20CB402/22CM402/20CS402/20DS402/20IT402

Hall Ticket Number:

II/IV B. Tech (Regular) DEGREE EXAMINATION

July, 2025 — **Common to CB, CM, CS, DS & IT**

Fourth Semester — **Web Technologies**

Time: Three Hours — **Maximum:** 70 Marks

Answer question 1 compulsorily. **(14X1 = 14 Marks)**

Answer one question from each unit. **(4X14 = 56 Marks)**

			CO	BL	M
1	a)	Define the purpose of the <canvas> element in HTML5.	CO1	L1	1M
	b)	How an image can be included in HTML page.	CO1	L2	1M
	c)	List CSS selectors.	CO1	L1	1M
	d)	How does the target attribute work in the <a> tag?	CO1	L1	1M
	e)	What is the CSS box model?	CO2	L1	1M
	f)	Define java script.	CO2	L2	1M
	g)	Name three JavaScript event handlers and their uses	CO2	L1	1M
	h)	Differentiate internal and external styles sheets.	CO2	L1	1M
	i)	Define built-in objects in JavaScript.	CO3	L1	1M
	j)	What is the role of the window object in JavaScript?	CO3	L1	1M
	k)	How does the Node interface represent HTML elements in the DOM?	CO3	L1	1M
	l)	What is the difference between XML and HTML?	CO4	L1	1M
	m)	Define AJAX and list its core components.	CO4	L1	1M
	n)	What role does jQuery play in AJAX implementation?	CO4	L1	1M

Unit-I

2	a)	List and describe the purpose of all semantic tags introduced in HTML5. Explain how each improves document structure compared to traditional -based layouts.	CO1	L2	7M
	b)	Compare and contrast the functionality of HTML form elements (<input type="text">, <select>, <datalist>, <textarea>, and <output>). When would you choose one over another in user registration forms?	CO1	L2	7M
		(OR)			
3		Write an HTML5 form to collect student registration details (name, email, course, gender, file upload). Use labels and appropriate input types.	CO1	L2	7M

Unit-II

4	a)	Explain the differences between static, relative, absolute, and fixed positioning in CSS.	CO2	L2	7M
	b)	Explain the usage of font properties such as font-family, font-size, font-style, and how they affect the readability of web content.	CO2	L2	7M
		(OR)			
5	a)	Describe how functions improve code reusability and modularity in JavaScript.	CO2	L2	7M
	b)	Write a Java Script that inputs three integers from the user and outputs their sum, average and largest. Use alert dialog box to display results.	CO2	L3	7M

Unit-III

6	a)	List and describe all methods of the Document object for node selection (getElementById, querySelector, etc.). Include their return types and performance implications.	CO3	L2	7M
	b)	Explain the event propagation mechanism in the DOM (capturing, target, bubbling phases). How does event.stopPropagation() differ from event.preventDefault()?	CO3	L2	7M
		(OR)			
7	a)	Identify and describe the purpose of core JavaScript objects such as Date, Array, and Math.	CO3	L3	7M
	b)	Explain the DOM hierarchy and its interfaces (Node, Element, Document) with code examples.	CO3	L2	7M

Unit-IV

8		What is XML schema? Differentiate DTD and XML Schema with suitable example.	CO4	L3	14M

(OR)

9	a)	Explain the architecture and working of AJAX with the help of a diagram. How does it improve user experience compared to traditional web page requests?	CO4	L2	7M
	b)	List and describe all components of a well-formed XML document (declaration, elements, attributes, entities, etc.). Provide an example document following these rules.	CO4	L3	7M

20CB402/22CM402/20CS402/20DS402/20IT402

Hall Ticket Number:

II/IV B. Tech (Regular) DEGREE EXAMINATION

July, 2025 — **Common to CB, CM, CS, DS & IT**

Fourth Semester — **Web Technologies**

Time: Three Hours — **Maximum:** 70 Marks

Answer question 1 compulsorily. **(14X1 = 14 Marks)**

Answer one question from each unit. **(4X14 = 56 Marks)**

			CO	BL	M
1	a)	**Define the purpose of the <canvas> element in HTML5.** Canvas element is introduced in HTML5 to display 2D shapes and graphics on a web page.It is also used to apply various transformations, such as rotate and blur on image.	CO1	L1	1M
	b)	**How an image can be included in HTML page.** HTML allows you to insert an image in web page with the help of IMG element. IMG element can represent by using <img> tag. Syntax of img tag <img src=”file name”>	CO1	L2	1M
	c)	**List CSS selectors.** We have 9 different selectors as follows 1. The universal selector: * 2. The type selector: element name: h1,h2 3. The class selector: .classname 4. The id selector: #id name 5. The child selector: parent>child 6. The descendant selector: parent child 7. The adjacent sibling selector: child+child 8. The attribute selector: [attribute=value] 9. The query selector	CO1	L1	1M
	d)	**How does the target attribute work in the <a> tag?** The A element uses the target attribute to specify the window where you open a document when hyperlink is clicked. You can open a document in the same window or another window by sing target attribute.	CO1	L1	1M
	e)	**What is the CSS box model?** CSS converts the data of HTML elements in the form of rectangular boxes is called box model.The box model allows placing a border around the elements and also provides space between elements.	CO2	L1	1M
	f)	**Define java script.** JavaScript is a client and server-side object-based scripting language that is used to make interactive Web pages.	CO2	L2	1M
	g)	**Name three JavaScript event handlers and their uses** • **Mouse Events:** These events are triggered by user interactions with a mouse. • **Keyboard Events:**	CO2	L1	1M

These events are triggered by user interactions with the keyboard.

- **Form Events:**

These events relate to interactions with HTML forms and form elements.

- **Window Events:**

These events are related to the browser window or document.

- **Drag and Drop Events:**

These events are triggered during drag-and-drop operations.

- **Clipboard Events:**

These events relate to clipboard operations.

h) **Differentiate internal and external styles sheets.** CO2 L1 1M

Internal style sheet :The internal style sheet is written within the HEAD element of the HTML document.

External style sheet :The external style sheet is written outside of the HTML document.

i) **Define built-in objects in JavaScript.** CO3 L1 1M

built-in objects serve as **containers for methods and properties** that allow you to perform operations related to specific concepts or data types.

j) **What is the role of the window object in JavaScript?** CO3 L1 1M

The window object is used to open a window in a browser to display the web page.It is a global object, i.e that provides the global access to the associated variables and functions.

k) **How does the Node interface represent HTML elements in the DOM?** CO3 L1 1M

Node interface is the base interface in the DOM tree. It means that all the other interfaces are derived from the Node interface.The Node interface is simply used to represent a node in the DOM tree

l) **What is the difference between XML and HTML?** CO4 L1 1M

S.No	HTML	XML
1	HTML is used to display data and focuses on how data looks.	XML is a software and hardware independent tool used to transport and store data. It focuses on what data is.
2	HTML is a markup language itself.	HTML is a markup language itself.
3	HTML is not case sensitive. XML is case sensitive.	HTML is not case sensitive. XML is case sensitive.
4	HTML is a presentation language.	XML is neither a presentation language nor aprogramming language.
5	HTML has its own predefined tags.	You can define tags according to your need.

m) **Define AJAX and list its core components.** CO4 L1 1M

AJAX (Asynchronous JavaScript And XML) is a collection of web technologies to develop more interactive web applications or web pages.

AJAX core components: HTML/XHTML, CSS, JavaScript, the DOM, and XMLHttpRequest,

n) **What role does jQuery play in AJAX implementation?** CO4 L1 1M

The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOMmanipulation.

Unit-I

2	a)	**List and describe the purpose of all semantic tags introduced in HTML5. Explain how each improves document structure compared to traditional -based layouts.**	CO1	L2	7M

HTML5 introduced a range of **semantic elements** that clearly describe their purpose in human and machine-readable language. Unlike non-semantic elements, which provide no information about their content, semantic elements clearly define their content.

For instance, **<form>**, **<table>**, and **<article>** tags clearly define the content and purpose, to the browser.

Why Use Semantic HTML Tags?

- **Accessibility:** Semantic elements make web pages more accessible. Screen readers and other assistive technologies can interpret the structure and navigate the content more efficiently.
- **SEO:** Better structured data leads to better SEO. Search engines prioritize well-structured content that uses semantic elements correctly, as it's easier to index.
- **Maintainability:** Semantic HTML helps create a logically structured document, which is easier to read and maintain.

Semantic Elements

Here are some of the fundamental HTML5 semantic elements that you should use to structure your web content:

1. <article>
2. <aside>
3. <details>
4. <figcaption>
6. <footer>
7. <header>
8. <main>
9. <mark>
10. <nav>
11. <section>

1. The <article> Tag

The <article> tag is used for content that stands alone and can be independently distributed or reused, such as a blog post or news article.

Marks content as portable and distinct; makes RSS and syndication easier.

2. The <aside> Tag

It is used to place content in a sidebar i.e. aside from the existing content. It is related to surrounding content.

Clarifies that the content is supplementary, not core—unlike .

3. The Details and Summary Tag

The "details" defines additional details that the user can hide or view. "summary" defines a visible heading for a "details" element.

Provides accessible, interactive disclosure without JavaScript or misused s.

4. The Figure and Figcaption Tag

These are used to add an image to a web page with a small description. Provides better semantics for images or visual aids compared to .

5. The Header Tag

As the name suggests, it is for the header of a section introductory of a page. There can be multiple headers on a page.

Clearly defines page or section headers, instead of using .

6. The Footer Tag

Footer located at the bottom of any article or document, they can contain contact details, copyright information etc. There can be multiple footers on a page.

Differentiates bottom content without relying on .

7. The Main Tag

It defines the main content of the document. The content inside the main tag should be unique.

Helps assistive technologies focus on the primary content of the page.

8.The Section Tag

A page can be split into sections like Introduction, Contact Information, Details, etc and each of these sections can be in a different section tag.

Groups related content meaningfully (like <section class="about">), rather than using vague containers.

Example:

>9. The nav Tag

It is used to define a set of navigation links in the form of a navigation bar or nav menu.

Identifies navigational blocks for screen readers and search engines; has no such meaning.

10. The Mark Tag

It is used to highlight the text.

Conveys meaning programmatically, unlike styling with background color.

b)	**Compare and contrast the functionality of HTML form elements (<input type="text">, <select>, <datalist>, <textarea>, and <output>). When would you choose one over another in user registration forms?**	CO1	L2	7M

INPUT element creates interactive controls for web based form that enables the user to enter the data.

1. **<input type="text">**

   Provides a single-line input field for short text entries (e.g., name, username, email).

Use Case in Registration Form:

- Username

- First Name / Last Name
- Email
- Phone Number (sometimes with type="tel")

Pros:

- Simple and versatile
- Can apply attributes like maxlength, pattern, placeholder

2. <textarea>

Provides a **multi-line** input field for longer text input.

Use Case in Registration Form:

- User bio or description
- Address (optional)
- Additional notes or comments

Pros:

- Adjustable rows/columns
- Can grow dynamically
- Handles large input better

3. <select>

Creates a **dropdown list** with predefined options. One or more options can be selected.

Use Case in Registration Form:

- Country selection
- Gender
- Language preference
- Security question

Pros:

- Enforces controlled input
- Compact UI
- Can allow single or multiple selections (multiple attribute)

4. <datalist>

Provides an **autocomplete suggestion list** for an <input> field. Unlike <select>, users can type freely or pick from the suggestions.

Use Case in Registration Form:

- City (with flexibility)
- Profession or skill suggestions
- Educational institution (optional field)

Pros:

- Offers guidance without restriction
- Keeps the field flexible
- Reuses suggestion list for multiple inputs

5. <output>

Displays the **result of a calculation or action** (e.g., live character count, password strength, BMI).

Use Case in Registration Form:

- Showing password strength score
- Displaying calculated age from date of birth
- Summarizing selected data

Pros:

- Dynamic feedback
- Integrated with JavaScript
- Non-editable but visible to user

(OR)

3	**Write an HTML5 form to collect student registration details (name, email, course, gender, file upload). Use labels and appropriate input types.**	CO1	L2	14M

```
<!DOCTYPE html>

<head>

  <title>Student Registration Form</title>

</head>

<body>

  <h1>Student Registration Form</h1>




    <label for="name">Full Name:</label><br>

    <input type="text" id="name" name="name" required><br><br>

    <label for="email">Email Address:</label><br>

    <input type="email" id="email" name="email" required><br><br>

    <label for="course">Course:</label><br>

    <select id="course" name="course" required>

      <option value="">--Select Course--</option>

      <option value="CSE"> CSE </option>

      <option value="IT"> IT </option>

      <option value="ECE">ECE</option>

      <option value="EEE">EEE</option>

    </select><br><br>

    <label>Gender:</label><br>

    <input type="radio" id="male" name="gender" value="male" required>

    <label for="male">Male</label><br>

    <input type="radio" id="female" name="gender" value="female">

    <label for="female">Female</label><br>

    <input type="radio" id="other" name="gender" value="other">

    <label for="other">Other</label><br><br>

    <label for="file">Upload Photo:</label><br>

    <input type="file" id="file" name="photo" accept="image/*"
required><br><br>
```

```
<button type="submit">Register</button>

</form>

</body>

</html>
```

Unit-II

4	a)	**Explain the differences between static, relative, absolute, and fixed positioning in CSS.**	CO2	L2	7M

Position of element using CSS:

CSS controls the position of element with respect to the normal flow of content of a web page. Syntax: position : [value]

Example: p{position : fixed}

Static Default value. Elements render in order, as they appear in the document flow

Absolute The element is positioned relative to its first positioned (not static) ancestor element

fixed The element is positioned relative to the browser window

RelativeThe element is positioned relative to its normal position, so "left:20px" adds 20 pixels to the element's LEFT position

	b)	**Explain the usage of font properties such as font-family, font-size, font-style, and how they affect the readability of web content.**	CO2	L2	7M

Font represents the style and size of the text in web browser.
It is used to differentiate the content.
For example, you can easily differentiate main level heading and sub heading based on font size and
styles.

1. font-family:

It is used to specify the name of the font family to apply the specified font style on the text.

- The font families are

a) Serif fonts have a small stroke at the edges of each letter. They create a sense of formality and
elegance.
b) Sans-serif fonts have clean lines (no small strokes attached). They create a modern and
minimalistic look.
c) Monospace fonts - here all the letters have the same fixed width. They create a mechanical look.
d) Cursive fonts imitate human handwriting.
e) Fantasy fonts are decorative/playful fonts.
Font Family Examples of Font Names
Serif Times New Roman, Georgia, Garamond
Sans-serif Arial, Verdana, Helvetica
Monospace Courier New, Lucida Console, Monaco
Cursive Brush Script MT, Lucida Handwriting

Effect on Readability:

- **Sans-serif fonts** (like Arial, Helvetica) are cleaner and better for **digital screens**.
- **Serif fonts** may aid readability in **printed content** but can be harder to read on screens at small sizes.
- Avoid overly decorative fonts for body text.

2. font-size:

The font-size property is used to change the size of the text.
The value of font-size property is specified in pixels.
Example: p{ font-size : 1.2em;20px; }
The value of font-size property is specified in another three ways
a) Absolute value: absolute value refers to the absolute size of the font i.e,. Size was fixed that

cannot be changed by used. The absolute values are xx-small, x-small, small, medium, large, xlarge,
xx-large.
b) Relative value: Relative values are calculated on the basis of the current font values or
surrounding element values.
c) Percentage value: Percentage values the size is represented by percentage to increase or decrease the font size.

3. font-size-adjust:

- The font-size-adjust property is used to change the aspect value of the text on a web page.
- Aspect value is the ratio between the font height of lowercase letter and actual height of the font.

Example: p{ font-size-adjust : 0.5; }

Effect on Readability:

- Text that is too **small** strains the eyes.
- Text that is too **large** reduces reading flow and may appear unprofessional.
- Responsive font sizing improves user experience across devices.

3. font-style:

The font-style property is used to specify the style of the font.
font-style properties are : normal, italic and oblique
Example: p{ font-style : italic; }

Effect on Readability:

- Italic text can add emphasis but is harder to read in large blocks.
- Use sparingly to maintain legibility.

(OR)

5	a)	**Describe how functions improve code reusability and modularity in JavaScript.**	CO2	L2	7M

Function in JavaScript, use function keyword, followed by the function name, which is

followed by the parentheses (contains parameter list).

Syntax to define a function in JavaScript:

function funName (parameter1, parameter2, parameter3) {

Code to be executed here

}

Example:

<script type="text/javascript">

function alertmsg()

{

alert("Hello JavaScript, I am Statements of Function Definition");

}

</script>

2. Reusability

Write a block of logic once, and reuse it in multiple places.

- **Avoids duplication** of code.
- Makes code **shorter and cleaner**.
- Easier to **update or fix bugs** in one place.

🔧 **Example:**

javascript

CopyEdit

```
function calculateArea(width, height) {
  return width * height;
}

let area1 = calculateArea(5, 10);
let area2 = calculateArea(3, 7);
```

Instead of rewriting the formula every time, the logic is reused.

Modularity

Breaks down large programs into **smaller, manageable parts**.

- Improves **readability**.
- Simplifies **debugging and testing**.
- Encourages **separation of concerns** (each function does one task).
- Promotes **collaborative development** (different people can work on different functions).

b) **Write a Java Script that inputs three integers from the user and outputs their sum, average and largest. Use alert dialog box to display results.** CO2 L3 7M

```
<!DOCTYPE html>
<html>
<head>
  <title>Sum, Average, and Largest</title>
  <script>
    function calculateResults() {
      let num1 = parseInt(prompt("Enter first integer:"));
      let num2 = parseInt(prompt("Enter second integer:"));
      let num3 = parseInt(prompt("Enter third integer:"));

      let sum = num1 + num2 + num3;
      let average = sum / 3;
      let largest = Math.max(num1, num2, num3);

      // Output
      alert(
        "Results:\n" +
        "Sum: " + sum + "\n" +
        "Average: " + average.toFixed(2) + "\n" +
        "Largest: " + largest
      );
    }
  </script>
</head>
<body onload="calculateResults()">
</body>
</html>
```

Unit-III

6 a) **List and describe all methods of the Document object for node selection (getElementById, querySelector, etc.). Include their return types and performance implications.** CO3 L2 7M

The document object is child object of window object, which refers to a browser.
 The document object is created when HTML document is loaded in a browser.
 The document object is an object that provides to access the all the HTML elements of a document.
 The document object stores the elements of HTML document, such as HEAD, BODY, and HTML as
objects.
 You can access document object either window.document or document properties.

Document object methods:
 The method is call by objectName . methodName(arguments)

Method	Description
open()	Opens an HTML document to display the output
close()	Closes the HTML document
getElementById()	Returns the element that has the ID attribute with the specified value
getElementsByName()	Returns an live NodeList containing all elements with the specified name
getElementsByTagName()	Returns an live NodeList containing all elements with the specified name
getElementsByClassName()	Returns an HTMLCollection containing all elements with the specified class name
write()	Writes HTML expressions or JavaScript code to a document
writeln()	Same as write(), but adds a newline character after each statement
querySelector(selector)	Returns the **first** element within the document that matches the specified CSS selector.

Performance Comparison

Method	Speed	Best Use Case
getElementById	**Fastest**	Selecting a unique element by
getElementsByClassName	Very Fast	Group of elements with same c
getElementsByTagName	Fast	All elements of a specific tag
getElementsByName	Fast	Form inputs with the same nan
querySelector	Moderate	First match for complex CSS s

b) **Explain the event propagation mechanism in the DOM (capturing, target, bubbling phases). How does event.stopPropagation() differ from event.preventDefault()?** CO3 L2 7M

The **event propagation mechanism in the DOM** defines how events travel through the document tree when an event is triggered on an element. It consists of **three phases**:

Event Propagation Phases

1. Capturing Phase (also called *capture phase*)

- The event starts from the **document root** and travels **downward** to the **target element**.
- Handlers registered with addEventListener(type, listener, { capture: true }) (or true) are invoked in this phase.
- Rarely used, but helpful when you want to intercept events before they reach the target.

2. Target Phase

- The event reaches the **target element**.
- Event listeners on the target element are called, regardless of whether they were registered for capture or bubble.

3. Bubbling Phase

- After reaching the target, the event **bubbles back up** from the target to the root (document).

- Handlers registered with addEventListener(type, listener) (default { capture: false }) are triggered in this phase.
- This is the most commonly used phase.

Illustration of Event Flow

html

<button id="inner">Click Me</button>

When clicking the button, the event propagation order is:

1. **Capturing**: document → html → body → #outer → #inner
2. **Target**: #inner (click event happens here)
3. **Bubbling**: #inner → #outer → body → html → document

event.stopPropagation() vs event.preventDefault()

Method	**What it does**
event.stopPropagation()	Stops the event from propagating to parent elements (no bubbling/capturing)
event.preventDefault()	Prevents the default action associated with the event

(OR)

7 a **Identify and describe the purpose of core JavaScript objects such as Date, Array, and Math.** CO3 L3 7M

Date Object:
 The Date object used to display the date on web page or time stamp in numerical or mathematical
calculations.
 Date objects are created with new Date ().
 There are five ways of instantiating (creating) a date:
var date1 = new Date();
or
var date1 = new Date(milliseconds);
or
var date1 = new Date("mm dd, yyyy");
or
var date1 = new Date("mm dd, yyyy hr:min:sec");
or
var date1 = new Date(yyyy, mm, dd, [, hr, min, sec, millisec]);

properties:

Name	Description
Constructor	Returns the function that created the Date object's prototype
Prototype	Allows you to add properties and methods to an object

Array object:
 An Array is used to store a number of values (called as elements) in order with a single variable.
 An array object can created in two ways
Using array constructor:
 Creates an empty array: var ar = new Array()
 Creates an array with given size: var ar = new Array(size)
 Creates an array based on number of elements:
var ar = new Array(element0, element1, ..., elementN)

Using array literal notation:
 Array literal notation are coma separated list of items enclosed by square brackets.
 creates an empty array: var ar = [ ]
 creates an empty array with elements: var ar = [ele1,ele2,...,elen]
 example: var ar = [5,"hello", true]

Javascript Array Objects Property

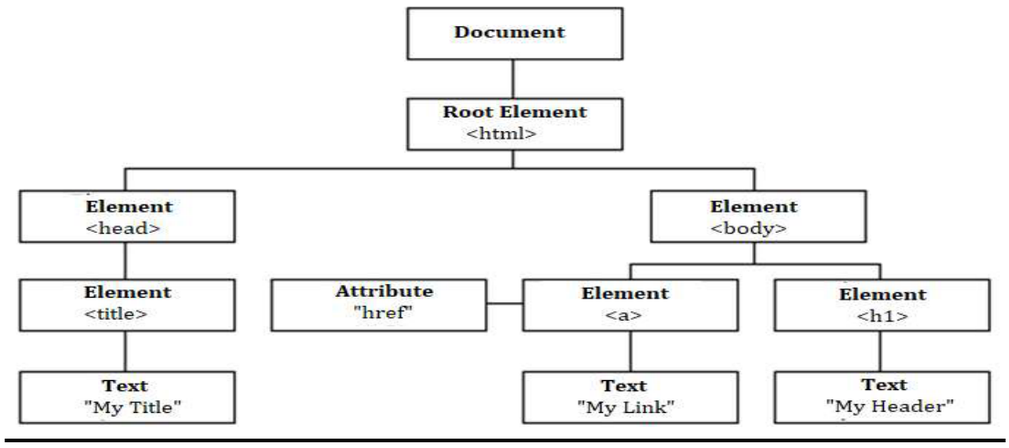
Name	Description
Length	Returns the number of elements in an array
Prototype	Use to add new properties to all objects.
constructor	Specifies the function which creates an object's prototype

Math Object:
 The math object is used to perform simple and complex arithmetic operation.
 JavaScript math object is used to perform mathematical task.

Properties:

Property	Description	
	Holds	**approximate value**
E	Euler's number	2.718
LN2	natural logarithmic of 2	0.693
LN10	natural logarithmic of 10	2.302
LOG2E E	base-2 logarithmic of	1.442
LN10	natural logarithmic of 10	2.302
LOG2E	base-2 logarithmic of	1.442

b **Explain the DOM hierarchy and its interfaces (Node, Element, Document) with code examples**. CO3 L2 7M



Understanding DOM Interfaces

The DOM API provides interfaces to implement DOM. Each interface is associated with a particular type of node that is defined in the inheritance hierarchy, which are described in the following table:

The Node Interface

This interface is the base interface in the DOM tree. It means that all the other interfaces are derived from the Node interface.

The Node Interface has Properties and Methods

The Document Interface:

 This interface is the root node in the DOM tree. It defines the documentElement property of the document, which returns the root node by looping through all the child nodes of the Document nodes to test whether or not they are Element nodes

The Element Interface:

 This interface represents an element in an HTML document.

Example

```
<!DOCTYPE HTML>
<html>
<head>
<title>Node Interface</title>
<script type="text/javascript">
```

```
function create_node() {
var link = document.createElement("a");
link.setAttribute("id","l1");
link.setAttribute("href","D://images/beclogo.png");
link.setAttribute("width","100");
link.setAttribute("height","100");
text = document.createTextNode("My Link");
link.appendChild(text);
document.body.appendChild(link);
}
function delete_node() {
child = document.getElementById("l1");
document.body.removeChild(child);
}
function clone_node() {
link = document.getElementById("l1");
c = link.cloneNode(true);
document.body.appendChild(c);
}
function insert_node() {
link = document.getElementById("l1");
para = document.createElement("p");
para.setAttribute("id","p1");
text = document.createTextNode("My Paragraph before link");
para.appendChild(text);
document.body.insertBefore(para,link);
}
</script>
</head>
<body >
<div id="d">
<h2>Node interface methods</h2>
<button onclick="create_node()">create Link</button>
<button onclick="delete_node()">remove Link</button>
<button onclick="clone_node()">clone Link</button>
<button onclick="insert_node()">Insert Paragraph before Link</button>
</div>
</body>
</html>
```

UNIT-IV

8	**What is XML schema? Differentiate DTD and XML Schema with suitable example.**	CO4	L3	14M

Describing XML Schema:

- An XML schema is similar to DTD file.
- Xml schema is a document that describes the structure of an xml document.
- It is used to validate the structure of an xml document with the help of an xml parser.
- The main difference between DTD and XML is that the XML schema use XSD (XML schemadocument)
- XSD is an XML based language describes the structure of an XML document where as the DTD fileused a list of elements to describe the document.

Syntax for define element: <xs:element name="name" type="type"/>

Example: <xs:element name="apple" type="xs:string"/>

Syntax for define attribute: <xs:attribute name="name" type="type"/>

Example: <xs:attribute name="lang" type="xs:string"/>

Here the name attribute specifies the name of element or attribute and the type attribute specifies thedata type of content.

The schema document can contain various data types which can be divided into two categories.

the predefine data types available in XML scheme are as follows:

xs:string

xs:decimal

xs:integer

xs:boolean

xs:date

xs:time

user-define:

The user-define data types divides into

i) Simple type element.

ii) Complex type elements

i) Simple type element:

Simple type element can contain only text cannot have any child elements or attributes

Syntax: <xs:element name="name" type="type"/>

Example:

XML document elements

<lastname>Apple</lastname>

<age>36</age>

<dateborn>1970-03-27</dateborn>

Simple type element for preceding XML elements

<xs:element name="lastname" type="xs:string"/>

<xs:element name="age" type="xs:integer"/>

<xs:element name="dateborn" type="xs:date"/>

The important differences between DTD and XSD are given below:

S.NO	DTD	XSD
1	DTD stands for Document Type Definition.	XSD stands for XML Schema Definition.
2	DTDs are derived from SGML syntax.	XSDs are written in XML.
3	DTD doesn't support datatypes.	XSD supports datatypes for elements and attributes
4	DTD doesn't support namespace.	XSD supports namespace
5	DTD doesn't define order for child elements.	XSD defines order for child elements.
6	DTD is not extensible.	XSD is extensible.
7	DTD is not simple to learn.	XSD is simple to learn because you don't need to learn new language
8	DTD provides less control on XML structure.	XSD provides more control on XML structure.

(OR)

9 a) **Explain the architecture and working of AJAX with the help of a diagram. How does it improve user experience compared to traditional web page requests?** CO4 L2 7M

Exploring AJAX:

- AJAX is a new technique for creating better, faster, and more interactive web applications with the help

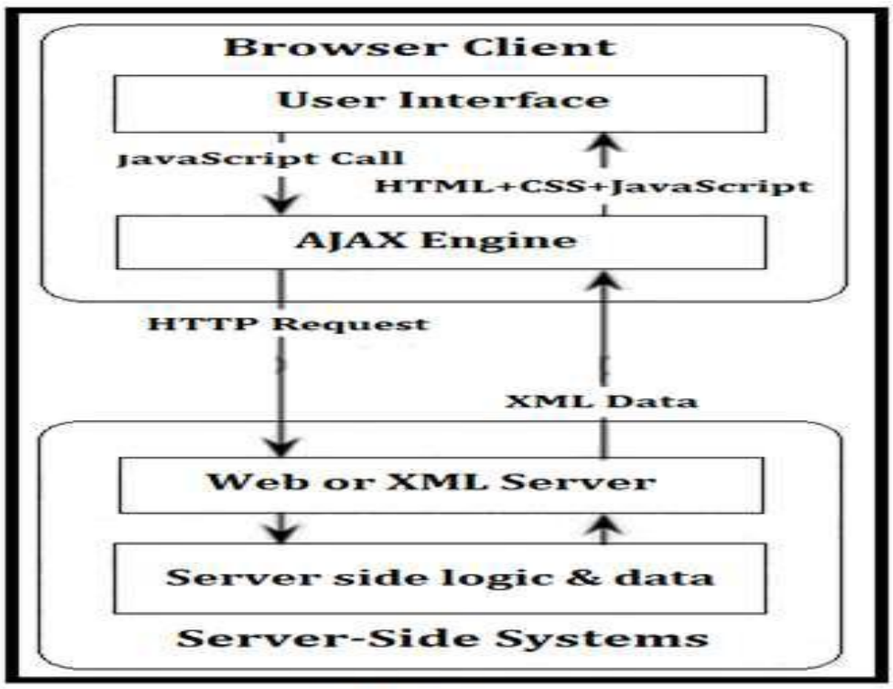
of XML, HTML, CSS and JavaScript.

- AJAX is not a programming language but way to use existing standards such as JavaScript and XML.
- AJAX is based on some internet standards, are
- XMLHttpRequest: Refers to an object that is used to exchange data asynchronously with a server.
- JavaScript and DOM: Display and interact with the information.
- CSS: provides styles to display the data.
- XML: provides a format to transfer the data from server to a client.

AJAX Web Application Model:

- The AJAX we application eliminates the start-stop-start-stop nature.
- At beginning it loads an ASP engine. An Asp engine code written in JavaScript and it establish the

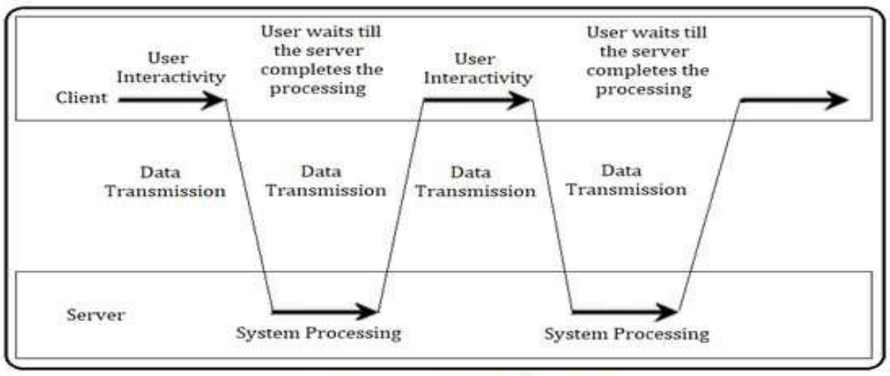
communication between a user interface and the AJAX engine on the client side.



AJAX Web Application Model

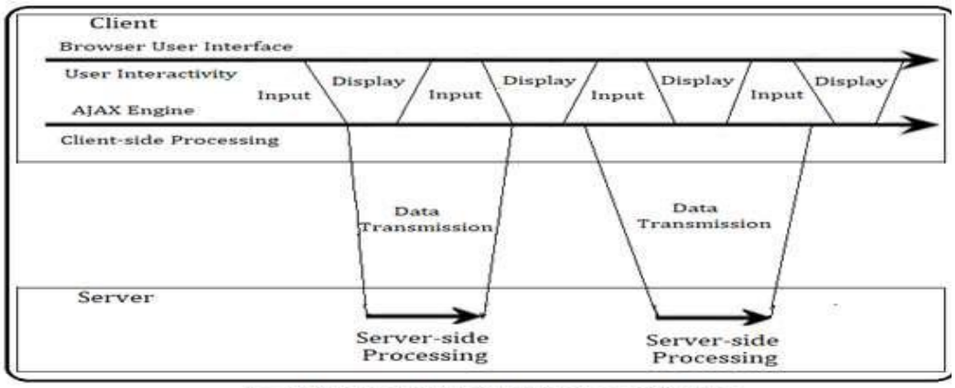
A web page sends its request to the server using JavaScript function.
The server response contains data and not style. The style on the data was implemented by using markup language.
Most of the page does not change only some parts of the page that need change are updated.
JavaScript dynamically updates the Webpage, without reloading the entire Web page.
This prevents the user from waiting for the server to complete its processing.
The AJAX engine takes care of displaying the user interface and interacts with the server on the users behalf.

- In traditional web application, a synchronous mode of communication exists between the client and the server as shown below.



Synchronous Mode of Communication

In AJAX, an asynchronous mode of communication exists between the client and the server as shown below.



Asynchronous Mode of Communication

It is seen that in the synchronous mode of interaction, there is no scope for user to wait until the server side processing gets over.

How AJAX works:

Following steps are involved in the working of AJAX:

Step 1:

Creating an instance of XMLHttpRequest object to send HttpRequest from client to server.

Syntax: variable = new XMLHttpRequest();

Example: req = new XMLHttpRequest();

Step 2:

Creating a request to the server by using the open( ) method of the XMLHttpRequest object.

Syntax: open(method, URL, async);

Example: req.open(GET, “file.asp”, true);

Here, method specifies the type of request (GET, POST),
URL specifies the location of file on the server,async specifies that the request is handled or not. The true indicate asynchronous and false indicates synchronous.

Step 3:

Sending a request to the server by using send( ) method of the XMLHttpRequest object.

It having two metods send( ) and send(string) for sending request.

If request types is GET then use send( ) constructor.

Syntax: open(method, URL, async);
send( )

Example: req.open(GET, “file.asp”, true);
send( );

If request type is POST the use send(string) constructor.

Syntax: open(method, URL, async);
send(string);

Example: req.open(GET, “file.asp”, true);
send(uname=appe&password=12345);

b) **List and describe all components of a well-formed XML document (declaration, elements, attributes, entities, etc.). Provide an example document following these rules.** CO4 L3 7M

Exploring XML:

Extensible Markup Language (XML) is a markup language on simple and platform-independent tool for storing and transporting data.

HTML describes how to display data on the screen.

For example:

```
<p>
<B>Mr. Ram Sharma</B><br/>
6-3-2, Main Street<br/>
Bapatla, 522101
</p>
```

Structure of an XML document:

XML define certain rules for its syntax that specify how to create or structure an XML document.

- The syntax used to create an XML document is called markup syntax.
- While creating an XML document you must remember the following points.
- XML document must have starting and closing taga.
- XML tags are case sensitive.
- XML elements must be properly nested.
- XML document must have one root element.
- XML attributes values must be enclosed in double quotes.

The structure of XML document as follow

```
<?xml version="1.0" encoding="UTF-8" ?>
<Employee>
<FirstName>Mabasha</FirstName>
<LastName>Shaik</LastName>
<Age>28</Age>
<EmpID id=”BEC14”></EmpID>
</Employee>
```

Xml document having the following sections:

- XML declaration
- XML elements
- XML attributes
- XML tree
- XML comments

XML declaration:

- XML declaration is the first line in the document.
- The XML declaration statement is used to indicate that the specified document is an XML document
- XML declaration defines the version and character encoding.

Syntax: <?xml version="1.0" encoding="UTF-8"?>

XML declaration is optional but it is good practice to include it.

XML elements:

- Elements are the building blocks of XML document.
- These divide the document into a hierarchy of sections, each serving a specific purpose.
- Elements are represented by tags.
- A start tag is delimited by the < and > and an end tag is delimited by the </ and > characters.

Example: <Employee></Employee>

A document must have single root element and it is the top most parent element in an XML.

Example: <?xml version="1.0" encoding="UTF-8"?>

```
<Employee>
.
.
```

.
</ Employee>

Elements are categorized into two types

Empty element:

- An empty element does not contain any content or other element within it.
- It can have attribute that helps in identifying an entity.

Example: <?xml version="1.0" encoding="UTF-8"?>
<items>
<item id=”1234” />
</ items >

Nested element:

An element contain another elements are known as nested element.
In above example <item> is nested in <items> element.

Rules while defining elements as follow

- Elements name must starts with letters and underscore (_) character.
- Elements name cannot contain spaces.
- Elements name cannot contain the : character.
- Elements name cannot start with the word like xml in upper, or lower or mixed.
- There cannot be space after < but may be space before > characters.

XML attributes:

- Attributes are used to provide additional information about the properties and behavior of HTML

elements.

- Attributes are name-values pairs separated by the equal (=) sign.
- Attribute values are enclosed within single quotes or double quotes.
- Example: <person gender="male" age="25"> Apple </person>

  Let see the xml document representation:

  <?xml version="1.0">
  <address>
  <NAME>
  <title>Mr.</title>
  <first-name>Ram</first-name>
  <last-name>Sharma</last-name>
  </NAME>
  <street>6-3-2, Main Street</street>
  <city>Bapatla</city>
  <zip-code>522101</zip-code>
  </address>