Objectives

- In this session, you will learn to:
  - Understand scripting
  - Implement JavaScript in Web pages
  - Use variables, operators, and control structures
  - Use variables, operators, and control structures
  - Implement functions

- Scenario:

Facilitates Online Registration

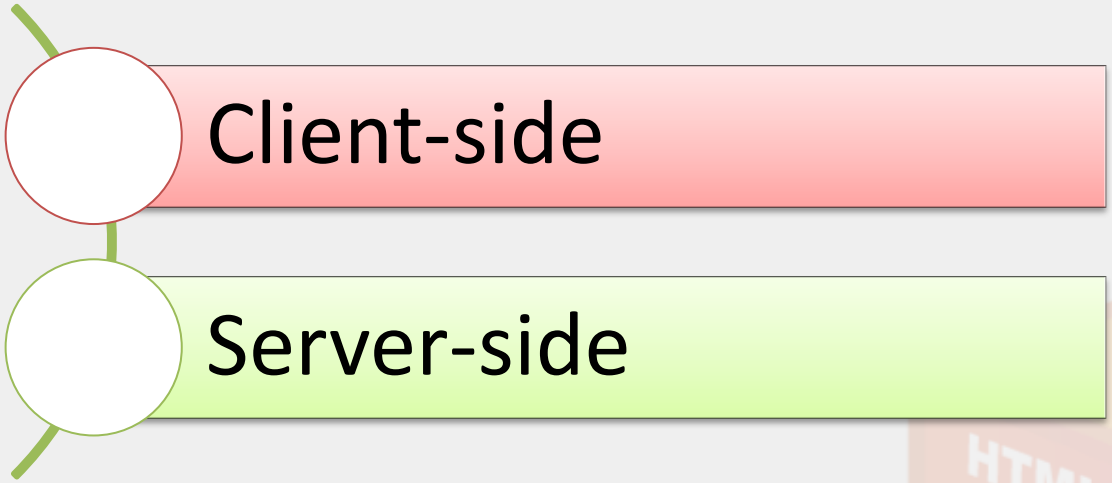
Users are asked to specify mail or email as the mode of communication

User chooses the desired option

Based on the choice made, user is provided fields to enter details

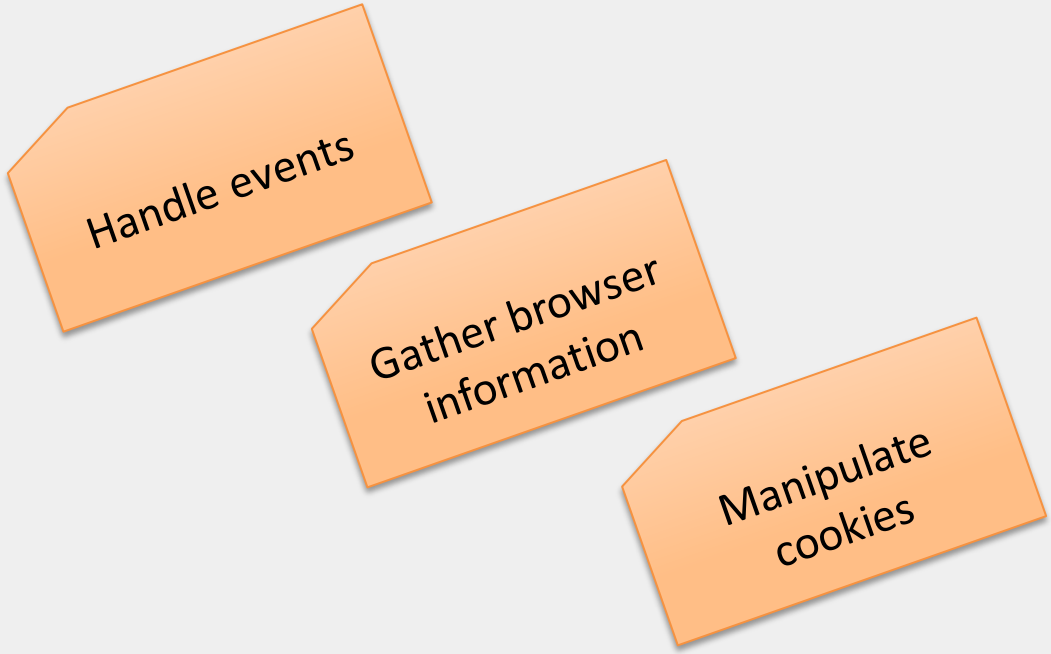
Types of Scripting

- A script:
  - Is a block of code that is incorporated in Web pages to make them dynamic and interactive.
  - Can be of two types:



Identifying the Benefits of JavaScript

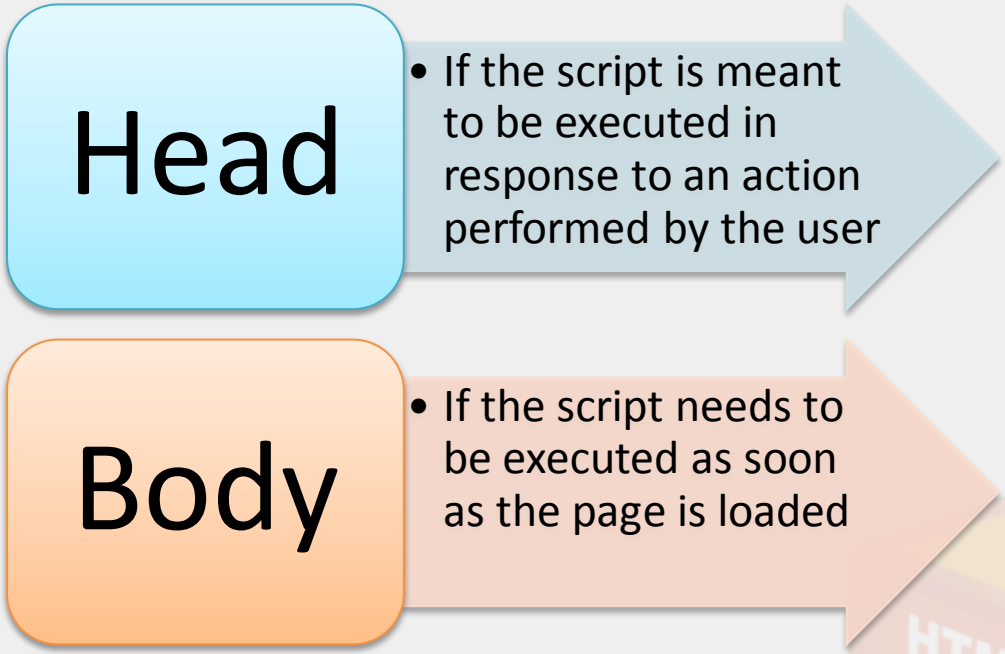
- JavaScript provides the following benefits:



- JavaScript:
  - Is a client-side scripting language.
  - Can be directly embedded into a Web page by writing the code inside the `<SCRIPT>` tag.
  - Code can also be written in an external JavaScript (.js) file.

Embedding a Script into a Web Page

- The JavaScript code:
  - Can be inserted in the following sections of the HTML document by using the `<SCRIPT>` tag:



  - Can be embedded into a Web page by using the following syntax:

```
<SCRIPT type="text/javascript"> JavaScript statements
</SCRIPT>
```

Creating and Using an External File

- An external JavaScript file:
  - Is saved with the .js extension.
  - Can be referred inside an HTML document using the `src` attribute of the `<SCRIPT>` tag.

Identifying Rules and Conventions Used in JavaScript

- JavaScript rules and conventions:

Semicolons

Quotes

Case sensitivity

Comments

- In JavaScript, to compare values and evaluate expressions, you need:

Variables

Operators

Conditional Constructs

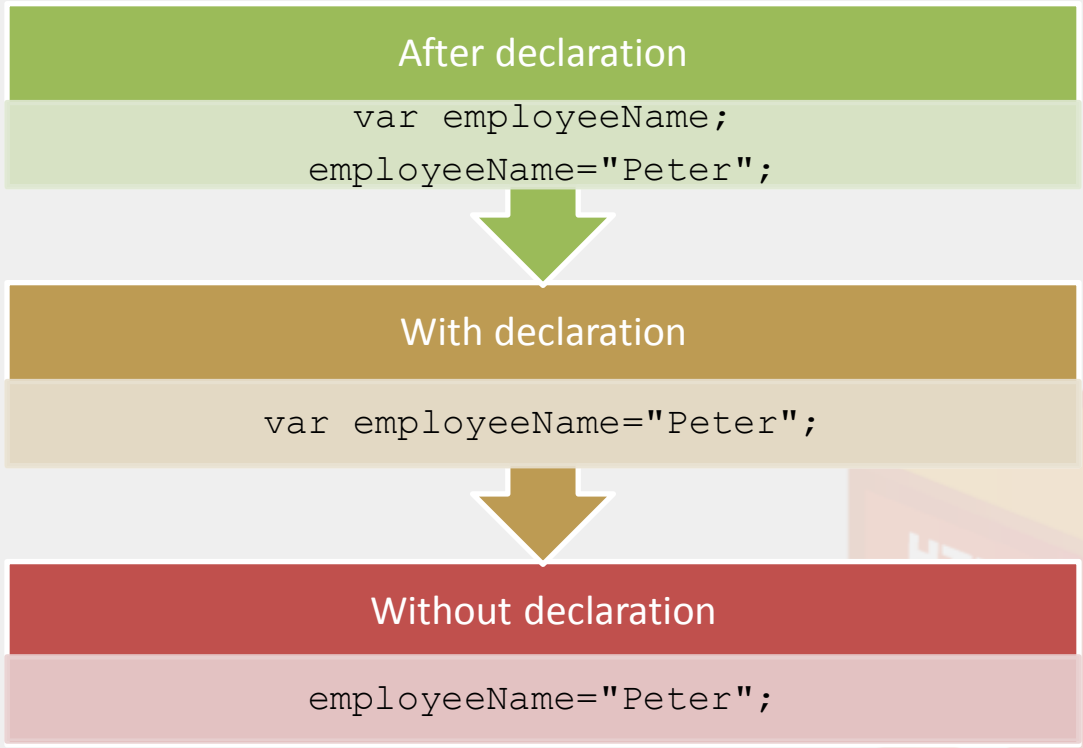
Looping Constructs

Defining Variables

- A variable:
  - Is a named location in memory that is used to store a value.
  - Is declared by using the following syntax:

```
var var_name;
```

  - Can be assigned a value in the following ways:



- An operator:
  - Is a set of one or more characters that is used for computations or comparisons.
  - Can be used to modify the values stored in the variables.
  - Can belong to any one of the following categories:

Arithmetic	Assignment	Arithmetic Assignment
Comparison	Logical	

Using Operators (Contd.)

- An arithmetic operator:
  - Is used to perform arithmetic operations on variables and literals.
  - Can be of the following types:
    - +
    - −
    - *
    - /
    - %
- An assignment operator:
  - Is used to assign a value or a result of an expression to a variable.

Using Operators (Contd.)

- Arithmetic assignment operators:
  - Are used to perform arithmetic operations and assign the value to the variable at the left side of the operator.
  - Are of the following types:
    - +=
    - -=
    - *=
    - /=
    - %=

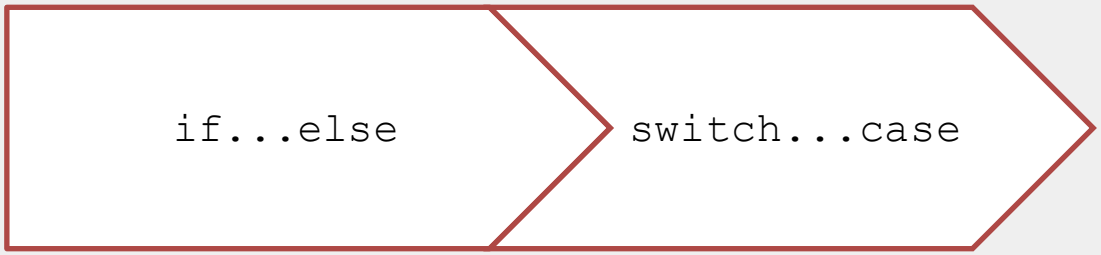
Using Operators (Contd.)

- Comparison operators:
  - Are used to compare two values and perform an action on the basis of the comparison.
  - Are of the following types:
    - <
    - >
    - <=
    - >=
    - ==
    - !=
    - ===

Using Operators (Contd.)

- Logical operators:
  - Are used to evaluate complex expressions.
  - Return a boolean value.
  - Are of the following types:
    - &&
    - !
    - ||

- Conditional constructs:
  - Allow you to execute a block of statements based on the result of the expression being evaluated.
  - Can be of the following types:



Using Conditional Constructs (Contd.)

- The `if…else` construct:
  - Is used to evaluate the specified condition and perform actions on the basis of the result of evaluation.
  - Has the following syntax:

```
if (exp)
{
// Statements;
}
else
{
// Statements;
}
```

- The `switch…case` construct is used when you need to evaluate a variable for multiple values.
- The following embedded Notepad file contains the code to understand the use of `switch…case` construct in HTML:

switch case

- Loop structures:
  - Are used to repeatedly execute one or more lines of code.
  - Can be of the following types:

`while` | `do...while` | `for`

- The `while` loop:
  - Is used to repeatedly execute a block of statements till a condition evaluates to true.
  - Always checks the condition before executing the statements in the loop.
  - Has the following syntax:

```
while (expression)
{
statements;
}
```

- The `do…while` loop:
  - Is executed at least once, even if the condition evaluates to false.
  - Has the following syntax:

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

- The `for` loop:
  - Allows the execution of a block of code depending on the result of the evaluation of the test condition.
  - Has the following syntax:

```
for (initialize variable; test condition; step value)
{
// code block
}
```

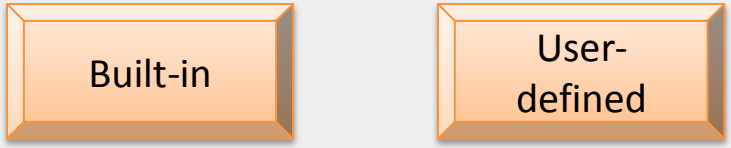
Break and Continue Statements

- The `break` statement:
  - Is used to exit the loop.
  - Prevents the execution of the remaining statements of the loop.
  - Is usually placed within an `if` construct inside the loop.
- The `continue` statement:
  - Is used to skip all the subsequent instructions and take the control back to the beginning of the loop.

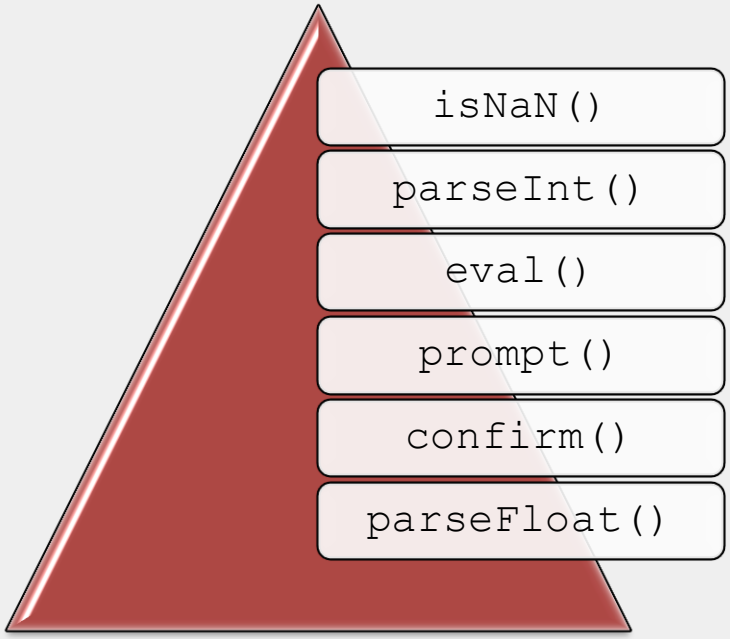
Implementing Functions

- Functions are used to write the code that needs to reused.
- They optimize the performance of the code.

- Functions:
  - Are a self-contained block of statements that have a name.
  - Are of the following types:



- Built-in functions:
  - Are ready to use as they are already coded.
  - For example:



- User-defined functions:
  - Are defined according to the need of the user.

Creating Functions

- Functions:
  - Are created by using the keyword, `function`, followed by the function name and the parentheses.
  - Are normally defined in the head section of a Web page.
  - Can optionally accept a list of parameters.
  - Are created using the following syntax:

```
function [functionName] (Variable1, Variable2)
{
//function statements
}
```

Accessing Functions

- A function is called by using the following syntax:

```
functionName ();
        or
functionName (parameter1, parameter 2...);
```

- A function returns a value by using the `return` statement as displayed in the following example:

```
function functionName()
{
var variable=10;
return variable;
}
```

Summary

- In this session, you learned that:
  - In JavaScript, the following conditional constructs can be used:
    - `if...else`
    - `switch...case`
  - In JavaScript, the following loop structures can be used:
    - `while`
    - `do...while`
    - `for`
  - A function is a self-contained block of statements that has a name.
  - Some of the built-in functions supported by JavaScript are:
    - `isNaN()`
    - `parseInt()`
    - `parseFloat()`
    - `eval()`
    - `prompt()`
    - `confirm()`
  - Functions are created by using the keyword, `function`, followed by the function name and the parentheses.
  - Functions can return a value using the `return` statement.