MASTERING COMPUTER SCIENCE

Mastering jQuery

A Beginner's Guide

EDITED BY

Sufyan bin Uzayr

CRC Press
Taylor & Francis Group

Mastering jQuery

With a combination of flexibility and feature-rich design, jQuery has changed the way millions of people write JavaScript. It makes tasks such as HTML document conversion, animation, event management, and Asynchronous JavaScript and XML (AJAX) much easier with an easy-to-use API that works on a wide variety of browsers. This book is a detailed guide for beginners to understand jQuery and its programming techniques. It discusses the fundamental concepts of jQuery and provides a roadmap for creating websites using jQuery programming.

Key Features:

- Follows a hands-on approach and offers practical lessons and tutorials related to jQuery
- Provides a special focus on animations and effects using jQuery programming
- Includes detailed tutorials meant for beginners to jQuery

Mastering jQuery: A Beginner's Guide is an exciting journey for anyone who wants to create an animated website. This is a valuable resource for developers already familiar with HTML and CSS and little understanding of JavaScript. After finishing this book, readers will be able to quickly build their website with absolute ease

About the Series

The *Mastering Computer Science* covers a wide range of topics, spanning programming languages as well as modern-day technologies and frameworks. The series has a special focus on beginner-level content, and is presented in an easy-to-understand manner, comprising:

- Crystal-clear text, spanning various topics sorted by relevance,
- Special focus on practical exercises, with numerous code samples and programs,
- A guided approach to programming, with step-by-step tutorials for the absolute beginners,
- Keen emphasis on real-world utility of skills, thereby cutting the redundant and seldom-used concepts and focusing instead of industry-prevalent coding paradigm,
- A wide range of references and resources, to help both beginner and intermediate-level developers gain the most out of the books.

Mastering Computer Science series of books start from the core concepts, and then quickly move on to industry-standard coding practices, to help learners gain efficient and crucial skills in as little time as possible. The books assume no prior knowledge of coding, so even the absolute newbie coders can benefit from this series.

Mastering Computer Science series is edited by Sufyan bin Uzayr, a writer and educator with over a decade of experience in the computing field.

For more information about this series, please visit: https://www.routledge.com/Mastering-Computer-Science/book-series/MCS

Mastering jQuery
A Beginner's Guide

Edited by
Sufyan bin Uzayr

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

First Edition published 2024
by CRC Press
2385 NW Executive Center Drive, Suite 320, Boca Raton, FL 33431

and by CRC Press
2 Park Square, Milton Park, Abingdon, Oxon, OX14 4RN

CRC Press is an imprint of Taylor & Francis Group, LLC

Library of Congress Cataloging-in-Publication Data

Names: Bin Uzayr, Sufyan, editor.
Title: Mastering jQuery : a beginner's guide / edited by Sufyan bin Uzayr.
Description: Boca Raton : CRC Press, 2024. | Series: Mastering computer science | Includes bibliographical references and index.
Identifiers: LCCN 2023007409 (print) | LCCN 2023007410 (ebook) | ISBN 9781032415192 (paperback) | ISBN 9781032415208 (hardback) | ISBN 9781003358497 (ebook)
Subjects: LCSH: JavaScript (Computer program language) | Cascading style sheets. | Web site development.
Classification: LCC QA76.73.J39 M3785 2024 (print) | LCC QA76.73.J39 (ebook) | DDC 005.2/762--dc23/eng/20230424
LC record available at https://lccn.loc.gov/2023007409
LC ebook record available at https://lccn.loc.gov/2023007410

ISBN: 9781032415208 (hbk)
ISBN: 9781032415192 (pbk)
ISBN: 9781003358497 (ebk)

DOI: 10.1201/9781003358497

Typeset in Minion
by KnowledgeWorks Global Ltd.

Access the Support Material: www.routledge.com/9781032415192

For Mom

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

Contents

About the Editor

Sufyan bin Uzayr is a writer, coder, and entrepreneur with over a decade of experience in the industry. He has authored several books in the past, pertaining to a diverse range of topics, ranging from History to Computers/IT.

Sufyan is the Director of Parakozm, a multinational IT company specializing in EdTech solutions. He also runs Zeba Academy, an online learning and teaching vertical with a focus on STEM fields.

Sufyan specializes in a wide variety of technologies, such as JavaScript, Dart, WordPress, Drupal, Linux, and Python. He holds multiple degrees, including ones in Management, IT, Literature and Political Science.

Sufyan is a digital nomad, dividing his time between four countries. He has lived and taught in universities and educational institutions around the globe. Sufyan takes a keen interest in technology, politics, literature, history, and sports, and in his spare time, he enjoys teaching coding and English to young students.

Learn more at sufyanism.com

Acknowledgments

There are many people who deserve to be on this page, for this book would not have come into existence without their support. That said, some names deserve a special mention, and I am genuinely grateful to:

- My parents, for everything they have done for me.
- The Parakozm team, especially Divya Sachdeva, Jaskiran Kaur, and Simran Rao, for offering great amounts of help and assistance during the book-writing process.
- The CRC team, especially Sean Connelly and Danielle Zarfati, for ensuring that the book's content, layout, formatting, and everything else remain perfect throughout.
- Reviewers of this book, for going through the manuscript and providing their insight and feedback.
- Typesetters, cover designers, printers, and everyone else, for their part in the development of this book.
- All the folks associated with Zeba Academy, either directly or indirectly, for their help and support.
- The programming community in general, and the web development community in particular, for all their hard work and efforts.

Sufyan bin Uzayr

Zeba Academy – Mastering Computer Science

The "Mastering Computer Science" series of books are authored by the Zeba Academy team members, led by Sufyan bin Uzayr, consisting of:

- Divya Sachdeva
- Jaskiran Kaur
- Simran Rao
- Aruqqa Khateib
- Suleymen Fez
- Ibbi Yasmin
- Alexander Izbassar

Zeba Academy is an EdTech venture that develops courses and content for learners primarily in STEM fields, and offers educational consulting and mentorship to learners and educators worldwide.

Additionally, Zeba Academy is actively engaged in running IT Schools in the CIS countries, and is currently working in partnership with numerous universities and institutions.

For more info, please visit https://zeba.academy

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

CHAPTER 1

jQuery

Introduction

IN THIS CHAPTER

- Introduction
- History
- Installation
- jQuery Methods

In this chapter, you will learn about jQuery, its fundamentals, history, advantages and disadvantages, and features. We will also cover various built-in methods used to create websites. There is various events method in jQuery such as keyboard, mouse, DOM, traversing etc. So let's begin with its introduction.

INTRODUCTION

jQuery is a fast JavaScript Library developed by John Resig in 2006 with a great theme: write less, do more. jQuery makes it easy to split HTML documents, event management, animation, and AJAX collaborations for faster web development.

It is fast, light, lightweight, and rich in features and designed to simplify HTML text on the client side. It makes things like DOM cut and trick, event management, animation, and AJAX much easier. With a combination

DOI: 10.1201/9781003358497-1

of flexibility and extensibility, jQuery has changed the way millions of people write JavaScript. As of May 2019, jQuery used 73% of the 10 million most popular websites.

The purpose of jQuery is to make JavaScript faster, easier, shorter, more efficient, and more attractive. jQuery helps developers to make websites more dynamic and collaborative with jQuery.

jQuery takes many common Vanilla JavaScript functions that require multiple lines of code and wraps them with pre-defined and built-in methods that you can call one line of code in jQuery.

jQuery is best known for its: "Write less, do more." This philosophy can be broadly defined as three concepts:

- You can find some specific elements (with CSS options) and do something about them (via jQuery methods), that is, find a set of items in DOM, and then do something with that set of elements.
- Combining multiple jQuery modes in a set of items.
- Using a jQuery wrapper and subtle repetition.

DIFFERENCE BETWEEN FRAMEWORK AND LIBRARY

FRAMEWORK

Software development is a complex process. This includes many phrases such as coding, designing, testing, etc. When it comes to coding, engineers have to worry about syntax, announcements, garbage collection, statements, exceptions, and more. Software components make development easier by providing a common platform where engineers can control all or part of a software development process.

Benefits

- Software frameworks are highly flexible, robust, and efficient because they are often developed, tested, and developed by many software developers.
- It facilitates better processes of planning and the proper use of design patterns.
- Avoiding duplication and inadequate code results in fewer errors and inconsistent development processes.

- It can create a personal software framework or contribute to an open-source framework. As a result, the performance of software programs will continue to improve.
- Many parts of the code and function are framed and tested early, which makes the applications more reliable.
- Even non-code engineers can test and repair it.
- Building an App takes very little time as it provides code to perform normal tasks and uses code provided by the developer in customization.
- As a result of using the software framework, you can focus on high-level performance, while low-level tasks are managed by the framework itself.

WHAT IS A LIBRARY?

A library is a collection of pre-written code that can be used to simplify the tasks. The term library simply refers to the code that is used repeatedly. It is actually a set of pre-defined tasks and classes that programmers can use to simplify the work and speed up the development process. Therefore, developers do not have to write code in order to perform a specific task because the library already compiles the code for those functions. General libraries are available in many editing languages, but editors can also create their own custom libraries.

Often, most libraries do not have a large size. Typically, libraries focus on smaller scales such as cables, sockets, IO, etc., so their APIs are also smaller and require fewer dependencies. jQuery, NumPy, etc. are examples of libraries.

WHY DO WE NEED A LIBRARY?

There is only a single answer to this question, and that is a reuse of a code already written by someone (or another engineer). Engineers can avoid writing code that has already been written in the library by using it. The result is more efficient and less time spent on coding. As other people may also use it, you will benefit from them by finding and fixing any bugs. This is one of the reasons why libraries are so useful.

HISTORY[1]

John Resig developed the first version of jQuery in 2005 and released it in 2006 at an event called BarCampNYC. On the first jQuery website, he wrote:

> jQuery is a JavaScript library that takes writing a JavaScript code should be fun. jQuery achieves this goal by doing regular, repetitive tasks, removing all unnecessary breaks, and leaving it short, smart and understandable.

jQuery had two key value propositions. The first was to provide an ergonomic API for web page fraud. In particular, it has provided powerful ways to select elements. In addition to selecting items based on their id or classes, jQuery allows for complex expressions, such as selecting items based on their interactions with other features. The second point of sale was to distinguish between the browsers. At that time, it was difficult to write code that would work strictly for all browsers.

jQuery was founded in January 2006 at BarCamp NYC by John Resig, influenced by Dean Edwards's cssQuery library. It is currently run by a team of engineers led by Timmy Willison (with jQuery select engine, Sizzle, led by Richard Gibson).

jQuery was originally licensed under CC BY-SA 2.5, and licensed by MIT in 2006. At the end of 2006, it had two licenses under the GPL and MIT licenses. As this led to some confusion, in 2012 the GPL was revoked and is now licensed only under the MIT license.

NET FRAMEWORK AND LIBRARIES

A number of web frameworks have emerged since the release of jQuery, with some of the current ones being React, Angular, and Vue. These frameworks have two important advantages over jQuery.

It is first that they make it easy to split the UI into sections. They are designed to manage page submissions and reviews. jQuery is usually used only to update a page, depending on the server to provide the first page.

The React, Angular, and Vue sections, on the other hand, allow for strong integration between HTML, code, and CSS. In the same way that we can split a codebase into multiple functions and classes contained in it, breaking the UI into usable components makes it easier to build and maintain a complex website.

The second advantage is that the new frameworks promote the declaration paradigm, in which the developer defines what the UI should look like and leaves it in the framework to make the necessary changes to get there. This method is contrary to the enforcement method identified by the jQuery code.

With jQuery, you clearly record the steps to make any changes. In a declaration framework, he says, "Based on this data, this is how the UI should be." This can greatly reduce the amount of memory you have to do to write code without interruption. Developers have embraced these new ways of building websites, reducing jQuery affiliates.

SYNTAX TO USE jQuery

jQuery has a very basic syntax to perform any functionalities.

Basic syntax is: $(selector).action();

- The $ sign is to define/access jQuery.
- The (selector) is the query for any DOM (Document Object Model) elements. jQuery uses CSS selectors here.
- The action() is the jQuery handler to perform any functionality to the selected DOM element.

WHY DO WE USE jQuery IN THE APP?

We all know that jQuery is a fast, rich, and light JavaScript library. The main motive of using jQuery is to make it easier to use JavaScript on your modern and smart website. It is highly recommended to have a basic knowledge of HTML, CSS, and JavaScript.

jQuery was upgraded to save developers' time by reducing code. It takes a lot of tasks that require multiple lines of JavaScript to be executed and wrap up the strategies you can say with one line of code.

REASONS TO USE jQuery IN YOUR APPLICATION

Easy to understand: It has simpler code than JavaScript. So you have to write a few lines of code to do the same thing. In addition, builders do not have to be experts in web design or web design to create amazing patterns on their sites. Any developer who has spent hours typing and experimenting with CSS documents will be able to use the simple features that jQuery brings to the table. There is also a set of solid jQuery UI ingredients that builders can link to their websites.

DIFFERENCE BETWEEN JavaScript AND jQuery

JavaScript: JavaScript is a great programming language used to make websites responsive and engaging. It is one of the identified areas near HTML and CSS that is used to build web pages. When HTML and CSS embellish and create web pages like that, JavaScript makes web pages flexible (we can say it gives them life). JavaScript is a large client language. It is used not only for website development but also in many desktop and server applications (Node.js, for example) and other information sites, such as MongoDB and CouchDB, which also use JavaScript. Whenever your browser analyzes a web page, its responsibility is to create a tree layout presentation with memory.

jQuery: jQuery is a JavaScript framework created from JavaScript. It is a very popular JavaScript library founded by John Resign and released in January 2006 at BarCamp NYC. It is a free, open-source library; a fast, short, rich JavaScript library; and compatible with browsers. The purpose of jQuery is to make life easier for most people so that they can easily develop browser-based websites and applications using JavaScript. In a nutshell, we can say that "jQuery is a library to provide a better place for web client development" to the developer with the help of its rich library.

Here is the coding example of JavaScript and jQuery how to hide element in both given below:

```
<!DOCTYPE html>
<html>
  <title> How to hide element using JavaScript
</title>
  <head>
    <style>
      body{
        text-align: center;
        margin: 5%;
        padding: 5%;
      }
      .container{
        border-radius: 20px;
        padding:10%;
        border: 5px solid rgb(136, 77, 77);
```

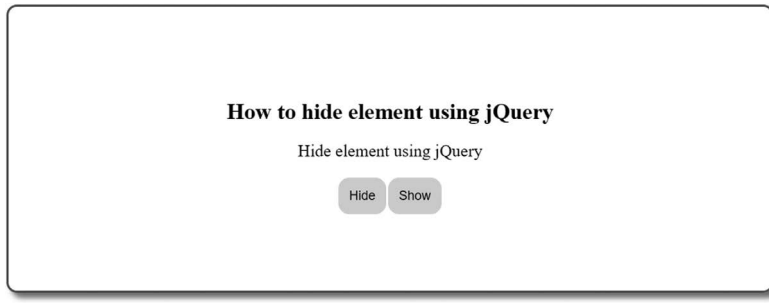
```
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color: RGB(80, 243, 47)
      }
      h1{
        font-size: 42px;
      }
      p{
        font-size: 32px;
      }
    </style>
  </head>
 <body>
   <div class="container">
     <h1> How to hide element using JavaScript </h1>
    <p id = "pid"> Hide element using JavaScript </p>

    <button type = "button" onclick = "displayHide()
"> Hide </button>
    <button type = "button" onclick = "displayShow()
"> Show </button>

    <script>
     function displayHide() {
       document.getElementById("pid").style.visibility
= "hidden";
     }
     function displayShow() {
       document.getElementById("pid").style.visibility
= "visible";
     }
    </script>
   </div>

 </body>
</html>
```

How to hide element using JavaScript.

Example of jQuery:

```
<!DOCTYPE html>
<html>
  <title>How to hide element using JavaScript
</title>
  <head>
    <style>
      body{
        text-align: center;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:10%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
```

```
      h1{
        font-size: 42px;
      }
      p{
        font-size: 32px;
      }
    </style>
  </head>
 <body>
   <div class="container">
    <!DOCTYPE html>
    <html>
    <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.
min.js"></script>
    <script>
    $(document).ready(function(){
     $(".btn1").click(function(){
      $("p").hide();
     });
     $(".btn2").click(function(){
      $("p").show();
     });
    });
    </script>
    </head>
    <body>
      <h1> How to hide element using jQuery </h1>
      <p id = "pid"> Hide element using jQuery </p>
    <button class="btn1"> Hide </button>
    <button class="btn2"> Show </button>

    </body>
    </html>

   </div>

 </body>
</html>
```



How to hide element using jQuery.

HOW TO INSTALL jQuery?[2]

There are various ways to install and use jQuery on your web site.

- By using jQuery with a CDN
- The jQuery library from jQuery.com
- The jQuery using npm or Yarn
- The jQuery using Bower

USING jQuery WITH CDN

You can use Content Delivery Networks (CDNs) to add jQuery to your web page. It can host jQuery on servers over the globe. jQuery hosted CDN by https://code.jquery.com. By adding its code for minified jQuery lib., you can copy and paste it into the head section on the HTML page.

Example:

```
<head>
  <script integrity="sha256-CSXorXvZcTkaix6Yvo6Hpp
cZGetbYMGWSFlBw8HfCJo=" crossorigin="anonymous
src="https://code.jquery.com/jquery-3.4.1.min.js"
"></script>
  </head>
```

Other CDN links:

- CDNJS CDN
- jsDelivr CDN

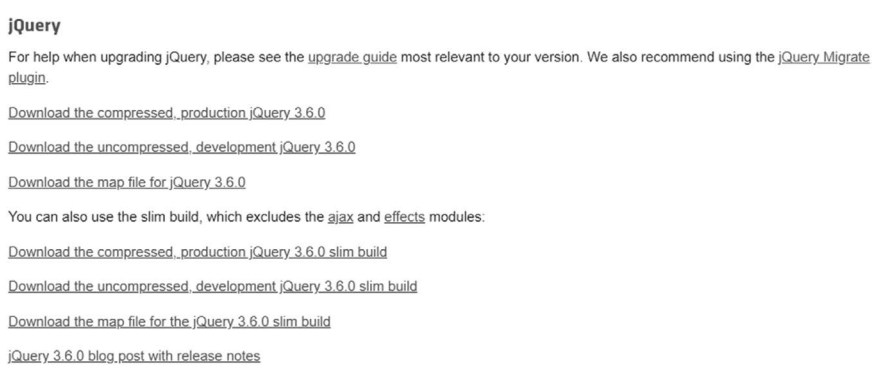
- Google CDN
- Microsoft CDN

DOWNLOAD THE jQuery LIBRARY FROM jQuery.com

There are two types of versions available for its download. Go to the "https://jquery.com/download/" official site and download the required version from the list. On its official site, you will see some links and you can download as per your needs:

- Download the compressed, production jQuery 3.6.0.
- Download the uncompressed, development jQuery 3.6.0.

The given figure will help you to find the installation link for jQuery installation.



jQuery installation.

DOWNLOAD THE jQuery LIBRARY USING npm (NODE PACKAGE MANAGER)

It is registered as a package on npm. You can install the newer version of jQuery using the npm CLI command, such as:

```
$ npm install jquery
```

As another option, you can also use the Yarn CLI command:

```
$ yarn add jquery
```

DOWNLOAD THE jQuery LIBRARY USING BOWER

It is registered as a package on npm. You can also install the latest version of jQuery using the bower CLI command:

```
$ bower install jquery
```

It will install jQuery to install the directory, the default bower_components. Within its bower_components/jquery/dist/ you will find an uncompressed release and a compressed release.

If you want to install the compressed jQuery file, you can install just that file using the following command:

```
$ bower install https://code.jquery.com/jquery-
3.4.1.min.js
```

WHY DO WE USE CDN?

Let's discuss the benefits of using a CDN:

- Google, Microsoft, and Yahoo offer great volume and high ratings. Better CDNs offer high availability and lower network latency.
- jQuery is used on many popular websites. If the user has already visited a jQuery webpage from CDN, then when it comes to your page the jQuery file is already cached by the browser, and there is no need to download the file again.
- There is a browser limit on the simultaneous connection (download of files) from a given domain. This number varies from browser to browser. For example, the browser allows two active connections, so the third download is blocked until one previous file is fully restored. CDN files are hosted on various domains. In fact, one CDN allows that browser to download two more files at a time.
- The HTTPS request is hosted by the CDN server, so the load on your web server is minimized. This also means that there are savings in the bandwidth of your website which will reduce your hosting costs.
- Page loading time is important for SEO techniques because search engines regard us as one of the factors. Therefore, developers can use jQuery CDN to ensure the improvement of their web pages in search engines.

Another example:

```
<!DOCTYPE html>
<html>
  <title>How to hide element using JavaScript</
title>
  <head>
    <style>
      body{
        text-align: center;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:10%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
      h1{
        font-size: 42px;
      }
      p{
        font-size: 32px;
      }
    </style>
  </head>
 <body>
   <div class="container">
    <!DOCTYPE html>
    <html>
    <head>
      <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.
min.js"></script>
      <script>
```

```
        $(document).ready(function () {
          $('#demo').click(function () {
            $('.test').css('backgroundColor','#86
f2f7');
          });
        });
      </script>
    </head>
    <div class="test">
      <h1> How to change color of paragraph? </h1>
      <p> Click the button to change the color</p>
    </div>
    <button id="demo">Click on it to change
color</button>

    </body>
    </html>

   </div>

 </body>
</html>
```

How to change color of paragraph.

$ (document) .ready () function in jQuery

Here, we will learn the $ (document) .ready () function in jQuery with some examples. This is an important part, which we need to understand in jQuery.

ready () function in jQuery

$ (document) .ready () is a jQuery expulsion event only when all DOM is fully loaded and ready for use by jQuery. This document.ready event is to prevent any of its code from running before the document finishes uploading (ready). This is the first secure area of the page loading process where we can trick our DOM features. This allows you to write jQuery (or JavaScript) code before the body tag.

The document.ready event only works after the DOM is loaded but before all the images, CSS, frames, etc. are fully loaded.

Here is the syntax to use $(document).ready():

```
$(document).ready(function () {
 // Write your jQuery codes
  });
```

Or you can use this:

```
$ (function () {
   // Write your jQuery codes
    });
```

Example: How to use this ready function to get alert:

```
<!DOCTYPE html>
<html>
  <title>How to hide element using JavaScript</title>
  <head>
 <script type = "text/javascript" src="https://ajax.
googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.
js"></script>
  <script>
       $(document).ready(function () {
          $(' #demo ').click(function () {
            alert(" Button is Clicked ");
          });
        });
      </script>
    </head>

    <style>
      body{
        text-align: center;
```

```
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:10%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
      h1{
        font-size: 42px;
      }
      p{
        font-size: 32px;
      }
    </style>
  </head>
<body>
  <div class="container">
  <h1> How to get alert in jQuery </h1>
     <button id="demo" > Click Me </button>
   </body>
   </html>
```

How to get alert in jQuery.

TYPES OF jQuery METHODS

jQuery | SELECTION AND EVENT METHODS

jQuery is a powerful JavaScript library. It is more powerful than JavaScript. jQuery codes are more accurate, shorter, and simpler than standard JavaScript codes. It can perform a variety of tasks.

In this section, we will learn about jQuery selectors, jQuery event methods, and some useful techniques.

Here is brief description of all selectors in jQuery:

jQuery Selectors	Symbol	Description
*	$("*")	It selects all elements at once
#id	$("#name")	It selects only that element with id="name"
.class	$(".name")	It selects only that element with class="name"
.class, .class	$(".name1,.name2")	It selects elements with the class "intro" or "demo"
Element	$("div")	It selects all <p> elements
element 1, element2, element 3	$("h1,div,p")	It selects all <h1>, and <p> elements
:first	$("p:first")	The first <p> element
:last	$("p:last")	The last <p> element
:even	$("li:even")	It selects all even elements
:odd	$("li:odd")	It selects all odd elements
:first-child	$("p:first-child")	It selects all the <p> <p> elements that are the 1st child of their parent
:first-of-type	$("p:first-of-type")	It selects all <p> elements that the first <p> element of their parent
:last-child	$("p:last-child")	It selects all <p> elements that the last child of their parent
:last-of-type	$("p:last-of-type")	It selects all <p> elements that the last <p> element of their parent
:nth-child(n)	$("p:nth-child(2)")	It selects all elements that the second child of their parent
:nth-last-child(n)	$("p:nth-last-child(2)")	It selects all <p> elements that the second child of their parent, from counting the last-child
:nth-of-type(n)	$("p:nth-of-type(2)")	It selects all <p> elements that the 2nd <p> element of their parent

(Continued)

jQuery Selectors	Symbol	Description
:nth-last-of-type(n)	$("p:nth-last-of-type(2)")	It selects all <p> elements that the 2nd <p> element of the parent, counting from the last child
:only-child	$("p:only-child")	It selects all <p> elements that the only child of their parent
:only-of-type	$("p:only-of-type")	It selects all elements that the only child, of its type, of their parent
parent > child	$("div > p")	It selects all elements that a direct child of a element
parent descendant	$("div p")	It selects all <p> elements that the descendants of a element
element + next	$("div + p")	The <p> element that are next to every elements such as
element ~ siblings	$("div ~ p")	It selects all <p> tags elements the appear after the element
:eq(index)	$("ul li:eq(3)")	The fourth element in a given list (index starts at 0)
:gt(number)	the $("ul li:gt(2)")	It can list elements with an index greater than 3
:lt(no)	$("ul li:lt(3)")	It lists elements with an index less than 3
:not(selector)	$("input:not(:empty)")	It selects all input elements that are not empty
:header	$(":header")	It selects all header elements <h1>, <h2>, and so on
:animated	$(":animated")	It selects all animated elements
:focus	$(":focus")	The element that currently has focus
:contains(text)	$(":contains('Hello')")	It selects all elements which contains the text "Hello"
:has(selector)	$("div:has(p)")	It selects all elements that have a <p> element
:empty	$(":empty")	It selects all elements that are empty
:parent	$(":parent")	It selects all elements that are a parent of another element
:hidden	$("p:hidden")	It hides all <p> elements
:visible	$("table:visible")	All visible tables
:root	$(":root")	The document's root element
:lang(language)	$("p:lang(en)")	It sets all <p> elements with a lang attribute value starting with "en"
[attribute]	$("[href]")	It selects all the elements with a href attribute

(Continued)

jQuery Selectors	Symbol	Description
[attribute=value]	$("[href='default.htm']")	It selects elements with a href attribute that equal to "default.htm"
[attribute!=value]	$("[href!='default.htm']")	It selects elements with a attribute value not equal to "default.htm"
[attribute$=value]	$("[href$='.jpg']")	It selects all elements with a attribute value ending with ".jpg"
[attribute|=value]	$("[title|='Tom']')	It selects every element with a title attribute value equal to 'Tom', or starting with 'Tom' followed by a hyphen
[attribute^=value]	$("[title^='Tom']")	It selects every element with a title attribute that starting with "Tom"
[attribute~=value]	$("[title~='tom']")	It selects every element with a title attribute that containing the specific word "tom"
[attribute*=value]	$("[title*='tom']")	It selects every element with a title attribute that contains the word "tom"
:input	$(":input")	It selects all the input elements
:text	$(":text")	It selects all the input elements with type="text"
:password	$(":password")	It selects all input elements with type="password"
:radio	$(":radio")	It selects all the input elements with type="radio"
:checkbox	$(":checkbox")	It selects all the input elements with type="checkbox"
:submit	$(":submit")	It selects all the input elements with type="submit"
:reset	$(":reset")	It selects all the input elements with type="reset"
:button	$(":button")	Its selects all the input elements with type="button"
:image	$(":image")	It selects all the input elements with type="image"
:file	$(":file")	It selects all the input elements with type="file"
:enabled	$(":enabled")	It enabled all input elements
:disabled	$(":disabled")	It disabled all input elements
:selected	$(":selected")	It selected all input elements
:checked	$(":checked")	It checked all input elements

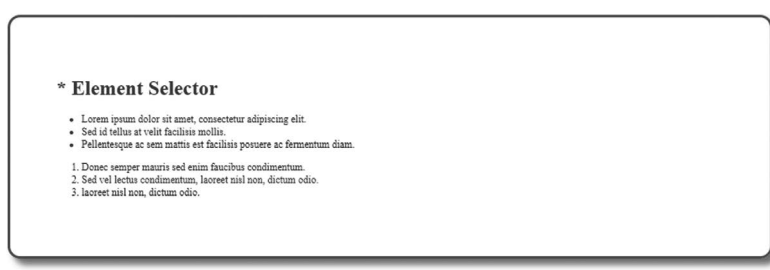
Now we will see example of some of the selector as follows, other selector will be seen in the another example.

* Selector

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script type="text/javascript" src="http://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.
min.js"></script>
  <script>
    $(document).ready(function () {
      $('#demo').click(function () {
        alert("Button is Clicked");
      });
    });
  </script>
</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
      box-shadow: 10px 10px 10px gray;
    }
    button{
      font-size: 24px;
      border: none;
      border-radius: 20px;
      padding: 20px;
      background-color:RGB(80, 243, 47)
    }
    *{
      color:blue
    }
```

```
    </style>
  </head>
 <body>

<div class="container">
    <h1> * Element Selector </h3>

    <ul>
      <li>Lorem ipsum dolor sit amet, adipiscing
elit. </li>
      <li> Sed id tellus at velit facilisis
mollis. </li>
      <li>
        Pellentesque ac sem mattis est facilisis
posuere ac fermentum diam. </li>
    </ul>

      <ol>
        <li> Donec semper mauris sed enim faucibus
condimentum. </li>
        <li>
          Sed vel lectus condimentum, laoreet nisl
non, dictum odio.</li>
        <li> laoreet nisl non, dictum odio. </li>
      </ol>

      <div class="">
        <p> </p>
      </div>
   </div>
    </body>
    </html>
```



Selector asterisk.

jQuery SELECTORS

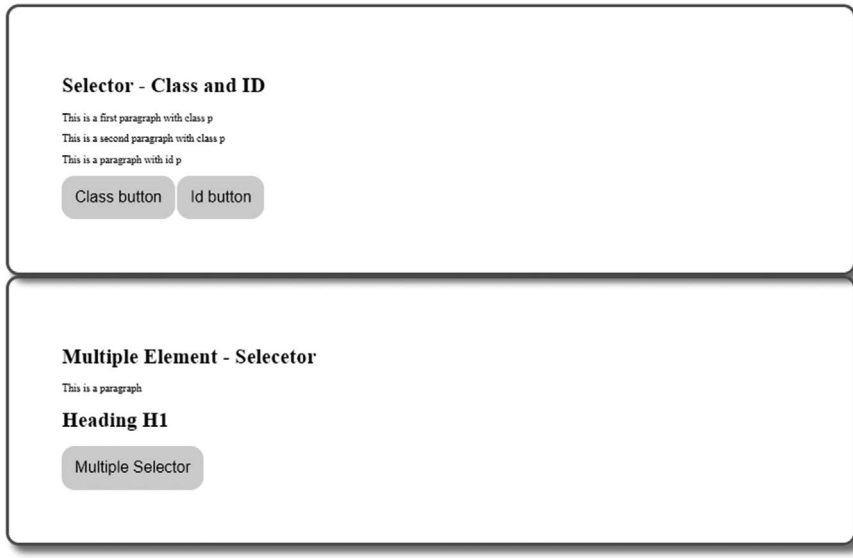
These selectors are used to select any HTML element (s) and allow you to change the HTML element (s) in the way we want. It selects HTML features in variable parameters like their name, classes, id, genres, attributes, attribute values, etc. All selectors in jQuery are selected using a special symbol, that is, the dollar symbol and brackets:

```
$ ("selector-name")
```

We have various selectors, for example, id, class, multiple elements like h1, h, p, etc. So let's have an example of each of the following.

```
<!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://ajax.
googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.
js"></script>
    <script>
      $(document).ready(function () {
        $('.para_class').click(function () {
          $(".p").css("color","red");
        });
        $('#para_id').click(function () {
          $("#p").css("color","blue");
        });
        $('#multiple_selector').click(function () {
          $("p, h1").css("background","lightpink");
        });
      });


    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
```

```
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

    </style>
  </head>
 <body>

<div class="container">
  <h1>Selector - Class and ID </h1>
        <p class="p"> This is a first paragraph with
class p</p>
        <p class="p"> This is a second paragraph with
class p </p>

        <p id="p"> This is a paragraph with id p</p>

  <button class="para_class"> Class button </button>
  <button id="para_id"> Id button </button>

   </div>
   <div class="container">
    <h1> Multiple Element - Selecetor </h1>
          <p> This is a paragraph</p>
          <h1> Heading H1 </h1>
    <button id="multiple_selector"> Multiple Selector
</button>

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



jQuery – selector.

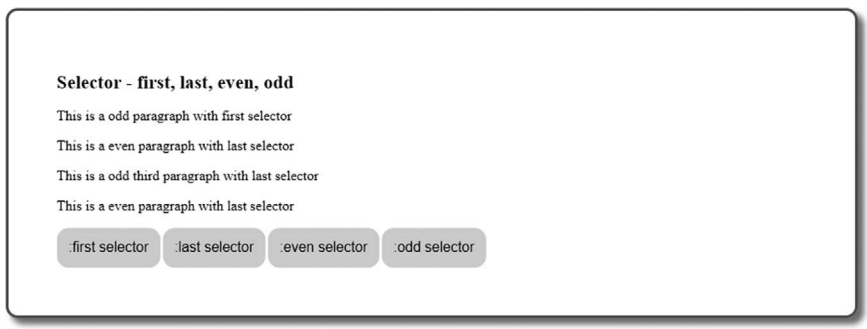
Now we are going to discuss another selector such as first, last, even, odd.

Example:

```
<!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.
min.js"></script>
    <script>
      $(document).ready(function () {
        $('.para_first').click(function () {
          $(".p:first").css("color","gray");
        });
        $('.para_last').click(function () {
          $(".p:last").css("color","blue");
        });
        $('.para_odd').click(function () {
          $("p:odd").
css("background","lightpink");
        });
        $('.para_even').click(function () {
```

```
          $("p:even").css("background","lightco
ral");
        });
      });


    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;

      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
    p{
      font-size: 24px;
    }
    </style>
  </head>
 <body>

<div class="container">
  <h1> Selector - first, last, even, odd </h1>
        <p class="p"> This is a odd paragraph with
first selector </p>
        <p class="p"> This is a even paragraph
with last selector </p>
        <p class="p"> This is a odd third
paragraph with last selector </p>
```

```
        <p class="p"> This is a even paragraph
with last selector </p>

  <button class="para_first"> :first selector
</button>
  <button class="para_last"> :last selector
</button>
  <button class="para_even"> :even selector
</button>
  <button class="para_odd"> :odd selector
</button>

   </div>

    </body>
    </html>
```



jQuery – selectors 2.

jQuery METHODS

You learned about jQuery selectors in the previous section. The jQuery selector finds a DOM element (s) and wraps it up in a jQuery object. For example, document.getElementById () in JavaScript will replace the DOM object and $ ("# id") will replace the jQuery object.

The document.getElementById function returns the div element while the jQuery selector returns the jQuery object which is a div of the div object. So now, you can call the jQuery methods of the jQuery object returned by the jQuery selector. jQuery offers a variety of different functions, for example, manage DOM, events, ajax, etc.

There are various jQuery methods. The complete list is given below:

- DOM Manipulation
- Traversing
- CSS
- Attributes
- Events
- Effects
- Dimensions
- Forms
- AJAX
- Core
- Data
- Miscellaneous
- Utilities

Now our first topic is DOM Manipulation in jQuery. Let's understand this in brief.

DOM MANIPULATION METHODS

DOM stands for Document Object Model. It can only be understood as a node tree created by the browser. Each of these nodes has its own properties and functions that can be changed using JavaScript.

The ability to change DOM is one of the unique and useful JavaScript abilities. It provides various methods to add, edit, or delete element(s) on the page.

The following lists some methods to add/remove new DOM elements.

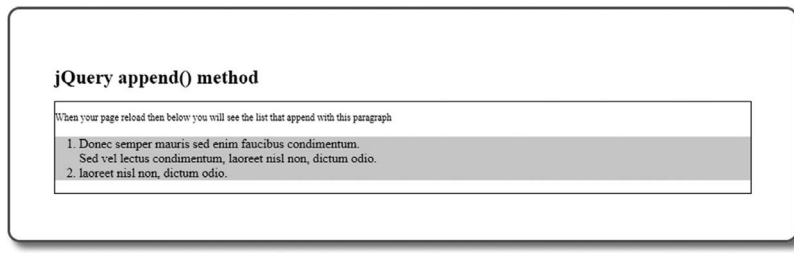
- append(): It can insert content to the end of the element(s) that is specified by a selector.

  Example:

```
<!DOCTYPE html>
<html>
```

```
  <head>
    <script type="text/javascript" src="https:
//ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {

$('p').append('<div style="background-color:
pink ; font-size:21px"> <ol> <li> Donec semper
mauris sed enim faucibus condimentum. </li> Sed
vel lectus condimentum, laoreet nisl non, dictum
odio. <li> laoreet nisl non, dictum odio. </li>
</ol> </div>');

});
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }

      .div1{
      border: 1px solid;
      margin: 2px 0 2px 0;
    }
    </style>
  </head>
 <body>

   <div class="container">
    <h1>jQuery append() method </h1>
      <div class="div1">
```

```
        <p> When your page reload then below you
will see the list that append
          with this paragraph </p>
      </div>
   </div>
    </body>
    </html>
```



Append method.

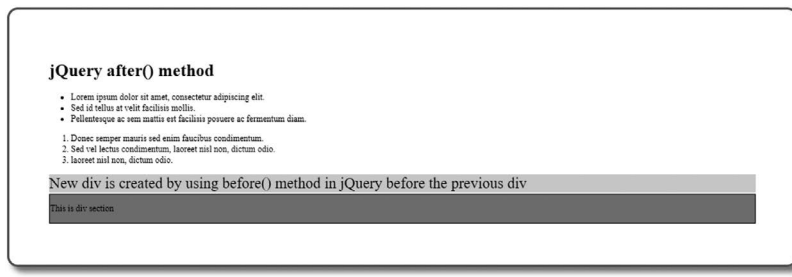
- before(): It can insert content (new or existing DOM elements) before an element which is specified by a selector.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {

$('.div1').before('<div style="background-color:
pink ; font-size:28px"> New div is created by
using before() method in jQuery before the
previous div</div>');

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

```
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
      .div1{
      border: 1px solid;
      background-color:red;
      margin: 2px 0 2px 0;
    }
    </style>
  </head>
 <body>

   <div class="container">
    <h1>jQuery after() method </h1>
    <ul>
      <li> Lorem ipsum dolor sit, consectetur
adipiscing elit. </li>
      <li> Sed id tellus at velit facilisis
mollis. </li>
      <li>
        Pellentesque ac sem mattis est facilisis
posuere ac fermentum diam. </li>
    </ul>

      <ol>
        <li> Donec semper mauris sed enim
faucibus condimentum. </li>
```

```
        <li>
          Sed vel lectus condimentum, laoreet
nisl non, dictum odio.</li>
        <li> laoreet nisl non, dictum odio. </li>
      </ol>

      <div class="div1">
        <p> This is div section </p>
   </div>
      </div>
    </body>
    </html>
```



Before method.

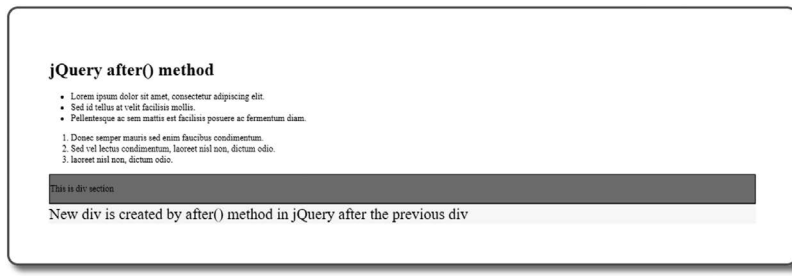
- after(): It can insert content (new or existing DOM elements) after an element(s) which is specified by a selector.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {

$('.div1').after('<div style="background-color:
yellow ; font-size:28px"> New div is created by
using after() method in jQuery the previous
div</div>');
```

```
});
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
      .div1{
      border: 1px solid;
      background-color:red;
      margin: 2px 0 2px 0;
    }
    </style>
  </head>
 <body>

   <div class="container">
    <h1>jQuery after() method </h1>
    <ul>
      <li>Lorem ipsum dolor sit, consectetur
adipiscing elit. </li>
      <li> Sed id tellus at velit facilisis
mollis. </li>
      <li>
        Pellentesque ac sem mattis est facilisis
posuere ac fermentum diam. </li>
    </ul>
```

```
      <ol>
        <li> Donec semper mauris sed enim
faucibus condimentum. </li>
        <li>
          Sed vel lectus condimentum, laoreet
nisl non, dictum odio.</li>
        <li> laoreet nisl non, dictum odio. </li>
      </ol>

      <div class="div1">
        <p> This is div section </p>
      </div>
   </div>
    </body>
    </html>
```



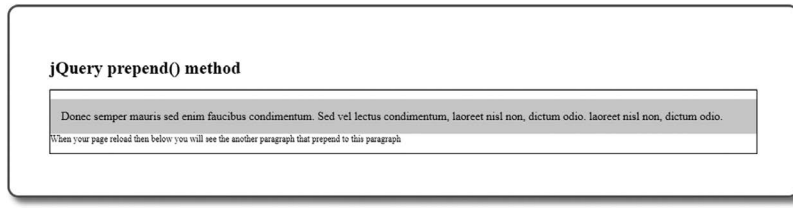
After method.

- prepend(): It can insert content at the beginning of an element(s) specified by a selector.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {
```

```
$('p').prepend('<div style="background-color:
pink ; font-size:21px; padding: 20px"> Donec
semper mauris sed enim faucibus condimentum. Sed
vel lectus condimentum, laoreet nisl non, dictum
odio. laoreet nisl non, dictum odio.</div>');

});
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }

      .div1{
      border: 1px solid;
      margin: 2px 0 2px 0;
    }
    </style>
  </head>
 <body>

   <div class="container">

    <h1>jQuery prepend() method </h1>
      <div class="div1">
        <p> When your page reload then below you
will see the another paragraph that prepend to
          this paragraph </p>
      </div>
   </div>
    </body>
    </html>
```



Prepend method.

- remove(): It can removes element(s) from DOM which is specified by selector.
- empty(): It can remove the child elements from the selected element.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {
       $('#btn').click(function () {
        $('.div2').remove();
       });

       $('button').click(function () {
        $('.div1').empty();
       });

          });
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding: 5%;
      }
      .container{
```

```
        border-radius: 20px;
        padding :5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }

      .div1{
      border: 1px solid;
      height:100px;
      background-color: palegreen;
      margin: 2px 0 2px 0;
    }

    .div2{
      border: 1px solid;
      height: 100px;
      background-color: pink;
      margin: 2px 0 2px 0;
    }

    </style>
  </head>
 <body>
<div class="container">

    <h1>jQuery remove() and empty() method </h1>
      <div class="div1">
        <p> This div will not remove </p>
      </div>
      <div class="div2">
        <p> Remove this div </p>
      </div>
      <button id="btn"> Click to Hide </button>
   </div>
    </body>
    </html>
```

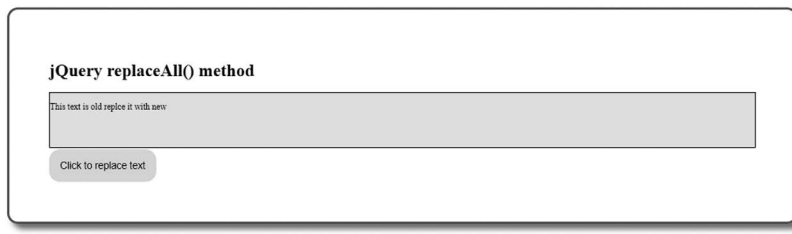
- replaceAll(): It can replace target element(s) with specified element.

Example:

```
<!DOCTYPE html>
<html>
```

```
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {
       $('#btn').click(function () {
        $('<div class="div2"> This text is new
</div>').replaceAll('.div1');
       });

       $('button').click(function () {
        $('.div1').empty();
       });

          });
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }

      .div1{
      border: 1px solid;
      height:100px;
      background-color: palegreen;
      margin: 2px 0 2px 0;
    }

    .div2{
      border: 1px solid;
      height: 100px;
      background-color: pink;
      margin: 2px 0 2px 0;
    }
```

```
    button{
        font-size: 18px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color: gainsboro
      }

    </style>
  </head>
 <body>
<div class="container">
    <h1>jQuery replaceAll() method </h1>
      <div class="div1">
        <p> This text is old replce it with new
</p>
      </div>
      <button id="btn"> Click to replace text
</button>
   </div>
    </body>
    </html>
```



ReplaceAll method.

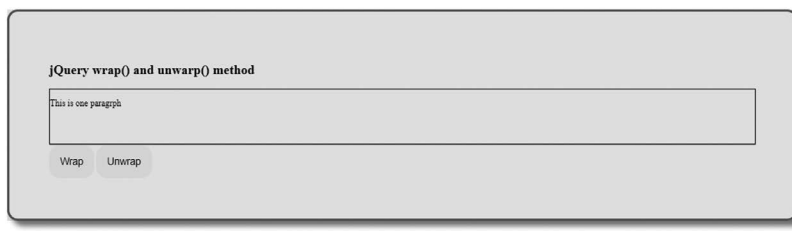
- wrap(): It can wrap an HTML structure around each element which is specified by the selector.
- .unwrap(): It can remove the parents of the set of elements from the DOM.

Example:

```
<!DOCTYPE html>
<html>
```

```
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {
       $('#btn1').click(function () {
        $('div').wrap('<section> </section');
       });

       $('#btn2').click(function () {
        $('div').unwrap();
       });
          });
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }

      .div1{
      border: 1px solid;
      height:100px;
      margin: 2px 0 2px 0;
    }

    section{
      background-color: pale green;

    }
    button{
        font-size: 18px;
        border: none;
        border-radius: 20px;
```

```
        padding: 20px;
        background-color: gainsboro
      }

    </style>
  </head>
 <body>
<div class="container">
  <h1> jQuery wrap() and unwarp() method </h1>
  <div class="div1">
    <p> This is one paragraph </p>
  </div>
  <button id="btn1"> Wrap</button>
  <button id="btn2"> Unwrap</button>

</div>

 </body>
</html>
```



Wrap and unwrap method.

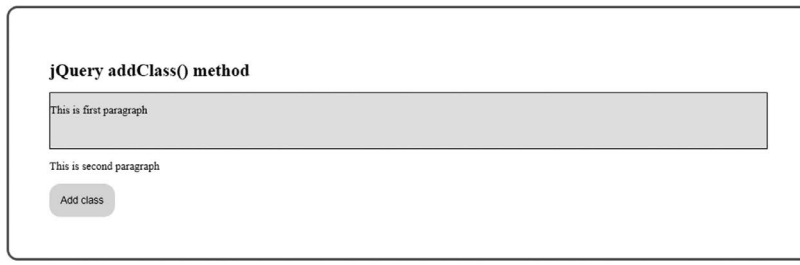
- .addClass(): It adds the specified classes to each element in the set of matched elements.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>
    <script>
     $(document).ready(function () {
```

```
        $('#btn1').click(function () {
         $(".div1").addClass("intro");
        });

           });
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }
    .intro{
      border: 1px solid;
      height:100px;
      margin: 2px 0 2px 0;
      background-color: palegreen;

    }
    button{
        font-size: 18px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color: gainsboro
      }

    </style>
  </head>
 <body>
<div class="container">
  <h1> jQuery addClass() method </h1>
  <div class="div1">
```

```
    <p> This is first paragraph</p>
  </div>
  <div class="div2">
    <p> This is second paragraph </p>
  </div>
  <button id="btn1"> Add class </button>

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



Add class method.

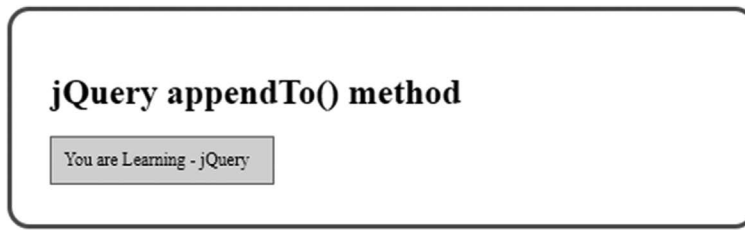
- .appendTo(): It can insert every element in the set of elements to the end of the target.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="http://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>
    <script>
     $(document).ready(function () {
       $('#btn1').click(function () {
        $(".div2").appendTo(".app");
       });

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

```
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }
    .app{
      background: light green;

    }
    button{
        font-size: 18px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color: gainsboro
      }

    #intro {
    background: light green;
    border: 2px solid green;
    padding: 10px;
    width: 180px;
    }
   </style>
   <script src="http://code.jquery.com/
jquery-1.10.2.js">
   </script>
 </head>
 <body>
    <div class="container">
   <span> jQuery </span>
   <h1> jQuery appendTo() method </h1>
  <div id="intro"> You are Learning - </div>
  </div>
```

```
    <script>
     $("span").appendTo("#intro");
    </script>
  </body>

  </body>
</html>
```



appendTo method.

- .attr(): It gets the value of an attribute for the element in the set of elements or a set of more than one attribute for every element.

Example:

- .clone(): It can create a deep copy of the set of elements.
- .css(): It gets the value of a style property for an element in the set of elements or a set of one or more of its properties for every element.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
```

```
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

</style>
      <script>
$(document).ready(function(){
 $("button").click(function(){
  $('.result').text(("Background color = " +
$("p").css("background-color")));
 });
});

</script>
</head>
<body>

  <div class="container">
    <h1> jQuery css() method </h1>
    <p style="background-color:pink">This is a
paragraph.</p>
    <p class="result"></p>
    <br>
    <button>Get CSS</button>

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

CSS method.

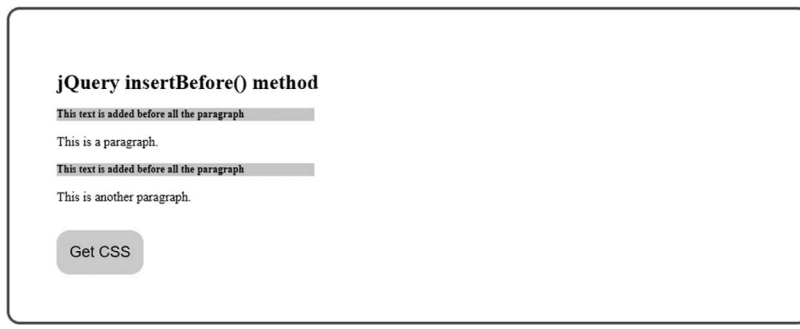
- .detach(): It can remove the set of matched elements from the DOM.
- .hasClass(): It determines whether any of the elements are assigned to the given class.
- .height(): It gets the current height for the first element in the set of elements or set the height of every matched element.
- .html(): It gets the contents of the first element in the set of elements or set the contents of every matched element.
- .innerHeight(): It gets the computed inner height for the first element in the set of elements or set inner height of every matched element.
- .innerWidth(): It gets the computed inner width for the first element in the set of elements or set inner width of every matched element.
- .insertAfter(): It can insert an element in the set of elements after the target.
- .insertBefore(): It inserts every element in the set of elements before the target.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
```

```
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

</style>
<script>
  $(document).ready(function(){
   $("button").click(function(){
    $("<h4 style='background-color:pink;
width:400px'> This text is added before all the
paragraph </h4>").insertBefore("p");
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery insertBefore() method </h1>
    <p >This is a paragraph.</p>
    <p >This is another paragraph.</p>
```

```
    <br>
    <button>Get CSS</button>

  </div>


 </body>


 </body>
</html>
```



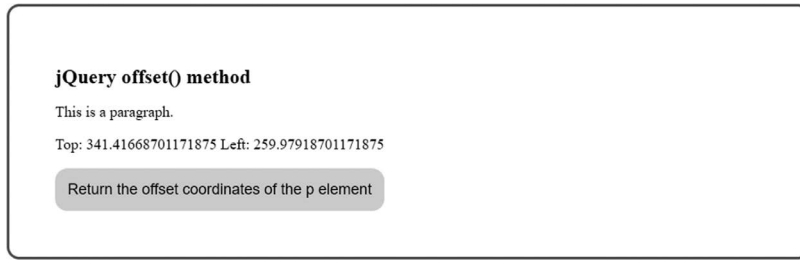
insertBefore method.

- .offset(): It gets the current coordinates of the first element, set the coordinates of the element, in the set of matched elements, relative to the document.

  Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
```

```
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

p{
  font-size:24px
}
</style>
<script>
  $(document).ready(function(){
   $("button").click(function(){
    var x = $("p").offset();
    $('.result').text(("Top: " + x.top + " Left:
" + x.left));
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery offset() method </h1>
    <p>This is a paragraph.</p>
<p class="result"> </p>
<button> You can return the offset coordinates
of the p element</button>
  </div>
 </body>
</html>
```



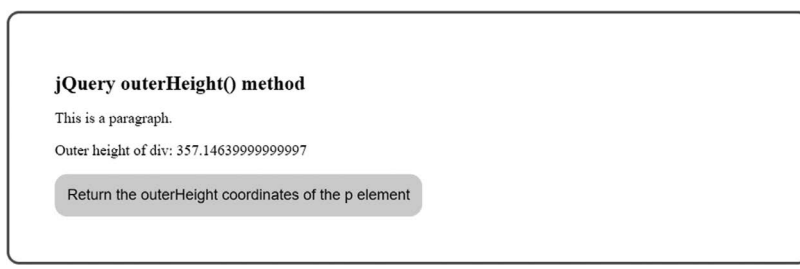
Offset method.

- .outerHeight(): It gets the current calculate outer height for the first element in the set of elements or outer height of every element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 20px;
     }

  button{
      font-size: 24px;
      border: none;
      border-radius: 20px;
      padding: 20px;
      background-color:RGB(80, 243, 47)
    }
```

```
p{
  font-size:24px
}
</style>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $('.result').text(("Outer height of div: " +
$("div").outerHeight()));
 });
});
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery outerHeight() method </h1>
    <p>This is a paragraph.</p>
<p class="result"></p>
<button> Return the outerHeight coordinates of
the p element </button>


  </div>


 </body>


 </body>
</html>
```



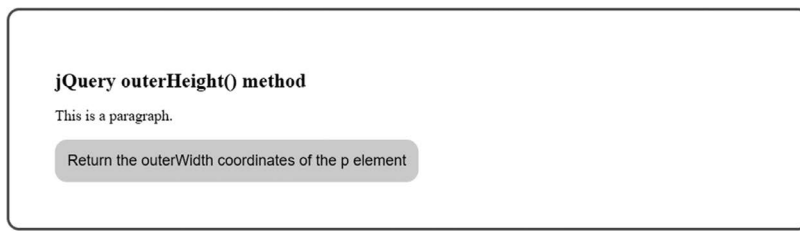
outerHeight method.

- .outerWidth(): It gets the current computed outer width for the first element in the set of matched elements or set the outer width of every matched element.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

p{
  font-size:24px
}
</style>
<script>
```

```
$(document).ready(function(){
 $("button").click(function(){
  $('.result').text(("Outer Width of div: " +
$("div").outerWidth()));
 });
});
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery outerHeight() method </h1>
    <p> This is a paragraph. </p>
<p class="result"></p>
<button> Return the outerWidth coordinates of
the p element </button>
  </div>
 </body>
</html>
```



outerWidth method.

- .position(): It gets the current coordinates of the first element in the set of elements, relative to the offset parent.

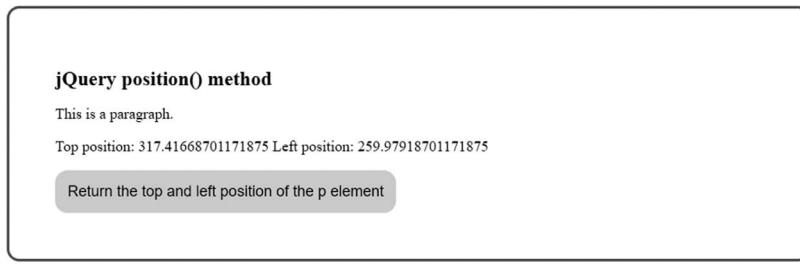
  It returns the position of the first matched element. It returns an object with two properties such as the top and left positions in pixels.

  Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
```

```
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

p{
  font-size:24px
}
</style>
<script>
  $(document).ready(function(){
   $("button").click(function(){
    var x = $("p").position();
    $('.result').text((" Top position: " + x.top
+ " Left position: " + x.left));
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery position() method </h1>
    <p> This is a paragraph. </p>
```

```
    <p class="result"></p>
    <button> It return the top and left position
of the element </button>
  </div>
 </body>
</html>
```



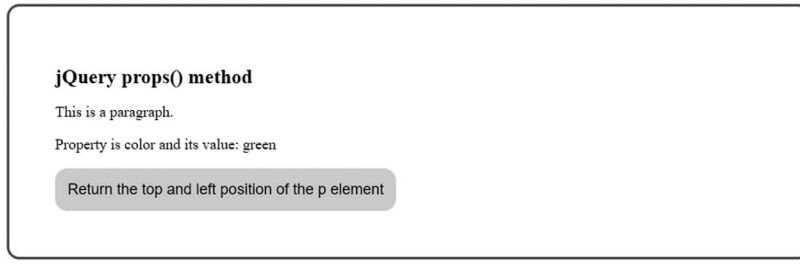
position method.

- .prop(): It gets the value of a property for the element in the set of matched elements or set more than properties for every matched element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
```

```
        p{
          font-size: 20px;
        }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

p{
  font-size:24px
}
</style>
<script>
 $(document).ready(function(){
      $("button").click(function(){
      var $x = $(".result");
      $x.prop("color", "green");
      $x.append(" Property is color and its
value: " +          $x.prop("color"));
      });
    });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery props() method </h1>
    <p> This is a paragraph. </p>
    <p class="result"></p>
    <button> It is return the top and left
position of the p element </button>
  </div>
 </body>
</html>
```



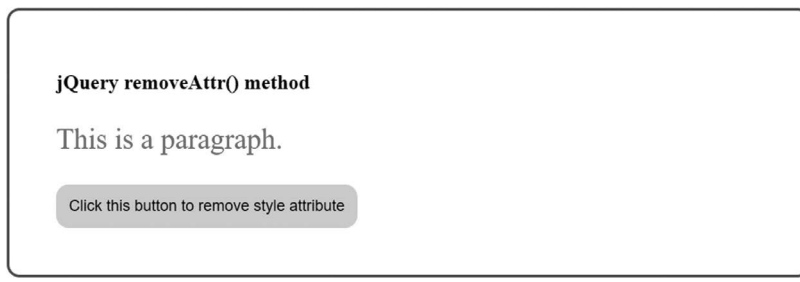
props method.

- removeAttr(): It can remove an attribute from each element in the set of elements.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 20px;
     }

  button{
      font-size: 24px;
      border: none;
```

```
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

</style>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $("p").removeAttr("style");
 });
});
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery removeAttr() method </h1>
    <p style="font-size:45px; color:red"> This
is a paragraph. </p>
    <p class="result"></p>
    <button> Click this button to remove style
attribute </button>
  </div>
 </body>
</html>
```



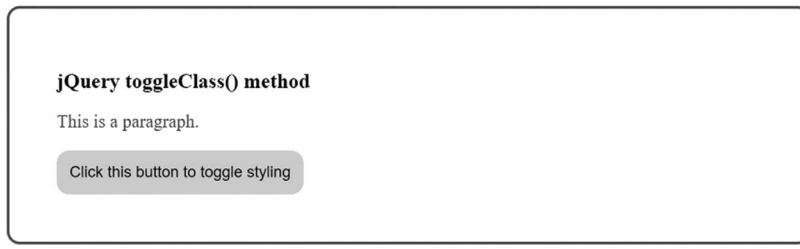
removeAttr method.

- toggleClass(): It can add or remove one or more classes from each element in the set of elements, depending on either the class presence or the value of the state argument.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

      .main {
 font-size: 28px;
 color:brown;
}
</style>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $("p").toggleClass("main");
 });
});
```

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

  <div class="container">
    <h1> jQuery toggleClass() method </h1>
    <p> This is a paragraph. </p>
    <button> Click this button to toggle styling
</button>
  </div>
 </body>
</html>
```



toggleCLass method.

- .width(): It gets the current computed width for the first element in the set of matched elements or set the width of every matched element.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
```

```
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

      .main {
 font-size: 28px;
 color:brown;
}
</style>
<script>
    $(document).ready(function(){
$("button").click(function(){
  $('.result').text($("div").width());
});
    });

  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery width() method </h1>
    <p> This is a paragraph. </p>
    <p class="result"> </p>
    <button> Click this button to get width </
button>
  </div>
 </body>
</html>
```

width method.

- .text(): It gets the combined text contents of an element in the set of elements, including the descendants or set the text contents of the elements. It returns the text content of the selected elements. When the method is used to return content, it returns the content of all matched elements (HTML markup will be removed).

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script type="text/javascript" src="https://
ajax.googleapis.com/ajax/libs/jquery/3.6.0/
jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }
```

```
    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

      .main {
 font-size: 28px;
 color:brown;
}
</style>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $("p").text("New text!");
 });
});
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery text() method </h1>
    <p> This is a paragraph. </p>
    <button> Click this button to get new text
</button>
  </div>
 </body>
</html>
```

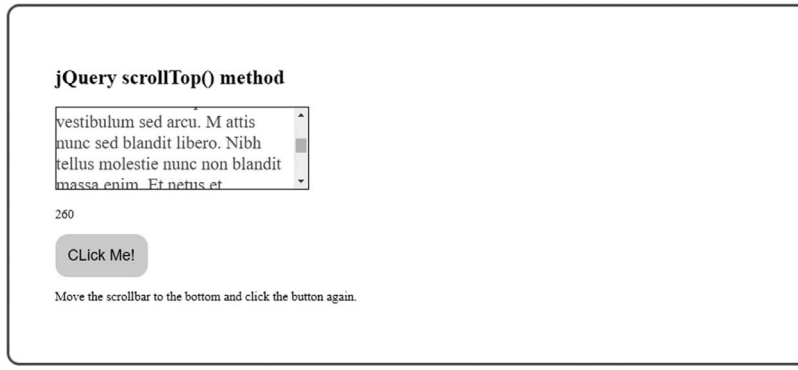
text method.

- .scrollTop(): It gets the current vertical position of the scroll bar for the first element in the set of matched elements or set the vertical position of the scroll bar for every matched element.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

      .scroll {
        left: 20px;
 font-size: 28px;
 color: brown;
}
</style>
```

```
<script>
  $(document).ready(function(){
    $("button").click(function(){
 $('.result').text(($("p").scrollTop()));
});
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery scrollTop() method </h1>
    <p class="scroll" style="border:1px solid
black;width:400px;height:130px;overflow:auto">
At urna condimentum mattis pellentesque id nibh
tortor id.
      Quis blandit turpis cursus in hac. Sed
blandit libero volutpat <br> sed
      cras ornare arcu. Magna ac placerat
vestibulum lectus mauris ultrices er
      os in cursus. At tempor commodo
ullamcorper a lacus <br> vestibulum sed arcu. M
      attis nunc sed blandit libero. Nibh tellus
molestie nunc non blandit massa enim.
       Et netus et malesuada fames. At risus
viverra <br> adipiscing at in. Vulputate mi sit
amet
       mauris commodo quis. Varius sit amet
mattis enim nulla aliquet porttitor
       lacus. Aliquam vestibulum morbi blandit
cursus risus. Pretium quam vulputate dignis

    </p>
    <p class="result"></p>
    <button> Click Me! </button>
    <p>Move the scrollbar to the bottom and
click the button again.</p>

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



scrollTop method.

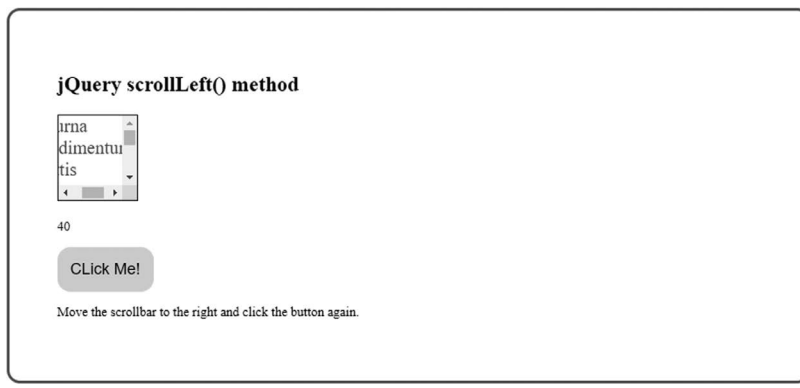
- .scrollLeft(): It gets the current horizontal position of the scroll bar for the first element in the set of matched elements or set the horizontal position of the scroll bar for every matched element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 20px;
     }
```

```
    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

      .scroll {
        left: 20px;
 font-size: 28px;
 color:brown;
}
</style>
<script>
  $(document).ready(function(){
    $("button").click(function(){
 $('.result').text(($("p").scrollLeft()));
});
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery scrollLeft() method </h1>
    <p class="scroll" style="border:1px solid
black;width:120px;height:130px;overflow:auto">
At urna condimentum mattis pellentesque id nibh
tortor id.
      Quis blandit turpis cursus in hac. Sed
blandit libero volutpat sed
      cras ornare arcu. Magna ac placerat
vestibulum lectus mauris ultrices er
      os in cursus. At tempor commodo
ullamcorper a lacu vestibulum sed arcu. M
      attis nunc sed blandit libero. Nibh tellus
molestie nunc non blandit massa enim.
       Et netus et malesuada fames. At risus
viverra adipiscing at in. Vulputate mi sit amet
       mauris commodo quis. Varius sit amet
mattis enim nulla aliquet porttitor
```

```
       lacus. Aliquam vestibulum morbi blandit
cursus risus. Pretium quam vulputate dignis

    </p>
    <p class="result"></p>
    <button> Click Me! </button>
    <p> Now, move the scrollbar to the right
side and click the button again.</p>

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



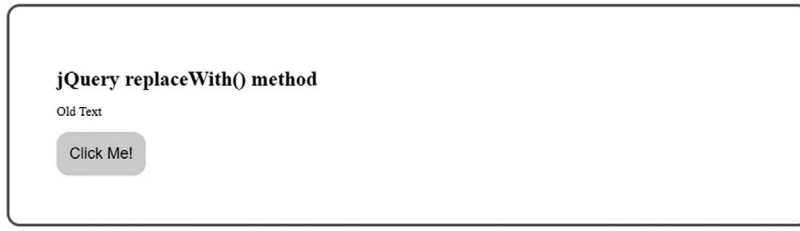
scrollLeft method.

- .replaceWith(): It replaces each element in the set of elements with the provided new content and returns the set of elements that was removed.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
```

```
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
</style>
<script>
  $(document).ready(function(){
    $("button").click(function(){
 $("p:first").replaceWith("<p> New Text! </p>");
});
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery replaceWith() method </h1>
    <p> Old Text </p>
    <p class="result"> </p>
    <button> Click Me! </button>
  </div>
 </body>
</html>
```



replaceWidth method.

jQuery EVENTS

The jQuery event is the result of a jQuery (JavaScript) action. Once these events are launched, you can use the custom functionality to do exactly what you want with the event. These custom functions are called Event Holders. jQuery Library provides ways to handle all DOM events and makes managing complete events much easier than we have in JavaScript.

jQuery provides a powerful and complete event API. The jQuery event API provides folding methods that allow multiple events in JavaScript. The jQuery event API also provides the ability to attach events that they do not explicitly support in their event modes.

The event API has the ability to apply events to objects that may not even exist document yet. Events can be neatly organized and labeled within jQuery, another feature that offers more than just the basics provided by JavaScript. Your events can be neat; it is organized into categories with a name, which makes it easier to manage events. Named events also make it possible to remove them easily. In this section, you learn everything you need to know to work with the jQuery event API. You learn to use jQuery event wrap methods such as click () or scroll up (). You read again how to use methods such as opening () and closing (). You can use the opening () and closing () methods to paste the event manager function for any event, whether a traditional JavaScript event or a custom event you created. The opening () and closing () modes can attach events to their elements that may not be present in the document yet. Additionally, on () and off () can compose and edit events, and is useful if you need to manage or delete events as easily as you create them. You learn how to create completely custom events for your Apps due to the trigger () method and methods of opening () and closing (). Custom events can make your Apps more popular and more comfortable.

Syntax of event methods: Most DOM events have the same jQuery method. To provide events by clicking on all the categories on the page, do this:

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

The next step describes what happens when an event occurs. You must transfer the function to the event.

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

jQuery Library provides ways to handle all DOM events and makes managing complete events much easier than we have in JavaScript. The examples of some common occurrences are as follows:-

1. Clicking the mouse
2. The web page is loading
3. Taking a mouse over an object
4. Submitting HTML form
5. Pressing a key on your keyboard, etc.

MOUSE EVENTS

This section will explain different mouse events occurring based on positions on a particular HTML element.

Mouse Events in jQuery:

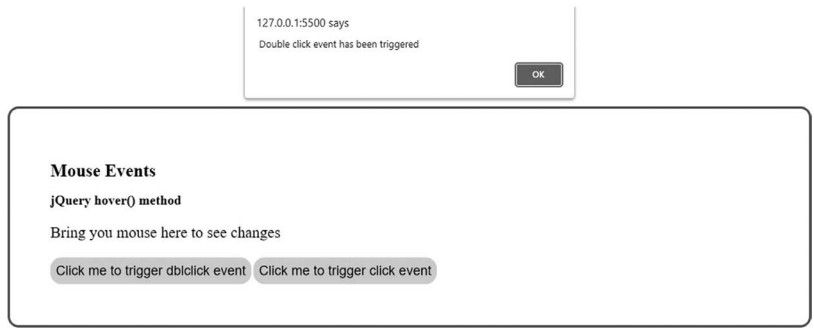
- Click
- dblclick

Example:

```
<!DOCTYPE html>
<html>
<head>
```

```
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 28px;
       }
           button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 10px;
        background-color:RGB(80, 243, 47)
      }


</style>
<script>
  $(document).ready(function() {
        $(".btn1").click(function() {
          $("p").dblclick();
        });
      });
      $(document).ready(function() {
        $(".btn2").click(function() {
          $("p").dblclick();
        });
      });
</script>

</head>
```

```
<body>

  <div class="container">
    <h1> Mouse Events </h1>
    <h2> jQuery double and single click method
</h2>
    <p id="key"> Bring you mouse here to see
changes </p>
    <button class=".btn1" ondblclick="alert('
Double click event has been triggered')">Click
me to trigger dblclick event</button>
    <button class=".btn2" onclick="alert(' Click
event has been triggered')"> Click me to trigger
click event </button>

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



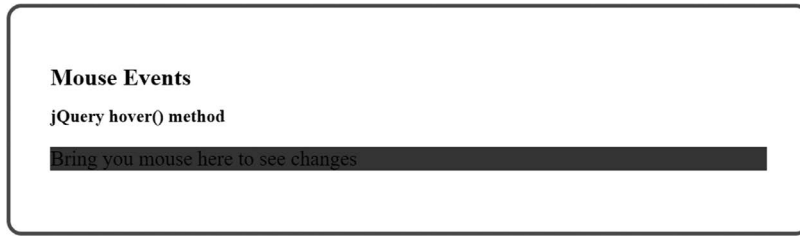
Mouse event – click and double click.

- hover

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
```

```
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 28px;
       }


</style>
<script>
$(document).ready(function(){
 $("p").hover(function(){
  $(this).css("background-color", "purple");
  }, function(){
  $(this).css("background-color", "green");
 });
});
</script>

</head>
<body>

  <div class="container">
    <h1> Mouse Events </h1>
    <h2> jQuery hover() method </h2>
    <p id="key"> Bring you mouse here to see
changes </p>
  </div>
 </body>
</html>
```



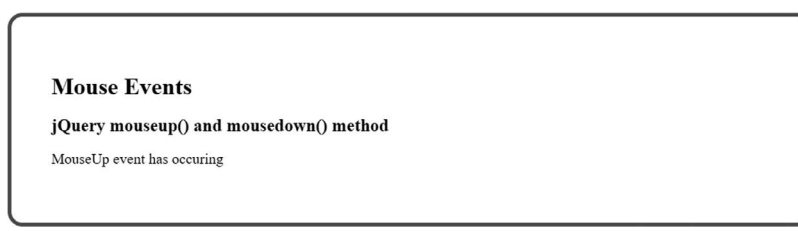
Mouse event - hover method.

- mousedown
- mouseup

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 20px;
       }


</style>
<script>
  $("document").ready(function()
```

```
 {
   $("#key").mouseup(up);
   $("#key").mousedown(down);
   function up()
    {
     $("#key").text(" MouseUp event has
occuring");
    }
   function down()
    {
     $("#key").text(" Mousedown event has
occuring");
    }
 });
</script>

</head>
<body>

  <div class="container">
    <h1> Mouse Events </h1>
    <h2> jQuery mouseup() and mousedown() method
</h2>
    <p id="key"> Click here to see changes </p>
    <p class="result"></p>
  </div>
 </body>
</html>
```



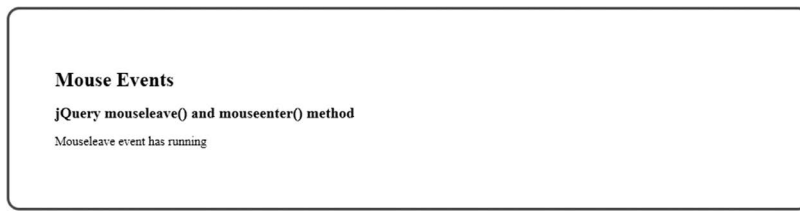
Mouse event – mouseup and mousedown method.

- mouseenter
- mouseleave

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
          font-size: 20px;
       }


</style>
<script>
  $("document").ready(function () {
    $("#key").mouseenter(enter);
    $("#key").mouseleave(leave);
    function enter() {
      $("#key").text(
        "Mouseenter event has running");
    }
    function leave() {
      $("#key").text(
        "Mouseleave event has running");
    }
  });
</script>

</head>
<body>
```

```
  <div class="container">
    <h1> Mouse Events </h1>
    <h2> jQuery mouseleave() and mouseenter()
method </h2>
    <p id="key"> Bring mouse here </p>
    <p class="result"></p>
  </div>
 </body>
</html>
```



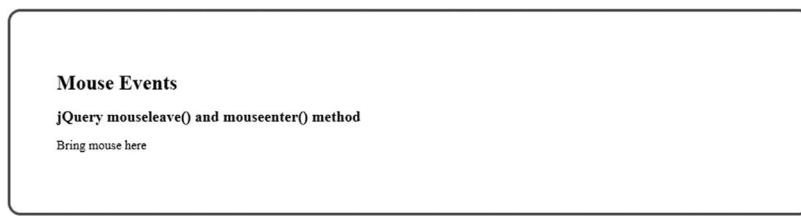
Mouse event – mouseleave and mouseenter method.

- mouseover
- mouseout

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
```

```
        p{
          font-size: 20px;
        }


</style>
<script>
  $("document").ready(function()
 {
   $("#key").mouseover(over);
   $("#key").mouseout(out);
   function over()
    {
     $("#key").text(" Mouseover event has
occuring");
    }
   function out()
    {
     $("#key").text(" Mouseout event has
occuring");
    }
 });
</script>

</head>
<body>

  <div class="container">
    <h1> Mouse Events </h1>
    <h2> jQuery mouseover() and mouseout()
method </h2>
    <p id="key"> Bring mouse here </p>
    <p class="result"></p>
  </div>
 </body>
</html>
```



Mouse events – mouseover and mouseout.

KEYBOARD EVENTS

All the different actions that a page can respond to, are called events (doing tasks). An event represents the moment when something happens. Here are some examples of keyboard handling:

- The moving a mouse over an element
- By selecting a radio button
- By clicking on an element

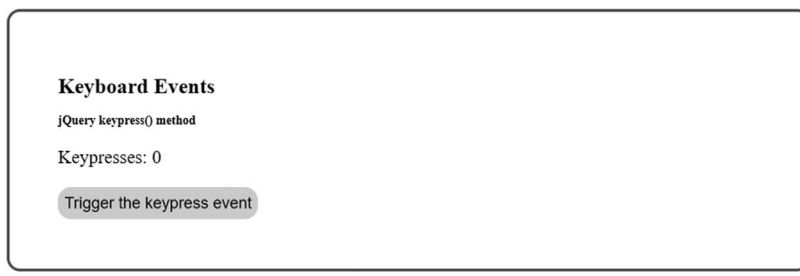
Also, we have some methods to perform keyboard actions like:

- Keypress: It can bind an event handler to the "keypress" Js event, or trigger that event on an element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }
         button{
      font-size: 24px;
      border: none;
      border-radius: 20px;
```

```
        padding: 10px;
        background-color:RGB(80, 243, 47)
      }


</style>
<script>
i = 0;
$(document).ready(function(){
 $("p").keypress(function(){
  $("span").text(i += 1);
 });
 $("button").click(function(){
  $("p").keypress();
 });
});
</script>

</head>
<body>

  <div class="container">
    <h1> Keyboard Events </h1>
    <h3> jQuery keypress() method </h3>
    <p>Keypresses: <span>0</span></p>

    <button>Trigger the keypress event</button>


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



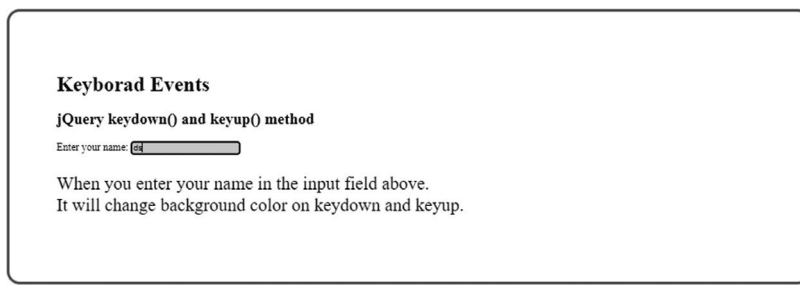
Keyboard event – keypress method.

- Keydown: It can bind an event handler to the "keydown" js event, or trigger that event on an element.
- keyup: It can bind an event handler to the "keyup" js event, or trigger that event on an element.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="https://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>

  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
      }
       p{
         font-size: 28px;
       }
           button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 10px;
        background-color:RGB(80, 243, 47)
      }


</style>
<script>
 $(document).ready(function(){
 $("input").keydown(function(){
  $("input").css("background-color", "yellow");
```

```
 });
 $("input").keyup(function(){
  $("input").css("background-color", "pink");
 });
});
</script>

</head>
<body>

  <div class="container">
    <h1> Mouse Events </h1>
    <h2> jQuery keydown() and keyup() method
</h2>
    Enter your name: <input type="text">

<p>When you enter your name in the input field
above.<br>
  It will change background color on keydown and
keyup.</p>

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



Keyboard event – keydown and keyup method.

FORM EVENTS

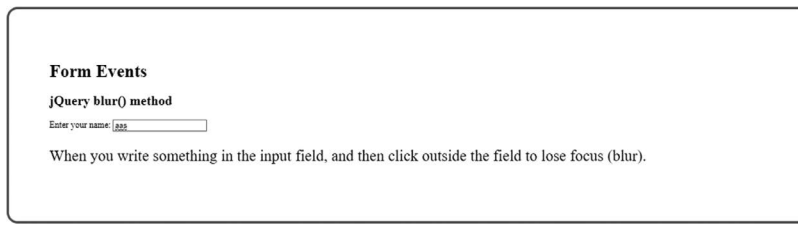
- .blur(): It binds an event handler to the "blur" js event, or trigger that event on an element. The blur event is an element when it loses focus. This event is applicable to form elements, such as <input>.

An element can lose focus via keyboard commands, such as the Tab key or by mouse clicks elsewhere on the page.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }
         button{
      font-size: 24px;
      border: none;
      border-radius: 20px;
      padding: 10px;
      background-color:RGB(80, 243, 47)
    }

</style>
<script>
$(document).ready(function(){
  $("input").blur(function(){
 alert("This input field has lost its focus.");
});});
</script>

</head>
```

```
<body>

<div class="container">
  <h1> Form Events </h1>
  <h2> jQuery blur() method </h2>

Enter your name: <input type="text">
<p>When you write something in the input field,
and then click outside the field to lose focus
(blur).</p>

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



Form event – blur method.

- .change(): It binds an event handler to the "change" js event, or trigger that event on an element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
```

```
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }
         button{
      font-size: 24px;
      border: none;
      border-radius: 20px;
      padding: 10px;
      background-color:RGB(80, 243, 47)
    }

</style>
<script>
$(document).ready(function(){
 $("input").change(function(){
  $(".result").text("The text has been
changed.");
 });
});
</script>

</head>
<body>

<div class="container">
  <h1> Form Events </h1>
  <h2> jQuery change() method </h2>

Enter your name: <input type="text">
<p class="result"></p>

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

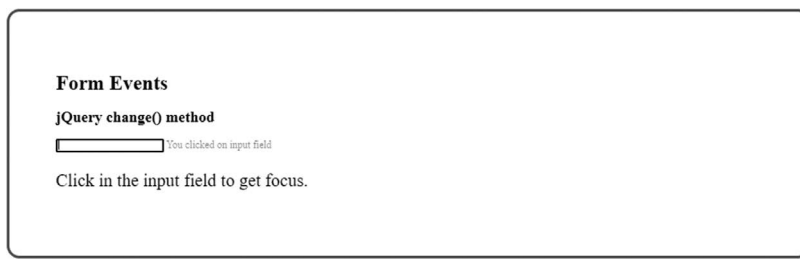
Form event – change method.

- .focus(): It binds an event handler to the "focus" js event, or trigger that event on an element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }
         button{
      font-size: 24px;
      border: none;
      border-radius: 20px;
      padding: 10px;
```

```
        background-color:RGB(80, 243, 47)
      }

</style>
<script>
$(document).ready(function(){
  $("input").focus(function(){
 $("span").css("display", "inline").
fadeOut(2000);
});
});
</script>

</head>
<body>

<div class="container">
  <h1> Form Events </h1>
  <h2> jQuery focus() method </h2>
 <input type="text">
 <span> You clicked on input field </span>
  <p> Click on the input field to get focus.
</p>
</div>
</body>
</html>
```



Form event – focus method.

- .focusin(): It binds an event handler to the "focusin" event.
- .focusout(): It binds an event handler to the "focus out" JavaScript event.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }


.para1{
  font-size: 26px;
  padding:30px
}
input{
  padding:10px 30px;
}
</style>
<script>
$(document).ready(function(){
 $(".para1").focusin(function(){
  $('.para1').css("background-color", "green");
 });
 $(".para1").focusout(function(){
  $('.para1').css("background-color", "pink");
 });
});
</script>
```

```
</head>
<body>

<div class="container">
  <h1> Form Events </h1>
  <h2> jQuery focusin() and focusout() method
</h2>
  <div class="para1"">
    <input type="text" placeholder="Name">
    <input type="email" placeholder="Email">

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

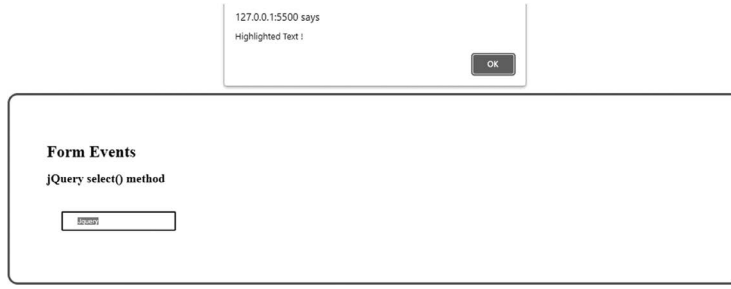
Form event – focusin and focusout method.

- .select(): It binds an event handler to the "select" js event, or trigger that event on an element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
```

```
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }


.para1{
  font-size: 26px;
  padding: 30px
}
input{
  padding: 10px 30px;
}
</style>
<script>
$(document).ready(function(){
  $("input").select(function(){
  alert("Highlighted Text !");
 });
});
</script>

</head>
<body>

<div class="container">
  <h1> Form Events </h1>
  <h2> jQuery select() method </h2>
  <div class="para1"">
    <input type="text" placeholder="Name">

  </div>

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



Form events – select method.

- .submit(): It can bind an event handler to the "submit: js event, or trigger that event on an element.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 28px;
     }


.para1{
  font-size: 26px;
  padding: 30px
}
```

```
input{
  padding: 10px 30px;
}
</style>
<script>
$(document).ready(function(){
 $("form").submit(function(){
  alert("Submitted");
 });
});
</script>

</head>
<body>

<div class="container">
  <h1> Form Events </h1>
  <h2> jQuery submit() method </h2>
  <div class="para1"">
    <form action="">
      First name: <input type="text"
name="FirstName" value="" required><br>
      Last name: <input type="text"
name="LastName" value="" required><br>
      <input type="submit" value="Submit">
     </form>

  </div>

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

Form event – submit method.

DOCUMENT EVENTS

- load
- resize
- scroll
- unload
- ready

Here is a list of jQuery methods that can be called on an Event Object:

- preventDefault(): It can prevent the browser from executing the default action.
- isDefaultPrevented(): It can returns whether event.preventDefault() was called on this event object.
- stopPropagation(): It can stop the bubbling of an event to the parent elements, preventing parent handlers from being notified of the event.
- isPropagationStopped(): It can returns whether event.stopPropagation() was called on this event object.
- stopImmediatePropagation(): It can stop the rest of the handlers from being executed.
- isImmediatePropagationStopped(): It can returns whether event. stopImmediatePropagation() was called on this event object.

EVENT MANIPULATION METHODS

The following list shows important event-related methods. jQuery also provides a collection of event helper functions that can be used either to trigger an event to bind any event types, and are mentioned below:

- bind(): It can bind a handler to one or more events (like click) for each matched element. It can also bind custom events. The bind (type, [data], fn) method binds a handler to one or more events (like click) for each matched element. It can also bind custom events. The event values can be blur, focus, load, resize, scroll, unload, click, etc.

- unbind(): it can remove bound events from each of the matched elements.

  Here is the simple syntax to use the method as given below:

```
selector.bind( type, [data], fn )
```

  Parameters: Here is a complete description of all the parameters used by this method:

  - type – More than one event types separated by a space.
  - data – It is an optional parameter and represents additional data passed to the event handler as event.data.
  - fn – It is a function to bind to the event on each of the set of matched elements.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
  .content{
      margin:10px;
      padding:12px;
      border:2px solid #666;
      width:60px;
      cursor:pointer
}

  </style>
```

```
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {
   $('div').bind('click', function( event ){
     alert('This will bind all the boxes with
each other);
   });
  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery bind method </h1>
    <div class = "content" style = "background-
color:blue;" > One </div>
    <div class = "content" style = "background-
color:green;" > Two </div>
    <div class = "content" style = "background-
color:red;" > Three </div>
  </div>
</body>
</html>
```

bind method.

- off(): It can remove a bound live event. It does the opposite of live. The off (events [, selector] [, handler(eventObject)]) jQuery method does the opposite of on() method, it removes a bound live event.
- on(): It can bind a handler to an event (like click) for all current future – matched element. It can also bind custom events.

Here is the syntax to use this method:

```
selector.on( event, selector, handler )
```

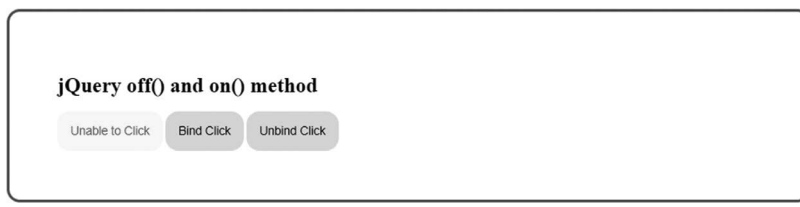
Parameters: Here is the description of the parameters used by this method:

- events – The event types separated by spaces.
- selector – It can select string.
- handler – A function to bind to the event on each of the set of matched elements.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    button{
        font-size: 18px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color: gainsboro
      }
    button#theone { color:red;
background:yellow; }

  </style>
```

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

function aClick() {
 $("div").show().fadeOut("slower");
}

$("#bind").click(function () {
 $("#theone").on("click", aClick).text("You can
click now!");
});

$("#unbind").click(function () {
 $("#theone").off("click", aClick).text(" Unable
to Click ");
});

});
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery off() and on() method </h1>
    <button id = "theone"> Unable to Click
</button>
   <button id = "bind">Bind Click</button>
   <button id = "unbind"> Unbind Click</button>

   <div style = "display:none;"> Click! </div>
  </div>

</body>
</html>
```



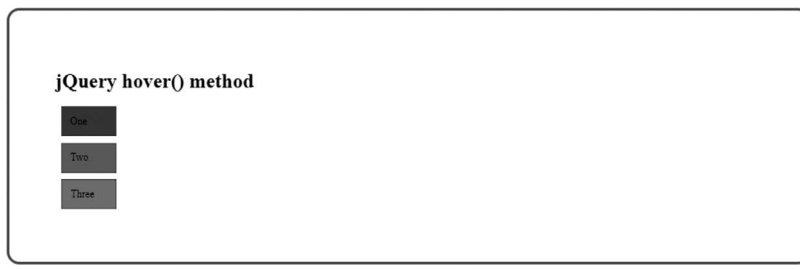
on and off method.

- hover( ): It can simulate hovering, for example, moving the mouse on, and off, an object.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    .content{
      margin:10px;
      padding:12px;
      border:2px solid #666;
      width:60px;
      cursor:pointer
}


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

   $('.content').hover(

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

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

  });
</script>
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery hover() method </h1>
    <div class = "content" style = "background-
color:blue;" > One </div>
    <div class = "content" style = "background-
color:green;" > Two </div>
    <div class = "content" style = "background-
color:red;" > Three </div>
  </div>

</body>
</html>
```



hover method.

- one(): It can bind a handler to one or more events to be executed once for each matched element. The one(type, [data], fn) method binds a handler to one or more events to be executed once for each matched element. The handler is executed only once for each element. Otherwise, the same rules as described in bind() apply.

  Here is the event value: blur, focus, load, resize, scroll, unload, click, etc.

Here is the common syntax to use this method:

```
selector.one( type, [data], fn )
```

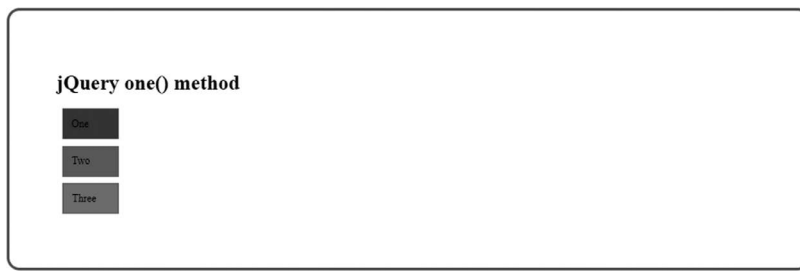
Parameters: Here is a complete description of all the parameters used by this method:

- type – More than one event types separated by a space.
- data – It is an optional parameter and represents additional data passed to the event handler as event.data.
- fn – It is a function used to bind to the event on each of the set of matched elements.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    .content{
      margin:10px;
      padding:12px;
      border:2px solid #666;
      width:60px;
      cursor:pointer
}


  </style>
```

```
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {
   $('.content').one('click', function( event ){
     alert('Clicked');
   });
  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery one() method </h1>
    <div class = "content" style = "background-
color:blue;" > One </div>
    <div class = "content" style = "background-
color:green;" > Two </div>
    <div class = "content" style = "background-
color:red;" > Three </div>
  </div>

</body>
</html>
```



one method.

- ready(): It can bind a function to be executed whenever the DOM is ready to be traversed and manipulated. The ready(fn) method binds a function to be executed whenever the document is ready to be traversed and manipulated. This method is a replacement for using window.onload.

  Here is the syntax to use this method:

```
selector.ready( fn )
```

Parameters: Here is a complete description of all the parameters used by this method:

- fn – The function to be executed when the DOM is ready.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    .content{
      margin:10px;
      padding:12px;
      border:2px solid #666;
      width:60px;
      cursor:pointer
}


  </style>
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {
   $("div").text(" <h1> </h1> The DOM is now
loaded...");
  });
</script>
</head>
<body>
```

```
  <div class="container">
    <h1> jQuery ready() method </h1>
    <div class = "content" > </div>

  </div>

</body>
</html>
```

ready method.

- trigger() : It can trigger an event on every matched element.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    .content{
      margin:10px;
      padding:12px;
      border:2px solid #666;
      width:60px;
```

```
      cursor:pointer
}


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

   $("#div1").click( function () {
     $("#div2").trigger('click');
   });

   $("#div2").click( function () {
     alert( " One is clicked");
   });

  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery trigger() method </h1>
    <div id = "div1" class="content" style =
"background-color:yellow;">ONE</div>
    <div id = "div2" class="content" style =
"background-color:lightcoral;">TWO</div>

  </div>

</body>
</html>
```

trigger method.

- triggerHandler( ): It can trigger all bound event handlers on an element. The triggerHandler(event, [data]) method triggers all bound event handlers on an element (for a specific event type) WITHOUT executing the browser's default actions, bubbling, or live events. This behaves similarly to the trigger method, with two major exceptions:
  - First – There is no default browser actions triggered, the triggered event does not bubble, and live events aren't triggered.
  - Second – The event is only triggered on the first element within the jQuery collection.

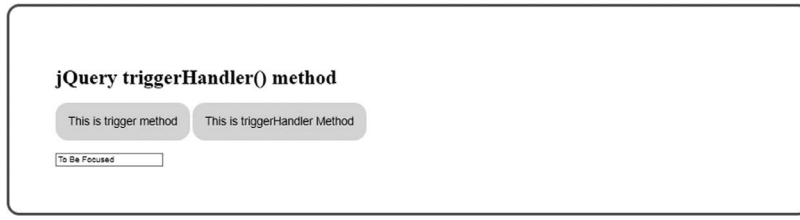
  Here is the syntax to use this method:

```
selector.triggerHandler( event, [data] )
```

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    .content{
      margin:10px;
      padding:12px;
      border:2px solid #666;
      width:60px;
      cursor:pointer
}
```

```
button{
        font-size: 18px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color: gainsboro
      }


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

   $("#old").click(function(){
     $("input").trigger("focus");
   });

   $("#new").click(function(){
     $("input").triggerHandler("focus");
   });

   $("input").focus(function(){
     $("<span> Focused! </span>").
appendTo("body").fadeOut(1000);
   });

  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery triggerHandler() method </h1>
    <button id = "old">This is trigger method
</button>
    <button id = "new"> This is triggerHandler
Method</button><br/><br/>

    <input type = "text" value = "To Be
Focused"/>
  </div>

</body>
</html>
```



triggerHandle method.

TRIGGER METHODS

Following is an example that would trigger the blur event on all paragraphs:

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

BINDING METHODS

Following is an example that binds a click event on every :

```
$("div").click( function () {
 // Here is your code
})
```

Here is a list of all the jQuery Support Methods:

- blur( ): It can trigger or binds the blur event of each matched element.
- change(): It can trigger or binds the change event of each matched element.
- click( ): It can trigger or binds the click event of each matched element.
- dblclick(): It can trigger or binds the dblclick event of each matched element.
- error(): It can trigger or binds the error event of each matched element.
- focus( ): It can trigger or binds the focus event of each matched element.

- keydown( ): It can trigger or binds the keydown event of each matched element.
- keypress(): It can trigger or binds the keypress event of each matched element.
- keyup(): It can trigger or binds the keyup event of each matched element.
- load(): It can bind a function to the load event of each matched element.
- mousedown(): It can bind a function to the mousedown event of each matched element.
- mouseenter(): It can bind a function to the mouseenter event of each matched element.
- mouseleave(): It can bind a function to the mouseleave event of each matched element.
- mousemove(): It can bind a function to the mousemove event of each matched element.
- mouseout(): It can bind a function to the mouseout event of each matched element.
- mouseover( ): It can bind a function to the mouseover event of each matched element.
- mouseup( ): It can bind a function to the mouseup event of each matched element.
- resize(): It can bind a function to the resize event of each matched element.
- scroll(): It can bind a function to the scroll event of each matched element.
- select(): It can trigger or binds the select event of each matched element.
- submit( ): It can trigger or binds the submit event of each matched element.
- unload(): It can bind a function to the unload event of each matched element.

EVENT ATTRIBUTES

Whenever a jQuery event is canceled, jQuery transfers the Event Item to the entire event manager function. The event item provides a variety of useful information about the event.

The event item is usually not required, and the parameter is omitted, as sufficient context is usually found where the handle should know exactly what to do when the handle is configured, however there are certain attributes you may need to access.

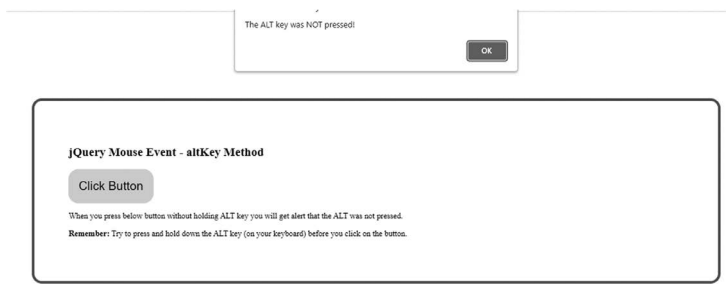
The following event attributes are available to access in a platform-independent manner:

- altKey: It sets to true if the Alt key was pressed when the event was triggered, if not false. The Alt key is labeled on most of the Mac keyboards.

  Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
```

```
#div1{
  font-size:26px
}
  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js"> </script>
<script>
  function isKeyPressed(event) {
   if (event.altKey) {
    alert("The ALT key was pressed!");
   } else {
    alert("The ALT key was NOT pressed!");
   }
  }
  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Mouse Event - altKey Method
</h2>
  <button onmousedown="isKeyPressed(event)">
Click Button </button>
  <p>When you press the below button without
holding the ALT key, you will get an alert that
the ALT was not pressed. </p>
  <strong> Remember: </strong>
   Try to press and hold down the ALT key (on
your keyboard) before you click on the button.
</p>
    </div>
</body>
</html>
```



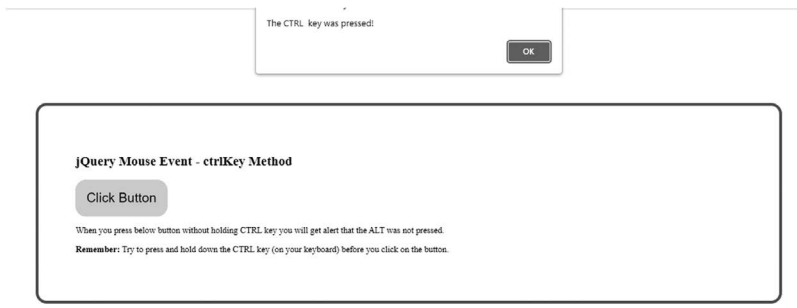
Mouse event – altKey method.

- ctrlKey: It sets to true if the Ctrl key was pressed when the event was triggered and false if not.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
#div1{
  font-size:26px
}
  </style>
<script src= "http://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"> </script>
<script>
  function isKeyPressed(event) {
   if (event.ctrlKey) {
    alert("The CTRL key was pressed!");
   } else {
```

```
    alert("The CTRL key was NOT pressed!");
   }
  }
  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Mouse Event - ctrlKey Method </h2>
  <button onmousedown="isKeyPressed(event)">
Click Button </button>
  <p>When you press the below button without
holding the CTRL key, you will get an alert that
the ALT was not pressed. </p>
  <strong> Remember: </strong>
   Try to press and hold down the CTRL key (on
your keyboard) before you click on the button.</p>
    </div>
</body>
</html>
```



Mouse event – ctrlKey method.

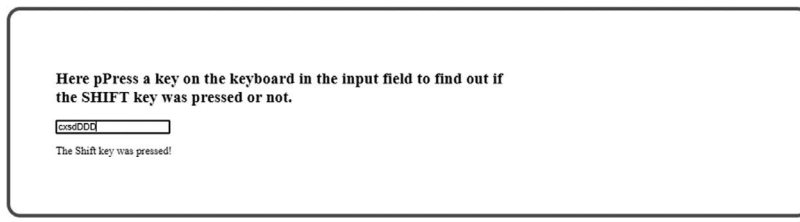
- shiftKey: It sets to true if the Shift key was pressed when the event was triggered and false if not.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
```

```
<script src="http://www.tutorialspoint.com/
jquery/jquery-3.6.0.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
#div1{
  font-size:26px
}
  </style>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"> </script>
<script>
   function isKeyPressed(event) {
   if (event.shiftKey) {
    $(".result").text("The Shift key was
pressed!");
   } else {
    $(".result").text("The Shift key was NOT
pressed!");
   }
  }

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

```
  <div class="container">
    <h2>
    Here pPress a key on the keyboard in the
input field to find out if <br>
        the SHIFT key was pressed or not. </h2>
    <input type="text" onkeydown="isKeyPressed
(event)">
  <p class="result"> </p>
</body>
</html>
```



Mouse event – shiftKey method.

- data: It is a value, if any, passed as the second parameter to the bind() command when the handler was established.
- keyCode: It is used for keyup and keydown events, this returns the key that was pressed.
- metaKey: It sets to true if the Meta key was pressed when the event was triggered, then vice versa. The Ctrl key is the Meta Key and the Command key on Macs.
- pageX: It is used for mouse events, and specifies the horizontal coordinate of the event relative to the page origin.
- pageY: It is used for mouse events, and specifies the vertical coordinate of the event relative to the page origin.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
```

```
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
#div1{
  font-size:26px
}
  </style>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"> </script>
<script>
  $(document).ready(function(){
   $(document).mousemove(function(event){
    $("p").text("X: " + event.pageX + ", Y:
" + event.pageY);
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery Mouse Event - pageX and pageY
</h1>
```

```
    <h3> Move your mouse cursor on the screen to
get its X and Y position</h3>
  <p></p>
</body>
</html>
```

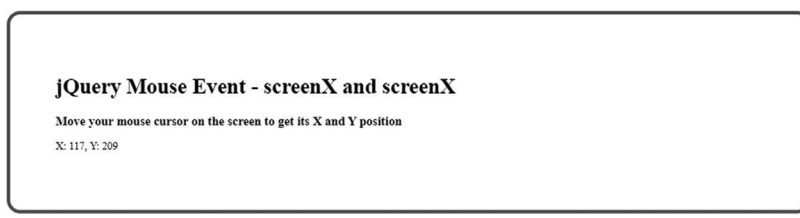
jQuery mouse event – pageX and pageY.

- relatedTarget: It is used for mouse events, it identifies the element that the cursor left or entered when the event was triggered.
- screenX: It is used for mouse events, it specifies the horizontal coordinate of the event relative to the screen origin.
- screenY: It is used for mouse events, it specifies the vertical coordinate of the event relative to the screen origin.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
```

```
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
#div1{
  font-size:26px
}
  </style>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"> </script>
<script>
  $(document).ready(function(){
   $(document).mousemove(function(event){
    $("p").text("X: " + event.screenX + ", Y: "
+ event.screenY);
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h1> jQuery Mouse Event - screenX and
screenX </h1>
    <h3> Move your mouse cursor on the screen to
get its X and Y position</h3>
  <p></p>
</body>
</html>
```



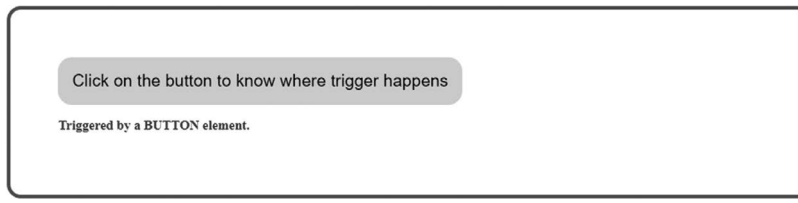
jQuery Mouse Event – screenX and screenY.

- target: It can identify the element for which the event was triggered.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
#div1{
  font-size:26px
}
  </style>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"> </script>
<script>
  $(document).ready(function(){
   $("p, button, h1").click(function(event){
    $(".result").html( "Triggered by a " +
event.target.nodeName + " element." );
   });
```

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

  <div class="container">
    <button> Click on the button to know where
trigger happens </button>
    <h3 class="result" style="color:blue;">
Result: </h3>
</body>
</html>
```



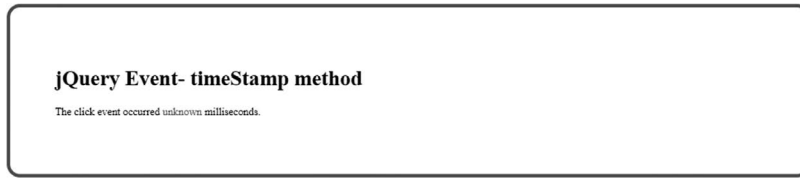
Event attributes – target.

- Timestamp: It is the timestamp (in milliseconds) when the event was created.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
```

```
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
#div1{
  font-size:26px
}
  </style>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.3.1/jquery.min.js"> </script>
<script>
  <!-- It's code to show the working of event.
timeStamp property -->
  $(document).ready(function() {
    $("div").click(function(event) {
      $(".result").text(event.timeStamp);
    });
  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery Event- timeStamp method </h1>
    <p>The click event occurred <span
class="result" style="color:green">unknown
</span>
      milliseconds.
      </p>
      </div>
      </body>
</html>
```



jQuery – timestamp.

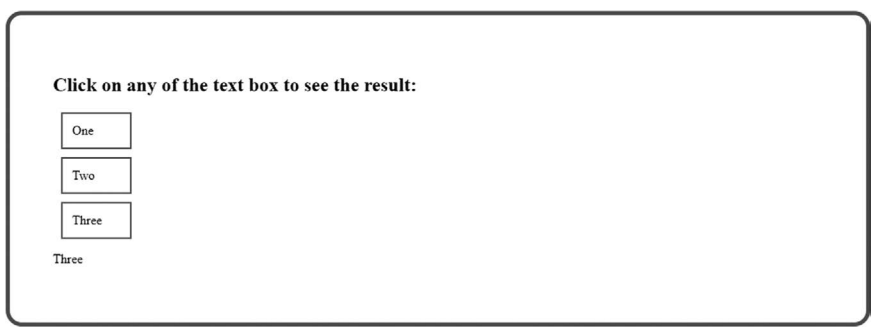
- type: It is used for all events, it specifies the type of event that was triggered (for example, click).
- which: It can specify the numeric code for the key that caused the event, and for mouse events, it specifies which button was pressed (1 for left, 2 for middle, 3 for right).

this Keyword in Event Handler

Many times it becomes easy to make use of this keyword inside an event handler. The keyword represents a DOM element which triggers the event. The following example will show the content of the clicked .

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/libs/
jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding: 5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
  .content{
      margin: 10px;
```

```
      padding: 12px;
      border: 2px solid #666;
      width: 60px;
      cursor: pointer}

  </style>
<script>
 $(document).ready(function() {
   $(".content").click(function(){
    $('.result').text($(this).text());
   });
 });
</script>

</head>
<body>

  <div class="container">
    <h2> Click on any of the text box to see the
result: </h2>
    <div class="content"> One </div>
    <div class="content"> Two </div>
    <div class="content"> Three </div>
    <p class="result"> <span> Result: </span>  </p>
  </div>

</body>
</html>
```



this keyword.

Removing Event Handlers

When you establish an event handler, it remains for the remainder of the life of the page. There might be a need when you would like to remove the event handler.

jQuery provides the unbind() function to remove an exiting event handler. The syntax of unbind() is as given below:

```
selector.unbind(eventType, handler)

or

selector.unbind(eventType)
```

Here is the description of the parameters:

- eventType – A string containing an event type, such as clicking or submitting.
- handler – It identifies the specific listener that's to be removed.

jQuery HTML/CSS METHODS

jQuery provides various methods to control the DOM in a better way. It provides methods which act as setter, getter, retrieving information from various DOM elements like .attr(), .html(), and .val().

All the selectors are supported in jQuery. CSS Methods are also used to apply CSS properties on DOM elements.

- addClass() Method: The addClass is a built-in method which is used to add more properties to each selected element. It can be used to change the property of the selected element. This method can be used in two different ways:

  Here, the class name can be used with the element which is going to be selected.

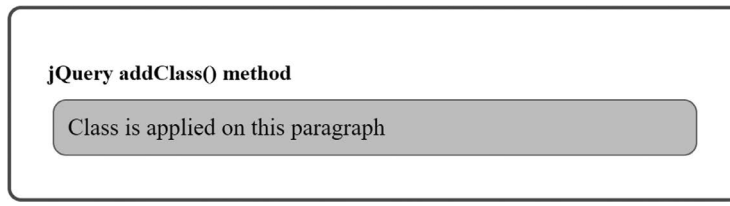
  Syntax:

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

  Example:

  ```
  <!doctype html>

  <html>
  ```

```
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    p {
      margin: 8px;
      font-size: 35px;
      border-radius: 20px;
      border: 2px solid green;
      background-color: light pink;
      padding: 20px;
    }


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

    $("p").last().addClass("selected");
  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery addClass() method </h1>
    <p> Class is applied on this paragraph </p>
  </div>

</body>
</html>
```



addClass method.

- after() Method: The after() is a built-in function which is used to insert content, specified by the parameter for the each selected element in the set of matched elements.

Syntax:

```
$(selector).after(A);
```

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    p {
      margin: 8px;
      font-size: 35px;
      border-radius: 20px;
      border: 2px solid green;
      background-color: light pink;
      padding: 20px;
    }


  </style>
```

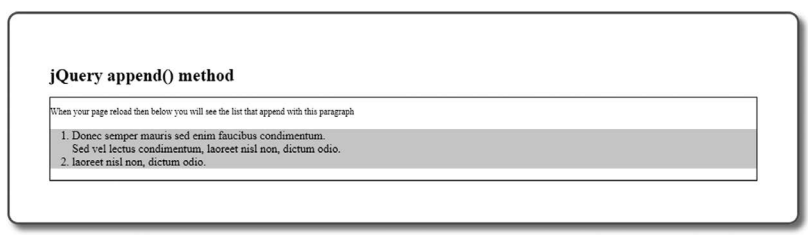
```
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {

    $("p").after(" This text will be after the
any paragraph ");
  });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery after() method </h1>
    <p> This is a paragraph </p>
    <p> This is another paragraph </p>
  </div>

</body>
</html>
```

- append() Method: It can insert content to the end of the element(s) that is specified by a selector.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="http://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>
    <script>
     $(document).ready(function () {

$('p').append('<div style="background-color:
pink ; font-size:21px"> <ol> <li> Donec semper
mauris sed enim faucibus condimentum. </li> Sed
vel lectus condimentum, laoreet nisl non, dictum
odio. <li> laoreet nisl non, dictum odio. </li>
</ol> </div>');

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

```
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }

      .div1{
      border: 1px solid;
      margin: 2px 0 2px 0;
    }
    </style>
  </head>
 <body>

   <div class="container">
    <h1>jQuery append() method </h1>
      <div class="div1">
        <p> When your page reload then below you
will see the list that append
          with this paragraph </p>
      </div>
   </div>
    </body>
    </html>
```



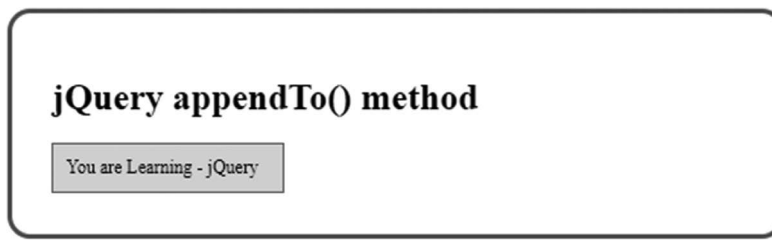
Append method.

- .appendTo(): It can insert every element in the set of matched elements to the end of the target.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <script src="http://ajax.googleapis.com/
ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <script>
   $(document).ready(function () {
     $('#btn1').click(function () {
      $(".div2").appendTo(".app");
     });

       });
  </script>
</head>
  <style>
    body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
     p{
       font-size: 20px;
     }
  .app{
    background: light green;

  }
  button{
      font-size: 18px;
      border: none;
      border-radius: 20px;
      padding: 20px;
      background-color: gainsboro
    }
```

```
    #intro {
    background: light green;
    border: 2px solid green;
    padding: 10px;
    width: 180px;
    }
   </style>
   <script src="https://code.jquery.com/
jquery-1.10.2.js">
   </script>
 </head>
 <body>
    <div class="container">
   <span> jQuery </span>
   <h1> jQuery appendTo() method </h1>
  <div id="intro"> You are Learning - </div>
  </div>

   <script>
    $("span").appendTo("#intro");
   </script>
 </body>

 </body>
</html>
```



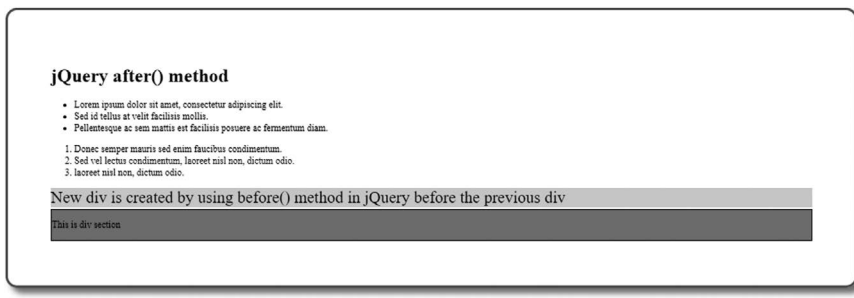
appendTo method.

- attr() Method: It can set or return attributes and values of the selected elements.
- before() Method: It can insert content (new or existing DOM elements) before an element(s) which is specified by a selector.

Example:

```
  <!DOCTYPE html>
  <html>
  <head>
    <script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
    <script>
     $(document).ready(function () {

$('.div1').before('<div style="background-color:
pink ; font-size:28px"> New div is created by
using before() method in jQuery before the
previous div</div>');

});
    </script>
  </head>
    <style>
      body{
        text-align: left;
        margin: 5%;
        padding:5%;
      }
      .container{
        border-radius: 20px;
        padding:5%;
        border: 5px solid rgb(136, 77, 77);
        box-shadow: 10px 10px 10px gray;
      }
      button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
      .div1{
      border: 1px solid;
      background-color:red;
      margin: 2px 0 2px 0;
    }
    </style>
```

```
  </head>
 <body>

   <div class="container">
    <h1>jQuery after() method </h1>
    <ul>
      <li>Lorem ipsum dolor sit, consectetur
adipiscing elit. </li>
      <li> Sed id tellus at velit facilisis
mollis. </li>
      <li>
        Pellentesque ac sem mattis est facilisis
posuere ac fermentum diam. </li>
    </ul>

      <ol>
        <li> Donec semper mauris sed enim faucibus
condimentum. </li>
        <li>
          Sed vel lectus condimentum, laoreet nisl
non, dictum odio.</li>
        <li> laoreet nisl non, dictum odio. </li>
      </ol>

      <div class="div1">
        <p> This is div section </p>
   </div>
      </div>
    </body>
    </html>
```



before method.

- clone() Method
- CSS() Method
- detach() Method
- empty() Method
- hasClass() Method
- HTML() Method
- insertAfter()Method
- insertBefore() Method
- offsetParent()Method
- prepend()Method
- prependTo()Method
- prop()Method
- remove()Method
- removeAttr()Method
- removeClass()Method
- removeProp()Method
- replaceAll()Method
- replaceWith()Method
- scrollLeft()Method
- scrollTop() Method
- text() Method
- toggleClass() Method
- unwrap() Method
- Val()Method
- wrap() Method
- wrapAll()Method
- wrapInner() Method

MANIPULATE DOM ELEMENT'S DIMENSIONS USING jQuery

The jQuery library includes various different methods to manipulate DOM element's dimensions like offset, position, height, width, etc.

The following lists all the jQuery methods to get or set the DOM element's dimensions:

- height(): It gets or sets height of the specified element(s).
- innerHeight(): It gets or sets inner height (padding + element's height) of the specified element(s).
- outerHeight(): It gets or sets outer height (border + padding + element's height) of the specified element(s).
- offset(): It gets or sets left and top coordinates of the specified element(s).
- position(): It gets the current coordinates of the specified element(s).
- width(): It gets or sets the width of the specified element(s).
- innerWidth(): It gets or sets the inner width (padding + element's width) of the specified element(s).
- outerWidth(): It gets or sets outer width (border + padding + element's width) of the specified element(s).

TRAVERSING DOM ELEMENTS USING jQuery

The jQuery library includes various methods to traverse DOM elements in a DOM hierarchy.

jQuery is a very powerful tool that provides a variety of DOM shortcuts to help us to select elements in HTML or XML documents randomly and sequentially. Elements in DOM are organized into a tree-like data structure that can be cut down for navigation, and retrieving content within an HTML or XML document.

THE DOM

Most DOM Traversal Methods do not convert the jQuery DOM object and are used to filter elements in a document based on specific circumstances. jQuery offers cross-cutting methods in the following three ways:

1. Traversing upwards – This method means crossing ancestors (Parent, Grandfather, Grandfather, Grandfather, etc.).
2. Traversing downwards – This method means termination of interest (Child, Granddaughter, Granddaughter, etc.).
3. Sideways – This method means tearing apart the ancestors (Brothers and sisters who are found at the same level).

Here is the complete list of jQuery method for traversing DOM elements:

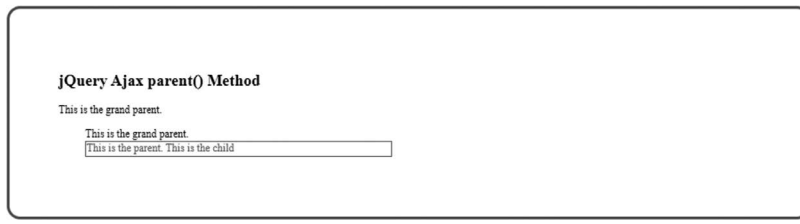
- children(): It gets all the child elements of the specified element(s).
- each(): It gets iterated over specified elements and executes a specified call back function for each element.
- find(): It gets all the specified child elements of each specified element(s).
- first(): It gets the first occurrence of the specified element.
- next(): It gets the immediately following sibling of the specified element.
- parent(): It gets the parent of the specified element(s).

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
```

```
        border: 5px solid rgb(136, 77, 77);
      }


      button{
          font-size: 24px;
          border: none;
          border-radius: 20px;
          padding: 20px;
          background-color:RGB(80, 243, 47)
        }

    </style>
  <script src=
  "http://ajax.googleapis.com/ajax/libs/
  jquery/3.3.1/jquery.min.js">
    </script>
     <script src="./javascript.js" ></script>
    <script>
    $(document).ready(function() {
        $("span").parent().css({
          "color": "green",
          "border": "2px solid green",
          "list-style" : "none"
        });
      });

    </script>
  </head>
  <body>

    <div class="container">
      <h2> jQuery Ajax parent() Method </h2>
      <div style="width:500px;">
        This is the grand parent.
          <ul>This is the grand parent.
            <li>This is the parent.
              <span> This is the child </span>
            </li>
          </ul>
        </div>
```

```
  <script>

  </script>
  </div>
</body>
</html>
```



parent method.

- parents(): It gets all the parents of the elements.
- prev(): It gets the immediately preceding sibling of the specified element.
- siblings(): It gets the siblings of each specified element(s).

CSS MANIPULATION USING jQuery

The jQuery library has various methods to manipulate style properties and CSS class of DOM element(s).

- CSS(): It gets or sets style properties to the specified element(s).
- addClass(): It is used to add one or more class to the specified element(s).
- hasClass(): It determines whether any of the specified elements are assigned the given CSS class.
- removeClass(): It removes a single class, multiple classes, or all classes from the specified element(s).
- toggleClass(): It toggles between adding/removing classes to the specified elements

jQuery EFFECTS

jQuery enables us to add results to a web page. The jQuery results can be categorized by blurring, slide, hide/show, and animation effects. jQuery provides a simple visual interface for performing various types of amazing effects like show, hide, blur inside, blur, slide, slide, rotate, etc. jQuery methods allow us to quickly apply the results commonly used with minimal configuration. This tutorial covers all the essential techniques of jQuery to create visual effects.

jQuery provides a simple encryption syntax with the help of encryption ():

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

You can use any jQuery selector to select any DOM item and use the jQuery hide () method to hide it. Here is a description of all the parameters that give you strict control over the effect of hiding:

1. speed – The parameter you select represents one of three predefined speeds ("slow," "normal," or "fast") or the number of milliseconds to start animation (e.g., 1000).
2. callback – This optional parameter represents the function to be performed whenever animation is terminated; apply it once on each part of the animation against it.

jQuery EFFECT METHODS

- .animate(): It can perform a custom animation of a set of CSS properties.
- .clearQueue(): It can remove from the queue all items that have not yet been run.
- .delay(): It can set a timer to delay the execution of subsequent items in the queue.
- .dequeue(): It can execute the next function on the queue for the matched elements.
- .fadeIn(): It can display the matched elements by fading them to opaque.
- .fadeOut(): It can hide the matched elements by fading them to transparent.

- .fadeTo(): It can adjust the opacity of the matched elements.
- .fadeToggle(): It can display or hide the matched elements by animating their opacity.
- .finish(): It can stop the currently-running animation, remove all queued animations, and complete all animations for the matched elements.
- .hide(): It can hide the matched elements.
- jQuery.fx.interval: It can rate (in milliseconds) at which animations fire.
- jQuery.fx.off: It can disable all animations.
- jQuery.speed: It can create an object containing a set of properties ready to be used in the definition of custom animations.
- .queue(): It can show or manipulate the queue of functions to be executed on the matched elements.
- .show(): It can display the matched elements.
- .slideDown(): It can display the matched elements with a sliding motion.
- .slideToggle(): It can display or hide the matched elements with a sliding motion.
- .slideUp(): It can hide the matched elements with a sliding motion.
- .stop(): It can stop the currently-running animation on the matched elements.
- .toggle(): It can display or hide the matched elements.

jQuery – ATTRIBUTES MANIPULATION

jQuery is widely used to manage various attributes related to HTML objects. Every HTML object can have standard and custom attributes (i.e., layouts) that are used to define features of that HTML object. jQuery gives us easy ways to easily manipulate (Find and Set) elements. First, let's try to understand a little bit about standard HTML and custom attributes.

jQuery gives us the means to manipulate (Get and Set) an element's attributes. Let's try to understand HTML standards and custom attributes.

STANDARD ATTRIBUTES

Some of the more common attributes are given below:

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

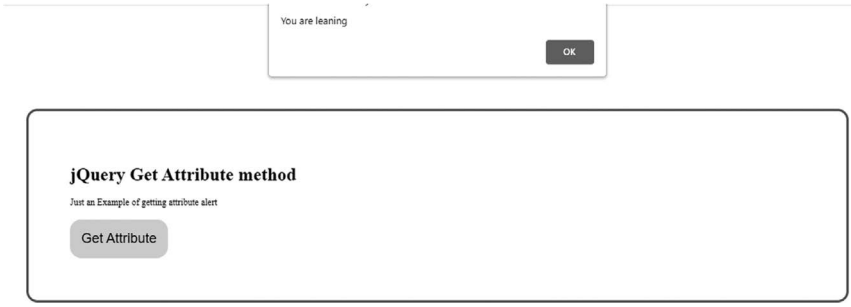
jQuery – GET DATA ATTRIBUTES

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

Example: Here 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="http://ajax.googleapis.com/ajax/libs/
jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
```

```
    p {
      margin: 8px;
      font-size: 35px;
      border-radius: 20px;
      border: 2px solid green;
      background-color: light pink;
      padding: 20px;
    }

    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {
   $("button").click(function(){
    alert( $("#text").data("heading"));
    alert( $("#text").data("language"));
   });
 });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery Get Attribute method </h1>

 <div id="text" data-heading="You are leaning"
data-language="jQuery">
  Just an Example of getting attribute alert
</div>
<br>
<button>Get Attribute</button>
  </div>

</body>
</html>
```



Get attribute.

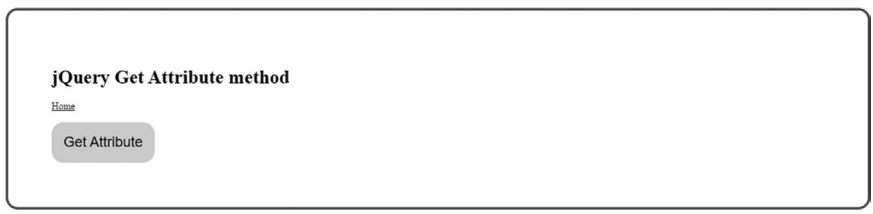
jQuery – GET STANDARD ATTRIBUTES

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

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/libs/
jquery/3.6.0/jquery.min.js"></script>
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }
    p {
      margin: 8px;
      font-size: 35px;
      border-radius: 20px;
```

```
        border: 2px solid green;
        background-color: light pink;
        padding: 20px;
      }

      button{
          font-size: 24px;
          border: none;
          border-radius: 20px;
          padding: 20px;
          background-color:RGB(80, 243, 47)
        }

  </style>
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {
   $("button").click(function(){
    alert( "Href = " + $("#home").attr("href"));
    alert( "Title = " + $("#home").attr("title"));
   });
 });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery Get Attribute method </h1>
    <p><a id="home" href="index.html" title="Index
Page"> Home </a></p>
    <button>Get Attribute</button>
  </div>
</body>
</html>
```



Get standard attributes.

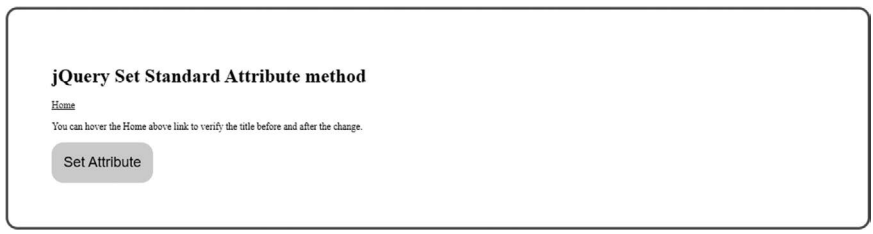
jQuery – SET STANDARD ATTRIBUTES

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

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/libs/
jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {
   $("button").click(function(){
    $("#home").attr("title", "Index Page");
```

```
    /* Let's get and display changed title */
    alert( "Changed Title = " + $("#home").
attr("title"));
   });
 });
</script>
</head>
<body>

  <div class="container">
    <h1> jQuery Set Standard Attribute method
</h1>
    <p><a id="home" href="index.htm" title="Index
Page">Home</a></p>
    <p>You can hover the Home above link to verify
the title before and after the change.</p>
 <button>Set Attribute</button>
  </div>
</body>
</html>
```



Set standard attribute method.

jQuery – SET CUSTOM ATTRIBUTES

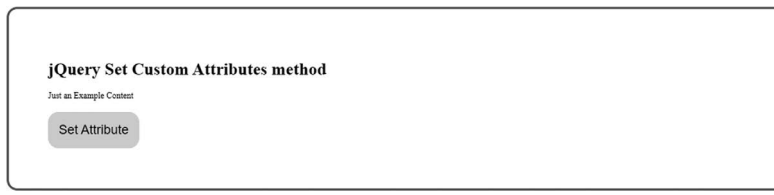
jQuery data(name, value) method is used to set the value of any attribute of the HTML element(s). We will use selectors to match the desired element(s) and then we will apply the attr(key, value) method to set the attribute value for the element.

Example:

```
<!doctype html>
<html>
```

```
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/libs/
jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script type = "text/javascript" language =
"javascript">
 $(document).ready(function() {
   $("button").click(function(){
    $("#home").data("language", " You are Learning
- jQuery ");
    /* Let's get and display changed author name
*/
    alert( "Changed Name = " + $("#home").
data("language"));
   });
 });
</script>
</head>
<body>
```

```
    <div class="container">
      <h1> jQuery Set Custom Attributes method </h1>
      <div id="home" data-language=" JavaScript " >
        Just an Example Content
      </div>
      <br>
      <button>Set Attribute</button>
    </div>
</body>
</html>
```



Set custom attributes.

jQuery – AJAX

AJAX is an acronym for Asynchronous JavaScript and XML, and this technology helps us to download data from the server without updating the browser page. If you are new to AJAX, then we would recommend that you check out our AJAX Tutorial before going tutorial. jQuery is a great tool that provides a rich set of AJAX ways to improve the next generation web application. The jQuery library includes various methods to send AJAX requests. These methods use XMLHttpRequest object of JavaScript.

- jQuery ajax(): It sends asynchronous HTTP request to the server.

The jQuery ajax() method provides the basic functionality of AJAX in jQuery. It sends asynchronous HTTP requests to the server.

Parameter description:

1. URL: It is the string URL to which you want to submit or retrieve the data.
2. options: It is configuration options for AJAX requests. It is an options parameter that can be specified using JSON format. This parameter is optional.

There are various parameters we can add in our AJAX method. So given below is the list of the parameters:

- accepts: It is content type sent in the request header that tells the server what kind of response it will accept in return.
- async: All requests are sent asynchronously. Set it to false to make it synchronous, by default.
- beforeSend: A callback function to be executed before the AJAX request is sent.
- cache: A boolean indicating browser cache. Default is true.
- complete: It is a callback function to be executed when the request finishes.
- contentType: It is a string containing a type of content when sending MIME content to the server. It is default "application/x-www-form-urlencoded"; charset=UTF-8.
- crossDomain: It is a boolean value indicating whether a request is a cross-domain.
- Data: It is data to be sent to the server. It can be JSON object, string, or array.
- dataType: It is a type of data that you're expecting back from the server.
- Error: It is a callback function to be executed when the request fails.
- Global: It is a Boolean indicating whether to trigger a global AJAX request handler or not. Its default is true.
- Headers: It is an object of header key/value pairs to send along with the request.
- ifModified: It allows the request to be successful only if the response has changed since the last request. This is done by checking the Last-Modified header. Its default value is false.
- isLocal: It allows the current environment to be recognized as local.
- Jsonp: It overrides the callback function name in a JSONP request.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>

  <script>
$(document).ready(function(){
 $("button").click(function(){
  $.ajax({url: "ajax.txt", success:
function(result){
   $("#div1").html(result);
  }});
```

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

  <div class="container">
    <h2> jQuery Ajax load() Method </h2>
    <div id="div1"><h2>Let jQuery AJAX Change
the text</h2></div>
    <button>Get External Content</button>

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

jQuery AJAX ajax() method.

- jQuery get(): It sends HTTP GET request to load the data from the server.

  The jQuery get() method sends asynchronous HTTP GET request to the server and retrieves the data.

  Syntax:

  ```
  $.get(URL, [data],[callback]);
  ```

  Here is the parameter:

  - URL: It is the request URL from which you want to retrieve the data.
  - data: It is data to be sent to the server with the request as a query string.
  - callback: It is a function to be executed when the request succeeds.

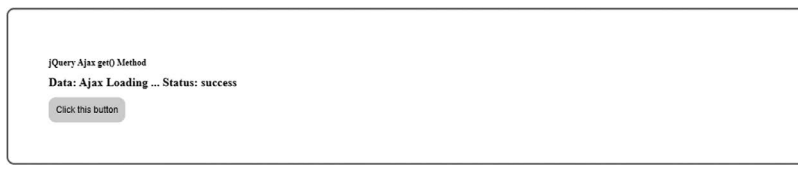
This is a way to write get code in ajax:

```
$.get('/data.txt', // URL
   function (data, textStatus, jqXHR) { // success
callback
     alert('status: ' + textStatus + ', data:'
+ data);
});
```

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
```

```
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>

  <script>
$(document).ready(function(){
 $("button").click(function(){
  $.get("ajax.txt", function(data, status){
   $('.result').text(("Data: " + data +
"\nStatus: " + status));
  });
 });
});

  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Ajax get() Method </h2>
    <h1 class="result"></h1>
    <button> Click this button </button>

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



jQuery AJAX get() method.

- jQuery Post(): It sends an HTTP POST request to submit or load the data to the server.

  The jQuery.getJSON(url, [data], [callback]) method loads JSON data from the server using a GET HTTP request. The method returns XMLHttpRequest object.

Here is the syntax to use this method:

```
$. post ( URL, [data], [callback] )
```

Here is the full description of all the parameters used by this method:

- URL – It is a string containing the URL to which the request is sent.
- data – It is an optional parameter that represents key/value pairs that will be sent to the server.
- callback – It is an optional parameter that represents a function to be executed whenever the data is loaded successfully.

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/libs/
jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }
```

```
  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>

  <script>
$(document).ready(function(){
 $("button").click(function(){
  $.post(url,
  {
   name: " Add your data ",
  },
  function(data,status){
   alert("Data: " + data + "\nStatus: " + status);
  });
 });
});

  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Ajax post() Method </h2>
    <h1 class="result"></h1>
    <button> Send an POST request to a page & get
the result back </button>

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

- jQuery getJSON(): It sends HTTP GET request to load JSON-encoded data from the server. Make your JSON file and add some data then save it with .extension.

  The jQuery.getJSON(url, [data], [callback]) method loads JSON data from the server using a GET HTTP request. The method returns XMLHttpRequest object.

  Here is the syntax to use this method:

```
$.getJSON( URL, [data], [callback] )
```

Here is the full description of all the parameters used by this method:

- URL – It is a string containing the URL to which the request is sent.
- data – It is an optional parameter that represents key/value pairs that will be sent to the server.
- callback – It is an optional parameter that represents a function to be executed whenever the data is loaded successfully.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
```

```
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>

  <script>
$(document).ready(function(){
 $("button").click(function(){
  $.getJSON("ajax.json", function(result){
    $(".result").append(result.Name + " " +
result.Age)
   });
  });
 });

  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Ajax getJSON() Method </h2>
    <h1 class="result"></h1>
    <button> Get your Json Data </button>

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

getJSON method.

- jQuery getScript(): It sends HTTP GET request to load the JavaScript file from the server and then executes it.

  The jQuery.getScript(url, [callback]) method loads and executes a JavaScript file using an HTTP GET request. It returns XMLHttpRequest object.

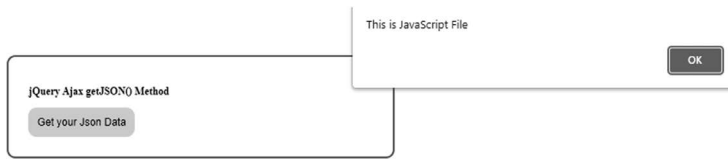
  Here is the syntax to use this method:

```
$.getScript( URL, [callback] )
```

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>
   <script src="./javascript.js" ></script>
  <script>
$(document).ready(function(){
 $("button").click(function(){
  $.getScript("javasscript.js");
  CheckJS();
 });
});
```

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

  <div class="container">
    <h2> jQuery Ajax getJSON() Method </h2>
    <h1 class="result"></h1>
    <button> Get your Json Data </button>
  <script>

  </script>
  </div>
</body>
</html>
```



getScript method.

- jQuery load(): It sends HTTP request to load the HTML or content from the server and add them to DOM element(s).

  The jQuery load () method is a simple, yet powerful AJAX method. The load () method loads data from the server and places the retrieved data in the selected element.

  Syntax:

  ```
  $ (selector) .load (URL, data, callback);
  ```

  It required URL parameter, it specifies the URL you wish to upload. The optional data parameter specifies a querystring key set/pair number for shipping and request. Optional call parameter is the name of the job to be performed after the loading () method is completed.

  The optional dialing parameter specifies the callback function that will be activated once the download method () has been completed. The callback function can have different parameters:

  - responseTxt – It contains content that results when a call is successful.

- statusTxt – It contains a call status.
- xhr – It contains the XMLHttpRequest item.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding: 5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>

  <script>
    $(document).ready(function(){
 $("button").click(function(){
  $("#div1").load("ajax.txt");
```

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

  <div class="container">
    <h2> jQuery Ajax load() Method </h2>
    <div id="div1"> <h2> Let jQuery AJAX Change
the text </h2> </div>
    <button> Get External Content </button>

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

jQuery AJAX load method.

The jQuery library also includes the following events which will be fired based on the state of the AJAX request.

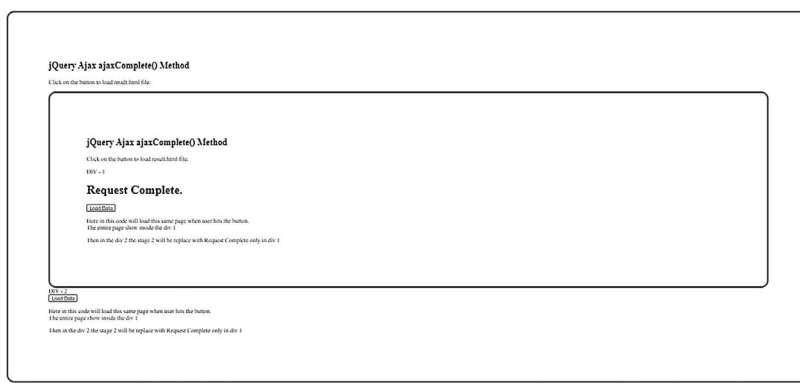
- ajaxComplete(): It is used to register a handler function to be called when AJAX requests complete. It specifies a function to be run when an AJAX request completes.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
```

```
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
  </script>
<script type = "text/javascript" language =
"javascript">
  $(document).ready(function() {

   $("#driver").click(function(event){
     $('#div1').load('jquery.html');
   });

   $(document).ajaxComplete(function(event,
request, settings){
     $("#div2").html("<h1> Request Complete.
</h1>");
   });

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

```
  <div class="container">
    <h2> jQuery Ajax ajaxComplete() Method </h2>
    <p> Click on the button to load result.html
file: </p>

    <div id = "div1" >
     DIV - 1
    </div>

    <div id = "div2" >
     DIV - 2
    </div>

    <input id = "driver" type = "button" value =
"Load Data" />
  <script>

  </script>
  <p> Here in this code will load this same page
when the user hits the button. <br> The entire
page show inside the div 1</p>
  <p> Then in the div 2 the stage 2 will be
replace with Request Complete only in div 1 </p>
</div>
</body>
</html>
```



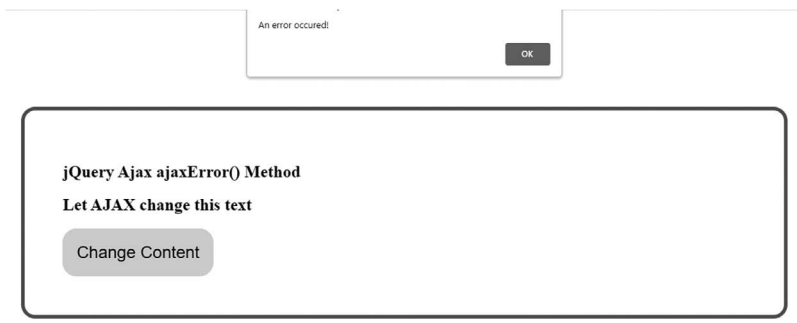
jQuery – ajaxComplete.

- ajaxError(): It is used to register a handler function to be called when AJAX requests complete with an error.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"htt://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
</script>
  <script>
$(document).ready(function(){
 $(document).ajaxError(function(){
  alert("An error occured!");
```

```
 });
 $("button").click(function(){
  $("div").load("wrongfile.txt");
 });
});
</script>
</head>
<body>

  <div class="container">
    <h2> jQuery Ajax ajaxError() Method </h2>

  <div> <h2> Let AJAX change this text </h2>
</div>
  <button> Change Content </button>
</div>
</div>
</body>
</html>
```



jQuery – ajaxerror.

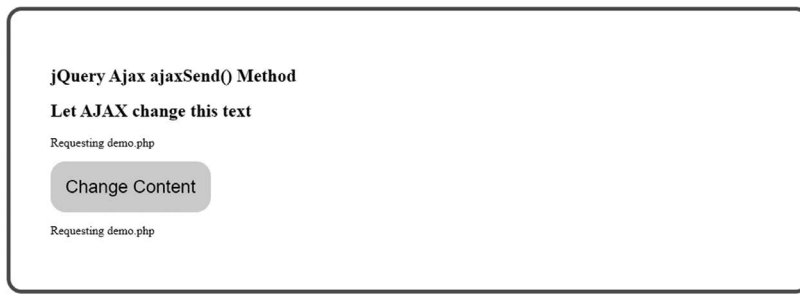
- ajaxSend(): It is used register a handler function to be called before AJAX request is sent.

  Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>
```

```
<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
</script>
<script>
  $(document).ready(function(){
   $(document).ajaxSend(function(e, xhr, opt){
    $("div").append("<p>Requesting " + opt.url +
"</p>");
   });
   $("button").click(function(){
    $("div").load("demo.php");
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Ajax ajaxSend() Method </h2>
    <div>
```

```
    <h2>Let AJAX change this text</h2>
     <button>Change Content</button>
     </div>
</body>
</html>
```



jQuery – ajaxSend method.

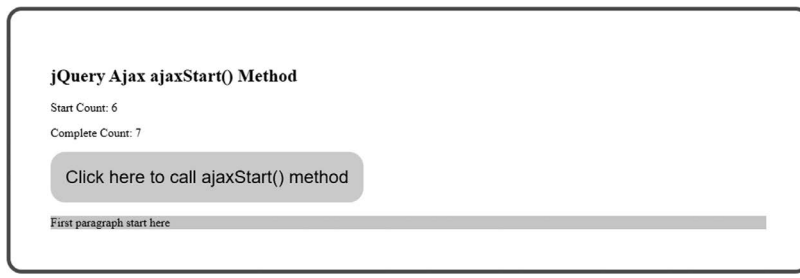
- ajaxStart(): It is used to register the handler function to be called when the first AJAX request begins. jQuery ajaxStart() method to be called first before the AJAX request begin to send. When an AJAX request to be sent, then jQuery checks whether there are any pending requests. If none of the requests pending, then jQuery triggers the ajaxStart event.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="http://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
```

```
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
</script>
<script>
  $(document).ready(function() {
   var count = 0;
   $("button").click(function(event){
    $("#msg").load("ajaxfile.html");
   });
   $(document).ajaxStart(function(){
    count++;
    $("#start").text("Start Count: " + count );
   });
   $(document).ajaxComplete(function(){
    count++;
    $("#complete").text("Complete Count: " +
count );
   });
  });
 </script>
</head>
<body>

  <div class="container">
    <h2> jQuery Ajax ajaxStart() Method </h2>
    <p id="start"> ajaxStart </p>
    <p id="complete"> ajaxComplete </p>
```

```
    <button> Click here to call ajaxStart()
method </button>
    <br /><br />
    <div id="msg" style="background-
color:pink;">First paragraph start here</div>

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



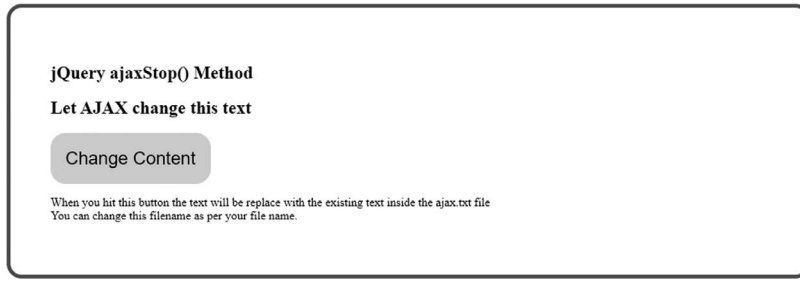
jQuery ajaxStart method.

- ajaxStop(): It is used to register the handler function to be called when all the AJAX requests have completed.

Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
```

```
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js">
</script>
<script>
  $(document).ready(function(){
   $(document).ajaxStop(function(){
    alert("All AJAX requests completed");
   });
   $("button").click(function(){
    $("div").load("ajax.txt");
    // $("div").load("demo_ajax_load.asp");
   });
  });
  </script>
</head>
<body>

  <div class="container">
    <h2> jQuery ajaxStop() Method </h2>
    <div><h2>Let AJAX change this text</h2>
</div>
    <button>Change Content</button>
    <p> When you hit this button the text will
be replace with the existing text inside the
ajax.txt file<br>
    You can change this filename as per your
file name.</p>
    </div>
</body>
</html>
```



jQuery – ajaxStop method.

- ajaxSuccess(): It is used to register a handler function to be called when AJAX request completes successfully. It attaches a function to be executed whenever an AJAX request completes successfully.

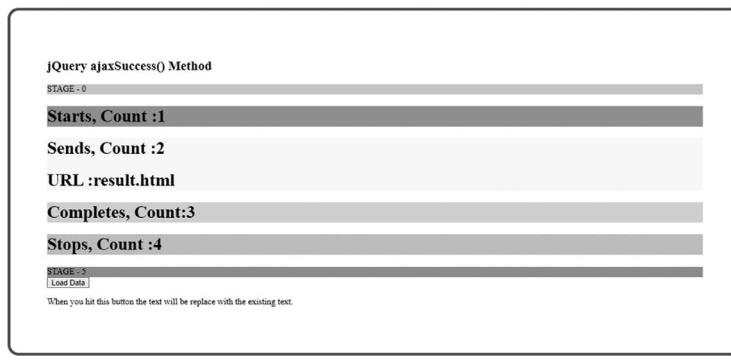
Example:

```
<!doctype html>
<html>
<head>
<title>The jQuery Example</title>
<script src="https://ajax.googleapis.com/ajax/
libs/jquery/3.6.0/jquery.min.js"></script>

<style>
      body{
      text-align: left;
      margin: 5%;
      padding:5%;
    }
    .container{
      border-radius: 20px;
      padding:5%;
      border: 5px solid rgb(136, 77, 77);
    }


    button{
        font-size: 24px;
        border: none;
        border-radius: 20px;
        padding: 20px;
```

```
        background-color:RGB(80, 243, 47)
      }

  </style>
<script src=
"http://ajax.googleapis.com/ajax/libs/
jquery/3.3.1/jquery.min.js"> </script>
<script type = "text/javascript" language =
"javascript">
    $(document).ready(function() {

      /* Global variable */
      var count = 0;

      $("#driver").click(function(event){
       $('#stage0').load('result.html');
      });

      /* Gets called when request starts */
      $(document).ajaxStart(function(){
       count++;
       $("#stage1").html("<h1>Starts, Count :" +
count + "</h1>");
      });

      /* Gets called when request is sent */
      $(document).ajaxSend(function(event, req,
set){
       count++;
       $("#stage2").html("<h1>Sends, Count :" +
count + "</h1>");
       $("#stage2").append("<h1> URL :" + set.
url + "</h1>");
      });

      /* Gets called when request completes */
      $(document).ajaxComplete(function(event,re
quest,settings){
       count++;
       $("#stage3").html("<h1> Completes,
Count:" + count + "</h1>");
      });
```

```
      /* Gets called when request is stopped */
      $(document).ajaxStop(function(event,reques
t,settings){
       count++;
       $("#stage4").html("<h1> Stops, Count :" +
count + "</h1>");
      });

      /* Gets called when all request completes
successfully */
      $(document).ajaxSuccess(function(event,req
uest,settings){
       count++;
       $("#stage5").html("<h1> Success, Count :"
+ count + "</h1>");
      });

    });
   </script>
</head>
<body>

  <div class="container">
    <h2> jQuery ajaxSuccess() Method </h2>
    <div id = "stage0" style =
"background-color:lightblue;">
    STAGE - 0
   </div>

   <div id = "stage1" style =
"background-color:lightcoral;">
    STAGE - 1
   </div>

   <div id = "stage2" style = "background-
color:light cyan;">
    STAGE - 2
   </div>

   <div id = "stage3" style =
"background-color:lightgreen;">
    STAGE - 3
   </div>
```

```
   <div id = "stage4" style = "background-
color:light pink;">
    STAGE - 4
   </div>

   <div id = "stage5" style =
"background-color:lightseagreen;">
    STAGE - 5
   </div>

   <input id = "driver" type = "button" id =
"driver" value="Load Data" />
    <p> When you hit this button the text will
be replaced with the existing text. </p>
    </div>
</body>
</html>
```



jQuery – ajaxSucces method.

BENEFITS OF jQuery

Web developers often use JavaScript to deliver the required functionality to the websites they create. jQuery is a single tool that offers great flexibility and power for web designers. This is a JavaScript library that helps to streamline and facilitate interaction between HTML objects and JavaScript code. Continue reading to learn the benefits of using jQuery instead of building your own library or writing raw JavaScript.

- jQuery is very popular with website developers because of its simplicity and ease of use. You will find a list of resources and information about jQueryon online, for example, code captions, blog posts,

quality tutorials, texts, and much more. With jQuery, you will never run out of resources and you will always find anything you need and all your questions will be answered when you work with this popular language.

- Most website developers find jQuery easy to read and accurate as the library is built using simple shortcodes. With its open coding standards and syntax, web designers can reduce the time it takes to run a site or application. With jQuery, developers do not have to be professional web designers or programmers to come up with the best styles for their websites. Web developers who have done the testing and encoding of CSS files will enjoy the easy implementation of jQuery.
- jQuery uses a powerful, clean, and simple syntax that makes it easy to select the DOM features on a web page you want to modify with JavaScript and enables you to combine results and actions to get a valid code. It is common to change multiple JavaScript lines with more than one line of jQuery code. jQuery uses version 3 of Cascading Style Sheets (CSS) to select elements, and this means that you do not need to learn new syntax when using this language.
- jQuery is a free open-source library and well supported in all various applications. This means that anyone can use this language in their programs without having to worry about any licensing or collaboration issues. In addition, jQuery is officially integrated into IDE Visual Studio 2010 by Microsoft. Additionally, jQuery intelligence is now well supported in Visual Studio 2010.
- To keep the jQuery library soft and lightweight, many functions have been removed and some have been moved to the plug-in section. If you need any of these omitted features, you can easily add them to your website as plugins. The soft library keeps coding at a limited level and helps save bandwidth to ensure faster uploads. The jQuery library content is 24 kb, which is smaller than the image on most websites, and the browser Apps will download and save it only once for use on all your web pages.
- The way a developer codes a website can greatly affect how it can be found in search engines. jQuery can be easily customized by search engines and has many plug-ins that can help developers to achieve this. One of the SEO-friendly practices you can use is to embed jQuery features using informal lists.

- jQuery provides App functions that help duplicate the unit of code, deductions, deception for similar members, and many other features. These functions provide integration between JavaScript and jQuery. With these important features of the App, the coding process will be smoother and easier.

DISADVANTAGES

- While jQuery has an impressive library in terms of quantity, also depending on much customization require on your website, the functionality may be limited thus using JavaScript.
- The jQuery JavaScript file is required to run commands. Although the size of this file is relatively small, it still can strain on the client system and also web server as well, if you intend to host the jQuery script on your own web server.

CHAPTER SUMMARY

In conclusion, we learned about jQuery, its fundamentals, history, advantages and disadvantages, features, and also various built-in methods used to create websites. There is various events method in jQuery such as keyboard, mouse, DOM, traversing, etc. In the next chapter, we will make our first jQuery project and will also explain the code.

NOTES

1. The history of jQuery - LogRocket
2. jQuery Installation - Medium

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

CHAPTER 2

jQuery Project One

To-Do List

IN THIS CHAPTER

- Introduction
- Requirement
- Code Explanation

The previous chapter is all about the jQuery introduction. Here in this chapter, we are going to cover one first jQuery project with a full explanation of each and every step.

INTRODUCTION

Before starting jQuery coding, you must know the requirements of the jQuery project.

You should be aware of HTML, some CSS, and a little bit of JavaScript to know how we link our files together for making projects. Let's talk about HTML first so that you get some hints about what we are going to do here.

HTML

This section provides an introduction to HTML. We cannot imagine web pages and the World Wide Web without HTML. It is the language mostly used to write web pages. It represents Hyper-Text Mark-up Language. You should know that any link found on web pages is usually called Hypertext,

DOI: 10.1201/9781003358497-2

and the mark refers to the tag or page layout in such a way that the text in the web page list is displayed in the correct format. The purpose of developing HTML was to understand the structure of any text: title, body, content, or categories.

So, basically, HTML provides a structured format to display the content of web pages. It is very simple and easy to understand. In the early 1990s, it was developed by Tim Berners-Lee and later went through many changes and modifications. HTML 5 is the latest version of HTML.

KEY HTML FEATURES

You know that HTML is a simple language that can use many tags to format content. All tags are enclosed within angular brackets <tagname>. With the exception of a few tags, most tags start with angular tags and end with corresponding angular tags.

<!DOCTYPE html> defines the document type and version of HTML. The HTML code starts just after the <html> angular marker and ends with </html>.

It usually has two major parts, namely the head and the body. Each category has its own set of requirements. You will get various sections in HTML structure. Let's discuss each.

Head Section

The header tag represents a web document header that you can tag <head> and <link> with. It starts with <head> and ends with </head>. It has parts of the title inside.

<title> This will be your webpage title. </title>>.

Body Section

It represents the body of a web document that usually contains titles, text, and sections. Topics start with <title> and end with </title>. Among these tags, the content can be labeled as "this is the first topic."

The paragraph will start with and end with . The content of the section should be written between these angular characters. The basic HTML code shown below in the overview section is used to create a simple HTML page.

```
<!DOCTYPE html>
<html lang="en">
<meta charset="utf-8">
```

```
<title> Page Title </title>
<body>
<h1> This is a Heading </h1>
<p> This is a paragraph. </p>

</body>
</html>
```

There are various elements we can add to our HTML code; let's take a look.

- HTML headings are elements defined by tags <h1> to <h6>, where <h1> defines the most important tag and <h6> defines the less important tag. The HTML section is an HTML element that will be defined using the <p> tag.
- HTML images are a feature of HTML and are defined by the <img> tag, and need to specify attributes such as image src, alt means other text, width, and height.
- HTML lists are elements and are defined using <ul> or <ol> tags, where <ul> is a random list and <ol> is an ordered list.
- The HTML table is part of the HTML and can be defined using tag <table> and tag lines <tr> and tag cells <td>.
- HTML links are elements and can be defined using the <a> tag.
- The HTML attribute style can be used with a combination of any HTML element such as <p>.
- In HTML, we use the lang attribute, we can say the language of the document using the <html> tag and the language defined using the lang attribute.
- In HTML, we can use the formatting elements to format the HTML document, and we can define special text features that have a special meaning. For example, HTML elements like <b> to bold and <i> to italics.
- In HTML, we can highlight specific text in a document using the <mark> feature to highlight text included in the <mark> element.

- In HTML, we can define text as the text above using the <sup> element in the HTML document so that the text embedded in the <sup> element becomes larger text.

The <!DOCTYPE> represents the document type and helps browsers to display web pages correctly. It must only appear once, at the top of the page (before any HTML tags). The <!DOCTYPE> declaration is not case sensitive.

Wherever the web is, it is HTML. HTML usage is distributed across all devices. Here is the list of features of HTML to know where we can use it.

- Browsers like Chrome, Firefox, and Safari all use HTML to render content on the web for better display.
- Various mobile browsers like Opera, Firefox Focus, Microsoft Edge, Dolphin, and Puffin all use HTML to better present and visualize online content on mobile phones.
- Various smart devices are embedded with HTML functions to better browse and navigate during their operations.
- HTML supports the first channel verification method on any web page to stop unwanted traffic.
- HTML accepts great content but gives the same visibility to smaller and larger screen devices.
- HTML supports a variety of colors, formats, and layouts.
- HTML uses templates that make website design easy.
- The HTML and XML syntax are very similar, so it is easy to work between the two domains.
- FrontPage, Dreamweaver, and many other development tools support HTML.
- HTML is a very useful search engine.

CSS

Cascading Style Sheets, better known as CSS, is a very simple design process used to make web pages more visible. CSS lets customize your web pages.

The best part about using these style features is that CSS is independent of the HTML way of creating web pages. The difference between the HTML and the CSS is that the first one is best known for providing a country layout on the web page while the other is intended to provide powerful coding and color coding techniques. CSS is used to control the composition of more than one web page at a time. All external style sheets are stored in the form of CSS files.

KEY CSS COMPONENTS

We just read the introduction to CSS, now let's move to the main CSS sections.

1. You can intend to make any changes in the world, just change the style, and you can see all the other features on all other web pages are updated automatically.
2. You can simply write the text once and use the same sheet as many times as you like.
3. CSS by comparison has a much wider range of attributes and lists compared to HTML. The HTML page can therefore have a sharper look and feel compared to standard HTML attributes.
4. CSS is considered a style sheet that is easy to use and read. This means that search engines do not have to make much effort in trying to read the text.
5. CSS can be used to store web applications locally with the help of the offline cache method, which can be used to view offline websites.

WHY DO WE USE CSS?

- For a website to work properly, it must have a fast download time. Nowadays, people usually wait a few seconds for a website to load. Therefore, it is important to ensure a fast pace. For companies looking to ensure fast and smooth website information, CSS becomes key to their success.
- CSS is easy to maintain due to the short maintenance time. This is because single-line code conversion affects the entire web page. Also, if upgrades are needed, make a small effort.

- You would not find many good and easy-to-use websites. One thing that is common to all of these websites is the consistency of construction. CSS empowers developers to ensure that style features are applied consistently across every few web pages.
- Due to its fast speed and easy maintenance, CSS saves a lot of time and effort in the web development process due to fast loading time. Here, a little time ensures the good performance of the designer.
- People use different smart devices to view a particular website. It can be a smartphone, PC, or laptop. For this purpose, websites are required to be compatible with the device. CSS ensures smooth operation by providing better alignment.
- You can change the location of the HTML tag with the help of CSS. You can set things as an image on any part of a web page as needed.

jQuery PROJECT: TO-DO LIST

PROJECT EXPLANATION

So first of all we make HTML structure from <html> tag to </html> closing tag. Now in the first line, it consists of head, title, meta tags with other tags like link inside, it has link tag for connecting Content Delivery Network (CDN) to the HTML file of Bootstrap, JavaScript, Popper.js for making all the scripts available to your HTML file. Suppose you are making some animation web page then this comes into use. Here we add the CDN of font-awesome.css for adding icons. You can download its file locally also. It comes in a zip file just you need to unzip that file and code the path where you keep it.

```
<link href="https://maxcdn.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css"
rel="stylesheet" crossorigin="anonymous">
```

Above, line is used to add icon fa icon such as trash, edit in the project. Next, we have <style> </style> tags in which we have some CSS attributes like body, h3, h2, .container, and so on for styling.

```
<style>

body {
```

```
    background: #746acd;  /* fallback for old browsers
*/
    background: linear-gradient(to right,  #57a77e,
#302b63, #57a77e);  /* Chrome 10-25, Safari 5.1-6 */
/* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera
12+, Safari 7+ */
    font-family: Roboto;
}

h3 {
    text-align: center;
    color: #CFD60A;
}

h2 {
    color: RGB(123,230,123);
}

.fa-plus {
    float: right;
}

.container {
    width: 400px;
    margin: 100px auto;
}

ul {
    list-style: none;
    margin: 0;
    padding: 0;
    position: relative;
}

li {
    height: 40px;
    line-height: 40px;
    color: RGB(230,230,12);
}

li:nth-child(2n) {
    background: rgba(123,123,123,0.5);
}
```

```
.left {
    background: #e74c3c;
    height: 40px;
    text-align: center;
    color: white;
    display: inline-block;
    width: 0;
    opacity: 0;
    transition: 0.2s linear;
}

.right {
    background: #e74c3c;
    height: 40px;
    text-align: center;
    color: white;
    display: inline-block;
    width: 0;
    opacity: 0;
    transition: 0.2s linear;
    position: absolute;
    right: 0px;
}

li:hover span{
    width: 35px;
    opacity: 1;
}

.todoinput {
    width: 100%;
    font-size: 18px;
    padding: 10px 10px 10px 15px;
    outline: none;
}

.editinput {
    height: 100%;
    width: 100%;
    outline: none;
}
```

```
.completed {
    text-decoration: line-through;
    color: gray;
}

.fa-pencil {
    font-size: 32px;
    }

    .fa-trash{
        font-size: 36px;
    }

</style>
```

Now, we have <body> tag in which we add div with container class, next <h3> tags with some text "To-Do List" and <h2> with text "Add your data here." So the main part of the coding begins here. After that, you will see an input field where anyone can add their data to the To-Do List.

```
 <h3> To-Do List</h3>
 <h2>Add your data here <i class="fa fa-plus"></i>
</h2>
 <input type="text" placeholder="Enter To-Do"
class="todoinput">
```

We apply class to input field to apply some CSS also attribute type="text" to tell the user to add alphanumeric data.

Next, we add tag <ul>, <li>. These tags are used to make lists on the HTML page. The syntax of making list is simple You need to follow this in the sequence.

```
<ul>
            <li>
                    I woke up early today.
            </li>
        </ul>
```

The code in the project is given below:

```
        <ul>
            <li>
```

```
            <span class="left">
                <i class="fa fa-trash"></i>
            </span>
            <span class="text">
                I woke up early today.
            </span>
            <span class="right">
                <i class="fa fa-pencil"> </i>
            </span>
        </li>
        <li>
            <span class="left">
                <i class="fa fa-trash"> </i>
            </span>
            <span class="text">
                Do some exercise
            </span>
            <span class="right">
                <i class="fa fa-pencil"> </i>
            </span>
        </li>
        <li>
            <span class="left">
                <i class="fa fa-trash"> </i>
            </span>
            <span class="text">
                Then go market
            </span>
            <span class="right">
                <i class="fa fa-pencil"> </i>
            </span>
        </li>
    </ul>
```

In this project you will find some pre-write sentences in To-Do List, you can change these as well. We discussed in brief about HTML and CSS, now let's discuss jQuery. Before starting it, let's memorize the concept of jQuery.

jQuery

jQuery is an open-source for JavaScript. It is simple and useful if we have to write a little and do more. jQuery is used to take a lot of common tasks that require multiple lines of JavaScript code and wrap it up

in a variety of ways. This method can be called a single line of code. So to make it easier to use JavaScript on your website. Explaining the difference between JavaScript and jQuery, Skillcrush states, "JavaScript is an independent programming language, whereas jQuery is a JavaScript code (not its language)."

"jQuery is a fast, small JavaScript library embedded in a single .js file. It offers a wide range of built-in functions that can accomplish a variety of tasks.

WHY DO WE CHOOSE TO USE jQuery?

It is noteworthy that most web developers prefer to use jQuery to improve their websites. This is because jQuery helps to simplify and measure the interaction between JavaScript and HTML code elements. It helps developers to build interactive and dynamic websites. It lets the developer upgrade AJAX templates easily.

jQuery is a powerful tool that will never run out of resources and you will always find whatever you need. It also promotes simplicity and coherence. It has clean and beautiful code and is easy with simple syntax, open-source libraries, animation, and amazing effects. It is extremely flexible, provides a load of fast pages and is SEO-friendly, making it widely used by web designers.

In this project, we add CDN of jQuery minified like:

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/
jquery/3.2.1/jquery.min.js"></script>
```

jQuery code always starts with $() symbol.

```
$(".todoinput").keypress(function(event) {
    if( event.which === 13)  {
        if($(this).val() === "" ) return;
      $( "ul" ).prepend(" <li> <span class='left' > <i
class='fa fa-trash'> </i> </span> " + $(this).val() +
"<span class='right'> <i class='fa fa-pencil'> </
li>");
                $(this).val( "" );
    }
});
```

First of all, we understand the use of " $ " as a selector function that selects DOM elements such as button, paragraph, input, and so on. $("todoinput")

here we select the input field and apply keypress function when we enter any input to it the condition checks that if the user hits the "Enter" key means its ASCII number is 13 and inside it we have another condition that checks empty string, it means the input is empty then we don't return nothing where === is that your data type and your data value should be same. In programming === it is called as strict data type operator.

Next, if the user adds some data in the input and hits enter, then first if condition works. Inside this condition, we append the class left and class right with trash and pencil with the user input content. This keyword is used to refer the current variable.

```
$(this).val( "" );
```

In the second section, we target .fa-plus means when the user clicks on the + symbol, the input will be trigger and perform slideToggle() method that is built-in in jQuery. This method checks the selected elements for visibility. slideDown() runs if an element is hidden. slideUp() runs if an element is visible – this creates a toggle effect.

```
$(" .fa-plus" ).on( "click", function() {
    $( "input" ).slideToggle();
});
```

In the third section, we have <ul> tag having on() with various parameters like "click," "span.left," and callback function. Here we get to know how to trigger using the class method.

```
$( "ul" ).on( "click", "span.text", function(event) {
    $(this).toggleClass( "completed" );
});
```

The "span.text" is in the <li> so that the user can add animation to the fa icon in the code. Here we use toggleClass for animation. This method toggles between adding and removing one or more class names from the selected elements.

```
$( "ul" ).on( "click", "span.left", function(event) {
    $(this).parent().fadeOut(1000, function(){
        $(this).remove();
```

```
    });
    event.stopPropagation();
});
```

In the above code we have same "ul" attribute, on function with various parameters such as "click" "span.left" and callback function. This keyword is looking at the current stage that performs parent() function. The parent() method returns the direct parent element of the selected element. These methods only traverse a single level up the DOM tree. When we hit the spna.left element that will edit trash icon, the current data of that span will be removed and also perform some fadeout(1000) function that is used for hiding the element slower as per normal speed. The remove() function helps to remove the current list element with all the CSS.

```
    event.stopPropagation();
```

This method stops the bubbling of an event to parent elements, preventing any parent event handlers from being executed. The event parameter comes from the event binding function.

Now the below code is the last step of this project:

```
$( "ul" ).on( "click", "span.right", function(event) {
    var parent = $(this).parent();
    var oldVal = parent.text();
    parent.html( "<input type='text'
class='editinput'>" );
    $( ".editinput" ).keypress(function(e) {
        if(e.which === 13) {
            if($(this).val() === "") {
                parent.html( "<span class='left'> <i
class='fa fa-trash'> </i> </span> <span class='text'>
" + oldVal + " </span> <span class='right'> <i
class='fa fa-pencil'>" );
            }
            else {
                var newVal = $(this).Val();
                parent.html( "<span class='left'> <i
class='fa fa-trash'> </i> </span> <span class='text'>
" + newVal + "</span> <span class='right'> <i
class='fa fa-pencil'>" );
            }
```

```
        }
        e.stopPropagation();
    });
```

Here what we are doing is the <ul>, "span.right", on() function with various parameters gets the events as the argument. Now all the happen is done by the pencil fa icon which means edit option.

```
    var parent = $(this).parent();
    var oldVal = parent.text();
```

This above code gets the parent and the value as oldVal. Next, we add some HTML() functions to the parent as given below:

```
    parent.html( "<input type='text'
class='editinput'>" );
```

Then we get input filed class for triggering the data. We select input field and apply keypress function. When we enter any input to it, the condition check, that if the user hits the "Enter" key means its ASCII number is 13 and inside it, we have another it means the input is empty then we don't return nothing.

If the user adds data and hits enter, the new data is attached with the rest of the list in the To-Do List. If the user adds nothing new, then this code will run as given below:

```
parent.html( "<span class='left'> <i class='fa
fa-trash'> </i> </span> <span class='text'> " + oldVal
+ " </span> <span class='right'> <i class='fa
fa-pencil'>" );
```

If the user adds something new, then this code will run as given below:

```
                parent.html( "<span class='left'> <i
class='fa fa-trash'> </i> </span> <span class='text'>
" + newVal + "</span> <span class='right'> <i
class='fa fa-pencil'>" );
```

Here, we have complete project coding:

```
<html>
<head>
    <meta name="viewport" content="width=device-width,
initial-scale=1">
    <title>To-Do List</title>
  <link href="https://maxcdn.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css"
rel="stylesheet" crossorigin="anonymous">
</head>
<style>

body {
    background: #746acd;  /* fallback for old browsers
*/
    background: linear-gradient(to right,  #57a77e,
#302b63, #57a77e);  /* Chrome 10-25, Safari 5.1-6 */
/* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera
12+, Safari 7+ */
    font-family: Roboto;
}

h3 {
    text-align: center;
    color: #CFD60A;
}

h2 {
    color: RGB(123,230,123);
}

.fa-plus {
    float: right;
}

.container {
    width: 400px;
<CODE>
    margin: 100px auto;
}
```

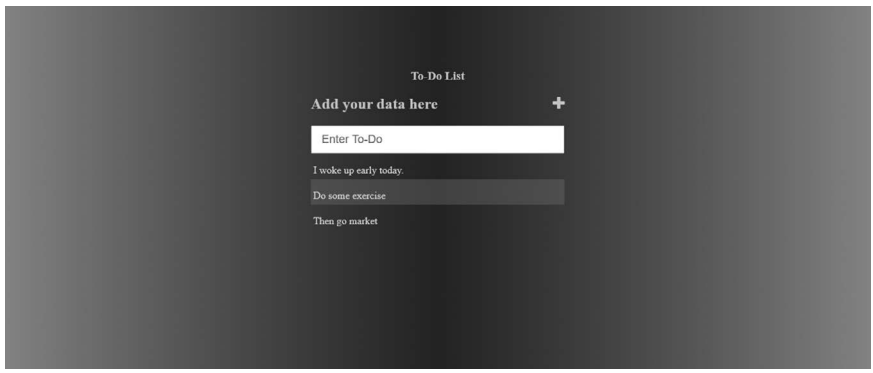
```
ul {
    list-style: none;
    margin: 0;
    padding: 0;
    position: relative;
}

li {
    height: 40px;
    line-height: 40px;
    color: RGB(230,230,12);
}

li:nth-child(2n) {
    background: rgba(123,123,123,0.5);
}

.left {
    background: #e74c3c;
    height: 40px;
    text-align: center;
    color: white;
    display: inline-block;
    width: 0;
    opacity: 0;
    transition: 0.2s linear;
}

.right {
    background: #e74c3c;
    height: 40px;
    text-align: center;
    color: white;
    display: inline-block;
    width: 0;
    opacity: 0;
    transition: 0.2s linear;
    position: absolute;
    right: 0px;
}

li:hover span{
    width: 35px;
```

```
    opacity: 1;
}

.todoinput {
    width: 100%;
    font-size: 18px;
    padding: 10px 10px 10px 15px;
    outline: none;
}

.editinput {
    height: 100%;
    width: 100%;
    outline: none;
}


.completed {
    text-decoration: line-through;
    color: gray;
}

.fa-pencil {
    font-size: 32px;
    }

    .fa-trash{
        font-size: 36px;
    }

     </style>
<body>

    <div class="container">
        <h3> To-Do List</h3>
        <h2>Add your data here <i class="fa
fa-plus"></i></h2>
        <input type="text" placeholder="Enter To-Do"
class="todoinput">
        <ul>
            <li>
                <span class="left">
                    <i class="fa fa-trash"></i>
                </span>
```

```
                <span class="text">
                    I woke up early today.
                </span>
                <span class="right">
                    <i class="fa fa-pencil"> </i>
                </span>
            </li>
            <li>
                <span class="left">
                    <i class="fa fa-trash"> </i>
                </span>
                <span class="text">
                    Do some exercise
                </span>
                <span class="right">
                    <i class="fa fa-pencil"> </i>
                </span>
            </li>
            <li>
                <span class="left">
                    <i class="fa fa-trash"> </i>
                </span>
                <span class="text">
                    Then go market
                </span>
                <span class="right">
                    <i class="fa fa-pencil"> </i>
                </span>
            </li>
        </ul>
    </div>

    <script src="https://cdnjs.cloudflare.com/ajax/
libs/jquery/3.2.1/jquery.min.js"></script>

    <script>

        $(".todoinput").keypress(function(event) {
            if( event.which === 13)  {
            if($(this).val() === "" ) return;
                $( "ul" ).prepend(" <li> <span
class='left' > <i class='fa fa-trash'> </i> </span> "
+ $(this).val() + "<span class='right'> <i class='fa
fa-pencil'> </li>");
```

```
                $(this).val( "" );
    }
});

$(" .fa-plus" ).on( "click", function() {
    $( "input" ).slideToggle();
});

$( "ul" ).on( "click", "span.text", function(event) {
    $(this).toggleClass( "completed" );
});

$( "ul" ).on( "click", "span.left", function(event) {
    $(this).parent().fadeOut(1000, function(){
        $(this).remove();
    });
    event.stopPropagation();
});

$( "ul" ).on( "click", "span.right", function(event) {
    var parent = $(this).parent();
    var oldVal = parent.text();
    parent.html( "<input type='text'
class='editinput'>" );
    $( ".editinput" ).keypress(function(e) {
        if(e.which === 13) {
            if($(this).val() === "") {
                parent.html( "<span class='left'> <i
class='fa fa-trash'> </i> </span> <span class='text'>
" + oldVal + " </span> <span class='right'> <i
class='fa fa-pencil'>" );
            }
            else {
                var newVal = $(this).Val();
                parent.html( "<span class='left'> <i
class='fa fa-trash'> </i> </span> <span class='text'>
" + newVal + "</span> <span class='right'> <i
class='fa fa-pencil'>" );
            }
        }
        e.stopPropagation();
    });
```

```
    event.stopPropagation();
});
        </script>
</body>
</html>
```



To-Do List.

CHAPTER SUMMARY

We covered our first project with a coding explanation of To-Do List. Before starting the project, we covered some basic explanations of HTML and CSS to use it in any project.

CHAPTER 3

jQuery Project Two

Portfolio

IN THIS CHAPTER

- Introduction
- Requirement
- Code Explanation

In the previous chapter, we covered the first jQuery. Here in this chapter, we are going to cover the second jQuery project with a full explanation of each and every step.

INTRODUCTION

In this project, we will be going to make a simple jQuery-based Portfolio where you will get some animation. Before starting jQuery coding, you must know the requirements of the jQuery project. You should be aware of HTML, some CSS, and a little bit of JavaScript to know how we link our files together for making projects. So let's talk about HTML first so that you get some hints about what we are going to do here.

HTML

This section provides an introduction to HTML. We cannot imagine web pages and the World Wide Web without HTML. It is the language mostly used to write web pages. It represents Hyper-Text Mark-up Language. You

DOI: 10.1201/9781003358497-3

should know that any link found on web pages is usually called Hypertext, and the mark refers to the tag or page layout in such a way that the text in the web page list is displayed in the correct format. The purpose of developing HTML was to understand the structure of any text: title, body, content, or categories.

So, basically, HTML provides a structured format to display the content of web pages. It is very simple and easy to understand. In the early 1990s, it was developed by Tim Berners-Lee and later went through many changes and modifications. HTML 5 is the latest version of HTML.

KEY HTML FEATURES

You know that HTML is a simple language that can use many tags to format content. All tags are enclosed within angular brackets <tagname>. With the exception of a few tags, most tags start with angular tags and end with corresponding angular tags.

<!DOCTYPE html> defines the document type and version of HTML. The HTML code starts just after the <html> angular marker and ends with </html>.

It usually has two major parts, namely the head and the body. Each category has its set of requirements. You will get various sections in HTML structure. Let's discuss each.

Head Section

The header tag represents a web document header that you can tag <head> and <link> with. It starts with <head> and ends with </head>. It has parts of the title inside.

<title> This will be your webpage title. </title>>.

Body Section

It represents the body of a web document that usually contains titles, text, and sections. Topics start with <title> and end with </title>. Among these tags, the content can be labeled as "this is the first topic."

The paragraph will start with and end with . The content of the section should be written between these angular characters. The basic HTML code shown below in the overview section is used to create a simple HTML page.

```
<!DOCTYPE html>
<html lang="en">
<meta charset="utf-8">
```

```
<title> Page Title </title>
<body>
<h1> This is a Heading </h1>
<p> This is a paragraph. </p>
</body>
</html>
```

There are various elements we can add to our HTML code; let's take a look.

- HTML headings are elements defined by tags <h1> to <h6>, where <h1> defines the important tag and <h6> defines the less important tag. The HTML section is an HTML element that will be defined using the <p> tag.
- HTML images are a feature of HTML and are defined by the <img> tag, and need to specify attributes such as image src, alt means other text, width, and height.
- HTML lists are elements and are defined using <ul> or <ol> tags, where <ul> is a random list and <ol> is an ordered list.
- The HTML table is part of the HTML and can be defined using tag <table> and tag lines <tr> and tag cells <td>.
- HTML links are elements, and can be defined using the <a> tag:
- The HTML attribute style can be used with a combination of any HTML element such as <p>.
- In HTML, we use the lang attribute, we can say the language of the document using the <html> tag and the language defined using the lang attribute.
- In HTML, we can use the formatting elements to format the HTML document, and we can define special text features that have a special meaning. For example, HTML elements like <b> to bold and <i> to italics.
- In HTML, we can highlight specific text in a document using the <mark> feature to highlight text included in the <mark> element.
- In HTML, we can define text as the text above using the <sup> element in the HTML document so that the text embedded in the <sup> element becomes larger text.

The <!DOCTYPE> represents the document type and helps browsers to display web pages correctly. It should appear once, at the top of the page (before any HTML tags). The <!DOCTYPE> declaration is not case sensitive.

Wherever the web is, it is HTML. HTML usage is distributed across all devices. Here is the list of features of HTML to know where we can use it.

- Browsers like Chrome, Firefox, and Safari all use HTML to render content on the web for better display.
- Various mobile browsers like Opera, Firefox Focus, Microsoft Edge, Dolphin, and Puffin all use HTML to better present and visualize online content on mobile phones.
- Various smart devices are embedded with HTML functions to better browse and navigate during their operations.
- HTML supports the first channel verification method on any web pages to stop unwanted traffic.
- HTML accepts great content but gives the same visibility to smaller and larger screen devices.
- HTML supports a variety of colors, formats, and layouts.
- HTML uses templates that make website design easy.
- The HTML and XML syntax are very similar, so it is easy to work between the two domains.
- FrontPage, Dreamweaver, and many other development tools support HTML.
- HTML is a very useful search engine.

CSS

Cascading Style Sheets, better known as CSS, is a very simple design process used to make web pages more visible. CSS lets customize your web pages. The best part about using these style features is that CSS is independent of the HTML way of creating web pages. The difference between the HTML and the CSS is that the first one is best known for providing a country layout on the web page while the other is intended to provide powerful coding and color coding techniques. CSS is used to control the

composition of more than one webpage at a time. All external style sheets are stored in the form of CSS files.

KEY CSS COMPONENTS

We just read the introduction to CSS, now let's move to the main CSS sections.

1. You can intend to make any changes in the world, just change the style, and you can see all the other features on all other web pages are updated automatically.
2. You can simply write the text once and use the same sheet as many times as you like.
3. CSS by comparison has a much wider range of attributes and lists compared to HTML. The HTML page can therefore have a sharper look and feel compared to standard HTML attributes.
4. CSS is considered a style sheet that is easy to use and read. This means that search engines do not have to make much effort in trying to read the text.
5. CSS can be used to store web applications locally with the help of the offline cache method, which can be used to view offline websites.

WHY DO WE USE CSS?

- For a website to work properly, it must have a fast download time. Nowadays, people usually wait a few seconds for a website to load. Therefore, it is important to ensure a fast pace. For companies looking to ensure fast and smooth website information, CSS becomes key to their success.
- CSS is easy to maintain due to the short maintenance time. This is because single-line code conversion affects the entire web page. Also, if upgrades are needed, make a small effort
- You would not find many good and easy-to-use websites. One thing that is common to all of these websites is the consistency of construction. CSS empowers developers to ensure that style features are applied consistently across every few web pages.

- Due to its fast speed and easy maintenance, CSS saves a lot of time and effort in the web development process due to fast loading time. Here, a little time ensures the good performance of the designer.
- People use different smart devices to view a particular website. It can also be a smartphone, PC, or laptop. For this purpose, websites are required to be compatible with the device. CSS ensures smooth operation by providing better alignment.
- You can change the location of the HTML tag with the help of CSS. You can set things as an image on any part of a web page as needed.

jQuery PROJECT: PORTFOLIO

A portfolio is a combination of beliefs, skills, qualifications, education, training, and experience. It provides insight into your personality and the way you work. Choosing the most important and easily understood experience will show the employer proof of organization, communication, and practical work-related skills.

PROJECT EXPLANATION

So first of all we make HTML structure from <html> tag to </html> closing tag. Now in the first line, it consists of head, title, meta tags with other tags like link inside, it has link tag for connecting CDN to the HTML file of Bootstrap, JavaScript, Popper.js for making all the scripts available to your HTML file. Suppose you are making some animation web page then this comes into use. Here we add the CDN of font-awesome.css for adding icons. You can download its file locally also. It comes in a zip file just you need to unzip that file and code the path where you keep it.

```
 <link rel="stylesheet" href="css/reset.css">
        <link rel="stylesheet" href="css/animate.css">
        <link rel="stylesheet" href="css/lity.min.
css">
        <link href="https://fonts.googleapis.com/css?f
amily=Merriweather:400,700|Muli:300,400"
rel="stylesheet">
        <link rel="stylesheet" href="https://use.
fontawesome.com/releases/v5.8.1/css/all.css"
integrity="sha384-50oBUHEmvpQ+1lW4y57PTFmhCaXp0ML5d60M
1M7uH2+nqUivzIebhndOJK28anvf" crossorigin="anonymous">
        <link rel="stylesheet" href="css/style.css">
```

Next, we have <style> </style> tags in which we have some CSS attributes like body, h3, h2, .container, and so on for styling.

```
* {
    -webkit-box-sizing: border-box;
    box-sizing: border-box;
}

HTML {
    scroll-behavior: smooth;
}

/*HELPER CLASS*/

.mg-bot-20 {
    margin-bottom: 20px;
}

.flex-spacer {
    -webkit-box-flex:1;
        -ms-flex:1;
            flex:1;
}

/*GLOBAL STYLES*/

.container {
    max-width: 1140px;
    margin: 0 auto;
    padding: 60px 20px 60px 20px;
}

p,
a {
    color: #808080;
    font-family: 'Muli', sans-serif;
    line-height: 24px;
    size: 16px;
}

a {
    text-decoration: none;
```

```
    color: #383838;
}

h1 {
    font-family: 'Merriweather', serif;
    font-size: 30px;
    line-height: 1.4em;
    color:#383838;
}

h2 {
    font-family: 'Merriweather', serif;
    font-size: 42px;
    color:#eabf3f;
}

h3 {
    font-family: 'Merriweather', serif;
    font-size: 24px;
}

.btn-primary {
    background-color: #383838;
    border:1px solid #342a0b;
    border-radius: 3px;
    color:#F5F5F5;
    display: inline-block;
    font-family: 'Merriweather', serif;
    font-size: 24px;
    font-weight: 300;
    padding: 15px 42px;
    transition: 0.3s;
    -webkit-transition: 0.3s;
}

.btn-primary:hover {
    background-color: #303030;
    -webkit-transition: 0.3s;
    transition: 0.3s;
}

/*END GLOBAL STYLES */
```

```
section#hero {
    background-color: #b0b0b0;
    background-image: -WebKit-gradient(linear,left
top, left bottom,from(rgba(176,176,176,.28)),to(r
gba(176,176,176,.28))), URL('../img/hero-bg.jpg');
    background-image: linear-gradient(rgba(176,176,176,
.28),rgba(176,176,176,.28));
    background-repeat: no-repeat;
    background-size: cover;
    background-position: center top;
    min-height: 68vh;
}

section#hero .container {
    padding-top:40px;
}

.site-name {
    text-decoration: none;
    color: #383838;
    font-family: 'Merriweather', serif;
    font-size:28px;
    text-shadow: 0px 2px 1.98px rgba(40, 40, 40, 0.41);
}
```

The CSS file is too lengthy, so you can file this file in the "css/style.css." You can also edit according to your need. There are other various files you will get in the CSS folder. All these are locally downloaded files.

Next our body section begins, we have various sections in the body tag.

This is the first section in which we get our navbar. It is located at the top of the website page. On the right side we have various navigating links and on the left side we can add any name.

```
 <section id="hero">
            <div class="container">
                <div class="site-header">
                    <div class="header-logo">
                        <a href="#" class="site-name">
Developer <span> Name </span></a>
                    </div>
                    <span class="flex-spacer"></span>
<!-- IE WORKAROUND -->
```

```
                    <nav class="site-nav">
                        <ul>
                            <!--<li><a href="#">Home
</a></li>-->
                            <li><a href="#about">
About</a></li>
                            <li><a href="#portfolio">
Portfolio</a></li>
                            <li><a class="contact-me
btn-primary" href="mailto:abecilla.paularmand@gmail.
com">Contact Me</a></li>
                        </ul>
                    </nav>
                    <!-- Mobile hidden nav -->
                    <nav class="mobile-nav overlay">
                        <div class="nav-burger toggle"
id="toggle">
                            <span class="bar top">
</span>
                            <span class="bar
middle"></span>
                            <span class="bar
bottom"></span>
                        </div>
                        <ul class="nav-overlay"
id="overlay">
                            <!--<li><a href="#">Home
</a></li>-->
                            <li><a href="#about"
class="toggle">About</a></li>
                            <li><a href="#portfolio"
class="toggle">Portfolio</a></li>
                            <li><a
href="mailto:abecilla.paularmand@gmail.com"
class="toggle">Contact Me</a></li>
                        </ul>
                    </nav>
                </div>
                <div class="hero-content">
                    <h1 class="cta">Hi. I'm "Your
Name"</h1>
                    <p>Web Designer / Developer</p>
                    <a href="#portfolio"
class="button-cta">View My Portfolio</a>
```

```
                </div>
            </div>
        </section>
```

Next you get about section here to tell about yourself who we are so that and whosoever check your profile summary will be highlighted to other.

```
  <section id="about">
            <div class="container">
                <div class="about-pic">
                    <img src="img/Avtar.png"
class="about-img" alt="Beach Picture 2018">
                </div>
                <div class="about-content">
                    <h2>About</h2>
                    <p class="mg-bot-20">Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Nam arcu
ante, convallis sit amet ex sed, dignissim aliquam
turpis. Quisque sollicitudin malesuada laoreet. Donec
orci orci, sodales vitae leo mattis, gravida finibus
velit. Class aptent taciti sociosqu ad litora conubia
nostra, per inceptos himenaeos. Etiam augue lectus,
fringilla et nisl et, varius maximus purus. </p>
                    <p class="mg-bot-20">Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Nam arcu
ante, convallis sit amet ex sed, dignissim aliquam
turpis. Quisque sollicitudin malesuada laoreet. Donec
orci orci, sodales vitae leo mattis, gravida finibus
velit. Class aptent taciti sociosqu ad litora conubia
nostra, per inceptos himenaeos. Etiam augue lectus,
fringilla et nisl et, varius maximus purus.
                        <span>Lorem </span>, Lorem
ipsum dolor sit amet, consectetur adipiscing elit. Nam
arcu ante, convallis sit amet ex sed, dignissim
aliquam turpis. Quisque sollicitudin malesuada
laoreet. Donec orci orci, sodales vitae leo mattis,
gravida finibus velit. Class aptent taciti sociosqu
litora torquent per nostra, per inceptos himenaeos. </p>
                </div>
            </div>
        </section>
```

The third section is about skills. The interviewer or any person can get the user skills by checking this section. You can also edit it if you learn new skill rather than listing:

```
 <section id="skills">
            <div class="container">
                <div class="skills-content">
                    <h2>Skills</h2>
                    <p class="mg-bot-20">Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Nam arcu
ante, convallis sit amet ex sed, dignissim aliquam
turpis. Quisque sollicitudin malesuada laoreet. Donec
orci orci, sodales vitae leo mattis, gravida finibus
velit. Class aptent taciti sociosqu adlitora torquent
conubia nostra, per inceptos himenaeos. Etiam augue
lectus, fringilla et nisl et, varius maximus purus. </p>
                    <div class="skills-icons">
                        <i class="fab fa-html5"></i>
                        <i class="fab fa-css3"></i>
                        <i class="fab fa-wordpress"></i>
                        <i class="fab fa-js"></i>
                        <i class="fab fa-adobe"></i>
                        <i class="fab fa-php"></i>
                        <i class="fas fa-database"></i>
                    </div>
                    <!--<a href="#" class="btn-
primary">View my Work</a>-->
                </div>
                <div class="skills-rating">
                    <p>HTML5 &amp; CSS3</p>
                    <div class="outer-meter">
                        <div class="inner-meter
html5"></div>
                    </div>
                    <p>Photoshop</p>
                    <div class="outer-meter">
                        <div class="inner-meter
photoshop"></div>
                    </div>
                    <p>Javascript &amp; jQuery</p>
                    <div class="outer-meter">
```

```
                        <div class="inner-meter
js-jquery"></div>
                    </div>
                    <p>Wordpress</p>
                    <div class="outer-meter">
                        <div class="inner-meter
wordpress"></div>
                    </div>
                    <p>PHP &amp; MySQL</p>
                    <div class="outer-meter">
                        <div class="inner-meter php-
mysql"></div>
                    </div>
                    <p>ReactJS and Redux (Learning in
progress)</p>
                    <div class="outer-meter">
                        <div class="inner-meter react-
redux"></div>
                    </div>
                </div>
            </div>
        </section>
```

The below section is about your recent project, you can add its images and zip files so that anyone can download your projects online.

```
 <section id="portfolio">
            <div class="container">
                <h2>Recent <span>Projects</span></h2>
                <div class="thumbnails-wrapper">
                    <img src="img/thumbnail-1.jpg"
alt="Garage Door Company sample" class="project-
thumbnails" data-lity-target="img/thumbnail-1.jpg">
                    <img src="img/thumbnail-2.jpg"
alt="Automotive template sample" class="project-
thumbnails" data-lity-target="img/thumbnail-2.jpg">
                    <img src="img/thumbnail-3.jpg"
alt="Construction sample template" class="project-
thumbnails" data-lity-target="img/thumbnail-3.jpg">
                </div>
            </div>
        </section>
```

The last section is the footer section. In this section, you find some social links to applications. The user can add social link such as Instagram, Facebook, and so on in this section.

```
   <section id="footer">
            <section id="quote">
                <div class="container">
                    <h3>"H Quisque sollicitudin
malesuada laoreet. Donec orci orci, sodales vitae leo
mattis, gravida finibus velit. Class aptent taciti
sociosqu ad litora torquent per conubia nostra, per
inceptos himenaeos.."</h3>
                    <p>-Lorem ipsum dolor sit amet</p>
                </div>
            </section>
            <section id="copyright">
                <div class="container">
                    <p>&copy;Lorem ipsum dolor sit
amet </p>
                    <span class="flex-spacer"></span>
<!-- IE WORKAROUND -->
                    <div class="social-icons">
                        <a href="https://www.facebook.
com/armand.abecilla" target="_blank"><i class="fab
fa-facebook-f"></i></a>
                        <a href="https://github.com/
armandAbecilla" target="_blank"><i class="fab
fa-github"></i></a>
                        <a href="https://open.spotify
.com/user/dnamra0216" target="_blank"><i class="fab
fa-spotify"></i></a>
                        <a href="https://soundcloud
.com/armand-abecilla" target="_blank"><i class="fab
fa-soundcloud"></i></a>
                    </div>
                </div>
            </section>
        </section>
```

In the end, you will find some script links. These are used to link some useful files of JavaScript. These files contain various types of properties which we can use as per needs.

```
        <script type="text/javascript" src="js/jquery-
3.3.1.min.js"></script>
        <script type="text/javascript" src="js/jquery
.waypoints.min.js"></script>
        <script type="text/javascript" src="js/wow
.min.js"></script>
        <script src="js/lity.min.js"
charset="utf-8"></script>
        <script type="text/javascript" src="js/main
.js"></script>
```

Here is the complete HTML file with you:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Armand Abecilla</title>
        <meta charset="utf-8">
        <meta HTTP-equiv="x-ua-compatible"
content="IE=edge, chrome=1"> <!-- for internet
explorer -->
        <meta name="viewport" content="width=device-
width, initial-scale=1"> <!-- mobile device optimized
-->
        <link rel="stylesheet" href="css/reset.css">
        <link rel="stylesheet" href="css/animate.css">
        <link rel="stylesheet" href="css/lity.min.
css">
        <link href="https://fonts.googleapis.com/css?f
amily=Merriweather:400,700|Muli:300,400"
rel="stylesheet">
        <link rel="stylesheet" href="https://use.
fontawesome.com/releases/v5.8.1/css/all.css"
integrity="sha384-50oBUHEmvpQ+1lW4y57PTFmhCaXp0ML5d60M
1M7uH2+nqUivzIebhndOJK28anvf" crossorigin="anonymous">
        <link rel="stylesheet" href="css/style.css">
    </head>
    <body>
        <section id="hero">
            <div class="container">
                <div class="site-header">
                    <div class="header-logo">
```

```
                        <a href="#" class="site-name">
Developer <span> Name </span></a>
                    </div>
                    <span class="flex-spacer"></span>
<!-- IE WORKAROUND -->
                    <nav class="site-nav">
                        <ul>
                            <!--<li><a href="#">Home
</a></li>-->
                            <li><a
href="#about">About</a></li>
                            <li><a
href="#portfolio">Portfolio</a></li>
                            <li><a class="contact-me
btn-primary" href="mailto:abecilla.paularmand@gmail.
com">Contact Me</a></li>
                        </ul>
                    </nav>
                    <!-- Mobile hidden nav -->
                    <nav class="mobile-nav overlay">
                        <div class="nav-burger toggle"
id="toggle">
                            <span class="bar top">
</span>
                            <span class="bar
middle"></span>
                            <span class="bar
bottom"></span>
                        </div>
                        <ul class="nav-overlay"
id="overlay">
                            <!--<li><a href="#">Home
</a></li>-->
                            <li><a href="#about"
class="toggle">About</a></li>
                            <li><a href="#portfolio"
class="toggle">Portfolio</a></li>
                            <li><a
href="mailto:abecilla.paularmand@gmail.com"
class="toggle">Contact Me</a></li>
                        </ul>
                    </nav>
                </div>
```

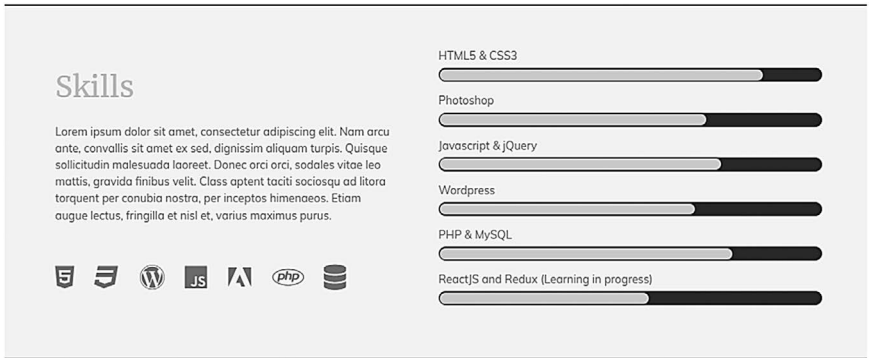
```
                <div class="hero-content">
                    <h1 class="cta">Hi. I'm "Your
Name"</h1>
                    <p>Web Designer / Developer</p>
                    <a href="#portfolio"
class="button-cta">View My Portfolio</a>
                </div>
            </div>
        </section>
        <section id="about">
            <div class="container">
                <div class="about-pic">
                    <img src="img/my-picture.jpg"
class="about-img" alt="Beach Picture 2018">
                </div>
                <div class="about-content">
                    <h2> About </h2>
                    <p class="mg-bot-20">Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Nam arcu
ante, convallis sit amet ex sed, dignissim aliquam
turpis. Quisque sollicitudin malesuada laoreet. Donec
orci orci, sodales vitae leo mattis, gravida finibus
velit. Class aptent taciti sociosqu ad litora torquent
conubia nostra, per inceptos himenaeos. Etiam augue
lectus, fringilla et nisl et, varius maximus purus. </p>

                    <p class="mg-bot-20">Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Nam arcu
ante, convallis sit amet ex sed, dignissim aliquam
turpis. Quisque sollicitudin malesuada laoreet. Donec
orci orci, sodales vitae leo mattis, gravida finibus
velit. Class aptent taciti sociosqu ad litora torquent
conubia nostra, per inceptos himenaeos. Etiam augue
lectus, fringilla et nisl et, varius maximus purus.

                        <span>Lorem </span>, Lorem
ipsum dolor sit amet, consectetur adipiscing elit. Nam
arcu ante, convallis sit amet ex sed, dignissim
aliquam turpis. Quisque sollicitudin malesuada
laoreet. Donec orci orci, sodales vitae leo mattis,
gravida finibus velit. Class aptent taciti sociosqu ad
litora torquent conubia nostra, per inceptos
himenaeos. </p>
```

```
                </div>
            </div>
        </section>
        <section id="skills">
            <div class="container">
                <div class="skills-content">
                    <h2>Skills</h2>
                    <p class="mg-bot-20">Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Nam arcu
ante, convallis sit amet ex sed, dignissim aliquam
turpis. Quisque sollicitudin malesuada laoreet. Donec
orci orci, sodales vitae leo mattis, gravida finibus
velit. Class aptent taciti sociosqu ad litora torquent
conubia nostra, per inceptos himenaeos. Etiam augue
lectus, fringilla et nisl et, varius maximus purus. </p>
                    <div class="skills-icons">
                        <i class="fab fa-html5"></i>
                        <i class="fab fa-css3"></i>
                        <i class="fab fa-WordPress"></i>
                        <i class="fab fa-js"></i>
                        <i class="fab fa-adobe"></i>
                        <i class="fab fa-PHP"></i>
                        <i class="fas fa-database"></i>
                    </div>
                    <!--<a href="#" class="btn-
primary">View my Work</a>-->
                </div>
                <div class="skills-rating">
                    <p>HTML5 &amp;amp; CSS3</p>
                    <div class="outer-meter">
                        <div class="inner-meter
html5"></div>
                    </div>
                    <p>Photoshop</p>
                    <div class="outer-meter">
                        <div class="inner-meter
photoshop"></div>
                    </div>
                    <p>Javascript &amp;amp; jQuery</p>
                    <div class="outer-meter">
                        <div class="inner-meter
js-jquery"></div>
```

```
                    </div>
                    <p>Wordpress</p>
                    <div class="outer-meter">
                        <div class="inner-meter
WordPress"></div>
                    </div>
                    <p>PHP &amp;amp; MySQL</p>
                    <div class="outer-meter">
                        <div class="inner-meter PHP-
MySQL"></div>
                    </div>
                    <p>ReactJS and Redux (Learning in
progress)</p>
                    <div class="outer-meter">
                        <div class="inner-meter react-
redux"></div>
                    </div>
                </div>
            </div>
        </section>
        <section id="portfolio">
            <div class="container">
                <h2>Recent <span>Projects</span></h2>
                <div class="thumbnails-wrapper">
                    <img src="img/thumbnail-1.jpg"
alt="Garage Door Company sample" class="project-
thumbnails" data-lity-target="img/project1.png">
                    <img src="img/thumbnail-2.jpg"
alt="Automotive template sample" class="project-
thumbnails" data-lity-target="img/project2.png">
                    <img src="img/thumbnail-3.jpg"
alt="Construction sample template" class="project-
thumbnails" data-lity-target="img/project3.png">
                </div>
            </div>
        </section>
        <section id="footer">
            <section id="quote">
                <div class="container">
                    <h3>"H Quisque sollicitudin
malesuada laoreet. Donec orci orci, sodales vitae leo
mattis, gravida finibus velit. Class aptent taciti
```

```
sociosqu ad litora torquent per conubia nostra, per
inceptos himenaeos.."</h3>
                    <p>-Lorem ipsum dolor sit amet</p>
                </div>
            </section>
            <section id="copyright">
                <div class="container">
                    <p>&copy;Lorem ipsum dolor sit
amet </p>
                    <span class="flex-spacer"></span>
<!-- IE WORKAROUND -->
                    <div class="social-icons">
                        <a href="https://www.facebook
.com/armand.abecilla" target="_blank"><i class="fab
fa-facebook-f"></i></a>
                        <a href="https://github.com/
armandAbecilla" target="_blank"><i class="fab
fa-GitHub"></i></a>
                        <a href="https://open.spotify
.com/user/dnamra0216" target="_blank"><i class="fab
fa-Spotify"></i></a>
                        <a href="https://soundcloud
.com/armand-abecilla" target="_blank"><i class="fab
fa-Soundcloud"></i></a>
                    </div>
                </div>
            </section>
        </section>

        <script type="text/javascript" src="js/jquery-
3.3.1.min.js"></script>
        <script type="text/javascript" src="js/jquery
.waypoints.min.js"></script>
        <script type="text/javascript" src="js/wow
.min.js"></script>
        <script src="js/lity.min.js"
charset="utf-8"></script>
        <script type="text/javascript" src="js/main
.js"></script>
    </body>
</html>
```

This is all about HTML files. Our main file is jQuery, where all the toggling and animation are occurring. You will learn each function step by step in the next section. Your jQuery file will be main.js, rest all the files remain as it is. So the first code is related to the skill section, where you will get user skills progress bar with various percentages as you see on other sites. If you don't know, you can see the following figure.



Portfolio – skill section.

Here is the code:

```
$(document).ready(function() {

}
```

Rest all the code will be written inside the { }. This code means that when your page is ready properly then rest of the code will run. As you know that our first code is related to the progress bar.

```
var pbAlreadyLoaded = false;

 $('.skills-rating').waypoint(function(direction){
     if (!pbAlreadyLoaded) {
         progressBar('.inner-meter.html5', 85);
         progressBar('.inner-meter.photoshop', 70);
         progressBar('.inner-meter.js-jquery', 74);
         progressBar('.inner-meter.wordpress', 67);
         progressBar('.inner-meter.php-mysql', 77);
         progressBar('.inner-meter.react-redux', 55);
```

```
            pbAlreadyLoaded = true;
        }
    }, {
        offset : '100%'
    });
```

In the above code, the progress bar loader is set to false when your page is not loaded properly; once it gets loaded, the progress bar is set to true which means the mentioned value will be occupied.

CHAPTER SUMMARY

We covered our first project with a coding explanation of Portfolio. Before starting the project, we covered some basic explanations of HTML and CSS also some features that's why we use it first in any project.

CHAPTER 4

jQuery Project Third

Flip Card

IN THIS CHAPTER

- Introduction
- Requirement
- Code Explanation

In the previous chapter, we covered the second jQuery project. Here in this chapter, we are going to cover third jQuery project with a full explanation of each and every step.

INTRODUCTION

Before starting jQuery coding, you must know the requirements of the jQuery project. You should be aware of HTML, some CSS, and a little bit of JavaScript to know how we link our files together for making projects. Let's talk about HTML first so that you will get some hints about what we are going to do here.

HTML

This section provides an introduction to HTML. We cannot imagine web pages and the World Wide Web without HTML. It is the language mostly used to write web pages. It represents Hyper-Text Mark-up Language. You should know that any link found on web pages is usually called Hypertext,

DOI: 10.1201/9781003358497-4

and the mark refers to the tag or page layout in such a way that the text in the web page list is displayed in the correct format. The purpose of developing HTML was to understand the structure of any text: title, body, content, or categories.

So, basically, HTML provides a structured format to display the content of web pages. It is very simple and easy to understand. In the early 1990s, it was developed by Tim Berners-Lee and later went through many changes and modifications. HTML 5 is the latest version of HTML.

KEY HTML FEATURES

You know that HTML is a simple language that can use many tags to format content. All tags are enclosed within angular brackets <tagname>. With the exception of a few tags, most tags start with angular tags and end with corresponding angular tags.

<!DOCTYPE html> defines the document type and version of HTML. The HTML code starts just after the <html> angular marker and ends with </html>.

It usually has two major parts, namely the head and the body. Each category has its own set of requirements. You will get various sections in HTML structure. Let's discuss each.

Head Section

The header tag represents a web document header that you can tag <head> and <link> with. It starts with <head> and ends with </head>. It has parts of the title inside.

<title> This will be your webpage title. </title>>.

Body Section

It represents the body of a web document that usually contains titles, text, and sections. Topics start with <title> and end with </title>. Among these tags, the content can be labeled as "this is the first topic."

The paragraph will start with and end with . The content of the section should be written between these angular characters. The basic HTML code shown below in the overview section is used to create a simple HTML page.

```
<!DOCTYPE html>
<html lang="en">
<meta charset="utf-8">
```

```
<title> Page Title </title>
<body>
<h1> This is a Heading </h1>
<p> This is a paragraph. </p>

</body>
</html>
```

There are various elements we can add to our HTML code; let's take a look.

- HTML headings are elements defined by tags <h1> to <h6>, where <h1> defines the most important tag and <h6> defines the less important tag. The HTML section is an HTML element that will be defined using the <p> tag.
- HTML images are a feature of HTML and are defined by the <img> tag, and need to specify attributes such as image src, alt means other text, width, and height.
- HTML lists are elements and are defined using <ul> or <ol> tags, where <ul> is a random list and <ol> is an ordered list.
- The HTML table is part of the HTML and can be defined using tag <table> and tag lines <tr> and tag cells <td>.
- HTML links are elements and can be defined using the <a> tag.
- The HTML attribute style can be used with a combination of any HTML element such as <p>.
- In HTML, we use the lang attribute, we can say the language of the document using the <html> tag and the language defined using the lang attribute.
- In HTML, we can use the formatting elements to format the HTML document, and we can define special text features that have a special meaning. For example, HTML elements like <b> to bold and <i> to italics.
- In HTML, we can highlight specific text in a document using the <mark> feature to highlight text included in the <mark> element.

- In HTML, we can define text as the text above using the <sup> element in the HTML document so that the text embedded in the <sup> element becomes larger text.

The <!DOCTYPE> represents the document type and helps browsers to display web pages correctly. It must only appear once, at the top of the page (before any HTML tags). The <!DOCTYPE> declaration is not case sensitive.

Wherever the web is, it is HTML. HTML usage is distributed across all devices. Here is the list of features of HTML to know where we can use it.

- Browsers like Chrome, Firefox, and Safari all use HTML to render content on the web for better display.
- Various mobile browsers like Opera, Firefox Focus, Microsoft Edge, Dolphin, and Puffin all use HTML to better present and visualize online content on mobile phones.
- Various smart devices are embedded with HTML functions to better browse and navigate during their operations.
- HTML supports the first channel verification method on any web page to stop unwanted traffic.
- HTML accepts great content but gives the same visibility to smaller and larger screen devices.
- HTML supports a variety of colors, formats, and layouts.
- HTML uses templates that make website design easy.
- The HTML and XML syntax are very similar, so it is easy to work between the two domains.
- FrontPage, Dreamweaver, and many other development tools support HTML.
- HTML is a very useful search engine.

CSS

Cascading Style Sheets, better known as CSS, is a very simple design process used to make web pages more visible. CSS lets customize your web pages. The best part about using these style features is that CSS is independent of

the HTML way of creating web pages. The difference between the HTML and the CSS is that the first one is best known for providing a country layout on the web page while the other is intended to provide powerful coding and color coding techniques. The CSS is used to control the composition of more than one webpage at a time. All external style sheets are stored in the form of CSS files.

KEY CSS COMPONENTS

We just read the introduction to CSS, now let's move to the main CSS sections.

1. You can intend to make any changes in the world, just change the style, and you can see all the other features on all other web pages are updated automatically.
2. You can simply write the text once and use the same sheet as many times as you like.
3. CSS by comparison has a much wider range of attributes and lists compared to HTML. The HTML page can therefore have a sharper look and feel compared to standard HTML attributes.
4. CSS is considered a style sheet that is easy to use and read. This means that search engines do not have to make much effort in trying to read the text.
5. CSS can be used to store web applications locally with the help of the offline cache method, which can be used to view offline websites.

WHY DO WE USE CSS?

- For a website to work properly, it must have a fast download time. Nowadays, people usually wait a few seconds for a website to load. Therefore, it is important to ensure a fast pace. For companies looking to ensure fast and smooth website information, CSS becomes key to their success.
- CSS is easy to maintain due to the short maintenance time. This is because single-line code conversion affects the entire web page. Also, if upgrades are needed, make a small effort
- You would not find many good and easy-to-use websites. One thing that is common to all of these websites is the consistency of

construction. CSS empowers developers to ensure that style features are applied consistently across every few web pages.

- Due to its fast speed and easy maintenance, CSS saves a lot of time and effort in the web development process due to fast loading time. Here, a little time ensures the good performance of the designer.
- People use different smart devices to view a particular website. It can be a smartphone, PC, or laptop. For this purpose, websites are required to be compatible with the device. CSS ensures smooth operation by providing better alignment.
- You can change the location of the HTML tag with the help of CSS. You can set things as an image on any part of a web page as needed.

Here we going to discuss the HTML page of the project.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta HTTP-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
    <script src="https://cdnjs.cloudflare.com/ajax/
libs/modernizr/2.8.3/modernizr.min.js" type="text/
javascript"></script>
<link href="styles.css" rel="stylesheet">
</head>
<body>
    <head>
        <title>Match Game</title>
        <link rel="shortcut icon" type="image/png"
href="./favicon.png"/>
        <link rel="stylesheet" href="https://cdnjs.
cloudflare.com/ajax/libs/meyer-reset/2.0/reset.min.
css">
        <link href="https://maxcdn.bootstrapcdn.com/
bootstrap/3.3.7/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-BVYiiSIFeK1dGmJRAky
```

```
cuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">
        <link href="https://fonts.googleapis.com/css?f
amily=Work+Sans:400,700,900" rel="stylesheet">
        <link rel="stylesheet" href="./resources/css/
style.css">
      </head>
      <body>
        <div class="container">
          <div class="flip-card">
            <div class=" text-center">
            <!-- Instructions -->
              <h1> jQuery Game - Flip Card </h1>
              <h4> There are following rules</h4>
              <p> Click on a card to reveal the number
on the other side. Click on a second card to try and
find a match to the first. If you succeed, the pair
will be removed from play. If not, then try again!</p>
pairs have been found.</p>
            </div>
            </div>
            <div class="row">
            <div id="game" >
            <!-- Grid appears via JS-->
            </div>
        </div>
        </div>
        <script
          src="https://code.jquery.com/jquery-
3.1.1.min.js"
          integrity="sha256-hVVnYaiADRTO2PzUGmuLJr8BLU
SjGIZsDYGmIJLv2b8="
          crossorigin="anonymous"></script>
      </body>
<script src="script.js"></script>
</body>
</html>
```

If you checked our last two projects then you will get a hint why HTML is an important page of any project. If not, then always keep in mind that the content shown on the browser is the content you have written on your HTML page. All CDN of CSS, Bootstrap, jQuery, and JavaScript are always

written on the HTML head tag. So in the above code, you get various link tags and script tags. The content in the body tags is visible on the browser.

On the HTML page, you get instructions about the Flip Card Game. Every div has an id or class for CSS and trigger in jQuery.

The next file we have is CSS named as styles.css. You can change styling as per your needs. The code is given below:

```
/* Universal Styles */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
  }

  body {
    font-family: 'Work Sans', sans-serif;
    font-size: 1em;
    background-color: RGB(255,242,242);
    padding: 5.25rem;
  }

  /* Instructions */
  .container h1 {
    font-weight: bold;
    font-size: 2.5rem;
    color: rgb(32,64,86);
  }
  .flip-card{
      width: 100%;
      margin: 0 auto;
  }
  .container h4 {
    font-weight: bold;
  }

  .container p {
    color: rgb(74,74,74);
  }

  /* Grid */

  .card {
```

```
    display: flex;
    justify-content: center;
    flex-direction: column;
    align-items: center;
    background-color: RGB(32,64,86);
    color: RGB(255,255,255);
    height: 12.5rem;
    border: 4px solid #fff;
    border-radius: 8px;
    font-weight: 900;
    font-size: 7rem;
  }

  .card:hover {
    cursor: pointer;
  }
```

Now, the next file is the jQuery file.

```
var MatchGame = {};

/*
 Sets up a new game after the HTML document has
loaded.
  Renders a 4x4 board of cards.
*/

$(function() {
  var $game = $('#game');
  var values = MatchGame.generateCardValues();
  MatchGame.renderCards(values, $game);
});

/*
  Generates and returns an array of matching card
values.
 */

MatchGame.generateCardValues = function() {
  var cardArray = [1,1,2,2,3,3,4,4,5,5,6,6,7,7,8,8];
  var cardValues = [];

  /* Move a random card from cardArray to card value
*/
```

```
  while (cardArray.length > 0) {
      var randomIndex = Math.floor(Math.
random()*cardArray.length);
      cardValues.push(cardArray[randomIndex]);
      cardArray.splice(randomIndex,1);
  }

  return cardValues;
};

/*
  Converts card values to jQuery card objects and adds
them to the supplied game
  object.
*/

MatchGame.renderCards = function(cardValues, $game) {
  /* create CSS styles for each number in cardValues
*/
  var cardColors = ["hsl(25,85%,65%)",
"hsl(55,85%,65%)", "hsl(90,85%,65%)",
"hsl(160,85%,65%)", "hsl(220,85%,65%)",
"hsl(265,85%,65%)", "hsl(310,85%,65%)",
"hsl(360,85%,65%)"];

  $game.empty(); /* empty HTML */
  $game.data('flippedCards',[]); /* Track number of
flippedCards */

  /* Loop through cardValues to create card objects */
  cardValues.forEach(function(value) {
    var $card = $('<div class="col-xs-3 card">
</div>');

    $card.data('value', value);
    $card.data('isFlipped', false);
    $card.data('color', cardColors[value - 1]);

    $game.append($card);
  });

/* Event listener for user clicking card */
```

```
  $('.card').click(function() {
    MatchGame.flipCard($(this), $('#game'));
  });
};

/*
  Flips over a given card and checks to see if two
cards are flipped over.
 Updates styles on flipped cards depending on whether
they are a match or not.
 */

MatchGame.flipCard = function($card, $game) {
  /* Prevent flipped card from getting triggered */
  if ($card.data('isFlipped')) {
    return;
  };

  $card.css('background-color', $card.data('color'));
  $card.text($card.data('value'));
  $card.data('isFlipped', true);

  /* Track flipped cards and observe match */
  var flippedCards = $game.data('flippedCards')
  flippedCards.push($card);

  /* Conditionals for styling cards upon match */

  /* Check if there are 2 active flippedCards */
  if (flippedCards.length === 2) {
    /* Check if flippedCards values match each other
*/
    if (flippedCards[0].data('value') ===
flippedCards[1].data('value')) {
      var matched = {
        backgroundColor: 'RGB(153,153,153)',
        color: 'RGB(204,204,204)'
      };

      flippedCards[0].css(matched);
      flippedCards[1].css(matched);

      } else {
```

```
      /* Reset flippedCards to original styling */
        window.setTimeout(function() {
          flippedCards[0].css('background-color',
'rgb(32, 64, 86)')
            .text('')
            .data('isFlipped',false);
          flippedCards[1].css('background-color',
'rgb(32, 64, 86)')
            .text('')
            .data('isFlipped',false);
        }, 350);
      };

    $game.data('flippedCards',[]); /* Reset
flippedCards tracker to 0 */
  };
};
```

jQuery FILE CODE EXPLANATION

When the page loads, you will see some instructions after that you will see various cards. The card number is hidden at first, however when you click on it, the card flips is shown. First, we initialize the variable MatchGame as given below:

```
var MatchGame = {};
```

Then after loading the page, below function runs. In this code, we use id as DOM that means we select id using code:

```
$(function() {
  var $game = $('#game');
  var values = MatchGame.generateCardValues();
  MatchGame.renderCards(values, $game);
});
```

The second line automatically runs the generateCardValues() as given below:

```
MatchGame.generateCardValues = function() {
  var cardArray = [1,1,2,2,3,3,4,4,5,5,6,6,7,7,8,8];
  var cardValues = [];
```

```
  /* Move a random card from cardArray to card value
*/
  while (cardArray.length > 0) {
      var randomIndex = Math.floor(Math.
random()*cardArray.length);
      cardValues.push(cardArray[randomIndex]);
      cardArray.splice(randomIndex,1);
  }

  return cardValues;
};
```

The function is about the various card number range of 16 digits. In the condition, first we check the length of the cardArray if it is greater than 0 then the user will get some cards on the screen.

```
var randomIndex = Math.floor(Math.random()*cardArray.
length);
```

In the above line, the cards will have random values every time when a user loads the page according to the card length. Then the user picks any card and that card value will be stored in another array as given below:

```
var randomIndex = Math.floor(Math.random()*cardArray.
length);
```

There are two extra methods used in the above code, that is, floor and random.

- math.floor(): It is a method that reduces the given number to the nearest integer value and returns it. If the given number is already a whole value, the floor method () returns it directly.
- math.random (): It is a function that returns a floating point, a random number ranges from 0 to less than 1.

```
cardArray.splice(randomIndex,1);
```

The above code runs when another array has some values in it and then it returns that cardValues.

```
  return cardValues;
```

Now the code which you see just above is a code that we see on the browser screen.

```
MatchGame.renderCards = function(cardValues, $game) {
  /* create CSS styles for each number in cardValues
*/
  var cardColors = ["hsl(25,85%,65%)",
"hsl(55,85%,65%)", "hsl(90,85%,65%)",
"hsl(160,85%,65%)", "hsl(220,85%,65%)",
"hsl(265,85%,65%)", "hsl(310,85%,65%)",
"hsl(360,85%,65%)"];

  $game.empty(); /* empty HTML */
  $game.data('flippedCards',[]); /* Track number of
flippedCards */

  /* Loop through cardValues to create card objects */
  cardValues.forEach(function(value) {
    var $card = $('<div class="col-xs-3 card">
</div>');

    $card.data('value', value);
    $card.data('isFlipped', false);
    $card.data('color', cardColors[value - 1]);

    $game.append($card);
  });

/* Event listener for user clicking card */
  $('.card').click(function() {
    MatchGame.flipCard($(this), $('#game'));
  });
};
```

Now create some CSS like coloring, as shown below:

```
var cardColors = ["hsl(25,85%,65%)",
"hsl(55,85%,65%)", "hsl(90,85%,65%)",
"hsl(160,85%,65%)", "hsl(220,85%,65%)",
"hsl(265,85%,65%)", "hsl(310,85%,65%)",
"hsl(360,85%,65%)"];
```

When we create an empty HTML using $game.empty(), we track the flipped card in an empty array. The code is given below:

```
  $game.empty(); /* empty HTML */
  $game.data('flippedCards',[]); /* Track number of
flippedCards */
```

Here, we create some working loop using some HTML tags with Bootstrap.

```
cardValues.forEach(function(value) {
    var $card = $('<div class="col-xs-3 card">
</div>');

    $card.data('value', value);
    $card.data('isFlipped', false);
    $card.data('color', cardColors[value - 1]);

    $game.append($card);
  });
```

Now, the clickable events work is given below:

```
  $('.card').click(function() {
    MatchGame.flipCard($(this), $('#game'));
  });
};
```

The above flipCard is a function that we call next:

```
MatchGame.flipCard = function($card, $game) {
  /* Prevent flipped card from getting triggered */
  if ($card.data('isFlipped')) {
    return;
  };

  $card.css('background-color', $card.data('color'));
  $card.text($card.data('value'));
  $card.data('isFlipped', true);
```

```
  /* Track flipped cards and observe match */
  var flippedCards = $game.data('flippedCards')
  flippedCards.push($card);

  /* Conditionals for styling cards upon match */

  /* Check if there are 2 active flippedCards */
  if (flippedCards.length === 2) {
    /* Check if flippedCards values match each other
*/
    if (flippedCards[0].data('value') ===
flippedCards[1].data('value')) {
      var matched = {
        backgroundColor: 'RGB(153,153,153)',
        color: 'RGB(204,204,204)'
      };

      flippedCards[0].css(matched);
      flippedCards[1].css(matched);

      } else {
      /* Reset flippedCards to original styling */
        window.setTimeout(function() {
          flippedCards[0].css('background-color',
' rgb(32, 64, 86)' )
            .text('')
            .data('isFlipped',false);
          flippedCards[1].css('background-color',
' rgb(32, 64, 86)' )
            .text('')
            .data('isFlipped ',false);
        }, 350);
      };

    $game.data('flippedCards',[]); /* Reset
flippedCards tracker to 0 */
  };
};
```

All the main work starts from here when the user hits the Card and the number will be shown. If the number is the same then the card color will be changed to gray. We cannot use that number again. If your number will

not match, then you can have an option to select another number. After matching the number the isFlipped will be false and no changes will be done.

CHAPTER SUMMARY

We covered our third project with a coding explanation of Flip Card. Before starting it, we covered some basic explanations of HTML and CSS and its features to use it in any project.

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

CHAPTER 5

Code Optimization

IN THIS CHAPTER

- Introduction
- Types of Code Enhancement
- Code Optimization
- The jQuery Security Model

In Chapter 4, we covered our last jQuery project with a code explanation. Here in this chapter, we are going to cover the code optimization process in jQuery.

In this chapter, we will discuss some of these common jQuery processes and code development techniques that improve the efficiency and effectiveness of our code. Most of the strategies outlined below are accurate once we get to the root of the problem.

INTRODUCTION

Code optimization, as the name suggests, can be defined as a method used to modify codes to improve the efficiency and quality of code. As a result of preparation, the system may be smaller in size, consume less memory, use it more quickly, or perform fewer tasks (input/output). A prepared plan should have the same effects and negative consequences as that of its unprepared plan.

DOI: 10.1201/9781003358497-5

Improvement may also be referred to as a program modification, performed by developers or program planners. Optimizer is a special software or built-in compiler unit and is also called a development compiler. Modern processors are also used to improve the orderliness of code commands. Code refinement is usually done at the end of the development phase as it reduces readability and adds code to increase performance.

IMPORTANT TYPES OF CODE ENHANCEMENT

1. Higher level preparation, medium level preparation, and lower level development: Advanced editing is a form of language-based development that works at a level close to the source code. Advanced configurations include installing inlining when the work call is replaced by a task theme and partial testing that uses loop rearrangement, matching alignment, alignment, structure, and tail removal. Most practical coding development falls under the medium-sized code development of language. These include:

    - Completion of common small expressions – It is a form of investigation to develop a combination of similar speech expressions by measuring the same number and then evaluating whether it is necessary to change it with a single variable that holds the computer value.
    - Regular distribution – Here, terms that can be analyzed during compilation are identified and replaced by their values.
    - Jump threading – It includes fine jumping practice. The second condition is terminated if the opposite or the first subset can be easily made with one step in the system. Acyclic chained escapes are followed until the coordinator arrives at the fixed location.
    - Fixed loop code movement – It is also known as Loop scale or scale increase. The invariant loop contains expressions that can be taken out of the loop body without affecting the semantics of the system. This movement is done automatically with the movement of the fixed loop code.
    - Dead code termination – Here codes that do not affect system results are deleted. It has many advantages that include system size reduction and operating time. It also simplifies system configuration. It is also known as DCE, dead code deletion, dead code removal, or dead code line.

- Reduction of power – This compiler setting replaces expensive functions with equally efficient but less expensive.

2. Low level optimization is very specific to the type of buildings. These include the following:
   - Register allocation – Here, a large number of variables for targeted programs are allocated to a small number of CPU registers. This can be done with a local register allocation or an international register allocation or a process register allocation.
   - Instruction planning – It is used to improve the standard of instruction, which also improves the performance of instructional equipment. It will not change the meaning of the code but redesign the order of the instructions to avoid piping stables. Mathematical ambiguous activities are also avoided.
   - Use of floating point units – These units are specially designed to perform floating point number operations such as addition, subtraction, etc. The features of these units are used in low-level developments that are very specific to the type of buildings.
   - Branch forecasting – This method helps to predict how a branch works even though it is not known for sure which will be of great help in improving results.
   - Peephole and profile-based optimization – Peephole optimization is done over small code sections at a time to change them by inserting short or quick instructions. This set is called the peephole.
   - Profile-based improvements are made to a complex combination of predicting the possible outcome of branches, the size of the same members, or the most widely used loops. They provide non-existent information, which allows editors to decide when needed.
3. Machine-based development and machine-based efficiency: The automated development section is trying to improve the code in the middle to get the best output. Improved centralized code does not include any complete memory locations or CPU registers.

Machine-based development is done after the production of a target code that is modified according to the design of the target machine. This includes CPU registers and may have complete memory indicators.

In conclusion, good code preparation is required as it provides a clean code base, high consistency, faster sites, better readability, and much more.

CODE OPTIMIZATION AND ORGANIZATION IN jQuery

jQuery is a very popular private library, in fact, almost every website uses it. It is popular because it is easy to read, speeds up development, provides clean code, and helps to run complex operations easily. It can slow down the performance of websites and can lead to slow loading, if not used properly. To sure the performance of jQuery code, one must know and follow the best practices. In this section, we list some of the most important tips for improving jQuery code performance.

1. **Upload jQuery from CDN (Use the latest version of jQuery, minified version, and from CDN):** A Content Delivery Network (CDN) is a system of distributed servers that deliver web pages and other web content to a user based on the user location, the origin of the web page, and the content delivery server. Uploading jQuery from a CDN is a good choice, as it reduces the loading time of your site (CDN speeds up the library), and reduces the load on your web server. Caching is another benefit of CDN. A cached version of a website is used when a user re-visits the same website. It will never be downloaded again. CDN Google is the preferred CDN for jQuery:

```
<script src = "https://ajax.googleapis.com/ajax/
libs/jquery/3.1.1/jquery.min.js> </script>
```

You can also use the jQuery CDN:

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

Note: There is no need for a protocol when referring to a jQuery library. The "protocol-less" URL is the best way to refer to third-party content found on both HTTP and HTTPS. When a protocol is omitted, the browser uses the default document protocol instead. If not, you will end up getting a mixed content warning.

The examples of alternatives that come out in stages are given below:

- .live () method removed from version 1.7.
- .load () method used to bind the event host to the "load" of JavaScript event, extracted from version 1.8.
- .attr () method no longer works in testing and removal of marking checkboxes. This happens in the version of jQuery 1.6. You should now use the .prop () checkbox or uncheck.
- unbind () method used to remove event handlers from selected elements and deprecated in jQuery version 3.0.

2. **Don't wait for the $ (document) .ready () function:** The function $(document).ready() is the most common way to store all your jQuery code. However, in the event of a large or heavy page with a lot of text and complex images, the correct document event may be delayed using this method. Most jQuery code does not need to wait for any of these items. $ (Document) .ready () is especially important when the code meets the DOM objects inserted after the script is released. If the code does not require the interaction of the DOM element, it can be removed from $ (document) .ready (). Consider the following example:

```
$ (document) .on ('click', '#btnClick', function
() {
    console.log ('Click on an event !!!');
});
$ (document) .ready (function () {});
```

Here, the click event is attached to the button by event posting, even before the "correct document" is called. This practice can improve the performance of your website.

3. **Customize your jQuery selector:** The following is one of the ways to improve the performance of jQuery while using selectors:

- Use ID as a selector – If possible, always use a feature ID as your selector instead of the HTML tag or CSS class. Regarding the ID selector, jQuery internally makes a call to the getElementById ()

Java script method that puts the map directly into the feature as given below:

```
$ (document) .ready (function () {
$ ("# elm") .css ("color", "red");
});
```

4. **Do not specifically use CSS class selector:** CSS class selectors are used to select elements with a specific CSS class. There are cases where you want to interact with many DOM objects, and in these cases the ID selector cannot be used. Working with the CSS class selector can be used by reducing the DOM interval time. For example, the following jQuery code selects all DOM elements with the CSS class name "dummy" and turns the background color yellow.

```
$ (document) .ready (function () {
    $ (". dummy") .css ("background color",
"yellow");
});
```

5. **Save your jQuery selector:** Saving time can greatly improve jQuery performance. One has to be very careful about the options for maintaining the archive properly as it can result in significant improvements. For example, the following jQuery code uses a new color (green), changes the font ("Tahoma"), and changes the element text ("sample text").

   The problem with the jQuery code is that it has to break DOM three times, once for each separate item (color, font, and text). Breaking the DOM is an expensive task. Another way to reduce the number of interruptions is to save the selector. For example:

```
var $ elm = $ ("# element"). CSS ("color", "yellow");
$ elm.css ("font size", "25px");
$ elm.text ("You are learning to use the code to
the fullest");
```

6. **Use jQuery chaining:** It is one of the most powerful features of jQuery. Chaining means connecting jQuery options or functions, respectively. For example, the code we used to display the cache of the jQuery repository can also be upgraded with integration.

```
$ ("# element ") .css ({" color ": " green ",
"font-family": "Tahoma"}). text (" You are
learning code optimization ");
```

In this case too, jQuery will only run once to get the feature. Chaining can be used for methods or functions. For example:

```
$ ("# btn"). click (function (event) {
    console.log ("click!");
}). mouse (function (e) {
    console.log (" Mouse over!")
}). mouseout (function (s) {
    console.log (" Move out the mouse!")
});
```

7. **Upload jQuery not from the title page but from the end of the body:** All developers usually add a jQuery link to the title of the page as given below:

```
<head>
<script src = "https://ajax.googleapis.com/ajax/
libs/jquery/3.2.1/jquery.min.js"> </script>
</head>
```

But we recommend that you place it before the end of the body tag.

```
<body>
…….
…….
<script src = "https://ajax.googleapis.com/ajax/
libs/jquery/3.2.1/jquery.min.js"> </script>

<script>
$ (document) .ready (function () {
console.log ("OK!");
});
</script>
</body>
```

The problem caused by the script is that it blocks the corresponding download. Browsers download more than two components in accordance with each hostname. So when you put text in the <head> tag, the browsers go to it, thus keeping some of the content waiting for the text to load, which results in slow page loading.

8. **Store result of selector inside a variable:** You should never select elements again and again. Instead, you should cache it in a variable.

jQuery doesn't have to track down element over and over again. It never selects element every time in the loop. The below code is a bad approach in terms of performance.

//Bad approach

```
for (var i = 0;  i < 100 ; i++) {
$('#elements').append(i);
}
```

Now, change the above code to store the selected item in a variable then use that variable inside the loop:

//Good approach

```
var item = $('#myItem');
for (var i = 0 ; i < 100; i++) {
item.append( i );
}
```

9. **Use jQuery On method to attach multiple event handlers:** The On method is useful when doing coding in jQuery. If you are attaching multiple events to an element then the code will become long.

   Let's take for example of these events such as — mouseover and mouseout.

   The code becomes:

```
$(" p ").mouseover(function(){
$(this).addClass(" box ");
});

$(" p ").mouseout(function(){
$(this).removeClass(" box ");
});
```

   Now, we can combine these two events together with. on() method.

```
$("p").on(" mouseover mouseout ", function(){
$(this).toggleClass(" box ");
});
```

10. **Always use the latest version:** jQuery is in development and continuous development. John and his team are constantly exploring new ways to improve program performance.

    If you want to stay up-to-date without having to download the library a thousand times, GIYF (Google Is Your Friend), even in this situation. Google offers a wide range of AJAX libraries to choose from Google's AJAX Libraries CDN.

11. **Use for instead of each:** The functions are always faster than the helper counterparts. Whenever you are looping through an object received as JSON, you would better rewrite your JSON and make it return an array through which you can loop easier.

12. **Cache. ALWAYS:** You should not make the mistake of reusing selectors time and time again. Instead, you should cache it in a variable. In this way, the DOM doesn't have to track down your element over and over again.

    Add this line of code at the end of the file:

```
console.timeEnd('cache');
```

13. **Return false:** You might have noticed when functions don't return false, you jump to the top of the page.

    When dealing with longer pages, this result can be quite annoying.

```
$(' #element ').click (function () {
    // code here
});
```

THE jQuery SECURITY MODEL

jQuery is a JavaScript UI framework that provides a rough background for many DOM deceptive tasks. It provides developers with a friendly interface so they can quickly update the DOM without reloading the entire page. The simple concept paved the way for a new web application development model and for many JavaScript frameworks. The jQuery security effects are not at all appealing, but they are often poorly understood and can have serious consequences. In many cases, we see this cost valuable time and money to large organizations. The vulnerability of jQuery is often raised by an accidental explosion and is a common source of disagreement between entry inspectors and engineers. It is good to note that almost all of the jQuery security issues surround activities that were so often misused that the jQuery team changed behavior to protect engineers. Although the changes are widely interpreted as bug fixes, it can easily be argued that the risk presented by jQuery is negligible without the error of the developer. We hope this article can be used by an organization to better assess the risks of common jQuery security issues.

The $() Function

$() is the same, and the most common written method for the jQuery () function, returns the jQuery object: actually part of the content to be written in DOM. In most cases, the jQuery function will take a selector, object, or object as a parameter. The selector, defined as hash (#), is the identifier for HTML content present in the current DOM.

jQuery "**XSS** Vulnerability"

If you came to this page today because of an accident called "jQuery XSS Vulnerability" suggested in the pentest report, you are not alone. At the time of this writing, there are no known XSS-specific threats in the jQuery framework (excluding jQuery plugins). Unfortunately, it is very common for behavioral changes to be interpreted as bug fixes. Most experienced jQuery developers are well aware of the dangers of introducing unreliable content into the jQuery object. Like other DOM modifiers (innerHTML, document.write (), etc.), the jQuery function should be used with proper care. It is no less dangerous than the traditional JavaScript functions that we call killer sinks. Let us take a look at the behavioral changes that have caused many headaches.

AJAX for jQuery $ .get ()

The jQuery ajax $ .get () function (not to be confused with the .get ()) function) is used for making ajax GET applications, as you might think. It has been found that in pre-1.12.0 versions, it will automatically check the response content, which has the ability to sign the script if it was content in the reply. Unlike the problem-solving problem described above, we believe that this behavior will be considered risky and potentially unexpected even for experienced developers. An important trace of that statement is the circumstances in which the matter may appear to be much less than the previous issue. These behaviors may help to reduce the vulnerability of the App:

- Apps that apply to unreliable domains may use text that can be viewed as secure content.
- Reliable API end point requests may be used for XSS attacks if the script is not embedded in data sources.

CHAPTER SUMMARY

Effective and organized coding improves the overall functionality of the application. All of the above tips can improve the performance of your jQuery code, especially by reducing the DOM horizontal time to create jQuery code. DOM traversal is a very expensive process and jQuery selectors play a major role in it. jQuery is awesome but without the advanced code it can lead to poor performance of the App, so be sure to use these tips.

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

Appraisal

jQuery is important in the world of web development. jQuery functions as library of JavaScript. It strives to simplify the lives of web developers by performing a wide range of tasks easier. jQuery started as a library to integrate browser collisions, while still providing generality for most browsers. As four browsers upgraded and packed with matching holes, jQuery becomes softer, more efficient, and better to accomplish the task of providing an API that simplifies JavaScript development.

jQuery has the ability to reduce multiple JavaScript lines into just a few lines, and, in most cases, just one JavaScript-enabled line.

The jQuery UI library also provides the ability to drag anything with the mouse cursor by clicking, holding, and moving the mouse, you can move objects around the page to any of your favorite position. It also makes easy to create a user interface for drag and drop. You can use jQuery UI for creating scrollable locations, where you take features from other parts of the page and release them to another, in the same way you can interact with the file manager of your operating system when you want to move a folder from one place to another. You can also create an orderly list using drag and drop, rearrangement of elements based on where you throw them. You can have a user interface where you drag the mouse cursor to select more than one file or folder, as you do in file manager of your operating system. The jQuery UI exposes the ability to resize features on the page using the mouse. All the things you can think of to do on your computer desktop, or within the App, you can do in a web browser with jQuery UI.

The jQuery UI also provides a day input widget using a beautiful, accessible JavaScriptdriven interface that opens when you click on the input field. You can customize pop-up dialogs such as pop-up windows, without opening a separate browser window that they display using Core, CSS, and JavaScript.

Another widget provided by the jQuery UI is a slide bar with images, similar to your media player.

DOI: 10.1201/9781003358497-6

As jQuery performs in JavaScript applications, in general, jQuery UI strives to perform unnecessary graphical user interface (GUI) tasks. The jQuery UI gives you the ability to customize the interface of professional interface with very little development effort. Before learning jQuery, you must have the basic knowledge of JavaScript.

WHAT CAN YOU DO WITH jQuery?

There are so many things you can do with jQuery. These are as follows:

- You can easily select elements to create a trick.
- You can easily create an effect as a show or hide elements, slide transitions, and more.
- You can easily create complex CSS animation with a few lines of code.
- You can easily manipulate DOM features and their attributes.
- You can easily use AJAX to enable incompatible data exchanges between client and server.
- You can easily cut around the DOM tree to find anything.
- You can easily perform multiple actions on an object with one line of code.
- You can easily find or set the size of HTML objects.

ADVANTAGES OF jQuery

There are a few benefits to why one should choose jQuery. These are as follows:

- Save a lot of time – You can save a lot of time and effort by using the built-in jQuery effects and selectors focusing on other development work.
- Simplify common JavaScript tasks – jQuery makes JavaScript tasks much easier. Now you can easily create a feature of rich and interactive web pages with a few lines of code, a common example using AJAX to update page content.
- Easy to use – jQuery is very easy to use. Anyone with a basic knowledge of HTML, CSS, and JavaScript can start to improve with jQuery.

- Compatible with browsers – jQuery is created with modern browsers and is compatible with all major modern browsers such as Chrome, Firefox, Safari, Internet Explorer, etc.
- It's free – And the best part is, it's completely free to download and use.
- Plugins – Several available JavaScript plugins can increase functionality that can be added according to one's requirements.

DISADVANTAGES OF jQuery

One of the worst things about jQuery is the large number of versions published in a short period of time. Whether you use the latest version of jQuery, you will need to download the library yourself (and update it regularly) or download the Google library. jQuery is easy to install and read, but it is not so easy when compared to CSS.

If jQuery is misused like Frame, the development site can get out of control.

PREREQUISITES FOR MAKING jQuery PROJECT

- HTML – It stands for Hypertext Markup Language. HTML is a standard markup language that is used for creating web pages to display in a web browser. Web browsers usually receive HTML documents from the server and then render them in the correct format from the client. Knowledge of HTML is a must for learning Bootstrap.
- CSS – It stands for Cascading Style Sheets. CSS is a style sheet language used to describe the presentation and styling of an HTML document. It was developed by World Wide Web Consortium (W3C) in 1996. The knowledge of CSS is essential for learning Bootstrap because to use any component of Bootstrap, you have to use the corresponding utility classes.
- JavaScript – It is considered the programming language of the web. Combined with HTML and CSS, these three languages are considered the first stepping stones for web development. Even though knowledge of JavaScript is not essential, it would prove to be useful as there are lots of JavaScript plugins in Bootstrap. You can also add your custom functionality by using Bootstrap.

REASON WHY CHOOSE jQuery

jQuery provides the standard syntax for selecting DOM features on a page. Instead of establishing its own rules of selection, it uses what the engineers are already familiar with – "Css Pickers." CSS options are more powerful than the syntax provided by native JavaScript DOM options.

jQuery has a feature where all its DOM functions return a changed feature. It allows us to bring voters and methods together in an easy-to-understand way. In jQuery, the DOM functions are written to handle most DOM features

As we have seen many times above, jQuery actually reduces the amount of code written by developers. When we remove the boilerplate, we are left only working on the specifics of the project.

If you ask anyone what they like about jQuery, the answer you will most likely find is the number of useful plugins provided by jQuery. It is not possible to list all useful plugins in the jQuery ecosystem. Since jQuery has been around for years, developers have had time to build and maintain it for a long time.

Bibliography

1. jQuery Introduction – https://www.geeksforgeeks.org/jquery-introduction, accessed on October 7, 2022.
2. jQuery Introduction – https://www.w3schools.com/jquery/jquery_intro.asp, accessed on October 7, 2022.
3. jQuery – https://www.javatpoint.com/what-is-jquery, accessed on October 7, 2022.
4. jQuery – https://en.wikipedia.org/wiki/JQuery, accessed on October 7, 2022.
5. About jQuery – https://www.codecademy.com/learn/learn-jquery/modules/learn-jquery-introduction, accessed on October 7, 2022.
6. jQuery – https://www.tutorialrepublic.com/jquery-tutorial/, accessed on October 8, 2022.
7. Library Vs Framework – https://www.interviewbit.com/blog/framework-vs-library/#:~:text=Libraries%20provide%20developers%20with%20predefined,build%20applications%20for%20specific%20platforms, accessed on October 8, 2022.
8. Framework and Library Difference – https://www.freecodecamp.org/news/the-difference-between-a-framework-and-a-library-bd133054023f/, accessed on October 8, 2022.
9. https://en.wikipedia.org/wiki/JQuery#:~:text=as%20a%20plugin.-,History,being%20led%20by%20Richard%20Gibson), accessed on October 8, 2022.
10. jQuery History – https://www.javatpoint.com/jquery-history, accessed on October 9, 2022.
11. Framework Vs Library – https://www.geeksforgeeks.org/software-framework-vs-library/, accessed on October 9, 2022.
12. jQuery History – https://blog.logrocket.com/the-history-and-legacy-of-jquery/, accessed on October 9, 2022.
13. jQuery History – https://www.w3schools.blog/jquery-history, accessed on October 10, 2022.
14. Version Release of jQuery – https://releases.jquery.com/jquery/, accessed on October 10, 2022.
15. History and Legacy of jquery – https://blog.logrocket.com/the-history-and-legacy-of-jquery/, accessed on October 10, 2022.
16. jQuery Syntax – https://www.w3schools.com/jquery/jquery_syntax.asp, accessed on October 10, 2022.

17. Use of jQuery in Application – https://www.geeksforgeeks.org/why-we-use-jquery-in-our-web-application/, accessed on October 10, 2022.
18. jQuery – https://hackr.io/blog/what-is-jquery, accessed on October 11, 2022.
19. jQuery vs JavaScript – https://www.geeksforgeeks.org/difference-between-javascript-and-jquery/#:~:text=JavaScript%20is%20a%20programming%20language,Application%20Programming%20Interface%20(API).&text=JavaScript%20boosts%20up%20the%20execution,for%20creating%20catchy%20web%20pages, accessed on October 11, 2022.
20. JavaScript vs jQuery – https://www.edureka.co/blog/javascript-vs-jquery/, accessed on October 11, 2022.
21. JavaScript vs jQuery – https://www.javatpoint.com/jquery-vs-javascript, accessed on October 11, 2022.
22. jQuery Installation – https://nishitvmaheta.medium.com/how-to-install - jquery-ea182b21ce5, accessed on October 11, 2022.
23. jQuery Core – https://learn.jquery.com/using-jquery-core/document-ready, accessed on October 11, 2022.
24. jQuery Events – https://www.w3schools.com/jquery/event_ready.asp, accessed on October 11, 2022.
25. jQuery Events – https://www.w3schools.com/jquery/jquery_events.asp, accessed on October 11, 2022.
26. jQuery DOM Manipulation – https://www.tutorialsteacher.com/jquery/jquery-dom-manipulation, accessed on October 11, 2022.
27. jQuery Event – https://www.w3schools.com/jquery/jquery_events.asp, accessed on October 11, 2022.
28. jQuery Event – https://www.tutorialsteacher.com/jquery/jquery-event, accessed on October 11, 2022.
29. jQuery Event – https://api.jquery.com/category/events/, accessed on October 11, 2022.
30. DOM Elements – https://www.tutorialsteacher.com/jquery/jquery-manipulate-attributes-of-dom-elements#:~:text=jQuery%20attribute%20methods%20allows%20you,text()%2C%20val()%20etc, accessed on October 12, 2022.
31. jQuery Attribute Manipluation – https://www.dotnettricks.com/learn/jquery/attribute-manipulation-example, accessed on October 12, 2022.
32. jQuery Manipulating – https://www.elated.com/jquery-manipulating-element-attributes/, accessed on October 12, 2022.
33. jQuery AJAX – https://api.jquery.com/jquery.ajax, accessed on October 12, 2022.
34. jQuery AJAX Method – https://www.javatpoint.com/jquery-ajax-method, accessed on October 12, 2022.
35. jQuery AJAX Method – https://www.tutorialsteacher.com/jquery/jquery-ajax-method, accessed on October 13, 2022.
36. jQuery References AJAX – https://www.w3schools.com/jquery/jquery_ref_ajax.asp, accessed on October 13, 2022.
37. jQuery AJAX – https://www.tutorialspoint.com/jquery/jquery-ajax.htm, accessed on October 13, 2022.

38. jQuery Benefits – https://www.sitepoint.com/benefits-jquery/, accessed on October 13, 2022.
39. jQuery Benefits – https://www.beacontechnologies.com/blog/2011/05/the-benefits-of-using-jquery.aspx, accessed on October 13, 2022.
40. Advanatges and Disadvantages – https://www.techquintal.com/advantages-and-disadvantages-of-jquery/, accessed on October 14, 2022.
41. HTML Introduction – https://www.w3schools.com/html/html_intro.asp, accessed on October 14, 2022.
42. HTML – https://en.wikipedia.org/wiki/HTML, accessed on October 14, 2022.
43. CSS – https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS, accessed on October 14, 2022.
44. Use of CSS – https://www.javatpoint.com/what-are-the-uses-of-css, accessed on October 15, 2022.
45. Use of CSS – https://devmountain.com/blog/what-is-css-and-why-use-it/, accessed on October 15, 2022.
46. CSS Used for – https://www.codecademy.com/resources/blog/what-is-css-used-for/, accessed on October 15, 2022.
47. jQuery Optimize – https://learningjquery.com/2017/03/how-to-optimize-jquery-code-for-better-performance, accessed on October 15, 2022.
48. jQuery Optimization Techniques – https://medium.com/codeodin/jquery-optimization-techniques-which-you-cant-miss-as-a-developer-b3ab17dbe7e9, accessed on October 15, 2022.
49. Code Optimization – https://www.innofied.com/code-optimization-organization-javascript-jquery, accessed on October 16, 2022.
50. jQuery Alternatives – https://www.spec-india.com/blog/jquery-alternatives, accessed on October 17, 2022.
51. jQuery Selector – https://learn.jquery.com/performance/optimize-selectors, accessed on October 18, 2022.
52. jQuery Performance Optimization – https://blog.dareboost.com/en/2014/04/jquery-performance-optimization, accessed on October 18, 2022.
53. JavaScript Optimization Tips – https://www.upwork.com/resources/javascript-optimization-tips, accessed on October 18, 2022.
54. https://web.eecs.umich.edu/~weimerw/2012-4610/scottcd/16a_cdsec.pdf
55. Code Optimization Organization JavaScript jQuery – https://www.innofied.com/code-optimization-organization-javascript-jquery, accessed on October 19, 2022.
56. Code Organization – https://learn.jquery.com/code-organization/concepts, accessed on October 19, 2022.
57. https://www.unleashed-technologies.com/blog/quick-tips-help-you-optimize-jquery-code-pro, accessed on October 19, 2022.
58. jQuery Alternatives – https://www.educba.com/jquery-alternatives/, accessed on October 19, 2022.
59. Alternatives to jQuery – https://www.techolac.com/alternatives-to-jquery, accessed on October 19, 2022.

60. Other UI jQuery Alternatives – https://www.g2.com/products/jquery-ui/competitors/alternatives, accessed on October 20, 2022.
61. jQuery – https://en.wikipedia.org/wiki/JQuery, accessed on October 20, 2022.
62. https://www.technotification.com/2019/06/5-lightweight-jquery-alternatives-2019.html, accessed on October 20, 2022.
63. Cash – https://github.com/fabiospampinato/cash#fnfilter, accessed on October 21, 2022.
64. Umberllajs – https://umbrellajs.com, accessed on October 21, 2022.
65. Chibi – https://github.com/kylebarrow/chibi, accessed on October 21, 2022.
66. Zeptojs – https://zeptojs.com, accessed on October 21, 2022.
67. Mootools – https://mootools.net, accessed on October 22, 2022.
68. Ext_JS – https://en.wikipedia.org/wiki/Ext_JS, accessed on October 22, 2022.
69. essential-js2 – https://www.syncfusion.com/products/whatsnew/essential-js2, accessed on October 22, 2022.

Index

C

D

E

F

T

U

W

X

Y