Web Technologies

SEC103

Bachelor of Computer Applications

I Year II Semester

G Prembihari Saran

Berhampur University

Introduction to JavaScript

JavaScript is a high-level, interpreted programming language widely used for client-side web development. JavaScript enables dynamic content, user interactivity, manipulation of HTML and CSS, and communication with servers without reloading the page.

JavaScript Functions and Events

- **Function**: A reusable block of code invoke with a name. Syntax:

```
function greet(name) {
  alert("Hello, " + name);
}
greet('World');
```

- **Events**: Actions that happen in the browser (e.g., user click, mouseover, keydown). You can attach events to HTML elements using event attributes (`onclick`, `onload`) or `addEventListener`:

```
document.getElementById('myBtn').onclick = function() { alert('Clicked!'); };
document.getElementById('myBtn').addEventListener('click', function() { alert('Clicked!'); });
```

Document Object Model (DOM) Traversing

- **DOM**: Represents the structure of an HTML document as a tree of objects.
- Access and modify DOM elements using JavaScript:

```
document.getElementById('header').innerHTML = 'New Title';
let elems = document.getElementsByClassName('main');
let allImages = document.getElementsByTagName('img');
```

Output System in JavaScript

- **alert()**: Displays a popup box with an OK button

```
alert('Hello, user!');
```

- **prompt()**: Asks the user for input in a popup box

```
let name = prompt('Enter your name:');
```

- **console.log()**: Prints information to the browser console (developer tools)

```
console.log('Debugging message');
```

- **document.write()**: Writes directly to the HTML document (rarely used in modern web apps)

Variables and Arrays in JavaScript

- **Variables**: Store data values. Declared with `var`, `let`, or `const`.

```
let age = 25;
const PI = 3.14;
var name = "Sam";
```

- **Arrays**: Collections of values, ordered and indexed by number

```
let fruits = ["Apple", "Banana", "Cherry"];
console.log(fruits[1]); // "Banana"
```

Date and String Handling in JavaScript

- **Date**: Create and manipulate dates using the `Date` object

```
let now = new Date();
let birthday = new Date('2000-05-28');
let year = birthday.getFullYear();
```

- **String**: JavaScript provides many string functions

```
let message = "Hello, World!";
message.length; // 13
message.toUpperCase(); // "HELLO, WORLD!"
```

```
message.substring(0, 5); // "Hello"
```

Manipulating CSS through JavaScript

- You can change the style of elements dynamically:

```
document.getElementById('myDiv').style.backgroundColor = 'yellow';
document.getElementById('main').classList.add('active');
document.documentElement.style.setProperty('--main-color', 'blue');
```

JavaScript Form Validation

HTML5 Validators

- `required`: Ensures a field must be filled before submitting
- `minlength`/`maxlength`: Restrict length of input
- `pattern`: Regular expression pattern that input must match

```
<input type="text" required minlength="3" pattern="[A-Za-z]{3,}" />
```

JavaScript Custom Validation

- Manual validation can be added for advanced checks

```
let value = document.getElementById('username').value;
if (!value) { alert('Username is required'); }
if (value.length < 4) { alert('Too short'); }
if (!/^[A-Za-z]+$/.test(value)) { alert('Only letters allowed'); }
```

Advanced JavaScript: Combining HTML, CSS, and JS Events

- Interact with styles, structure, and behaviors via event listeners

```
document.getElementById('btn1').onclick = function() {
  document.getElementById('target').style.color = 'red';
  document.getElementById('target').innerHTML = 'You clicked!';
}
```

- JavaScript can control navigation, open new windows, or interact with browser APIs

```
window.open('https://example.com');
history.back();
```

AJAX (Asynchronous JavaScript and XML)

Introduction & Purpose

- AJAX is a technique for updating parts of a web page without reloading the full page.
- Uses JavaScript to send HTTP requests in the background.
- **XMLHttpRequest** object powers AJAX (also possible via fetch API in modern browsers).

AJAX Implementation Example

```
let xhr = new XMLHttpRequest();
xhr.open('GET', 'data.json', true);
xhr.onreadystatechange = function() {
  if (xhr.readyState === 4 && xhr.status === 200) {
    let data = JSON.parse(xhr.responseText);
    console.log(data);
  }
};
xhr.send();
```

Advantages of AJAX

- No page reload for data fetch/update
- Reduces server load
- Responsive and dynamic UI

Disadvantages of AJAX

- Harder to debug and maintain
- May cause browser history/navigation issues
- Relies on JavaScript being enabled

AJAX Applications & Alternatives

- Used in web apps for live search, chat, content loaders, form validation
- Alternatives: WebSockets (real-time), server-sent events (push updates), Fetch API

Introduction to XML

- **XML** (Extensible Markup Language) is used to store, organize, and transport data.
- Tags are user-defined; document must be well-formed.
- Key Concepts: Elements, attributes, well-formedness, extensibility

DTD (Document Type Definition)

- Defines structure and valid elements/attributes in XML.
- Can be internal (in XML file) or external.
- Ensures data consistency; rigid, less flexible than XSD

XML Schema (XSD)

- Defines structure, element types, attributes for XML.
- Supports data types, namespaces, powerful than DTD

XSL, XSLT, and Transformations

- **XSL** (Extensible Stylesheet Language): Styles XML for display.
- **XSLT** (XSL Transformations): Transforms XML into another format (e.g., HTML)

```
<xsl:template match="/">
  <html><body><h2>Title</h2></body></html>
```

```
</xsl:template>
```

Introduction to XHTML

- **XHTML** (Extensible HyperText Markup Language) is a stricter reformulation of HTML based on XML.
- Follows XML rules: all tags closed, lower case, attribute values quoted, nesting required
- All tags must be closed and properly nested for validity.
- Greater error checking, compatibility with XML tools.

JSON (JavaScript Object Notation)

- Lightweight data-interchange format; easy for humans to read/write.
- Used for web APIs and config files.

Structure: Keys, Values, Arrays, Objects

- **Keys**: Always strings, unique within the same object
- **Values**: Can be string, number, boolean, array, object, or null
- **Objects**: Key-value pairs inside curly braces

```
{ "name": "Alice", "age": 30, "isStudent": false }
```

- **Arrays**: Ordered list of values inside square brackets

```
{ "colors": ["red", "green", "blue"] }
```

These notes provide detailed explanations, code examples, and context for all major topics of this unit, including core JavaScript, manipulating the DOM, AJAX, XML/XHTML fundamentals, and JSON data handling.