with *class=myCSSClass*
Find child element	$('p:first-child')	Find all <p> elements, which is the first child of its parent element. (parent element can be anything)
	$("p:last-child")	Selects all <p> elements which is the last child of its parent element. (parent element can be anything)
	$("p:nth-child(5)")	Selects all <p> elements which is the 5th child of its parent element. (parent element can be anything)
	$("p:nth-last-child(2)")	Selects all <p> elements which is the 2nd last child of its parent element. (parent element can be anything)

Find by input type	$(":input")	Selects all input elements.
:button	$(":button")	Selects all input elements where type="button".
:radio	$(":radio")	Selects all input types where type="radio"
:text	$(":text")	Selects all input elements where type="text" .
":checkbox"	$(":checkbox")	Selects all checkbox elements.
:submit	$(":submit")	Selects all input elements where type="submit".
:password	$(":password")	Selects all input elements where type="password".
:reset	$(":reset")	Selects all input elements where type="reset".
:image	$(':image')	Selects all input elements where type="image".
:file	$(':file')	Selects all input elements where type="file"

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	
$("ul li:first")	Selects the first element of the first	
$("ul li:first-child")	Selects the first element of every	
$("[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	

jQuery Methods

```
<html>
<body>
    <div id="myDiv"></div>
</body>
</html>
```



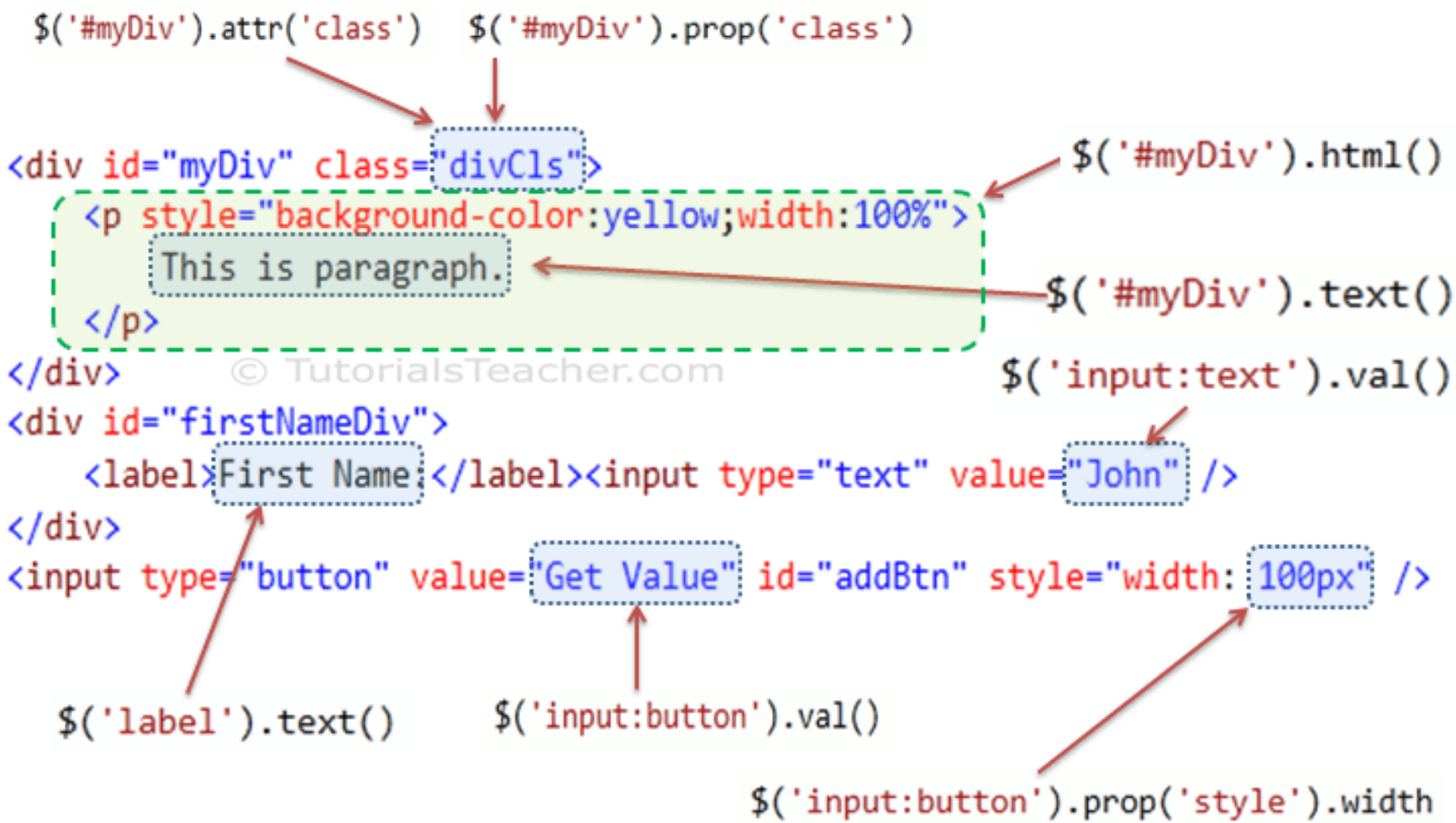
document.getElementById function returns div element
whereas jQuery selector returns jQuery object which is a wrapper around div element.
We have to call jQuery methods of jQuery object which is returned by jQuery selector.

Category	Description	Imp Methods
DOM Manipulation	These methods manipulate DOM elements in some manner e.g. changing attribute, style attribute, adding and removing elements etc.	after(), append(), attr(), before(), more..
Traversing	These methods help in navigating from DOM element to another element in a parent child hierarchy e.g. finding ancestors, descendants or sibling element of a specified element.	children(), closest(), each(), first(), next(), filter(), parent(), siblings(), more..
CSS	These methods get and set css related properties of elements.	addClass(), css(), hasClass(), removeClass(), toggleClass() more..

Attributes	These methods get and set DOM attributes of elements.	attr(), html(), removeAttr(), prop(), val(), more..
Events	These methods are used to handle DOM or JavaScript events.	bind(), blur(), change(), click(), dblclick(), focus(), keyup(), keydown(), more..
Effects	These methods are used to add animation to elements.	animate(), fadeIn(), fadeOut(), hide(), show(), stop(), more..

Manipulate HTML Attributes

- Get Content - text(), html(), and val()
- Three simple, but useful, jQuery methods for DOM manipulation are:
  - text() - Sets or returns the text content of selected elements
  - html() - Sets or returns the content of selected elements (including HTML markup)
  - val() - Sets or returns the value of specific target elements


$('#myDiv').attr('class')
$('#myDiv').prop('class')

This is paragraph.

© TutorialsTeacher.com

<label>First Name:</label><input type="text" value="John" />

<input type="button" value="Get Value" id="addBtn" style="width: 100px" />
$('#myDiv').html()
$('#myDiv').text()
$('input:text').val()
$('label').text()
$('input:button').val()
$('input:button').prop('style').width

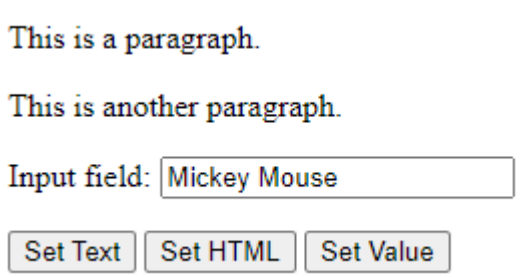
```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("#btn1").click(function(){
      alert("Text: " + $("#test").text());
   });
   $("#btn2").click(function(){
      alert("HTML: " + $("#test").html());
   });
});
</script>
</head>
<body>
<p id="test">This is some <b>bold</b> text in a paragraph.</p>
<button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>

</body>
</html>
```

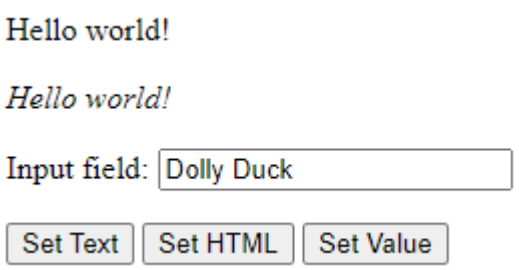
```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        alert("Value: " + $("#test").val());
    });
});
</script>
</head>
<body><p>Name: <input type="text" id="test" value="Mickey Mouse"></p>
<button>Show Value</button>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      alert($("#w3s").attr(“href"));
   });
});
</script>
</head>
<body>
<p><a href="https://www.w3schools.com" id="w3s">W3Schools.com</a></p>
<button>Show href Value</button>
</body>
</html>
```

- Set Content - text(), html(), and val() :use the previous three methods to **set content**:
- text() - Sets or returns the text content of selected elements
- html() - Sets or returns the content of selected elements (including HTML markup)
- val() - Sets or returns the value of form fields

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#btn1").click(function(){
        $("#test1").text("Hello world!");
    });
    $("#btn2").click(function(){
        $("#test2").html("<b>Hello world!</b>");
    });
    $("#btn3").click(function(){
        $("#test3").val("Dolly Duck");
    });
});
</script>
</head>
<body>
<p id="test1">This is a paragraph.</p>
<p id="test2">This is another paragraph.</p>
<p>Input field: <input type="text" id="test3" value="Mickey Mouse"></p>
<button id="btn1">Set Text</button>
<button id="btn2">Set HTML</button>
<button id="btn3">Set Value</button>
</body>
</html>
```





jQuery - Add Elements

- four jQuery methods that are used to add new content:

1. append() - Inserts content at the end of the selected elements
2. prepend() - Inserts content at the beginning of the selected elements
3. after() - Inserts content after the selected elements
4. before() - Inserts content before the selected elements

```
<div>

    <label>This is div.</label>

</div>

<p>     This is paragaph.    </p>
```

```
$('div').prepend('<p>This is prepended paragraph</p>');
                                        $('div').before('<p>This is new paragraph</p>');
<p>This is new paragraph</p>
<div>
    <p>This is prepended paragraph</p>
    <label>This is div.</label>
    <p>This is appended paragraph</p>
</div>
<p>This is new paragraph</p>
<p>This is paragaph.    </p>
                                        $('div').after('<p>This is new paragraph</p>');
$('div').append('<p>This is appended paragraph</p>');
```

jQuery - Remove Elements

- To remove elements and content, there are two jQuery methods:
  1. remove() - Removes the selected element (and its child elements):

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

     Ex: $("div1").remove();
  2. empty() - Removes the child elements from the selected element:

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

     Ex: $(“div1").empty();

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

     The parameter can be any of the jQuery selector syntaxes.
  - $("p").remove(".test");
    - removes all <p> elements with class="test"
  - $("p").r emove(".test, .demo");
    - removes all <p> elements with class="test" and class="demo"

jQuery Manipulating CSS

- addClass() - Adds one or more classes to the selected elements
- removeClass() - Removes one or more classes from the selected elements
- toggleClass() - Toggles between adding/removing classes from the selected elements
- css() - Get or set style properties to the specified element(s).

```
$('#myDiv').css('background-color', 'yellow');
```

```
<div id="myDiv">
    <p>
        This is first paragraph.
    </p>
    <div id="inrDiv">
        <p>This is second paragraph.</p>
    </div>
</div>
<div>
    <p>This is third paragraph.</p>
</div>
```

```
<div id="myDiv" style="background-color:yellow">
    <p>
        This is first paragraph.
    </p>
    <div id="inrDiv" class="myCSSClass myCSSClass2">
        <p>This is second paragraph.</p>
    </div>
</div>
<div>
    <p>This is third paragraph.</p>
</div>
```

```
$('#inrDiv').addClass('myCSSClass');
```

```
$('#inrDiv').toggleClass('myCSSClass2');
```

jQuery Manipulating CSS

- following stylesheet will be used for all the examples
- ```
  .important {
      font-weight: bold;
      font-size: xx-large;
  }

  .blue {
      color: blue;
  }
  ```

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

jQuery Manipulating CSS

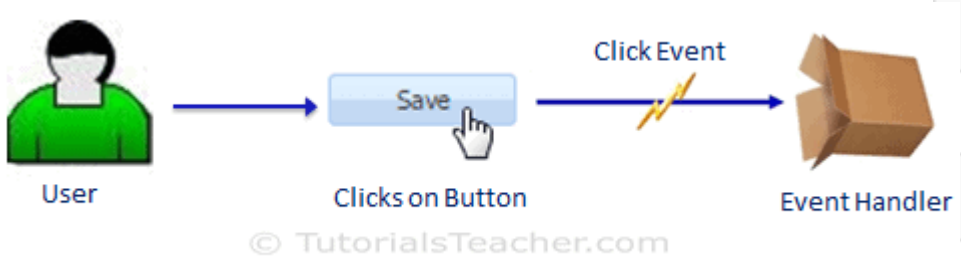
- removeClass() Method
  - $("button").click(function(){
    $("h1, h2, p").removeClass("blue");
    });

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

jQuery - css() Method

- Set a CSS Property:
  - To set a specified CSS property, use the following syntax:
  - css("*propertyname*","*value*");
  - $("p").css("background-color", "yellow");
- Set Multiple CSS Properties
  - To set multiple CSS properties, use the following syntax:
  - css({"*propertyname*":"*value*","*propertyname*":"*value*",...});
  - $("p").css({"background-color": "yellow", "font-size": "200%"});

jQuery Event Methods



- In most web applications, the user does some action to perform an operation.
- For example, user clicks on save button to save the edited data in a web page.
- Here, clicking on the button is a user's action, which triggers click event and click event handler (function) saves data.

Category	jQuery Method	DOM Event
Form events	blur	onblur
	change	onchange
	focus	onfocus
	focusin	onfocusin
	select	onselect
	submit	onsubmit
Keyboard events	keydown	onkeydown
	keypress	onkeypress
	keyup	onkeyup
	focusout	
Mouse events	click	onclick
	dblclick	ondblclick
	focusout	
	hover	
	mousedown	onmousedown
	mouseenter	onmouseenter

	mouseleave	onmouseleave
	mousemove	onmousemove
	mouseout	onmouseout
	mouseover	onmouseover
	mouseup	onmouseup
	Toggle	
Browser events	Error	onerror()
	Resize	onresize
	Scroll	onscroll
Document loading	Load	onload
	Ready	
	Unload	onunload

Commonly Used jQuery Event Methods

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

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

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

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

jQuery Effects

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

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

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

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

Query - Callback Functions

- A **jQuery Callback Function** is a function that will be executed only after the current effect gets completed.
- Following is a simple syntax of any jQuery effect method:

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

- jQuery callback function will be written as follows:

```
$(selector).effectName(speed, function(){
  <!-- function body -->
});
```

- jQuery callback functions are required due to asynchronous nature of Javascript (jQuery) code execution.
- jQuery effects may take sometime to complete, so there is a chance that the next lines of code may get executed while the effects are still being executed.
- To handle asynchronous execution of the code, jQuery allows to pass a callback in all the effect methods and the purpose of this callback function is to be executed only when the effect gets completed.
- The callback function is passed as an argument to the effect methods and they typically appear as the last argument of the method.