Introduction to React and Redux

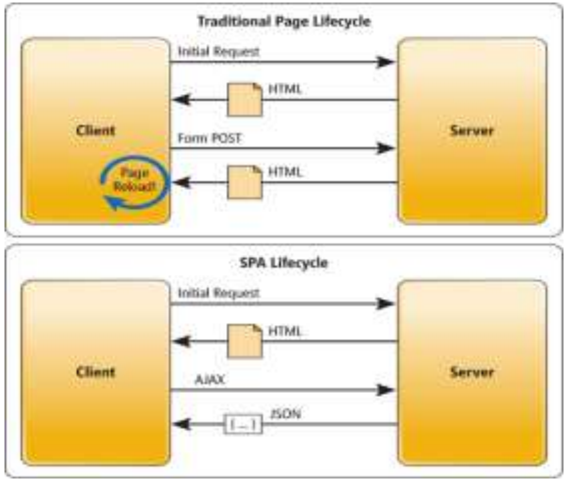
Ho Van Cuong
Front-End Developer

Table of contents

Single-page application (SPA)

A **single-page application** is an app that works inside a browser and does not require page reloading during use.

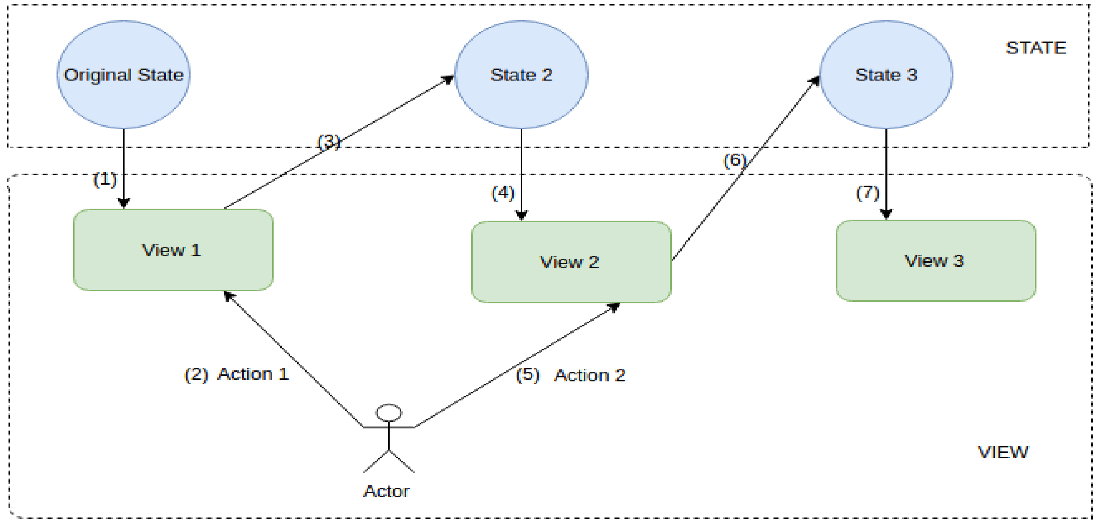
E.g. Gmail, Google Maps, Facebook or GitHub.



ASP.NET - Single-Page Applications: Build Modern, Responsive Web Apps with ASP.NET, n.d.   Web 15 Oct 2017 <https://msdn.microsoft.com/en-us/magazine/dn463786.aspx>

State-based approach

We manage state and view separately.



React

A JavaScript library for building user interface, built by an engineer at Facebook and released to the developer community under an open-source license in 2013.

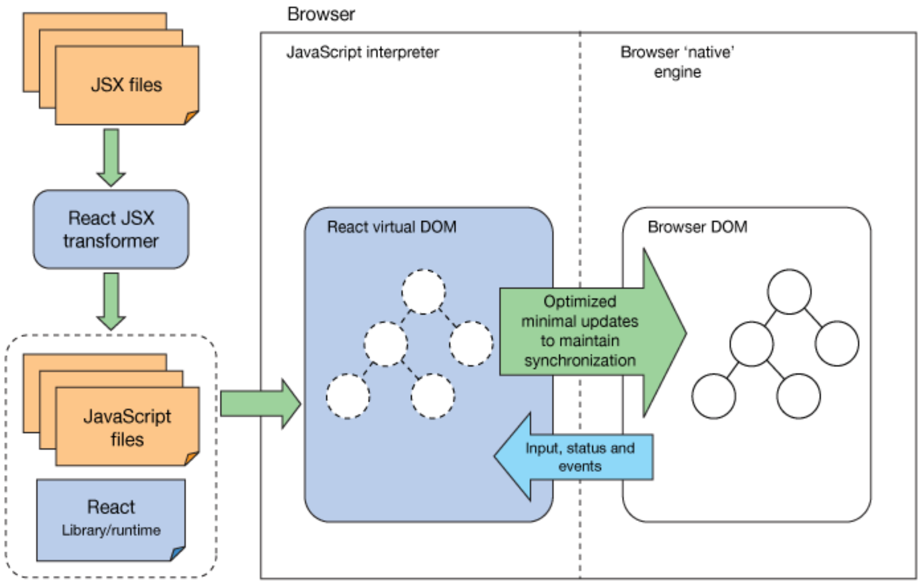
Some key concepts:

- Virtual DOM

- Component-Based
- One-way data flow
- Learn Once, Write Anywhere

How it works

“You declaratively specify your web UI components hierarchy and feed it to React's virtual DOM. Then React takes care of the synchronization of your UI with the browser's actual DOM at the appropriate time.”



React: Create maintainable, high-performance UI components, n.d. Web 15 Oct 2017 <https://www.ibm.com/developerworks/library/wa-react-intro/index.html>

JSX

A syntax extension to JavaScript to produce **React elements**.

```
<div>
    <MyLabel  text={TextLabel} />
    <MyTextfield />
    <MyButton textlabel='OK' />
</div>
```

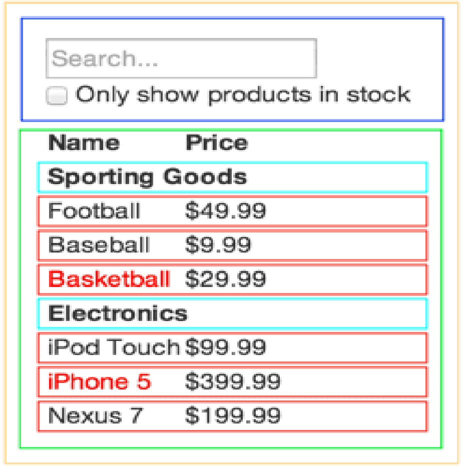
Is translated into JavaScript React API calls like this

```
React.createElement("div", null,
    React.createElement(MyLabel, {text: TextLabel}),
    React.createElement(MyTextfield, null),
    React.createElement(MyButton, {textlabel: "OK"}))
```

Components

Components let you split the UI into independent, reusable pieces, and think about each piece in isolation.

Single responsibility principle: A component should ideally only do one thing.



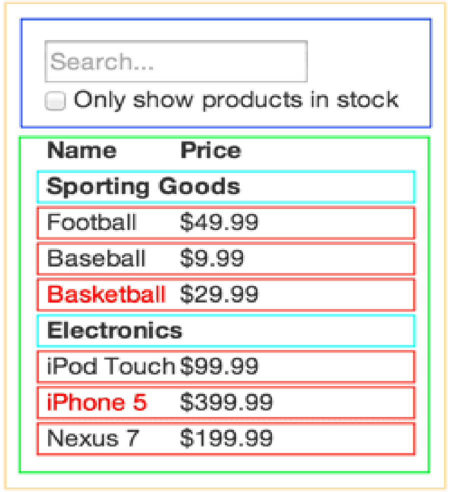
Name	Price
Sporting Goods	
Football	$49.99
Baseball	$9.99
Basketball	$29.99
Electronics	
iPod Touch	$99.99
iPhone 5	$399.99
Nexus 7	$199.99

Thinking in React, n.d.   Web 15 Oct 2017
<https://reactjs.org/docs/thinking-in-react.html>

Stateless Functional Components

Useful for dumb/presentational components.

```
const SearchBar = () => (
  <form>
      <input type="text" placeholder="Search..." />
      <p>
        <input type="checkbox" />
        {' '}
        Only show products in stock
      </p>
   </form>
);
```



Search...

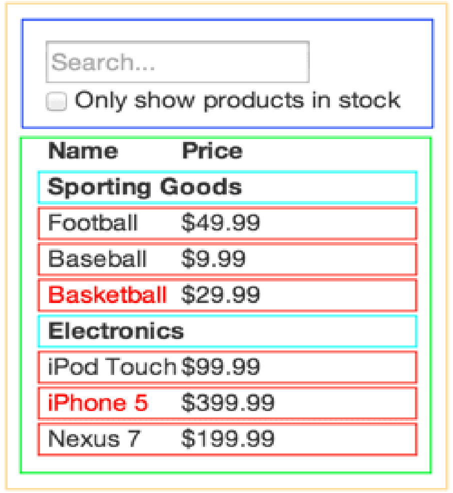
Only show products in stock

Name	Price
Sporting Goods	
Football	$49.99
Baseball	$9.99
Basketball	$29.99
Electronics	
iPod Touch	$99.99
iPhone 5	$399.99
Nexus 7	$199.99

Class Components

Convert a functional component to a class if you need lifecycle methods, refs, or want to optimize performance using shouldComponentUpdate.

```
class SearchBar extends React.Component {
 render() {
   return (
     <form>
       <input type="text" placeholder="Search..." />
       <p>
         <input type="checkbox" />
         {' '}
         Only show products in stock
       </p>
     </form>
   );
 }
}
```


Search...

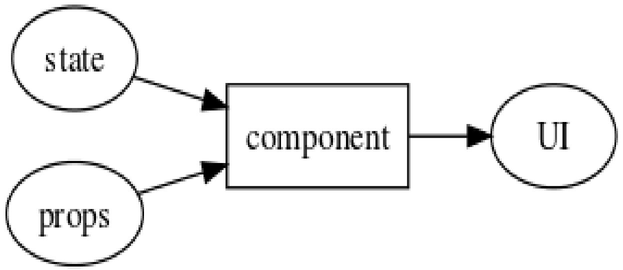
☐ Only show products in stock

Name	Price
Sporting Goods	
Football	$49.99
Baseball	$9.99
Basketball	$29.99
Electronics	
iPod Touch	$99.99
iPhone 5	$399.99
Nexus 7	$199.99

Props and State

```
class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { count: 0 }
  }

  render() {
    return (
      <div>
        <h1>{this.props.name}</h1>
         <button onClick={() =>
            this.setState({count: this.state.count + 1})} />
      </div>
    );
  }
}

ReactDOM.render(<App name=”Sara”/>,
document.getElementById('root'));
```

Props and state holds information about the component.



Composition in CycleJS, Choo, React and Angular2, n.d. Web 15 Oct 2017
<http://blog.krawaller.se/posts/composition-in-cyclejs-choo-react-and-angular2/>

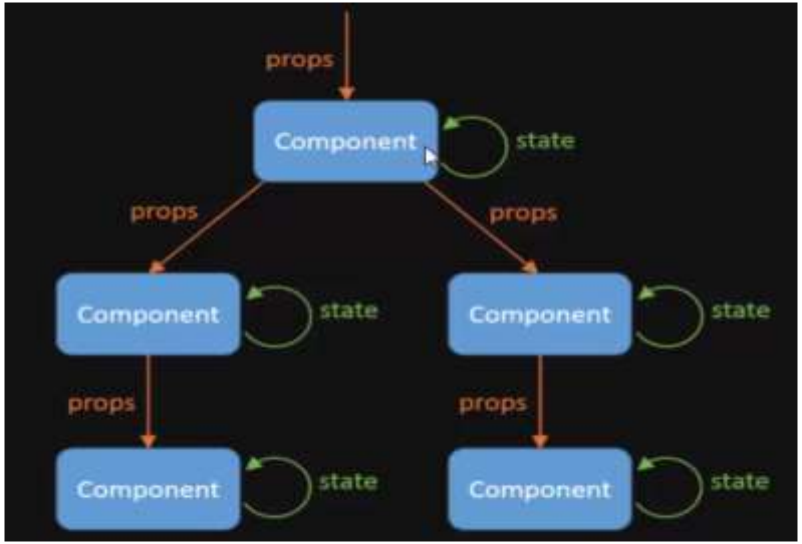
Props and State

Props

- From outside
- Immutable

State

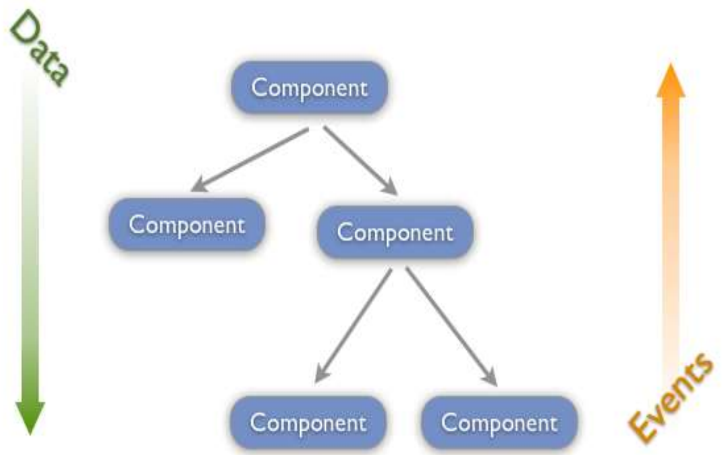
- Created within component
- Mutable



Flow of Data and Events

A parent-component should set the props of a child-component to pass any data from the parent to the child.

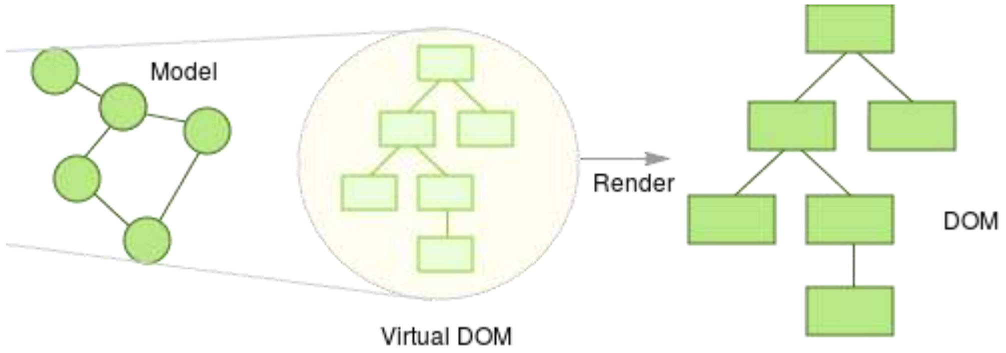
User-events (mouse, keyboard, touches) will always bubble up from the child all the way to the root component, unless handled in between.



Intro to the React Framework, n.d. Web 15 Oct 2017 <https://code.tutsplus.com/tutorials/intro-to-the-react-framework--net-35660>

Virtual DOM

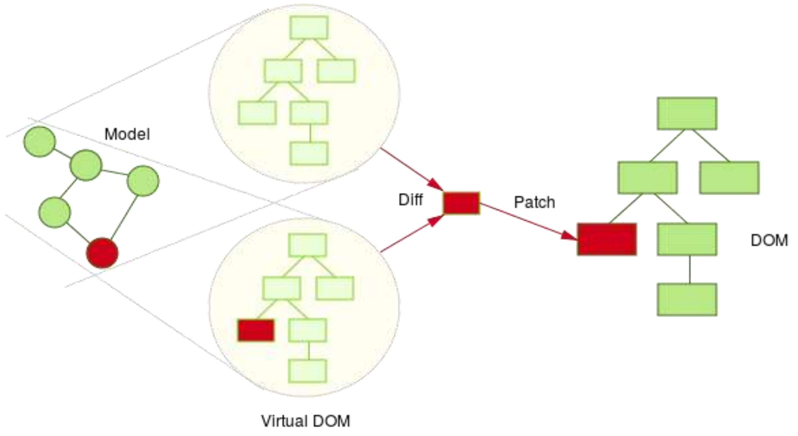
A light-weight, pure JavaScript data structure of plain objects and arrays that *represents* a real DOM object graph.



Change And Its Detection In JavaScript Frameworks, n.d. Web 15 Oct 2017
<http://teropa.info/blog/2015/03/02/change-and-its-detection-in-javascript-frameworks.html>

Virtual DOM

When something changes, render() runs to create new virtual DOM. React diffs with the old one and updates what needed.

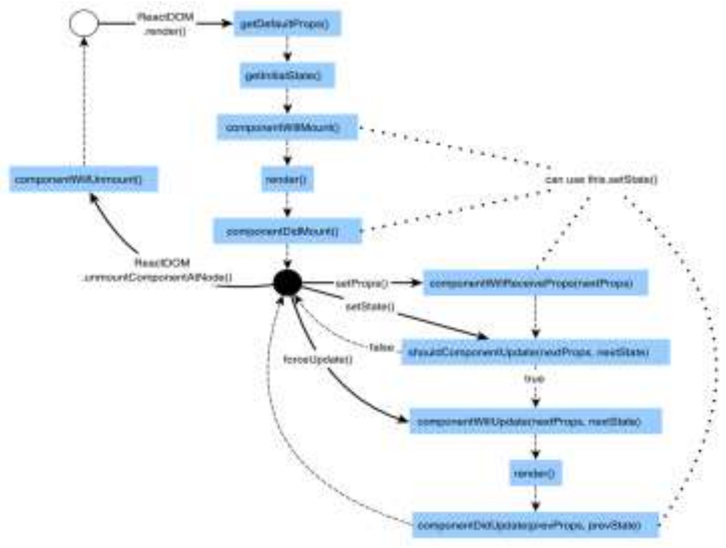


Change And Its Detection In JavaScript Frameworks, n.d. Web 15 Oct 2017
<http://teropa.info/blog/2015/03/02/change-and-its-detection-in-javascript-frameworks.html>

Life Cycle Events

The process where all stages that React component goes through are involved is called the **component's lifecycle**.

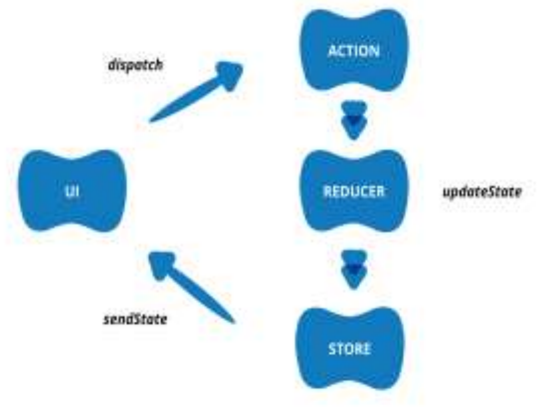
React provides several methods that notify us when certain stage of this process occurs. These methods are called the **component's lifecycle methods**.


ReactDOM .render()
getDefaultProps()
getInitialState()
componentWillMount()
componentWillUnmount()
render()
can use this.setState()
componentDidMount()
ReactDOM .unmountComponentAtNode()
setProps()
componentWillReceiveProps(nextProps)
setState()
false
shouldComponentUpdate(nextProps, nextState)
forceUpdate()
true
componentWillUpdate(nextProps, nextState)
render()
componentDidUpdate(prevProps, prevState)

An Introduction to Life Cycle Events in React, n.d.   Web 15 Oct 2017
<https://tylermcginnis.com/an-introduction-to-life-cycle-events-in-react-js/>

Redux

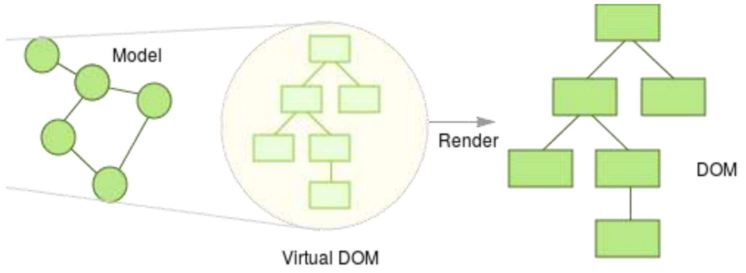
A tool for managing both data-state and UI-state in JavaScript applications.

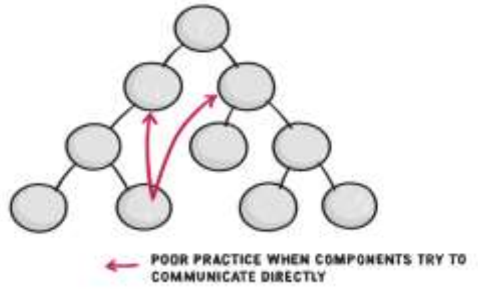


Giamir, Web 15 Oct 2017
<https://github.com/giamir/giamir.github.io/blob/master/_posts/2016-12-22-unit-testing-a-react-redux-app.md>

Why Redux?

- Our code must manage more state than ever before.
- New requirements becoming common in front-end product development.
- Redux attempts to make state mutations predictable by imposing certain restrictions on how and when updates can happen.

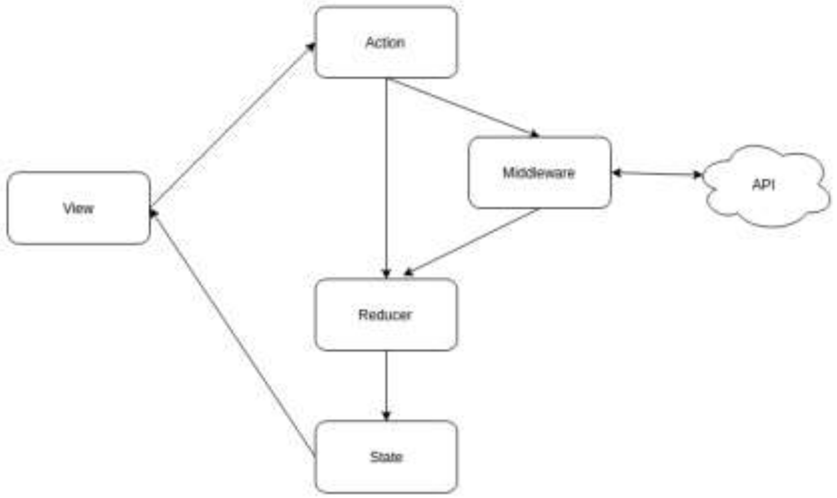




Leveling Up with React: Redux, n.d. Web 15 Oct 2017
<https://css-tricks.com/learning-react-redux/>

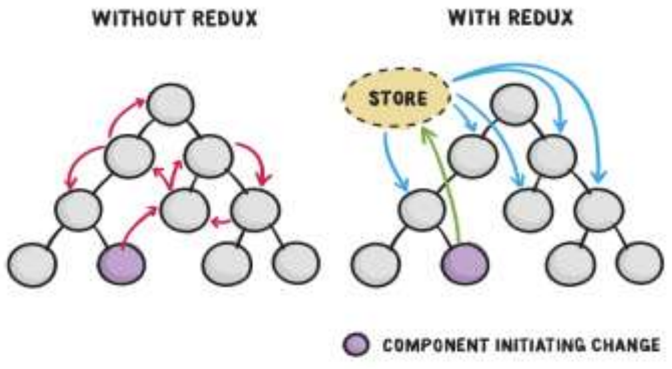
Three Principles

- Single source of truth
- State is read-only
- Changes are made with pure functions



Single source truth

The **state** of your whole application is stored in an object tree within a single **store**, typically a deeply nested object for a real application.



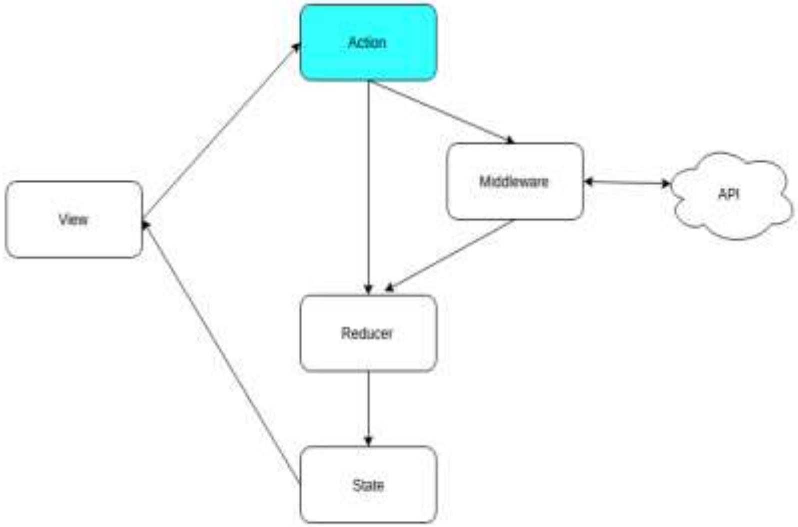
Leveling Up with React: Redux, n.d.   Web 15 Oct 2017
<https://css-tricks.com/learning-react-redux/>

```
console.log(store.getState());

{
  visibilityFilter: 'SHOW_ALL',
  todos: [
    {
      text: 'laundry',
      completed: true,
    },
    {
      text: 'read One Punch Man',
      completed: false
    }
  ]
}
```

State is read-only

The only way to change the state is to emit an **action**, an object describing what happened.

```
var action = {
  type: 'ADD_USER',
  user: {name: 'Dan'}
};

// Assuming a store object has been created already
store.dispatch(action);
```

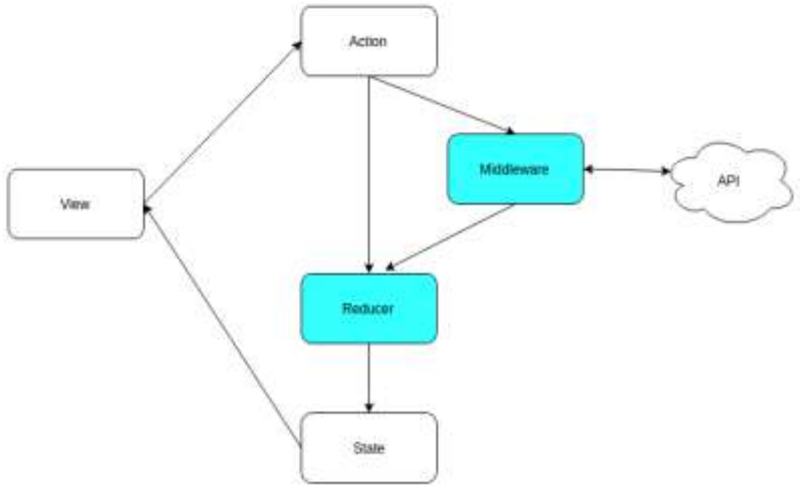


Changes are made with Pure Functions

To specify how the state tree is transformed by actions, you write pure **reducers**.

A reducer takes in current state as an argument and can only modify the state by returning new state.

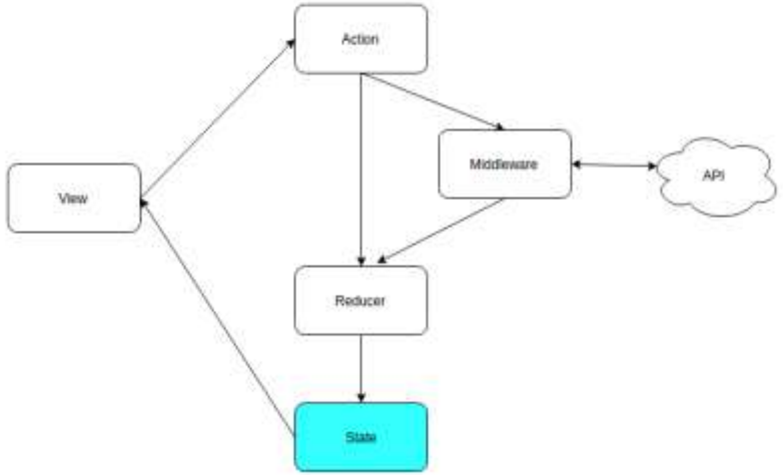
```
// Reducer Function
var someReducer = function(state, action)
{
  ...
  return state;
}
```



Access state from Store

To get state from store

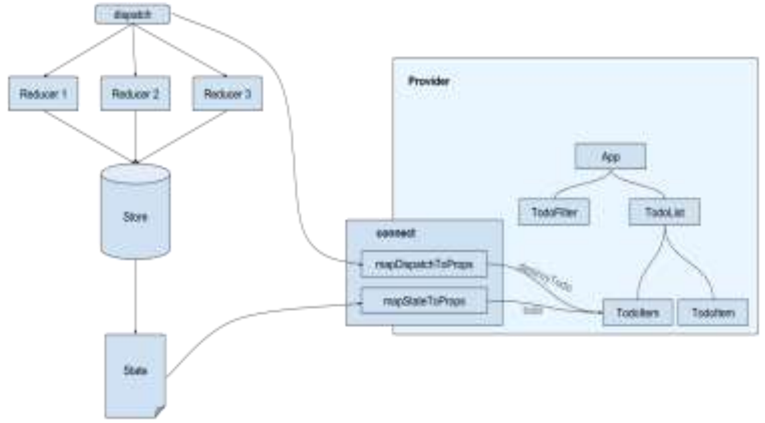
```
store.getState();
```



React Redux

Integrate Redux’s state management into a React application.

```
import { connect, Provider } from 'react-redux';
class compA extends React.Component {
  someFunction() {
    console.log(this.props.users);
    this.props.actionA();
  }
}
const mapStateToProps = (store)  => ({
  users: store.userState.users
});
const mapDispatchToProps = {
   actionA: importedActionA
}
export default
connect(mapStateToProps)(mapDispatchToProps)(compA);

ReactDOM.render(
  <Provider store={store}> <compA /> </Provider>,
document.getElementById('root'));
```



React-redux "connect" explained, n.d.   Web 15 Oct 2017
<http://www.sohamkamani.com/blog/2017/03/31/react-redux-connect-explained/>

References

React Redux demos
<https://github.com/cuonghovan/react-redux-demo>

React
<https://reactjs.org/>

Redux
<http://redux.js.org/>

Change And Its Detection In JavaScript Frameworks
<http://teropa.info/blog/2015/03/02/change-and-its-detection-in-javascript-frameworks.html>

React: Create maintainable, high-performance UI components
<https://www.ibm.com/developerworks/library/wa-react-intro/index.html>

Leveling Up with React: Redux
<https://css-tricks.com/learning-react-redux/>

Understanding the React Component Lifecycle
<http://busypeoples.github.io/post/react-component-lifecycle/>

Q&A