WEB APP DEVELOPMENT WITH REACTJS (INT252)

Lecture 1: ES6 JavaScript Essentials for React

Unit I – JavaScript Refresher & React Foundations

Syllabus Mapping:

- Modern JavaScript (ES6+)
- JavaScript features required for React

Target Learner: Beginner (HTML, CSS, JavaScript knowledge)

Why JavaScript Matters Before React

React is **not a new language** ❌

React is:

- A **JavaScript library**
- Built completely on **modern JavaScript**

👉 If ES6 is weak, React will feel confusing

What is ES6?

ES6 (ECMAScript 2015) introduced:

- Cleaner syntax
- Better variable handling
- Functional programming features

React **expects** ES6 usage

Variables: `var` vs `let` vs `const`

`var` (Old – Avoid)

- Function scoped
- Causes bugs

`let`

- Block scoped
- Value can change

`const`

- Block scoped
- Value **cannot be reassigned**

Example: let vs const

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

const name = "React";
// name = "JS" ❌ error
```

👉 In React, **use `const` by default**

Why React Prefers `const`

- Prevents accidental reassignment
- Makes code predictable
- Encourages immutability

React mindset = **no unexpected changes**

Arrow Functions

Traditional function:

```
function add(a, b) {
  return a + b;
}
```

Arrow function:

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

Why Arrow Functions in React

- Shorter syntax
- Cleaner components
- No confusion with `this`

React components are **mostly arrow functions**

Arrow Function in React Component

```
const Hello = () => {
  return <h1>Hello React</h1>;
};
```

This is a valid React component

Template Literals

Old way:

```
const msg = "Hello " + name;
```

ES6 way:

```
const msg = `Hello ${name}`;
```

Why Template Literals Matter in React

Used heavily in:

- JSX expressions
- Dynamic text
- Styling (later)

Default Parameters

```
function greet(name = "User") {
  return `Hello ${name}`;
}

greet(); // Hello User
```

Useful when props are missing

Destructuring (Preview)

```
const user = { name: "Aman", age: 20 };

const { name, age } = user;
```

You will see this **everywhere in React**

Why Destructuring is Important

Without destructuring:

```
props.name
props.age
```

With destructuring:

```
const { name, age } = props;
```

Cleaner components

ES6 Modules (Import / Export)

```
export default function App() {}
```

```
import App from './App';
```

React apps are **modular**

How React Uses ES6 Modules

- Each component = separate file
- Import only what you need
- Better organization

Visual Summary (Flow)

```
Modern JavaScript
        ↓
   ES6 Features
        ↓
   React Syntax
        ↓
   Clean Components
```

Practice Exercises

1. Convert a normal function to arrow function
2. Use template literals to print your name
3. Declare variables using `let` and `const`

✅ Answers – Practice Exercises

1. Arrow Function

```
const square = n => n * n;
```

✅ Answers – Continued

2. Template Literal

```
const name = "Student";
console.log(`Hello ${name}`);
```

✅ Answers – Continued

3. let vs const

```
let age = 20;
age = 21;

const course = "React";
```

Key Takeaways

- React depends on ES6
- Prefer `const` over `let`
- Arrow functions simplify code
- Template literals are everywhere
- ES6 modules power React structure

Next Lecture

Lecture 2: Arrays, Objects & Immutability

Unit I – JavaScript Refresher & React Foundations