NgRx State Management with Angular

Presented by Shane Jordan

THANKS TO ALL OUR SPONSORS!

BizStream™

THE C2 GROUP

<aptera>

KL&A
KUNZ, LEIGH AND ASSOCIATES

SpinDance

OST | OpenDigital

ATOMIC OBJECT

KFORCE®

Progress®

MAGIC WRIGHTER INC.

{M} MICHIGAN SOFTWARE LABS

Collective Idea

TEKsystems®

SALESPAD™

Microsoft

What we're going to do today

- General concepts
- Talk about NgRx
- Demo some code

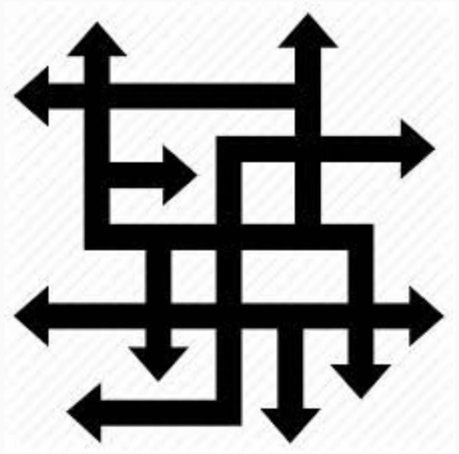
Handling State before Redux?

We passed our state around in services

Maybe even localstorage, session storage, cookies, etc. (there is still many good use cases for these)

This got complicated in large applications

This got complicated in large applications



What is Redux and NgRx?

Reactive libraries for Angular

NgRx is Redux implementation for Angular

Manages state for your application

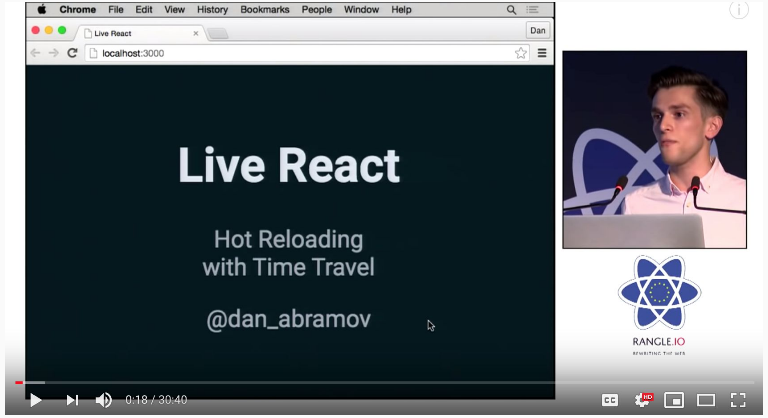
One way data flow

Redux History

Started with React and Facebook releasing Flux

Dan Abramov & Andrew Clark in 2015 created Redux, that was inspired by the Elm framework and its immutable concepts

https://www.youtube.com/watch?v=xsSnOQynTHs



Why use NgRx?

Centralized State that you can see in once place

Elegant pattern

Combats Complexity

Data Cache

Predictable Patterns (you know that only reducers are changing state)

First Some Concepts...

PURE FUNCTIONS

Deterministic - one goes in, always comes out. Time is an example of something NOT deterministic (it changes)

Consistent input and output with no side effects

IMPURE FUNCTIONS

Impure Functions exist to (think AJAX calls)...we will handle these later.

They have Side Effects. What's a side effect?

IMMUTABILITY

Immutability is a very hot pattern right now in order to counter the ever increasing complexity of our applications and even infrastructure

NO MUTATIONS OF STATE, you destroy and build again

Immutability helps combat complexity

Smart and Dumb Components

Smart Components = Containers

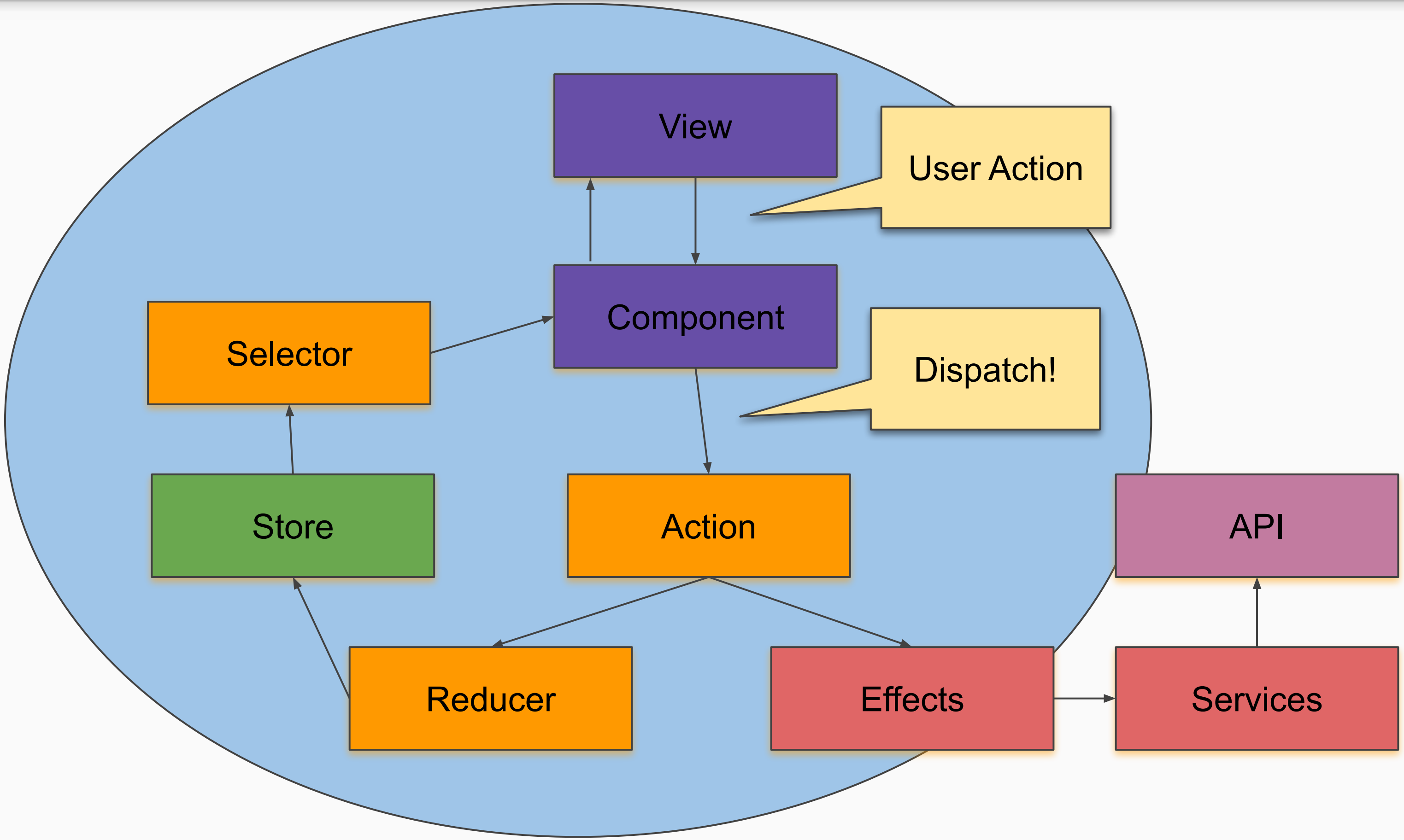
Contains all of your API calls, etc.

Dumb Components = Presentation Components

Only concerned with UX, not concerned with API calls, etc.

BUILDING BLOCKS

Redux Pattern with Effects



STORE

In-Memory “Property bag”. This of it as a big client side object database in memory.

Stores all of your application State

Subscribe to updates from store

Dispatch events to store

Organize by Feature Module for each Feature State (or slice)

Single source of truth.

Just JSON structure

ACTIONS

Dispatch Action to change store/state

Type and a Payload, simple JS object with typically two properties: Type and payload

Immutable

Actions are payloads of information that send data from your application to your store. They are the *only*source of information for the store. You send them to the store using store.dispatch().

REDUCERS

Reducers specify how the application's state changes in response to actions sent to the store. Remember that actions only describe what happened, but don't describe how the application's state changes.

Handle Transitions from One State to another

Nothing else should alter the store except Reducers!

MetaReducers are executed BEFORE normal reducers. Think of it like middleware. Great for logging.

Only Reducers can change state

Immutable

SELECTORS

These are how you select state from the store

Pure Functions

Slices of State

```
this.store.select(state
=> state.pizzas);
```

FEATURE MODULE STATE COMPOSITION

IN APP MODULE

Root Reducer - in the root of the app.module, the core reducer for your app

Meta Reducer - for cross cutting concerns like logging

IN FEATURE MODULES

Feature Reducer - go in each Feature Module and is only concerned with that Feature state

EFFECTS

Async processing

Impure Functions (hence the name side effect)

Like Services, it's how we call APIs

Always handle three states Request, Success, Failure

Anything Async, most of time API calls

1. Listen for an action
2. Do something (like call an API)
3. Dispatch new action

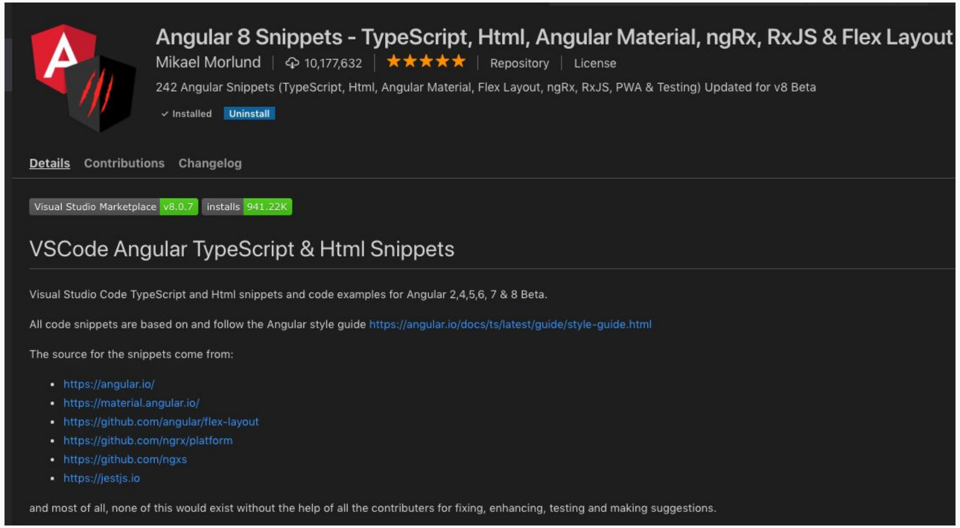
TOOLING

NgRx Snippets in VSCode

Installed Angular 8 Snippets in VSCode

Uses official Angular style guide

Include ngrx and a lot of other snippets



Angular 8 Snippets - TypeScript, Html, Angular Material, ngRx, RxJS & Flex Layout

Mikael Morlund | 10,177,632 | ★★★★★ | Repository | License

242 Angular Snippets (TypeScript, Html, Angular Material, Flex Layout, ngRx, RxJS, PWA & Testing) Updated for v8 Beta

VSCode Angular TypeScript & Html Snippets

Visual Studio Code TypeScript and Html snippets and code examples for Angular 2,4,5,6, 7 & 8 Beta.

All code snippets are based on and follow the Angular style guide https://angular.io/docs/ts/latest/guide/style-guide.html

The source for the snippets come from:

- https://angular.io/
- https://material.angular.io/
- https://github.com/angular/flex-layout
- https://github.com/ngrx/platform
- https://github.com/ngxs
- https://jestjs.io

and most of all, none of this would exist without the help of all the contributers for fixing, enhancing, testing and making suggestions.

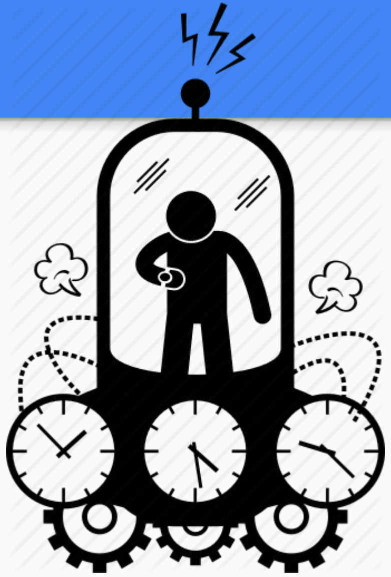
NgRx Chrome Dev Extension

You know it's Redux pattern since it uses the same NgRx Dev tool

Very helpful state change and debugging tool - you will live in this while developing

Time Traveling

https://chrome.google.com/webstore/detail/redux-devtools/lmhkpmbekcpmknklioeibfkpmmfibljd?hl=en



Angular CLI Generate

Generate consistent code quickly through the command line

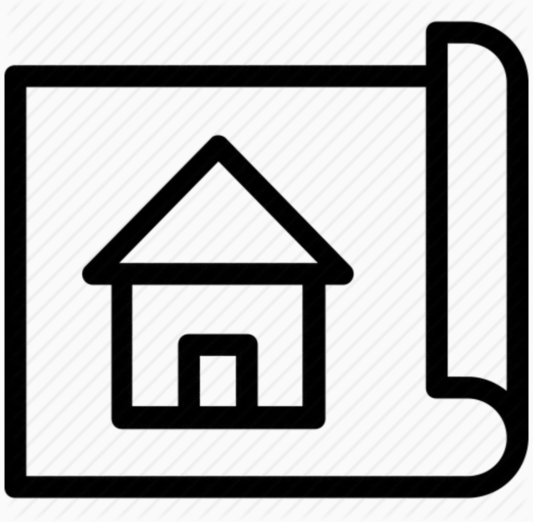
npm install vs ng add

Angular CLI custom Schematics

Extensions to the built in generators

NgRx has their own set

Wires up a lot of code automatically



DEMO

PUMPKIN SPICE CEO STOUT
BIG BAD BAPTIST
BIG BAD BAPTISTA
DOUBLE BARREL BIG BAD BAPTIST
Lizard of Koz
KBS
Black Note Stout
Pigeon Hill
Lizard Burial Ground
HIPSTER BRUNCH STOUT
RYE HIPSTER
BOURBON BARREL IMPERIAL MAYAN
HAZEL'S NUT

How I want this live coding demo to go

There is a good chance it might go like this. :)

Types of Maps in Effects and when to use them

switchMap (fastest)

GETs and read onlys (cancel previous request)

concatMap (safer)

Updates, let's all the calls finish

mergeMap (parallel requests)

HTTP requests in parallel. Doesn't guarantee order

exhaustMap (sequential waits)

Waits for initial calls to complete. Good for logins.

Memory Leaks - async pipe and unsubscribe

Unsubscribe

Used in component.ts

Manually controlling unsubscribe on ngDestroy. Must set and maintain a variable

.takeWhile(() => isComponentActive))

Async Pipes (automatic)

Used in templates

Automatically subscribes and unsubscribes for you!

Can’t use them in the component.ts

***ngFor=”let product of product$ | async”**

REASONS NOT TO USE NgRx

Too much boilerplate

If your app is small to medium in size, this might not be worth the complexity

If your team is not as experienced

Note: There are other ways to reduce boilerplate, such as NgRx Entities or 3rd party package NgRx Data.

Or other solutions such as NgXs, Mobx, Akita...

Should you use NgRx in your project?

Project Size and Complexity

Small - Probably Not

Medium - Maybe

Large “Enterprise-y”: Yes

Team skill set is also a factor, if the team is new to Angular, maybe not

DISPATCH ANY QUESTIONS

THANKS TO ALL OUR SPONSORS!

BizStream™

THE C2 GROUP

<aptera>

KL&A
KUNZ, LEIGH AND ASSOCIATES

KFORCE®

SpinDance

OST | OpenDigital

ATOMIC OBJECT

TEKsystems®

Progress®

MAGIC WRIGHTER INC.

SALESPAD™

Microsoft

{M} MICHIGAN SOFTWARE LABS

Collective Idea