🚀 JavaScript Refresher Ultimate Cheat Sheet

🚀 JavaScript Refresher Ultimate Cheat Sheet

🔷 1. Variables & Scope

Declaration:

```
var a = 10; // Function-scoped
let b = 20; // Block-scoped
const c = 30; // Block-scoped, cannot be reassigned
```

Hoisting:

```
console.log(x); // undefined (var is hoisted)
var x = 5;
```

```
console.log(y); // ReferenceError (let/const are not hoisted)
let y = 10;
```

Scope Differences:

```
function test() {
  var a = 1;
  if (true) {
    let b = 2;
    const c = 3;
  }
  console.log(a); // ✅ Works
  console.log(b); // ❌ ReferenceError
```

```
  console.log(c); // ❌ ReferenceError
}
```

🔷 2. Functions & Arrow Functions

Regular Function:

```
function greet(name) {
   return `Hello, ${name}!`;
}
```

Arrow Function:

```
const greet = (name) => `Hello, ${name}!`;
```

Arrow Functions & `this`:

```
const obj = {
  name: "Alice",
  regularFunction: function () {
    console.log(this.name);
  },
  arrowFunction: () => console.log(this.name),
};
obj.regularFunction(); // ✅ Alice
obj.arrowFunction(); // ❌ undefined (Arrow functions do not have their own `t
his`)
```

🔷 3. Destructuring & Spread Operator

Object Destructuring:

```
const user = { name: "Bob", age: 25 };
const { name, age } = user;
console.log(name, age); // Bob 25
```

Array Destructuring:

```
const numbers = [1, 2, 3];
const [first, second] = numbers;
console.log(first, second); // 1 2
```

Spread Operator (...)

```
const arr1 = [1, 2, 3];
const arr2 = [...arr1, 4, 5];
console.log(arr2); // [1, 2, 3, 4, 5]
```

```
const obj1 = { a: 1, b: 2 };
const obj2 = { ...obj1, c: 3 };
console.log(obj2); // { a: 1, b: 2, c: 3 }
```

🔷 4. Array & Object Methods

Array Methods:

```
const numbers = [1, 2, 3, 4, 5];

const doubled = numbers.map(num => num * 2); // [2, 4, 6, 8, 10]
const evens = numbers.filter(num => num % 2 === 0); // [2, 4]
const sum = numbers.reduce((acc, num) => acc + num, 0); // 15
```

Object Methods:

```
const user = { name: "Alice", age: 25 };
console.log(Object.keys(user)); // ["name", "age"]
console.log(Object.values(user)); // ["Alice", 25]
console.log(Object.entries(user)); // [["name", "Alice"], ["age", 25]]
```

🔷 5. Optional Chaining & Nullish Coalescing

```
const user = { profile: { email: "test@example.com" } };
console.log(user?.profile?.email); // "test@example.com"
console.log(user?.address?.street); // undefined (No error!)
```

```
const value = null;
console.log(value ?? "Default Value"); // "Default Value"
```

🔷 6. ES6+ Features Summary

Feature	Example
Template Literals	`Hello, ${name}!`
Arrow Functions	`const sum = (a, b) => a + b;`
Destructuring	`const { name } = user;`
Spread Operator	`const newArr = [...arr1, 4, 5];`
Array Methods	`arr.map(x => x * 2);`
Optional Chaining	`obj?.property?.value;`
Nullish Coalescing	`value ?? "default";`