25 Exam Questions

🚀 Explore JavaScript's Test Your Knowledge! 🚀

TEST YOUR SKILLS

Question 1

Q: What is the purpose of the Array.prototype.map method in JavaScript?

A: The Array.prototype.map method creates a new array populated with the results of calling a provided function on every element in the calling array.

Explanation: map is used for transforming an array. It applies a function to each element of the array and then returns a new array without altering the original one.

Question 2

Q: What will be the output of the following code?

```
console.log(typeof null);
```

A: The output will be "object".

Explanation: In JavaScript, typeof null is "object", which is a known bug since the first version of JavaScript. null is not actually an object; it's a primitive value.

Question 3

Q: How does JavaScript handle asynchronous operations?
A: JavaScript handles asynchronous operations using callbacks, promises, and async/await.
Explanation: Callbacks are functions passed into other functions as arguments to be executed later. Promises represent the eventual completion (or failure) of an asynchronous operation and its resulting value. async/await is syntactic sugar built on top of promises that allows for writing asynchronous code in a more synchronous manner.

Question 4

Q: What is the difference between == and === in JavaScript?
A: == compares two variables after converting them to a common type (coercion), while === compares both value and type without conversion.
Explanation: == (equality) will perform a type conversion when comparing two things, and === (strict equality) will do the same comparison without type conversion.

Question 5

Q: What is a closure in JavaScript?
A: A closure is a function that remembers its outer variables and can access them.

Explanation: In JavaScript, closures are created every time a function is created, at function creation time. It allows a function to access and manipulate variables that are external to that function.

Question 6

Q: What is the purpose of the this keyword in JavaScript?
A: this refers to the object it belongs to. It has different values depending on where it is used: in a method, alone, in a function, in a function (in strict mode), in an event, and with call()/apply()/bind().
Explanation: this is contextually based on the object that is invoking a function. In global scope, this refers to the global object. When used in a method, this refers to the owner object.

Question 7

Q: What is event bubbling in JavaScript?
A: Event bubbling is the concept in which an event starts at the most specific element and then flows upward toward the least specific element (document object).
Explanation: In event bubbling, when an event is fired on an element that has parent elements, the event is first captured and handled by the innermost element and then propagated to outer elements.

Question 8

Q: What is the difference between var, let, and const in JavaScript?

A: var is function-scoped, let and const are block-scoped. var variables can be redeclared and updated, let variables can be updated but not redeclared, and const variables can neither be updated nor redeclared.
Explanation: var declarations are globally or function scoped, and they can be redeclared and updated. let and const are block scoped and provide better scope control. let allows updating but not redeclaration, while const prevents both redeclaration and updating.

Question 9

Q: Explain the concept of hoisting in JavaScript.
A: Hoisting is JavaScript's default behavior of moving declarations to the top.
Explanation: In JavaScript, variables and function declarations are moved to the top of their scope before code execution. However, only declarations are hoisted, not initializations. If a variable is used before it's declared, it will result in undefined.

Question 10

Q: What is the output of the following code?

```
console.log(0.1 + 0.2 === 0.3);
```

A: The output will be false.

Explanation: This is due to floating point errors in internal representation of numbers in JavaScript. JavaScript uses binary floating point format to represent numbers, which leads to precision errors in calculations. In this case, 0.1 + 0.2 doesn't exactly equal 0.3.

Question 11

Q: What is a 'Promise' in JavaScript?
A: A Promise is an object representing the eventual completion or failure of an asynchronous operation.
Explanation: It's used to handle asynchronous computations, allowing you to write cleaner, more readable asynchronous code, especially when dealing with code that requires a sequence of asynchronous operations.

Question 12

Q: How do you create a class in JavaScript?
A: Classes in JavaScript are created using the class keyword.
Explanation: A JavaScript class is a blueprint for creating objects. It encapsulates data and functions that manipulate data, and can be used to instantiate new objects with the new keyword.

Question 13

Q: What is the output of the following code snippet?

```
console.log(typeof NaN);
```

A: The output will be "number".

Explanation: NaN stands for "Not-a-Number" but is technically of type number in JavaScript. It's used to represent any computational error that returns a non-numeric result.

Question 14

Q: What is event delegation in JavaScript?
A: Event delegation is a technique of handling events by adding an event listener to a parent element instead of multiple child elements.
Explanation: This technique takes advantage of the fact that an event bubbles up through the DOM. By using event delegation, you can reduce the number of event handlers you need and improve memory usage and performance.

Question 15

Q: What is the purpose of JSON.stringify and JSON.parse in JavaScript?
A: JSON.stringify converts a JavaScript object or value to a JSON string, while JSON.parse converts a JSON string into a JavaScript object.
Explanation: These methods are widely used for working with data in JSON format, the most common data exchange format on the web.

Question 16

Q: How can you stop event propagation in JavaScript?
A: Event propagation can be stopped using the event.stopPropagation() method.

Explanation: When an event occurs, it can be propagated in two phases: capturing and bubbling. The stopPropagation() method prevents further propagation of the current event in either phase.

Question 17

Q: What is the output of the following code?

```
console.log("1" + 2 + 3);
```

A: The output will be "123".

Explanation: JavaScript performs automatic type coercion. Here, the first operation "1" + 2 turns 2 into a string and concatenates it with "1", resulting in "12". Then, "12" + 3 turns 3 into a string, resulting in "123".

Question 18

Q: What are arrow functions in JavaScript?
A: Arrow functions are a shorthand syntax for writing function expressions. They allow for shorter syntax and lexically bind the this value.
Explanation: Arrow functions are often used for short function expressions and callbacks. They do not have their own this, arguments, super, or new.target keywords and cannot be used as constructors.

Question 19

Q: What is 'destructuring assignment' in JavaScript?

A: Destructuring assignment is a syntax that allows you to unpack values from arrays or properties from objects into distinct variables.
Explanation: It provides a more concise and readable way to access array elements or object properties and assign them to variables.

Question 20

Q: What does the forEach method do in JavaScript?
A: The forEach method executes a provided function once for each array element.
Explanation: It's a method on the Array prototype that helps you iterate over the elements of an array. Unlike map or filter, forEach doesn't return a new array but can be used for operations like printing elements or applying a function to each element.

Question 21

Q: Explain the concept of 'Prototype' in JavaScript.
A: Prototypes are the mechanism by which JavaScript objects inherit features from one another. In JavaScript, every object has a prototype (a reference to another object) from which it can inherit properties and methods.

Question 22

Q: What is the difference between null and undefined in JavaScript?

A: undefined is a type that indicates that a variable has been declared but has not yet been assigned a value. null is an assignment value that represents a deliberate non-value.

Question 23

Q: How do you add an element at the beginning and end of an array in JavaScript?
A: To add an element at the beginning of an array, use the unshift() method. To add an element at the end, use the push() method.

Question 24

Q: What is 'temporal dead zone' in JavaScript?
A: Temporal Dead Zone (TDZ) is a term to describe the state where variables are in a scope but have not yet been declared. This is particularly relevant for variables declared with let and const, which are not hoisted in the same way as var.

Question 25

Q: What is the output of the following code?

```
let x = 1;
function foo() {
 x = 10;
 return;
 function x() {}
}
foo();
console.log(x);
```

A: The output will be 1.

Explanation: Due to function hoisting, the local function x() is hoisted to the top of foo(), making x within foo() a function. The assignment x = 10 is actually reassigning the local function, not the global x. The global x remains unchanged.