What is React

Topics

1 / What is React

2 / Where React Came From

3 / What is React Used For

4 / How React Works

5 / Why Use React

6 / What Can React Do

Part 1

What it React

React is a **free**, **open-source**, **front-end** **Javascript library** used for building **user interfaces** based on **components**.

‘free’

‘free’ - React is ‘free’

Understanding Software Copyright

- All software is protected under copyright
- Authors can specify a **license** for distribution
- **Common licenses** are MIT, GPL, Apache
- **React** uses the **MIT License**

free | open-source | front-end | Javascript library | UIs | components

MIT License

“free”

You can use the **software** and do **almost anything you want** with it, **but** you must include the **original copyright notice** and **disclaimer** when you use it in your own projects.

‘open source’

free | open-source | front-end | Javascript library | UIs | components

“open-source”

Source Code is ‘Open’

Open source means that the **source code** (the ‘code’) of what React is, is open and public to be viewed or downloaded by anyone, through Github.

‘front-end’

free | open-source | front-end | Javascript library | UIs | components

“front-end”

Front-End vs Back-End

Front-end, as opposed to **back-end** refers to code that executes on the ‘client’, which means on the user’s device.

Express.js is a **back-end Javascript library**.

‘front-end’ - React is a front-end tool

Front-end developers work on

- Displaying content
- Receiving user input
- Providing a friendly user interface

Back-end developers work on

- Data processing
- Data storage
- Business logic

‘Javascript library’

free | open-source | front-end | **Javascript library** | UIs | components

“JS library”

Javascript ‘library’

A **library** is simply a set of files that act as an **API** to abstract a lower-layer of code away in order to write simpler code by calling the methods of that **API/library**.

‘Javascript library’ - React is a Javascript library

Key Characteristics of Libraries

1. Abstraction
    - **Simple inputs** provide controls to **complex actions**
    - **Example**: A **car** is a complex machine with **simple inputs**
2. Reusability
    - **Standard** items **reduce duplication** and **simplify design**
    - **Example**: **Tools** in a toolbox are reused for different actions & projects

‘user interfaces (UIs)’

free | open-source | front-end | Javascript library | **UIs** | components

“UIs”

Website vs Web Apps

Websites are often more ‘static’ (unchanging) and **web apps** are more ‘dynamic’ (often changing). **UIs** refer to the visual design and structure aspects of both.

‘components’

free | open-source | front-end | Javascript library | UIs | components

“components”

Components?

Components are the **building blocks**, each one representing a different parts of a UI. They allow us to separate a complex UI into a lot of **simple** and **independent files**.

Part 2

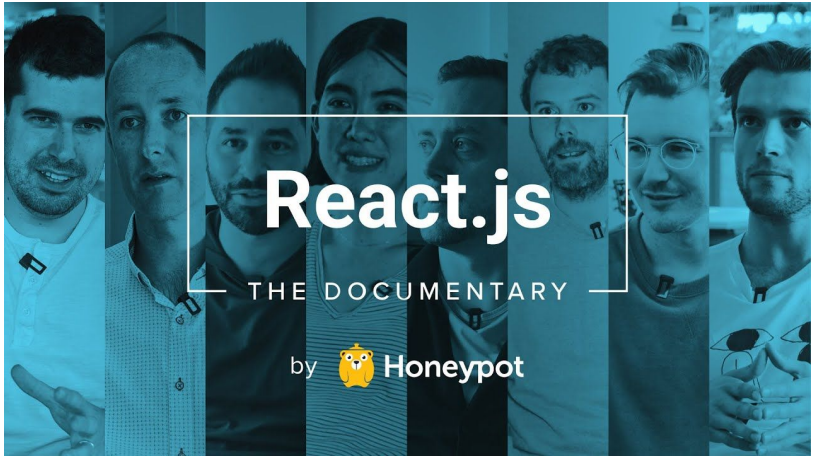
Where is React From

Founding Story

- **React** was created by a small team at **Facebook** around 2012
- **Facebook** was founded in 2004
  - Bought **Instagram** in April 2012
  - Went public with their **IPO** in May 2012
- **Facebook's** focus in 2012 was their **ads platform** and **mobile ecosystem**
- **Facebook's** main stack was PHP but was having scaling issues for complex apps

Founding Story

- **Angular** and **Backbone** emerge around 2011 as the first **Javascript frameworks**
- The internal team created **React** to be able to develop complex applications
- **React** eventually became the **dominant library** over time for several reasons
  - **Performance** (apps were fast)
  - **Simplicity** (easy to learn and use)
  - **Singular focus** (rather than all-in-one)
  - **Backwards compatibility**



Present Day

- **React** is maintained by **Meta** and a community of independent **developers** and **companies**
- **Next.js** marked a great paradigm shift in moving React towards server-side rendering in 2016
- The team behind **Next.js** is a company named **Vercel**, with members a part of the React Core team

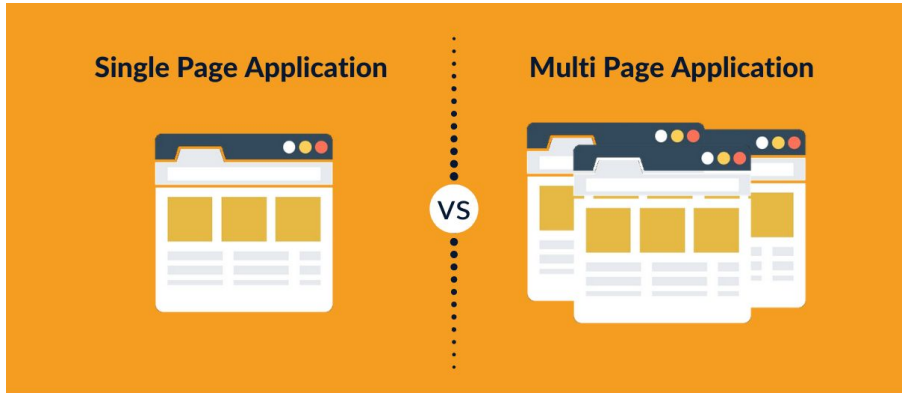
Vercel

Part 3

What is React Used For

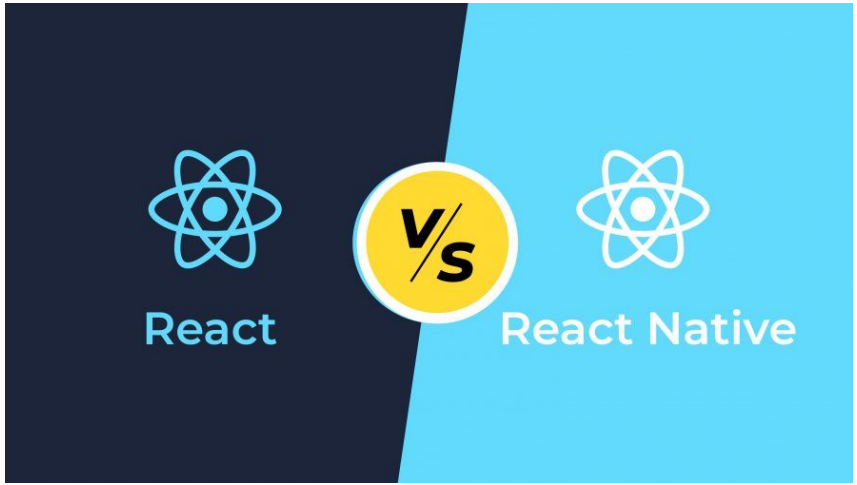
What is React Used For?

1. **Static** applications
2. **Single-page** applications (**SPAs**)
3. **Mobile** applications
4. **Server-rendered** applications



React’s Core Technologies

1. **React Core** is for **Web Apps**
2. **React Native** is for **Mobile Apps**
3. **Next.js** is for **Server-Side Apps**



Part 4

How React Works

Key Concepts in React

Components

JSX

Props

State

Events

Components

Components

- We learned that **components** in computer science provide **abstracted** and **reusable code**
- In React, **components** are just **Javascript functions**

App.jsx U

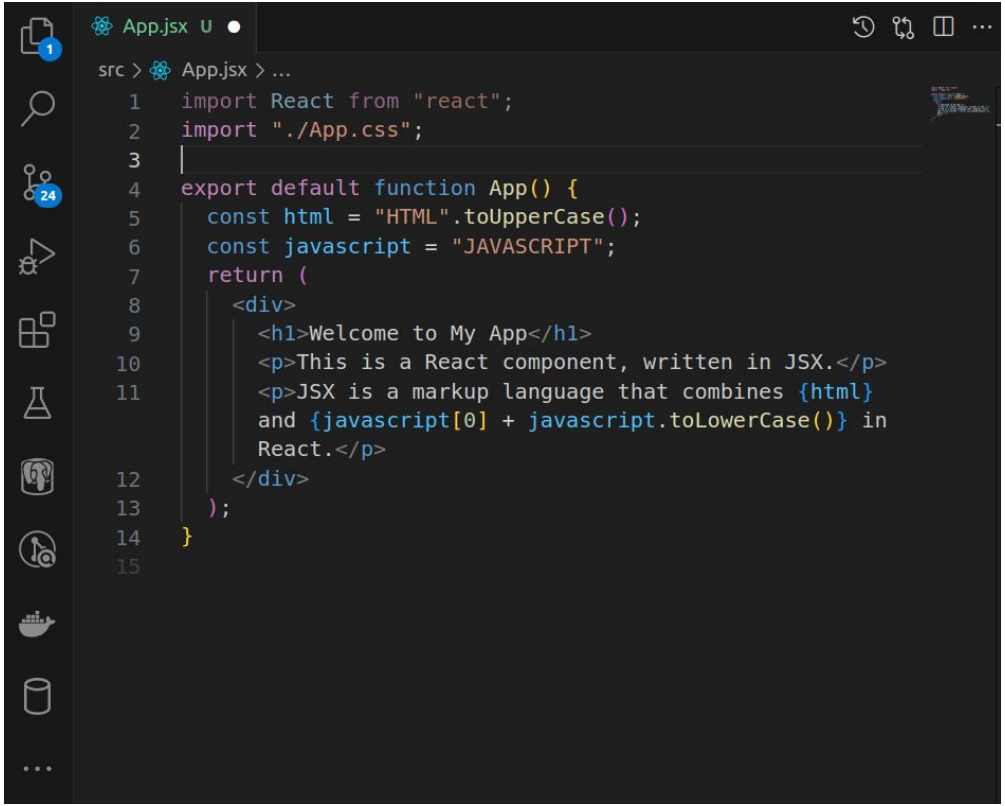
src > App.jsx > ...

```jsx
import React from "react";
import "./App.css";

function RandomNumber() {
  const randomNumber = Math.floor(Math.random() * 10) + 1;
  return (
    <div>
      <h2>Item</h2>
      <p>This is an random number: {randomNumber}.</p>
    </div>
  );
}

export default function App() {
  return (
    <div>
      <h1>Welcome to My App</h1>
      <p>This is a React component, written in JSX.</p>
      <RandomNumber />
      <RandomNumber />
    </div>
  );
}
```

JSX

JSX

- For these Javascript functions to be valid **components**, they must return an HTML-like syntax known as **JSX**
- JSX stands for **Javascript XML**, which is a **markup language** for React
- **HTML** and **XML** are other **markup languages**



Props

Props

- Props is shorthand for **properties**
- **Props** is a method to pass data from a **parent component** to a **child component**
- **Props** is a **Javascript object**, available to us through being the first argument in a component function
- **Props** allow us to create **reusable components** that differ only in the variables passed through the **props**

App.jsx U

src > App.jsx > ...

```react
import React from "react";
import "./App.css";

function Color(props) {
  return (
    <div>
      <h2>Item</h2>
      <p>This component's color is: {props.color}</p>
    </div>
  );
}

export default function App() {
  return (
    <div>
      <h1>Welcome to My App</h1>
      <p>This is a React component, written in JSX.</p>
      <Color color="Blue" />
      <Color color="Red" />
    </div>
  );
}
```

State

State

- **State** allows our app to be dynamic
- **Change** in our app requires three things
  - **Events** that the browser recognizes
  - **Inputs** that trigger those events
  - **Memory** that remembers the ‘state’
- **React** provides **memory** at the component level with the use of the **useState** function
- **useState** returns an array of two items, more commonly used with **array destructuring**

App.jsx 1, U

src > App.jsx > ...

```jsx
import React from "react";
import "./App.css";

export default function Alarm() {
  const [alarmState, setAlarmState] = React.useState("off");
  return (
    <div>
      <h1>Welcome to My App</h1>
      <p>The alarm is: {alarmState}</p>
    </div>
  );
}
```

Events

Events

- **Events** are a Javascript + browser concept
- The browser listens for **events**, like
  - Typing a letter of a keyboard
  - Clicking a button
- **Event handlers** are **Javascript functions**
- These **functions** are called **event handlers** because they **handle events**
- The function reference passed to **onClick** is an **event handler** and **'click'** is the type of event

App.jsx U

src > App.jsx > ...

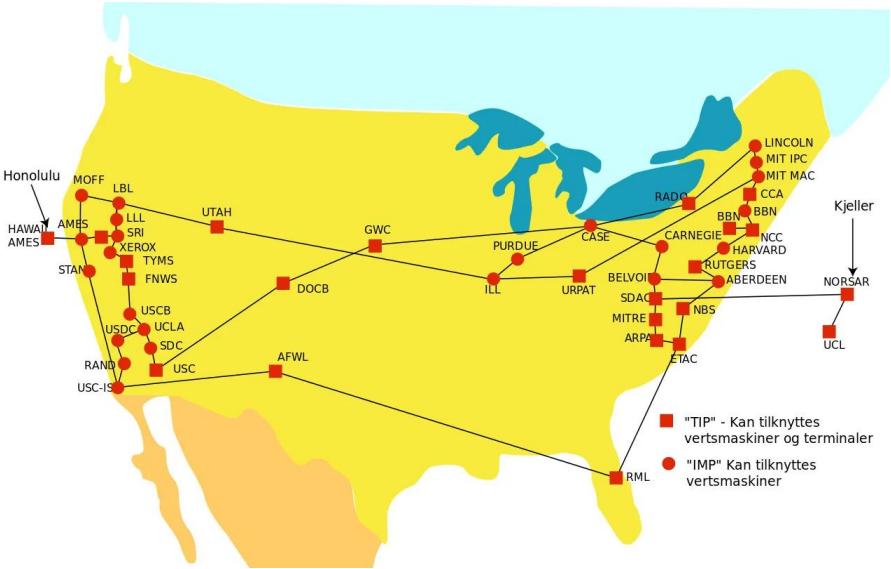
```jsx
import React from "react";
import "./App.css";

export default function Alarm() {
  const [alarmState, setAlarmState] = React.useState("off");
  function handleClick() {
    if (alarmState === "off") {
      setAlarmState("on");
    } else {
      setAlarmState("off");
    }
  }
  return (
    <div>
      <h1>Welcome to My App</h1>
      <p>The alarm is: {alarmState}</p>
      <button onClick={handleClick}>
        Turn Alarm {alarmState === "off" ? "On" : "Off"}
      </button>
    </div>
  );
}
```

Part 5

Why Use React

The Web Before React

- The **Internet** was a government funded program intended to share information across universities
- The **Internet** was originally designed to transmit **static documents**
- Over time, **HTML, CSS, Javascript** formed the base layer of creating static and dynamic pages for the web



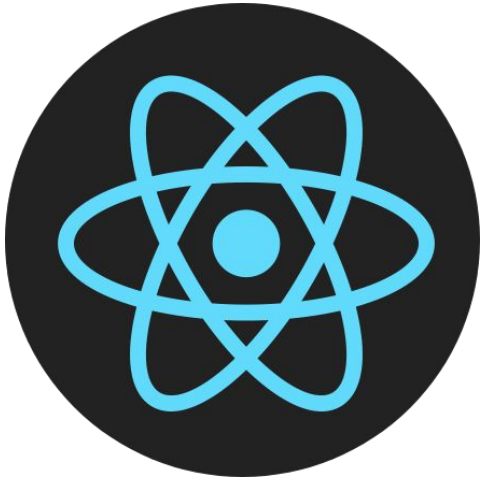
The Web Today

Benefits of Frameworks

- Scalability
- Modularity
- Developer productivity
- Community support
- Performance
- Security

Why React

1. Easy to Learn
2. High Adoption
3. Large Ecosystem



Part 6

What Can React Do

Analytics	Authentication	Roles	State Management
Logging	Local Storage	Internationalization	Testing
Forms	APIs	Routing	Styling
Performance	WebSockets	Responsive Design	Deployment

What’s **Next?**

Sections

Section 1

The React Universe

1. What is React (✓)
2. History of React
3. React Team
4. React Vision
5. React Community
6. React Ecosystem
7. React Installation
8. Common Questions
9. Recommended Resources

Section 2

Learn React

1. Components
2. JSX
3. Props
4. State
5. Conditional Rendering
6. Lists
7. Forms
8. App State
9. Refs
10. Effects
11. Hooks

Section 3

Common React Modules

1. Analytics
2. Authentication & Authorization
3. Role-Based Access Control
4. State Management
5. …
14. Real-time Updates & WebSockets
15. Responsive Web Design
16. Deployment and Hosting


Subscribe

Thanks for watching