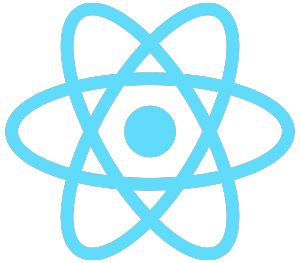
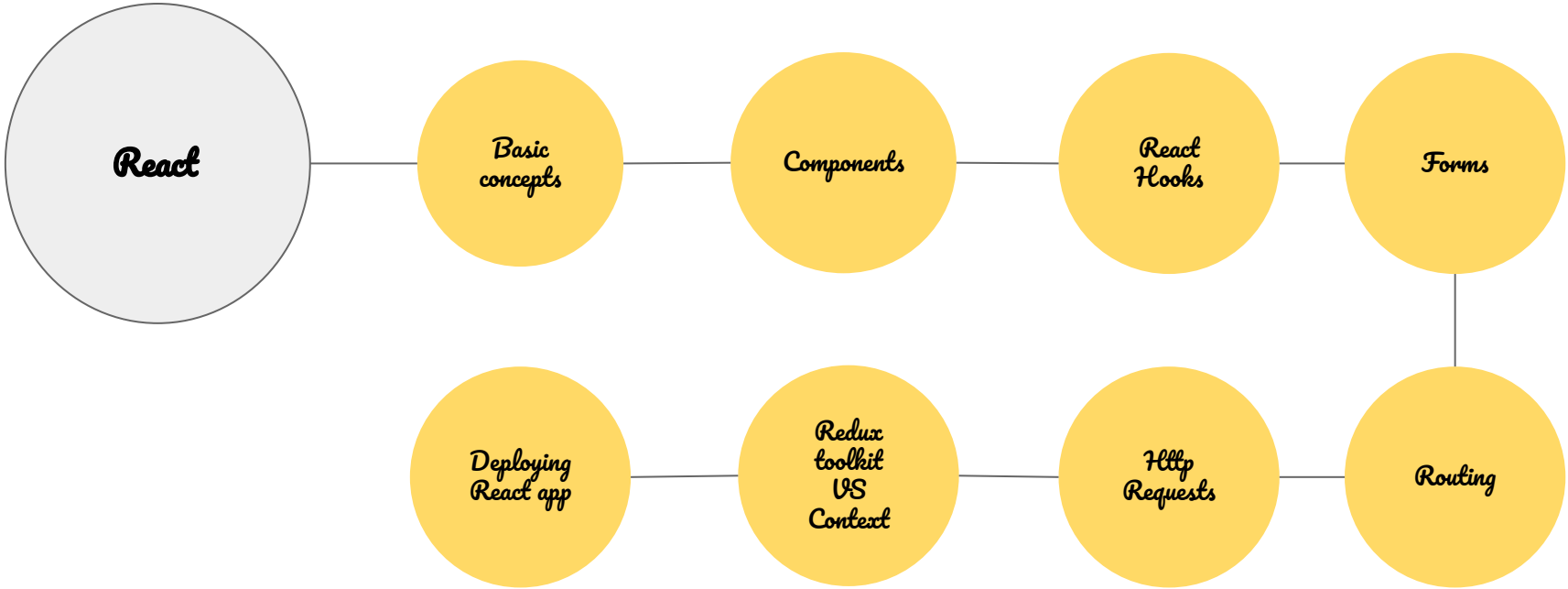


React.js

Lecture 1

Course Roadmap



Agenda

- Introduction about React
- Single page application
- Virtual Dom
- Environment setup.
- React app structure.
- Building reusable components.
- What is JSX?
- Handling events
- Conditional render and rendering list

React

React is a JavaScript library for building user interfaces, developed at Facebook and released to the world in 2013.

Current stable version : 19.0.0

React ...

React is a JavaScript library for rendering user interfaces (UI). UI is built from small units like buttons, text, and images. React lets you combine them into reusable, nestable components. From web sites to phone apps, everything on the screen can be broken down into components.

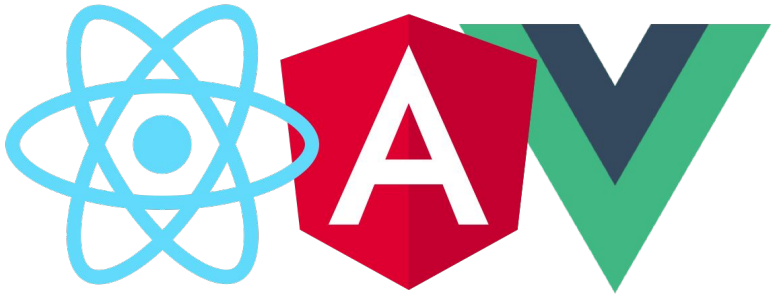
Why React ?

- Easy creation of dynamic applications.
- Improved performance using Virtual DOM.
- Reusable components
- Easy to learn.
- Dedicated tools for easy debugging.

React Alternates

Stack overflow Survey :

https://survey.stackoverflow.co/2023/#most-popular-technologies-webframe

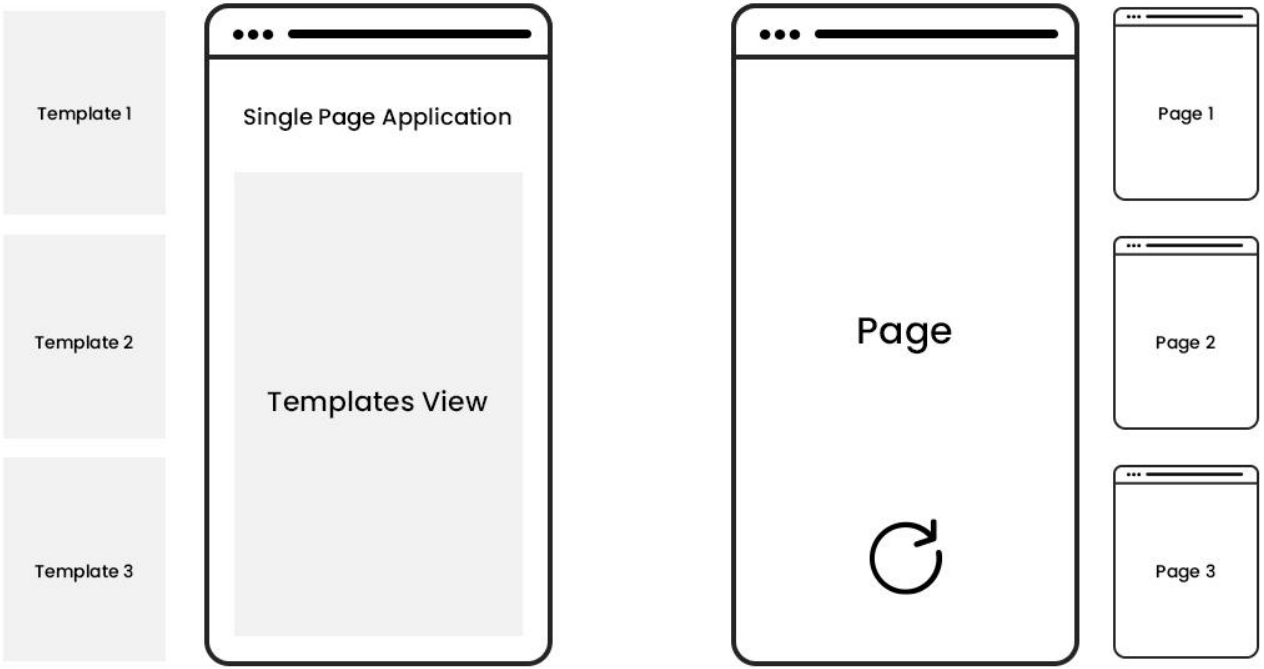


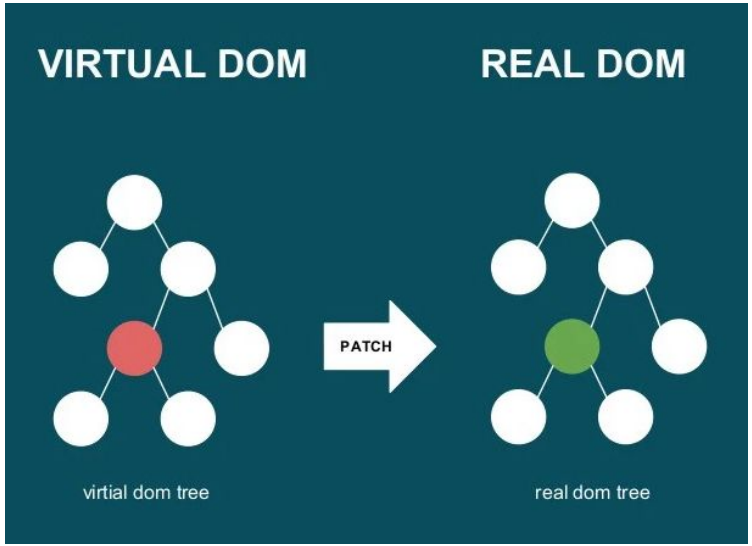
Single page application

- An SPA (Single-page application) is a web app implementation that loads only a single web document, and then updates the body content of that single document via JavaScript code.

- This therefore allows users to use websites without loading whole new pages from the server, which can result in performance gains and a more dynamic experience

Single Page Application





Virtual DOM

The virtual DOM is only a virtual representation of the DOM

- When new elements are added to the UI, a virtual DOM, which is represented as a tree is created.
- Each element is a node on this tree. If the state of any of these elements changes, a new virtual DOM tree is created.
- This tree is then compared with the previous virtual DOM tree, the virtual DOM calculates the best possible method to make these changes to the real DOM. This ensures that there are minimal operations on the real DOM.

https://adhithiravi.medium.com/react-virtual-dom-explained-in-simple-english-fc2d0b277bc5

Getting started: Using Vite

- Install node : https://nodejs.org/en/
- Open a new terminal in your directory.
- Using vite npm create vite@latest
- Then select the react app to be installed
- Once app installed, Enter your app directory
- Run the command, npm install, then npm run dev.

https://vite.dev/guide/

Getting started: or using npx

- Install node : https://nodejs.org/en/
- Open a new terminal in your directory.
- Create new react app : npx create-react-app app-name
- Enter your app folder.
- Run your React app: npm start

https://www.freecodecamp.org/news/npm-vs-npx-whats-the-difference/

Let's explore our

React App Structure

Structure

For the project to build, these files must exist with exact filenames:

- index.html is the page template;
- src/main.jsx is the JavaScript entry point.

You can delete or rename the other files.

What are components?

Components

- React applications are built from isolated pieces of UI called components. A React component is a JavaScript function that you can sprinkle with markup. Components can be as small as a button, or as large as an entire page.
- React component is a JavaScript function that you can sprinkle with markup.

Example : https://miro.medium.com/v2/resize:fit:1400/1*NX0GtVytAI8soIUMAuUSeQ.png

React components are regular JavaScript functions except:

- Their names always begin with a capital letter.
- They return JSX markup.

What is JSX ..?

- JSX stands for JavaScript XML and allows us to write HTML in React.
- With JSX :

  ```
  const myelement = <h1>I Love JSX!</h1>;

  ReactDOM.render(myelement, document.getElementById('root'));
  ```

- Without JSX :

  ```
  const myelement = React.createElement('h1', {}, 'I do not use JSX!');

  ReactDOM.render(myelement, document.getElementById('root'));
  ```

Try : https://babeljs.io/

The Rules of JSX

1. **Return a single root element**
    - To return multiple elements from a component, wrap them with a single parent tag.
    - If you don't want to add an extra to your markup, you can write <> and </> instead
2. **Close all the tags**
    - JSX requires tags to be explicitly closed: self-closing tags like <img> must become <img />
3. **camelCase all most of the things!**
    - JSX turns into JavaScript and attributes written in JSX become keys of JavaScript objects.like class would be className

JSX converter: https://transform.tools/html-to-jsx

Handling Events

Handling events with React elements is very similar to handling events on DOM elements.

Example :

```
<button onClick={addUsers}>
        Add Users
</button>

<button onClick={() => addUser()}>
        Add Users
</button>
```

Rendering Lists

Rendering Lists

- You will often want to display multiple similar components from a collection of data. You can use the JavaScript array methods to manipulate an array of data like filter() and map() with React to filter and transform your array of data into an array of components.

- You need to give each array item a key — a string or a number that uniquely identifies it among other items in that array.

- Keys tell React which array item each component corresponds to, so that it can match them up later. This becomes important if your array items can move (e.g. due to sorting), get inserted, or get deleted. A well-chosen key helps React infer what exactly has happened, and make the correct updates to the DOM tree.

https://react.dev/learn/rendering-lists#why-does-react-need-keys

Conditional Rendering

Conditional Rendering

Your components will often need to display different things depending on different conditions. In React, you can conditionally render JSX using JavaScript syntax like if statements, &&, and ? : operators.

Thank you

Lap

Task: Portfolio

Create a web page using React that contains the following sections :

- Hero section
- Bio section , button to download CV.
- Skills section [HTML , CSS ...]
- Education Section
- Contacts and social media icon links such as facebook , twitter , linkedin , github (fontawesome - [Bonus]) - could be in the hero section or footer or the side section as UI.

Using Bootstrap as UI library (search for it) and create reusable components for your page.