ES6 Super Detailed Notes With Multiple Examples + Interview Q&A;

---

1. let & const (with 3 examples)

let (block-scoped, can be reassigned)

Example 1:

```
{
let a = 10;
a = 20; // valid
}
```

Example 2:

```
for (let i = 0; i < 3; i++) {
console.log(i);
}
```

Example 3:

```
if (true) {
let x = "inside block";
console.log(x);
}
```

const (cannot be reassigned but mutable objects)

Example 1:

```
const num = 10;
// num = 20 ■ invalid
```

Example 2:

```
const arr = [1,2,3];
arr.push(4); // valid

Example 3:
const obj = {name: "Pushpa"};
obj.name = "Ram"; // valid
```

---

2. Arrow Functions (with 3 examples)

```
## Example 1: Basic Syntax
const add = (a, b) => a + b;

## Example 2: Multi-line Arrow Function
const greet = (name) => {
return `Hello ${name}`;
};

## Example 3: Arrow Function Without Own "this"
const person = {
name: "Pushpa",
show: () => console.log(this.name) // undefined because arrow functions do not bind 'this'
};
```

---

3. Template Literals (3 examples)

```
## Example 1: String interpolation
const name = "Pushpa";
```

console.log(`My name is ${name}`);

Example 2: Multi-line strings

console.log(`Line 1

Line 2

Line 3`);

Example 3: Embedded expressions

console.log(`2 + 3 = ${2+3}`);

---

4. Destructuring (3 examples)

Example 1: Array destructuring

const numbers = [10, 20, 30];

const [a, b] = numbers;

Example 2: Object destructuring

const student = {name: "Pushpa", age: 21};

const {name, age} = student;

Example 3: Nested destructuring

const data = {profile: {id: 101, score: 90}};

const {profile: {id, score}} = data;

---

5. Spread Operator (3 examples)

Example 1: Array expansion

const arr1 = [1, 2];

```
const arr2 = [...arr1, 3, 4];
```

Example 2: Object copy

```
const obj1 = {a: 10};
const obj2 = {...obj1, b: 20};
```

Example 3: Function arguments

```
function sum(a, b, c) {
return a + b + c;
}
const nums = [1,2,3];
sum(...nums);
```

---

6. Classes (3 examples)

Example 1: Basic class

```
class Person {
constructor(name) {
this.name = name;
}
}
```

Example 2: Method inside class

```
class Student {
constructor(name, marks) {
this.name = name;
this.marks = marks;
```

```
}
show() {
return `${this.name} scored ${this.marks}`;
}
}
```

Example 3: Class inheritance

```
class Animal {
speak() { return "Sound"; }
}
class Dog extends Animal {
speak() { return "Bark"; }
}
```

DETAILED ES6 INTERVIEW QUESTIONS & ANSWERS

Q1. Difference between var, let, and const? (VERY DETAILED)

Answer:

var

- Function scoped.
- Can be redeclared.
- Gets hoisted with default value `undefined`.

Example:

```
function test() {
```

console.log(a); // undefined

var a = 10;

}

let

- Block scoped.

- Cannot be redeclared.

- Hoisted but stays in "Temporal Dead Zone".

Example:

{

console.log(x); // ■ ReferenceError

let x = 5;

}

const

- Block scoped.

- Must be initialized immediately.

- Cannot be reassigned.

Example:

const pi = 3.14;

// pi = 4 ■ not allowed

Q2. Explain Arrow Functions in detail with examples.

Answer:

Arrow functions provide shorter syntax and do NOT bind their own `this`.

Example 1: Short Syntax

```
const add = (a, b) => a + b;
```

Example 2: No "this" binding

```
function Normal() {
this.value = 20;
setTimeout(function() {
console.log(this.value); // undefined
}, 1000);
}

function Arrow() {
this.value = 20;
setTimeout(() => {
console.log(this.value); // 20 (uses lexical this)
}, 1000);
}
```

Example 3: Return Objects

```
const getObj = () => ({id: 1});
```

Q3. Explain Promises with detailed real-world example.

Answer:

A Promise represents a task that will complete sometime in the future.

States:

- Pending

- Resolved

- Rejected

Real-world Example:

Like ordering food:

- You place order → pending

- Food delivered → resolved

- Food canceled → rejected

Example:

```
const orderFood = new Promise((resolve, reject) => {

const delivered = true;

if(delivered) resolve("Food delivered!");

else reject("Order failed");

});

orderFood.then(res => console.log(res))

.catch(err => console.log(err));
```

Q4. What is async/await? Explain with examples.

Answer:

Async/await is built on top of promises and makes asynchronous code look synchronous.

Example 1:

```
async function getData() {

let res = await fetch("https://jsonplaceholder.typicode.com/todos/1");

let data = await res.json();
```

console.log(data);

}

Example 2:

async function demo() {

await new Promise(res => setTimeout(res, 2000));

console.log("Finished after 2 seconds");

}

Q5. What is destructuring and why is it useful?

Answer:

Destructuring allows extracting values from objects/arrays easily.

Example 1: Array

const [a,b] = [10,20];

Example 2: Object

const {name, age} = {name: "Pushpa", age: 21};

Example 3: Function Parameters

function show({name, age}) {

console.log(name, age);

}
