1. 리덕스의 개념은 무엇이고 왜 쓰는가

**Redux**

- 상태관리 라이브러리

- 애플리케이션 클라이언트의 state를 관리하기 위한 거대한 이벤트 루프

**Redux 원칙**

- single source of truth : 앱 전체는 단일 스토어로 관리

- state is read only: 상태는 읽기전용. 상태를 변경하기 위해서는 액션객체를 이용하여 액션을 발생 시켜야 한다.

- changes are made with pure functions: 상태의 변화를 일으키기 위해서는 reducer를 사용한다. reducer는 이전의 상태와 액션을 받아서 다음 상태를 반환 하는 함수로서 순수 함수로 구현해야한다.

**Redux 사용하는 이유**

모던 자바스크립트 SPA 에서 예전보다 많은 인터렉션 및 복잡한 데이터를 사용하면서 클라이언트에서 관리해야할 상태들이 많아지고 복잡해짐에 따라 더욱 필요하게 되었다.

상태 관리 라이브러리 중 대표적인 Redux 는 데이터가 집중화되어 있어 (단일 스토어) 예측 가능하며 데이터 흐름이 단방향이기 때문에 디버깅이 쉽다. 특히 Redux는 거의 동일한state 개념(ex) read-only, setState() )이 존재하는 react 와 함께 사용하기 좋다.

**출처**

1. <https://www.bangseongbeom.com/redux-benefits-caveats.html>

2. <https://medium.com/@wooder2050/%EB%A6%AC%EB%8D%95%EC%8A%A4-redux-%EB%8A%94-%EC%99%9C-%EC%93%B0%EB%8A%94-%EA%B1%B4%EB%8D%B0-2eaafce30f27>

3. <https://voidsatisfaction.github.io/2017/02/24/what-is-redux/>

4, <https://velog.io/@velopert/Redux-1-%EC%86%8C%EA%B0%9C-%EB%B0%8F-%EA%B0%9C%EB%85%90%EC%A0%95%EB%A6%AC-zxjlta8ywt>

2. hook 과 mobx는 무엇이고 왜 쓰는가

**Hook**

- React 에서 상태와 생명주기를 관리하는 새로운 방법

- 기존 this.state와 life cycle API (didComponentMount) 등의 사용을 대체가능

- 특정 상황에 따라 HOC, render prop도 대체가능

- 함수형 컴포넌트에서도 Hook 을 통해 클래스형 컴포넌트에서 지원하는state 및 life cycle API를 어느정도 대체할 수 있다.

**Mobx**

- redux 와는 또다른 State 관리 라이브러리

- Component 와 State 를 연결하는 번잡한 보일러플레이트 코드들을 데코레이터 제공으로 깔끔하게 해결

- redux 와 비교하여 러닝커브가 낮은편이고 코드량이 적으며 객체 지향 친화적이다.

**출처:**

<http://pmionlinebusiness.com/2698875>

<https://boxfoxs.tistory.com/395>

<https://medium.com/@ehddnjs8989/react-hooks-%EC%82%AC%EC%9A%A9%EC%9D%B4%EC%9C%A0-ce03c66a53b0>

<https://woowabros.github.io/experience/2019/01/02/kimcj-react-mobx.html>

<https://ddwroom.tistory.com/75>

3. webpack과 babel은 무엇이고 왜 쓰는가

**webpack**

- webpack 은 모던 자바스크립트 애플리케이션을 위한 자바스크립트 정적 모듈 번들러이다.

- webpack 은 브라우저에서 사용되는 자바스크립트 파일 뿐만 아니라 스타일 시트, 이미지등 모든 것을 자바스크립트 모듈로 로딩해서 사용할 수 있게끔 해준다.

**Webpack 사용하는 이유**

- 네트워크 병목 현상을 해결할 수 있다.

- 모듈 단위로 개발이 가능하다

- 코드 압축/최적화

- 각 loader 를 통해 브라우저에서 사용하는 정적파일들을 불러와 사용할 수 있다.

**Babel**

- babel은 자바스크립트 컴파일러, 최신의 자바스크립트 코드를 이전 단계의 자바스크립트 코드로 변환가능하게 해주는 Transpiler 이다.

- babel은 구문변환, polyfill, 소스코드 변환등 기능을 제공한다.

**Babel 사용하는 이유**

- 다양한 웹브라우저 및 환경에 대해 대응하기 위하여 사용한다. (cross-browser 지원을 위해서)

**출처**

<https://brunch.co.kr/@topherlee/29>

<https://medium.com/@ljs0705/babel-%EC%9D%B4%ED%95%B4%ED%95%98%EA%B8%B0-a1d0e6bd021a>

<https://beomy.tistory.com/41>

<https://medium.com/@ljs0705/spa-single-page-app-%EC%97%90%EC%84%9C-webpack%EC%9D%84-%EC%82%AC%EC%9A%A9%ED%95%98%EB%8A%94-%EC%9D%B4%EC%9C%A0-ce7d3f82fe9>

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