FRONTEND DEVELOPMENT OVERVIEW

PROGRAMMING APPLICATIONS AND FRAMEWORKS (IT3030)

LEARNING OUTCOMES

- After completing this lecture, you will be able to,
  - Describe the main concepts associated with frontend web development.
  - Apply the knowledge gained in developing more robust frontend web applications.
  - Apply the learned concepts to create more secure web applications.

CONTENTS

- What is frontend development?
- Why is frontend development important?
- HTML, CSS, JS
- Document Object Model (DOM)
- State in Web apps
  - State Management
- Web Application Security
  - OWASP Top 10 Security Risks & Vulnerabilities
  - HTTP Security Headers
- Wrap up
- Summary

WHAT IS FRONTEND DEVELOPMENT?

- Frontend development is the development of visual and interactive elements of a website that users interact with directly.
- Also known as Client-side development.

WHY IS FRONTEND DEVELOPMENT IMPORTANT?

- Backend vs. Frontend
  - Frontend web development focuses on **creating a good look and feel for the user.**
  - Backend web development focuses on engineering the web application's structure, logic and data.

WHY IS FRONTEND DEVELOPMENT IMPORTANT?

- Frontend development can be considered under two aspects.
  - **Designing** a good look and feel
    - HCI Design, UI/ UX etc.
  - **Engineering** the designed good look and feel
    - Frontend technologies
      - E.g., HTML, CSS, JS, Different frameworks, libraries etc.

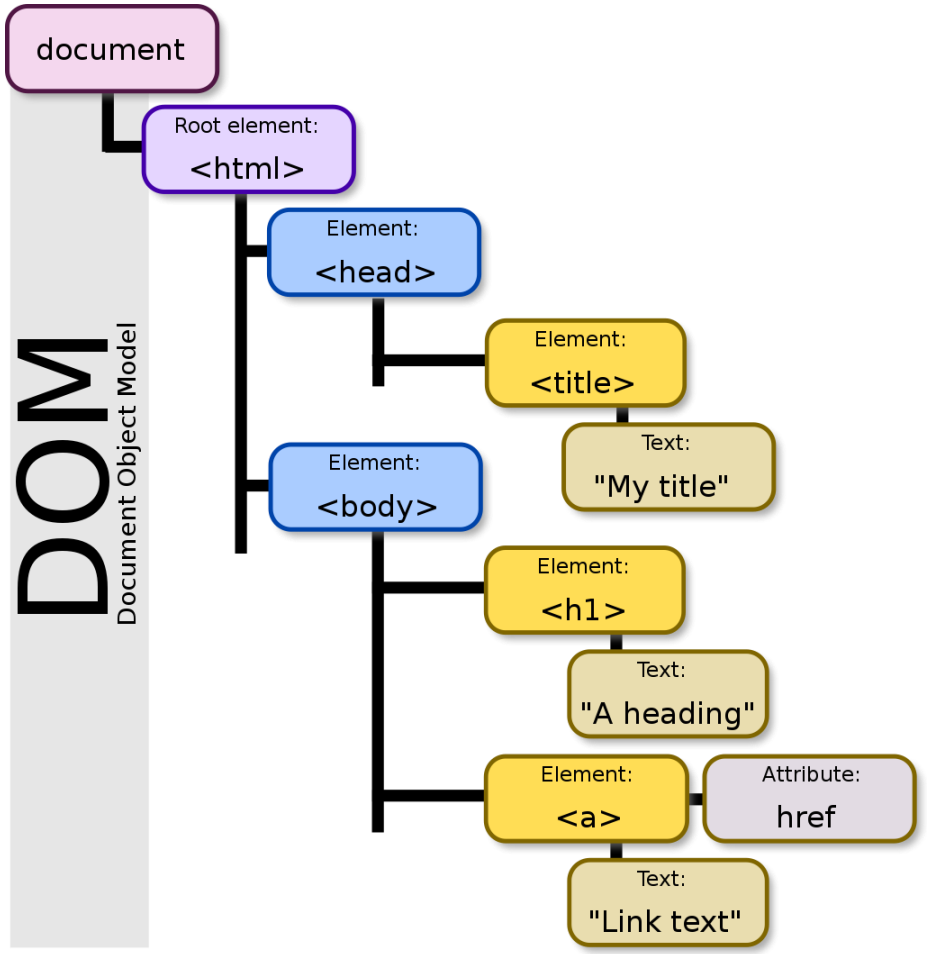
HTML, CSS, JS

- Frontend development is done with a combination of technologies in which,
  - **HTML** provides the structure
  - **CSS** provides the styling and layout
  - **JavaScript** provides the dynamic behavior and interactivity

DOCUMENT OBJECT MODEL (DOM)

- What do we actually manipulate with these technologies?
  - Document Object Model (DOM)!

DOCUMENT OBJECT MODEL (DOM)

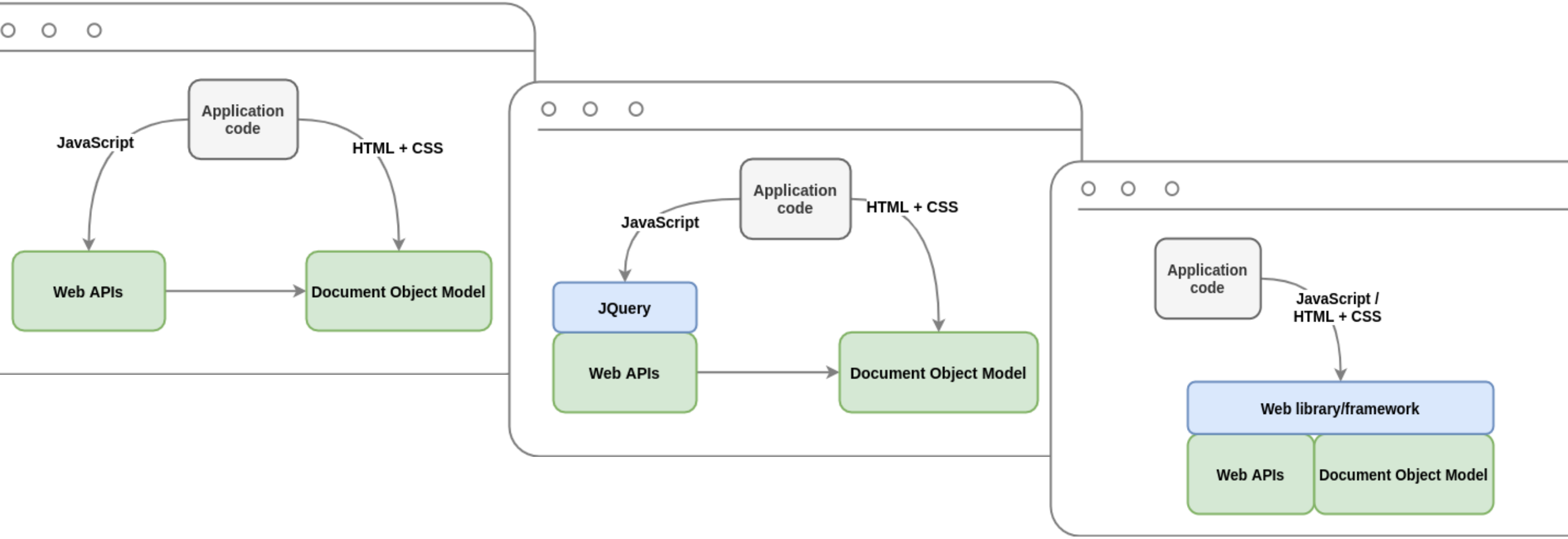


Source: https://en.wikipedia.org/wiki/Document_Object_Model

DOCUMENT OBJECT MODEL (DOM)

- Document Object Model (DOM) – An object-oriented representation of the HTML structure
- DOM is not a JS functionality.
  - JS is the mostly used language to manipulate it, but other languages can also be used.
- It is just a Web API.
- [YouTube] The DOM in 4 minutes

EVOLUTION OF FRONTEND DEVELOPMENT



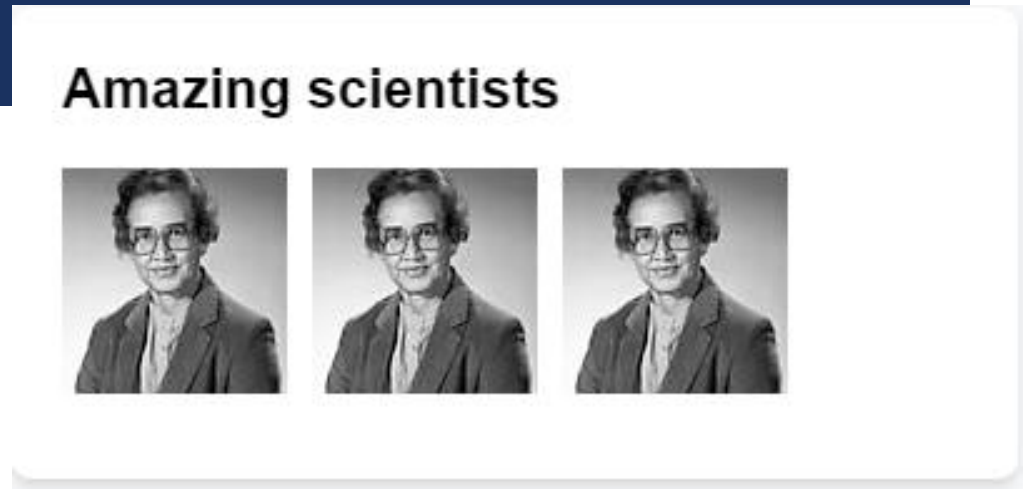
Source: https://www.epineda.net/the-evolution-of-front-end-development/amp/

REACT JS

- Developed and maintained by Facebook and Instagram.
- A JavaScript library for creating user interfaces.
- Serves as the view of MVC architecture.
- Suitable for large web application which use data and change over the time without reloading the entire page.
- React Native caters developing mobile application for various platforms and React VR caters developing VR applications.
- Aims at Speed, Simplicity and Scalability.

REACT : COMPONENTS

- Components are one of the core concepts of React.
- They are the foundation upon which you build 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.
- https://react.dev/learn/thinking-in-react



NOTABLE FEATURES

One-Way data flow.

- Single source of truth - Data is originated and kept in one place, data is immutable.
- Data flow from the parent component to the child component.
- Action flow from child component to parent component.

Virtual DOM

- DOM manipulation is cost instead react create a virtual DOM tree and compare it with the rendered tree and update only what is changed.

JSX

- React JS language for defining user interfaces.
- HTML/XML like syntax.
- Prevents cross-site scripting attacks by converting expressions to strings.

PROJECT STRUCTURE

```
/my-react-app
|-- /node_modules
|-- /public
|-- /src
    |-- /components  // React components, representing the 'View' in MVC
        |-- /shared  // Shared components like buttons, modals, etc.
        |-- /layout  // Layout components like headers, footers, etc.
        |-- /pages   // Page components, corresponding to different views/screens
    |-- /models      // Models in MVC, representing the data layer
        |-- user.js
        |-- product.js
        // other data models
    |-- /controllers // Controllers in MVC, orchestrating the app's data flow
        |-- UserController.js
        |-- ProductController.js
        // other controllers
    |-- /services    // Services for handling side effects, API calls, etc.
        |-- ApiService.js
        |-- UserService.js
        |-- ProductService.js
        // other services
    |-- /utils       // Utility functions and helpers
    |-- /hooks       // Custom React hooks, if any
    |-- /store       // If using Redux or a similar state management, to handle shared state
        |-- /actions
        |-- /reducers
    |-- App.js       // Main React component
    |-- index.js     // Entry point for React application
|-- package.json    // Node package configuration
|-- README.md
```

STATE IN WEB APPS

- **State** refers to the current condition or data of the application at a given point in time.
- State is important in web applications because it affects the behavior and appearance of the application.
- [YouTube] What is "State" in Programming?

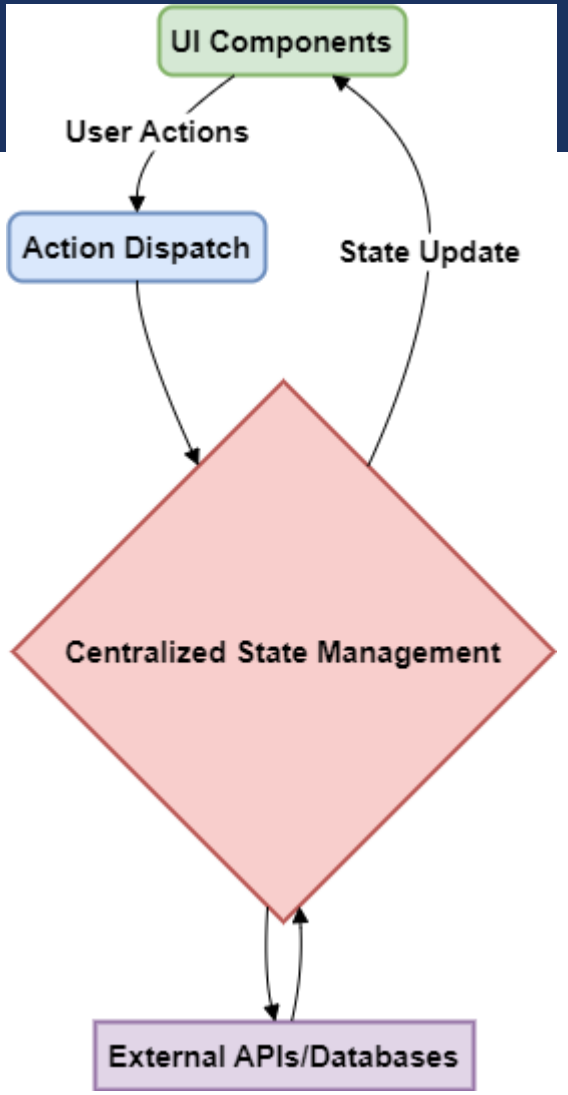
STATE MANAGEMENT IN WEB APPS

- **State management** is a concept in web development that refers to the management and manipulation of data or information within an application.
- In simple terms, it involves **controlling the data flow** and ensuring that the application's **data is consistent** and **up-to-date**.

STATE MANAGEMENT IN WEB APPS

- Proper management of state is important for the performance and reliability of web applications.
- Poorly managed state can result in slow application performance, data inconsistencies, or security vulnerabilities.

STATE MANAGEMENT IN WEB APPS



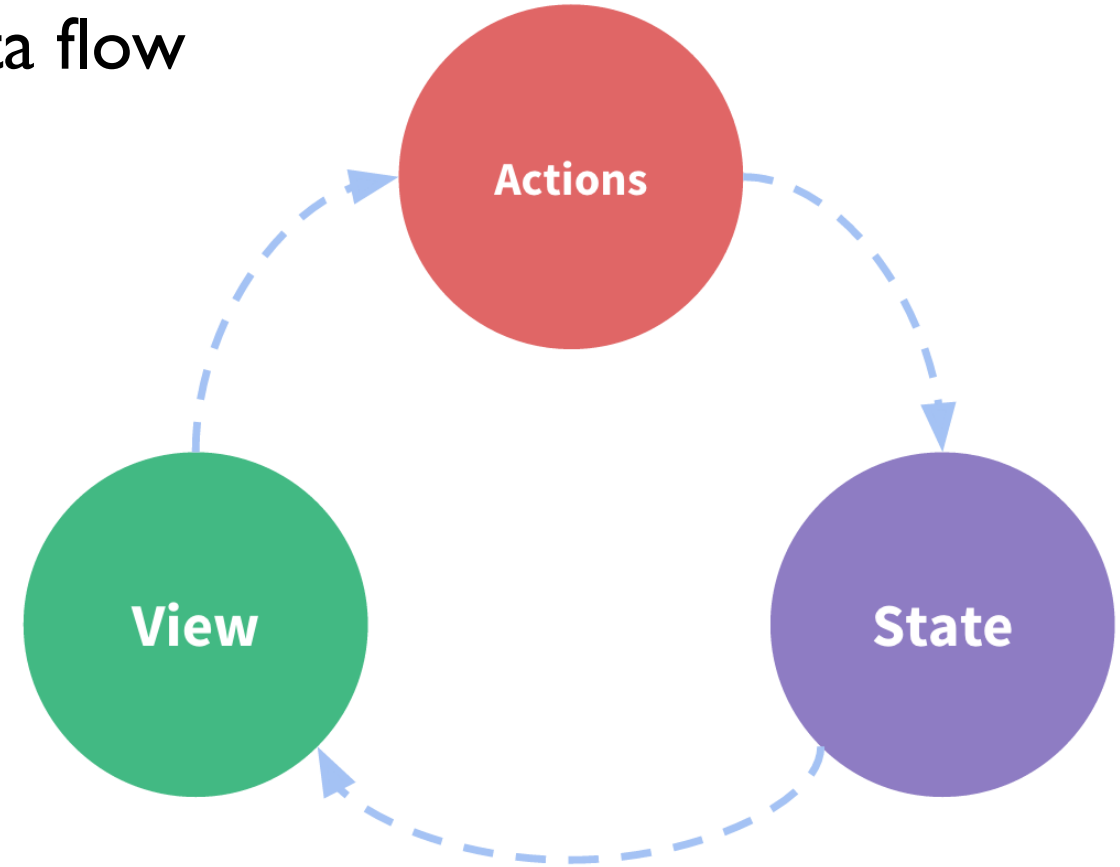
- How is State managed in web apps?
  - Local state management
  - Global state management
    - Redux

STATE MANAGEMENT IN WEB APPS

- Redux
  - Redux is a pattern and library for managing and updating application state.
  - Redux helps you manage "global" state - state that is needed across many parts of your application.
  - It serves as a centralized store for state that needs to be used across your entire application, with rules ensuring that the state can only be updated in a predictable fashion.
  - Redux is commonly used with React.
  - Read more on Redux here.

STATE MANAGEMENT IN WEB APPS

Redux one-way data flow



Source: https://redux.js.org/tutorials/essentials/part-1-overview-concepts

STATE MANAGEMENT IN WEB APPS

- Redux is not always needed.
  - **Do not use Redux until** you have problems with vanilla React.
- When to use Redux?
  - You have **large amounts of application state** that are needed in many places in the app.
  - The app state is updated frequently over time.
  - The logic to update that state may be complex.
  - The app has a medium or large-sized codebase and might be worked on by many people.

PROPS AND STATE

- Props: Props are used to pass data from a parent component to a child component.
- State: State is used to manage data that changes over time in a component.

EVENT HANDLERS

- In React, event handlers are functions that are called in response to user actions, such as clicks, keystrokes, or form submissions.
- React event handlers are typically defined as methods of a component class or as arrow functions within a functional component.
- In JSX, event handlers are specified as attributes of elements, using a naming convention where the event name is prefixed with on, followed by the name of the event in camelCase.
- Event handlers can receive an event object as a parameter, which contains information about the event, such as the target element, the mouse position, or the key that was pressed.

HOOKS

- React Hooks are a way to use state and other React features in functional components, without the need for class components.
- The most common Hooks are useState, useEffect, useContext, and useReducer. Each Hook has a specific purpose and usage.
- The useState Hook allows you to add state to a functional component. It returns an array with two values: the current state value and a function to update the state.
- The useEffect Hook allows you to perform side effects in a functional component, such as fetching data from an API or updating the DOM. It takes a function as its argument and runs it after every render.
- The useContext Hook allows you to access data from a React context in a functional component. It takes a context object as its argument and returns the current context value.
- The useReducer Hook is an alternative to useState that allows you to manage more complex state updates. It takes a reducer function and an initial state value as its arguments and returns the current state value and a dispatch function to update the state.
- Hooks should only be used at the top level of a functional component or another custom Hook. They should not be used inside loops or conditionals.

MANAGING STATE | REACTING TO INPUTS

- As your application grows, it helps to be more intentional about how your state is organized and how the data flows between your components.
- Redundant or duplicate state is a common source of bugs.
- React uses a declarative way to manipulate the UI.
- Instead of manipulating individual pieces of the UI directly, you describe the different states that your component can be in, and switch between them in response to the user input.
- This is similar to how designers think about the UI.

MANAGING STATE | STATE STRUCTURE

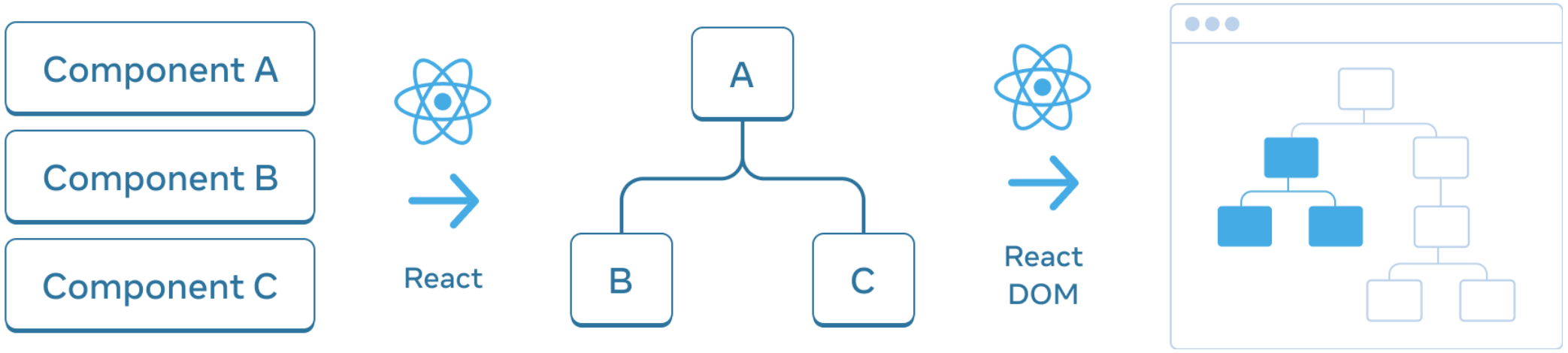
- Structuring state well can make a difference between a component that is pleasant to modify and debug, and one that is a constant source of bugs.
- **Group related state.** If you always update two or more state variables at the same time, consider merging them into a single state variable.
- **Avoid contradictions in state.** When the state is structured in a way that several pieces of state may contradict and "disagree" with each other, you leave room for mistakes. Try to avoid this.
- **Avoid redundant state.** If you can calculate some information from the component's props or its existing state variables during rendering, you should not put that information into that component's state.
- **Avoid duplication in state.** When the same data is duplicated between multiple state variables, or within nested objects, it is difficult to keep them in sync. Reduce duplication when you can.
- **Avoid deeply nested state.** Deeply hierarchical state is not very convenient to update. When possible, prefer to structure state in a flat way.

MANAGING STATE | STATE SHARING

- Sometimes, you want the state of two components to always change together.
- Remove state from both and move it to the closest parent component
- Then pass it down to them via props
- This is known as lifting state up, and it's one of the most common things you will do writing React code.

MANAGING STATE |PRESERVING & RESETTING STATE

- State is isolated between components.
- React keeps track of which state belongs to which component based on their place in the UI tree.
- You can control when to preserve state and when to reset it between re-renders.

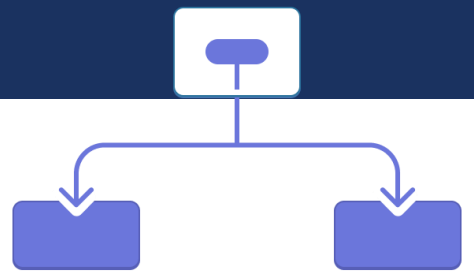


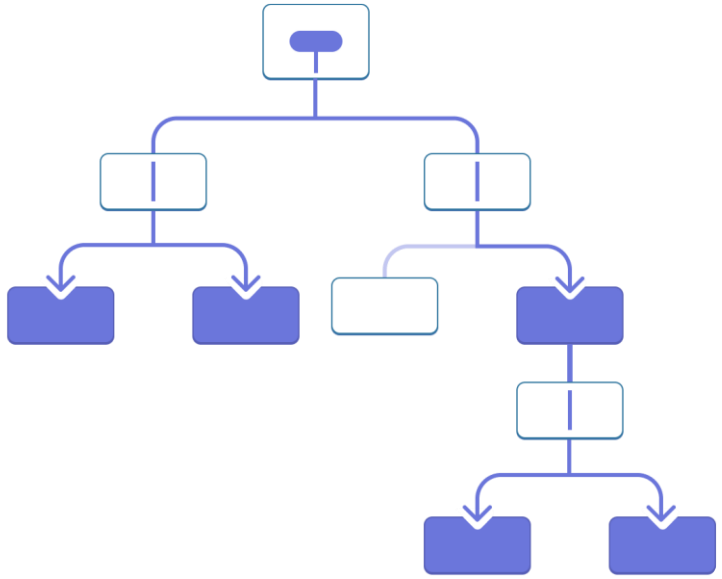
MANAGING STATE | USING REDUCER

- Components with many state updates spread across many event handlers can get overwhelming
- For these cases, you can consolidate all the state update logic outside your component in a single function, called a reducer.
- Although reducers can "reduce" the amount of code inside your component, they are named after the reduce() operation that you can perform on arrays.

MANAGING STATE | WITH CONTEXT

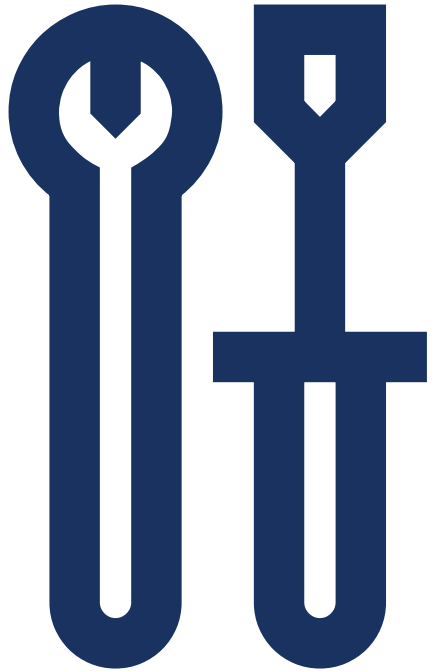
- Usually, you will pass information from a parent component to a child component via props.
- But passing props can become verbose and inconvenient if you must pass them through many components in the middle, or if many components in your app need the same information.
- Context lets the parent component make some information available to any component in the tree below it no matter how deep without passing it explicitly through props.





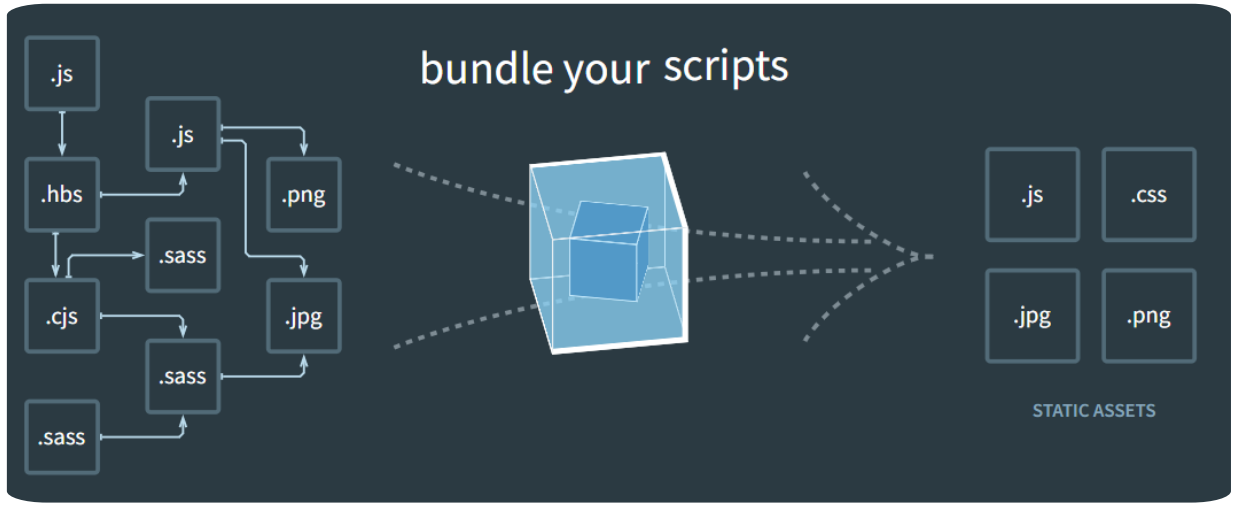
BUNDLING

- In web development, bundling refers to combining multiple files into a single file.
- Bundling is typically done with JavaScript or CSS files.
- Bundling reduces the number of HTTP requests needed to fetch all the required resources for a web page.
- This can improve the page's performance by reducing network latency.
- Bundling tools like Webpack or Parcel are used to create the bundle file.
- These tools analyze dependencies between files to create the bundle.
- Optimization techniques like minification and tree shaking may also be used as part of the bundling process.



WEBPACK

- Webpack is an open-source bundling tool and module bundler used in web development.
- It is used to bundle JavaScript files and other assets like CSS, images, and fonts for web applications.
- Webpack analyzes the dependencies between modules in the application and generates a single optimized bundle file.
- It includes a built-in development server that allows developers to preview their applications in a browser and see live changes.
- Webpack offers features like code splitting, lazy loading, tree shaking, and hot module replacement to optimize and speed up performance.
- It is highly configurable and can be customized according to project requirements.
- Webpack is widely used in modern web development frameworks like React, Vue, and Angular.
- https://webpack.js.org/



PARCEL

- Parcel is an open-source web application bundler and build tool, similar to Webpack.
- It is designed to be zero-config, meaning that it requires minimal setup and configuration to get started.
- Parcel can bundle a variety of web assets, including JavaScript, CSS, HTML, images, and more.
- It automatically handles file transformations, such as transpiling ES6 code to ES5 or compressing images.
- Parcel can use multiple cores to build and bundle projects in parallel, which can reduce build times.
- It includes a built-in development server that allows developers to see their changes in real-time as they make modifications to their code.
- Parcel has gained popularity among developers due to its ease of use and simplicity.
- It has been adopted by popular frameworks such as React, Vue, and Angular.
- https://parceljs.org/

SETTING UP REACT

- Official Documentation recommend to use a framework for react
- Frameworks provide features that most apps and sites eventually need, including routing, data fetching, and generating HTML.
- Popular Frameworks
  - Next.js
  - Remix
  - Gatsby
  - Expo (for native apps)

VITE

- Vite is a build tool and development server for modern web applications.
- It was created by Evan You, the creator of Vue.js.
- Vite is known for its fast startup time.
- It uses a development server that leverages modern browser features like native ES modules and HTTP/2 to quickly serve application code.
- Vite provides a build tool that includes features like code splitting and tree shaking to optimize the final application bundle for deployment.
- Vite is primarily used for developing front-end applications built with frameworks like React or Vue.js.

WEB APPLICATION SECURITY

- OWASP, or the Open Web Application Security Project, is a nonprofit organization focused on software security.
- OWASP Top 10 is a list of the 10 most common web application security risks.

WEB APPLICATION SECURITY

- OWASP Top 10 Vulnerabilities 2021
  1. Broken Access Control
  2. Cryptographic Failures
  3. Injection
  4. Insecure Design
  5. Security Misconfiguration
  6. Vulnerable and Outdated Components
  7. Identification and Authentication Failures
  8. Software and Data Integrity Failures
  9. Security Logging and Monitoring Failures
  10. Server-Side Request Forgery

WEB APPLICATION SECURITY

- Testing an application for the OWASP vulnerabilities is a **crucial step** in application development.
- [YouTube] What is The OWASP Top 10? | A Radware Minute
- [Mandatory] Read OWASP Top Security Risks & Vulnerabilities 2021.

WEB APPLICATION SECURITY

- HTTP Security Headers
  - HTTP security headers are a subset of HTTP headers that is related specifically to security.
  - Setting suitable headers in your web applications and web server settings is an easy way to greatly improve the resilience of your web application against many common attacks.
  - Read this link on HTTP Security Headers.
  - Some additional information can be found here too.

WRAP UP

- See this link for a step-by-step guide on becoming a frontend developer.
- Here's a nice overview on Frontend development: [YouTube] Frontend web development - a complete overview

SUMMARY

- What is frontend development?
- Why is frontend development important?
- HTML, CSS, JS
- Document Object Model (DOM)
- State in Web apps
  - State Management
- Web Application Security
  - OWASP Top 10 Security Risks & Vulnerabilities
  - HTTP Security Headers

REFERENCES

1. What is the Document Object Model?
2. Introduction to the DOM
3. The Best React State Management Tools for Enterprise Applications
4. Redux Essentials, Part 1: Redux Overview and Concepts
5. OWASP Top 10 Vulnerabilities

THANK YOU

VISHAN.J@SLIIT.LK

NELUM.A@SLIIT.LK