

Getting Started

The course presents with a full blown understanding for React.

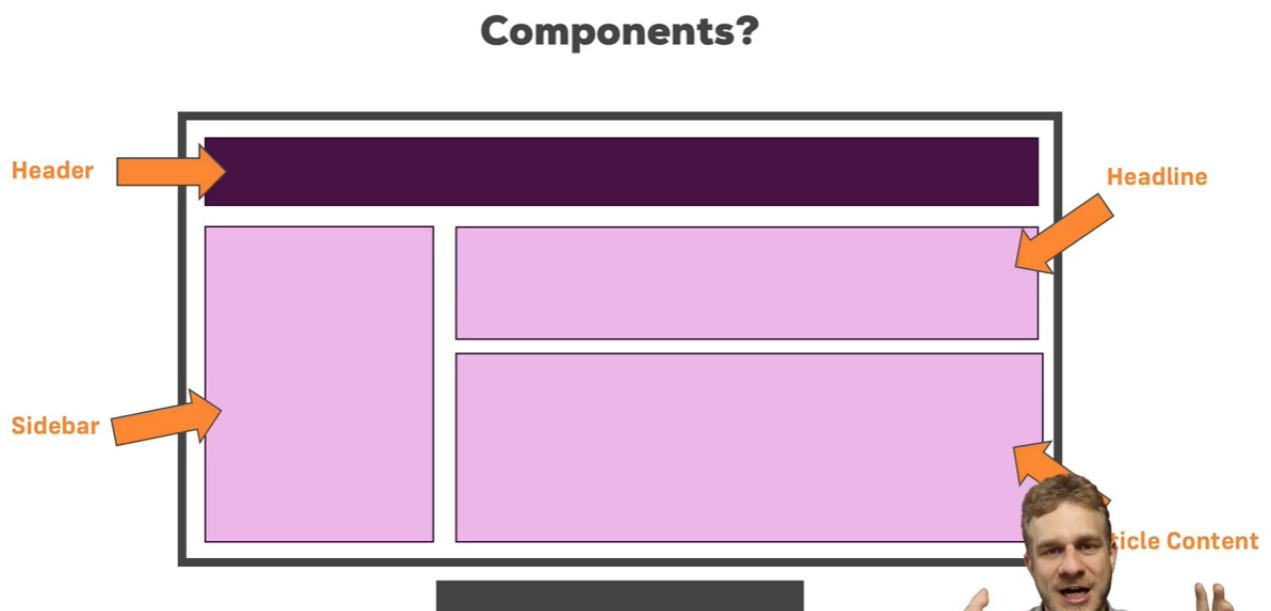
We will look into :

1. Basic javascript and ES6 syntax of javascript
2. Creating components with React
3. Understanding what are Props
4. Handling States for Components
5. Learning Redux for better state management of components
6. We will create a Burger ordering application with React
7. We will learn about front-end routing of SPAs in React
8. Learn authentication through javascript
9. Unit testing of react components
10. Deploying the react application to the Web
11. React Hooks

Getting Started – What, why, how

What is React?

React is a javascript library for building user interfaces. They run in the browser. Things happen instantly and is server-independent. Given any user interface we can always split the user interface into components. Consider the following example.



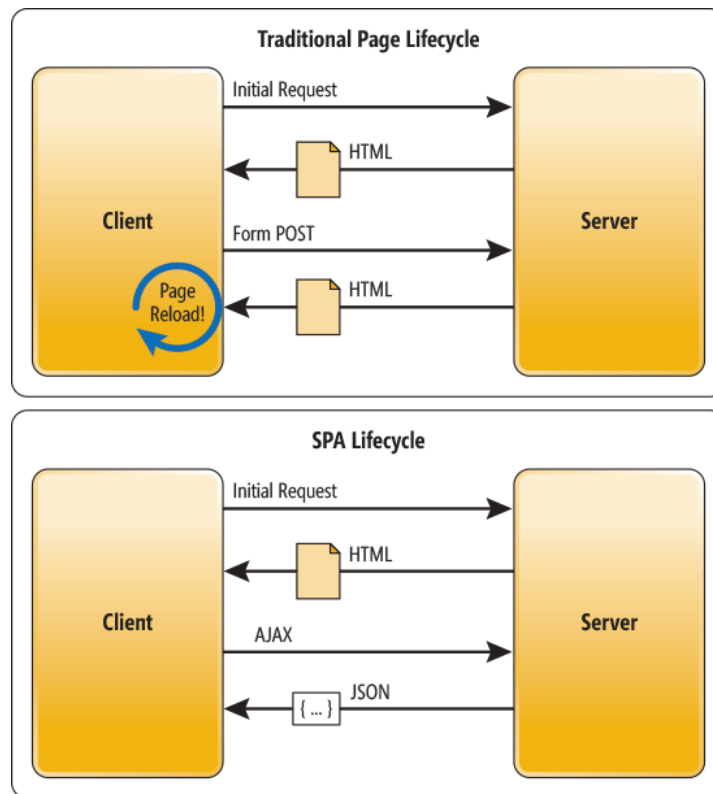
We can reuse the components. This makes the whole code manageable while helping promote team work. If we want to change the behavior of a reusable component, we can make it at one place and it can reflect at other places.

React components are custom HTML elements which can have complex user interface behavior attached to it.

Real-world SPAs and React applications

The React home-page is built with React itself. We can break it down into components. What are SPAs?

Single-Page Applications (SPAs) are Web apps that load a single HTML page and dynamically update that page as the user interacts with the app. SPAs use AJAX and HTML5 to create fluid and responsive Web apps, without constant page reloads. However, this means much of the work happens on the client side, in JavaScript.



Writing out First React Code using Codepen

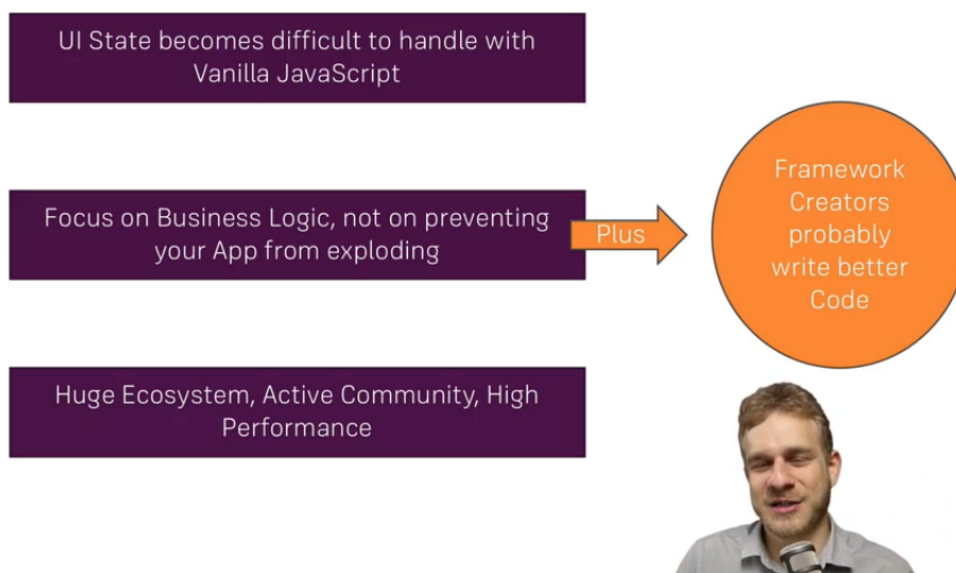
We did a simple ReactDOM render on Codepen. We used Babel for understanding the JSX syntax referred to in the rendering.

Here's the link to the codepen : <https://codepen.io/mohit2494/pen/OYZrLy>

Why React?

1. Helps better manage state than vanilla javascript, jquery
2. focus on business logic
3. Huge ecosystem

Why React?



React Alternatives

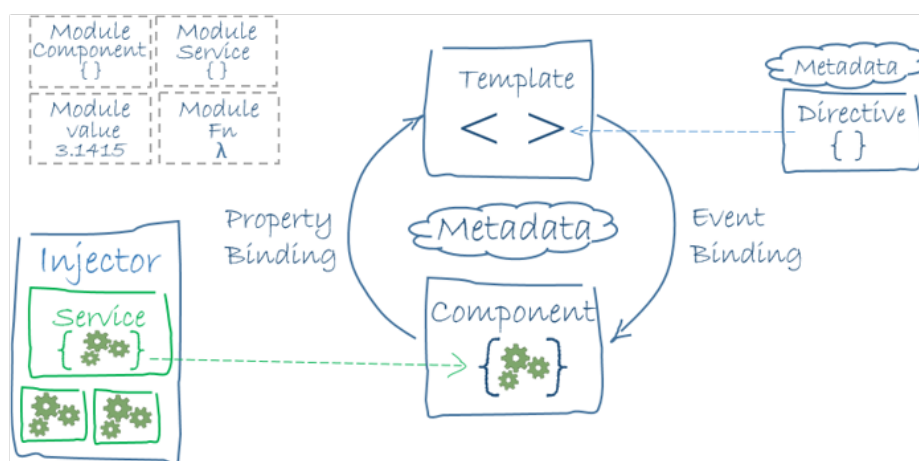
The most popular alternatives for React for building scalable web applications are Angular and Vue. JQuery on the other hand focuses more on DOM manipulation, its traversal etc. Hence, JQuery is not a popular alternative.

Here's a brief introduction about the 2 :

Angular - <https://angular.io/guide/architecture>

Angular is a platform and framework for building client applications in HTML and TypeScript. Angular is written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that you import into your apps.

The basic building blocks of an Angular application are NgModules, which provide a compilation context for components. NgModules collect related code into functional sets; an Angular app is defined by a set of NgModules. An app always has at least a root module that enables bootstrapping, and typically has many more feature modules.

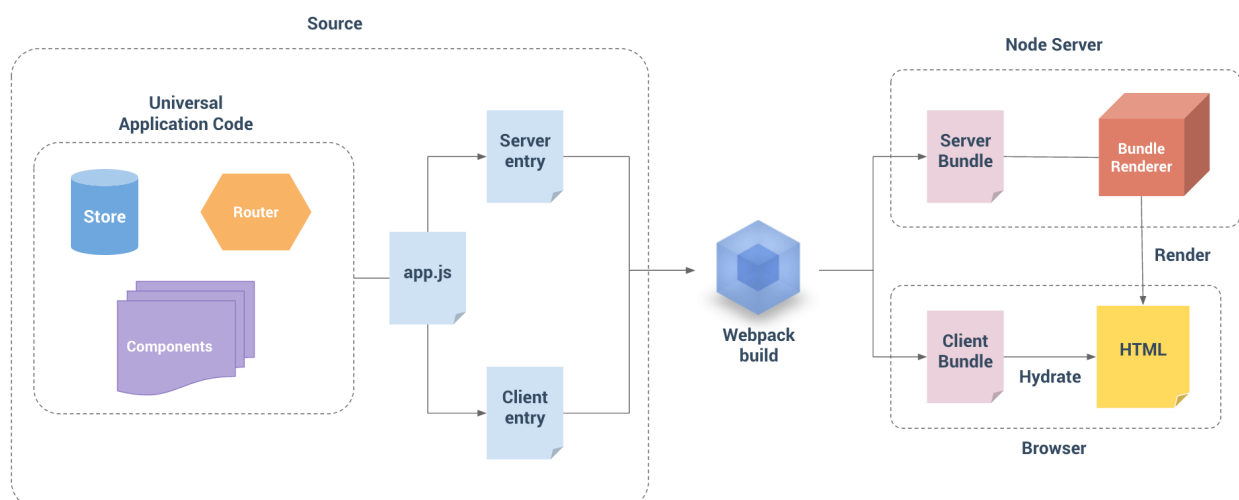


Vue

Vuex doesn't really restrict how you structure your code. Rather, it enforces a set of high-level principles: Application-level state is centralized in the store. The only way to mutate the state is by committing mutations, which are synchronous transactions.

Asynchronous logic should be encapsulated in, and can be composed with actions.

As long as you follow these rules, it's up to you how to structure your project. If your store file gets too big, simply start splitting the actions, mutations and getters into separate files.



Backbone

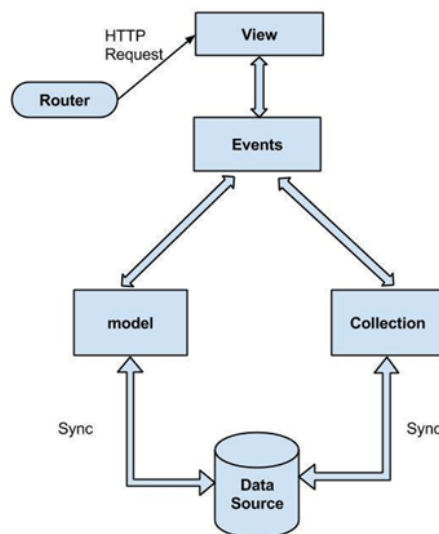
You really don't need to spend so much time with the architecture - it's a framework that can either be used to help put together using simple OO constructs or an event-based design. You basically have ONLY 4 sets of classes (so to speak)

Models - that store actual data that you need to store/manipulate and sync with server in a restful fashion (using JSON/ajax)

Collections - to help you store a list of models and use the wonderful underscore.js to help iterate over it using various operations to make your life A LOT easier

Views - Helps separate concerns. You restrict the rendering operations to this class and also use it to act as a "controller" - to capture events and perform operations on the model. Or to listen to a model's or collection's events so as to update the view when the underlying model changes.

Router - Based on the url fragments you can choose to 'route' your applications logic - loosely speaking. Based on your url fragments you can choose what functions to invoke so you effectively 'route' to the right set of methods based on your logic.



Understanding Single Page Applications vs Multi Page Applications

Ten years later the SPA pattern emerged. SPA is essentially an evolution of the MPA+AJAX design pattern, where the only shell page is generated on the server and all UI is rendered by browser JavaScript code. SPA requests the markup and data separately, and renders pages directly in browser. We can do this because of the new advanced MVVM JavaScript frameworks emerged like AngularJS and KnockoutJS.

So what are the advantages and disadvantages the SPA has over MPA?

SPA advantages over MPA:

Faster page loading times

Improved user experience because the data is loading in the background from server

No need to write the code to render pages on the server

Decoupling of front-end and back-end development

Simplified mobile development; you can reuse the same backend for web application and native mobile application

SPA disadvantages to MPA:

Heavy client frameworks which are required to be loaded to the client

UI code is not compiled, so it's harder to debug and it's exposed to potential malicious user

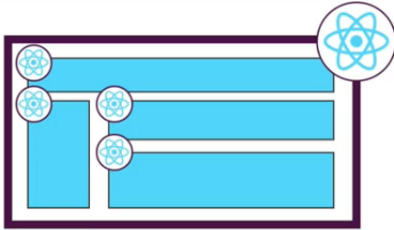
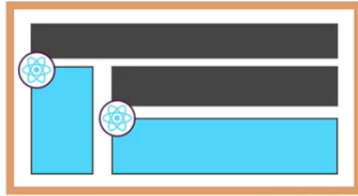
SEO (search engine optimization) implications; since your pages are built in the browser, the search engine crawler will see a different version of the page than that of your users.

Link : <http://www.eikospartners.com/blog/multi-page-web-applications-vs.-single-page-web-applications>

Stackoverflow : <https://stackoverflow.com/questions/21862054/single-page-application-advantages-and-disadvantages>

7. Understanding Single Page Applications and Multi Page Applications

Two Kinds of Applications

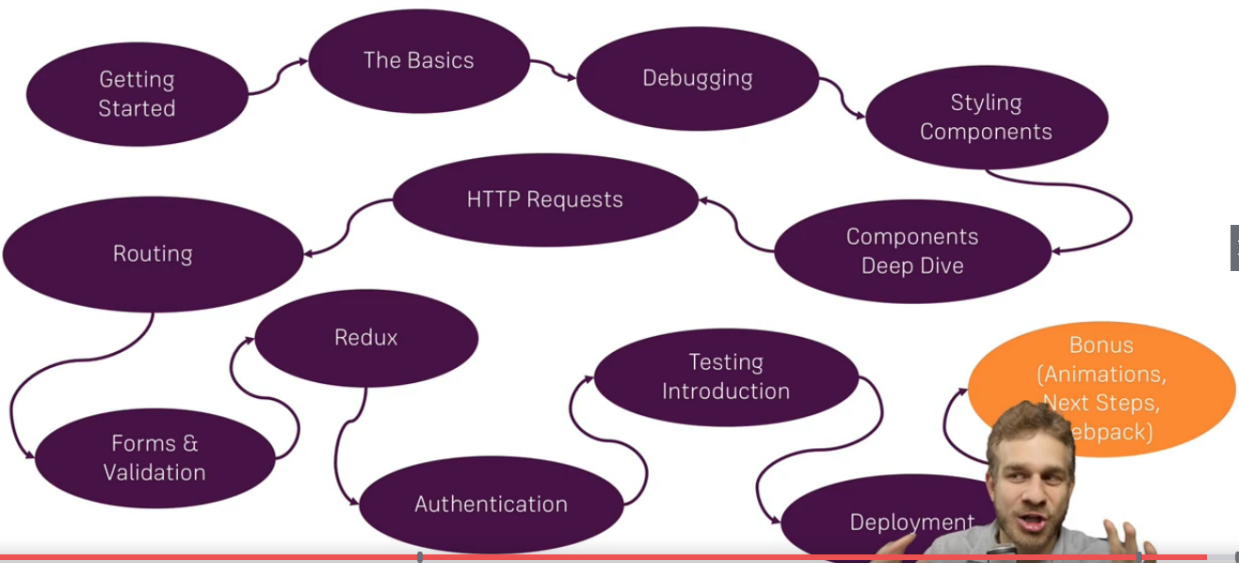
Single Page Applications	Multi Page Applications
Only ONE HTML Page, Content is (re)rendered on Client	Multiple HTML Pages, Content is rendered on Server
	
Typically only ONE ReactDOM.render() call	One ReactDOM.render() call per "widget"

Video player controls: 1.25x, 3:09 / 3:38

Course Outline

8. Course Outline

Course Outline



Video player controls: 1.25x, 7:15 / 7:28