My Interview Plan

- Angular

Topic: Angular

Q1: What is Routing Guard in Angular? ☆

Answer:
Angular’s route guards are interfaces which can tell the router whether or not it should allow navigation to a requested route. They make this decision by looking for a true or false return value from a class which implements the given guard interface.

Source: *medium.com*

Q2: What are the differences between AngularJS (angular 1.x) and Angular (Angular 2.x and beyond)? ☆☆

Answer:
Angular and AngularJS is basically a different framework with the same name.

Angular is more ready for the current state of web standards and the future state of the web (ES6\7, immutiablity, components, shadow DOM, service workers, mobile compatibilty, modules, typescript and so on and so on...)

Angular killed many main features in AngularJS like - controllers, $scope, directives (replaced with `@component` annotations), the module definition, and much more, even simple things like ng-repeat has not left the same as it was.

Also:

1. They added an angular `cli`.
2. Your angular code is written in ES6 Typescript and it compiles at runtime to Javascript in the browser.
3. You bind to your HTML similarly like how you would if in an Angular 1 directive. So variable like $scope and $rootScope have been deprecated.

Source: *stackoverflow.com*

Q3: What is a component? Why would you use it? ☆☆

Answer:
Components are the most basic building block of an UI in an Angular application. An Angular application is a tree of Angular components. Angular components are a subset of directives. Unlike directives, components always have a template and only one component can be instantiated per an element in a template.

A component must belong to an NgModule in order for it to be usable by another component or application. To specify that a component is a member of an NgModule, you should list it in the `declarations` field of that NgModule.

```
@Component({selector: 'greet', template: 'Hello {{name}}!'})
class Greet {
  name: string = 'World';
}
```

Source: *angular.io*

Q4: What is the minimum definition of a component? ☆☆

Answer:
The absolute minimal configuration for a `@Component` in Angular is a template. Both template properties are set to optional because you have to define either `template` or `templateUrl`.

When you don't define them, you will get an exception like this:

```
No template specified for component 'ComponentName'
```

A selector property is not required, as you can also use your components in a route.

Source: *stackoverflow.com*

Q5: What's the difference between an Angular component and module? ☆☆

Answer:
Components control views (html). They also communicate with other components and services to bring functionality to your app.

Modules consist of one or more components. They do not control any html. Your modules declare which components can be used by components belonging to other modules, which classes will be injected by the dependency injector and which component gets bootstrapped. Modules allow you to manage your components to bring modularity to your app.

Source: *stackoverflow.com*

Q6: What is a module, and what does it contain? ☆

Answer:
An Angular module is set of Angular basic building blocks like component, directives, services etc. An app can have more than one module.

A module can be created using `@NgModule` decorator.

```
@NgModule({
  imports:      [ BrowserModule ],
  declarations: [ AppComponent ],
  bootstrap:    [ AppComponent ]
})
export class AppModule { }
```

Source: *stackoverflow.com*

Q7: What is a service, and when will you use it? ☆☆

Answer:
Angular services are singleton objects which get instantiated only once during the lifetime of an application. They contain methods that maintain data throughout the life of an application, i.e. data does not get refreshed and is available all the time. The main objective of a service is to organize and share business logic, models, or data and functions with different components of an Angular application.

The *separation of concerns* is the main reason why Angular services came into existence. An Angular service is a stateless object and provides some very useful functions.

Source: *dzone.com*

Q8: What is the equivalent of ngShow and ngHide in Angular? ☆☆

Answer:
Just bind to the `hidden` property

Source: *stackoverflow.com*

Q9: What are pipes? Give me an example. ☆

Answer:
A **pipe** takes in data as input and transforms it to a desired output. You can chain pipes together in potentially useful combinations. You can write your own custom pipes. Angular comes with a stock of pipes such as `DatePipe`, `UpperCasePipe`, `LowerCasePipe`, `CurrencyPipe`, and `PercentPipe`.

Consider:

```
<p>The hero's birthday is {{ birthday | date }}</p>
```

In this page, you'll use pipes to transform a component's birthday property into a human-friendly date.

Source: *angular.io*

Q10: You have an HTML response I want to display. How do I do that? ☆☆

Answer:
The correct syntax is the following:

```
<div [innerHTML]="theHtmlString"></div>
```

Working in `5.2.6`

Source: *medium.com*

Q11: How can I select an element in a component template? ☆☆

Answer:
You can get a handle to the DOM element via ElementRef by injecting it into your component's constructor:

```
constructor(myElement: ElementRef) { ... }
```

Source: *medium.com*

Q12: What is the equivalent of "ngShow" and "ngHide" in Angular? ☆☆

Answer:
Just bind to the hidden property:

```
[hidden]="!myVar"
```

Source: *medium.com*

Q13: What is the difference between *ngIf vs [hidden]? ☆☆

Answer:
`*ngIf` effectively removes its content from the DOM while `[hidden]` modifies the `display` property and only instructs the browser to not show the content but the DOM still contains it.

Source: *medium.com*

Q14: What is the difference between "@Component" and "@Directive" in Angular? ☆☆

Answer:

- **Directives** add behaviour to an existing DOM element or an existing component instance.
- **A component**, rather than adding/modifying behaviour, actually creates its own view (hierarchy of DOM elements) with attached behaviour.

Write a component when you want to create a reusable set of DOM elements of UI with custom behaviour. Write a directive when you want to write reusable behaviour to supplement existing DOM elements.

Source: *medium.com*

Q15: What does this line do? ☆☆

Details:

```
@HostBinding('[class.valid]') isValid;
```

Answer:
`@HostBinding` lets you set properties on the element or component that hosts the directive.

The code applies the css class `valid` to whatever is using this directive conditionally based on the value of isValid.

Source: *alligator.io*

Q16: How would you protect a component being activated through the router? ☆☆

Answer:
The Angular router ships with a feature called guards. These provide us with ways to control the flow of our application. We can stop a user from visitng certain routes, stop a user from

leaving routes, and more. The overall process for protecting Angular routes:

- Create a guard service: `ng g guard auth`
- Create `canActivate()` or `canActivateChild()` methods
- Use the guard when defining routes

```
// import the newly created AuthGuard
const routes: Routes = [
  {
    path: 'account',
    canActivate: [AuthGuard]
  }
];
```

Some other available guards:

- `CanActivate`: Check if a user has access
- `CanActivateChild`: Check if a user has access to any of the child routes
- `CanDeactivate`: Can a user leave a page? For example, they haven't finished editing a post
- `Resolve`: Grab data before the route is instantiated
- `CanLoad`: Check to see if we can load the routes assets

Source: *scotch.io*

Q17: What are some differences between Angular 2 and 4? ☆☆

Answer:
Just to name a few:

- Improvements in AOT,
- allowing the "else" clause in ngIf,
- support for TypeScript 2.1
- breaking out the animations package

Source: *github.com/WebPredict*

Q18: How would you run unit test? ☆☆

Answer:
The Angular CLI downloads and install everything you need to test an Angular application with the Jasmine test framework.

The project you create with the CLI is immediately ready to test. Just run this one CLI command:

```
ng test
```

Source: *angular.io*