Ref 1 : <http://www.code-sample.com/2016/06/what-is-angular-2.html> ( A lot of questions are there)

Ref 2 : https://dzone.com/articles/angular-2-gotchas-and-interview-questions

1)Difference between angular 2 and angular 1

Ans :

The basic differences and many more advantages on Angular2 vs. Angular 1 as following,

1.      Angular2 has better performance.

2.      Angular2 has more powerful template system.

3.      Angular2 provide simpler APIs, lazy loading and easier to application debugging.

4.      Angular2 much more testable

5.      Angular2 provides to nested level components.

6.      Angular2 execute run more than two programs at the same time.

7.      Angular1 is controllers and $scope based but Angular2 is component based.

8.      The Angular2 structural directives syntax is changed like ng-repeat is replaced with \*ngFor etc.

9.      In Angular2, local variables are defined using prefix (#) hash. You can see the below \*ngFor loop Example.

Angular 1 Controller:-

var app = angular.module("userApp", []);

app.controller("productController", function($scope) {

$scope.users = [{ name: "Anil Singh", Age:30, department :"IT" },

{ name: "Aradhya Singh", Age:3, department :"MGMT" }];

});

Angular 2 Components using TypeScript:-

Here the @Component annotation is used to add the metadata to the class.

import { Component } from 'angular2/core';

@Component({

selector: 'usersdata',

template: `<h3>{{users.name}}</h3>`

})

export class UsersComponent {

users = [{ name: "Anil Singh", Age:30, department :"IT" },

{ name: "Aradhya Singh", Age:3, department :"MGMT" }];

}

Bootstrapping in Angular 1 using ng-app,

angular.element(document).ready(function() {

angular.bootstrap(document, ['userApp']);

});

Bootstrapping in Angular 2,

import { bootstrap } from 'angular2/platform/browser';

import { UsersComponent } from './product.component';

bootstrap(UserComponent);

The Angular2 structural directives syntax is changed like ng-repeat is replaced with \*ngFor etc.

For example as,

//Angular 1,

<div ng-repeat="user in users">

Name: {{user.name}}, Age : {{user.Age}}, Dept: {{user.Department}}

</div>

//Angular2,

<div \*ngFor="let user of users">

Name: {{user.name}}, Age : {{user.Age}}, Dept: {{user.Department}}

</div>

2)Difference between angular 1 and angular 2 ?

3)What are the core component of angular2 ?

Ans : The core components of angular2 are as follows :

The Angular 2 architecture diagram identifies the eight main building blocks as.

1.       [Module](http://www.code-sample.com/2016/04/cannot-find-module-angular2-core.html)

2.       [Component](http://www.code-sample.com/2016/06/angular-2-components.html)

3.       [Template](http://www.code-sample.com/2016/06/angular-2-template-components.html)

4.       [Outpouts](http://www.code-sample.com/2016/06/angular-2-outputs.html" \t "_blank)

5.       [Data Binding](http://www.code-sample.com/2016/06/angular-2-template-components.html)

6.       [Directive](http://www.code-sample.com/2016/06/angular-2-directives-components.html)

7.       Service

8.       [Dependency Injection](http://www.code-sample.com/2016/04/dependency-injection-in-angular-2.html)

4)Difference between constructor and onInit methods ?

Ans :

1.      The constructor is a default method runs when component is being constructed.

2.      The constructor is a typescript feature and it is used only for a class instantiations and nothing to do with Angular 2.

3.      The constructor called first time before the ngOnInit().

Angular 2 ngOnInit:-

1.      The ngOnInit event is an Angular 2 life-cycle event method that is called after the first ngOnChanges and the ngOnInit method is use to parameters defined with @Input otherwise the constructor is OK.

2.      The ngOnInit is called after the constructor and ngOnInit is called after the first ngOnChanges.

3.      The ngOnChanges is called when an input or output binding value changes.

5)What are new in angular 4 ?

Ans :

1. Smaller & Faster Apps - Angular 4 applications is smaller & faster in comparison with Angular 2.

2. View Engine Size Reduce - Some changes under to hood to what AOT generated code compilation that means in Angular 4, improved the compilation time. These changes reduce around 60% size in most cases.

3. Animation Package- Animations now have their own package i.e. @angular/platform-browser/animations

4. Improvement - Some Improvement on \*ngIf and \*ngFor.

5. Template - The template is now ng-template. You should use the “ng-template” tag instead of “template”. Now Angular has its own template tag that is called “ng-template”.

6. NgIf with Else – Now in Angular 4, possible to use an else syntax as,

<div \*ngIf="user.length > 0; else empty"><h2>Users</h2></div>

<ng-template #empty><h2>No users.</h2></ng-template>

7. AS keyword – A new addition to the template syntax is the “as keyword” is use to simplify to the “let” syntax.

Use of as keyword,

<div \*ngFor="let user of users | slice:0:2 as total; index as = i">

{{i+1}}/{{total.length}}: {{user.name}}

</div>  
To subscribe only once to a pipe “|” with “async” and If a user is an observable, you can now use to write,

<div \*ngIf="users | async as usersModel">

<h2>{{ usersModel.name }}</h2> <small>{{ usersModel.age }}</small>

</div>

8. Pipes - Angular 4 introduced a new “titlecase” pipe “|” and use to changes the first letter of each word into the uppercase.   
The example as,

<h2>{{ 'anil singh' | titlecase }}</h2>

<!-- OUPPUT - It will display 'Anil Singh' -->  
9. Http - Adding search parameters to an “HTTP request” has been simplified as,

//Angular 4 -

http.get(`${baseUrl}/api/users`, { params: { sort: 'ascending' } });

//Angular 2-

const params = new URLSearchParams();

params.append('sort', 'ascending');

http.get(`${baseUrl}/api/users`, { search: params });  
10. Test- Angular 4, overriding a template in a test has also been simplified as,

//Angular 4 -

TestBed.overrideTemplate(UsersComponent, '<h2>{{users.name}}</h2>');

//Angular 2 -

TestBed.overrideComponent(UsersComponent, {

set: { template: '<h2>{{users.name}}</h2>' }

});  
11. Service- A new service has been introduced to easily get or update “Meta Tags” i.e.

@Component({

selector: 'users-app',

template: `<h1>Users</h1>`

})

export class UsersAppComponent {

constructor(meta: Meta) {

meta.addTag({ name: 'Blogger', content: 'Anil Singh' });

}

}

12. Forms Validators - One new validator joins the existing “required”, “minLength”, “maxLength” and “pattern”. An email helps you validate that the input is a valid email.

13. Compare Select Options - A new “compareWith” directive has been added and it used to help you compare options from a select.

<select [compareWith]="byUId" [(ngModel)]="selectedUsers">

<option \*ngFor="let user of users" [ngValue]="user.UId">{{user.name}}</option>

</select>

14. Router - A new interface “paramMap” and “queryParamMap” has been added and it introduced to represent the parameters of a URL.

const uid = this.route.snapshot.paramMap.get('UId');

this.userService.get(uid).subscribe(user => this.name = name);

15. CanDeactivate - This “CanDeactivate” interface now has an extra (optional) parameter and it is containing the next state.

16. I18n - The internationalization is tiny improvement.

//Angular 4-

<div [ngPlural]="value">

<ng-template ngPluralCase="0">there is nothing</ng-template>

<ng-template ngPluralCase="1">there is one</ng-template>

</div>

//Angular 2-

<div [ngPlural]="value">

<ng-template ngPluralCase="=0">there is nothing</ng-template>

<ng-template ngPluralCase="=1">there is one</ng-template>

</div>

**6) How to declare global variable in angular 2 :**

Ans :

1.     Create Global Variables.

2.     Import and Use the Global Variables in the Component.

3.     Result

**Create Global Variables :- “app.global.ts”**

import { Injectable } from '@angular/core';

@Injectable()

export class AppGlobals {

    readonly baseAppUrl: string = 'http://localhost:57431/';

    readonly baseAPIUrl: string = 'https://api.github.com/';

}

**Import and Use the Global Variables in the Component:- “user.component.ts”**

import { Component, Injectable} from '@angular/core';

import { CommonModule } from '@angular/common';

import { HttpModule, Http } from '@angular/http';

import { UserService } from '../service/user.service';

**import { AppGlobals } from '../shared/app.globals';**

@Component({

    selector: 'user',

    templateUrl: './user.component.html',

    styleUrls: ['./user.component.css'],

**providers**: [UserService, **AppGlobals**]

})

export class UserComponent {

    //USERS DECLARATIONS.

    users = [];

    //HOME COMPONENT CONSTRUCTOR

    constructor(private userService: UserService, **private \_global: AppGlobals**) { }

    //GET USERS SERVICE ON PAGE LOAD.

    ngOnInit() {

        this.userService.getAPIUsers(**this.\_global.baseAPIUrl** + 'users/hadley/orgs').subscribe(data => this.users = data);

        this.userService.getAppUsers(**this.\_global.baseAppUrl** + 'api/User/GetUsers').subscribe(data => console.log(data));

    }

}

//END BEGIN – USERCOMPONENT

**“user.server.ts” :-**

import { Injectable, InjectionToken } from '@angular/core';

import { Http, Response } from '@angular/http';

import 'rxjs/add/operator/map';

//BEGIN-REGION - USERSERVICE

@Injectable()

export class UserService {

    constructor(private \_http: Http) {

    }

    getAPIUsers(apiUrl) {

        return this.\_http.get(apiUrl).map((data: Response) => data.json());

    }

    getAppUsers(apiUrl) {

        return this.\_http.get(apiUrl).map((data: Response) => data);

    }

}

//END BEGIN – USERSERVICE

7)What is primeNg in angular2 :

Ans :

<https://yakovfain.com/2016/10/06/primeng-ui-components-for-angular-2/>

<https://www.primefaces.org/primeng/#/>

8)Security checks for angular2 ?

Ans :

Just like any other client side or web application, angular 2 application should also follow some of the basic guidelines to mitigate the security risks. Some of them are:

* Avoid using/injecting dynamic Html content to your component.
* If using external Html, that is coming from database or somewhere outside the application, sanitize it.
* Try not to put external urls in the application unless it is trusted. Avoid url re-direction unless it is trusted.
* Consider using AOT compilation or offline compilation.
* Try to prevent XSRF attack by restricting the api and use of the app for known or secure environment/browsers.

9)What is AOT compilation ?

Ans : AOT compilation stands for Ahead Of Time compilation, in which the angular compiler compiles the angular components and templates to native JavaScript and HTML during the build time. The compiled Html and JavaScript is deployed to the web server so that the compilation and render time can be saved by the browser.