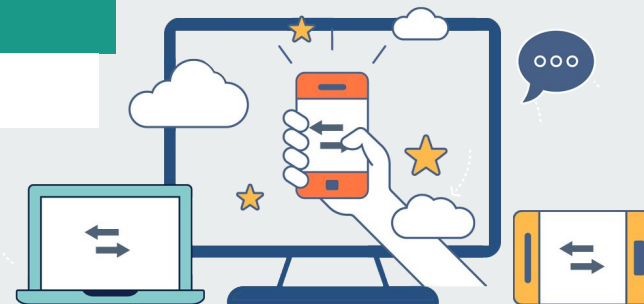


-Lecture 5- Chapter 3 – Introduction to React.JS

Part I

Adil **CHEKATI**, PhD

adil.chekati@univ-constantine2.dz





Prerequisites

- ❑ JavaScript Fundamentals (Knowledge and Application)
- ❑ Front-End Development Basics (Application)
- ❑ Understanding of ES6 (Knowledge)



Introduction to React.JS

Objectives

- Introducing Front End Frameworks and React
- Understanding React Components and their types
- Understanding JSX and its use
- Understanding styling in React

1. Introduction to front-end frameworks

- ❑ Front-end development refers to the creation of user interfaces and the implementation of user interactions on websites and web applications.
- ❑ Over the years, web development has evolved from simple static pages to complex, dynamic applications.
- ❑ **The need for efficient and structured front-end development practices grew.**



The rise of *front-end frameworks*, which provide developers with a set of **pre-written** code and **tools** to simplify the process of building robust and intuitive user interfaces.

1. Introduction to front-end frameworks

1.1. What is a Front-End Framework?

- Collection of HTML, CSS, and JavaScript files that provide a base structure and functionality to build web applications.
- Consist of **reusable** and **customizable** components, styles, and utilities that follow best practices and conventions.



1. Introduction to front-end frameworks

1.2. Why using Front-End Framework?

a) Rapid Development

With the availability of multiple ready-to-use elements, developers can focus on implementing business logic and user-specific features rather than reinventing the wheel.

b) Consistent User Experience

By utilizing the style guides and predefined UI components provided by the framework, developers can maintain a standardized look and feel across an application.

c) Responsiveness and Compatibility

Responsive design built into frameworks components and layouts, making it easier to create applications that adjust accordingly.

d) Community Support and Documentation

Frameworks are widely adopted by developers worldwide, resulting in thriving communities and extensive documentation

1. Introduction to front-end frameworks

1.3. Popular Front-End Frameworks



1. Introduction to front-end frameworks

1.3. Popular Front-End Frameworks

Framework	Strengths	Weaknesses
Angular	Complete solution	Steeper learning curve
	Scalability	Performance with large datasets
	CLI tooling	Architecture may not fit all projects
React	Excellent performance	Requires additional libraries
	Reusability	JSX syntax may be unfamiliar
	Strong ecosystem	Lack of official opinions on architecture
Vue.js	Easy to learn and integrate	Less widely adopted
	Clear syntax	Flexibility can lead to inconsistencies
	Documentation and community	May not suit very large-scale projects

2. REACT framework



- ❑ Open-source JavaScript library that is used for building user interfaces (UIs) for web applications.
- ❑ Developed by Facebook and is now maintained by Facebook and a community of individual developers and companies.
- ❑ Allows developers to create reusable UI components and efficiently update and render those components when the data changes.



React.JS is often referred to as a "JavaScript framework," although it is **technically a library**.

2. REACT framework

2.2. Why React.JS?

a) Component-Based Architecture

React.JS enables developers to break down the UI into independent and reusable components.

b) Virtual DOM

In-memory representation of the actual DOM.

c) Declarative Syntax

Developers can describe what they want the UI to look like, and React.JS takes care of updating the actual DOM to match that description.

d) Ecosystem and Community Support

Community around React.JS is highly active, providing support, regularly releasing updates, and continuously improving the library.

2. REACT framework

2.3. Key Concepts in React.JS

a) Components

Building blocks of React.JS applications.

Allow developers to split the UI into independent and reusable parts, each responsible for its own rendering and behavior.

b) Props (properties)

Used to pass data from a parent component to its child components. Immutable.

c) State

Internal data and state of a component. Mutable.

d) Lifecycle Methods

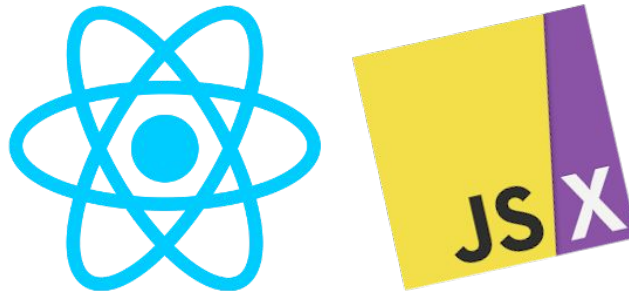
enable developers to hook into specific moments in a component's lifecycle

2. REACT framework

2.3. Key Concepts in React.JS

e) JSX

Syntax extension for JavaScript that allows developers to write HTML-like code within JavaScript. It is used by React.JS to define the structure and appearance of components.



2. REACT framework

2.4. React Separation of Concerns

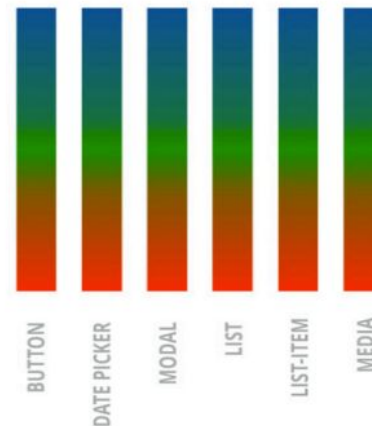
Unlike a traditional web application, React does not divide a web application into separate files: HTML, CSS and JavaScript, but **integrates** these three types into components.

Separation of Concerns



Separation of Concerns

(only, from a different point of view)



3. REACT Components



- ❑ Reusable piece of code that describes the appearance and behavior of a part of the user interface.
- ❑ Allows developers to break your user interface into smaller, independent parts, making it easier to manage and maintain your code.
- ❑ Created using: JavaScript functions or ES6 classes (functional components and class components).

3. REACT Components


a) Functional components

- ❑ Simple JavaScript functions that receive props (short for properties) as input and return a React element.
- ❑ Typically used for presentational and stateless components, as they are easy to read, test, and understand.

b) Class components

- ❑ ES6 classes that extend the `React.Component` class.
- ❑ Additional features compared to functional components: use lifecycle methods and manage internal state.
- ❑ Used for more complex components that need to maintain their own state.

3. REACT Components



```
1 // Class components
2 class FreePalestine extends React.Component {
3   render() {
4     return <h2>Palestine, will be FREE!</h2>;
5   }
6 }
7
8 // Functional components
9 function FreePalestine() {
10   return <h2>Palestine, will be FREE!</h2>;
11 }
```


Lab Exercises Submission Guidelines

- **Deadline:**
At the end of each Lab session (no later than Saturday at 23:59)
To: adil.chekati@univ-constantine2.dz
- **Link to be submitted:**
Github repository link.



Textbook

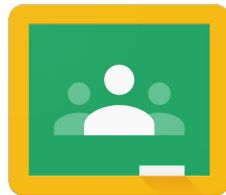
→ All academic materials will be available on:

Google Drive.

E-learning platform of Constantine 2 University.

Google Classroom.

aoa5lne



Google Classroom





References



Book:

Alex Banks, Eve Porcello - *Learning React: Modern Patterns for Developing React Apps* (2020)

MOOC

React - The Complete Guide (incl. Hooks, React Router, Redux)" on Udemy

<https://github.com/PacktPublishing/React---The-Complete-Guide-includes-Hooks-React-Router-and-Redux-Second-Edition>

Online Resource:

React.js official documentation

<https://react.dev/learn>



Next Lecture

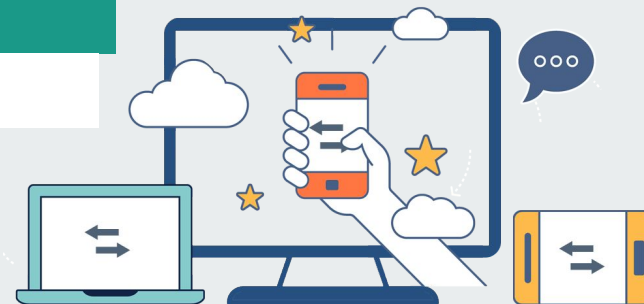
-Lecture 6-

Chapter 3 – Introduction to React.JS

Part II

Adil **CHEKATI**, PhD

adil.chekati@univ-constantine2.dz



Questions, & comments...

 adil.chekati@univ-constatine2.dz
