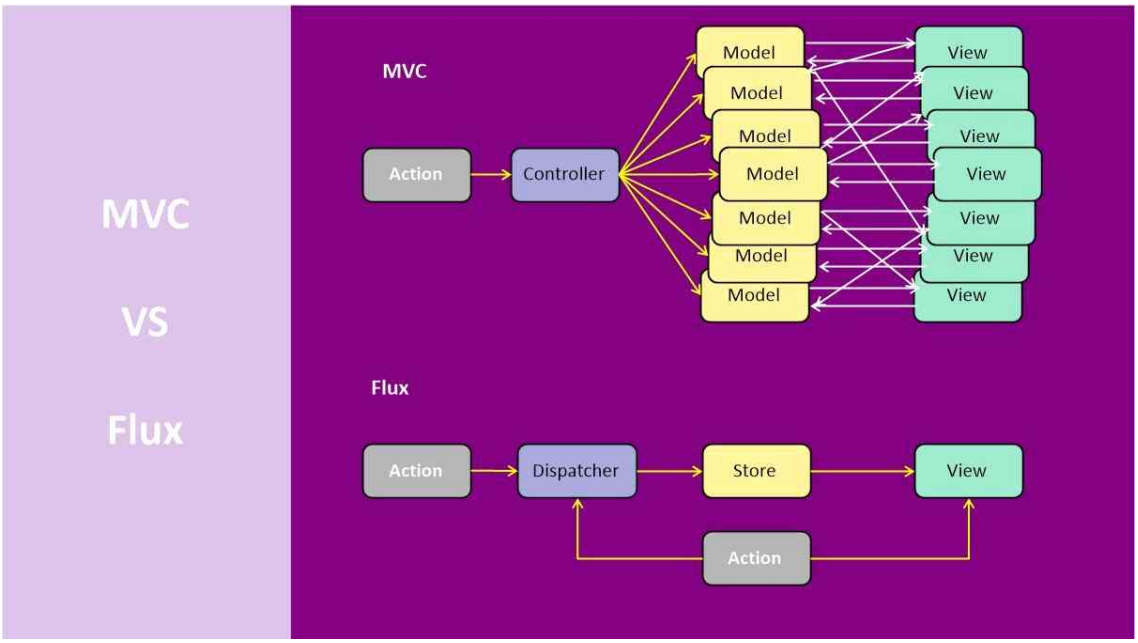
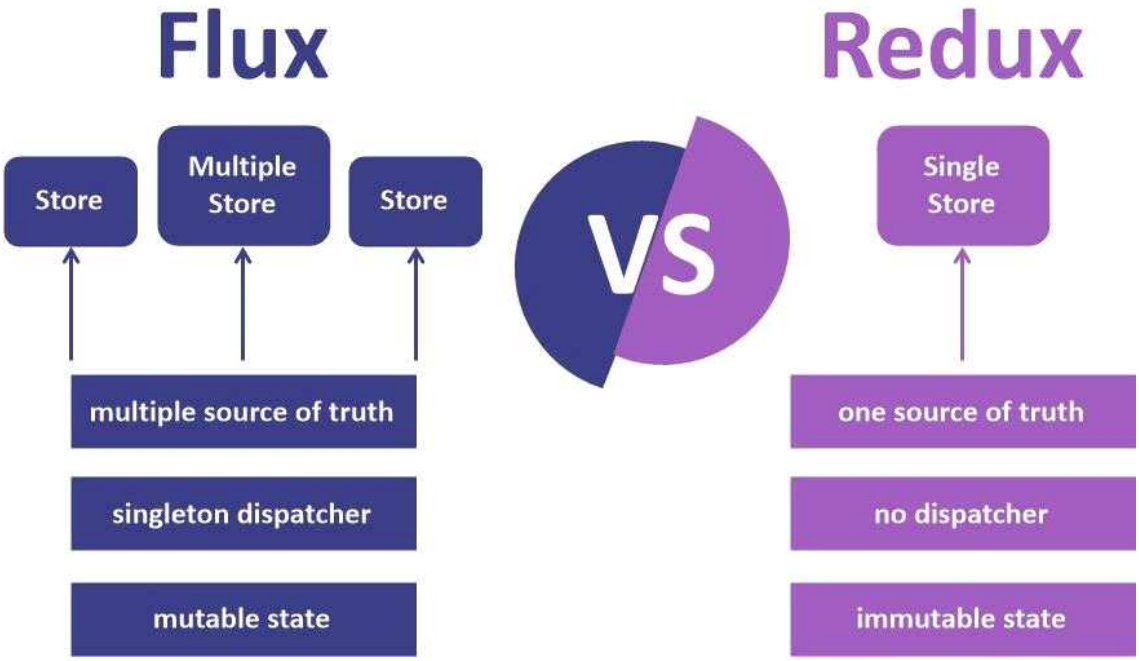




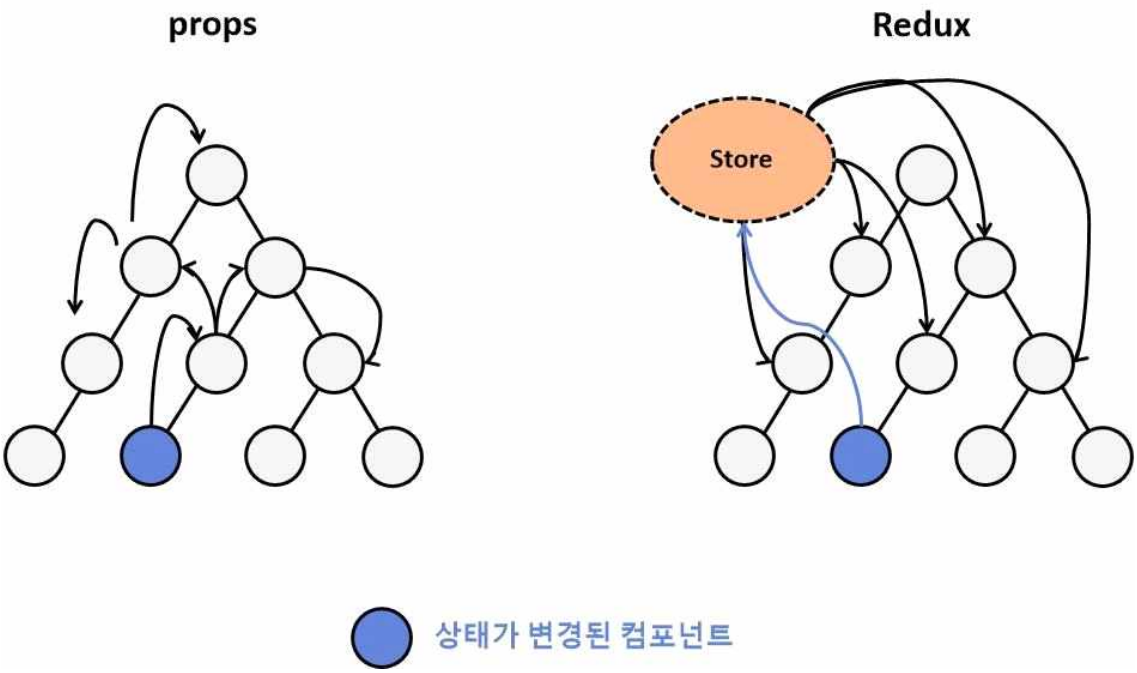
MVC : Model View Controller

- 프로그래밍 디자인 패턴으로 데이터의 테이블에 대응 되는 Model과 이를 조작하기 위한 Controller 그리고 model의 값을 보여주는 View로 구분 된다.
- 사용자의 인터페이스와 비즈니스 로직을 분리해 서로 영향 없이 쉽게 어플리케이션을 수정할 수 있는 패턴

Flux : Facebook 에서 만든 clinet-side web application을 구축 할 때 사용하는 application architecture (앱구조)

design pattern(디자인패턴)이다. Flux는 대규모 프로젝트에서 복잡해지는 MVC구조의 단점을 보완하는 단방향 데이터 흐름의 구조이다.

REDUX : Flux의 몇 가지 중요한 특성에 영감을 받아 개발이 되었다. Flux와 달리 리덕스는 dispatcher라는 개념과 다수의 Store도 존재 하지 않았다. 대신 리덕스는 하나의 Root에 하나의 Store만이 존재하며 순수함수의 의존하는데 이 순수 함수는 이것들을 관리하는 추가적인 entity 없이도 조합하기 쉽다.



Redux가 필요한 이유
상태 관리의 필요성이 있기 때문이다.

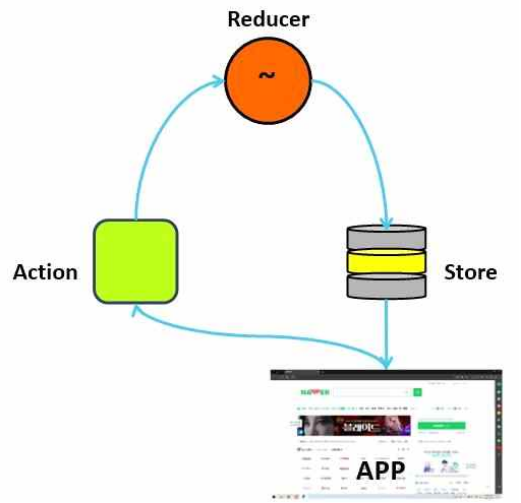
독립된 형태의 컴포넌트들로 구성이 된 트리로 형성이 되면서 각 컴포넌트의 관계가 형성이 됩니다.

Props를 사용을 했을 때
특정 컴포넌트의 상태가 변경이 되었을 때 그 변경된 상태를
자신을 포함하고 있는 상위 컴포넌트에게 전달을 통해
다른 부분에 영역에서도 그 변화를 감지해 애플리케이션에서 전체적으로 업데이트 하려고 하면 복잡성이 높아진다.

왜냐하면 전역에서 상태관리를 해줄 수 있는게 없기 때문이다.

Redux를 사용을 했을 때는
Store를 통해 특정컴포넌트의 상태가 변경되었을 때 이를
Store에 알리는 dispatch라는 작업을 하게 되면
store는 dispatch된 내용을 토대로 연결되어있는 컴포넌트들을 업데이트 하게 됩니다.
이와 같이 상태관리를 해줘야지 어플리케이션을 효율적으로 관리할 수 있습니다.

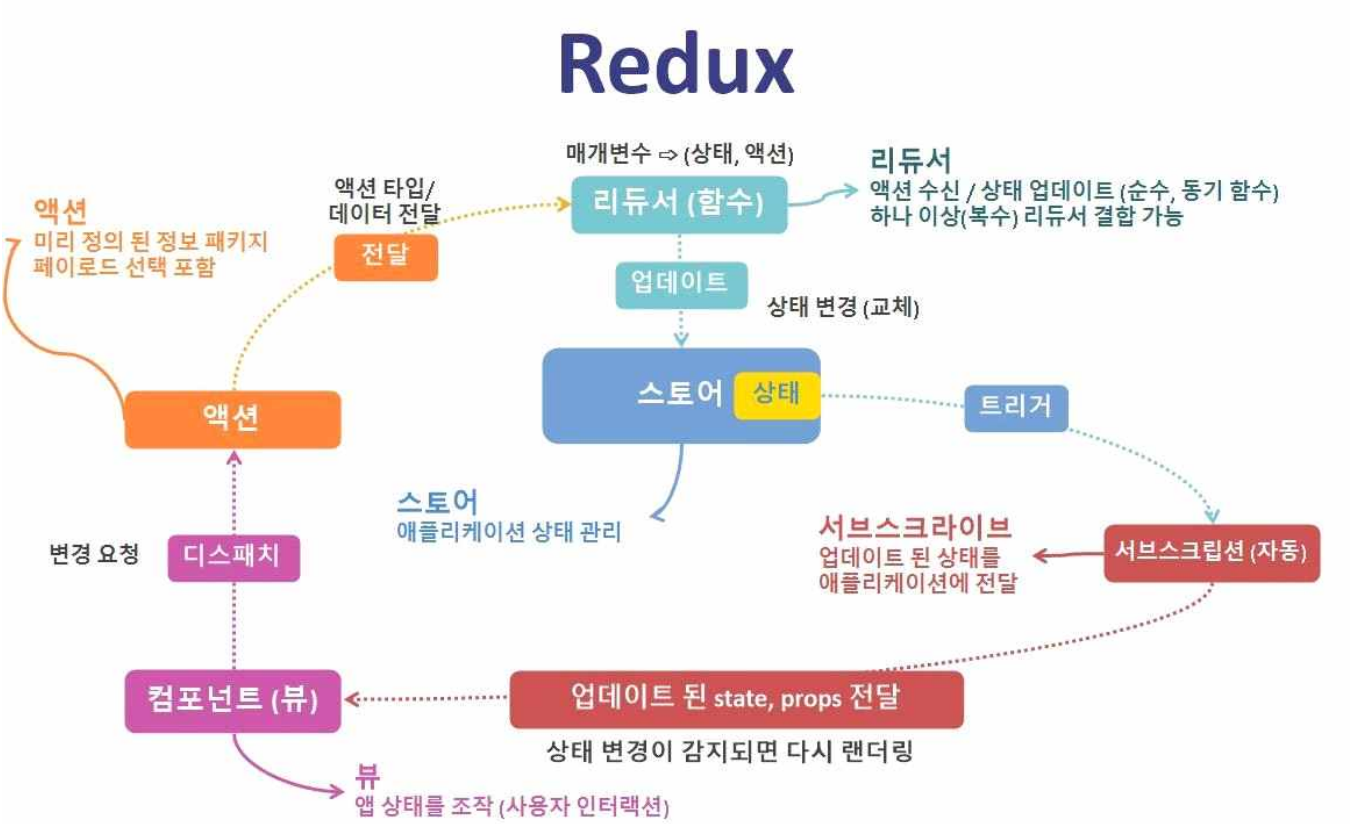
물론 리액트는 이런 문제를 개선하기 위해 context api또한 리덕스와 비슷한 형태를 가지고 있습니다. provider를 통해서 공급받은 데이터를 처리할 수 있는데 상대적으로 규모가 큰 상태를 관리 할 때는 context api보다 리덕스를 사용하는 편이 좋습니다.



Redux의 작동 흐름 (3원칙)

1. 애플리케이션의 상태는 모두 한 곳에서 집중 관리됩니다
스토어에서 관리가 됩니다. 별도의 컴포넌트마다 동기화가 필요없이 스토어에서 연결되어 있는 app에 있는 컴포넌트에게 일괄처리를 해준다

2. 상태는 불변데이터 입니다. 오직 action 만이 상태 교체를 할 수 있습니다. 즉 app에 잇는 특정 컴포넌트가 상태를 교채될 경우 app이 상태를 가지고 있지 않기 때문에 action 한테 요청을 하고 액션은 리듀서 한테 함수한테 상태변경하기를 요구합니다. 이 과정에서 app 컴포넌트가 액션에 상태를 요청하기 때문에 예측이 가능합니다
action의 상태변경 흐름을 디버깅 하기가 용이합니다.
3. action이 상태 변경 요청을 받으면 상태 변경을 action이 하는게 아니라 액션은 상태를 변경 할 수 있는 리듀서 함수한테 상태를 바꿔달라고 요청을 합니다. 리듀서 함수는 액션이 요청을 받아서 자신이 가지고 있는 상태 데이터를 변경한 다음 store 한테 말하고 store는 변경된 상태에 따라서 연결된 app컴포넌트를 업데이트


Redux
매개변수 ⇨ (상태, 액션)
리듀서
액션 수신 / 상태 업데이트 (순수, 동기 함수)
하나 이상(복수) 리듀서 결합 가능
액션 타입/
데이터 전달
리듀서 (함수)
액션
미리 정의 된 정보 패키지
페이로드 선택 포함
전달
업데이트
상태 변경 (교체)
스토어
상태
액션
트리거
스토어
애플리케이션 상태 관리
서브스크라이브
업데이트 된 상태를
애플리케이션에 전달
서브스크립션 (자동)
변경 요청
디스패치
컴포넌트 (뷰)
업데이트 된 state, props 전달
상태 변경이 감지되면 다시 랜더링
뷰
앱 상태를 조작 (사용자 인터랙션)