React cheat sheet

create-react-app

```
create-react-app YOUR_APP_NAME
```

Stateless component

```
import React from 'react'

const YourComponent = () => <div>aaa</div>

export default YourComponent
```

Class component

```
import React from 'react'

class YourComponent extends React.Component {
  render() {
```

```
    return <div>aaa</div>
  }
}

export default YourComponent
```

Properties in stateless component

```
const YourComponent = ({ propExample1, example2 }) => (
  <div>
    <h1>properties from parent component:</h1>
    <ul>
      <li>{propExample1}</li>
      <li>{example2}</li>
    </ul>
  </div>
)

// <YourComponent propExample1="aaa" example2="bbb" />
```

Properties in class component

Children

```
class YourComponent extends React.Component {
  render() {
    return (
      <div>
        <h1>
          properties from parent component:
        </h1>
        <ul>
          <li>{this.props.propExample1}</li>
          <li>{this.props.example2}</li>
        </ul>
      </div>
    )
  }
}
```

```
const Component1 = (props) => (
  <div>{props.children}</div>
)

const Component2 = () => (
  <Component1>
    <h1>Component 1</h1>
  </Component1>
)
```

Architecture

State

```
class CountClicks extends React.Component {
  state = {
    clicks: 0
  }

  onButtonClick = () => {
    this.setState(prevState => ({
      clicks: prevState.clicks + 1
    }))
  }

  render() {
    return (
      <div>
        <button onClick={this.onButtonClick}>
          Click me
```

```
          </button>
          <span>
            The button clicked
            {this.state.clicks} times.
          </span>
        </div>
      )
    }
  }
```

useRef hook

```
function TextInputWithFocusButton() {
  const inputEl = useRef(null);
  const onButtonClick = () => {
    // `current` points to the mounted text input element
    inputEl.current.focus();
  };
  return (
    <>
      <input ref={inputEl} type="text" />
      <button onClick={onButtonClick}>Focus the input</button>
    </>
  );
```

```
}
```

Author: Jonathan Gilyadov

React Router

[React-router Cheat Sheet]

```
import {
  BrowserRouter,
  Route
} from 'react-router-dom'

const Hello = () => <h1>Hello world!</h1>

const App = () => (
  <BrowserRouter>
    <div>
      <Route path="/hello" component={Hello}/>
    </div>
  </BrowserRouter>
)

// open: http://localhost:3000/hello
```

react-redux provider

[Redux Cheat Sheet]

```
import React from 'react'
import { render } from 'react-dom'
import { Provider } from 'react-redux'
import { createStore } from 'redux'
import todoApp from './reducers'
import App from './components/App'

const store = createStore(todoApp)

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

```
import { connect } from 'react-redux'

YourComponent = connect(
  mapStateToProps,
  mapDispatchToProps
)(YourComponent)

export default YourComponent
```

```
import React, { useState } from 'react'

function Example() {
  const [count, setCount] = useState(0)

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click here to increase the counter.
      </button>
    </div>
  )
}
```

Ref

```
class AutoFocusTextInput extends React.Component {
  constructor(props) {
    super(props);
    this.textInput = React.createRef();
```

```
  }

  componentDidMount() {
    this.textInput.current.focus();
  }

  render() {
    return (
      <input ref={this.textInput} />
    );
  }
}
```

Higher Order Component

```
import React from 'react'
import Loading from '../components/Loading'

const withLoading = WrappedComponent => {
  return (props = {}) => {
    if (props.isLoading) {
      return <Loading />
    }
    return <WrappedComponent {...props} />
```

```
  }
}

export default withLoading

// --------

const MyComponent = ({}) => <div /> // ...
const WithLoadingComponent = withLoading(MyComponent)
```

Render Props

```
import React from 'react'

const MOBILE_VIEW_WIDTH_THRESHOLD = 600

class MediaQuery {
  state = {
    shouldShowMobileView: false
  }

  componentDidMount() {
    addEventListener('resize', this.updateResizeStatus)
  }
```

```
  componentWillUnmount() {
    removeEventListener('resize', this.updateResizeStatus)
  }

  updateResizeStatus() {
    if (screen.width <= MOBILE_VIEW_WIDTH_THRESHOLD) {
      this.setState({
        shouldShowMobileView: true
      })
    }
  }

  render() {
    return this.props.children(
      this.state.shouldShowMobileView
    )
  }
}

export default MediaQuery

// --------

import React from 'react'
```

```
import MobileView from '../components/MobileView'
import DefaultView from '../components/DefaultView'

const Screen = () => (
  <MediaQuery>
    {shouldShowMobileView =>
      shouldShowMobileView ? (
        <MobileView />
      ) : (
        <DefaultView />
      )
    }
  </MediaQuery>
)
```

Author: Daler Asrorov