

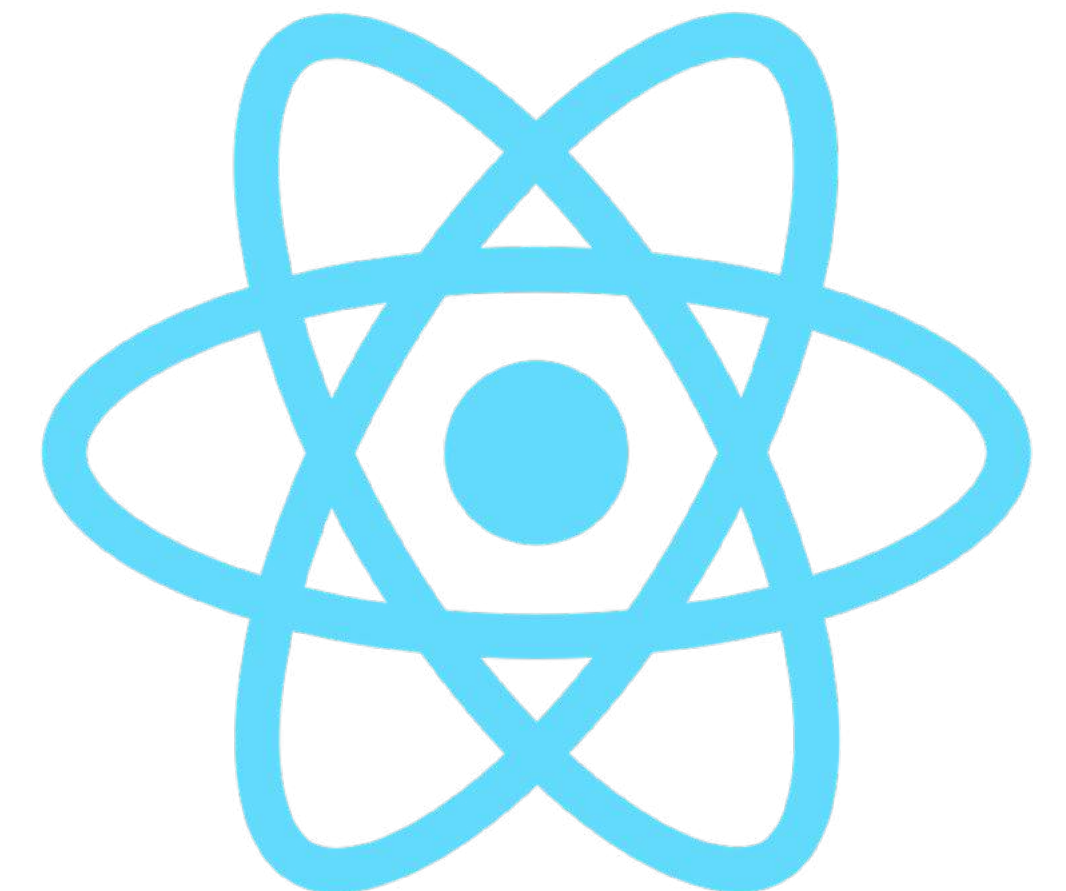
ReactJS I (Introduction to React)

Objective

- MVC
- Introduces ReactJS (a JavaScript library for building user interfaces)
- Setting up your environment
- Creating React Elements using JSX.









What is React?

- React is a JavaScript library used to generate user interfaces (UIs) that is used for building user interfaces specifically for single-page applications. It's used for handling the view layer for web and mobile apps. React also allows us to create reusable UI components.
- React allows developers to create large web applications that can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application. This corresponds to the view in the MVC template.
- JavaScript can be used to make web pages dynamic. This is where React comes in. With React, we use JavaScript to write HTML.
- React is declarative. This means we tell React what we want to do, not how to do it.

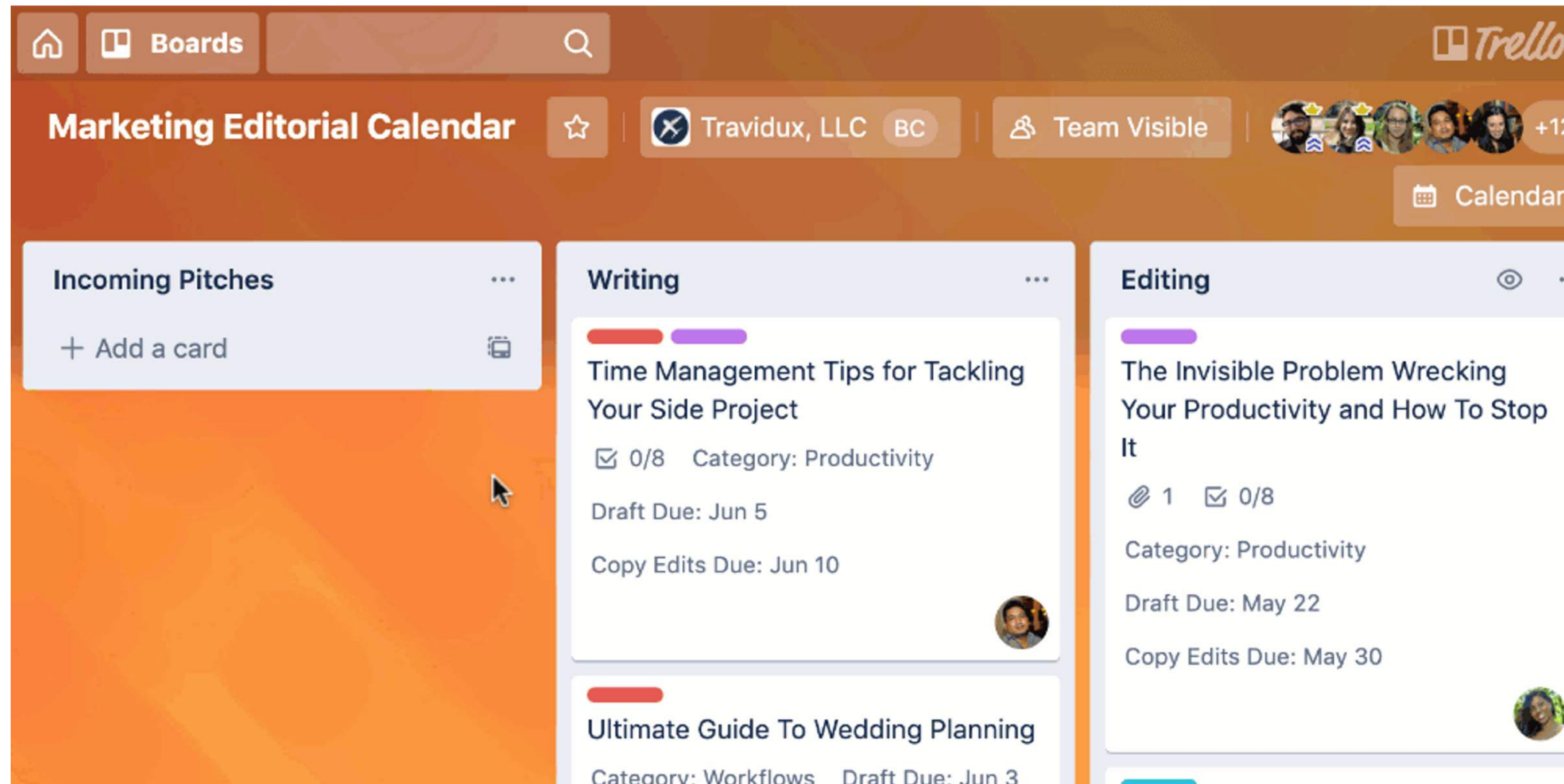


What Is a Single Page Application? (SPA)

- Applications that are contained within one web page are called single-page applications (SPA).
- The browser downloads the entire app data when you visit SPA web applications. Thus, you can browse through different parts of the app seamlessly and the page won't reload every time you click on something.
- This is because a single page application executes the logic in the browser, instead of a server. It does so with JavaScript frameworks that can lift this heavy data on the client-side. JavaScript also enables an SPA to reload only those parts of the app that a user requests for, not the entire app. As a result, SPAs are known to deliver fast and efficient performance.

Single page apps	Multi page apps
    	  

What Is a Single Page Application? (SPA)



Advantages of Single-Page Applications

What benefits can SPA provide to your business?



High speed



Hassle-free user
experience



Mobile-
friendliness



Makes your
website unique

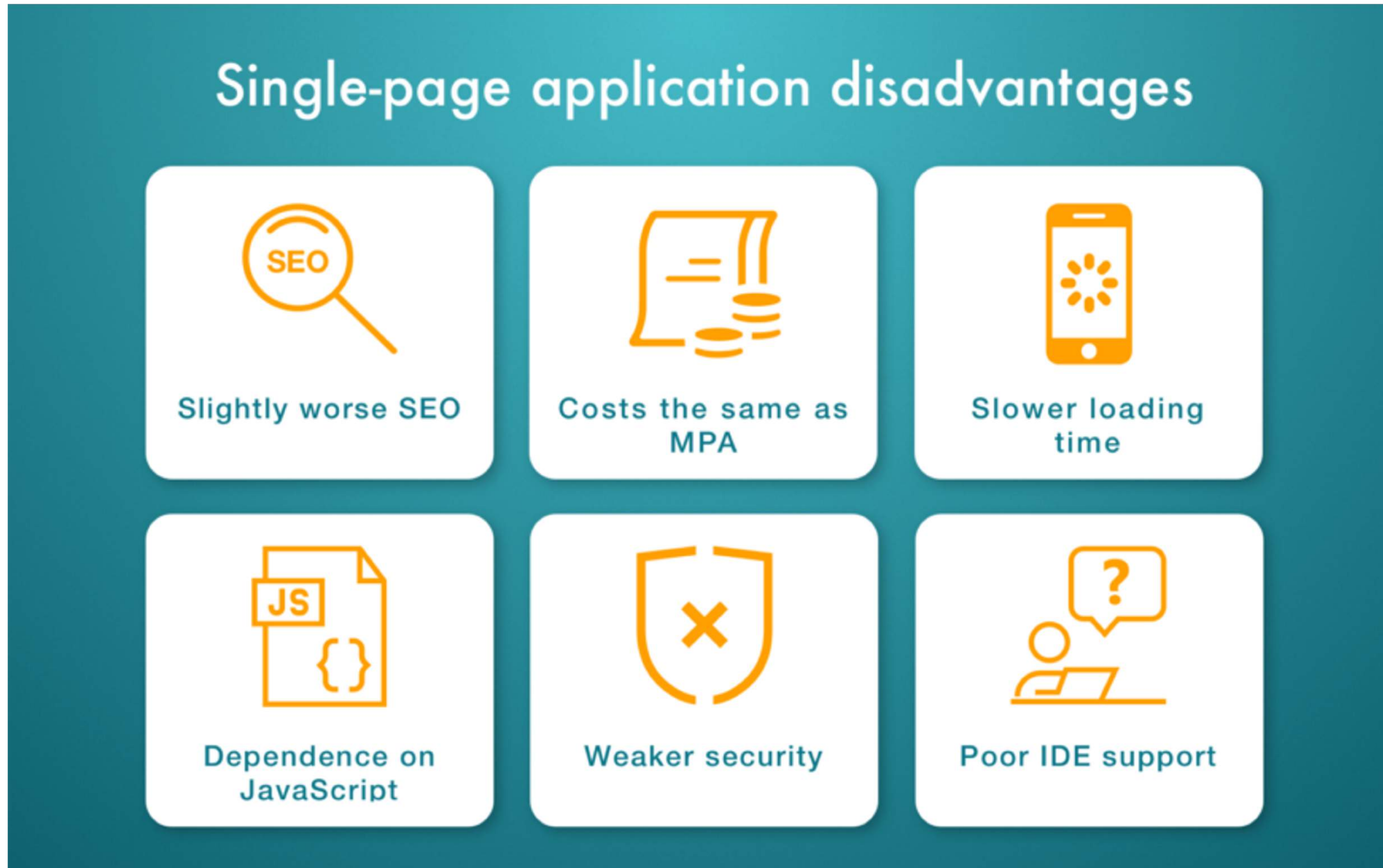


Better
conversions

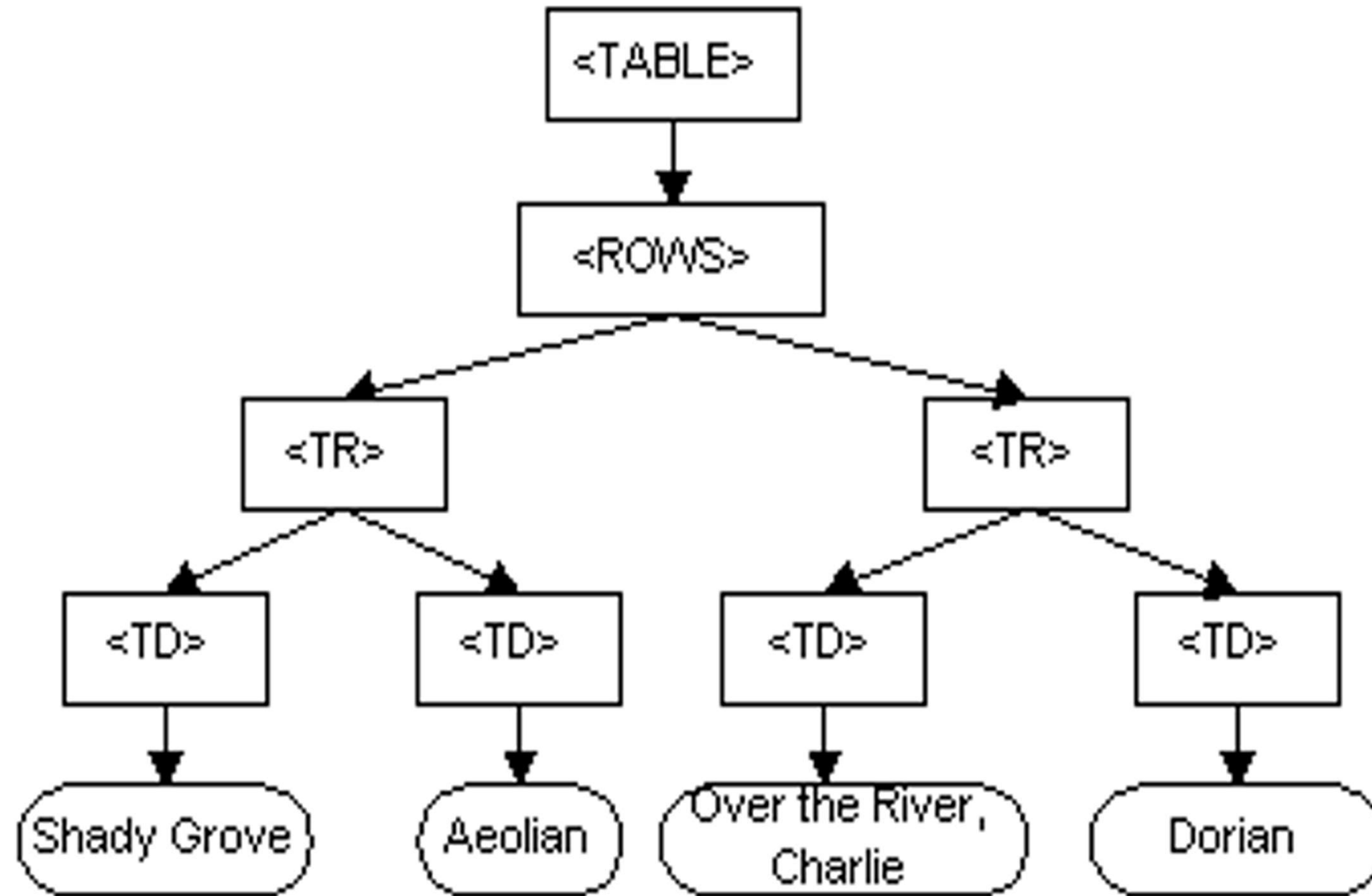


Brand
storytelling

Disadvantages of Single-Page Applications

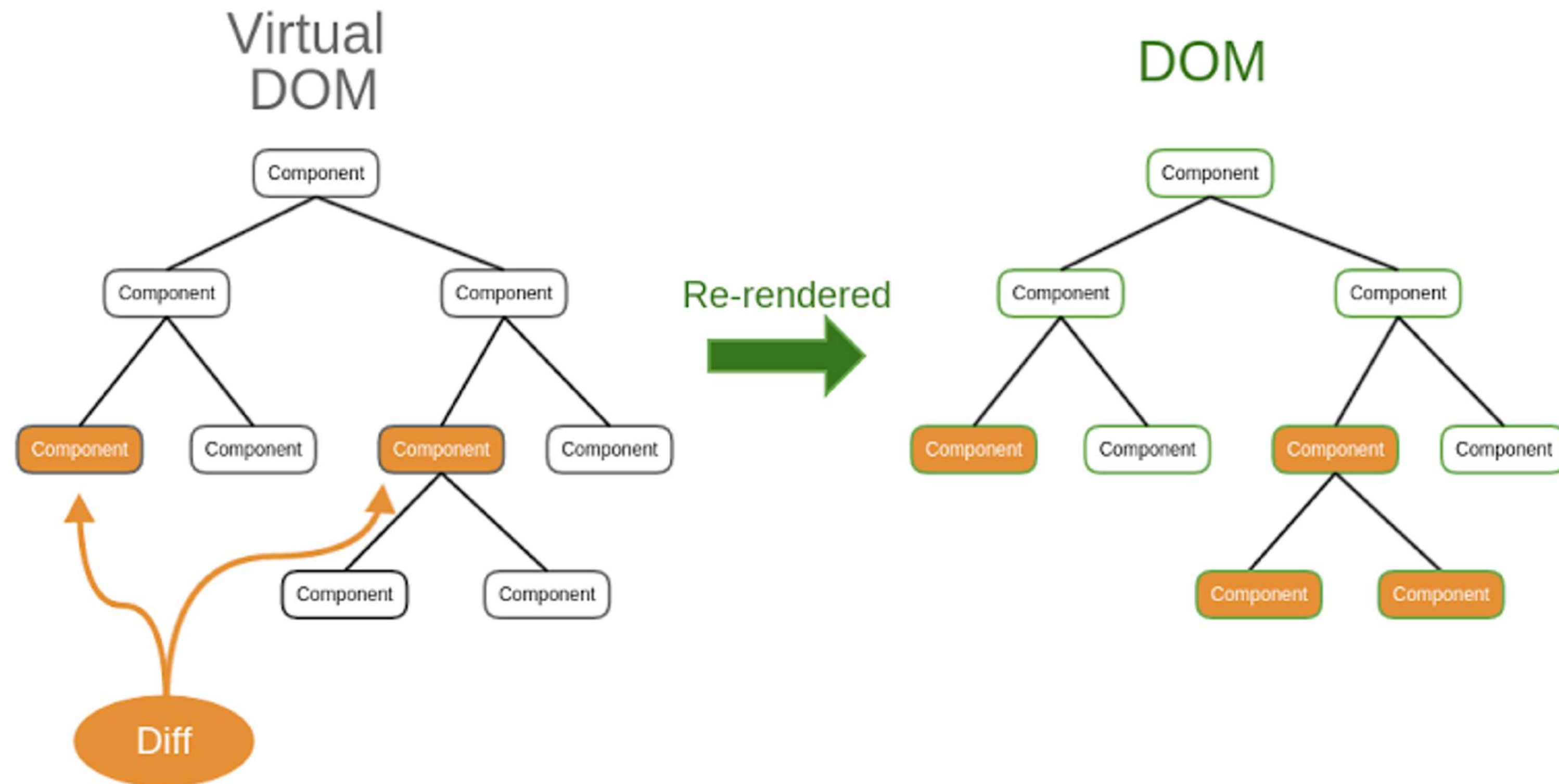


DOM for an HTML Table



Virtual DOM

- Virtual DOM: representation of the HTML document in memory.
- React works by taking a 'snapshot' of the DOM changes are made and, as changes are made virtual DOM is updated by making those changes.



JSX

- JSX is a syntax extension for JavaScript that can make creating React applications a lot quicker and easier.
- JSX stands for JavaScript XML.
- JSX allows us to write HTML in React.
- JSX makes it easier to write and add HTML in React.
- JSX allows us to write HTML elements in JavaScript and place them in the DOM without any createElement() and/or appendChild() methods.
- JSX converts HTML tags into react elements.

```
function formatName(user) {  
  return user.firstName + ' ' + user.lastName;  
}  
  
const user = {  
  firstName: 'Dave',  
  lastName: 'Hyperion',  
  profileImg: './images/12345.gif'  
};  
  
const element = (  
  <h1>  
    Hello, {formatName(user)}!  
  </h1>  
)  
);  
  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
)
```

Example of JSX

JSX Rules

- You can specify various elements that can have different attributes.
- You can use normal HTML elements and create user-specified elements.
- If an attribute value is enclosed in quotes, it is a string. If it is in braces, it is an enclosed JavaScript expression.

Here are two examples. The first uses JSX and the second does not:

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

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

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

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

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

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

Execute the expression **5 + 5**:

```
const myelement = <h1>React is {5 + 5} times better with JSX</h1>;
```


JSX Rules

- Attributes can be assigned using either:
- String literals e.g. ``
- String literals should be enclosed by quotation marks, e.g. `"./images/logo.gif"`
- Or JavaScript expressions, e.g. ``
- JavaScript expressions should always be enclosed by curly braces instead of quotation marks, e.g. `{user.profileImg}`

React Elements

- Elements are the smallest building blocks of React apps. An element specifies what should be there in our UI. An Element is a plain object describing what we want to appear in terms of the DOM nodes.

Official definition:

React elements are the building blocks of React applications. An element describes what you want to see on the screen. React elements are immutable (they cannot be modified after they are created).

```
const element = <h1>Hello, world</h1>;
```

Typically, elements are not used directly, but get returned from components.

Creating React Elements

3 Steps to create any React elements:

1. Create React and ReactDOM objects by importing React and ReactDOM.

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


Creating React Elements

2. Create the React Element using either JavaScript or JSX.

a. You could declare a variable that stores HTML or JSX

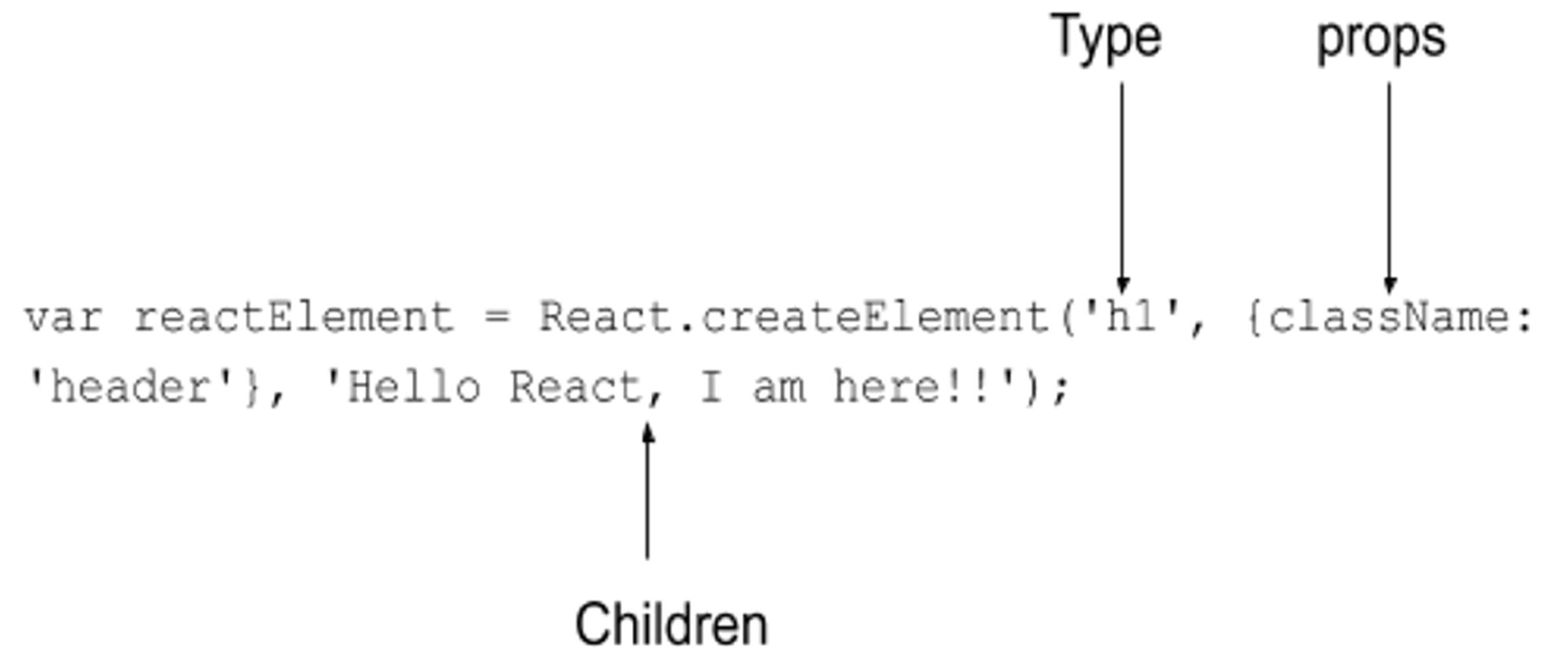
b. Your variable could consist of more than one element if you wrap them in a parent element and enclose the elements with `()`

```
const element = (  
  
  <div>  
    <h1>I'm learning React with Hyperion</h1>  
      
  </div>  
  
);  
ReactDOM.render(element, document.getElementById('root'));
```

Creating React Elements

c. You could use the createElement() method to create React elements.

```
const reactElement = React.createElement('h1', {}, 'Hello World!');  
ReactDOM.render(reactElement, document.getElementById('root'));
```



The diagram illustrates the parameters of the `createElement()` method. It shows the code snippet: `var reactElement = React.createElement('h1', {className: 'header'}, 'Hello React, I am here!!');`. Three arrows point from labels to specific parts of the code: 'Type' points to the string `'h1'`, 'props' points to the object `{className: 'header'}`, and 'Children' points to the string `'Hello React, I am here!!'`.

```
var reactElement = React.createElement('h1', {className:  
'header'}, 'Hello React, I am here!!');
```

Creating React Elements

3. Render the element that you have created to the DOM using the `render()` method. Once you have created an element, you still have to render it so that it is visible to the user.

أكاديمية طويق
TUWAIQ ACADEMY



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


Styling React Elements

As with creating HTML pages, you can create your own custom stylesheets.

To apply the rules in the stylesheet to the elements, simply import the .css file to the JavaScript file where you are creating the element. E.g. `import './css/custom.css';`

There are several CSS libraries that can make creating components quicker and easier by reusing and modifying existing code, e.g. React-Bootstrap and Reactstrap

Reactstrap: fewer features. A good choice for projects that need smaller builds.

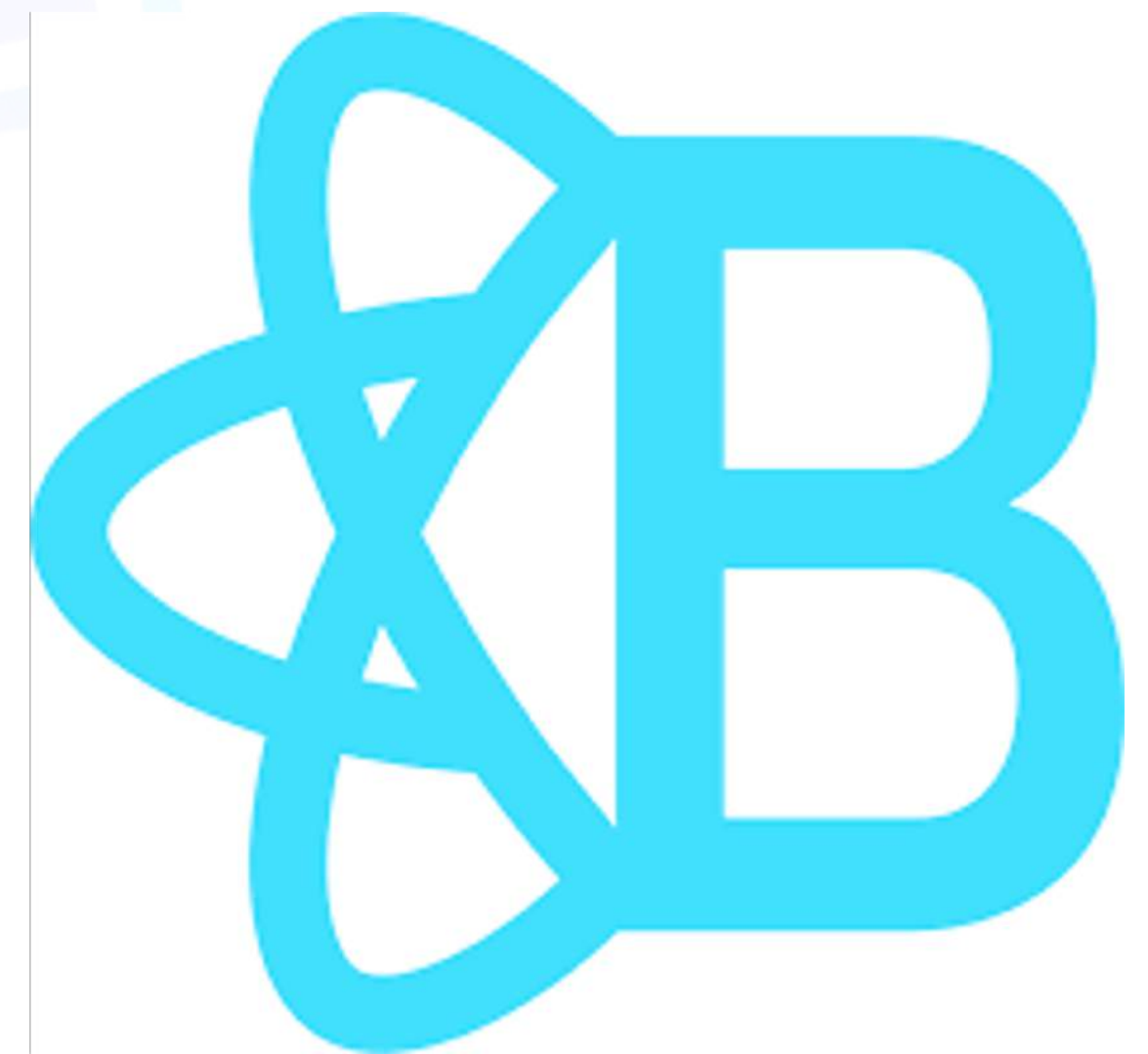
React-Bootstrap

React-Bootstrap is a complete re-implementation of the Bootstrap components using React. To use:

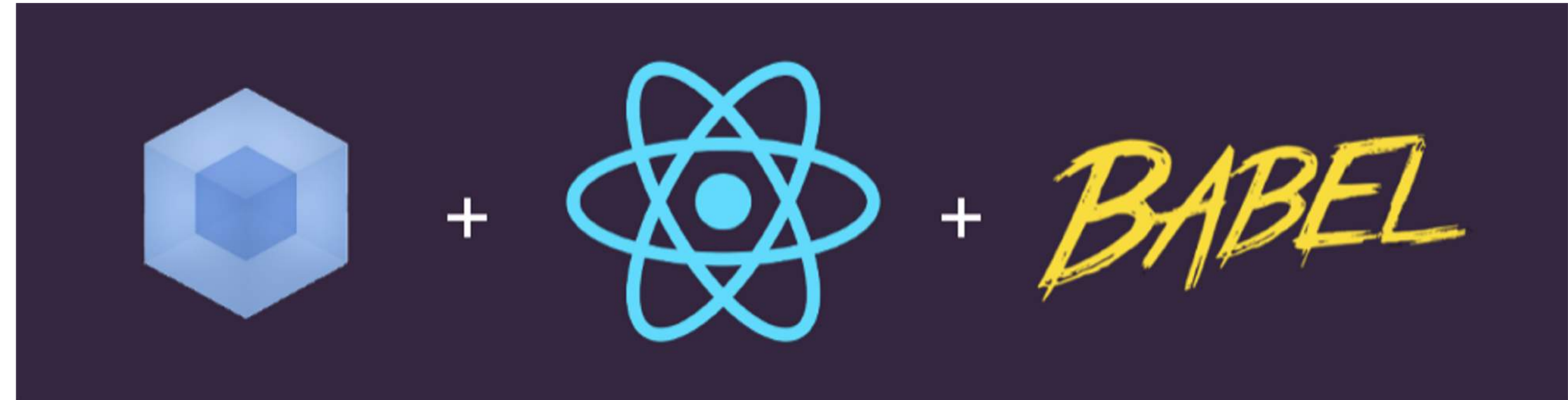
Install using NPM. Navigate to the project directory and type: `npm install react-bootstrap bootstrap`

Import the React-Bootstrap component you want to use. E.g. `import Button from 'react-bootstrap/Button';`

Include the latest styles from the React-Bootstrap CDN. Get the link from the official React-Bootstrap



React Starter Kits



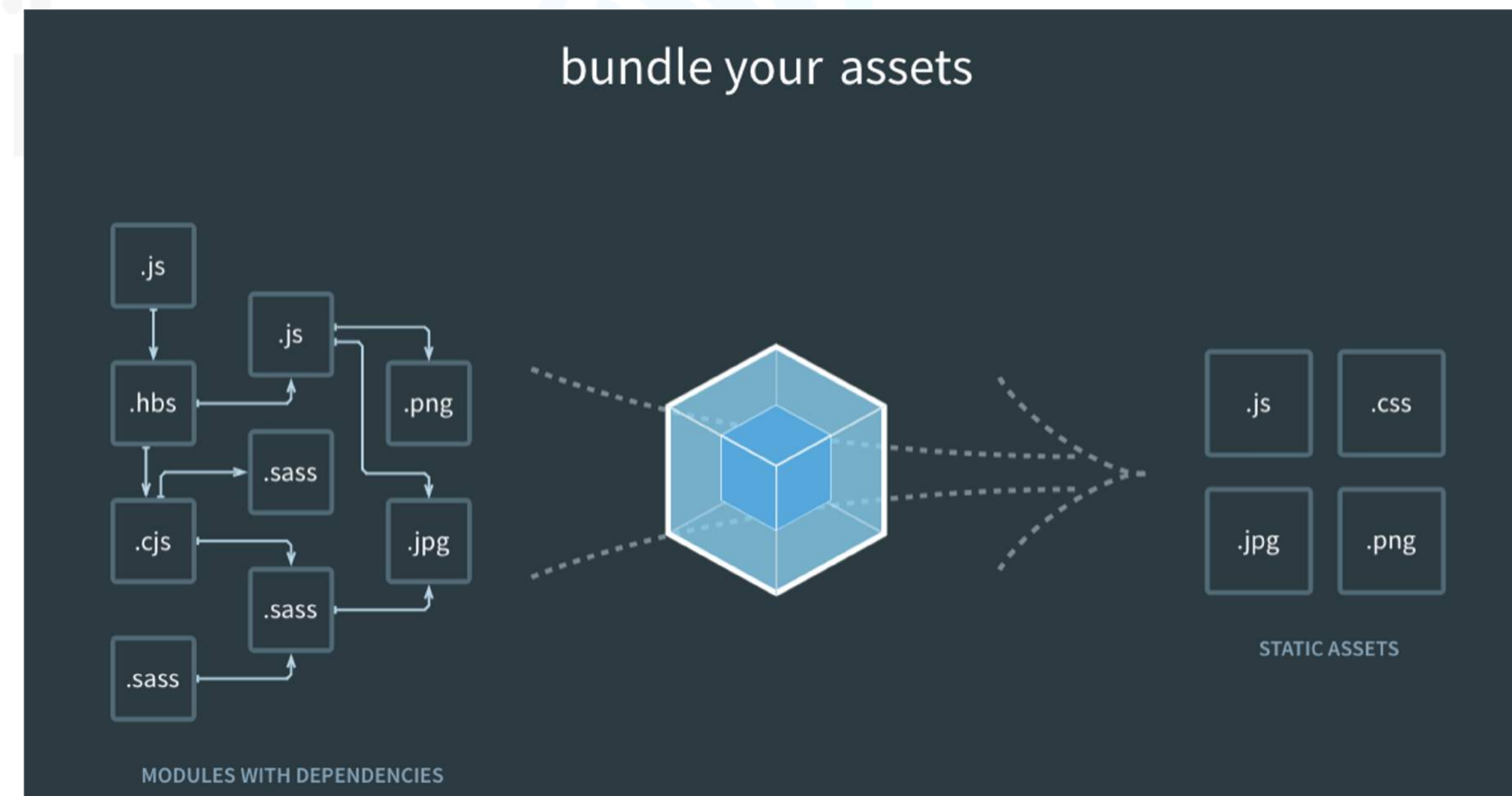
- React relies heavily on other libraries to work.
- Babel is a JavaScript compiler used for writing next generation JavaScript.
- It is particularly relevant to developers who work with React because it can convert JSX syntax.
- Babel is a toolchain that is mainly used to convert ECMAScript 2015+ code into a backwards compatible version of JavaScript in current and older browsers or environments. Here are the main things Babel can do for you:

```
// Babel Input: ES2015 arrow function [1, 2, 3].map(n => n + 1);
```

```
// Babel Output: ES5 equivalent [1, 2, 3].map(function(n) { return n + 1; });
```


React Starter Kits

- React also works best when used with Webpack.
- Webpack is a module bundler.
- It creates a dependency graph that is used to handle all the resources you need for your app.



React Starter Kits

- You could manually install and configure React and all its dependent libraries separately but Facebook, the creators of React, recommend that you use a starter kit instead.
- We will use Create React App to create single-page applications and (later in this level) Next.js for creating static and server-rendered applications.

Debugging a React App

- The developers of React have created various tools to assist with debugging React apps. To access these debugging tools see here (<https://reactjs.org/community/debugging-tools.html>).
- You can also use CodeSandBox for simple debugging.
- It is also important to configure your editor for React development.



Resources

- <https://www.thirdrocktechkno.com/blog/single-page-apps-vs-multi-page-apps-what-to-choose-for-web-development/>
- <https://www.c-sharpcorner.com/article/what-and-why-reactjs/>
- <https://andrewly.medium.com/pros-and-cons-between-single-page-and-multi-page-apps-8f4b26acd9c9>
- <https://www.freecodecamp.org/news/the-model-view-controller-pattern-mvc-architecture-and-frameworks-explained/>
- <https://towardsdatascience.com/everything-you-need-to-know-about-mvc-architecture-3c827930b4c1>
- https://www.w3schools.com/react/react_jsx.asp