Angular - Architecture overview

- Aditya Kumar
Chief Technology Officer, edwisor.com

edWisor

We will cover with the following concepts

1) What is Angular ?
2) Modules
3) Components
4) Templates
5) Metadata
6) Data Binding
7) Directives

Library Vs Framework

Libraries

- Are less opinionated
- Large amount of control
- Low entry barrier as easy to learn and use in the existing application.
- Coding standards have to be set internally and steps have to be taken to ensure they are met.
- Handling such issues can take up a lot of effort from the programmer.

Framework

- Are highly opinionated
- Slightly less amount of control
- High entry barrier as difficult to learn and use in the existing application.
- Powerful in organizing large code bases and applications. Coding standards are usually a part of the framework
- Easy to avoid common errors, inefficient or less optimized code

Angular is a framework ...

1) Angular is a framework for building client applications in Html and either Javascript or a language like TypeScript which compiles to javascript.

2) Angular is a platform that makes it easy to build applications with the web.

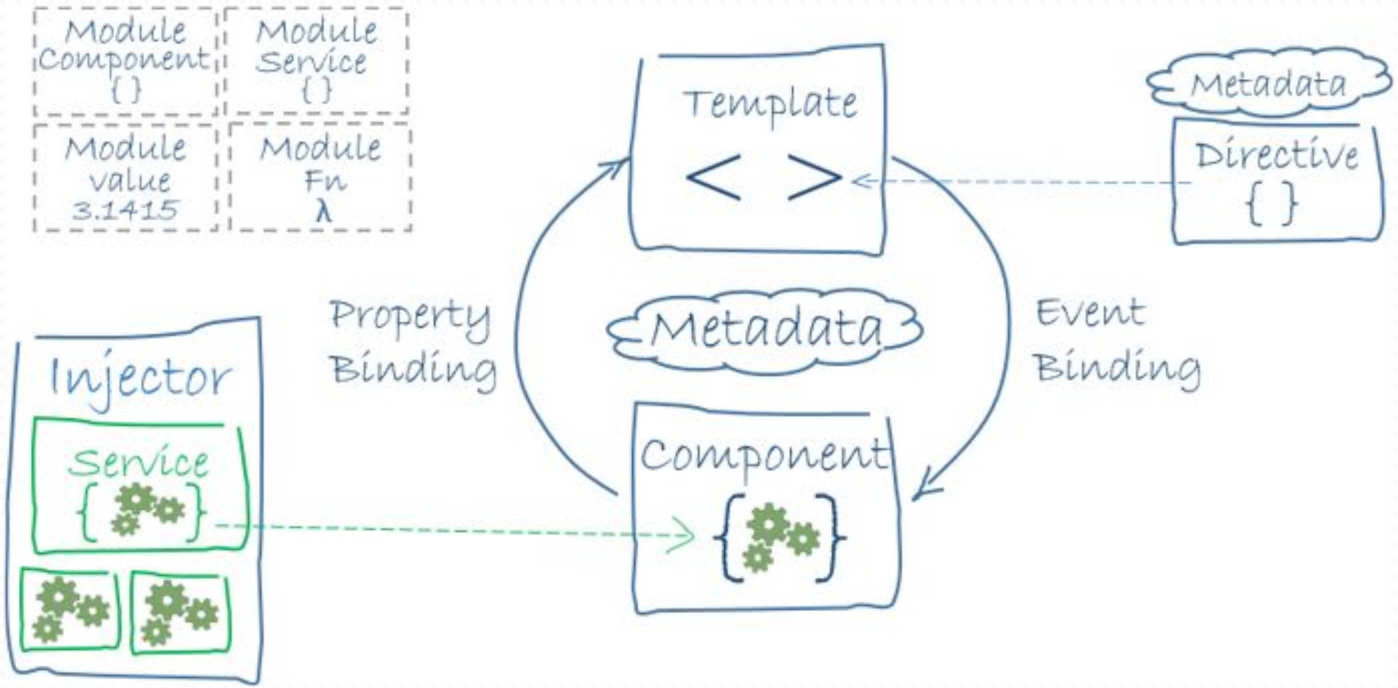
   Angular combines declarative templates, dependency injection, end to end tooling, and integrated best practices to solve development challenges.

   Angular empowers developers to build applications that live on the web, mobile, or the desktop

edWisor

Understanding the architecture is important!

Angular team has made a commendable effort of explaining this on their official documentation as well -
https://angular.io/guide/architecture#architecture-overview



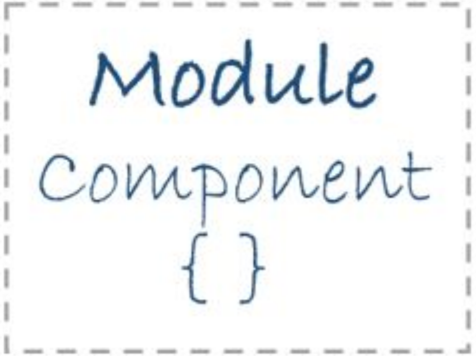
Metadata is what help angularize the Javascript(typescript)

1) Everything is just a Javascript Class. It's the metadata associated with that class that makes it a part of Angular.
2) Metadata are the instructions which tell the Angular - what are the things to be done and what are the resources needed for that.
3) They are like configuration option for calling a library.

edWisor

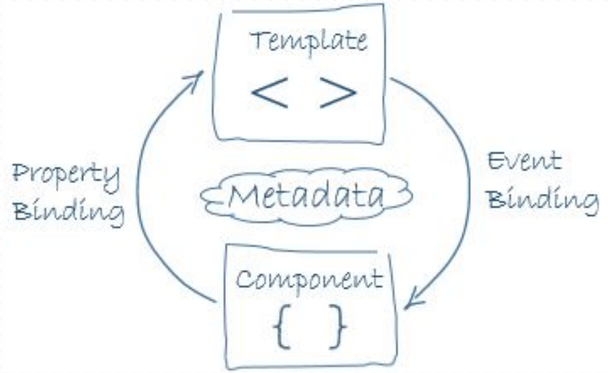
Modules are what package an application

1) Every Angular app has at least one NgModule call Root Module which is by convention called AppModule.
2) It's typically used to
    a) Declare classes related to that part of application
    b) Declare the parts of this module that can be used by other modules
    c) Declare the parts of other module that are to be used in current one.
    d) Declare the global collection of services
    e) Set up the starting point of this application.
3) https://angular.io/guide/architecture#modules



edWisor

Components are the base of modular programming



1) The Angular application manages what the user sees and can do, achieving this through the interaction of a component class instance (the *component*) and its user-facing template.
2) You may be familiar with the component/template duality from your experience with model-view-controller (MVC) or model-view-viewmodel (MVVM). In Angular, the component plays the part of the controller/viewmodel, and the template represents the view.
3) Whenever you think of a view or even a special UI element, you can be sure that there is a component involved.

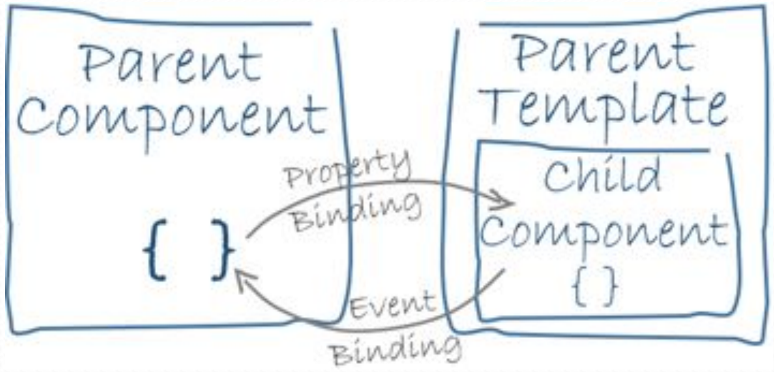
Template is your HTML companion

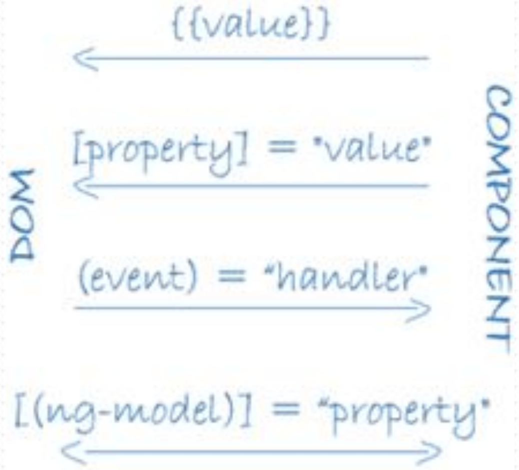
1) Templates contain the natural HTML (with css) along with the extensions provided to HTML by Angular as a framework.
2) They give you the pick of the look and the dynamism associated with it and provide a way to create robust dynamic view while maintaining the harmony of various complex logics involved in an Average application
3) https://angular.io/guide/architecture#templates

Data Bindings save us from the nightmares Of Jquery

1) Interpolation, Property Binding and Event binding make our life easier when it comes to exchange data between component and even between different components.

2) https://angular.io/guide/architecture-components#data-binding





You have been working with directives All this time

1) Directives are typically used in views to extend the natural HTML abilities of the dynamic Angular Views
2) https://angular.io/guide/architecture-components#directives

Why did you make me go through all that pain?

Because this is the knowledge which is going to give you edge over others.

Services and DI will be discussed towards the end

We will be looking deeper into the concept of services and dependency injection later towards the end of the Level

Next Steps are -

Setting up our chat application