Introduction to JavaScript

What is JavaScript?

- JavaScript was first known as **LiveScript**
- Netscape changed its name to JavaScript
- The ECMA-262 Specification defined a standard version of the core JavaScript language
  - JavaScript is a lightweight, interpreted programming language
  - Designed for creating network-centric applications
  - Complementary to and integrated with Java
  - Complementary to and integrated with HTML
  - Open and cross-platform

Client-side JavaScript

- Client-side JavaScript is the most common form of the language
- he script should be included in or referenced by an HTML document
- Web page can include programs that interact with the user, control the browser, and dynamically create HTML content.
- The JavaScript client-side mechanism provides many advantages over traditional CGI server-side scripts
  - Check if the user has entered a valid e-mail address
- The JavaScript code is executed when the user submits the form
- JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions

Advantages of JavaScript

- **Less server interaction**: validate user input before sending the page off to the server => saves server traffic
- **Immediate feedback to the visitors**: they don't have to wait for a page reload to see if they have forgotten to enter something.
- **Increased interactivity** – You can create interfaces that react when the user hovers over them with a mouse or via the keyboard.
- **Richer interfaces** – You can use JavaScript to include such items as drag-and-drop components.

Limitations of JavaScript

- Cannot treat JavaScript as a full-fledged programming language.
- It lacks the following important features
  - Does not allow the reading or writing of files because of security reason.
  - Cannot be used for networking applications because there is no such support available.
  - Doesn't have any multithreading or multiprocessor capabilities.

Syntax

- JavaScript can be implemented using JavaScript statements that are placed within the **<script>... </script>** HTML tags in a web page
  - <script language="javascript" type="text/javascript">
  - JavaScript code
  - </script>
- **Language** – This attribute specifies what scripting language you are using.
- **Type** – should be set to "text/javascript".

Your first JavaScript Script

```
<html>
  <body>
    <script language="javascript" type="text/javascript">
      <!–
        document.write("Hello World!")
      //-->
    </script>
  </body>
</html>
```

Whitespace and Line Breaks

- JavaScript ignores spaces, tabs, and newlines that appear in JavaScript programs.
- You can use spaces, tabs, and newlines freely in your program.
- You are free to format and indent your programs in a neat and consistent way that makes the code easy to read and understand

Semicolons are Optional

```
<script language="javascript” type="text/javascript">
<!--
        var1 = 10
        var2 = 20
//-->
</script>

<script language="javascript" type="text/javascript">
<!--
        var1 = 10; var2 = 20;
//-->
</script>
```

Case sensitivity

- JavaScript is a case-sensitive language.
- This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.
    - The identifiers **Time** and **TIME** will convey different meanings in JavaScript.

Comments in JavaScript

- Any text between a // and the end of a line is treated as a comment and is ignored by JavaScript.
- Any text between the characters /* and */ is treated as a comment. This may span multiple lines.
- JavaScript also recognizes the HTML comment opening sequence <!--. JavaScript treats this as a single-line comment, just as it does the // comment.
- The HTML comment closing sequence --> is not recognized by JavaScript so it should be written as //-->.

JavaScript - Placement in HTML File

- Script in <head>...</head> section.
- Script in <body>...</body> section.
- Script in <body>...</body> and <head>...</head> sections.
- Script in an external file and then include in <head>...</head> section.

JavaScript in <head>...</head> section

```
<html>
  <head>
    <script type="text/javascript">
    <!–
      function sayHello()
      {
        alert("Hello World")
      }
    //-->
  </script>
</head>
<body>
<input type="button" onclick="sayHello()" value="Say Hello" />
</body>
</html>
```

JavaScript in <body>...</body> section

```
<html>
<head>
</head>
<body>
 <script type="text/javascript">
 <!–
  document.write("Hello World")
 //-->
 </script>
 <p>This is web page body </p>
</body>
</html>
```

JavaScript in <body> and <head> Sections

```
<html>
<head>
<script type="text/javascript">
  <!–
  function sayHello()
  {
    alert("Hello World")
  }
  //-->
</script>
</head>
```

```
<body>
<script type="text/javascript">
  <!–
  document.write("Hello World")
  //-->
</script>
<input type="button" onclick="sayHello()"
value="Say Hello" />
</body>
</html>
```

JavaScript in External File

```
<html>
<head>
 <script type="text/javascript" src="filename.js" >
 </script>
</head>
<body>
 .......
</body>
</html>
```

JavaScript Datatypes

- **Numbers,** eg. 123, 120.50 etc.
- **Strings** of text e.g. "This text string" etc.
- **Boolean** e.g. true or false.
- **null** and **undefined**
- JavaScript does not make a distinction between integer values and floating-point values
  - numbers use the 64-bit floating-point format defined by the IEEE 754 standard

JavaScript Variables

- you must declare variable before you use it.
- Variables are declared with the **var** keyword as follows
- Example:

```
<script type="text/javascript">
<!–
 var money; var name;
//-->
</script>
```

JavaScript Variable Scope

- JavaScript variables have only two scopes.
  - **Global Variables** − A global variable has global scope which means it can be defined anywhere in your JavaScript code.
  - **Local Variables** − A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

Example

```
<html>
<body onload = checkscope();>
<script type = "text/javascript">
  <!–
  var myVar = "global"; // Declare a   global variable
  function checkscope( )  {
    var myVar = "local"; // Declare a local variable
    document.write(myVar);
  }
//-->
</script>
</body>
</html>
```

Operators

- Arithmetic Operators
- Comparision Operators
- Logical (or Relational) Operators
- Assignment Operators
- Conditional (or ternary) Operators

Arithmetic Operators

- + (Addition)
- - (Subtraction)
- * (Multiplication)
- / (Division)
- % (Modulus)
- ++ (Increment)
- -- (Decrement)

Comparison Operators

- = = (Equal)
- != (Not Equal)
- > (Greater than)
- < (Less than)
- >= (Greater than or Equal to)
- <= (Less than or Equal to)

Logical Operators

- && (Logical AND)
- || (Logical OR)
- ! (Logical NOT)

Bitwise Operators

- & (Bitwise AND)
- | (BitWise OR)
- ^ (Bitwise XOR)
- ~ (Bitwise Not)
- << (Left Shift)
- >> (Right Shift)
- >>> (Right shift with Zero)

Assignment Operators

- = (Simple Assignment)
- += (Add and Assignment)
- −= (Subtract and Assignment)
- *= (Multiply and Assignment)
- /= (Divide and Assignment)
- %= (Modules and Assignment)

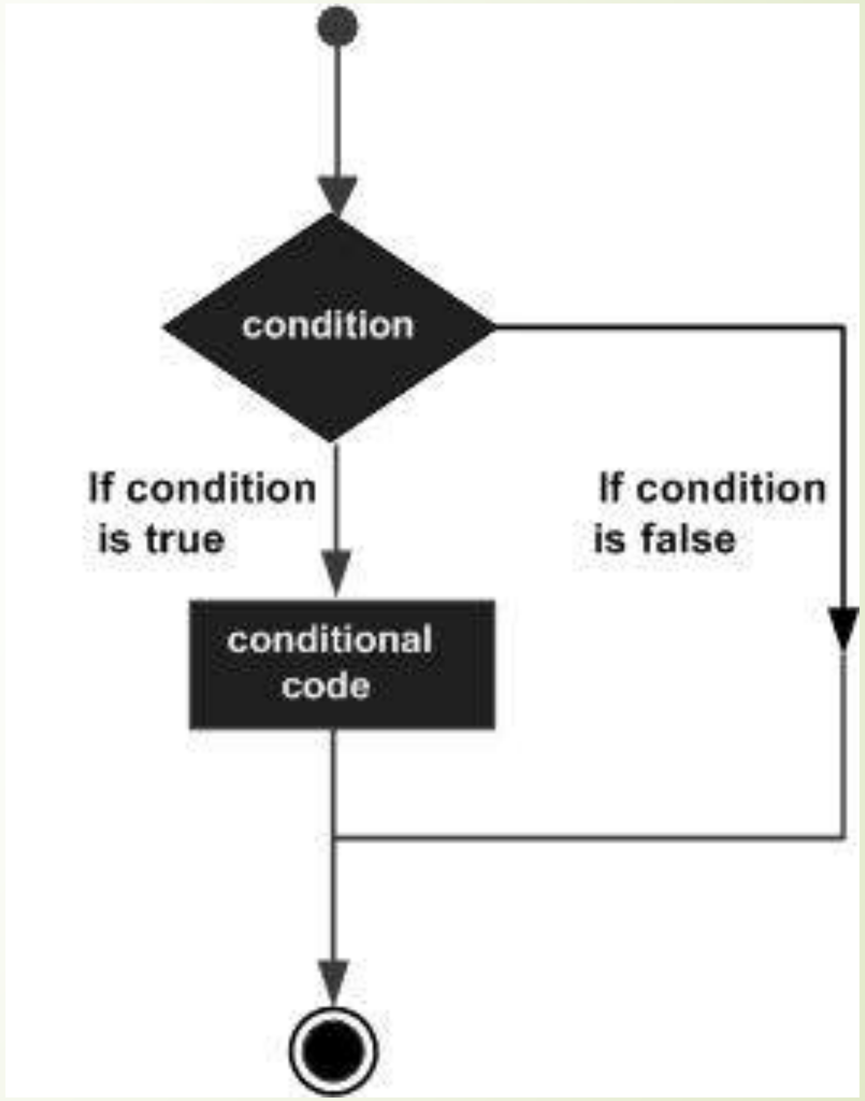
Miscellaneous Operator

- **? : (Conditional)**
- typeof Operator

Number	"number"
String	"string"
Boolean	"boolean"
Object	"object"
Function	"function"
Undefined	"undefined"
Null	"object"

if...else Statement

- Flow Chart of if-else
- JavaScript supports the following forms of **if..else** statement
  - if statement
  - if...else statement
  - if...else if... statement.

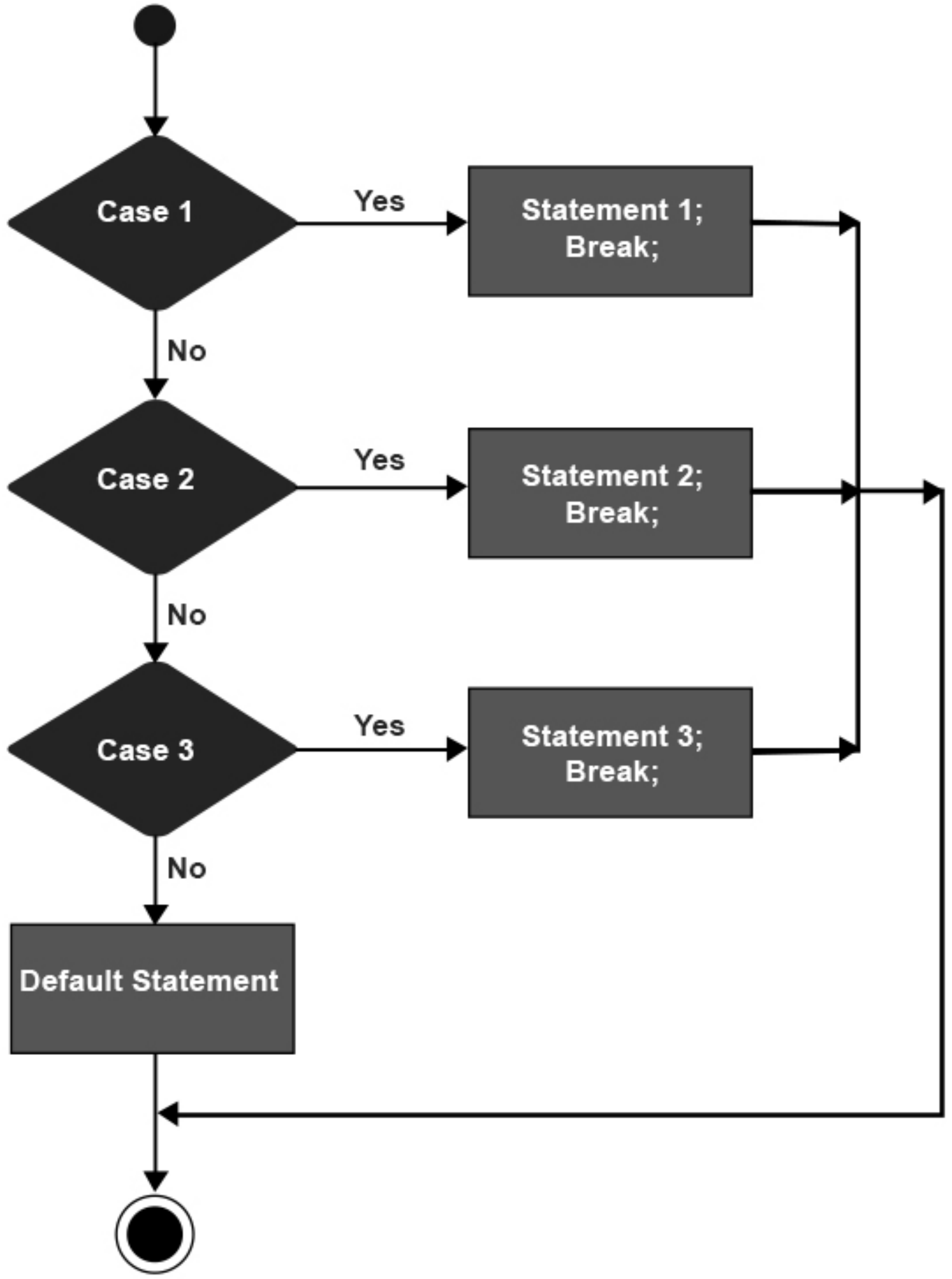


Example

```
<html>
<body>
<script type="text/javascript">
  <!–
  var age = 15;
  if( age > 18 ) {
    document.write("<b>Qualifies for driving</b>");
  } else {
    document.write("<b>Does not qualify for driving</b>");
  }
//-->
</script>
<p>Set the variable to different value and then try...</p>
</body>
</html>
```

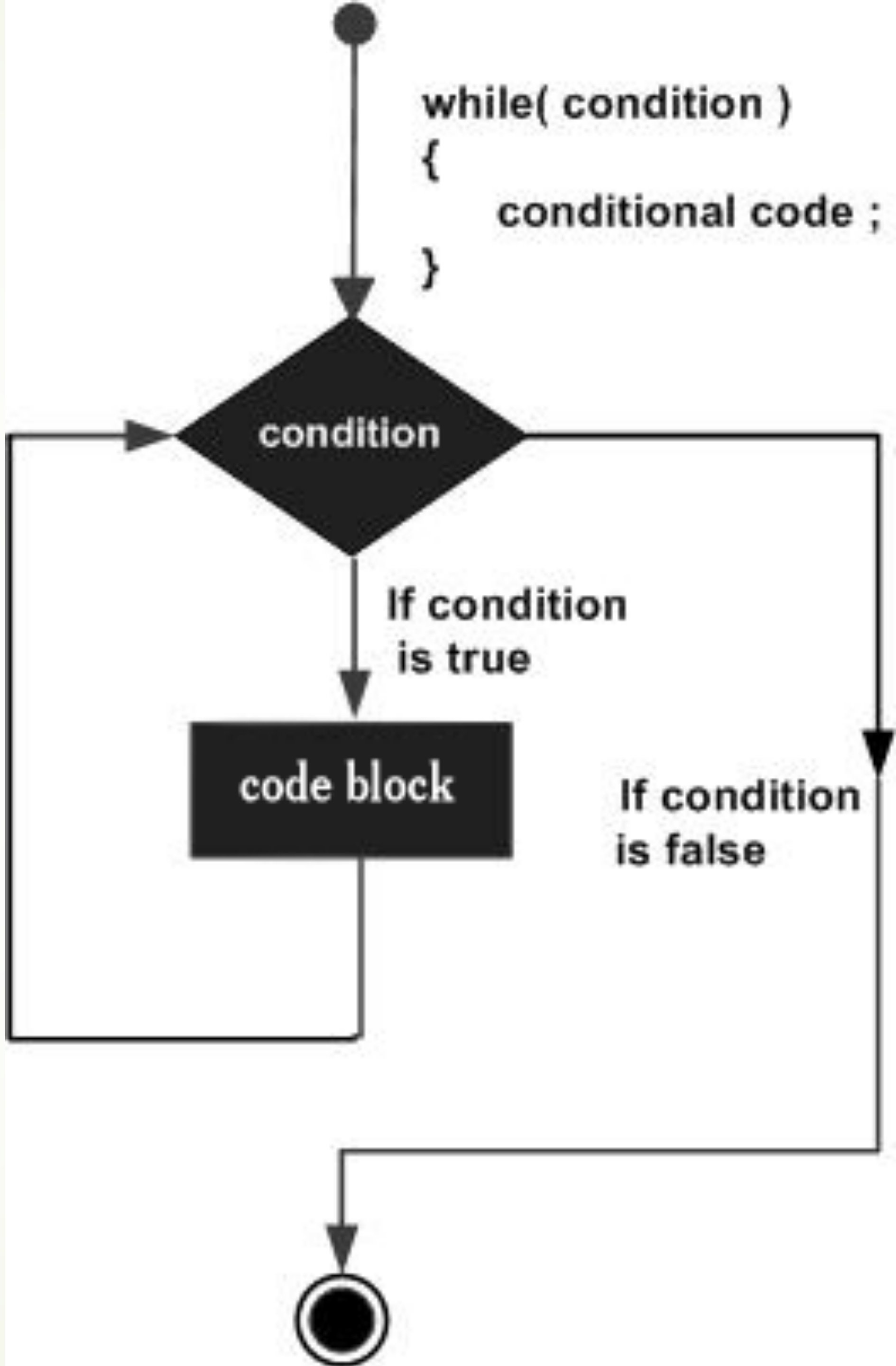
Switch Case

- Flow Chart



While Loops

- Flow Chart

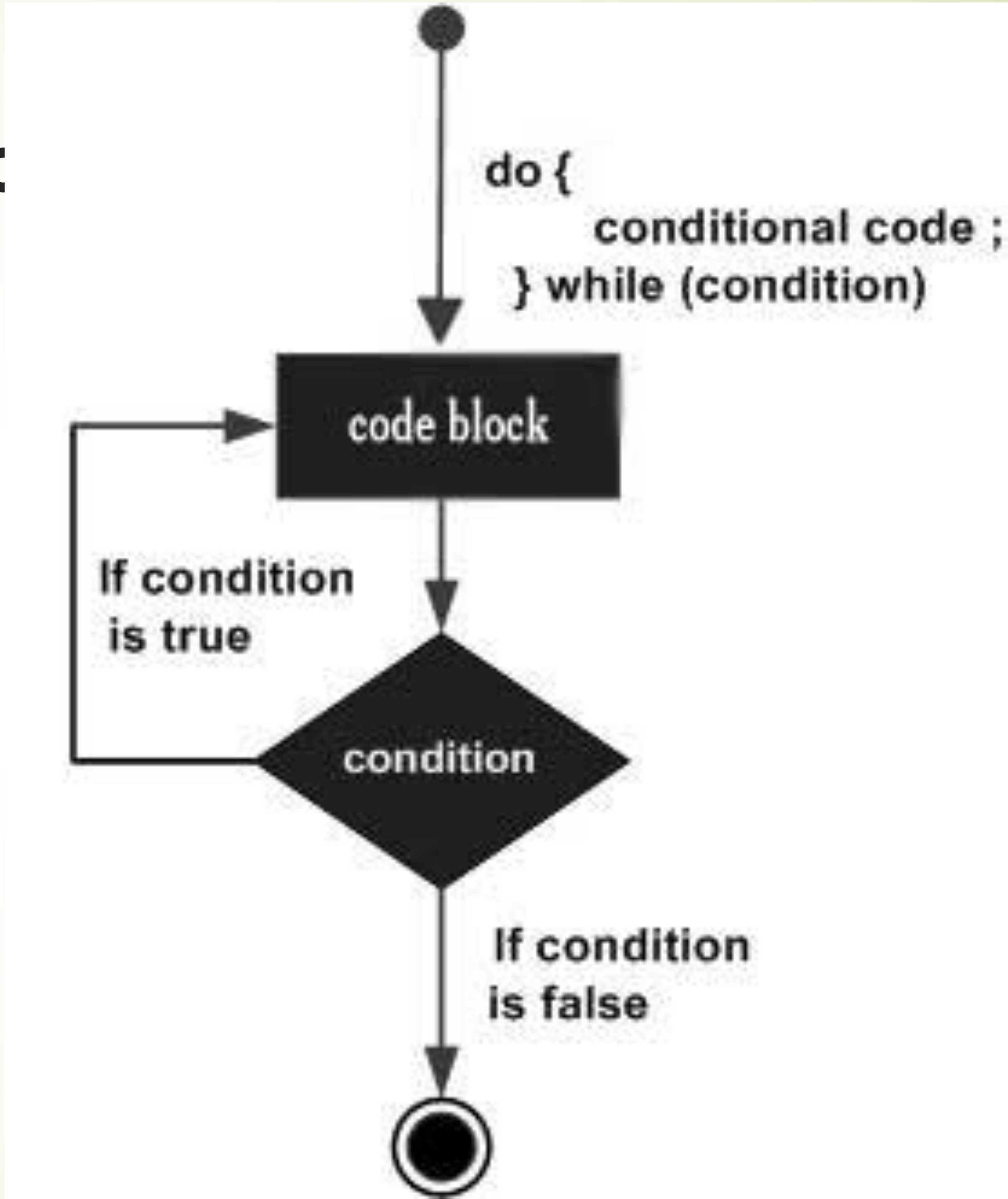


Example

```
<html>
<body>
<script type="text/javascript">
<!–
  var count = 0;
  document.write("Starting Loop ");
  while (count < 10) {
    document.write("Current Count : " + count + "<br />");
    count++;
  }
  document.write("Loop stopped!");
//-->
</script>
<p>Set the variable to different value and then try...</p>
</body>
 </html>
```

The do...while Loo

- Flow Chart

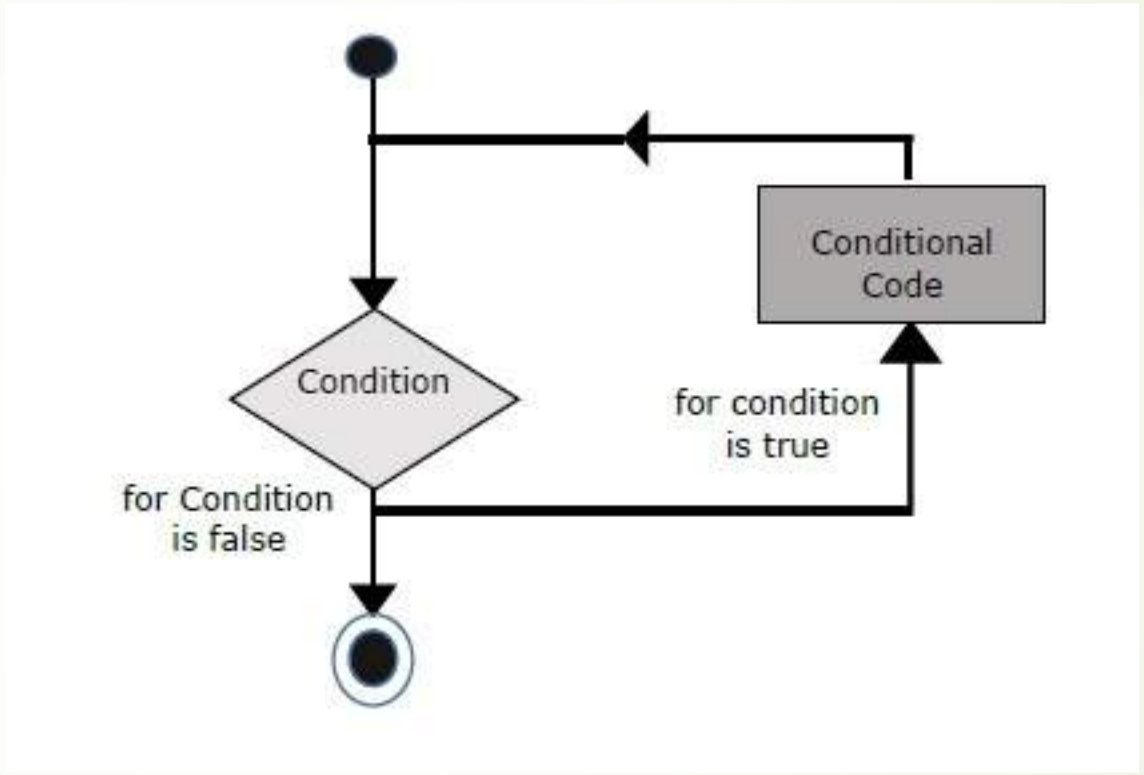


Example

```
<html>
<body>
<script type="text/javascript">
<!–
 var count = 0;
 document.write("Starting Loop ");
 do {
  document.write("Current Count : " + count + "<br />");
  count++;
 } while (count < 10);
 document.write("Loop stopped!");
//-->
</script>
<p>Set the variable to different value and then try...</p>
</body>
 </html>
```

For Loop

- Flow Chart



- Syntax

```
for (initialization; test condition; iteration statement) {
    Statement(s) to be executed if test condition is true
}
```

Example

```
<html>
<body>
<script type="text/javascript">
<!–
 var count;
 document.write("Starting Loop" + "<br />");
 for(count = 0; count < 10; count++) {
   document.write("Current Count : " + count );
   document.write("<br />");
 }
 document.write("Loop stopped!");
//-->
</script>
<p>Set the variable to different value and then try...</p>
</body>
</html>
```

Functions

- Function Definition

```
<script type="text/javascript">
<!–
 function functionname(parameter-list) {
  statements
 }
//-->
</script>
```

Event

- JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page.
- When the page loads, it is called an event.
- When the user clicks a button, that click too is an event.
- Other examples include events like pressing any key, closing a window, resizing a window, etc.
- Developers can use these events to execute JavaScript coded responses:
    - buttons to close windows
    - messages to be displayed to users
    - data to be validated

onclick Event Type

- Event type occurs when a user clicks the left button of his mouse
- You can put your validation, warning etc.

Example

```
<html>
<head>
<script type="text/javascript">
<!–
function sayHello() {
  alert("Hello World")
}
//-->
</script>
</head>
<body>
<p>Click the following button and see result</p>
<form>
  <input type="button" onclick="sayHello()" value="Say Hello" />
</form>
</body>
</html>
```

onsubmit Event type

- **onsubmit** is an event that occurs when you try to submit a form.
- You can put your form validation against this event type.

Example

```
<html>
<head>
<script type="text/javascript">
<!–
function validation() {
  all validation goes here .........
  return either true or false
}
//-->
</script>
</head>
<body>
<form method="POST" action="t.cgi" onsubmit="return validate()">
  .......
  <input type="submit" value="Submit" />
</form>
</body>
</html>
```

onmouseover and onmouseout

- The **onmouseover** event triggers when you bring your mouse over any element
- The **onmouseout** triggers when you move your mouse out from that element

Summary

- What is JavaScript
- Adding JavaScript to a page
- Anatomy of JavaScript
- Browser Object
- Events