

# Front End Software Development

Web App Design w/React (weeks 13 - 18)

Week 01

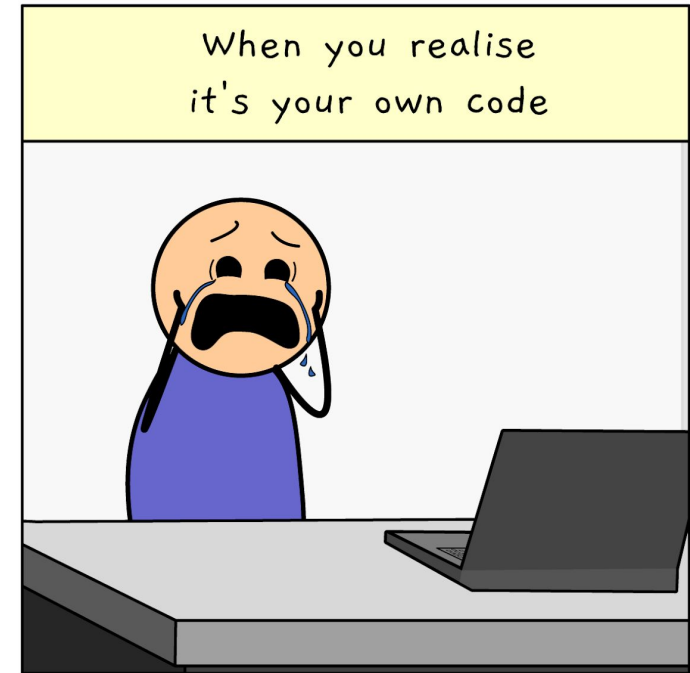
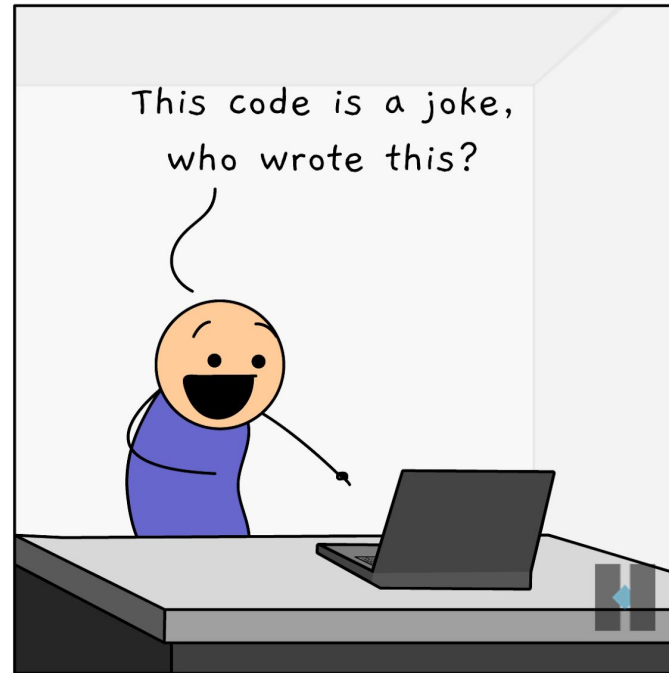


# Agenda

- Questions
- What is React?
- Installing NPM
- Webpack
- Getting Started With React
  - create-react-app
- React Components
- JSX



# Questions



f /techindustan

t /techindustan

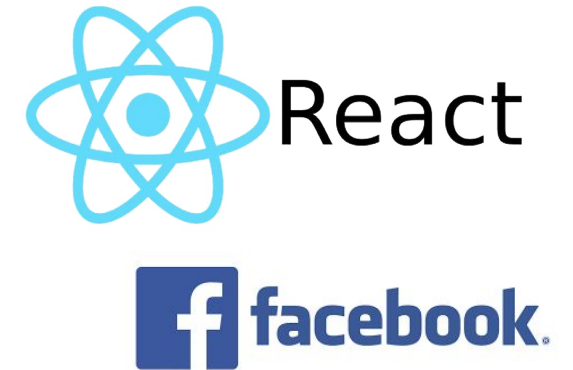
ig /techindustan

# What is?

*(React)*

- User Interface (UI)  
broken into components
- Created & used by Facebook
- Components
  - Custom HTML Elements

```
<MyWelcome>
  <h1>Welcome</h1>
</MyWelcome>
```
  - Encapsulate logic & code
  - Reuse
- Virtual DOM
  - Detects different, only renders changes.



# Installing NPM

(Node Package  
Manager)



- Node.js
  - Server-side JavaScript
  - <https://nodejs.org/en/download/>
- Node Package Manager
  - Manage packages (install, update, manage)
  - To check if it is installed, from command line:

```
npm -v
```

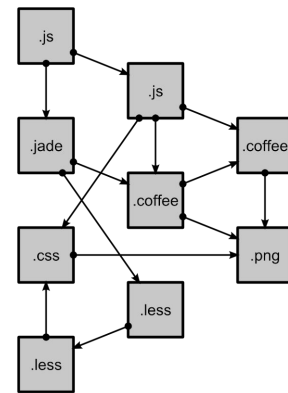
*Note: By default, npm downloads and installed packages locally in the current folder.*



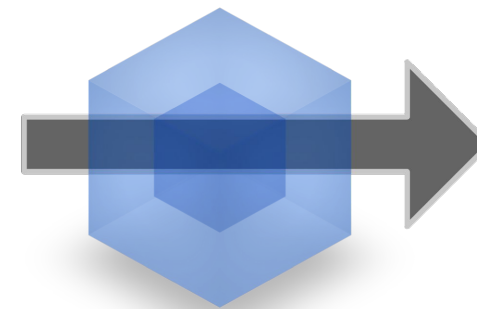
# Webpack



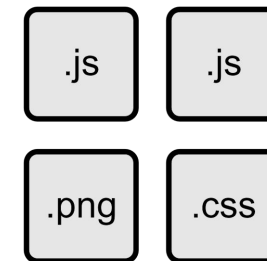
- Multiple dependencies
  - CSS, JavaScript, HTML
  - Separate file per class
    - Manually linking files with `<script />` tags not maintainable.
- Bundles into single files / resources (JS, CSS)
  - Minification (*Removes spaces, line feeds, shortens function names, etc.*)



modules  
with dependencies



webpack  
MODULE BUNDLER



static  
assets

# Webpack

(Usage)

- Installation

```
npm init -y
npm install --save-dev webpack-cli
npm install --save-dev webpack
```



```
npx webpack
```

*Creates a  
dist/main.js file.*

- Classes

- `export default class ClassName`
- `import $ from 'jquery';`
- `import Person from './person';`

***Note:** The `init` defaults to expecting a `src/index.js` file (defined in `package.json`). Create `src/index.js` as a blank file. Don't forget to create the folder!*

## Alternatives

Browserify, Gulp, Babel, Parcel, Grunt, RequireJS, Bower, etc. (and too many more...)

# Getting Started

*(Default Template)*

- Create Project

```
mkdir src dist
npm init -y
npm install --save-dev webpack webpack-cli react react-dom
```

- Create dist/index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <div id="app"><div>
    <script src="main.js"></script>
  </div>
</body>
</html>
```

- Create src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
```

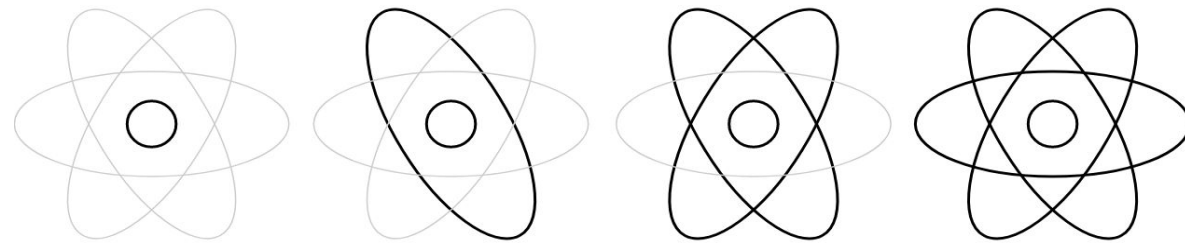
- Build / Pack

```
npx webpack
```



# Getting Started

*(Under the hood)*



- Create HTML element

```
let element =  
  React.createElement('h1', {},  
    'This is React!');
```

HTML Tag

Properties  
(props)

Content / Children  
(Text, Nested  
Elements, etc.)

- Render

```
ReactDOM.render(element,  
  document.getElementById('app'));
```

# **create-react-app**

(Quick start in 2 steps)

1. `npx create-react-app  
appName`
2. `npm run`

- Folder & File Structure

- **src\**  
*React related code. We'll mostly work here.*
- **src\App.css**  
*Our page / global CSS and styles.*
- **src\App.js**  
*Our main application page and content.*
- **public\**  
*Our static assets (fonts, images, etc.)*



Edit `src/App.js` and save to reload.

[Learn React](#)

# React Components (Elements)

- Break up user interface into reusable components

- components \

```
import React from 'react';
```

```
export class AddButton extends React.Component {
```

```
  render() {
```

```
    return React.createElement('button',  
      { className: 'btn btn-primary',  
        'Add' });
```

```
  }  
}
```

```
...
```

```
export class SubtractButton extends React.Component {
```

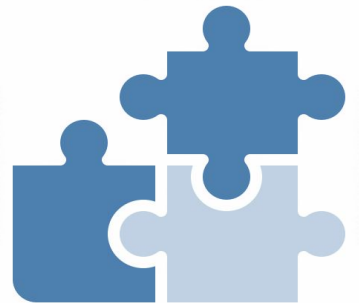
```
  render() {
```

```
    return React.createElement('button',  
      { className: 'btn btn-danger',  
        'Subtract' });
```

```
  }  
}
```

Custom HTML  
Element / Tag

Draws to virtual  
DOM



# React Components

(Composition)

- Composition

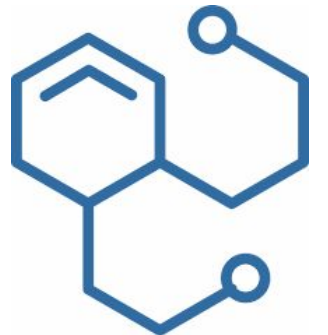
```
import React from 'react';
import AddButton from './add-button';
import SubtractButton from './subtract-button';

let e = React.createElement;

render() {
  return e('div', { className: 'container' },
    e('h1', {},
      'Count: ',
      e('span', { className: 'badge bg-secondary' }, 0)
    ),
    e('div', {},
      e(AddButton, {}),
      e(SubtractButton, {})
    )
  );
}
```

Custom HTML  
Element / Tag

Custom HTML  
Element / Tag

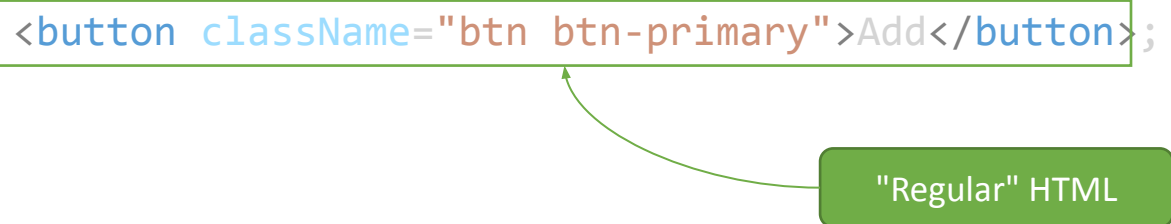


# JSX

*(Elements)*

- JavaScript XML
  - Reduces need for `React.createElement()`
  - Write "regular" HTML

```
export default class AddButton extends React.Component {  
  render() {  
    return <button className="btn btn-primary">Add</button>;  
  }  
}
```



- Note: Need to use `className` vs `class` due to `class` being a reserved word in JavaScript.

# JSX (Composition)

- Use Components

```
export default class Counter extends React.Component {  
  render() {  
    return (

<h1>Count: <span className='badge bg-secondary'>0</span></h1>  
      <div>  
        <AddButton /> <SubtractButton />  
      </div>  
    </div>);  
  }  
}


```

Which code do you  
want to write?

Vs.

```
render() {  
  return e('div', { className: 'container' },  
    e('h1', {},  
      'Count: ',  
      e('span', { className: 'badge bg-secondary' }, 0)  
    ),  
    e('div', {},  
      e(AddButton, {}),  
      e(SubtractButton, {})  
    )  
  );  
}
```



# DEMO

React From Scratch