CodeWithHarry

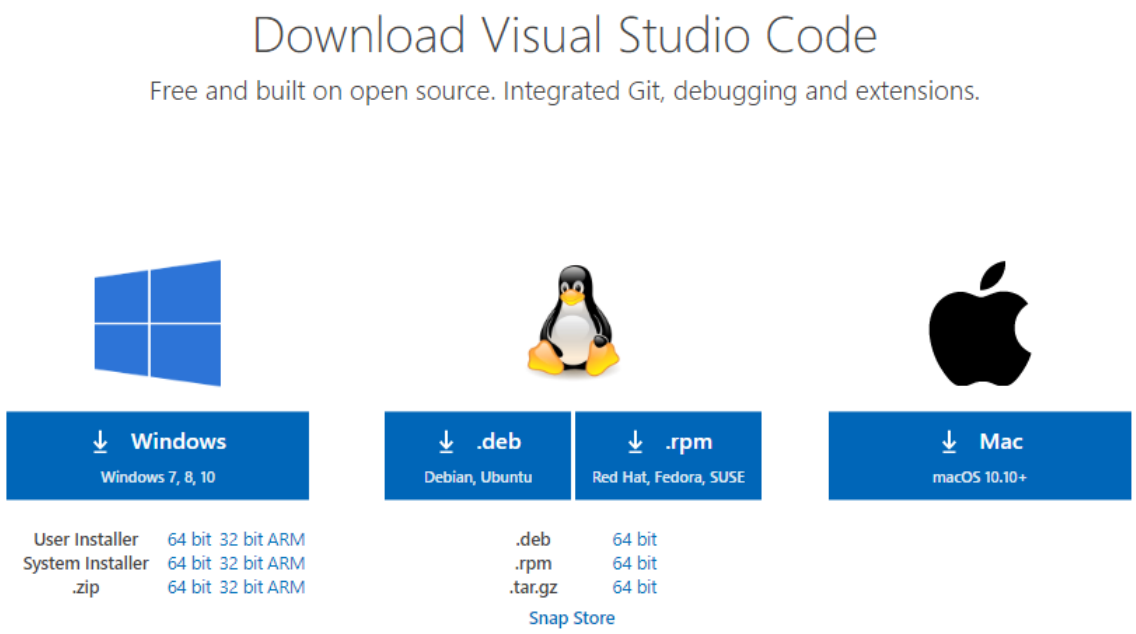
HTML CSS JS C C++ JAVA PYTHON PHP

Functions in JavaScript | JavaScript Tutorial In Hindi #10

Overview Q&A Downloads Announcements

Functions in JavaScript | JavaScript Tutorial In Hindi #10

For this series, we are using Virtual Studio Code (VS Code). ***Visual Studio Code*** is a fast source code editor and provides the tools that a developer needs for a quick code-build-debug cycle. To download VS Code, click on ***Download Virtual Studio Code***. For guidance, check the tutorial ***Installing VS Code, Extensions & Setup***



In today's tutorial, we will study about ***functions in JavaScript***. As we know, function is a group of reusable code which can be called anywhere in the program. This eliminates the need to rewrite the same code. Functions allow a programmer to divide a big program into a smaller and manageable function. JavaScript provides functions just like other programming languages. A JavaScript function can be defined using **the function** keyword.

There are mainly two advantages of JavaScript functions.

1. **Code reusability**: We can call a function several times whenever we need it.
2. **Less coding**: It reduces the line of code and makes our program compact.

Define a Function:-

There are two different ways to define a function in JavaScript:

Function Declaration:-

A **Function Declaration** defines a named function. To create a function declaration, use the function keyword, and then write the function's name. When using function declarations, the function definition is hoisted. Thus the function is allowed to be used before it is defined. The syntax of the function declaration is as follows:

```
function name(parameters){
   //statements
 }
```

Inside the function body, we can implement the logic. **For example**, the following print_message () function simply shows a message to the console:

```
function print_message(message) {
console.log(message);
}
print_message("Hello World");
//Output: Hello World
```

In the body of the print_message() function, we call the console.log() function to output a message to the console.

Function Expressions:-

A **Function Expressions** defines a named or anonymous function. An anonymous function is a function that does not have any name. Function Expressions are not hoisted, and therefore we cannot call them before they are defined. In the example below, we are setting the anonymous function object equal to a variable. Following is the syntax of function expression :

```
let variable_name = function(parameters){
  // statements
 }
```

Example:-

```
let add=function(a, b) {
    return a + b;
}
console.log(add(4,3));
//Output: 7
```

Returning a value:-

JavaScript function returns undefined. See the following example:

```
function print_message(message) {
    console.log(message);
}
let result = say('Hello World');
console.log('Result:', result);
//Output:
//Hello World
//Result: undefined
```

To specify a return value for a function, use the return statement followed by an expression or a value, like this:
return expression;
Let's create a new function called get_distance:

```
function get_Distance(speed, time) {
      let dist = speed * time;
      return dist;
}
```

To call the get_Distance function, we can call it as part of initializing a variable:

```
var myDistance = get_Distance(8, 5);
```

When the getDistance function gets called, it gets evaluated and returns a numerical value that is then assigned to the myDistance variable. That's all there is to it.

Key Takeaways:-

In today's tutorial, we have learned that the function is a *subprogram* designed to perform a particular task. Functions are executed when they are called. This is

known as *invoking* a function. Arguments can be *passed* into functions and used within the function. Functions *always* return a value. In JavaScript, if no return value is specified, the function will return undefined

Code index.html as described/written in the video

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Tutorial on Js</title>
</head>
<body>
    <h1>This is Js tutorial by Harry</h1>
</body>

<!-- <script src="js/tut2.js"></script> -->
<!-- <script src="js/tut3.js"></script> -->
<!-- <script src="js/tut4.js"></script> -->
<!-- <script src="js/tut5.js"></script> -->
<!-- <script src="js/tut6.js"></script> -->
<!-- <script src="js/tut7.js"></script> -->
<!-- <script src="js/tut8.js"></script> -->
<!-- <script src="js/tut9.js"></script> -->
<!-- <script src="js/tut10.js"></script> -->
<script src="js/tut11.js"></script>
</html>
```

Js code as described/written in the video

```
  console.log('this is tut 10');

if(1){
    let i =234;
    console.log(i)
}
```

```
console.log(i);

function ui(name)
{
    let i = 9;
    console.log(i);
    return `This is a ${name} ui`;
}

console.log(ui("harry"), i)
// const mygreet = function(name, thank='Thank You'){
//     let msg = `Happy Birthday ${name} How I wish I could fly to you right now and be with you on this special day of yours. But rem
//     return msg;

// }


// let name = 'SkillF';
// let name2 ='Rohan';

// let val = mygreet(name, 'Thanks a lot');
// console.log(val);


// const myobj = {
//     name: "SkillF",
//     game: function(){
//         return "GTA";
//     }
// }
// console.log(myobj.game())

// arr = ['fruit', 'vegetable', 'furniture'];

// arr.forEach(function(element, index, array) {
//     console.log(element, index)
// });
```

Previous

Next

CodeWithHarry