Metadata Classes

- Angular uses Metadata to decorate classes, methods and properties.
- The most notable Metadata is the `@component` Metadata.
- Metadta classes are very convenient and they make it easy to work with components, services and the dependency injection system

Below is a list of Angular's core Metadata classes categorized under directives/components, pipes and di.

Directive/component Meta-data

- Component: used to define a component
  - View: used to define the template for a component
  - ViewChild: used to configure a view query
  - ViewChildren: used to configure a view query
- Directive: used to define a directive
  - Attribute used to grab the value of an attribute on an element hosting a directive
  - ContentChild: used to configure a content query
  - ContentChildren: used to configure a content query
  - Input: used to define the input to a directive/component
  - Output: used to define the output events of a directive/component
  - HostBinding: used to declare a host property binding
  - HostListener: used to declare a host listener

Pipes

- Pipe: used to declare reusable pipe function

DI

- Inject: parameter metadata that specifies a dependency.

- Injectable: a marker metadata that marks a class as available to Injector for creation.
- Host: Specifies that an injector should retrieve a dependency from any injector until reaching the closest host.
- Optional: parameter metadata that marks a dependency as optional
- Self: Specifies that an Injector should retrieve a dependency only from itself.
- SkipSelf: Specifies that the dependency resolution should start from the parent injector.
- Query: Declares an injectable parameter to be a live list of directives or variable bindings from the content children of a directive.
- ViewQuery: Similar to `QueryMetadata`, but querying the component view, instead of the content children.