MCA 303: Web Technologies

UNIT I

Introduction to Internet-Browser Architecture-IE: Chrome-Search Engines-Introduction to HTML-5-HTML-5 Tags-Audio, Video Tags – HTML-5 Forms-Controls-CSS Styling-CSS Tags-Attributes.

UNIT II

Java Script-JQuery: JavaScript Programming Scripts- Control structures- Functions-Document, Browser, Date, Math, String objects-Events- JQuery Libraries-JQuery Objects, Functions – JQuery Events-Animations.

UNIT III

AJAX Concepts: Simple AJAX objects-Ajax Libraries-Examples, Webservers IIS, Tomcat-Hosting Website in a Web servers.

UNIT IV

Introduction to PHP: Control Structures-Arrays-Functions-Database connectivity-Introduction to ZEND Framework and applications

UNIT-V

Introduction to Java Servlets: Servlet classes and interfaces - Java Database Connectivity- Introduction to JSP-Java Server Page scriptlets -JSP Objects-JSP Web applications.

Text Books:

1. Deitel, Deitel and Goldberg Internet & World Wide Wide how to program"by End. Pearson Education
2. Ivan Bayross, Webenavled commercial Application Development in Java 2.0 BPB.
3. HTML 5 Black book, Kogent Learning Solutions Inc.

Reference Books:

1. Raj Kamal Internet and web Technologies, Tata Mc Graw Hill, 2002.
2. Chirs Bates, Web Programming, John Wiley, 2nd Edition

Lecture Notes
UNIT-1

Web browser:

- It is an application that we use when we browse the WWW.
- It renders text-based HTML documents into visual pages, which are what we see inside a browser.
- It speaks HTTP protocol and communicates with web servers.
- It understands URL and knows how to translates URL into web resources, e.g., HTML text files, images, videos, etc.
- It is a virtual machine that runs the JavaScript programs embedded inside HTML documents.
- It understands CSS rules and applies the rules to layout the pages.
- It interacts with a user in front of a browser and translates user inputs into browser events, e.g., clicking a link, clicking a button, submitting text inside a text box.
- It makes the WWW come alive!

Browser functionality

- caching: keep local copies of documents that have been downloaded and viewed before;
- authentication: validate the credential of the user when visiting a secure web site;
- state maintenance: keep "cookies" within a session across many request/response transactions;
- on-demand downloading: as an HTML document is being rendered, missing items (e.g., CSS style sheets, JavaScript code, images, audios, videos, etc.) will be loaded on demand;
- header processing: the header of an HTML document may contain "redirection";
- content type processing: process the embedded content (i.e., image formats, audio formats, video formats) accordingly;
- handling errors.

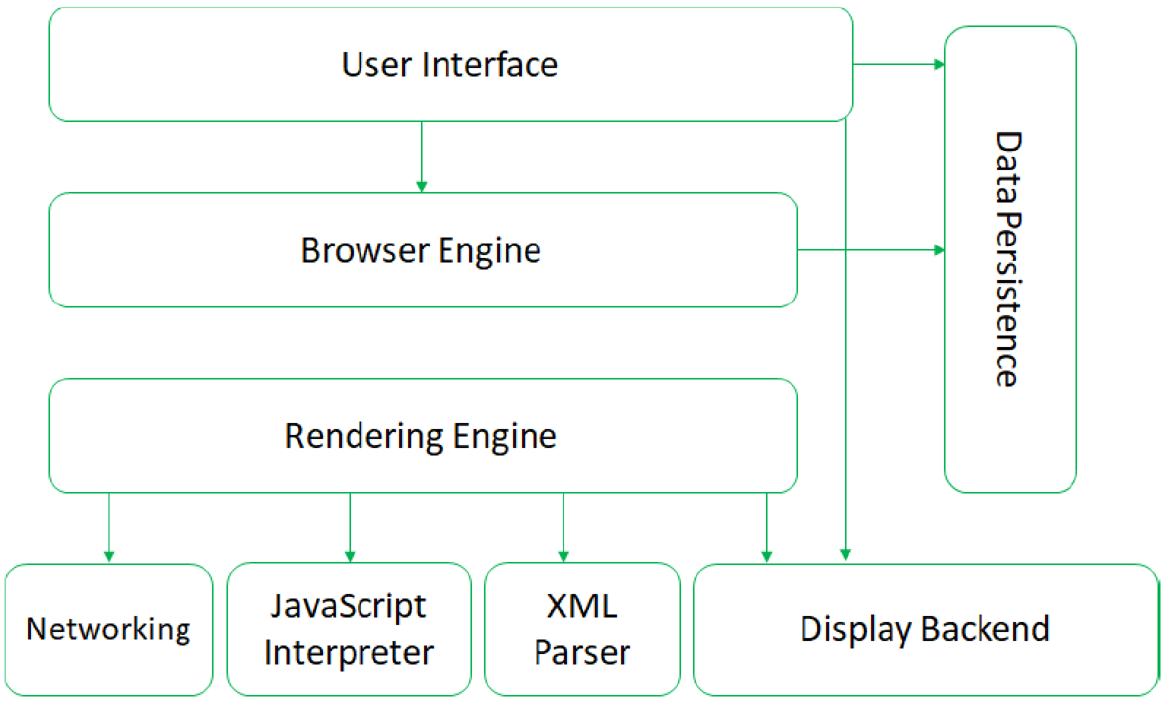
Browser Functionality and Architecture.

Browser Architecture

The Web Browser architecture is shown in below Figure. It has eight major subsystems and their dependencies:

1. The User Interface subsystem is the layer between the user and the Browser Engine. It provides features such as toolbars, visual page-load progress, smart download handling, preferences, and printing. It may be integrated with the desktop environment to provide browser session management or communication with other desktop applications.

2. The Browser Engine subsystem is an embedded component that provides a high-level interface to the Rendering Engine(Which is the next layer). It loads a given URL and supports primitive browsing actions such as forward, back, and reload. It provides hooks for viewing various aspects of the browsing session such as current page load progress and JavaScript alerts. It also allows the querying and manipulation of Rendering Engine settings.



3. The Rendering Engine subsystem produces a visual representation for a given URL. It is capable of displaying HTML and Extensible Markup Language (XML) documents, optionally styled with CSS, as well as embedded content such as images.

4. The Networking subsystem implements file transfer protocols such as HTTP and FTP

5. The JavaScript Interpreter evaluates JavaScript code, which may be embedded in web pages.

6. The XML Parser subsystem parses XML documents into a Document Object Model (DOM) tree. This is one of the most reusable subsystems in the architecture. In fact, almost all browser implementations leverage an existing XML Parser rather than rewriting their own from scratch.

7.The Display Back-end subsystem provides drawing and windowing primitives, a set of user interface widgets, and a set of fonts. It may be tied closely with the operating system.

8. The Data Persistence subsystem stores various data associated with the browsing session on disk. This may be high-level data such as bookmarks or toolbar settings, or it may be low-level data such as cookies, security certificates, or cache.

3. HTML5 and its features.

HTML5 is the next major revision of the HTML standard superseding HTML 4.01, XHTML 1.0, and XHTML 1.1. HTML5 is a standard for structuring and presenting content on the World Wide Web.

HTML5 is a cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG).

The new standard incorporates features like video playback and drag-and-drop that have been previously dependent on third-party browser plug-ins such as Adobe Flash, Microsoft Silverlight, and Google Gears.

Browser Support

The latest versions of Apple Safari, Google Chrome, Mozilla Firefox, and Opera all support many HTML5 features and Internet Explorer 9.0 will also have support for some HTML5 functionality.

New Features

HTML5 introduces a number of new elements and attributes that can help you in building modern websites. Here is a set of some of the most prominent features introduced in HTML5.

- **New Semantic Elements** − These are like <header>, <footer>, and <section>.
- **Forms 2.0** − Improvements to HTML web forms where new attributes have been introduced for <input> tag.
- **Persistent Local Storage** − To achieve without resorting to third-party plugins.
- **WebSocket** − A next-generation bidirectional communication technology for web applications.
- **Server-Sent Events** − HTML5 introduces events which flow from web server to the web browsers and they are called Server-Sent Events (SSE).
- **Canvas** − This supports a two-dimensional drawing surface that you can program with JavaScript.
- **Audio & Video** − You can embed audio or video on your webpages without resorting to third-party plugins.
- **Geolocation** − Now visitors can choose to share their physical location with your web application.
- **Microdata** − This lets you create your own vocabularies beyond HTML5 and extend your web pages with custom semantics.
- **Drag and drop** − Drag and drop the items from one location to another location on the same webpage.

HTML- basic Tags:

The **<!DOCTYPE html>** statement must always be the first to appear on an HTML page and tells the browser which version of the language is being used. In this case, we are working with HTML5.

The <html> and </html> tags tell the web browser where the HTML code starts and ends.

The HTML <head> Element

The HTML <head> element is a container for metadata. HTML metadata is data about the HTML document. Metadata is not displayed.

The <head> element is placed between the <html> tag and the <body> tag:

HTML Headings

Headings are defined with the <h1> to <h6> tags.

<h1> defines the most important heading. <h6> defines the least important heading.

Example

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
```

```
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>

<!DOCTYPE html>
<html>
<head>
  <title>My First HTML</title>
  <meta charset="UTF-8">
</head>
<body>
<p>The HTML head element contains meta data.</p>
<p>Meta data is data about the HTML document.</p>
</body>
</html>
```

HTML Paragraphs
The HTML <p> element defines a paragraph:

Example
```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

HTML Line Breaks
The HTML <br> element defines a line break.
Use <br> if you want a line break (a new line) without starting a new paragraph:

The HTML <pre> Element
The HTML <pre> element defines preformatted text.

The text inside a <pre> element is displayed in a fixed-width font (usually Courier), and it preserves both spaces and line breaks:

HTML Formatting Elements
In the previous chapter, you learned about the HTML style attribute.

HTML also defines special elements for defining text with a special meaning.

HTML uses elements like <b> and <i> for formatting output, like bold or italic text.

Formatting elements were designed to display special types of text:

<b> - Bold text
<strong> - Important text
<i> - Italic text
<em> - Emphasized text
<mark> - Marked text
<small> - Small text
<del> - Deleted text

<ins> - Inserted text
<sub> - Subscript text
<sup> - Superscript text

HTML Links - Hyperlinks

HTML links are hyperlinks.

You can click on a link and jump to another document.

When you move the mouse over a link, the mouse arrow will turn into a little hand.

Note: A link does not have to be text. It can be an image or any other HTML element.

HTML Links - Syntax

Hyperlinks are defined with the HTML `<a>` tag: `<a href="url">link text</a>`

```
<a href="https://www.w3schools.com/html/">Visit our HTML tutorial</a>
```

HTML Images

Images can improve the design and the appearance of a web page.

HTML Images Syntax

In HTML, images are defined with the `<img>` tag.

The `<img>` tag is empty, it contains attributes only, and does not have a closing tag.

The `src` attribute specifies the URL (web address) of the image:

```
<img src="url">
<img src="img_chania.jpg" alt="Flowers in Chania">
```

Lists in HTML

Unordered HTML List

An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.

The list items will be marked with bullets (small black circles) by default:

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Unordered HTML List - Choose List Item Marker

The CSS `list-style-type` property is used to define the style of the list item marker:

Value	Description
Disc	Sets the list item marker to a bullet (default)
Circle	Sets the list item marker to a circle
square	Sets the list item marker to a square
None	The list items will not be marked

Example - Disc

```
<ul style="list-style-type:disc;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Ordered HTML List

An ordered list starts with the `<ol>` tag. Each list item starts with the `<li>` tag.

The list items will be marked with numbers by default:

Example

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Ordered HTML List - The Type Attribute

The `type` attribute of the `<ol>` tag, defines the type of the list item marker:

Type	Description
type="1"	The list items will be numbered with numbers (default)
type="A"	The list items will be numbered with uppercase letters
type="a"	The list items will be numbered with lowercase letters
type="I"	The list items will be numbered with uppercase roman numbers
type="i"	The list items will be numbered with lowercase roman numbers

Numbers:

```
<ol type="1">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

HTML Description Lists

HTML also supports description lists.

A description list is a list of terms, with a description of each term.

The <dl> tag defines the description list, the <dt> tag defines the term (name), and the <dd> tag describes each term:

Example

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

A Description List

Coffee
 - black hot drink
Milk
 - white cold drink

Table Tags:

Defining an HTML Table

An HTML table is defined with the `<table>` tag.

Each table row is defined with the `<tr>` tag. A table header is defined with the `<th>` tag. By default, table headings are bold and centered. A table data/cell is defined with the `<td>` tag.

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

HTML Table - Adding a Border

If you do not specify a border for the table, it will be displayed without borders.

A border is set using the CSS `border` property:

Example

```
table, th, td {
  border: 1px solid black;
}
```

HTML Table - Cells that Span Many Columns

To make a cell span more than one column, use the `colspan` attribute:

Example

```
<table style="width:100%">
  <tr>
    <th>Name</th>
    <th colspan="2">Telephone</th>
  </tr>
  <tr>
    <td>Bill Gates</td>
    <td>55577854</td>
    <td>55577855</td>
  </tr>
</table>
```

HTML Table - Cells that Span Many Rows

To make a cell span more than one row, use the `rowspan` attribute:

Example

```
<table style="width:100%">
  <tr>
    <th>Name:</th>
    <td>Bill Gates</td>
  </tr>
  <tr>
    <th rowspan="2">Telephone:</th>
    <td>55577854</td>
  </tr>
  <tr>
    <td>55577855</td>
  </tr>
</table
```

HTML Frames:

The <frameset>.

- The <frameset> tag defines a frameset.
- The <frameset> element holds one or more <frame> elements. Each <frame> element can hold a separate document.
- The <frameset> element specifies HOW MANY columns or rows there will be in the frameset, and HOW MUCH percentage/pixels of space will occupy each of them.

Attribute	Value	Description
Cols	Pixels % *	It specifies the number and size of column spaces in the frameset. (Not Supported in HTML5)
Rows	Pixels % *	It specifies the number and size of the rows spaces in the frameset. (Not Supported in HTML5)

Tag-specific attribute

Attribute	Value	Description
frameborder	0 1	It specifies whether to display a border around the frame or not, and its default value is 1
longdsec	URL	It specifies a page which contains the long description of the content of the frame.
marginheight	Pixels	It specifies the top and bottom margins of the frame.
marginwidth	Pixels	It defines the height of the margin between frames.
name	Text	It is used to assign the name to the frame.
noresize	Noresize	It is used to prevent resizing of the frame by the user.
scrolling	yes no auto	It specifies the existence

```
<!DOCTYPE html>
<html>
<head>
   <title>Frame tag</title>
</head>
  <frameset cols="50%,50%">
```

```
    <frame  src="https://www.javatpoint.com/html-table">
    <frame  src="https://www.javatpoint.com/css-table">
   </frameset>
</html>
```

HTML iframes

HTML Iframe is used to display a nested webpage (a webpage within a webpage). The HTML <iframe> tag defines an inline frame, hence it is also called as an Inline frame.

An HTML iframe embeds another document within the current HTML document in the rectangular region.
The webpage content and iframe contents can interact with each other using JavaScript.

Iframe Syntax

An HTML iframe is defined with the <iframe> tag:

```
<iframe src="URL"></iframe>
```

Here, "src" attribute specifies the web address (URL) of the inline frame page.

Set Width and Height of iframe

You can set the width and height of iframe by using "width" and "height" attributes. By default, the attributes values are specified in pixels but you can also set them in percent. i.e. 50%, 60% etc.

```
<html>
<body>
<h2>HTML Iframes example</h2>
<p>Use the height and width attributes to specify the size of the iframe:</p>
<iframe src="https://www.javatpoint.com/" height="300" width="400"></iframe>
</body>
</html>
```

HTML-5 tags.

Ans: **add basic tags here. Give in question-4**

HTML Audio Tag

HTML audio tag is used to define sounds such as music and other audio clips. Currently there are three supported file format for HTML 5 audio tag.

1. mp3

2. wav
3. ogg

HTML Audio - How It Works

The `controls` attribute adds audio controls, like play, pause, and volume.

The `<source>` element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.

The text between the `<audio>` and `</audio>` tags will only be displayed in browsers that do not support the `<audio>` element.

HTML Audio Tag Example

Let's see the code to play mp3 file using HTML audio tag.

```
1. <audio controls>
2.   <source src="koyal.mp3" type="audio/mpeg">
3. Your browser does not support the html audio tag.
4. </audio>
```

The HTML <video> Element

To show a video in HTML, use the <video> element:

How it Works

The `controls` attribute adds video controls, like play, pause, and volume.

If height and width are not set, the page might flicker while the video loads.

The `<source>` element allows you to specify alternative video files which the browser may choose from. The browser will use the first recognized format.

The text between the `<video>` and `</video>` tags will only be displayed in browsers that do not support the `<video>` element.

Example

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
```

</video>

The HTML `<canvas>`

The HTML `<canvas>` element is used to draw graphics on a web page.

The graphic to the left is created with `<canvas>`. It shows four elements: a red rectangle, a gradient rectangle, a multicolor rectangle, and a multicolor text.

The HTML `<canvas>` element is used to draw graphics, on the fly, via JavaScript.

The `<canvas>` element is only a container for graphics. You must use JavaScript to actually draw the graphics.

The HTML `<canvas>` element is used to draw graphics, on the fly, via JavaScript.

The `<canvas>` element is only a container for graphics. You must use JavaScript to actually draw the graphics.

```
<!DOCTYPE html>
<html>
<body>
<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.moveTo(0,0);
ctx.lineTo(200,100);
ctx.stroke();
</script>
</body>
</html>
```

HTML iframe Tag

An iframe is used to display a web page within a web page.

- An HTML iframe is defined with the `<iframe>` tag:

```
<iframe src="URL"></iframe>
```

The `src` attribute specifies the URL (web address) of the inline frame page.

- Iframe - Set Height and Width

Use the height and width attributes to specify the size of the iframe.
The height and width are specified in pixels by default:
Example
<iframe src="demo_iframe.htm" height="200" width="300"></iframe>

```
<iframe src="demo_iframe.htm" name="iframe_a"></iframe>
<p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>
```

HTML <map> tag

HTML <map> tag is used with <area> tag to define a client-side image map.

An image map is consist of an image with clickable areas, where you can click on the image, and it will open to new or the provided destination.

The <map> tag can consist of more than one <area> elements which define the coordinates and type of the area.

1. <img src="image1.png" usemap="#web">
2. <map name="web">
3. <area shape="rect" coords="66,117,131,168" href="https://www.javatpoint.com/html-tutorial">

HTML 5 Forms and elements.

The <form> Element

The HTML `<form>` element defines a form that is used to collect user input:

```
<form>
.
form elements
.
</form>
```

An HTML form contains **form elements.**

Form elements are different types of input elements, like: text fields, checkboxes, radio buttons, submit buttons, and more.

The <input> Element

The `<input>` element is the most important form element.

The `<input>` element is displayed in several ways, depending on the **type** attribute.

Here are some examples:

These input elements use the **type** attribute to specify the data type.HTML4 provides following types – also in HTML 5.

Sr.No.	Type & Description
1	**Text** A free-form text field, nominally free of line breaks.
2	**Password** A free-form text field for sensitive information, nominally free of line breaks.
3	**Checkbox** A set of zero or more values from a predefined list.
4	**Radio** An enumerated value.
5	**Submit** A free form of button initiates form submission.
6	**File** An arbitrary file with a MIME type and optionally a file name.
7	**Image** A coordinate, relative to a particular image's size, with the extra semantic that it must be the last value selected and initiates form submission.
8	**Hidden** An arbitrary string that is not normally displayed to the user.
9	**Select** An enumerated value, much like the radio type.
10	**Textarea** A free-form text field, nominally with no line break restrictions.
11	**Button** A free form of button which can initiates any event related to button.

Example:-

```
<form action = "http://example.com/cgiscript.pl" method = "post">

  <p>

    <label for = "firstname">first name: </label>

    <input type = "text" id = "firstname"><br />

      <label for = "lastname">last name: </label>

    <input type = "text" id = "lastname"><br />

      <label for = "email">email: </label>

    <input type = "text" id = "email"><br>

      <input type = "radio" name = "sex" value = "male"> Male<br>

    <input type = "radio" name = "sex" value = "female"> Female<br>

    <input type = "submit" value = "send"> <input type = "reset">
```

```
</p> </form>
```

Newly Defined HTML 5 Elements in Form

Sr.No.	Type & Description
1	datetime A date and time (year, month, day, hour, minute, second, fractions of a second) encoded according to ISO 8601 with the time zone set to UTC.
2	datetime-local A date and time (year, month, day, hour, minute, second, fractions of a second) encoded according to ISO 8601, with no time zone information.
3	date A date (year, month, day) encoded according to ISO 8601.
4	month A date consisting of a year and a month encoded according to ISO 8601.
5	week A date consisting of a year and a week number encoded according to ISO 8601.
6	time A time (hour, minute, seconds, fractional seconds) encoded according to ISO 8601.
7	number It accepts only numerical value. The step attribute specifies the precision, defaulting to 1.
8	range The range type is used for input fields that should contain a value from a range of numbers.
9	email It accepts only email value. This type is used for input fields that should contain an e-mail address. If you try to submit a simple text, it forces to enter only email address in email@example.com format.
10	url It accepts only URL value. This type is used for input fields that should contain a URL address. If you try to submit a simple text, it forces to enter only URL address either in http://www.example.com format or in http://example.com format.

```
<!DOCTYPE HTML>
<html>

   <body>

      <form action = "/cgi-bin/html5.cgi" method = "get">
         Date and Time : <input type = "datetime" name = "newinput"
/>
Enter email : <input type = "email" name = "newinput" />
         <input type = "submit" value = "submit" />
      </form>
   </body>
</html>
```

CSS:

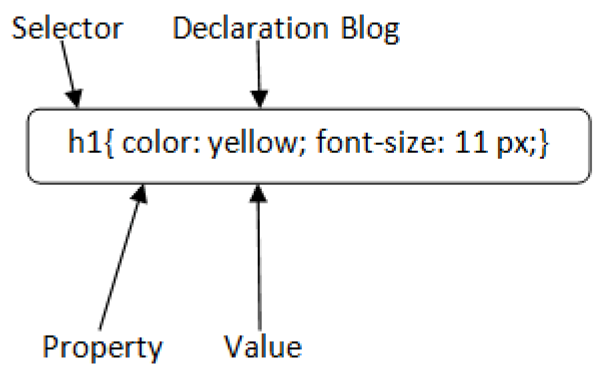
CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. It can also be used with any kind of XML documents including plain XML, SVG and XUL.

CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications.

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

CSS Syntax

A CSS rule set contains a selector and a declaration block.



Selector: Selector indicates the HTML element you want to style. It could be any tag like <h1>, <title> etc.

Declaration Block: The declaration block can contain one or more declarations separated by a semicolon. For the above example, there are two declarations:

Each declaration contains a property name and value, separated by a colon.

Property: A Property is a type of attribute of HTML element. It could be color, border etc.

Value: Values are assigned to CSS properties. In the above example, value "yellow" is a Selector{Property1: value1; Property2: value2; ....;} ssigned to color property.

CSS is added to HTML pages to format the document according to information in the style sheet. There are three ways to insert CSS in HTML documents.

1. Inline CSS
2. Internal CSS
3. External CSS

1) Inline CSS

In For example:

<p style="color:blue">Hello CSS</p>

2) Internal CSS

Internal CSS is used to apply CSS on a single document or page. It can affect all the elements of the page. It is written inside the style tag within head section of html.

For example:line CSS is used to apply CSS on a single line or element.

1. **<style>**
2. p{color:blue}
3. **</style>**

3) External CSS

External CSS is used to apply CSS on multiple pages or all pages. Here, we write all the CSS code in a css file. Its extension must be .css for example style.css.

For example: p{color:blue}

You need to link this style.css file to your html pages like this:

```
<link rel="stylesheet" type="text/css" href="style.css">
```

1. All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:
2. Inline style (inside an HTML element)
3. External and internal style sheets (in the head section)
4. Browser default

5. So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

CSS selectors:

CSS selectors are used to "find" (or select) the HTML elements you want to style.

The CSS element Selector

The element selector selects HTML elements based on the element name.

Example

Here, all <p> elements on the page will be center-aligned, with a red text color:

```
p {
  text-align: center;
  color: red;
}
```

The CSS id Selector

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element is unique within a page, so the id selector is used to select one unique element!

To select an element with a specific id, write a hash (#) character, followed by the id of the element.

```
<style>
#para1 {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>
```

The CSS class Selector

The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

```
<style>
.center {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<h1 class="center">Red and center-aligned heading</h1>
<p class="center">Red and center-aligned paragraph.</p>
```

The CSS Universal Selector

The universal selector (*) selects all HTML elements on the page.

```
<style>
* {
  text-align: center;
  color: blue;
}
</style>
</head>
<body>
<h1>Hello world!</h1>
<p>Every element on the page will be affected by the style.</p>
<p id="para1">Me too!</p>
<p>And me!</p>
```

The CSS Grouping Selector

The grouping selector selects all the HTML elements with the same style definitions.

Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

```
h1, h2, p {
  text-align: center;
  color: red;
}
```

CSS properties :

CSS Background Color

You can set the background color for HTML elements:

```
<h1 style="background-color:DodgerBlue;">Hello World</h1>
<p style="background-color:Tomato;">Lorem ipsum...</p>

body {
  background-color: lightblue;
}
```

CSS Text Color

You can set the color of text:

```
<h1 style="color:Tomato;">Hello World</h1>
<p style="color:DodgerBlue;">Lorem ipsum...</p>
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>
```

CSS Border Color

You can set the color of border

```
<h1 style="border:2px solid Tomato;">Hello World</h1>
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>
<h1 style="border:2px solid Violet;">Hello World</h1>
```

CSS Border Style

The `border-style` property specifies what kind of border to display.

The following values are allowed:

- `dotted` - Defines a dotted border
- `dashed` - Defines a dashed border
- `solid` - Defines a solid border
- `double` - Defines a double border
- `groove` - Defines a 3D grooved border. The effect depends on the border-color value
- `ridge` - Defines a 3D ridged border. The effect depends on the border-color value

CSS background-image

The `background-image` property specifies an image to use as the background of an element.

By default, the image is repeated so it covers the entire element.

```
body {
  background-image: url("paper.gif");
}
```

CSS Margins

The CSS `margin` properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`

```
p {
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
}
```

CSS Padding

The CSS `padding` properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

CSS has properties for specifying the padding for each side of an element:

- `padding-top`
- `padding-right`
- `padding-bottom`

- `padding-left`

```
div {
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
```

The CSS Box Model

All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

Explanation of the different parts:

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

```
div {
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
```

Styling Links

Links can be styled with any CSS property (e.g. color, font-family, background, etc.).

Example

```
a {
  color: hotpink;
}
```

In addition, links can be styled differently depending on what **state** they are in.

The four links states are:

- `a:link` - a normal, unvisited link

- `a:visited` - a link the user has visited
- `a:hover` - a link when the user mouses over it
- `a:active` - a link the moment it is clicked

```css
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}

/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
```

CSS Layout - The Display Property

The `display` property specifies if/how an element is displayed.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is `block` or `inline`.

Block-level Elements

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

The <div> element is a block-level element.

Examples of block-level elements:

- <div>

- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

Display: none;

display: none; is commonly used with JavaScript to hide and show elements without deleting and recreating them. Take a look at our last example on this page if you want to know how this can be achieved.

The <script> element uses display: none; as default.

```
span {
  display: block;
}
```

CSS properties for Text Formatting:

Text Color

The color property is used to set the color of the text. The color is specified by:

a color name - like "red"

a HEX value - like "#ff0000"

an RGB value - like "rgb(255,0,0)"

The default text color for a page is defined in the body selector.

Example

```
body {
color: blue;
}
h1 {
 color: green;
}
```

Text Alignment

The `text-align` property is used to set the horizontal alignment of a text.

A text can be left or right aligned, centered, or justified.

```
h1 {
  text-align: center;
}

h2 {
  text-align: left;
}

h3 {
  text-align: right;
}
```

Text Direction

The direction and unicode-bidi properties can be used to change the text direction of an element:

Example

```
p {
  direction: rtl;
  unicode-bidi: bidi-override;
}
```

Text Decoration

The text-decoration property is used to set or remove decorations from text.

```
h1 {
  text-decoration: overline;
}
h2 {
  text-decoration: line-through;
}
h3 {
  text-decoration: underline;
}
```

Text Indentation

The text-indent property is used to specify the indentation of the first line of a text:

Example

```
p {
  text-indent: 50px;
}
```

Letter Spacing

The letter-spacing property is used to specify the space between the characters in a text.
The following example demonstrates how to increase or decrease the space between characters:

Example

```
h1 {
  letter-spacing: 3px;
}

h2 {
  letter-spacing: -3px;
}
```

CSS Font Style

The font-style property is mostly used to specify italic text.

This property has three values:

- normal - The text is shown normally
- italic - The text is shown in italics
- oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

```
p.normal {
  font-style: normal;
}
p.italic {
  font-style: italic;
}
p.oblique {
  font-style: oblique;
}
```

Font Weight

The font-weight property specifies the weight of a font:

Example

```
p.normal {
  font-weight: normal;
}
p.thick {
  font-weight: bold;
}
```

Font Size

The font-size property sets the size of the text.

Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs.

Example

```
h1 {
  font-size: 40px;
}

h2 {
  font-size: 30px;
}

p {
  font-size: 14px;
}
```

CSS TABLE formatting properties:

Table Borders

To specify table borders in CSS, use the border property.

The example below specifies a black border for <table>, <th>, and <td> elements:

```
table, th, td {
  border: 1px solid black;
}
```

Collapse Table Borders

The border-collapse property sets whether the table borders should be collapsed into a single border:

```
table {
  border-collapse: collapse;
}

table, th, td {
  border: 1px solid black;
}
```

Table Width and Height

Width and height of a table are defined by the width and height properties.

```
table {
  width: 100%;
}

th {
  height: 50px;
}
```

Horizontal Alignment

The text-align property sets the horizontal alignment (like left, right, or center) of the content in <th> or <td>.

By default, the content of <th> elements are center-aligned and the content of <td> elements are left-aligned.

```
th {
  text-align: left;
}
```

Vertical Alignment

The vertical-align property sets the vertical alignment (like top, bottom, or middle) of the content in <th> or <td>.

By default, the vertical alignment of the content in a table is middle (for both <th> and <td> elements).

```
td {
  height: 50px;
  vertical-align: bottom;
}

th {
  background-color: #4CAF50;
  color: white;
}
```

Short questions

1.Explain about web browser?
2.Explain about browser functionality?
3.Write about HTML tags?
4.List out CSS properties?

Long questions

1.Explain about HTML5 and its features?
2.Discuss about list in HTML?
3.Write about HTML frames?
4.What is CSS and how they linked with HTML?

UNIT-2

JavaScript

JavaScript is a lightweight, interpreted **programming** language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. **JavaScript** is very easy to implement because it is integrated with HTML. It is open and cross-platform..It is a client-Side scripting language.

Advantages of Javascript:

- Javascript is the most popular **programming language** in the world and that makes it a programmer's great choice. Once you learnt Javascript, it helps you developing great front-end as well as back-end softwares using different Javascript based frameworks like jQuery, Node.JS etc.

- Javascript is everywhere, it comes installed on every modern web browser and so to learn Javascript you really do not need any special environment setup. For example Chrome, Mozilla Firefox , Safari and every browser you know as of today, supports Javascript.
- Javascript helps you create really beautiful and crazy fast websites. You can develop your website with a console like look and feel and give your users the best Graphical User Experience.
- JavaScript usage has now extended to mobile app development, desktop app development, and game development. This opens many opportunities for you as Javascript Programmer.
- Due to high demand, there is tons of job growth and high pay for those who know JavaScript. You can navigate over to different job sites to see what having JavaScript skills looks like in the job market.
- Great thing about Javascript is that you will find tons of frameworks and Libraries already developed which can be used directly in your software development to reduce your time to market.

Sample Code:-

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

There are many useful **Javascript frameworks** and libraries available:

- Angular,React,jQuery,Vue.js,Ext.js,Ember.js,Meteor,

Mithril,Node.js Limitations of JavaScript

We cannot treat JavaScript as a full-fledged programming language. It lacks the following important features −

- Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
- JavaScript cannot be used for networking applications because there is no such support available.
- JavaScript doesn't have any multi-threading or multiprocessor capabilities.

Once again, JavaScript is a lightweight, interpreted programming language that allows you to build interactivity into otherwise static HTML pages.

JavaScript Development Tools

- **Microsoft FrontPage** − Microsoft has developed a popular HTML editor called FrontPage. FrontPage also provides web developers with a number of JavaScript tools to assist in the creation of interactive websites.

- **Macromedia Dreamweaver MX** − Macromedia Dreamweaver MX is a very popular HTML and JavaScript editor in the professional web development crowd. It provides several handy prebuilt JavaScript components, integrates well with databases, and conforms to new standards such as XHTML and XML.
- **Macromedia HomeSite 5** − HomeSite 5 is a well-liked HTML and JavaScript editor from Macromedia that can be used to manage personal websites effectively.

Java Script Coding Syntax

JavaScript can be implemented using JavaScript statements that are placed within the **<script>... </script>** HTML tags in a web page.

You can place the **<script>** tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the **<head>** tags.

The <script> tag alerts the browser program to start interpreting all the text between these tags as a script. A simple syntax of your JavaScript will appear as follows.

```
<script ...>
  JavaScript code
</script>
```

The script tag takes two important attributes −

- **Language** − This attribute specifies what scripting language you are using. Typically, its value will be javascript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.
- **Type** − This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".

So your JavaScript segment will look like −

```
<script language = "javascript" type = "text/javascript">
   JavaScript code
</script>
```

This function can be used to write text, HTML, or both. Take a look at the following code.

Live Demo

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

This code will produce the following result −

```
Hello World!
```

Semicolons are Optional

Simple statements in JavaScript are generally followed by a semicolon character, just as they are in C, C++, and Java. JavaScript, however, allows you to omit this semicolon if each of your statements are placed on a separate line. For example, the following code could be written without semicolons.

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

But when formatted in a single line as follows, you must use semicolons −

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

Note − It is a good programming practice to use semicolons.

- 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.

JavaScript supports both C-style and C++-style comments, Thus −

- 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 Datatypes and variables:

One of the most fundamental characteristics of a programming language is the set of data types it supports. These are the type of values that can be represented and manipulated in a programming language.

JavaScript allows you to work with three primitive data types −

- **Numbers,** eg. 123, 120.50 etc.
- **Strings** of text e.g. "This text string" etc.
- **Boolean** e.g. true or false.

JavaScript also defines two trivial data types, **null** and **undefined,** each of which defines only a single value. In addition to these primitive data types, JavaScript supports a composite data type known as **object**. We will cover objects in detail in a separate chapter.

JavaScript Variables

Like many other programming languages, JavaScript has variables. Variables can be thought of as named containers. You can place data into these containers and then refer to the data simply by naming the container.

Before you use a variable in a JavaScript program, you must declare it. Variables are declared with the var keyword as follows.

Ex:-

```
<script type = "text/javascript">

  <!--

    var money;

    var name;

  //-->

</script>
```

JavaScript Global Variable

A JavaScript global variable is declared outside the function or declared with window object. It can be accessed from any function.

```
<script>

var value=50;//global variable

function a(){
```

```
alert(value);
}
function b(){
alert(value);
}
</script>
```

JavaScript non-primitive data types

The non-primitive data types are as follows:

Data Type	Description
Object	represents instance through which we can access members
Array	represents group of similar values
RegExp	**represents** regular expression

Operators in JavaScript

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on the operands. The following operators are known as JavaScript arithmetic operators.

JavaScript operators are symbols that are used to perform operations on operands. For example:

There are following types of operators in JavaScript.

1. Arithmetic Operators
2. Comparison (Relational) Operators
3. Bitwise Operators
4. Logical Operators
5. Assignment Operators
6. Special Operators

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on the operands. The following operators are known as JavaScript arithmetic operators.

Operator	Description	Example
+	Addition	10+20 = 30
-	Subtraction	20-10 = 10
*	Multiplication	10*20 = 200
/	Division	20/10 = 2
%	Modulus (Remainder)	20%10 = 0

++	Increment	var a=10; a++; Now a = 11
--	Decrement	var a=10; a--; Now a = 9

JavaScript Comparison Operators

The JavaScript comparison operator compares the two operands. The comparison operators are as follows:

Operator	Description	Example
==	Is equal to	10==20 = false
===	Identical (equal and of same type)	10==20 = false
!=	Not equal to	10!=20 = true
!==	Not Identical	20!==20 = false
>	Greater than	20>10 = true
>=	Greater than or equal to	20>=10 = true
<	Less than	20<10 = false
<=	Less than or equal to	20<=10 = false

JavaScript Bitwise Operators

The bitwise operators perform bitwise operations on operands. The bitwise operators are as follows:

Operator	Description	Example
&	Bitwise AND	(10==20 & 20==33) = false
|	Bitwise OR	(10==20 | 20==33) = false
^	Bitwise XOR	(10==20 ^ 20==33) = false
~	Bitwise NOT	(~10) = -10
<<	Bitwise Left Shift	(10<<2) = 40
>>	Bitwise Right Shift	(10>>2) = 2
>>>	Bitwise Right Shift with Zero	(10>>>2) = 2

JavaScript Logical Operators

The following operators are known as JavaScript logical operators.

Operator	Description	Example

&&	Logical AND	(10==20 && 20==33) = false
||	Logical OR	(10==20 || 20==33) = false
!	Logical Not	!(10==20) = true

JavaScript Assignment Operators

The following operators are known as JavaScript assignment operators.

Operator	Description	Example
=	Assign	10+10 = 20
+=	Add and assign	var a=10; a+=20; Now a = 30
-=	Subtract and assign	var a=20; a-=10; Now a = 10
*=	Multiply and assign	var a=10; a*=20; Now a = 200
/=	Divide and assign	var a=10; a/=2; Now a = 5
%=	Modulus and assign	var a=10; a%=2; Now a = 0

JavaScript Special Operators

The following operators are known as JavaScript special operators.

Operator	Description
(?:)	Conditional Operator returns value based on the condition. It is likeif-else.
,	Comma Operator allows multiple expressions to be evaluated assingle statement.
delete	Delete Operator deletes a property from the object.
in	In Operator checks if object has the given property
instanceof	checks if the object is an instance of given type
new	creates an instance (object)
typeof	checks the type of object.
void	it discards the expression's return value.
yield	checks what is returned in a generator by the generator's iterator.

JavaScript Decision making Statements

The **JavaScript if-else statement** is used *to execute the code whether condition is true or false*. There are three forms of if statement in JavaScript.

1. If Statement
2. If else statement
3. if else if statement

JavaScript If statement

It evaluates the content only if expression is true. The signature of JavaScript if statement is given below.

```
if(expression){
//content to be evaluated
}
```

JavaScript If...else Statement

It evaluates the content whether condition is true of false. The syntax of JavaScript if-else statement is given below.

```
if(expression){
//content to be evaluated if condition is true
}
else{
//content to be evaluated if condition is false
}
```

JavaScript If...else if statement

It evaluates the content only if expression is true from several expressions. The signature of JavaScript if else if statement is given below.

```
if(expression1){
//content to be evaluated if expression1 is true
}
else if(expression2){
//content to be evaluated if expression2 is true
}
else if(expression3){
//content to be evaluated if expression3 is true
}
else{
//content to be evaluated if no expression is true
}
```

Example:-

```
<script>
var a=20;
if(a==10){
document.write("a is equal to 10");
}
else if(a==15){
document.write("a is equal to 15");
}
else if(a==20){
document.write("a is equal to 20");
}
else{
document.write("a is not equal to 10, 15 or 20");
}
</script>
```

JavaScript Switch

The JavaScript switch statement is used to execute one code from multiple expressions. It is just like else if statement that we have learned in previous page. But it is convenient than if..else..if because it can be used with numbers, characters etc.

The signature of JavaScript switch statement is given below.

```
switch(expression){
case value1:
 code to be executed;
 break;
case value2:
 code to be executed;
 break;
......

default:
 code to be executed if above values are not matched;
}
```

Example

```
<script>
var grade='B';
var result;
switch(grade){
case 'A':
result="A Grade";
break;
case 'B':
result="B Grade";
break;
case 'C':
result="C Grade";
break;
default:
result="No Grade";
}
document.write(result);
</script>
```

6. Discuss about JavaScript Loops

The **JavaScript loops** are used *to iterate the piece of code* using for, while, do while or for-in loops. It makes the code compact. It is mostly used in array.

There are four types of loops in JavaScript.

1. for loop
2. while loop
3. do-while loop
4. for-in loop

1) JavaScript For loop

The JavaScript for loop iterates the elements for the fixed number of times. It should be used if number of iteration is known. The syntax of for loop is given below.

```
for (initialization; condition; increment)

{     code to be executed

}
```

1. **Ex:-**
2. **<script>**
3. for (i=1; i**<**=5; i++)
4. {

```
document.write(i + "<br/>")
}
</script>
```

2) JavaScript while loop

The JavaScript while loop iterates the elements for the infinite number of times. It should be used if number of iteration is not known. The syntax of while loop is given below.

```
while (condition)
{
   code to be executed
}
```

Let’s see the simple example of while loop in javascript.

```
<script>
var i=11;
while (i<=15)
{
document.write(i + "<br/>");
i++;
}
</script>
```

3. 3) JavaScript do while loop

The JavaScript do while loop iterates the elements for the infinite number of times like while loop. But, code is executed at least once whether condition is true or false. The syntax of do while loop is given below.

```
do{
   code to be executed
}while (condition);
```

Let’s see the simple example of do while loop in javascript.

Ex:-

```
<script>
var i=21;
do{
document.write(i + "<br/>");
i++;
}while (i<=25);
</script>
```

for…in loop is used to loop through an object's properties. As we have not discussed Objects yet, you may not feel comfortable with this loop. But once you understand how objects behave in JavaScript, you will find this loop very useful.

Syntax

The syntax of ‘for..in’ loop is −

```
for (variablename in object) {
  statement or block to execute
}
```

In each iteration, one property from object is assigned to variablename and this loop continues till all the properties of the object are exhausted.

Example

8. Discuss about creation of arrays in JavaScript

JavaScript array is an object that represents a collection of similar type of elements.

There are 3 ways to construct array in JavaScript

- By array literal

- By creating instance of Array directly (using new keyword)
- By using an Array constructor (using new keyword)

By Array Literal Syntax:- var arrayname=[value1,value2.....valueN];

```
<script>
var emp=["Sonoo","Vimal","Ratan"];
for (i=0;i<emp.length;i++){
document.write(emp[i] + "<br/>");
}
</script>
```

By Using new Keyword

Syntax :- var arrayname=new Array();

```
<script>
var i;
var emp = new Array();
emp[0] = "Arun";
emp[1] = "Varun";
emp[2] = "John";

for (i=0;i<emp.length;i++){
document.write(emp[i] + "<br>");
}
</script>
```

JavaScript array constructor (new keyword)

to create instance of array by passing arguments in constructor so that we don't have to provide value explicitly.

The example of creating object by array constructor is given below.

```
<script>
var emp=new Array("Jai","Vijay","Smith");
for (i=0;i<emp.length;i++){
document.write(emp[i] + "<br>");
}
</script>
```

JavaScript Array Methods

Let's see the list of JavaScript array methods with their description.

Methods	Description
concat()	It returns a new array object that contains two or more merged arrays.
every()	It determines whether all the elements of an array are satisfying theprovided function conditions.
fill()	It fills elements into an array with static values.
from()	It creates a new array carrying the exact copy of another array element.
filter()	It returns the new array containing the elements that pass the providedfunction conditions.
find()	It returns the value of the first element in the given array that satisfiesthe specified condition.
findIndex()	It returns the index value of the first element in the given array thatsatisfies the specified condition.

forEach()	It invokes the provided function once for each element of an array.
indexOf()	It searches the specified element in the given array and returns the indexof the first match.
pop()	It removes and returns the last element of an array.
push()	It adds one or more elements to the end of an array.
reverse()	It reverses the elements of given array.
shift()	It removes and returns the first element of an array.
slice()	It returns a new array containing the copy of the part of the given array.
sort()	It returns the element of the given array in a sorted order.

Ex:-
```
var arr=[2,4,1,8,5];
var result=arr.sort(function compare(a,b)
{
 return b-a;
});
document.writeln(result);
</script>
```

9. Explain about JavaScript Functions

JavaScript functions are used to perform operations. We can call JavaScript function many times to reuse the code.

Advantage of JavaScript function

There are mainly two advantages of JavaScript functions.

- Code reusability: We can call a function several times so it save coding.
- Less coding: It makes our program compact. We don't need to write many lines of code each time to perform a common task.

Syntax:
```
function functionName([arg1, arg2, ...argN]){
 //code to be executed
}
```

Example
```
<html>
<body>
<script>
function getcube(number){
alert(number*number*number);
}
</script>
<form>
<input type="button" value="click" onclick="getcube(4)"/>
</form>
</body>
</html>
```

JavaScript Objects (in exam write any 4 or 5 objects)

A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc.

JavaScript is an object-based language. Everything is an object in JavaScript.

A) The JavaScript string is an object that represents a sequence of characters.

There are 2 ways to create string in JavaScript

- By string literal
- By string object (using new keyword)

By string literal

The string literal is created using double quotes. The syntax of creating string using string literal is given below:

Ex:-

```
<script>
var str="This is string literal";
document.write(str);
</script>
```

By string object (using new keyword)

The syntax of creating string object using new keyword is given below:

```
var stringname=new String("string literal");
```

Ex:-

```
<script>
var stringname=new String("hello javascript string");
document.write(stringname);
</script>
```

Methods	Description
charAt()	It provides the char value present at the specified index.
charCodeAt()	It provides the Unicode value of a character present at thespecified index.
concat()	It provides a combination of two or more strings.
indexOf()	It provides the position of a char value present in the givenstring.
lastIndexOf()	It provides the position of a char value present in the givenstring by searching a character from the last position.
search()	It searches a specified regular expression in a given string andreturns its position if a match occurs.
match()	It searches a specified regular expression in a given string andreturns that regular expression if a match occurs.
replace()	It replaces a given string with the specified replacement.
substr()	It is used to fetch the part of the given string on the basis of thespecified starting position and length.
substring()	It is used to fetch the part of the given string on the basis of thespecified index.
slice()	It is used to fetch the part of the given string. It allows us toassign positive as well negative index.
toLowerCase()	It converts the given string into lowercase letter.
toLocaleLowerCase()	It converts the given string into lowercase letter on the basis of

	host?s current locale.
toUpperCase()	It converts the given string into uppercase letter.

B) JavaScript Date Object

The JavaScript date object can be used to get year, month and day. You can display a timer on the webpage by the help of JavaScript date object.

You can use different Date constructors to create date object. It provides methods to get and set day, month, year, hour, minute and seconds.

Constructor

You can use 4 variant of Date constructor to create date object.
Date()
Date(milliseconds)
Date(dateString)
Date(year, month, day, hours, minutes, seconds, milliseconds)

JavaScript Date Methods

Let's see the list of JavaScript date methods with their description.

Methods	Description
getDate()	It returns the integer value between 1 and 31 that represents theday for the specified date on the basis of local time.
getDay()	It returns the integer value between 0 and 6 that represents theday of the week on the basis of local time.
getFullYears()	It returns the integer value that represents the year on the basis oflocal time.
getHours()	It returns the integer value between 0 and 23 that represents thehours on the basis of local time.
getMinutes()	It returns the integer value between 0 and 59 that represents theminutes on the basis of local time.
getMonth()	It returns the integer value between 0 and 11 that represents themonth on the basis of local time.
getSeconds()	It returns the integer value between 0 and 60 that represents theseconds on the basis of local time.
setDate()	It sets the day value for the specified date on the basis of localtime.
setDay()	It sets the particular day of the week on the basis of local time.
setFullYears()	It sets the year value for the specified date on the basis of localtime.
setHours()	It sets the hour value for the specified date on the basis of localtime.

setMinutes()	It sets the minute value for the specified date on the basis of localtime.
setMonth()	It sets the month value for the specified date on the basis of localtime.
setSeconds()	It sets the second value for the specified date on the basis of localtime.
setUTCMonth()	It sets the month value for the specified date on the basis ofuniversal time.
setUTCSeconds()	It sets the second value for the specified date on the basis ofuniversal time.
toDateString()	It returns the date portion of a Date object.
toISOString()	It returns the date in the form ISO format string.
toString()	It returns the date in the form of string.
toTimeString()	It returns the time portion of a Date object.
valueOf()	It returns the primitive value of a Date object.

Example:-

```
<script>
var date=new Date();
var day=date.getDate();
var month=date.getMonth()+1;
var year=date.getFullYear();
document.write("<br>Date is: "+day+"/"+month+"/"+year);
</script>
```

To Display digital Timer in a Browser

```
Current Time: <span id="txt"></span>
<script>
window.onload=function(){getTime();}
function getTime(){
var today=new Date();
var h=today.getHours();
var m=today.getMinutes();
var s=today.getSeconds();
// add a zero in front of numbers<10
m=checkTime(m);
s=checkTime(s);
document.getElementById('txt').innerHTML=h+":"+m+":"+s;
setTimeout(function(){getTime()},1000);
}
//setInterval("getTime()",1000);//another way
function checkTime(i){
if (i<10){
  i="0" + i;
 }
return i;
}
</script>
```

c)JavaScript Math object

The JavaScript math object provides several constants and methods to perform mathematical operation. Unlike date object, it doesn't have constructors.

JavaScript Math Methods
Let's see the list of JavaScript Math methods with description.

Methods	**Description**
abs()	It returns the absolute value of the given number.

acos()	It returns the arccosine of the given number in radians.
asin()	It returns the arcsine of the given number in radians.
atan()	It returns the arc-tangent of the given number in radians.
cbrt()	It returns the cube root of the given number.
ceil()	It returns a smallest integer value, greater than or equal to the given number.
cos()	It returns the cosine of the given number.
cosh()	It returns the hyperbolic cosine of the given number.
exp()	It returns the exponential form of the given number.
floor()	It returns largest integer value, lower than or equal to the given number.
hypot()	It returns square root of sum of the squares of given numbers.
log()	It returns natural logarithm of a number.
max()	It returns maximum value of the given numbers.
min()	It returns minimum value of the given numbers.
pow()	It returns value of base to the power of exponent.

Example:

```
Random Number is: <span id="p2"></span>
<script>
document.getElementById('p2').innerHTML=Math.random();
</script>
```

D) Number Object

The **JavaScript number** object *enables you to represent a numeric value*. It may be integer or floating-point. JavaScript number object follows IEEE standard to represent the floating-point numbers.

JavaScript Number Methods

Let's see the list of JavaScript number methods with their description.

Methods	**Description**
isFinite()	It determines whether the given value is a finite number.
isInteger()	It determines whether the given value is an integer.
parseFloat()	It converts the given string into a floating point number.
parseInt()	It converts the given string into an integer number.
toExponential()	It returns the string that represents exponential notation of the given number.
toFixed()	It returns the string that represents a number with exact digits after a decimal point.
toPrecision()	It returns the string representing a number of specified precision.
toString()	It returns the given number in the form of string.

E) Boolean object

JavaScript Boolean is an object that represents value in two states: true or false. You can create the JavaScript Boolean object by Boolean() constructor as given below.

Boolean b=new Boolean(value);

JavaScript Boolean

JavaScript Boolean is an object that represents value in two states: true or false. You can create the JavaScript Boolean object by Boolean() constructor as given below.

Boolean b=new Boolean(value);

The default value of JavaScript Boolean object is false.

JavaScript Boolean Example

```
<script>
document.write(10<20);//true
document.write(10<5);//false
</script>
```

JavaScript Boolean Methods

Method	Description
toSource()	returns the source of Boolean object as a string.
toString()	converts Boolean into String.
valueOf()	converts other type into Boolean.

f) Window Object

Window Object

The window object represents a window in browser. An object of window is created automatically by the browser.Window is the object of browser, it is not the object of javascript. The javascript objects are string, array, date etc.

Method	Description
alert()	displays the alert box containing message with ok button.
confirm()	displays the confirm dialog box containing message with okand cancel button.
prompt()	displays a dialog box to get input from the user.
open()	opens the new window.
close()	closes the current window.
setTimeout()	performs action after specified time like calling function, evaluating expressions etc.

Java Script events:

The change in the state of an object is known as an Event. In html, there are various events which represents that some activity is performed by the user or by the browser. When javascript code is included in HTML, js react over these events and allow the execution. This process of reacting over the events is called Event Handling. Thus, js handles the HTML events via Event Handlers.

Mouse events:

Event Performed	Event Handler	Description
click	Onclick	When mouse click on an element
mouseover	onmouseover	When the cursor of the mouse comes over the element
mouseout	onmouseout	When the cursor of the mouse leaves an element
mousedown	onmousedown	When the mouse button is pressed over the element

mouseup	onmouseup	When the mouse button is released over the element
mousemove	onmousemove	When the mouse movement takes place.

Keyboard events:

Event Performed	Event Handler	Description
Keydown & Keyup	onkeydown & onkeyup	When the user press and then releasethe key

Form events:

Event Performed	Event Handler	Description
focus	Onfocus	When the user focuses on an element
submit	onsubmit	When the user submits the form
blur	Onblur	When the focus is away from a form element
change	onchange	When the user modifies or changes the value of aform element

Window/Document events

Event Performed	Event Handler	Description
load	Onload	When the browser finishes the loading of the page
unload	onunload	When the visitor leaves the current webpage, the browser unloads it
resize	Onresize	When the visitor resizes the window of the browser

Example:-

```
<html>
<head> Javascript Events </head>
<body>
<script language="Javascript" type="text/Javascript">
   <!--
   function clickevent()
   {
      document.write("This is JavaTpoint");
   }
   //-->
</script>
<form>
<input type="button" onclick="clickevent()" value="Who's this?"/>
</form>
```

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

Example-2 to find simple interest

```
<html>

<form name="form1">
Enter Principle:<input type="text" name="pr"/><br>
Enter Time:<input type="text" name="tm"/><br>
Enter Rate:<input type="text" name="rate"/>    <br>
SI Value:<input type="text" name="simp"/>
<input type="button" onclick="printvalue()" value="Find SI"/>
</form>

<script type="text/javascript">
function printvalue(){
var p=parseFloat(document.form1.pr.value);
var t=parseFloat(document.form1.tm.value);
var r=parseFloat(document.form1.rate.value);
var si=(p*t*r)/100.0;
document.form1.simp.value=si;
}
</script>
</html>
```

Example-3

```
<html>
<head> Javascript Events</head>
<body>
<h2> Enter something here</h2>
<input type="text" id="input1" onfocus="focusevent()"/>
<script>
<!--
   function focusevent()
   {
      document.getElementById("input1").style.background=" aqua";
   }
//-->
</script>
</body>
</html>
```

JQuery

Features of JQuery and give a sample program .

jQuery is a small and lightweight JavaScript library created by John Resig in 2006 at google with a nice motto: **Write less, do more**. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code. Here is the list of important core features supported by jQuery −

- **DOM manipulation** − The jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called **Sizzle**.
- **Event handling** − The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- **AJAX Support** − The jQuery helps you a lot to develop a responsive and featurerich site using AJAX technology.
- **Animations** − The jQuery comes with plenty of built-in animation effects which you can use in your websites.
- **Lightweight** − The jQuery is very lightweight library - about 19KB in size (Minified and gzipped).
- **Cross Browser Support** − The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+
- **Latest Technology** − The jQuery supports CSS3 selectors and basic XPath syntax.

to use jQuery

- Local Installation − You can download jQuery library on your local machine and include it in your HTML code.

- Go to the https://jquery.com/download/ to download the latest version available.
- Now put downloaded **jquery-2.1.3.min.js file** in a directory of your website, e.g. /jquery.

Many of the biggest companies on the web use jQuery.
Some of these companies are:

- Microsoft
- Google
- IBM
- Netflix

JQuery Sample Code:

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript" src = "/jquery/jquery-2.1.3.min.js">
    </script>

    <script type = "text/javascript">
      $(document).ready(function() {
        document.write("Hello, World!");
      });
    </script>
  </head>

  <body>
    <h1>Hello</h1>
  </body>
</html>
```

```
<html>
<head>
 <title>First jQuery Example</title>
 <script type="text/javascript"
  src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
(directly call from google library if we have online connection)
 </script>
 <script type="text/javascript" language="javascript">
 $(document).ready(function() {
 $("p").css("background-color", "cyan");
 });
 </script>
 </head>
<body>
<p>The first paragraph is selected.</p>
<p>The second paragraph is selected.</p>
<p>The third paragraph is selected.</p>
</body>
</html>
```

JQuery Syntax

jQuery Syntax
The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Basic syntax is: $(selector).action()

- A $ sign to define/access jQuery
- A (selector) to "query (or find)" HTML elements
- A jQuery action() to be performed on the element(s)

Examples:
$(this).hide() - hides the current element.
$("p").hide() - hides all <p> elements.
$(".test").hide() - hides all elements with class="test".
$("#test").hide() - hides the element with id="test"

$(document).ready() and $()

The code inserted between $(document).ready() is **executed only once when page is ready for JavaScript code to execute.**

$(document).ready() and $()
The code inserted between $(document).ready() is executed only once when page is ready for JavaScript code to execute.

In place of $(document).ready(), you can use shorthand notation $() only.

```
$(document).ready(
function() {
$("p").css("color", "red");
}
);
```

The above code is equivalent to this code.

```
$(
function() {
$("p").css("color", "red");
}
);
```

How to use Selectors

The jQuery selectors can be used single or with the combination of other selectors. They are required at every steps while using jQuery. They are used to select the exact element that you want from your HTML document.

S.No.	Selector	Description
1)	Name:	It selects all elements that match with the givenelement name.
2)	#ID:	It selects a single element that matches with the givenid.
3)	.Class:	It selects all elements that matches with the givenclass.
4)	Universal(*)	It selects all elements available in a DOM.
5)	Multiple Elements A,B,C	It selects the combined results of all the specifiedselectors A,B and

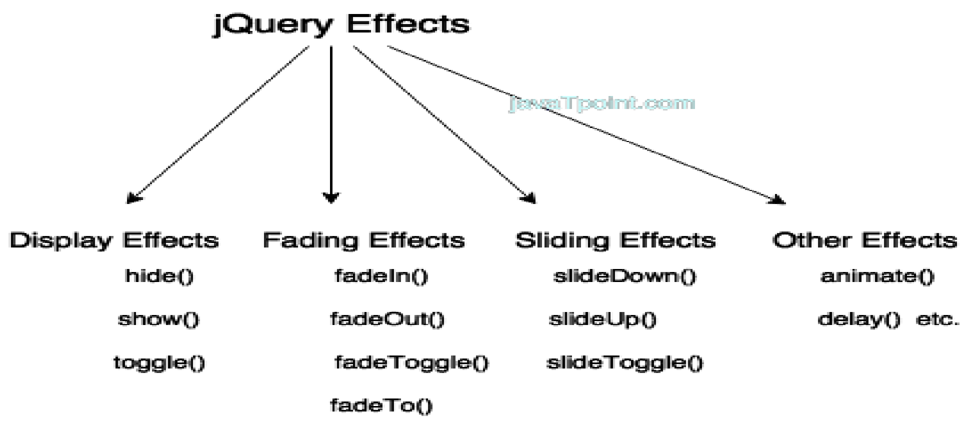
Different jQuery Selectors

Selector	Example	Description
*	$("*")	It is used to select all elements.
#id	$("#firstname")	It will select the element with id="firstname"
.class	$(".primary")	It will select all elements with class="primary"

class,.class	$(".primary,.secondary")	It will select all elements with the class"primary" or "secondary"
element	$("p")	It will select all p elements.
el1,el2,el3	$("h1,div,p")	It will select all h1, div, and p elements.
:first	$("p:first")	This will select the first p element
:last	$("p:last")	This will select he last p element
:even	$("tr:even")	This will select all even tr elements
:odd	$("tr:odd")	This will select all odd tr elements

jQuery Effects

jQuery enables us to add effects on a web page. jQuery effects can be categorized into fading, sliding, hiding/showing and animation effects



jQuery provides many methods for effects on a web page. A complete list of jQuery effect methods are given below:

No.	Method	Description
1)	animate()	performs animation.
2	clearQueue()	It is used to remove all remaining queued functions from theselected elements.
3)	delay()	sets delay execution for all the queued functions on theselected elements.
4	dequeue()	It is used to remove the next function from the queue, andthen execute the function.
5)	fadein()	shows the matched elements by fading it to opaque. In otherwords, it fades in the selected elements.
6)	fadeout()	shows the matched elements by fading it to transparent. Inother words, it fades out the selected elements.
7)	fadeto()	adjusts opacity for the matched element. In other words, it

		fades in/out the selected elements.
8)	fadetoggle()	shows or hides the matched element. In other words, togglesbetween the fadeIn() and fadeOut() methods.
9)	finish()	It stops, removes and complete all queued animation for theselected elements.
10)	hide()	hides the matched or selected elements.
11)	queue()	shows or manipulates the queue of methods i.e. to be executedon the selected elements.
12)	show()	displays or shows the selected elements.
13)	slidedown()	shows the matched elements with slide.
14)	slidetoggle()	shows or hides the matched elements with slide. In otherwords, it is used to toggle between the slideUp() and slideDown() methods.
15)	slideup()	hides the matched elements with slide.
16)	stop()	stops the animation which is running on the matched elements.
17)	toggle()	shows or hides the matched elements.

Ex:- to hide and show effects.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#hide").click(function(){
    $("p").hide();
  });
  $("#show").click(function(){
    $("p").show();
  });
});
</script>
</head>
<body>

<p>If you click on the "Hide" button, I will disappear.</p>

<button id="hide">Hide</button>
<button id="show">Show</button>

</body>
</html>
```

Ex:- Toggle Effect

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#flip").click(function(){
        $("#panel").slideToggle("slow");
    });
});
```

```
</script>
 <style>
#panel, #flip {
   padding: 5px;
   text-align: center;
   background-color: #00FFFF;
   border: solid 1px #c3c3c3;
}
#panel {
   padding: 50px;
   display:none;
}
</style>
</head>
<body>
<div id="flip">Click to slide toggle panel</div>
<div id="panel">Hello javatpoint.com!
It is the best tutorial website to learn jQuery and other languages.</div>
</body>
</html>
```

// animate effect

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("div").animate({left: '450px'});
   });
});
</script>
</head>
<body>
<button>Start Animation</button>
<p>A simple animation example:</p>
<div    style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
</body>
</html>
```

jQuery html()

jQuery html() method is used to change the entire content of the selected elements. It replaces the selected element content with new contents.

- $(selector).html() It is used to return content.
- $(selector).html(content) It is used to set content.
- $(selector).html(function (index, currentcontent))

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("p").html("Hello <b>Javatpoint.com</b>");
   });
});
</script>
</head>
<body>
<button>Click here to change the content of all p elements</button>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
</body>
```
- </html>

Short questions

1.Explain about Javascript advantages?
2.Write about Javascript datatypes and variables?
3.Explain about Javascript functions?
4.Write about Javascript objects?

Long questions

1.Discuss about Javascript loops?
2.Write about Operators in Javascript?
3.Describe about Javascript events?
4.Explain about features of JQuery and give a sample program?

UNIT-3

Ajax:

AJAX stands for **A**synchronous **Ja**vaScript and **X**ML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script.

- Ajax uses XHTML for content, CSS for presentation, along with Document Object Modeland JavaScript for dynamic content display.
- Conventional web applications transmit information to and from the sever using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.
- With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server.
- XML is commonly used as the format for receiving server data, although any format, including plain text, can be used.
- AJAX is a web browser technology independent of web server software.

- A user can continue to use the application while the client program requests information from the server in the background.
- Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger.
- Data-driven as opposed to page-driven.

Rich Internet Application Technology

AJAX is the most viable Rich Internet Application (RIA) technology so far. It is getting tremendous industry momentum and several tool kit and frameworks are emerging. But at the same time, AJAX has browser incompatibility and it is supported by JavaScript, which is hard to maintain and debug.

AJAX is Based on Open Standards

AJAX is based on the following open standards −

a) Browser-based presentation using HTML and Cascading Style Sheets (CSS).
b) Data is stored in XML format and fetched from the server.
c) Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
d) JavaScript to make everything happen.

- Google Maps A user can drag an entire map by using the mouse, rather than clicking on a button.
- Google Suggest As you type, Google offers suggestions. Use the arrow keys to navigate the results.
- Gmail Gmail is a webmail built on the idea that emails can be more intuitive, efficient, and useful.
- Facebook,Twitter,Youtube etc.. **All the browsers support AJAX technologies.**

Ajax Technologies

AJAX cannot work independently. It is used in combination with other technologies to create interactive webpages.

JavaScript

- Loosely typed scripting language.
- JavaScript function is called when an event occurs in a page.
- Glue for the whole AJAX operation.

DOM

- API for accessing and manipulating structured documents.
- Represents the structure of XML and HTML documents.

CSS

- Allows for a clear separation of the presentation style from the content and may bechanged programmatically by JavaScript

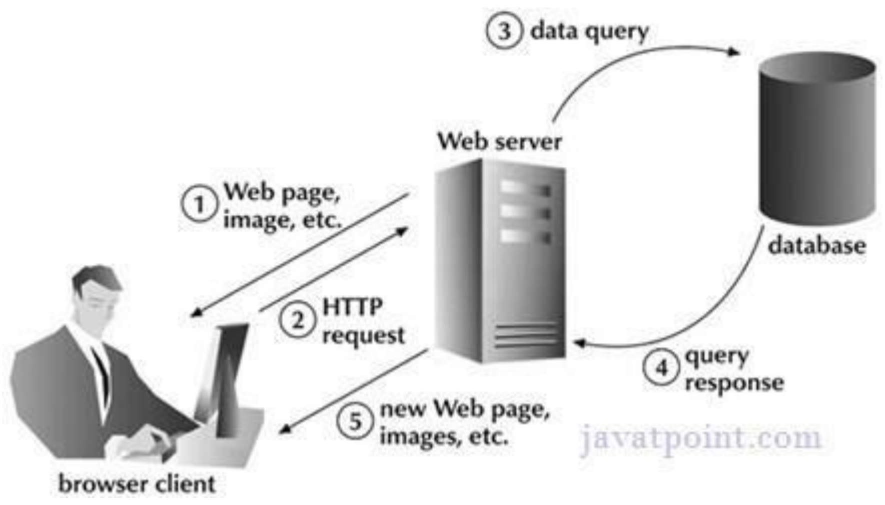
XMLHttpRequest

- JavaScript object that performs asynchronous interaction with the server.

2. How Ajax Works

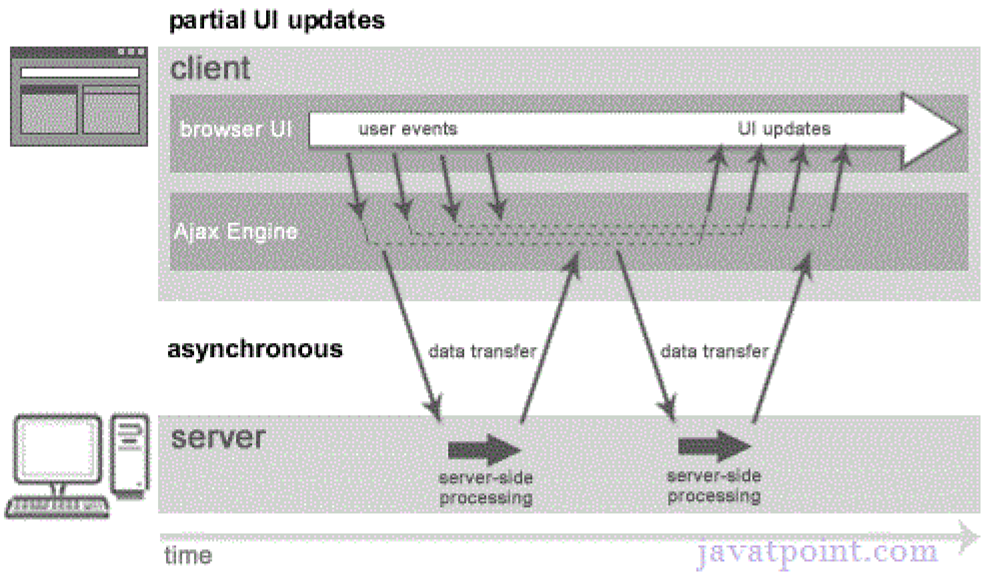
Synchronous (Classic Web-Application Model)

A synchronous request blocks the client until operation completes i.e. browser is unresponsive. In such case, javascript engine of the browser is blocked.



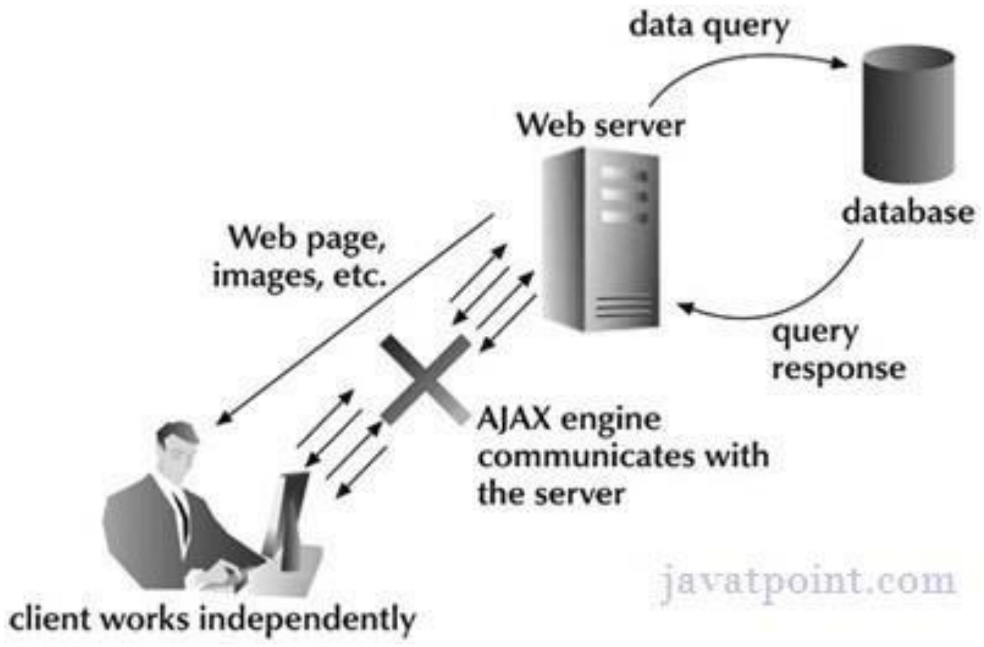
Asynchronous (AJAX Web-Application Model)

An asynchronous request doesn't block the client i.e. browser is responsive. At that time, user can perform another operations also. In such case, javascript engine of the browser is not blocked.



As you can see in the above image, full page is not refreshed at request time and user gets response from the ajax engine.

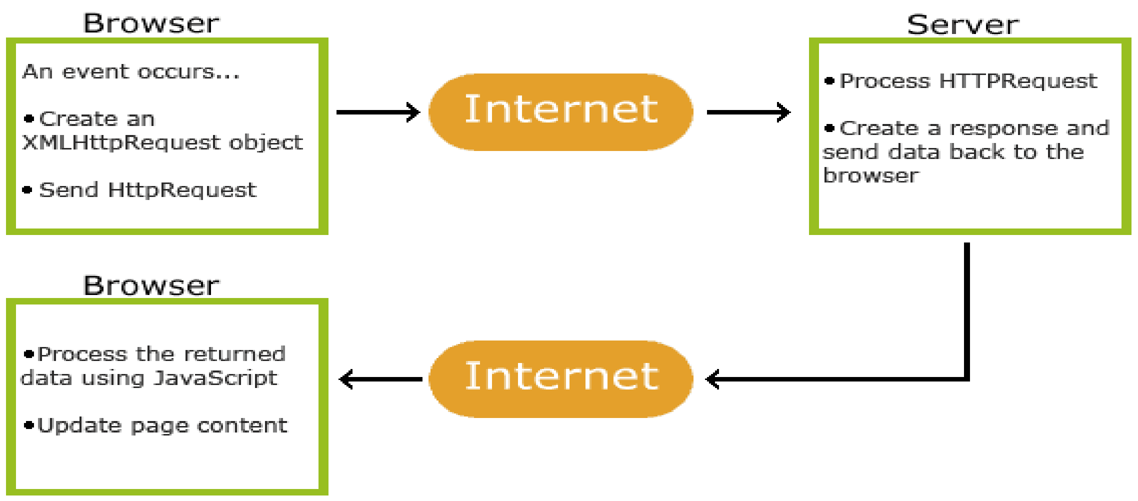
Let's try to understand asynchronous communication by the image given below.



An object of **XMLHttpRequest** is used for asynchronous communication between client andserver.

It performs following operations:

- ➢ Sends data from the client in the background
- ➢ Receives the data from the server
- ➢ Updates the webpage without reloading it.


Browser
An event occurs...
•Create an XMLHttpRequest object
•Send HttpRequest
Internet
Server
•Process HTTPRequest
•Create a response and send data back to the browser
Browser
•Process the returned data using JavaScript
•Update page content
Internet

1. An event occurs in a web page (the page is loaded, a button is clicked)
2. An XMLHttpRequest object is created by JavaScript
3. The XMLHttpRequest object sends a request to a web server
4. The server processes the request
5. The server sends a response back to the web page
6. The response is read by JavaScript

7. Proper action (like page update) is performed by JavaScript

Properties of XMLHttpRequest object

The common properties of XMLHttpRequest object are as follows:

Property	**Description**
onReadyStateChange	It is called whenever readystate attribute changes. It must notbe used with synchronous requests.
readyState	represents the state of the request. It ranges from 0 to 4. **0** UNOPENED open() is not called. **1** OPENED open is called but send() is not called.

	2 HEADERS_RECEIVED send() is called, and headers and statusare available. **3** LOADING Downloading data; responseText holds the data. **4** DONE The operation is completed fully.
reponseText	returns response as text.
responseXML	returns response as XML

he important methods of XMLHttpRequest object are as follows:

Method	Description
void open(method, URL)	opens the request specifying get orpost method and url.
void open(method, URL, async)	same as above but specifies asynchronous or not.
void open(method, URL, async, username, password)	same as above but specifies usernameand password.
void send()	sends get request.
void send(string)	send post request.
setRequestHeader(header,value)	it adds request headers.

Example using AJAX

The XMLHttpRequest Object

All modern browsers support the XMLHttpRequest object.

The XMLHttpRequest object can be used to exchange data with a server behind the scenes. Thismeans that it is possible to update parts of a web page, without reloading the whole page.

Create an XMLHttpRequest Object

All modern browsers (Chrome, Firefox, IE7+, Edge, Safari Opera) have a built-in XMLHttpRequestobject.

Syntax for creating an XMLHttpRequest object: **variable = new XMLHttpRequest();**

The onreadystatechange Property

With the XMLHttpRequest object you can define a function to be executed when the requestreceives an answer.

The function is defined in the **onreadystatechange property of the XMLHttpResponse** object:

The onreadystatechange Property

The **readyState** property holds the status of the XMLHttpRequest.

The **onreadystatechange** property defines a function to be executed when the readyState changes.

The **status** property and the **statusText** property holds the status of the XMLHttpRequest object.

Property	**Description**
onreadystatechange	Defines a function to be called when the readyState property changes
readyState	Holds the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
status	200: "OK" 403: "Forbidden" 404: "Page not found" For a complete list go to the Http Messages Reference
statusText	Returns the status-text (e.g. "OK" or "Not Found")

```
<!DOCTYPE html>
<html>
<body>

<h1>The XMLHttpRequest Object</h1>

<p id="demo">Let AJAX change this text.</p>

<button type="button" onclick="loadDoc()">Change Content</button>
<script>
function loadDoc()
 {var xhttp;
 if (window.XMLHttpRequest) {
   // code for modern browsers
   xhttp = new
   XMLHttpRequest();
   } else {
   // code for IE6, IE5
   xhttp = new ActiveXObject("Microsoft.XMLHTTP");
 }
 xhttp.onreadystatechange = function() {
   if (this.readyState == 4 && this.status == 200) {
     document.getElementById("demo").innerHTML =
     this.responseText;
   }
 };
 xhttp.open("GET",
 "ajax_info.txt", true);
 xhttp.send();
}
</script>

</body>
</html
```

Web Server:

Web server" can refer to hardware or software, or both of them working together.

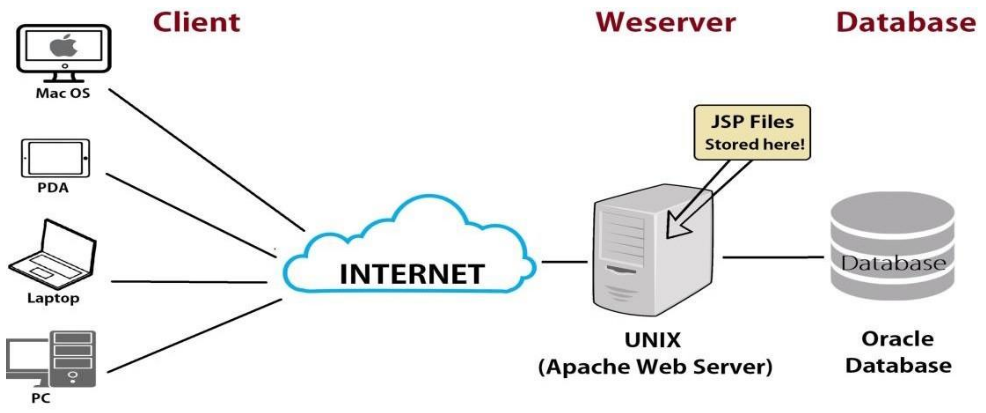
On the hardware side, a web server is a computer that stores web server software and a website's component files (e.g. HTML documents, images, CSS stylesheets, and JavaScript files). It is connected to the Internet and supports physical data interchange with other devices connected to the web.
On the software side, a web server includes several parts that control how web users access hosted files, at minimum an HTTP server. An HTTP server is a piece of software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages). It can be accessed through the domain names (like mozilla.org) of websites it stores, and delivers their content to the end-user's device.
A web server processes incoming network requests over HTTP and several other related protocols. The primary function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP).

At the most basic level, whenever a browser needs a file which is hosted on a web server, the browser requests the file via HTTP. When the request reaches the correct web server (hardware), the HTTP server (software) accepts request, finds the requested document (if it doesn't then a 404 response is returned), and sends it back to the browser, also through HTTP.

Basic representation of a client/server connection through HTTP
To publish a website, you need either a static or a dynamic web server.

A static web server, or stack, consists of a computer (hardware) with an HTTP server (software). We call it "static" because the server sends its hosted files "as-is" to your browser.

A dynamic web server consists of a static web server plus extra software, most commonly an application server and a database. We call it "dynamic" because the application server updates the hosted files before sending them to your browser via the HTTP server.



Various services provided by the web server are:

1. **Cost Efficient:** Web server is the most cost efficient method to use, maintain and upgrade. Traditional desktop software costs companies a lot in terms of finance. On the other hand. It is available at much cheaper rates, Besides, there are many one-time-payment, pay-as-you-go and other salable options available, which makes it very reasonable for the company.
2. **Resource Sharing:** Web Server has the capability to store unlimited information such as Google Drives, Cloud computing etc. The space where the data can be stored is shared by the other users at the same time like hard disk can be shared on physical network as LAN.
3. **Data Sharing:** With the help of web servers one can easily access the information from anywhere, where there is an Internet connection using Google docs such as Documents, Excel sheets, Drawings, power point presentations etc.
4. **Backup and Recovery:** As all the data now a days is stored on web servers, backing it up and restoring the same is relatively much easier than storing the same on a physical device. Hence, the entire process of backup and recovery much simpler than other traditional methods of data storage.

Types of Servers:

5. **Mail Server:** Mail Server provides a centrally-located pool of disk space for network users to store and share various documents in the form of emails. Since, all the data is stored in one location, administrators need only backup files from one computer.
6. **Application Server:** An application server acts as a set of components accessible to the software developer through an API defined by the platform itself. For Web applications. These components are usually performed in the same running environment as its web server(s), and their main job is to support the construction of dynamic pages.
7. **File Transfer Protocol (FTP) Server:** FTP uses separate control and data connections between the client and the server. FTP users may authenticate themselves in the form of a username and password. But can connect anonymously if the server is configured to allow it. For secure transmission username and password must be encrypted using FTP and SSL.
8. **Database Server:** A database server is a computer program that provides database services to other computer programs or computers using client-server functionality, and some DBMSs (e.g., Mysql) depend on the client-server model for database access. Such a server is accessed either through a "front end" running on the user's computer where the request is made or the "back end" where the request is served such as data analysis and storage.
9. **Domain Name System (DNS) Server:** A name server is a computer server that hosts a network service for providing responses to queries. It maps a numeric identification or addressing component. This service is performed by the server in response to a network service protocol request.

IIS(INTERNET INFORMATION SERVER) webserver:

A web server processes incoming network requests over HTTP and several other related protocols. The primary function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP).

IIS is a web server that runs on the Microsoft .NET platform on the Windows OS.
The two main process models for web servers are to either handle all requests on a single thread, or to spawn a new thread for each request. Although the single-thread model (Node.js, for example) has some worker threads available, it typically only uses them for certain kinds of work, such as file system access. **The thread-per-request model that uses will grab a thread from a thread pool for each request**.

IIS is rich with features.

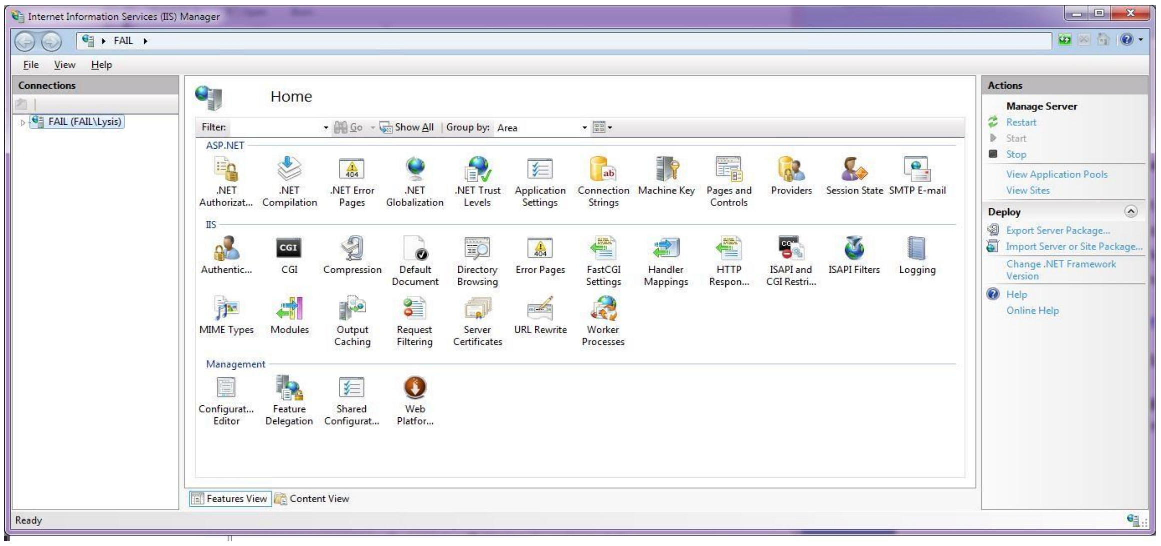
- Most commonly, IIS is used to host ASP.NET web applications and static websites. It can also be used as an FTP server, host WCF services, and be extended to host web applications built on other platforms such as PHP.

- There are built-in authentication options such as Basic, ASP.NET, and Windows auth. The latter is useful if you have a Windows Active Directory environment—users can be automatically signed into web applications using their domain account.

- Other built-in security features include TLS certificate management and binding for enabling HTTPS and SFTP on your sites, request filtering for whitelisting or blacklisting traffic, authorization rules, request logging, and a rich set of FTP-specific security options.

- One key feature of IIS is the application pool. We'll have to take a closer look at the application pool, as it's a critical component of the IIS process model.

The main IIS configuration utility is found in the Windows Control Panel. It's in the "System and Security" section in the "Administrative Tools" option. The below image gives you a quickoverview of the IIS 7.5 main manager window:



Virtual Directories

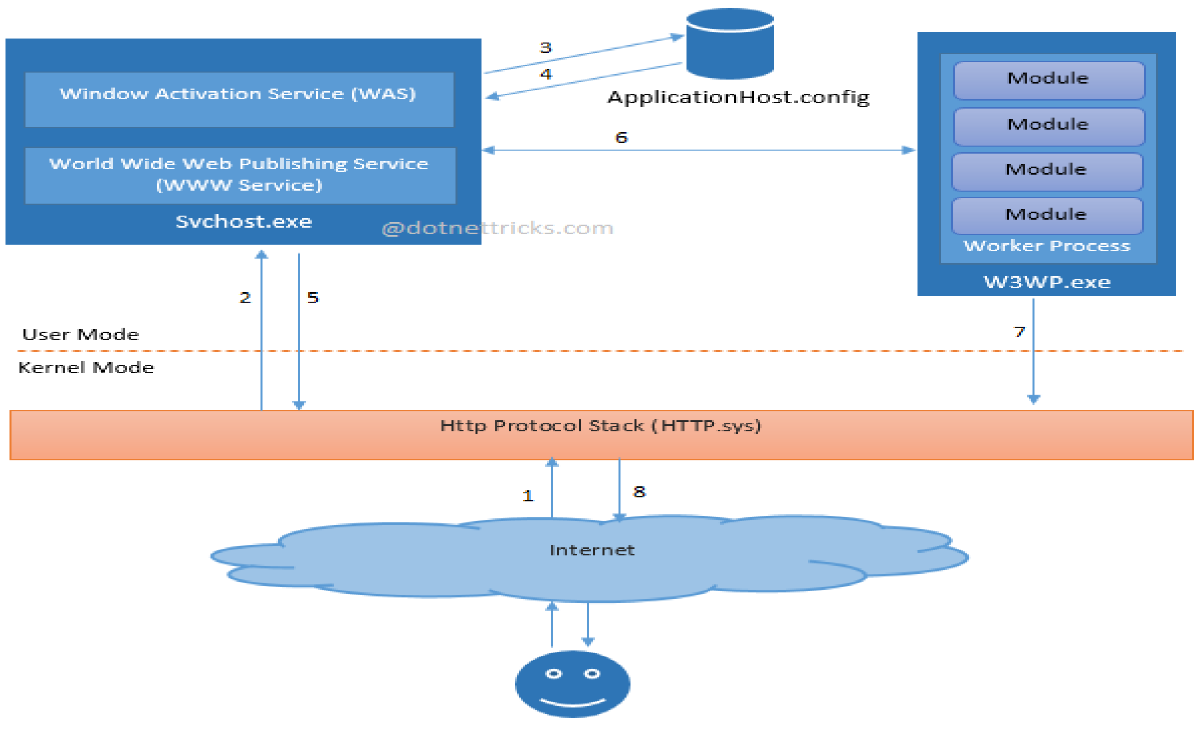
- IIS allows you to create sites, applications, and virtual directories to share information with users over the Internet or internally on an intranet such as a home network.
- A virtual directory is a name that you specify in IIS and that maps to a physical directory on a server similar to how DNS maps a URL to an IP address.

Ports

- Typically your server will use port 80 for HTTP traffic however this can be adjusted to meet your needs or the needs of another application on your computer.
- You can find a full list of ports and the purpose they each serve here. Changing a port within IIS 7 to 10 is simple. First Open Internet Information Services Manage

IIS Architecture:

IIS has two main layers - Kernel Mode and User Mode. The Kernel Mode contains the HTTP.SYS and User Mode contains WAS and W3 service. The subsection of both are shown in fig.



IIS Architecture

The above diagrams shows the flow of an HTTP request in process. The request-processing flowis described as:

10. An HTTP request first goes to HTTP.sys and now, HTTP.SYS is responsible for passing the request to a particular application pool.
11. HTTP.sys contacts to WAS and WAS requests configuration information from the xml file.
12. The configuration information is sent to WWW service receives.
13. The WWW service uses the configuration information to configure HTTP.sys.
14. Configured HTTP.sys contacts to WAS and now, WAS starts a worker process for the application pool to which the request was made.
15. The worker process processes the request and returns a response to HTTP.sys. The request is passed through an ordered series of module in the processing pipeline.

Role of HTTP.sys in IIS

HTTP.SYS is the part of kernel mode of IIS. Every client request is passes through the kernel mode, Http.sys then makes a queue for each and individual application pool based on the request. Whenever we create any application pool IIS automatically registers the pool with HTTP.sys to identify the particular during request processing. Itprovides the following services in IIS:

1. Routing HTTP requests to the correct request queue.
2. Caching of responses in kernel mode.
3. Performing all text-based logging for the WWW service.
4. Implementing quality of service functionality, which includes connection limits, connection timeouts, queue-length limits, and bandwidth throttling.

ISAPI Filter

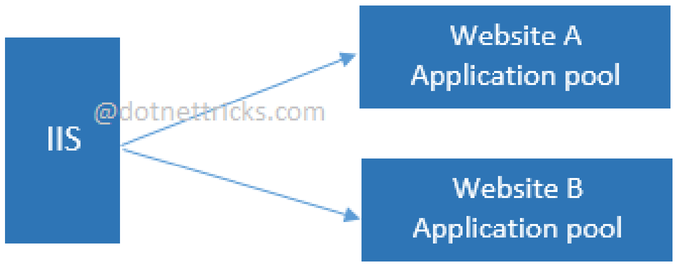
ISAPI filters are DLL files that can be used to modify and enhance the functionality provided by IIS. ISAPI filters always run on an IIS server, filtering every request until they find one they need to process.
ISAPI filters can be registered with IIS to modify the behavior of a server. It can perform the following tasks:

1. Change request data (URLs or headers) sent by the client
2. Control which physical file gets mapped to the URL
3. Control the user name and password used with anonymous or basic authentication
4. Modify or analyze a request after authentication is complete
5. Modify a response going back to the client
6. Run processing when a request is complete
7. Run processing when a connection with the client is closed
8. Perform special logging or traffic analysis.
9. Handle encryption and compression.

Different Security Settings Available in IIS

IIS provides a variety of authentication schemes:

1. Anonymous (enabled by default)
2. Basic
3. Digest
4. Integrated Windows authentication (enabled by default)
5. Client Certificate Mapping



Key Points About Application Pool

1. Provides isolation between different web applications.
2. Every web application has individual worker process.
3. Improve manageability of web application.
4. Provides better performance.

In IIS 6.0, **WWW Service** manages the following main areas in IIS:

- HTTP administration and configuration
- Process management
- Performance monitoring

Windows Process Activation Service (WAS)

In IIS 7 and later, Windows Process Activation Service (WAS) manages application pool configuration and worker processes instead of the WWW Service. This enables you to use the same configuration and process model for HTTP and non-HTTP sites.

WAS manages application pools and worker processes for both HTTP and non-HTTP requests. When a protocol listener picks up a client request, WAS determines if a worker process is running or not. If an application pool already has a worker process that is servicing requests, the listener adapter passes the request onto the worker process for processing.

An **IIS Worker Process (w3wp.exe**) handles the web requests sent to the IIS web server for the configured IIS application pool. IIS application pools also provide a bunch of advanced settings. These impact the behavior of w3wp and your IIS worker process.
IIS application pools also provide a bunch of advanced settings. These impact the behavior of w3wp and your IIS worker process. Including things like what Windows user account it runs as, auto restarting of the process, auto shutdown, and more. It is also possible for one IIS applicationpool to create multiple IIS worker processes in what is called a web garden.

Apache Tomcat WebServer:

Apache Tomcat is a webcontainer which allows to run servlet and JavaServer Pages (JSP) based web applications. Most of the modern Java web frameworks are based on servlets, e.g. JavaServer Faces, Struts, Spring.

Apache Tomcat also provides by default a HTTP connector on port 8080, i.e., Tomcat can also be used as HTTP server. But the performance of Tomcat is not as good as the performance of a designated web server, like the Apache HTTP server.

Tomcat is used in conjunction with Apache HTTP Server where Apache HTTP Server attends static content like html, images etc., and forwards the requests for dynamic content to Tomcat. This is because Apache HTTP Server supports more advanced options than that of Tomcat.

Jasper 2

Jasper is the JSP Engine for Tomcat. Jasper is responsible for parsing JSP files and compilationof JSP’s Java code as servlets.

Jasper is capable of background compilation, which means if any changes are made to JSP files, then the older versions of those JSP files are still retained by the server, until the updated JSP files are recompiled.

Installing Apache Tomcat on Ubuntu

To install Tomcat on Ubuntu, you could use command line interface and run the following command :

```
~$ sudo apt-get install
```

Start Apache Tomcat

Once you install Tomcat, it is started automatically.

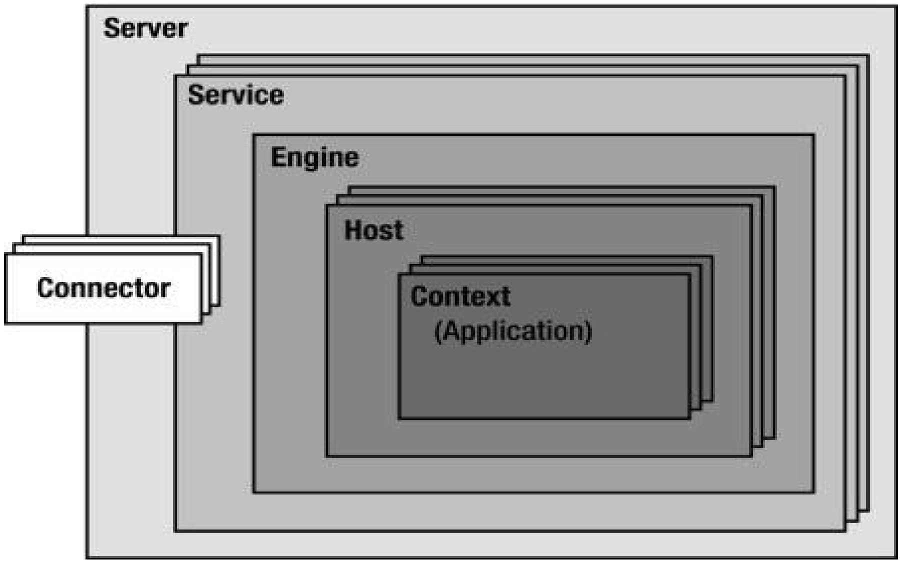
In case if you have stopped it manually, and would like to start Apache Tomcat again, open aterminal and run the following command :

```
~$ sudo /etc/init.d/tomcat8
start
```

Deploying Dynamic Web-Applications with Apache Tomcat

.war is the format of the web application that Apache Tomcat Server could deploy. If you are building a web application using an IDE like Eclipse, you could export the application as a WARfile

Tomcat Architecture



Context

A Context is the innermost element of a group of Tomcat components called containers,and it **represents a single web application**. Tomcat automatically instantiates and configures a standard context upon loading your application. As part of the configuration, Tomcat also processes the properties defined in the `\WEB-INF\web.xml` file of your application folder and makes them available to the application.

Connector

A Connector handles communications with the client. There are multiple connectors available with Tomcat e.g. HTTP connector for most of the HTTP traffic and AJP connector which implements the AJP protocol used when connecting Tomcat to anotherweb server such as Apache HTTPD server.

The default configuration of Tomcat includes a connector to handle HTTP communication. By default, this connector waits for requests coming through port **8080**. This is why the URLs of our examples always start with `http://localhost:8080/`. Note that the requests for all applications go through a single instance of this connector. Each new request causes the instantiation of a new thread that remains alive within the connector for the duration of the request. Tomcat often refer to this connector as "**`Coyote`**".AJP connector lets Tomcat only handle dynamic web pages and lets a pure HTML server (e.g., the Apache Web Server) handle the requests for static pages. This maximizes the efficiency with which the requests are handled.

Host

A Host is an association of a network name, e.g. www.yourdomain.com, to the Tomcat server. A host can contain any number of contexts (i.e. applications). You can define several hosts on the same server. For example, if you have registered the domain `yourdomain.com`, you can define host names such as `w1.yourdomain.com` and `w2.yourdomain.com`. Keep in mind that it will only be accessible from the Internet if a domain name server maps its name to the IP address of your computer. The default configuration of Tomcat includes the host named **localhost**

Engine

An Engine represents request processing pipeline for a specific Service. As a Service may have multiple Connectors, the Engine receives and processes all requests from these connectors, handing the response back to the appropriate connector for transmission to the client.

An engine must contain one or more hosts, one of which is designated as the default host. The default Tomcat configuration includes the engine Catalina, which contains the host localhost (obviously designated to be the default host because it is the only one). The Catalina engine handles all incoming requests received via the HTTP connector and sends back the corresponding responses. It forwards each request to the correct host and context on the basis of the information contained in the request header.

Service

A Service is an intermediate component which lives inside a Server and ties one or more Connectors to exactly one Engine. Tomcat's default configuration includes the service Catalina which associates the HTTP and AJP connectors to the Catalina engine. Accordingly, Connector and Engine are subelements of the Service element.

Server

The Server is the top component and represents an instance of Tomcat. It can contain one or more services, each with its own engine and connectors.

Working of Tomcat

The lifecycle of a typical servlet running on Tomcat might look something like this:

16. Tomcat receives a request from a client through one of its connectors.
17. Tomcat maps this request to the appropriate Engine for processing. These Engines are contained within other elements, such as Hosts and Servers, which limit the scope of Tomcat's search for the correct Engine.
18. Once the request has been mapped to the appropriate servlet, Tomcat checks to see if that servlet class has been loaded. If it has not, Tomcat compiles the servlet into Java bytecode, which is executable by the JVM, and creates an instance of the servlet.
19. Tomcat initializes the servlet by calling its init method. The servlet includes code that is able to read Tomcat configuration files and act accordingly, as well as declare any resources it might need, so that Tomcat can create them in an orderly, managed fashion.
20. Once the servlet has been initialized, Tomcat can call the servlet's service method to process the request, which will be returned as a response.
21. During the servlet's lifecycle, Tomcat and the servlet can communicate through the use of listener classes, which monitor the servlet for a variety of state changes. Tomcat can retrieve and store these state changes in a variety of ways, and allow other servlets access to them, allowing state to be maintained and accessed by various components of a given context across the span of a single or

multiple user sessions. An example of this functionality in action is an e-commerce application that remembers what the user has added to their cart and is able to pass this data to a checkout process.

22. Tomcat calls the servlet's destroy method to smoothly remove the servlet. This action is triggered either by a state change that is being listened for, or by an external command delivered to Tomcat to undeploy the servlet's Context or shut down the server.

Web hosting provider:

The first thing to consider when starting your website is to choose a web hosting provider (if you already have a web domain[1]). The web hosting provider provides the web **space** (i.e. special computers called web servers) where your website files are stored, as well as the **technologies** and **services** needed for your website to be viewed on the Internet.

Add-on services provided by a web hosting provider typically include data backup, firewall protection, technical support, email services, domain name registration, website building tools, and applications.

- Round the clock support
- Unlimited[2] storage
- Unlimited (professional) email addresses
  (with Unlimited Autoresponders, Mail Forwards, Email Aliases, Mailing Lists, etc.)
- 1-click installation
  (WordPress, Joomla, Magento, Drupal, phpBB, Gallery and many other CMSs)
- Latest cPanel

5. Discuss about to Hosting a Website.

Step 1: Decide What Type of Website You Want

You will typically find 2 types of websites:

Static or Basic Websites: Static websites are simple websites with one or more web pages (called HTML pages). You can build them on your computer with software like Dreamweaver and then upload the pages to your host's server using any FTP software (such as FileZilla). Since they cannot be modified dynamically, such websites are called static websites.Static websites are cheaper than dynamic websites (below) but come with limited functionality and no option for e- commerce or interactivity.
Dynamic Websites: Dynamic websites contain information that changes, depending on the time of day, the viewer and other factors. They make use of both client-side and server-side scripts to create and update content. Client-side scripts, which run on a user's computer, are mainly used for appearance and interaction purposes. Server-side scripts, which reside on a server and are extensively used by E-commerce and social networking sites, allow users to have individual accounts and provide a customized response for each user. Examples of dynamic websites include blogs, forums, photo galleries and e-commerce sites.

Step 2: Choose Your Hosting Server

Unlike static HTML sites which can be hosted on most web servers, when it comes to web applications, there are basically two types of hosting platforms. Depending on your hosting needs and what you're most comfortable with, you can choose from:

- **Linux Hosting,** which allows running scripts written in PHP, Perl, Python and other Unix- originated languages, and usually supports PostgreSQL and MySQL databases. This isthe most commonly used system today.
- **Windows Hosting,** which allows running ASP scripts utilizing .NET and other Microsoft technologies, and supports Microsoft SQL Server and Access database.

Step 3: Select Your Web Hosting Plan

You will typically find a wide range of services in web hosting, such as:

- **Shared Hosting:** In shared hosting, you get to share the physical server with other website owners. However, you will have your own separate account (secured with login credentials). Shared hosting is very affordable because the cost of operating the server is shared between you and the other website owners.
- **VPS Hosting (Virtual Private Server Hosting):** In VPS hosting, every website is stored on a very powerful server that is divided into several virtual compartments. The server software is configured separately so that each unit can function independently. It should be your preferred option if you have high-security concerns but don't want to invest in a faster (but costlier) dedicated server.
- **Dedicated Hosting:** Dedicated hosting offers you an entire server for yourself, thereby making it faster, more secure…and costlier. It is the ideal solution for larger businesses and high-traffic websites because it allows for maximum customization, configuration,installation and flexibility.
- **Cloud Hosting:** Cloud hosting allows multiple virtual servers (clouds) to work together to host a website or a group of websites. It offers unlimited ability to handle sudden traffic spikes. A cloud-hosted website is not limited to a single server, and the resources allocated to it can shrink or expand dynamically, depending on how much traffic you get. It's a great option for large websites, including e-commerce websites, newsletters and blogs.

Step 4: Change Your DNS Address

After you have purchased your web hosting, you will get Name Servers (also known as Domain Name Servers or DNS) – which is the Internet's equivalent of a phone book that contains IP Addresses. To get your website up and working, you will need to change the Name Servers of your domain. It's a simple but mandatory step for you to get started.

1. Go to your Domain Control Panel via http://manage.hostgator.in/customer.
2. Enter your registered **email address** and **password**.
3. Click on the **Domain Name** for which you need to change the Name Servers.
4. In the Domain Registration section, click on the **Name Servers** option.
5. Replace the existing Name Servers with the ones **provided by your current web host,** and click on the **Update Name Servers** button.

Step 5: Upload Your Website

You can now upload your website to your account by connecting to the server using either cPanel's **File Manager** or **FTP Client** (such as FileZilla) – after which your website will go live.

Step 6: share the DNS address of your website to client.

Short questions

1.Write about ajax technologies?
2.What is web browser?
3.What are the types of servers?
4.Features of web hosting provider?

Long questions

1. **Discuss about to hosting a website?**
2. **Discuss about apache tomcat webserver?**
3. **Discuss about IIS?**
4. **What is ajax? Write about its features and applications?**

UNIT-4

Features of PHP

The **PHP Hypertext Preprocessor (PHP)** is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications.

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

PHP is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning PHP:

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".

- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

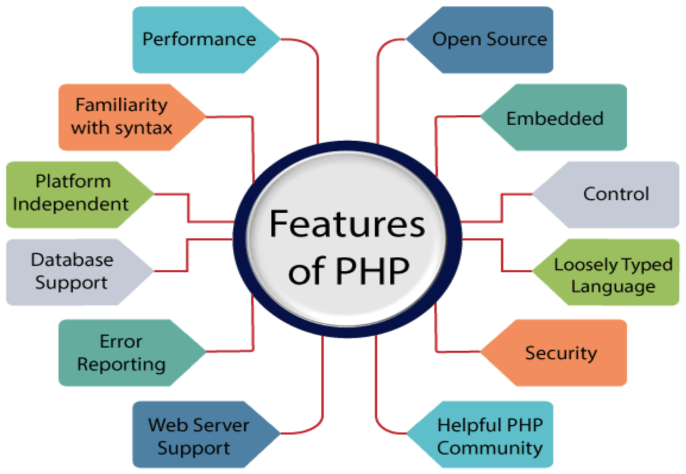
Characteristics of PHP

Five important characteristics make PHP's practical nature possible −

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

Applications of PHP

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data



How to run PHP

In order to develop and run PHP Web pages three vital components need to be installed on your computer system.

- **Web Server** – PHP will work with virtually all Web Server software, including Microsoft's Internet Information Server (IIS) but then most often used is freely available Apache Server. Download Apache for free here – https://httpd.apache.org/download.cgi . We CAN USE **XAMP SERVER.**
- **Database** – PHP will work with virtually all database software, including Oracle and Sybase but most commonly used is freely available MySQL database. Download MySQL for free here – https://www.mysql.com/downloads/
- **PHP Parser** – In order to process PHP script instructions a parser must be installed to generate HTML output that can be sent to the Web Browser. This tutorial will guide you how to install PHP parser on your computer.

// This is a single-line comment /* ………*/ for multiline.

# This is also a single-line comment

Php Code is enclosed between

```
    <?php...
    ….. 
…..

    ?>
<html>

        <head>
            <title>Hello World</title>
        </head>

        <body>
            <?php echo "Hello, World!";?></body></html>
```

In PHP, a variable is declared using a **$ sign** followed by the variable name. Here, some important points to know about variables:

As PHP is a loosely typed language, so we do not need to declare the data types of the variables. It automatically analyzes the values and makes conversions to its correct datatype.

After declaring a variable, it can be reused throughout the code.Assignment Operator (=) is used to assign the value to a variable.

Syntax of declaring a variable in PHP is given below:

$variablename=value;

PHP Constants

PHP constants are name or identifier that can't be changed during the execution of the script except for magic constants, which are not really constants. PHP constants can be defined by 2 ways:

- Using define() function
- **Using const keyword**

```
<?php
define("MESSAGE","Hello JavaTpoint PHP",true);//not
case sensitive echo MESSAGE, "</br>";
echo
message;
const
pi=3.14;
echo pi;
?>
```

PHP Data Types

Variables can store data of different types, and different data types can do different things. PHP supports the following data types:

- String
- Integer
- Float (floating point numbers - also called double)
- Boolean
- Array
- Object
- NULL

PHP String

A string is a sequence of characters, like "Hello world!".
A string can be any text inside quotes. You can use single or double quotes:

Example

```
<?php
$x = "Hello world!";
$y = 'Hello
world!'; echo
$x;
```

```
echo "<br>";  echo $y;?>
?>
```

PHP Operators

Operators are used to perform operations on variables and values.

PHP divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Increment/Decrement operators
- Logical operators
- String operators
- Conditional assignment operators

PHP Arithmetic Operators

- The PHP arithmetic operators are used with numeric values to perform common arithmetical operations, such as addition, subtraction, multiplication etc.

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y
**	Exponentiation	$x ** $y	Result of raising $x to the $y'th power

PHP Assignment Operators

The PHP assignment operators are used with numeric values to write a value to a variable.

The basic assignment operator in PHP is "=". It means that the left operand gets set to the value of the assignment expression on the right.

Assignment	**Same as...**	**Description**
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

PHP Comparison Operators

The PHP comparison operators are used to compare two values (number or string):

Operator	**Name**	**Example**	**Result**
==	Equal	$x == $y	Returns true if $x is equal to $y
===	Identical	$x === $y	Returns true if $x is equal to $y, andthey are of the same type

!=	Not equal	$x != $y	Returns true if $x is not equal to $y
<>	Not equal	$x <> $y	Returns true if $x is not equal to $y
!==	Not identical	$x !== $y	Returns true if $x is not equal to $y, orthey are not of the same type
>	Greater than	$x > $y	Returns true if $x is greater than $y
<	Less than	$x < $y	Returns true if $x is less than $y
>=	Greater thanor equal to	$x >= $y	Returns true if $x is greater than or equalto $y
<=	Less than or equal to	$x <= $y	Returns true if $x is less than or equal to $y
<=>	Spaceship	$x <=> $y	Returns an integer less than, equal to, or greater than zero, depending on if $x is less than, equal to, or greater than $y. Introduced in PHP 7.

PHP Increment / Decrement Operators

The PHP increment operators are used to increment a variable's value.
The PHP decrement operators are used to decrement a variable's value.

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x

$x--	Post-decrement	Returns $x, then decrements $x by one

PHP Logical Operators

- The PHP logical operators are used to combine conditional statements.

Operator	Name	Example	Result
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

PHP String Operators

- PHP has two operators that are specially designed for strings.

Operator	Name	Example	Result
.	Concatenation	$txt1 . $txt2	Concatenation of $txt1 and $txt2
.=	Concatenation assignment	$txt1 .= $txt2	Appends $txt2 to $txt1

?:	Ternary	$x = expr1 ? expr2 : expr3	Returns the value of $x.

The value of $x is expr2 if expr1 = TRUE,The value of $x is expr3 if expr1 = FALSE

PHP Echo

PHP echo is a language construct, not a function. Therefore, you don't need to use parenthesis with it. But if you want to use more than one parameter, it is required to use parenthesis.

The syntax of PHP echo is given below: **void echo (string $arg1 [, string $...])**

PHP Print

Like PHP echo, PHP print is a language construct, so you don't need to use parenthesis with the argument list. Print statement can be used with or without parentheses: print and print(). Unlike echo, it always returns 1.

The syntax of PHP print is given below: **int print(string $arg)**

```
<?php
print "Hello by PHP
print "; print ("Hello by
PHP print()");
?>
```

2. write about $ and $$.

PHP $ and $$ Variables

The **$var** (single dollar) is a normal variable with the name var that stores any value like string, integer, float, etc.

The **$$var** (double dollar) is a reference variable that stores the value of the $variable inside it.

```
<?php
$x = "abc";
$$x = 200;
echo
$x."<br/>";
echo
$$x."<br/>";
echo $abc;
?>
```

5. Discuss about Decision making in PHP.

PHP If Else

PHP if else statement is used to test condition. There are various ways to use if statement in PHP.

- if
- if-else

- if-else-if
- nested if

PHP If Statement

PHP if statement allows conditional execution of code. It is executed if condition is true.

If statement is used to executes the block of code exist inside the if statement only if the specified condition is true.

Syntax:

```
if(condition){
    //code to be executed
}
```

Ex:-

```
<?php
$num=12;
if($num<100){
echo "$num is less than 100";
}
?>

```

PHP If-else Statement

PHP if-else statement is executed whether condition is true or false.

If-else statement is slightly different from if statement. It executes one block of code if the specified condition is true and another block of code if the condition is false.

Syntax:-

```
if(condition)
{
//code to be executed if true
}else{
//code to be executed if false
}
```

```
<?php
$num=12;
if($num%2==0){
echo "$num is even number";
}else{
echo "$num is odd number";
}
?>
```

If-elsif statement

Syntax:-

```
if (condition1){
//code to be executed if condition1 is true
} elseif (condition2){
//code to be executed if condition2 is true
} elseif (condition3){
//code to be executed if condition3 is true
....
}  else{
//code to be executed if all given conditions are false
}
```

```
<?php
   $marks=69;
   if ($marks<33){
   echo "fail";
   }
   else if ($marks>=34 && $marks<50) {
   echo "D grade";
   }
   else if ($marks>=50 && $marks<65) {
      echo "C grade";
   }
   else if ($marks>=65 && $marks<80) {
       echo "B grade";
   }
   else if ($marks>=80 && $marks<90) {
       echo "A grade";
   }
   else if ($marks>=90 && $marks<100) {
       echo "A+ grade";
   }
  else {
       echo "Invalid input";
   }
?>
```

```
//nested IF statement
<?php
            $age = 23;
    $nationality = "Indian";
    //applying conditions on nationality and age
    if ($nationality == "Indian")
    {
        if ($age >= 18) {
            echo "Eligible to give vote";
        }
        else {
            echo "Not eligible to give vote";
        }
    }
?>
```

The Switch Statement

If you want to select one of many blocks of code to be executed, use the Switch statement.

The switch statement is used to avoid long blocks of if..elseif..else code.

Syntax

```
switch (expression){
   case label1:
      code to be executed if expression = label1;
      break;

   case label2:
      code to be executed if expression = label2;
      break;
      default:

   code to be executed
   if expression is different
   from both label1 and label2;
}
```

Important points to be noticed about switch case:

1. The **default** is an optional statement. Even it is not important, that default must always be the last statement.

2. There can be only one **default** in a switch statement. More than one default may lead to a **Fatal** error.
3. Each case can have a **break** statement, which is used to terminate the sequence of statement.
4. The **break** statement is optional to use in switch. If break is not used, all the statements will execute after finding matched case value.
5. PHP allows you to use number, character, string, as well as functions in switch expression.
6. Nesting of switch statements is allowed, but it makes the program more complex and less readable.
7. You can use semicolon (;) instead of colon (:). It will not generate any error.

```php
<?php
$num=20;
switch($num){
case 10:
echo("number is equals to 10");
break;
case 20:
echo("number is equal to 20");
break;
case 30:
echo("number is equal to 30");
break;
default:
echo("number is not equal to 10, 20 or30");
}
?>
```

```php
<?php
    $ch = "B.Tech";
    switch ($ch)
    {
        case "BCA":
echo "BCA is 3 years course";
            break;
        case "Bsc":
echo "Bsc is 3 years course";
            break;
        case "B.Tech":
echo "B.Tech is 4 years course";
            break;
        case "B.Arch":
echo "B.Arch is 5 years course";
            break;
        default:
            echo "Wrong Choice";
            break;
    }
?>
```

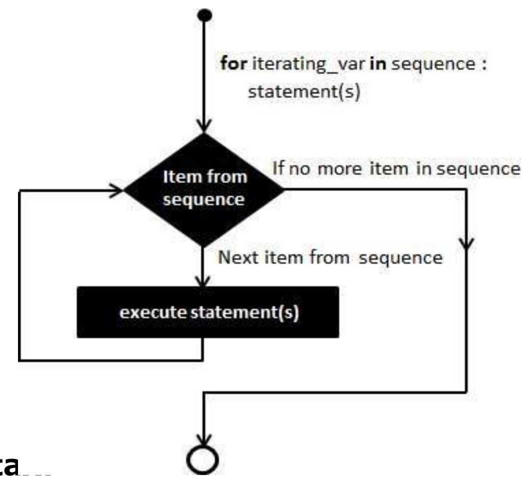
Repetition statements/Loops in PHP:

Loop in PHP are used to execute the same block of code a specified number of times. PHP supports following four loop types.

- **for** – loops through a block of code a specified number of times.
- **while** – loops through a block of code if and as long as a specified condition is true.
- **do...while** – loops through a block of code once, and then repeats the loop as long as a special condition is true.
- **foreach** – loops through a block of code for each element in an array.

The for loop statement

The for statement is used when you know how many times you want to execute a statement or a block of statements.



Synta

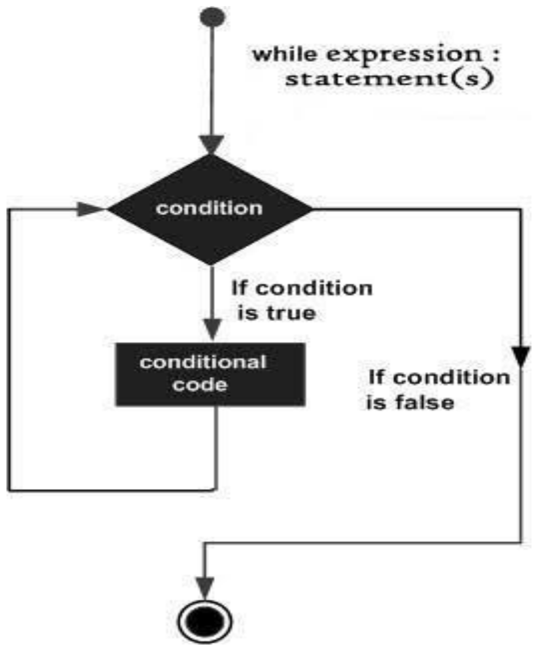
```
for (initialization; condition;
  increment){ code to be
  executed;
}
```

```
 <html>
<body>
    <?php
      $a = 0;
      $b = 0;

      for( $i = 0; $i<5; $i++ ) {
        $a += 10;
        $b += 5;
      }
          echo ("At the end of the loop a = $a and b = $b" );
    ?>

   </body>
 </html>
```

The while loop statement

The while statement will execute a block of code if and as long as a test expression is true.

If the test expression is true then the code block will be executed. After the code has executed the test expression will again be evaluated and the loop will continue until the test expression is found to be false.



Syntax:-

```
while (condition)
  { code to be
  executed;
}

<html>
  <body>
   <?php
      $i = 0;
      $num = 50;

      while( $i < 10) {
        $num--;
        $i++;
      }
       echo ("Loop stopped at i = $i and num = $num" );
    ?>
   </body>
</html>
```

The do...while loop statement

The do...while statement will execute a block of code at least once - it then will repeat the loop as long as a condition is true.

Syntax

```
do {
      code to be executed;
}
    while (condition);
```

```
<?php
         $i = 0;
         $num = 0;

         do {
            $i++;
         }

         while( $i < 10 );
         echo ("Loop stopped at i = $i" );
      ?>
```

The foreach loop statement

The foreach statement is used to loop through arrays. For each pass the value of the current array element is assigned to $value and the array pointer is moved by one and in the next pass next element will be processed.

Syntax

```
foreach (array as value) {
   code to be executed;
}
```

```
<html>
   <body>

      <?php
         $array = array( 1, 2, 3, 4, 5);

         foreach( $array as $value ) {
            echo "Value is $value <br />";
         }
      ?>

   </body>
</html>
```

PHP Arrays

An array is a data structure that stores one or more similar type of values in a single value. There are three different kind of arrays and each array value is accessed using an ID c which is called array index.

- **Numeric array** – An array with a numeric index. Values are stored and accessed in linear fashion.
- **Associative array** – An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.
- **Multidimensional array** – An array containing one or more arrays and values are accessed using multiple indices

Numeric Array

These arrays can store numbers, strings and any object but their index will be represented by numbers. By default array index starts from zero.

Example

Following is the example showing how to create and access numeric arrays.

Create an Array in PHP

In PHP, the `array()` function is used to create an array: `array();`

```
<html>
  <body>

    <?php
     /* First method to create array. */
     $numbers = array( 1, 2, 3, 4, 5);

     foreach( $numbers as
       $value ) { echo "Value
       is $value <br />";
     }
          /* Second method to create array. */
     $numbers[0] = "one";
     $numbers[1] = "two";
     $numbers[2] = "three";
     $numbers[3] = "four";
     $numbers[4] = "five";

     foreach( $numbers as
       $value ) { echo "Value
       is $value <br />";
    }
   ?>

  </body>
</html>
```

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";
?>
```

PHP Associative Arrays

Associative arrays are arrays that use named keys that you assign to them.

It is created as array(key=>value,key=>value..);

There are two ways to create an associative array:

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
```

Ex:-1

```
<?php
$salary=array("Sonoo"=>"350000","John"=>"450000","Kartik"=>"200000");
echo "Sonoo salary: ".$salary["Sonoo"]."<br/>";
echo "John salary: ".$salary["John"]."<br/>";
echo "Kartik salary: ".$salary["Kartik"]."<br/>";
?>
```

Ex:-2

```
<?php
$size=array("Big","Medium","Short");
foreach( $size as $s )
{
  echo "Size is: $s<br />";
}
?>
```

Ex:-3

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($age as $x => $x_value) {
  echo "Key=" . $x . ", Value=" .
  $x_value; echo "<br>";
}
?>
```

PHP Multidimensional Array

PHP multidimensional array is also known as array of arrays. It allows you to store tabular data in an array. PHP multidimensional array can be represented in the form of matrix which is represented by row * column.

```
<?php
$emp = array
  (
  array(1,"sonoo",400000),
  array(2,"john",500000),
  array(3,"rahul",300000)
  );

```

```
for ($row = 0; $row < 3; $row++) {
 for ($col = 0; $col < 3; $col++) {
   echo $emp[$row][$col]."  ";
 }
 echo "<br/>";
}
?>
```

Array Functions

1) PHP array() function

PHP array() function creates and returns an array. It allows you to create indexed, associative and multidimensional arrays.

2) PHP array_change_key_case() function

PHP array_change_key_case() function changes the case of all key of an array. array array_change_key_case (array $array [, int $case = CASE_LOWER])

3) PHP count() function

PHP count() function counts all elements in an array.

4)PHP sort() function

PHP sort() function sorts all the elements in an array

Ex:-<?php

```
$season=array("summer","winter","spring","
autumn"); sort($season);
foreach( $season as $s )
{
 echo "$s<br />";
}
?>
```

5) PHP array_reverse() function

PHP array_reverse() function returns an array containing elements in reversed order.

6)PHP array_search() function

PHP array_search() function searches the specified value in an array. It returns key if search is successful.

```
<?php
$season=array("summer","winter","spring","autumn");
$key=array_search("spring",$season);echo
$key;
?>
```

8.Explain about PHP strings and functions.

PHP string is a sequence of characters i.e., used to store and manipulate text. PHP supports only 256-character set and so that it does not offer native Unicode support. There are 4 ways to specify a string literal in PHP.

```
<?php
$str1='Hello
text multiple
line
text within single quoted string';
$str2='Using double "quote" directly inside single quoted string';
$str3='Using escape sequences \n in single
quoted string'; echo "$str1 <br/> $str2 <br/>
$str3"; ?>
```

PHP String Functions

strlen() - Return the Length of a String

```
<?php
    echo strlen("Hello world!"); // outputs 12
?>
```

str_word_count() - Count Words in a String

The PHP str_word_count() function counts the number of words in a string.

```
<?php
    echo str_word_count("Hello world!"); // outputs 2
?>
```

strrev() - Reverse a String

The PHP strrev() function reverses a string.

```
<?php
    echo strrev("Hello world!"); // outputs !dlrow olleH
?>
```

strpos() - Search For a Text Within a String

The PHP strpos() function searches for a specific text within a string. If a match is found, the function returns the character position of the first match. If no match is found, it will return FALSE.

```
    <?php
    echo strpos("Hello world!", "world"); // outputs 6
    ?>
```

str_replace() - Replace Text Within a String

The PHP str_replace() function replaces some characters with some other characters in a string.

```
<?php
    echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!
?>
```

9. Discuss about SUPERGLOBAL variables.

Some predefined variables in PHP are "superglobals", which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

The PHP superglobal variables are:

$GLOBALS
$_SERVER
$_REQUEST
$_POST
$_GET
$_FILES
$_ENV
$_COOKIE
$_SESSION

$GLOBALS

$GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods). PHP stores all global variables in an array called $GLOBALS[index]. The index holds the name of the variable.

```
<?php
    $x = 75;
$y = 25;
```

```
    function addition() {
      $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
    }
 addition();
echo $z;
    ?>
```

$_SERVER
$_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

```
<?php
    echo $_SERVER['PHP_SELF'];
echo "<br>";
    echo $_SERVER['SERVER_NAME'];
echo "<br>";
    echo $_SERVER['HTTP_HOST'];
echo "<br>";
    echo $_SERVER['HTTP_REFERER'];
echo "<br>";
    echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
    echo $_SERVER['SCRIPT_NAME'];
    ?>
```

The following table lists the most important elements that can go inside $_SERVER:

Element/Code	Description
$_SERVER['PHP_SELF']	Returns the filename of the currently executing script
$_SERVER['GATEWAY_INTERFACE']	Returns the version of the Common Gateway Interface(CGI) the server is using
$_SERVER['SERVER_ADDR']	Returns the IP address of the host server
$_SERVER['SERVER_NAME']	Returns the name of the host server (such as www.w3schools.com)
$_SERVER['SERVER_SOFTWARE']	Returns the server identification string (such asApache/2.2.24)

$_SERVER['SERVER_PROTOCOL']	Returns the name and revision of the information protocol (such as HTTP/1.1)
$_SERVER['REQUEST_METHOD']	Returns the request method used to access the page(such as POST)
$_SERVER['REQUEST_TIME']	Returns the timestamp of the start of the request(such as 1377687496)
$_SERVER['QUERY_STRING']	Returns the query string if the page is accessed via aquery string
$_SERVER['HTTP_ACCEPT']	Returns the Accept header from the current request

$_REQUEST

PHP $_REQUEST is a PHP super global variable which is used to collect data after submitting an HTML form.

```
<html>
<body>

<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_REQUEST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>

</body>
</html>
```

$_POST

PHP $_POST is a PHP super global variable which is used to collect form data after submitting an HTML form with method="post". $_POST is also widely used to pass variables.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to the file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $_POST to collect the value of the input field:

```
<html>
<body>

<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_POST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>

</body>
</html>
```

$_GET

PHP $_GET is a PHP super global variable which is used to collect form data after submitting an HTML form with method="get".

$_GET can also collect data sent in the URL.

Assume we have an HTML page that contains a hyperlink with parameters:

```
<html>
<body>

<a href="test_get.php?subject=PHP&web=W3schools.com">Test $GET</a>

</body>
</html>
```

Cookie:

A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

Create Cookies With PHP

A cookie is created with the `setcookie()` function. Syntax setcookie(*name, value, expire, path, domain, secure, httponly*);

PHP Create/Retrieve a Cookie

The following example creates a cookie named "user" with the value "John Doe". The cookie will expire after 30 days (86400 * 30). The "/" means that the cookie is available in entire website (otherwise, select the directory you prefer).

We then retrieve the value of the cookie "user" (using the global variable $_COOKIE). We also use the isset() function to find out if the cookie is set:

```
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>
</body>
</html>
```

10. Explain about PHP User Defined Functions and its usage

Besides the built-in PHP functions, it is possible to create your own functions.

- A function is a block of statements that can be used repeatedly in a program.
- A function will not execute automatically when a page loads.
- A function will be executed by a call to the function.

Create a User Defined Function in PHP

A user-defined function declaration starts with the word function:

Syntax

```
function functionName() {
    code to be executed;
}
```

```
<?php
function writeMsg() {
    echo "Hello world!";
}

writeMsg(); // call the function
?>
```

PHP Function Arguments

Information can be passed to functions through arguments. An argument is just like a variable.

Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

```php
<?php
function familyName($fname, $year) {
    echo "$fname Refsnes. Born in $year <br>";
}

familyName("Hege", "1975");
familyName("Stale", "1978");
familyName("Kai Jim", "1983");
?>
```

PHP Functions - Returning values

To let a function return a value, use the `return` statement:

```php
<?php declare(strict_types=1); // strict requirement
function sum(int $x, int $y) {
    $z = $x + $y;
    return $z;
}

echo "5 + 10 = " . sum(5, 10) . "<br>";
echo "7 + 13 = " . sum(7, 13) . "<br>";
echo "2 + 4 = " . sum(2, 4);
?>
```

Date and Time functions in PHP:

The PHP Date() Function

The PHP date() function formats a timestamp to a more readable date and time.
Syntax date(format,timestamp)

Get a Date

The required *format* parameter of the date() function specifies how to format

the date (or time). Here are some characters that are commonly used for

dates:

- d - Represents the day of the month (01 to 31)
- m - Represents a month (01 to 12)
- Y - Represents a year (in four digits)
- l (lowercase 'L') - Represents the day of the week

Other characters, like"/", ".", or "-" can also be inserted between the characters to add additional formatting.

```
    <?php
    echo "Today is " . date("Y/m/d") . "<br>";
echo "Today is " . date("Y.m.d") . "<br>";
    echo "Today is " . date("Y-m-d") . "<br>";
    echo "Today is " . date("l");
?>
```

Get a Time

Here are some characters that are commonly used for times:

- H - 24-hour format of an hour (00 to 23)
- h - 12-hour format of an hour with leading zeros (01 to 12)
- i - Minutes with leading zeros (00 to 59)
- s - Seconds with leading zeros (00 to 59)
- a - Lowercase Ante meridiem and Post meridiem (am or pm)

```
<?php
    echo "The time is " . date("h:i:sa");
    ?>
```

Create a Date From a String With strtotime()

The PHP strtotime() function is used to convert a human readable date string into a Unix timestamp(the number of seconds since January 1 1970 00:00:00 GMT).

Syntax strtotime(time, now)

The example below creates a date and time from the

strtotime() function: Example

<?php

$d=strtotime("10:30pm April 15 2014");

echo "Created date is " . date("Y-m-d h:i:sa", $d);

?>

PHP Date/Time Functions

Function	**Description**
checkdate()	Validates a Gregorian date
date_add()	Adds days, months, years, hours, minutes, and seconds to a date
date_diff()	Returns the difference between two dates
date_format()	Returns a date formatted according to a specified format
date_parse()	Returns an associative array with detailed info about a specified date
date_sub()	Subtracts days, months, years, hours, minutes, and seconds from a date

date_sunrise()	Returns the sunrise time for a specified day and location
date_sunset()	Returns the sunset time for a specified day and location
date_time_set()	Sets the time
date_timestamp_get()	Returns the Unix timestamp
date_timestamp_set()	Sets the date and time based on a Unix timestamp
date()	Formats a local date and time
getdate()	Returns date/time information of a timestamp or the current local date/time
gettimeofday()	Returns the current time

Examples:-

```
    1.<?php
$date1=date_create("2013-03-15");
    $date2=date_create("2013-12-12");
$diff=date_diff($date1,$date2);

    2. <!DOCTYPE html>

<html>
    <body>
<?php
    $date=date_create("2013-03-15");
echo date_format($date,"Y/m/d H:i:s");
    ?>
</body>
    </html>>
```

PHP form handling:

PHP Form Handling

We can create and use forms in PHP. To get form data, we need to use **PHP superglobals $_GET and $_POST.**

The form request may be get or post. To retrieve data from get request, we need to use $_GET, for post request
$_POST.

PHP Get Form

Get request is the default form request. The data passed through get request is visible on the URL browser so it is not secured. You can send limited amount of data through get request.

```
<form action="welcome.php" method="get">
Name: <input type="text" name="name"/>
<input type="submit" value="visit"/>
</form>
```

File: welcome.php

```
<?php
```

```
$name=$_GET["name"];//receiving name field value in $name variable
echo "Welcome, $name";
?>
```

PHP Post Form

Post request is widely used to submit form that have large amount of data such as file upload, image upload, login form, registration form etc.

The data passed through post request is not visible on the URL browser so it is secured. You can send large amount of data through post request.

Let's see a simple example to receive data from post request in PHP.

File: form1.html

```
<form action="login.php" method="post">
<table>
<tr><td>Name:</td><td> <input type="text" name="name"/></td></tr>
<tr><td>Password:</td><td> <input type="password" name="password"/></td></tr>
<tr><td colspan="2"><input type="submit" value="login"/>  </td></tr>
</table>
</form>
```

File: login.php

```
<?php
$name=$_POST["name"];//receiving name field value in $name variable
$password=$_POST["password"];//receiving password field value in $password variable

echo "Welcome: $name, your password is: $password";
?>
```

MySQL and its features

MySQL is one of the most popular relational database system being used on the Web today. It is freely available and easy to install, however if you have installed Wampserver it already there onyour machine. MySQL database server offers several advantages:

1. MySQL is easy to use, yet extremely powerful, fast, secure, and scalable.
2. MySQL runs on a wide range of operating systems, including UNIX or Linux, Microsoft Windows, Apple Mac OS X, and others.
3. MySQL supports standard SQL (Structured Query Language).
4. MySQL is ideal database solution for both small and large applications.
5. MySQL is developed, and distributed by Oracle Corporation.
6. MySQL includes data security layers that protect sensitive data from intruders.

Note:- All the commands are SQL commands with minor changes.

PHP Database Handling with example

a) Ways of Connecting to MySQL through PHP

In order to store or access the data inside a MySQL database, you first need to connect to the MySQL database server. PHP offers two different ways to connect to MySQL server: MySQLi (Improved MySQL) and PDO (PHP Data Objects) extensions.

While the PDO extension is more portable and supports more than twelve different databases, MySQLi extension as the name suggests supports MySQL database only. MySQLi extension however provides an easier way to connect to, and execute queries on, a MySQL database server.

b) Connecting to MySQL Database Server

In PHP you can easily do this using the mysqli_connect() function. All communication between PHP and the MySQL database server takes place through this connection. Here're the basic syntaxes for connecting to MySQL using MySQLi and PDO extensions:

Syntax: MySQLi, Object Oriented way

```
$mysqli = new mysqli("hostname", "username", "password", "database");
```

Syntax: PHP Data Objects (PDO) way

```
$pdo = new PDO("mysql:host=hostname;dbname=database", "username", "password");
```

The *hostname* parameter in the above syntax specify the host name (e.g. `localhost`), or IP address of the MySQL server, whereas the *username* and *password* parameters specifies the credentials to access MySQL server, and the *database* parameter, if provided will specify the default MySQL database to be used when performing queries.

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Close the Connection

The connection will be closed automatically when the script ends. To close the connection before, use the following:

MySQLi Object-Oriented:

```
$conn->close();
```

Create a MySQL Database Using MySQLi and PDO

The CREATE DATABASE statement is used to create a

database in MySQL. The following examples create a

database named "myDB":

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// Create database
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
  echo "Database created successfully";
} else {
  echo "Error creating database: " . $conn->error;
}

$conn->close();
?>
```

Insert Data Into MySQL Using MySQLi and PDO

After a database and a table have been created, we can start

adding data in them. Here are some syntax rules to follow:

- The SQL query must be quoted in PHP
- String values inside the SQL query must be quoted
- Numeric values must not be quoted
- The word NULL must not be quoted

The INSERT INTO statement is used to add new records to a MySQL table:

INSERT INTO table_name (column1, column2, column3,...) VALUES (value1, value2, value3,...)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

Select Data From a MySQL Database

The SELECT statement is used to select data from one or more tables:

```
SELECT column_name(s) FROM table_name
```

or we can use the * character to select ALL columns from a table:

```
SELECT * FROM table_name
```

Select Data With MySQLi

The following example selects the id, firstname and lastname columns from the MyGuests table and displays it on the page:

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
```

```
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
  // output data of each row while($row =
  $result->fetch_assoc()) {
    echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
  }
} else {
  echo "0 results";
}
$conn->close();
?>
```

Delete Data From a MySQL Table Using MySQLi and

PDO The DELETE statement is used to delete records from a

table: DELETE FROM table_name WHERE some_column =

some_value

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";

if ($conn->query($sql) === TRUE) { echo
  "Record deleted successfully";
}
else{
echo "Error deleting record: " . $conn->error;
    }

    $conn->close();
    ?>
```

Update Data In a MySQL Table Using MySQL

The UPDATE statement is used to update existing records in a table:

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "UPDATE MyGuests SET lastname='Doe' WHERE

id=2"; if ($conn->query($sql) === TRUE) {
  echo "Record updated successfully";
} else {
  echo "Error updating record: " . $conn->error;
}

$conn->close();
?>
```

Short questions

1.What are the features of php?
2.Write the applications of php?
3.What are the loops in php?
4.What are the php arrays?

Long questions

1.Write about array functions?
2.How to run php?
3.Write about date and time functions in php?
4.Write about MySQL and its features?

UNIT-5

Servlets:

Java Servlets are programs that run on a Web or Application server and act as a middle layer between a requests coming from a Web browser or other HTTP client and databases or applications on the HTTP server.

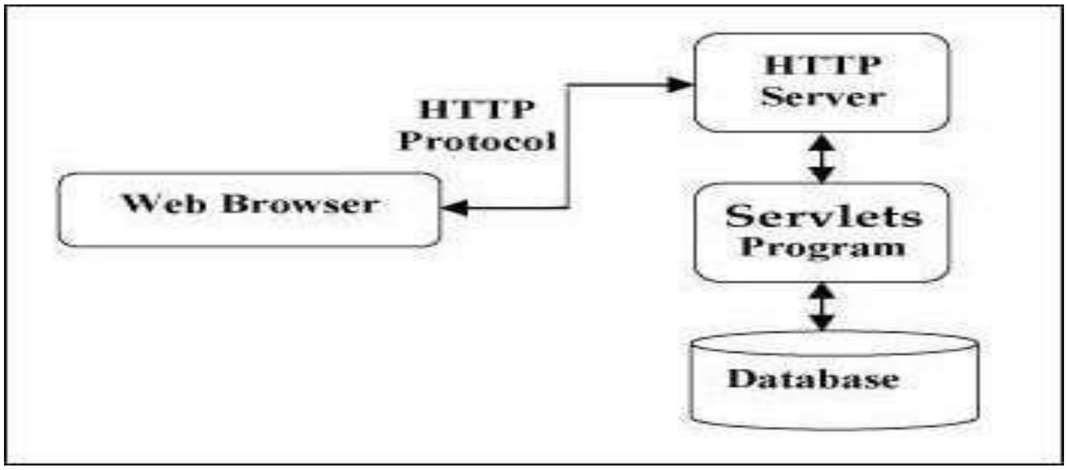
Using Servlets, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.

Java Servlets often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But Servlets offer several advantages in comparison with the CGI.

- Performance is significantly better.
- Servlets execute within the address space of a Web server. It is not necessary to create a separate process to handle each client request.
- Servlets are platform-independent because they are written in Java.
- Java security manager on the server enforces a set of restrictions to protect the resources on a server machine. So servlets are trusted.
- The full functionality of the Java class libraries is available to a servlet. It can communicate with applets, databases, or other software via the sockets and RMI mechanisms that you have seen already.

Servlets Architecture

The following diagram shows the position of Servlets in a Web Application.



Servlets Tasks

Servlets perform the following major tasks −

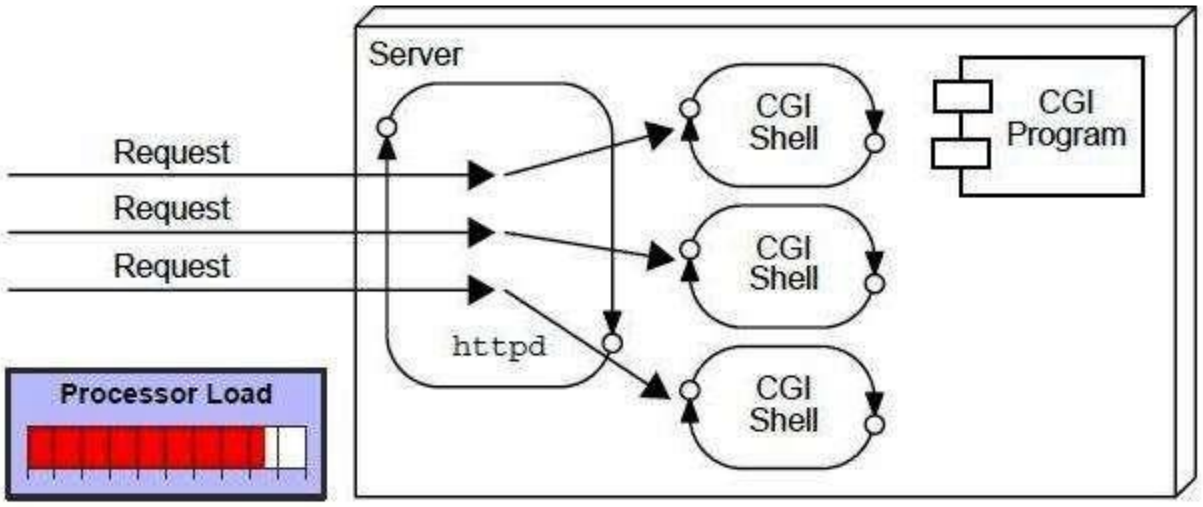
- Read the explicit data sent by the clients (browsers). This includes an HTML form on a Web page or it could also come from an applet or a custom HTTP client program.
- Read the implicit HTTP request data sent by the clients (browsers). This includes cookies, media types and compression schemes the browser understands, and so forth.
- Process the data and generate the results. This process may require talking to a database, executing an RMI or CORBA call, invoking a Web service, or computing the response directly.
- Send the explicit data (i.e., the document) to the clients (browsers). This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), Excel, etc.
- Send the implicit HTTP response to the clients (browsers). This includes telling the browsers or other clients what type of document is being returned (e.g., HTML), setting cookies and caching parameters, and other such tasks.

What is web application?

A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter etc. and other components such as HTML. The web components typically execute in Web Server and respond to HTTP request.

CGI(Commmon Gateway Interface)

CGI technology enables the web server to call an external program and pass HTTP request information to the external program to process the request. For each request, it starts a new process.

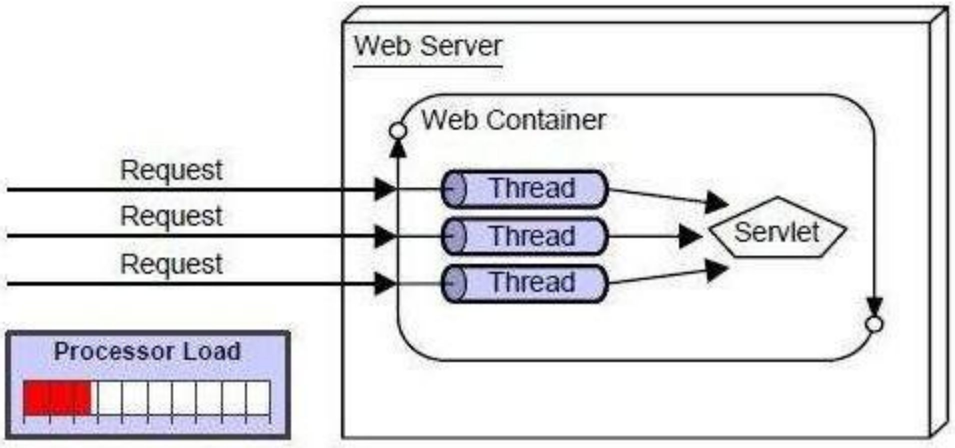


Disadvantages of CGI

There are many problems in CGI technology:

1. If number of clients increases, it takes more time for sending response.
2. For each request, it starts a process and Web server is limited to start processes.
3. It uses platform dependent language e.g. C, C++, perl.

Advantage of Servlet



There are many advantages of servlet over CGI. The web container creates threads for handling the multiple requests to the servlet. Threads have a lot of benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low. The basic benefits of servlet are as follows:

1. **better performance:** because it creates a thread for each request not process.
2. **Portability:** because it uses java language.
3. **Robust:** Servlets are managed by JVM so we don't need to worry about memory leak, garbage collection etc.
4. **Secure:** because it uses java language..

Website

Website is a collection of related web pages that may contain text, images, audio and video. The first page of a website is called home page. Each website has specific internet address (URL) that you need to enter in your browser to access a website.

Website is hosted on one or more servers and can be accessed by visiting its homepage using a computer network. A website is managed by its owner that can be an individual, company or an organization.

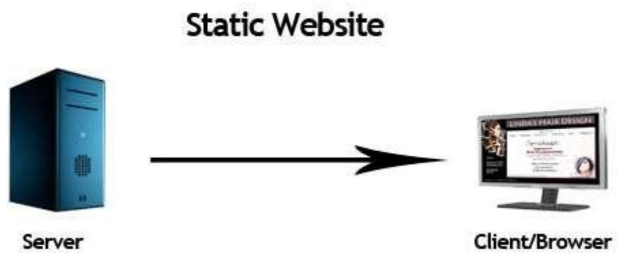
A website can be of two types:

- Static Website
- Dynamic Website

Static website

Static website is the basic type of website that is easy to create. You don't need web programming and database design to create a static website. Its web pages are coded in HTML.

The codes are fixed for each page so the information contained in the page does not change and it looks like a printed page.



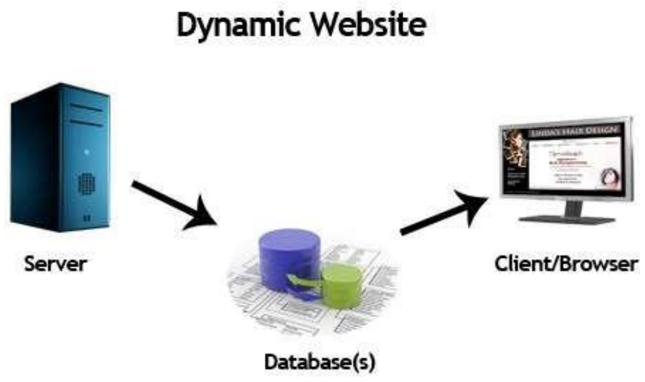
Dynamic website

Dynamic website is a collection of dynamic web pages whose content changes dynamically. It accesses content from a database or Content Management System (CMS). Therefore, when you alter or update the content of the database, the content of the website is also altered or updated.

Dynamic website uses client-side scripting or server-side scripting, or both to generate dynamic content.

Client side scripting generates content at the client computer on the basis of user input. The web browser downloads the web page from the server and processes the code within the page to render information to the user.

In server side scripting, the software runs on the server and processing is completed in the server then plain pages are sent to the user.



Static vs Dynamic website

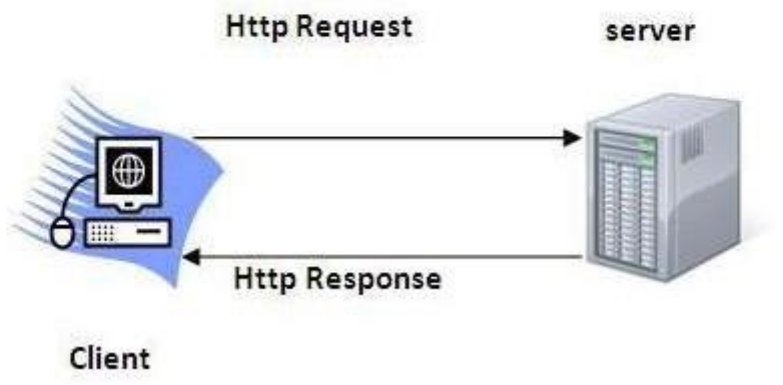
Static Website	**Dynamic Website**
Prebuilt content is same every time the pageis loaded.	Content is generated quickly and changes regularly.
It uses the **HTML** code for developing a website.	It uses the server side languages suchas **PHP,SERVLET, JSP, and ASP.NET** etc. for developing a website.

It sends exactly the same response for every request.	It may generate different HTML for each ofthe request.
The content is only changes when someone publishes and updates the file (sends it to the web server).	The page contains "server-side" code it allows the server to generate the unique content when the page is loaded.
Flexibility is the main advantage of staticwebsite.	Content Management System (CMS) is themain advantage of dynamic website.

HTTP (Hyper Text Transfer Protocol)

The Hypertext Transfer Protocol (HTTP) is application-level protocol for collaborative, distributed, hypermedia information systems. It is the data communication protocol used to establish communication between client and server.

HTTP is TCP/IP based communication protocol, which is used to deliver the data like image files, query results, HTML files etc on the World Wide Web (WWW) with the default port is TCP 80. It provides the standardized way for computers to communicate with each other.



The Basic Characteristics of HTTP (Hyper Text Transfer Protocol):

- It is the protocol that allows web servers and browsers to exchange data over the web.
- It is a request response protocol.
- It uses the reliable TCP connections by default on TCP port 80.
- It is stateless means each request is considered as the new request. In other words, server doesn't recognize the user by default.

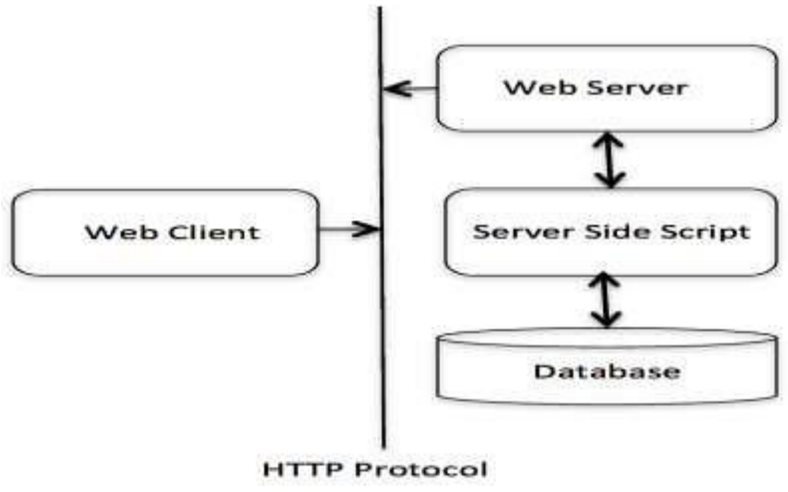
The Basic Features of HTTP (Hyper Text Transfer Protocol):

There are three fundamental features that make the HTTP a simple and powerful protocol used for communication:

- **HTTP is media independent:** It refers to any type of media content can be sent by HTTP as long as both the server and the client can handle the data content.
- **HTTP is connectionless:** It is a connectionless approach in which HTTP client i.e., a browser initiates the HTTP request and after the request is sends the client disconnects from server and waits for the response.
- **HTTP is stateless:** The client and server are aware of each other during a current request only. Afterwards, both of them forget each other. Due to the stateless nature of protocol, neither the client nor the server can retain the information about different request across the web pages.

The Basic Architecture of HTTP (Hyper Text Transfer Protocol):

The below diagram represents the basic architecture of web application and depicts where HTTP stands:



HTTP is request/response protocol which is based on client/server based architecture. In this web browser, search engines, etc behaves as a HTTP clients, and the Web server like Servlet behaves as a server.

The HTTP request methods are:

HTTP Request	Description

GET	Asks to get the resource at the requested URL.
POST	Asks the server to accept the body info attached. It is like GET requestwith extra info sent with the request.
HEAD	Asks for only the header part of whatever a GET would return. Justlike GET but with no body.
TRACE	Asks for the loopback of the request message, for testing ortroubleshooting.
PUT	Says to put the enclosed info (the body) at the requested URL.
DELETE	Says to delete the resource at the requested URL.
OPTIONS	Asks for a list of the HTTP methods to which the thing at the requestURL can respond

Get vs. Post

There are many differences between the Get and Post request. Let's see these differences:

GET	POST
1) In case of Get request, only **limited amount of data** can be sent because data issent in header.	In case of post request, **large amount of data** can be sent because data is sent in body.
2) Get request is **not secured** because data isexposed in URL bar.	Post request is **secured** becausedata is not exposed in URL bar.
3) Get request **can be bookmarked.**	Post request **cannot be bookmarked.**

4) Get request is **idempotent** . It means second request will be ignored until response of first request is delivered	Post request is **non-idempotent.**
5) Get request is **more efficient** and usedmore than Post.	Post request is **less efficient** andused less than get.

GET and POST

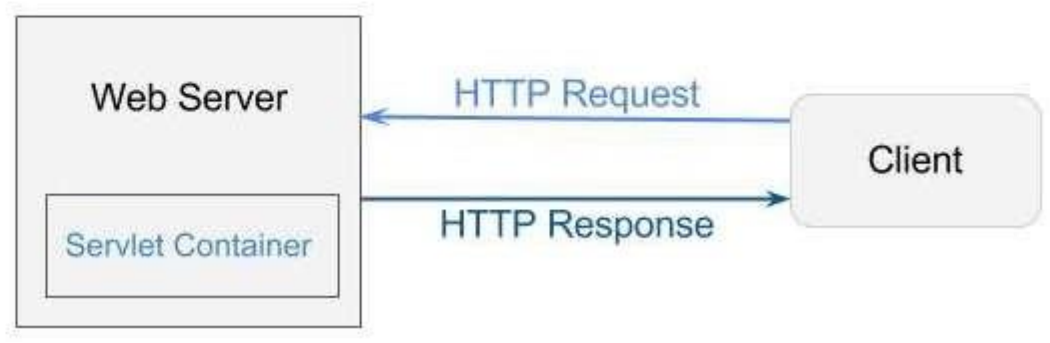
Two common methods for the request-response between a server and client are:

- **GET**- It requests the data from a specified resource
- **POST**- It submits the processed data to a specified resource

Servlet Container

It provides the runtime environment for JavaEE (j2ee) applications. The client/user can request only a static WebPages from the server. If the user wants to read the web pages as per input then the servlet container is used in java.

The servlet container is used in java for dynamically generate the web pages on the server side. Therefore the servlet container is the part of a web server that interacts with the servlet for handling the dynamic web pages from the client.



Servlet Container States

The servlet container is the part of web server which can be run in a separate process. We can classify the servlet container states in three types:

- **Standalone:** It is typical Java-based servers in which the servlet container and the web servers are the integral part of a single program. For example:- Tomcat running by itself
- **In-process:** It is separated from the web server, because a different program is runs within the address space of the main server as a plug-in. For example:- Tomcat running inside the JBoss.
- **Out-of-process:** The web server and servlet container are different programs which are run in a different process. For performing the communications between them, web server uses the plug-in provided by the servlet container.

The Servlet Container performs many operations that are given below:

- Life Cycle Management
- Multithreaded support
- Object Pooling
- Security etc.

Servlet API

1. Servlet API
2. Interfaces in javax.servlet package
3. Classes in javax.servlet package
4. Interfaces in javax.servlet.http package
5. Classes in javax.servlet.http package

The javax.servlet and javax.servlet.http packages represent interfaces and classes for servlet api.

The **javax.servlet** package contains many interfaces and classes that are used by the servlet or web container. These are not specific to any protocol.

The **javax.servlet.http** package contains interfaces and classes that are responsible for http requests only.

Let's see what are the interfaces of javax.servlet package.

Interfaces in javax.servlet package

There are many interfaces in javax.servlet package. They are as follows:

1. Servlet
2. ServletRequest
3. ServletResponse
4. RequestDispatcher
5. ServletConfig
6. ServletContext
7. SingleThreadModel
8. Filter
9. FilterConfig
10. FilterChain
11. ServletRequestListener
12. ServletRequestAttributeListener
13. ServletContextListener
14. ServletContextAttributeListener

Classes in javax.servlet package

There are many classes in javax.servlet package. They are as follows:

1. Generic Servlet
2. Servlet Input Stream
3. Servlet Output Stream
4. Servlet Request Wrapper
5. Servlet Response Wrapper
6. Servlet Request Event
7. Servlet Context Event
8. Servlet Request Attribute Event
9. Servlet Context Attribute Event
10. Servlet Exception
11. Unavailable Exception

Interfaces in javax.servlet.http package

There are many interfaces in javax.servlet.http package. They are as follows:

1. HttpServletRequest
2. HttpServletResponse
3. HttpSession
4. HttpSessionListener
5. HttpSessionAttributeListener
6. HttpSessionBindingListener
7. HttpSessionActivationListener
8. HttpSessionContext (deprecated now)

Classes in javax.servlet.http package

There are many classes in javax.servlet.http package. They are as follows:

1. HttpServlet
2. Cookie
3. HttpServletRequestWrapper
4. HttpServletResponseWrapper
5. HttpSessionEvent
6. HttpSessionBindingEvent
7. HttpUtils (deprecated now)

Servlet LifeCycle

A servlet life cycle can be defined as the entire process from its creation till the destruction. The following are the paths followed by a servlet.

- The servlet is initialized by calling the **init()** method.
- The servlet calls **service()** method to process a client's request.
- The servlet is terminated by calling the **destroy()** method.
- Finally, servlet is garbage collected by the garbage collector of the JVM.

Now let us discuss the life cycle methods in detail.

The init() Method

The init method is called only once. It is called only when the servlet is created, and not called for any user requests afterwards. So, it is used for one-time initializations, just as with the init method of applets.

The servlet is normally created when a user first invokes a URL corresponding to the servlet, but you can also specify that the servlet be loaded when the server is first started.

When a user invokes a servlet, a single instance of each servlet gets created, with each user request resulting in a new thread that is handed off to doGet or doPost as appropriate. The init() method simply creates or loads some data that will be used throughout the life of the servlet.

The init method definition looks like this −

```
public void init() throws ServletException {

   // Initialization code...

}
```

The service() Method

The service() method is the main method to perform the actual task. The servlet container (i.e. web server) calls the service() method to handle requests coming from the client(browsers) and to write the formatted response back to the client.

Each time the server receives a request for a servlet, the server spawns a new thread and calls service. The service() method checks the HTTP request type (GET, POST, PUT, DELETE, etc.) and calls doGet, doPost, doPut, doDelete, etc. methods as appropriate.

Here is the signature of this method −

```
public void service(ServletRequest request, ServletResponse response)

   throws ServletException, IOException {

}
```

The service () method is called by the container and service method invokes doGe, doPost, doPut, doDelete, etc. methods as appropriate. So you have nothing to do with service() method but you override either doGet() or doPost() depending on what type of request you receive from the client.

The doGet() and doPost() are most frequently used methods with in each service request. Here is the signature of these two methods.

The doGet() Method

A GET request results from a normal request for a URL or from an HTML form that has no METHOD specified and it should be handled by doGet() method.

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
  throws ServletException, IOException {
  // Servlet code
}
```

The doPost() Method

A POST request results from an HTML form that specifically lists POST as the METHOD and it should be handled by doPost() method.

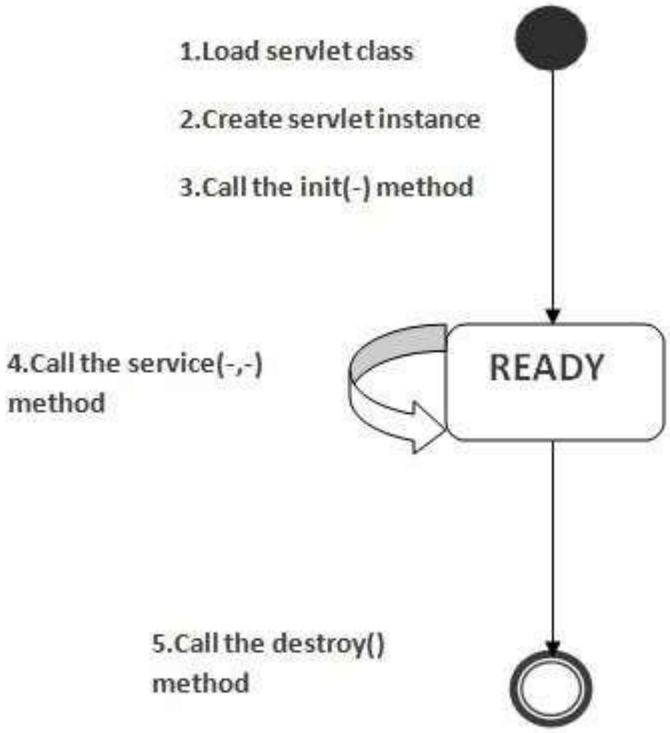
```
public void doPost(HttpServletRequest request, HttpServletResponse response)
  throws ServletException, IOException {
  // Servlet code
}
```

The destroy() Method

The destroy() method is called only once at the end of the life cycle of a servlet. This method gives your servlet a chance to close database connections, halt background threads, write cookie lists or hit counts to disk, and perform other such cleanup activities.

After the destroy() method is called, the servlet object is marked for garbage collection. The destroy method definition looks like this −

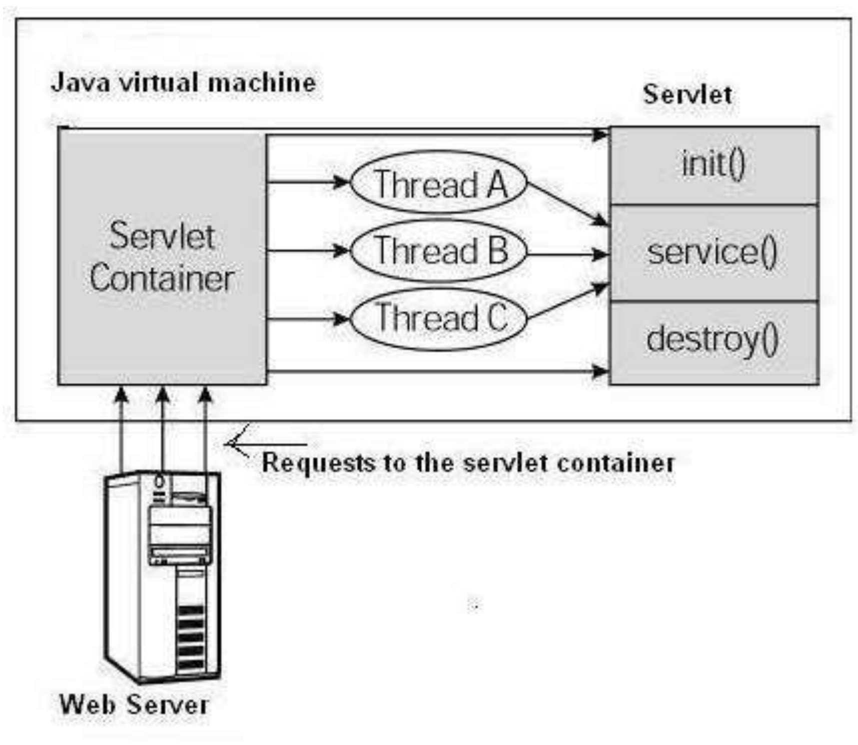
```
public void destroy() {
  // Finalization code...
}
```


1.Load servlet class
2.Create servlet instance
3.Call the init(-) method
4.Call the service(-,-) method
READY
5.Call the destroy() method

Architecture Diagram

The following figure depicts a typical servlet life-cycle scenario.

- First the HTTP requests coming to the server are delegated to the servlet container.
- The servlet container loads the servlet before invoking the service() method.
- Then the servlet container handles multiple requests by spawning multiple threads, each thread executing the service() method of a single instance of the servlet.



Servlet Interface

1. Servlet Interface
2. Methods of Servlet interface

Servlet interface provides common behaviour to all the servlets.

Servlet interface needs to be implemented for creating any servlet (either directly or indirectly). It provides 3 life cycle methods that are used to initialize the servlet, to service the requests, and to destroy the servlet and 2 non-life cycle methods.

Methods of Servlet interface

There are 5 methods in Servlet interface. The init, service and destroy are the life cycle methods of servlet. These are invoked by the web container.

Method	Description
public void init(ServletConfig config)	initializes the servlet. It is the life cycle method of servlet and invoked by the web container only once.

public void service(ServletRequest request,ServletResponseresponse)	provides response for the incoming request. It is invoked at each request by the web container.
public void destroy()	is invoked only once and indicates that servletis being destroyed.
public ServletConfig getServletConfig()	returns the object of ServletConfig.
public String getServletInfo()	returns information about servlet such aswriter, copyright, version etc

GenericServlet class

1. GenericServlet class
2. Methods of GenericServlet class
3. Example of GenericServlet class

GenericServlet class implements **Servlet**, **ServletConfig** and **Serializable**interfaces. It provides the implementation of all the methods of these interfaces except the service method.

GenericServlet class can handle any type of request so it is protocol-independent.

You may create a generic servlet by inheriting the GenericServlet class and providing the implementation of the service method.

Methods of GenericServlet class

There are many methods in GenericServlet class. They are as follows:

1. **public void init(ServletConfig config)** is used to initialize the servlet.
2. **public abstract void service(ServletRequest request, ServletResponse response)** provides service for the incoming request. It is invoked at each time when user requests for a servlet.
3. **public void destroy()** is invoked only once throughout the life cycle and indicates that servlet is being destroyed.

4. **public ServletConfig getServletConfig()** returns the object of ServletConfig.
5. **public String getServletInfo()** returns information about servlet such as writer, copyright, version etc.
6. **public void init()** it is a convenient method for the servlet programmers, now there is no need to call super.init(config)
7. **public ServletContext getServletContext()** returns the object of ServletContext.
8. **public String getInitParameter(String name)** returns the parameter value for the given parameter name.
9. **public Enumeration getInitParameterNames()** returns all the parameters defined in the web.xml file.
10. **public String getServletName()** returns the name of the servlet object.
11. **public void log(String msg)** writes the given message in the servlet log file.
12. **public void log(String msg,Throwable t)** writes the explanatory message in the servlet log file and a stack trace.

HttpServlet class

1. HttpServlet class
2. Methods of HttpServlet class

The HttpServlet class extends the GenericServlet class and implements Serializable interface. It provides http specific methods such as doGet, doPost, doHead, doTrace etc.

Methods of HttpServlet class

There are many methods in HttpServlet class. They are as follows:

1. **public void service(ServletRequest req,ServletResponse res)** dispatches the request to the protected service method by converting the request and response object into http type.
2. **protected void service(HttpServletRequest req, HttpServletResponse res)** receives the request from the service method, and dispatches the request to the doXXX() method depending on the incoming http request type.
3. **protected void doGet(HttpServletRequest req, HttpServletResponse res)** handles the GET request. It is invoked by the web container.
4. **protected void doPost(HttpServletRequest req, HttpServletResponse res)** handles the POST request. It is invoked by the web container.

5. **protected void doHead(HttpServletRequest req, HttpServletResponse res)** handles the HEAD request. It is invoked by the web container.
6. **protected void doOptions(HttpServletRequest req, HttpServletResponse res)** handles the OPTIONS request. It is invoked by the web container.
7. **protected void doPut(HttpServletRequest req, HttpServletResponse res)** handles the PUT request. It is invoked by the web container.
8. **protected void doTrace(HttpServletRequest req, HttpServletResponse res)** handles the TRACE request. It is invoked by the web container.
9. **protected void doDelete(HttpServletRequest req, HttpServletResponse res)** handles the DELETE request. It is invoked by the web container.
10. **protected long getLastModified(HttpServletRequest req)** returns the time when HttpServletRequest was last modified since midnight January 1, 1970 GMT.

Steps to Deploy a servlet example

1. Steps to create the servlet using Tomcat server

1. Create a directory structure
2. Create a Servlet
3. Compile the Servlet
4. Create a deployment descriptor
5. Start the server and deploy the application

There are given 6 steps to create a **servlet example**. These steps are required for all the servers.

The servlet example can be created by three ways:

1. By implementing Servlet interface,
2. By inheriting GenericServlet class, (or)
3. By inheriting HttpServlet class

The mostly used approach is by extending HttpServlet because it provides http request specific method such as doGet(), doPost(), doHead() etc.

Here, we are going to use **apache tomcat server** in this example. The steps are as follows:

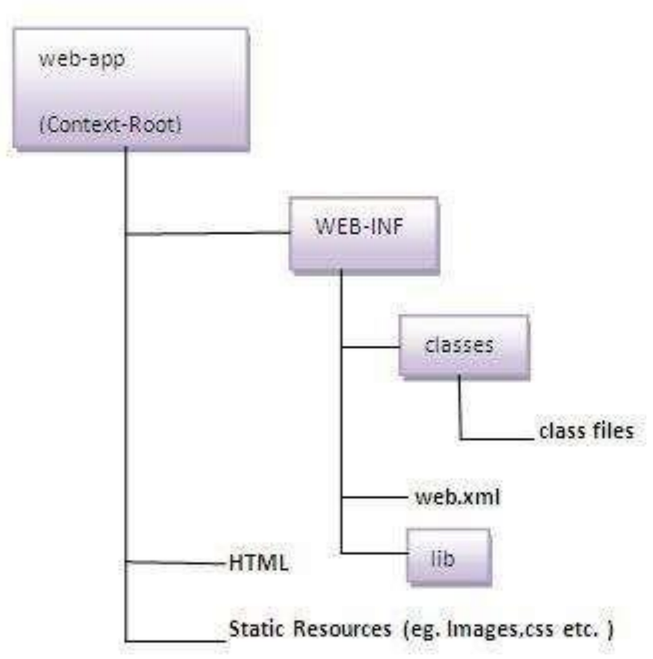
1. Create a directory structure
2. Create a Servlet

3. Compile the Servlet
4. Create a deployment descriptor
5. Start the server and deploy the project

1)Create a directory structures

The **directory structure** defines that where to put the different types of files so that web container may get the information and respond to the client.

The Sun Microsystem defines a unique standard to be followed by all the server vendors. Let's see the directory structure that must be followed to create the servlet.



As you can see that the servlet class file must be in the classes folder. The web.xml file must be under the WEB-INF folder.

---

2) Create a Servlet

There are three ways to create the servlet.

1. By implementing the Servlet interface

2. By inheriting the GenericServlet class
3. By inheriting the HttpServlet class

The HttpServlet class is widely used to create the servlet because it provides methods tohandle http requests such as doGet(), doPost, doHead() etc.

In this example we are going to create a servlet that extends the HttpServlet class. In this example, we are inheriting the HttpServlet class and providing the implementation ofthe doGet() method. Notice that get request is the default request.

DemoServlet.java

```java
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
public class DemoServlet extends HttpServlet{
public void doGet(HttpServletRequest req,HttpServletResponse res)
throws ServletException,IOException
{
res.setContentType("text/html");//setting the content type
PrintWriter pw=res.getWriter();//get the stream to write the data

//writing html in the stream
pw.println("<html><body>");
pw.println("Welcome to servlet");
pw.println("</body></html>");

pw.close();//closing the stream
}}
```

3) Compile the servlet

For compiling the Servlet, jar file is required to be loaded. Different Servers provide different jar files:

Jar file	Server
1) servlet-api.jar	Apache Tomcat

2) weblogic.jar	Weblogic
3) javaee.jar	Glassfish
4) javaee.jar	JBoss

Two ways to load the jar file

1. set classpath
2. paste the jar file in JRE/lib/ext folder

Put the java file in any folder. After compiling the java file, paste the class file of servlet in **WEB-INF/classes** directory.

---

4) Create the deployment descriptor (web.xml file)

The **deployment descriptor** is an xml file, from which Web Container gets the information about the servet to be invoked.

The web container uses the Parser to get the information from the web.xml file. There are many xml parsers such as SAX, DOM and Pull.

There are many elements in the web.xml file. Here is given some necessary elements to run the simple servlet program.

web.xml file

```
<web-app>

<servlet>
<servlet-name>sonoojaiswal</servlet-name>
<servlet-class>DemoServlet</servlet-class>
</servlet>

<servlet-mapping>
<servlet-name>sonoojaiswal</servlet-name>
<url-pattern>/welcome</url-pattern>
</servlet-mapping>
```

12.
13.</web-app>

Description of the elements of web.xml file

There are too many elements in the web.xml file. Here is the illustration of some elements that is used in the above web.xml file. The elements are as follows:

<web-app> represents the whole application.

<servlet> is sub element of <web-app> and represents the servlet.

<servlet-name> is sub element of <servlet> represents the name of the servlet.

<servlet-class> is sub element of <servlet> represents the class of the servlet.

<servlet-mapping> is sub element of <web-app>. It is used to map the servlet.

<url-pattern> is sub element of <servlet-mapping>. This pattern is used at client sideto invoke the servlet.

---

5) Start the Server and deploy the project

To start Apache Tomcat server, double click on the startup.bat file under apache-tomcat/bin directory. In URL type the url to view the servlet.

Session Tracking in Servlets

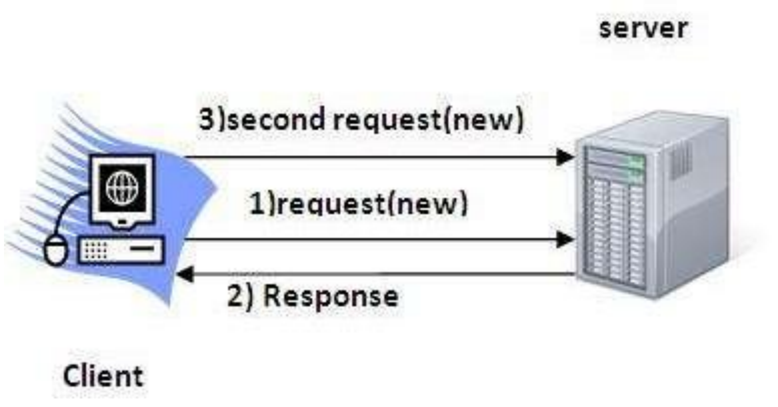
1. Session Tracking
2. Session Tracking Techniques

Session simply means a particular interval of time.

Session Tracking is a way to maintain state (data) of an user. It is also known as **session management** in servlet.

Http protocol is a stateless so we need to maintain state using session tracking techniques. Each time user requests to the server, server treats the request as the new request. So we need to maintain the state of an user to recognize to particular user.

HTTP is stateless that means each request is considered as the new request. It is shown in the figure given below:



Why use Session Tracking?

To recognize the user It is used to recognize the particular user.

Session Tracking Techniques

There are four techniques used in Session tracking:

1. **Cookies**
2. **Hidden Form Field**
3. **URL Rewriting**
4. **HttpSession**

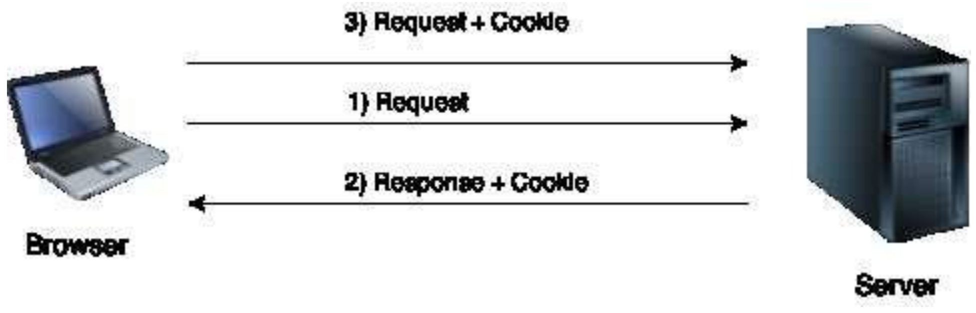
Cookies in Servlet

A **cookie** is a small piece of information that is persisted between the multiple client requests.

A cookie has a name, a single value, and optional attributes such as a comment, path and domain qualifiers, a maximum age, and a version number.

How Cookie works

By default, each request is considered as a new request. In cookies technique, we add cookie with response from the servlet. So cookie is stored in the cache of the browser. After that if request is sent by the user, cookie is added with request by default. Thus, we recognize the user as the old user.



Types of Cookie

There are 2 types of cookies in servlets.

1. Non-persistent cookie
2. Persistent cookie

Non-persistent cookie

It is **valid for single session** only. It is removed each time when user closes the browser.

Persistent cookie

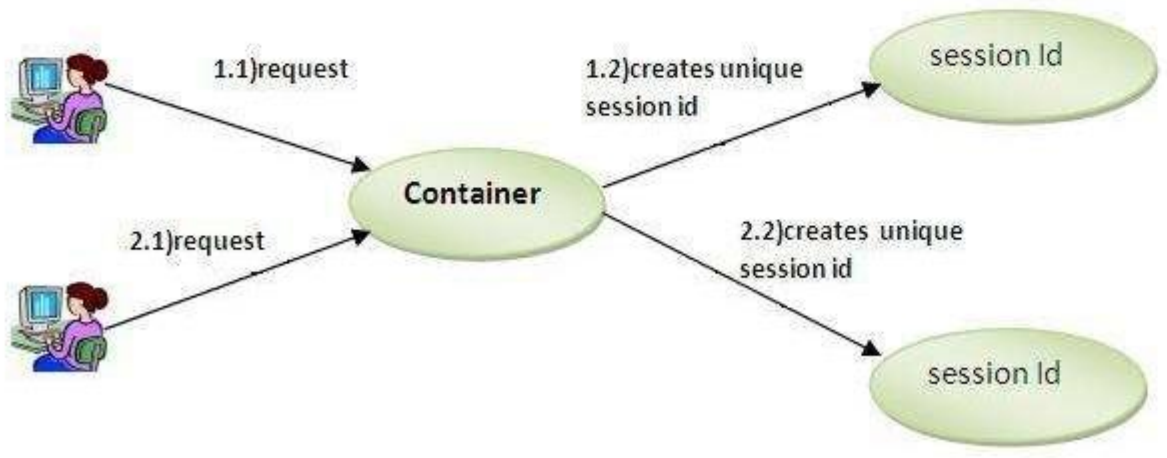
It is **valid for multiple session** . It is not removed each time when user closes the browser. It is removed only if user logout or signout.

4) HttpSession interface

1. HttpSession interface
2. How to get the HttpSession object
3. Commonly used methods of HttpSession interface
4. Example of using HttpSession

In such case, container creates a session id for each user.The container uses this id to identify the particular user.An object of HttpSession can be used to perform two tasks:

1. bind objects
2. view and manipulate information about a session, such as the session identifier, creation time, and last accessed time.



How to get the Http Session object ?

The Http Servlet Request interface provides two methods to get the object of HttpSession:

1. **public Http Session get Session():**Returns the current session associated with this request, or if the request does not have a session, creates one.
2. **public Http Session get Session(boolean create):**Returns the current HttpSessionassociated with this request or, if there is no current session and create is true, returnsa new session.

Commonly used methods of HttpSession interface

1. **public String getId():**Returns a string containing the unique identifier value.
2. **public long get Creation Time():**Returns the time when this session was created, measured in milliseconds since midnight January 1, 1970 GMT.
3. **public long get Last Accessed Time():**Returns the last time the client sent a requestassociated with this session, as the number of milliseconds since midnight January 1,1970 GMT.
4. **public void invalidate():**Invalidates this session then unbinds any objects bound to it.

3) URL Rewriting

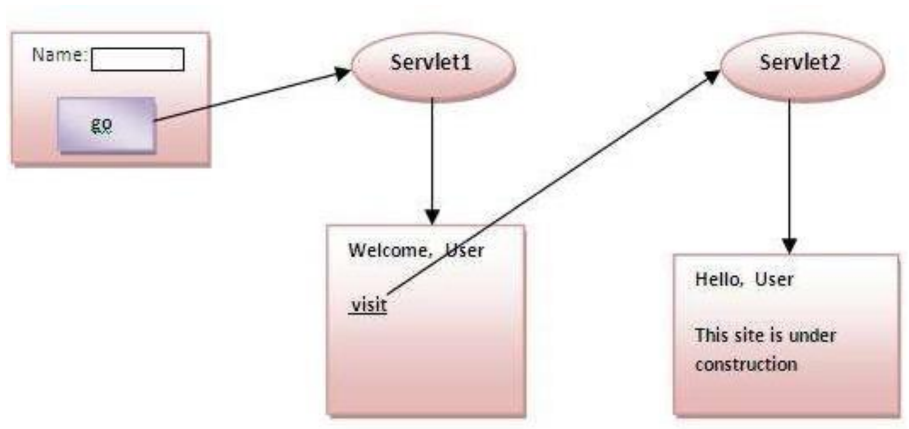
1. URL Rewriting
2. Advantage of URL Rewriting
3. Disadvantage of URL Rewriting

4. Example of URL Rewriting

In URL rewriting, we append a token or identifier to the URL of the next Servlet or the next resource. We can send parameter name/value pairs using the following format:

url?name1=value1&name2=value2&??

A name and a value is separated using an equal = sign, a parameter name/value pair is separated from another parameter using the ampersand(&). When the user clicks the hyperlink, the parameter name/value pairs will be passed to the server. From a Servlet, we can use getParameter() method to obtain a parameter value.



Advantage of URL Rewriting

1. It will always work whether cookie is disabled or not (browser independent).
2. Extra form submission is not required on each pages.

Disadvantage of URL Rewriting

1. It will work only with links.
2. It can send Only textual information

Java Server Pages

Java Server Pages (JSP) is a technology for developing Webpages that supports dynamic content. This helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.

A JavaServer Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.

Using JSP, you can collect input from users through Webpage forms, present records from a database or another source, and create Webpages dynamically.

JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages, and sharing information between requests, pages etc.

Why Use JSP?

JavaServer Pages often serve the same purpose as programs implemented using the **Common Gateway Interface (CGI)**. But JSP offers several advantages in comparison with the CGI.

- Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having separate CGI files.
- JSP are always compiled before they are processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is requested.
- JavaServer Pages are built on top of the Java Servlets API, so like Servlets, JSP also has access to all the powerful Enterprise Java APIs, including **JDBC, JNDI, EJB, JAXP,** etc.
- JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

Finally, JSP is an integral part of Java EE, a complete platform for enterprise class applications. This means that JSP can play a part in the simplest applications to the most complex and demanding.

Advantages of JSP

Following table lists out the other advantages of using JSP over other technologies −

vs. Active Server Pages (ASP)

The advantages of JSP are twofold. First, the dynamic part is written in Java, not Visual Basic or other MS specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and non-Microsoft Web servers.

vs. Pure Servlets

It is more convenient to write (and to modify!) regular HTML than to have plenty of println statements that generate the HTML.

vs. Server-Side Includes (SSI)

SSI is really only intended for simple inclusions, not for "real" programs that use form data, make database connections, and the like.

vs. JavaScript

JavaScript can generate HTML dynamically on the client but can hardly interact with the web server to perform complex tasks like database access and image processing etc.

vs. Static HTML

Regular HTML, of course, cannot contain dynamic information.

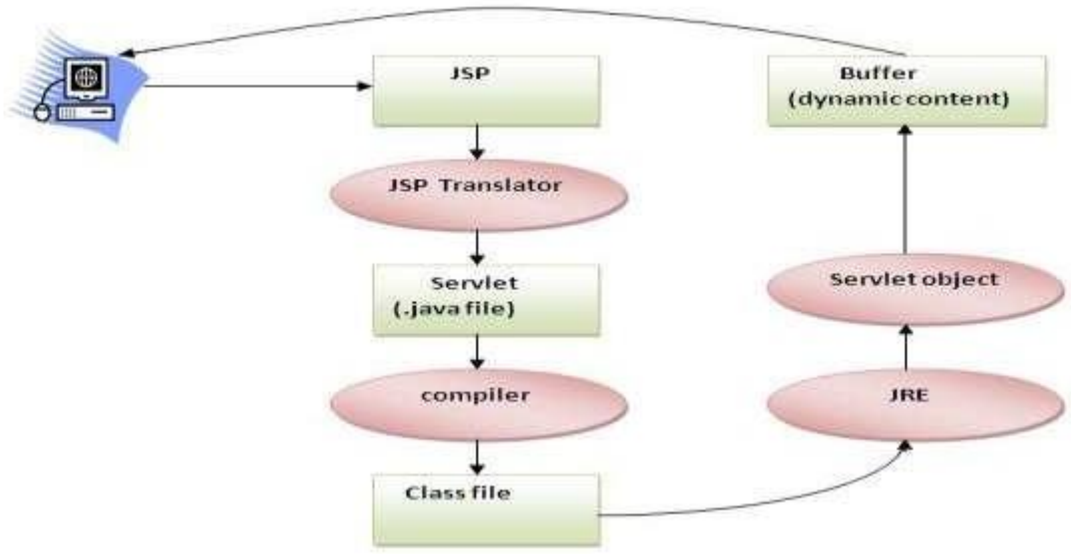
Life cycle of a JSP Page

The JSP pages follows these phases:

- Translation of JSP Page
- Compilation of JSP Page
- Classloading (class file is loaded by the classloader)
- Instantiation (Object of the Generated Servlet is created).

- Initialization (jspInit() method is invoked by the container).
- Reqeust processing (_jspService() method is invoked by the container).
- Destroy (jspDestroy() method is invoked by the container).

Note: jspInit(), _jspService() and jspDestroy() are the life cycle methods of JSP.



As depicted in the above diagram, JSP page is translated into servlet by the help of JSP translator. The JSP translator is a part of webserver that is responsible to translate the JSP page into servlet. Afterthat Servlet page is compiled by the compiler and gets converted into the class file. Moreover, all the processes that happens in servlet is performed on JSP later like initialization, committing response to the browser and destroy.

Creating a simple JSP Page

To create the first jsp page, write some html code as given below, and save it by .jsp extension. We have save this file as index.jsp. Put it in a folder and paste the folder in the web-apps directory in apache tomcat to run the jsp page.

index.jsp

Let's see the simple example of JSP, here we are using the scriptlet tag to put java code in the JSP page. We will learn scriptlet tag later.

```
<html>
<body>
<% out.print(2*5); %>
</body>
```

5. </html>

It will print **10** on the browser.

How to run a simple JSP Page ?

Follow the following steps to execute this JSP page:

- Start the server
- put the jsp file in a folder and deploy on the server
- visit the browser by the url http://localhost:portno/contextRoot/jspfile e.g. http://localhost:8888/myapplication/index.jsp

The JSP API

1. The JSP API
2. javax.servlet.jsp package
3. The JspPage interface
4. The HttpJspPage interface

The JSP API consists of two packages:

1. javax.servlet.jsp
2. javax.servlet.jsp.tagext

javax.servlet.jsp package

The javax.servlet.jsp package has two interfaces and classes.The two interfaces are as follows:

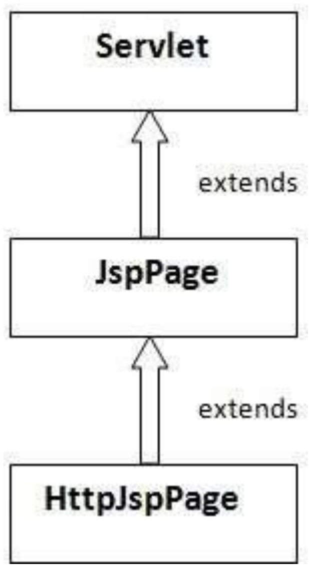
1. JspPage
2. HttpJspPage

The classes are as follows:

- JspWriter
- PageContext
- JspFactory
- JspEngineInfo
- JspException,JspError

The JspPage interface

According to the JSP specification, all the generated servlet classes must implement the JspPage interface. It extends the Servlet interface. It provides two life cycle methods.



Methods of JspPage interface

1. **public void jspInit():** It is invoked only once during the life cycle of the JSP when JSP page is requested firstly. It is used to perform initialization. It is same as the init() method of Servlet interface.
2. **public void jspDestroy():** It is invoked only once during the life cycle of the JSP before the JSP page is destroyed. It can be used to perform some clean up operation.

The HttpJspPage interface

The HttpJspPage interface provides the one life cycle method of JSP. It extends the JspPage interface.

Method of HttpJspPage interface:

1. **public void _jspService():** It is invoked each time when request for the JSP page comes to the container. It is used to process the request. The underscore _ signifies that you cannot override this method.

JSP Scriptlet tag (Scripting elements)

1. Scripting elements
2. JSP scriptlet tag
3. Simple Example of JSP scriptlet tag
4. Example of JSP scriptlet tag that prints the user name

In JSP, java code can be written inside the jsp page using the scriptlet tag. Let's see what are the scripting elements first.

JSP Scripting elements

The scripting elements provides the ability to insert java code inside the jsp. There are three types of scripting elements:

- scriptlet tag
- expression tag
- declaration tag

JSP scriptlet tag

A scriptlet tag is used to execute java source code in JSP. Syntax is as follows:

```
<%  java source code %>
```

Example of JSP scriptlet tag

In this example, we are displaying a welcome message.

```
<html>
<body>
<% out.print("welcome to jsp"); %>
</body>
```

JSP expression tag

The code placed within **JSP expression tag** is *written to the output stream of the response*. So you need not write out.print() to write data. It is mainly used to print the values of variable or method.

Syntax of JSP expression tag

```
<%=  statement %>
```

Example of JSP expression tag

```
<html>
<body>
<%= "welcome to jsp" %>
</body>
</html>
```

JSP Declaration Tag

1. JSP declaration tag
2. Difference between JSP scriptlet tag and JSP declaration tag
3. Example of JSP declaration tag that declares field
4. Example of JSP declaration tag that declares method

The **JSP declaration tag** is used *to declare fields and methods*.

The code written inside the jsp declaration tag is placed outside the service() method of auto generated servlet.

So it doesn't get memory at each request.

Syntax of JSP declaration tag

The syntax of the declaration tag is as follows:

```
<%!  field or method declaration %>
```

```
<html>
<body>
<%! int data=50; %>
<%= "Value of the variable is:"+data %>
</body>
</html>
```

JSP Implicit Objects

1. JSP Implicit Objects

2. out implicit object
3. Example of out implicit object

There are **9 jsp implicit objects**. These objects are *created by the web container* that are available to all the jsp pages.

The available implicit objects are out, request, config, session, application etc.

A list of the 9 implicit objects is given below:

Object	Type
Out	JspWriter
Request	HttpServletRequest
Response	HttpServletResponse
Config	ServletConfig
Application	ServletContext
Session	HttpSession
pageContext	PageContext
Page	Object
Exception	Throwable

1) JSP out implicit object

For writing any data to the buffer, JSP provides an implicit object named out. It is the object of JspWriter. In case of servlet you need to write:

```
PrintWriter out=response.getWriter();
```

But in JSP, you don't need to write this code.

Example of out implicit object

In this example we are simply displaying date and time.

index.jsp

```
<html>
<body>
<% out.print("Today is:"+java.util.Calendar.getInstance().getTime()); %>
</body>
</html>
```

JSP request implicit object

The **JSP request** is an implicit object of type HttpServletRequest i.e. created for each jsp request by the web container. It can be used to get request information such as parameter, header information, remote address, server name, server port, content type, character encoding etc.

It can also be used to set, get and remove attributes from the jsp request scope.

Let's see the simple example of request implicit object where we are printing the name of the user with welcome message.

Example of JSP request implicit object

index.html

```
<form action="welcome.jsp">
<input type="text" name="uname">
<input type="submit" value="go"><br/>
</form>
```

welcome.jsp

```
<%
String name=request.getParameter("uname");
out.print("welcome "+name);
%>
```

3) JSP response implicit object

In JSP, response is an implicit object of type HttpServletResponse. The instance of HttpServletResponse is created by the web container for each jsp request.

It can be used to add or manipulate response such as redirect response to another resource, send error etc.

Let's see the example of response implicit object where we are redirecting the response to the Google.

Example of response implicit object

index.html

```
<form action="welcome.jsp">
<input type="text" name="uname">
<input type="submit" value="go"><br/>
</form>
```

welcome.jsp

```
<%
response.sendRedirect("http://www.google.com");
%>
```

4) JSP config implicit object

In JSP, config is an implicit object of type *ServletConfig*. This object can be used to get initialization parameter for a particular JSP page. The config object is created by the web container for each jsp page.

Generally, it is used to get initialization parameter from the web.xml file.

5) JSP application implicit object

In JSP, application is an implicit object of type *ServletContext*.

The instance of ServletContext is created only once by the web container when application or project is deployed on the server.

This object can be used to get initialization parameter from configuaration file (web.xml). It can also be used to get, set or remove attribute from the application scope.

This initialization parameter can be used by all jsp pages.

6) session implicit object

In JSP, session is an implicit object of type HttpSession.The Java developer can use this object to set,get or remove attribute or to get session information.

Example of session implicit object

index.html

```
<html>
<body>
<form action="welcome.jsp">
<input type="text" name="uname">
<input type="submit" value="go"><br/>
</form>
</body>
</html>
```

welcome.jsp

```
<html>
<body>
<%

String name=request.getParameter("uname");
out.print("Welcome "+name);

session.setAttribute("user",name);

<a href="second.jsp">second jsp page</a>

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

7) pageContext implicit object

In JSP, pageContext is an implicit object of type PageContext class.The pageContext object can be used to set,get or remove attribute from one of the following scopes:

- page
- request
- session
- application

In JSP, page scope is the default scope.

JSP directives

1. JSP directives
1. page directive
2. Attributes of page directive

The **jsp directives** are messages that tells the web container how to translate a JSP page into the corresponding servlet.

There are three types of directives:

- page directive
- include directive
- taglib directive

Syntax of JSP Directive

```
1. <%@ directive attribute="value" %>
```

JSP page directive

The page directive defines attributes that apply to an entire JSP page.

Syntax of JSP page directive

```
1. <%@ page attribute="value" %>
```

Jsp Include Directive

1. Include directive
2. Advantage of Include directive
3. Example of include directive

The include directive is used to include the contents of any resource it may be jsp file, html file or text file. The include directive includes the original content of the included resource at page translation time (the jsp page is translated only once so it will be better to include static resource).

Advantage of Include directive

Code Reusability

Syntax of include directive

```
<%@ include file="resourceName" %>
```

JSP Taglib directive

1. JSP Taglib directive
2. Example of JSP Taglib directive

The JSP taglib directive is used to define a tag library that defines many tags. We use the TLD (Tag Library Descriptor) file to define the tags. In the customtag section we will use this tag so it will be better to learn it in custom tag.

Syntax JSP Taglib directive

```
1. <%@ taglib uri="uriofthetaglibrary" prefix="prefixoftaglibrary" %>
```

Short questions

1.Write about servlets?
2.What is web applications?
3.What is website and its types?
4.What is servlet container?

Long questions

1.What is servlet architectuter?
2.Write about HTTP?
3.Write about servlet interface?
4.Write about java server pages?

Question papers for Sample:

S.V.U. COLLEGE OF COMMERCE MANAGEMENT AND COMPUTER SCIENCE :: TIRUPATHI
DEPARTMENT OF COMPUTER SCIENCE
Time:2hours INTERNAL EXAMINATIONS -I MAX MARKS:30

Section-A

Answer any five from the following 5*2=10m
1.Explain about web browser?
2.Explain about browser architecture?
3.Write about HTML-5 basic tags?
4.Write about HTML frames?
5.Discuss about javascript advantages?
6.Write about javascript coding syntax?
7.Write about javascript decision making statements?
8.Discuss about javascript loops?

Section-B

Answer any one Question from each unit 2*10=20marks

Unit-1

9. a) Write about HTML and its features?
b) Discuss about table tags?

(Or)

10. a) Write about HTML 5 forms and elements?
b) What is CSS and how they are linked with HTML?

Unit-2

11. a) Write about javascript datatypes and variables?
b) write about operators in javascript?

(OR)

12. a) Explain about javascript functions ?
b) Write about javascript objects?

S.V.U COLLEGE OF COMMERCE MANAGEMENT AND COMPUTER SCIENCE :: TIRUPATHI
DEPARTMENT OF COMPUTER SCIENCE
TIME:2 Hours INTERNAL EXAMINATIONS-2 Max.Marks:30

SECTION-A

Answer any five of the following 5*2=10M

1.Write about ajax?
2.Write about ajax technologies?
3.How ajax works?
4.What are the features of php?
5.What are the functions of php?
6.Explain about php operators?
7.Explain about servlet architecture?
8.Explain about static and its types ?

SECTION-B

Answer any one question from each unit 2*10=20M

UNIT-1

9.a) Write about ajax features and applications?
b) What is webserver and give examples?

(or)

10.What are the types of servers?

UNIT-2

11.Explain about php arrays?

(or)

12.Explain about servlet lifecycle?

MASTER OF COMPUTER APPLICATIONS DEGREE EXAMINATION
SECOND SEMESTER
Paper MCA 303: WEB TECHNOLOGIES
(Under C.B.S.C Revised Regulations w.e.f.2021-2023)
(Common paper to University and all Affliated Colleges)
Time:3 hours Max.Marks:70

PART-A
(Compulsory)

Answer any five of the following questions each question carries 2 marks(5*2=10)

1. a) Create a HTML form with five basic features?
b) What are the various styles in CSS?
c) How events are handled in javascript?Explain.

d)Explain about JQuery objects?
e) What is script manager in ajax?
f) What is synchronous request in ajax?
g) Explain features of php?
h) Differentiate between Get and Post methods in php?
i) Explain about security issues in servlet?
j) List the JSP implicit objects?

PART-B

Answer any ONE full question from each unit
Each question carries 12 Marks (5*12=60)

UNIT-1

2.Discuss about the Browser Architecture with diagram?

(or)

3.Explain the classifications of HTML tags with examples?

UNIT-2

4.Write about the various objects in javascript?

(or)

5.Discuss about the control structures of javascript?

UNIT-3

6.Explain step by step installation procedure of IIS webserver?

(or)

7.Discuss how to run php using IIS webserver?

UNIT-4

8.Explain about database connectivity with php with suitable examples?

(or)

9.Explain about cookies in php with examples?

UNIT-5

10.Explain JSP applications design with suitable example ?

(or)

11.Explain lifecycle of a servlet.Illustrate with an example program?