# How to build an Angular library


ANGULAR
TRAINING

@AlainChautard - angulartraining.com

# About me - Alain Chautard (or just Al)

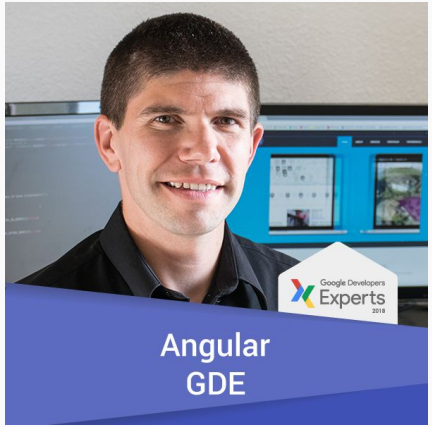
Google Developer Expert in Web technologies / Angular

Java developer since 2006

Angular JS addict since 2011

Organizer of the Sacramento Angular Meetup group

Web consultant (60%) / trainer (40% of the time) @ angulartraining.com



# Why would you create your own library?

- To promote reuse of your Angular code between projects
- To publish your code on NPM so it becomes open-source

- In the end, it's all about sharing some of your code outside of your current project

# We’re going to create a simple mask library

- One simple pipe to hide sensitive data (like SSN, account number)

```
it('should format 123456 to ******3456', () => {
    const pipe = new MaskPipe();
    expect(pipe.transform('123456')).toBe('******3456');
});
```

# How to generate a library?

- As always with Angular CLI, everything starts with a simple command:

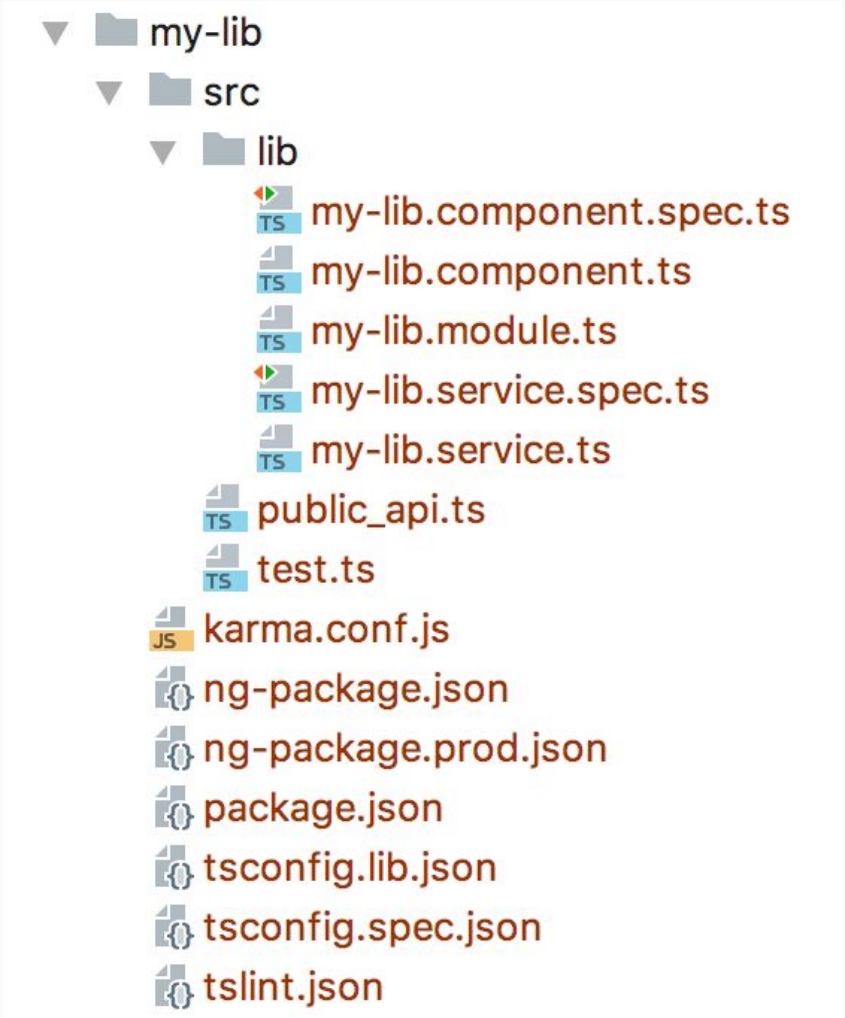
**ng generate library my-lib**

- The above command is going to generate a **`projects/my-lib`** folder structure at the root of your current project

# Library structure

In **`projects/src/lib`**, you'll find the source code of your library

Note that Angular CLI also generates a Karma config file as well as **`tsconfig`** and **`tslint`** for your library.



# How to add code to your library?

- Using Angular CLI:

**ng generate component myComponent --project=my-lib**

```
[ng2-demo]$ ng g c myComponent --project=my-lib
Two or more projects are using identical roots. Unable to determine project using
rrent working directory. Using default workspace project instead.
CREATE projects/my-lib/src/lib/my-component/my-component.component.css (0 bytes)
CREATE projects/my-lib/src/lib/my-component/my-component.component.html (31 bytes)
CREATE projects/my-lib/src/lib/my-component/my-component.component.spec.ts (664 by
s)
CREATE projects/my-lib/src/lib/my-component/my-component.component.ts (292 bytes)
UPDATE projects/my-lib/src/lib/my-lib.module.ts (320 bytes)
```

# Export the public API of your library

- Two files to keep in sync:

## my-lib.module.ts

```
@NgModule({
      imports: [],
      declarations: [MyLibComponent, ...],
      exports: [MyLibComponent]
})
export class MyLibModule { }
```

## public_api.ts

```
/*
* Public API Surface of my-lib
*/

export * from './lib/my-lib.service';
export * from './lib/my-lib.component';
export * from './lib/my-lib.module';
```

# How to build your library?

- A simple command allows you to create a build:

**ng build my-lib**

- The output of the build is going to go to **`dist/my-lib`** folder structure at the root of your current project
- Also supports the **`--prod`** flag for production builds

# How to develop your library?

- Automatic rebuild after every change made to your code:

**ng build my-lib --watch**

- Note: The watch feature requires the compiler option **enableResourceInlining** to be enabled in **`tsconfig.lib.json`**

```
"angularCompilerOptions": {
    "enableResourceInlining": true
},
```

# How to test your library?

- Unit tests can easily be run with **`ng test`**:

**ng test my-lib**

- The above command will re-run the tests after every change you make to your library

# How to use your library?

- Apps and libraries in the same project can use the library directly without having it published:

```
import {MyLibService} from 'my-lib';
```

# How to publish your library?

- If you want to make your library public in the NPM repository, you can run the following commands:

```
ng build my-lib --prod
cd dist/my-lib
npm publish
```

# Thanks for your attention

Link to slides:

https://goo.gl/AQ2kW3

My blog:

https://blog.angulartraining.com

Twitter:

**@AlainChautard**


ANGULAR TRAINING