Angular Lisbon Meetup 8.11.2017

Rapid App Development with Angular and LoopBack

João Ribeiro, joao@altar.io
Peter Bouda, peter.bouda@apiax.com

About João

- Add description

About Peter

- Web Developer since the 90s
- LoopBack + Angular since 2015
- Senior Web Architect at Apiax
- Angular trainer at ng-lisbon.com
- The rest is LEGO
- https://www.peterbouda.eu/

LoopBack Intro

- Quickly cerate REST APIs
- Based on Express.js
- Model-driven development
- IBM bought StrongLoop in 2015 and integrates LoopBack in their cloud offer (IBM API Connect)
- Comes with Android, iOS and Angular(JS) SDKs
- http://loopback.io

LoopBack SDK Builder Intro

- A fork from the official AngularJS SDK to support Angular 2
- Generates front-end code to acccess back-end endpoints

```
this.accountApi.patchAttributes(userId, { email: newEmail });
```

- Real-time communication via FireLoop (think FireBase but with your own stack)
- ngrx and ORM support
- https://github.com/mean-expert-official/loopback-sdk-builder

Get started with LoopBack CLI

Generate LoopBack app:

```
$ npm install -g loopback-cli
$ lb
? What's the name of your application? the-next-big-thing
? Enter name of the directory to contain the project: the-next-big-thing

? Which version of LoopBack would you like to use? 3.x (current)
? What kind of application do you have in mind?
                   api-server (A LoopBack API server with local User auth)

I'm all done. Running npm install for you to install the required dependencies.
If this fails, try running the command yourself.
```

Windows preparation

- Two steps before building loopack-cli
- Install windows-build-tools

```
npm install --global --production windows-build-tools
```

- Install OpenSSL version 1.0.2L: https://slproweb.com/products/Win32OpenSSL.html

Setup database

```
$ lb datasource
? Enter the data-source name: postgres
? Select the connector for postgres: PostgreSQL (supported by StrongLoop)
[...]
```

- Support for different database types as MySQL, PostgreSQL, MongoDB, etc.
- Installs a LoopBack Connector module
- Configure via JSON files (datasources.json)
- Mail is "datasource", too

Generate models

```
$ lb model
```

- Models consist of JSON description and optional JS
- Make a model public and you will get all the REST endpoints
- Fined-gained control to disable and restrict endpoints
- Access Control Lists to allow access to user roles
- Define model relations
- Custom remote methods
- LoopBack API Explorer to exercise all the generated API endpoints

Generate Angular SDK

```
$ npm install --save @mean-expert/loopback-sdk-builder
$ ./node_modules/.bin/lb-sdk server/server.js client/src/app/shared/sdk
```

- NativeScript support
- Real-time communication support
- React support :P
- Generates front-end code for:
  - Models
  - Services
  - Local storage (for auth)

Access Control, Users and Roles

```
{
  "accessType": "WRITE",
  "principalType": "ROLE",
  "principalId": "$owner",
  "permission": "ALLOW"
}
```

- LoopBack comes with a User model and supports authentication
- Authentication is also available in Angular SDK (`LoopBackAuth` and `UserApi` services)
- Built-in dynamic roles: $everyone, $authenticated, $unauthenticated, $owner
- You can specify static user roles (e.g. 'admin') and dynamic role handlers (e.g. 'teamMember')
- Access to REST endpoints and remote methods is controled via ACLs

Related models and queries

```
this.orderApi.findById(id, { include: [ { relation: 'items',
  scope: { order: 'id DESC' } } ] }).subscribe([...]);
```

- Create model relations with hasMany, hasOne, belongsTo, HasAndBelongsToMany, ...
- LoopBack will publish REST endpoints for those (`GET /<model1-name>/<instanceID>/<model2-name>`)
- LoopBack adds helper methods to model class's prototype (`order.items(function(err, items) { [...] })`)
- Powerful `include` filter supports nested queries (compare GraphQL)

The Angular+LoopBack Seed

- Basic full-stack app with User registration, login and profile page
- Extend to your needs!
- https://github.com/ng-lisbon/angular-loopback-seed

Angular SDK + NGRX (Redux)

```
lb-sdk ... -l angular2 -d ng2web -n [ngrx|orm]
```

- Generates:
  - Actions
  - Effects
  - Reducers
  - Guards and Resolvers
  - ORM*
  - Plugable State
- only for orm flag

Extendable NGRX

- Actions: **Action**, **ActionSuccess**, **ActionFail**. With `{meta: any}`.
- Effects consume **Action** and dispatch **ActionSuccess** or **ActionFail**
- Reducers consume **ActionSuccess**

You can use any to create you own functionality with custom Effects and Reducers

```
@Effect()
public signupSuccess$ = this.actions$
  .ofType(UserActionTypes.SIGNUP_SUCCESS)
  .map((action: LoopbackAction) => new UserActions.login({
    email: action.payload.credentials.email,
    password: action.payload.credentials.password
  }, ['user']))
```

- You can pass any **meta tag** to actions for extra functionality.
- Meta tags will be pass on between actions

Local database representation with ORM

- One **Reducer** per **Model**
- Relation's data resolved to it's **Reducer**
- **ORM** query aggregates data from local **Store**

```
this.orm.Room.find({ include: ['messages'] })
```

```
[{
  id: 1,
  name: 'ng-lisbon',
  messages: [{ id: 1, text: 'Hello NG-Lisbon' }]
}]
```

```
rooms: { ids: [...], entities: {...} },
messages: { ids: [...], entities: {...} }
```

ORM special meta tags

- `{ io: true }`
  - tells **ORM** to **Sync** the query with the server using Fireloop Real-Time API
- `{ justCache: true }`
  - uses only local **State** bypassing the query to the backend
- `{ resetStore: true }` (comming soon)
  - resets the **Store** before applying new **State**

ORM by example

```
this.rooms$ = this.orm.Room.find({
  where: {
    name: { like: 'ng-lisbon' }
  },
  order: 'id DESC',
  limit: 10,
  include: ['categories', 'accounts', 'messages', 'likes']
}, {
  io: true
})
```

- io meta tag tells ORM to sync the query with the server using Fireloop Real-Time API

Thanks!

Any questions?

João Ribeiro, joao@altar.io
Peter Bouda, peter.bouda@apiax.com