PPT Program Assignment

Web Development Assignment - 4

Answer 1 : Hoisting is a feature in JavaScript that allows variable and function declarations to be recognized and accessible throughout their containing scope, regardless of where they are physically placed in the code. It happens during the compilation phase before the code is executed, and it essentially moves these declarations to the top of their respective scopes. So, even if you declare a variable or function later in the code, you can still use it earlier in the same scope because it gets hoisted to the top.

Answer 2: Temporal Dead Zone refers to a behavior in JavaScript where variables declared with the 'let' and 'const' keywords are inaccessible or undefined within a certain scope, specifically before they are declared. This means that if you try to access such a variable before its declaration, JavaScript will throw an error. The Temporal Dead Zone helps catch potential issues with accessing variables before they are properly initialized.

Answer 3: 'var' has function scope and is hoisted, allowing access before declaration.
'let' has block scope, is not hoisted, and throws a reference error if accessed before declaration.
Using 'let' or 'const' is generally recommended for better scoping and avoiding hoisting-related issues.

Answer 4: ECMAScript 6 introduced several major features to JavaScript:
1. Block-scoped variables: 'let' and 'const' for block-level variable scoping.
2. Arrow functions: A shorter syntax for creating anonymous functions.
3. Classes: A more structured and object-oriented way to define and inherit objects.
4. Modules: A standardized way to organize and import/export code between files.

Answer 5: The main difference between 'let' and 'const' in JavaScript is how they handle variable assignment and reassignment:
1. Assignment: Variables declared with 'let' can be assigned a new value, meaning their value can be changed throughout the program. Variables declared with 'const' must be assigned a value when declared and cannot be reassigned afterwards.
2. Mutability: Variables declared with 'let' are mutable, allowing their value to be modified. Variables declared with 'const' are immutable, meaning their value cannot be changed once assigned.
3. Block scoping: Both 'let' and 'const' have block-level scoping, meaning they are only accessible within the block statement where they are defined.
4. Recommendation: It is generally recommended to use 'let' for variables that need to be reassigned, and 'const' for variables that should remain constant. Using 'const' can help enforce immutability and make the code more predictable and easier to reason about.

Answer 6: Template literals in ES6 are a way to create strings that allow for easier interpolation of variables and expressions. They are enclosed by backticks (`) instead of single or double quotes.

To use template literals, you can insert variables or expressions within `${}` placeholders directly within the template string. For example, `const name = "John"; console.log(`Hello, ${name}!`);` will output "Hello, John!".
Template literals also support multi-line strings without the need for escape characters, making it more convenient for formatting and readability.

Answer 7: The main differences between 'map' and 'forEach' in JavaScript are as follows:

1. Return value: 'map' returns a new array with the same length as the original, containing the results of applying a provided function to each element. 'forEach' does not return anything; it simply iterates over the array and executes a provided function for each element.
2. Modifying the original array: 'map' does not modify the original array; it creates a new array with the transformed values. 'forEach' does not create a new array but can modify the original array elements if the provided function makes changes.
3. Usage of return values: With 'map', you can capture and use the return values from each iteration. 'forEach' does not provide a way to access or utilize the return values.
4. Chaining: 'map' can be easily chained with other array methods, allowing for more complex transformations and operations. 'forEach' cannot be chained directly since it does not return a value.
5. Purpose: 'map' is typically used when you want to transform each element of an array into a new value and collect the results. 'forEach' is used when you want to perform an action or side effect for each element of an array without creating a new array.

Answer 8: In ES6, you can destructure objects and arrays using the following syntax:
1. Object Destructuring: To extract values from an object and assign them to variables with matching property names, use '{}' curly braces and specify the variable names inside. Example: const { name, age } = person;

2. Array Destructuring: To extract values from an array and assign them to variables in a specific order, use '[ ]' square brackets and specify the variable names inside, separated by commas. Example: const [first, second, third] = array;

Answer 9: In ES6, you can define default parameter values in functions by assigning a default value to a parameter in the function declaration.
If an argument is not passed or is 'undefined', the default value will be used.
Example: function greet(name = 'Anonymous') { ... }
When calling the function, if an argument is provided, it will override the default value, otherwise, the default value will be used.

Answer 10: The spread operator (...) in ES6 allows for the expansion of iterable objects, such as arrays or strings, into individual elements.
It is used to make copies, merge or combine arrays, pass multiple arguments to a function, or create new objects with shared properties.
It provides a concise way to work with multiple elements and simplifies array manipulation and object creation.