Angular js

AngularJS is the most popular, open-source, structural [JavaScript](https://www.interviewbit.com/courses/fast-track-js/)-based framework, developed by Google, that was mainly built for developing large-scale, enterprise-level, dynamic, single-page web applications by extending the HTML syntax and following the MVC (Model-View-Controller) architecture. AngularJS is used for creating easily maintainable, responsive, and cross-browser-compatible enterprise-level web applications.

## Define AngularJS and what are its key features?

The main features of AngularJS are listed below:

* **Applications** developed in AngularJS are testable.
* **Data-binding** − AngularJS provides the most important feature of data binding which facilitates the synchronization of data between the model and the view components in the framework.
* **Controller** − AngularJS is built on JavaScript components and the JavaScript functions bound to scope are called controllers.
* **Services** − AngularJS has many in-built services such as $http which helps in making XMLHttpRequests and AJAX calls.
* **Scope** − AngularJS provides special objects called Scope which refer to the models and is a glue between the view and the controller.
* **Filters** − AngularJS supports several in-built filters as well as provides the ability to define custom filters that aid in subsetting the array items and filtering based on required conditions.
* **Directives** − Directives represent the markers of the DOM elements like attributes, elements, CSS, etc. They are used for creating custom HTML tags that act as widgets. AngularJS supports in-built directives like ngBind, ngModel, ngHide, etc, and also supports the creation of user-defined directives to achieve code reusability.
* **Routing** − Routing is the most important concept supported by AngularJS that involves switching of the views based on any condition.
* **MVC pattern** − MVC pattern also stands for Model-View-Controller pattern is followed by AngularJS that helps it allocate responsibilities appropriately. Model does the task of managing the application data. Views do the task of displaying the application data and the controllers act as an interface between the Model and View to implement application logic.
* **Dependency Injection** − AngularJS was mainly created to demonstrate the feature of dependency injection. This feature helps developers to develop, maintain and test applications easily by defining the interactions and resolving the dependencies between various components.

### 2. Define Scope in AngularJS.

Scopes are special objects in AngularJS that act as a glue between the view and the controller. They refer to the model component of the MVC architecture. They are arranged in a hierarchical way to mimic the DOM structure hierarchy. AngularJS has an in-built $scope object that has all the application data and the corresponding methods bound to that scope.

### 3. What do the services represent in AngularJS?

* services in AngularJS represent reusable components or functionality that can be shared and injected into different parts of an application.
* They encapsulate specific functionality, promote code reusability, enable dependency injection, and contribute to modularity and testability.
* Services are singletons, ensuring consistent data and state throughout the application. They can be both built-in (e.g., $http, $rootScope) and custom-created to suit application-specific needs.

# What is two-way data binding in AngularJS?

* two-way data binding in AngularJS is a feature that allows automatic synchronization of data between the model and the view.
* It enables changes in the model to update the view, and changes in the view to update the model, without the need for manual handling or event listeners.
* This simplifies the development process and keeps the model and view in sync with each other.

# Explain the concept of directives in AngularJS.

Directives in AngularJS are a feature that allows you to extend HTML with custom behaviors and create reusable components. They are markers on DOM elements that instruct AngularJS to attach specific functionality to those elements. Directives enable you to enhance HTML by adding new behavior or modifying existing behavior, making your code more expressive and declarative.

The concept of directives in AngularJS offers the following benefits:

* Reusability: Directives promote code reusability by allowing you to create modular and self-contained components. Once defined, directives can be used across multiple elements or views in your application.
* Custom Functionality: Directives provide a way to define custom HTML elements, attributes, classes, or comments that encapsulate specific behavior. You can define directives to handle DOM manipulation, event handling, data binding, and more.
* Enhanced HTML: Directives enrich HTML by extending its capabilities and making it more expressive. They allow you to create domain-specific language (DSL)-like syntax, improving the readability and maintainability of your code.
* Controller Linking: Directives facilitate communication between different parts of your application through a technique called "controller linking." They can interact with parent controllers and other directives, enabling collaboration and modular design.
* AngularJS Built-in Directives: AngularJS provides a set of built-in directives, such as ng-model, ng-repeat, ng-show, etc., which cover common use cases. These directives allow you to quickly achieve dynamic behavior and interact with data in your application.

# What is the difference between ng-model and ng-bind directives?

1. ng-model: The ng-model directive is used for two-way data binding. It binds the value of an HTML form element (like input, select, textarea) to a property in the AngularJS controller or scope. Any changes made to the form element by the user are automatically reflected in the bound property, and any changes to the property are immediately updated in the form element.
2. ng-bind: The ng-bind directive is used for one-way data binding. It binds the value of an expression in the AngularJS controller or scope to the inner HTML of an HTML element. It evaluates the expression and updates the HTML content with the resulting value. However, it does not establish a two-way data binding, so changes made in the HTML element will not update the bound expression.

# What is the role of filters in AngularJS?

filters in AngularJS are used to format, transform, and filter data in the view layer of an application. They provide a way to manipulate data before it is displayed to the user.

Here are the key roles of filters in AngularJS:

1. Formatting: Filters can format data in a desired way, such as formatting dates, numbers, or currency. They enable the presentation of data in a more readable and user-friendly format.
2. Data Transformation: Filters allow data transformation by applying various operations, such as converting text to uppercase or lowercase, truncating strings, or extracting specific parts of a string.
3. Filtering: Filters help in filtering data based on specific criteria. They enable the creation of search functionality or the filtering of lists based on user input.
4. Chaining: Multiple filters can be chained together to apply a sequence of transformations or filters to the data. This allows for complex data manipulation and formatting.

# How does dependency injection work in AngularJS?

dependency injection (DI) in AngularJS is a design pattern that allows the components of an application to be loosely coupled and easily testable. It enables the framework to manage and provide the required dependencies to the components when they are needed.

Here's a brief explanation of how dependency injection works in AngularJS:

Dependency Registration: In AngularJS, dependencies are registered using the module system. You define the dependencies required by a component (such as a controller, service, or directive) and register them with the AngularJS module.

Dependency Resolution: When a component needs its dependencies, AngularJS resolves them based on the registered dependencies. It analyzes the dependencies of the component and automatically injects the appropriate instances or values into the component.

Explicit or Implicit Injection: Dependency injection in AngularJS can be done explicitly or implicitly. Explicit injection is when you specify the dependencies as function parameters of the component's constructor or function. Implicit injection is when you rely on the function parameter names to infer the dependencies.

Singleton Services: AngularJS uses a singleton pattern for services. When a service is injected into multiple components, AngularJS ensures that only one instance of the service is created and shared among the components.

Injection Hierarchies: AngularJS creates an injection hierarchy based on the component structure. If a dependency is not found at the current level, it will traverse up the hierarchy until it finds a matching dependency or throws an error.

Dependency injection in AngularJS promotes modularity, reusability, and testability. It allows components to be easily replaced, extended, or tested in isolation by providing their dependencies from the outside. This approach helps to reduce tight coupling, increase code maintainability, and improve overall application design.

# What are services in AngularJS? Give examples of built-in services.

services in AngularJS are reusable components that provide functionality and data to various parts of an application. They are responsible for implementing business logic, data manipulation, and communication with external resources. Services promote code modularity, reusability, and maintainability by separating concerns and providing a centralized way to manage shared functionality.

Here are examples of built-in services in AngularJS:

* $http: This service facilitates communication with remote servers using HTTP methods such as GET, POST, PUT, and DELETE. It allows fetching data from APIs, sending data to the server, and handling asynchronous operations.
* $rootScope: It represents the root scope of the AngularJS application and serves as a global scope accessible across all controllers and views. It provides a way to share data and functions between different parts of the application.
* $timeout: This service is used to execute a function after a specified delay. It is often used for adding delays, timeouts, or scheduling tasks within an application.
* $filter: It provides a set of built-in filters for formatting and manipulating data in the view layer. Filters allow transformations such as formatting dates, numbers, currency, sorting arrays, and filtering lists based on specific criteria.
* $location: This service provides methods to interact with the browser's URL. It allows reading and modifying the URL, accessing query parameters, and navigating to different routes within the application.

# What is the difference between $scope and $rootScope?

* In AngularJS, $scope and $rootScope are two important objects used for data binding and sharing data between different components. The main difference between them is the scope hierarchy and their accessibility.
* $scope: $scope is an object that represents the scope of a specific controller, directive, or component. It is created for each instance of the controller or directive. It provides a context for data binding and allows you to expose properties and methods to the view. The scope is hierarchical, meaning that child scopes inherit properties and methods from their parent scopes. $scope is limited to the specific component or directive where it is defined.
* $rootScope: $rootScope is the root scope of the AngularJS application. It is the parent scope for all other scopes and is accessible throughout the entire application. Any properties or methods defined on $rootScope are accessible from any child scope. It acts as a global scope that can be used to share data or functions across different components or controllers.

# How does routing work in AngularJS?

# What is the purpose of $http service in AngularJS?

The $http service in AngularJS is used to make HTTP requests to external servers or APIs. It provides a simplified way to interact with web services and retrieve data asynchronously.

The main purpose of the $http service is to handle AJAX (Asynchronous JavaScript and XML) requests and retrieve data from a server without causing a full page refresh. It supports various HTTP methods like GET, POST, PUT, DELETE, etc., allowing you to perform different types of operations on remote resources.

Here are some key points about the $http service:

* HTTP Requests: The $http service allows you to send HTTP requests to remote servers by specifying the URL, method, headers, and data payload. It provides a simple and consistent API for making AJAX calls.
* Promises: The $http service returns a promise, which represents the future result of the HTTP request. Promises allow you to handle asynchronous operations and handle the response once it's available.
* Response Handling: Once the server responds to the HTTP request, the $http service returns a response object containing the data, status code, headers, and other related information. You can process this response in your application and update the UI accordingly.
* Error Handling: The $http service also provides error handling capabilities. You can handle errors, such as network failures or server errors, and take appropriate actions based on the error response received.

# Explain the concept of controllers in AngularJS.

In AngularJS, controllers are responsible for managing the data and behavior of a specific part of the application. They act as intermediaries between the data model and the view, exposing data and functions to be used in the UI. Controllers handle user interactions, update the data model, and update the view accordingly. They help in separating concerns and maintaining a modular code structure.

# What is the role of the $digest cycle in AngularJS?

the $digest cycle in AngularJS is responsible for updating the bindings and detecting changes in the data model. It is an essential part of the AngularJS framework that ensures that the view reflects the most up-to-date state of the application.

The main role of the $digest cycle is to:

* Detect Changes: The $digest cycle iterates through all the watchers (data bindings) in the application and checks if their associated data has changed. It compares the current value of each watcher with its previous value to identify any changes.
* Update the View: If changes are detected, the $digest cycle updates the corresponding parts of the view to reflect the new values. It triggers the re-evaluation of expressions, re-renders templates, and updates the DOM elements affected by the changes.
* Stabilize the Model: The $digest cycle continues to run until all the watchers are stable, meaning that no further changes are detected in subsequent iterations. It ensures that the model and the view are synchronized and consistent.

# How can you communicate between controllers in AngularJS?

In AngularJS, there are several ways to communicate between controllers. Here are a few common approaches:

Using a Service: You can create a shared service that acts as a mediator between controllers. The service can hold shared data or provide functions to update and retrieve data. Controllers can inject and use this service to communicate and share information.

Using $rootScope: The $rootScope is the parent scope for all other scopes in an AngularJS application. You can use it to broadcast events using $broadcast or emit events using $emit. Controllers can listen to these events and respond accordingly.

Using Events: AngularJS provides an event system where you can define and dispatch custom events. Controllers can use $scope.$on to listen to these events and perform actions based on them. Events can be broadcasted from a parent controller to child controllers or between sibling controllers.

Using Route Parameters: If controllers are associated with different routes, you can pass data between them using route parameters. Controllers can retrieve the data from the route parameters and use it as needed.

Using $routeParams: In AngularJS, $routeParams provides access to the parameters of the current route. Controllers can access these parameters to communicate and share data.

Using $location Service: The $location service allows you to read and modify the URL. Controllers can use this service to set URL parameters or query strings, which can be accessed by other controllers