
ANGULARJS
by Google

Bienvenidos !!!

Programación con Angulas JS

Instructor: Borja Cabeza

Herramientas: Adobe Brackets 

Donwload desde http://brackets.io

Duración: 24 Horas

1. Introducción a Angular JS

¿ Qué es Angular JS ?

¿ Qué es la arquitectura MVC ?

Angular JS vs jQuery

Angular JS vs bootstrap

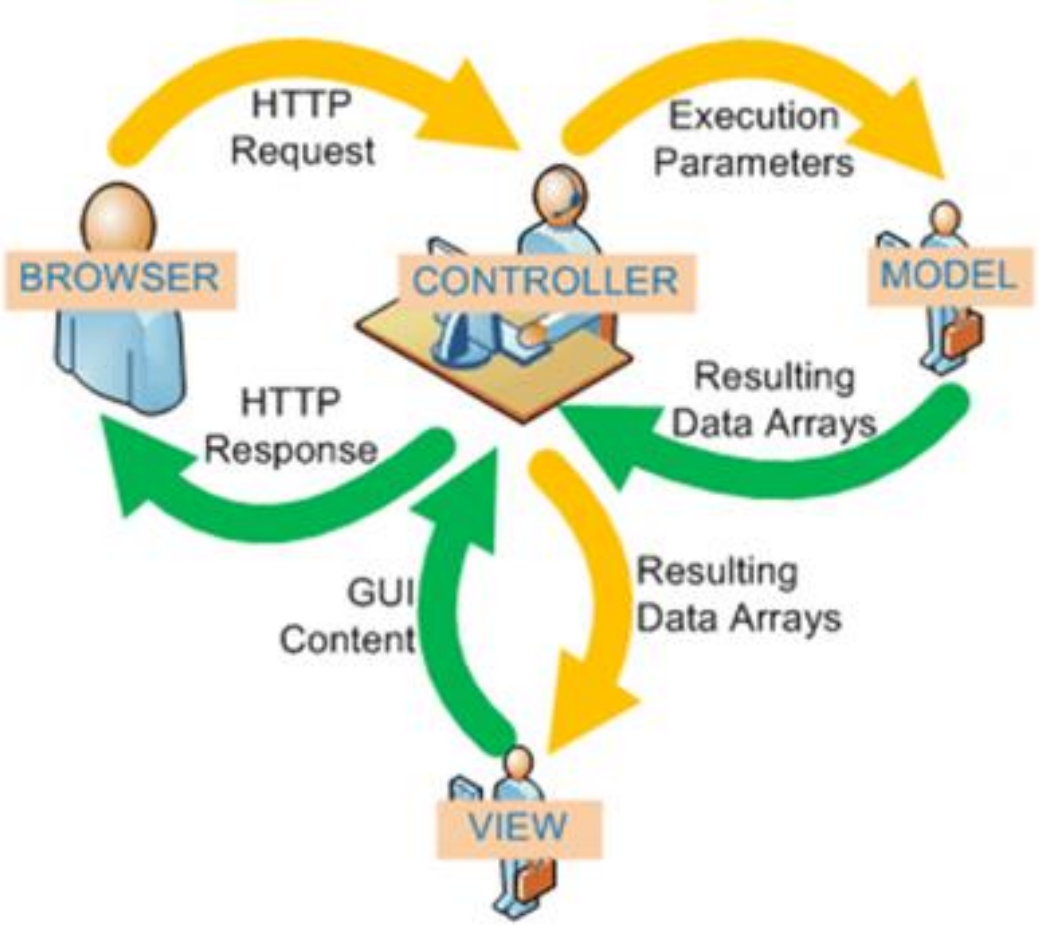
Angular JS vs IONIC

Angular JS vs ngCordova

Plantillas

¿ Qué es la arquitectura MVC ?

Model-View-Controller



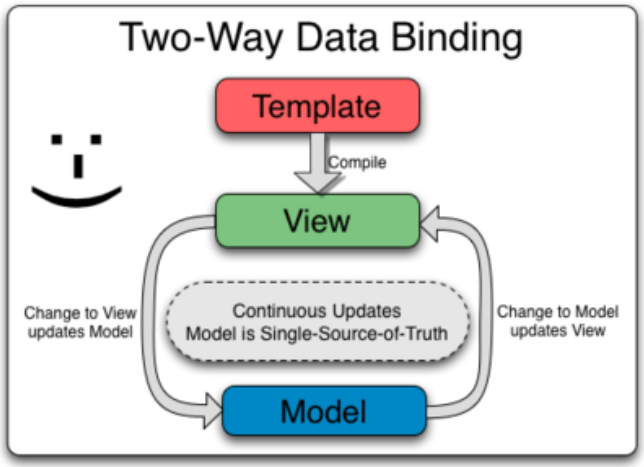
Angular JS vs …

2. Características Angular JS

- Framework popular mantenido por Google
- Sin componentes gráficos
- Ligero y eficiente (155kb)
- Sintaxis sencilla
- Coexistencia con otros frameworks sin problemas de incompatibilidad
- Data Binding
- Inyección de dependencias
- Directivas

Data Binding

Mediante el Data Binding Angular JS es capaz de transforma un plantilla estática en dinámica.



Data Binding

- Une dos elementos en tiempo real
- El cambio en los datos se reflejan en tiempo real en la vista
- No requiere eventos o funciones adicionales para su funcionamiento

Inyección de dependencias

Angular JS es modular. Se compone de una librería básica y otras extras especializadas.

Index of /1.5.3/

../		
docs/	25-Mar-2016 20:35	-
i18n/	25-Mar-2016 20:35	-
angular-1.5.3.zip	25-Mar-2016 20:35	10250035
angular-animate.js	25-Mar-2016 20:35	149421
angular-animate.min.js	25-Mar-2016 20:35	25367
angular-animate.min.js.map	25-Mar-2016 20:35	69998
angular-aria.js	25-Mar-2016 20:35	14958
angular-aria.min.js	25-Mar-2016 20:35	3807
angular-aria.min.js.map	25-Mar-2016 20:35	8409
angular-cookies.js	25-Mar-2016 20:35	9750
angular-cookies.min.js	25-Mar-2016 20:35	1444
angular-cookies.min.js.map	25-Mar-2016 20:35	3396
angular-csp.css	25-Mar-2016 20:35	343
angular-loader.js	25-Mar-2016 20:35	17202
angular-loader.min.js	25-Mar-2016 20:35	1729
angular-loader.min.js.map	25-Mar-2016 20:35	3838
angular-message-format.js	25-Mar-2016 20:35	37844
angular-message-format.min.js	25-Mar-2016 20:35	10306
angular-message-format.min.js.map	25-Mar-2016 20:35	28130
angular-messages.js	25-Mar-2016 20:35	27840
angular-messages.min.js	25-Mar-2016 20:35	2911
angular-messages.min.js.map	25-Mar-2016 20:35	7736
angular-mocks.js	25-Mar-2016 20:35	103355
angular-resource.js	25-Mar-2016 20:35	31396
angular-resource.min.js	25-Mar-2016 20:35	4486
angular-resource.min.js.map	25-Mar-2016 20:35	10975
angular-route.js	25-Mar-2016 20:35	37889
angular-route.min.js	25-Mar-2016 20:35	4582
angular-route.min.js.map	25-Mar-2016 20:35	11513
angular-sanitize.js	25-Mar-2016 20:35	25655
angular-sanitize.min.js	25-Mar-2016 20:35	5828
angular-sanitize.min.js.map	25-Mar-2016 20:35	9886
angular-scenario.js	25-Mar-2016 20:35	1458347
angular-touch.js	25-Mar-2016 20:35	26523
angular-touch.min.js	25-Mar-2016 20:35	3942
angular-touch.min.js.map	25-Mar-2016 20:35	10877
angular.js	25-Mar-2016 20:35	1141759
angular.min.js	25-Mar-2016 20:35	155877
angular.min.js.map	25-Mar-2016 20:35	421029
errors.json	25-Mar-2016 20:35	9444
version.json	25-Mar-2016 20:35	307
version.txt	25-Mar-2016 20:35	5

Módulos

- Código fácil de mantener
- Aplicativos más ligeros y con mayor rendimiento
- Módulos a medida

3. Creando un Proyecto con Angular JS

- Instalación y descarga de Angular JS
- Configurar mi Aplicación Web utilizando ng-app
- Inicializar la Aplicación Web mediante la sentencia javascript *angular.module*
- Crear las primeras expresiones {{ }}
- Crear la primera plantilla con expresiones dinámicas y configurar un controlador ng-controller
- Utilización del objeto *Scope*
- Utilización de la directiva *ng-model* y *ng-click*

Laboratorio 1

Vamos a practicar como crear una aplicación y un controlador de Angular JS y el uso de expresiones. Utiliza las directivas *ng-app*, *ng-controller* y *ng-model*.

Instrucciones para realizar el Laboratorio

- Completa el código Javascript
- Completa el código HTML

Ficheros para realizar el Laboratorio

- Descarga de la URL:

https://github.com/borjacabeza/Curso-Angular-JS/blob/master/Laboratorios/Lab1/index.html

Tiempo Estimado: 20 minutos

4. Filtros en expresiones

- Sirven para dar formato a textos y número
- Sirven para buscar entre los datos manipulados
- Sirven para ordenar los datos manipulados
- Se pueden combinar hasta obtener el resultado deseado
- Se pueden crear filtros personalizados

Formato HTML: *{{ expresión | filtro | filtro }}*

Formato Javascript: *$filter('filtro')()*

Filtros para formatear contenido

- Usamos **lowercase** y **uppercase** para transformar alfanuméricos:

```
<p>{{ t1 | uppercase }}</p>


$scope.t1 = “Titulo de la Página";
$scope.t2 = $filter(‘uppercase’)($scope.t1);
```

- Usamos **currency** para la representación de divisas.
- Usamos **number** para la representación de números formateando su parte decimal.

Filtros para formatear fechas y objetos

- Usamos **date** para formatear la representación de fechas:

```
<p>{{ fecha | date : format : timezone }}</p>


$scope.f1 = new Date();
$scope.f2 = $filter('date')($scope.f1, 'dd/MM/yyyy');
$scope.f3 = $filter('date')(new Date(), 'hh:mm:ss');
$scope.f4 = $filter('date')(new Date(), 'medium');
```

- Usamos **json** para serializar un objeto javascript a JSON

Filtros para representación de elementos

- Usamos **limitTo** para determinar el número de elementos que se representan:

```
<p>{{ expresión | limitTo : ítems : comienzo }}</p>


$scope.r1 = $filter('limitTo')('abcdef', 4);
$scope.r2 = $filter('limitTo')([8, 9, 1, 0], 3, 1);
```

Filtros personalizados

- Creamos un nuevo filtro personalizado utilizando el método *filter*.

```
var miApp = angular.module('filtros', []);

miApp.filter('capitalize' function () {
        var temp = function(datos, arg) {

                //Escribe el código de tu filtro

        };

        return temp;
});
```

Laboratorio 2

Vamos a practicar como crear filtro personalizado en Angular JS. Utiliza el metodo *filter*.

Instrucciones para realizar el Laboratorio

- Completa el código Javascript
- Utiliza tres parámetros. Uno para datos y dos para argumentos
- Controla el valor *undefined* de los parámetros

Ficheros para realizar el Laboratorio

- Descarga de la URL:

https://github.com/borjacabeza/Curso-Angular-JS/blob/master/Laboratorios/Lab2/index.html

Tiempo Estimado: 30 minutos

Filtros para búsqueda y filtrado de datos

- Usamos **filter** para la búsqueda o filtrado de los datos.
- Mediante data binding hacemos posible que sea en tiempo real.

```
{{ expresión | filter : expresión }}
```

- Utilizaremos la directiva *ng-repeat* para mostrar colecciones que posteriormente filtraremos

Filtros para ordenar de datos

- Usamos **orderBy** para ordenar los datos.
- Mediante data binding hacemos posible que sea en tiempo real.

```
{{ expresión | orderBy : expresión : reverse }}
```

- Utilizaremos la directiva *ng-repeat* para mostrar colecciones que posteriormente filtraremos

5. Controladores de Angular JS

- Un aplicación puede tener uno o varios controladores
- Los controladores mantienen el principio de encapsulamiento frente a otros controladores de la aplicación
- Creamos un controlador con el método *controller*
- Activamos un controlador con la directiva *ng-controller*
- El objeto *$scope* hace de enlace entre los controladores y las vistas
- Todos los controladores comparten el objeto *$rootScope*

Laboratorio 3

Vamos a practicar una aplicación con dos controladores y los objetos *$scope* y *$rootScope*.

Instrucciones para realizar el Laboratorio

- Completa el código Javascript
- Añade las operaciones al array controlado por el *$rootScope*
- Muestra los datos del array en el segundo controlado usando *ng-repeat*

Ficheros para realizar el Laboratorio

- Descarga de la URL:

https://github.com/borjacabeza/Curso-Angular-JS/blob/master/Laboratorios/Lab3/index.html

Tiempo Estimado: 30 minutos

6. Gestión de eventos Angular JS

- Los eventos en Angular JS son controlados mediante directivas.
- Los eventos en Angular JS se enlazan con funciones definidas en el objeto *$scope* del controlador

Directivas para gestión eventos

ng-click: Cuando se produce el evento click.
ng-dblclick: Cuando se produce el evento doble click.
ng-blur: Cuando se produce el evento blur.
ng-change: Cuando se cambia el contenido de un <INPUT> o <SELECT> pero no cuando se cambia a consecuencias del propio modelo.
ng-cut: Cuando se produce el evento cut.

ng-keydown: Cuando se produce el evento keydown.
ng-keyup: Cuando se produce el evento keyup.
ng-keypress: Cuando se produce el evento keypress.

ng-mousedown: Cuando se produce el evento mousedown.
ng-mouseenter: Cuando se produce el evento mouseenter.
ng-mouseleave: Cuando se produce el evento mouseleave.
ng-mousemove: Cuando se produce el evento mousemove.
ng-mouseover: Cuando se produce el evento mouseover.
ng-mouseup: Cuando se produce el evento mouseup.

Laboratorio 4

Vamos a practicar una aplicación la gestión de diferentes eventos.

Instrucciones para realizar el Laboratorio

- Completa el código Javascript
- Añade las directivas necesarias para controlar los eventos

Ficheros para realizar el Laboratorio

- Descarga de la URL:

https://github.com/borjacabeza/Curso-Angular-JS/blob/master/Laboratorios/Lab4/index.html

Tiempo Estimado: 30 minutos

7. Directivas de Angular JS

- Son atributos insertados en las etiquetas HTML que habilitan conductas complejas para la etiqueta y su contenido
- Para pasar las validación de W3C añadimos el prefijo *data-* al nombre de las directivas ya que realmente se trata de atributos personalizado de HTML

  http://validator.w3.org/#validate_by_input

- La directiva *ng-bind* cumple la misma función que *{{ }}*
- Podemos crear directivas personalizadas y definir como se deben utiliza, *A*tributo, *E*tiqueta, *C*lase o *M*Comentario.

Directivas más comunes en Angular JS

- Usamos *ng-app* para declarar una aplicación.
- Usamos *ng-controller* para definir el trozo de HTML afectado o contralo por un controlador
- Usamos *ng-model* para enlazar elementos del modelo con <INPUT> o elementos de formulario
- Usamos *ng-bind* para indicar contenidos de las etiquetas HTML, igual que *{{ }}*
- Usamos *ng-repeat* para repetir código HTML en función del contenido de un array de elementos
- Usamos *ng-show* para mostrar o ocultar elementos

Directivas más comunes en Angular JS II

- Usamos *ng-disable* y *ng-readonly* para habilitar o deshabilitar elementos HTML
- Usamos *ng-maxlength* y *ng-minlength* para el número de caracteres de un <INPUT>
- Usamos *ng-pattern* para indicar expresiones regulares de un <INPUT>
- Usamos *ng-true-value* y *ng-false-value* para especificar valores en los <INPUT> de tipo *checked*
- Usamos *ng-options* para rellenar un <SELECT>

Directivas más comunes en Angular JS III

- Usamos *ng-if* muestra un elemento si es *true* o no lo muestra si el *false*
- Usamos *ng-switch* determinar que conjunto de código HTML se muestra en función de un valor

8. Acceso a datos remotos con Angular JS

- Usamos el objeto *$http* para realizar comunicaciones remotas sobre HTTP.
- El servicio *$http* utiliza los objetos nativo de Javascript XMLHttpRequest y JSONP
- Admite diferentes acciones cada una de ellas representada en un método

  $http.get() *$http.post()* *$http.put()* *$http.delete()*

  $http.jsonp() *$http.head()* *$http.patch()*

- El uso del objeto *$http* sigue el patrón de promesas.

$http.get()

- Usamos **$http.get()** para enviar y recibir datos con servidores remotos

```
$http.get(url).success(function (datos) {
        ...
});

$http.get(url).then(function (datos) {
        ...
});
```

$http.post()

- Usamos **$http.post()** para enviar y recibir datos con servidores remotos

```
$http.post(url, datos).success(function (datos) {
        ...
});

$http.post(url, datos).then(function (datos) {
        ...
});
```

$http.jsonp()

- Usamos **$http.jsonp()** para enviar y recibir datos con servidores remotos saltando las restricciones de la política same-origin
- Usamos un parámetro *callback* con el valor *JSON_CALLBACK*

http://example.com/url.json?callback=JSON_CALLBACK

```
$http.jsonp(url).success(function (datos) {
        ...
});

$http.jsonp(url).then(function (datos) {
        ...
});
```

Laboratorio 5

Vamos a practicar el método POST del servicio $http.

Instrucciones para realizar el Laboratorio

- Completa el código Javascript
- Añade las expresiones necesarias

Ficheros para realizar el Laboratorio

- Descarga de la URL:

https://github.com/borjacabeza/Curso-Angular-JS/blob/master/Laboratorios/Lab5/index.html

Tiempo Estimado: 30 minutos

9. Plantillas y Rutas con Angular JS

- El *routing* es posiblemente una de las herramientas más poderosas de Angular JS
- Podemos desarrollar aplicaciones de una única página que muestre diferentes vistas en función de la url del navegador
- El *routing* mejora los tiempos de carga y reduce el uso de ancho de banda
- El *routing* favorece la indexación de nuestra aplicación

http://ejemplo.com/index.html

http://ejemplo.com/index.html#/seccion1

http://ejemplo.com/index.html#/seccion2

Instalación de ngRoute

Inyectaremos en nuestra aplicación el modulo ngRoute para habilitar el recurso de *routing* en la misma

Instalación de ngRoute

- Añadir el código javascript de modulo
- Inyectamos la dependencia al crear el módulo cuando

```
<script src="angular-route.min.js"></script>

var miApp = angular.module('miApp', ['ngRoute']);
```

Creando un Proyecto con Rutas

- Instalación de ngRoute
- Inyectar dependencias en la declaración de modulo
- Utilizamos el método *config* de la variable aplicación para configurar las rutas
- Utilizamos la directiva ng-app para determinar el alcance de la aplicación
- Utilizamos la directiva *ng-view* para determinar donde se presentas las vistas de cada ruta
- Podemos definir rutas con parámetros. Los parámetros son accesibles mediante el objeto *$routeParams*
- El objeto *$location* nos permite modificar la ruta desde Javascript

Laboratorio 6

Vamos a practicar con las plantillas y las rutas en Angular JS.

Instrucciones para realizar el Laboratorio

- Completa el código Javascript
- Añade las expresiones necesarias

Ficheros para realizar el Laboratorio

- Descarga de la URL:

https://github.com/borjacabeza/Curso-Angular-JS/blob/master/Laboratorios/Lab6/index.html

Tiempo Estimado: 30 minutos

10. Factorías de Angular JS

- Las factorías son contenedores de código Javascript.
- Las factorías se instancia una única vez dentro de la aplicación estado.
- Las factorías se puede utilizar en varios controladores compartiendo la información y funcionalidad programada.

```
miApp.factory('name', function () {
        return { ... }
});

miApp.controller('ctrl', function(name) {

});
```

11. Servicios de Angular JS

- Los servicios son contenedores de código Javascript.
- Los servicios se instancia una única vez dentro de la aplicación.
- Los servicios están pensado para compartir funcionalidad Javascript entre todos los elementos de la aplicación.

```
miApp.service('name', function () {
        this.property = function() {}
        ...
});

miApp.controller('ctrl', function(name) {

});
```

12. Animaciones con Angular JS

- El recurso de animaciones permite definir animaciones que se relacionada con eventos que producen algunas directivas.
- La tabla de directivas y eventos son:

Animación	Directivas	Class Inicial	Class Final
enter	ngIf, ngInclude, ngRepeat, ngSwitch, ngView	.ng-enter	.ng-enter-active
leave	ngIf, ngInclude, ngRepeat, ngSwitch, ngView	.ng-leave	.ng-leave-active
move	ngRepeat	.ng-move	.ng-move-active
add	ngClass, ngShow, ngHide, ngModel	.ng-add	.ng-add-active
remove	ngClass, ngShow, ngHide, ngModel	.ng-remove	.ng-remove-active

- Podemos usar la ayuda online *http://nganimate.org*

Instalación de ngAnimate

Inyectaremos en nuestra aplicación el modulo ngAnimate para habilitar el recurso de *animaciones* en la misma

Instalación de ngAnimate

- Añadir el código javascript de modulo
- Inyectamos la dependencia al crear el módulo cuando

```
<script src="angular-animate.min.js"></script>

var miApp = angular.module('miApp', ['ngAnimate']);
```

Curso de Angular JS.

Gracias.

Recuerda **Angular JS** es un framework MVC de código abierto desarrollado por Google y escrito en Javascript.

Cuando programes **No trates de Inventar la rueda,** utiliza los recursos Angular JS.

Referencias:
http://angularjs.org
http://nganimate.org