Introducción

Que es angular

Porque Angular

La aplicación Shell

Instalar angular CLI

Crear una nueva aplicación

Aplicación Server

Componentes de Angular

Cambiar el título de la aplicación

Agregando estilos a la aplicación

Building Blocks of Angular Apps

03:41

Components

09:38

Generating Components Using Angular CLI

04:41

Templates

02:28

Directives

03:27

Services

04:29

Dependency Injection

07:20

Generating Services Using Angular CLI

02:11

Arquitectura de angular

Modulos

Componentes

Templetes

Metadato

Data binding

Directivas

Servicios

Inyección de dependencias

2 Introducción a Angular 5

– ¿Angular5 o **angular.io**?  
– ¿Qué nos ofrece?

4 TypeScript

– Introducción a TS  
– Programación orientada a componentes  
– Angular 5 en TS  
[/vc\_column\_text]

5 Componentes

– Arquitectura de una aplicación  
– Los componentes en detalle  
– Binding de datos

Angular es un marco de JavaScript de código abierto que simplifica el código de enlace entre los objetos de JavaScript y los elementos de la IU HTML.

Se muestra una simple función de "Cliente" con la propiedad "Nombre del cliente". También hemos creado un objeto llamado "Cust" que es del tipo de clase "Cliente".

function Customer()

{

this.CustomerName = "AngularInterview";

}

var Cust = new Customer();

Ahora digamos que el objeto lo queremos vincular a un cuadro de texto HTML dinamico llamado como "TxtCustomerName".

<input type=text id="TxtCustomerName" onchange="UitoObject()"/>

La función "UitoObject" toma datos de la IU y los establece en el objeto, mientras que la otra función, "ObjecttoUI", toma los datos del objeto y los establece en la IU.

function UitoObject()

{

Cust.CustomerName = $("#TxtCustomerName").val();

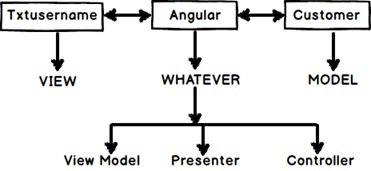
}

function ObjecttoUi()

{

$("#TxtCustomerName").val(Cust.CustomerName);

}



Ahora que el código de enlace tiene diferentes vocabularios.

    Algunos lo llamaron "ViewModel" porque conecta el "Modelo" y la "Vista".

    Algunos lo llaman "Presentador" porque esta lógica no es más que lógica de presentación.

    Y el "Controlador" porque controla cómo se comunicarán la vista y el modelo.

Para evitar esta confusión de vocabulario, el equipo Angular ha denominado este código como "WhatEver". Es ese código " WhatEver " que une la interfaz de usuario y el modelo. Es por eso que escuchará a muchos desarrolladores que dicen que Angular implementa la arquitectura "MVW".

Entonces, concluir que el objetivo de Angular es el enlace, el enlace y la unión.