Mushkin's House of Killer Plants

To get started run the following commands

- `$ git clone https://github.com/lnewsom/app-of-horrors.git`
- `$ cd app-of-horrors/`
- `$ git checkout step-one`
- `$ npm install`
- `$ ng serve -o`

If everything is set up properly, the app should be running
in your default browser on localhost 4200


LITTLE APP
OF
HORRORS
Saving the Day with NgRx

LARA NEWSOM

 Source Allies Software Engineer

 Twitter: @LaraNerdsom

 GitHub: /lnewsom/app-of-horrors

 source allies

My Lil' App ❤

Awww, CUTE!!!

Ha ha, don't look at my trash...

“I’m going in!”

-Any Developer Working on a Nightmare Component


ME
FEED ME
MY
NO
LONGER
LIL'
APP

You can have NgRx installed and working with one command!


ARRANGEMENTS

NgRx is not all or nothing.

You can move one piece at a time!!

OH

YASS

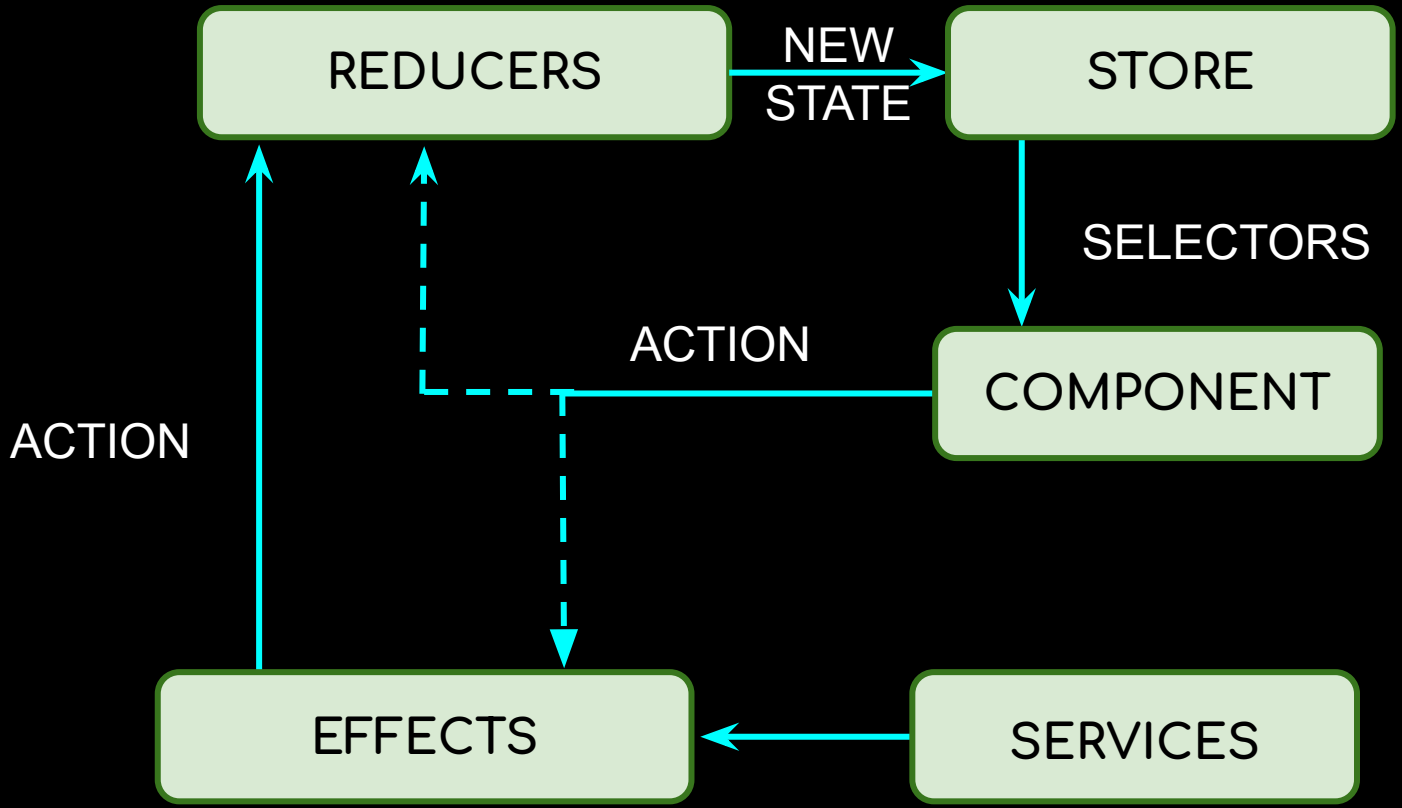
You get Chrome Redux Devtools for FREEEEEEE!!!

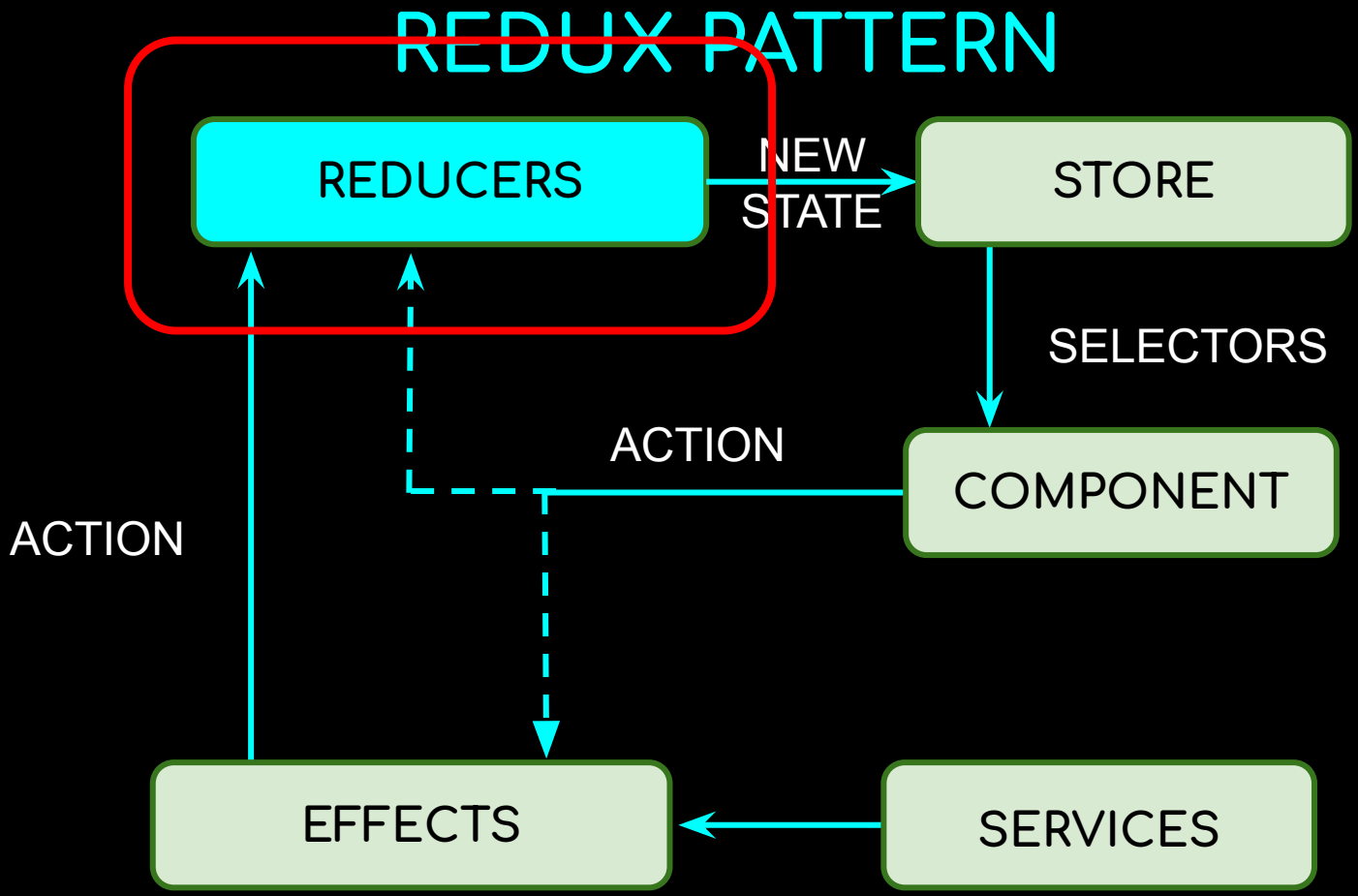

Save Me
NgRx

What is NgRx?

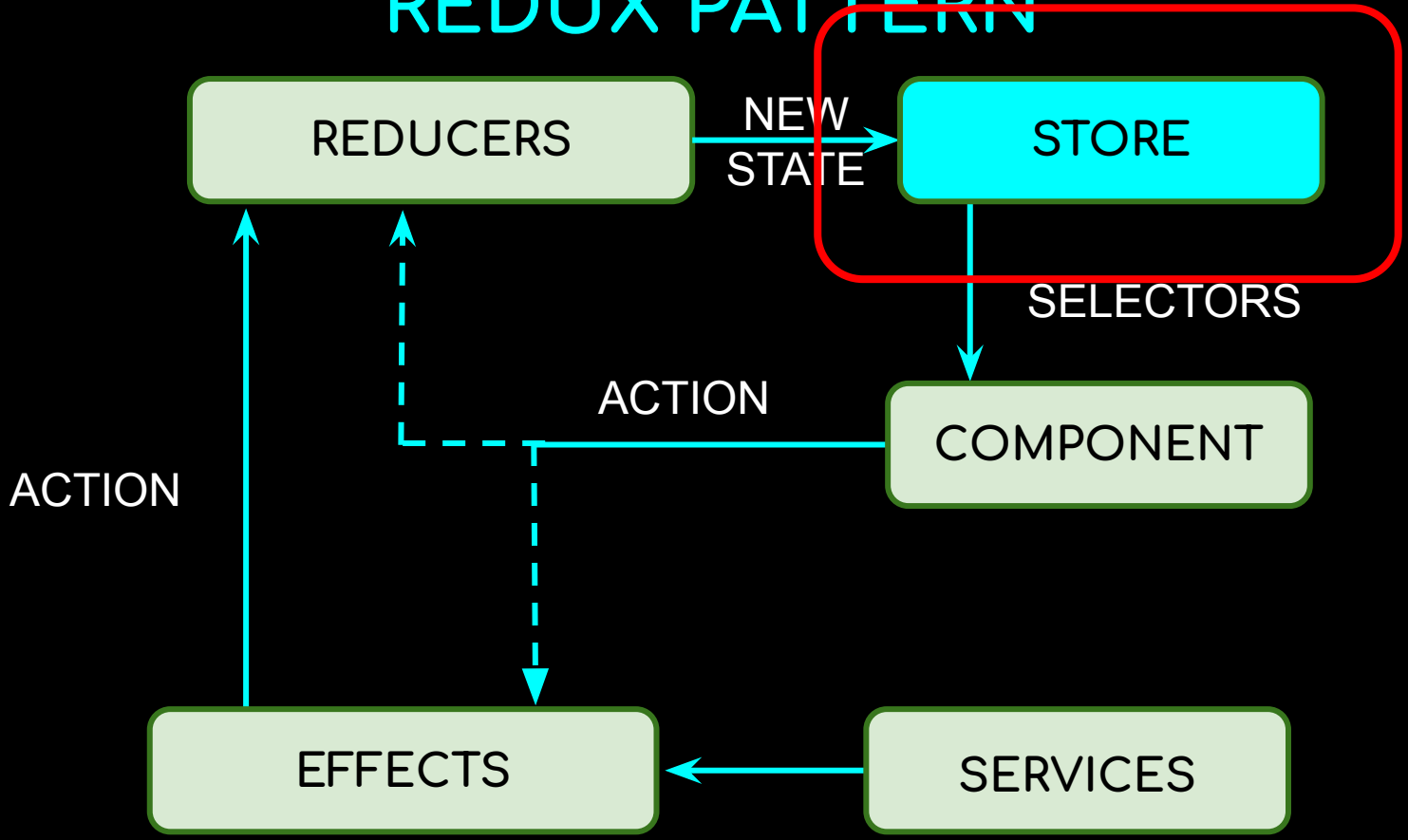
NgRx is a set of libraries developed for Angular that follow the Redux pattern.

REDUX PATTERN

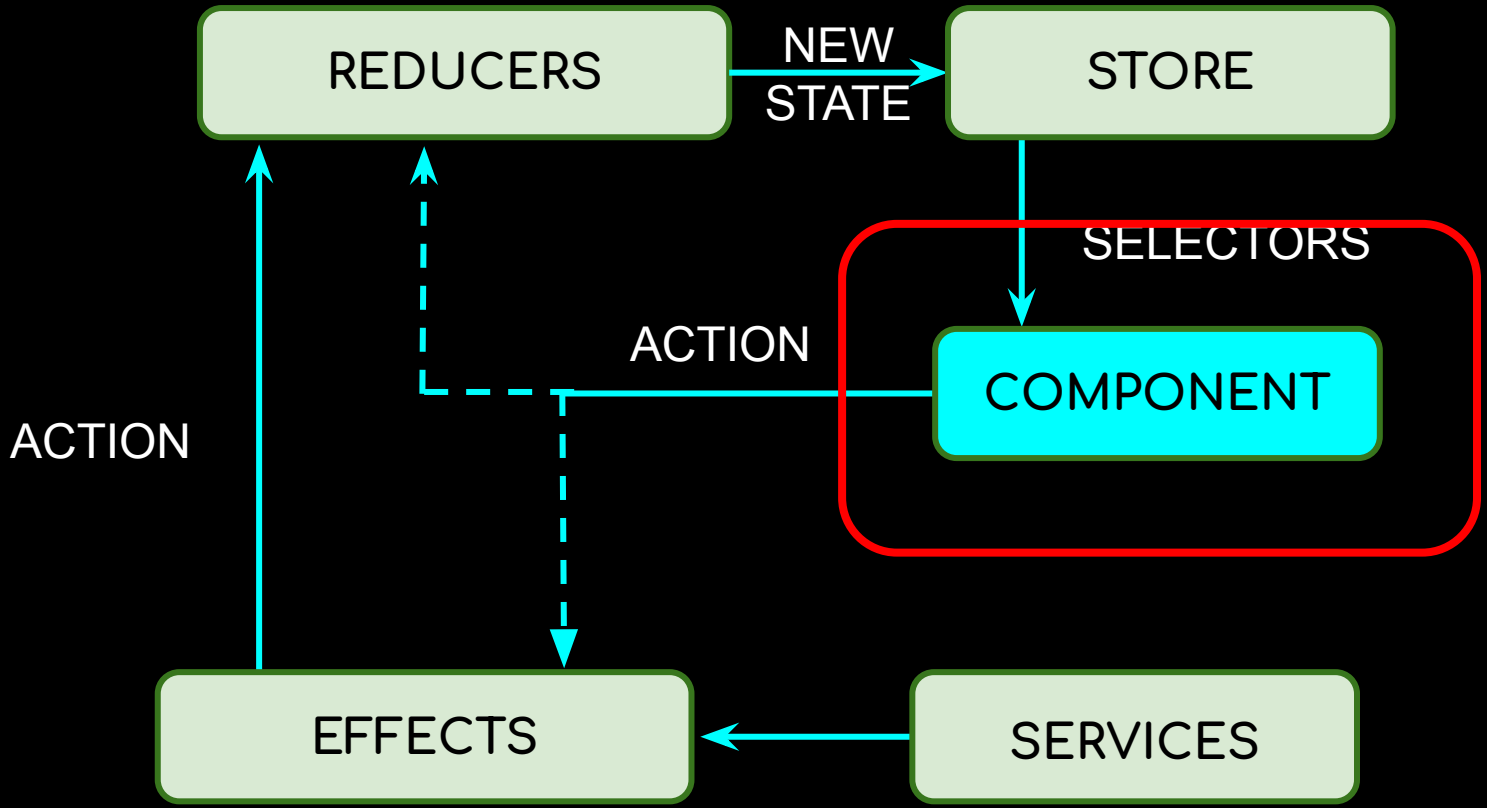



REDUX PATTERN
REDUCERS
NEW STATE
STORE
SELECTORS
ACTION
COMPONENT
ACTION
EFFECTS
SERVICES

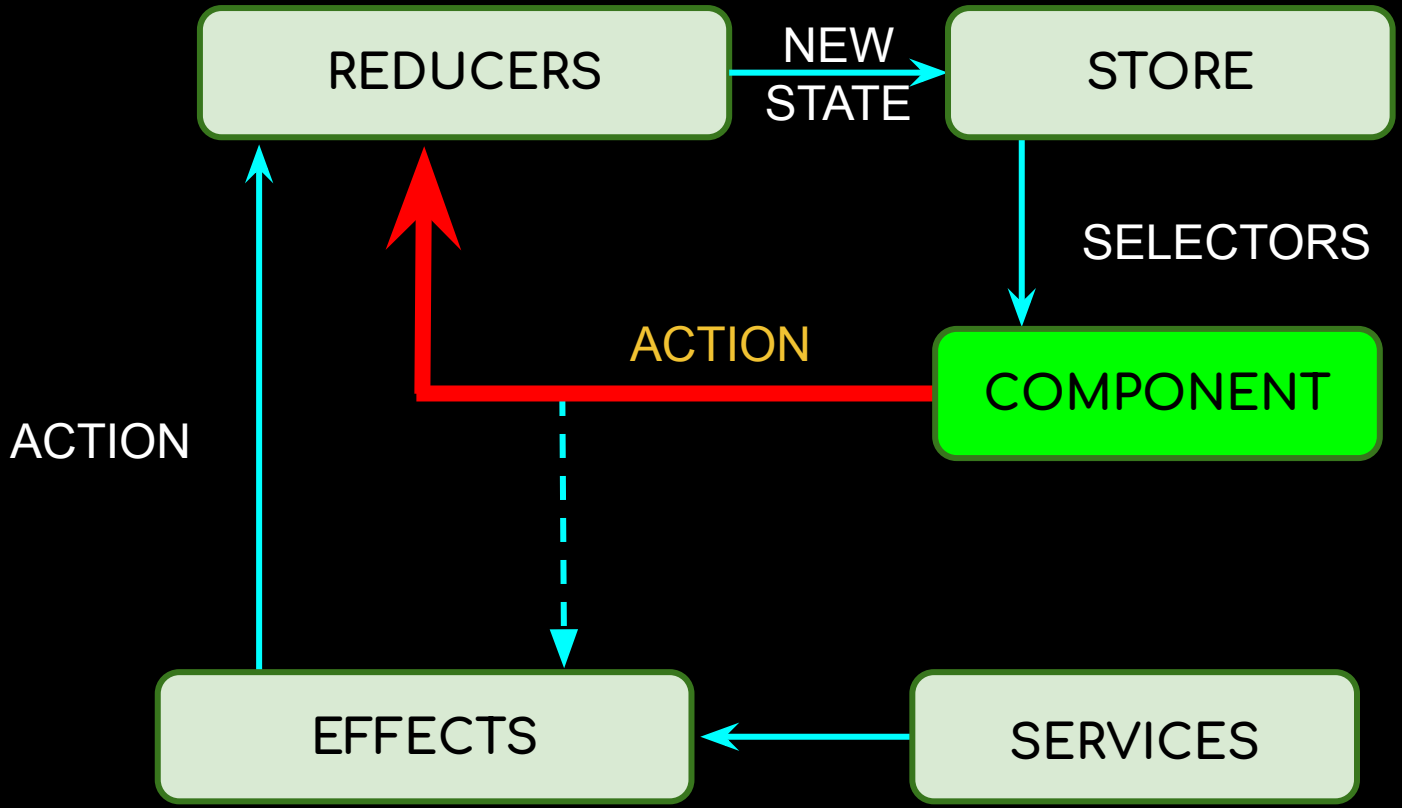
REDUX PATTERN



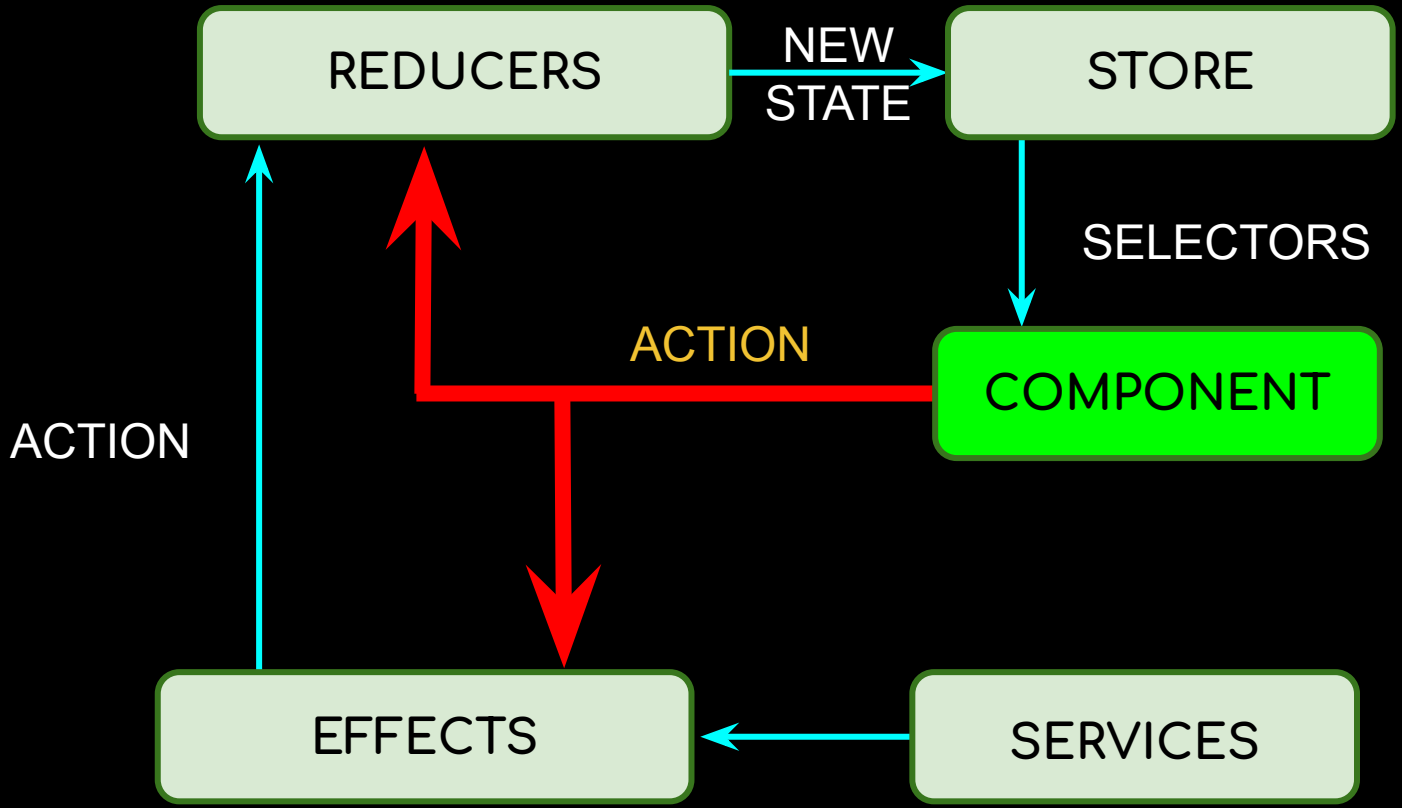
REDUX PATTERN



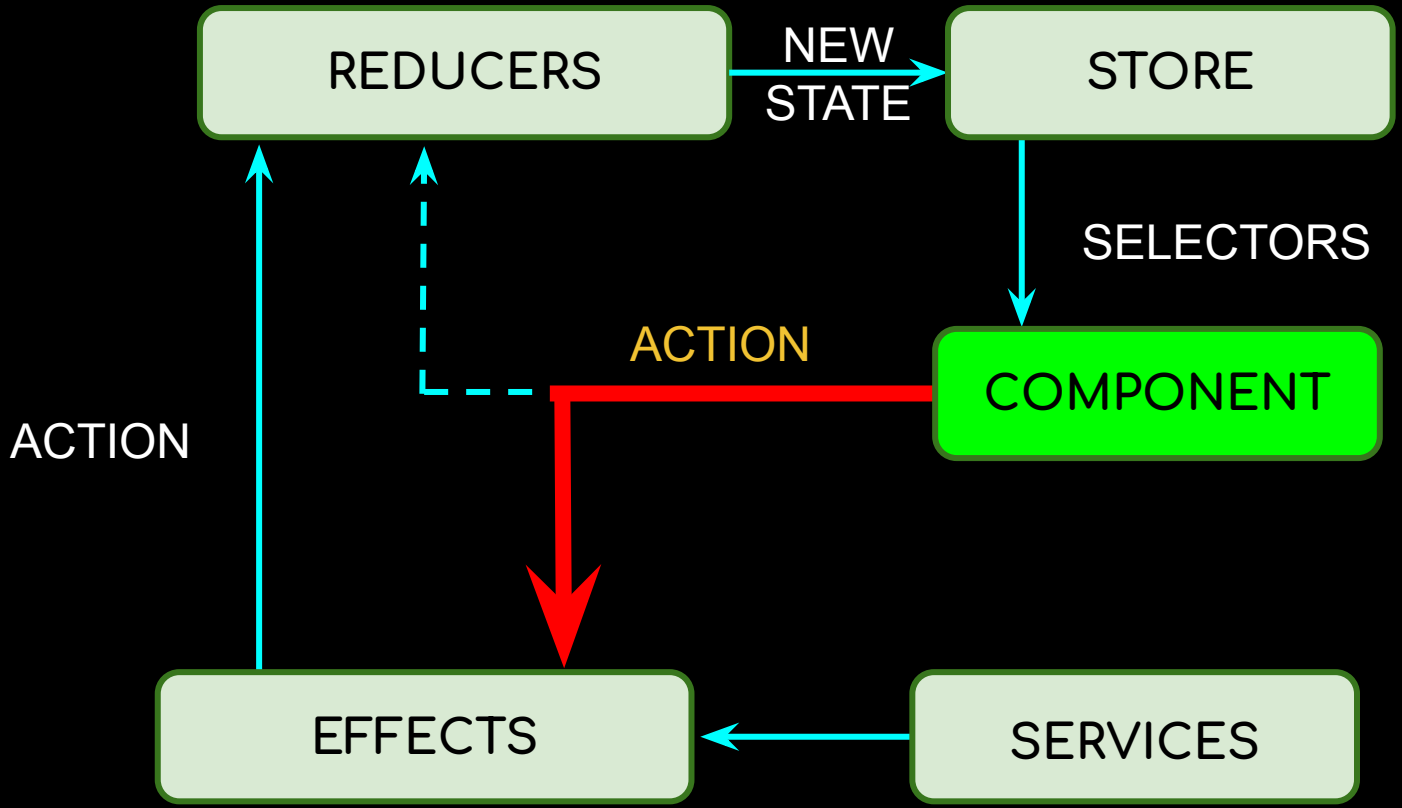
REDUX PATTERN



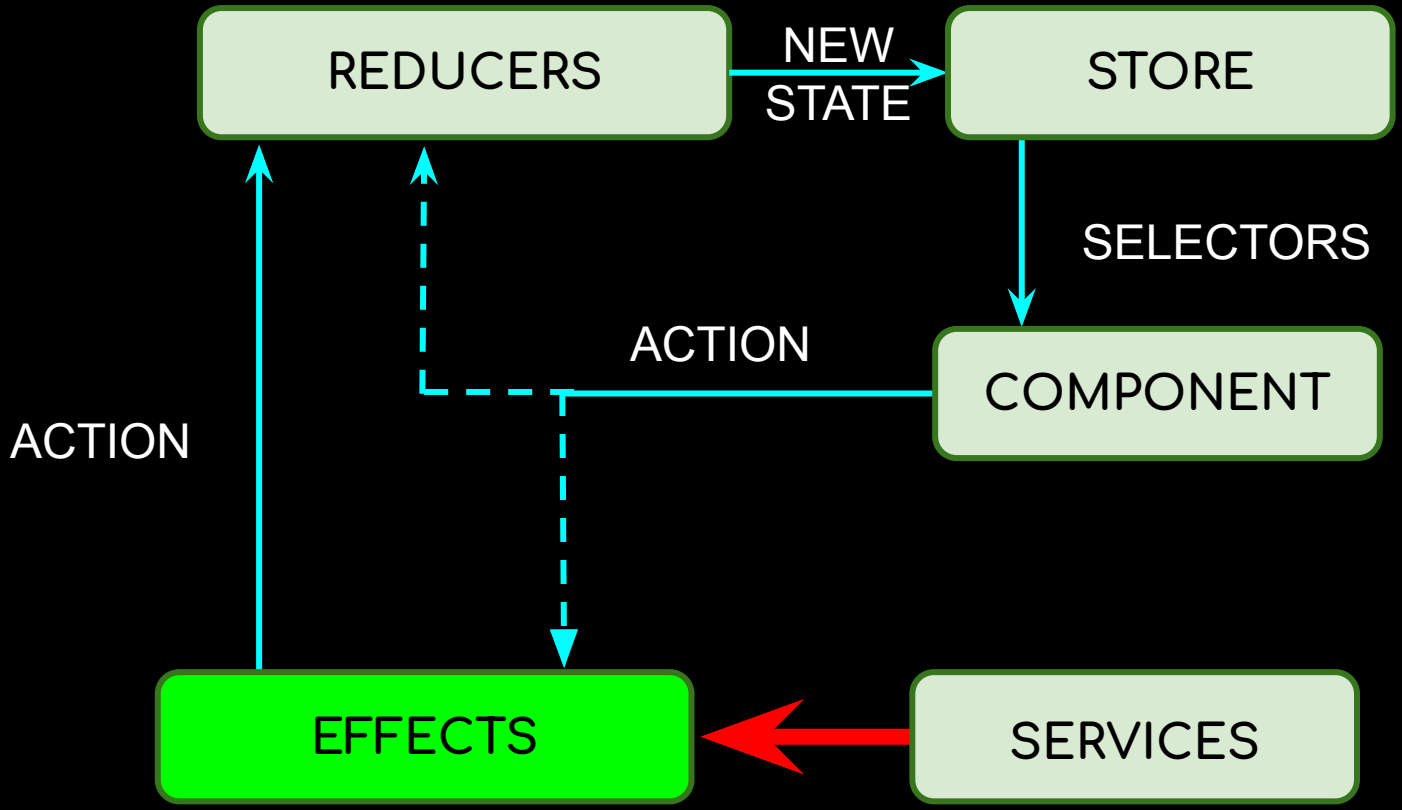
REDUX PATTERN



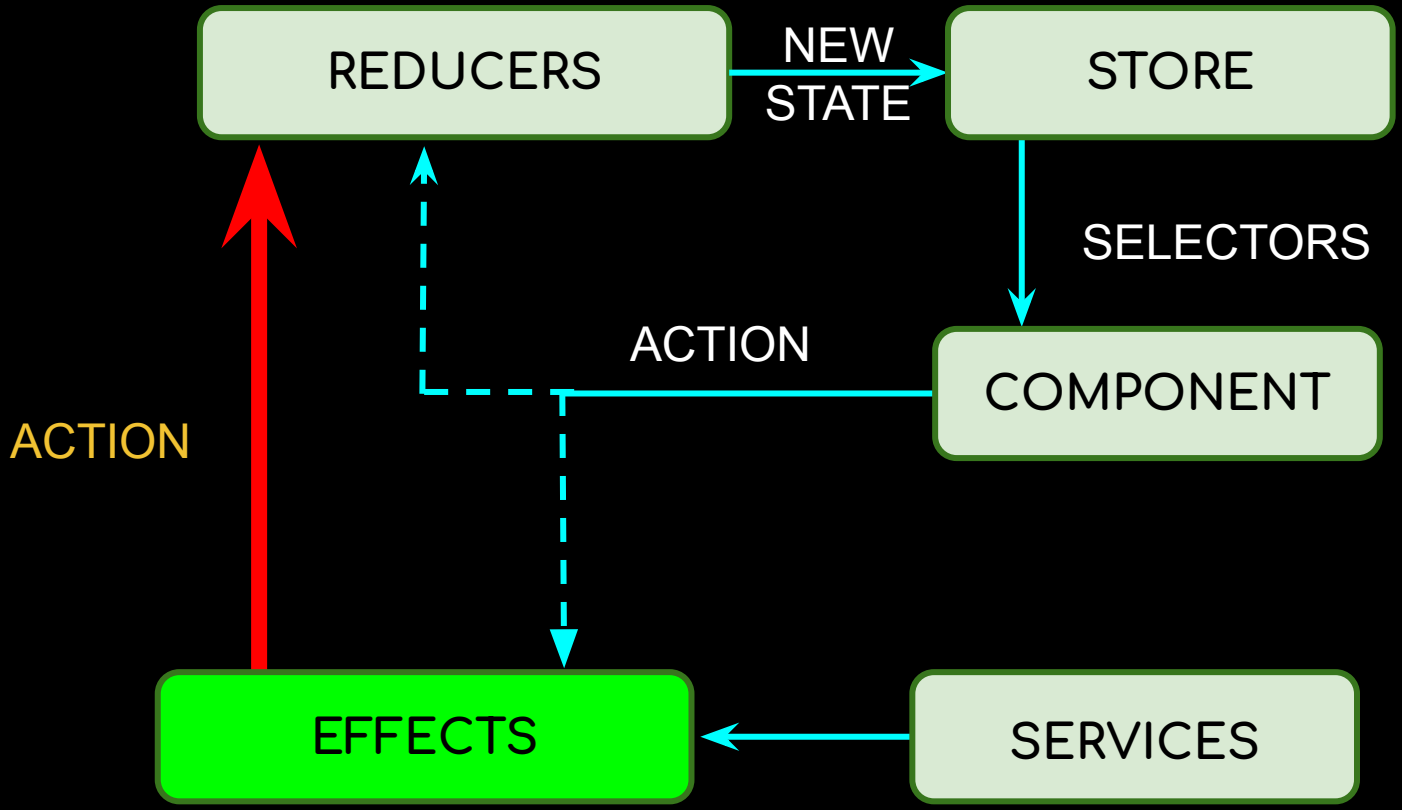
REDUX PATTERN



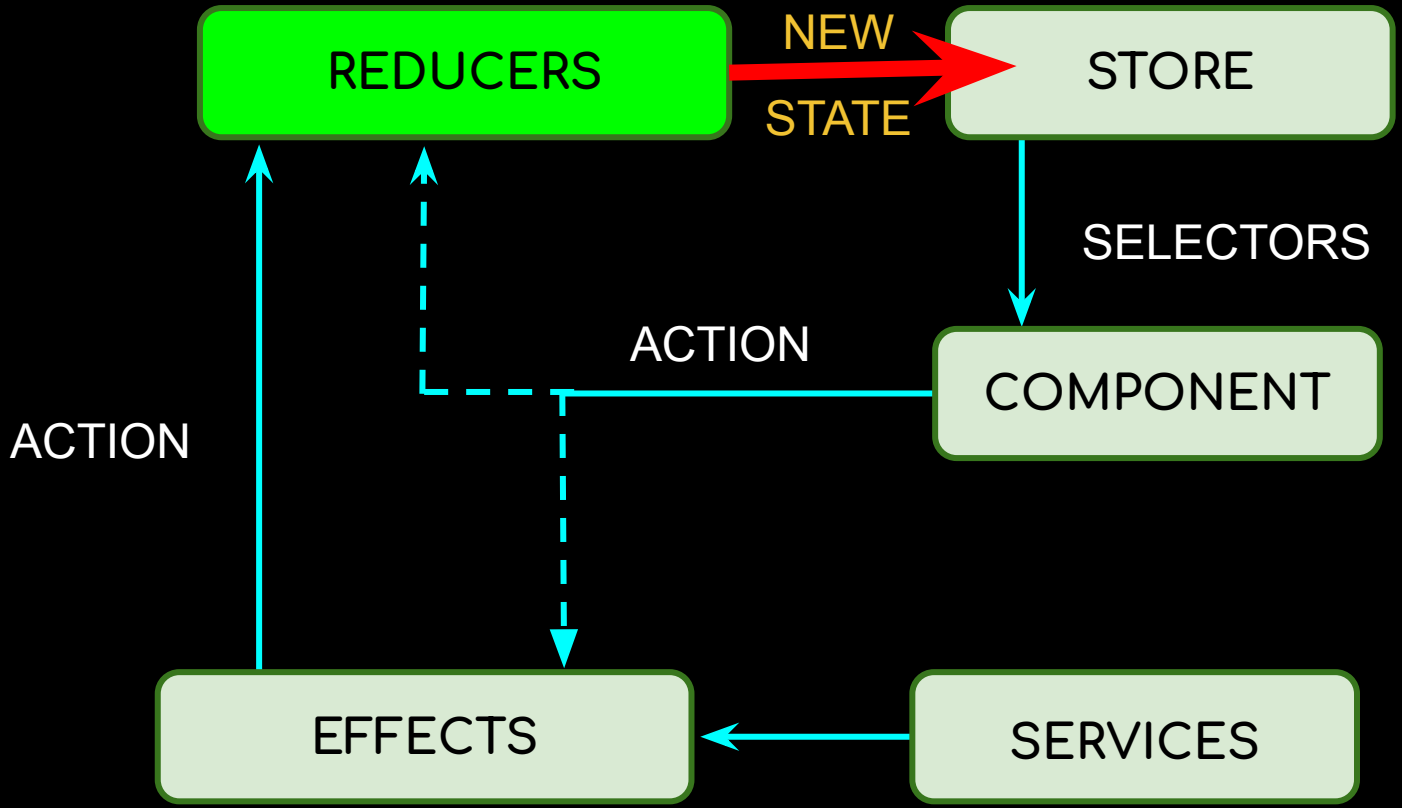
REDUX PATTERN



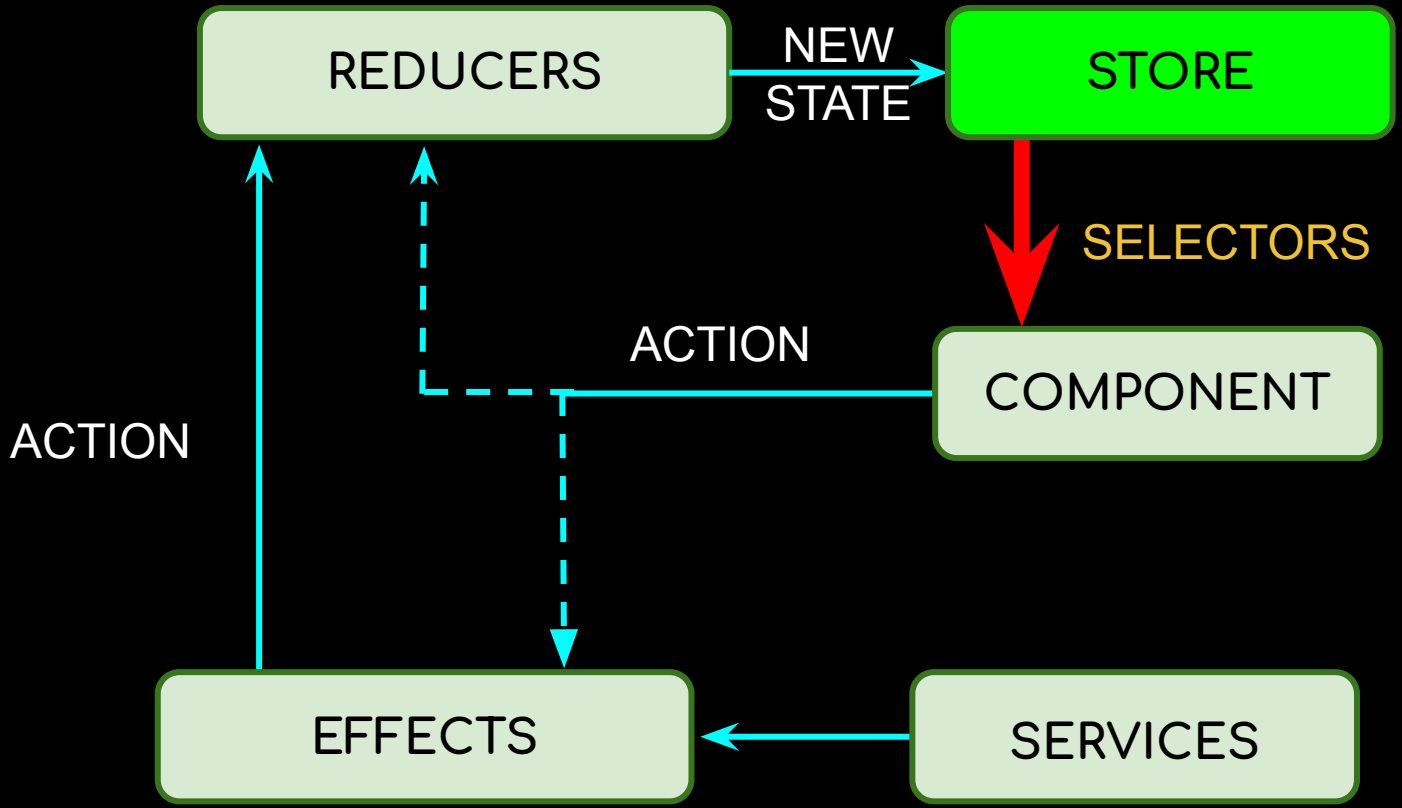
REDUX PATTERN



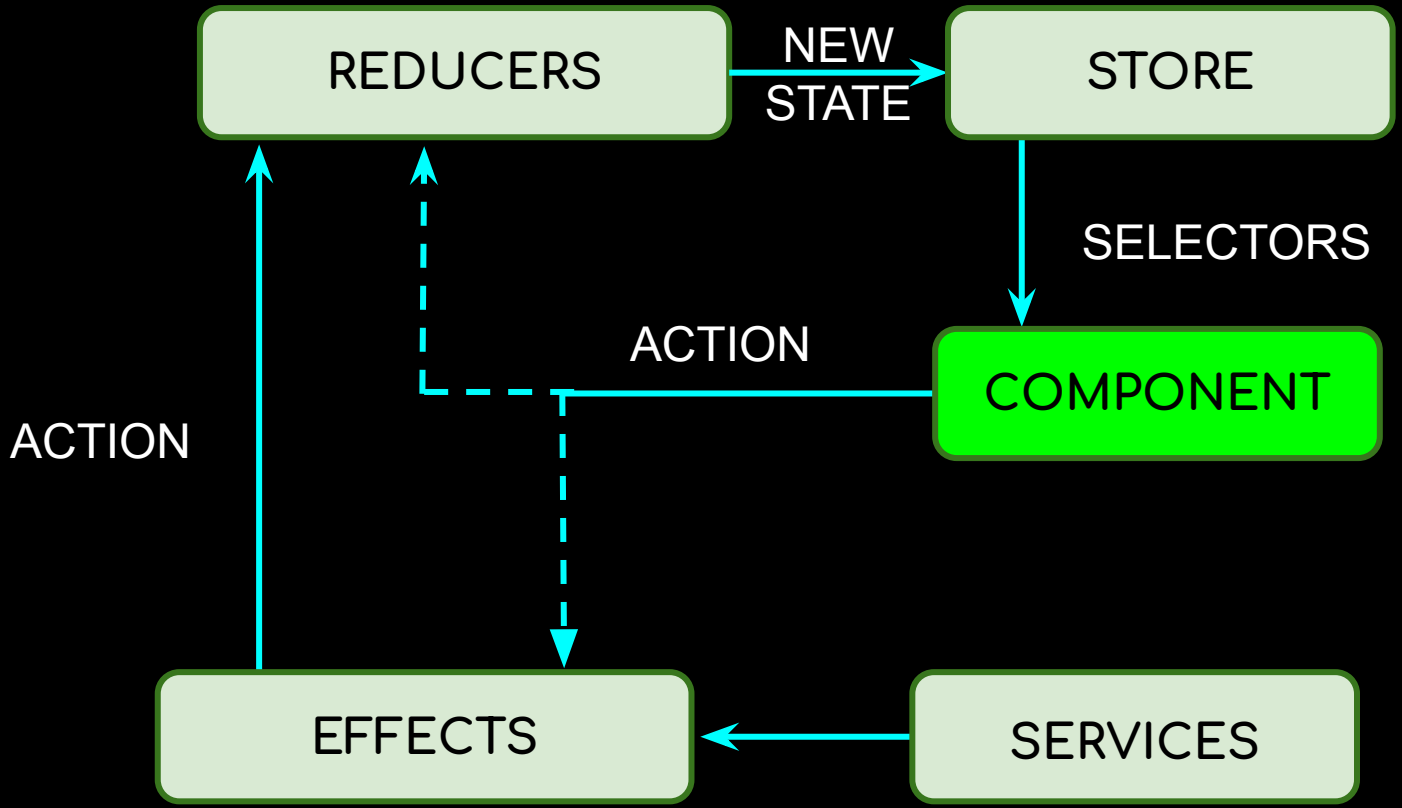
REDUX PATTERN



REDUX PATTERN



REDUX PATTERN

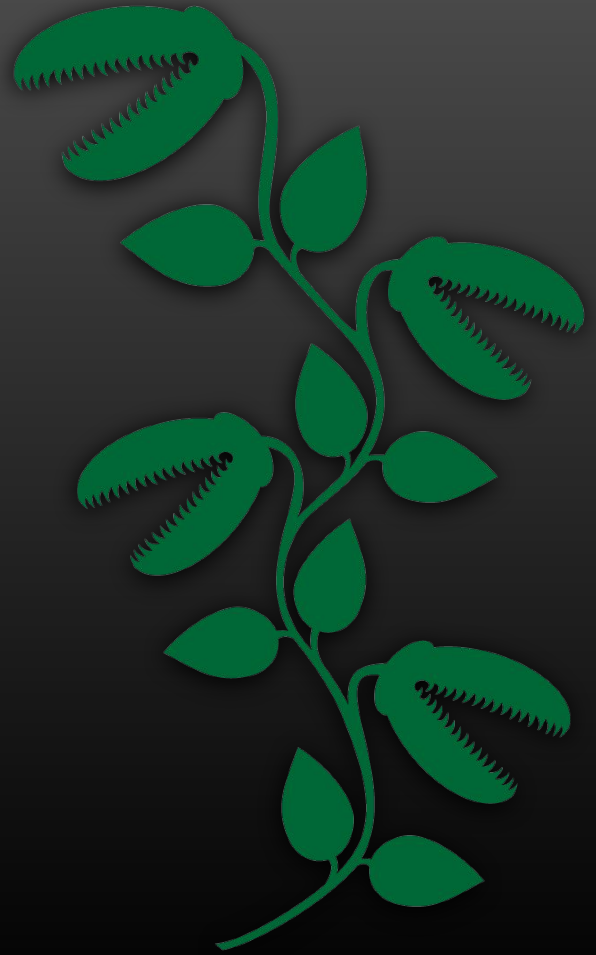


NgRx Pros

- Single Source of Truth
- Easier Debugging
- Easier Testing
- Easier Growth

NgRx Cons

- Learning Curve
- Boilerplate
- External Library



Let's Get Started Before It's Too Late!

Mushnik's House of Killer Plants

To get started run the following commands

- `$ git clone https://github.com/lnewsom/app-of-horrors.git`
- `$ cd app-of-horrors/`
- `$ npm install`
- `$ git checkout step-one`
- `$ ng serve -o`

The app should now be running in your default browser on localhost 4200

Installing NgRx with Redux DevTools

Run the following commands:

```
$ ng add @ngrx/store
$ ng add @ngrx/store-devtools
$ ng add @ngrx/router-store
$ ng add @ngrx/schematics
```

Installing NgRx with Redux DevTools

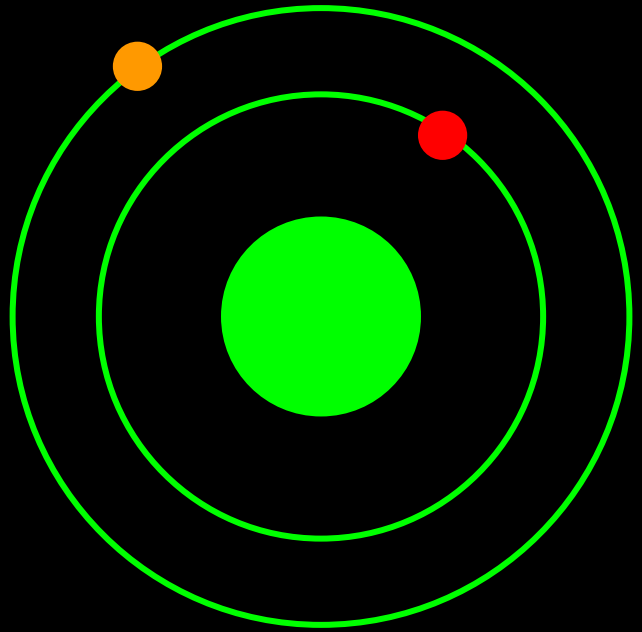
Then in the browser, search for Redux Devtools, and add the extension.

If the app is still running:

```
$ ctrl + c (to stop the app)
$ ng serve
```

This checks that the app still compiles & works as expected.

Check Out Those Devtools!



Look for this icon in the upper right of your browser.

Installing NgRx Router-Store

```
import * as fromRouter from '@ngrx/router-store';

export interface State {
    router: fromRouter.RouterReducerState<any>
}

export const reducers: ActionReducerMap<State> = {
    router: fromRouter.routerReducer
};
```

Installing NgRx Router-Store

```
export const selectRouter = createFeatureSelector<
  State,
  fromRouter.RouterReducerState<any>
>('router');

const {
  selectQueryParams,    // select the current route query params
  selectQueryParam,     // factory function to select a query param
  selectRouteParams,    // select the current route params
  selectRouteParam,     // factory function to select a route param
  selectRouteData,      // select the current route data
  selectUrl,            // select the current url
} = fromRouter.getSelectors(selectRouter);
```

Installing NgRx Router-Store

```
export const getPlantType =
  createSelector(selectRouteParam('plantType'), (plantType) => plantType);
```

```
export class SelectedPlantComponent implements OnInit {
    public plantType$: Observable<string>;

    public constructor(
        private store: Store<fromRoot.State>
    ) {
    }

    public ngOnInit(): void {
        this.plantType$ = this.store.pipe(
            select(fromRoot.getPlantType)
        );
```

```
*ngIf="plantType$ | async as plantType">
```

Adding the User Slice of State

We will start from the `version-two` branch

- `ng g a reducers/user-state/user-state -c -a=false`
- `ng g r reducers/user-state/user-state -c -a=false`
- `index.ts`

user-state.actions.ts

```
import { createAction, props } from '@ngrx/store';

import { User } from '../../models/user';

export const setUser: any =
    createAction('[User State] Set User Data', props<{ user: User }>());
```

user-state.reducer.ts

```ts
export const userStateFeatureKey = 'userState';

export interface State {
  user: User;
}

export const initialState: State = {
  user: undefined
};
```

user-state.reducer.ts

```
const userStateReducer = createReducer(
  initialState,
  on(setUser, (state, action)=> ({ ...state, user: action.user }))
);

export function reducer(state: State | undefined, action: Action) {
  return userStateReducer(state, action);
}
```

reducers/user-state/index.ts

```
export * from './user-state.actions';
export * from './user-state.reducer';
```

reducers/index.ts

```
import * as userState from './user-state';

export interface State {
    router: fromRouter.RouterReducerState<any>,
    userState: userState.UserState
}

export const initialState: State = {
    router: null,
    userState: userState.initialState
}
```

reducers/index.ts

```
export const reducers: ActionReducerMap<State> = {
    router: fromRouter.routerReducer,
    userState: userState.reducer
};

export const selectUserState: any =
  createFeatureSelector<userState.UserState>(userState.featureKey);

export const selectUser: any =
  createSelector(selectUserState, (state: userState.UserState) => state.user);
```

```
this.store.dispatch(userState.setUser({ user }))
```

```
this.user$ = this.store.pipe(
    select(fromRoot.selectUser)
);
```

Composed Selectors

```
export const loadPlantListingTable: any =
  createSelector(
    getPlantType,
    selectUser,
    (plantType, user) => ({ plantType, user }
  )
);
```

It’s like riding a bike...

Installing NgRx Effects

Run the following command:

```
$ ng add @ngrx/effects
```

You can find the docs for ngrx/router-store at

https://ngrx.io/guide/effects

```
export class AppEffects {
  @Effect()
  public getPlantListings$ = this.actions$.pipe(
      ofType(getPlantListings),
      mergeMap(() => {
        return zip(
          this.restService.getPlantListings(),
          this.restService.getPlantQuantities()
        )
        .pipe(map((response) => {
          const listings: PlantListing[] = response[0];
          const quantities: { plantId: number, quantity: number }[] = response[1];
          return this.plantQuantityService.mapQuantities(listings, quantities);
          })
        );
      }),
      map((plantListings) => setPlantListings({ plantListings })
      )
  );
```

NOW GET OUT THERE AND

MANAGE THAT STATE!

NgRx Resources

NgRx Docs - https://ngrx.io/docs

RxJs Docs - https://rxjs.dev

source allies