HTML, CSS & JavaScript

Description

HTML, Structure, Common Elements, Rendering the DOM

Objectives

- HTML, Structure, Common Elements, Rendering the DOM

Introduction to HTML and Strucutre

Before we introduce an HTML itselfs let us start with the the term web-browser. The web-browser is the required application that runs on the target system. Web browser is responsible for downloading the desired content from the specified location and rendering it a user’s views. The location may be defined by the link somewhere on the internet that is reachable, or it could be the link to the local storage file. The contents of the local file or provided link is downloaded by the web-browser client. After the finished download a web-browser attempts to interpret the downloaded content as the user view, it means render.

The content requires a specific form, structure, that a web-browser is able to interpret it. The structure is specified by the HyperText Markup Language, shortly HTML. HTML may contain other languages or tags, such as JavaScript, CSS or etc. It meant that the installed web-browser requires the user to contain appropriate interpreters to properly render the content.

Let us start with the HTML language and dive a bit deeper into its semantics. The HTML language is interpreted by a web-browser, it has a defined structure described by elements (Example 5.1). The element is the building block of the HTML. The HTML language contains a series of such elements such as, html, head, body p1 and ect. Those elements tell the web-browser how to render the content.

```
<!DOCTYPE html>
<html>
<head>
    <title>Title: Welcome HTML</title>
</head>
<body>
    <h1>Heading: Welcome to HTML</h1>
    <p>Paragraph: Hello HTML <b>world!</b></p>
</body>
</html>
```

Example 5.1: Simple HTML content structure.

Understnading HTML structures

Most of the HTML tags work in pairs, **paired-tags**. Some of the tags may be used as **singular tags** (unpaired tags). Singular tags means that it doesn not have a closing.

1. **paired tag** example: `<b>`...`</b>`
2. **singular tag** example: `<br>` or `<img/>`

Each valid HTML continent must start with `<!DOCTYPE>` identifier. The doctype is not a html tag. It gives a web-browser information that the content should be interpreted as the HTML web-page (Example 5.1, first line).

The doctype tag is followed by the paired tag `<html>` which represents the root of the HTML document and is closed by the closing tag `</html>`. (Example 5.1, `<html></html>` tags)

The `<HTML>` tag,te representation of the document, has two parts:

1. **header** - paired tag `<head>`
2. **body** - paired tag `<body>`

The **header** structure contains information about the data that is rendered to the user’s view. Basically the header contains information about rendered content. Header may contain following tags:

- Singular tags: `<base>`, `<link>`, `<meta>`
- Pair tags: `<title>`, `<style>`, `<script>`, `<noscript>`

The **body** section contains information that is rendered to the user view. The section contains data with tags, eg. paragraphs, headings, images, tables etc., to modify displayed content view(Example 5.1)

Understand common HTML Elements

The following section is dedicated to the commonly used tags which resides in the body section. Those help to manage modify the texts rendering. Each of following tags is paird tag, it means it requires to have a closing tag (Example 2.)

`<tag_name>`….`</tag_name>` Example 2.: Paired body tag structure

Body Tags:

- <h1>to<h6> : Headings
- <p> : Text paragraph
- <img> : Insert image tag

Text formatting tags:

- <strong>, or <b> : bold text formatting
- <i>, <em> : italic text formatting
- <small> : small text size formatting
- <del> : strike a text formatting
- <ins> : underling a text formatting
- <mark> : higlighting text formatting
- <sup> : supercript text formatting
- <sub> : subscript text formatting

Table formatting tags:

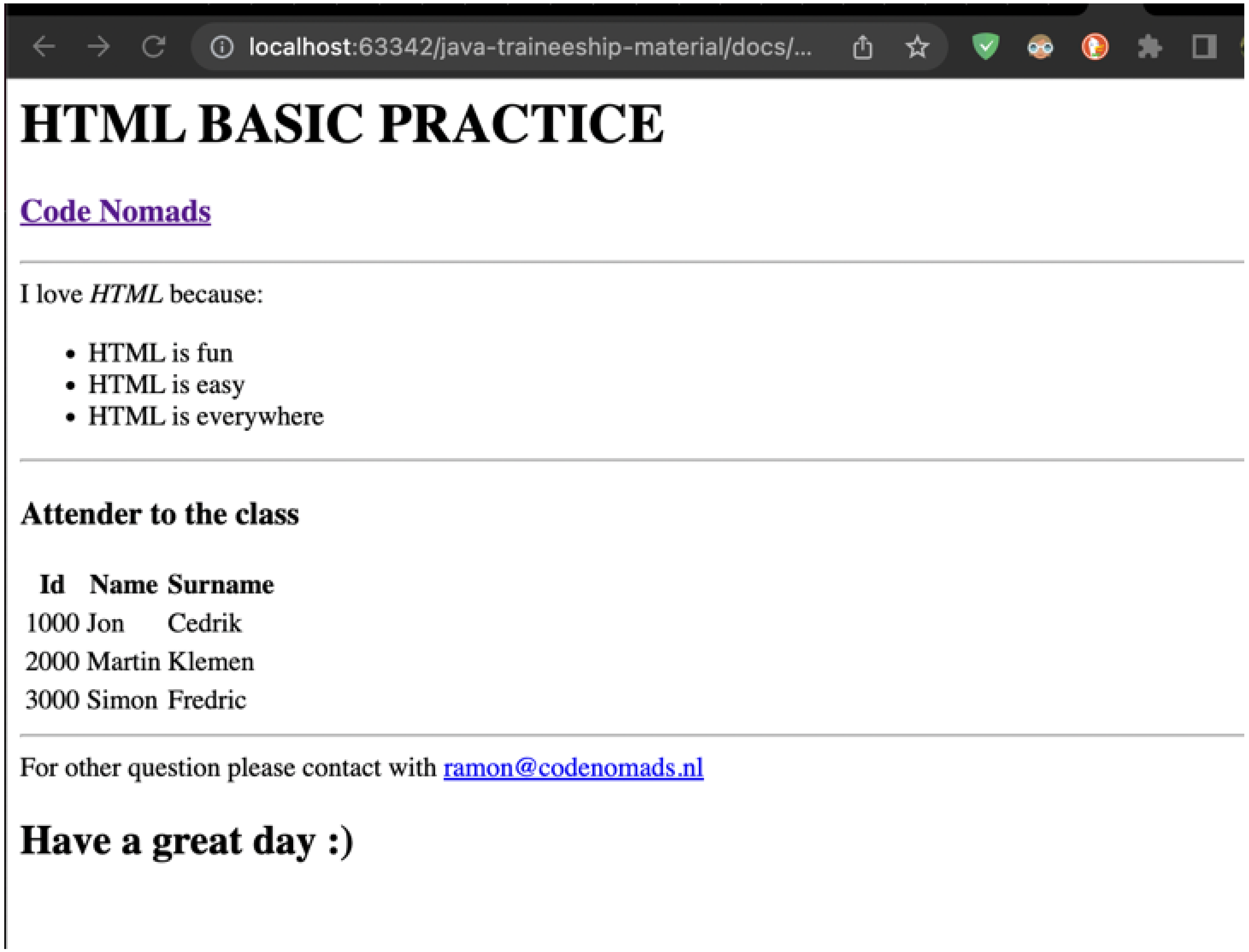
- <table> : defined a table tag
- <tr> : define a table row
- <th> : defien a table header
- <td> : define a table cell or data

Exercises

1. Write a program to create a webpage to print "I have started learning HTML" in body
2. Write a program to create a webpage to print “Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. ” in paragraph tag

3. Write a program to show "apple, orange, banana, grape, melon" in the list.
4. Write program to print ‘REGISTRATION FORM’ in first line with Header tag. On second line print name with strong tag and put input text next to it.
5. Write a program that shows image of given source "https://images.freeimages.com/images/large-previews/e04/yellow-frontal-with-ivy-1228121.jpg"
6. Write a program that gives a link to the https://www.w3schools.com/html/default.asp
7. Write a program that shows a table with 3 columns. First column is Id, second is name and third is surname. Create a table to print 4 of your friends' id, name, surname in it.
8. Write a program to show 3 button of cancel, add, delete.
9. Look at the image of the web page below.
    - Using HTML coding, replicate the page exactly.
    - The hypertext links for the codenomads and ramon@codenomads.nl must be working links.
    - Don’t forget to include the title by putting the phrase “I Love HTML!!!” in your title tag.

Tip: There is one of each in the coding – H1, H2, and H3. Do not use the font size and bold tags. The title tag should not be bold, even thought looks bold in the tab on the screenshot.



localhost:63342/java-traineeship-material/docs/...

HTML BASIC PRACTICE

Code Nomads

I love *HTML* because:

- HTML is fun
- HTML is easy
- HTML is everywhere

Attender to the class

Id	Name	Surname
1000	Jon	Cedrik
2000	Martin	Klemen
3000	Simon	Fredric

For other question please contact with ramon@codenomads.nl

Have a great day :)

Module: HTML, CSS & JavaScript

Introduction

HTML, Structure, Common Elements, Rendering the DOM

What is CSS

CSS is a rule-based language — you define the rules by specifying groups of styles that should be applied to particular elements or groups of elements on your web page.

How it looks like

For example, you can decide to have the main heading on your page to be shown as large red text. The following code shows a very simple CSS rule that would achieve the styling described above:

```
h1 {
    color: red;
    font-size: 5em;
}
```

How it looks like

- In the above example, the CSS rule opens with a selector . This selects the HTML element that we are going to style. In this case, we are styling level one headings (<h1>).
- We then have a set of curly braces { }.
- Inside the braces will be one or more declarations, which take the form of property and value pairs. We specify the property (color in the above example) before the colon, and we specify the value of the property after the colon (red in this example).

- This example contains two declarations, one for color and the other for font-size. Each pair specifies a property of the element(s) we are selecting (<h1> in this case), then a value that we'd like to give the property.

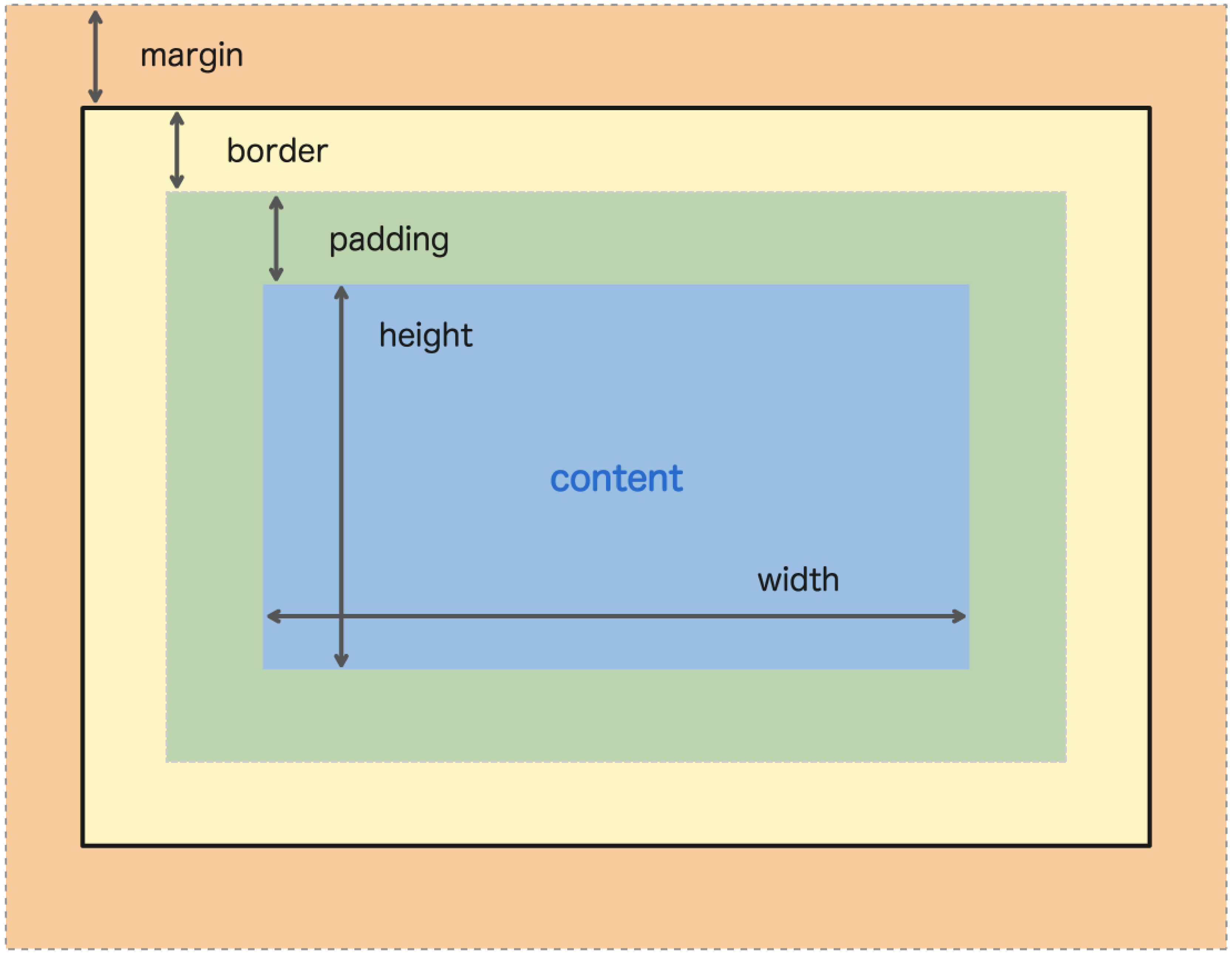
How it looks like

A CSS stylesheet will contain many such rules, written one after the other.

```
h1 {
    color: red;
    font-size: 5em;
}

p {
    color: black;
}
```

How it looks like


margin
border
padding
height
content
width

Example:

```
h1 {
    margin: 20px;
    paddig: 15%;
    border-width: thick;
}
```

Multiple ways to refer to html elements

```
<p class="example-text" id="example-text1">This is an example paragraph text</p>
```

By element

```
p {
    ...
}
```

By class attribute (everywhere this class is used)

```
.example-text {
    ...
}
```

or (everywhere this class is used on <p> elements)

```
p.example-text {
    ...
}
```

By id attribute

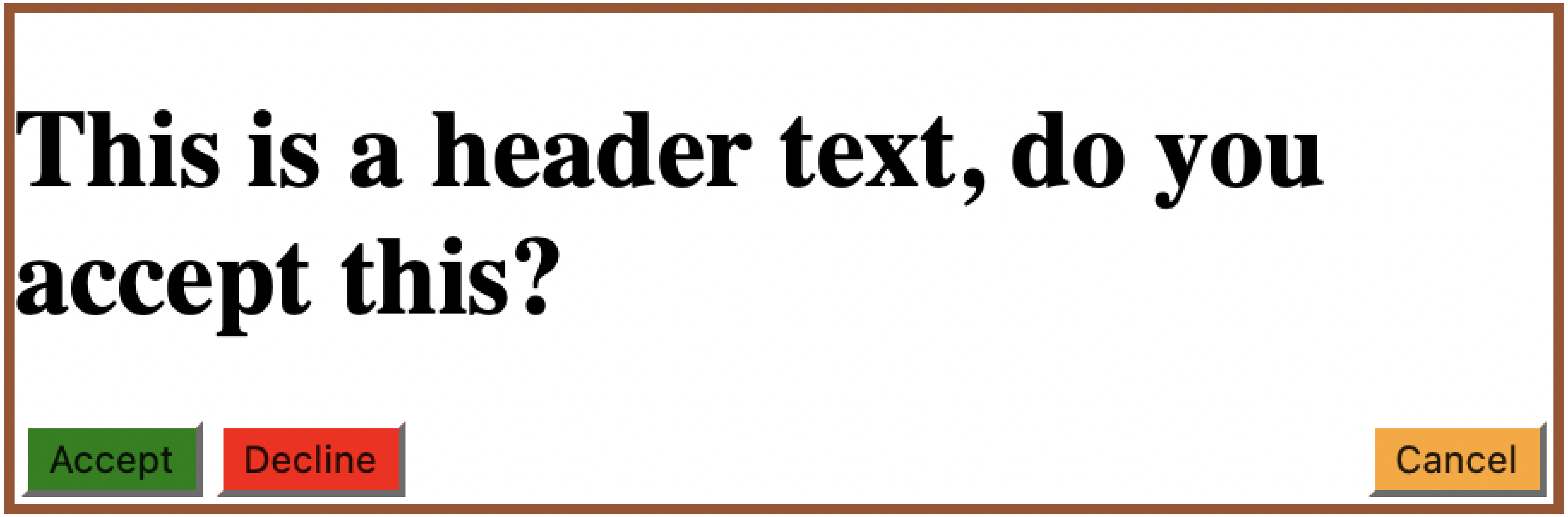
```
#example-text1 {
    ...
}
```

And many other ways!

Valuable resource

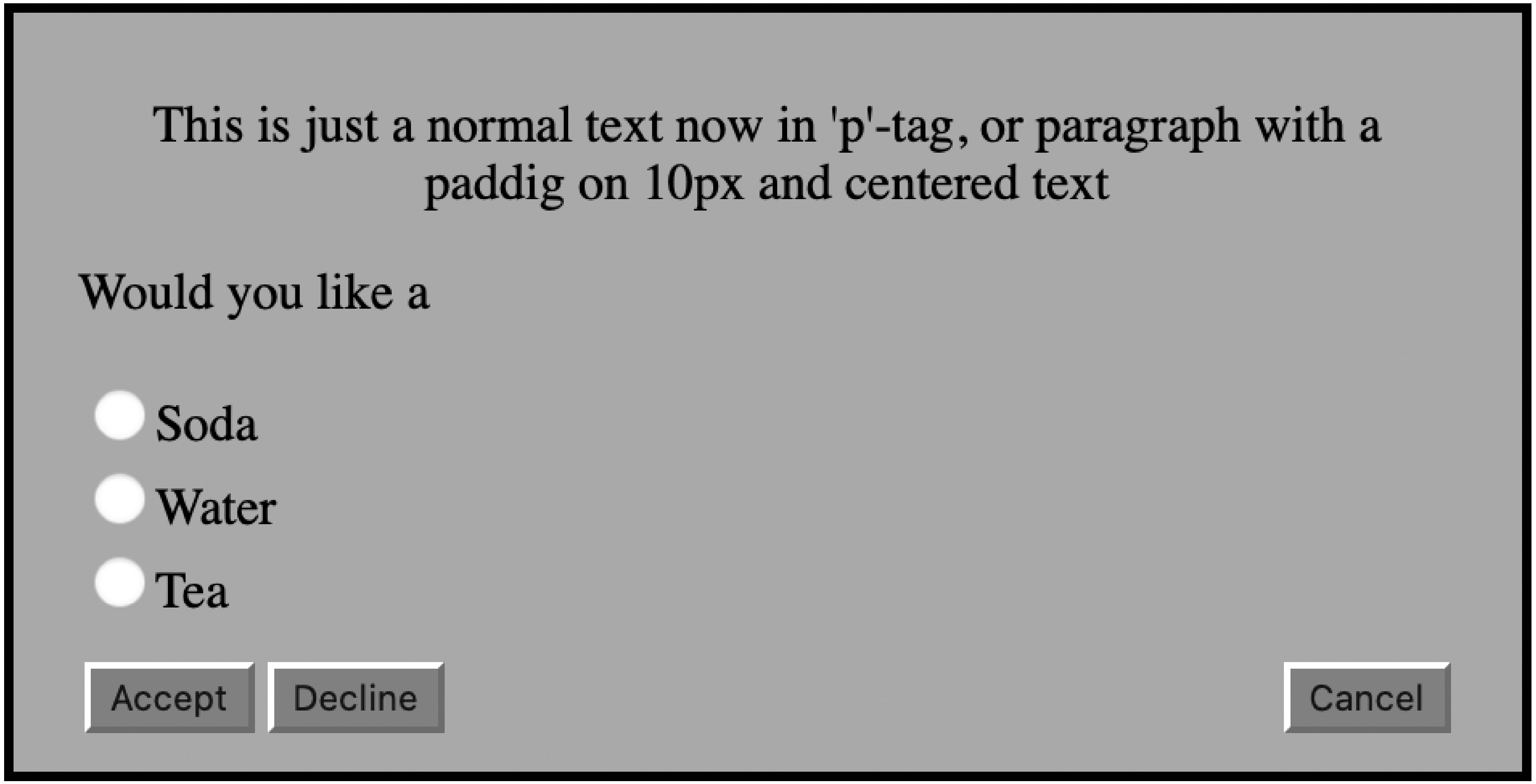
Here you can find all the properties and their possible values https://www.w3schools.com/Css/default.asp Introductory exercises (get as far as you can in 15 minutes): https://www.w3schools.com/css/exercise.asp

First exercise



- Box width: 30% of viewport
- Border color: sienna
- Button colors: green, red, #ffA525

First exercise - making things a bit more aesthetic


This is just a normal text now in 'p'-tag, or paragraph with a paddig on 10px and centered text

Would you like a

- The radio buttons can now get the same class, so make a css selector like this:

```
    .radiobutton{
        ...
    }
```

And use it like this:

```
        <input type="radio" class="radiobutton">Soda</input></br>
```

Second exercise - tables

Exercises: table.html

Third exercise - fixing error

Exercises: navbar-bad.html What is supposed to look line: navbar-solution.html

Tinker with the css tags in the developer tools until it looks like the solution. There are some hints in the text of what is wrong

Module: HTML, CSS & JavaScript

Introduction

JavaScript

Some interesting facts about JavaScript (1/7)

Along with HTML an CSS, JavaScript is one of the three main things of the www (World Wide Web). It enables interactive web pages and thus is an essential part of web applications. A majority of websites use it and all major web browsers have a devoted JavaScript engine to execute it.

Some interesting facts about JavaScript (2/7)

JavaScript is single threaded. This is the reason lots of people who use multi-threaded programming thinks its working is slow as it would not be able to make use of all the cores of the CPU properly.

Some interesting facts about JavaScript (3/7)

Despite the fact that there are similarities between JavaScript and Java, including language name, respective standard libraries and syntax, these two languages are distinct and differ significantly in design.

Some interesting facts about JavaScript (4/7)

Like all other scripting languages, arrays and objects can be created with a brief shortcut syntax. These literals structure the basis of JSON data format.

Some interesting facts about JavaScript (5/7)

JavaScript supports regular expressions in a manner similar to Perl, which provides a concise and powerful syntax for text manipulation that is more sophisticated than the built-in string functions.

Some interesting facts about JavaScript (6/7)

There is a CSRF attack known as “JavaScript hijacking” in which a tag on an attacker’s site damages a page on the victim’s site that returns private information such as JavaScript or JSON (Reference: Wikipedia)

Some interesting facts about JavaScript (7/7)

JavaScript is supported by all modern Web browsers with the built-in interpreters.

JavaScript Syntax

JavaScript syntax is the set of rules, how JavaScript programs are constructed:

```
// How to create variables:
var x;
const pi=3.14;
let y;
// How to use variables:
x = 5;
y = 6;
let z = x + y;
```

The JavaScript syntax defines two types of values:

- Fixed values
- Variable values

Fixed values are called **Literals**.

Variable values are called **Variables**.

JavaScript Literals

- **Numbers** are written with or without decimals:

```
var decimalVal=10.50;
```

```
decimalVal=1001;
```

- **Strings** are text, written within double or single quotes:

```
var name="John Doe";

var otherVay='John Doe';
```

JavaScript Variables

JavaScript uses the keywords **var**, **let** and **const** to declare variables.

`let` and `const` variables were lately added to js library ES6 (2015).

- let is used for the mutable variables

```
let x=10;
x=5*4;

var y=30;
y=20*4;
```

- const is used for the immutable variables.

```
const pi=3.14;

pi=5; // this will give error. Reassigning const is not possible
```

JavaScript DataTypes

JavaScript variables can hold different data types: numbers, strings, objects and more.

```
let x="string values";// string

let b=true; // boolean

let y= 15; // decimal

let k= 15.5; // decimal

let nullVal=null;

let z=new Date(); // date

let arr=[];// array

let obj={}; // object

obj={name:'klemen', surname:'clark', age:1};
//...
```

JavaScript Function

A JavaScript function is defined with the `function` keyword, followed by a **name**, followed by parentheses **()**.

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).

The parentheses may include parameter names separated by commas: **(parameter1, parameter2, …)**

The code to be executed, by the function, is placed inside curly brackets: **{}**

```
function sumOfTwoVariable( x, y ){
    return x+y;
}

function printToTheConsole( message ){
    console.log(message)
}

// you can pass function value to the variable

const result=sumOfTwoVariable(4,5);

console.log(result); // it will print 9

// Java Script is also functional programing, you can pass a function as a parameter to the other function

let val=genericOperation(9,5,sumOfTwoVariable) // val will be 14


function genericOperation(x,y,fun){

    return fun(x,x)
}
```

JavaScript Object

A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc.

JavaScript is an object-based language. Everything is an object in JavaScript.

JavaScript is template based not class based. Here, we don’t create class to get the object. But, we direct create objects.

Creating Objects in JavaScript

There are 3 ways to create objects.

1- By object literal

```
const carObject={brand:"volvo",year:2010,isNew:true};

console.log(carObject.brand);// volvo

carObject.year=2022;

console.log(year);// 2022
```

2- By creating instance of Object directly (using new keyword)

```
const carObject=new Object();
carObject.model='Ax';
carObject.name='mercedes'
```

3- By using an object constructor (using new keyword)

```
function employee(id,name,salary){
this.id=id;
this.name=name;
this.salary=salary;
}
```

```
const emp=new employee(103,"Vimal Jaiswal",30000);
```

JavaScript If-Else statements

The JavaScript if-else statement is used to execute the code whether condition is true or false.

```
var a=20;
if(a>10){
document.write("value of a is greater than 10");
} else {
    document.write("value of a is less than 10");
}
```

JavaScript Loops

The JavaScript loops are used to iterate the piece of code using for, while, do while or for-in loops. It makes the code compact. It is mostly used in array.

There are four types of loops in JavaScript.

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

```
  // for loop
    for (i=1; i<=5; i++)
```

```
     {
         console.log(i);
     }
     /*
     1
     2
     3
     4
     5
     */


     // while loop
     var i=11;

     while (i<=15)
     {
         document.write(i + "<br/>");
         i++;
     }

     // do while loop
     var i=21;
     do{
         document.write(i + "<br/>");
         i++;
     }while (i<=25);

     // for in loop
     var stringArr=['car','phone','key'];

     for(x in stringArr){
         console.log(x);
     }
```

JavaScript Arrays

The JavaScript you can use array to hold more than one value inside.

You can create an array in 3 different ways;

```
// 1 way
const cars=['audi','honda','mercedes']; // create an array with items

cars.push('reno');// adding still possible

//2 way
const cars=[]; // create an empty array
cars[0]='audi';
cars.push('mercedes');
cars[2]='honda';

// 3 way
const cars= new Array('audi','honda','mercedes'); // not a good way to create an array !!!
cars[3]='audi';
cars.push('mercedes');
cars[5]='honda';
```

For other data structures please read the W3Schools set,map,etc

Opening developer tools in the browser

Writing the first lines of JS

What could be the output of the following lines, and why?

```
console.log(false == '0');
console.log(false === '0');
console.log(false == '');
console.log(false == null);
console.log(false == 0);
```

Writing the first lines of JS

Solution

```
console.log(false == '0'); //true
console.log(false === '0'); //false
console.log(false == ''); //true
console.log(false == null); //false
console.log(false == 0); //true
```

Writing the first lines of JS - but why?

- "==" vs. "==="

Writing the first lines of JS - falsy values

The following values are always falsy:

- `false`
- `0 (zero)`

- `-0 (minus zero)`
- `0n (BigInt zero)`
- `''` and `""` (empty string)
- `null`
- `undefined`
- `NaN`

Writing the first lines of JS - truthy values

Everything else is truthy. That includes:

- `'0'` (a string containing a single zero)
- `'false'` (a string containing the text “false”)
- `[]` (an empty array)
- `{}` (an empty object)
- `function(){}` (an “empty” function)

Writing the first lines of JS

```
const obj = { 1: 'a', 2: 'b', 3: 'c' };
const arr = [1, 2, 3, 4, 5];

arr.push(6);

console.log(arr); // [1, 2, 3, 4, 5, 6]
```

```
obj.hasOwnProperty('1');
obj.hasOwnProperty(1);
arr.includes(1); // true
arr.includes(7); // false
arr.hasOwnProperty(4); // true

arr.splice(0,1); // start removing from index 0 and repeat 1 time
console.log(arr); // [2, 3, 4, 5, 6]

arr.splice(2,3); // start removing from index 2 and repeat 3 times

console.log(arr); // [2, 3]
```

Call alert message

```
window.alert("BOOOO!");
```

JavaScripts Events

The change in the state of an object is known as an Event. In html, there are various events which represents that some activity is performed by the user or by the browser. When javascript code is included in HTML, js react over these events and allow the execution. This process of reacting over the events is called **Event Handling**. Thus, js handles the HTML events via **Event Handlers**.

For example, when a user clicks over the browser, add js code, which will execute the task to be performed on the event.

Some of the HTML events and their event handlers are:

Event Performed	Event Handler	Description
click	onclick	When mouse click on an element
mouseover	onmouseover	When the cursor of the mouse comes over the element
Keydown & Keyup	onkeydown & onkeyup	When the user press and then release the key
focus	onfocus	When the user focuses on an element

Click Event

```
<html>
<head> Javascript Events </head>
<body>
<script language="Javascript" type="text/Javascript">
    <!--
    function clickevent()
    {
        document.write("This is JavaTpoint");
    }
    //-->
</script>
<form>
<input type="button" onclick="clickevent()" value="Who's this?"/>
</form>
</body>
</html>
```

MouseOver Event

```
<html>
<head>
<h1> Javascript Events </h1>
</head>
<body>
<script language="Javascript" type="text/Javascript">
    <!--
    function mouseoverevent()
    {
        alert("This is JavaTpoint");
    }
    //-->
</script>
<p onmouseover="mouseoverevent()"> Keep cursor over me</p>
</body>
</html>
```

So JavaScript But Where To

The <script> Tag

In HTML, JavaScript code is inserted between `<script>` and `</script>` tags.

Example

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

```
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>
<body>
<h2>Demo JavaScript in Head</h2>

<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>

</body>
</html>
```

External JavaScript

Scripts can also be placed in external files:

External file myScript.js

```
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

Referencing the js file in the html

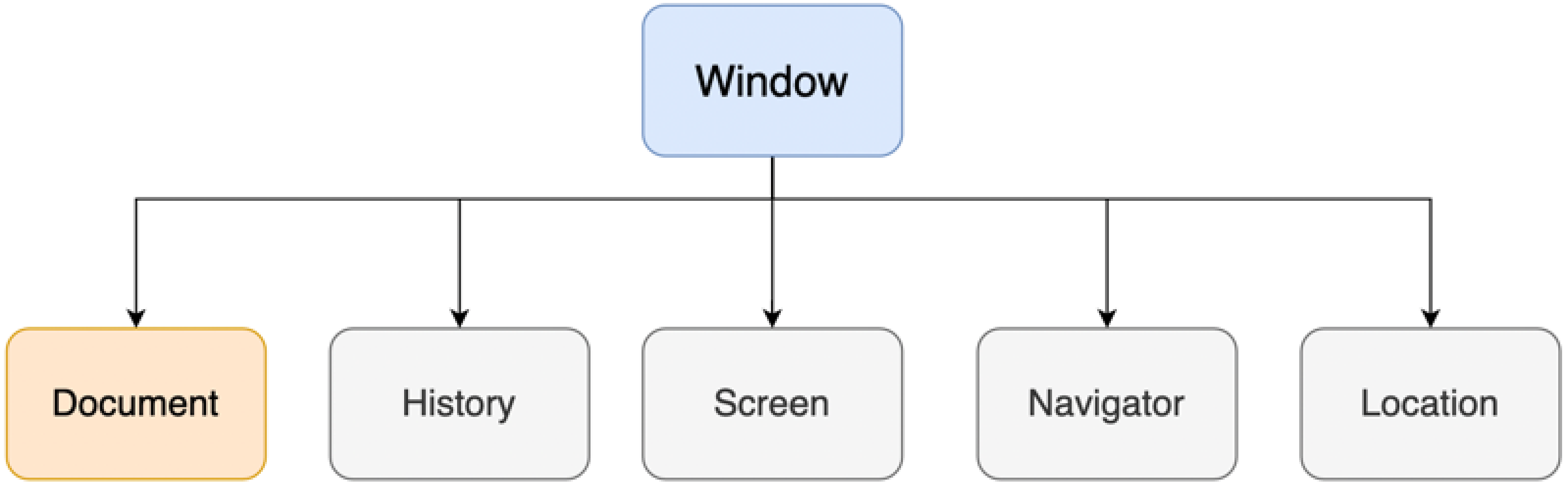
```
<script src="myScript.js"></script>
```

What is the BOM (Browser Object Model)

TThe **Browser Object Model** (BOM) is used to interact with the browser.

The default object of browser is window means you can call all the functions of window by specifying window or directly. For example:

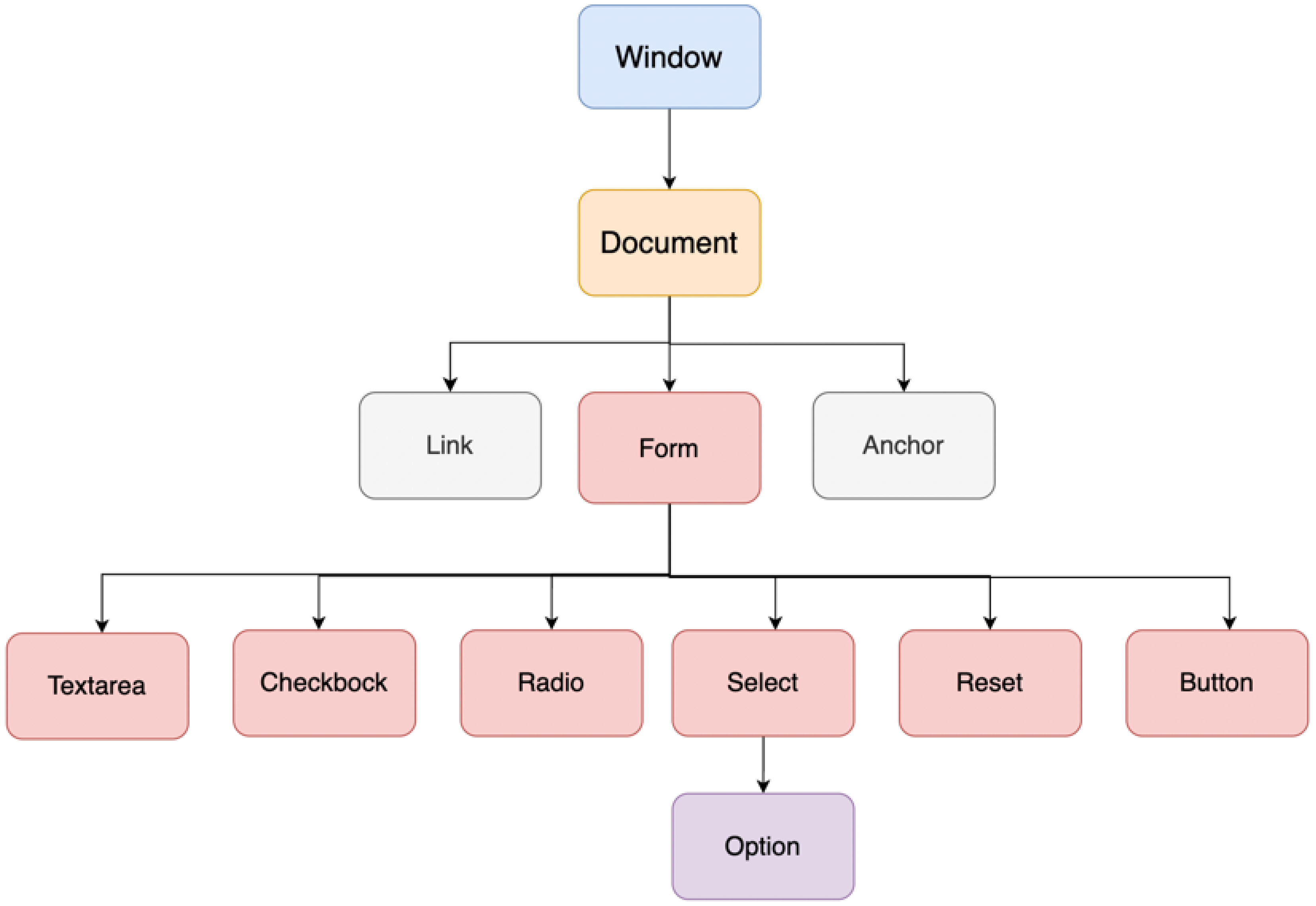
```
window.alert("Hello JavaScript");
// or
alert("Hello JavaScript")
```



What is the DOM (Document Object Model)

The **document** object represents the whole html document.

When html document is loaded in the browser, it becomes a document object. It is the root element that represents the html document. It has properties and methods. By the help of document object, we can add dynamic content to our web page.


Window
Document
Link
Form
Anchor
Textarea
Checkbock
Radio
Select
Reset
Button
Option

Method	**Description**
write("string")	writes the given string on the document.
getElementById()	returns the element having the given id value.
getElementsByName()	returns all the elements having the given name value.
getElementsByClassName()	returns all the elements having the given class name.
getElementsByTagName()	returns all the elements having the given tag name.

Example;

```
<script type="text/javascript">
function printvalue(){
var name=document.form1.name.value;
alert("Welcome: "+name);
}
</script>

<form name="form1">
Enter Name:<input type="text" name="name"/>
<input type="button" onclick="printvalue()" value="print name"/>
</form>
```

Exercises

1. Write a javascript function to show typed text on result.

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

```
<head>
  <meta charset=utf-8 />
  <title>Write a java script function print given text on Result </title>
</head>
<body>
<form>
 Your Input : <input type="text" id="text-in" />

  <input type="button" onClick="result()" Value="save" />
</form>
<p>You wrote :
  <span id = "result"></span>
</p>

<script>
  function result()
  {
    // uncomment the given codes
   // const text = document.getElementById("text-in").value;
   //document.getElementById("result").innerHTML = text;
  }

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

2. Please write the javascript function to calculate multiplication and division of two numbers on given html.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset=utf-8 />
  <title>JavaScript program to calculate multiplication and division of two numbers </title>
  <style type="text/css">
```

```
    body {margin: 30px;}
  </style>
</head>
<body>
<form>
  1st Number : <input type="text" id="firstNumber" /><br>
  2nd Number: <input type="text" id="secondNumber" /><br>
  <input type="button" onClick="multiplyBy()" Value="Multiply" />
  <input type="button" onClick="divideBy()" Value="Divide" />
</form>
<p>The Result is : <br>
  <span id = "result"></span>
</p>

<script>
  function multiplyBy()
  {
    // fill here by yourself
  }

  function divideBy()
  {
   // fill here by yourself
  }
</script>
</body>
</html>
```

3. Please open this HTML on chrome and use developer tools to test test50 function on console.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
    <title>JavaScript program to check two given numbers and return true if one of the number is 50 or 
if their sum is 50.</title>
</head>
<body>

<script>
    function test50(x, y) 
    {
        return ((x == 50 || y == 50) || (x + y == 50));
    }
</script>

</body>
</html>
```

4. Write a JavaScript program to check whether two given integer values are in the range 50..99 (inclusive). Return true if either of them are in the said range.
5. Write a JavaScript program to create new string with first 3 characters are in lower case from a given string. If the
6. Write a JavaScript program to compute the sum of three elements of a given array of integers of length 3
7. Write a HTMl to convert Celsius to Fahrenheit or vice versa. **Formula** is c/5 = (f-32)/9; [where c = temperature in Celsius and f = temperature in Fahrenheit] Expected Output : 60°C is 140 °F 45°F is 7.2°C

For More exercises you can do by yourself on W3School or W3Resource

Resources

- https://www.javatpoint.com/javascript-tutorial

- https://www.w3schools.com/js/default.asp
- https://www.w3resource.com/javascript-exercises/javascript-basic-exercises.php

Module: HTML, CSS & JavaScript

Introduction

HTML, Structure, Common Elements, Rendering the DOM