THE NEW COLLEGE (AUTONOMOUS), CHENNAI-14.

DEPARTMENT OF COMPUTER APPLICATIONS – SHIFT - II

STUDY MATERIAL

SUBJECT: OPEN SOURCE TECHNOLOGIES SUB CODE: 20BHM615

CLASS: III BCA STAFF: Dr. K. SANKAR

UNIT-IV

1. INTRODUCTION TO JQUERY

jQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: **Write less, do more**. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.

jQuery can be used to find a particular HTML element in the HTML document with a certain ID, class or attribute and later we can use jQuery to change one or more of attributes of the same element like color, visibility etc. jQuery can also be used to make a webpage interactive by responding to an event like a mouse click.

2. JQUERY - SELECTORS

jQuery Selectors are used to select HTML element(s) from an HTML document. Consider an HTML document is given and you need to select all the from this document. This is where jQuery Selectors will help.

jQuery Selectors can find HTML elements (ie. Select HTML elements) based on the following:

- HTML element Name
- Element ID
- Element Class
- Element attribute name
- Element attribute value
- Many more criteria

The jQuery library harnesses the power of **Cascading Style Sheets** (CSS) selectors to let us quickly and easily access elements or groups of elements in the Document Object Model (DOM).

jQuery Selectors works in very similar way on an HTML document like an **SQL Select Statement** works on a Database to select the records.

Syntax

```
$(document).ready(function(){
   $(selector)
});
```

A jQuery selector starts with a dollar sign **$** and then we put a **selector** inside the braces **()**. Here **$()** is called **factory function**, which makes use of following three building blocks while selecting elements in a given document:

Selector Name	Description
The **element** Selector	Represents an HTML element name available in the DOM. For example **$('p')** selects all paragraphs <p> in the document.
The **#id** Selector	Represents a HTML element available with the given ID in the DOM. For example **$('#some-id')** selects the single element in the document that has **some-id** as element Id.
The **.class** Selector	Represents a HTML elements available with the given class in the DOM. For example **$('.some-class')** selects all elements in the document that have a class of **some-class**.

All the above selectors can be used either on their own or in combination with other selectors. All the jQuery selectors are based on the same principle except some tweaking.

The element Selector

The jQuery **element** selector selects HTML element(s) based on the element name. Following is a simple syntax of an element selector:

```
$(document).ready(function(){
   $("Html Element Name")
});
```

Please note while using element name as jQuery Selector, we are not giving angle braces alongwith the element. For example, we are giving only plain **p** instead of **<p>**.

The #id Selector

The jQuery **#id** selector selects an HTML element based on the element **id** attribute. Following is a simple syntax of a **#id** selector:

```
$(document).ready(function(){
   $("#id of the element")
});
```

To use jQuery **#id** selector, you need to make sure that **id** attribute should be uniquely assigned to all the DOM elements. If your elements will have similar ids then it will not produce correct result.

The .class Selector

The jQuery **.class** selector selects HTML element(s) based on the element **class** attribute. Following is a simple syntax of a **.class** selector:

```
$(document).ready(function(){
   $(".class of the element")
});
```

Because a class can be assigned to multiple HTML elements with in an HTML document, so it is very much possible to find out multiple elements with a single **.class** selector statement.

3. JQUERY - ATTRIBUTES

jQuery is being heavily used in manipulating various attributes associated with HTML elements. Every HTML element can have various standard and custom attributes (i.e. properties) which are used to define the characteristics of that HTML element.

jQuery gives us the means to easily manipulate (Get and Set) an element's attributes. First let's try to understand a little about HTML standard and custom attributes.

Standard Attributes

Some of the more common attributes are −

- className
- tagName
- id
- href
- title
- rel
- src
- style

Example

Let's have a look at the following code snippet for HTML markup for an image element −

```
<img id="id" src="image.gif" alt="Image" class="class" title="This is an image"/>
```

In this element's markup, the tag name is img, and the markup for id, src, alt, class, and title represents the element's attributes, each of which consists of a name and a value.

Custom data-* Attributes

HTML specification allows us to add our own custom attributes with the DOM elements to provide additional detail about the element. These attributes names start with **data-**.

Example

Below is an example where we provided information about copyright of the image using **data-copyright** which is a custom attribute −

```
<img data-copyright="Tutorials Point" id="imageid" src="image.gif" alt="Image"/>
```

jQuery - Get Standard Attributes

jQuery **attr()** method is used to fetch the value of any standard attribute from the matched HTML element(s). We will use **jQuery Selectors** to match the desired element(s) and then we will apply **attr()** method to get the attribute value for the element.

Example

Following is a jQuery program to get href and title attributes of an anchor <a> element:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://www.tutorialspoint.com/jquery/jquery-3.6.0.js"></script>
<script>
  $(document).ready(function() {
    $("button").click(function(){
      alert( "Href = " + $("#home").attr("href"));
      alert( "Title = " + $("#home").attr("title"));
    });
  });
</script>
</head>
<body>
  <p>Click the below button to see the result:</p>
```

```
  <p><a id="home" href="index.htm" title="Tutorials Point">Home</a></p>
  <button>Get Attribute</button>
</body>
</html>
```

jQuery - Get Data Attributes

jQuery **data()** method is used to fetch the value of any custom data attribute from the matched HTML element(s). We will use **jQuery Selectors** to match the desired element(s) and then we will apply **data()** method to get the attribute value for the element.

Example

Following is a jQuery program to get author-name and year attributes of a element:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://www.tutorialspoint.com/jquery/jquery-3.6.0.js"></script>
<script>
  $(document).ready(function() {
    $("button").click(function(){
      alert( "Author = " + $("#home").data("author-name"));
      alert( "Year = " + $("#home").data("year"));
    });
  });
</script>
</head>
<body>
  <p>Click the below button to see the result:</p>

  <div id="home" data-author-name="Zara Ali" data-year="2022">
    Just an Example Content
  </div>
  <br>
  <button>Get Attribute</button>
</body>
</html>
```

jQuery - Set Standard Attributes

jQuery **attr(name, value)** method is used to set the value of any standard attribute of the matched HTML element(s). We will use **jQuery Selectors** to match the desired element(s) and then we will apply **attr(key, value)** method to set the attribute value for the element.

Example

Following is a jQuery program to set the title attribute of an anchor <a> element:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://www.tutorialspoint.com/jquery/jquery-3.6.0.js"></script>
<script>
  $(document).ready(function() {
    $("button").click(function(){
      $("#home").attr("title", "New Anchor Title");

      /* Let's get and display changed title */
      alert( "Changed Title = " + $("#home").attr("title"));
    });
  });
</script>
</head>
<body>
  <p>Click the below button to see the result:</p>

  <p><a id="home" href="index.htm" title="Tutorials Point">Home</a></p>
  <button>Set Attribute</button>
  <p>You can hover the Home link to verify the title before and after the change.</p>
</body>
</html>
```

jQuery - Set Custom Attributes

jQuery **data(name, value)** method is used to set the value of any custom attribute of the matched HTML element(s). We will use **jQuery Selectors** to match the desired element(s) and then we will apply **attr(key, value)** method to set the attribute value for the element.

4. JQUERY - DOM TRAVERSING

jQuery is a very powerful tool which provides a variety of DOM traversal methods to help us select elements in an HTML or XML document randomly as well as in sequential method. Elements in the DOM are organized into a tree-like data structure that can be traversed to navigate, locate the content within an HTML or XML document.

The DOM tree can be imagined as a collection of **nodes** related to each other through parent-child and sibling-sibling relationships and the root start from the top parent which is HTML element in an HTML document.

Before we start traversing a DOM, Let's understand the terminology of **parent**, **child** and **sibling**. Let's see an example:

```
<body>
  <p>This is paragrah</p>
  <div><span>This is div</span></div>
  <button id="b1">Get width</button>
  <button id="b2">Set width</button>
</body>
```

In the above example, we have a **<body>** element at the top, which is called a parent for all the elements. The , <p> and <button> elements inside the <body> element are called siblings. Again element inside is a child of and is called a parent of element. The element is a next sibling of the <p> element and <p> is the previous sibling of the element.

Traversing the DOM

Most of the DOM Traversal Methods do not modify the jQuery DOM object and they are used to filter out elements from a document based on given conditions. jQuery provides methods to traverse in the following three directions:

- **Traversing Upwards** - This direction means traversing the ancestors (Parent, Grandparent, Great-grandparent etc.)
- **Traversing Downwards** - This direction means traversing the descendants (Child, Grandchild, Great-grandchild etc.)
- **Sideways** - This direction means traversing the ancestors the siblings (Brother, sisters available at the same level)

5. JQUERY - CSS CLASSES

jQuery provides three methods **addClass()**, **removeClass()** and **toggleClass()** to manipulate CSS classes of the elements. We have divided our CSS manipulation discussion into two parts. This chapter will discuss about manipulating CSS classes and the next chapter will discuss about manipulating CSS properties.

jQuery - Adding CSS Classes

jQuery provides **addClass()** method to add a CSS class to the matched HTML element(s). Following is the syntax of the **addClass()** method:

```
$(selector).addClass(className);
```

This method takes a parameter which is one or more space-separated classes to be added to the class attribute of each matched element. More than one class may be added at a time, separated by a space, to the set of matched elements, like so:

```
$(selector).addClass("Class1 Class2");
```

jQuery - Removing CSS Classes

jQuery provides **removeClass()** method to remove an existing CSS class from the matched HTML element(s). Following is the syntax of the **removeClass()** method:

```
$(selector).removeClass(className);
```

This method takes a parameter which is one or more space-separated classes to be removed from the class attribute of each matched element. More than one class may be removed at a time, separated by a space, from the set of matched elements, like so:

```
$(selector).removeClass("Class1 Class2");
```

jQuery - Toggle CSS Classes

jQuery provides **toggleClass()** method to toggle an CSS class on the matched HTML element(s). Following is the syntax of the **toggleClass()** method:

```
$(selector).toggleClass(className);
```

This method takes a parameter which is one or more space-separated classes to be toggled. If an element in the matched set of elements already has the class, then it is removed; if an element does not have the class, then it is added.

6. JQUERY DOM MANIPULATION

jQuery provides methods such as **attr()**, **html()**, **text()** and **val()** which act as getters and setters to manipulate the content from HTML documents. Here are some basic operations which you can perform on DOM elements with the help of jQuery standard library methods −

- Extract the content of an element
- Change the content of an element
- Adding a child element under an existing element
- Adding a parent element above an existing element
- Adding an element before or after an existing element
- Replace an existing element with another element
- Delete an existing element
- Wrapping content with-in an element

We have already covered **attr()** method while discussing jQuery Attributes and remaining DOM content manipulation methods **html()**, **text()** and **val()** will be discussed in this chapter.

jQuery - Get Content

jQuery provides **html()** and **text()** methods to extract the content of the matched HTML element. Following is the syntax of these two methods:

```
$(selector).html();

$(selector).text();
```

The jQuery **text()** method returns plain text value of the content where as **html()** returns the content with HTML tags. You will need to use jQuery selectors to select the target element.

```
</html>
```

Get Form Fields

jQuery **val()** method is used to get the value from any form field. Following is simple syntax of this method.

```
$(selector).val();
```

jQuery - Set Content

jQuery **html()** and **text()** methods can be used to set the content of the matched HTML element. Following is the syntax of these two methods when they are used to set the values:

```
$(selector).html(val, [function]);
$(selector).text(val, [function]);
```

Here **val** is he HTML of text content to be set for the element. We can provide an optional callback function to these methods which will be called when the value of the element will be set.

The jQuery **text()** method sets plain text value of the content where as **html()** method sets the content with HTML tags.

Set Form Fields

jQuery **val()** method is also used to set the value from any form field. Following is simple syntax of this method when it is used to set the value.

```
$(selector).val(val);
```

Here **val** is he value to be set for the input field. We can provide an optional callback function which will be called when the value of the field will be set.

jQuery - Replacement Elements

The jQuery **replaceWith()** method can be used to replace a complete DOM element with another HTML or DOM element. Following is the syntax of the method:

```
$(selector).replaceWith(val);
```

Here **val** is what you want to have instead of original element. This could be HTML or simple text.

7. JQUERY - EVENT HANDLING

A jQuery Event is the result of an action that can be detected by jQuery (JavaScript). When these events are triggered, you can then use a custom function to do pretty much whatever you want with the event. These custom functions are called **Event Handlers**.

Following are the examples of some common events −

- A mouse click
- A web page loading
- Taking mouse over an element
- Submitting an HTML form
- A keystroke on your keyboard, etc.

The following table lists some of the important DOM events.

Mouse Events	Keyboard Events	Form Events	Document Events
click	keypress	submit	load
dblclick	keydown	change	resize
hover	keyup	select	scroll
mousedown		blur	unload
mouseup		focusin	ready

8. JQUERY - AJAX

AJAX is an acronym standing for Asynchronous JavaScript and XML and this technology helps us to load data from the server without a browser page refresh. If you are new with AJAX, I would recommend you go through our Ajax Tutorial before proceeding further. JQuery is a great tool which provides a rich set of AJAX methods to develop next generation web application.

Loading Simple Data

This is very easy to load any static or dynamic data using JQuery AJAX. JQuery provides **load()** method to do the job −

Syntax

Here is the simple syntax for **load()** method −

[selector].load(URL, [data], [callback]);

Here is the description of all the parameters −

- **URL** − The URL of the server-side resource to which the request is sent. It could be a CGI, ASP, JSP, or PHP script which generates data dynamically or out of a database.
- **data** − This optional parameter represents an object whose properties are serialized into properly encoded parameters to be passed to the request. If specified, the request is made using the **POST** method. If omitted, the **GET** method is used.
- **callback** − A callback function invoked after the response data has been loaded into the elements of the matched set. The first parameter passed to this function is the response text received from the server and second parameter is the status code.

9. JQUERY - EFFECTS

jQuery effects add an X factor to your website interactivity. jQuery provides a trivially simple interface for doing various kind of amazing effects like show, hide, fade-in, fade-out, slide-up, slide-down, toggle etc. jQuery methods allow us to quickly apply commonly used effects with a minimum configuration. This covers all the important jQuery methods to create visual effects.

jQuery Effect - Hiding Elements

jQuery gives simple syntax to hide an element with the help of **hide()** method:

```
$(selector).hide( [speed, callback] );
```

You can apply any jQuery **selector** to select any DOM element and then apply jQuery **hide()** method to hide it. Here is the description of all the parameters which gives you a solid control over the hiding effect −

- **speed** − This optional parameter represents one of the three predefined speeds ("slow", "normal", or "fast") or the number of milliseconds to run the animation (e.g. 1000).
- **callback** − This optional parameter represents a function to be executed whenever the animation completes; executes once for each element animated against.

jQuery Effect - Show Elements

jQuery gives simple syntax to show a hidden element with the help of **show()** method:

```
$(selector).show( [speed, callback] );
```

You can apply any jQuery **selector** to select any DOM element and then apply jQuery **show()** method to show it. Here is the description of all the parameters which gives you a control over the show effect −

- **speed** − An optional string representing one of the three predefined speeds ("slow", "normal", or "fast") or the number of milliseconds to run the animation (e.g. 1000).
- **callback** − This optional parameter represents a function to be executed whenever the animation completes; executes once for each element animated against.

jQuery Effect - Fading Elements

jQuery gives us two methods - **fadeIn()** and **fadeOut()** to fade the DOM elements **in** and **out** of visibility.

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

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

The jQuery **fadeIn()** method is used to fade in a hidden element where as **fadeOut()** method is used to fade out a visible element. Here is the description of all the parameters which gives you a control over the fading effects −

- **speed** − An optional string representing one of the three predefined speeds ("slow", "normal", or "fast") or the number of milliseconds to run the animation (e.g. 1000).
- **callback** − This optional parameter represents a function to be executed whenever the animation completes; executes once for each element animated against.

jQuery Effect - Sliding Elements

jQuery gives us two methods - **slideUp()** and **slideDown()** to slide up and slide down the DOM elements respectively. Following is the simple syntax for these two methods:

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

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

The jQuery **slideUp()** method is used to slide up an element where as **slideDown()** method is used to slide down. Here is the description of all the parameters which gives you more control over the effects −

- **speed** − An optional string representing one of the three predefined speeds ("slow", "normal", or "fast") or the number of milliseconds to run the animation (e.g. 1000).
- **callback** − This optional parameter represents a function to be executed whenever the animation completes; executes once for each element animated against.

10. JQUERY – INTERACTIONS

Interactions could be added basic mouse-based behaviors to any element. Using with interactions, We can create sortable lists, resizable elements, drag & drop behaviors. Interactions also make great building blocks for more complex widgets and applications.

Sr.No.	Interactions & Description
1	Drag able Enable drag able functionality on any DOM element.
2	Drop able Enable any DOM element to be drop able.

3	Resize able Enable any DOM element to be resize-able.
4	Select able Enable a DOM element (or group of elements) to be selectable.
5	Sort able Enable a group of DOM elements to be sortable.

11. JQUERY – WIDGETS

A jQuery UI widget is a specialized jQuery plug-in. using plug-in, we can apply behaviors to the elements. However, plug-ins lack some built-in capabilities, such as a way to associate data with its elements, expose methods, merge options with defaults, and control the plug-in's lifetime.

Sr.No.	Widgets & Description
1	Accordion Enable to collapse the content, that is broken into logical sections.
2	Autocomplete Enable to provides the suggestions while you type into the field.
3	Button Button is an input of type submit and an anchor.
4	Datepicker It is to open an interactive calendar in a small overlay.
5	Dialog Dialog boxes are one of the nice ways of presenting information on an HTML page.
6	Menu Menu shows list of items.
7	Progressbar It shows the progress information.
8	Select menu Enable a style able select element/elements.
9	Slider The basic slider is horizontal and has a single handle that can be moved with the

	mouse or by using the arrow keys.
10	Spinner It provides a quick way to select one value from a set.
11	Tabs It is used to swap between content that is broken into logical sections.
12	Tooltip Its provides the tips for the users.

12. JQUERY - THEMING

Jquery has two different styling themes as A And B.Each with different colors for buttons, bars, content blocks, and so on.

The syntax of J query theming as shown below −

~~~ End of Unit-IV~~~
~~~