Day 1

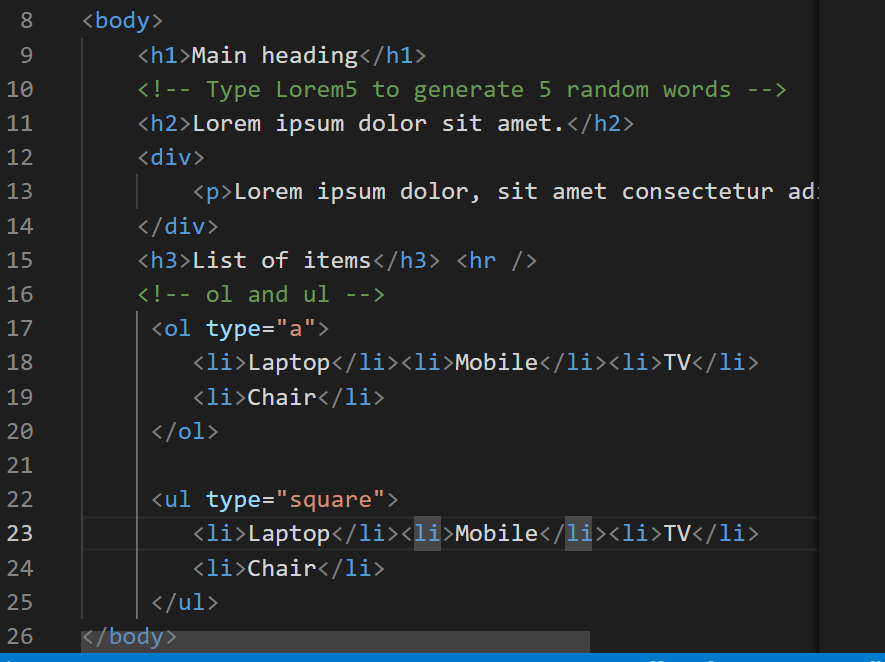
HTML, CSS & SASS/SCSS

HTML: Hyper Text Markup Language - Display the content

CSS - Cascading Style Sheet - Style

HTML file

Lists & headings



Output:



You can create HTML tables, forms and other elements

CSS:

It is used mainly to add the styles to HTML elements

RWD

It helps to fit your web page to all types of devices it could be desktop, mobiles, printing devices, tablets

Same HTML content to render differently in different devices

Medi Query: it applies the styles based on the conditions.

Grid: It arranges the elements in row & column format

Flex: It automatically arranges the elements based on the size like horizontally or vertically.

{display: flex }

SASS

Syntactically Awesome Style sheets

It avoids lot of boiler plate code which you write in CSS

SASS -> SCSS (this resembles css syntax), but sass follows indentation syntax

SASS syntax  
$font-style: arial;  
$color: blue

h1   
 color: $color;  
 font-family: $font-style

SCSS syntax:

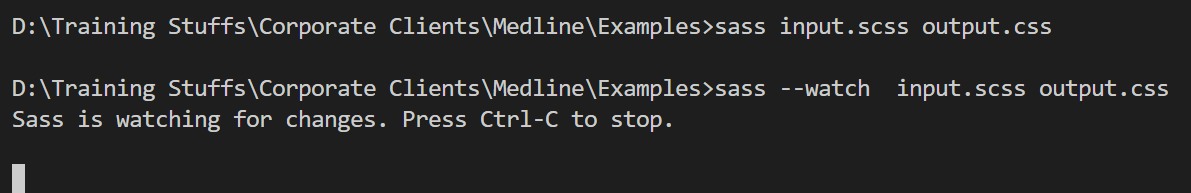
$color: blue;  
$font-style: arial;  
h1 {   
 color: $color;  
 font-family: $font-style;  
}

Install sass

npm install -g sass

To compile scss to css

sass input.scss output.css



--watch lets you to automatically compile scss to css

Mixins

These allow you to reuse the group of CSS declarations.

@mixin shape($color, $width, $height) {   
 width: $width;  
 height: $height;  
 background-color: $color;  
}   
.square {  
 @include shape(blue, 400px, 400px);   
}  
.rectangle {  
 @include shape(red, 400px, 200px);  
}

Parent selector

It is used in nested selector to refer the outer selector

.alert {  
 &:hover { color: red }  
}

In CSS you write

alert : hover { color: red; }

Placeholder

It is to reuse some selectors using inheritance

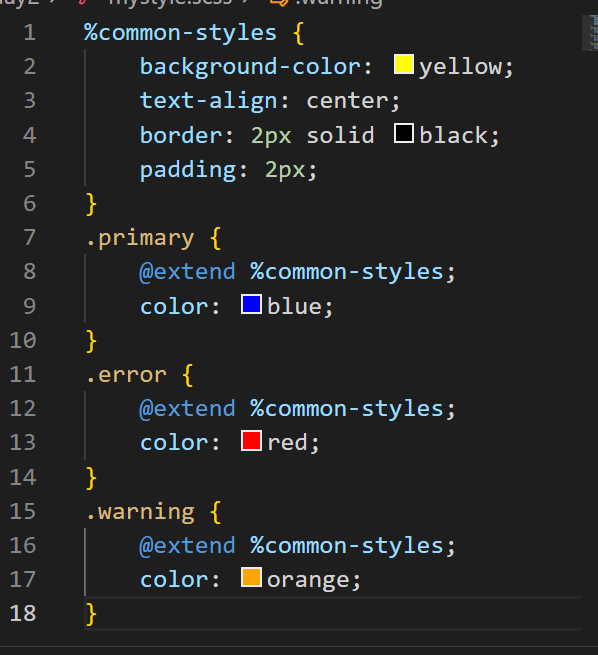
%name {  
 // css selectors  
}

.primary {  
 @extend %name;  
 color: blue;  
}  
.secondary {  
 @extend %name;  
 color: grey;  
}

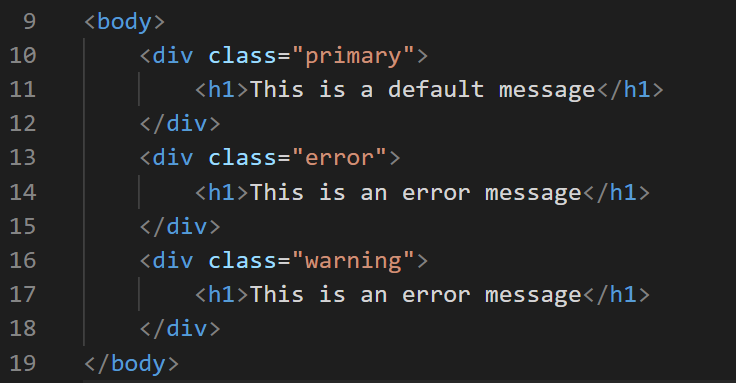
How to compile multiple SASS files

sass --watch file1.scss : file1.css file2.scss : file2.css

day2/mystyle.scss



index.html



Output:



Special functions:

SCSS provides various functions to perform complex operations

$colors: red, green, blue;  
.error { color : nth($colors, 1) }

.warning { color: nth($colors, 2) }

Mixing colors using mix()

.mixed-color { color: mix(red, green, 60%) }

Here 60% red & 40% green

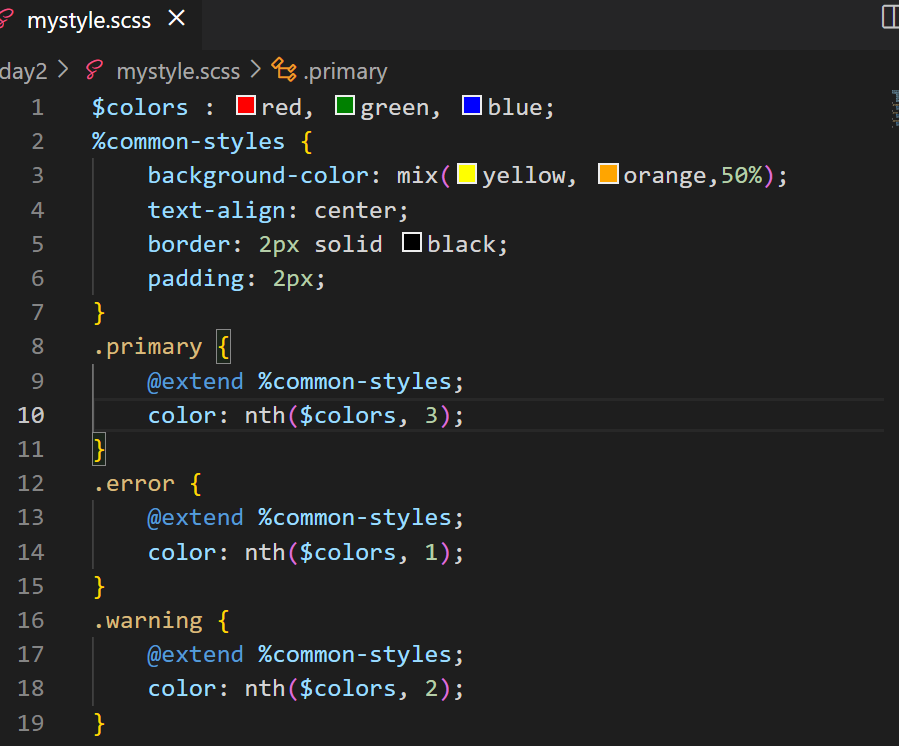
Note: You can only mix two colors, however you can get the reference of two colors and pass the color again to the mix

mix(mix(red, green, 60%), blue, 50%)

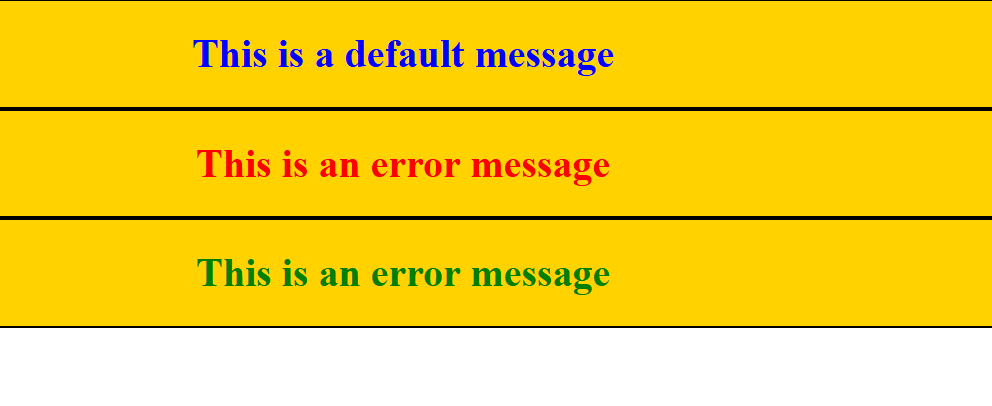
Round up the numbers

font-size: round(19.6px); // font-size: 20px

scss



Output:



SASS official document

@if & @else if : these are used to apply the CSS on a conditional based

!default: to provide a default values when the property doesn’t have a value

<https://sass-lang.com/guide/>

Javascript

HTML - displaying the content

CSS - styling the HTML

Javascript - It adds effects the web page by dynamically accessing HTML & CSS, it has various features like

* variables
* operators
* datatypes
* functions
* objects
* arrays
* conditional statement
* loops
* form validations

There are two ways you can execute Javascript

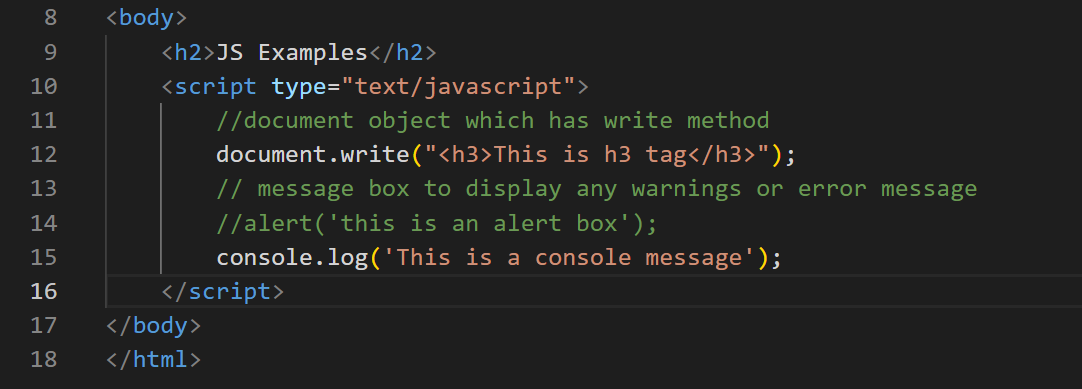
1. Using browser at the front-end
2. Using Node.js at the backend

Strings in Javascript: You can either use single or double quotes

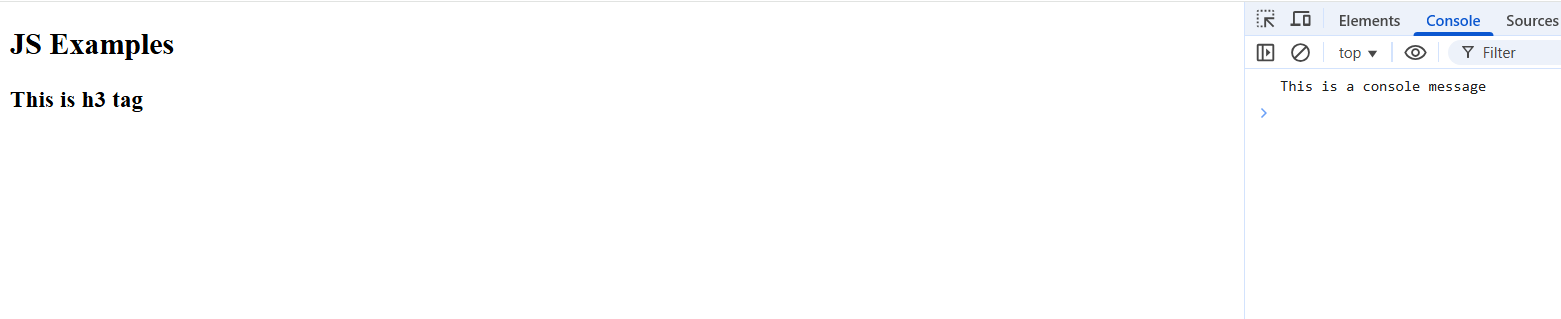
Note: single quote is not treated as character it is treated as string itself

Note: semicolon is optional in Javascript as long as you keep the statements in the new line

JS hello world program



Output:



Fundamentals of Javascript

1. Variables
2. Datatypes
3. Operators
4. Conditional Statements
5. Loops

Variables: It is to store the data in Javascript

old approach is to use var keyword

var employeeId = 1234;  
var employeeName = “Siddharth”;

I can assign different values to the same variable with different types

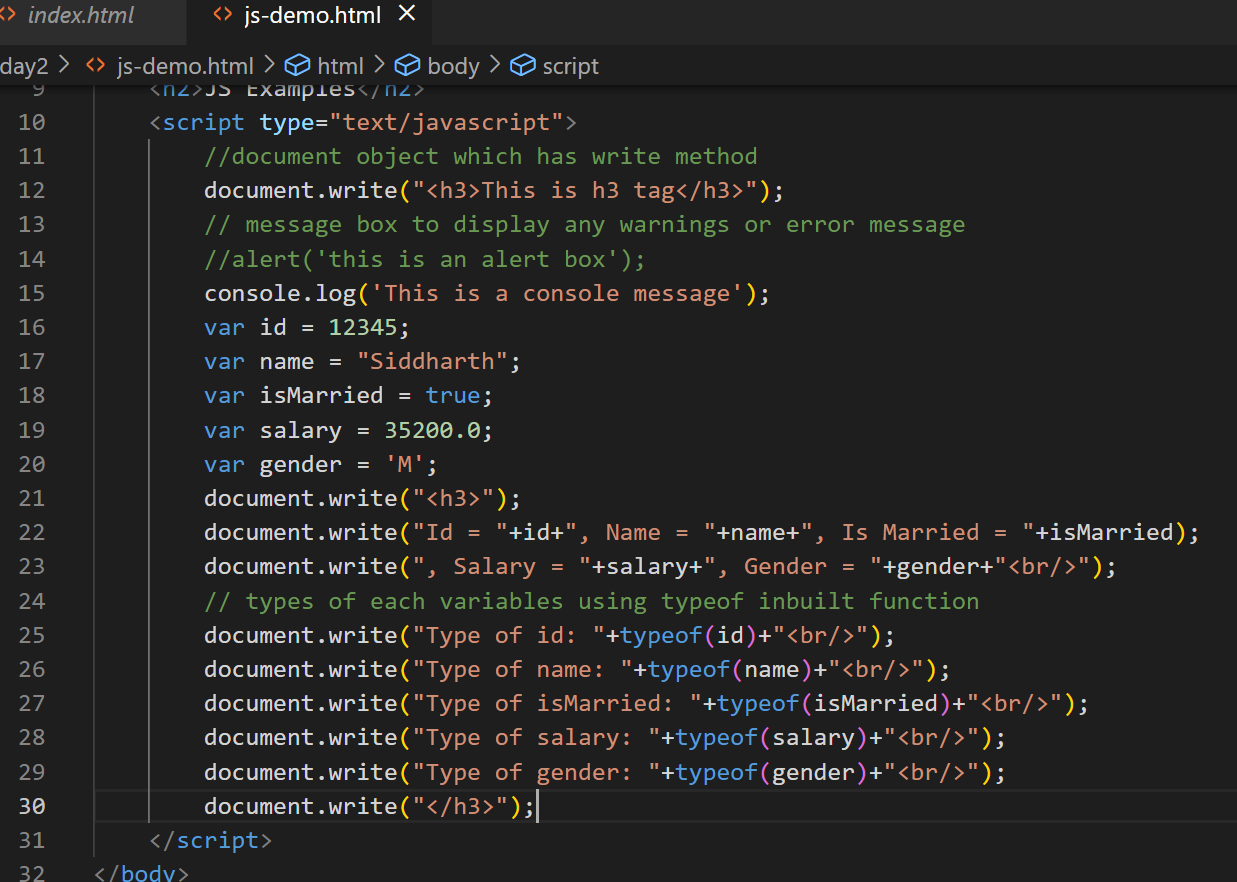
employeeId = “12345” or employeeId = true

Datatypes in Javascript

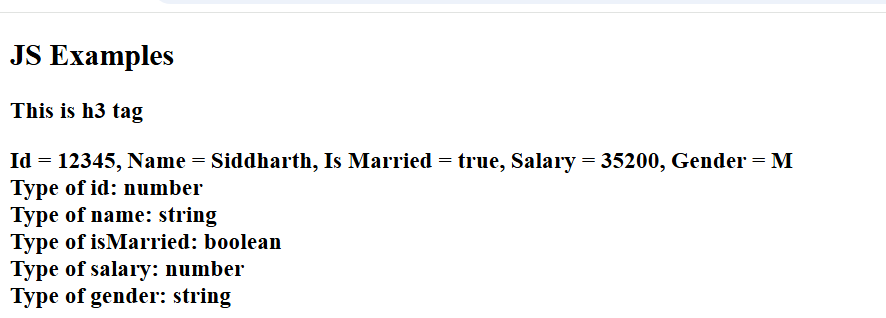
In Javascript variables get the types based on the type of value, you can use an inbuilt function in javascript typeof(variableName) to get the datatype

document.write(typeof(employeeId); // number or string or boolean

jsdemo.html



Output:



Javascript operators

Arithmetic operators: +, -, \*, /, %

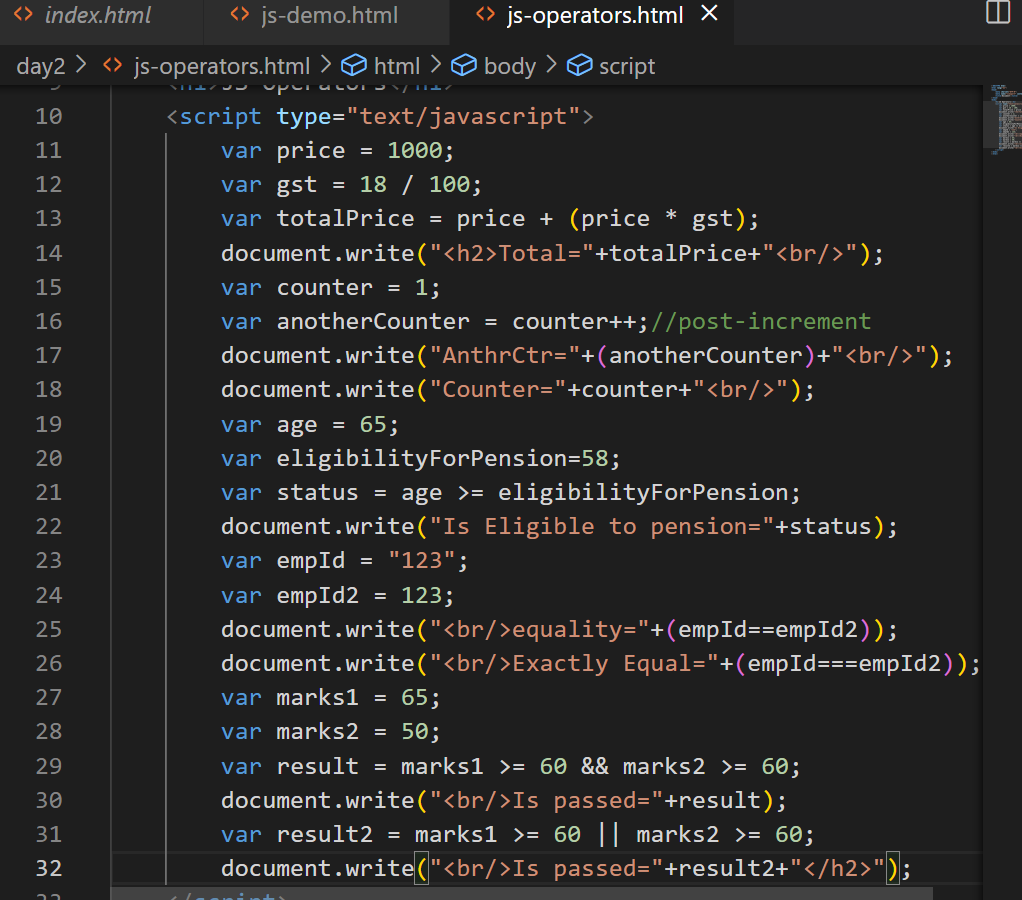
Assignment operator: =

Comparison operators: <, >, <=, >=, !=, ==, ===

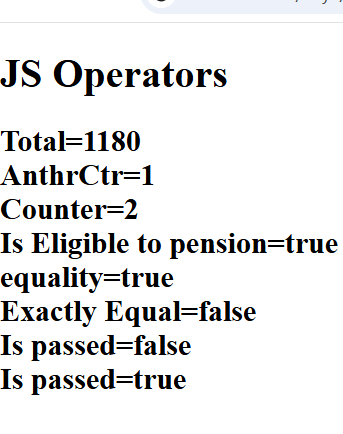
Increment & Decrement operators: ++, --

Logical operators: &&, ||

Operators Demo



Output:



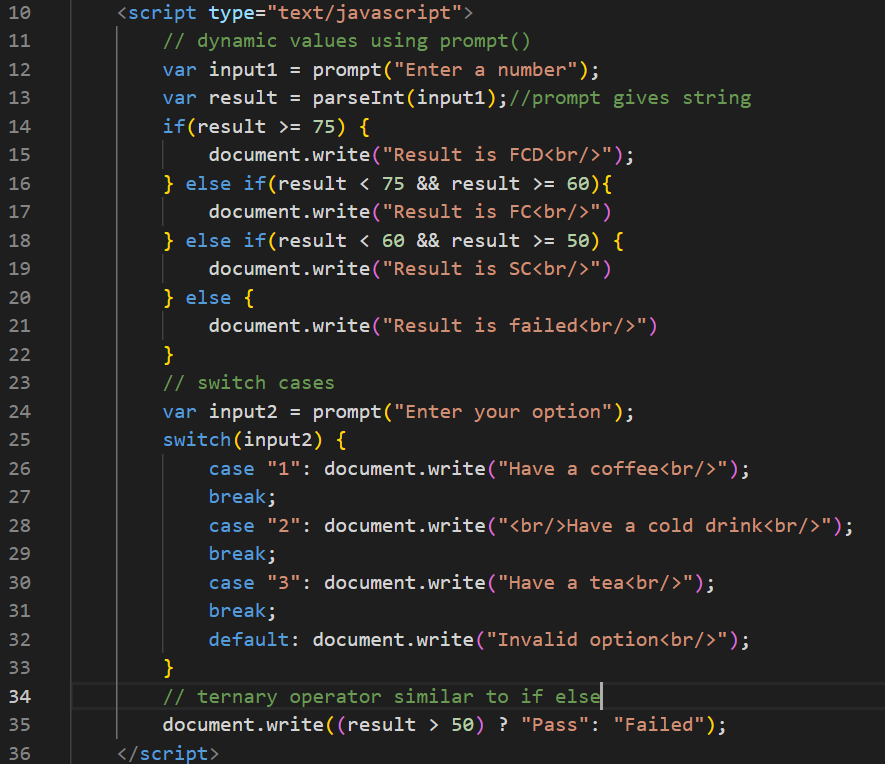
Conditional statements:

Statements that you want to execute only if its true or false

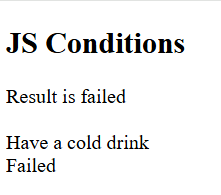
In Javascript we have following conditional statements

1. if
2. if else
3. if else if else if … else
4. switch

Conditional Statements demo



Output:



Day 3 agenda

**Loops**

* While Loop
* For Loop
* For...In Loop
* Loop Control

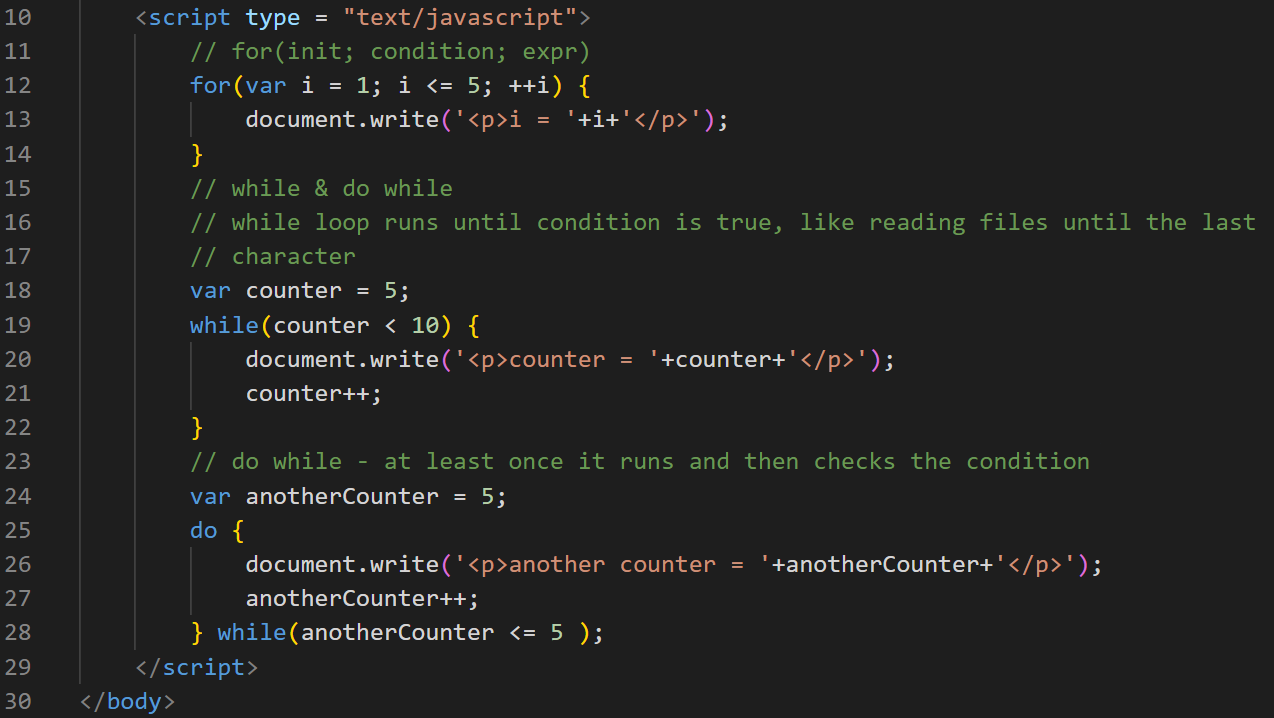
**JavaScript Functions & Objects**

* JavaScript Functions
* The Object Constructor
* Defining Methods for an Object
* JavaScript Boolean
* JavaScript String
* JavaScript Arrays
* JavaScript Arrays Method
* JavaScript Arrays Sort

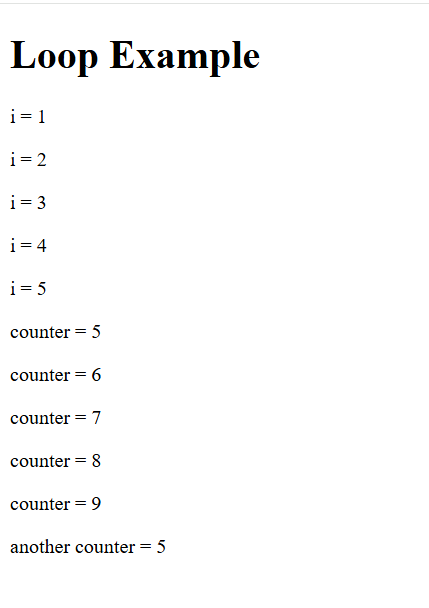
Loops: Same statements to execute until certain condition is true, there 3 types of loops

1. for loop
2. while
3. do-while

for loop: When you want to iterate fixed set of iterations then you can use for loops, it is used while reading array elements also.



Output:



Objects in Javascript

In Javascript you can create objects that will have properties & functions/methods, there are many ways you can create objects in Javascript

1. Literal style
2. functional style with object constructor
3. class style (modern syntax)

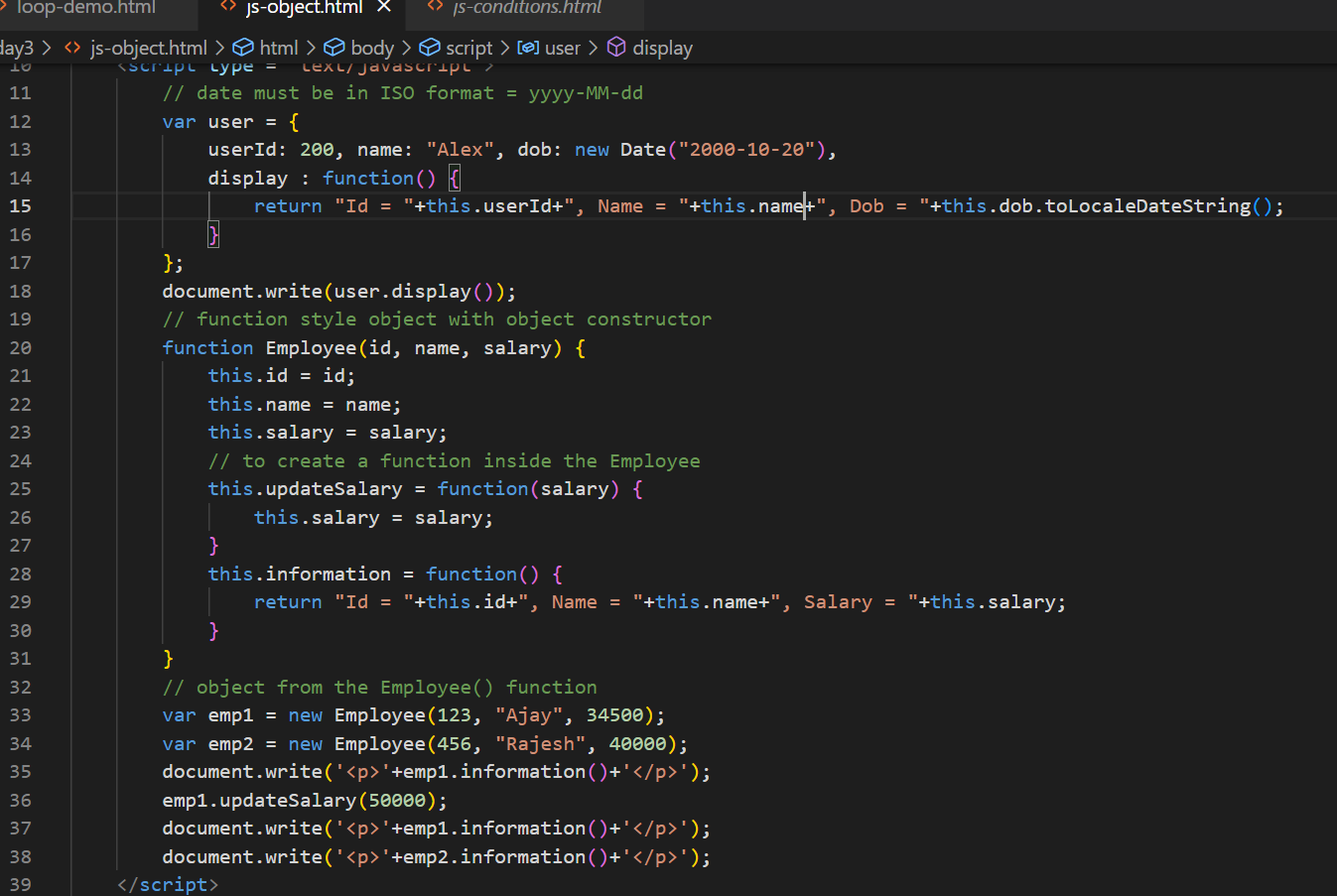
Literal style: This is better when you want one or two objects to be created or when you want some mock objects to be created

{ property : value, property : value, property : function() { } }

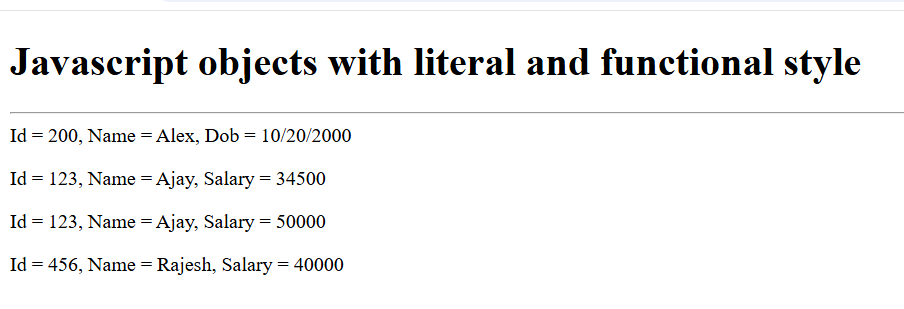
Function style: When you want multiple objects to be created with the same template then this is better, this style is more preferred in Javascript

function FunctionName(parameters) { … }

js-objects.html

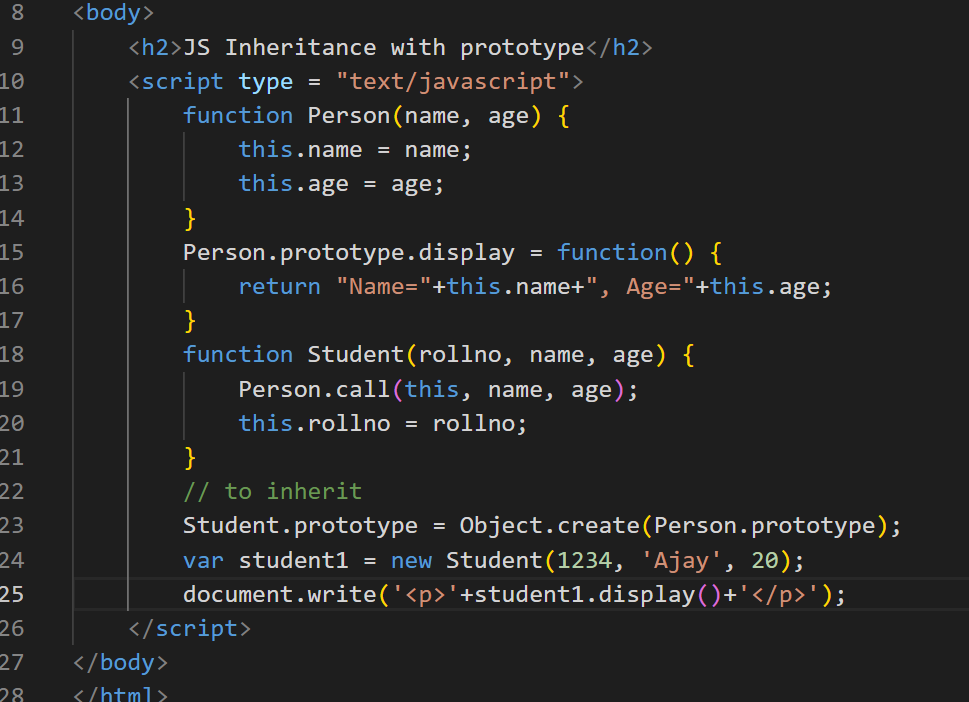


Output:



How to achieve inheritance in javascript

We can achieve inheritance with prototype if you are using old syntax, and a newer syntax uses classes & extends keyword



Output:



Arrays: It is a container object to store multiple values of same types or different types (in javascript it is allowed)

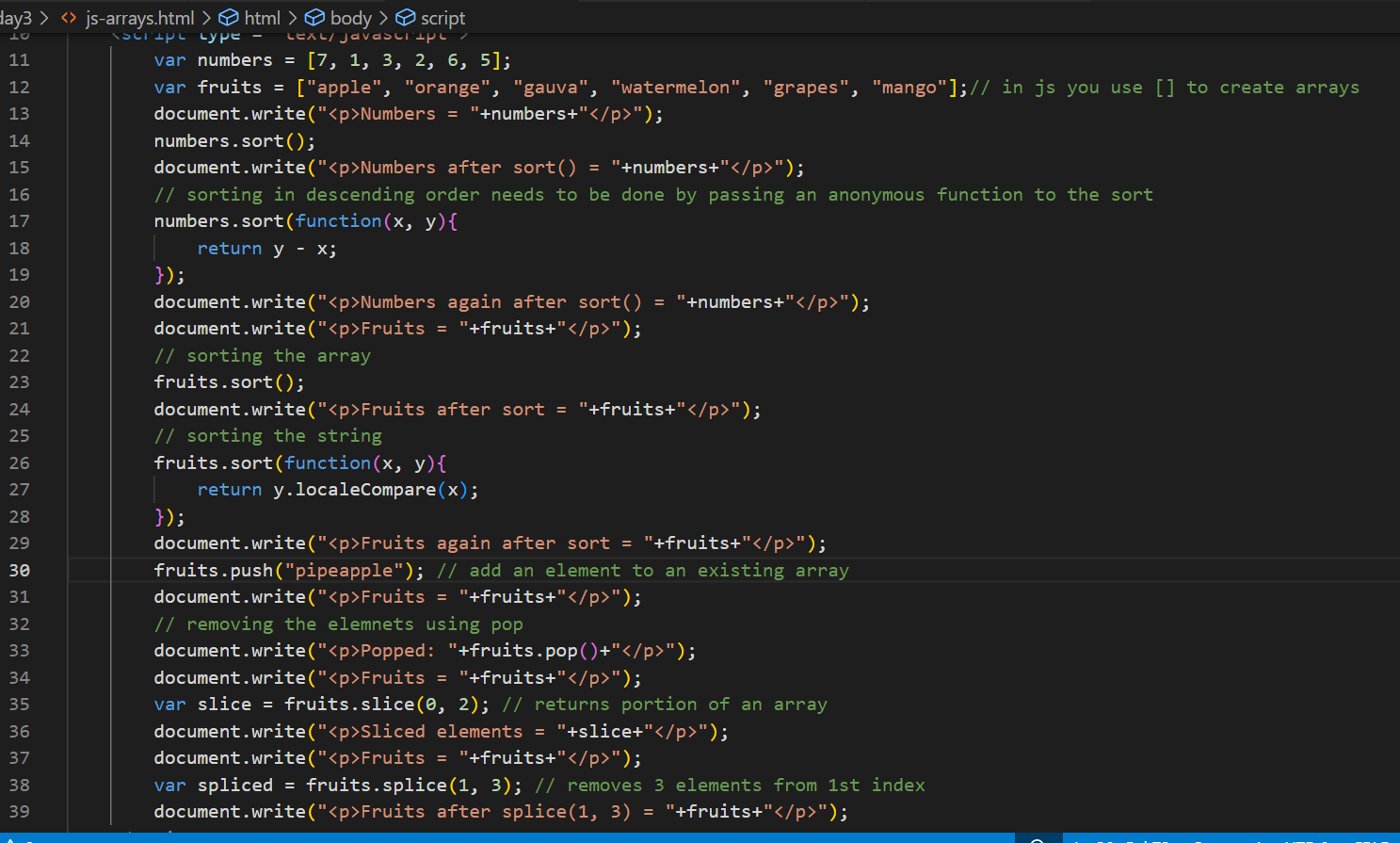
You can create simple to complex arrays

In Javascript you have lot of inbuilt functions in arrays like

push(): to add the element in the array  
pop(): to delete the element from the array  
sort(): to sort the elements  
slice()  
splice()

sort(): by default it sorts in ascending order, to sort in descending order you must use a function as a parameter that returns an int value to the sort to arrange the elements

sort(function(x, y){   
 return   
})  
Arrays methods & sorting with comparator anonymous function

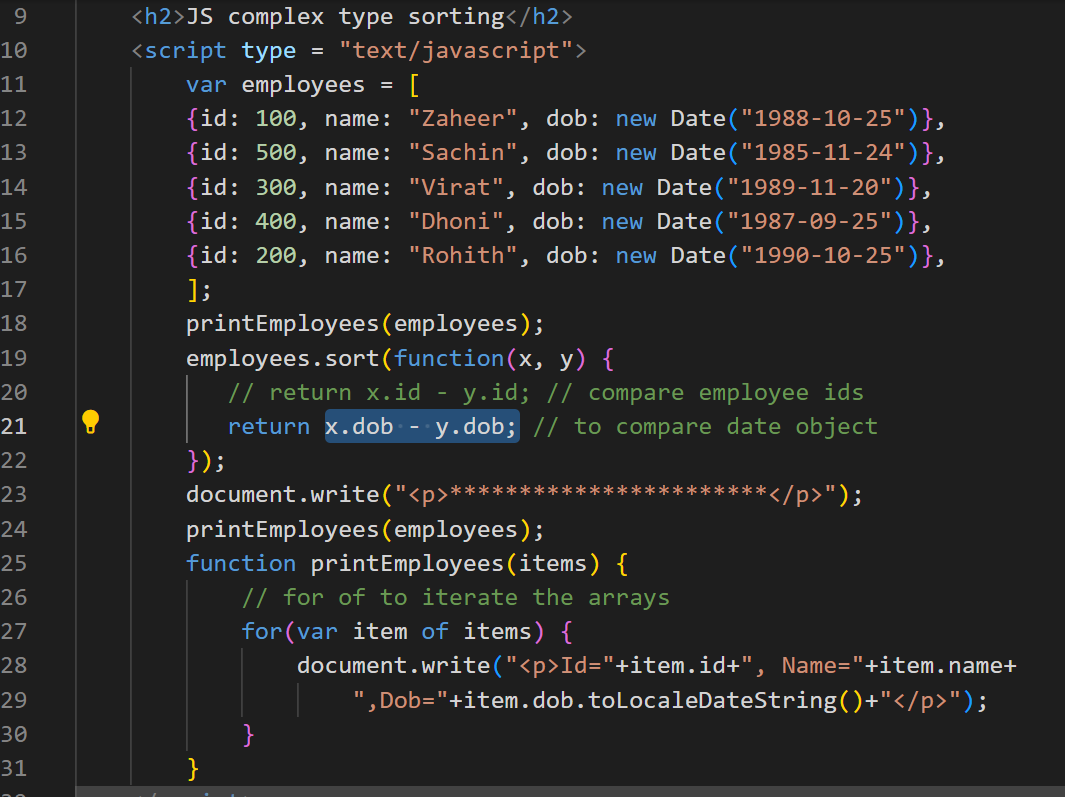


Output:

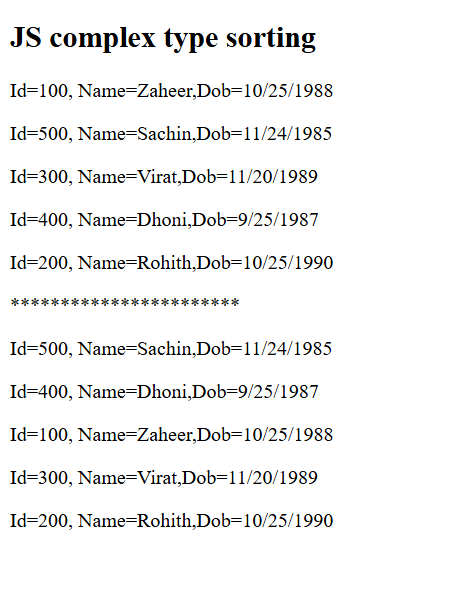


How to sort complex objects

You must use the same anonymous function and access the property of the objects and compare them.



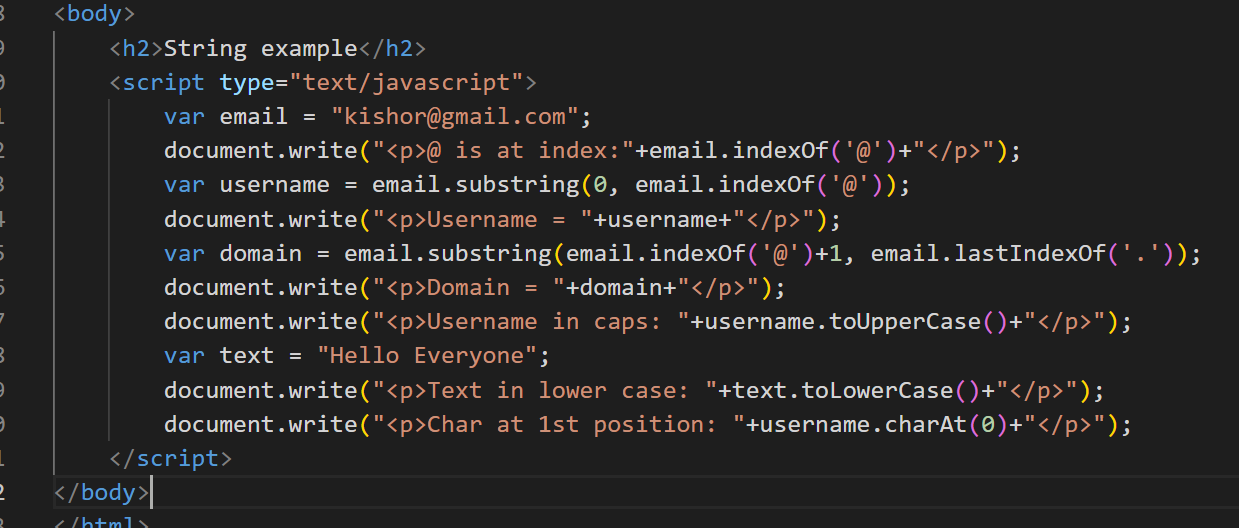
Output:



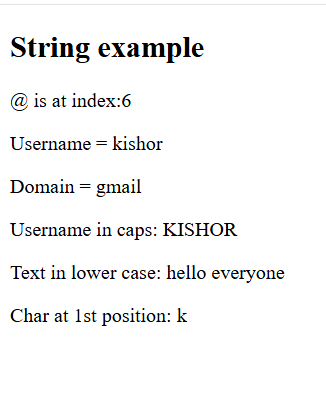
String methods:

In String there are various inbuilt methods like-

toUpperCase(), substring(), concat(), indexOf(), lastIndexOf()



Output:



Day 4 Agenda

**JavaScript HTML DOM**

* Introduction
* DOM Elements
* DOM EventListener

**JavaScript Form Validation**

* Create Form
* Styling Form
* Validate Username Input
* Validate E-Mail Input
* Validate Password & Radio Button

For in Loop

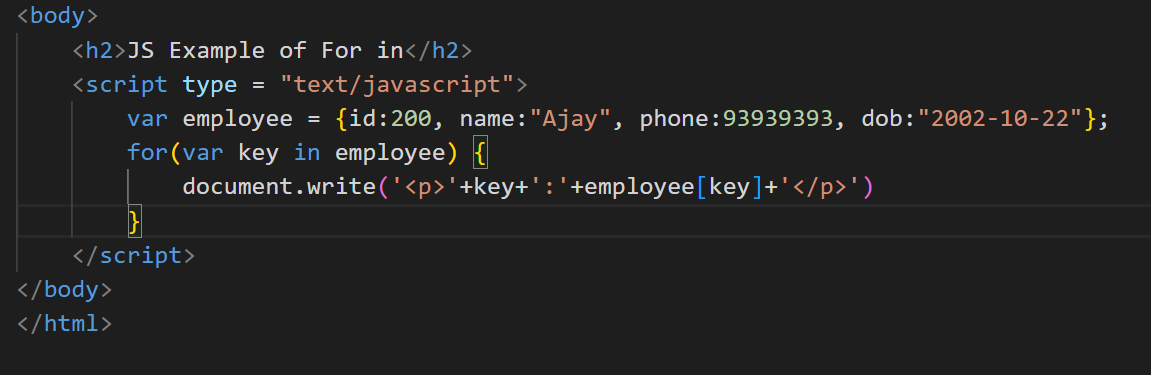
It is used to iterate the properties of an object.

let employee = { id:101, name:”Raj”, phone:939933, dob:”2004-10-22” };

let name = employee[“name”];

for(var key in employee) {   
 // key will be the property of an object

employee[key]  
}

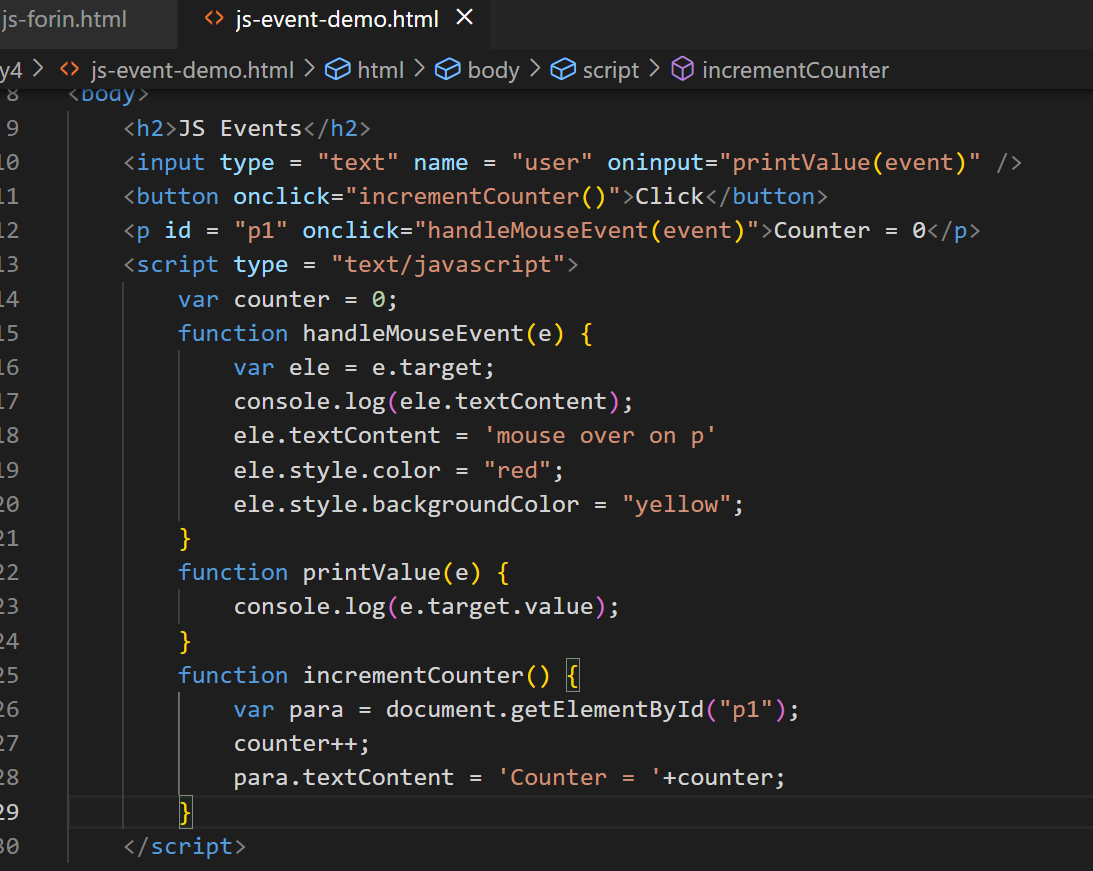


DOM events

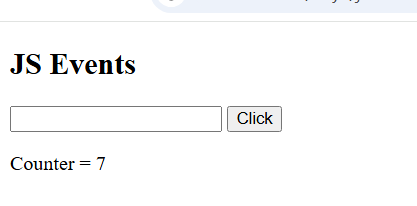
DOM stands for Document Object Model, it is a tree structure of HTML elements loaded in the browser, HTML elements would generate events when you do something with the element like mouseover, click, onchange, onsubmit

<button onclick=”fun()”>Click</button>

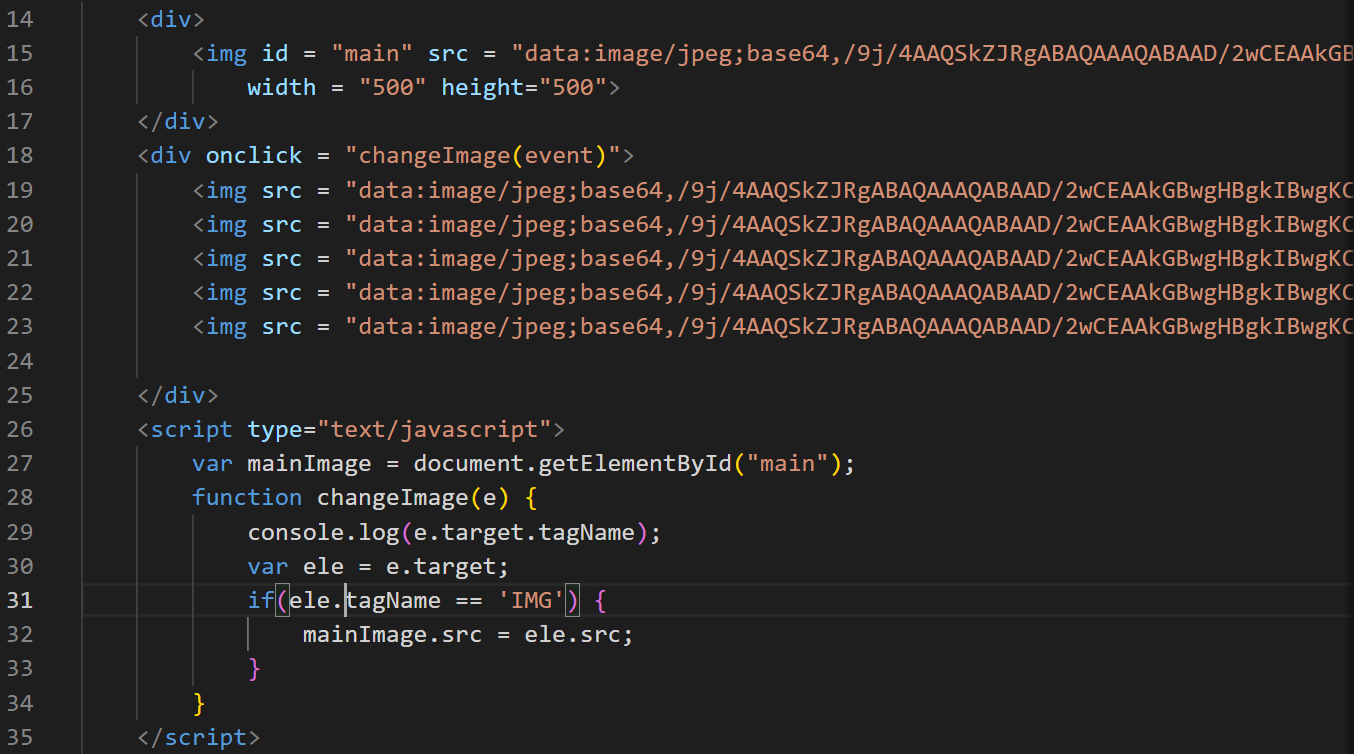
js-event-demo.html



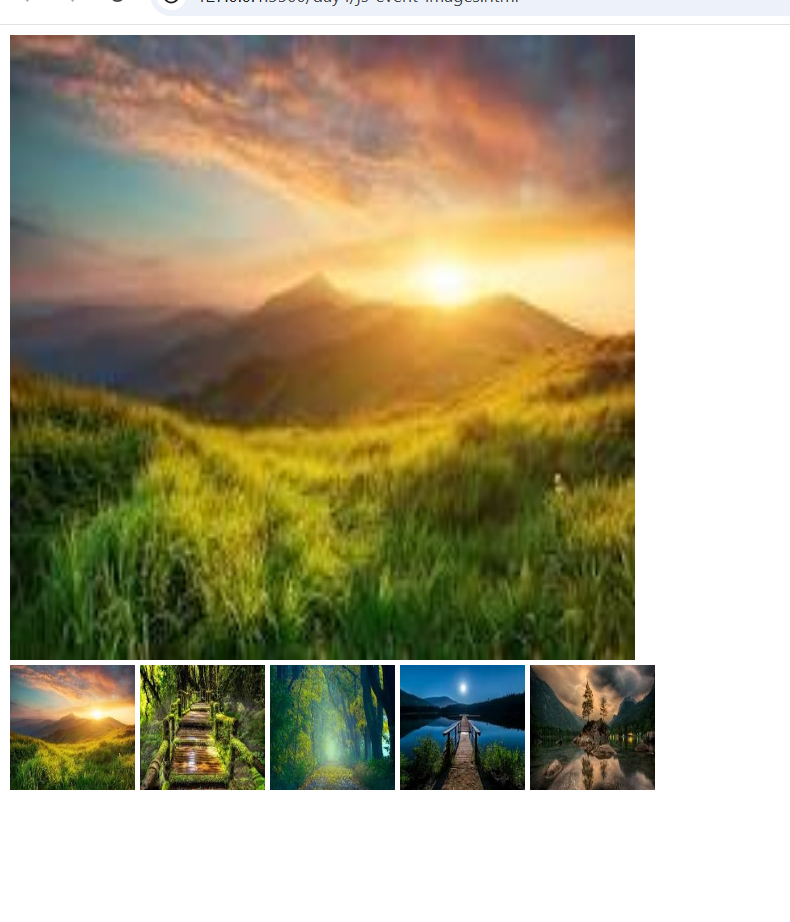
Output:



Changing the images based on the image you select



Output;



List of ways to access the element

1. document.getElementById(“id”): Pass the id to get the element
2. event.target: Get the element who generated the event
3. document.getElementsByTagName(“tag”): You get all the elements having the tag name in the form array
4. document.getElementsByClassName(“className”): You get all the elements having the class name in the form array
5. document.querySelector(“selector”): You can use a single method to get the elements by passing the id or class name or tag name.
6. document.querySelectorAll(“selector”): You can get all the elements having the same selector

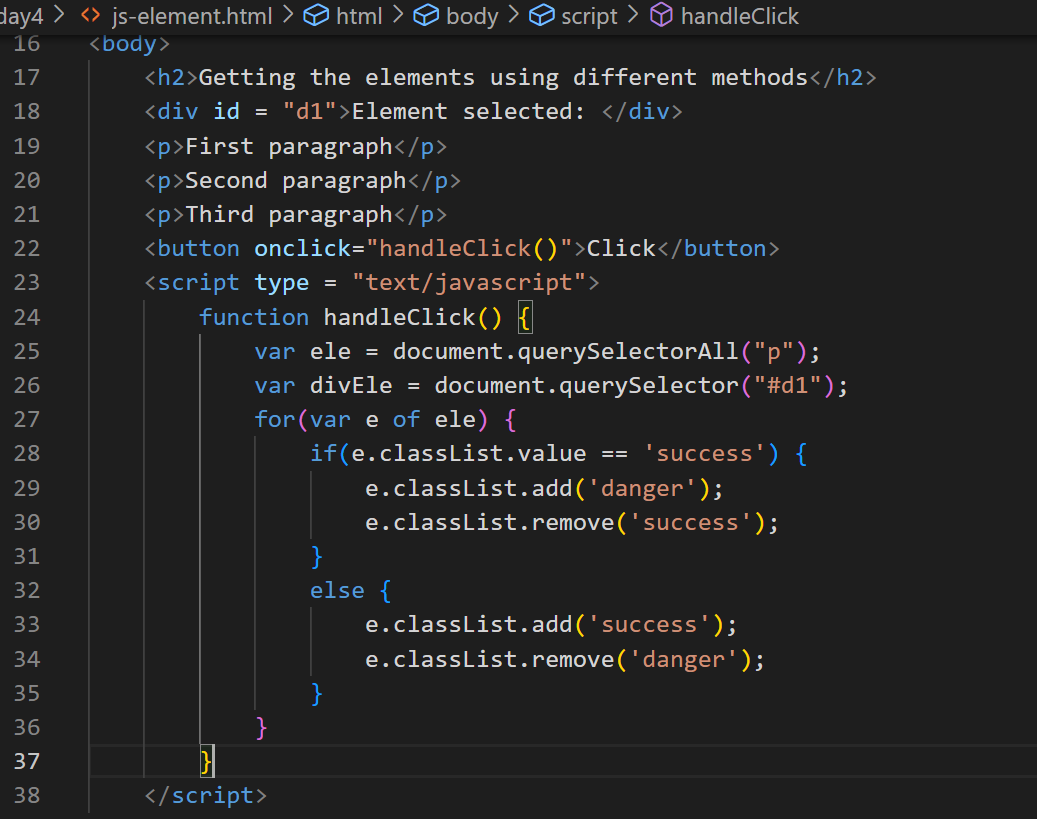
ex: document.querySelector(“#p1”): gets the element whose id is p1d

ex: document.querySelector(“.p1”): gets the elements which has the class name p1

ex: document.querySelectorAll(“.p1”): get array of elements using the same class name

ex: document.querySelectorAll(“p”): gets all the p

ex: document.querySelector(“p”): gets the first p element in the DOM



Output:

