

Lecture 8

JavaScript Loops

Different Types of Loops in JavaScript

Loops are used to execute the same block of code again and again, as long as a certain condition is met. The basic idea behind a loop is to automate the repetitive tasks within a program to save the time and effort. JavaScript now supports five different types of loops:

- **while** — loops through a block of code as long as the condition specified evaluates to true.
- **do...while** — loops through a block of code once; then the condition is evaluated. If the condition is true, the statement is repeated as long as the specified condition is true.
- **for** — loops through a block of code until the counter reaches a specified number.
- **for...in** — loops through the properties of an object.
- **for...of** — loops over iterable objects such as arrays, strings, etc.
- **IF** statement — is a conditional branching statement.
- **Switch** statement — is used to perform different actions on different conditions.

In the following sections, we will discuss each of these loop statements in detail.

1. If Statement

- IF statement is a conditional branching statement.

- In 'IF' statement, if the condition is true a group of statement is executed. And if the condition is false, the following statement is skipped.

Syntax : If statement

```
if(condition)
{
    //Statement 1;
    //Statement 2;
}
```

Example : Simple Program for IF Statement

```
<html>
  <body>
    <script type="text/javascript">
      var num = prompt("Enter Number");
      if (num > 0)
      {
        alert("Given number is Positive!!!");
      }
    </script>
  </body>
</html>
```

Output:

JavaScript

Enter Number

2

Cancel OK

JavaScript Alert

Given number is Positive!!!

☐ Prevent this page from creating additional dialogs.

OK

2. If – Else Statement

- If – Else is a two-way decision statement.
- It is used to make decisions and execute statements conditionally.

Flow Diagram of If – Else Statement

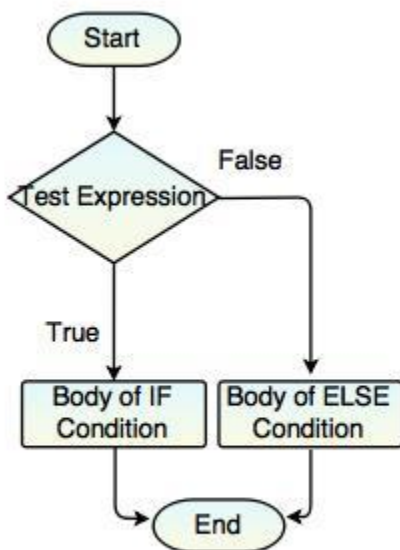


Fig. Flow Diagram of IF - ELSE Statement

Syntax : If-Else statement

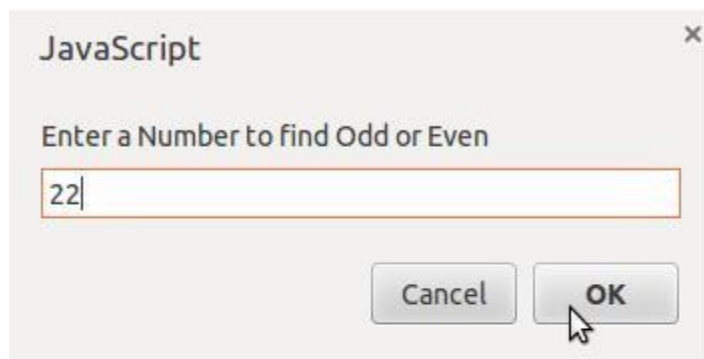
```
if (condition)
{
    //Statement 1;
}
else if(condition)
{
    //Statement 2;
}
else
{
    //Statement 3;
}
```

Example : Simple Program for If-Else Statement

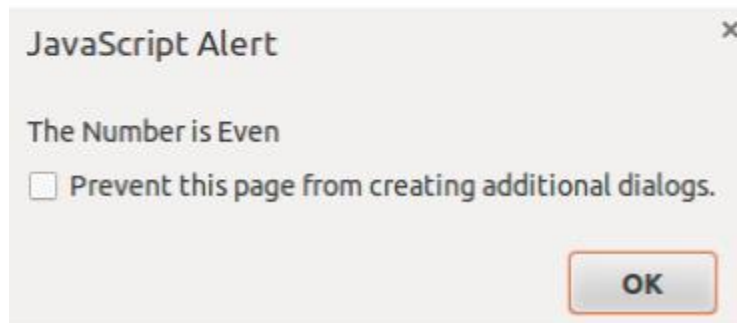
```
<html>
<head>
<script type="text/javascript">
    var no = prompt("Enter a Number to find Odd or Even");
    no = parseInt(no);
    if (isNaN(no))
    {
        alert("Please Enter a Number");
    }
    else if (no == 0)
    {
        alert("The Number is Zero");
    }
    else if (no % 2)
    {
        alert("The Number is Odd");
    }
}
```

```
        else
        {
            alert("The Number is Even");
        }
    </script>
</head>
</html>
```

Output:



A JavaScript dialog box titled "JavaScript" with a close button (X) in the top right corner. The text inside reads "Enter a Number to find Odd or Even". Below the text is a text input field containing the number "22". At the bottom of the dialog are two buttons: "Cancel" and "OK". A mouse cursor is pointing at the "OK" button.



A JavaScript Alert dialog box titled "JavaScript Alert" with a close button (X) in the top right corner. The text inside reads "The Number is Even". Below the text is a checkbox with the label "Prevent this page from creating additional dialogs." which is currently unchecked. At the bottom right of the dialog is an "OK" button.

JavaScript ×

Enter a Number to find Odd or Even

33

Cancel OK

JavaScript Alert ×

The Number is Odd

☐ Prevent this page from creating additional dialogs.

OK

JavaScript ×

Enter a Number to find Odd or Even

ABC

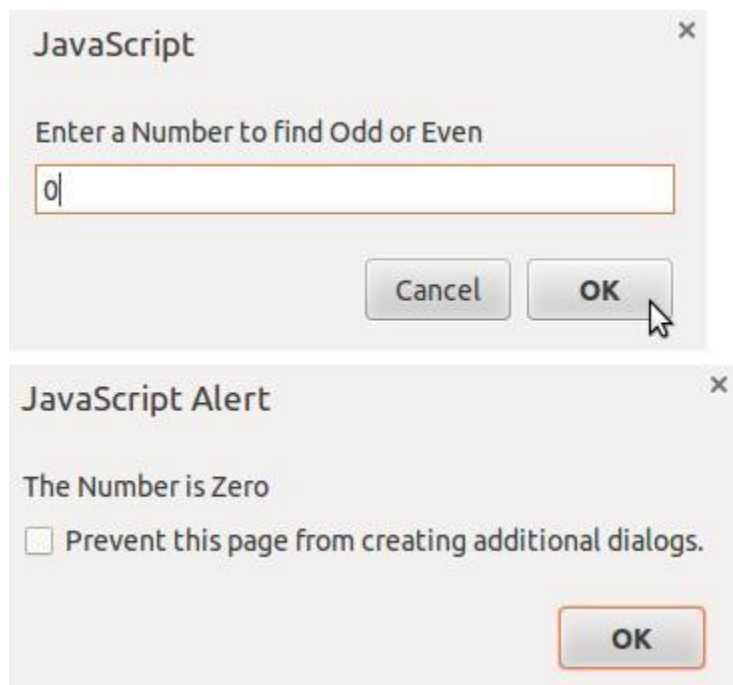
Cancel OK

JavaScript Alert ×

Please Enter a Number

☐ Prevent this page from creating additional dialogs.

OK



3. Switch Statement

- Switch is used to perform different actions on different conditions.
- It is used to compare the same expression to several different values.

Flow Diagram of Switch Statement

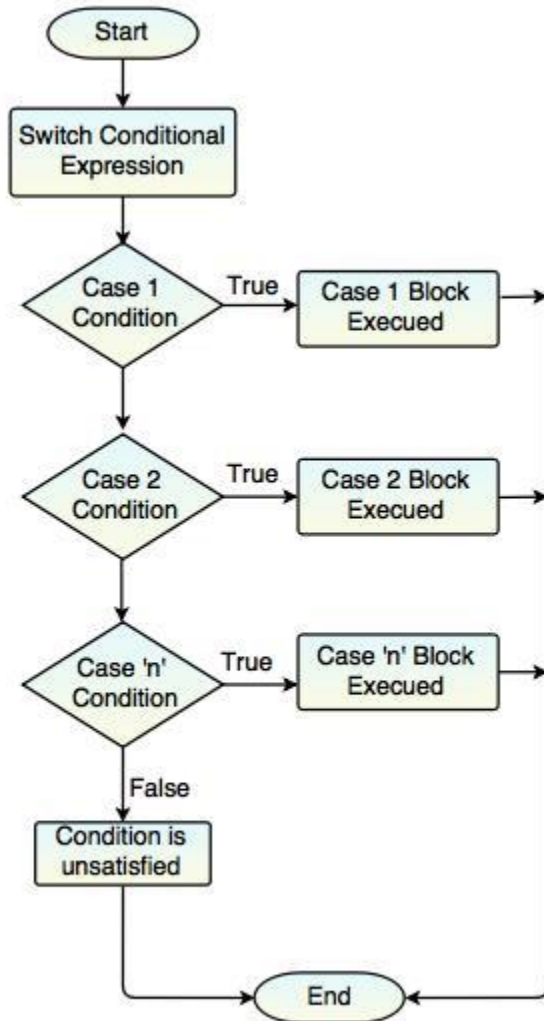


Fig. Flow Diagram of Switch Statement

Syntax

```
switch(expression)
{
    case condition 1:
        //Statements;
        break;
    case condition 2:
        //Statements;
        break;
```



```

case condition 3:
    //Statements;
    break;
.
.
case condition n:
    //Statements;
    break;
default:
    //Statement;
}

```

Example : Simple Program for Switch Statement

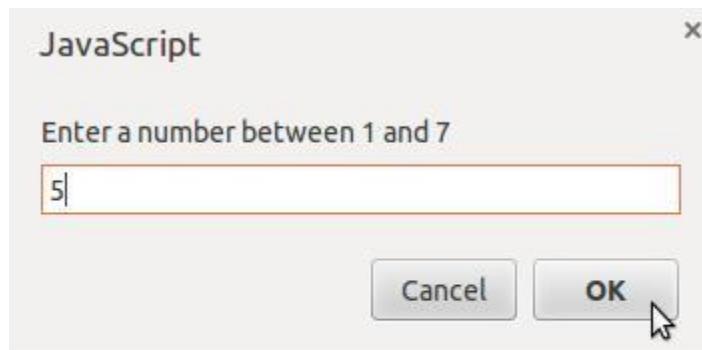
```

<html>
<head>
    <script type="text/javascript">
var day = prompt("Enter a number between 1 and 7");
switch (day)
{
    case (day="1"):
        document.write("Sunday");
        break;
    case (day="2"):
        document.write("Monday");
        break;
    case (day="3"):
        document.write("Tuesday");
        break;
    case (day="4"):
        document.write("Wednesday");
        break;
    case (day="5"):

```

```
        document.write("Thursday");
        break;
    case (day="6"):
        document.write("Friday");
        break;
    case (day="7"):
        document.write("Saturday");
        break;
    default:
        document.write("Invalid Weekday");
        break;
}
</script>
</head>
</html>
```

Output:

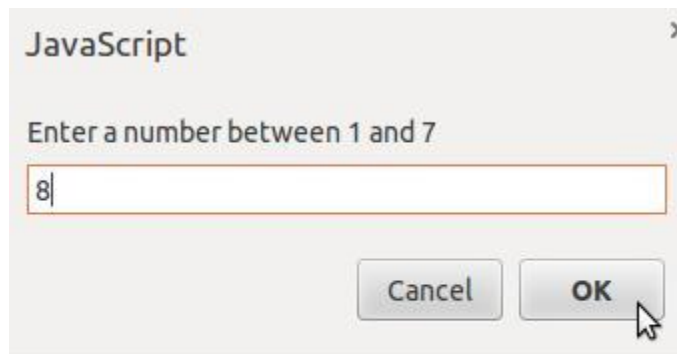
A JavaScript dialog box titled "JavaScript" with a close button (X) in the top right corner. The dialog contains the text "Enter a number between 1 and 7" above a text input field. The input field contains the number "5". Below the input field are two buttons: "Cancel" and "OK". A mouse cursor is pointing at the "OK" button.

JavaScript

Enter a number between 1 and 7

Cancel OK

Thursday



Invalid Weekday

4. For Loop

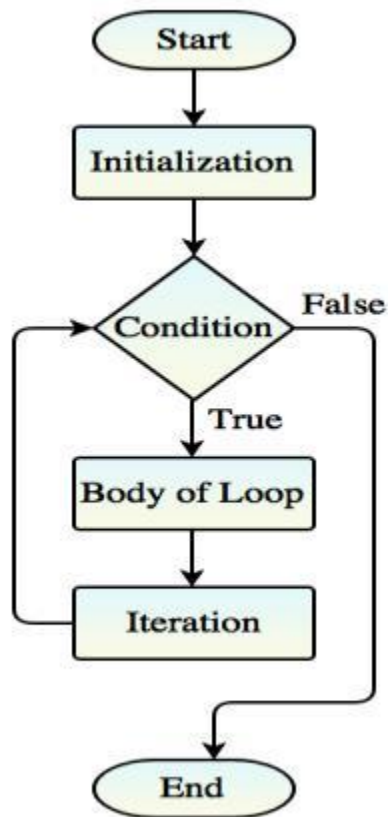
- For loop is a compact form of looping.

It includes three important parts:

1. Loop Initialization
2. Test Condition
3. Iteration

- All these three parts come in a single line separated by semicolons(;).

Flow Diagram of 'For' Loop



Syntax

```
for(initialization; test-condition; increment/decrement)
{
    //Statements;
}
```

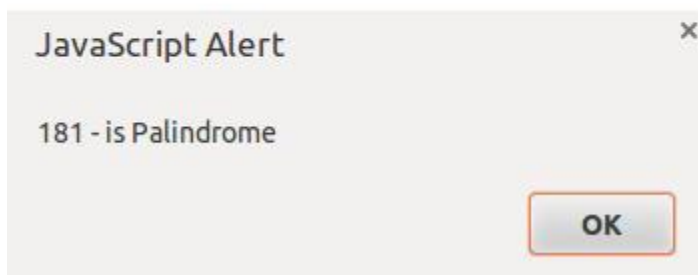
Example : Palindrome Program using For Loop

```
<html>
<body>
    <script type="text/javascript">
        function palindrome()
        {
            var revstr = " ";
            var strr = document.getElementById("strr").value;
```

```
var i = strr.length;
for(var j=i; j>=0; j--)
{
    revstr = revstr+strr.charAt(j);
}
if(strr == revstr)
{
    alert(strr+" - is Palindrome");
}
else
{
    alert(strr+" - is not a Palindrome");
}
}
</script>
<form>
    Enter a String or Number: <input type="text" id="strr"
name="checkpalindrome"><br>
    <input type="submit" value="Check" onclick="palindrome();">
</form>
</body>
</html>
```

Output:

Enter a String or Number:



Enter a String or Number:



5. For-in Loop

- The `for-in` loop is a special type of a loop that iterates over the properties of an [object](#), or the elements of an array. The generic syntax of the `for-in` loop is:

Syntax

```
for (variable_name in Object)
{
    //Statements;
}
```

6. While Loop

- While loop is an entry-controlled loop statement.
- It is the most basic loop in JavaScript.
- It executes a statement repeatedly as long as expression is true.
- Once the expression becomes false, the loop terminates.

Flow Diagram of While Loop

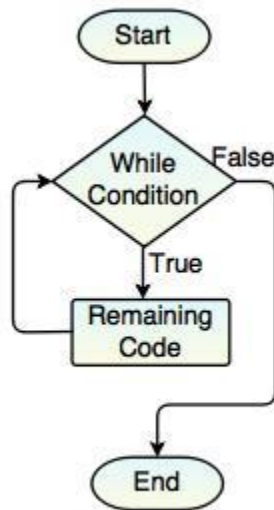


Fig. Flow Diagram of While Loop

Syntax

```
while (condition)
{
    //Statements;
}
```

Example : Fibonacci Series Program using While Loop

```
<html>
<body>
    <script type="text/javascript">
        var no1=0,no2=1,no3=0;
        document.write("Fibonacci Series:"+"<br>");
        while (no2<=10)
        {
            no3 = no1+no2;
            no1 = no2;
            no2 = no3;
            document.write(no3+"<br>");
        }
    </script>
</body>
</html>
```

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

Output:

Fibonacci Series:

1
2
3
5
8
13

7. Do-While Loop

- Do-While loop is an exit-controlled loop statement.
- Similar to the While loop, the only difference is condition will be checked at the end of the loop.
- The loop is executed at least once, even if the condition is false.

Flow Diagram of Do – While

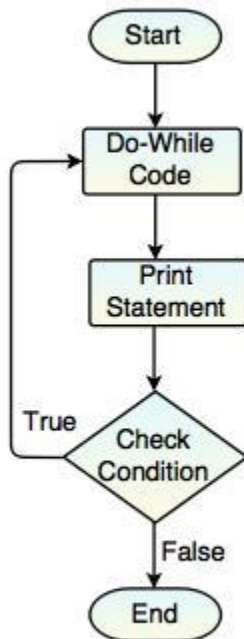


Fig. Flow Diagram of Do-While Loop

Syntax

```
do
{
    //Statements;
}
while(condition);
```

Example : Simple Program on Do-While Loop

```
<html>
<body>
    <script type = "text/javascript">
        var i = 0;
        do
        {
            document.write(i+"<br>")
            i++;
        }
        while (i <= 5)
    </script>
</body>
</html>
```

Output:

```
0
1
2
3
4
5
```

Difference between While Loop and Do – While Loop

While Loop	Do – While Loop
In while loop, first it checks the condition and then executes the program.	In Do – While loop, first it executes the program and then checks the condition.
It is an entry – controlled loop.	It is an exit – controlled loop.
The condition will come before the body.	The condition will come after the body.
If the condition is false, then it terminates the loop.	It runs at least once, even though the conditional is false.
It is a counter-controlled loop.	It is a iterative control loop.

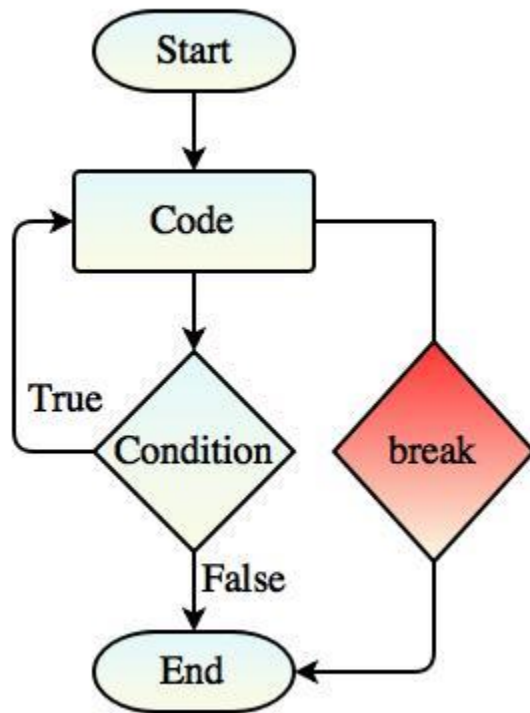
8. Break Statement

- Break statement is used to jump out of a loop.
- It is used to exit a loop early, breaking out of the enclosing curly braces.

Syntax:

break;

Flow Diagram of Break Statement



9. Continue Statement

- Continue statement causes the loop to continue with the next iteration.
- It skips the remaining code block.

Flow Diagram of Continue Statement

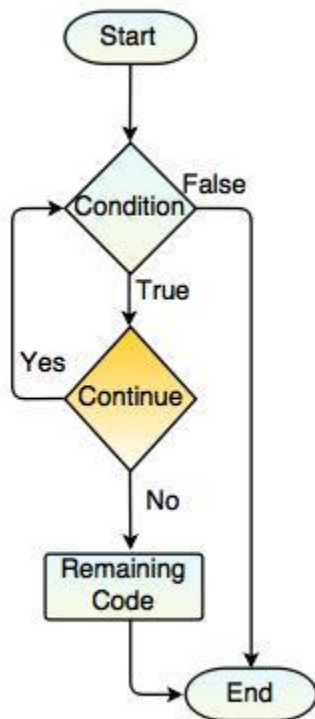


Fig. Flow Diagram of Continue Statement

Syntax:
`continue`