



Angular 101

A fastboot to Angular

Content



1. What is angular, who made it, who uses it and why should I use it?
2. Installation (Linux Only)
3. Angular-cli
4. Understanding the Architecture
5. Components & Templates
6. Forms
7. Services & HTTP Client
8. Routing
9. Observables*
10. Websocket Clients*



What is angular, who made it and why should i use it?

Front end, client side JS based web framework.

Maintained and created by Google.

Part of MEAN stack. Used to build SPA (Single Page Applications)

We can use it to create dynamic web applications.



Installation

Download node v8+ from <https://nodejs.org/en/>

Extract and set into PATH in .bashrc

NPM is node package manager. Just like PIP for Python.

Install angular cli. A tool to create and manage angular projects Much like django-admin utility for django.

```
npm install -g @angular/cli
```



Angular - CLI

`ng new PROJET_NAME` ← creates a new project

`cd` to the project.

`ng serve --host HOST --port PORT` ← run the webapp on
`HOST:PORT`

`ng generate component <NAME>` ← generate a component named
`NAME`



Understanding the architecture

PROJECT_NAME_DIRECTORY - your project.

NODE_MODULES - files of external modules you are using.

Package.json - list of external modules you are using.

src - source code of project.

- app - All the application related files.

- app.module.ts - Where all the services and components are bundled.

- assets- static things in project.



Components & Templates

Components are sections in a single view in your webapp. A bit similar to apps in Django.

Whatever your component will look like is saved in `name.component.html`, and however it will be handled is saved in `name.component.ts` and the corresponding css for this is handled inside your `name.component.html`.

This html is a template not a static html. Its exactly same as templates in django where we could pass values from code. The only difference is that we can do it asynchronously.



Forms

We can bind a certain variable inside our component.ts to a form or tag inside our component.html, It is called two-way-data-binding and can be used using **ngModels**.

In order to bind a tag to a variable named "VARIABLE" we can add an attribute to a tag as -

```
<tag [(ngModel)]="VARIABLE"/>
```




Services & HTTPClient

Services perform the task of communication that is common between components.

We can create them in our project using

```
ng generate service SERVICE_NAME
```

Services are mostly used for data that may take some time to come to our frontend. Mostly REST API interactions are based on HTTPClients from @angular/common/http module.



Routing

Routes are the same as path. They define which component logic should be executed and when it should be executed.

Its available in @anuglar/router as RouterModule and Routes.

It is quite similar to urlpatterns in Django.



Observables

Observables are a part of ReactiveJavascript aka RxJS, they are a structured asynchronous way of passing messages between two endpoints IN our application.

Observables offer significant benefits over other techniques for event handling, asynchronous programming, and handling multiple values.

Some benefits are that Observables are declarative, ie. you declare a Observable once and then you subscribe to it by passing functions as arguments. Observables can also deliver values of many types.

An observable works on the regex `next* (error|complete)`



Websocket Clients

Websocket clients can be created in Angular using Observables.

```
Observable.fromEvent(Targetted EventEmitter , Event Name String );
```

eg.

```
Observable.fromEvent(this.ws,'message')
```