Ngrx

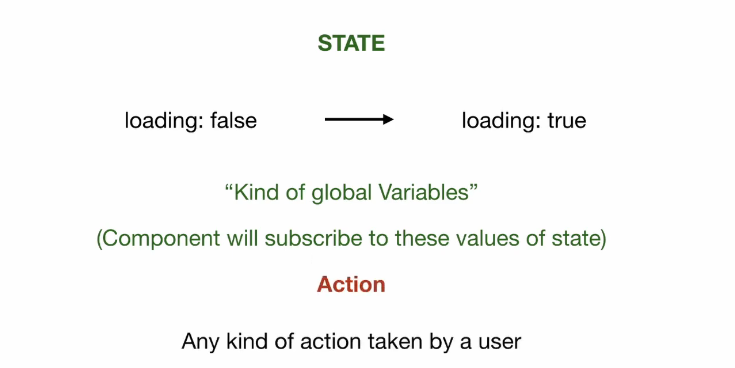
RxJS + Redux = NgRx

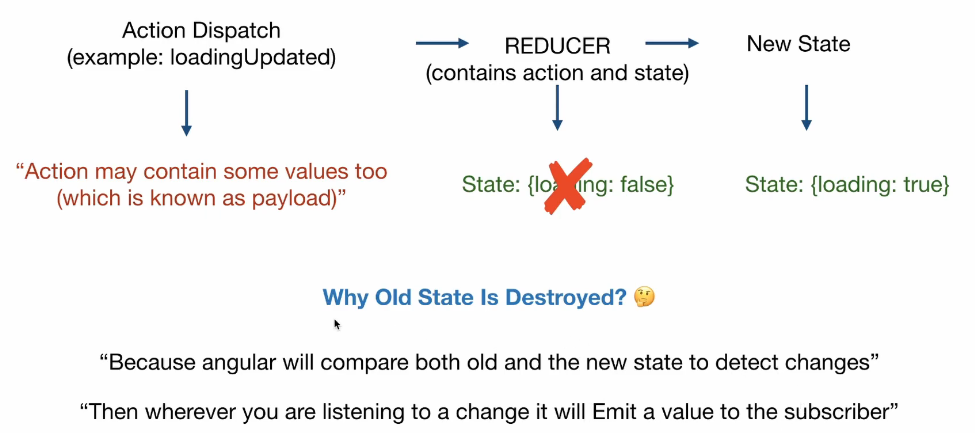
Redux is basically a global state management library taken from react.

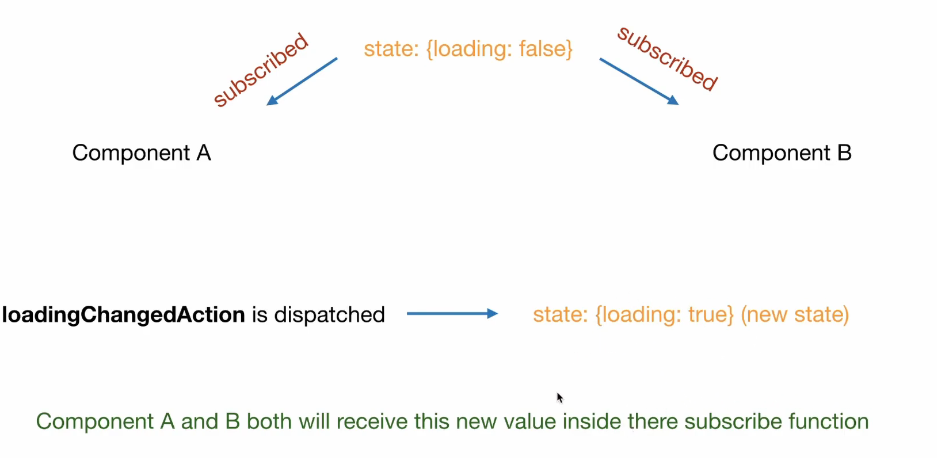
All components can share the same data among themselves.

Lets have 2 folders in app directory, one for components and another for containers.

There are three main parts of redux: Actions, State, and Reducer







Reducer does not edit the current state, instead it creates a new state which is emitted to its subscribers later.

Steps:

1. Create action  
   export const increment = createAction('[Counter Component] Increment');
2. Define a reducer to handle changes in the values based on the provided actions  
   [export](https://ngrx.io/api/store-devtools/DevToolsFeatureOptions#export) const initialState = 0; [export](https://ngrx.io/api/store-devtools/DevToolsFeatureOptions#export) const counterReducer = [createReducer](https://ngrx.io/api/store/createReducer)( initialState, [on](https://ngrx.io/api/store/on)(increment, (state) => state + 1), [on](https://ngrx.io/api/store/on)(decrement, (state) => state - 1), [on](https://ngrx.io/api/store/on)(reset, (state) => 0) );
3. Import the StoreModule from @ngrx/store and the reducer created above from reducer file
4. Add the StoreModule.forRoot function in the imports array of your AppModule with an object containing the state\_variable i.e. count (for example) and the reducer that manages the state. The StoreModule.forRoot() method registers the global providers needed to access the Store throughout your application.  
   example: StoreModule.forRoot({ count: counterReducer })
5. Now create your component. Explore more on use of async in templates  
   <div>Current Count: {{ count$ | async }}</div>
6. Inject the store into your component and connect to the store’s state\_variable state.  
     
   constructor(private store: [Store](https://ngrx.io/api/store/Store)<{ count: number }>) { this.count$ = store.select('count'); }
7. Implement functions and dispatch actions to the store.  
     
   increment() { this.store.dispatch(increment()); }