**UNIVERSIDAD CIENTÍFICA DEL SUR**

Facultad:

Ingeniería de Sistemas

Tema:

AngularJS

Asignatura:

Taller De Programación

Profesor:

Eric Gustavo Coronel

Integrantes:

Garcia Chamba, Alexys

Galindo Vidal, Camilo

Peramás Ceras, Joan

**Índice**

Resumen 3

Antecedentes 5

Fundamentos Teórico 7

Caso Demostrativo 13

Discusión 20

Conclusiones 21

Recomendaciones 22

Bibliografía 23

**Resumen**

Recientemente han aparecido una oleada de frameworks, librerías y aplicaciones web que han situado Javascript en otro nivel. **AngularJS** es uno de ellos. Son los frameworks que vienen a aportar herramientas y patrones de diseño con los que Javascript se convierte en un lenguaje capaz de servir como motor de aplicaciones enormes.

[AngularJS](http://www.angularjs.org/) es un framework javascript  de código abierto y desarrollado por Google para crear aplicaciones web dinámicas en el cliente.

Actualmente las computadoras modernas, por muy modestas que sean, son capaces de procesar con velocidad ciertas procesos. Son capaces de recibir simples datos y "procesarlos" ellos mismos el HTML para visualizarlos a base de plantillas. Antes el servidor era el que tenía que enviar el HTML completo al cliente, ahora la tendencia es que solo envíe los datos y que el cliente (navegador o cualquier otro sistema donde desees ver esos datos) sea el que los trate y los muestre debidamente.

Esto ha producido que una parte de la lógica de maquetado y de presentación de la información se haya trasladado del servidor hacia los clientes. La ventaja obvia es que el servidor se ha descargado de trabajo, puesto que simplemente tiene que enviar los datos a través de JSON al cliente y es éste el que se encargará de producir el HTML que sea necesario. Pero no es solo una mejora en relación al servidor en términos de procesamiento, también en términos de bits, porque es más ligero transferir datos simples que el HTML completo para mostrarlos.

En definitiva, el servidor ha repartido la carga de trabajo que solía recaer sobre él entre todos los clientes que se conectan a su servicio. Pero la mejora no se queda solamente en el servidor, sino que el usuario también percibe un mejor desempeño, puesto que las acciones que realiza contra el servidor tienen una respuesta más rápida. Con ello poco a poco las aplicaciones cliente/servidor tienen un desempeño más parecido a las aplicaciones de escritorio. El usuario es el rey y demanda aplicaciones que sean rápidas y no le hagan esperar y eso se lo dan los frameworks como AngularJS.

Al programador además le facilitan las cosas, no solo por disponer de un conjunto de librerías, sino porque los frameworks nos traen un conjunto de paradigmas y patrones que facilitan el desarrollo del software y sobre todo su mantenimiento. Nos referimos principalmente al llamado MVC, que es la separación del código en diferentes responsabilidades.

El presente trabajo nos dará una visión amplia acerca del framework angularjs y su aplicación en un proyecto.

**Antecedentes**

AngularJS comenzó como un proyecto personal de Miško Hevery en 2009 con el fin de desarrollar rápidamente aplicaciones web, la solución de los principales problemas encontrados en las solicitudes de página (SPA – Aplicaciones de una sola página). El nombre AngularJS se refiere a los soportes de ángulo, conocidos como etiquetas (signo menor que o mayor que) "<>" tal como se utiliza en HTML.



Algún tiempo después de la creación de los AngularJS, Miško Hevery estaba trabajando en un proyecto llamado Google Feedback donde aplicó por primera vez su marco en un proyecto de Google y pudo reducir miles de líneas de código.

Con el impacto positivo que AngularJS género dentro de Google, el framework comenzó a ser adoptado de forma gradual. En el ng-conf (Conferencia de AngularJS) del año pasado se informó de que Google tiene internamente alrededor de 110 proyectos, utilizando el marco creado por Hevery.

Actualmente Google es el principal contribuyente al código AngularJS, pero la comunidad técnica es muy activo, que proporciona información continua sobre el funcionamiento de las nuevas características y ayudando en el desarrollo porque el proyecto es completamente de código abierto.

Actualmente Varias compañías están utilizando AngularJS como el marco principal para el desarrollo de aplicaciones web para usuario, de acuerdo con - Libscore - más de 10.500 sitios utilizan AngularJS hoy.

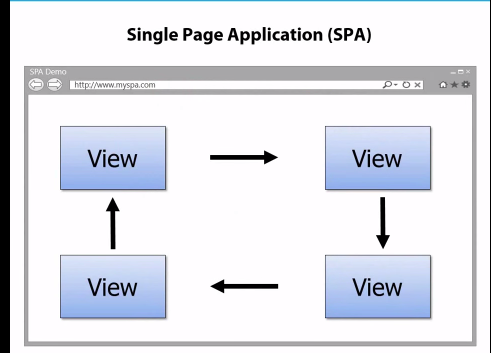
Desde 2014 se convirtió en el marco de ángulo web principal en uso, número de desarrolladores, participación de la comunidad técnica y su adopción por las grandes empresas.

Este año 2016 se lanzó la versión beta de AngularJS 2.0 la cual está siendo desarrollado por Google en colaboración con la comunidad y con el apoyo de empresas como Microsoft.

**Fundamento teórico**

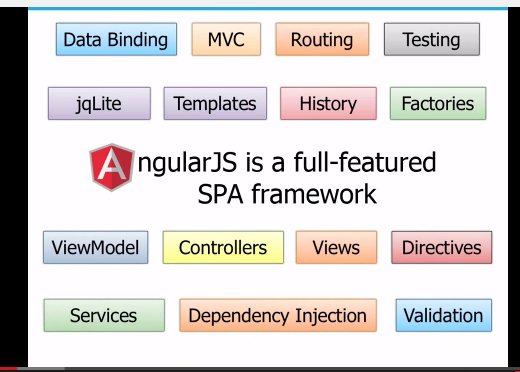


AngularJS es un framework para crear Webapps en lenguaje cliente con Javascript ejecutándose en el conocido [Single-Page Applications (SPA)](http://en.wikipedia.org/wiki/Single-page_application) (aplicación de una sóla página) que extiende el tradicional HTML con etiquetas propias (directivas) como pueden ser ng-app, ng-controller, ng-model, ng-view...



AngularJS es lo que debió ser HTML desde el inicio si hubiera sido diseñado para la creación de aplicaciones de software. HTML es perfecto como lenguaje de definición estática de documentos y mediante AngularJS se puede extender para abarcar las aplicaciones web.

AngularJS es un framework completo en funcionalidades:



Permite utilizar un sistema de plantillas y extender HTML con tags personalizados para expresar de forma clara y concisa los componentes de la aplicación.

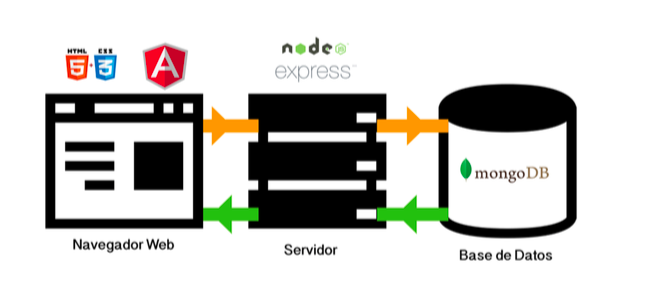
AngularJS permite con su avanzado sistema de vinculación de datos (data-binding) mediante las variables vista/controllador e inyección de dependencias eliminar y simplificar mucho código.

Consultas Ajax con peticiones HTTP, sistema óptimo de templating, manipulación de datos en JSON, inyección de dependencias, deep linking, formularios de validación, desacoplamiento del DOM de Javascript, internacionalización i18n y l10n, filtros, unit testing... serán otras de las genialidades que veremos en AngularJS.

AngularJS es compatible con los navegadores de última generación (Chrome, Firefox, Safari, Opera, Webkits, IE9+).

JavaScript era el lenguaje que se utilizaba en la web para añadir efectos y animaciones a las páginas. Pero este lenguaje ha evolucionado mucho en nuestros días hasta llegar al punto de ser llevado al servidor con Node.js.

Estructura de una aplicación web con AngularJS, NodeJS, y MongoDB.



Empezaremos a revisar y entender cómo funciona angularjs.

**AngularJS y sus características**

Con este framework tendremos la posibilidad de escribir una aplicación de manera fácil, que con solo leerla podríamos entender qué es lo que se quiere lograr sin esforzarnos demasiado. Además de ser un framework que sigue el patrón MVC nos brinda otras posibilidades como la vinculación de datos en dos vías y la inyección de dependencia. Sobre estos términos estamos tratando más adelante.

**Plantillas AngularJS**

Nos permite crear aplicaciones de una sola página, o sea podemos cargar diferentes partes de la aplicación sin tener que recargar todo el contenido en el navegador. Este comportamiento es acompañado por un motor de plantillas que genera contenido dinámico con un sistema de expresiones evaluadas en tiempo real. El mismo tiene una serie de funciones que nos ayuda a escribir plantillas de una forma organizada y fácil de leer, además de automatizar algunas tareas como son: las iteraciones y condiciones para mostrar contenido. Este sistema es realmente innovador y usa HTML como lenguaje para las plantillas. Es suficientemente inteligente como para detectar las interacciones del usuario, los eventos del navegador y los cambios en los modelos actualizando solo lo necesario en el DOM¹¹ y mostrar el contenido al usuario.

**Estructura MVC**

La idea de la estructura MVC no es otra que presentar una organización en el código, donde el manejo de los datos (**M**odelo) estará separado de la lógica (**C**ontrolador) de la aplicación, y a su vez la información presentada al usuario (**V**istas) se encontrará totalmente independiente. Es un proceso bastante sencillo donde el usuario interactúa con las vistas de la aplicación, éstas se comunican con los controladores notificando las acciones del usuario, los controladores realizan peticiones a los modelos y estos gestionan la solicitud según la información brindada. Esta estructura provee una organización esencial a la hora de desarrollar aplicaciones de gran escala, de lo contrario sería muy difícil mantenerlas o extenderlas. Es importante aclarar mencionar que en esta estructura el modelo se refiere a los diferentes tipos de servicios que creamos con Angular.

**Vinculación de datos**

Desde que el DOM pudo ser modificado después de haberse cargado por completo, librerías como jQuery hicieron que la web fuera más amigable. Permitiendo de esta manera que en respuesta a las acciones del usuario el contenido de la página puede ser modificado sin necesidad de recargar el navegador. Esta posibilidad de modificar el DOM en cualquier momento es una de las grandes ventajas que utiliza AngularJS para vincular datos con la vista.

Pero eso no es nuevo, jQuery ya lo hacía antes, lo innovador es, ¿Que tan bueno sería si pudiéramos lograr vincular los datos que tenemos en nuestros modelos y controladores sin escribir nada de código? Seria increíble verdad, pues AngularJS lo hace de una manera espectacular. En otras palabras, nos permite definir que partes de la vista serán sincronizadas con propiedades de Javascript de forma automática. Esto ahorra enormemente la cantidad de código que tendríamos que escribir para mostrar los datos del modelo a la vista, que en conjunto con la estructura **MVC** funciona de maravillas.

**Directivas**

Si vienes del dominio de jQuery esta será la parte donde te darás cuenta que el desarrollo avanza de forma muy rápida y que seleccionar elementos para modificarlos posteriormente, como ha venido siendo su filosofía, se va quedando un poco atrás comparándolo con el alcance de AngularJS. jQuery en si es una librería que a lo largo de los años ha logrado que la web en general se vea muy bien con respecto a tiempos pasados.

A su vez tiene una popularidad que ha ganado con resultados demostrados y posee una comunidad muy amplia alrededor de todo el mundo. Uno de los complementos más fuertes de AngularJS son las directivas, éstas vienen a remplazar lo que en nuestra web haría jQuery. Más allá de seleccionar elementos del DOM, AngularJS nos permite *extender* la sintaxis de HTML. Con el uso del framework nos daremos cuenta de una gran cantidad de atributos que no son parte de las especificaciones de HTML. AngularJS tiene una gran cantidad de directivas que permiten que las plantillas sean fáciles de leer y a su vez nos permite llegar a grandes resultados en unas pocas líneas.

Pero todo no termina ahí, AngularJS nos brinda la posibilidad de crear nuestras propias directivas para extender el HTML y hacer que nuestra aplicación funcione mucho mejor.

**Inyección de dependencia**

AngularJS está basado en un sistema de inyección de dependencias donde nuestros controladores piden los objetos que necesitan para trabajar a través del constructor.

Luego AngularJS los inyecta de forma tal que el controlador puede usarlo como sea necesario. De esta forma el controlador no necesita saber cómo funciona la dependencia ni cuáles son las acciones que realiza para entregar los resultados. Así estamos logrando cada vez más una organización en nuestro código y logrando lo que es una muy buena práctica: “Los controladores deben responder a un principio de responsabilidad única”. En otras palabras, el controlador es para controlar, o sea recibe peticiones y entregar respuestas basadas en estas peticiones, no genera el mismo las respuestas. Si todos nuestros controladores siguen este patrón nuestra aplicación será muy fácil de mantener incluso si su proceso de desarrollo es retomado luego de una pausa de largo tiempo. Si no estás familiarizado con alguno de los conceptos mencionados anteriormente o no te han quedado claros, no te preocupes, todos serán explicados en detalle más adelante. Te invito a que continúes ya que a mi modo de pensar la programación es más de código y no de tantos de conceptos. Muchas dudas serán aclaradas cuando lo veas en la práctica.

**Caso Demostrativo**

**Creando un Proyecto con AngularJS**

Una aplicación AngularJS en su mínima expresión es, en síntesis, un documento HTML enlazado a un archivo javascript en el cual se encuentran un grupo de comandos e instrucciones. Sin embargo, para implementar una aplicación AngularJS necesitas seguir algunos pasos, que aunque sencillos, son indispensables para que tu aplicación se ejecute correctamente. Dedicaremos este capítulo a enumerar y revisar cada uno de los pasos y elementos que componen una aplicación AngularJS: cómo instalar, configurar e inicializar AngularJS. Al final de este capítulo aplicaremos todos los conceptos y técnicas para crear nuestra primera aplicación.

**Descargar AngularJS**

Sin duda el primer paso para comenzar a utilizar esta librería es tener acceso a su código, AngularJS nos ofrece varias formas de utilizar sus rutinas y procesos. AngularJS se debe instalar desde el sitio oficial de Google, aunque puedes encontrarlo en algunos otros lugares, repositorios o enlaces, es recomendable que en la medida de lo posible utilices la versión del sitio oficial para estar 100% seguro de que tienes no solo una versión estable sino también un programa seguro sin ningún tipo de modificación de su código original. El sitio oficial es: https://angularjs.org En cuanto ingreses al sitio oficial encontrarás un botón que te invita a descargar el código de AngularJS, al hacer click en el encontraras una ventana con varias opciones para personalizar tu descarga.



Todas estas opciones están pensadas para que puedas adaptar AngularJS a diferentes escenarios. Vamos a revisar cada uno de estos apartados para ayudarte a elegir la versión perfecta para tus necesidades.

**Librerías adicionales**

Las librerías adicionales te permitirán incluir nuevas modalidades, están separadas del núcleo central de código para reducir el tamaño total de la instalación. Al utilizar un esquema modular tu aplicación nunca va a cargar código que no necesite, solamente lo importante y con causa será instalado en tu aplicación. Una vez elegidas las opciones de tu archivo, procede al proceso de descarga. Si elegiste las opciones recomendadas, al presionar download la aplicación te enviará a un documento que posiblemente se vea similar a este.

**Instalar AngularJS en un documento HTML**

AngularJS depende directamente de tu código HTML, por eso, para instalarlo te recomiendo crear primero un documento HTML bien estructurado, con código validado y siguiendo los standards para asegurarte que todo esta perfecto desde el inicio. Puedes comenzar tu aplicación, utilizando este código base

1 *<!DOCTYPE html>*

2 **<html>**

3 **<head>**

4 **<meta** charset="utf-8"**>**

5 **<title>**AngularJS**</title>**

6 **<meta** name="description" content=""**>**

7 **<meta** name="viewport" content="width=device-width"**>**

8 **</head>**

9 **<body>**

10

11 **</body>**

12 **</html>**

Ahora que tienes el marco perfecto para tu aplicación es momento de instalarlo en tu documento. El proceso de instalación no puede ser más sencillo, solo debes colocar esta linea de código justo antes de la etiqueta </body>

<script src="http//ajax.googleapis.com/ajax/libs/angularjs/1.3.9/angular.min.js"></script>

Para este punto tienes un documento listo para utilizar AngularJS, sin embargo, aunque el documento tiene acceso a la librería no tiene ninguna indicación de cómo y dónde usarla, para ello es necesario definir e inicializar una aplicación.

**Configurar una aplicación AngularJS**

Uno de los elementos fundamentales de AngularJS son las directivas, elementos que podemos inyectar en el código HTML que nos permiten realizar procesos complejos.

Más adelante hablaremos de ellas en detalle, por ahora vamos a ver nuestra primera directiva y una de las más importantes ya que al incluirla en nuestro documento, le indicaremos a AngularJS que puede comenzar a procesar esa parte del documento.

Hablamos de la directiva *ng-app* y la forma de utilizarla es incluyéndose dentro de nuestro documento en forma de un parámetro HTML. Veamos un ejemplo: supongamos que yo deseo que el documento HTML en el que estamos trabajando se comporte como una aplicación angular, solo debo incluir esta directiva dentro de cualquier etiqueta HTML y definir el nombre de mi aplicación. En este caso, si mi aplicación se llama “miAplicacionAngular” la forma de inicializarla sería así

1 **<body** ng-app=“miAplicacionAngular”**>**

2

3 **</body>**

El valor que le estoy asignando a “ng-app” es el nombre con el que identificare a mi aplicación y tienes libertad de darle el nombre que desees. Toma en cuenta que aunque es técnicamente posible incluir únicamente la directiva sin asignarle un nombre a la aplicación y de hecho encontraras muchos ejemplos en internet que declaran la directiva “ng-app” de esta forma:

1 **<body** ng-app **>**

2

3 **</body>**

Aunque en teoría una aplicación tiene la capacidad de funcionar con ese código, es mejor que siempre le definas un nombre desde el inicio. Conforme tu aplicación se vuelva más compleja, tarde o temprano estarás obligado a asignarle un nombre, así que es mejor arrancar con un documento de buena calidad desde el primer momento.En este libro usaremos esta práctica en todos los ejemplos y te recomiendo aplicarlo en tus propios documentos.

**Alcance de ng-app**

La directiva “ng-app” no solo define una aplicación AngularJS, también la delimita a un área de acción específica. Puedes incluir la directiva “ng-app” en cualquier parte de tu documento, pero debes tomar en cuenta que las acciones de tu aplicación AngularJS tendrán efecto únicamente dentro de la etiqueta en que la asignes. Cualquier elemento que esté fuera de la etiqueta elegida, será totalmente ignorado por la aplicación.

Tienes libertad total para usar la directiva “ng-app” en cualquier parte de tu documento y esta posición dependerá muchas veces de las necesidades específicas de tu aplicación, sin embargo te recomiendo que de ser posible incluyas esta directiva en las etiquetas <html> o <body> (solo en una de ellas a la vez) así tendrás una sola app para englobar todas las rutinas del sitio, te aseguras que todo el contenido será parte de la aplicación y tendrás control sobre todos los elementos visibles del documento HTML.

**Inicializar la aplicación**

Ahora que ya tienes declarada tu aplicación usando la directiva “ng-app” es momento de inicializarla. Para realizar este paso debes crear una variable donde declararemos el módulo correspondiente a tu nueva aplicación. Por ejemplo si tenemos una aplicación que fue declarada con:

ng-app='holamundo'

Debemos inicializarla con el código:

var miaplicacionAngular = angular.module('holamundo',[])

En este caso le asignó a la variable el nombre “miaplicacionAngular”, pero tu puedes usar cualquier nombre. El valor que almacena esta variable declara un nuevo módulo de angularJS en el cual estará tu aplicación.

**Cargando módulos adicionales**

Notarás también que agregué un arreglo vacío después del nombre del módulo “[]” , la intención de este arreglo es declarar dependencias adicionales que le añadirán funcionalidades nuevas a tu aplicación. AngularJS nos permite incluir módulos adicionales a la hora de declarar nuestra aplicacion y asi mantener un codigo modular.

Cuando descargas el framework tienes incluyen únicamente las funciones más básicas y frecuentes para mantener la carga rápida y eficiente, pero si necesitas servicios adicionales puedes descargar nuevos módulos e incorporarlos en este arreglo para que estén disponibles en tu aplicación. Por ejemplo, en el caso de que queramos agregarle soporte para administrar los URL’s a través del módulo adicional de routing, lo inyectariamos de esta forma:

var miaplicacionAngular = angular.module('holamundo',['ngRoute'])

**Ejemplo Hola Mundo Final**

1 **<script** src="http://ajax.googleapis.com/ajax/libs/angularjs/1.3.9/angular.min.js\

2 "**></script>**

Luego insertar la directiva ng-app con el nombre de tu aplicación, en este caso la pondremos en la etiqueta <html>

1 <html ng-app='holamundo'>

Finalmente declaramos el modulo de la aplicación para inicializarla

1 var miAppAngular = angular.module('holamundo',[]);

Ahora tenemos una aplicación AngularJS lista para trabajar, vamos a adelantarnos al tema del próximo capítulo y usaremos una expresión para que pruebes lo genial que es AngularJS. Reemplaza el <h1> actual por este código:

1 **<h1>** {{ "Mi Aplicación AngularJS #" + 1 |uppercase }}**</h1>**

La expresión que hemos incluido esta mostrando ahora una concatenación y modificando el texto para que solo se despliegue en mayúsculas. El código completo de tu primera aplicación seria este:

2 **<html** ng-app='holaGustavoCoronel'**>**

3

4 **<head>**

5 **<meta** charset="utf-8"**>**

6 **<title>**AngularJS - Hola Mundo!**</title>**

7 **<meta** name="viewport" content="width=device-width"**>**

8 **</head>**

9 **<body>**

10

11 **<h1>** {{ "Mi Aplicación AngularJS #" + 1 |uppercase }}**</h1>**

12

13 **<script** src="http://ajax.googleapis.com/ajax/libs/angularjs/1.3.9/angular.mi\

14 n.js"**></script>**

15 **<script>**

16

17 var miAppAngular = angular.module('holamundo',[]);

18

19 **</script>**

20 **</body>**

21

22 **</html>**

**Discusión**

**Performance**:

Como se ha comentado anteriormente, Angular tiene el denominado ‘two-way-data-binding‘, que permite que las variables del modelo y la vista están relacionadas y se actualicen en ‘real time’, para lograr esto, pero, hay que evaluar cada variable de las usadas por Angular En cada ‘digest cycle‘ de la aplicación. Esto supone un problema potencial en la ‘performance’ de la aplicación si el desarrollador no lo ha tenido en cuenta y hay estructuras de datos complejas a evaluar.

**Inyección de dependencias:**

El sistema de inyección de dependencias de Angular carga todos los módulos necesarios al cargar la aplicación, esto supone que incluso si el usuario no accede a ciertas partes de esta, se cargan los módulos necesarios para que dichas partes funcionen.  Esto implica un volumen de tráfico mayor del necesario, frente a otros modelos, como el implementado por [RequireJS](http://requirejs.org/), que carga los módulos únicamente cuando estos son necesarios y se van a usar.

Otra cuestión a tratar sobre la inyección de dependencias en Angular, es la anotación que usa, permitiendo tres formas diferentes de anotar las dependencias a un módulo y considerando que una de estas posibles formas, no funcionará si se planea minificar el código de la aplicación

**Conclusiones**

**Two way data binding**

A diferencia de la mayoría de sistemas de ‘templates’ y/o frameworks Angular usa un sistema en el que vista y modelo están en relación constante, se considera el modelo como ‘Single-Source-of-Truth‘. Gracias a esto, se logra que todo cambio visual, se actualice a tiempo real en el modelo y viceversa, evitando que sea el desarrollador el encargado de lograr la sincronía entre modelo y vista, como es el caso de otros frameworks.

**Directivas**

Consisten en marcadores en un elemento de DOM que indican al compilador de Angular que dicho elemento tiene un comportamiento específico, gracias a esto, se puede trabajar fácilmente a nivel de componentes, siendo estos componentes reutilizables en toda la aplicación.

**Comunidad**

La comunidad de desarrolladores han dado gran soporte a este framework (una búsqueda por ‘angular’ en [github](https://github.com/), da 117428 resultados),  por lo que hay gran cantidad de módulos ya creados. Esto permite facilitar y agilizar el desarrollo de aplicaciones, pues el desarrollador se puede focalizar en las partes más complejas del desarrollo.

**Recomendaciones**

* AngularJS es un buen framework para el desarrollo de una aplicación web, sobretodo si en esta prima lograr una interfaz fluida y una buena experiencia de usuario, en la que no se note la comunicación entre cliente y servidor.
* También destacar que el funcionamiento del propio framework anima a optar por diseños basados en componentes de forma que cada uno de ellos tenga su propia lógica (mediante el uso de directivas) y logrando así que la aplicación está compuesta de módulos independientes, evitando problemas de estado compartido.
* AngularJS es un framework de Front-End, y este es únicamente una parte del stack de tecnologías que constituyen el total de un proyecto, por ello hay que tener en cuenta el total de la arquitectura de este, intentando que Angular se sustenta en un Back-End potente que realice toda la lógica de negocio, para poder así delegar en Angular únicamente la lógica visual de la aplicación.

**Bibliografía**

[**https://eladrodriguez.gitbooks.io/angularjs/content/index.html**](https://eladrodriguez.gitbooks.io/angularjs/content/index.html)

Libro online

[**https://docs.angularjs.org/tutorial**](https://docs.angularjs.org/tutorial)

Tutorial oficial

[**http://www.w3schools.com/angular/**](http://www.w3schools.com/angular/)

Tutorial

[**https://www.codecademy.com/learn/learn-angularjs**](https://www.codecademy.com/learn/learn-angularjs)

Curso online