Reducer

Generates a reducer file that contains a state interface, an initial state object for the reducer, and a reducer function.

Command

```
ng generate reducer ReducerName [options]
```

Options

Nest the reducer file within a folder based on the reducer `name`. ^P • `--flat`

Provide the path to a file containing an `Angular Module` and the entity reducer will be added to its `imports` array using `StoreModule.forFeature`.

- `--module`

Provide the path to a `reducers` file containing a state interface and an object map of action reducers. The generated reducer interface will be imported and added to the first defined interface within the file. The reducer will be imported and added to the first defined object with an `ActionReducerMap` type.

- `--reducers`

Generate a spec file alongside the reducer file.

- `--spec`

Examples

Generate a `User` reducer file add it to a defined map of reducers generated from a feature state.

```
ng generate reducer User --reducers reducers/index.ts
```

Generate a `User` reducer file within a nested folder based on the reducer name.

```
ng generate reducer User --flat false
```

Generate a `User` reducer and register it within the `Angular Module` in `app.module.ts`.

```
ng generate reducer User --module app.module.ts
```

Generate a `User` reducer file within a nested `reducers` folder.

```
ng generate reducer User --group
```

Store

Generates the initial setup for state management and registering new feature states. It registers the `@ngrx/store-devtools` integration and generates a state management file containing the state interface, the object map of action reducers and any associated meta-reducers.

```
ng generate store State [options]
```

Options

Provide the path to a file containing an `Angular Module` and the feature state will be added to its `imports` array using `StoreModule.forFeature` or `StoreModule.forRoot`.

- `--module`
  - Alias: `-m`
  - Type: `string`

When used with the `--module` option, it registers the state within the `Angular Module` using `StoreModule.forRoot`. The `--root` option should only be used to setup the global `@ngrx/store` providers.

- `--root`
  - Type: `boolean`
  - Default: `false`

Provide the folder where the state files will be created.

- `--statePath`
  - Type: `string`
  - Default: `reducers`

Provide the name of the interface exported for your state. When defining with the `--root` option, the name of the store will be used to define the interface name.

- `--stateInterface`
  - Type: `string`
  - Default: `State`

Examples

Generate the initial state management files and register it within the `app.module.ts`.

```
ng generate store State --root --module app.module.ts
```

Generate an `Admin` feature state within the `admin` folder and register it with the `admin.module.ts` in the same folder.

```
ng generate module admin --flat false
ng generate store admin/Admin -m admin.module.ts
```

Generate the initial state management files within a `store` folder and register it within the `app.module.ts`.

```
ng generate store State --root --statePath store --module app.module.ts
```