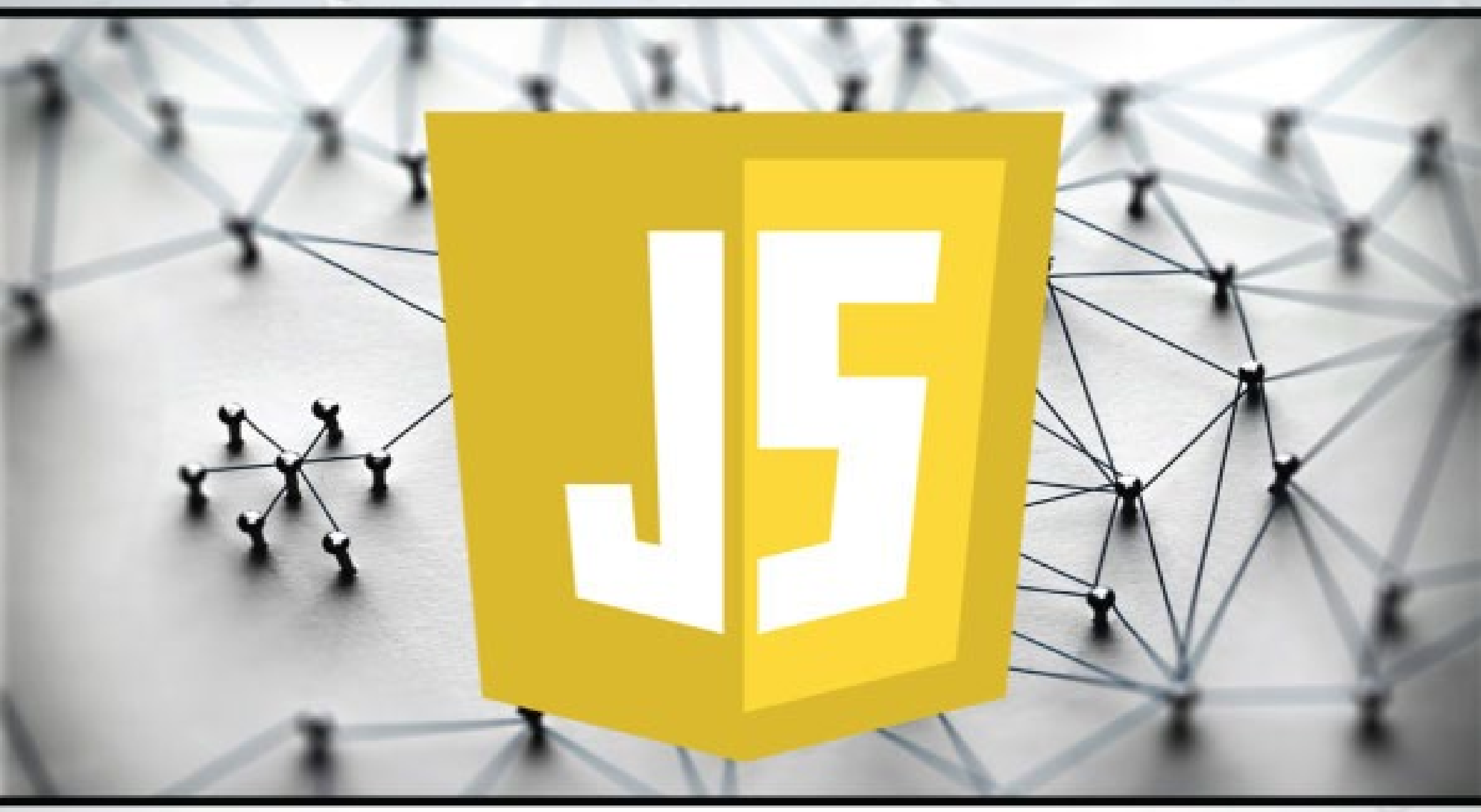
Beginners Guide



LEARN JAVASCRIPT

for Absolute Beginners

First Edition

PHPBOOTCAMP.COM

EASY TO PRACTISE CODE SAMPLES

About this Book

JAVASCRIPT is a client-side programming language that allows web developers to create scripts that can run on Client Browser. JAVASCRIPT Language is basically used to run program at client side. This book will help you understand the basics of JAVASCRIPT Language and how to put it in practice to build Websites.

Audience

This tutorial has been designed to meet the requirements of all those readers who are keen to learn the basics of JAVASCRIPT.

Prerequisites

This book assumes you have no prior knowledge on Programming knowledge and assume you are at a beginner level.

How to use this Book

This book contains JAVASCRIPT Language Basics, Exercises and Examples which are part of the PHP Bootcamp Program. This bootcamp has helped many students to become PHP Full Stack Web Developer in Just 30 days.

>>>Check out more about this program here...

Copyright & Disclaimer

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. PHPBootcamp.com provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@phpbootcamp.com.

Found Typos & Broken Link

If you found any typos and broken links in the course or book, then please let me know using this email address.

contact@phpbootcamp.com

Support

You can reach me for technical discussions and other support related queries from here.

support@phpbootcamp.com

Thanks for your support!

Free Courses

	Learn HTML in 1 Hour
	Learn CSS in 1 Hour
	Learn JAVASCRIPT in 1 Hour

Table of Contents

1. JavaScript Basics

1 JavaScript Basics

1.1 Introduction to JavaScript

Invention of JavaScript

During 1990's, When Internet and HTML was introduced to the IT industry web pages main purpose was to display content on the browser.

Webpages was created and stored on server which are connected to internet. These Web page main purpose is to connect all the other resource on the internet.

Connecting things in internet and accessing all resources via web page was the main concept.

In 1995, Brandon Eich Wrote Java Script and main purpose of it was to change the DOM which was created by the HTML.

JavaScript was born.

JavaScript main purpose is to execute program at the browser.

With JavaScript you are do following things:

- Change the Page Data at Runtime.
- Validate what user is doing on the Page.
- Show and Hide things on the page based on the User Action.
- Download and Upload data in the background from the browser.
- Animation Effects

- Control the timer.

History of JavaScript

In 1995, Brandon Eich Wrote Java Script at Netscape. Original name of JavaScript was Mocha. It was renamed to LiveScript and then to JavaScript.

Soon many browsers were released and which adopted the specification.

JavaScript is a Scripting Language. Know more about Scripting language from here.

Future of JavaScript

The main advantage of JavaScript is that it runs on users browser. There is no need to refresh the page or send the page to server and do some calculation.

All the operations happens on Browser so it makes the JavaScript extra ordinary faster than any web language.

Latest Famous Languages like Angular are completely built on JavaScript which handles front and back of the application from browser itself.

New Web Programming Language are based on JavaScript.

1.2 Internal JavaScript

Usage of Internal JavaScript

JavaScript can be written in many different places in HTML page.

One of the method of defining the JavaScript is inside the same HTML page.

This type of JavaScript includes is restricted to page level only means you cannot reuse this code in some other pages.

Benefit of using this internal JavaScript is when you want specific changes to apply for that page level only.

SYNTAX:

<head>
<script>

alert('This will show an alert box');

</script>
</head>

alert is a function to show alert box on the web page.

alert('some text here');

Add semicolor ';' at the end of the line.

<style> tag is used to write the JavaScript inside this tag.

This tag <style> can be defined in <head> or <body>

JavaScript code written inside the <style> tag are executed as it is written.

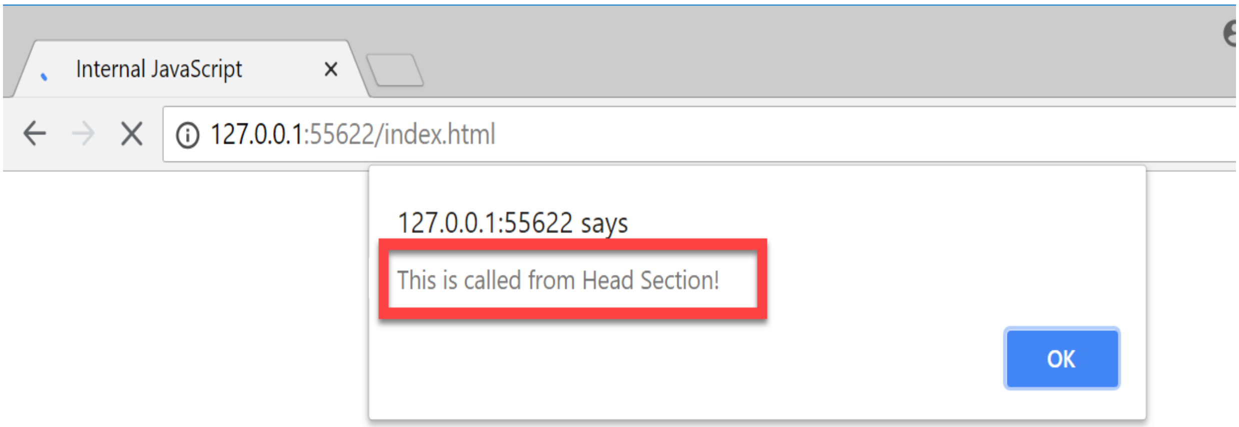
If you have two <script> tag one after the other then code inside the <script> executes one after the other.

If you put the <script> tag at the end of the <body> tag then the script is executed at the last after the page is loaded.

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Internal JavaScript</title>
    <script>

        alert('This is called from Head Section!');

    </script>

</head>

<body>
<h1>Heading</h1>
<p>Paragraph Text</p>
<ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
</body>

</html>
```



127.0.0.1:55622 says

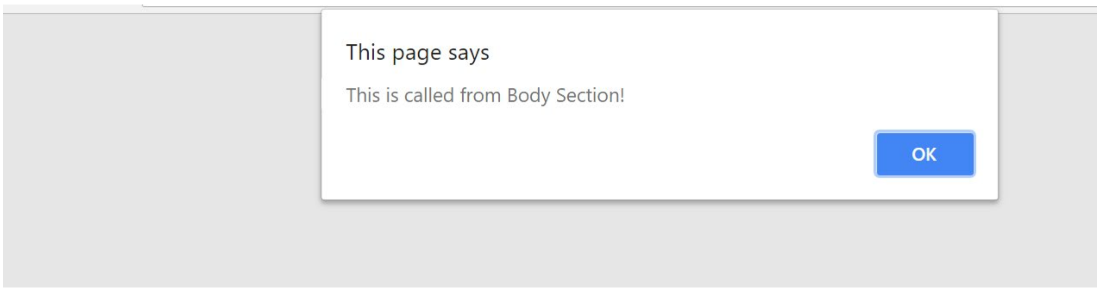
This is called from Head Section!

[Live Preview]

Exercise 1

[Download the Exercise 1]

Exercise 1: Put the <script> tag with alert at the end of the page in the <body> section.



This page says

This is called from Body Section!

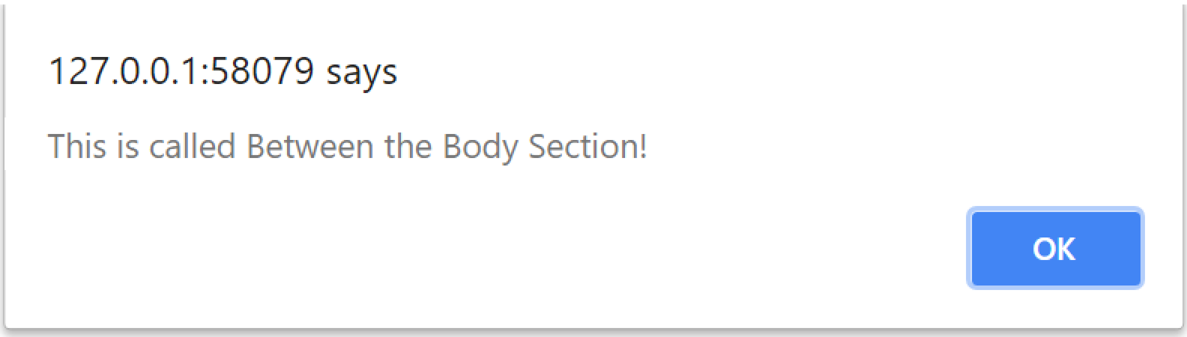
Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Put the script tag with alert with tags in the <body> section and see if that works.

Also, Press “OK” button and then Press F5 to see if the alert box comes again.



127.0.0.1:58079 says

This is called Between the Body Section!

Live Preview

1.3 Comments JavaScript

Usage of Comments

You can use the special notation to comment the code inside the JavaScript.

Comments helps to document about the code with a single or multiple line.

Comments are ignored by the JavaScript Engine and it is not displayed on the browser.

HTML comments and JavasScript comments are not same. Don’t get confused with the comments in JavaScript vs comments in HTML.

There are two types of Comments in JavaScript:

Single Line Comments

Multi Line Comments

// – This is Single Line Comments. Use to comment one line.

/* – is used to indicate the comments are starting.

***/** – is used to indicate the comments are ended.

Anything between **/*** and ***/** will not be executed by the browser.

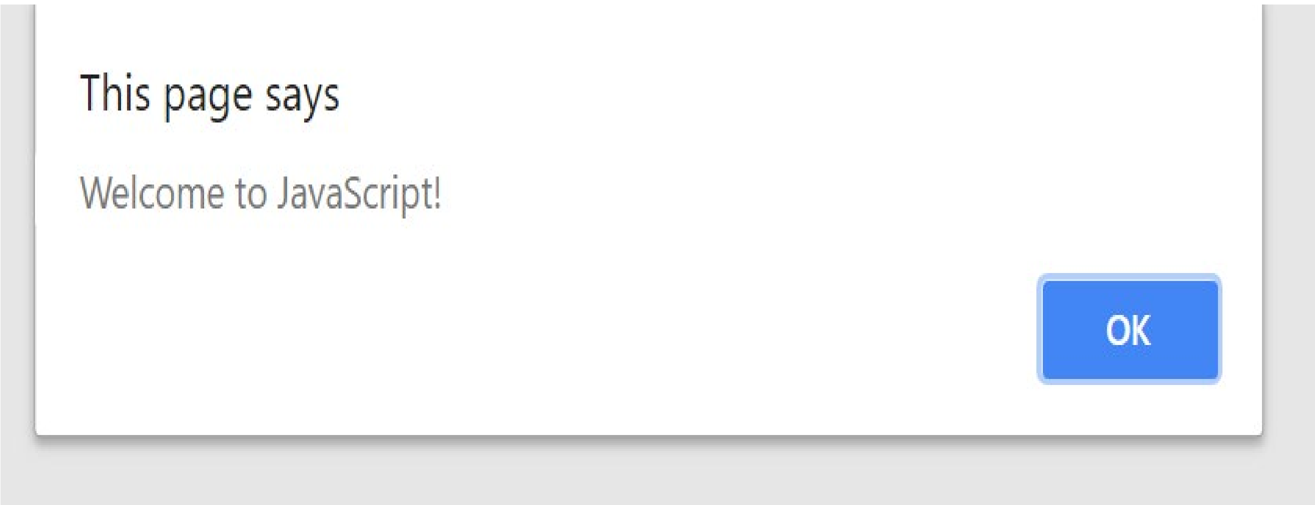
SYNTAX:

```
<head>
<script>

// This will alert the user – Single Line Comments
alert(‘This is alerted on the browser’);

/* Starting of the Multi Line Comment

alert(‘This is not executed by the browser’);

Ending of the Multi Line Comments */

</sript>
</head>
```

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <meta name="description" content="Page Description">
        <title>How to Write Comments in JavaScript</title>

        <!-- This is HTML Comments -->

        <script>
            /*
                This is a multi-line comment in JS.
                This is different than the HTML comments.
                Anything inside this block will be ignored.

                alert('This is Multi Line Comment Blok. Ignored by the
Browser');
            */

            //Alert the user with the text
            alert('Welcome to JavaScript!');

        </script>
        <!-- This is HTML Comments -->

    </head>

    <body>
        <h1>Heading</h1>
        <p>Paragraph Text</p>
        <ul>
            <li>Item 1</li>
            <li>Item 2</li>
            <li>Item 3</li>
        </ul>
    </body>
</html>
```



This page says

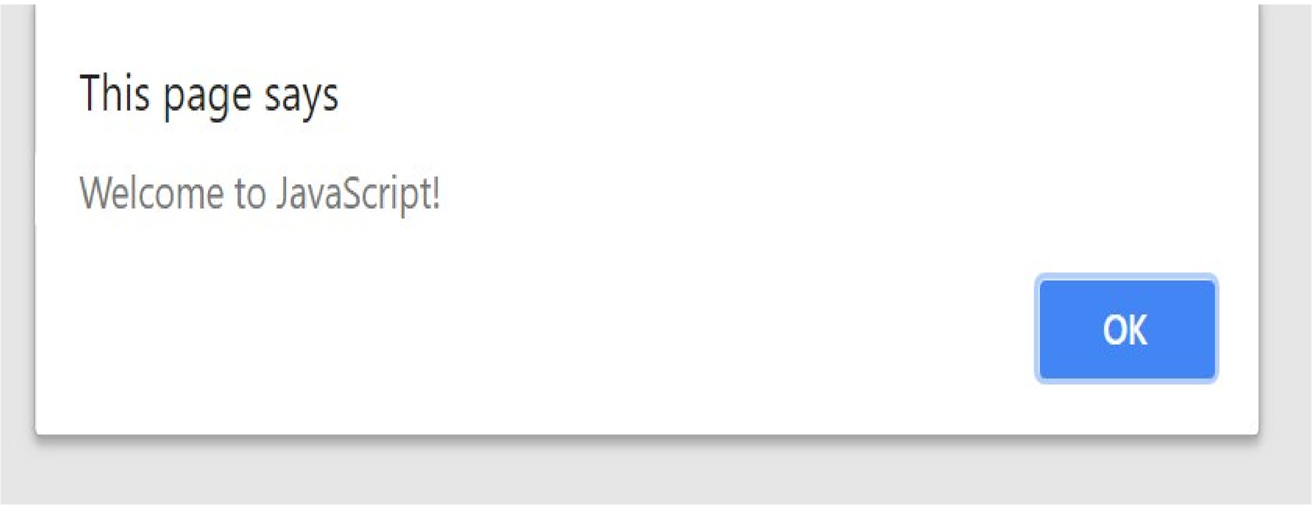
Welcome to JavaScript!

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Make the multi-line comments that looks like single line.



This page says

Welcome to JavaScript!

Live Preview

Exercise 2

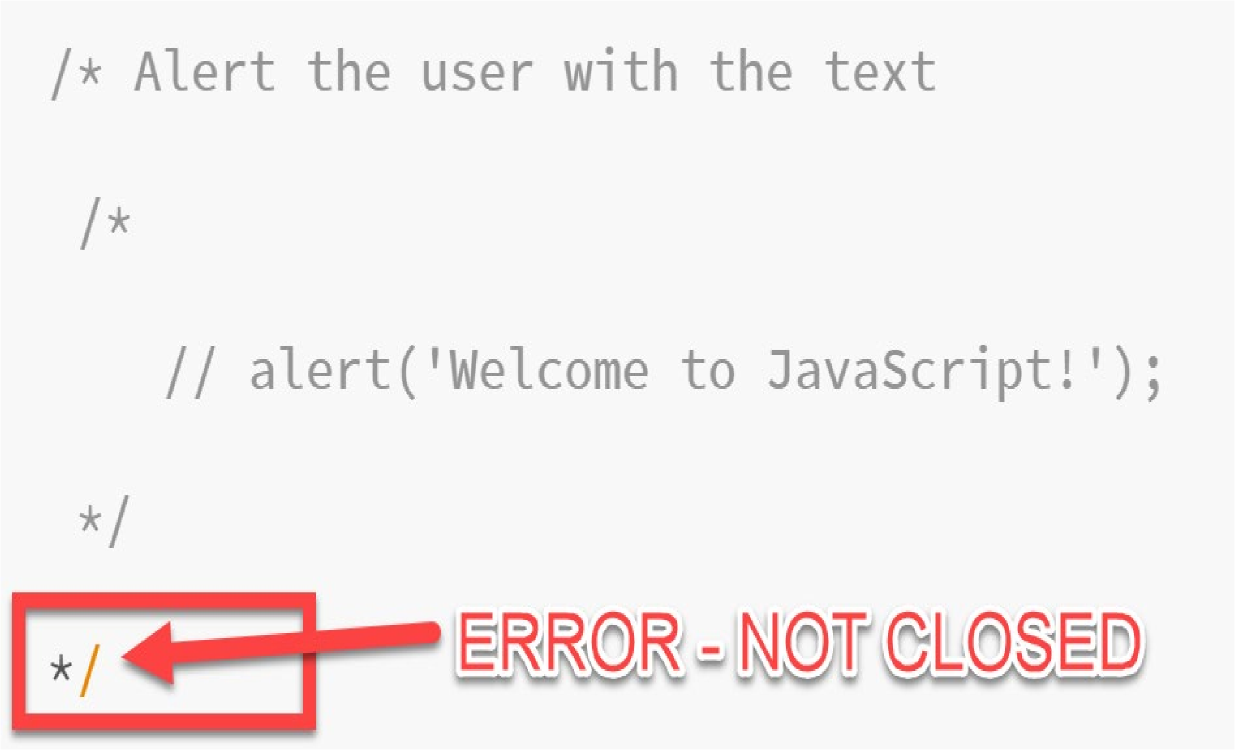
Download the Exercise 2

Exercise 2:

1. Comment the entire **<script>** block with HTML comments.

Observe how the HTML comments can comment all the **<script>** tags

2. Write Multi Line Comments inside the Multi Line comments and see it shows an error on the page.



```
/* Alert the user with the text

 /*

    // alert('Welcome to JavaScript!');

 */

*/
```

ERROR - NOT CLOSED

Live Preview

1.4 External JavaScript

Usage of External JavaScript

JavaScript (JS) can be written in another file and included inside the HTML page.

This type of external JS is very powerful and helpful technique which is commonly used in every website development.

Benefit of using this external JS is that you have one JS file that is included in all the website pages.

By just changing at one place in the JS it will impact the overall site design look and feel.

This is one of the best practice to separate the design with the html tags and store them in a external file and include it in all the HTML pages.

External JS filename should be .js and it can be included anywhere inside the HTML page with

<script type=’text/javascript‘ src=”location of the file></script>

SYNTAX:

```
<head>

<!– Make sure javascriptcode.js file exists in the ‘javascript’ folder –>

<script type=’text/javascript‘
src=”javascript/javascriptcode.js”></script>

</head>
```

<script> tag is used to link the resource to the HTML page. The attribute of script tag will let the browser knows what type of resource it is.

src attribute is similar to <img> tag src to map the location of the file in the server with the path and filename. You can even mention the foldername/file name to refer the file path.

Sample Example

Download the Example

There are two files:

index.html

javascript/javascriptcode.js

javascriptcode.js file is linked inside the index.html file with <script> tag.

javascript is the folder name it could be any name and not mandatory to have that name.

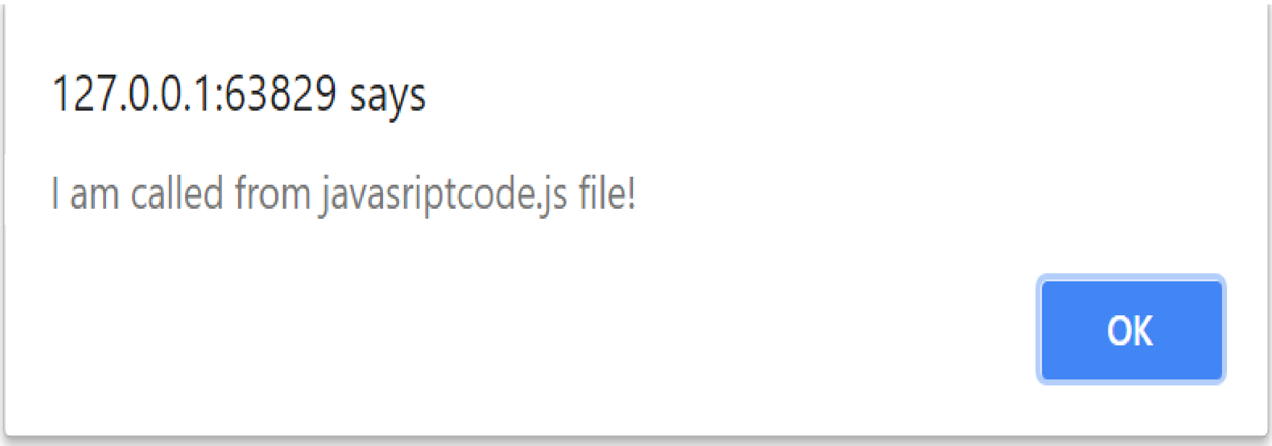
type attribute tells the type of the content in the file. In this case, it is text/javascript

FileName: index.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>External JavaScript</title>
    <script type='text/javascript'
src="javascript/javascriptcode.js"></script>
</head>

<body>
<h1>External JavaScript!</h1>
<p>Paragraph Text</p>
<ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
</body>

</html>
```

FileName: javascriptcode.js

```
alert('I am called from javasriptcode.js file!');
```



127.0.0.1:63829 says

I am called from javasriptcode.js file!

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Create js1.js and js2.js and link them in the HTML page.

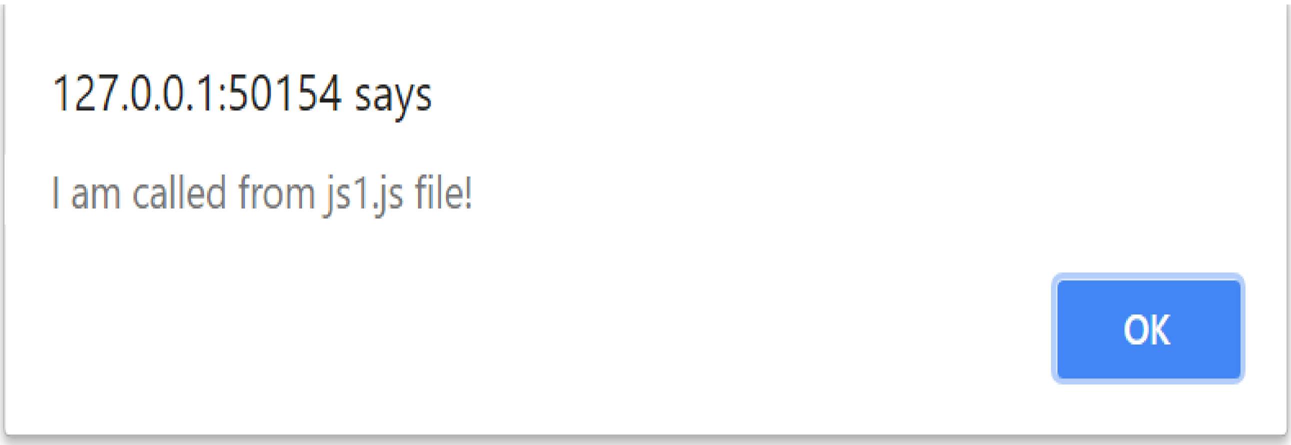
Do not create folder and place the js files along with the HTML file.

Filename: js1.js

alert('This is called from js1.js');

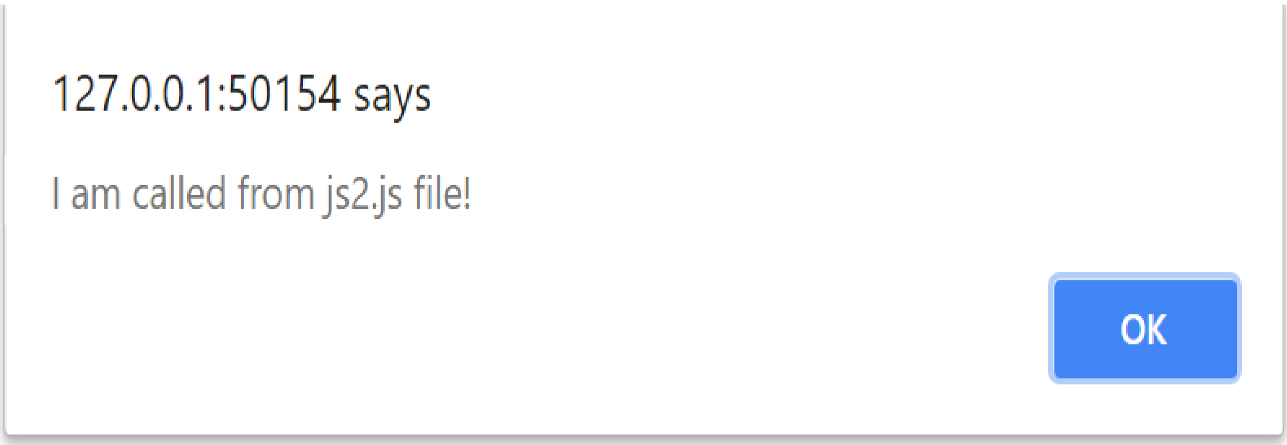
Filename: js2.js

alert('This is called from js2.js');



127.0.0.1:50154 says

I am called from js1.js file!



127.0.0.1:50154 says

I am called from js2.js file!

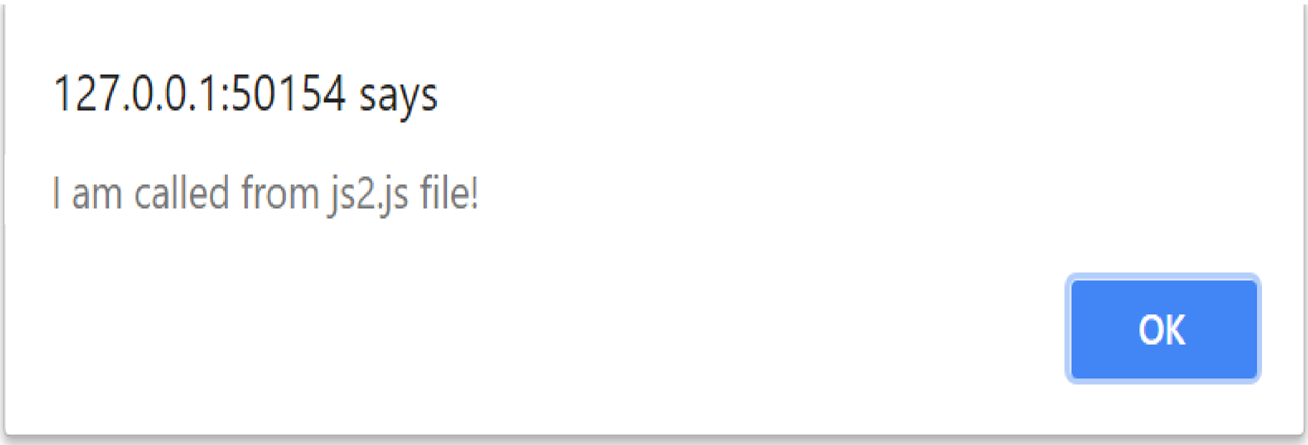
Live Preview

Exercise 2

Download the Exercise 2

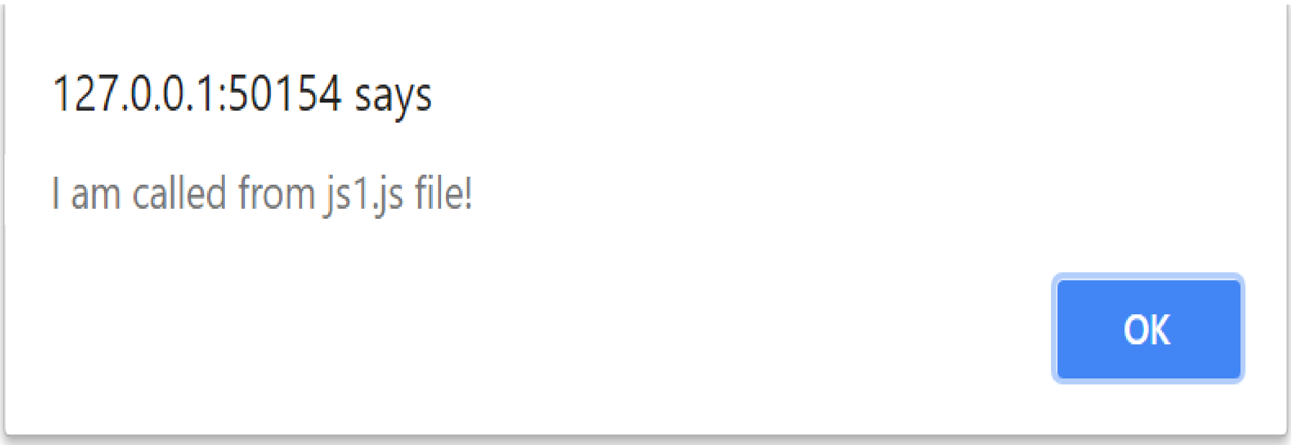
Exercise 2: Place one <script> tag inside the <head> tag and put another <script> tag at the end of <body>.

Swap the sequence, put js2.js first and js1.js last.



127.0.0.1:50154 says

I am called from js2.js file!



127.0.0.1:50154 says

I am called from js1.js file!

Live Preview

1.5 Inline JavaScript

Usage of Inline JavaScript

Inline JavaScript is defined inside the HTML tag itself like an attribute.

Inline JS overrides all the styles defined in internal JS and External JS.

Inline JS code is executed first.

This is defined in the HTML tag as a attribute.

SYNTAX:

<a href=”#” onclick=”alert(‘Welcome to JavaScript!’);”>Click Me</a>

onclick is a event for <a> tag and it is called when user clicks on the link.

Calling this alert is handled by the browser itself. When user clicks on the link, browser will raise the click event on this tag and because we ask to raise an alert when this event is raised. This method is called.

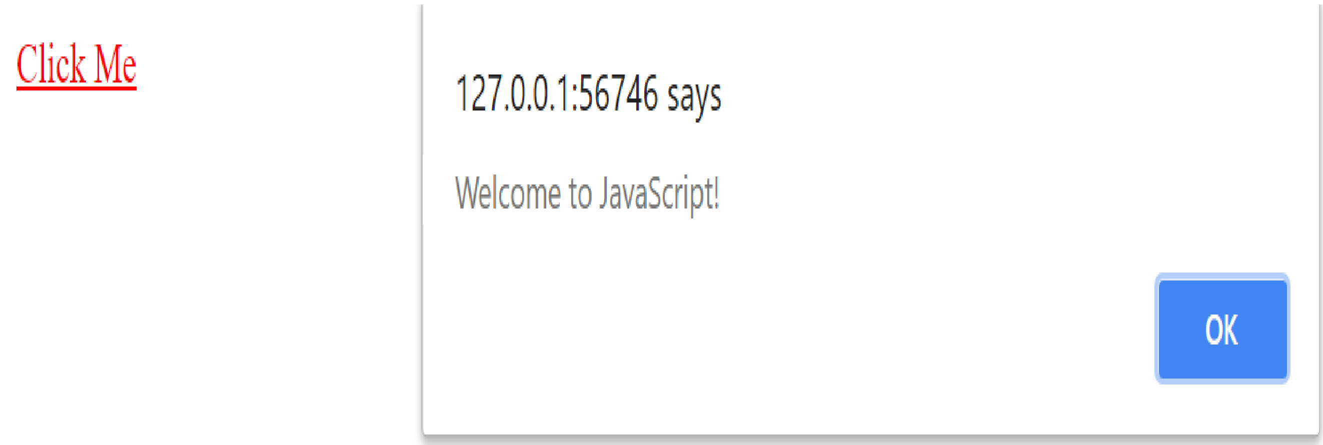
Whatever is mentioned inside the onClick value will be executed as JavaScript.

JavaScript is written inside this onClick event within double quotes “”. All the JavaScript is exactly similar as mentioned in internal and external JavaScript.

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Inline JavaScript</title>
</head>

<body>
    <a href="#" onclick="alert('Welcome to JavaScript!');">Click Me</a>
</body>

</html>
```



Click Me

127.0.0.1:56746 says

Welcome to JavaScript!

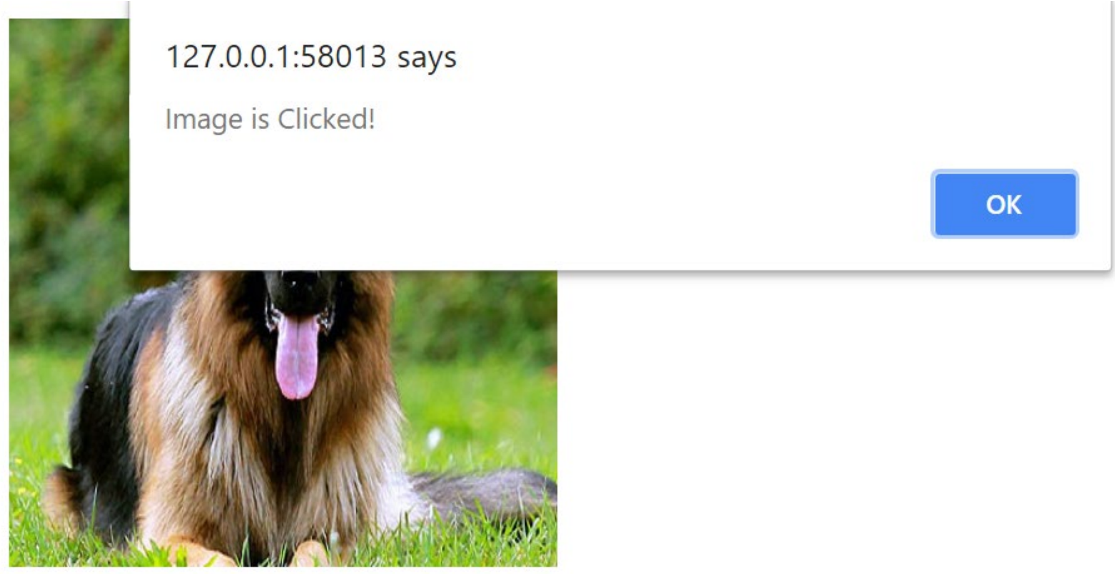
Live Preview

Become PHP Full Stack Web Developer in Just 30 Days

Exercise 1

Download the Exercise 1

Exercise 1: Add an Image and on clicking the image show an alert message.



127.0.0.1:58013 says

Image is Clicked!

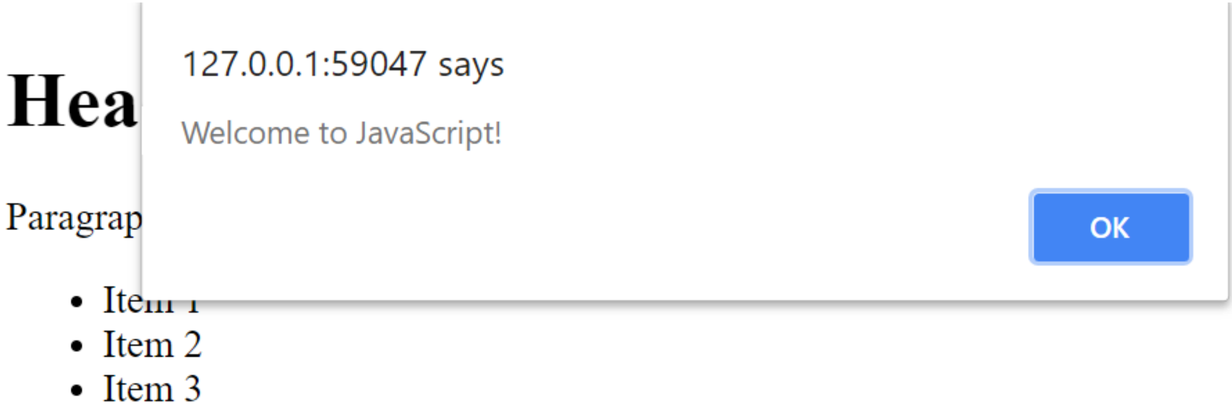
Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Show a alert message when clicked anywhere on the page.

Find the area when clicked – alert is raised and find the area where alerts are not raised on the body.



127.0.0.1:59047 says

Welcome to JavaScript!

[Live Preview]

1.6 Hello Sample JavaScript

Hello World Program

You can choose to write JavaScript Internal, External or Inline and it will work.

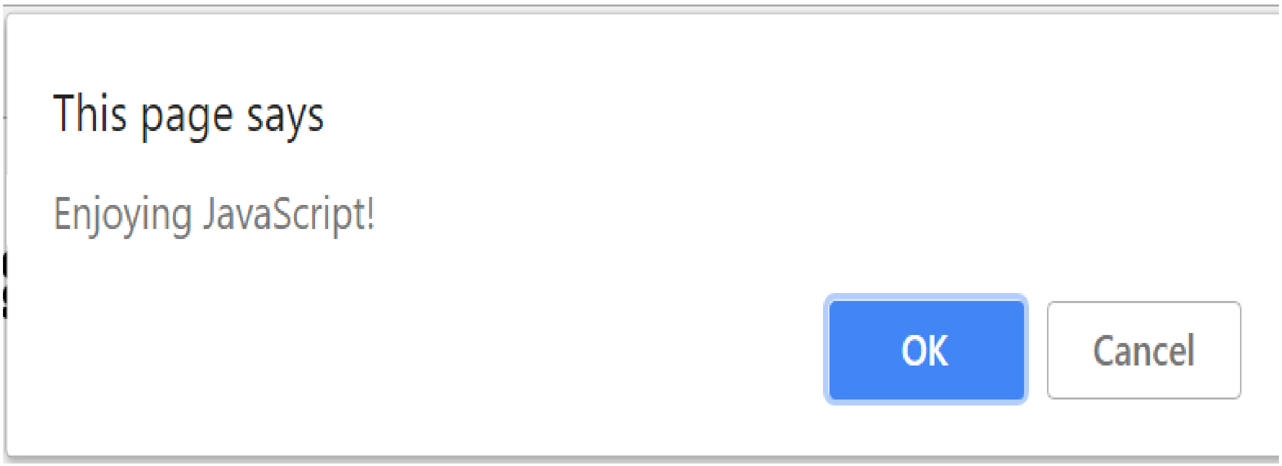
But the best way is to organize your code into one file so that it is easy to fix issues at one place for your entire site.

For this hello world sample, we will use the inline javascript. Once we know how to write functions and call them from on click even then we improve this program.

Instead of alert we will use confirm method that shows ok and cancel button.

However, we don’t take any action on what user pressed so just show a different confirm box when user click the link.

confirm('Enjoying JavaScript!');



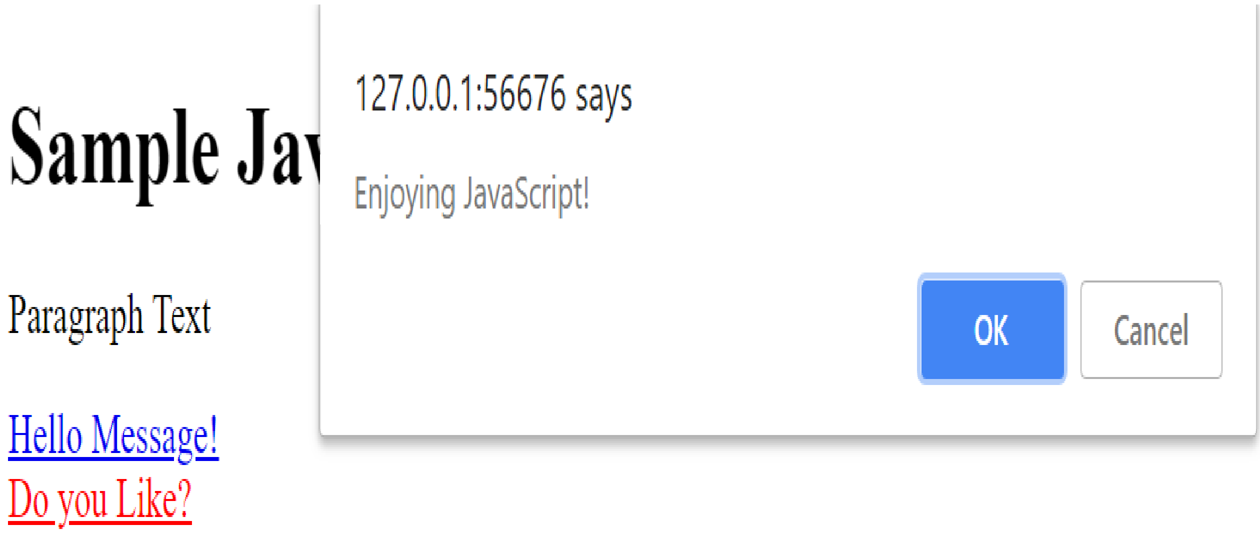
This page says

Enjoying JavaScript!

Sample Example

[Download the Example]

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Sample JavaScript Program!</title>
</head>

<body>
<h1>Sample JavaScript Program</h1>
<p>Paragraph Text</p>
<a href="#" onclick="alert('Hello World!');">Hello Message!</a>
    <br>
<a href="#" onclick="confirm('Enjoying JavaScript!');">Do you Like?</a>

</body>
</html>
```



Sample Jav

127.0.0.1:56676 says

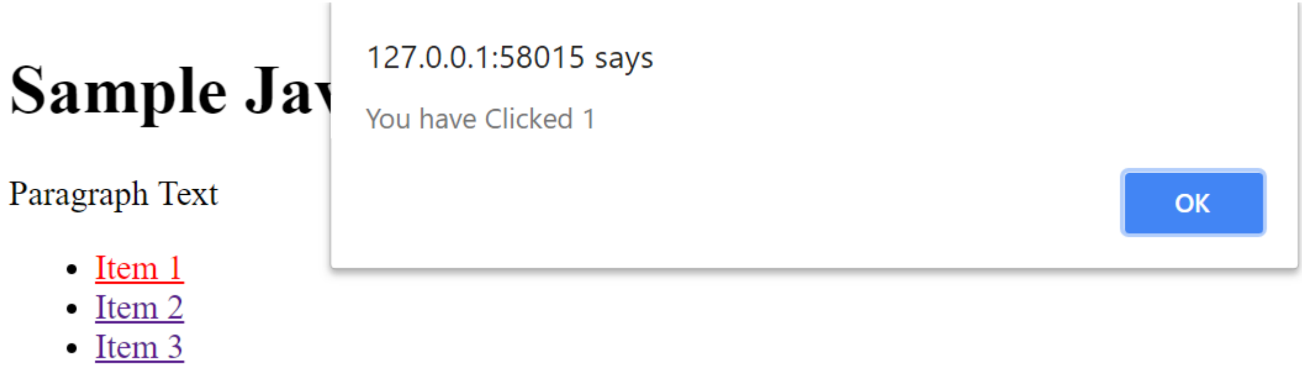
Enjoying JavaScript!

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Show list of items 1, 2 and 3. When user clicks any links show what they have clicked.



Sample Jav

127.0.0.1:58015 says

You have Clicked 1

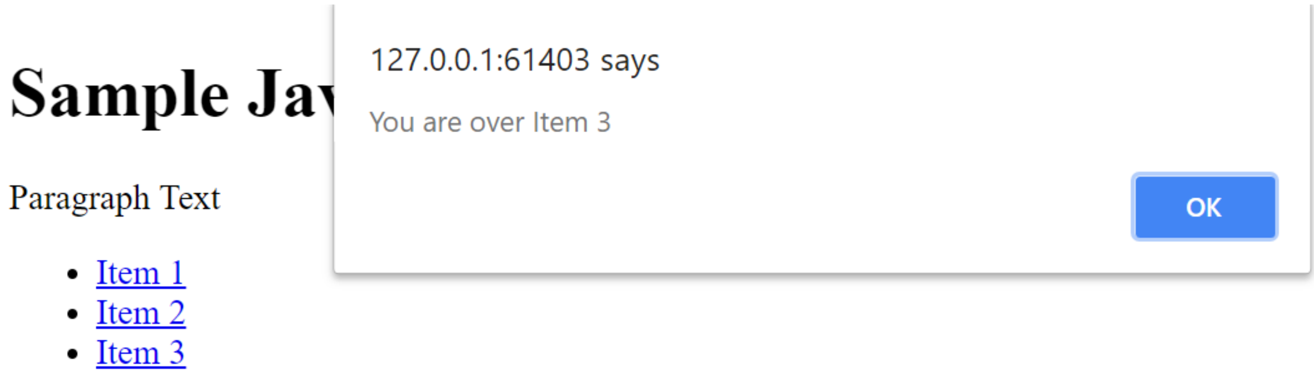
Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Replace **onclick** with **onmouseover** in the above example and see how alert is raised when mouse is over the link.

Become PHP Full Stack Web Developer in Just 30 Days



127.0.0.1:61403 says

You are over Item 3

Live Preview

1.7 JavaScript Terminology

These are the terms you need to know if you are a JavaScript Programmer.

Semicolon (;) – Semicolor symbol is used to indicate the browser that line is finished.

Brackets – [] – These are used to represent Arrays.

Braces – { } – Flower brackets are used to define the scope like starting point and ending point and we write the code inside this block.

Like starting of function and ending of a function.

parantheses – () – Is used to call a function.

Identifiers – are the name given to variables, functions, properties or object. Any thing you name it is identifier.

Variable – Any identifier that stores a value.

Operators – Special symbols that are used to perform some operation. Like Arithmetic operator + is used to add two numbers.

Example : +, -, ++

Expressions – They are variables or operators that resolves into something. a = 1 + 2 is an expression.

a = 1 + 2; //Expression always evaluate to some other value.

Statements – Group of commands to perform an action or could be a single statement. They end with ‘;’. Statement are meant to do some action.

```
{
y = sum(1, 2);
alert(“value of y” + y);
}
```

1.8 Testing & Debugging JavaScript

Testing Debugging JavaScript

Because JavaScript runs in the browser the only way to debug the JavaScript program is from the browser tools

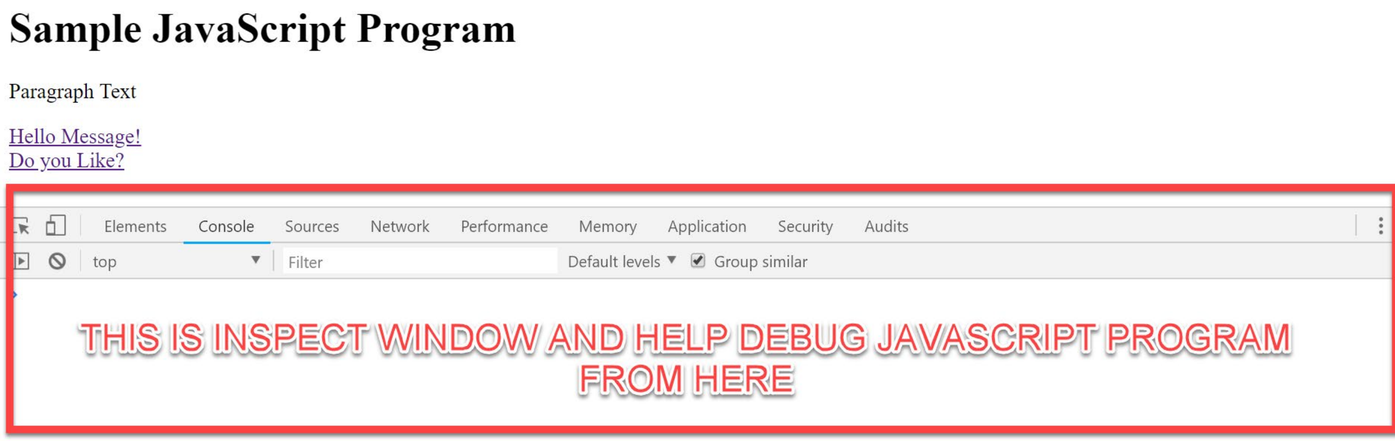
Chrome and Firefox has plenty of debugging tools that can help to find the issue and fix it. You can pause the execution of javascript program and debug it steps by step.

You need to learn this method to find the issues in JavaScript and Fix it.

Inspect Window:

Load any HTML program with JavaScript in Chrome and press CTRL + SHIFT + I.

This will load the Inspect Window.



Sample JavaScript Program

Paragraph Text

Hello Message!
Do you Like?

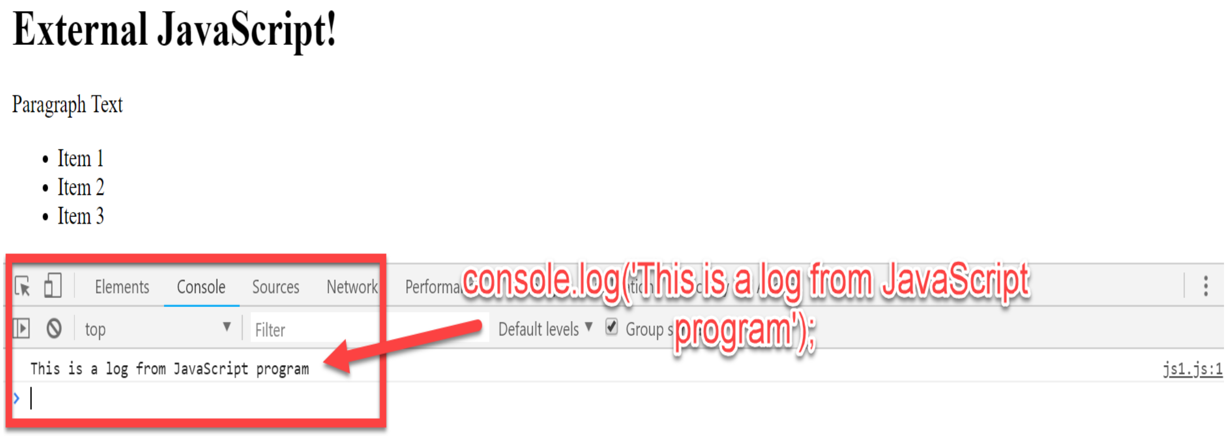
THIS IS INSPECT WINDOW AND HELP DEBUG JAVASCRIPT PROGRAM FROM HERE

Console Tab:

Console tab is the output tab where you can write the logs of your program and view it.

Even browser will it is own log here if there is any issue in the HTML program.

console.log('This is a log from JavaScript program');



External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

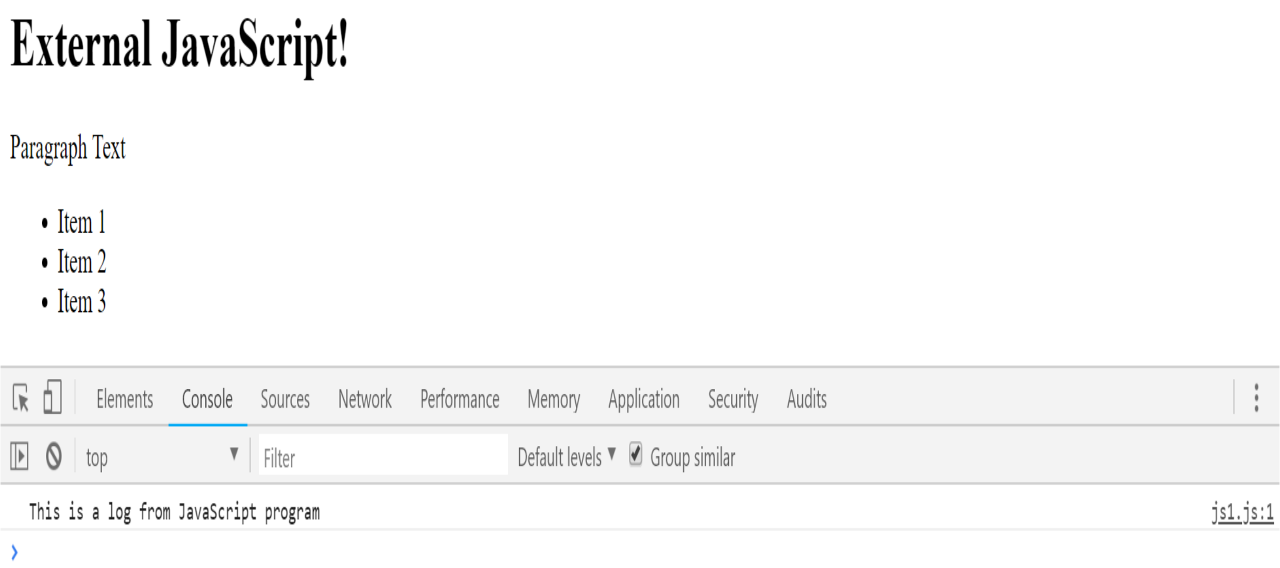
console.log('This is a log from JavaScript program');

```
This is a log from JavaScript program                js1.js:1
```

This is the easiest way to debug your program by writing console.log at many places in your code and verify it from the inspect -> console window.

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>External JavaScript</title>
    <script type='text/javascript' src="js1.js"></script>
</head>

<body>
<h1>External JavaScript!</h1>
<p>Paragraph Text</p>
<ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>

</body>

</html>
```



External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

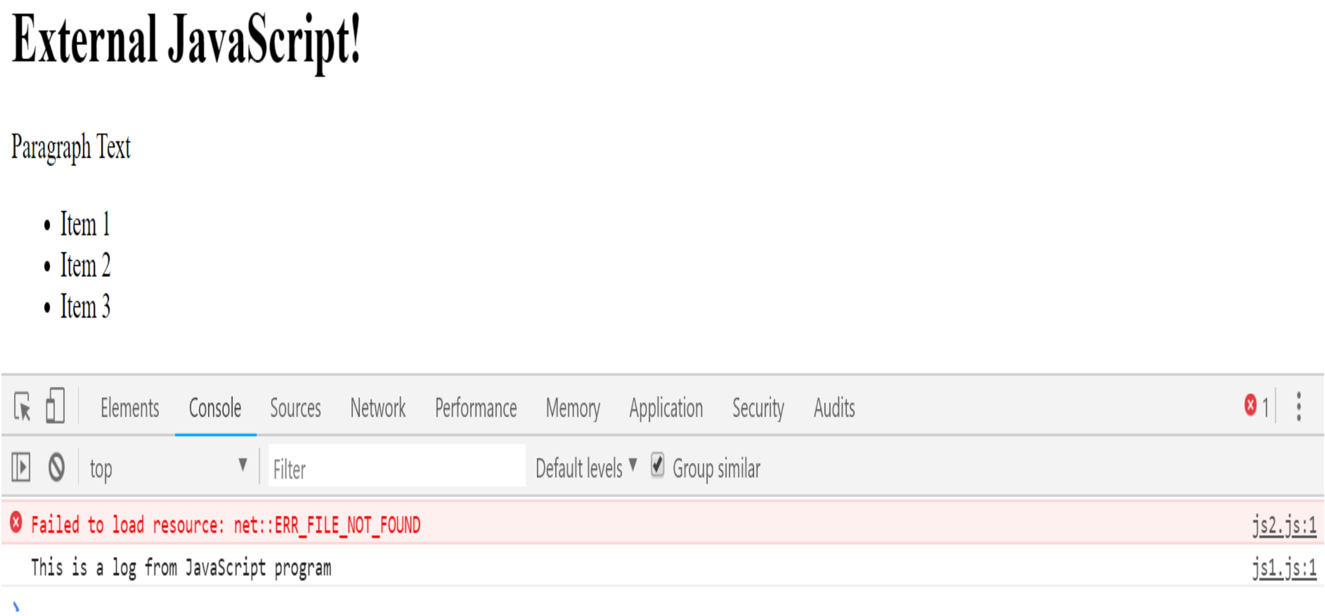
```
This is a log from JavaScript program                                    js1.js:1
›
```

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Include another js file in the head section. js2.js but do not create the file. Verify the error in the console window.



External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

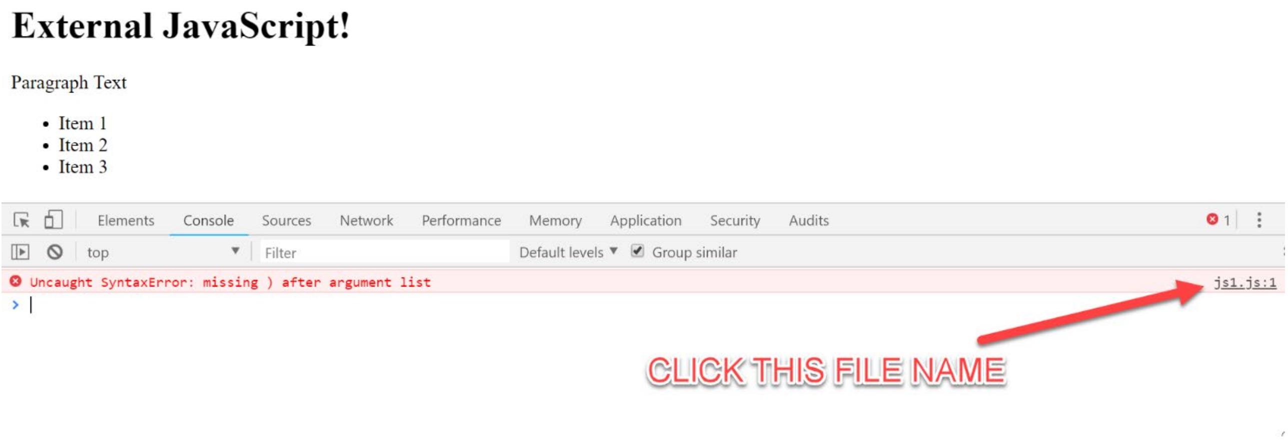
```
Failed to load resource: net::ERR_FILE_NOT_FOUND                js2.js:1
This is a log from JavaScript program                           js1.js:1
```

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Remove the **)** at the end of the console.log or alert and see the error in the console window.



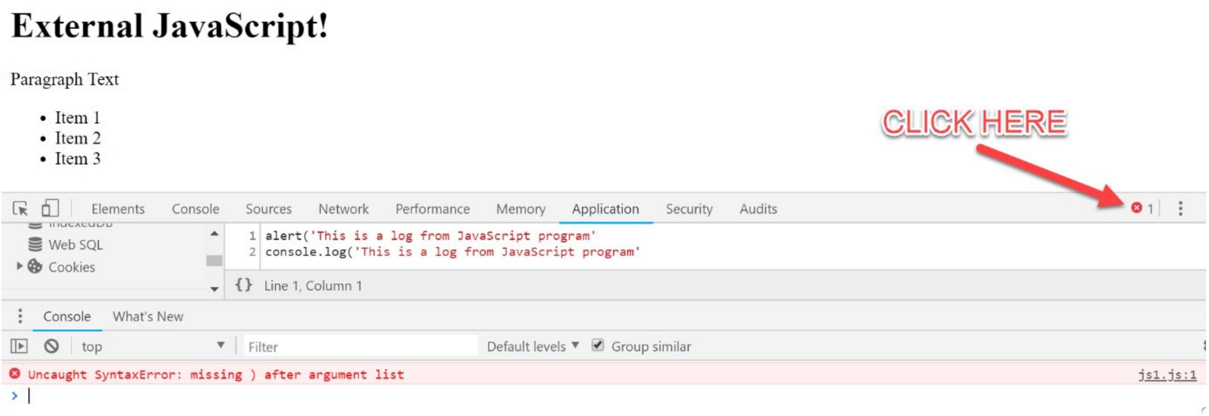
External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

```
Uncaught SyntaxError: missing ) after argument list        js1.js:1
```

CLICK THIS FILE NAME



External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

CLICK HERE

```
alert('This is a log from JavaScript program'
console.log('This is a log from JavaScript program'
```

```
Uncaught SyntaxError: missing ) after argument list        js1.js:1
```

[Live Preview]

2. Working with Data

2 Working with Data

2.1 Identifiers

Usage of Identifiers

Identifiers are the name that we give to variables, functions, objects, properties and events.

There are some rules that you need follow to define the Identifiers.

- Identifiers can contain only letters, numbers, underscore and dollar sign
- Identifiers cannot start with a number.
- Identifiers are case sensitive
- Identifiers can be any length.
- Identifiers cannot use keywords and reserved words.

Some of the Valid Identifiers:

- strFirstName
- $var
- index_1
- crashReport

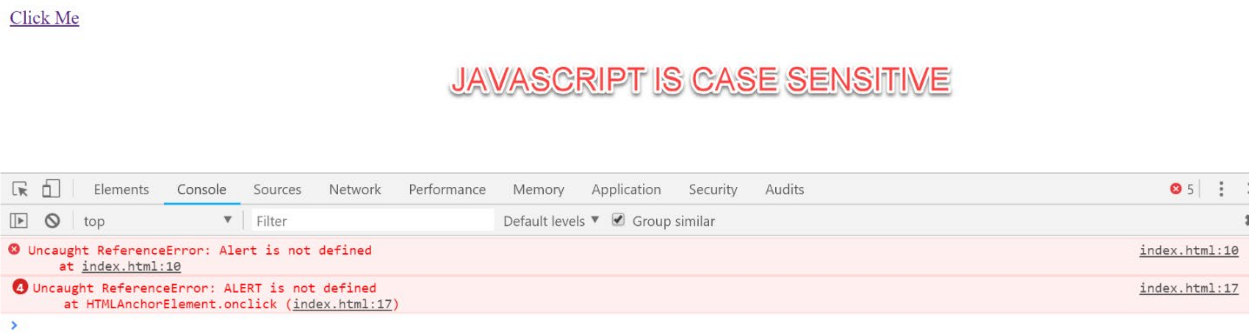
InValid Identifiers:

- 1PhoneNumber
- false
- long
- package

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Identifiers JavaScript</title>
    <script>

        Alert('JavaScript is Case Sensitive');

    </script>

</head>

<body>
    <a href="#" onclick="ALERT('Welcome to JavaScript!');">Click Me</a>
</body>

</html>
```



Click Me

JAVASCRIPT IS CASE SENSITIVE

```
Uncaught ReferenceError: Alert is not defined                index.html:10
    at index.html:10
Uncaught ReferenceError: ALERT is not defined                index.html:17
    at HTMLAnchorElement.onclick (index.html:17)
>
```

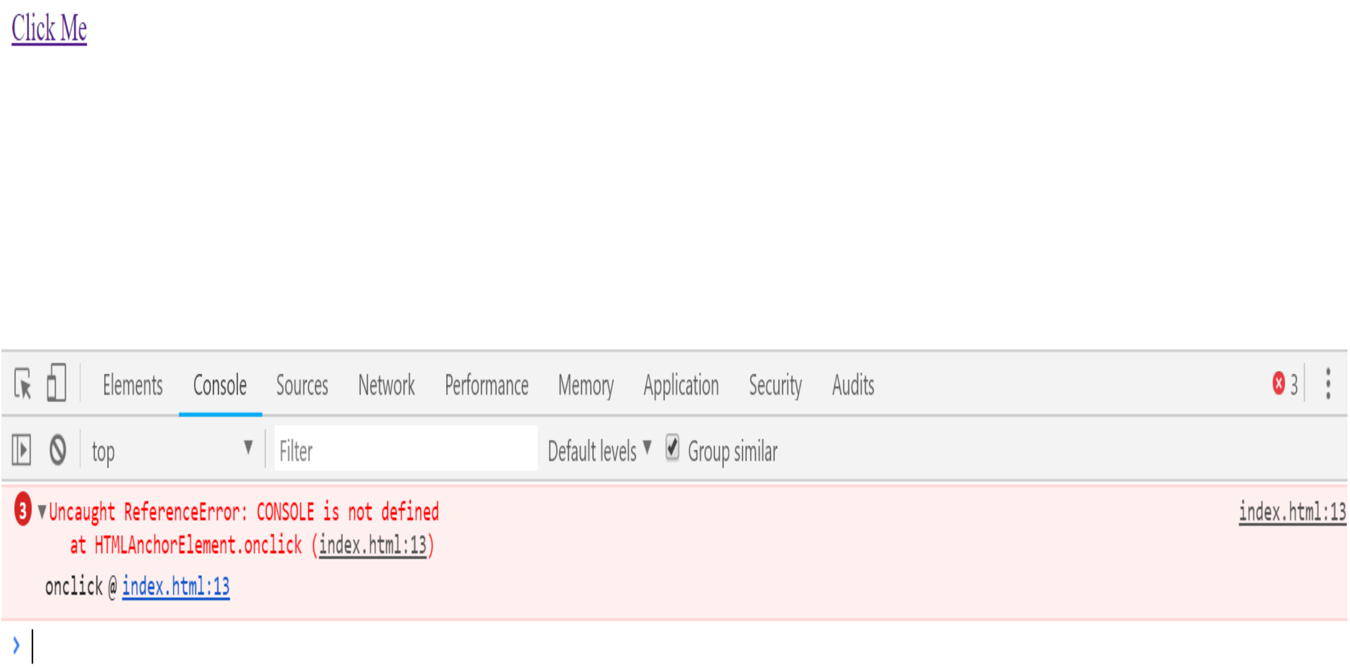
Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Try to change the console.log as CONSOLE.LOG and see what happens.

CONSOLE.LOG('JavaScript is Case Sensitive');



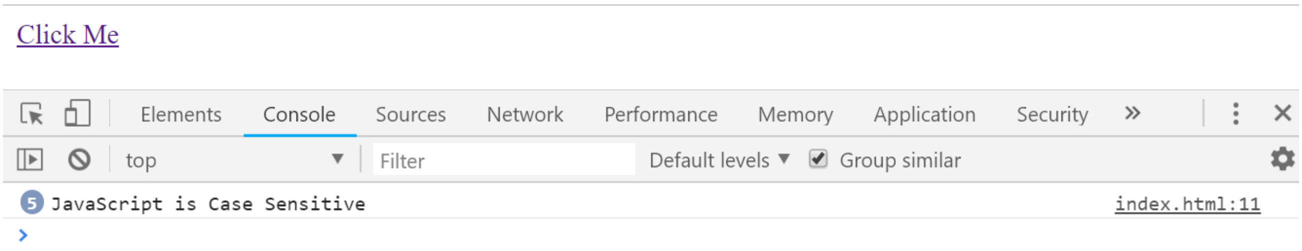
```
3 ▼Uncaught ReferenceError: CONSOLE is not defined          index.html:13
      at HTMLAnchorElement.onclick (index.html:13)
   onclick @ index.html:13
```

[Live Preview]

Exercise 2

[Download the Exercise 2]

Exercise 2: Make onclick as ONCLICK and see if the click command still works. Write the correct JavaScript Syntax.



```
5 JavaScript is Case Sensitive                              index.html:11
```

OnClick is an attribute of the HTML tag not JavaScript.

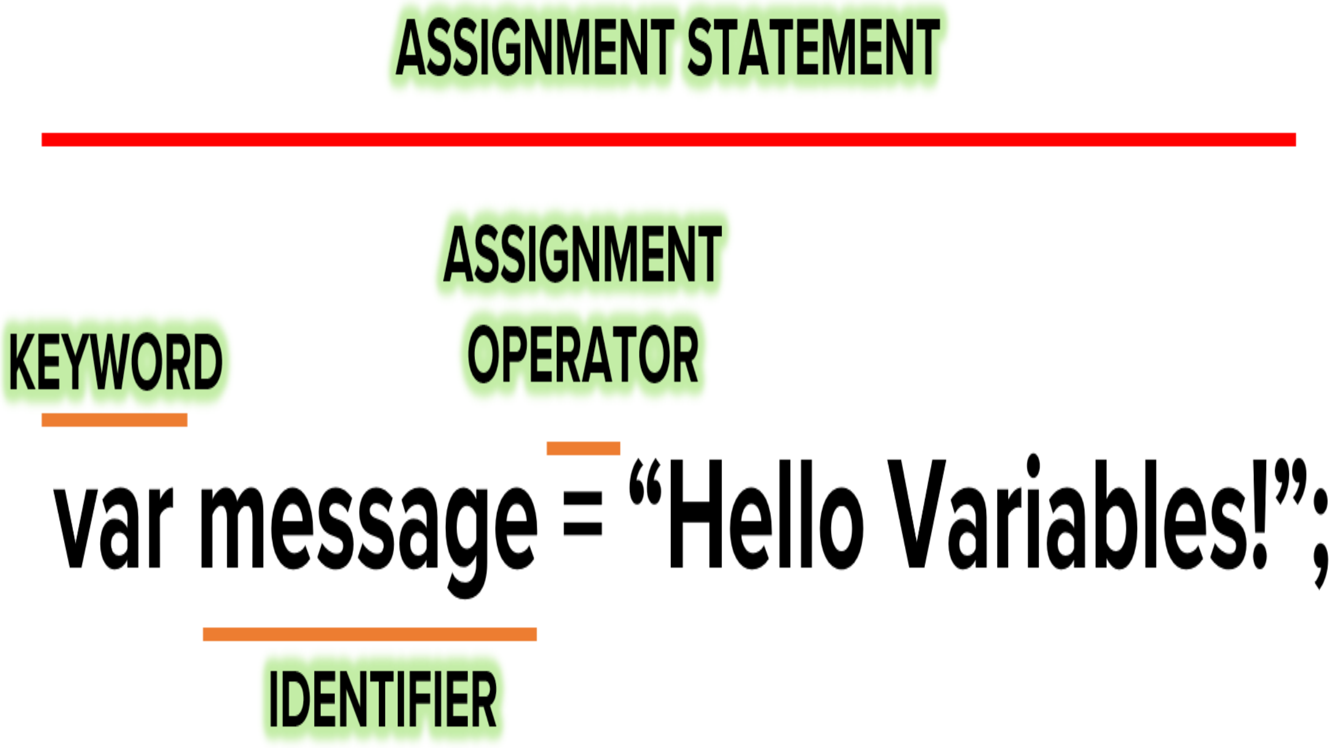
[Live Preview]

[Become PHP Full Stack Web Developer in Just 30 Days]

2.2 Variables

Usage of Variables

Variables are used to store information which are used inside the program.



var keyword is used to define a variable.

var message = “This is a test message”;

= is the assignment operator used to assign the value to the variable.

“” double quotes is used to represent string value.

; semicolor is used to end the assignment statement.

SYNTAX:

```
//Declaring a Variable
var variableName;
```

```
//Initialize the variable with empty string
variableName = “”;
```

```
//Assigning Value to the Variable.
variableName = “This is a Test Message!”;
```

```
//Multiple Variable
var name1, name2, name3;
nam1 = “firstName”, name2 = “middleName”, name3 =”lastName”;
```

```
//Defining the Integers & Decimals
var counter = 0;
counter = 10;
var total = 10.98;
counter = total;
```

Show Variable in alert message:

```
var firstName = “WPFreelancer.com”;
alert(‘Welcome to ” + firstName );
```

+ symbol is used to attach a variable to the string and display it.

Write Variables on HTML Document

document.write() method can be used to write anything on the HTML page.

As alert is used to show alerts similarly document.write() method is used to write content on the HTML page. It can also access variable defined in the head -> script section.

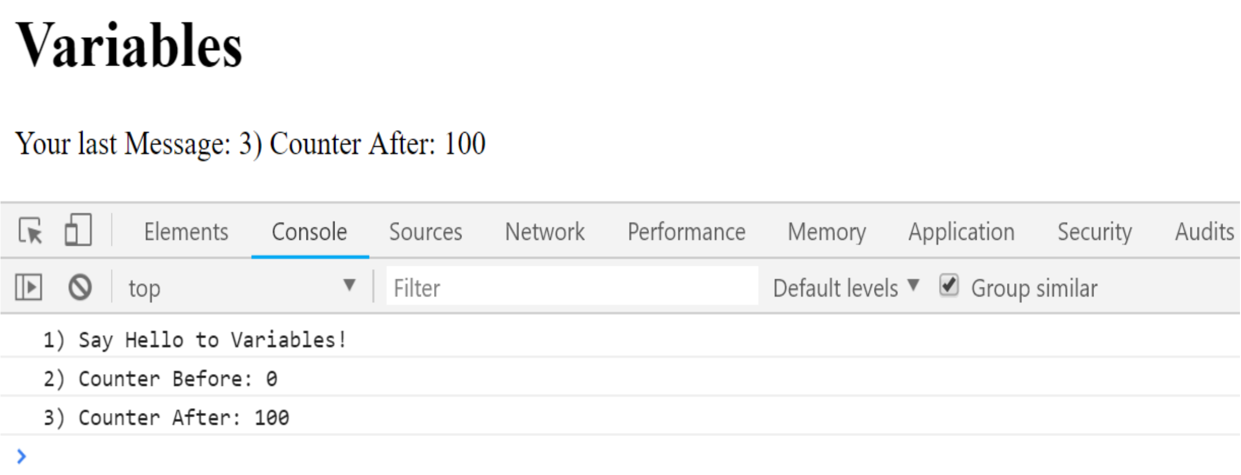
<script type=”text/javascript”>
document.write(“Your Message” + message + “<br>”);
</script>

Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Variables</title>
    <script>

        //Declaring a Variable
        var message;

        //Initalizing a Variable with Empty String
        message = "";

        //Assigning value to the Variables
        message = "1) Say Hello to Variables!";

        //Displaying the Variables
        alert(message);
        console.log(message);

        var counter = 0;
        alert("2) Counter Before: " + counter);
        message = "2) Counter Before: " + counter;
        console.log(message);

        counter = 100;
        alert("3) Counter After: " + counter);
        message = "3) Counter After: " + counter;
        console.log(message);

    </script>

</head>

<body>
<h1>Variables</h1>
<script type="text/javascript">
document.write("Your last Message: " + message);
</script>
</body>
</html>
```

```
//Initalizing a Variable with Empty String
message = "";

//Assigning value to the Variables
message = "1) Say Hello to Variables!";

//Displaying the Variables
alert(message);
console.log(message);

var counter = 0;
alert("2) Counter Before: " + counter);
message = "2) Counter Before: " + counter;
console.log(message);

counter = 100;
alert("3) Counter After: " + counter);
message = "3) Counter After: " + counter;
console.log(message);
```



Variables

Your last Message: 3) Counter After: 100

```
1) Say Hello to Variables!
2) Counter Before: 0
3) Counter After: 100
```

Live Preview

Exercise 1

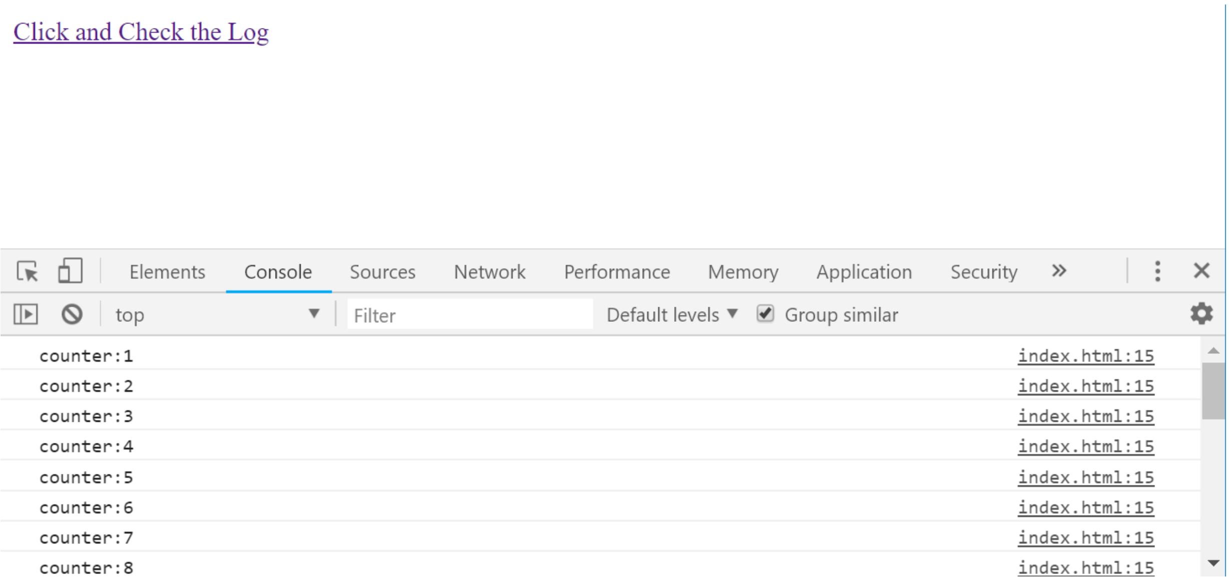
Download the Exercise 1

Exercise 1: Count the number of times the user clicked the link and show it for every click on the link.

Define the counter variable in the head -> script section and track the number of clicks and show it in the console.log.

Snippet:

onclick="counter = counter + 1; console.log('counter:' + counter);"



```
counter:1                                              index.html:15
counter:2                                              index.html:15
counter:3                                              index.html:15
counter:4                                              index.html:15
counter:5                                              index.html:15
counter:6                                              index.html:15
counter:7                                              index.html:15
counter:8                                              index.html:15
```

Live Preview

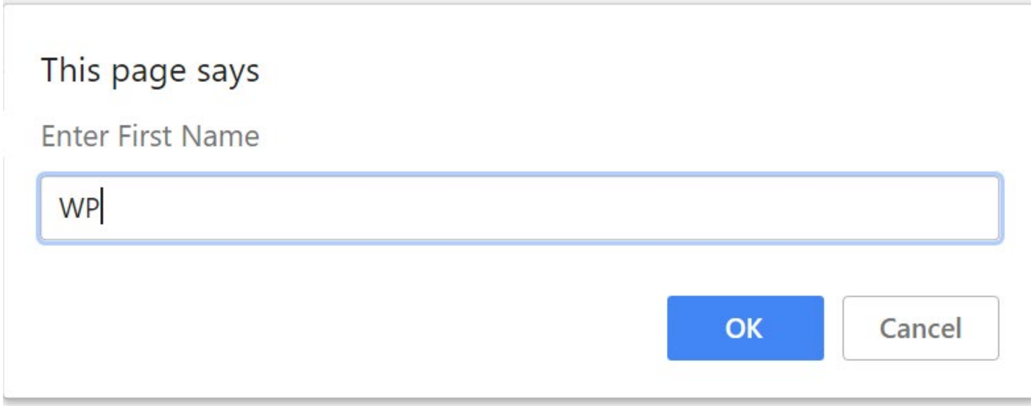
Exercise 2

Download the Exercise 2

Exercise 2: Accept two names from User and Display them on the HTML page.

Accept Name: firstName = **prompt**("Enter First Name");

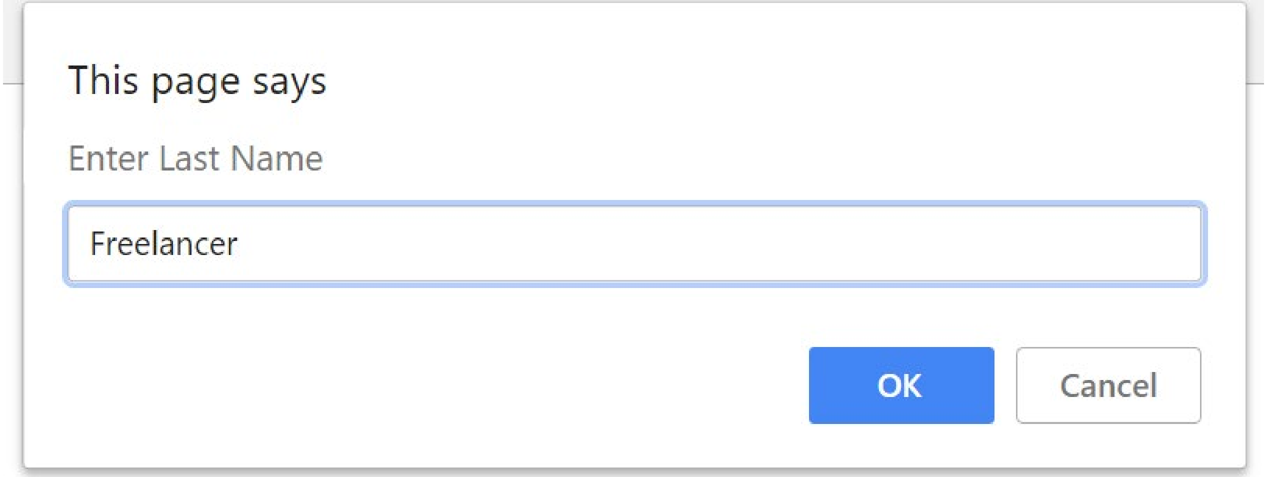
Display Name: document.write("First Name: " + firstName + "<br>");



This page says

Enter First Name

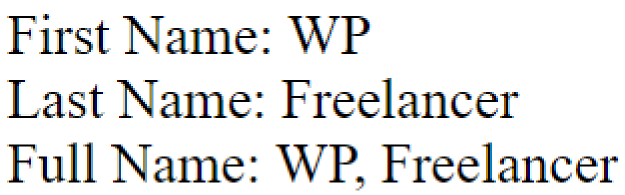
WP



This page says

Enter Last Name

Freelancer



First Name: WP
Last Name: Freelancer
Full Name: WP, Freelancer

Live Preview

2.3 Primitives

Usage of Primitive Data Types

Primitive data types means the basics data types that can be used in JavaScript programs.

There are 3 Primitive Data Types:

1. Number Data Type
2. String Data Type
3. Boolean Data Type

Number Data Type is used to store whole, positive, negative and decimal numbers.

String Data Type is used to store character data.

Boolean Data Type is used to store true and false values.

Numeric Data:

```
var count = 1;
```

String Data:

```
// Use Single Quote or Double Quotes

var fullName = “WPFreelancer”;
var lastName = ‘WPFreelancer’;
```

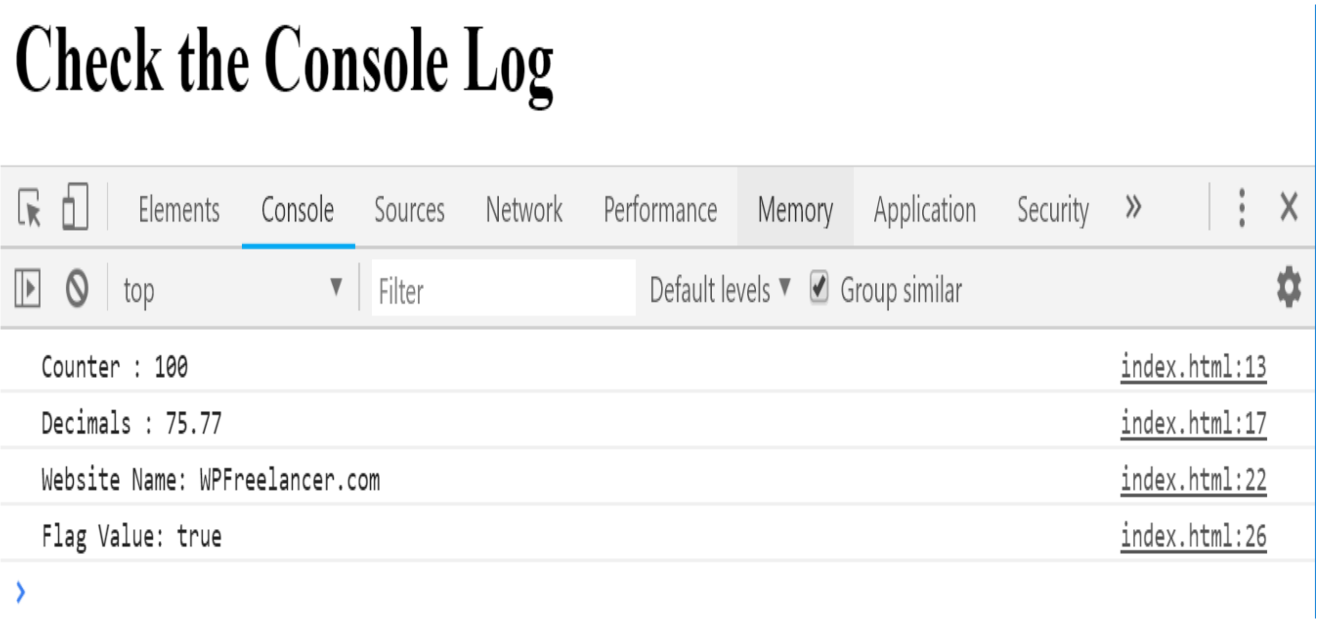
Boolean Data:

```
var flag = true;
var results = false;
```

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Primitive</title>
    <script>

        //Numeric Data Type
        var counter = 0;
        counter = counter + 100;
        console.log("Counter : " + counter);

        var decimalcount = 1.45;
        decimalcount = decimalcount + 74.32;
        console.log("Decimals : " + decimalcount);

        //String Data Type
        var fullName = "WPFreelancer";
        fullName = fullName + '.com';
        console.log("Website Name: " + fullName);

        //Boolean Data Type
        var flag = true;
        console.log("Flag Value: " + flag);


    </script>

</head>

<body>
<h1>Check the Console Log</h1>

</body>

</html>
```



Check the Console Log

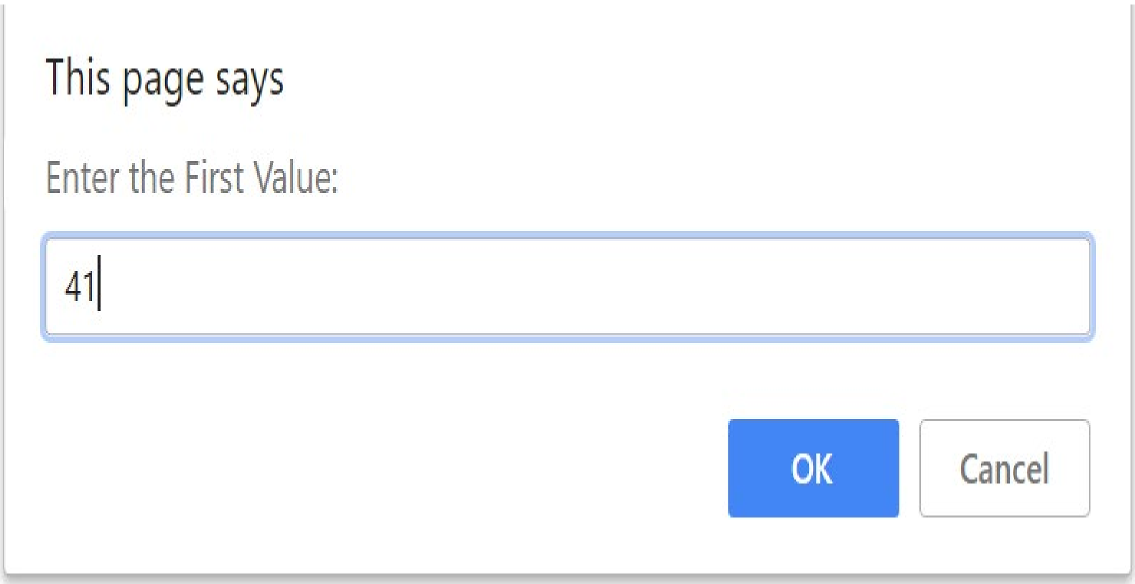
```
Counter : 100                          index.html:13
Decimals : 75.77                       index.html:17
Website Name: WPFreelancer.com         index.html:22
Flag Value: true                       index.html:26
```

Live Preview

Exercise 1

Download the Exercise 1

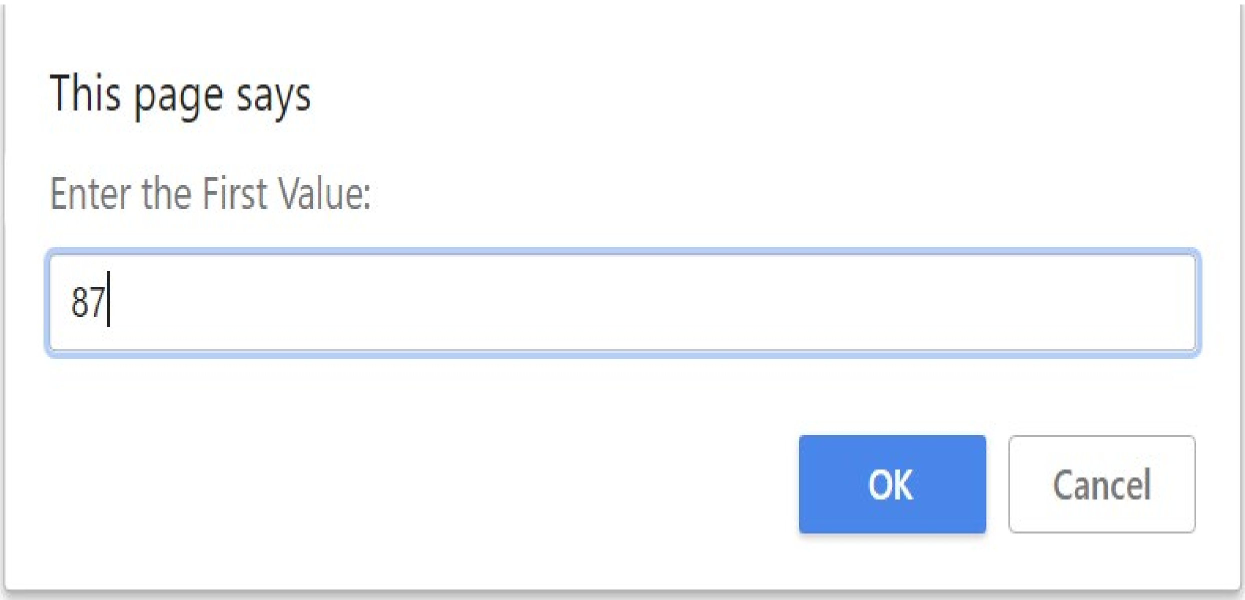
Exercise 1: Accept two numbers using prompt. Add those numbers and display on the browser.



This page says

Enter the First Value:

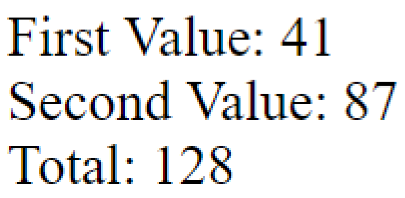
41



This page says

Enter the First Value:

87



```
First Value: 41
Second Value: 87
Total: 128
```

Tip:

Use parseInt() method to convert the string to integer.

```
var firstValue = prompt("Enter the First Value: ");
var firstValue = parseInt( firstValue);
```

[Live Preview]

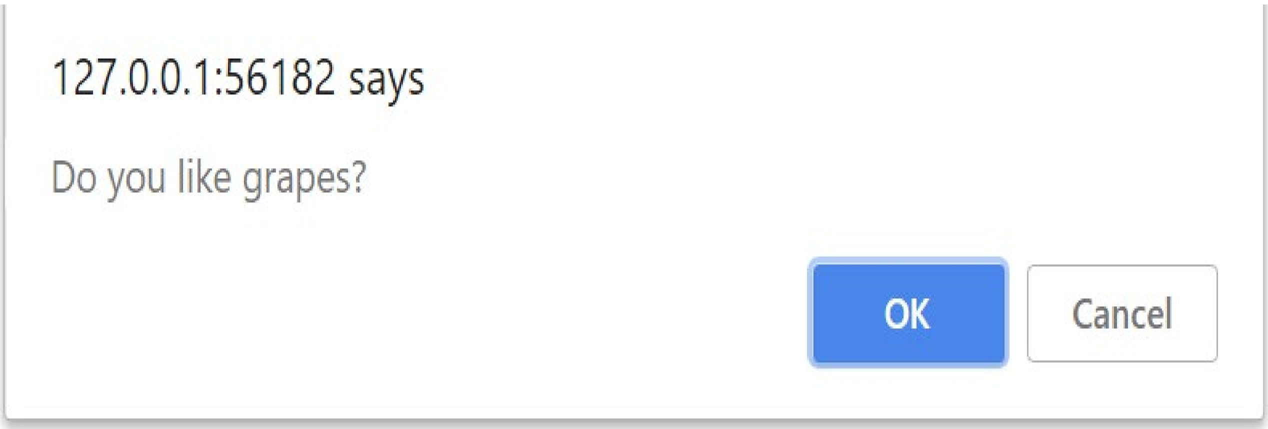
Exercise 2

[Download the Exercise 2]

Exercise 2: Ask user a question and print if they click ok or cancel.

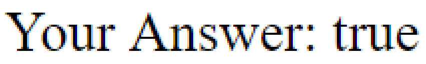
Tip:

```
var userAnswer = confirm("Do you like grapes?");
```



127.0.0.1:56182 says

Do you like grapes?



Your Answer: true

[Live Preview]

2.4 Keywords

Keywords

These are the list of the keywords that you should not use to define variables.

Keywords should not be used as Identifiers.

Keywords separated with space:

abstract arguments boolean break byte case catch char
class const continue debugger default delete do
double else enum eval export extends false final
finally float for function goto if implements import
in instanceof int interface let long native new
null package private protected public return short static super

switch synchronized this throw throws transient true
try typeof var void volatile while with yield

2.5 Reserved Words

Reserved Words

These are the list of the reserved words that you should not use to define variables, functions, objects or properties.

Reserved words should not be used as Identifiers.

Reserved Words separated with space:

Array Date eval function hasOwnProperty Infinity isFinite
isNaN isPrototypeOf length Math NaN name Number Object
prototype String toString undefined valueOf

3.Expression in

3 Expressions in JavaScript

3.1 Assignment Expressions

Usage of Assignment Expressions

Expressions are evaluated into a result value or final value or single value.

There are couple of Expressions:

1. **Assignment Expressions**
2. Comparison Expressions
3. Arithmetic Expressions
4. Logical Expressions

We will look at Assignment Expressions in this topic.

Expressions uses Operators to perform the activity.

= is the **assignment operator** which assigns the value to the variable.

message = “something”; is a assignment expression that assign value to the message variable.

Compound Assignment Operators

Compound assignment operators helps to do more than one operators job. It combines two operators together to perform an action.

Compound Assignment Operators are:

- +=

- -=
- *=

counter **+=** 1;
counter = counter + 1;

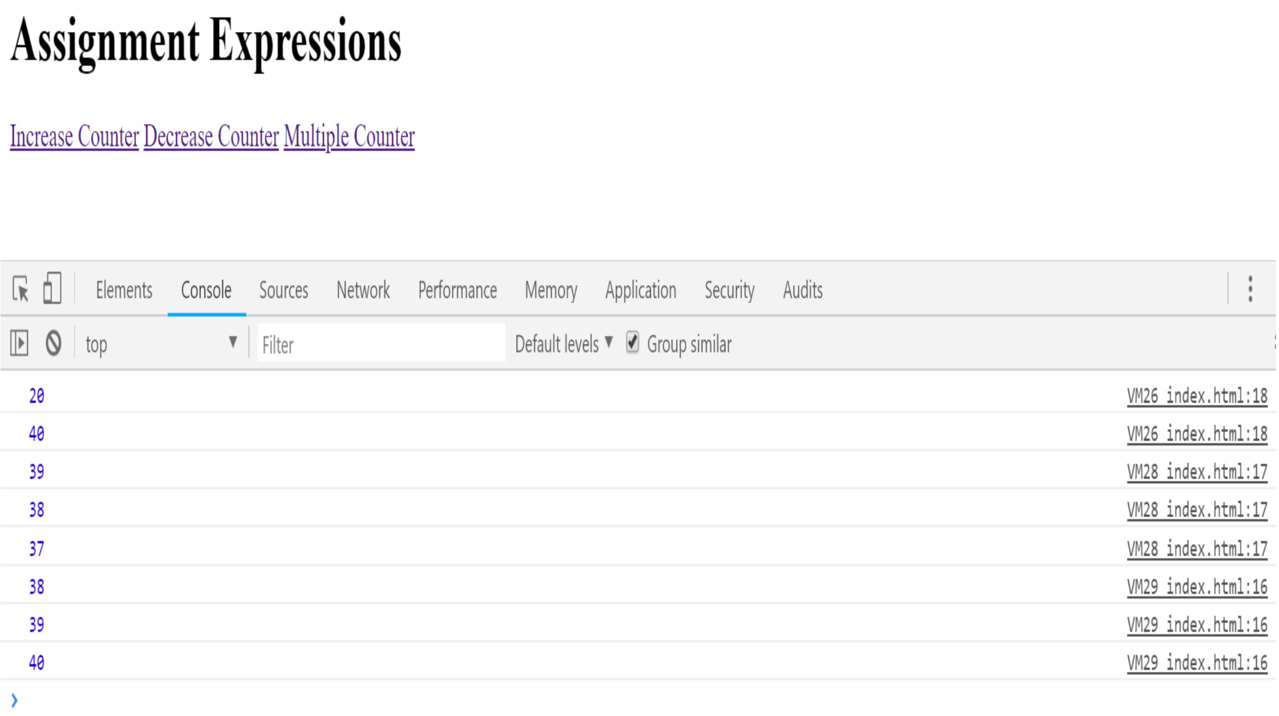
counter **-=** 1;
counter = counter – 1;

counter ***=** 1;
counter = counter * 1;

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Assignment Expressions</title>
    <script>
        var counter = 10;
    </script>

</head>

<body>
<h1>Assignment Expressions</h1>
    <a href="#" onclick="counter+=1;console.log(counter);">Increase
Counter</a>
    <a href="#" onclick="counter-=1;console.log(counter);">Decrease
Counter</a>
    <a href="#" onclick="counter*=2;console.log(counter);">Multiple
Counter</a>
</body>
</html>
```



Assignment Expressions

Increase Counter Decrease Counter Multiple Counter

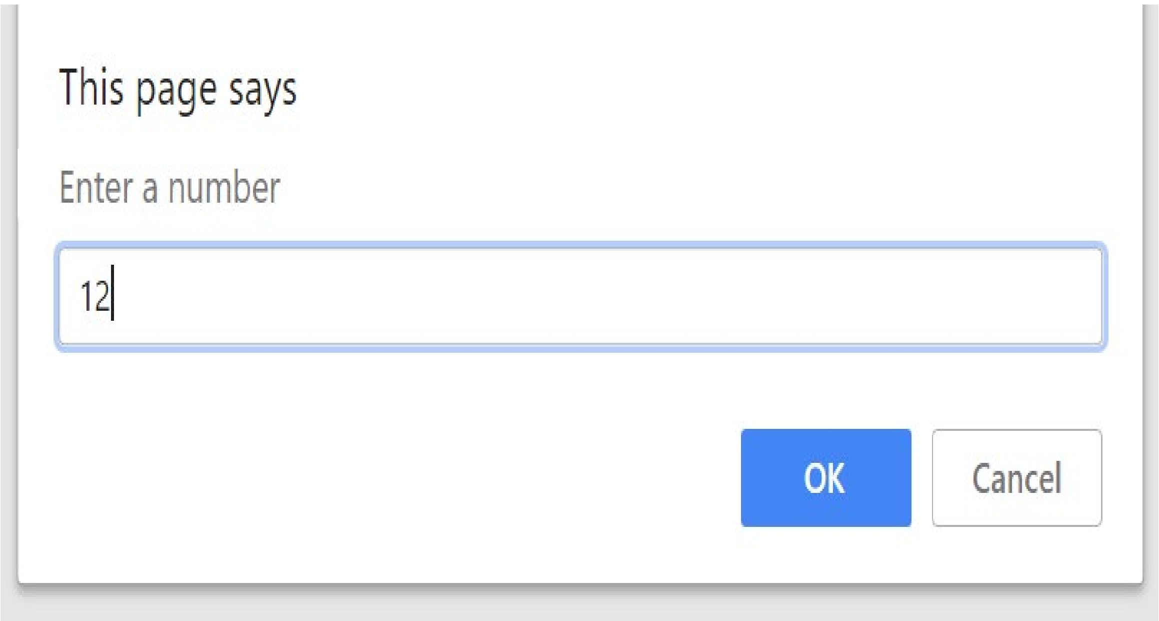
```
20                                                          VM26 index.html:18
40                                                          VM26 index.html:18
39                                                          VM28 index.html:17
38                                                          VM28 index.html:17
37                                                          VM28 index.html:17
38                                                          VM29 index.html:16
39                                                          VM29 index.html:16
40                                                          VM29 index.html:16
>
```

[Live Preview]

Exercise 1

[Download the Exercise 1]

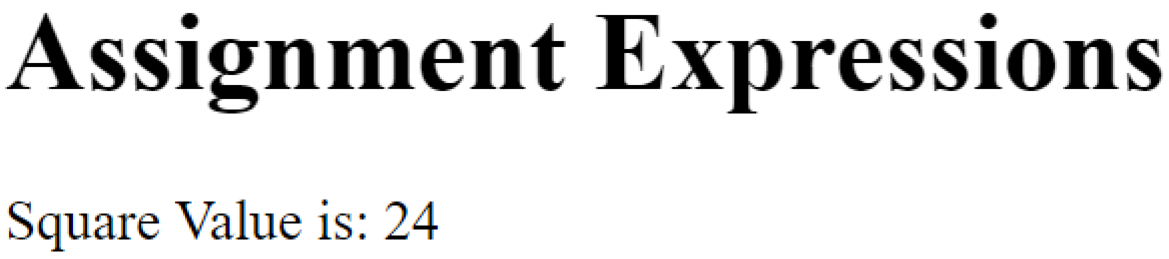
Exercise 1: Ask user to enter a number and display the square of this number.



This page says

Enter a number

12



Assignment Expressions

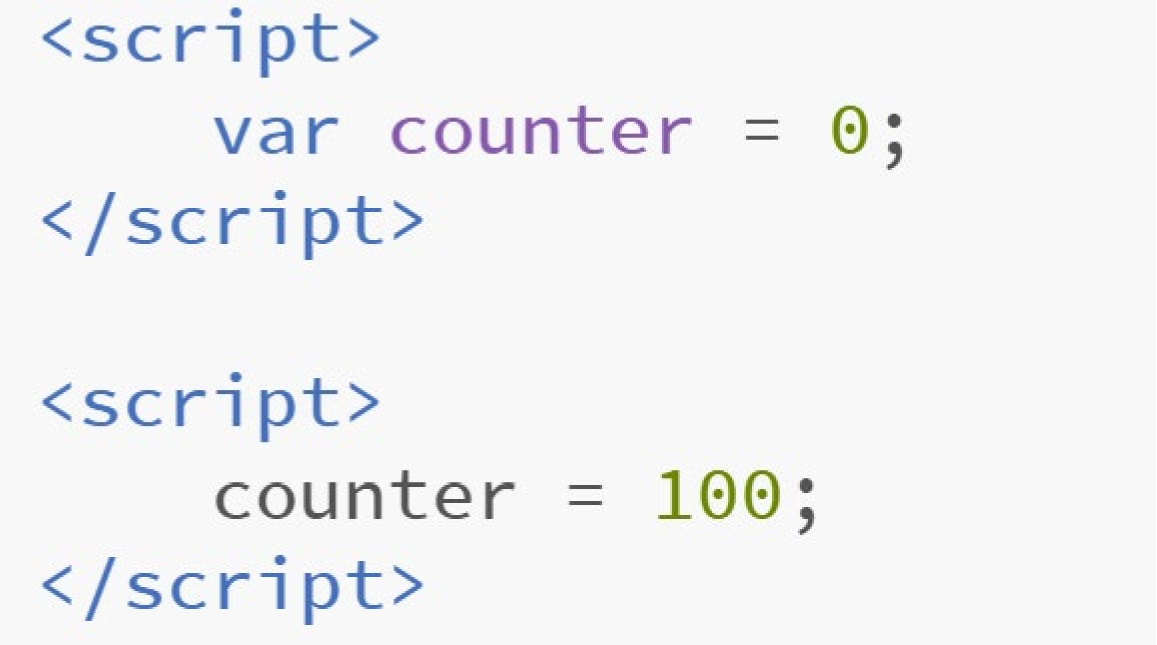
Square Value is: 24

Live Preview

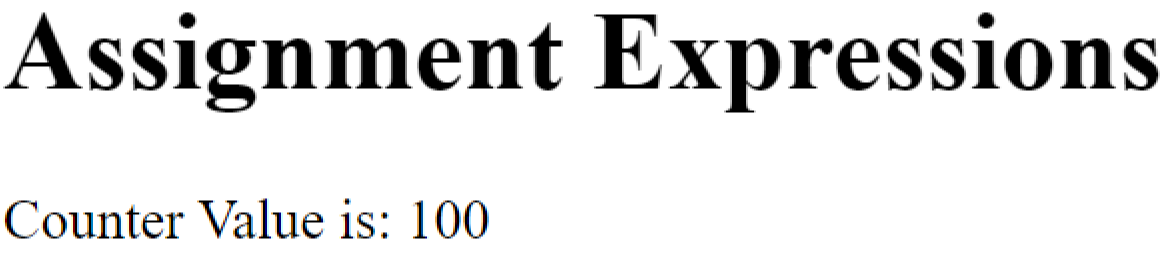
Exercise 2

Download the Exercise 2

Exercise 2: Create two <script> tags. Define variable in first <script> tag and assign value in another <script>. Finally display it on the body.



```
<script>
    var counter = 0;
</script>

<script>
    counter = 100;
</script>
```



Assignment Expressions

Counter Value is: 100

Live Preview

3.2 Comparison Expressions

Usage of Comparison Expressions

Expressions are evaluated into a result value or final value or single value.

There are couple of Expressions:

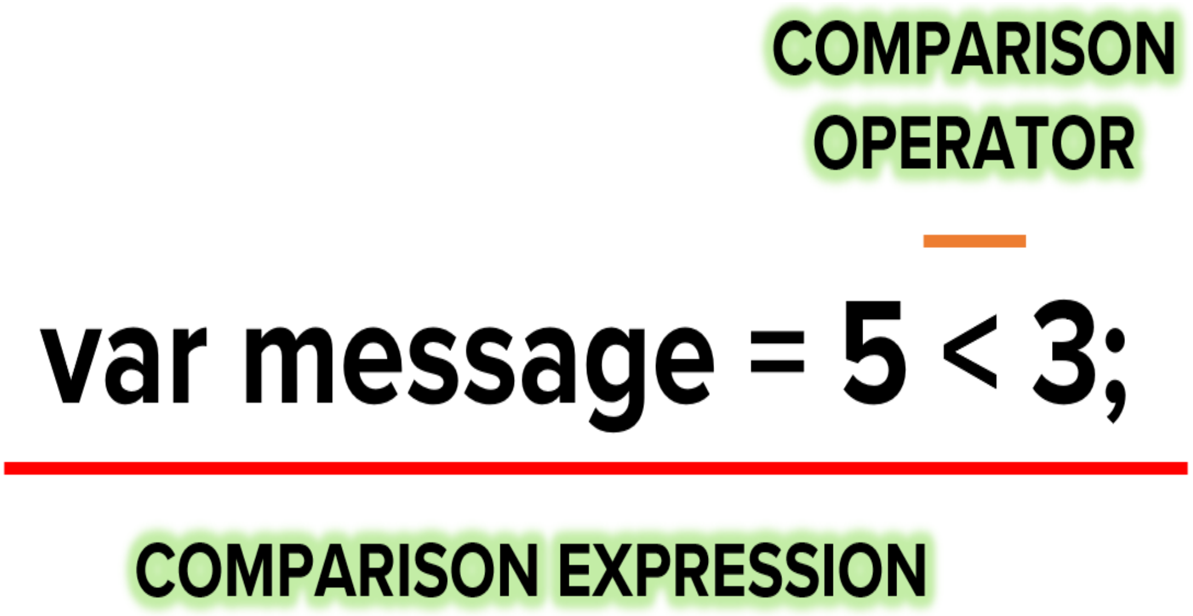
3 Expressions in JavaScript

1. Assignment Expressions
2. Comparison Expressions
3. Arithmetic Expressions
4. Logical Expressions

We will look at Comparison Expressions in this topic.

Expressions uses Operators to perform the activity.

Comparison expression always evaluate into a true or false value.



> is the **comparison operator** which compares the two values.

message variable will be have a true or false.

Comparison Operations:

- < – Less than
- > – Greater than
- == – Equal to
- === – Equal value and Equal Data Type
- !== – Not Equal Value and Equal Data Type
- != – Not Equal
- >= – Greater than or Equal to

Become PHP Full Stack Web Developer in Just 30 Days

- <= – Less than or Equal to

Conditional (Ternary) Operator

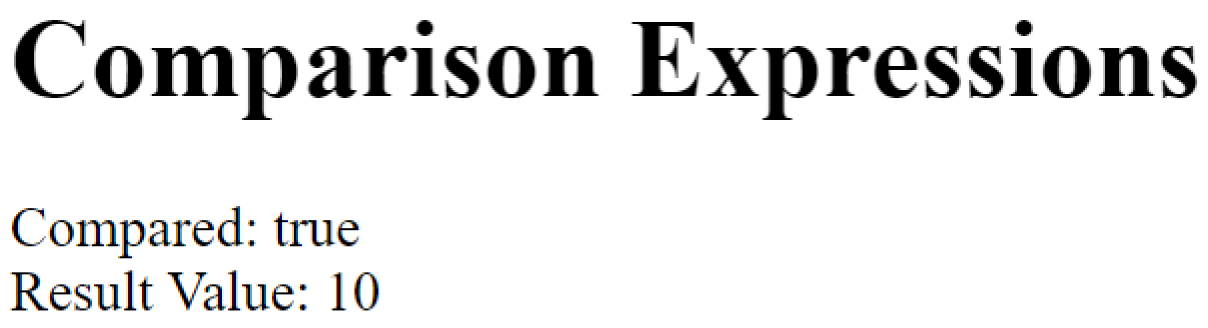
JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.

variablename = (condition) ? value1:value2
counter = (10<=10) ? 10 : 0;

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Comparison Expressions</title>
    <script>
        var counter = 10 <= 10;

        //Conditional (Ternary) Operator
        var resultValue = (counter) ? 10:0;

    </script>

</head>

<body>
<h1>Comparison Expressions</h1>
    <script type="text/javascript">
        document.write("Compared: " + counter + "<br>");
        document.write("Result Value: " + resultValue);
    </script>
</body>
</html>
```



Comparison Expressions

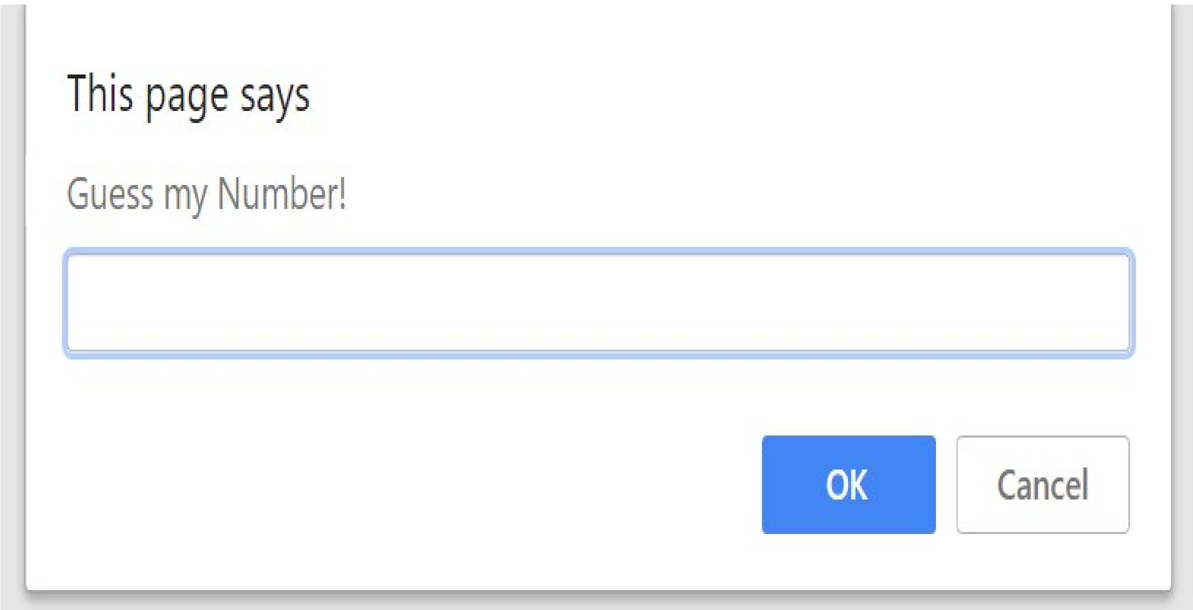
Compared: true
Result Value: 10

[Live Preview]

Exercise 1

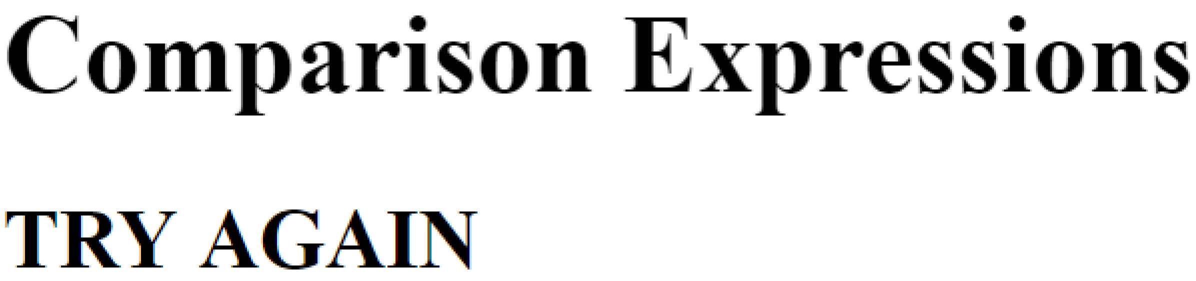
[Download the Exercise 1]

Exercise 1: Ask user to guess your number. When the number is matched with 7 then show "WINNER' word or show 'TRY AGAIN'



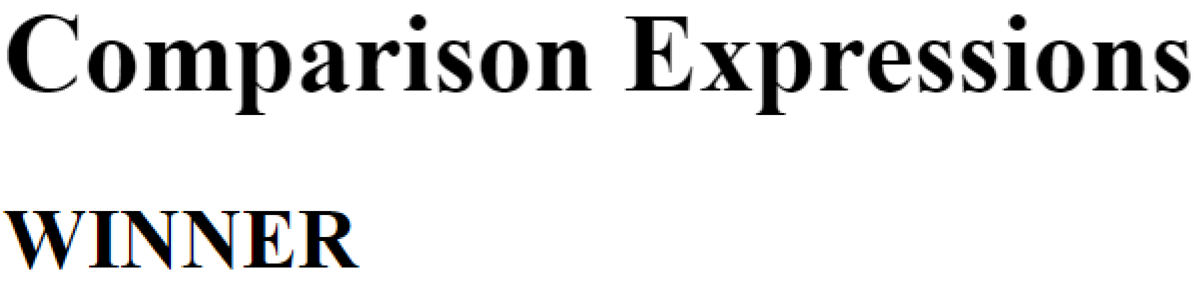
This page says

Guess my Number!



Comparison Expressions

TRY AGAIN



Comparison Expressions

WINNER

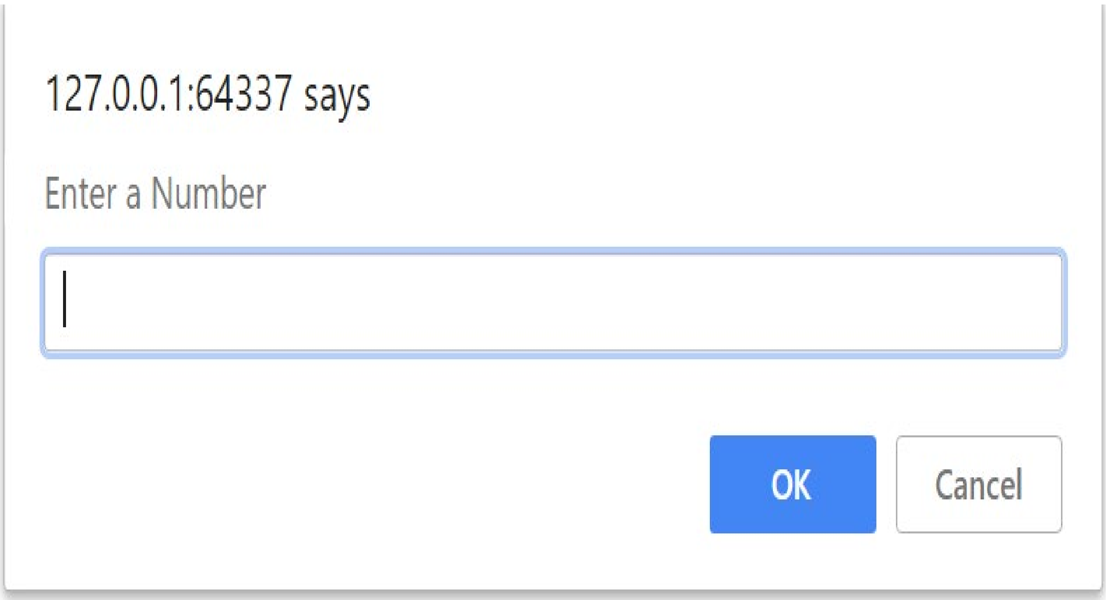
Live Preview

Exercise 2

Download the Exercise 2

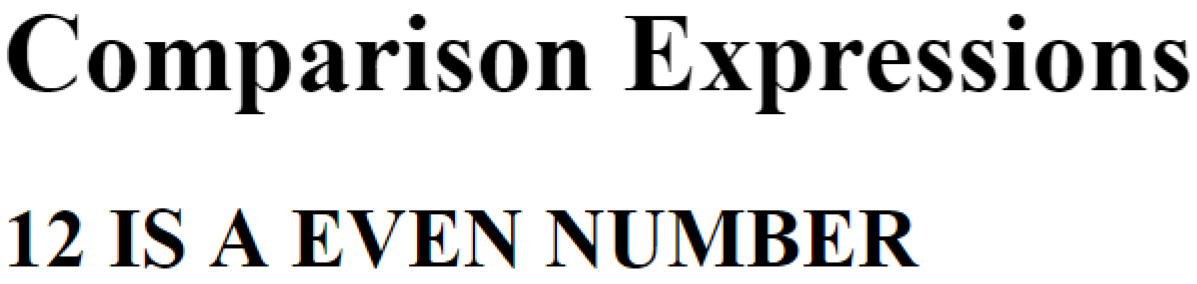
Exercise 2: Find out a Even number and Odd Number from the number entered by the User.

Tips:
var resultValue = (counter%2) ? “ODD NUMBER”: “EVEN NUMBER”;



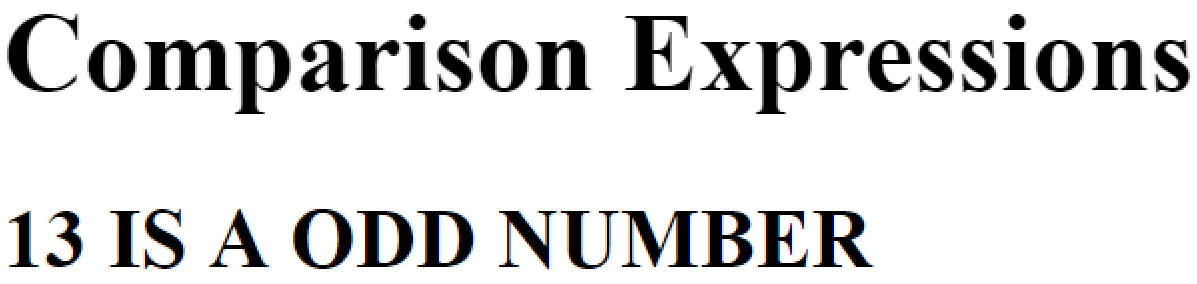
127.0.0.1:64337 says

Enter a Number



Comparison Expressions

12 IS A EVEN NUMBER



Comparison Expressions

13 IS A ODD NUMBER

Live Preview

3.3 Arithmetic Expressions

Usage of Arithmetic Expressions

Expressions are evaluated into a result value or final value or single value.

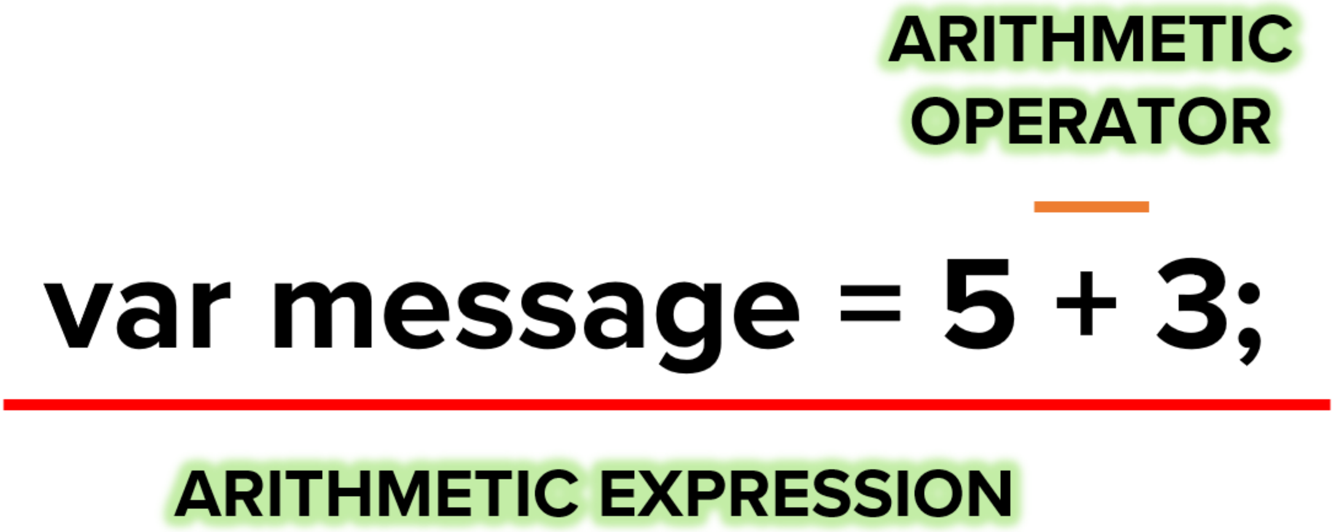
There are couple of Expressions:

1. Assignment Expressions
2. Comparison Expressions
3. **Arithmetic Expressions**
4. Logical Expressions

We will look at Comparison Expressions in this topic.

Expressions uses Operators to perform the activity.

Arithmetic expression always evaluate into a single value. A series of operations that results into a single value.



```
var message = 5 + 3;
```

+ is the **arithmetic operator** which adds two values.

message variable will be have 8 value.

Arithmetic Operations:

- + – Addition
- – Subtraction
- * – Multiple
- / – Division
- % – Modulus
- ++ – Increment
- — Decrement

Order of Precedence

Order of precedence decides which operates evaluates first.

From Left to Right, these operators has higher priority

- ++
- —
- * / %
- + –

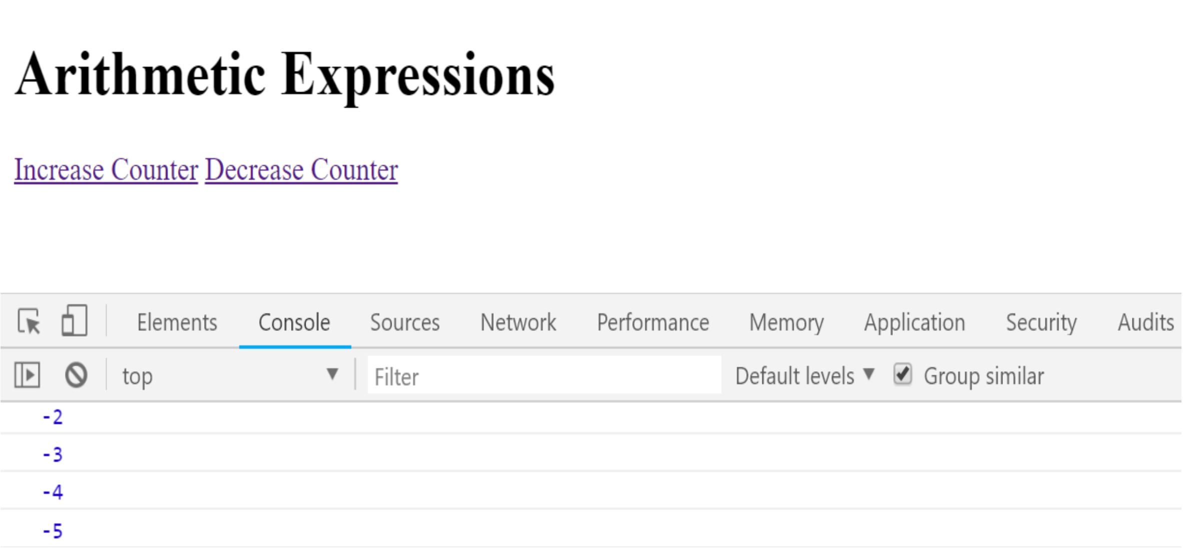
Increment Example:

counter = 10;
counter++; // This means counter = counter + 1;

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Arithmetic Expressions</title>
    <script>
        var counter = 10;
        console.log(counter);
        counter = counter + 10;
    </script>

</head>

<body>
<h1>Arithmetic Expressions</h1>
    <a href="#" onclick="counter++;console.log(counter);">Increase
Counter</a>
    <a href="#" onclick="counter--;console.log(counter);">Decrease
Counter</a>
</body>
</html>
```



Arithmetic Expressions

Increase Counter Decrease Counter

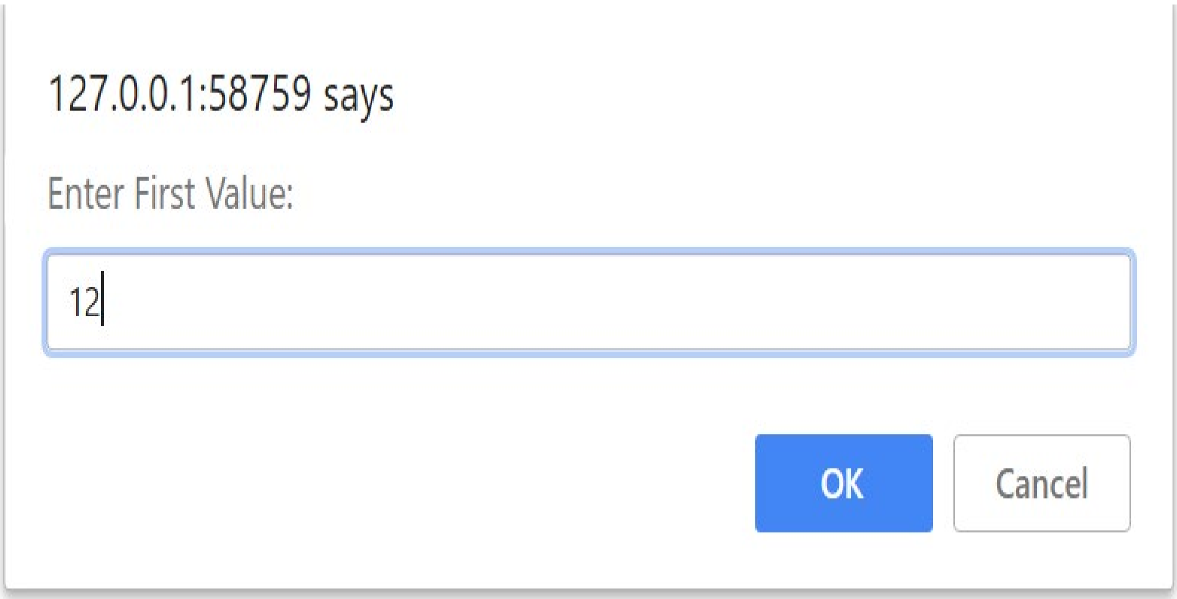
```
-2
-3
-4
-5
```

Live Preview

Exercise 1

Download the Exercise 1

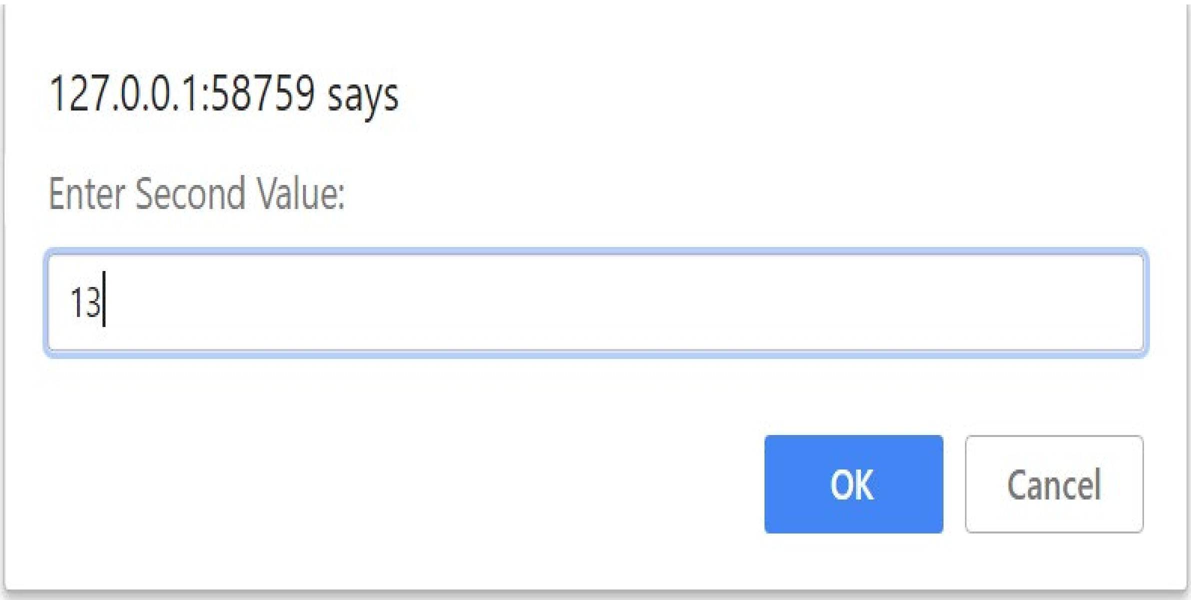
Exercise 1: Accept two numbers from user and show addition, subtraction, multiplication and division of two numbers.



127.0.0.1:58759 says

Enter First Value:

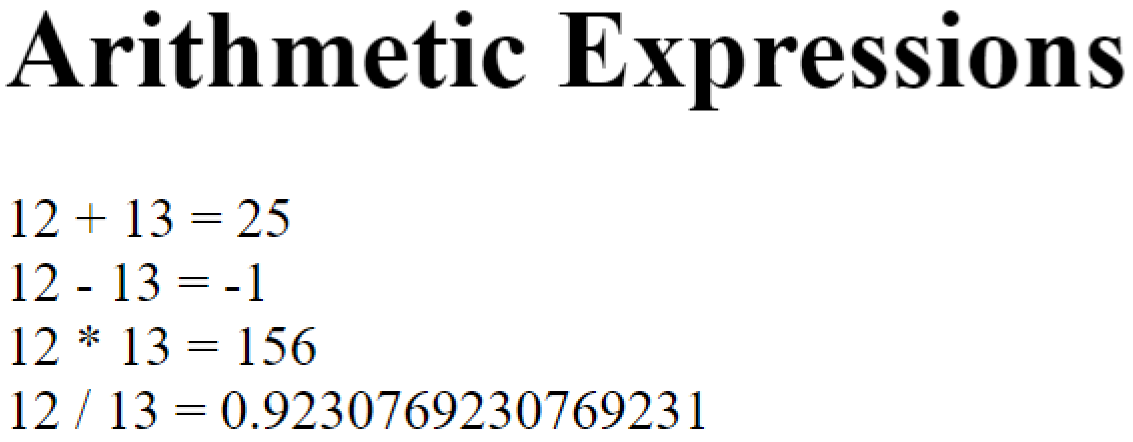
12



127.0.0.1:58759 says

Enter Second Value:

13



Arithmetic Expressions

12 + 13 = 25
12 - 13 = -1
12 * 13 = 156
12 / 13 = 0.9230769230769231

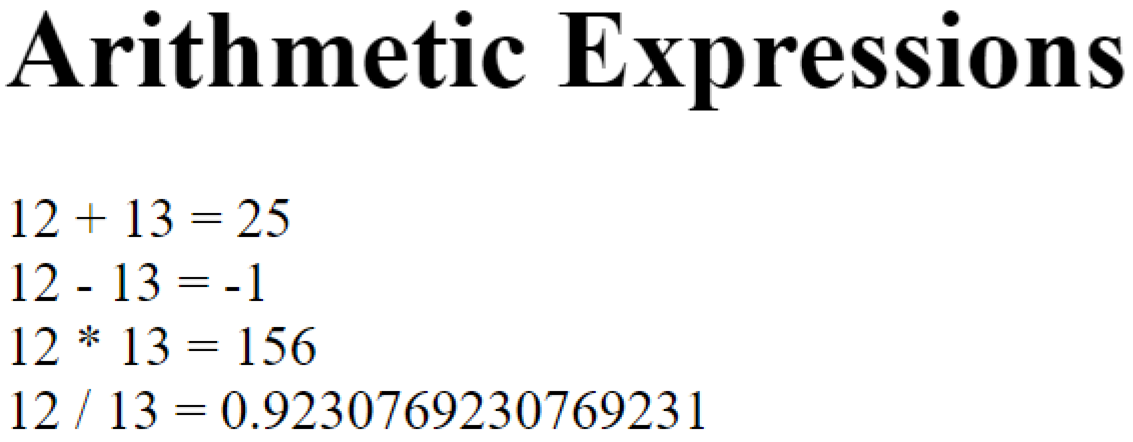
Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Ask length and breadth from the user and calculate the area of a rectangle and display on the page.

Tips:
var resultValue = length * breadth;



Arithmetic Expressions

12 + 13 = 25
12 - 13 = -1
12 * 13 = 156
12 / 13 = 0.9230769230769231

Live Preview

3.4 Logical Expressions

Usage of Logical Expressions

Expressions are evaluated into a result value or final value or single value.

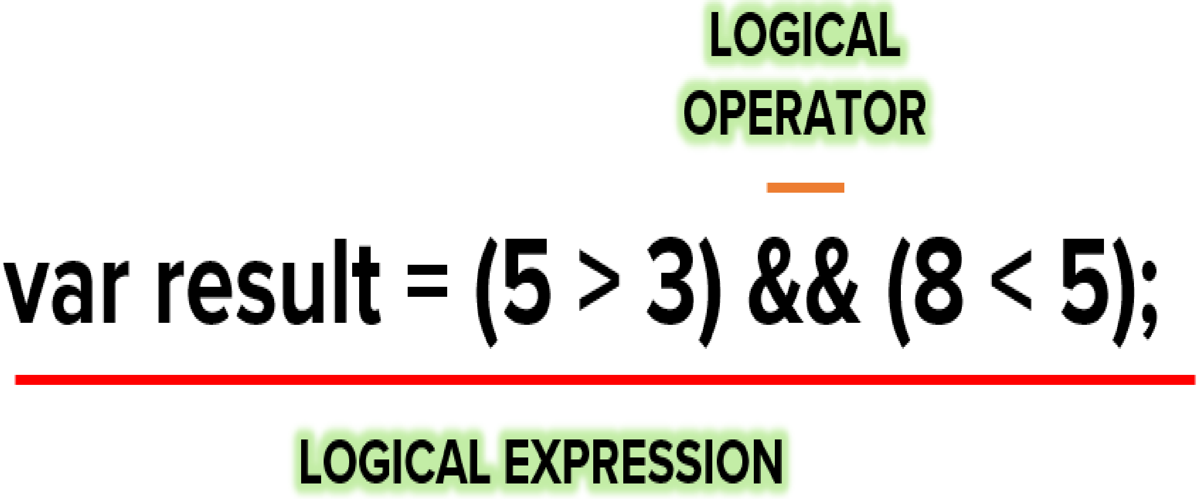
There are couple of Expressions:

1. Assignment Expressions
2. Comparison Expressions
3. Arithmetic Expressions
4. **Logical Expressions**

We will look at Logical Expressions in this topic.

Expressions uses Operators to perform the activity.

Logical Operators are used to check the if the condition is true or false based on many conditions.



```
var result = (5 > 3) && (8 < 5);
```

&& is the **logical operator** which checks left side and right side value and decides if the condition is true or false.

(5 > 3) – true
(8 < 5) – false
true && false = false

result variable will have **false** boolean value.

Logical Operations:

- && – AND
- || – OR
- ! – NOT

Order of Precedence

Order of precedence decides which operates evaluates first.

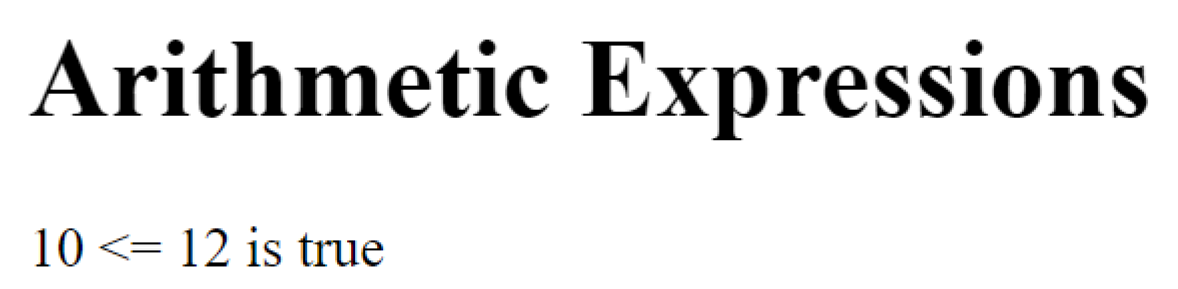
- NOT
- AND
- OR

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Arithmetic Expressions</title>
    <script>
        var input1 = prompt("Enter First Value: ");
        var input2 = prompt("Enter Second Value: ");

        var result = (input1 < input2) || (input1 == input2)

    </script>

</head>

<body>
```

```
<h1>Arithmetic Expressions</h1>
    <script type="text/javascript">
        document.write(input1 + " <= " + input2 + " is "+ result);
    </script>
</body>
</html>
```



Arithmetic Expressions

10 <= 12 is true

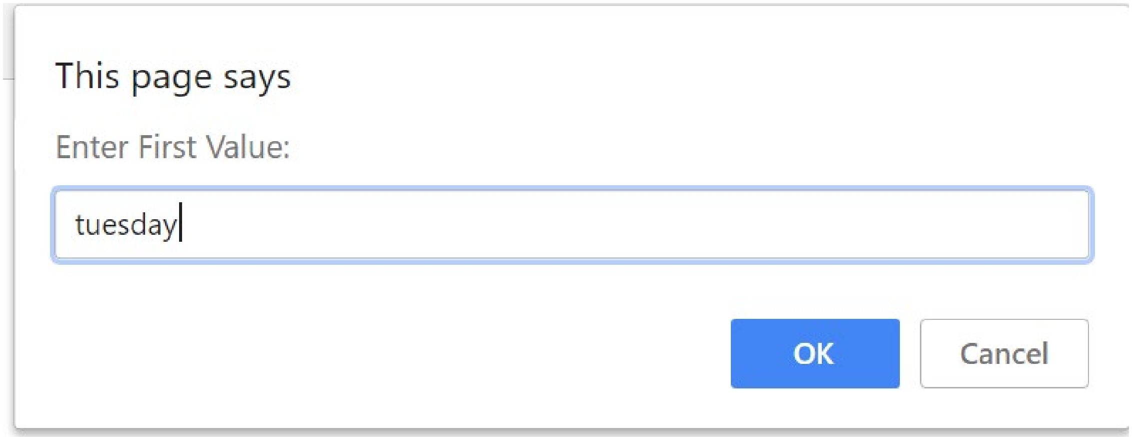
[Live Preview]

Exercise 1

[Download the Exercise 1]

Exercise 1: Ask user to enter spelling of days of the week in lowercase and check if the spelling is correct.

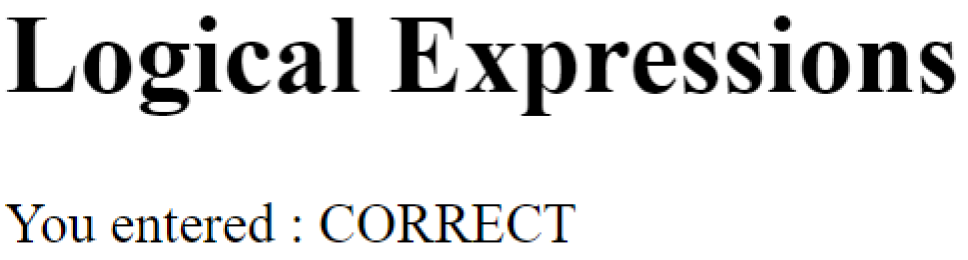
Print “CORRECT” or “WRONG” based on the condition met.



This page says

Enter First Value:

tuesday



Logical Expressions

You entered : CORRECT

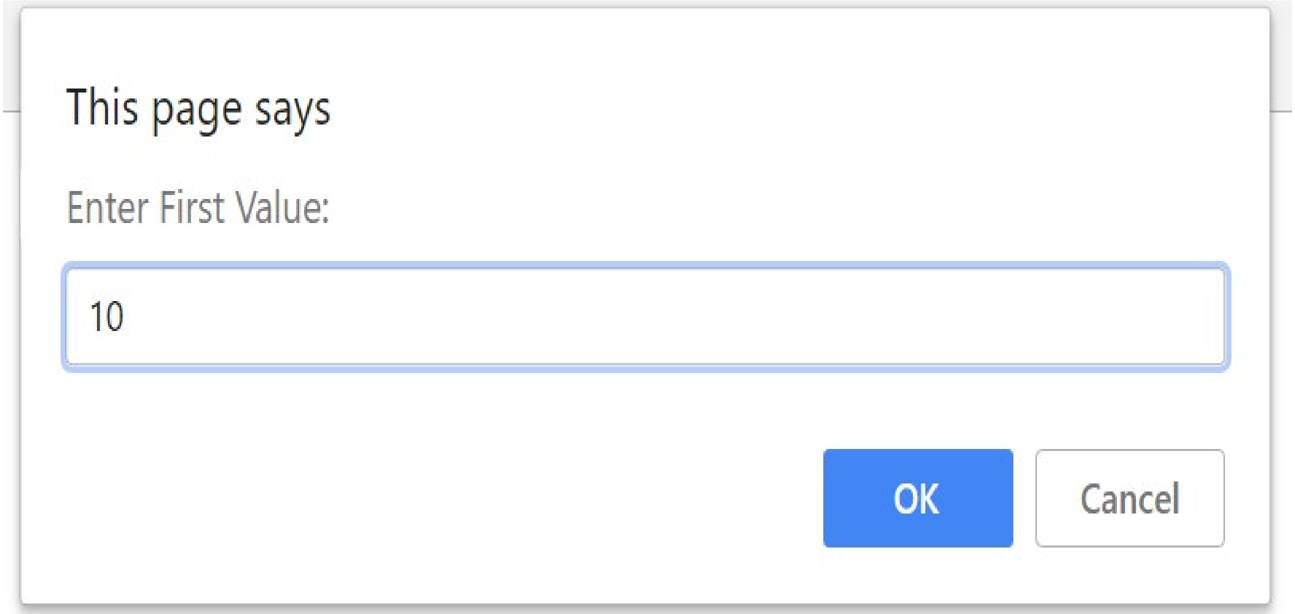
[Live Preview](#)

Exercise 2

[Download the Exercise 2](#)

Exercise 2: Ask user to enter a value between 1 to 10 and confirm it is correct or wrong.

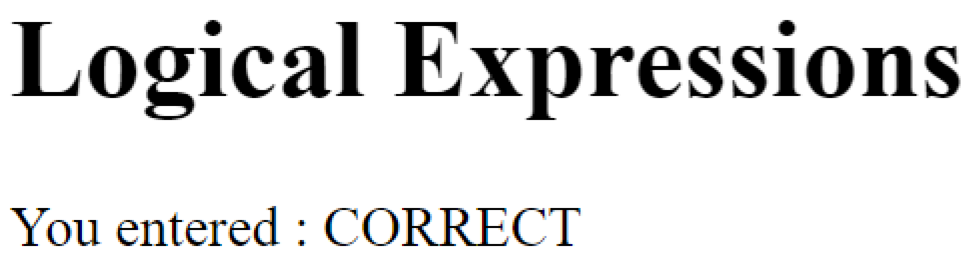
Print “CORRECT” or “WRONG” based on the condition met.



This page says

Enter First Value:

10



Logical Expressions

You entered : CORRECT

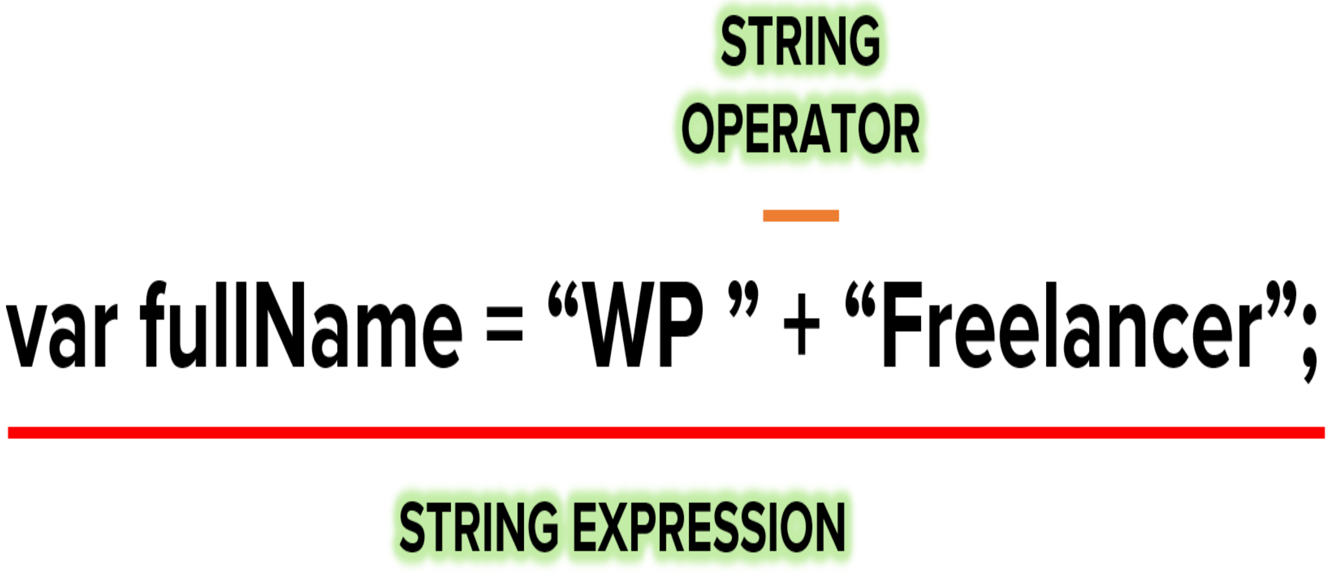
[Live Preview]

3.5 String Operations

Usage of String Operations

There two string operators that can be used to join the string:

1. +
2. +=



```
var fullName = “WP ” + “Freelancer”;
```

+ is the string operator that can be used anywhere to join two strings.

Examples:

var name = “Firstname” + “lastname”;
var name += “!”; //Add the value at the last.

String Methods:

- indexOf(search, position)
- substr(start, length)
- substr(start, stop)
- toLowerCase()
- toUpperCase()

var name = “WP Freelancer”;

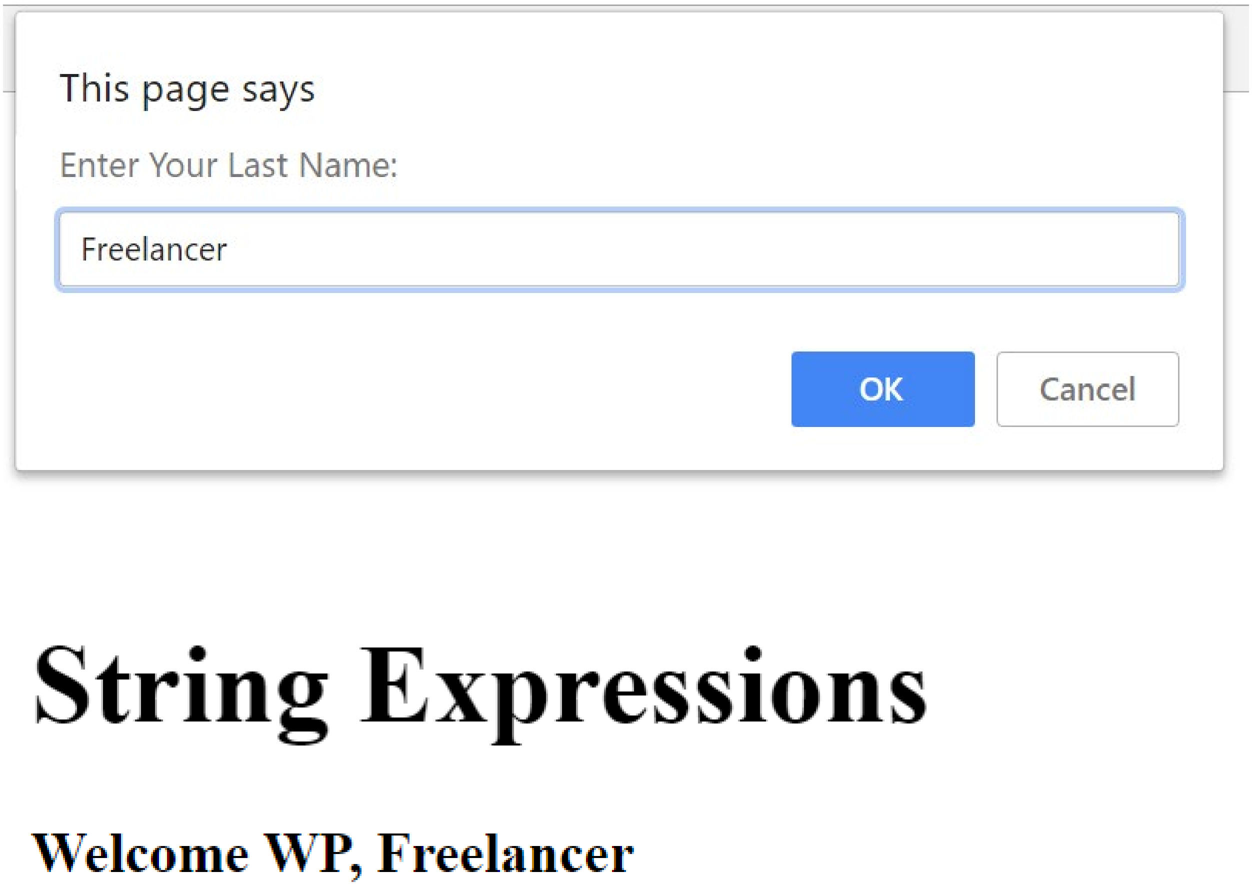
alert(name.toLowerCase());

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>String Expressions</title>
    <script>
        var input1 = prompt("Enter Your First Name: ");
        var input2 = prompt("Enter Your Last Name: ");

        var userName = "<strong>Welcome " + input1 + ", " + input2 +
"</strong>";

    </script>

</head>

<body>
<h1>String Expressions</h1>
    <script type="text/javascript">
        document.write(userName);
    </script>
```

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



This page says

Enter Your Last Name:

Freelancer

String Expressions

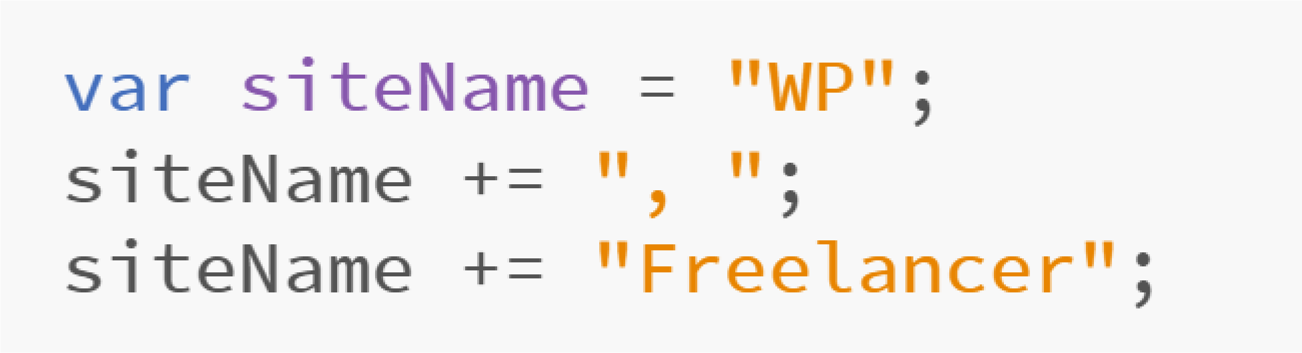
Welcome WP, Freelancer

[Live Preview]

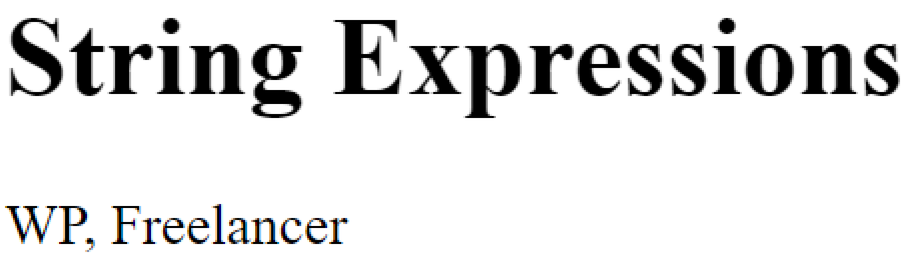
Exercise 1

[Download the Exercise 1]

Exercise 1: Try to use a the + and += operator and prepare a string and print it.



```
var siteName = "WP";
siteName += ", ";
siteName += "Freelancer";
```



String Expressions

WP, Freelancer

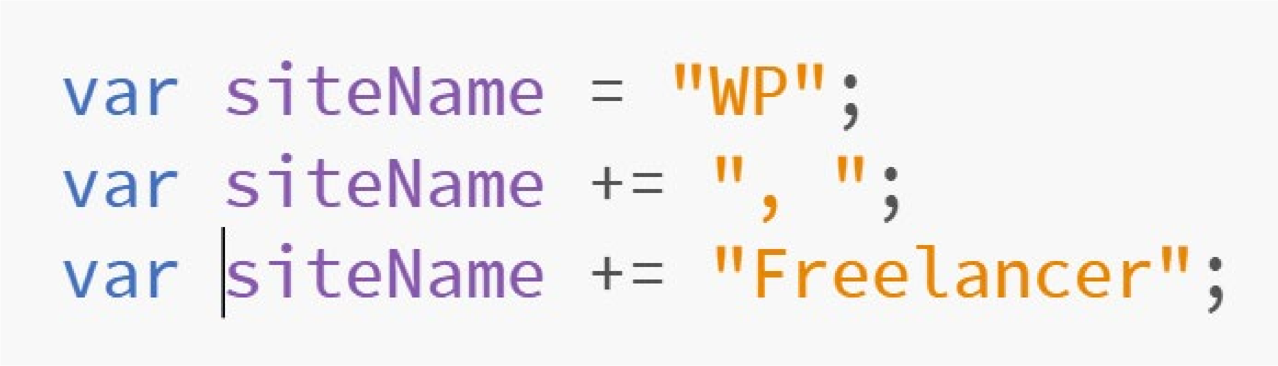
[Live Preview](#)

Exercise 2

[Download the Exercise 2](#)

Exercise 2: Define the String variable like and observe the output.

Avoid this common mistake with the string operations.



```
var siteName = "WP";
var siteName += ", ";
var siteName += "Freelancer";
```

Guess the Output.

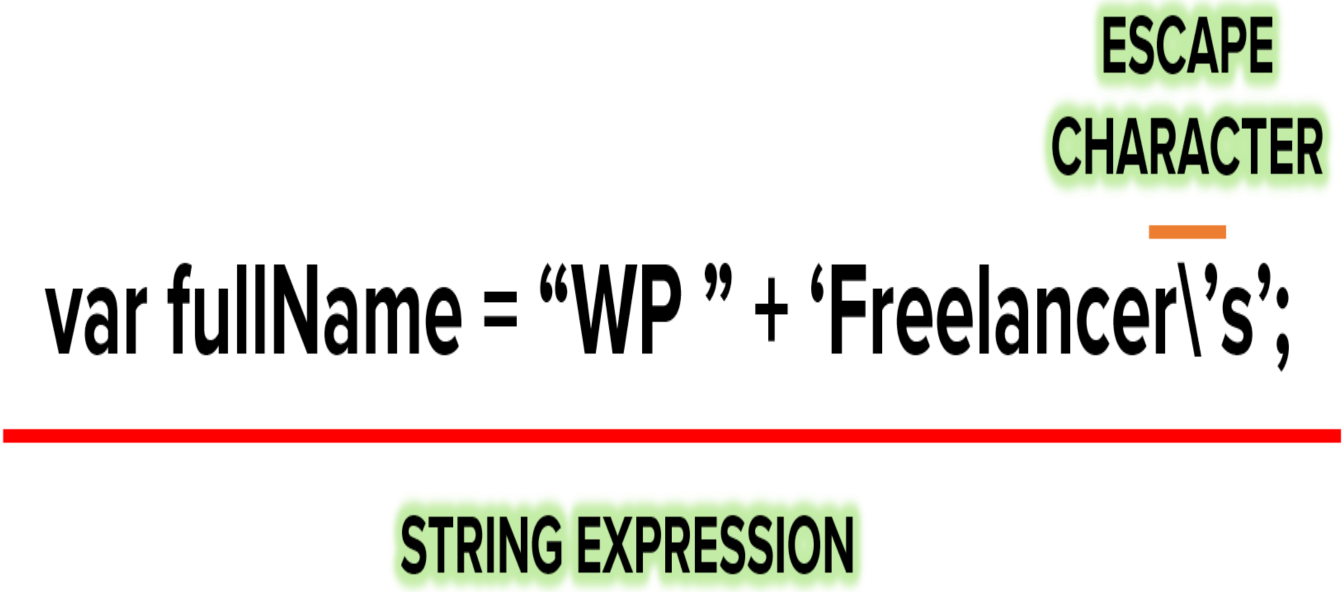
[Become PHP Full Stack Web Developer in Just 30 Days](#)

Live Preview

3.6 Quotes

Usage of Quotes in Strings

Single Quotes and Double Quotes can be used to represent String in JavaScript.



```
var fullName = “WP ” + ‘Freelancer\’s’;
```

Escape Sequence:

Sometime you need to show some special characters like single quotes or double quotes inside the content.

var message = ‘We\’ll be going to BootCamp soon!’;

\ is used to escape the characters in the string.

\n is used to add new line

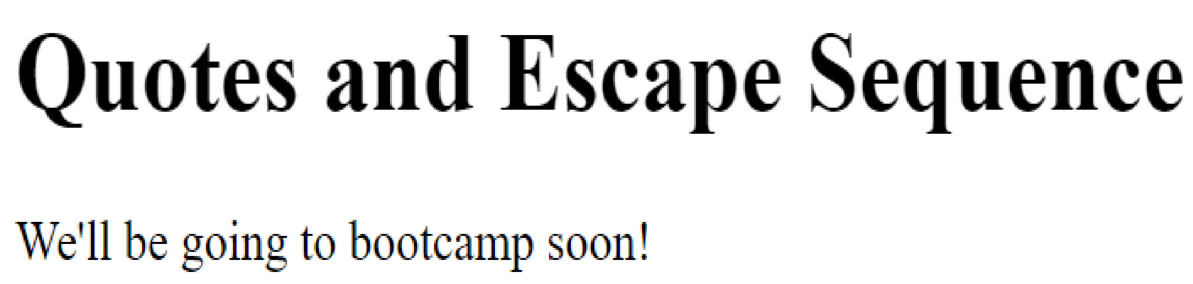
\” to escape the double quotes.

Become PHP Full Stack Web Developer in Just 30 Days

Sample Example

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>String Expressions</title>
    <script>
        var input1 = prompt("Enter Your First Name: ");
        var input2 = prompt("Enter Your Last Name: ");

        var userName = "<strong>Welcome " + input1 + ", " + input2 +
"</strong>";

    </script>

</head>

<body>
<h1>String Expressions</h1>
    <script type="text/javascript">
        document.write(userName);
    </script>
</body>
</html>
```



Quotes and Escape Sequence

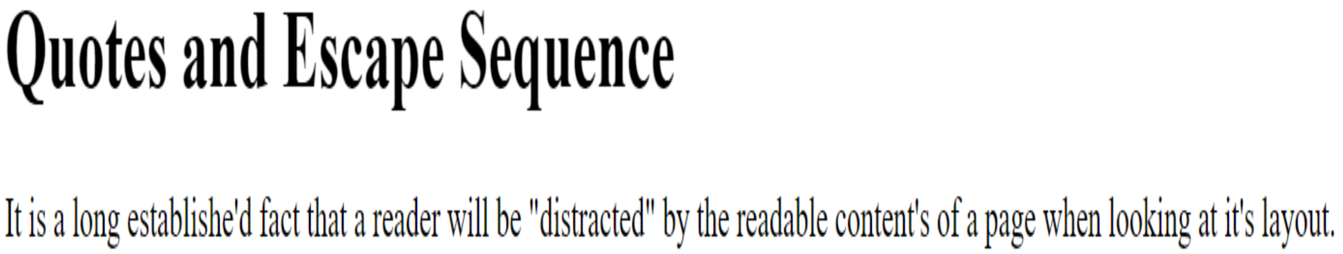
We'll be going to bootcamp soon!

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Write a paragraph with single quotes and double quote and escape single quotes.



Quotes and Escape Sequence

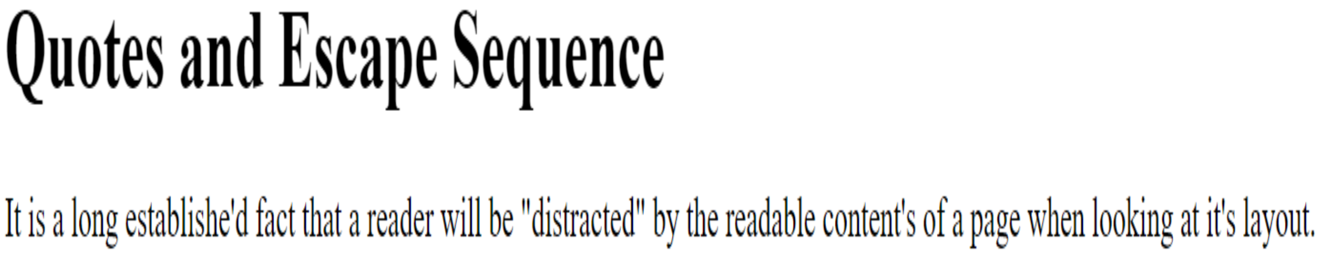
It is a long establishe'd fact that a reader will be "distracted" by the readable content's of a page when looking at it's layout.

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Write the same paragraph from above in double quotes and escape double quotes.



Quotes and Escape Sequence

It is a long establishe'd fact that a reader will be "distracted" by the readable content's of a page when looking at it's layout.

Live Preview

3.7 Boolean

Usage of Boolean

Boolean Variables helps to make decisions or store a decision based on an expression.

Boolean values can be true or false.

You can use boolean variable as a condition to check if the value is true or false.

var result = 2 > 1;
var message = (result) ? “CORRECT”: “WRONG”;

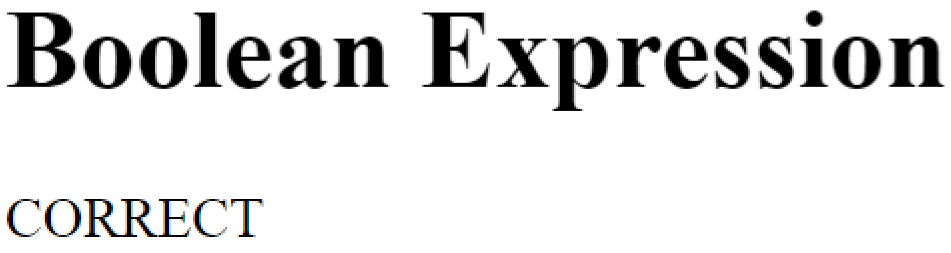
Example:
var isTrue= “Henry” == “H3nry”;

Based on the conditional operator, the output can be stored in the boolean primitive data type.

isTrue will hold false value.

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Boolean Expressions</title>
    <script>

       var result = 2 > 1;
       var message = (result) ? "CORRECT": "WRONG";

    </script>

</head>

<body>
<h1>Boolean Expression</h1>
    <script type="text/javascript">
        document.write(message);
    </script>
</body>
</html>
```



Boolean Expression

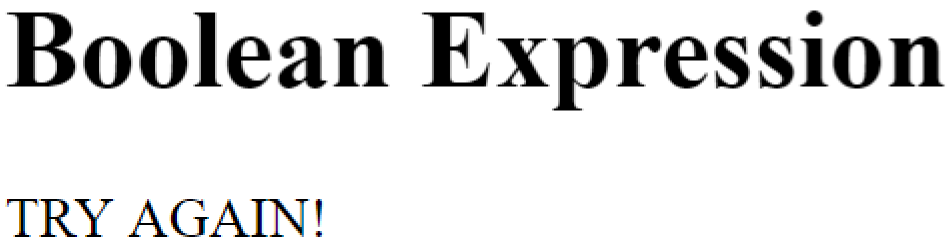
CORRECT

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Accept a number from user and if the number is 7 then show “FOUND” and if not say “TRY AGAIN!



Boolean Expression

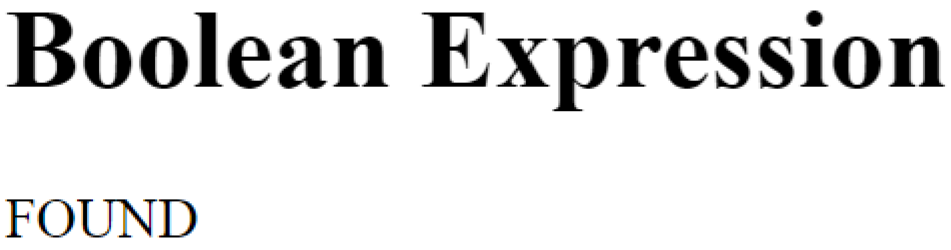
TRY AGAIN!

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Ask user to guess a word that starts with A and then match it with the word that you guessed it.



Boolean Expression

FOUND

Live Preview

3.8 Arrays

Usage of Arrays

Arrays are special type of Objects that holds one or more items called as elements.

Each element could be primitive data type or object.

length is used to indicate the number of elements in the array.

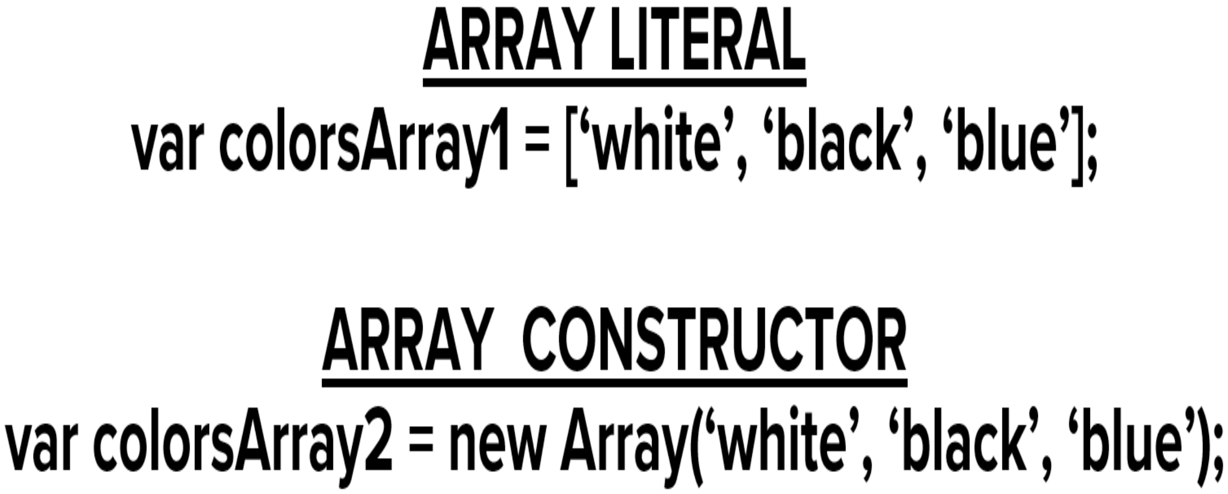
Define an Array

ARRAY LITERAL:

var arrayName = [1, 2, ‘white’, false];

ARRAY CONSTRUCTOR:

var arrayName = new Array(length / values);



ARRAY LITERAL

var colorsArray1 = [‘white’, ‘black’, ‘blue’];

ARRAY CONSTRUCTOR

var colorsArray2 = new Array(‘white’, ‘black’, ‘blue’);

Access an Array Element

```
arrayName[index]
```

Length of an Array Element

```
var lenofArray = arrayName.length;
```

Add Values to Array

```
var newArray = [];

newArray[0] = ‘white’;

newArray[1] = ‘red’;
```

Access the Values from the Array

```
var color1 = newArray[0];
```

Add new element at the end

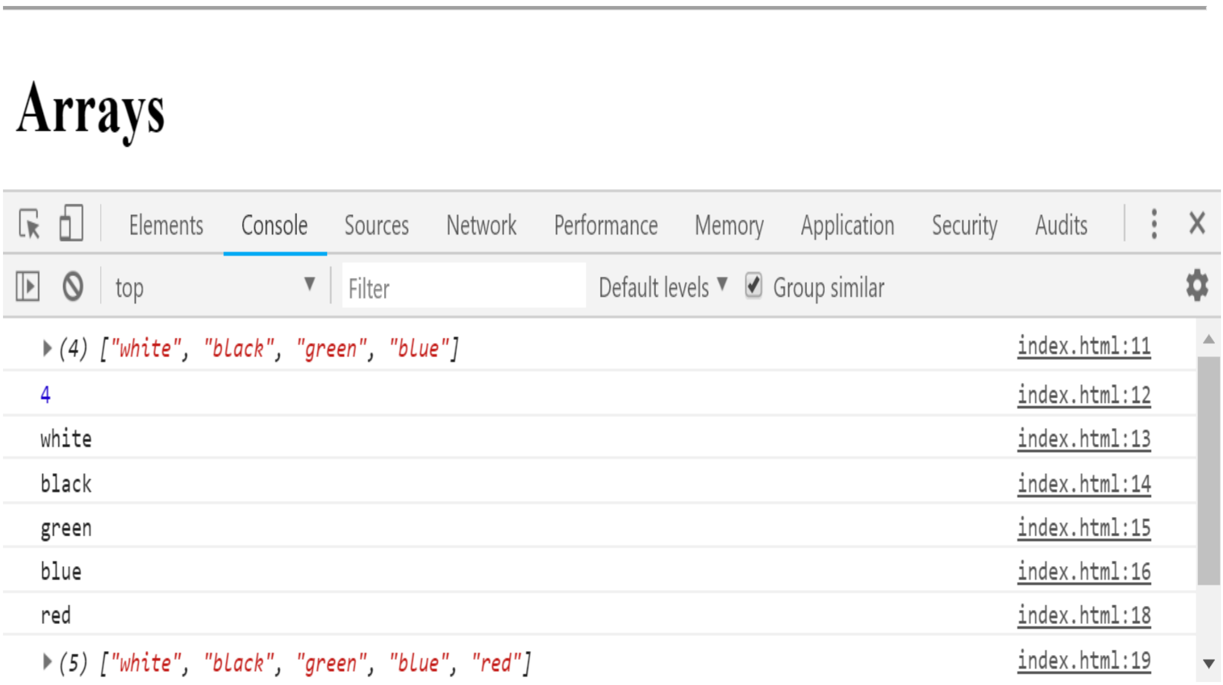
```
newArray[ newArray.length] = ‘Black’;
```

First Element of Array

```
var firstElement = newArray[0];
```

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Boolean Expressions</title>
    <script>

        var colorsArray = ['white', 'black', 'green', 'blue'];
        console.log(colorsArray);
        console.log(colorsArray.length);
        console.log(colorsArray[0]);
        console.log(colorsArray[1]);
        console.log(colorsArray[2]);
        console.log(colorsArray[3]);
        colorsArray[colorsArray.length] = 'red';
        console.log(colorsArray[4]);
        console.log(colorsArray);
        console.log(colorsArray.length);
    </script>

</head>

<body>
<h1>Arrays</h1>
</body>
</html>
```



Arrays

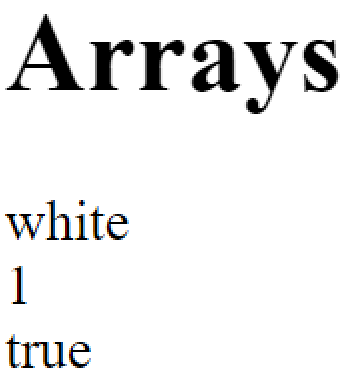
```
▸ (4) ["white", "black", "green", "blue"]              index.html:11
  4                                                    index.html:12
  white                                                index.html:13
  black                                                index.html:14
  green                                                index.html:15
  blue                                                 index.html:16
  red                                                  index.html:18
▸ (5) ["white", "black", "green", "blue", "red"]       index.html:19
```

Live Preview

Exercise 1

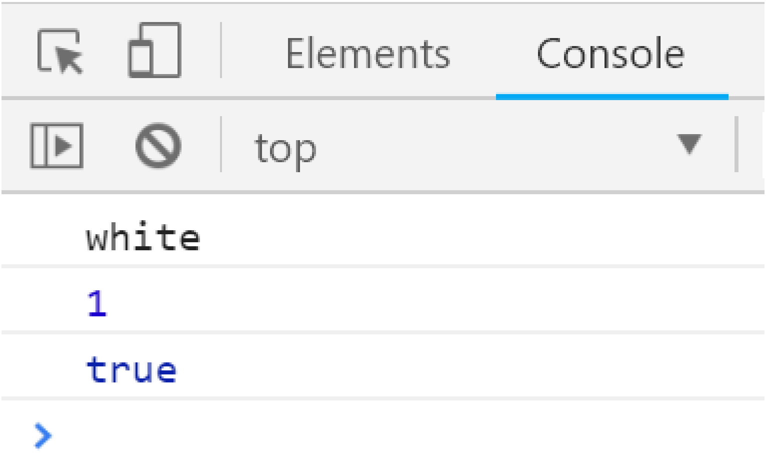
Download the Exercise 1

Exercise 1: Create an Array with String, Number and Boolean and display them.



Arrays

white
1
true



```
white
1
true
>
```

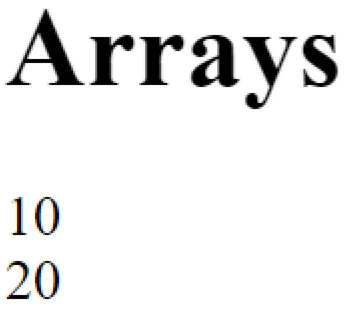
Live Preview

Exercise 2

Download the Exercise 2

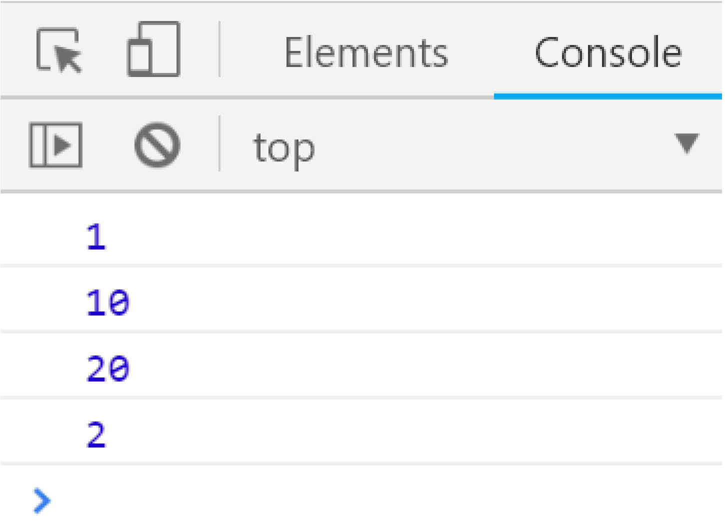
Exercise 2: Do the following exercise with Arrays

1. Define an Empty Array
2. Add 1 Element
3. Display the Length
4. Add 2 Element
5. Show the two Elements
6. Show the Length of the Array



Arrays

10
20



```
1
10
20
2
>
```

Live Preview

3.9 Date and Time

Usage of Date and Time

Date() is a class library available to access the date.

However, Date is not a primitive data type. You need to create a instance of Date to access the date functions.

var today = new Date();

By default, dt will have user system date and it can be displayed with its methods.

Date Methods

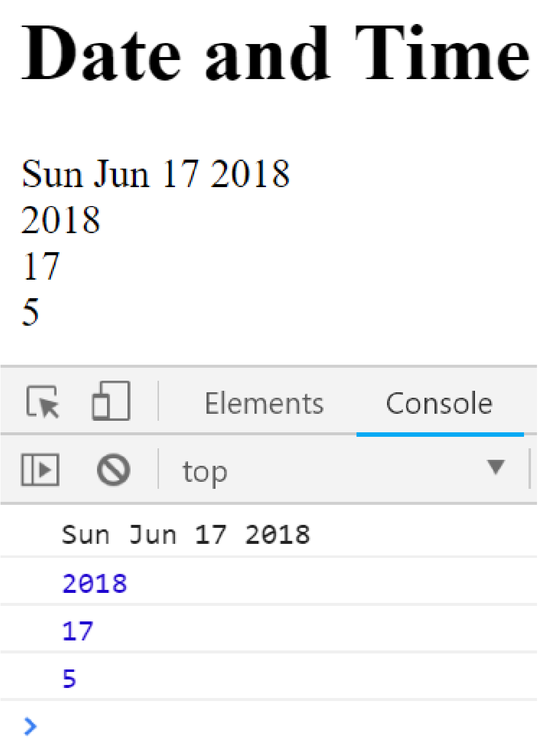
- toDateString() – Returns a date with formatted String.
- getFullyear() – Returns the 4 digit year from the Date.
- getDate() – Returns the day of the month from the Date.
- getMonth() – Fetch the month.

Examples:

var today = new Date();
alert(today.toDateString());

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Date and Time</title>
    <script>

        var today = new Date();
        console.log(today.toDateString());
        console.log(today.getFullYear());
        console.log(today.getDate());
        console.log(today.getMonth());

    </script>

</head>

<body>
<h1>Date and Time</h1>
    <script type="text/javascript">

        document.write(today.toDateString()+"<br>");
        document.write(today.getFullYear()+"<br>");
        document.write(today.getDate()+"<br>");
        document.write(today.getMonth()+"<br>");

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



Date and Time

```
Sun Jun 17 2018
2018
17
5
```

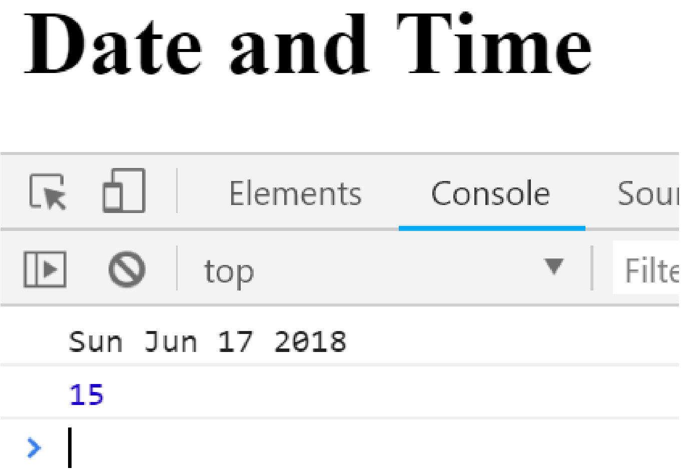
```
Sun Jun 17 2018
2018
17
5
>
```

Live Preview

Exercise 1

[Download the Exercise 1]

Exercise 1: Display length of the Date.



Date and Time

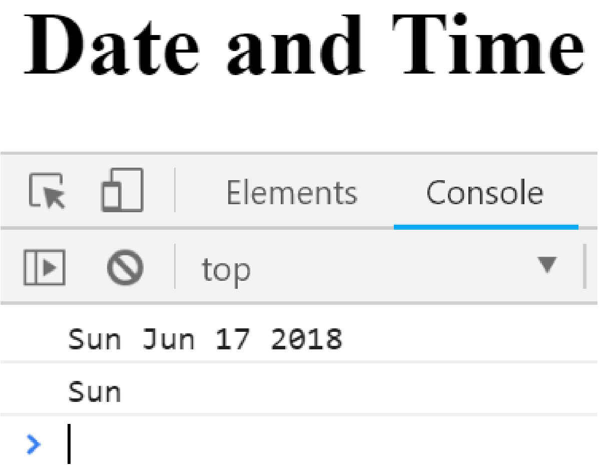
```
Sun Jun 17 2018
15
>
```

[Live Preview]

Exercise 2

[Download the Exercise 2]

Exercise 2: Display only day of the week from the Date.



Date and Time

```
Sun Jun 17 2018
Sun
>
```

Live Preview

4.Statements

4 Statements

4.1 if Statements

Usage of if Statements

if statement are used to check a condition and make a decision based on the result of the condition.

It can choose some action when the condition is true and also take some action when it false.

You can nest multiple conditions together and decide to choose one action based on multiple conditions.

Syntax:

```
if ( condition1 ){

// Statements

} else if( condition1 || condition2 ){

// Statements

} else {

// Statements

}
```

Example 1:

```
if( marks > 35 ){

alert(“You are Passed!”);
```

}else{

alert(“Try Again!”);

}

Example 2:

if(marks > 35 && marks < 60){

alert(“You are Passed with Grade C”);

}else if (marks > 60 && marks < 80){

alert(“You are Passed with Grade B”);

}else if (marks > 80){

alert(“You are Passed with Grade A”);

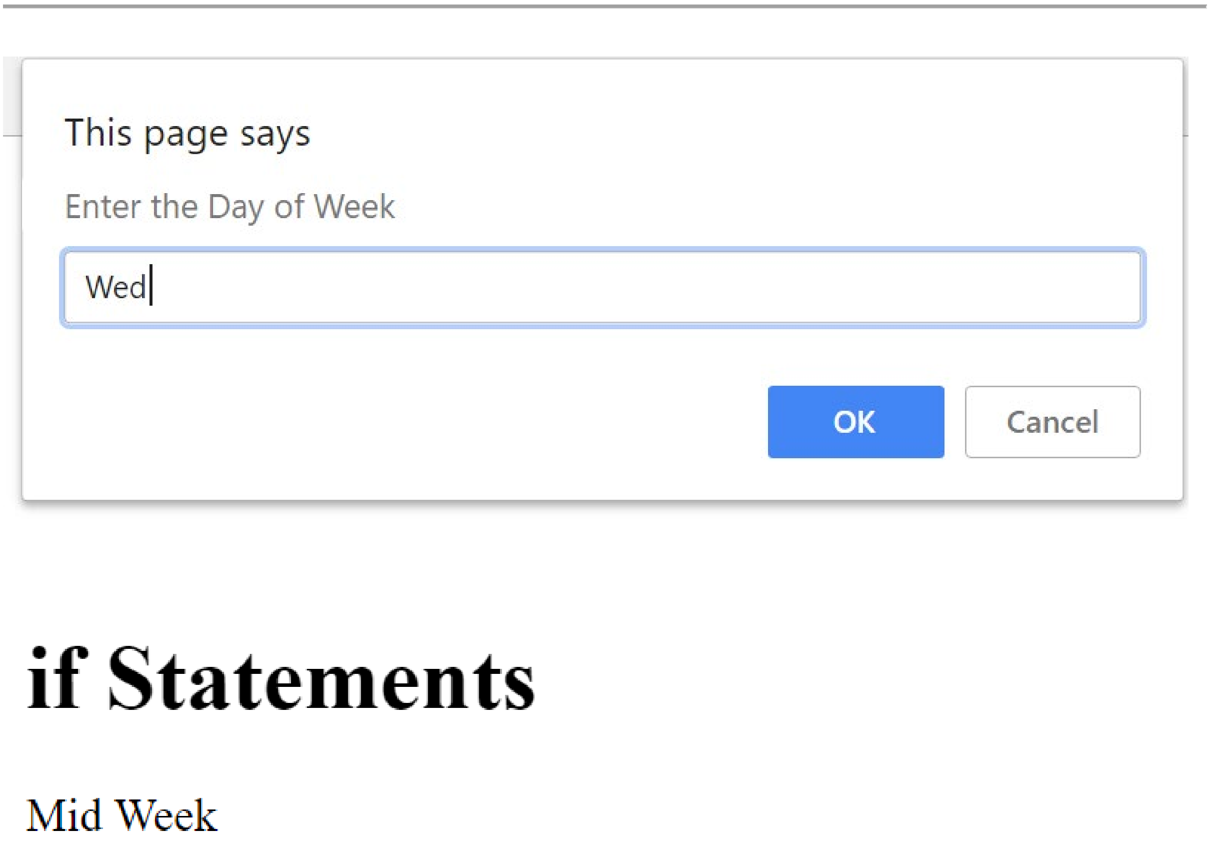
}else{

alert(“Try Again!”);

}

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>if Statements</title>
    <script>

        var dayOfWeek = prompt("Enter the Day of Week");
        var result;
        dayOfWeek = dayOfWeek.toLowerCase();

        if( dayOfWeek == 'monday' || dayOfWeek == 'mon'){
            result = "First Day of Week";
        }else if( dayOfWeek == 'tuesday' || dayOfWeek == 'tue' ){
            result = "Second Days of Week";
        }else if( dayOfWeek == 'wednesday' || dayOfWeek == 'wed' ){
            result = "Mid Week";
        }else if( dayOfWeek == 'thursday' || dayOfWeek == 'thurs'){
            result = "Preparing for Weekend";
        }else if( dayOfWeek == 'friday' || dayOfWeek == 'fri'){
            result = "It's Friday!";
        }else if( dayOfWeek == 'saturday' || dayOfWeek == 'sat' ){
            result = "Enjoying Day!";
        }else if( dayOfWeek == 'sunday' || dayOfWeek == 'sun'){
            result = "Resting Day!";
        }else{
            result = "Cannot find that Value!";
        }

    </script>

</head>

<body>

    <h1>if Statements</h1>
    <script type="text/javascript">
        document.write( result );
    </script>

</body>
</html>
```



This page says

Enter the Day of Week

Wed

if Statements

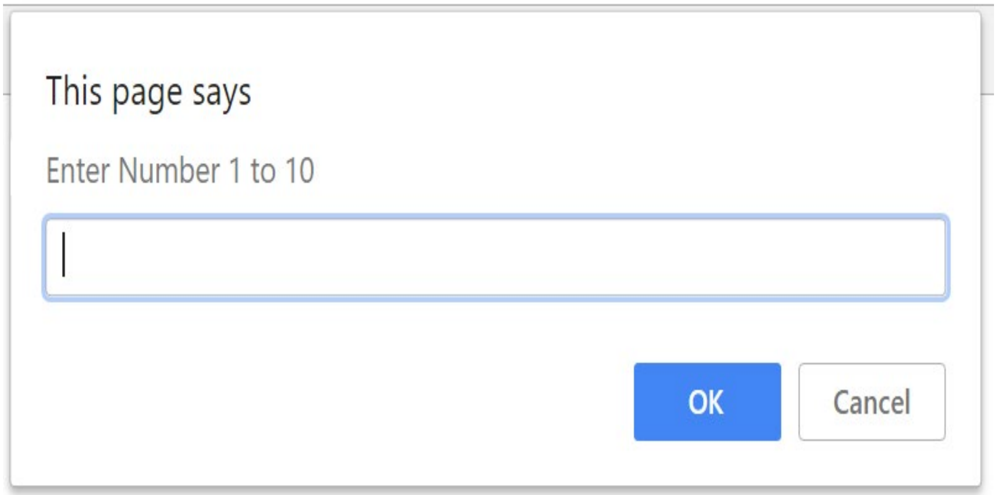
Mid Week

Live Preview

Exercise 1

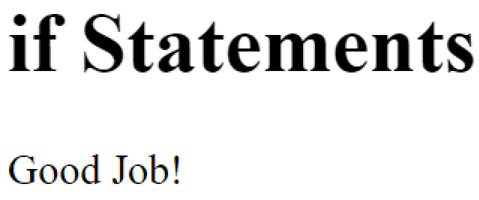
Download the Exercise 1

Exercise 1: Accept number 1 to 10 and check if the user entered correctly between 1 to 10 as requested with the if Statement.



This page says

Enter Number 1 to 10



if Statements

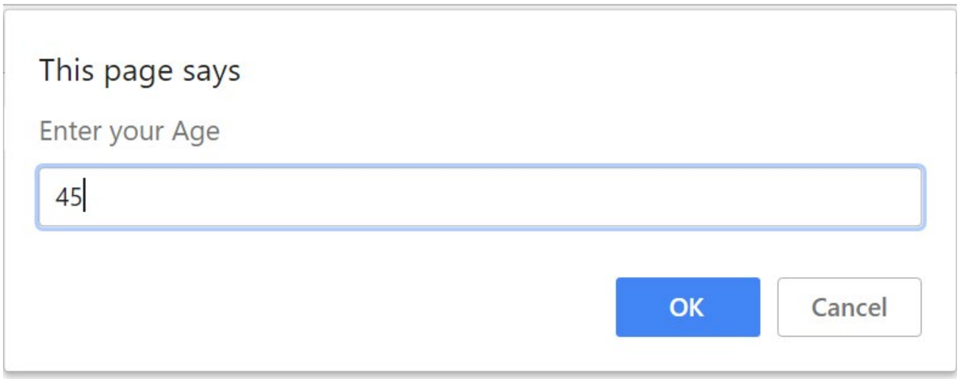
Good Job!

Live Preview

Exercise 2

Download the Exercise 2

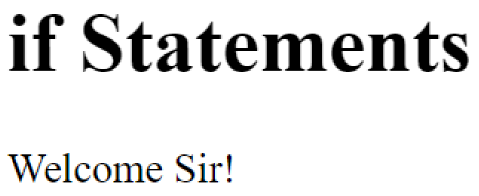
Exercise 2: Ask user to enter the age and decides if he is kid, man (age > 21) or senior citizen (age > 55).



This page says

Enter your Age

45



if Statements

Welcome Sir!

Live Preview

4.2 switch Statements

Usage of Switch Statements

switch statement are used to check a value and make a decision based on the result of the value matching.

Syntax:

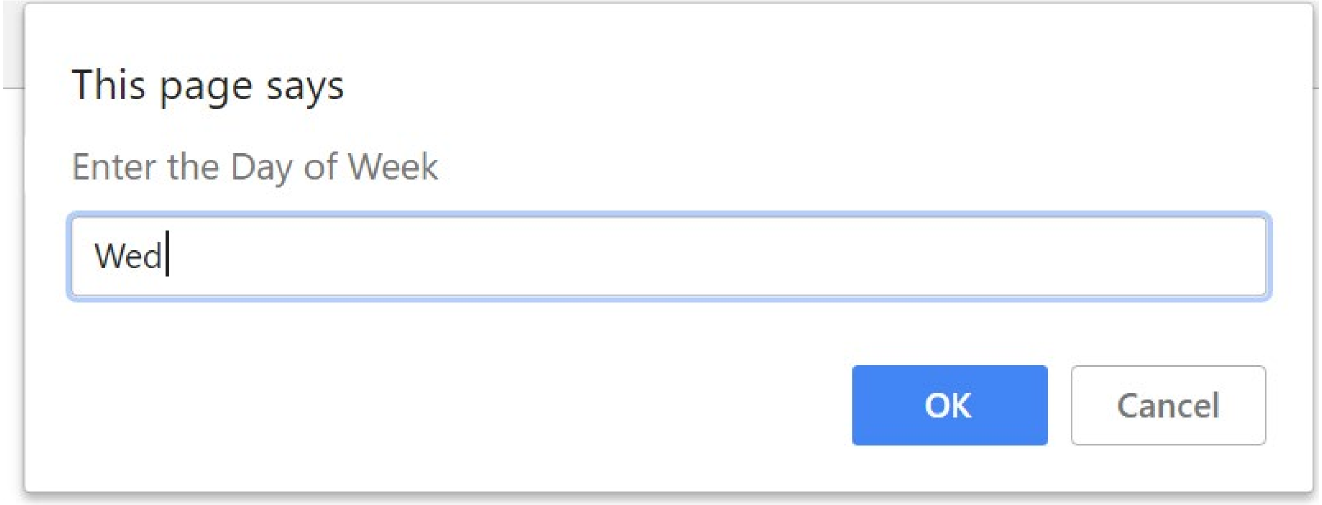
```
switch( variable ){

case value:

//Statement

break;
case value:
//Statement
break;
default:
//Statement
break;
}
```

Example 1:

```
switch ( dayOfWeek ){
case ‘Mon’:
alert(“Welcome Monday”);
break;
case ‘Tuesday’:
alert(“Welcome Tuesday”);
break;
default:
alert(“Try Again!”);
break;
}
```

Download the Example

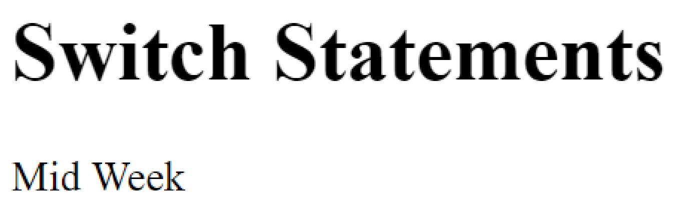
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Switch Statements</title>
    <script>

        var dayOfWeek = prompt("Enter the Day of Week");
        var result;
        dayOfWeek = dayOfWeek.toLowerCase();

        switch(dayOfWeek){
            case 'monday':
                result = "First Day of Week";
                break;
            case 'tuesday':
                result = "Second Days of Week";
                break;
            case 'wednesday':
                result = "Mid Week";
                break;
            case 'thursday':
                result = "Preparing for Weekend";
                break;
            case 'friday':
                result = "It's Friday!";
                break;
            case 'saturday':
                result = "Enjoying Day!";
                break;
            case 'sunday':
                result = "Resting Day!";
                break;
            default:
                result = "Cannot find that Value!";
        }

    </script>

</head>

<body>

    <h1>Switch Statements</h1>
    <script type="text/javascript">
        document.write( result );
    </script>

</body>

</html>
```



This page says

Enter the Day of Week

Wed



Switch Statements

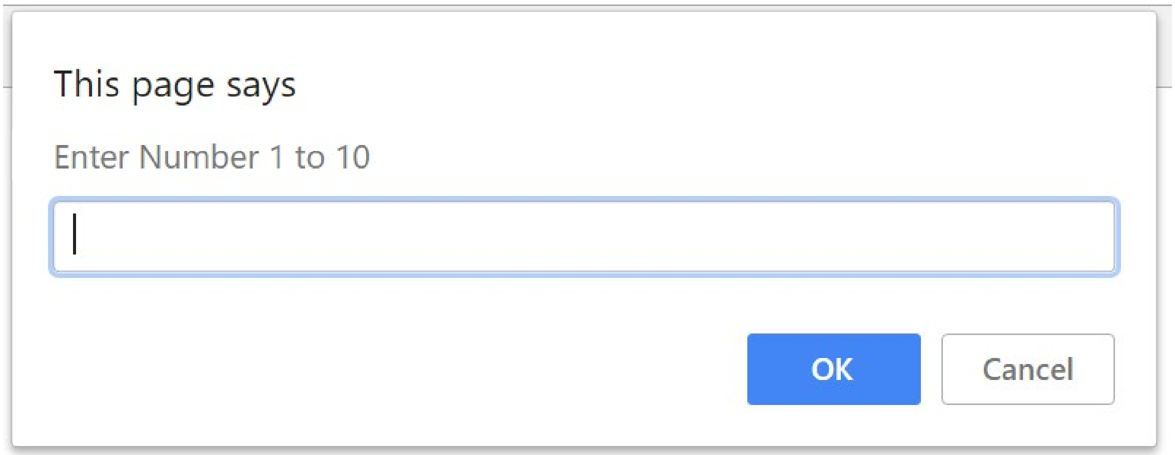
Mid Week

Live Preview

Exercise 1

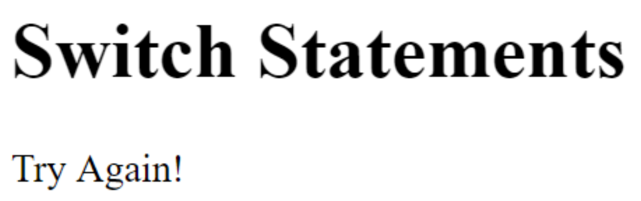
Download the Exercise 1

Exercise 1: Accept number 1 to 10 and check if the user entered correctly between 1 to 10 as requested with the Switch Statement.



This page says

Enter Number 1 to 10



Switch Statements

Try Again!

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Ask user to enter the age and decides if he is kid, man (age > 21) or senior citizen (age > 55). Use the Switch Statement.

Tips:

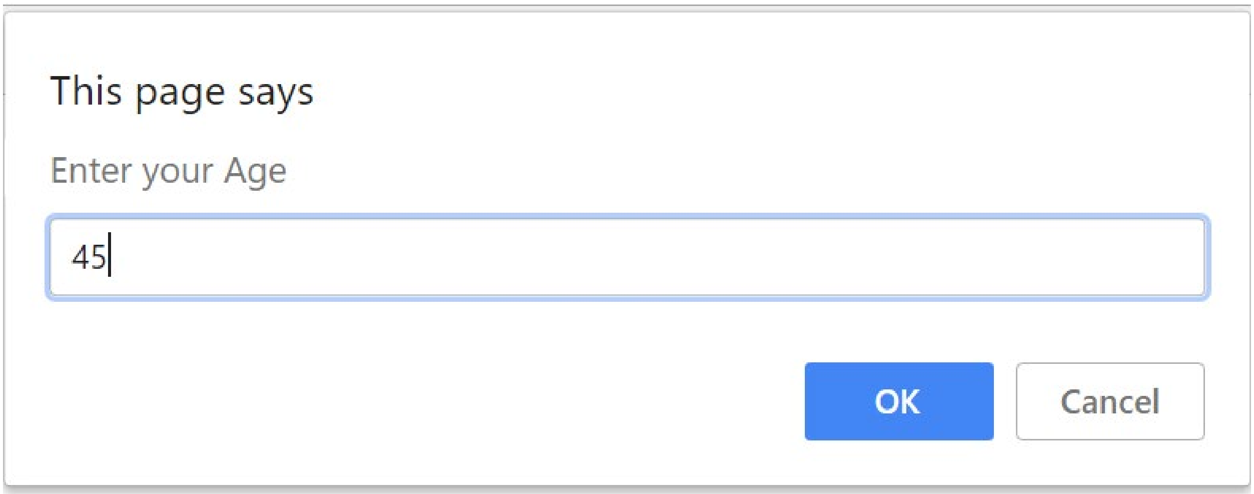
switch(true){

case (input > 21):

//Statements

break

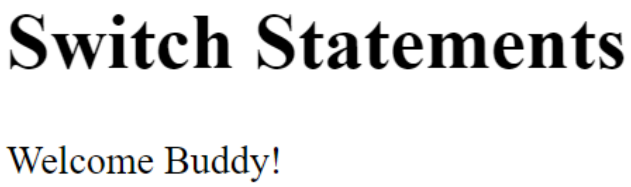
}



This page says

Enter your Age

45



Switch Statements

Welcome Buddy!

Live Preview

4.3 while Statements

Usage of While Loop

While statement are used to loop a block code and run it until a condition is met.

Running the same block of code until the condition is satisfied.

Syntax:

```
while( condition ) {

//Statements

}
```

Example 1:

```
var counter = 0;

while ( counter < = 10 ){

counter++;

}
```

Syntax:

```
do{
```

//Statement

}while (condition);

Example 1:

var counter = 0;

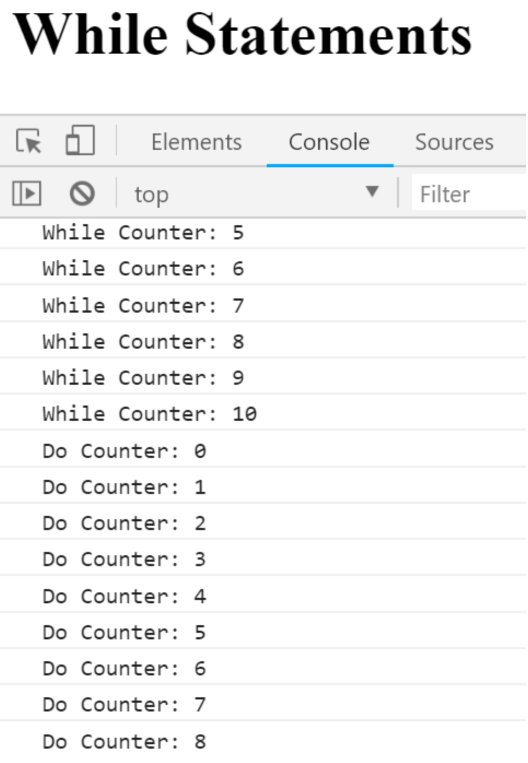
do{

counter++;

} while (counter < = 10);

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>While Statements</title>
    <script>

        var counter = 0;
        while( counter <= 10){
            console.log("While Counter: " + counter++);
        }
        counter = 0;
        do{
            console.log("Do Counter: " + counter++);
        }while( counter <= 10);

    </script>

</head>

<body>

    <h1>While Statements</h1>

</body>

</html>
```



While Statements

```
While Counter: 5
While Counter: 6
While Counter: 7
While Counter: 8
While Counter: 9
While Counter: 10
Do Counter: 0
Do Counter: 1
Do Counter: 2
Do Counter: 3
Do Counter: 4
Do Counter: 5
Do Counter: 6
Do Counter: 7
Do Counter: 8
```

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Accept two number from user and print all the numbers between them. Print only maximum of 10 numbers.

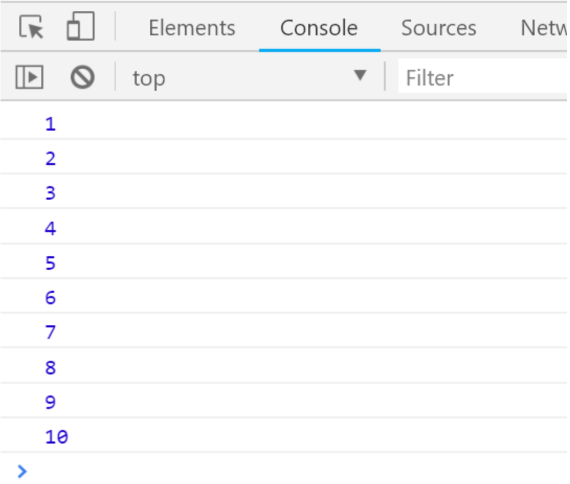
break; – is a keyword that can be used in the loop to break from the loop.

continue; – is a keyword that can be used in the loop to continue the loop by skipping the below statements.

Tip:

if(counter >= 10){
break;
}

While Statements



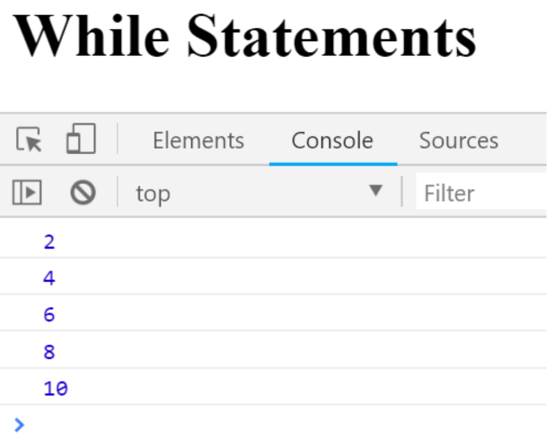
```
1
2
3
4
5
6
7
8
9
10
>
```

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Accept two number from user and print **EVEN** numbers only between them. Print only maximum of 10 numbers.



While Statements

```
2
4
6
8
10
>
```

Live Preview

4.4 for Statements

Usage of For Loop

for Loop are used to loop a block code and run it until a condition is met.

Running the same block of code until the condition is satisfied.

Syntax:

for(counter initialization; condition; increments) {

//Statements

}

Example 1:

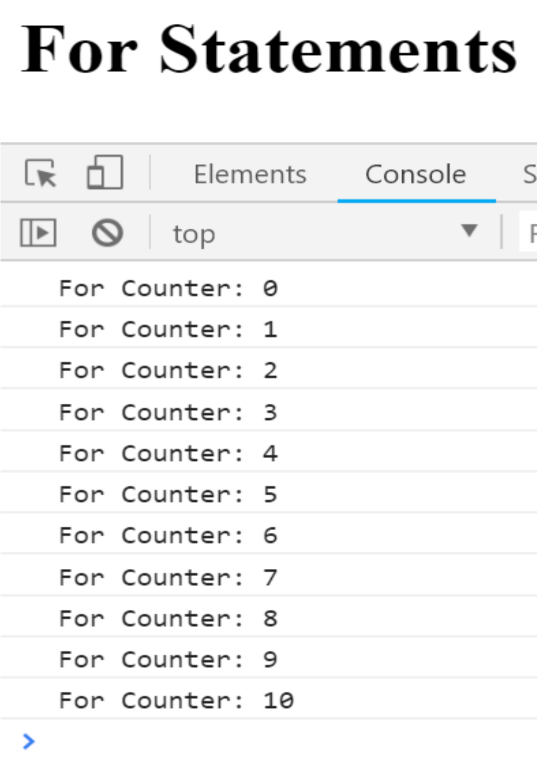
for(var counter = 0; counter <= 10; counter++){

alert(counter);

}

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>For Statements</title>
    <script>

        for(var counter=0; counter<=10; counter++){
            console.log("For Counter: " + counter);
        }
    </script>

</head>

<body>

    <h1>For Statements</h1>

</body>

</html>
```



For Statements

```
For Counter: 0
For Counter: 1
For Counter: 2
For Counter: 3
For Counter: 4
For Counter: 5
For Counter: 6
For Counter: 7
For Counter: 8
For Counter: 9
For Counter: 10
>
```

Live Preview

Exercise 1

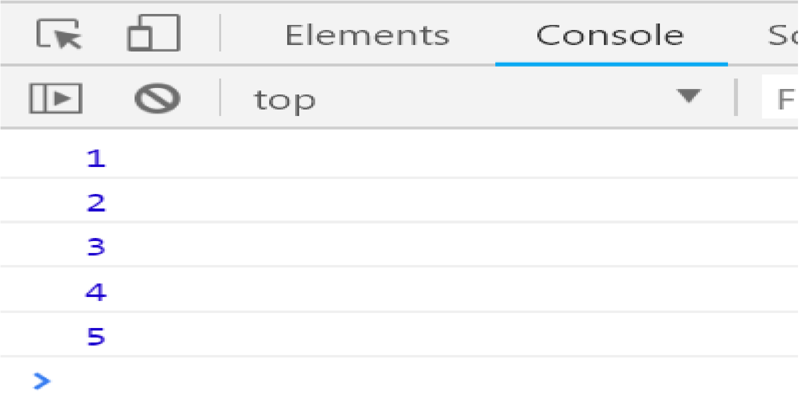
Download the Exercise 1

Exercise 1: Accept two number from user and print all the numbers between them. Print only maximum of 10 numbers.

break; – is a keyword that can be used in the loop to break from the loop.

continue; – is a keyword that can be used in the loop to continue the loop by skipping the below statements.

For Statements



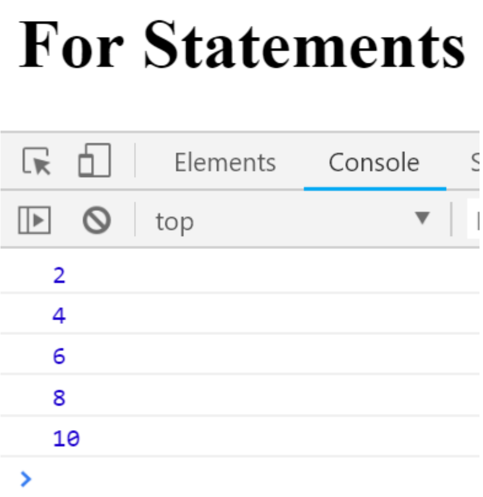
```
1
2
3
4
5
>
```

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Accept two number from user and print **EVEN** numbers only between them. Print only maximum of 10 numbers.



For Statements

```
2
4
6
8
10
>
```

Live Preview

5.Functions, Objects & Events

5 Function, Objects and Events

5.1 Functions

Usage of Functions

Functions is a block of statements that performs an action.

You can pass parameters to functions and it can return a value from the function using “**return**” keyword.

Syntax of Function Declaration:

```
function nameOfFunction(Parameters){

return someValue;

}
```

Example 1:

```
function area(width, height){

return width * height;

}

var size = area(10, 20);
```

Syntax of Function Expression:

```
var nameOfFunction = function(Parameters){

return someValue;

}
```

Example 1:

```
var area = function(width, height){
```

return width * height;

}

var size = area(10, 20);

Example of Immediately Invoked Functions:

var size = (function(){

var width = 10;

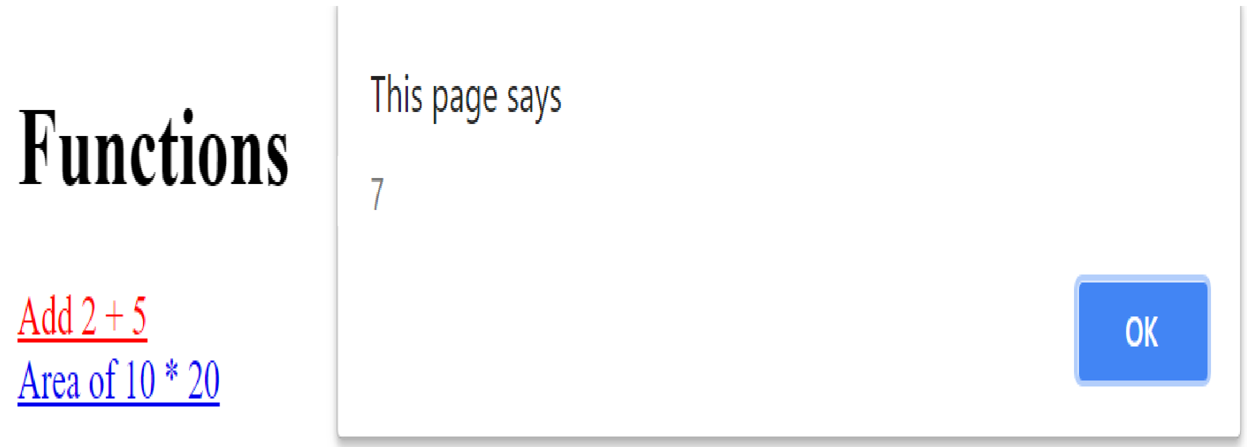
var height = 20;

return width * height;

}());

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Functions</title>
    <script>

        function sum(a, b){
            return a + b;
        }

        var size = function(width, height){
            return width * height;
        }

    </script>

</head>

<body>
<h1>Functions</h1>
<a href="#" onclick="alert( sum(2,5) );">Add 2 + 5</a>
    <br>
<a href="#" onclick="alert( size(10,20) );">Area of 10 * 20</a>

</body>

</html>
```



Functions

Add 2 + 5
Area of 10 * 20

This page says

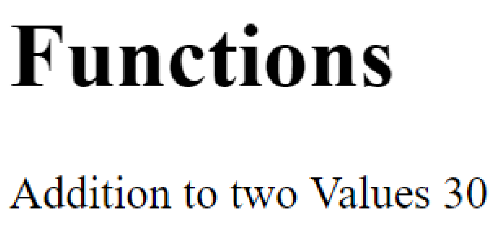
7

[Live Preview]()

Exercise 1

[Download the Exercise 1]()

Exercise 1: Accept two number from users and add those two numbers using functions.



Functions

Addition to two Values 30

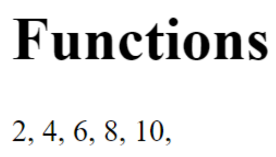
[Live Preview]()

Exercise 2

[Download the Exercise 2]()

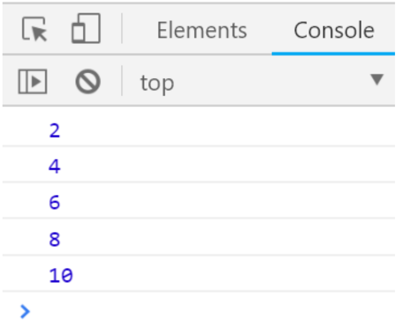
Exercise 2: Accept two number from user and print **EVEN** numbers only between them. Print only maximum of 10 numbers.

Create functions for this and execute it.



Functions

2, 4, 6, 8, 10,



```
2
4
6
8
10
>
```

Live Preview

5.2 Objects

Usage of Objects

Object groups variables and functions together.

Variables in objects are called as Properties.

function in Objects are called as Methods.

If Hotel is an Object then numberOfRoom is a Property and checkAvailability() is a function.

Object Literal:

var hotel = {

numberOfRooms: 10,

bookedRooms: 5,

```
checkAvailability(): function(){
return numberOfRooms – bookedRooms;
}
};
var hotelTotalRooms = hotel.numberOfRooms;
```

or

```
var hotelTotalRooms = hotel[‘numberOfRooms’];
var roomsFree = hotel.checkAvailability();
```

Creating an Object:

```
var hotel = new Object();
hotel.numberOfRooms = 10;
hotel.bookedRooms = 5;
hotel.checkAvailability = function(){
return this.numberOfRooms – this.bookedRooms;
}
```

Object Function Notation:

Creating Object as a Function

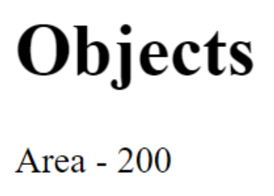
```
function hotel(totalRooms, roomLeft){
this.numberOfRooms = totalRooms;
this.bookedRooms = roomLeft;
this.checkAvailability = function(){
return this.numberOfRooms – this.bookedRooms;
}
```

}

var h1 = new hotel(10, 5);
var h2 = new hotel(10, 4);

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Objects</title>
    <script>

        function Area(a, b)
        {
            this.input1 = a;
            this.input2 = b;

            this.getArea = function(){
                return this.input1 * this.input2;
            }
        }

        var input1 = prompt("Enter First Value");
        var input2 = prompt("Enter Second Value");
        var result = 0;

        input1 = parseInt(input1);
        input2 = parseInt(input2);

        var a1 = new Area(input1, input2);
        result = a1.getArea();

        console.log("area - " + result);

    </script>

</head>

<body>

    <h1>Objects</h1>
    <script type="text/javascript">
        document.write("Area - " + result);
    </script>

</body>

</html>
```



Objects

Area - 200

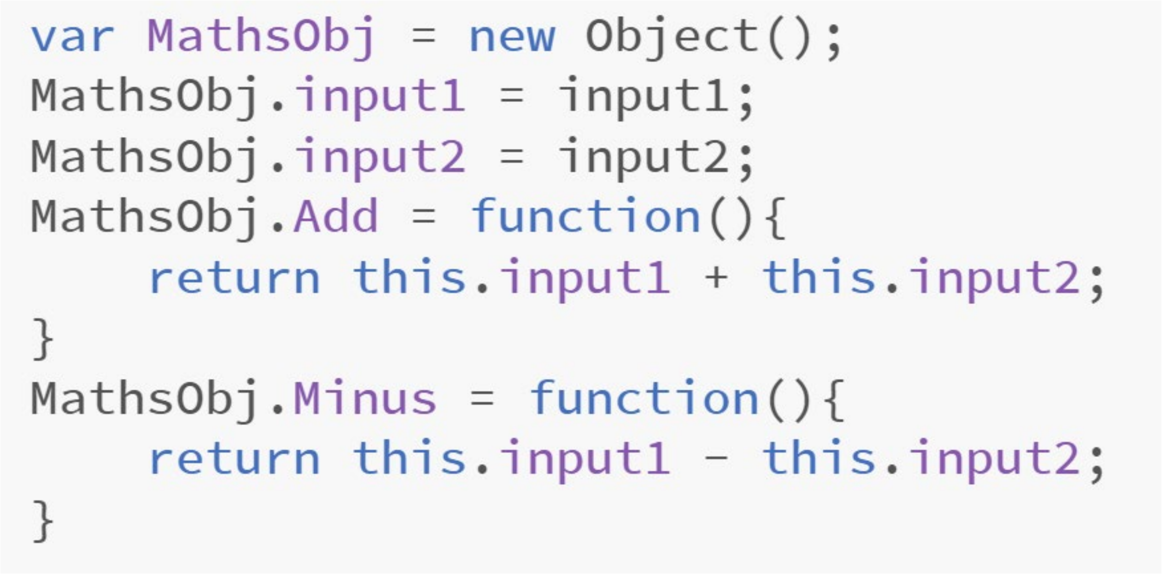
Live Preview

Exercise 1

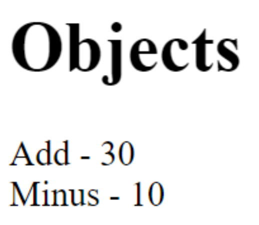
Download the Exercise 1

Exercise 1: Accept two number from users and add those two numbers using Objects.

Create a MathsObj Objects and Make Add and Subtract functions.



```
var MathsObj = new Object();
MathsObj.input1 = input1;
MathsObj.input2 = input2;
MathsObj.Add = function(){
    return this.input1 + this.input2;
}
MathsObj.Minus = function(){
    return this.input1 - this.input2;
}
```



```
Objects

Add - 30
Minus - 10
```

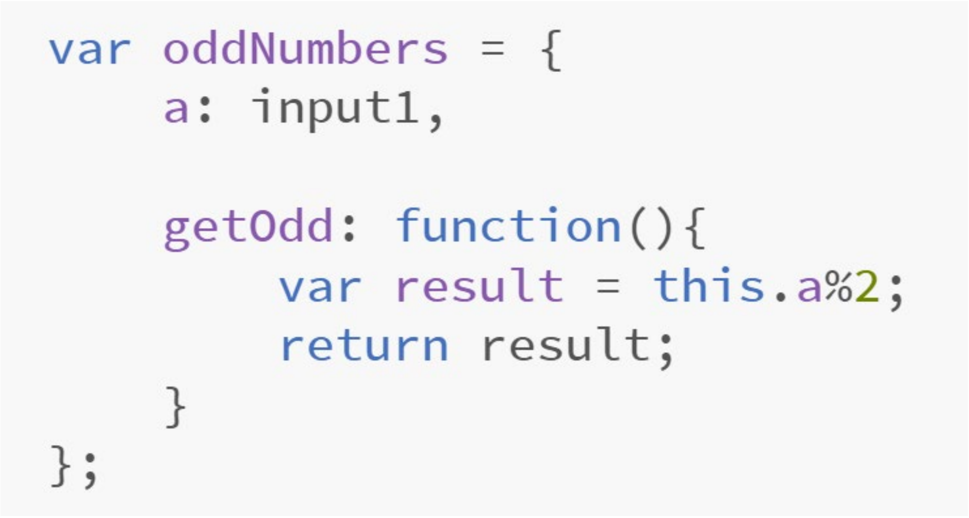
[Live Preview]

Exercise 2

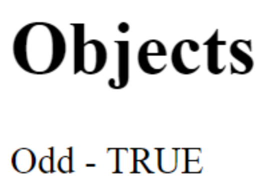
[Download the Exercise 2]

Exercise 2: Accept two number from user and print **EVEN** numbers only between them. Print only maximum of 10 numbers.

Create Object for this and use it.



```
var oddNumbers = {
    a: input1,

    getOdd: function(){
        var result = this.a%2;
        return result;
    }
};
```



```
Objects

Odd - TRUE
```

Live Preview

5.3 Events

Usage of Events

Events are actions that user performs on the browsers.

Functions that handle the events are called as Event Handlers.

Common Events with the elements:

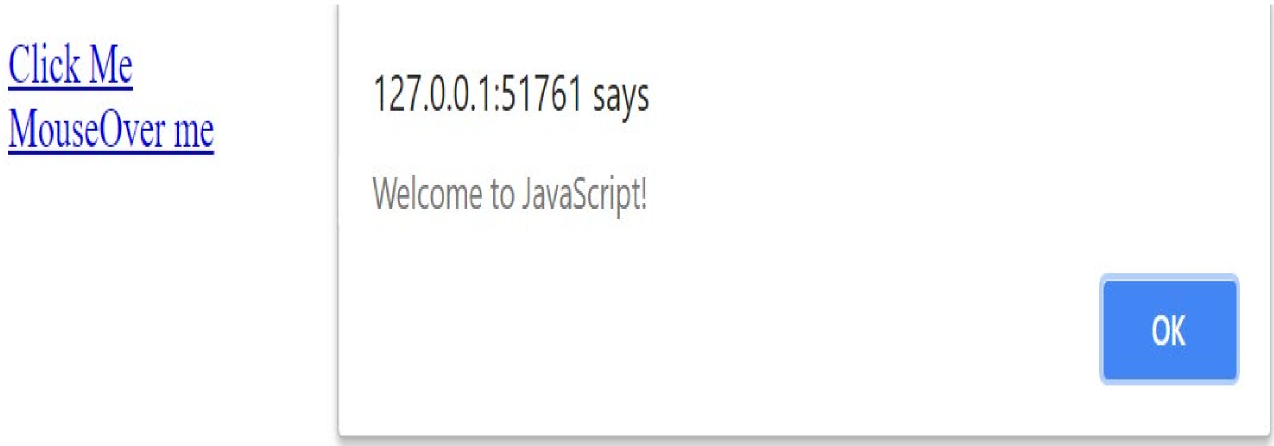
- onClick
- ondblClick
- onmouseover
- onmouseup
- onmouseout

Example:

<a href=”#” onmouseover=”alert(‘hello’);”>Mouse Over Me</a>

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Events JavaScript</title>
</head>

<body>
    <a href="#" onclick="alert('Welcome to JavaScript!');">Click Me</a>
    <br>
    <a href="#" onmouseover="alert('Welcome to JavaScript!');">MouseOver
me</a>
</body>

</html>
```



127.0.0.1:51761 says

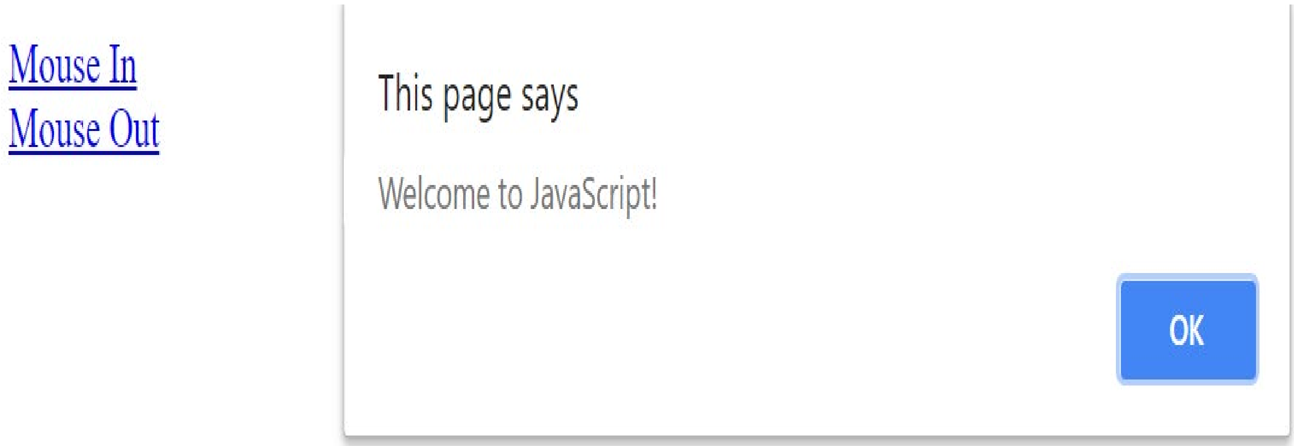
Welcome to JavaScript!

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Alert when user mouseover and mouseout events on link.



This page says

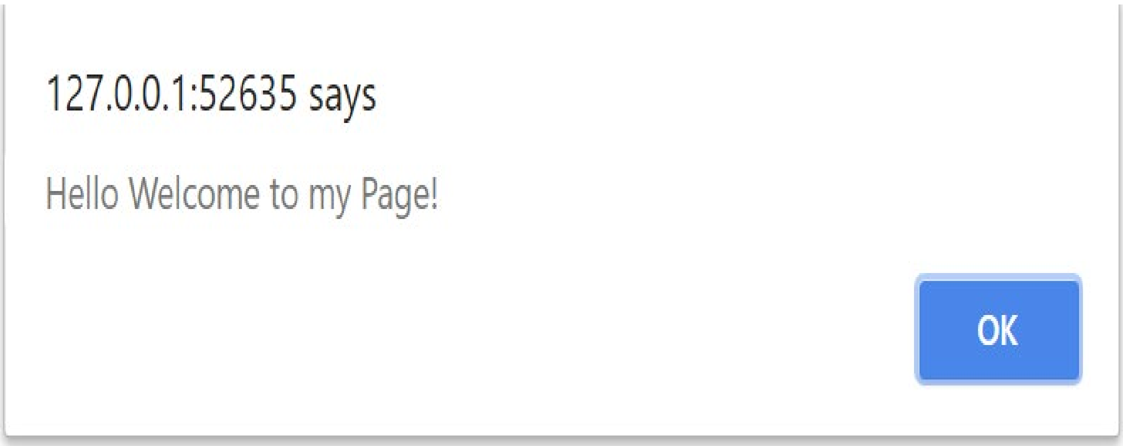
Welcome to JavaScript!

Live Preview

Exercise 2

Download the Exercise 2

Exercise 2: Use **onload()** event on the body to greet the user welcome message.


127.0.0.1:52635 says

Hello Welcome to my Page!

Live Preview

6.Testing

6 Testing

6.1 Debugging

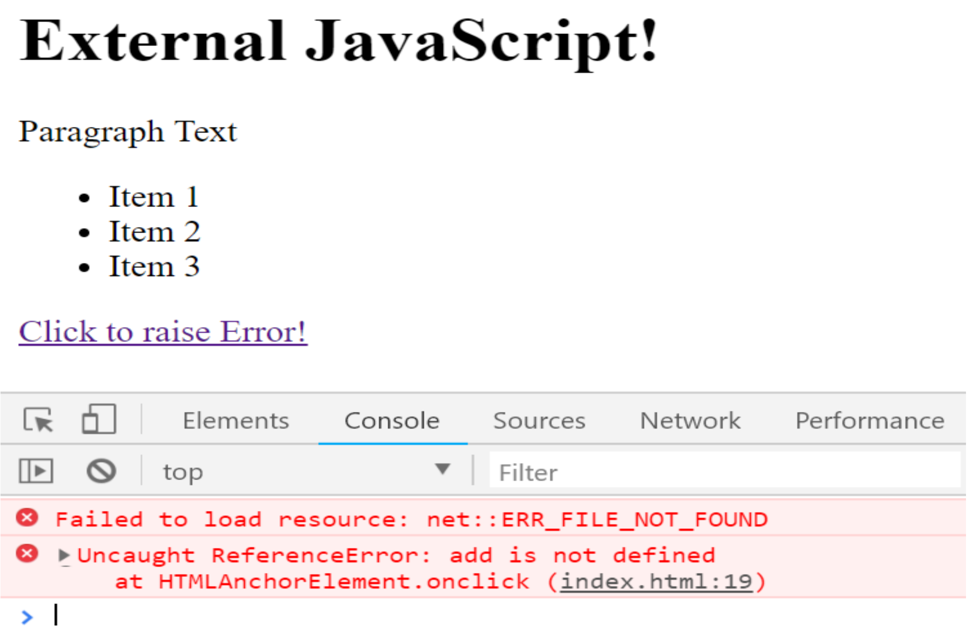
Step 1: Write JavaScript Code with Errors

Write a External JavaScript file “**scripts.js”** and include it in the HTML file.

Try the following Errors:

1. Give the wrong javascript file link name as **scripting.js** instead of **scripts.js**
   - Verify and Fix the program using Developer Tools shown below.

Download the Exercise 1



External JavaScript!

Paragraph Text

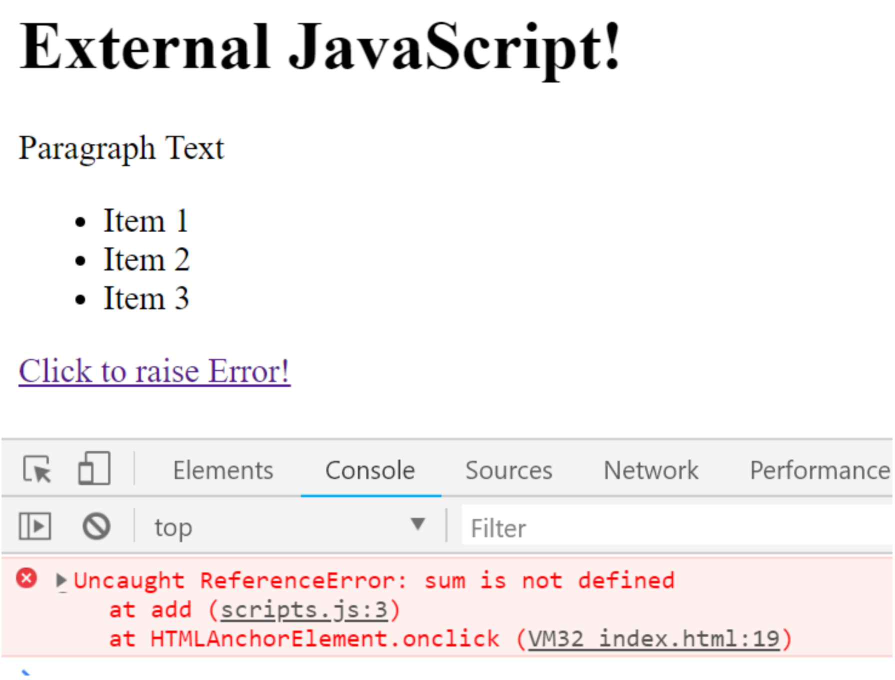
- Item 1
- Item 2
- Item 3

Click to raise Error!

```
Failed to load resource: net::ERR_FILE_NOT_FOUND
▶Uncaught ReferenceError: add is not defined
    at HTMLAnchorElement.onclick (index.html:19)
>
```

Live Preview 1

1. Add Sum() method in the Javascript file and do not create that function. This will cause the program to fail.



External JavaScript!

Paragraph Text

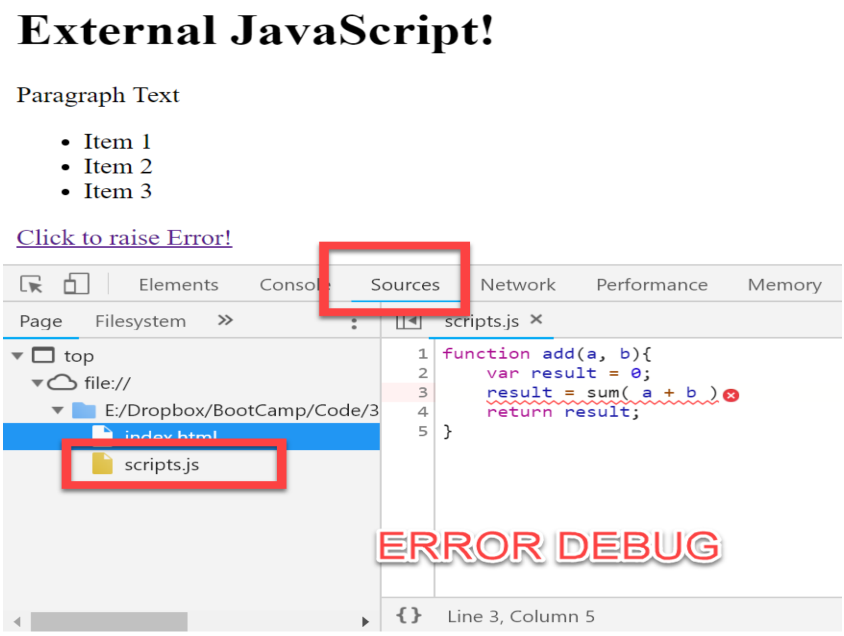
- Item 1
- Item 2
- Item 3

Click to raise Error!

```
Uncaught ReferenceError: sum is not defined
    at add (scripts.js:3)
    at HTMLAnchorElement.onclick (VM32 index.html:19)
```

Click on the file name – **scripts.js**

Download the Exercise 2



External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

Click to raise Error!

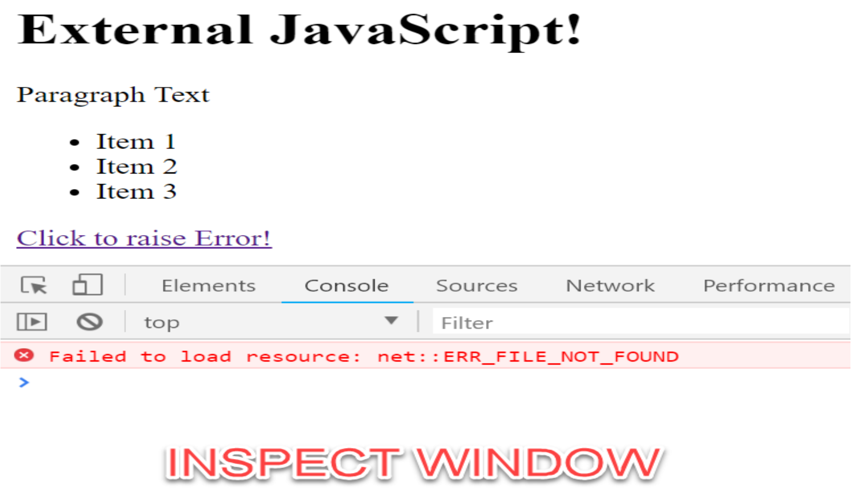
```
function add(a, b){
    var result = 0;
    result = sum( a + b )
    return result;
}
```

[Live Preview 2]

Step 2: Open HTML Page Using Chrome

Download the Source code and open the HTML file using the chrome browser and test it.

Step 3: Open Developer Tools



External JavaScript!

Paragraph Text

- Item 1
- Item 2
- Item 3

Click to raise Error!

```
Failed to load resource: net::ERR_FILE_NOT_FOUND
>
```

3 Ways to open the Chrome Developer Tool:

1. CTRL + SHIFT + I
2. F12
3. Top-Right Menu -> More Tools -> Developer Tools

Step 4: Debug

Two ways to find the error file:

1. Once you open the console window it show the errors with red color. Clicking on the link will take you to the javascript error code.
2. Click on Source Tab and open the file. It will show any errors in the file with red cross marks.

6.2 Common Errors

Common JavaScript Errors

These are the most common JavaScript errors that you might make while writing the JavaScript.

Forgetting the Semicolors ';' at the last.

JavaScript is Case Sensitive. So make sure the names matches as it is described.

Difference between Assignment and Comparison Operator.

Example:

```
//Using Assignment Operator instead of Comparison Operator.
if( a = b ){
// Statement
}
```

Comparing Number with String.

Forgetting the ‘**+**‘ sign in the string.

Example: “This is a sample ” + message + ” Program” + message1 ” to do it!”;

Forgetting the Single Quotes Opening and Closing.

Example: ‘This is a sample ‘ + message + ‘ Program’s ‘ + message1 + ‘ to do it!’;

Missing Parenthesis in If Statements

Example:

```
//Missing Parenthesis
if ( x > y) && (y > 100) {

}
```

Using Wrong Keywords Case.

Example:

```
Funtion add(){
}
```

Using Keywords as Variable name.

Example:

```
//Keywords as Variable
var name = “hello”;
```

Forget to close the parenthesis

Example:

```
function add(){
}

var total = add;
```

Accessing the Wrong Array Index.

Example:

```
var myArray = new Array();
myArray[0] = “hello”;
alert( myArray[1] );
```

6.3 Try Catch Block

Usage of Try Catch Blocks

Exceptions in programming are referred as Runtime Errors.

When Runtime Errors happens there is a way to catch those errors and pass it to the application and make a clean exit from the program.

This is called as Exception Handling.

The process of making sure the code will not break and if it does it know the reason for it and make a clean exit.

To handle Runtime errors in the JavaScript we have try-catch blocks. Using this we can catch the errors and decide what to do next.

SYNTAX:

```
try{

//Statements

}catch(errorName) {

//Statments

}
```

Example:

```
try{
```

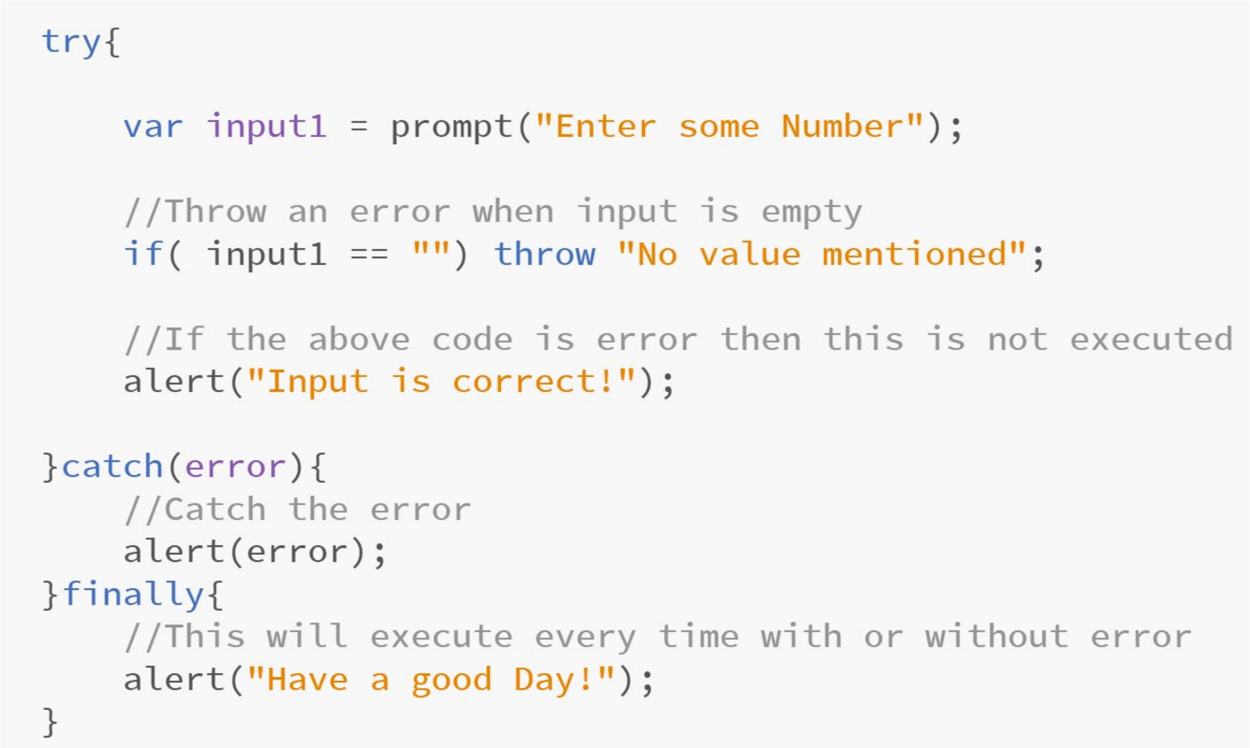
```
var firstName = “”;

if( firstName == “” ) throw “Name is empty”;

}catch(error){

alert(error);

}finally{

alert(“Thanks for playing!”);

}
```

try {} block will have all the statements

throw is a keyword to throw an error from the program. Once the program throw an error it will stop processing the next steps. It will jump to catch {} block.

catch {} block will be executed once the throw is called. The program will execute out from the catch block safely.

finally{} block will execute every time irrespective of error or not. It is good place to close all open connections. Handle clean exit.



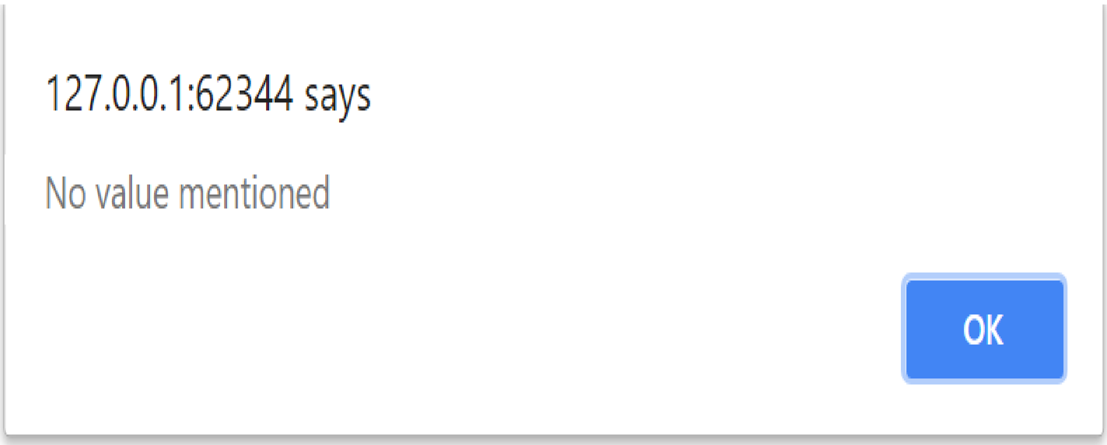
```
try{

    var input1 = prompt("Enter some Number");

    //Throw an error when input is empty
    if( input1 == "") throw "No value mentioned";

    //If the above code is error then this is not executed
    alert("Input is correct!");

}catch(error){
    //Catch the error
    alert(error);
}finally{
    //This will execute every time with or without error
    alert("Have a good Day!");
}
```

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Try Catch JavaScript</title>
    <script>

        try{

            var input1 = prompt("Enter some Number");

            //Throw an error when input is empty
            if( input1 == "") throw "No value mentioned";

            //If the above code is error then this is not executed
            alert("Input is correct!");

        }catch(error){
            //Catch the error
            alert(error);
        }finally{
            //This will execute every time with or without error
            alert("Have a good Day!");
        }

    </script>
```

Become PHP Full Stack Web Developer in Just 30 Days

```
</head>

<body>
<h1>Heading</h1>
<p>Paragraph Text</p>
<ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
</body>

</html>
```



127.0.0.1:62344 says

No value mentioned

[Live Preview]

Exercise 1

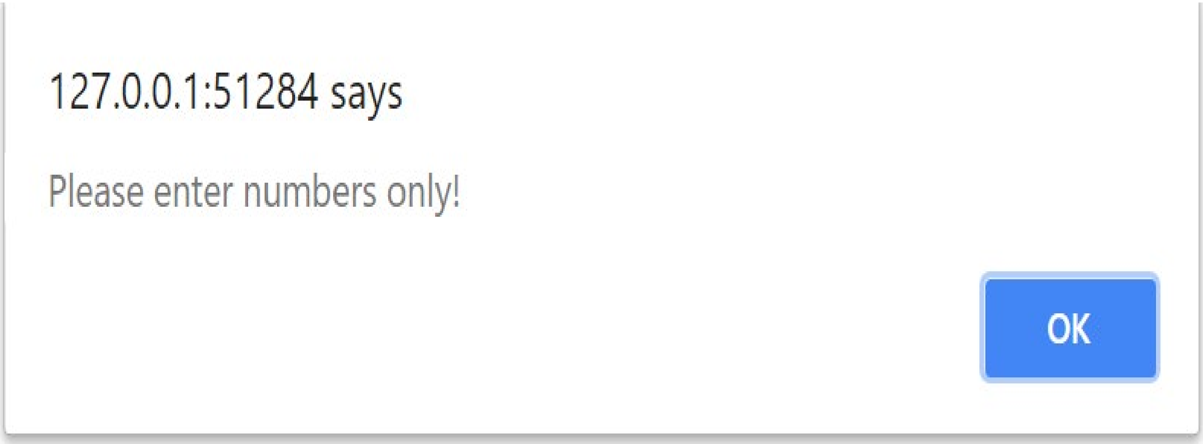
[Download the Exercise 1]

Exercise 1: Accept one number from user and raise error if the number is not integer.

Use Function: isNaN() to check if the value is number or not.

if(isNaN(input1)){

}



127.0.0.1:51284 says

Please enter numbers only!

[Live Preview]

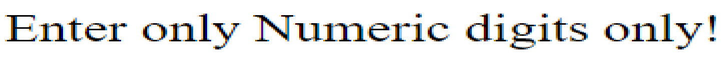
Exercise 2

[Download the Exercise 2]

Exercise 2: Ask user to enter number between 1 to 100.

Show the following message on the page:

- If the input is between 1 to 100 – Thank the user
- If the input is greater than 100 – Ask user to enter between 1 to 100
- If the input is not number – Ask user to enter numeric values only. throw the error message here.



Enter only Numeric digits only!

[Live Preview]

7.Document Object Model

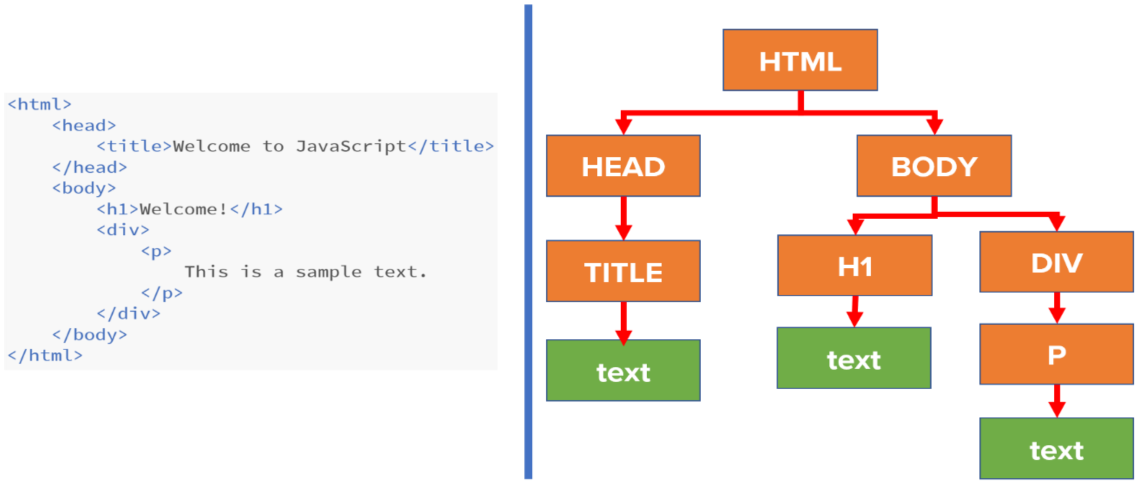
7 Document Object Model

7.1 Find an Element by ID

What is DOM?

Document Object Model or DOM in short is nothing but the hierarchy representation of all the HTML elements like a tree.

DOM is created by the browser for the HTML page so that it can easily navigate and make changes to the elements.



```
<html>
    <head>
        <title>Welcome to JavaScript</title>
    </head>
    <body>
        <h1>Welcome!</h1>
        <div>
            <p>
                This is a sample text.
            </p>
        </div>
    </body>
</html>
```

Document – Parent element of the DOM

Element – Each Orange box is an element

text – Each Green box is an text.

JavaScript can modify the DOM elements. By adding, removing and reading the elements from the DOM.

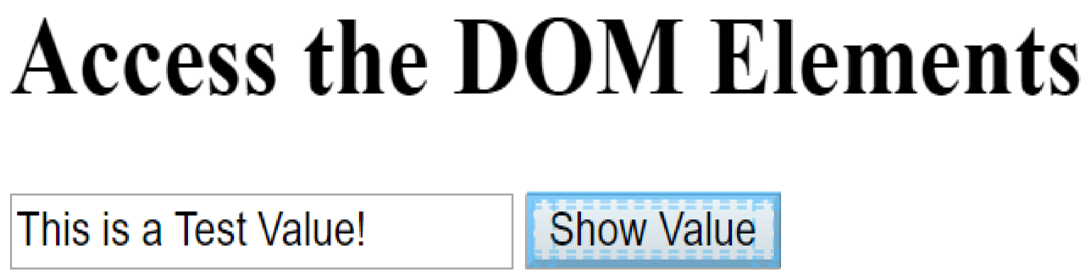
Read a Value of an Element

You can read any element from the DOM using JavaScript functions:

1. getElementById(idName)
2. getElementsByTagName(tagName)
3. getElementByName(name)
4. getElementByClassName(className)

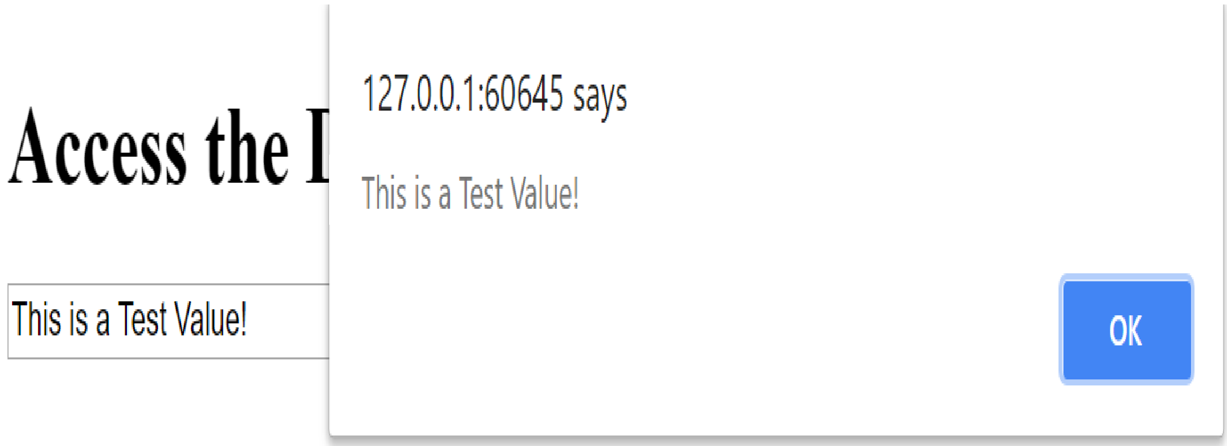
Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Inline JavaScript</title>
    <script>

        function show(){
            var txtValue = document.getElementById('txtName').value;
            alert( txtValue );
        }
    </script>
</head>

<body>
    <h1>Access the DOM Elements</h1>
    <form>

        <input type="text" id="txtName" />
        <input type="button" onclick="show();" Value="Show Value" />
    </form>
</body>

</html>
```



Access the DOM Elements

This is a Test Value!



Access the I

This is a Test Value!

127.0.0.1:60645 says

This is a Test Value!

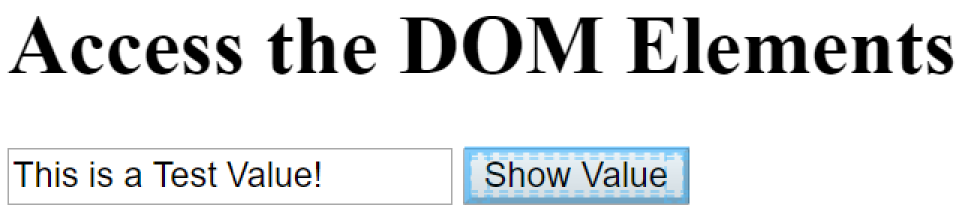
Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Copy the example and change the id=”txtName” to name=”txtName” and use this method to access the same value getElementByName.

Tips:
var txtValue = document.getElementsByName(‘txtName’)[0].value;



Access the DOM Elements

This is a Test Value!

Live Preview

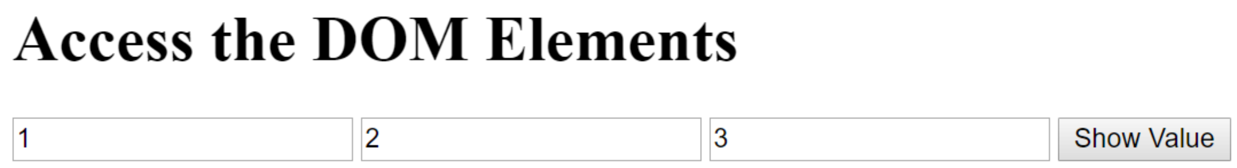
Exercise 2

Download the Exercise 2

Exercise 2: Access all the form text box elements using arrays and access them by name.

Tip:

<input type=”text” name=”txtName” />
<input type=”text” name=”txtName” />
<input type=”text” name=”txtName” />



Access the DOM Elements

[Live Preview]

7.2 Update the Data

Update the Element Data

innerHTML property can be used to update data for any element.

Adding data to this attribute will modify the DOM element and it will reflect immediately on the page.

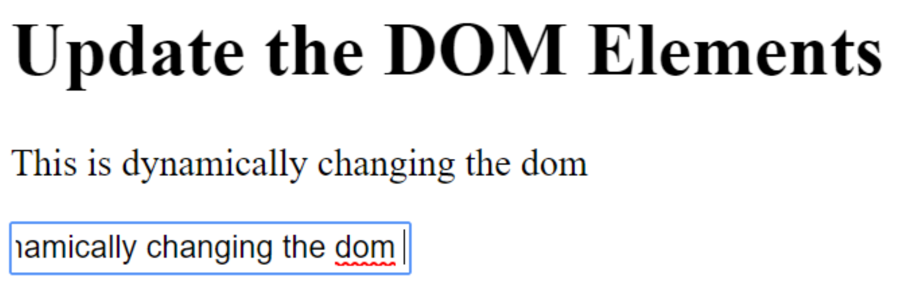
Example:

document.getElementById(id).innerHTML = “This is a added element”;

[Download the Example]

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Inline JavaScript</title>
    <script>

        function update(){

            document.getElementById("txtValue").innerHTML =
document.getElementById("txtName").value;
```

```
        }
    </script>
</head>

<body>
    <h1>Update the DOM Elements</h1>
    <p id="txtValue"></p>
    <form>
        <input type="text" id="txtName" onkeypress="update();"/>

    </form>
</body>

</html>
```



Update the DOM Elements

This is dynamically changing the dom

ıamically changing the dom

[Live Preview]

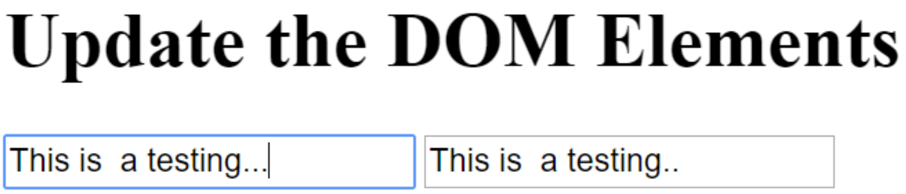
Exercise 1

[Download the Exercise 1]

Exercise 1: Create two Text box and if user writes the data in one text box then display the text in other text box.

Tip:

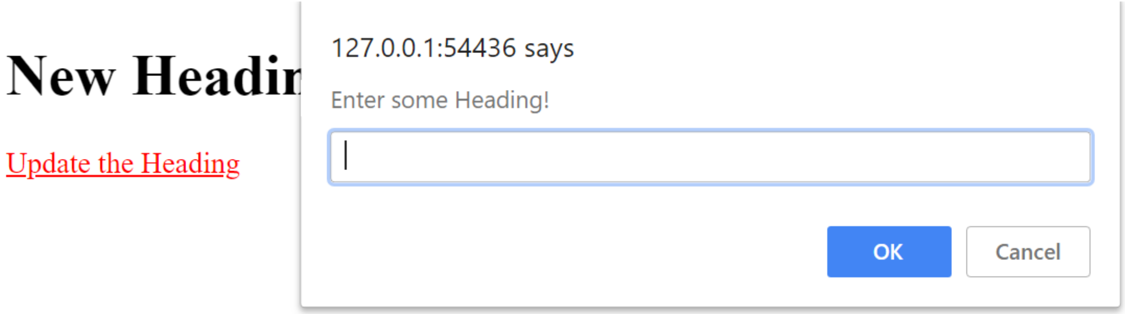
document.getElementsByName(“txtName”)[1].value = document.getElementsByName(“txtName”)[0].value;


Update the DOM Elements

Exercise 2

[Download the Exercise 2]

Exercise 2: Show a button and when user press ask them to type heading then change the heading with the new value.


New Headin

127.0.0.1:54436 says

Enter some Heading!

Live Preview

7.3 Access Form Elements

Access the Form Elements

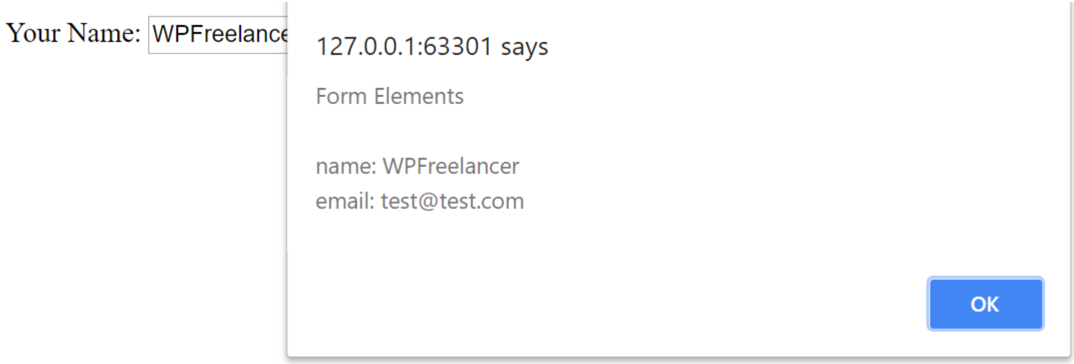
You can access all the form elements using the DOM.

This is very helpful when you want to check what user has entered the data before sending the data to server to save it.

It is called as Client Side Validation.

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Inline JavaScript</title>
    <script>

        function checkFormData(){

            //Query all elements that tag name as input
            var inputs = document.getElementsByTagName("input");

            var message = "Form Elements\n\n";

            for (var i=0; i < inputs.length; i++)
            {

               if (inputs[i].getAttribute('type') == 'text')
               {
                  message += inputs[i].getAttribute('name') + ": ";
                  message += inputs[i].value + "\n";
               }
            }
            alert(message);
        }
    </script>
</head>

<body>
<form name="user" id="userfrm" action="#">

Your Name: <input type="text" name="name" id="txt_name" />

Your Email: <input type="text" name="email" id="txt_email" />

<input type="button" name="submit" value="Submit"
onclick="checkFormData();" />

</form>

</body>

</html>
```



Your Name: WPFreelance

127.0.0.1:63301 says

Form Elements

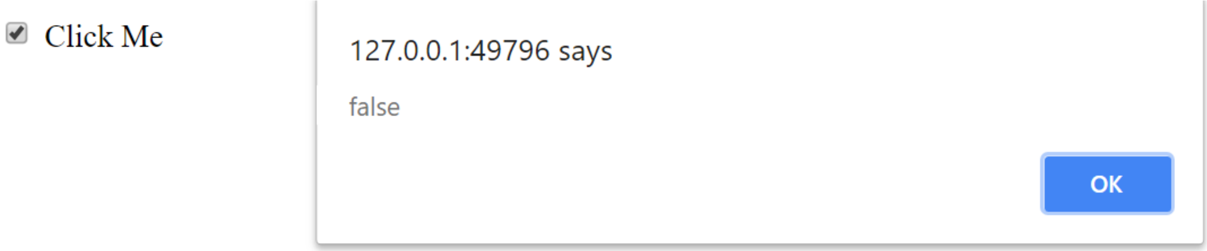
name: WPFreelancer
email: test@test.com

Live Preview

Exercise 1

Download the Exercise 1

Exercise 1: Read the Checkbox checked on the form using JavaScript.



☑ Click Me

127.0.0.1:49796 says

false

Tip:

<input type=”checkbox” id=”chkbox” onclick=”showFormData()”>

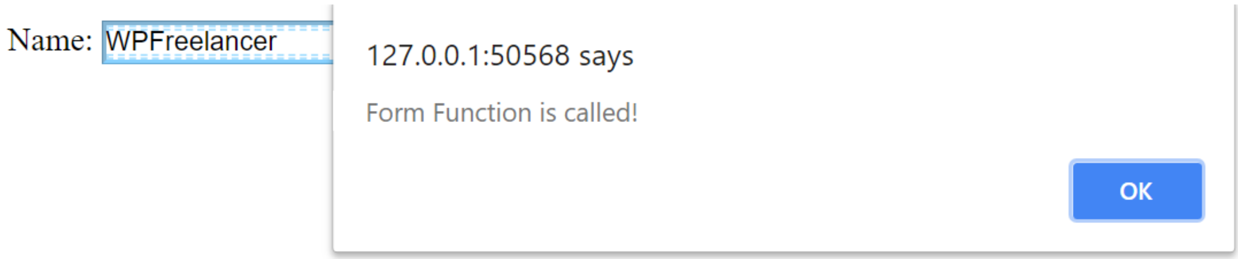
document.getElementById(“chkbox”).checked

Exercise 2

Download the Exercise 2

Exercise 2: Call JavaScript function when the form is submitted.

TIP:
<form name=”search” onsubmit=”return callFunction()” >



Name: WPFreelancer

127.0.0.1:50568 says

Form Function is called!

[Live Preview]

8.Snippets

8 Snippets JavaScript

8.1 Access the Browser URL

Usage Browser History Snippet

This snippet of code will help you to navigate the page front or back based on the page history.

You can make the page go back and front from the JavaScript.

Download the Snippet

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Snippets JavaScript</title>
    <script>

        function goBack(){
            history.back();
        }
    </script>
</head>

<body>

<a href="#" onclick="goBack();">Go Back</a> <br>
<a href="#" onclick="history.forward();">Go Forward</a>
</body>

</html>
```

Live Preview

9.Projects

9 Projects JavaScript

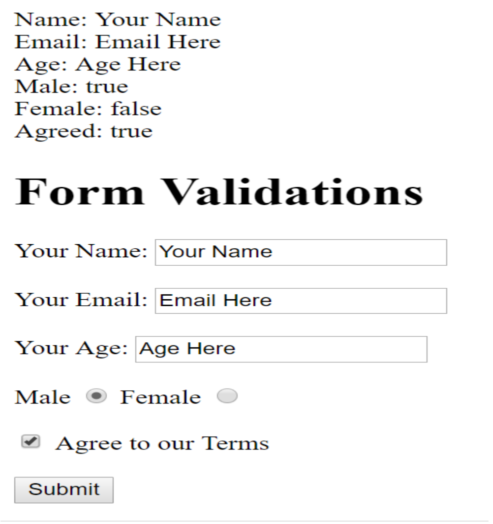
9.1 Form Validations

Form Validation

This project will show how to validate the form using the JavaScript.

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Form Validations JavaScript</title>
    <script>

        function checkFormData(){
            var message = "";

            //Access the TextBox
            var tname = document.getElementById("txt_name").value;
            var temail = document.getElementById("txt_email").value;
            var tage = document.getElementById("txt_age").value;

            //Access the Radio Button
            var tgender1 = document.getElementById("radio1").checked;
            var tgender2 = document.getElementById("radio2").checked;

            //Access the Checkbox
            var tchkbox = document.getElementById("chkbox").checked;

            message = "Name: " + tname + "<br>";
            message += "Email: " + temail + "<br>";
            message += "Age: " + tage + "<br>";
            message += "Male: " + tgender1 + "<br>";
            message += "Female: " + tgender2 + "<br>";
            message += "Agreed: " + tchkbox + "<br>";

            document.getElementById("tmessage").innerHTML = message;
        }
    </script>
</head>

<body>
    <p id="tmessage"></p>
    <h1>Form Validations</h1>
<form action="https://wpfreelancer.com/" method="get">

Your Name: <input type="text" name="tname" id="txt_name" /> <br><br>
Your Email: <input type="text" name="temail" id="txt_email" /> <br><br>
Your Age: <input type="text" name="tage" id="txt_age" /> <br><br>

Male <input id ="radio1" name="gender" type="radio" checked>
Female <input id ="radio2" name="gender" type="radio"> <br><br>

<input type="checkbox" id="chkbox" /> Agree to our Terms <br> <br>

<input type="button" name="submit" value="Submit"
onclick="checkFormData();" />

</form>

</body>

</html>
```



Name: Your Name
Email: Email Here
Age: Age Here
Male: true
Female: false
Agreed: true

Form Validations

Your Name: Your Name

Your Email: Email Here

Your Age: Age Here

Male ● Female ○

☑ Agree to our Terms

Live Preview

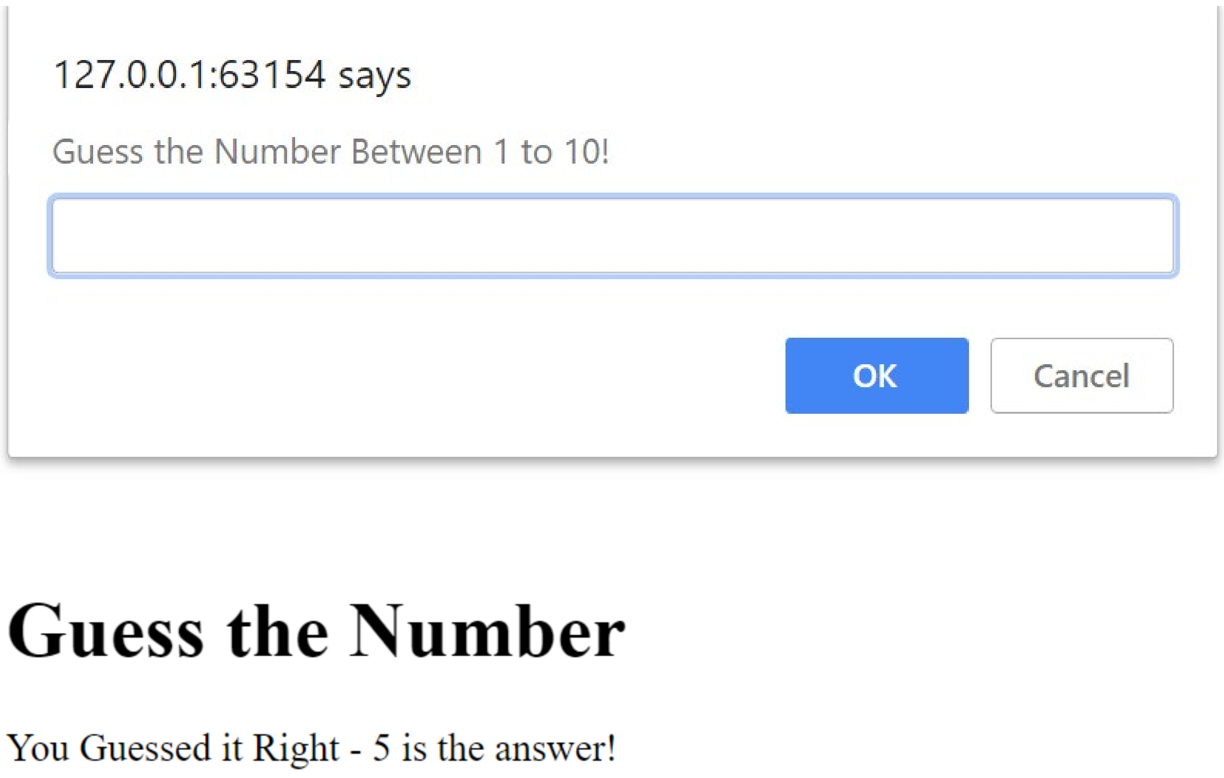
9.2 Guess the Number

Guess the Number

This project will ask user to guess the right number between 1 to 10. It will keep asking the number until the user guess it.

Download the Example

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Page Description">
    <title>Guess The Number</title>
    <script>

        var message = "";
        var answer = 5;




        do{
            var input1 = prompt("Guess the Number Between 1 to 10!");
            input1 = parseInt(input1);

            if( input1 != answer ){
                alert("Nope - Wrong.");
            }

        }while(answer != input1)

        message =  "You Guessed it Right - " + answer + " is the answer!";

    </script>

</head>

<body>

    <h1>Guess the Number</h1>
    <script type="text/javascript">
        document.write(message);
    </script>

</body>

</html>
```



127.0.0.1:63154 says

Guess the Number Between 1 to 10!

Guess the Number

You Guessed it Right - 5 is the answer!

Live Preview