# React Course basic & advance concepts

Basic :

1.Environment Set up:

<https://www.codementor.io/kakarganpat/react-js-environment-setup-with-webpack-kvt082917>

<http://blog.tamizhvendan.in/blog/2015/11/23/a-beginner-guide-to-setup-react-dot-js-environment-using-babel-6-and-webpack/>

create-react-app my-app

cd my-app

npm start

2.Jsx: <https://www.fullstackreact.com/30-days-of-react/day-2/>

<https://reactjs.org/docs/introducing-jsx.html>

3.Rendering Elements:

4.Props: <https://www.agiliq.com/blog/2018/05/understanding-react-state-and-props/>

<https://blog.cloudboost.io/props-vs-state-in-react-js-9c186dcf39b8>

<https://hackernoon.com/understanding-state-and-props-in-react-94bc09232b9c>

<https://hackernoon.com/pass-props-to-react-routers-link-component-2eebda7a63e1>

5.State, setstate::: <https://medium.freecodecamp.org/understanding-setstate-in-react-ea8982168b49>

<https://medium.com/react-tutorials/react-state-14a6d4f736f5>

<https://medium.com/@baphemot/understanding-reactjs-setstate-a4640451865b>

6.Lifecycle methods:: <https://hackernoon.com/reactjs-component-lifecycle-methods-a-deep-dive-38275d9d13c0>

<https://blog.bitsrc.io/react-16-lifecycle-methods-how-and-when-to-use-them-f4ad31fb2282>

<https://programmingwithmosh.com/javascript/react-lifecycle-methods/>

<https://medium.com/react-ecosystem/react-components-lifecycle-ce09239010df>

<https://engineering.musefind.com/react-lifecycle-methods-how-and-when-to-use-them-2111a1b692b1>

http://busypeoples.github.io/post/react-component-lifecycle/

7.Handling events:::: <https://blog.logrocket.com/a-guide-to-react-onclick-event-handlers-d411943b14dd>

<https://medium.freecodecamp.org/the-best-way-to-bind-event-handlers-in-react-282db2cf1530>

<https://medium.freecodecamp.org/this-is-why-we-need-to-bind-event-handlers-in-class-components-in-react-f7ea1a6f93eb>

<https://codeburst.io/react-component-event-handling-660acb1cfd07>

<https://www.agiliq.com/blog/2019/01/working-with-events-reactjs/>

8.Conditional rendering::: <https://blog.logrocket.com/conditional-rendering-in-react-c6b0e5af381e>

<https://www.robinwieruch.de/conditional-rendering-react/>

<https://programmingwithmosh.com/react/conditional-rendering-react/>

<https://hackernoon.com/conditional-rendering-on-react-57a864c2d04d>

9.Lists& keys:: <https://daveceddia.com/display-a-list-in-react/>

<https://o7planning.org/en/12135/react-lists-and-keys-tutorial>

<https://medium.com/@nothingisfunny/keys-and-refs-in-react-fd954457fd75>

10.Forms :;; <https://medium.com/@agoiabeladeyemi/the-complete-guide-to-forms-in-react-d2ba93f32825>

<https://www.codementor.io/blizzerand/building-forms-using-react-everything-you-need-to-know-iz3eyoq4y>

<https://dzone.com/articles/working-with-forms-in-reactjs-using-the-basic-tool>

11.Lifting state up::: <https://chenglou.github.io/react/docs/lifting-state-up.html>

Advance Concepts

1.Context : *Context provides a way to pass data through the component tree without having to pass props down manually at every level.*

prop drilling

<https://medium.com/@MCapoz/passing-data-in-react-with-prop-drilling-904aeb3cb5e>

context Api

<https://hackernoon.com/how-do-i-use-react-context-3eeb879169a2>

2.Error Boundaries : *Error boundaries are React components that catch JavaScript errors anywhere in their child component tree, log those errors, and display a fallback UI instead of the crashed component tree.*

https://hackernoon.com/error-boundaries-in-react-16-32fb8e185a3

3.Refs : refs are used to get reference to a DOM(Document Object Model) node or an instance of a component in a React Application i.e. refs would return the node we are referencing

<https://hackernoon.com/refs-in-react-all-you-need-to-know-fb9c9e2aeb81>

https://medium.com/@mrewusi/a-gentle-introduction-to-refs-in-react-f407101a5ea6

4.Forward Refs : **Ref** forwarding is a technique for automatically passing a **ref** through a component to one of its children. This is typically not necessary for most components in the application.

5.Fragments : A common pattern in React is for a component to return multiple elements. Fragments let you group a list of children without adding extra nodes to the DOM.

https://codeburst.io/fragment-in-react-1cd69f158910

6.Higher-order Components : **a higher-order component is a function that takes a component and returns a new component.**

<https://medium.com/@soorajchandran/introduction-to-higher-order-components-hoc-in-react-383c9343a3aa>

**https://levelup.gitconnected.com/understanding-react-higher-order-components-by-example-95e8c47c8006**

<https://tylermcginnis.com/react-higher-order-components/>

7.Pure components : <https://www.fullstackreact.com/30-days-of-react/day-11/>

<https://codeburst.io/when-to-use-component-or-purecomponent-a60cfad01a81>

https://logrocket.com/blog/pure-functional-components/

8.Jsx in Depth: https://reactjs.org/docs/jsx-in-depth.html

9.Portals : Portals provide a first-class way to render children into a DOM node that exists outside the DOM hierarchy of the parent component.

https://codeburst.io/reacts-portals-in-3-minutes-9b2efb74e9a9

<https://medium.com/@siobhanpmahoney/portals-in-react-js-5d98bb89797c>

<https://www.cronj.com/blog/overview-react-portal/>

<https://hackernoon.com/using-a-react-16-portal-to-do-something-cool-2a2d627b0202>

10.Render Props :  [“render prop”](https://cdb.reacttraining.com/use-a-render-prop-50de598f11ce) refers to a technique for sharing code between React components using a prop whose value is a function.

<https://levelup.gitconnected.com/understanding-react-render-props-by-example-71f2162fd0f2>

<https://medium.com/byteconf/render-props-in-react-6081b6fa3593>

<https://reactgo.com/react-render-props-tutorial/>

https://codeburst.io/reacts-render-props-technique-in-3-minutes-462862bf462e

11.Hooks : <https://medium.freecodecamp.org/an-introduction-to-react-hooks-12843fcd2fd9>

<https://www.codementor.io/shaikh9996/building-a-basic-react-hooks-example-pfy5hy57b>

12.Styles : <https://blog.bitsrc.io/5-ways-to-style-react-components-in-2019-30f1ccc2b5b>

<https://codeburst.io/4-four-ways-to-style-react-components-ac6f323da822>

13.Memo : <https://scotch.io/tutorials/react-166-reactmemo-for-functional-components-rendering-control>

https://medium.com/@trekinbami/using-react-memo-and-memoization-1970eb1ed128

React Routing

: <https://www.npmjs.com/package/react-router-dom>

<https://blog.pshrmn.com/entry/simple-react-router-v4-tutorial/>

https://codeburst.io/getting-started-with-react-router-5c978f70df91

https://reactgo.com/reactrouter/tutorial/

Axios

**Axios** GET and POST request Tutorial in **React**.js. ... **Axios** is a lightweight HTTP client based similar to a Fetch API. **Axios** is promise-based async/await library for the readable asynchronous code

<https://alligator.io/react/axios-react/>

<https://www.npmjs.com/package/react-axios>

<https://flaviocopes.com/axios/#get-requests>

https://reactjs.org/docs/faq-ajax.html

React Redux

<https://reactgo.c>

[https://redux.js.org/basics/usage-with-react m/react-redux-beginners-tutorial/](https://redux.js.org/basics/usage-with-react%20m/react-redux-beginners-tutorial/)

<https://www.valentinog.com/blog/redux/>

<https://www.edureka.co/blog/react-redux-tutorial/>



React Animations

https://reactgo.com/react-animation-tutorial-examples/

Helpul link

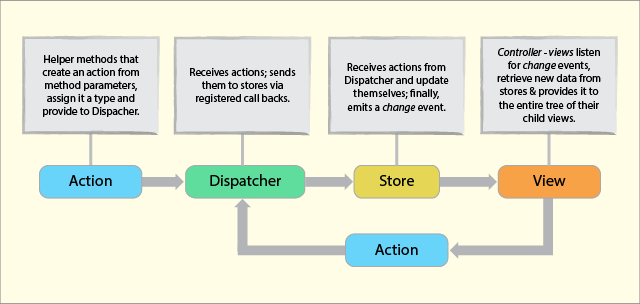
<https://reactjs.org/>

React Flux

<https://scotch.io/tutorials/getting-to-know-flux-the-react-js-architecture>

<https://www.cabotsolutions.com/2017/01/detailed-study-flux-react-js-application-architecture>

<https://www.fullstackreact.com/p/intro-to-flux-and-redux/>



1.Actions : Helper methods that facilitate passing data to the Dispatcher

2.Dispather: Receives actions and broadcasts payloads to registered callbacks

3.Store: Containers for application state & logic that have callbacks registered to the dispatcher

4.View: React Components that grab the state from Stores and pass it down via props to child components.

React Imp interview Questions

https://www.edureka.co/blog/interview-questions/react-interview-questions/

https://www.fullstackreact.com/p/appendix-a-proptypes/

<http://typecast.com/blog/a-more-modern-scale-for-web-typography>

<http://typecast.com/how-it-works>

<https://learnui.design/blog/mobile-desktop-website-font-size-guidelines.html>

ReactRouter Links Documentation

<https://reacttraining.com/react-router/web/example/basic>

TodoLIst

<https://medium.com/@aghh1504/1-simple-react-todo-list-52186b62976b>

<https://egghead.io/>

simple api project

<https://github.com/harinij/100DaysOfCode/tree/master/Day%20005%20-%20React%20App%20with%20Axios%20for%20server%20requests>

react product build

<https://medium.com/@svinkle/how-to-deploy-a-react-app-to-a-subdirectory-f694d46427c1>

<https://blog.bitsrc.io/understanding-fragments-in-react-a1b6e534b01a>

<https://tylermcginnis.com/blog/>

Full React Concepts book 30 days react

<https://www.fullstackreact.com/30-days-of-react/>

<https://github.com/fullstackreact/react-daily-ui>

<https://codepen.io/collection/DoLZRm/>

<https://medium.com/@siddharthac6/getting-started-with-react-js-using-webpack-and-babel-66549f8fbcb8>

# React + Webpack 4 + Babel 7 Setup Tutorial

<https://www.robinwieruch.de/minimal-react-webpack-babel-setup/>

<https://medium.freecodecamp.org/how-to-use-reactjs-with-webpack-4-babel-7-and-material-design-ff754586f618>

<https://webpack.js.org/>

<https://babeljs.io/>

file structure:

<https://hackernoon.com/the-100-correct-way-to-structure-a-react-app-or-why-theres-no-such-thing-3ede534ef1ed>

<https://hackernoon.com/how-to-structure-your-react-app-98c48e102aad>

<https://blog.bitsrc.io/structuring-a-react-project-a-definitive-guide-ac9a754df5eb>

<https://medium.com/@alexmngn/how-to-better-organize-your-react-applications-2fd3ea1920f1>

https://blog.bitsrc.io/5-tools-for-faster-development-in-react-676f134050f2