1. What is dependency Injection?

Ans: Injecting the required dependencies to a dependent object is known as Dependency Injection. And this is one of the technique of "Inversion of Control". And it's a software design pattern which deals with how components get hold of their dependencies.

2. Why Dependency Injection (DI)?

Ans: The fundamental requirement of using this pattern is to for passing the sevice to the client rather than allowing client to built (or) find the service. The main intention behind DI is to decouple objects.

3. How to inject dependencies?

Ans: Angular js knows predefined dependencies by their names, so to inject them you need to specify its name exactly how its defined in angular. But order of the dependecies can be changed because its not similar to functional parameters which we declare in javascript.

Ex: app.controller("myController", function ($scope) { $scope.message = "Hello World"; });

4. What are the different ways of injecting dependencies?

Ans:

1) as functional arguments.
2) as array arguments.
3) using $inject service.
4) using Named Functions
5) using Inline functions.

5. What are the core objects and components to achieve DI?

Ans:

1) Factory
2) Service
3) Value
4) Constant
5) Provider

6. Differences between service and factory?

Ans:

Services: Is a constructor function which will create object using new keyword. It doesn't return any objects. We can add properties and functions to service object using this keyword, where this refers to the current constructor. Can inject dependencies like $http, if needed.

It's a singleton object which will share single object throughout the application. ex: sharing user details in the application.

syntax: app.service('serviceName',function(){})

Factory:

It is a functions which should return an object which can include properties and logical functions. It can be called as a wrapper of logical functions like a class and it can be instantiated in different controllers when you are using it with constructor function.

syntax: app.factory('serviceName',function(){ return serviceObj;})

7. Difference between constant and value?

Ans:

Contant:

1) It can be injected into configurations anywhere.
2) It cannot be intercepted by decorator i.e it cannot be changed once its initialized.

Value:

1) It cannot be injected into configurations anywhere like constant.
2) It can be intercepted by the decorator.