AngularJS

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.  
  
Angular is what HTML would have been, had it been designed for applications. HTML is a great declarative language for static documents. It does not contain much in the way of creating applications, and as a result building web applications is an exercise in what do I have to do to trick the browser into doing what I want?  
The impedance mismatch between dynamic applications and static documents is often solved with:

* a library - a collection of functions which are useful when writing web apps. Your code is in charge and it calls into the library when it sees fit. E.g., jQuery.
* frameworks - a particular implementation of a web application, where your code fills in the details. The framework is in charge and it calls into your code when it needs something app specific. E.g., durandal, ember, etc.

Angular takes another approach. It attempts to minimize the impedance mismatch between document centric HTML and what an application needs by creating new HTML constructs. Angular teaches the browser new syntax through a construct we call directives.  
Examples include:

* Data binding, as in . DOM control structures for repeating, showing and hiding DOM fragments.
* Support for forms and form validation.
* Attaching new behavior to DOM elements, such as DOM event handling.
* Grouping of HTML into reusable components.

## **Angular2 Interview Questions.**

**Q: What are the new features of Angular2?**

**A:**Angular 2 is written entirely in Typescript and meets the ECMAScript 6 specification.

* **Component-Based**- Angular 2 is entirely component based. Controllers and $scope are no longer used. They have been replaced by components and directives.
* **Directives**- The specification for directives is considerably simplified, although they are still subject to change. With the @Directive annotation, a directive can be declared.
* **Dependency Injection**- Because of the improved dependency injection model in Angular2 there are more opportunities for component / object-based work.
* **Use of TypeScript**-TypeScript is a typed super set of JavaScript which has been built and maintained by Microsoft and chosen by the AngularJS team for development. The presence of types makes the code written in TypeScript less prone to run-time errors. In recent times, the support for ES6 has been greatly improved and a few features from ES7 have been added as well.
* **Generics**- TypeScript has generics which can be used in the frontend.
* **Lambdas with TypeScript**- In TypeScript, lambdas are available.
* **Forms and Validations**- Forms and validations are an important aspect of frontend development. Within Angular 2 the Form Builder and Control Group are defined.

**Q: What is the need of Angular2?  
A:**Angular 2 is not just a typical upgrade but a totally new development. The whole framework is rewritten from the ground. Angular 2 got rid of many things like $scope, controllers, DDO, jqLite, angular.module etc. It uses components for almost everything. Imagine that even the whole app is now a component. Also it takes advantage of ES6 / TypeScript syntax. Developing Angular 2 apps in TypeScript has made it even more powerful. . Apart from that, many things have evolved and re-designed like the template engine and many more.  
  
**Q: What is ECMA Script ?  
A:**ECMAScript is a subset of JavaScript. JavaScript is basically ECMAScript at its core but builds upon it. Languages such as ActionScript, JavaScript, JScript all use ECMAScript as its core. As a comparison, AS/JS/JScript are 3 different cars, but they all use the same engine... each of their exteriors is different though, and there have been several modifications done to each to make it unique. Angular2 supports ES6 and higher versions.  
  
**Q: What is @NgModule?  
A:**@NgModule is a decorator function. A decorator function allows users to mark something as Angular 2 thing (could be a module or component or something else) and it enables you to provide additional data that determines how this Angular 2 thing will be processed, instantiated and used at the runtime. So, whenever user writes @NgModule, it tells the Angular 2 module, what’s going to be included and used in and using this module.  
  
**Q: What is Traceur compiler ?  
A:**Traceur is a JavaScript.next-to-JavaScript-of-today compiler that allows you to use features from the future today. Traceur supports ES6 as well as some experimental ES.next features. Traceur's goal is to inform the design of new JavaScript features which are only valuable if they allow you to write better code.

**Q: What is TypeScript ?  
A:**TypeScript is a typed super set of JavaScript which has been built and maintained by Microsoft and chosen by the AngularJS team for development. The presence of types makes the code written in TypeScript less prone to run-time errors. In recent times, the support for ES6 has been greatly improved and a few features from ES7 have been added as well.  
  
**Q: What is component in AngularJs 2 ?  
A:**In Angular, a Component is a special kind of directive that uses a simpler configuration which is suitable for a component-based application structure.  
  
**Q: What is @Inputs in Angular 2? ?  
A:**@Input allows you to pass data into your controller and templates through html and defining custom properties. This allows you to easily reuse components and have them display different values for each instance of the renderer.  
  
**Q: What is @Outputs in Angular?  
A:** Components push out events using a combination of an @Output and an EventEmitter. This allows a clean separation between reusable Components and application logic.  
  
**Q: What are differences between Components and Directives?  
A:**

Components

For register component we use @Component meta-data annotation.

Component is a directive which use shadow DOM to create encapsulate visual behavior called components. Components are typically used to create UI widgets.

Component is used to break up the application into smaller components.

Only one component can be present per DOM element.

@View decorator or templateurl template are mandatory in the component.

Directives

For register directives we use @Directive meta-data annotation.

Directives is used to add behavior to an existing DOM element.

Directive is use to design re-usable components.

Many directive can be used in a per DOM element.

Directive don’t have View.

**Q: What is primeng? How can it be used with angular2?  
A:**PrimeNG is a collection of rich UI components for Angular 2. PrimeNG is a sibling of the popular JavaServer Faces Component Suite, PrimeFaces. All widgets are open source and free to use under Apache License 2.0, a commercial friendly license. PrimeNG is developed by PrimeTek Informatics, a company with years of expertise in developing open source UI components. AngularJS makes it possible to use predefined components for development like tables etc. This helps developers save time and efforts. Using PrimeNG developers can create awesome applications in no time