JavaScript Assignment Questions

1. What is JavaScript ? Why do we need JavaScript ?

Answer :

JavaScript is a high-level, interpreted programming language primarily used for front-end web development. It allows developers to add interactivity, dynamic content, and behavior to websites. JavaScript is an essential component of web development alongside HTML and CSS, forming the trio of core technologies for building web pages and applications.

Key features and aspects of JavaScript include:-

1].Client-Side Scripting: JavaScript is mainly used for client-side scripting, meaning it runs in the user's browser rather than on the server. This enables dynamic changes to the content and behavior of a web page after it has been loaded.

2].Interactivity: JavaScript enables developers to create interactive elements on web pages, such as forms, buttons, sliders, and more. It allows for user input validation, feedback, and a smoother user experience.

3].Asynchronous Programming: JavaScript supports asynchronous programming, which is crucial for handling tasks that may take some time to complete, like fetching data from a server or loading images. This is achieved through features like callbacks, promises, and async/await.

4].Manipulating the Document Object Model (DOM): JavaScript can manipulate the DOM, representing the structure of a web page, allowing for dynamic updates and changes to the content and structure without requiring a full page reload.

5].Cross-Browser Compatibility: JavaScript is supported by all major web browsers, making it a versatile language for developing applications that can run on various platforms and devices.

6].Open Source and Community-driven: JavaScript has a large and active community, contributing to its constant improvement and evolution. There are numerous libraries and frameworks built on top of JavaScript, such as React, Angular, and Vue.js, which facilitate the development of complex web applications.

2. What is variables ? Explain the rules of variables with examples.

Answer :

In programming, a variable is a named storage location that holds a value or data. Variables are used to store and manipulate information within a program. Each variable has a unique identifier, or name, which is used to reference and access the stored data.

Here are the basic rules for declaring and using variables in JavaScript:

***Variable Declaration:**

1. Variables are declared using the var, let, or const keyword.

2. var was traditionally used for variable declaration, but let and const are more modern alternatives introduced in ECMAScript 6 (ES6).

3. let is used for variables whose values can be reassigned, while const is used for variables with a constant (unchangeable) value.

```
var variable1;

let variable2;

const variable3 = 10;
```

***Variable Naming Rules:**

1. Variable names must begin with a letter (a-z, A-Z), an underscore (_), or a dollar sign ($).

2. Subsequent characters can also be digits (0-9).

3. JavaScript is case-sensitive, so variableName and VariableName are treated as different variables.

```
let firstName;

let _totalAmount;
```

let $discount;

***Reserved Keywords:**

1. Avoid using reserved keywords as variable names since they have special meanings in JavaScript.

```
// Incorrect
let if = 5;


// Correct
let condition = 5;
```

***Variable Assignment:**

1.Values are assigned to variables using the assignment operator =.

```
let age = 25;
```

***Variable Scope:**

1.The scope of a variable defines where it is accessible. Variables declared with var are function-scoped, while those declared with let and const are block-scoped.

```
if (true) {
   var x = 5; // Function-scoped
   let y = 10; // Block-scoped
}
```

***Reassignment (for let):**

1.Variables declared with let can be reassigned a new value.

```
let count = 1;
count = 2; // Reassignment is allowed
```

***Constants (for const):**

1.Variables declared with const cannot be reassigned after their initial assignment.

```
const pi = 3.14;
// pi = 3.14159; // Error: Assignment to constant variable
```

These rules ensure clarity, consistency, and maintainability in code. Choose variable names that are descriptive and reflect the purpose of the data they hold. Follow these conventions to write clean and readable JavaScript code.

3. Explain Primitive data types in JS with examples.

Answer :

In JavaScript, data types are classified into two main categories: primitive data types and objects (reference types). Primitive data types are the basic building blocks of data and are immutable, meaning their values cannot be changed directly. The primitive data types in JavaScript include:

1. **String:Represents a sequence of characters enclosed in single (') or double (") quotes.**

```
let name = "John";

let message = 'Hello, World!';
```

2. **Number:Represents numeric values, including integers and floating-point numbers.**

```
let age = 25;

let pi = 3.14;
```

3. **Boolean:Represents a logical value indicating either true or false.**

```
let isStudent = true;

let hasLicense = false;
```

4. **Undefined:Represents a variable that has been declared but not assigned a value.**

```
let undefinedVar;
```

5. **Null:Represents the intentional absence of any object value.**

```
let nullValue = null;
```

6. **Symbol (ES6 and later):Represents a unique and immutable data type, often used as an identifier for object properties.**

```
let sym1 = Symbol("symbol1");

let sym2 = Symbol("symbol2");
```

7. **BigInt (ES2020 and later):Represents whole numbers larger than the maximum safe integer represented by the Number type.**

```
let bigIntNumber = 9007199254740991n;
```

These primitive data types are fundamental to JavaScript and are used to represent simple values. When you work with these types, keep in mind that they are passed by value, meaning a copy of the value is passed to functions or assigned to other variables. Additionally, primitive values are compared by value, so two variables with the same primitive value are considered equal.

```
let a = 10;
let b = 10;

console.log(a === b); // true
```

4. Explain difference between null and undefined With examples.

Answer :

In this example, the variable pi is declared as a constant with the value 3.14. If you try to reassign a new value to pi (as shown in the commented line), it will result in a TypeError, and your code will throw an error at runtime.

Constants are useful when you want to ensure that a variable retains a consistent value throughout the execution of your program. They are commonly used for values that are not meant to be changed, such as mathematical constants or configuration settings that should remain constant.

***Undefined:**

1. undefined is a primitive data type that is automatically assigned to a variable that has been declared but not yet initialized with a value.

example :let variable;

console.log(variable); // undefined

2. It is also the default return value of functions that do not explicitly return a value.

example:function noReturn() {

// No return statement

}

console.log(noReturn()); // undefined

3. In certain contexts, undefined can be explicitly assigned to a variable.

example:let explicitUndefined = undefined;

console.log(explicitUndefined); // undefined

***** Null:**

1.null is a special value that represents the intentional absence of any object value. It is often used to indicate that a variable intentionally has no value or that a function deliberately returns no value.

example:let nullVariable = null;

2.It is explicitly assigned by the developer.

example : let age = null; // Represents unknown or intentionally no value

3. Unlike undefined, null is a value that needs to be assigned explicitly, and it is not automatically assigned to uninitialized variables.

5. Write a JS program that calculates the area of a triangle.

Answer:

```
Taking user input for base and height
rl.question("Enter the base of the triangle:", function (baseInput) {
let base = Number(baseInput);

if (isNaN(base)) {
console.log("Please enter a valid number for the base.");
rl.close();
} else {
rl.question("Enter the height of the triangle:", function (heightInput) {
let height = Number(heightInput);

if (isNaN(height)) {
console.log("Please enter a valid number for the height.");
} else {
// Calculating and displaying the area
let area = calculateTriangleArea(base, height);
console.log(
`The area of the triangle with base ${base} and height ${height} is: ${area}`
);
}

rl.close();
}
});
```

6. Write a JS program in that create two variables representing a first name, middle name and last name concatenate(+) them to form a full name.

Answer:

```
Variables representing first name, middle name, and last name
let firstName = "John";
let middleName = "Doe";
let lastName = "Smith";

Concatenating the variables to form a full name
// let fullName = firstName + " " + middleName + " " + lastName;

Displaying the full name
console.log("Full Name: " + fullName);
```

7. How do you declare a constant variable in JS. What happens if you try to reassign a new value to a constant with example.

Answer :

In JavaScript, you can declare a constant variable using the const keyword. Constants are variables whose values cannot be reassigned after their initial assignment. Here's an example of how you declare and use a constant variable:

```
// Declaring a constant variable
const pi = 3.14;

// Trying to reassign a new value to the constant (will result in an error)
// Uncommenting the line below will result in a TypeError
pi = 3.14159; // Error: Assignment to constant variable
```

In this example, the variable pi is declared as a constant with the value 3.14. If you try to reassign a new value to pi (as shown in the commented line), it will result in a TypeError, and your code will throw an error at runtime.

Constants are useful when you want to ensure that a variable retains a consistent value throughout the execution of your program. They are commonly used for values that are not meant to be changed, such as mathematical constants or configuration settings that should remain constant.

```
let num1 = 5;
console.log(num1);

num1 = 2;
console.log(num1);
```