

리덕스 라이브러리 이해하기

Redux: 리액트에서 가장 많이 사용하는 상태관리 라이브러리

- 특징
 - 컴포넌트의 상태 업데이트 관련 로직을 분리할 수 있음
 - 컴포넌트간 같은 상태를 공유가 용이 함
- v16.3 이후 부터는 똑같은 작업이 가능하지만 규모가 큰 프로젝트의 경우 리덕스가 유리

action

- 상태에 어떠한 변화가 필요할 경우 **action**이 발행 됨
- action은 **type** 필드를 반드시 가지고 있어야 함

```
{  
  type: 'TOGGLE_VALUE'  
}
```

```
{  
  type: 'ADD_TODO',  
  data : {  
    id: 1, text: 'learn redux'  
  }  
}
```

action creator

- 사용에 용이하게 액션 객체를 만들어 주는 함수

```
function addTodo(data) {  
  return {  
    type: 'ADD_TODO',  
    data  
  }  
}
```

reducer

- 변화를 일으키는 함수
- 액션을 발생시키면 리듀서는 현재상태와 비교하여 새로운 상태를 만들어 반환해 줌

```
const initialState = {  
  counter: 1  
};  
function reducer(state = initialState, action) {  
  switch (action.type) {  
    case INCREMENT: return { counter: state.counter + 1};  
    default: return state;  
  }  
}
```

store

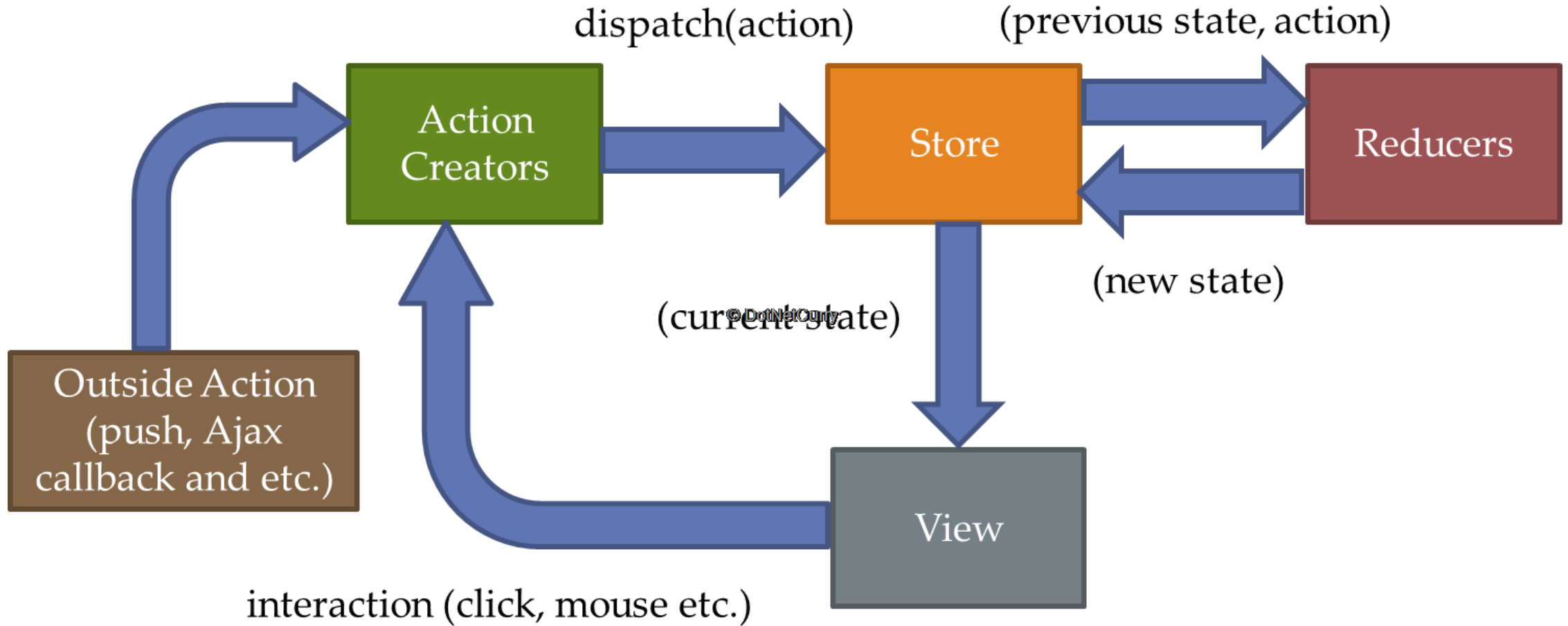
- 한 개의 프로젝트는 단 하나의 스토어만 가질 수 있음
- 현재 App 상태와 리듀서가 들어 있음

dispatch

- store의 내장 함수 중 하나
- action을 발생시키는 역할
- `dispatch(action)` 과 같은 형태로 사용
- 이 함수가 호출되면 스토어는 리듀서 함수를 실행시켜 새로운 상태를 만들어 줌

subscribe

- store의 내장 함수 중 하나
- 리스너 함수가 action이 dispatch될 때마다 호출됨



리액트 없이 사용하는 리덕스

리덕스는 리액트에 종속되는 라이브러리가 아님

CODE 확인

리덕스의 세 가지 규칙

- 단일 스토어
 - 하나의 app 안에는 하나의 redux
- 읽기 전용 상태
 - 상태를 업데이트 할 때 기존의 객체는 건드리지 않고 새로운 객체를 생성해야 함
 - 데이터가 변경되는 것을 감지하기 위해 shallow equality 검사를 하기 때문
- 리듀서는 순수 함수
 - 순수 함수
 - 리듀서는 이전 상태와 액션 객체를 파라미터를 받음
 - 파라미터 외의 값에는 의존하면 안됨
 - 이전 상태는 건드리지 않고 변화를 준 새로운 상태 객체 반환
 - 똑같은 파라미터로 호출된 리듀서 함수는 언제나 똑같은 결과 값 반환
 - 랜덤 값, 시간 값, 네트워크 요청 등의 함수 요청 하면 안됨
 - 바깥에서(액션을 만드는 과정, 리덕스 미들웨어 등) 호출해야함