|  |  |  |
| --- | --- | --- |
| **Web Programming Laboratory**  **Week 1: Introduction** |  | |
| Name: Rutathorn Worraphatthada | ID:6231302026 | Section:01 |
| Date: 29/08/2019 | Due: Tuesday Midnight | |

**Objective**

* To review basic CSS and JavaScript programming

**References**

* <http://www.w3schools.com/css/default.asp>
* <http://www.w3schools.com/js/default.asp>
* <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Classes>

**Requirements**

* Visual Studio Code
* Any web browser
* Basic HTML, CSS and JavaScript knowledge

Web development is mainly composed of two big parts: front-end and back-end developments.

* Front end refers to client side including UI/UX design, input and output to end users
* Back end is related to server side which takes care of the most processing tasks including database connection

Generally the basic of web programming on client side requires three foundations: HTML, CSS and JavaScript. However, for businesses, there are various web frameworks to help develop webs more efficiently e.g. Bootstrap, Angular, React, Vue etc.

In this course, the front-end part will be based on HTML, CSS and JavaScript with Bootstrap. Then the back end will cover MySQL database connection, server-side processing with NodeJS and ExpressJS.

**Exercise 1** Style an app using “CSS”

CSS (Cascading Style Sheet) is a set of commands used to format the display of webpages.



Create a new HTML file such as “ex01.html” and paste the following codes.

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <html>  <head>  <style>  p {  font-size: 30px;  text-align: center;  color: blue;  }  </style>  </head>  <body>  <p>Hello</p>  <p>World</p>  <p>I love programming!</p>  </body>  </html> |

Then try to modify the CSS as below.

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <style>  p {  font-size: 30px;  font-weight: bold;  text-align: center;  color: orange;  }  body {  background-color: rgb(100, 120, 225);  }  </style>  </head>  <body>  <p>Hello</p>  <p>World</p>  <p>I love programming!</p>  </body>  </html> |

After refreshing, the new display should be:



**Exercise 2** Class and ID selectors

* id is unique and is assigned to a single widget
* class is common and is for multiple widgets

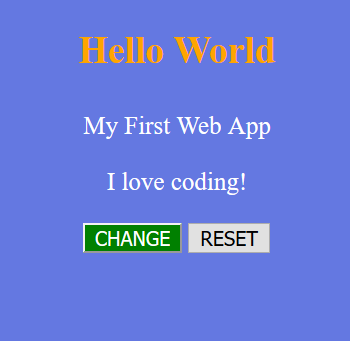
|  |
| --- |
| <!DOCTYPE html>  <html lang="en">      <head>  <meta charset="utf-8">          <style>              p {                  text-align: center;              }              body {                  background-color: rgb(100,120,225);              }              #title {                  font-size: 30px;                  font-weight: bold;                  color: orange;              }              .normal {                  font-size: 20px;                  color: white;              }          </style>      </head>      <body>          <p id="title">Hello</p>          <p class="normal">World</p>          <p class="normal">I love programming!</p>      </body>  </html> |

Here is the result.



Observe that the symbol “#” is an ID selector and “.” is a class selector for CSS.

**Assignment 1** Modify the previous exercise to get the result below. When you click any button, nothing happens.



*assign1.html*

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <html>    <head>  <meta charset="utf-8">  <style>  p{  text-align:center;  }  body{  background-color:rgb(100,120,255);  }  #title{  font-size:30px;  font-weight: bold;  color:orange;  }  .normal{  font-size:20px;  color:white;  }  input[type=button]{  font-size:16px;  }  .bcenter{  text-align:center;  }  #bttChange{  background-color:green;  color:white;  }    </style>  </head>  <body>  <p id="title">Hello World</p>  <p class="normal">My childhood Web App</p>  <p class="normal">I love Coding!</p>  <div class="bcenter">  <input type="button" id="bttChange" value="CHANGE">  <input type="button" value="RESET">  </div>  </body>  </html> |

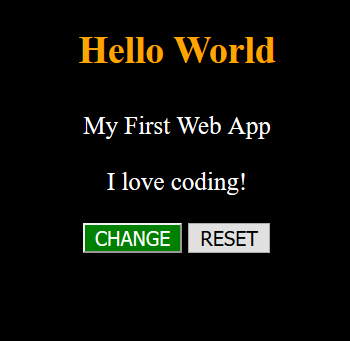
**Exercise 3** Click events

Let us try to add click event to a button “CHANGE” to render page background to black. Finally tap “RESET” will return background to purple.

To add functionality to this page, we need a client-side script such as JavaScript.



Clicking “CHANGE” will

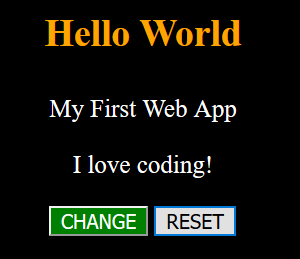


|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <style>              p {                  text-align: center;              }              body {                  background-color: purple;              }              #title {                  font-size: 30px;                  font-weight: bold;                  color: orange;              }              .normal {                  font-size: 20px;                  color: white;              }              input[type=button] {                  font-size: 16px;              }              .bcenter {                  text-align: center;              }              #bttChange {                  background-color: green;                  color: white;              }          </style>      </head>      <body>          <p id="title">Hello World</p>          <p class="normal">My First Web App</p>          <p class="normal">I love coding!</p>          <div class="bcenter">              <input type="button" id="bttChange" value="CHANGE" onclick="changeBackground('black')">              <input type="button" value="RESET" onclick="changeBackground('purple')">          </div>            <script>              function changeBackground(color) {                   document.body.style.backgroundColor = color;              }          </script>      </body>  </html> |

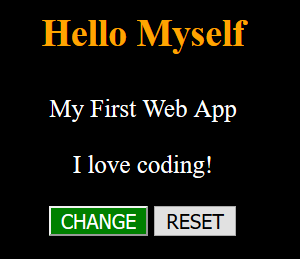
**Exercise 4** Refer to a widget through its ID

To access a widget by code for some purposes such as changing its properties, we can refer to its ID. Therefore the widget we need to access must be given an ID.

We will modify the previous exercise so that clicking “CHANGE” will alter the title text and tapping “RESET” will revert it.



After clicking “CHANGE”, the title is modified.



|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <style>  p {  text-align: center;  }  body {  background-color: black;  }  #title {  font-size: 30px;  font-weight: bold;  color: orange;  }  .normal {  font-size: 20px;  color: white;  }  input[type=button] {  font-size: 16px;  }  .bcenter {  text-align: center;  }  #bttChange {  background-color: green;  color: white;  }  </style>  </head>  <body>  <p id="title">Hello World</p>  <p class="normal">My First Web App</p>  <p class="normal">I love coding!</p>  <div class="bcenter">  <input type="button" id="bttChange" value="CHANGE" onclick="changeTitle('Myself')">  <input type="button" value="RESET" onclick="changeTitle('World')">  </div>  <script>  function changeTitle(message) {  document.getElementById("title").innerHTML = "Hello " + message;  // document.querySelector("#title").innerHTML = "Hello " + message;  }  </script>  </body>  </html> |
|  |

**Remark**

It is possible to bind JS event to HTML elements directly by JS. Let’s try to modify the previous exercise to get the same output.

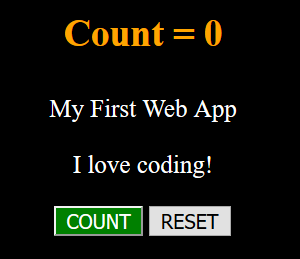
|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <style>  p {  text-align: center;  }  body {  background-color: black;  }  #title {  font-size: 30px;  font-weight: bold;  color: orange;  }  .normal {  font-size: 20px;  color: white;  }  input[type=button] {  font-size: 16px;  }  .bcenter {  text-align: center;  }  #bttChange {  background-color: green;  color: white;  }  </style>  </head>  <body>  <p id="title">Hello World</p>  <p class="normal">My First Web App</p>  <p class="normal">I love coding!</p>  <div class="bcenter">  <input type="button" id="bttChange" value="CHANGE"">  <input type="button" id="bttReset" value="RESET">  </div>  <script>  //find change button  let bttChange = document.querySelector("#bttChange");  //bind click event to the button  bttChange.addEventListener("click", changeTitle);  //find reset button  let bttReset = document.querySelector("#bttReset");  //bind click event to the button  bttReset.addEventListener("click", resetTitle);  function changeTitle() {  let title = document.querySelector("#title");  title.innerHTML = "Hello MySelf";  }  function resetTitle() {  let title = document.querySelector("#title");  title.innerHTML = "Hello World";  }  </script>  </body>  </html> |
|  |

Or if we want to share a function for both buttons, we can add an anonymous function to handle function’s arguments.

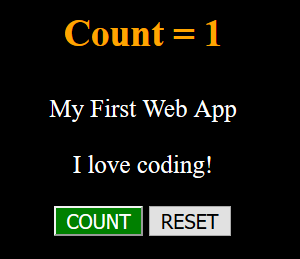
|  |
| --- |
| <script>  //find change button  let bttChange = document.querySelector("#bttChange");  //bind click event to the button  bttChange.addEventListener("click", function() {changeTitle("Myself")});  //find reset button  let bttReset = document.querySelector("#bttReset");  //bind click event to the button  bttReset.addEventListener("click", function() {changeTitle("World")});  function changeTitle(message) {  let title = document.querySelector("#title");  title.innerHTML = "Hello " + message;  }  </script> |
|  |

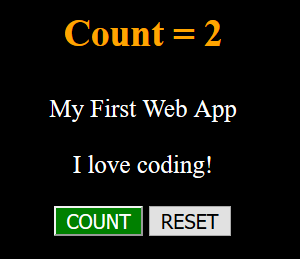
**Assignment 2** Click counting

Develop a web app to count a number of clicks and provide the reset function.

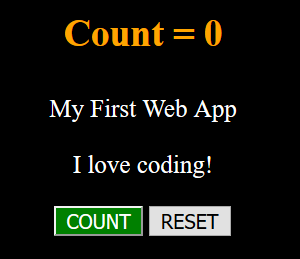


Clicking “CHANGE” will increase the counter.





Clicking “RESET” will reset the counter.



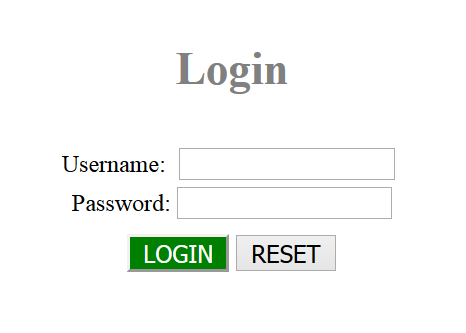
Paste your code below.

*assign2.html*

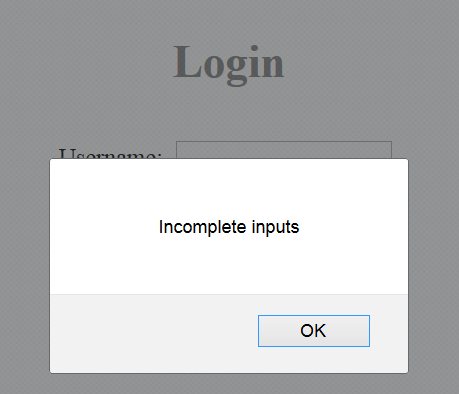
|  |
| --- |
|  |

**Assignment 3** Get inputs from TextField

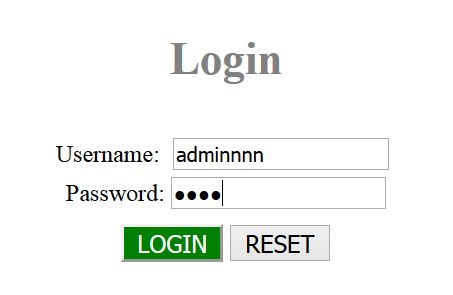
Create a login app to get username and password which are assumed to be “admin” and “1234”. Then alert for the login result.

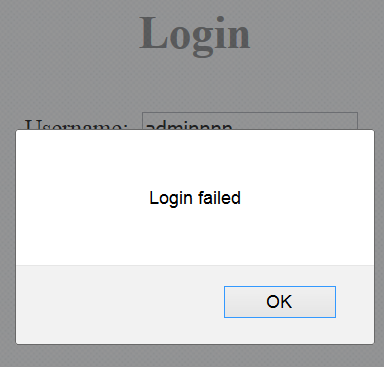


If we miss username or password and click “LOGIN”,

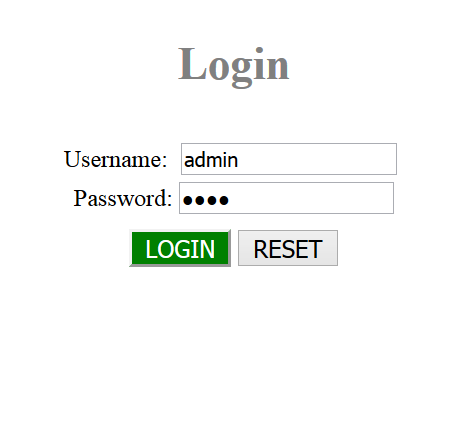


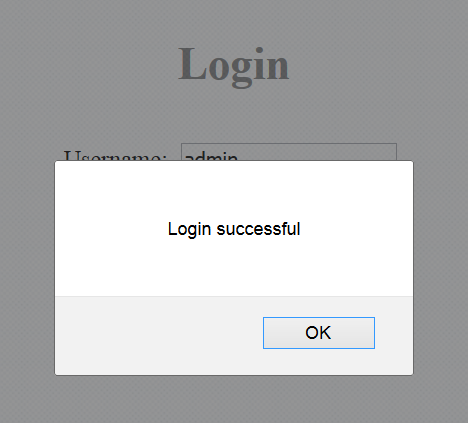
If we put the wrong username or password,



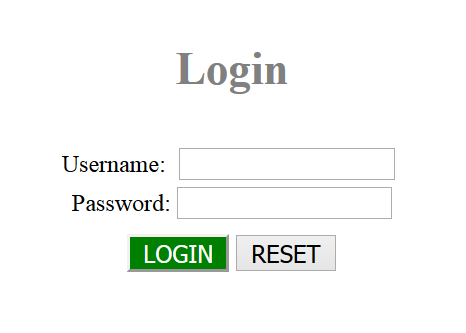


If correct login,





Clicking “RESET” will reset all inputs.

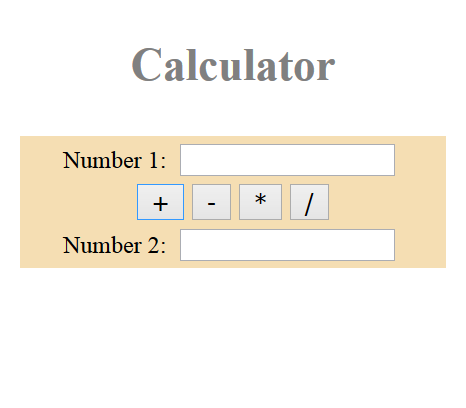


Paste your code below.

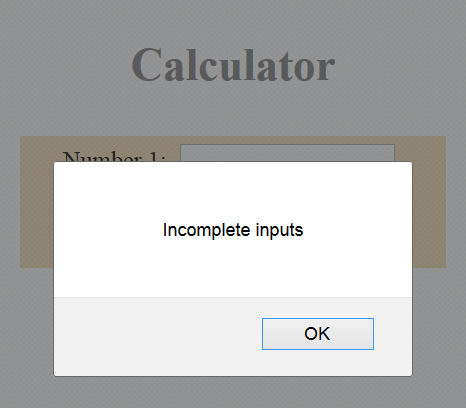
*assign3.html*

|  |
| --- |
|  |

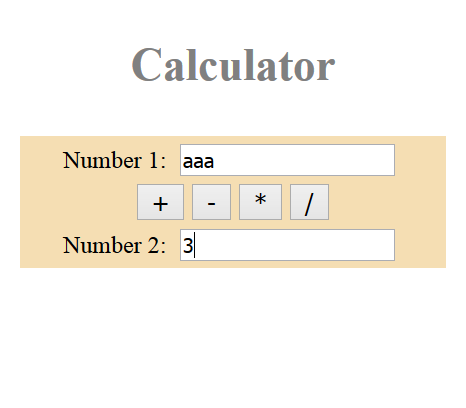
**Assignment 4** Simple calculator

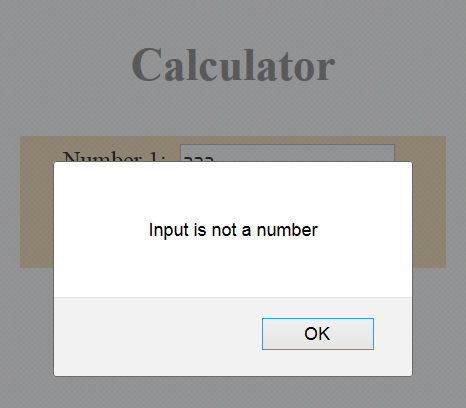


If missing input and click any button,

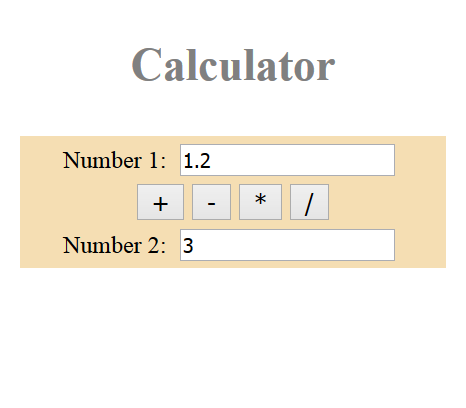


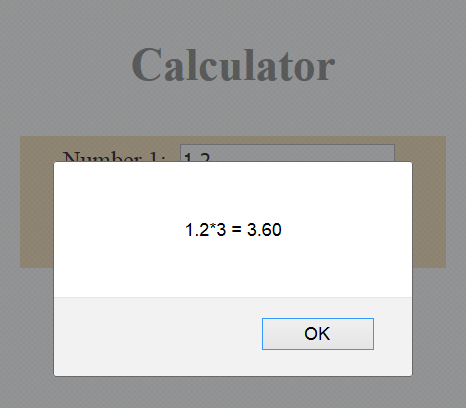
If input is not a number,





If correct inputs and then we click \*.



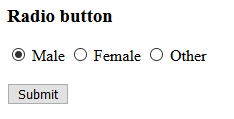


Paste your code below.

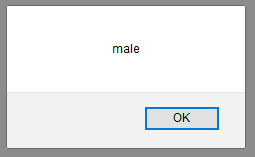
*assign4.html*

|  |
| --- |
|  |

**Exercise 5** HTML radio buttons and select

****

When clicking a button,

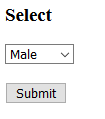


|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>HTML Inputs</title>  </head>  <body>  <h3>Radio button</h3>  <input type="radio" name="gender" value="male" checked> Male  <input type="radio" name="gender" value="female"> Female  <input type="radio" name="gender" value="other"> Other <br><br>  <button onclick="getRadio()">Submit</button>  <script>  function getRadio() {  //get all elements whose name is "gender", the return is an array  var arr\_gender = document.getElementsByName("gender");  var gender = "male";  //loop to check which radio is checked and get its value  for(var i=0; i<arr\_gender.length; i++) {  //is this radio checked?  if(arr\_gender[i].checked) {  gender = arr\_gender[i].value;  break;  }  }  alert(gender);  }  </script>  </body>  </html> |

We can simplify this code using “document.querySelector()”.

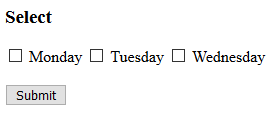
|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>HTML Inputs</title>  </head>  <body>  <h3>Radio button</h3>  <input type="radio" name="gender" value="male" checked> Male  <input type="radio" name="gender" value="female"> Female  <input type="radio" name="gender" value="other"> Other <br><br>  <button onclick="getRadio()">Submit</button>  <script>  function getRadio() {  //query and get radio's value  var gender = document.querySelector("input[name='gender']:checked").value;  alert(gender);  }  </script>  </body>  </html> |

Let us change radios to select.

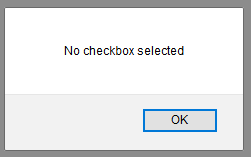


|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>HTML Inputs</title>  </head>  <body>  <h3>Select</h3>  <select id="gender">  <option value="male">Male</option>  <option value="female">Female</option>  <option value="other">Other</option>  </select>  <br><br>  <button onclick="getGender()">Submit</button>  <script>  function getGender() {  //query and get select's value  var element = document.querySelector("#gender");  // var element = document.getElementById("gender");  var gender = element.options[element.selectedIndex].value;  alert(gender);  }  </script>  </body>  </html> |

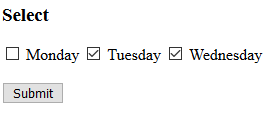
**Exercise 6** HTML checkboxes

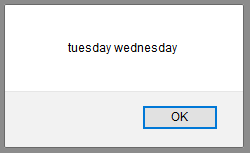


If select nothing,



If select some,





|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>HTML Inputs</title>  </head>  <body>  <h3>Select</h3>  <input type="checkbox" name="day" value="monday"> Monday  <input type="checkbox" name="day" value="tuesday"> Tuesday  <input type="checkbox" name="day" value="wednesday"> Wednesday  <br><br>  <button onclick="getDays()">Submit</button>  <script>  function getDays() {  //get all checkboxes  var days = document.querySelectorAll("input[name='day']:checked");  //if nothing is checked  if(days.length == 0) {  alert("No checkbox selected");  return;  }  //show all selected values  var values = "";  for(var i=0; i<days.length; i++) {  values = values + days[i].value + " ";  }  alert(values);  }  </script>  </body>  </html> |

**Assignment 5** Work in group up to 3 members to choose a web application project and fill in the following forms.

Group members (ID and name)

|  |
| --- |
| 1.  2.  3. |

Project title

|  |
| --- |
|  |

Objectives

|  |
| --- |
|  |

Functions

|  |
| --- |
|  |

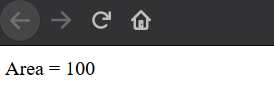
**JavaScript Classes (OPTIONAL)**

Classes must be declared before use unlike functions which can be declared anywhere.

Class declaration

|  |
| --- |
| class Rectangle {  constructor(height, width) {  this.height = height;  this.width = width;  }  } |

**Exercise 6** Using classes



|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Welcome</title>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  </head>  <body>  <script>  class Rectangle {  constructor(height, width) {  this.height = height;  this.width = width;  }  findArea() {  return this.width \* this.height;  }  }  var square = new Rectangle(10, 10);  document.writeln("Area = " + square.findArea());  </script>  </body>  </html> |

We can neglect the constructor by using the default one. Let’s see this example.

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Welcome</title>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  </head>  <body>  <script>  //Class declaration, must exist before usage  class Rectangle {  constructor(height, width) {  this.height = height;  this.width = width;  }  findArea() {  return this.width \* this.height;  }  setArea(height, width) {  this.height = height;  this.width = width;  }  }  var square = new Rectangle();  square.setArea(10, 10);  document.writeln("Area = " + square.findArea());  </script>  </body>  </html> |

In addition, we can inherit the superclass.

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Inheritance</title>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  </head>  <body>  <script>  //Class declaration, must exist before usage  class Rectangle {  constructor(height, width) {  this.height = height;  this.width = width;  }  findArea() {  return this.width \* this.height;  }  setArea(height, width) {  this.height = height;  this.width = width;  }  }  class Square extends Rectangle {  constructor(width) {  super(width, width);  }  }  var square = new Square(10);  document.writeln("Area = " + square.findArea());  </script>  </body>  </html> |

**JavaScript ES6 Arrow function (OPTIONAL)**

Normal function

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Functions</title>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  </head>  <body>  <script>  function square(num) {  return num\*num;  }  var answer = square(5);  document.writeln(answer);  </script>  </body>  </html> |

ES5 function

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Functions ES5</title>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  </head>  <body>  <script>  //In ES5 we can write  const square = function (num) {  return num\*num;  }  var answer = square(5);  document.writeln(answer);  </script>  </body>  </html> |

ES6 Arrow function

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Functions ES5</title>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  </head>  <body>  <script>  //In ES6 we can write  const square = (num) => {  return num\*num;  }  //if only single function statement, {} and return can be neglected  const square1 = (num) => num\*num;  var answer = square(5);  var answer1 = square1(5);  document.writeln(answer + ", " + answer1);  </script>  </body>  </html> |