Usman Institute of Technology

Department of Computer Science

Course Code: SE319

Course Title: Web Engineering

Fall 2022

Lab 03

Objective:

This experiment is related to the library of JavaScript i.e. JQuery and also the fundamentals of AJAX which one of the important method used in Web Applications for the consumption of APIs.

Student Information

Student Name	
Student ID	
Date	

Assessment

Marks Obtained	
Remarks	
Signature	

Usman Institute of Technology
Department of Computer Science
SE319 – Web Engineering

Lab 03

Instructions

- Come to the lab in time. Students who are late more than 15 minutes, will not be allowed to attend the lab.
- Students have to perform the examples and exercises by themselves.
- Raise your hand if you face any difficulty in understanding and solving the examples or exercises.
- Lab work must be submitted on or before the submission date.

1. Objective

This experiment is related to the library of JavaScript i.e. JQuery and also the fundamentals of AJAX which one of the important method used in Web Applications for the consumption of APIs.

2. Theory

JQuery is a lightweight, "write less, do more", and JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript on your website. JQuery is a fast, small and feature-rich JavaScript library.

JQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax.

JQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

The jQuery library is a single JavaScript file, and you reference it with the HTML `<script>` tag (note that the `<script>` tag should be inside the `<head>` section):

```
<head>
   <script src="https://code.jquery.com/jquery-3.6.1.min.js"></script>
</head>
```

AJAX is an acronym for Asynchronous JavaScript and XML. It is a group of inter-related technologies like JavaScript, DOM, XML, HTML/XHTML, CSS, and XML Http Request etc. It allows us to send and receive data asynchronously without reloading the web page. So it is fast. The ajax() method in jQuery performs an AJAX request. JQuery provides a rich set of AJAX methods for developing web applications. It is widely used for the requests.

AJAX method allows you to send asynchronous http requests to submit or retrieve data from the server without reloading the whole page.

AJAX is the art of exchanging data with a server, and update parts of a web page - without reloading the whole page. AJAX can be used to send http GET, POST, PUT, DELETE etc. request. It can retrieve any type of response from the server. Use option parameter to customize ajax request as per your need.

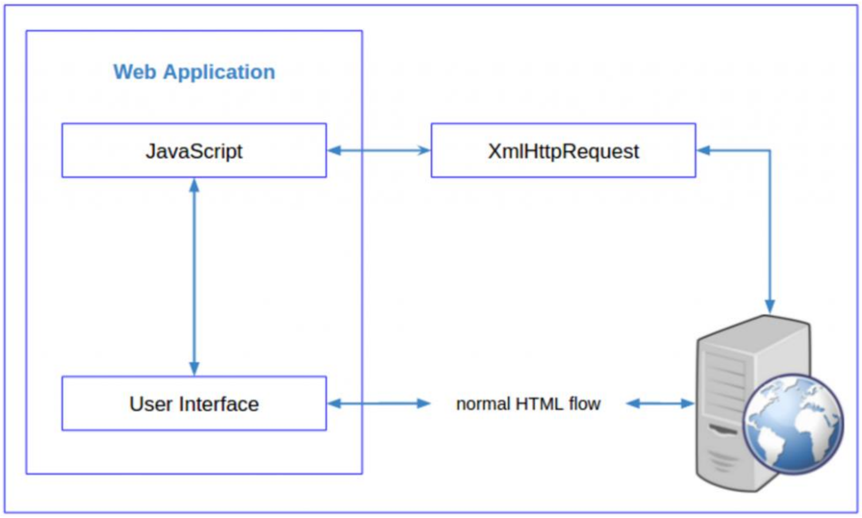


Figure 1: Ajax request flow (Source: https://www.clickittech.com/developer/ajax-technique-and-jquery-methods/)

3. JQuery Syntax

The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

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:

```
$(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".
```

4. JQuery - Selectors

jQuery selectors allow you to select and manipulate HTML element(s). jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more. It is based on the existing CSS Selectors, and in addition, it has some own custom selectors. All selectors in jQuery start with the dollar sign and parentheses: $().

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")
```

Task 1
When a user clicks on a button, all `<p>` elements will be hidden:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Lab # 3 - JQuery & AJAX</title>
        <script src="https://code.jquery.com/jquery-
3.6.0.min.js"></script>
    </head>
    <body>
        <button>Click me</button>
        <p>This is dummy Text.</p>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $("p").hide();
                });
            });
        </script>
    </body>
</html>
```

#id selector:
The jQuery `#id` selector uses the id attribute of an HTML tag to find the specific element. An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element. To find an element with a specific id, write a hash character, followed by the id of the HTML element:

```
$("#test")
```

Task 2
When a user clicks on a button, the element with `id="test"` will be hidden:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title> Lab # 3 - JQuery & AJAX</title>
        <script src="https://code.jquery.com/jquery-
3.6.0.min.js"></script>
    </head>
    <body>
```

```
        <button>Click me</button>
        <div id="test">
            <p>This is dummy Text.</p>
        </div>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $("#test").hide();
                });
            });
        </script>
    </body>
</html>
```

.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:

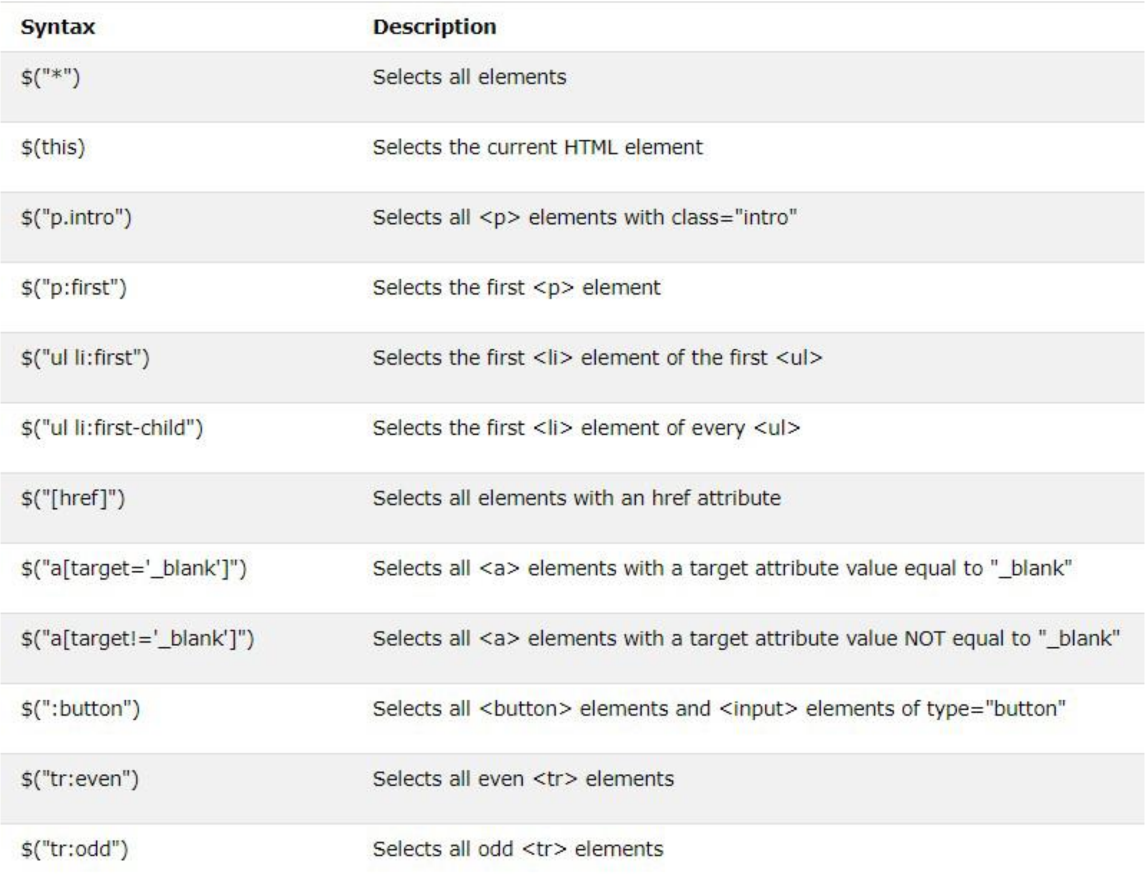
```
$(".test")
```

Task 3

When a user clicks on a button, the element with `class="test"` will be hidden:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello! WDD Class</title>
        <script src="https://code.jquery.com/jquery-
3.6.0.min.js"></script>
    </head>
    <body>
        <button>Click me</button>
        <div class="test">
            <p>This is dummy Text.</p>
        </div>
        <script>
            $(document).ready(function(){
                $("button").click(function(){
                    $(".test").hide();
                });
            });
        </script>
    </body>
</html>
```

Different Types of Selectors:



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 <li> element of the first <ul>
$("ul li: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

5. JQuery – Event Methods

In jQuery, most DOM events have an equivalent jQuery method. To assign a click event to all paragraphs on a page, you can do this:

```
$("p").click();
```

The next step is to define what should happen when the event fires. You must pass a function to the event:

```
$("p").click(function(){
    // action goes here!!
});
```

Some of the commonly used JQuery Event Methods are:

- click()
- dblclick()
- mouseenter()
- mouseleave()
- mousedown()
- mouseup()
- hover()

- focus()
- blur()
- on()

Task 4
Attach multiple event handlers to a `<p>` element:

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

Task 5
With jQuery, you can hide and show HTML elements with the show() and hide() methods:

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

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

Task 6
The jQuery fadeIn() method is used to fade in a hidden element.

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the fading completes. The following example demonstrates the fadeIn() method with different parameters:

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

Try same syntax for fadeout and fadetoggle.

Task 7

The jQuery fadeTo() method allows fading to a given opacity (value between 0 and 1).

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

The required speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The required opacity parameter in the fadeTo() method specifies fading to a given opacity (value between 0 and 1).

The optional callback parameter is a function to be executed after the function completes.

The following example demonstrates the fadeTo() method with different parameters:

```
$("button").click(function(){
     $("#div1").fadeTo("slow", 0.15);
     $("#div2").fadeTo("slow", 0.4);
     $("#div3").fadeTo("slow", 0.7);
});
```

Task 8

The jQuery slideUp() method is used to slide up an element.

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

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the sliding completes.

The following example demonstrates the slideUp() method:

```
$("#flip").click(function(){
     $("#panel").slideUp();
});
```

Task 9

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

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

The required params parameter defines the CSS properties to be animated.

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

The optional callback parameter is a function to be executed after the animation completes.

The following example demonstrates a simple use of the animate() method; it moves a <div> element to the right, until it has reached a left property of 250px:

```
$("button").click(function(){
     $("div").animate({left: '250px'});
});
```

6. AJAX Method

Task 10

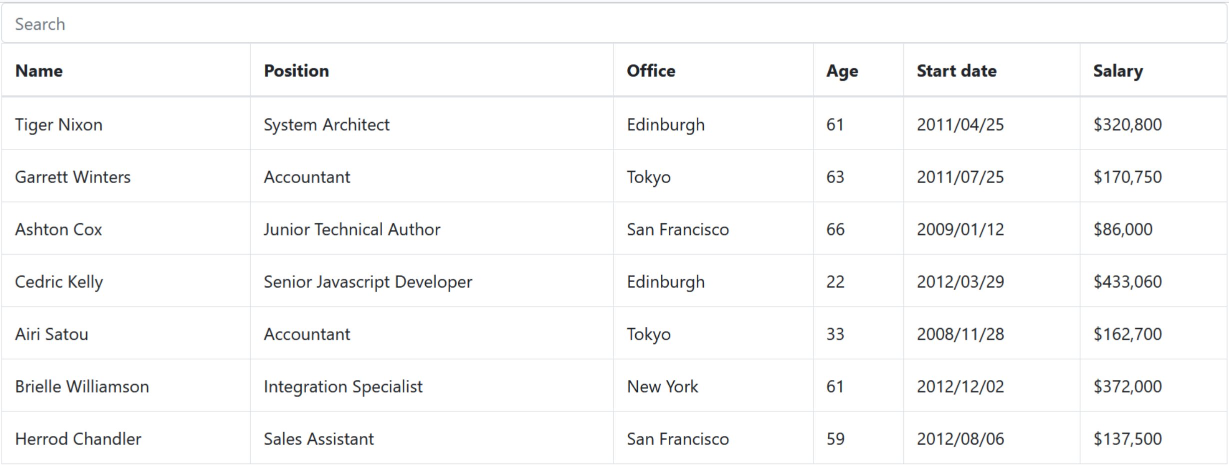
```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Lab # 3 - JQuery & AJAX</title>
        <script src="https://code.jquery.com/jquery-
3.6.0.min.js"></script>
        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap
.min.css" >
    </head>
<body>
    <div class="container">
        <div class="row">
            <input type="text" id="myInput" class="form-control"
placeholder="Search"/>
        </div>
        <div class="row">
            <table class="table table-bordered" id="myTable"
width="100%" cellspacing="0">
                <thead>
                    <tr>
                        <th>Currency Code</th>
                        <th>Conversion</th>
                    </tr>
                </thead>
                <tbody id="TBody">

                </tbody>
            </table>
        </div>
    </div>
    <script>
    $(document).ready(function(){
        $("#myInput").on("keypress", function(e) {
            if(e.which == 13)
            {
                var value = $(this).val().toUpperCase();
                $.ajax({
                    url: 'https://open.er-
api.com/v6/latest/'+value,
                    dataType: 'json',
                    type: 'GET',
                    success: function(data,status,xhr){
                        var oRates = data.rates;
                        var html = '';
                        for(var o in oRates)
                        {
                            if(oRates.hasOwnProperty(o)) {
                                html += `<tr>
                                        <td>${o}</td>
                                        <td>${oRates[o]}</td>
                                        </tr>`;
                            }
                        }
                        $('#TBody').html(html);
                    },
                    error: function(jqXhr, textStatus,
errorMessage) {
                        alert(errorMessage);
                    }
                });
            }
        });
    });
    </script>
</body>
</html>
```

7. Lab Task

Task A

Use jQuery to filter/search for specific elements. Perform a case-insensitive search for items in a table. Type something in the input field to search from the table:



Search

Name	Position	Office	Age	Start date	Salary
Tiger Nixon	System Architect	Edinburgh	61	2011/04/25	$320,800
Garrett Winters	Accountant	Tokyo	63	2011/07/25	$170,750
Ashton Cox	Junior Technical Author	San Francisco	66	2009/01/12	$86,000
Cedric Kelly	Senior Javascript Developer	Edinburgh	22	2012/03/29	$433,060
Airi Satou	Accountant	Tokyo	33	2008/11/28	$162,700
Brielle Williamson	Integration Specialist	New York	61	2012/12/02	$372,000
Herrod Chandler	Sales Assistant	San Francisco	59	2012/08/06	$137,500

Use filter on each row of table and toggle function to hide and show the rows.
You can also use bootstrap CSS to show the proper styling of table using the given below link
`https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css`

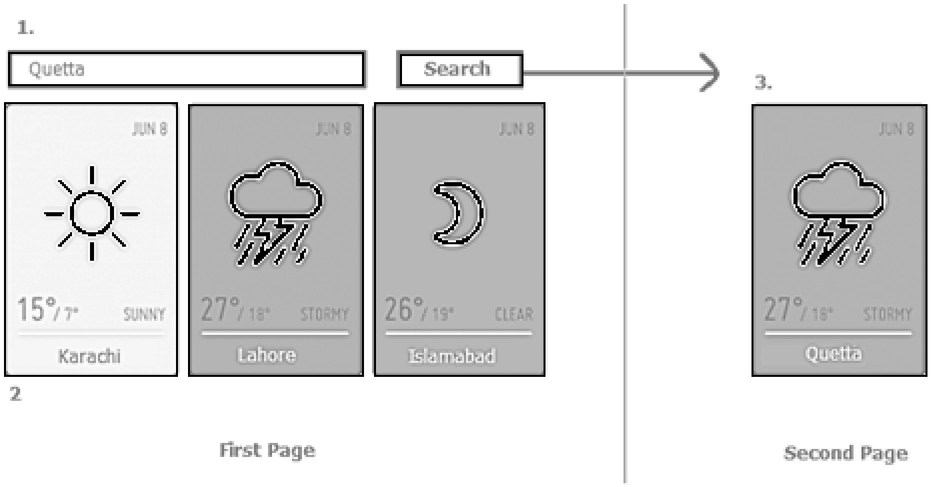
Task B

Replica of Minimeteo

☠ **Don't** Use any JavaScript popup boxes. Use something else to show error!

Design Guide & App Information Link: https://goo.gl/espVRo

Create/Design a 2 pages, minimal design, and weather information responsive application for Pakistan's major cities.



Process Flow.

- ✓ On load/start,
  - **a.** An Ajax is request is created & send to any open/free weather API & return response of **3 major cities of Pakistan.**
  - b. On successful response, populate/set response data in weather card as shown in **[2]** on 1st/Main page.
  - c. Also store response in local storage for future processing.
  - d. On error response [case like internet isn't working etc.], just show error & reuse last local storage values if exists.

- ✓ On search click**[1]**,
  - a. An Ajax is request is created & send to any open/free weather API with provided city name & return response of **particular city**.
  - b. On successful response, populate/set response data in weather card as shown in **[3]** on 2nd page.

  Follow above point c & d

- ✓ Weather card is consist of,
  - a. City Name
  - b. Temperature
  - c. Date
  - d. Weather Icon
  - e. Weather status