PROJECT TRAINING WORKSHOP

HTML & CSS

RAAFI

Builds You Best

USER INTERFACE

- **What is User Interface?**
  - The point of interaction between a user and a digital product, such as a software application, website, or any other digital interface

- **What are key concepts in User Interface Design?**

User-Centered Design	Visual Design	Usability	Navigation
Consistency	Hierarchy and Information Architecture	Feedback	Affordances
Accessibility	Responsive Design	User Flows	Gestalt Principles
White Space	Error Handling	Microinteractions	Prototype and Testing

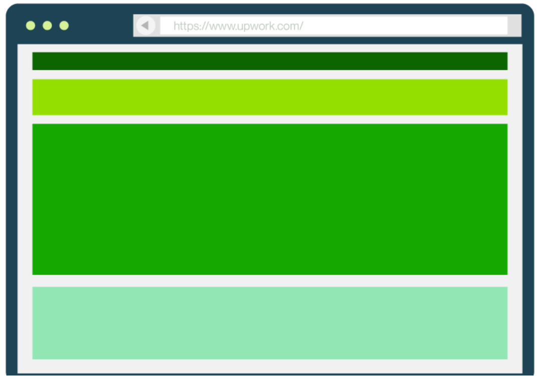
USER INTERFACE

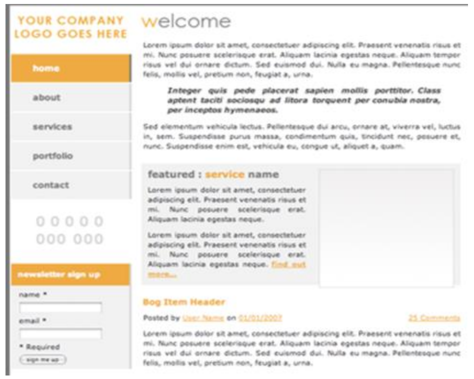
- **What is User Interface Layout?**
  - The point of interaction between a user and a digital product, such as a software application, website, or any other digital interface

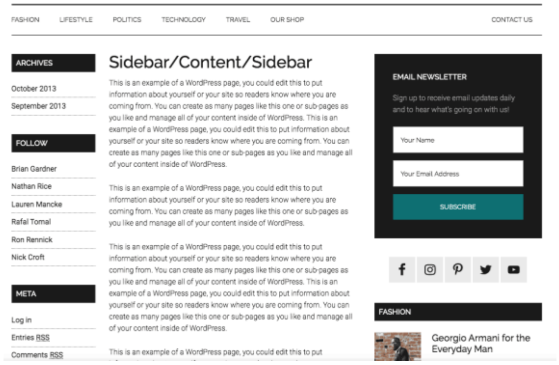
- **What are different types User Interface Layout?**

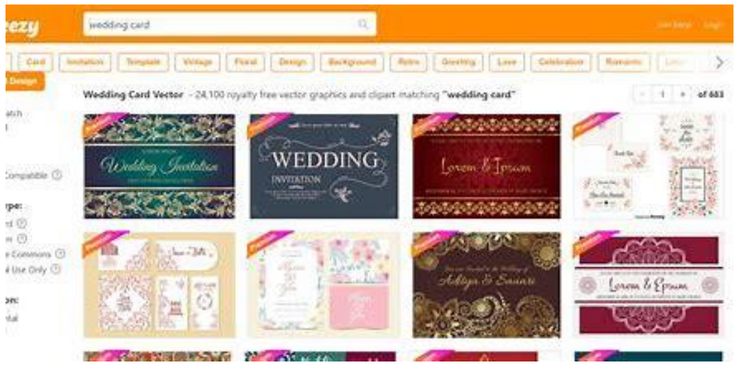
Single Column Layout	Two-Column Layout	Three-Column Layout	Grid Layout
Card-Based Layout	Full-Screen Layout	Tabbed Layout	Floating Action Button (FAB) Layout
Stacked Layout	Masonry Layout	Overlay Layout	Fixed Header/Footer Layout
Split Screen Layout	Centered Layout		

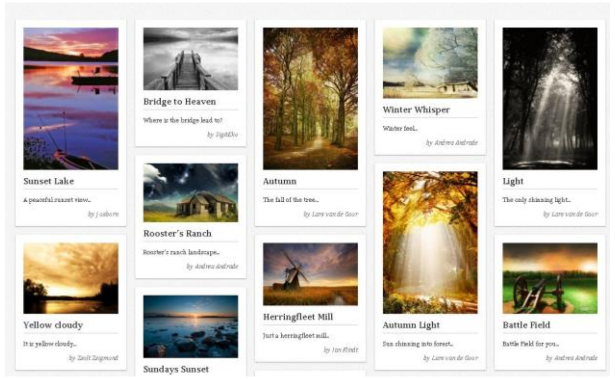
USER INTERFACE

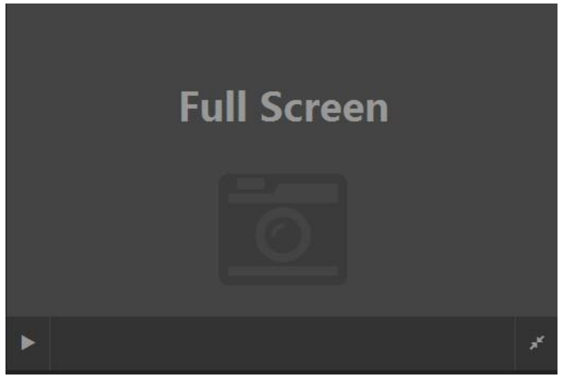


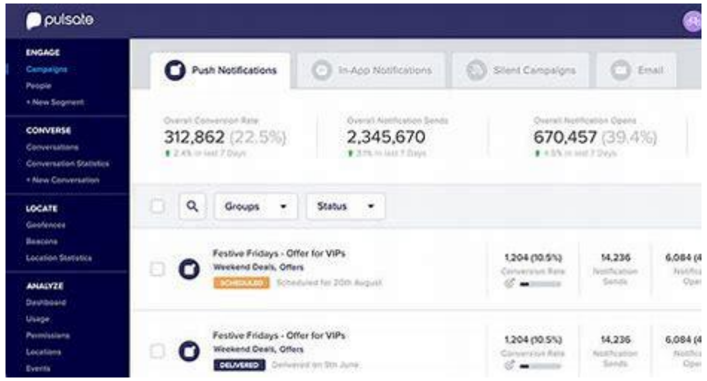




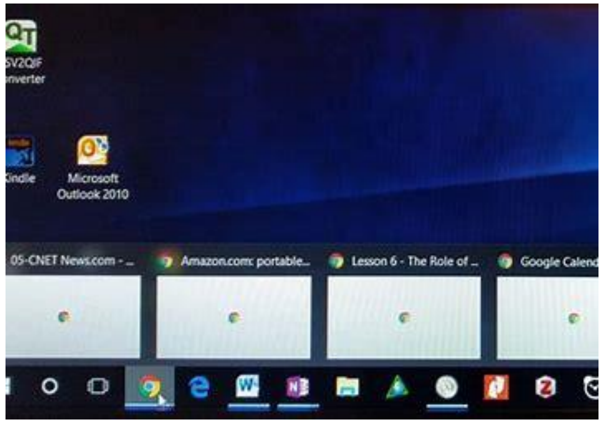


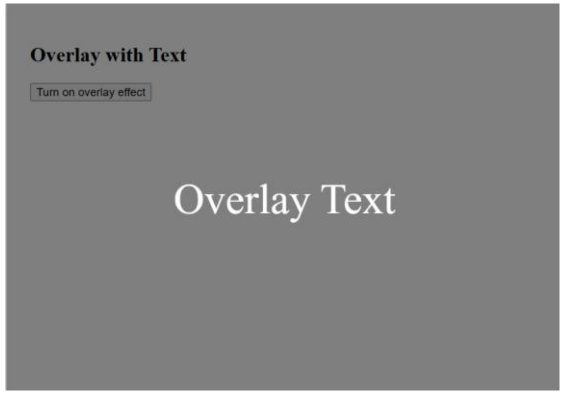


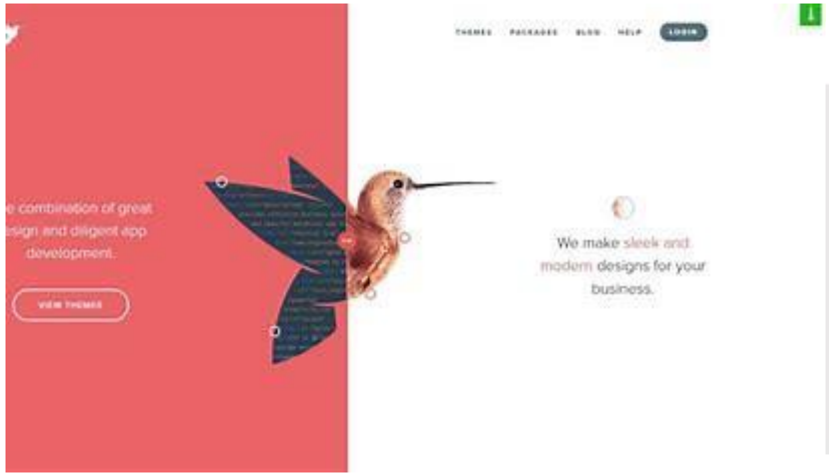




USER INTERFACE







e combination of great design and diligent app development.

We make sleek and modern designs for your business.

Well, hello. Bonjour, salut.

Incub is a community of active people who are passionate about their work and the impact it has on others. Our work is at the crosspoint of science, art, culture, technology and design.

Read more →

RAAFI Builds You Best

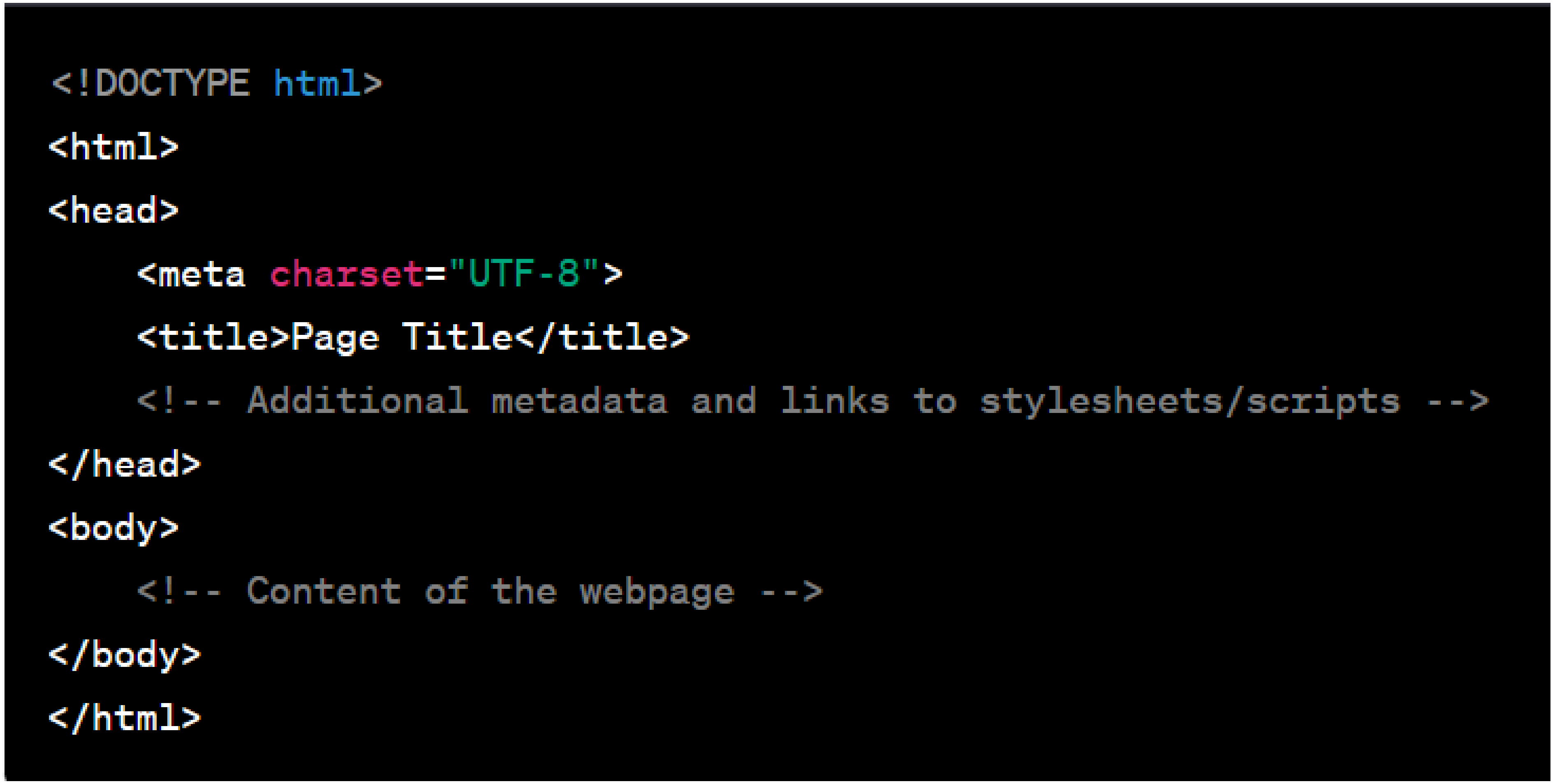
HTML

- **What is HTML?**
  - **Hypertext Markup Language, is the standard markup language used for creating and structuring content on the World Wide Web**

- **What is the role of HTML in web development?**

Structuring Content	Semantic Meaning	Hyperlinks and Navigation	Forms and User Input
Media Embedding	Document Metadata	Compatibility	Foundation for Styling and Layout
Integration with Other Technologies			

HTML STRUCTURE



```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Page Title</title>
    <!-- Additional metadata and links to stylesheets/scripts -->
</head>
<body>
    <!-- Content of the webpage -->
</body>
</html>
```

RAAFI
Builds You Best

HTML KEY CONCEPTS

- **What are the key concepts of HTML?**

Document Structure	Elements and Tags	Semantic Elements	Headings and Paragraphs
Lists	Hyperlinks	Images	Forms
Input Types	Attributes	Formatting	Comments
HTML Entities	Validation	Escape Characters	Nesting
Responsive Design	HTML5 APIs	Inline vs. Block Elements	

CSS

- **What is CSS?**
  - **Cascading Style Sheets)** is a stylesheet language used for describing the presentation and layout of HTML documents. (Color, Font, Spacing, Position, responsiveness)**

- **What is the main role of CSS?**

Separation of Concerns	Visual Styling	Layout Control	Consistency
Accessibility	Ease of Maintenance	Efficiency	Loading Speed
Cross-Browser Compatibility	Responsive Design	Animation and Interactivity	

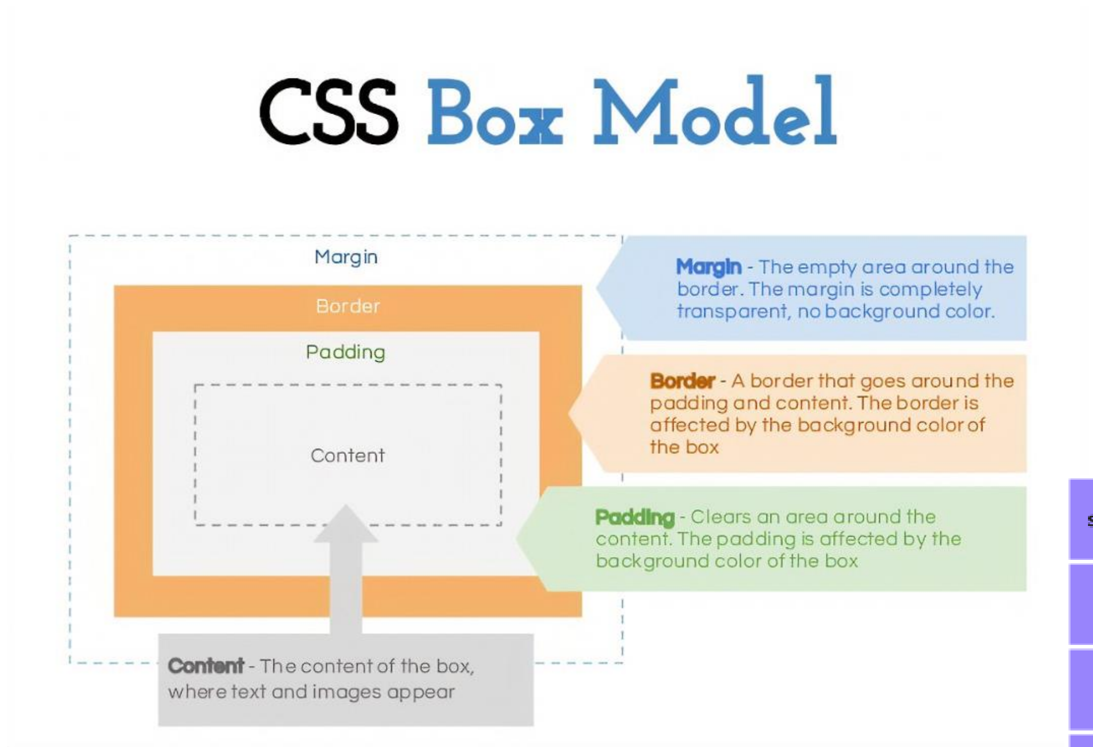
CSS

- **What are the key concepts of CSS?**

Selectors	Properties and Values	Declaration Blocks	Rules or Selectors
External, Internal, and Inline CSS	Cascading and Specificity	Inheritance	Box Model
Units of Measurement	Colors	Fonts	Layout Techniques
Responsive Design	Transitions and Animations	Pseudo-classes and Pseudo-elements	Vendor Prefixes
CSS Preprocessors			

UNDERSTANDING BOX MODEL

- **What is Box Model?**

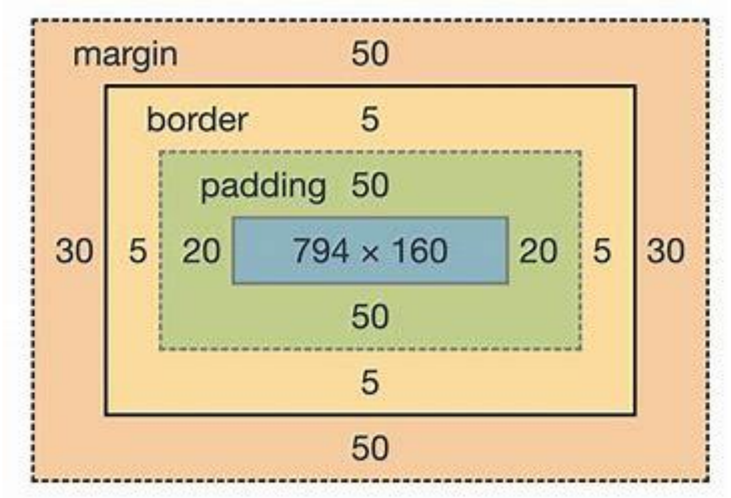


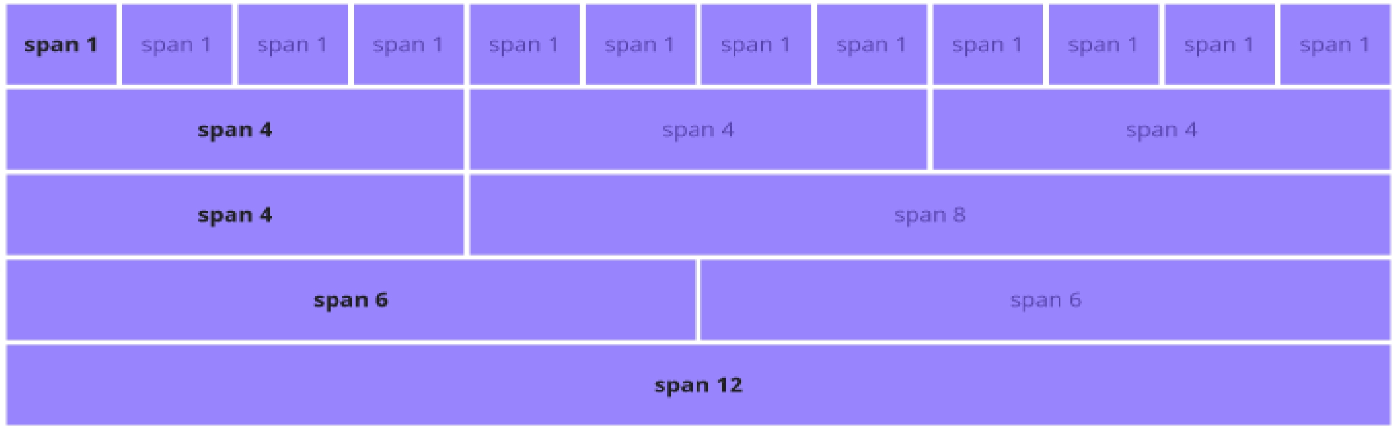
Margin - The empty area around the border. The margin is completely transparent, no background color.

Border - A border that goes around the padding and content. The border is affected by the background color of the box

Padding - Clears an area around the content. The padding is affected by the background color of the box

Content - The content of the box, where text and images appear





JAVASCRIPT

- **What is JavaScript?**
  - Programming Language. **Its primary role in web development is to provide interactivity and dynamic behavior to websites and web applications**

- **What are the key roles of JavaScript?**

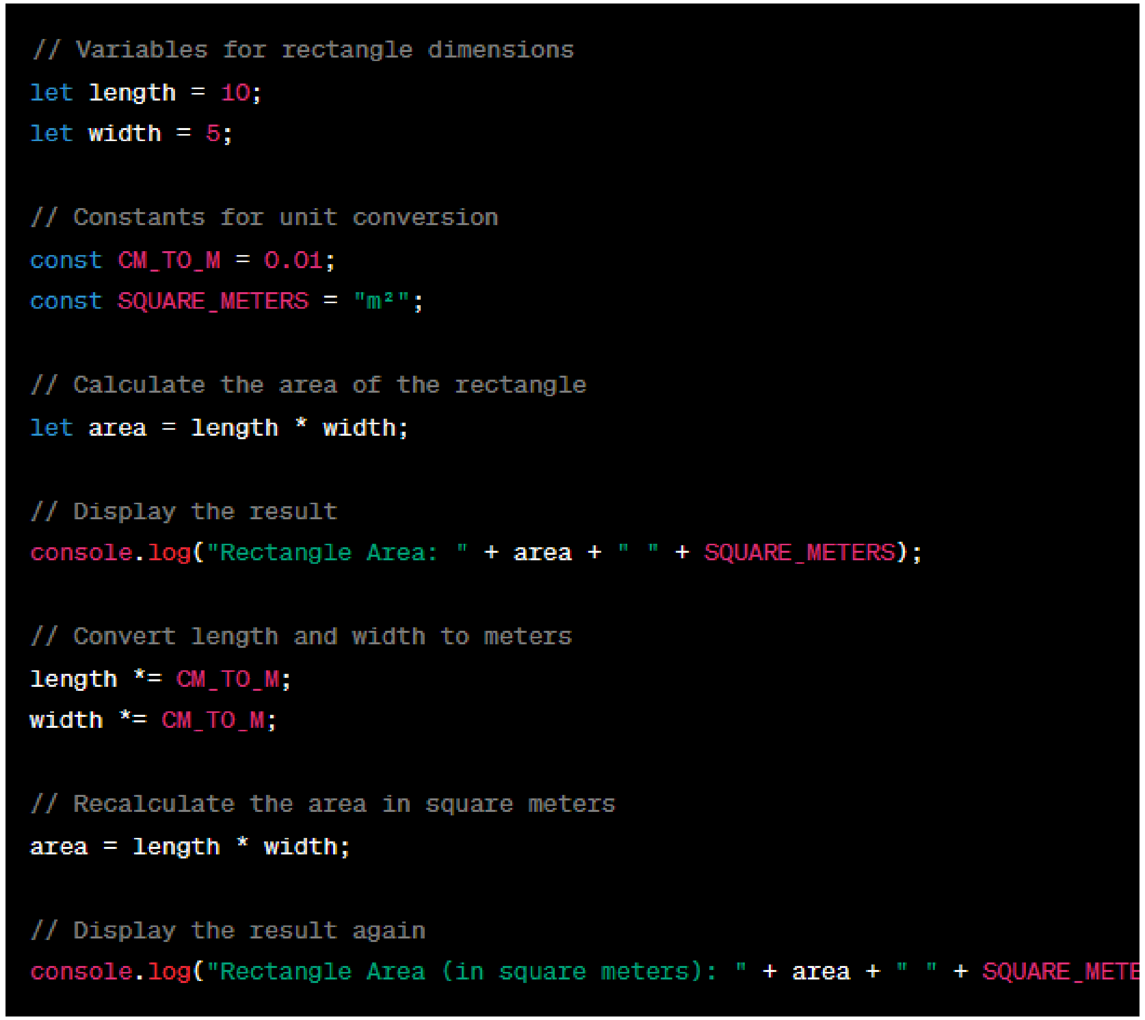
Client-Side Scripting	Enhancing User Experience	DOM Manipulation	Event Handling
AJAX (Asynchronous JavaScript and XML)	Front-End Frameworks	Browser APIs	Form Validation
Cookies and Local Storage	Security Enhancements	Data Manipulation	Dynamic Content Loading
Responsive Design	Web APIs and Integration	Progressive Web Apps (PWAs)	

JavaScript

- **What are the key concepts of JavaScript?**

Variables and Data Types	Operators	Control Structures	Functions
Scope and Closures	Arrays	Objects	DOM Manipulation
Events and Event Handling	Asynchronous Programming	Error Handling	Prototypes and Inheritance
JSON (JavaScript Object Notation)	Modules and Modularization	ES6+ Features	Promises and Async/Await
Browser APIs and Events	AJAX (Asynchronous JavaScript and XML)	Scoping and Hoisting	

JavaScript – Variables, Constants, Operators



```
// Variables for rectangle dimensions
let length = 10;
let width = 5;

// Constants for unit conversion
const CM_TO_M = 0.01;
const SQUARE_METERS = "m²";

// Calculate the area of the rectangle
let area = length * width;

// Display the result
console.log("Rectangle Area: " + area + " " + SQUARE_METERS);

// Convert length and width to meters
length *= CM_TO_M;
width *= CM_TO_M;

// Recalculate the area in square meters
area = length * width;

// Display the result again
console.log("Rectangle Area (in square meters): " + area + " " + SQUARE_METE
```

RAAFI
Builds You Best

JavaScript – Controls Structures

```javascript
let age = 18;

if (age >= 18) {
    console.log("You are eligible to vote.");
} else {
    console.log("You are not eligible to vote yet.");
}
```

```javascript
for (let i = 1; i <= 5; i++) {
    console.log("Count: " + i);
}
```

```javascript
let count = 0;

while (count < 5) {
    console.log("Current count: " + count);
    count++;
}
```

```javascript
let day = "Wednesday";

switch (day) {
    case "Monday":
        console.log("It's the start of the week.");
        break;
    case "Wednesday":
        console.log("It's the middle of the week.");
        break;
    default:
        console.log("It's some other day.");
}
```

```javascript
let person = {
    name: "Alice",
    age: 30,
    occupation: "Engineer"
};

for (let key in person) {
    console.log(key + ": " + person[key]);
}
```

```javascript
for (let i = 1; i <= 3; i++) {
    for (let j = 1; j <= 3; j++) {
        console.log("i: " + i + ", j: " + j);
    }
}
```

```javascript
let temperature = 25;
let isRaining = false;

if (temperature > 20 && !isRaining) {
    console.log("It's a great day to go outside!");
} else {
    console.log("Maybe it's better to stay indoors.");
}
```

RAAFI
Builds You Best

JavaScript – Functions

```
function calculateRectangleArea(length, width) {
    return length * width;
}

let area = calculateRectangleArea(10, 8);
console.log("Rectangle Area:", area);  // Output: Rectangle Area: 80
```

```
const multiply = function(x, y) {
    return x * y;
};

let product = multiply(3, 4);
console.log("Product:", product);  // Output: Product: 12
```

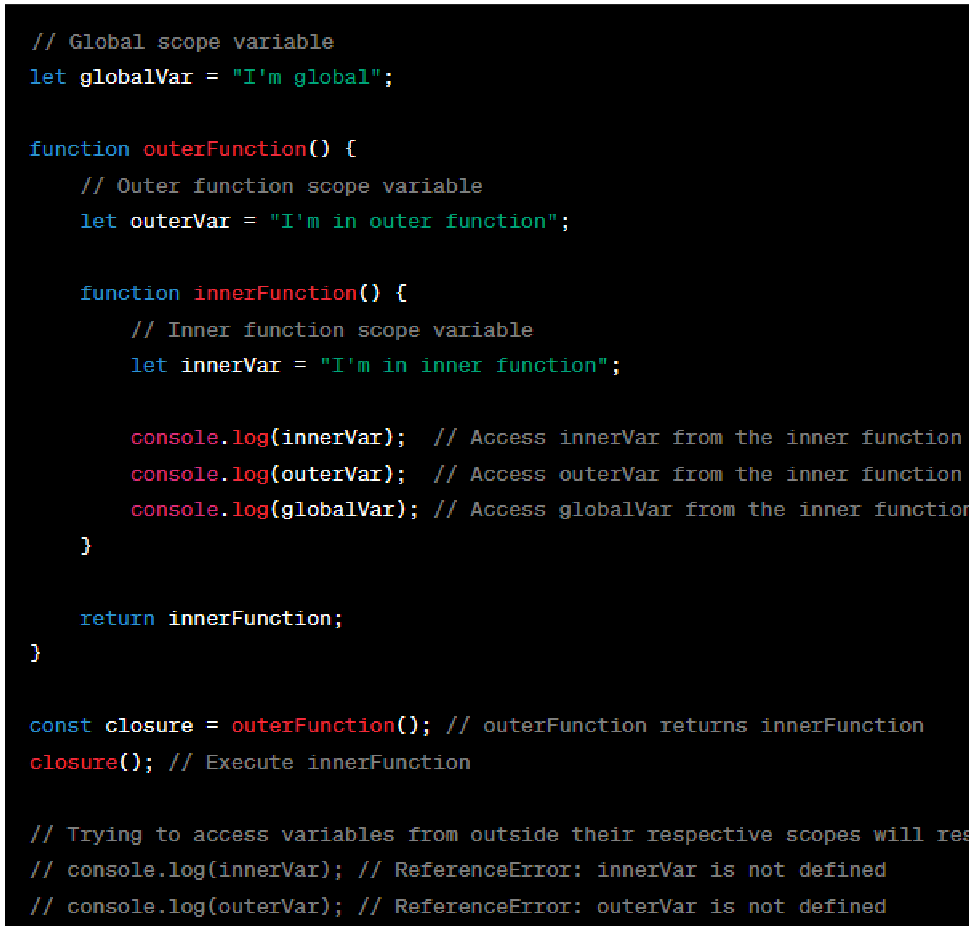
```
const divide = (a, b) => a / b;

let quotient = divide(10, 2);
console.log("Quotient:", quotient);  // Output: Quotient: 5
```

```
setTimeout(function() {
    console.log("Delayed message!");
}, 2000);  // Output after 2 seconds: Delayed message!
```

```
// A function that simulates an asynchronous operation (e.g., fetching data
function fetchData(callback) {
    setTimeout(function() {
        const data = { message: "Hello, world!" };
        callback(data);
    }, 2000); // Simulate a delay of 2 seconds
}

// A callback function to handle the fetched data
function processData(data) {
    console.log("Received data:", data.message);
}

// Call the fetchData function and provide processData as the callback
console.log("Fetching data...");
fetchData(processData);
console.log("Request sent.");
```

RAAFI
Builds You Best

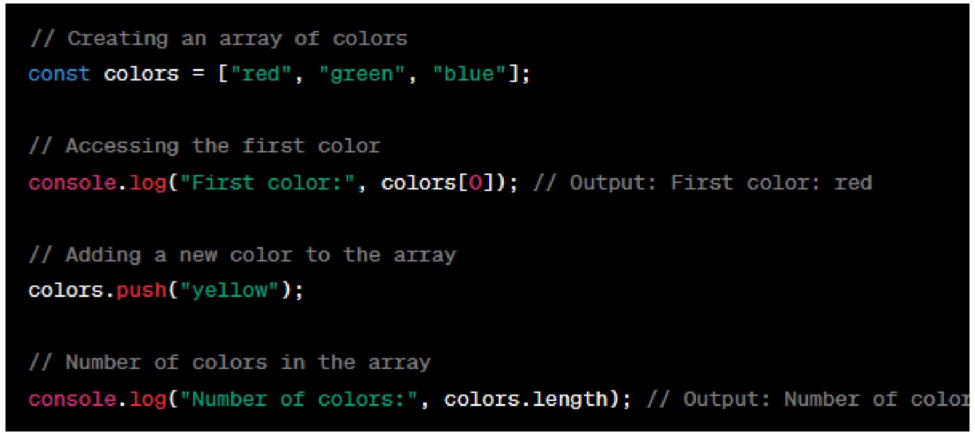
JavaScript – Scope and Closures



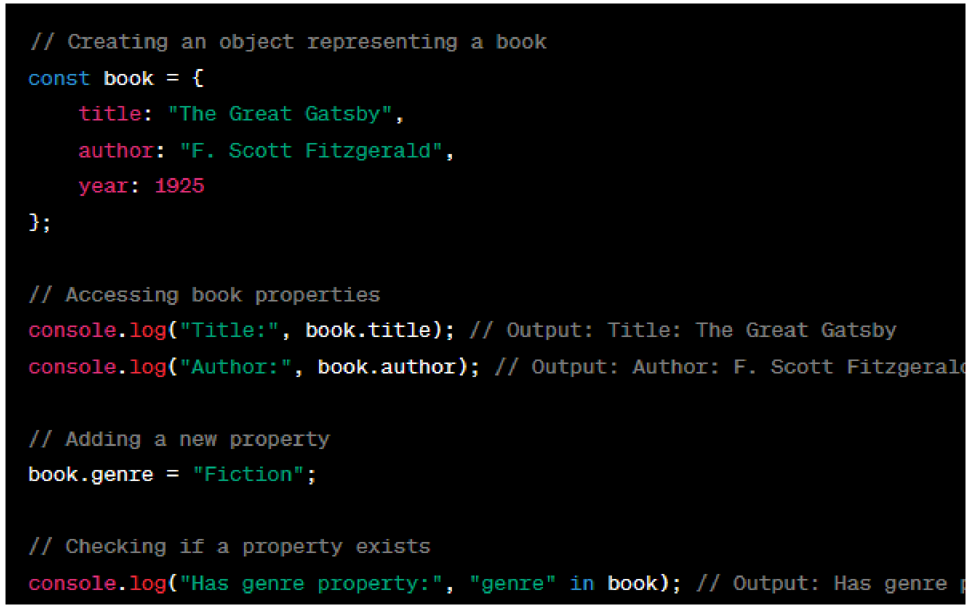
```javascript
// Global scope variable
let globalVar = "I'm global";

function outerFunction() {
    // Outer function scope variable
    let outerVar = "I'm in outer function";

    function innerFunction() {
        // Inner function scope variable
        let innerVar = "I'm in inner function";

        console.log(innerVar);  // Access innerVar from the inner function
        console.log(outerVar);  // Access outerVar from the inner function
        console.log(globalVar); // Access globalVar from the inner functio
    }

    return innerFunction;
}

const closure = outerFunction(); // outerFunction returns innerFunction
closure(); // Execute innerFunction

// Trying to access variables from outside their respective scopes will res
// console.log(innerVar); // ReferenceError: innerVar is not defined
// console.log(outerVar); // ReferenceError: outerVar is not defined
```

RAAFI
Builds You Best

JavaScript – Arrays and Objects

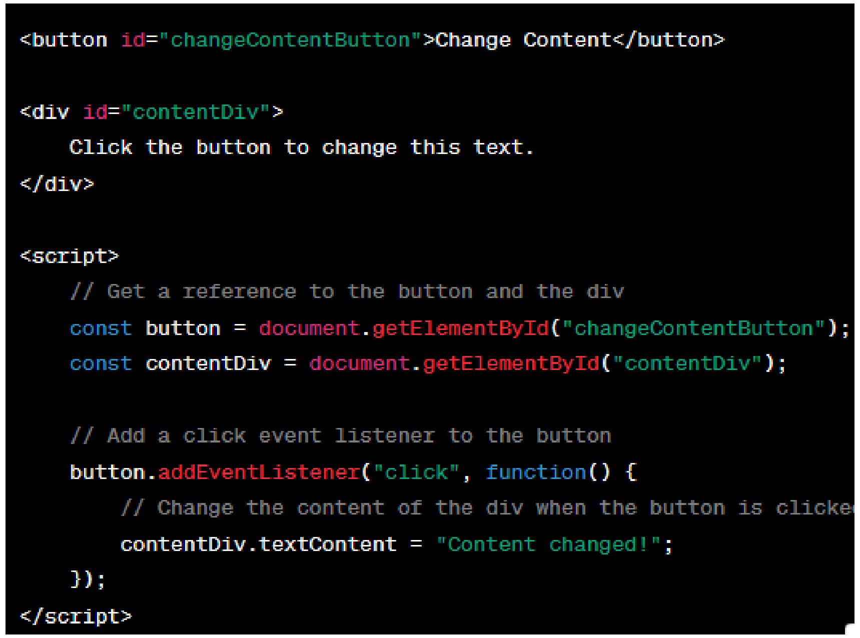


```
// Creating an array of colors
const colors = ["red", "green", "blue"];

// Accessing the first color
console.log("First color:", colors[0]); // Output: First color: red

// Adding a new color to the array
colors.push("yellow");

// Number of colors in the array
console.log("Number of colors:", colors.length); // Output: Number of color
```



```
// Creating an object representing a book
const book = {
    title: "The Great Gatsby",
    author: "F. Scott Fitzgerald",
    year: 1925
};

// Accessing book properties
console.log("Title:", book.title); // Output: Title: The Great Gatsby
console.log("Author:", book.author); // Output: Author: F. Scott Fitzgeral

// Adding a new property
book.genre = "Fiction";

// Checking if a property exists
console.log("Has genre property:", "genre" in book); // Output: Has genre
```

JavaScript – DOM Manipulation & Events Handling



```
<button id="changeContentButton">Change Content</button>

<div id="contentDiv">
    Click the button to change this text.
</div>

<script>
    // Get a reference to the button and the div
    const button = document.getElementById("changeContentButton");
    const contentDiv = document.getElementById("contentDiv");

    // Add a click event listener to the button
    button.addEventListener("click", function() {
        // Change the content of the div when the button is clicke
        contentDiv.textContent = "Content changed!";
    });
</script>
```

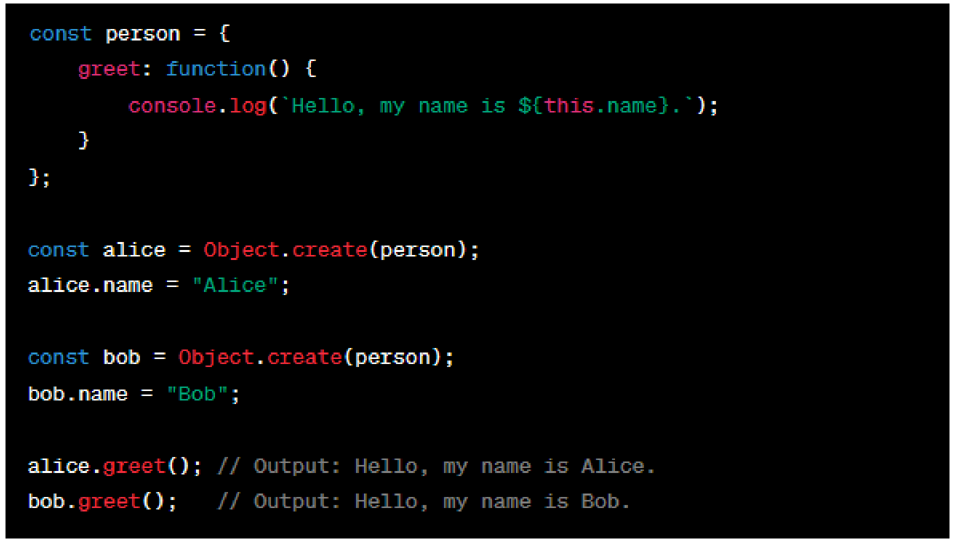
RAAFI
Builds You Best

JavaScript – Asynchronous Operations , Exception Handling

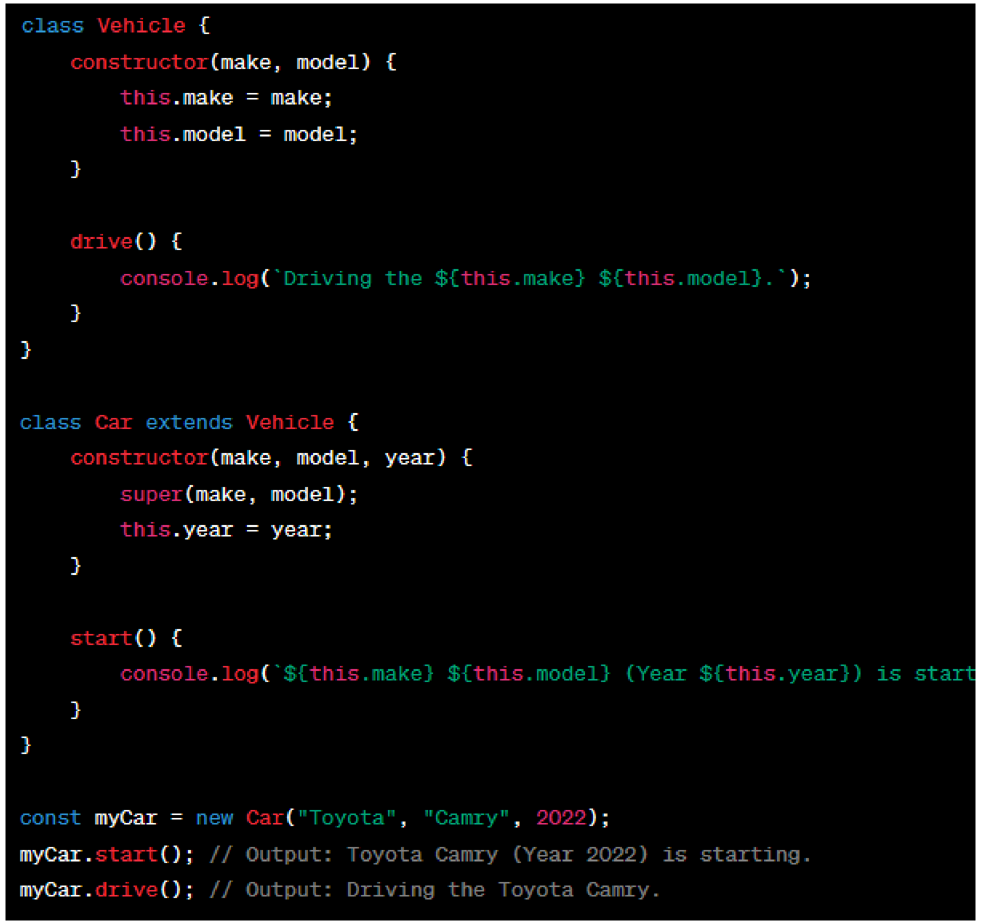
```javascript
console.log("Start of the program");

// Simulate an asynchronous operation with setTimeout
setTimeout(function() {
    console.log("Asynchronous operation completed after 2 seconds");
}, 2000);

console.log("Continuing with other tasks");
```

```javascript
try {
    // Code that might throw an error
    const result = someFunction(); // Assuming someFunction is not defined
    console.log(result); // This line will not execute
} catch (error) {
    // Code to handle the error
    console.error("An error occurred:", error.message);
} finally {
    // Code that always runs, whether there's an error or not
    console.log("Execution completed.");
}
```

JavaScript – Prototypes and Inheritance



```javascript
const person = {
    greet: function() {
        console.log(`Hello, my name is ${this.name}.`);
    }
};

const alice = Object.create(person);
alice.name = "Alice";

const bob = Object.create(person);
bob.name = "Bob";

alice.greet(); // Output: Hello, my name is Alice.
bob.greet();   // Output: Hello, my name is Bob.
```



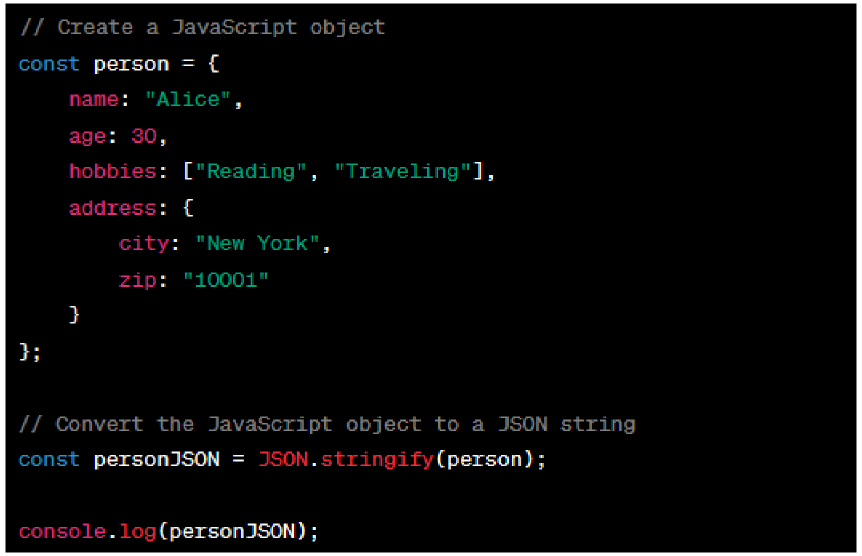
```javascript
class Vehicle {
    constructor(make, model) {
        this.make = make;
        this.model = model;
    }

    drive() {
        console.log(`Driving the ${this.make} ${this.model}.`);
    }
}

class Car extends Vehicle {
    constructor(make, model, year) {
        super(make, model);
        this.year = year;
    }

    start() {
        console.log(`${this.make} ${this.model} (Year ${this.year}) is start
    }
}

const myCar = new Car("Toyota", "Camry", 2022);
myCar.start(); // Output: Toyota Camry (Year 2022) is starting.
myCar.drive(); // Output: Driving the Toyota Camry.
```

JavaScript – JSON

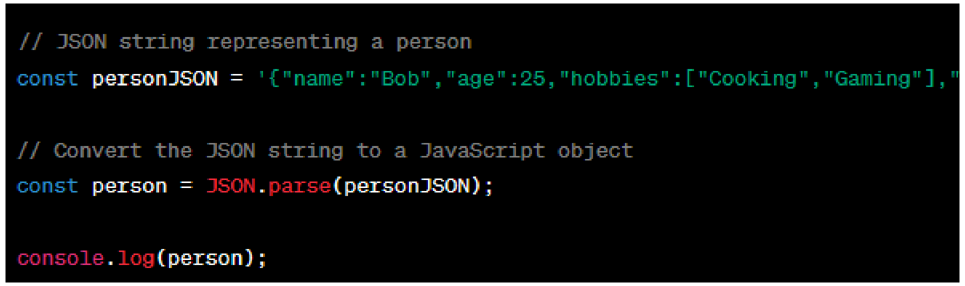
- ❑ Data is in name/value pairs
- ❑ Data is separated by commas
- ❑ Curly braces hold objects
- ❑ Square brackets hold arrays

```xml
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>
```

```json
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```



```javascript
// Create a JavaScript object
const person = {
    name: "Alice",
    age: 30,
    hobbies: ["Reading", "Traveling"],
    address: {
        city: "New York",
        zip: "10001"
    }
};

// Convert the JavaScript object to a JSON string
const personJSON = JSON.stringify(person);

console.log(personJSON);
```



```javascript
// JSON string representing a person
const personJSON = '{"name":"Bob","age":25,"hobbies":["Cooking","Gaming"],"

// Convert the JSON string to a JavaScript object
const person = JSON.parse(personJSON);

console.log(person);
```

RAAFI
Builds You Best

JavaScript – Modularization and ES6

```javascript
// mathUtils.js
function add(a, b) {
    return a + b;
}

function subtract(a, b) {
    return a - b;
}

// Export the functions as a module
module.exports = { add, subtract };
```

```javascript
// main.js
const mathUtils = require('./mathUtils.js');

const sum = mathUtils.add(5, 3);
console.log(sum); // Output: 8
```

```javascript
// ES5 function
function add(x, y) {
    return x + y;
}

// ES6 arrow function
const add = (x, y) => x + y;
```

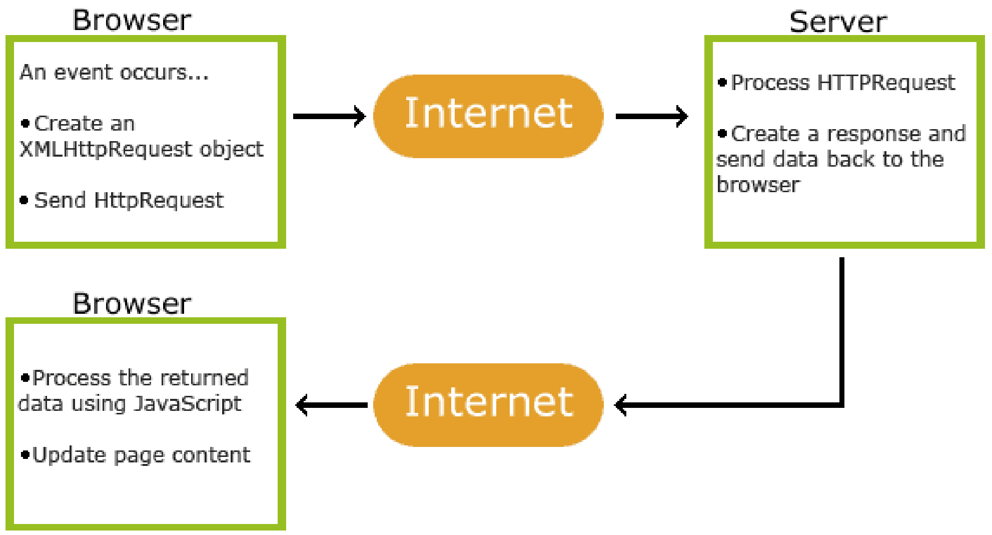
JavaScript – Callbacks, Promises and Async/Await

```javascript
function fetchData(callback) {
    setTimeout(() => {
        const data = "Data fetched successfully";
        callback(data);
    }, 1000);
}

function process(data) {
    console.log(data);
}

fetchData(process);
```

```javascript
function fetchData() {
    return new Promise((resolve, reject) => {
        setTimeout(() => {
            const data = "Data fetched successfully";
            resolve(data);
        }, 1000);
    });
}

fetchData()
    .then(data => console.log(data))
    .catch(error => console.error(error));
```

```javascript
function fetchData() {
    return new Promise((resolve, reject) => {
        setTimeout(() => {
            const data = "Data fetched successfully";
            resolve(data);
        }, 1000);
    });
}

async function fetchDataAndProcess() {
    try {
        const data = await fetchData();
        console.log(data);
    } catch (error) {
        console.error(error);
    }
}

fetchDataAndProcess();
```

JavaScript – AJAX


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

```
<form action="">
  <select name="customers" onchange="showCustomer(this.value)">
    <option value="">Select a customer:</option>
    <option value="ALFKI">Alfreds Futterkiste</option>
    <option value="NORTS ">North/South</option>
    <option value="WOLZA">Wolski Zajazd</option>
  </select>
</form>
<br>
<div id="txtHint">Customer info will be listed here...</div>

<script>
function showCustomer(str) {
  if (str == "") {
    document.getElementById("txtHint").innerHTML = "";
    return;
  }
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("txtHint").innerHTML = this.responseText;
  }
  xhttp.open("GET", "getcustomer.php?q="+str);
  xhttp.send();
}
</script>
```

JavaScript – Browser APIs

```
const paragraph = document getElementById("demo")

fetch("https://jsonplaceholder.typicode.com/posts/1")

navigator.geolocation.getCurrentPosition

localStorage getItem("username")


const audioContext = new AudioContext();
const oscillator = audioContext.createOscillator

oscillator.connect(audioContext.destination);
oscillator.start();
oscillator.stop(audioContext.currentTime + 2)
```

RAAFI
Builds You Best

JavaScript – Scoping and Hoisting

```javascript
function greet() {
    var message = "Hello, ";
    console.log(message + name);
}

var name = "Alice";
greet(); // Output: Hello, Alice
console.log(message); // Error: message is not defined (out of scope)
```

```javascript
greet(); // Works fine

function greet() {
    console.log("Hello");
}
```

```javascript
// Function Declaration (hoisted)
functionDeclaration(); // Works fine
function functionDeclaration() {
    console.log("Function Declaration");
}

// Function Expression (not hoisted)
functionExpression(); // Error: functionExpression is not a function
var functionExpression = function() {
    console.log("Function Expression");
};
```