CHAPTER 1

Web Development

The big picture of web* basics for the development of AI applications that can run on the user's device/gadget is shown in Fig1.1

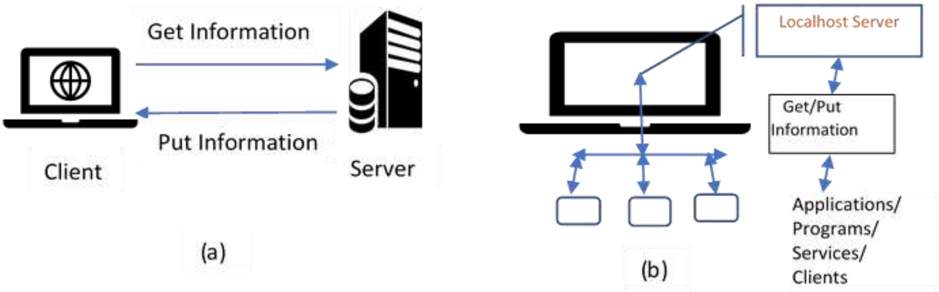


Fig1.1 Web Communication through Internet and localhost

The three web development essentials are:

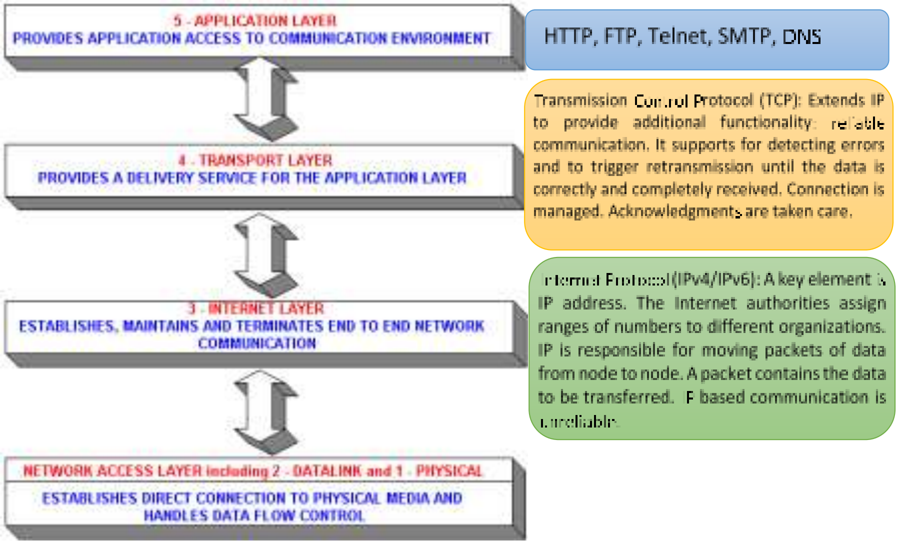
i) Client: Web browsers, used to surf the web

ii) Server systems: used to supply information to the browsers

iii) Computer Networks: used to support browser-server communication

**Web is a collection of interconnected documents and resources*.

The functioning of the web activity as shown in Fig1.1(a) indicate the Internetworking principle, where the communication between client and server is done through protocols such as Internet Protocol (IP), Transmission Control Protocol (TCP), HyperText Transfer Protocol (HTTP), and File Transfer Protocol (FTP). Fig1.1(b) depicts the communication between the client (browser/services/applications), and the server (localhost) exists locally to provide the required information to the respective applications (client/browser/services). The conceptual theme of Fig1.1(a) is based on the TCP/IP protocol suite, as shown in Fig1.2.

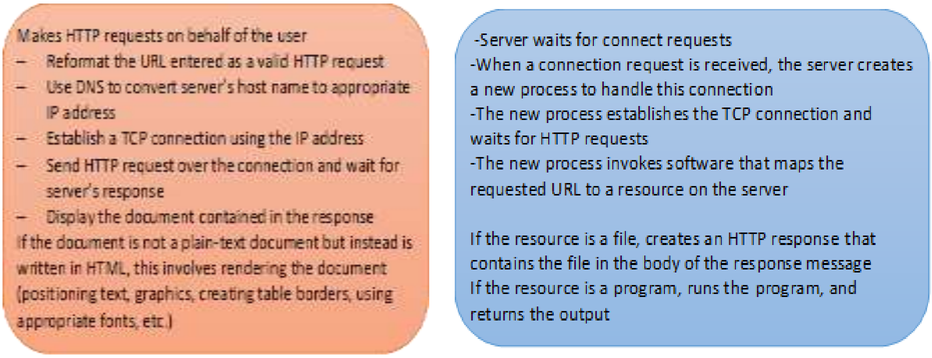

5 - APPLICATION LAYER
PROVIDES APPLICATION ACCESS TO COMMUNICATION ENVIRONMENT
4 - TRANSPORT LAYER
PROVIDES A DELIVERY SERVICE FOR THE APPLICATION LAYER
3 - INTERNET LAYER
ESTABLISHES, MAINTAINS AND TERMINATES END TO END NETWORK COMMUNICATION
NETWORK ACCESS LAYER including 2 - DATALINK and 1 - PHYSICAL
ESTABLISHES DIRECT CONNECTION TO PHYSICAL MEDIA AND HANDLES DATA FLOW CONTROL
HTTP, FTP, Telnet, SMTP, DNS

Transmission Control Protocol (TCP): Extends IP to provide additional functionality: reliable communication. It supports for detecting errors and to trigger retransmission until the data is correctly and completely received. Connection is managed. Acknowledgments are taken care.

Internet Protocol (IPv4/IPv6): A key element is IP address. The Internet authorities assign ranges of numbers to different organizations. IP is responsible for moving packets of data from node to node. A packet contains the data to be transferred. IP based communication is unreliable.

Fig1.2 TCP/IP protocol suite

The World Wide Web (WWW): Is a system of interlinked, hypertext documents that runs over the Internet. Two types of software exist i) Client: A system that wishes to access the information provided by servers must run client software (e.g., Web browser), and ii) Server: an (internet) connected computer that wishes to provide information to others must run server software. The client and server applications communicate over the Internet by following a protocol built on TCP/IP-i.e HTTP. Fig.1.3 depicts the overall communication



Makes HTTP requests on behalf of the user

- Reformat the URL entered as a valid HTTP request
- Use DNS to convert server's host name to appropriate IP address
- Establish a TCP connection using the IP address
- Send HTTP request over the connection and wait for server's response
- Display the document contained in the response

If the document is not a plain-text document but instead is written in HTML, this involves rendering the document (positioning text, graphics, creating table borders, using appropriate fonts, etc.)

-Server waits for connect requests
-When a connection request is received, the server creates a new process to handle this connection
-The new process establishes the TCP connection and waits for HTTP requests
-The new process invokes software that maps the requested URL to a resource on the server

If the resource is a file, creates an HTTP response that contains the file in the body of the response message
If the resource is a program, runs the program, and returns the output

Fig1.3 The communication between web client (browser) and the webserver

Hypertext: A format of information that allows one to move from one part of a document to another or one document to another through hyperlinks.
Uniform Resource Locator (URL): Unique identifiers used to locate a particular resource on the network.
Markup Language: Defines the structure and content of hypertext documents.

Organizing the web with HTML

To design and develop web pages, practice through Hypertext Markup Language (HTML) programming is preferred- the crucial innovation used to characterize the structure of a website page. HTML is utilized to determine whether web substance ought to be perceived as a section, list, heading, connect, picture, mixed media player, structure, or one of the numerous other accessible components or even another component that can be characterized.

What is HTML?

HTML is not a programming language. It is a markup language that advises Internet browsers on how to structure the site pages that a user visits. It is very well designed/convoluted by the web designer as the requirements it to be. HTML comprises of a progression of components, which you use to encase, wrap, or increase various pieces of substance to cause it to show up or act with a specific goal in mind. The encasing labels can cause content into a hyperlink to associate with another page and to emphasize words.

CHAPTER

Web Development using Integrated Development Environment (IDE)/Editors

The difference between IDE and editors (text) for web development is that IDE does all the things from fundamental content managers to advanced development capabilities that cannot be done with text editors. For example, the editors such as ***Sublime***, ***Notepad++,*** or ***Atom*** can be utilized for HTML, and Cascading Style Sheet (CSS), composing the code for designing the web pages. They accompany a lot of advantageous highlights; for example, language structure featuring, adaptable interfaces, and broad navigation tools that a web developer require extra features to make a useful application. For instance, a web developer may require a debugger and a compiler for developing web applications effectively.

Nonetheless, with the best IDEs, you would not be stressed over that. They regularly accompany extra features for automation, testing, and envisioning the advancement procedure. Mainly, they provide support to the web developer for transforming code into a working application.

Most commonly used IDE for Web development

i) *Visual Studio Code:* Visual Studio Code is conceivably the best JavaScript IDE for Windows, Mac, and Linux platforms. In addition to the fact that it supports JavaScript functionality, yet it likewise supports Node.js and TypeScript features, and it accompanies a system of extensions for different programming dialects, including C++, C#, Python, and PHP. Visual Studio Code gives incredible syntactical featuring and auto-complete with IntelliSense dependent on factor types, word definitions, and imported modules for programmer-friendly operations.

ii) *NetBeans:* NetBeans is high on the rundown for the best web advancement IDE because it lets you create a neat work area, versatile, and web applications in a matter of moments. It works similarly as great with programming codes for JavaScript, HTML5, PHP, and C/C++. It is a free JavaScript IDE and an incredible HTML5 IDE for your everyday use. This web advancement IDE accompanies correct code dissecting and altering devices good with the most recent Java 8 innovations.

iii) *PyCharm:* PyCharm is not the best free JavaScript IDE; the paid Professional Edition is unquestionably worth looking at on the off chance that you are searching for a solid web advancement IDE for Python developers.

iv) IntelliJ IDEA: is an excellent web improvement IDE. There is a free network form; however, if you need to exploit all the Java Script improvement apparatuses it brings to the table, you ought to consider looking at the paid-for Ultimate Edition. Time and energy can be minimized in the web development process while using the IntelliJ Idea. IntelliJ IDEA is an excellent CSS IDE. However, it additionally bolsters a wide assortment of programming dialects.

Note: In this book, the programming snippets (codes) are illustrated with the Notepad++, Sublime, and Atom text editors IDEs. The Fig.1.4 shows the web programming environments of the three editors.

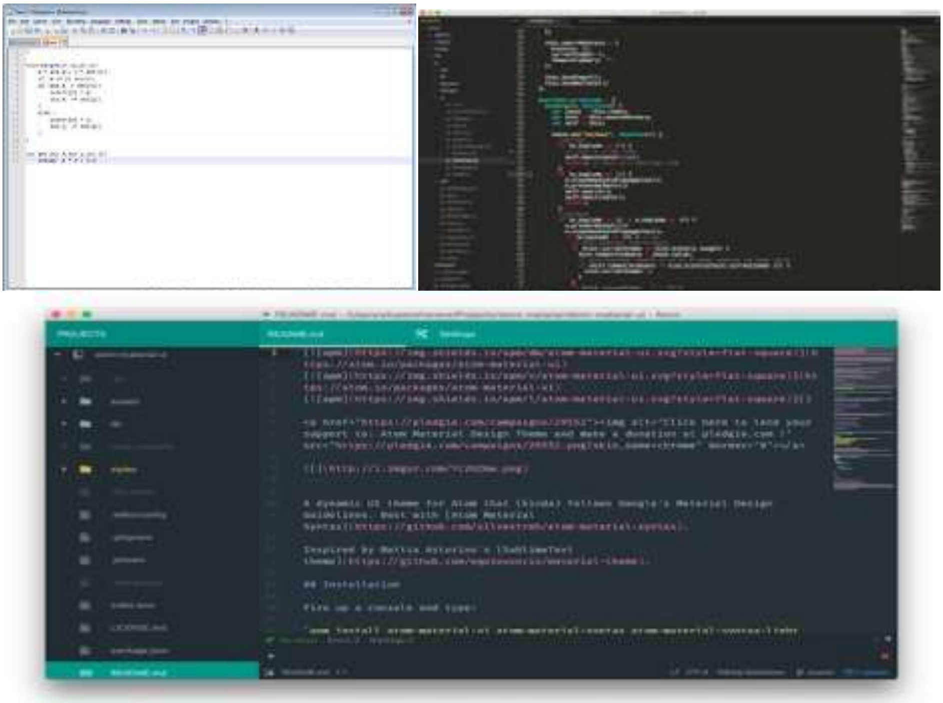


Fig.1.4 Notepad++, Sublime, and Atom editors environment

Building blocks of web development:

There are three building blocks of web development.

i) ***HTML:*** Using HTML 5, Computers can now understand what is on your website rather than just displaying the contents of the website dynamically. Earlier versions of HTML are used to provide static or dynamic information to the users. The content of the website is shown from HTML code.

ii)Cascading Style Sheet (CSS): CSS is meant for how the website/pages look alike. HTML is for making the content while CSS is for showing the content in the way we like it using different styles, colors, backgrounds, and layout (design of the web page). CSS makes a website/webpage look interesting.
iii)Java Script (JS): Is a programming language used to manipulate HTML and CSS. The main functionality is to provide interactive features for the user. It is sturdy and can be used for creating full web applications (apps).

Resources: For developing and testing the web applications/code snippets given in this book, it is recommended to consider Google's Chrome browser, which has versions for Mac, Windows, and Linux.

HTML and CSS Programming:

The foremost step in the web development process is to define what you want to say (HTML) before defining how you want to say it (CSS). The following code shows the structure and some of the HTML elements.

A HTML component(element) is the combination of start tag, its characteristics (attributes), an end tag and everything in the middle. A HTML tag either opening(< >) or closing (</ >) is utilized to check the beginning or end of a component(element).

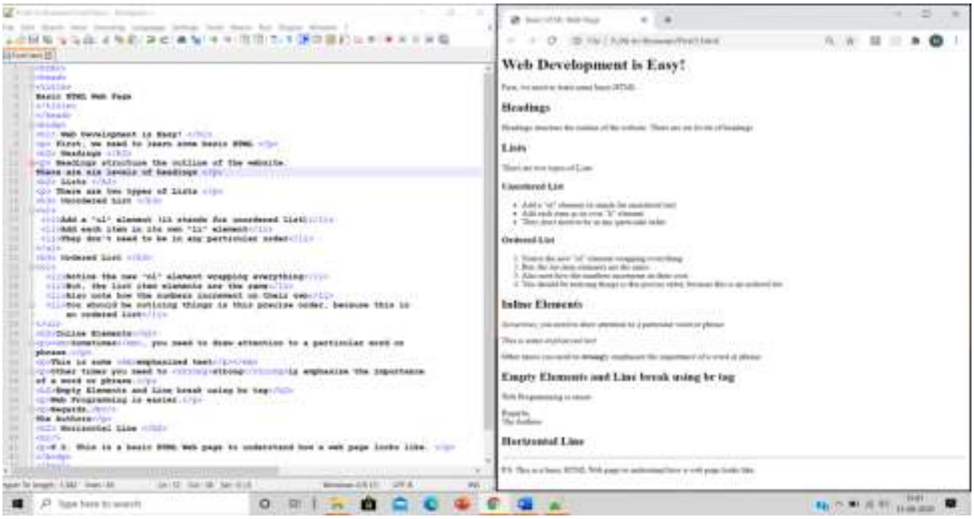
Structure of an HTML document

```
<!DOCTYPE html>
<html>
  <head>
    <!-- Metadata goes here -->
  </head>
  <body>
    <!-- Content goes here -->
  </body>
</html>
```

The first line <!DOCTYPE html> is to inform the browser that it is an HTML5 version web page. The entire web page contents are to be wrapped in <html> tags. The actual <html> text is called an "opening tag", while </html> is called a "closing tag". Everything inside of these tags are considered part of the <html> "element", which is this actual thing that gets created when a web browser parses your HTML tags. “Inside of the <html> element, we have two more elements called <head> and <body>. A web page’s head contains all of its metadata, like the page title, any CSS stylesheets, and other things that are required to render the page, but you do not necessarily want the user to see. The bulk of our HTML markup will live in the <body> element, which represents the visible content of the page. Comments are given in between the tags `<!-- and ends with -->”.` Fig1.5 shows the corresponding output.

Basic html contents of a Web Page

```
<html>
<head>
<title>
Basic HTML Web Page
</title>
</head>
<body>
<h1> Web Development is Easy! </h1>
<p> First, we need to learn some basic HTML </p>
<h2> Headings </h2>
<p> Headings structure the outline of the website.
There are six levels of headings </p>
<h2> Lists </h2>
<p> There are two types of Lists </p>
<h3> Unordered List </h3>
<ul>
 <li>Add a "ul" element (it stands for unordered
list)</li>
 <li>Add each item in its own "li" element</li>
 <li>They don't need to be in any particular
order</li>
</ul>
<h3> Ordered List </h3>
<ol>
  <li>Notice the new "ol" element wrapping
everything</li>
  <li>But, the list item elements are the same</li>
  <li>Also note how the numbers increment on their
own</li>
  <li>You should be noticing things is this precise
order, because this is an ordered list</li>
</ol>
<h2>Inline Elements</h2>
<p><em>Sometimes</em>, you need to draw attention to
a particular word or
phrase.</p>
<p>This is some <em>emphasized text</p></em>
<p>Other times you need to <strong>strong</strong>ly
emphasize the importance of a word or phrase.</p>
<h2>Empty Elements and Line break using br tag</h2>
<p>Web Programming is easier.</p>
<p>Regards,<br/>
The Authors</p>
<h2> Horizontal Line </h2>
<hr/>
<p>P.S. This is a basic HTML Web page to understand
how a web page looks like. </p>
</body>
</html>
```



Web Development is Easy!

Headings

Lists

Unordered List

Ordered List

Inline Elements

Empty Elements and Line break using br tag

Horizontal Line

Fig1.5 Screen print of the basic html code and its Output.

Dynamic HTML: DHTML is used by the web developer to control how to display and position HTML elements in a browser window.

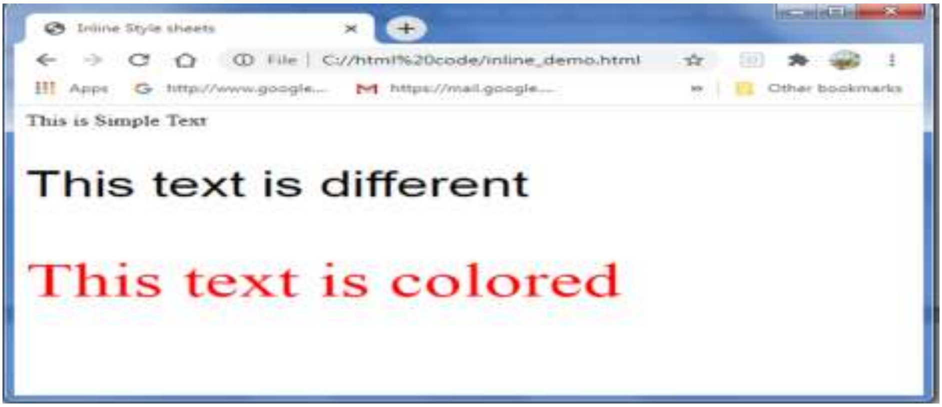
- HTML is used to create static web pages and DHTML is used to create dynamic web pages.
- HTML consists of simple HTML tags, on the other hand DHTML is made of HTML tags+ Cascading Style Sheets+ Javascripts.

Cascading Style Sheets (CSS): Cascading Style Sheets helps us to specify presentation of elements on the webpage. Hence, using CSS we can determine the style and layout of the web page. There are three ways of using style sheets:

1.Inline Style Sheets
2.Embedded Style Sheets
3.External Style Sheets

Inline Style Sheets: An Inline CSS is used to apply various unique style to a single element. It is also used to define a style for a special type of elements, add a class attribute to the element. The following code is to show how inline styles can be used along with the HTML elements. Fig1.6 depicts the corresponding output.

```
<HTML>
<head>
<TITLE> Inline Style sheets </TITLE>
</head>
<Body>
<p> This is Simple Text </p>
<p Style="font-size:30pt; font-family:arial"> This
text is different </p>
<p style="font-size:40pt;color:#ff0000"> This text is
colored </p>
</Body>
</HTML>
```



This is Simple Text

This text is different

This text is colored

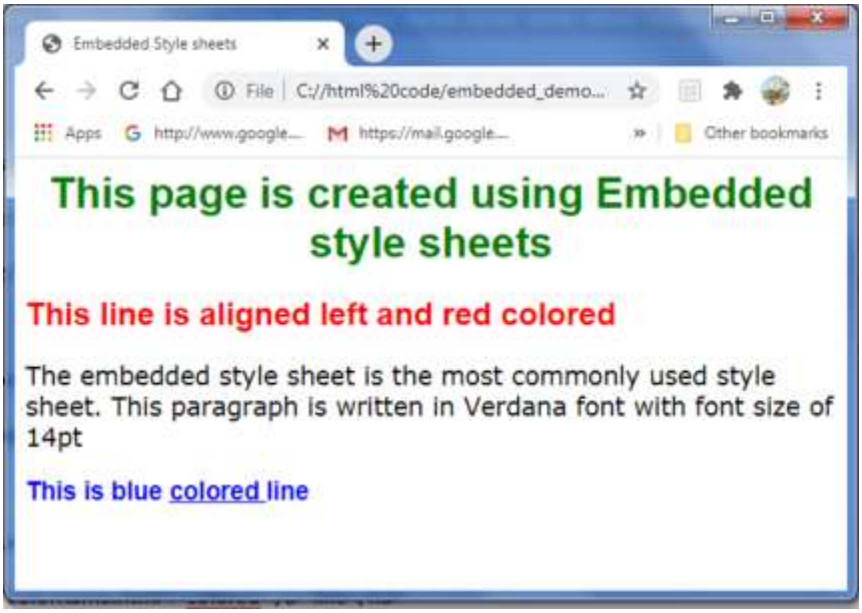
Fig1.6 Output of the inline CSS style settings

Embedded Style Sheets: For embedded style sheet, we write all desired selectors along with the properties and values in the head section. And in the body section then newly defined selector tags are used with the actual contents. In the following DHTML script we have defined h1, h2, h3 and p selectors with different properties and values. Fig1.7 shows the corresponding output of the embedded style settings along with the HTML code.

Note: to define Embedded style sheets we have to mention **style type= “test/css”** in head section.

```
<HTML>
<head>
<TITLE> Embedded Style sheets </TITLE>
<style type="text/css">
h1,h2,h3{font-family:arial;}
h2 {color:red;left:20px }
h3 {color:blue;}
p {font-size:14pt;font-family:verdana;}
.special {color:green}
</style>
</head>
<Body>
<h1 class="special"> <center>
This page is created using Embedded style sheets
</center> </h1>
```

```
<h2> This line is aligned left and red colored </h2>
<p> The embedded style sheet is the most commonly
used style sheet. This paragraph is written in
Verdana font with font size of 14pt</p>
<h3> This is blue
<a href="colorname.html"> colored </a> line</h3>
</Body>
</HTML>
```



This page is created using Embedded style sheets

This line is aligned left and red colored

The embedded style sheet is the most commonly used style sheet. This paragraph is written in Verdana font with font size of 14pt

This is blue colored line

Fig1.7-Output of the embedded style settings

External Style sheets: Sometimes, we need to apply the particular style to more than one web page in such cases external style sheets can be used. The central idea in this type of style sheet is that it is stored in one .css file, and the name of that file has to be mentioned in our web pages, then the styles defined in .css are applied to these web pages. Here is the simple program in which external style sheets are used:

```
<HTML>
<head>
<TITLE> External Style sheets </TITLE>
```

```
<link rel="stylesheet" type="txt/css" href=C:\html
code\ex1.css"/>
</head>
<Body>
<h1 class="special"> <center>
This page is created using External style sheets
</center> </h1>
<h2> This line is aligned left and red colored </h2>
<p> The embedded style sheet is the most commonly
used style sheet. This paragraph is written in
Verdana font with font size of 14pt</p>
<h3>This is blue
<a href="colorname.html"> colored </a> line</h3>
</Body>
</HTML>
```

Create a file with name ex1.css

```
h1 {font-family:arial;}
h2 {
font-family:times new roman;
color:red;
left:20px;
}
h3 {
font-family:arial;
color:blue
}
p {
font-size:14pt;
font-family:cambria;
}
special {color:green }
```

Basics of Java Script (JS)

What is JavaScript?

- It is a scripting language (a lightweight programming language)
- It is an Interpreted language (it executes without preliminary compilation)
- Usually embedded directly into HTML pages
- It is designed to add interactivity to HTML pages
- Java and JavaScript are different.

What can a JavaScript do?

- JavaScript gives HTML designers a programming tool
- JavaScript can put dynamic text into an HTML page
- JavaScript can react to events
- JavaScript can read and write HTML elements
- JavaScript can be used to validate data
- JavaScript can be used to apply AI,ML and DL techniques in the browser
- JavaScript can be used to create cookies Store and retrieve information on the visitor’s computer

JavaScript How To:

The HTML `<script>` tag is used to insert a JavaScript into an HTML page:

```
<script type=“text/javascript”>
     document.write(“Hello World!”);
</script>
```

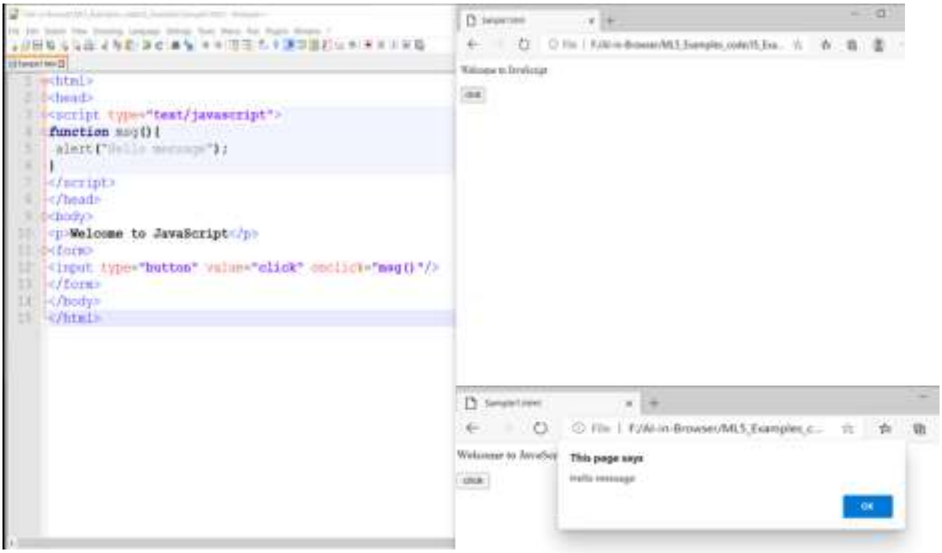
The <script> tag is used to embed a client-side script. The <script> element either contains scripting statements, or it points to an external script file through the “src” attribute .

JavaScript Where To:

- You can include JavaScripts in head, body, or simply use external JavaScript file (.js).
- JavaScripts in the body section will be executed while the page loads.
- JavaScripts in the head section will be executed when the head section is invoked.

Example Java Script inside the head section and Fig1.8 shows the corresponding output

```
<html>
<head>
<script type="text/javascript">
function msg(){
 alert("Hello message");
}
</script>
</head>
<body>
<p>Welcome to JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
```



```
<html>
<head>
<script type="text/javascript">
function msg(){
 alert("Hello message");
}
</script>
</head>
<body>
<p>Welcome to JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
</body>
</html>
```

Welcome to JavaScript

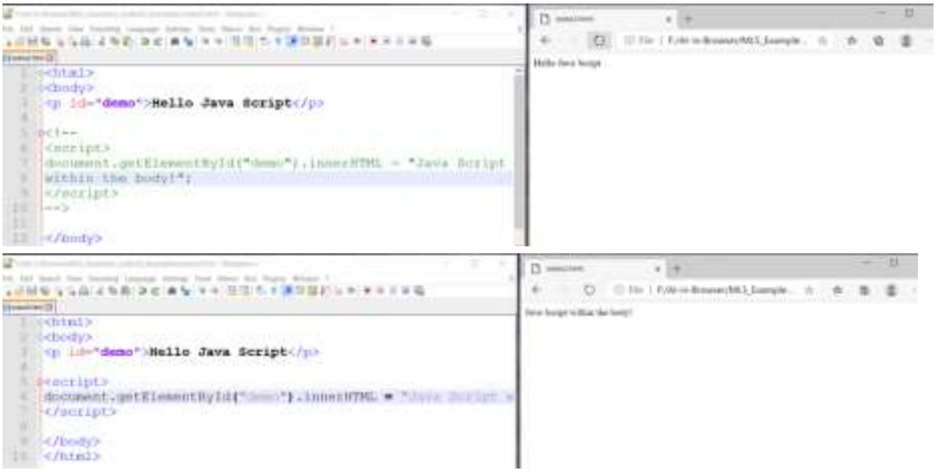
This page says

Hello message

Fig1.8.Example Java Script inside the head section

Example Java Script inside the body section and Fig1.9 shows the corresponding output

```
<html>
<body>
<p id="demo">Hello Java Script</p>
<script>
document.getElementById("demo").innerHTML = "Java
Script within the body!";
</script>
</body>
</html>
```



```
<html>
<body>
<p id="demo">Hello Java Script</p>

<!--
<script>
document.getElementById("demo").innerHTML = "Java Script
within the body!";
</script>
-->

</body>
```

```
Hello Java Script
```

```
<html>
<body>
<p id="demo">Hello Java Script</p>

<script>
document.getElementById("demo").innerHTML = "Java Script w
</script>

</body>
</html>
```

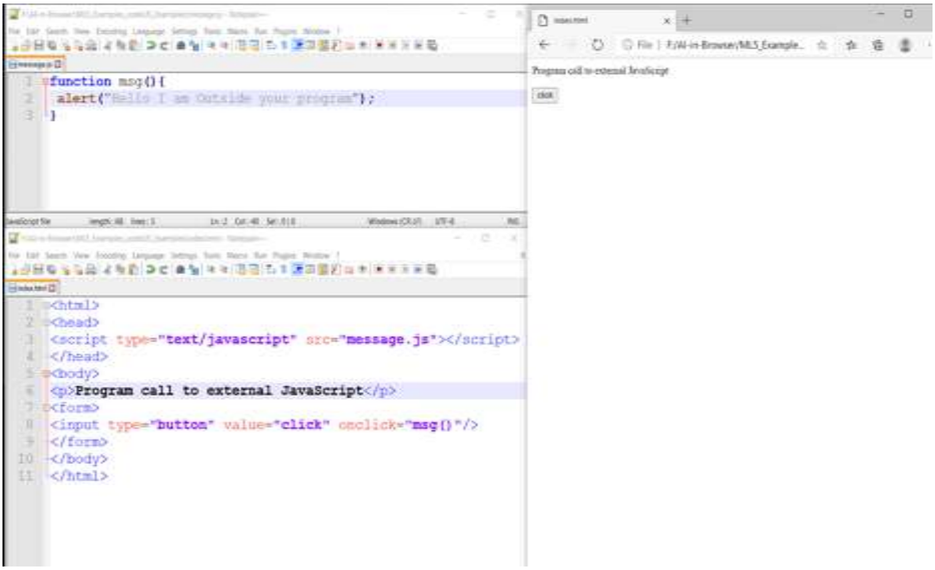
```
Java Script within the body!
```

Fig1.9 Example Java Script inside the body section

Example: Java Script external to the program and Fig1.10 shows the corresponding output

```
message.js
function msg(){
 alert("Hello I am Outside your program");
}
```

```
Index.html
<html>
<head>
<script type="text/javascript"
src="message.js"></script>
</head>
<body>
<p>Program call to external JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
</body>
</html>
```



```
function msg(){
 alert("Hello I am Outside your program");
}
```

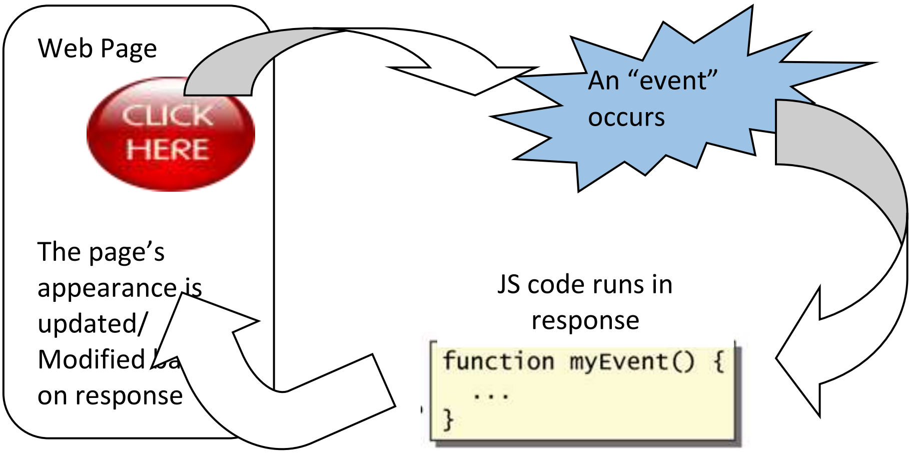
```
<html>
<head>
<script type="text/javascript" src="message.js"></script>
</head>
<body>
<p>Program call to external JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
</body>
</html>
```

Program call to external JavaScript

Fig1.10 Example-Java Script external to the program

Java Script for Event Driven Process:

Event driven programming with JavaScript is a valuable method to make dynamic websites. Commonly, after the website page has stacked, the JavaScript program keeps on running for an event to occur. In the event that you interface, a JavaScript will execute the corresponding code and the behavior of the page changes based on the events. Fig1.11 shows the pictorial representation of a typical event driven process with the help of js features.



```
function myEvent() {
   ...
}
```

Fig1.11: Event driven process in web page with the help of js code

Document Object Model (DOM) Manipulation

At the point when a website page is stacked, the browser (program) creates a Document Object Model of the page. The HTML DOM model is developed as a “tree of Objects”. The difference between HTML and DOM is that, HTML is the thing that returns from the worker in the appropriate response demand, gets parsed and the DOM is the thing that the program works with it and can be controlled with JavaScript. This has a colossal effect in the event that you miss a component from a website page.

The HTML DOM is an Object Model for HTML. It characterizes:

- HTML components as items(elements)
- Properties(attributes) for all HTML components
- Techniques for all HTML components
- Events for all HTML components

The HTML DOM is an API (Programming Interface) for JavaScript:

- JavaScript can include/change/delete HTML components
- JavaScript can include/change/delete HTML traits
- JavaScript can include/change/delete CSS styles
- JavaScript can respond to HTML events
- JavaScript can include/change/delete HTML events

When composing site pages and applications, one of the most widely recognized things you'll need to do is control the archive structure here and there. This is typically done by utilizing the DOM, a lot of APIs for controlling HTML and styling data that utilizes the Document object.

A basic model page at dom-example.html is available at ***(see it live too***). Have a go at opening this up in your program — it is a straightforward page containing a <section> component inside which you can discover a picture, and a passage with a connection inside.

Discovering HTML Elements: At the point when you need to get to HTML components with JavaScript, you need to discover the components first. There are a few different ways to do this:

- Discovering HTML elements by id
- Discovering HTML elements by label name
- Discovering HTML elements by class name
- Discovering HTML elements by CSS selectors
- Discovering HTML elements by HTML object assortments

The following HTML source code resembles the above mentioned features:

```
<!DOCTYPE html>
<html>
<body>
<h2>Finding HTML Elements by Id</h2>
<p id="intro">Hello World!</p>
<p>This example demonstrates the <b>getElementsById</b> method.</p>
<p id="demo1"></p>
<p id=”demo2”></p>
<p class="intro">The DOM is very useful.</p>
<p class="intro">This example demonstrates the
<b>getElementsByClassName</b> method.</p>
<script>
var myElement = document.getElementById("intro");
document.getElementById("demo1").innerHTML =
"The text from the intro paragraph is " + myElement.innerHTML;
var x = document.getElementsByTagName("p");
document.getElementById("demo2").innerHTML =
'The text in first paragraph (index 0) is: ' + x[0].innerHTML;
var x1 = document.getElementsByClassName("intro");
document.getElementById("demo").innerHTML =
'The first paragraph (index 0) with class="intro": ' + x1[0].innerHTML;
</script>
</body>
</html>
```

jQuery

jQuery is a “light-weight javascript library” which signifies compose less, accomplish more. jQuery library allows the web programmers to do following tasks easily:

- HTML and DOM elements manipulation.
- CSS management and control.
- Provides event driven techniques to trigger and react to an event on a web page such as mouse click, button click, keypress and so on.
- Improves the functionality of Asynchronous Java Script and XML (AJAX) calls for the exchange of information.

There are two ways to use the jQuery library in your program:

i) Download the jQuery library (*https://code.jquery.com/jquery-3.5.1.min.js*)from jQuery.com and store it in the same folder as that of the source code(html and css code) location and use it as shown below:

```
<head>
<script src="jquery-3.5.1.min.js"></script>
</head>
```

or

ii)If you are connected to Internet then the following link from Google CDN can be used for jQuery features:

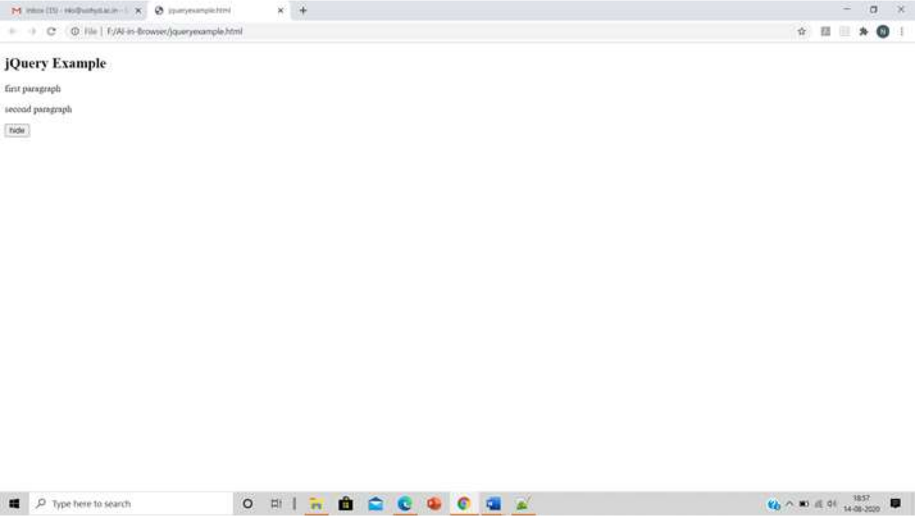
```
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
</head>
```

The following example shows how to use the jQuery library and its functionality:

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>
```

```
<body>
<h2>jQuery Example</h2>
<p>first paragraph</p>
<p>second paragraph</p>
<button>hide</button>
</body>
</html>
```

The output for the above script is as follows before the button hide is clicked:



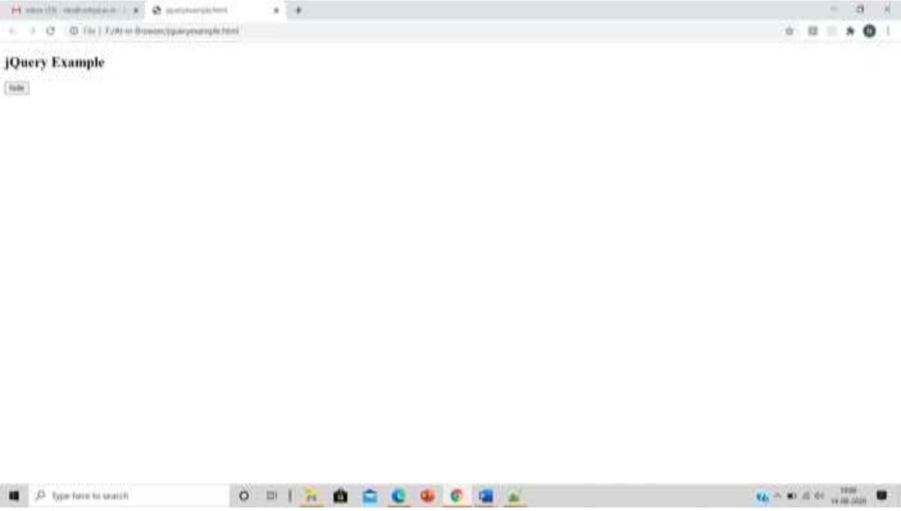
jQuery Example

first paragraph

second paragraph

hide

After the hide button is clicked:



jQuery Example

hide

Fig.1.12 jQuery library and its functionality

Programming through Java Script Libraries for manipulating HTML elements:

p5*JS: A p5.js library is a JavaScript library for imaginative coding, with an emphasis on making coding available and comprehensive for designers, developers, teachers, learners, and any other person!. p5.js is free and open-source programming, the tools to learn it are available to everybody. p5.js is an understanding of ***Processing*** for the present web.

A p5.js library can be any JavaScript code that stretches out or adds to the existing java script code. There are two classifications of libraries, ***Core libraries and Contributed Libraries***. Utilizing the illustration of a programmer java script code, p5.js has a full arrangement of drawing utilities that supports HTML5 objects for text, input, video, webcam, and sound.

To include a p5.js library in the program, link it into your HTML file, after you have linked in p5.js. An example HTML file with p5.js library linked look like the following code snippet:

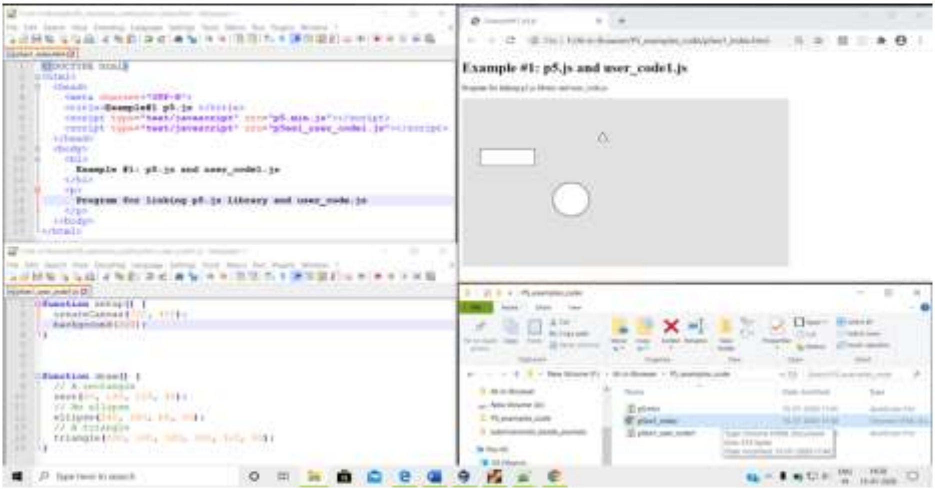
```
<!doctype html>
<html>
<head>
<script src="p5.js"></script>
<script src="user_code.js"></script>
</head>
<body>
</body>
</html>
```

Note: *The p5.js (Single file: Full uncompressed version) or p5.min.js (Single file: Compressed version) library can be downloaded from this link and make sure the p5.js/p5.min.js library and the html file are in the same folder (computer system path).*

Requirements for development and executing programs involving p5.js libraries: Editor: Notepad++ and Web browser: Chrome or Firefox

Example : Inclusion of p5.js library along with the user code:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example#1 p5.js </title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex1_user_code1.js"></script>
  </head>
  <body>
    <h1> Example #1: p5.js and user_code1.js </h1>
    <p>Program for linking p5.js library and user_code.js </p>
  </body>
</html>

function setup() {
  createCanvas(720, 400);
  background(200);
}
function draw() {
  // A rectangle
  rect(40, 120, 120, 40);
  // An ellipse
  ellipse(240, 240, 80, 80);
  // A triangle
  triangle(300, 100, 320, 100, 310, 80);
}
```



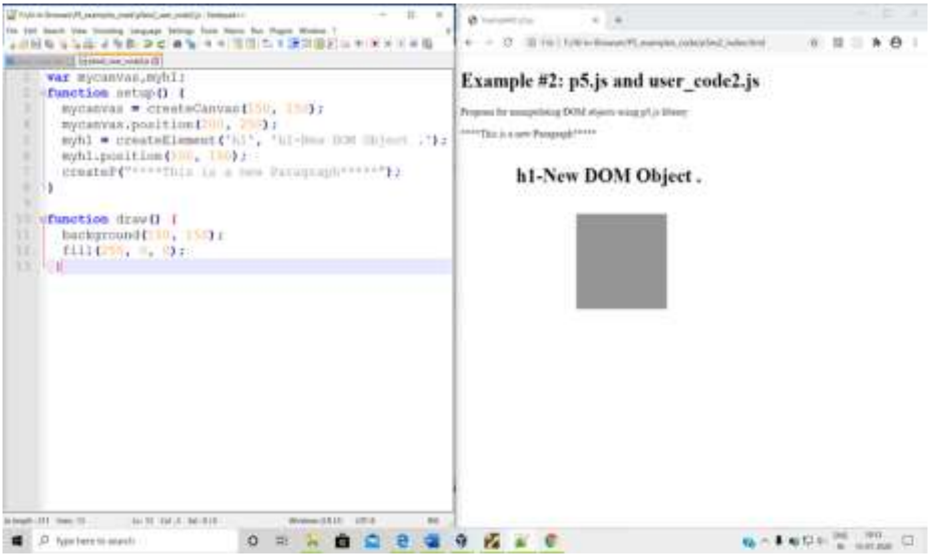
Example #1: p5.js and user_code1.js

Fig1.13 Inclusion of p5.min.js library with the user java script code

Example Creating DOM objects using JS libraries(p5.js)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example#2 p5.js </title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex2_user_code2.js"></script>
  </head>
  <body>
    <h1> Example #2: p5.js and user_code2.js </h1>
    <p>Program for manipulating DOM objects using p5.js library </p>
  </body>
</html>
```

```
var mycanvas,myh1;
function setup() {
  mycanvas = createCanvas(150, 150);
  mycanvas.position(200, 250);
  myh1 = createElement('h1', 'h1-New DOM Object .');
  myh1.position(100, 150);
  createP("****This is a new Paragraph*****");
}
function draw() {
  background(150, 150);
  fill(255, 0, 0);
 }
```



```
var mycanvas,myh1;
function setup() {
  mycanvas = createCanvas(150, 150);
  mycanvas.position(200, 250);
  myh1 = createElement('h1', 'h1-New DOM Object .');
  myh1.position(100, 150);
  createP("****This is a new Paragraph*****");
}

function draw() {
  background(150, 150);
  fill(255, 0, 0);
}
```

Example #2: p5.js and user_code2.js

Program for manipulating DOM objects using p5.js library

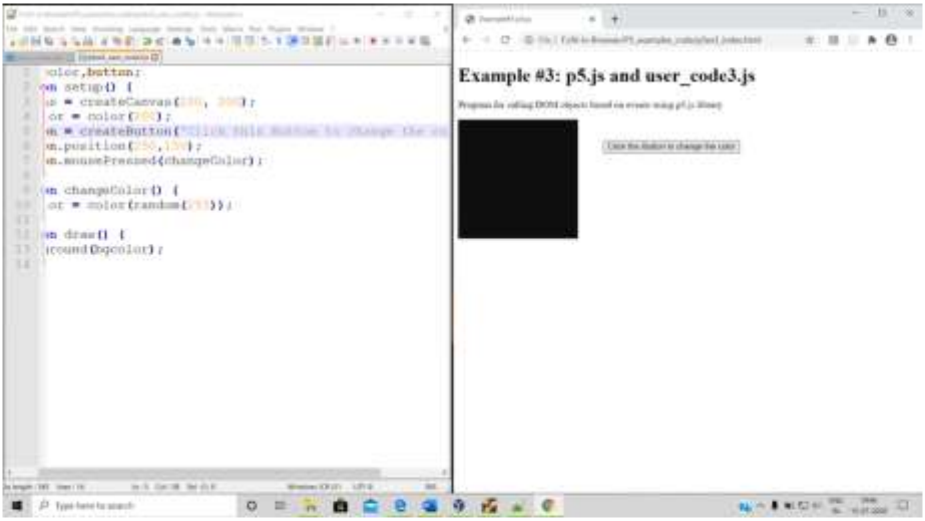
****This is a new Paragraph*****

h1-New DOM Object .

Fig1.14 Creating DOM objects using JS libraries(p5.js)

Example Program for calling DOM objects based on events using p5.js library

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example#3 p5.js </title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex3_user_code3.js"></script>
  </head>
  <body>
    <h1>Example #3: p5.js and user_code3.js </h1>
    <p>Program for calling DOM objects based on events using p5.js
library </p>
  </body>
</html>
var bgcolor,button;
function setup() {
  canvas = createCanvas(200, 200);
  bgcolor = color(200);
  button = createButton('Click this Button to change the color');
  button.position(250,150);
  button.mouseПressed(changeColor);
}
function changeColor() {
  bgcolor = color(random(255));
}
function draw() {
  background(bgcolor);
 }
```



Example #3: p5.js and user_code3.js

Program for calling DOM objects based on events using p5.js library

Fig1.15 calling DOM objects based on events using p5.js library

Example for interacting various HTML elements using p5.js library

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example#4 p5.js </title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex4_user_code4.js"></script>
  </head>
  <body>
    <h1> Example #4: p5.js and user_code4.js </h1>
    <p> Program for interacting various HTML elements using p5.js library
    </p>
  </body>
</html>
var mybgcolor,mybutton,myslider1,myinput1,myname;
function setup() {
  mycanvas = createCanvas(200, 200);
  mybgcolor = color(200);
  myname = createP('Your name!');
  mybutton = createButton('Click to resize the circle');
  mybutton.mousePressed(changeColor);
  myslider1 = createSlider(10, 100, 86);
  myinput1 = createInput('Enter your name::');
}
function changeColor() {
  mybgcolor = color(random(255));
}
function draw() {
  background(mybgcolor);
  fill(255, 0, 175);
  ellipse(100, 100, myslider1.value(), myslider1.value());
  myname.html(myinput1.value());
  text(myinput1.value(), 10, 20);
}
```

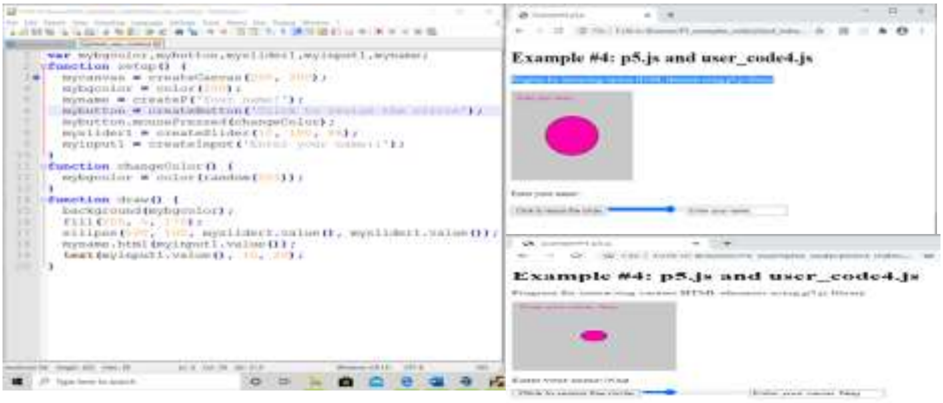
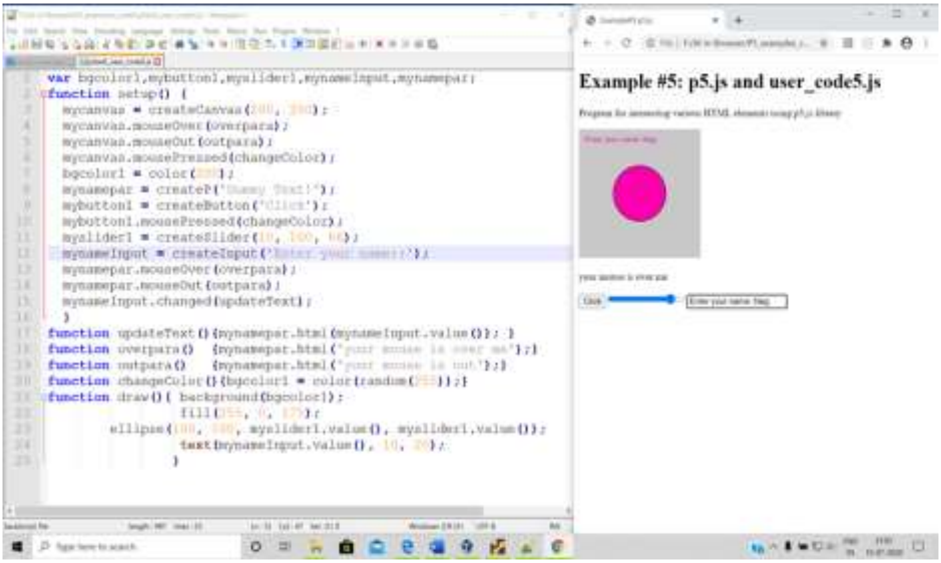

Example #4: p5.js and user_code4.js
Example #4: p5.js and user_code4.js

Fig1.16 interacting various HTML elements using p5.js library

Another example for interacting various HTML elements using p5.js library

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example#5 p5.js </title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex5_user_code5.js"></script>
  </head>
  <body>
    <h1>Example #5: p5.js and user_code5.js</h1>
    <p> Program for interacting various HTML elements using p5.js library
    </p>
  </body>
</html>
var bgcolor1,mybutton1,myslider1,mynameInput,mynamepar;
function setup() {
  mycanvas = createCanvas(200, 200);
  mycanvas.mouseOver(overpara);
  mycanvas.mouseOut(outpara);
  mycanvas.mousePressed(changeColor);
  bgcolor1 = color(200);
  mynamepar = createP('Dummy Text!');
  mybutton1 = createButton('Click');
  mybutton1.mousePressed(changeColor);
  myslider1 = createSlider(10, 100, 86);
  mynameInput = createInput('Enter your name::');
  mynamepar.mouseOver(overpara);
  mynamepar.mouseOut(outpara);
  mynameInput.changed(updateText);
  }
function updateText(){mynamepar.html(mynameInput.value()); }
function overpara()  {mynamepar.html('your mouse is over me');}
function outpara()   {mynamepar.html('your mouse is out');}
function changeColor(){bgcolor1 = color(random(255));}
function draw(){ background(bgcolor1);
                 fill(255, 0, 175);
ellipse(100, 100, myslider1.value(), myslider1.value());
         text(mynameInput.value(), 10, 20);
}
```



```
var bgcolor1,mybutton1,myslider1,mynameInput,mynamepar;
function setup() {
  mycanvas = createCanvas(200, 200);
  mycanvas.mouseOver(overpara);
  mycanvas.mouseOut(outpara);
  mycanvas.mousePressed(changeColor);
  bgcolor1 = color(200);
  mynamepar = createP('Dummy Text!');
  mybutton1 = createButton('Click');
  mybutton1.mousePressed(changeColor);
  myslider1 = createSlider(10, 100, 80);
  mynameInput = createInput('Enter your name!!');
  mynamepar.mouseOver(overpara);
  mynamepar.mouseOut(outpara);
  mynameInput.changed(updateText);
  }
function updateText(){mynamepar.html(mynameInput.value()); }
function overpara()   {mynamepar.html('your mouse is over me');}
function outpara()    {mynamepar.html('your mouse is out.');}
function changeColor(){bgcolor1 = color(random(255));}
function draw(){ background(bgcolor1);
                 fill(255, 0, 175);
         ellipse(100, 100, myslider1.value(), myslider1.value());
                 text(mynameInput.value(), 10, 20);
                }
```

Example #5: p5.js and user_code5.js

Program for interacting various HTML elements using p5.js library

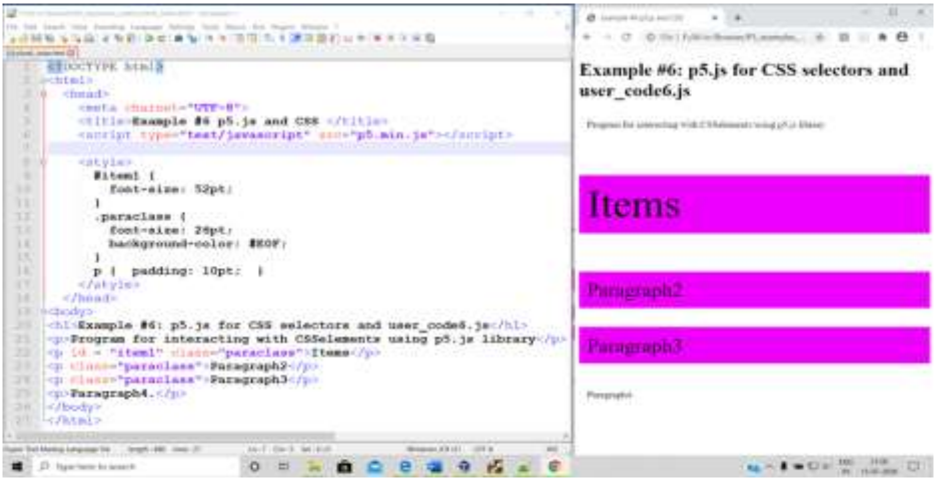
your mouse is over me

Fig1.17 Another example for interacting various HTML elements using p5.js library

Example P5.js with CSS Elements:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example #6 p5.js and CSS </title>
    <script type="text/javascript" src="p5.min.js"></script>
    <style>
      #item1 {
        font-size: 52pt;
      }
      .paraclass {
        font-size: 26pt;
        background-color: #E0F;
      }
      p {  padding: 10pt;  }
    </style>
  </head>
<body>
<h1>Example #6: p5.js for CSS selectors and user_code6.js</h1>
<p>Program for interacting with CSSelements using p5.js library</p>
<p id = "item1" class="paraclass">Items</p>
<p class="paraclass">Paragraph2</p>
<p class="paraclass">Paragraph3</p>
<p>Paragraph4.</p>
</body>
</html>
```

```
function setup() {}
function draw() {}
```



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example #6 p5.js and CSS </title>
    <script type="text/javascript" src="p5.min.js"></script>

    <style>
      #item1 {
        font-size: 52pt;
      }
      .paraclass {
        font-size: 26pt;
        background-color: #F0F;
      }
      p {  padding: 10pt;  }
    </style>
  </head>
<body>
<h1>Example #6: p5.js for CSS selectors and user_code6.js</h1>
<p>Program for interacting with CSSelements using p5.js library</p>
<p id = "item1" class="paraclass">Items</p>
<p class="paraclass">Paragraph2</p>
<p class="paraclass">Paragraph3</p>
<p>Paragraph4.</p>
</body>
</html>
```

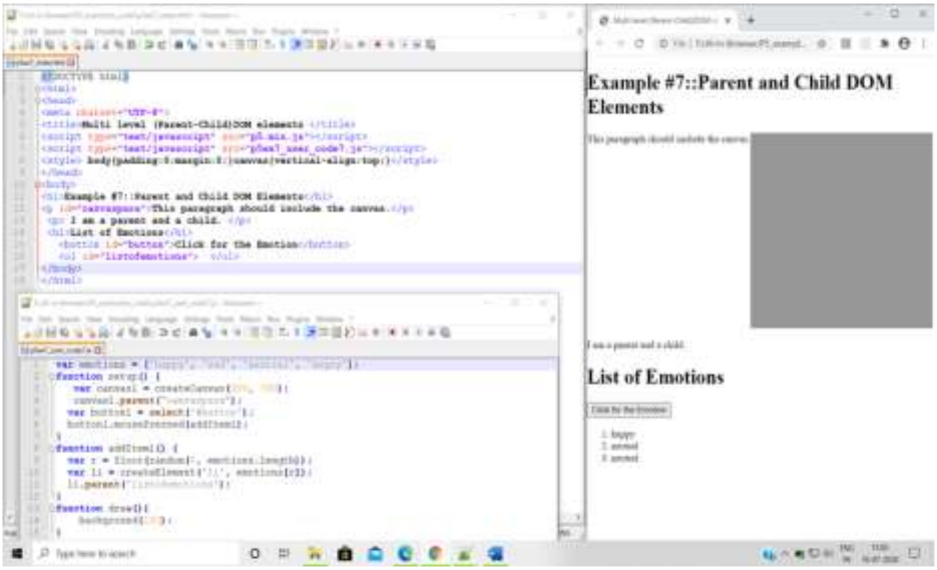
```
Example #6: p5.js for CSS selectors and user_code6.js

Program for interacting with CSSelements using p5.js library

Items

Paragraph2

Paragraph3

Paragraph4.
```

Fig1.18 P5.js with CSS ELEMENTS

Example P5.js Parent-Child DOM Elements

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Multi level (Parent-Child)DOM elements </title>
<script type="text/javascript" src="p5.min.js"></script>
<script type="text/javascript" src="p5ex7_user_code7.js"></script>
<style> body{padding:0;margin:0;}canvas{vertical-align:top;}</style>
</head>
<body>
<h1>Example #7::Parent and Child DOM Elements</h1>
<p id="canvaspara">This paragraph should include the canvas.</p>
 <p> I am a parent and a child. </p>
 <h1>List of Emotions</h1>
   <button id="button">Click for the Emotion</button>
   <ol id="listofemotions">  </ol>
</body>
</html>
var emotions = ['happy', 'sad', 'neutral', 'angry'];
function setup() {
   var canvas1 = createCanvas(300, 300);
   canvas1.parent("canvaspara");
  var button1 = select('#button');
  button1.mousePressed(addItem1);
}
```

```
function addItem1() {
  var r = floor(random(0, emotions.length));
  var li = createElement('li', emotions[r]);
  li.parent('listofemotions');
}
function draw(){
background(150);}
```



```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Multi level (Parent-Child)DOM elements </title>
<script type="text/javascript" src="p5.min.js"></script>
<script type="text/javascript" src="p5ex7_user_code7.js"></script>
<style> body{padding:0;margin:0;}canvas{vertical-align:top;}</style>
</head>
<body>
<h1>Example #7::Parent and Child DOM Elements</h1>
<p id="canvasparent">This paragraph should include the canvas.</p>
 <p> I am a parent and a child. </p>
 <h1>List of Emotions</h1>
  <button id="button">Click for the Emotion</button>
  <ol id="listofemotions">  </ol>
</body>
</html>
```

```
var emotions = [[illegible]];
function setup() {
   var canvas1 = createCanvas([illegible], [illegible]);
   canvas1.parent('canvasparent');
  var button1 = select('#button');
  button1.mousePressed(addItem1);
}
function addItem1() {
  var r = floor(random([illegible], emotions.length));
  var li = createElement('li', emotions[r]);
  li.parent('listofemotions');
}
function draw(){
    background([illegible]);
}
```

Example #7::Parent and Child DOM Elements

This paragraph should include the canvas.

I am a parent and a child.

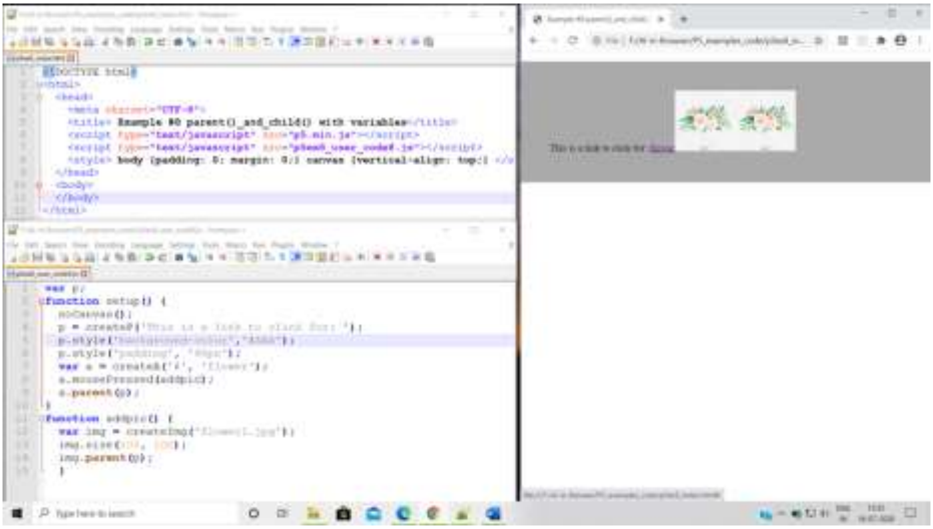
List of Emotions

1. happy
2. [illegible]
3. [illegible]

Fig1.19 P5.js along with parent Child elements

Example Parent-Child with variables：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title> Example #8 parent()_and_child() with variables</title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex8_user_code8.js"></script>
    <style> body {padding: 0; margin: 0;} canvas {vertical-align: top;}
    </style>
  </head>
  <body>
  </body>
</html>
var p;
function setup() {
  noCanvas();
  p = createP('This is a link to click for: ');
  p.style('background-color','#AAA');
  p.style('padding', '48px');
  var a = createA('#', 'flower');
  a.mousePressed(addpic);
  a.parent(p);
}
function addpic() {
  var img = createImg('flower1.jpg');
  img.size(100, 100);
  img.parent(p);
  }
```



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title> Example #0 parent()_and_child() with variables</title>
    <script type="text/javascript" src="p5.min.js"></script>
    <script type="text/javascript" src="p5ex0_user_code#.js"></script>
    <style> body {padding: 0; margin: 0;} canvas {vertical-align: top;} </s
  </head>
  <body>
  </body>
</html>
```

```
var p;
function setup() {
  noCanvas();
  p = createP('This is a link to click for: ');
  p.style('background-color','#AAA');
  p.style('padding', '4px');
  var a = createA('#', 'flower');
  a.mousePressed(addpic);
  a.parent(p);
}
function addpic() {
  var img = createImg('flower1.jpg');
  img.size(100, 100);
  img.parent(p);
  }
```

This is a link to click for: flower

Fig1.20 P5.js with Parent-Child variables

Graphics and Interactive processing in the browser using Java Script libraries (P5.js)

P5.js is another understanding of Processing written in JavaScript that makes it simple to associate with HTML5 objects, including text, input, video, webcam, and sound. It helps in how to apply information perception methods to receive information from and to webcam sound and video and handle them in the program. The following examples illustrate some of the graphics features that are managed by the P5.js library and its functions.

Example P5.js graphics functions to illustrate mouse press event

```
<!DOCTYPE html>
<html>
<head>
  <script src="p5.min.js"></script>
  <meta charset="utf-8" />
</head>
<body>
<script src="p5_graphics_ex1_user_code1.js"></script>
</body>
</html>

let x = 100; let y = 100; let extraCanvas1;
function setup() {
  createCanvas(300, 300);
  extraCanvas1 = createGraphics(300, 300);
  extraCanvas1.clear();
 }
function draw() {
  background(255,204,0);
  x += random(-5, 5);
  y += random(-5, 5);
  if (mouseIsPressed) {
    extraCanvas1.fill(255, 150);
    extraCanvas1.noStroke();
    extraCanvas1.ellipse(mouseX, mouseY, 60, 60);
  }
  image(extraCanvas1, 0, 0);
  fill('blue');  stroke(255);
  rectMode(CENTER);  rect(x, y, 20, 20);}
```

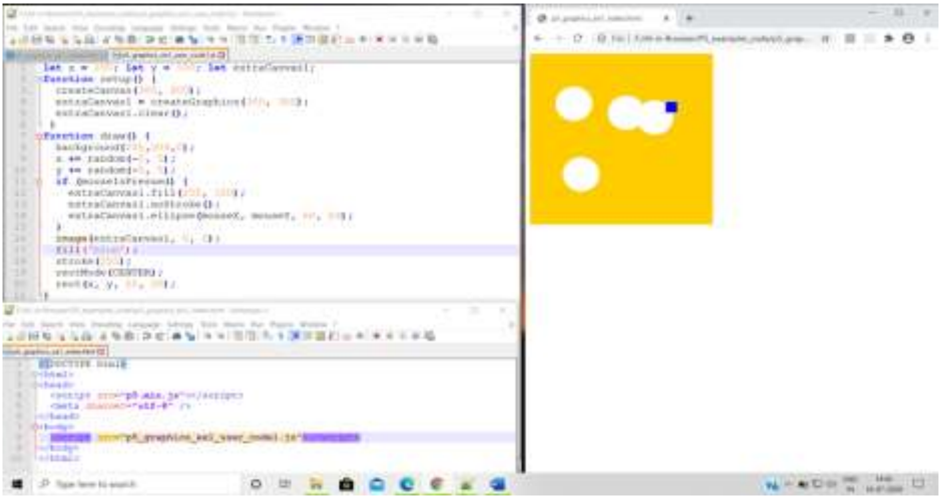
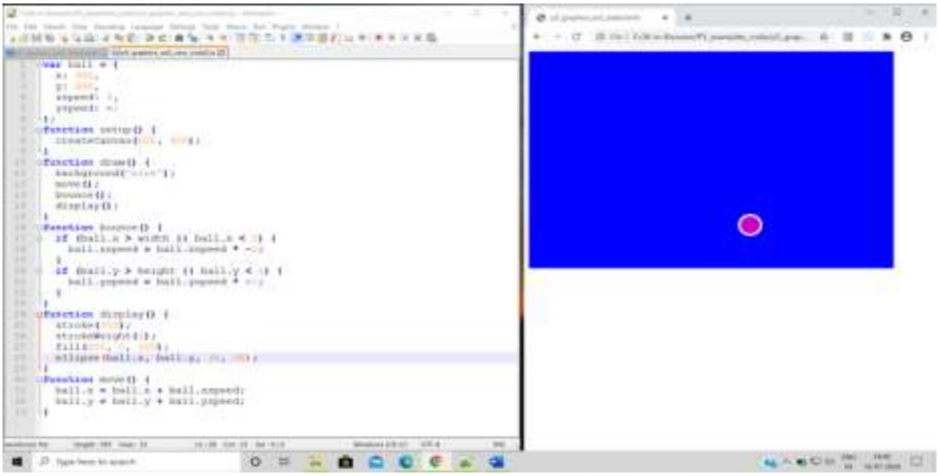


Fig1.21 P5.js graphics with mouse press event

Example P5.js graphics script for Bouncing Ball illustration

```
<!DOCTYPE html>
<html>
<head>
  <script src="p5.min.js"></script>
  <meta charset="utf-8" />
</head>
<body>
<script src="p5_graphics_ex2_user_code2.js"></script>
</body>
</html>
```

```
var ball = {
  x: 300,
  y: 200,
  xspeed: 4,
  yspeed: -3
};
function setup() {
  createCanvas(600, 400);
}
function draw() {
  background('blue');
  move();
  bounce();
  display();
}
function bounce() {
  if (ball.x > width || ball.x < 0) {
    ball.xspeed = ball.xspeed * -1;
  }
  if (ball.y > height || ball.y < 0) {
    ball.yspeed = ball.yspeed * -1;
  }
}
function display() {
  stroke(255);
  strokeWeight(4);
  fill(200, 0, 200);
  ellipse(ball.x, ball.y, 36, 36);
}
function move() {
  ball.x = ball.x + ball.xspeed;
  ball.y = ball.y + ball.yspeed;
}
```



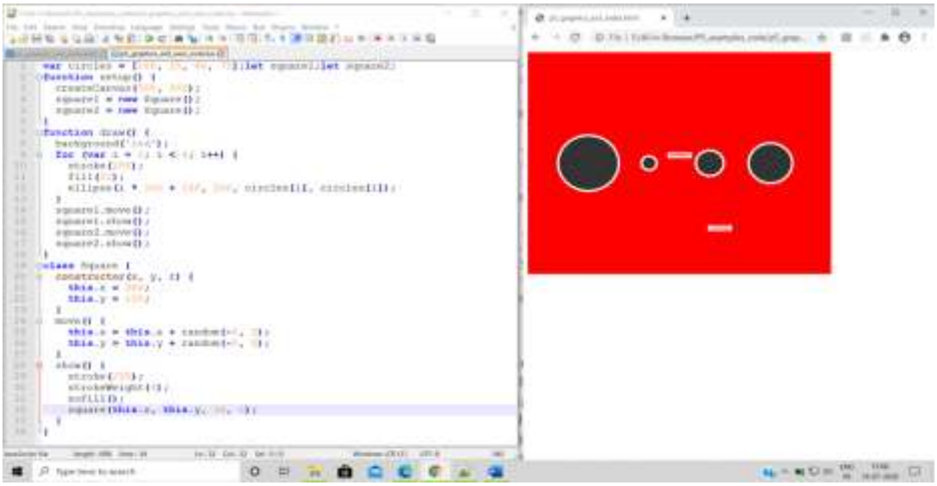
```
var ball = {
  x: [illegible],
  y: [illegible],
  xspeed: [illegible],
  yspeed: [illegible]
};
function setup() {
  createCanvas([illegible], [illegible]);
}
function draw() {
  background('blue');
  move();
  bounce();
  display();
}
function bounce() {
  if (ball.x > width || ball.x < [illegible]) {
    ball.xspeed = ball.xspeed * -[illegible];
  }
  if (ball.y > height || ball.y < [illegible]) {
    ball.yspeed = ball.yspeed * [illegible];
  }
}
function display() {
  stroke([illegible]);
  strokeWeight([illegible]);
  fill([illegible], [illegible], [illegible]);
  ellipse(ball.x, ball.y, [illegible], [illegible]);
}
function move() {
  ball.x = ball.x + ball.xspeed;
  ball.y = ball.y + ball.yspeed;
}
```

Fig1.22 P5.js graphics script for Bouncing Ball illustration

Example P5.js graphics function on how to use Class, Objects, Arrays, and loops features

```
<!DOCTYPE html>
<html>
<head>
  <script src="p5.min.js"></script>
  <meta charset="utf-8" />
</head>
<body>
<script src="p5_graphics_ex3_user_code3.js"></script>
</body>
</html>

var circles = [100, 25, 46, 72];let square1;let square2;
function setup() {
  createCanvas(500, 400);
  square1 = new Square();
  square2 = new Square();
}
function draw() {
  background('red');
  for (var i = 0; i < 4; i++) {
    stroke(255);
    fill(51);
    ellipse(i * 100 + 100, 200, circles[i], circles[i]);
  }
  square1.move();
  square1.show();
  square2.move();
  square2.show();
}
```

```
class Square {
  constructor(x, y, r) {
    this.x = 200;
    this.y = 150;
  }
 move() {
    this.x = this.x + random(-5, 5);
    this.y = this.y + random(-5, 5);
  }
  show() {
    stroke(255);
    strokeWeight(4);
    noFill();
    square(this.x, this.y, 36, 6);
  }
}
```



```
var circles = [[illegible]];let square1;let square2;
function setup() {
  createCanvas([illegible]);
  square1 = new Square();
  square2 = new Square();
}
function draw() {
  background('[illegible]');
  for (var i = [illegible]; i < [illegible]; i++) {
    stroke([illegible]);
    fill([illegible]);
    ellipse(i * [illegible] + [illegible], [illegible], circles[i], circles[i]);
  }
  square1.move();
  square1.show();
  square2.move();
  square2.show();
}
class Square {
  constructor(x, y, r) {
    this.x = [illegible];
    this.y = [illegible];
  }
  move() {
    this.x = this.x + random(-[illegible], [illegible]);
    this.y = this.y + random(-[illegible], [illegible]);
  }
  show() {
    stroke([illegible]);
    strokeWeight([illegible]);
    noFill();
    square(this.x, this.y, [illegible], [illegible]);
  }
}
```

Fig1.23 P5.js graphics function on how to use Class, Objects, Arrays, and loops features

Example P5.js graphics Object Interaction functionality

```
<!DOCTYPE html>
<html>
<head>
  <script src="p5.min.js"></script>
  <meta charset="utf-8" />
</head>
<body>
  <script src="p5_graphics_ex4_user_code4.js"></script>
</body>
</html>
let bubbles = [];let flower;let kittens = [];
function preload() {
  flower = loadImage('kittens/flower.png');
  for (let i = 0; i < 5; i++) {
    kittens[i] = loadImage(`kittens/kitten${i}.jpg`);
  }
}
function setup() {
  createCanvas(600, 400);
  for (let i = 0; i < 10; i++) {
    let x = random(width);
    let y = random(height);
    let r = random(50, 150);
    // let kitten = random(kittens);
    let b = new Bubble(x, y, r);
    bubbles.push(b);
  }
}
function mousePressed() {
  for (let i = 0; i < bubbles.length; i++) {
    bubbles[i].clicked(mouseX, mouseY);
  }
}
function draw() {
  background(0);
  for (let i = 0; i < bubbles.length; i++) {
    bubbles[i].move();
    bubbles[i].show();
  }
}
class Bubble {
  constructor(x, y, r, img) {
    this.x = x;
    this.y = y;
    this.r = r;
    this.kitten = random(kittens);
  }
  clicked(px, py) {
    //let d = dist(px, py, this.x, this.y);
    //if (d < this.r) {
    if (
      px > this.x &&
      px < this.x + this.r &&
      py > this.y &&
      py < this.y + this.r
    ) {
      this.kitten = flower; //random(kittens);
    }
  }
  move() {
    this.x = this.x + random(-2, 2);
    this.y = this.y + random(-2, 2);
  }
  show() {
    image(this.kitten, this.x, this.y, this.r, this.r);
    // stroke(255);
    // strokeWeight(4);
    // fill(this.brightness, 125);
    // ellipse(this.x, this.y, this.r * 2);
  }
}
```

Getting started with Machine Learning-ML5.js + P5.js libraries:

ML5.js is a library that intends to make AI congenial for a full imaginative coder. The library is created at New York University and has been openly released in July 2018. The library gives access to AI methods and models in the program, expanding on the head of TensorFlow.js with no other outside dependencies. *ML5.js* makes life simpler for individuals who are new to the Machine Learning field. More details are provided at this link.

Requirements for development and executing programs involving p5.js and ML5.js libraries:

Editor: VScode or Notepad++
Web browser: Chrome or Firefox and
Sample Images and Datasets for developing ML applications

Design, Develop, and Execute programs locally: There are **two** methods for running a local web server on your computer to develop and execute your programs on the local computer.

Method1: Using python – httpserver:

Step1: Install python 3+ on your computer.
Step 2: cd /path_to/ml5_p5-examples (at the command prompt, goto the folder where your
ml5/p5 programs exist)
Step 3: python -m http.server 8081
Step 4: In the browser URL type: localhost:8081/indexfilename

Example How to use ML5.js library using python webserver for Image Classification

```
<html>
<head>
  <meta charset="UTF-8">
  <title>Image Classification Example</title>
  <script src="ml5.min.js" type="text/javascript"></script>
</head>
<body>
```

```
  <h1>Image classification using MobileNet model</h1>
  <p>The MobileNet model labeled this as
    <span id="result">... </span> with a confidence of
    <span id="probability">...</span>.
  </p>
  <img src="images/bird1.png" id="image" width="400" />
  <script src="ml5_ex1_user_code1.js"></script>
</body>
</html>
const image = document.getElementById('image');
const result = document.getElementById('result');
const probability = document.getElementById('probability');
ml5.imageClassifier('MobileNet')
  .then(classifier => classifier.classify(image))
  .then(results => {
    result.innerText = results[0].label;
    probability.innerText = results[0].confidence.toFixed(4);
  });
```

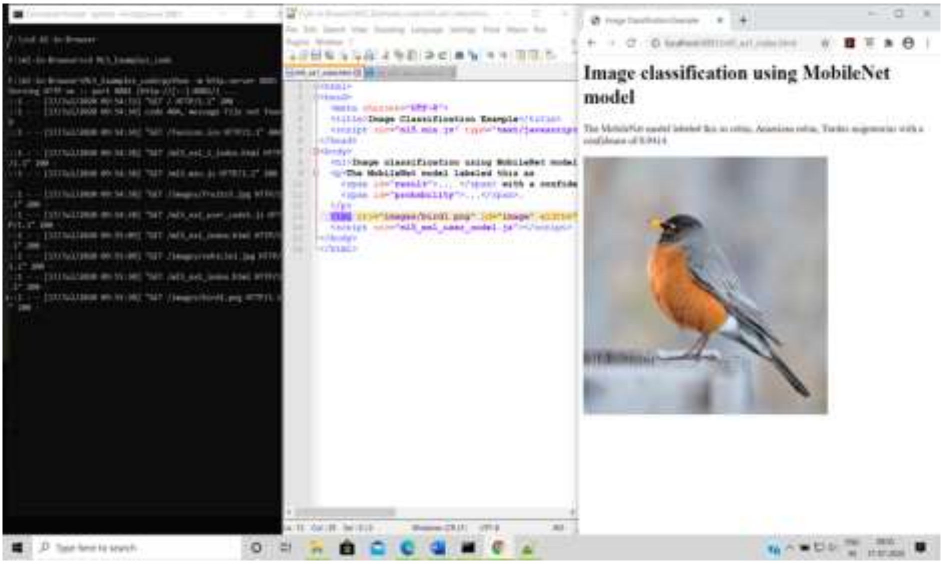


Image classification using MobileNet model

The MobileNet model labeled this as robin, American robin, Turdus migratorius with a confidence of 0.9414

Fig1.24 ML5.js library along with python server and notepad++ for Image Classification

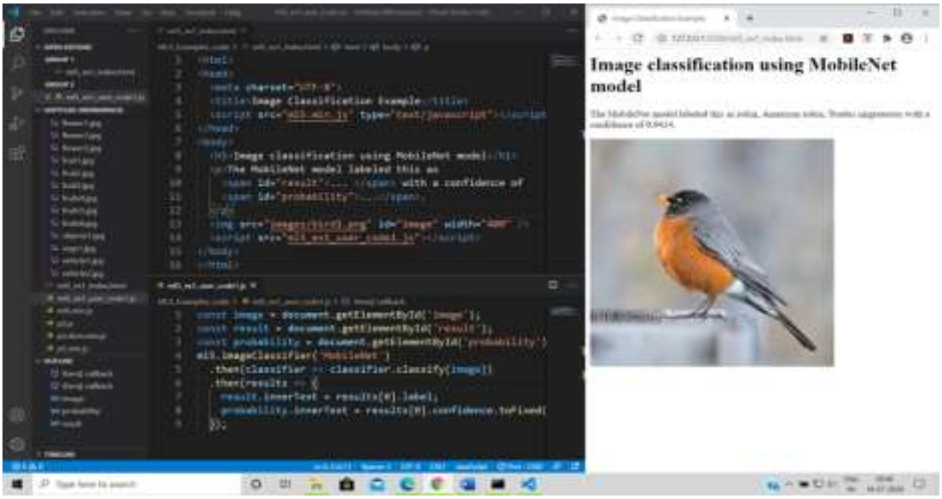
Method2: Using Visual Studio Code Editor with node.js Live Server:

Step1: Download and Install node.js as given in this *link*
Step 2: Installing NPM (Node Package Manager) as given in this **link**
Step 3: Download and Install VSCode by following the steps given in this *link*

Step 4: Open the VSCode editor and click the Golive button, so that the output of the program can be seen in the browser. The following examples depict the execution of the programs under this method.

Example Image classification using node.js server

```
<html>
<head>
  <meta charset="UTF-8">
  <title>Image Classification Example</title>
  <script src="ml5.min.js" type="text/javascript"></script>
</head>
<body>
  <h1>Image classification using MobileNet model</h1>
  <p>The MobileNet model labeled this as
    <span id="result">... </span> with a confidence of
    <span id="probability">...</span>.
  </p>
  <img src="images/fruits5.jpg" id="image" width="400" />
  <script src="ml5_ex1_user_code1.js"></script>
</body>
</html>

const image = document.getElementById('image');
const result = document.getElementById('result');
const probability = document.getElementById('probability');
ml5.imageClassifier('MobileNet')
  .then(classifier => classifier.classify(image))
  .then(results => {
    result.innerText = results[0].label;
    probability.innerText = results[0].confidence.toFixed(4);
  });
```



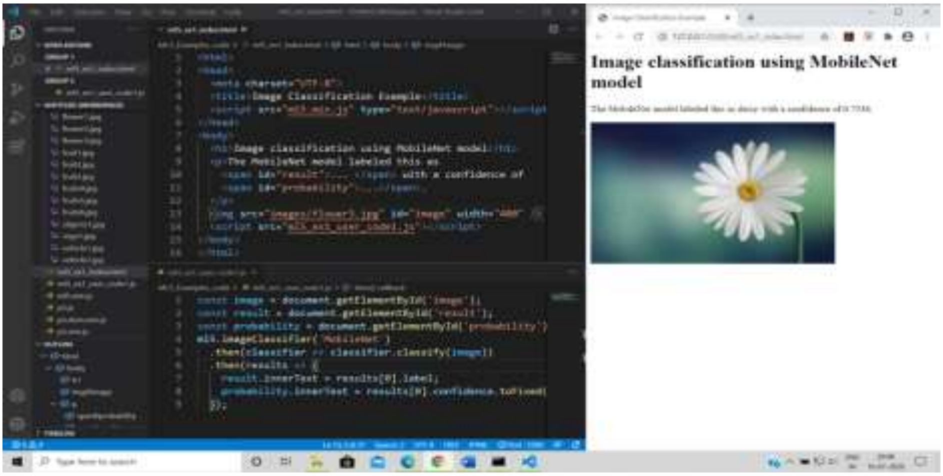
```
<html>
<head>
  <meta charset="UTF-8">
  <title>Image Classification Example</title>
  <script src="ml5.min.js" type="text/javascript"></script>
</head>
<body>
  <h1>Image classification using MobileNet model</h1>
  <p>The MobileNet model labeled this as
    <span id="result">... </span> with a confidence of
    <span id="probability">...</span>.
  </p>
  <img src="images/bird1.png" id="image" width="400" />
  <script src="ml5_ex1_user_code1.js"></script>
</body>
</html>
```

```
const image = document.getElementById('image');
const result = document.getElementById('result');
const probability = document.getElementById('probability')
ml5.imageClassifier('MobileNet')
  .then(classifier => classifier.classify(image))
  .then(results => {
    result.innerText = results[0].label;
    probability.innerText = results[0].confidence.toFixed(
  });
```

Image classification using MobileNet model

The MobileNet model labeled this as robin, American robin, Turdus migratorius with a confidence of 0.9424.

Fig1.25 (a) Image classification using node.js server and Visual Studio code



```
<html>
<head>
  <meta charset="UTF-8">
  <title>Image Classification Example</title>
  <script src="ml5.min.js" type="text/javascript"></script>
</head>
<body>
  <h1>Image classification using MobileNet model</h1>
  <p>The MobileNet model labeled this as
    <span id="result">...</span> with a confidence of
    <span id="probability">...</span>.
  </p>
  <img src="images/flower1.jpg" id="image" width="400"
  <script src="ml5_ex1_user_code1.js"></script>
</body>
</html>
```

```
const image = document.getElementById('image');
const result = document.getElementById('result');
const probability = document.getElementById('probability')
ml5.imageClassifier('MobileNet')
  .then(classifier => classifier.classify(image))
  .then(results => {
    result.innerText = results[0].label;
    probability.innerText = results[0].confidence.toFixed(
  });
```

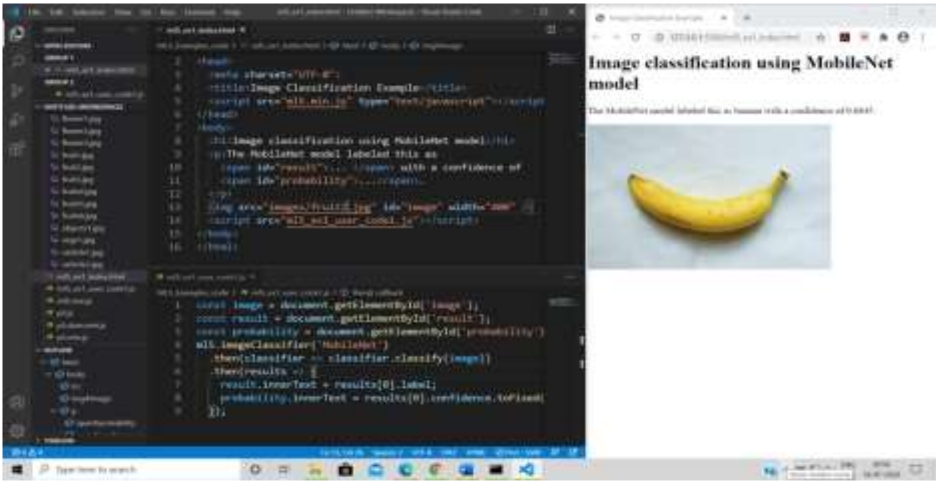
Image classification using MobileNet model

The MobileNet model labeled this as daisy with a confidence of [illegible]

Fig1.25 (b) Image classification using node.js server and Visual Studio code

“ML⁵.js methods and functions are asynchronous (because machine learning models can take significant amounts of time to process inputs and generate outputs!).

Using Promises: ML⁵.js also supports Promises. If no callback is provided to any asynchronous function then a Promise is returned. With Promises, the image classification example can be used in the following way:”



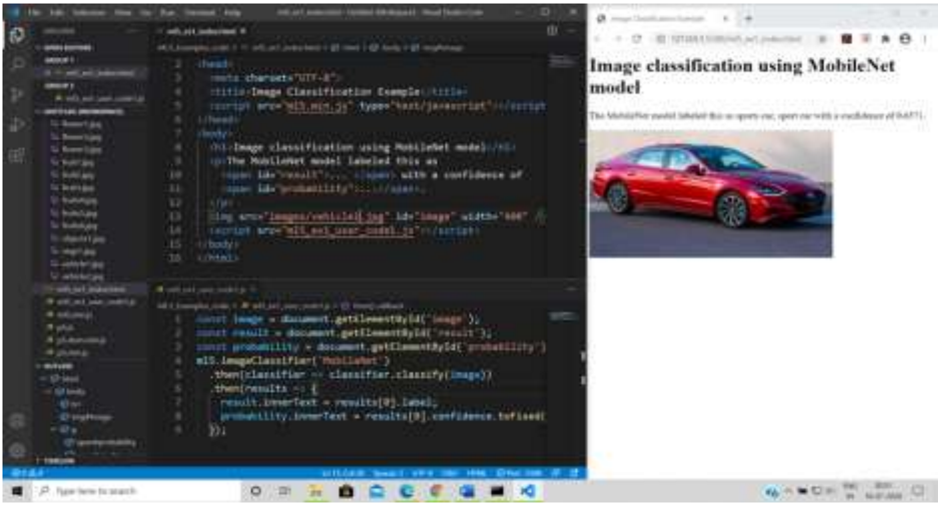
```
<head>
  <meta charset="UTF-8">
  <title>Image Classification Example</title>
  <script src="ml5.min.js" type="text/javascript"></script>
</head>
<body>
  <h1>Image classification using MobileNet model</h1>
  <p>The MobileNet model labeled this as
    <span id="result">...</span> with a confidence of
    <span id="probability">...</span>.
  </p>
  <img src="images/fruit2.jpg" id="image" width="400"
  <script src="ml5_ex1_user_code1.js"></script>
</body>
</html>
```

```
const image = document.getElementById('image');
const result = document.getElementById('result');
const probability = document.getElementById('probability')
ml5.imageClassifier('MobileNet')
  .then(classifier => classifier.classify(image))
  .then(results => {
    result.innerText = results[0].label;
    probability.innerText = results[0].confidence.toFixed(
  });
```

Image classification using MobileNet model

The MobileNet model labeled this as banana with a confidence of [illegible]

Fig1.25 (c) Image classification using node.js server and Visual Studio code



```
<head>
  <meta charset="UTF-8">
  <title>Image Classification Example</title>
  <script src="ml5.min.js" type="text/javascript"></script
</head>
<body>
  <h1>Image classification using MobileNet model</h1>
  <p>The MobileNet model labeled this as
    <span id="result">...</span> with a confidence of
    <span id="probability">...</span>.
  </p>
  <img src="images/vehicle1.jpg" id="image" width="400" />
  <script src="ml5_ex1_user_code1.js"></script>
</body>
</html>
```

```
const image = document.getElementById('image');
const result = document.getElementById('result');
const probability = document.getElementById('probability')
ml5.imageClassifier('MobileNet')
  .then(classifier => classifier.classify(image))
  .then(results => {
    result.innerText = results[0].label;
    probability.innerText = results[0].confidence.toFixed(
  });
```

Image classification using MobileNet model

The MobileNet model labeled this as sports car, sport car with a confidence of 0.4771.

Fig1.25 (d) Image classification using node.js server and Visual Studio code

References

Book: Beginning Machine Learning in the Browser

Nagender Kumar Suryadevara