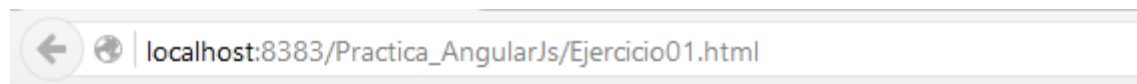


Práctica AngularJS

Ejercicio 1

Desarrolla una página web equivalente que permita introducir los campos de un jugador de baloncesto (te puedes basar en los atributos que hemos utilizado con Spring), y muestre en tiempo real el valor introducido. Explica, de manera muy resumida, el mecanismo que utiliza AngularJS para sincronizar los elementos de la interfaz de usuario HTML (explica la directiva específica que se utiliza en este ejemplo).



Introduce los datos del jugador:

Nombre :

Apellidos :

País :

Posición en el campo :

Nombre y apellidos del jugador : David Casaoliva Rey

País de origen del jugador : España

Posición en el campo : Pibot

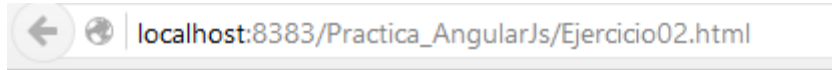
Explicación

En este ejercicio se utiliza la propiedad ng-model, lo que permite, es que a la vez que se está introduciendo un nuevo valor instantáneamente está actualizando el controlador. Al escribir el nombre del jugador a la vez se está actualizando el controlador que lo muestra en la parte de debajo de la pantalla.

Ejercicio 2

Extiende el ejercicio número 1, de manera que se incluya un controlador que gestione los datos de la vista. El controlador ha de proporcionar unos datos iniciales por defecto que se mostrarán en la vista. Cuando el usuario actualiza los elementos de la vista (los elementos de la interfaz de usuario en HTML), ¿se actualizan automáticamente los datos en el controlador? Justifica tu respuesta.

Campos con datos por defecto:



Cambia los datos por defecto del jugador.

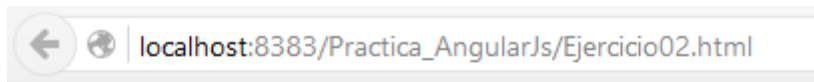
Nombre:
Apellidos:

Nombre completo: David Casaoliva Rey

Explicación:

Con las funciones ng-app y ng-controller, se gestiona el mostrar los valores predeterminados en los campos donde podemos modificarlos. La función ng-app nos permite crear un controlador con el scope y poder definir valores por defecto para los campos que queremos modificar.

Campos con modificaciones de los datos por defecto en los inputs:



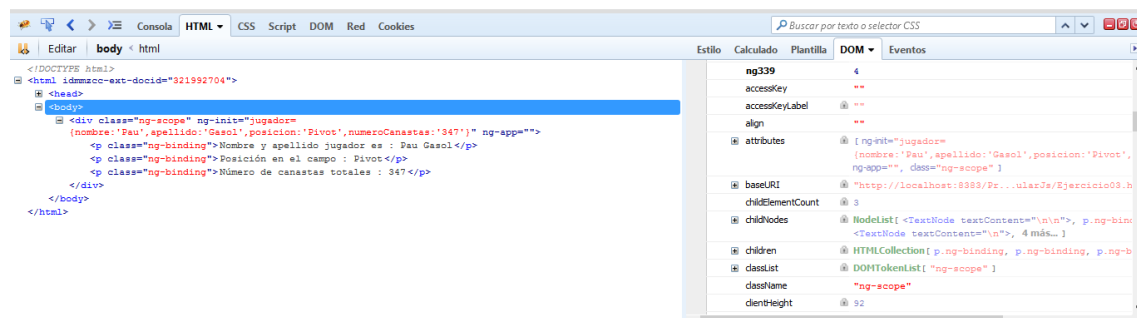
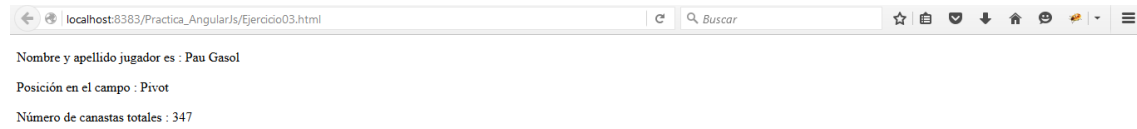
Cambia los datos por defecto del jugador.

Nombre:
Apellidos:

Nombre completo: Pepe Perez Rey

Ejercicio 3

Utilizando la directiva ng-init, inicializa un objeto jugador de baloncesto con sus diversos atributos. Posteriormente, muestra sus atributos en diversos elementos HTML mediante la directiva {{}}. Explica en qué componentes de AngularJS se debe manipular el DOM: directivas, controladores, vistas o servicios. Justifica tu respuesta.



Explicación

Permite evaluar una expresión en el scope, creando el objeto jugador i definiendo sus atributos. Al examinar el DOM, comprobamos que están los datos del objeto cargados y en el scope.

Ejercicio 4

Inicializa un array para almacenar diversos jugadores de baloncesto. Posteriormente, utiliza la directiva ng-repeat para mostrar todos los jugadores de baloncesto con sus diversos atributos. Explica cómo funciona la gestión de scopes con la directiva ng-repeat.

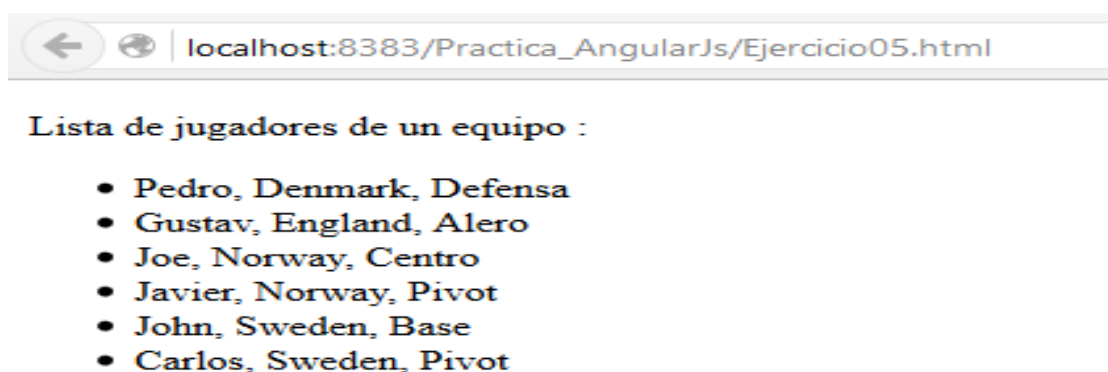


Explicación:

La directiva ng-repeat nos permite mostrar los objetos contenidos en una variable. Al definir una serie de objetos, en este caso jugadores, podemos ir mostrando a usar esta directiva combinada en un li, mostramos una lista con los atributos que hemos guardado de cada jugador dentro de la variable jugadores.

Ejercicio 5:

Inicializa un array que almacene diversos jugadores de baloncesto en un controlador. Posteriormente, utiliza la directiva ng-repeat para mostrar todos los jugadores. Utiliza también un filtro, para mostrar los jugadores ordenando mediante varios atributos (por ejemplo, nombre, total canastas, total asistencias, país, etc.). La idea es que puedes experimentar con diversos atributos para comprobar que la ordenación funciona correctamente con todos los tipos de datos. Justifica en qué casos es preferible realizar la ordenación en el cliente mediante AngularJS, y en qué casos es preferible realizar la ordenación en el servidor mediante Spring. Explica ventajas e inconvenientes de cada enfoque.




Explicación:

La ordenación es más eficiente y segura en Spring, por qué el usuario no puede acceder directamente al scope y poderlo manipular. En cambio, en AngularJS es menos seguro y el usuario puede manipular los resultados. Es mejor en Spring por el tema de la seguridad y la eficiencia. El ejemplo está ordenado por país.

Ejercicio 6:

Repaso sobre la sincronización automática que realiza angular entre la vista, el controlador y el modelo. Experimenta con el siguiente código de ejemplo, comprobando la sincronización que se efectúa con los atributos de un jugador de baloncesto.

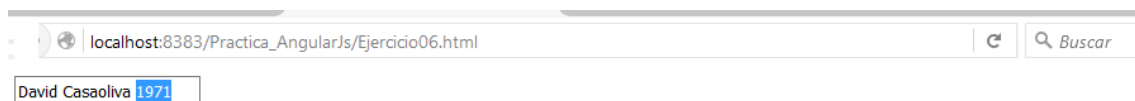


A screenshot of a web browser window. The address bar shows 'localhost:8383/Practica_AngularJs/Ejercicio06.html'. Below the address bar, there is a text input field containing the name 'David Casaoliva'.

El nombre del jugador es : David Casaoliva

When you change the name in the input field, the changes will affect the model, and it will also affect the name property in the controller.

Modificación del input:



A screenshot of a web browser window, similar to the previous one. The text input field now contains 'David Casaoliva 1971', with '1971' highlighted in blue.

El nombre del jugador es : David Casaoliva 1971

When you change the name in the input field, the changes will affect the model, and it will also affect the name property in the controller.

Explicación:

Con ng-controller, conseguimos que se actualice la salida directamente con lo que introducimos en el input, en este caso, al incorporar al nombre del jugador ya guardado en el scope con un número.

Ejercicio 7:

Con JHipster y Spring, generamos un listado de equipos de la base de datos.

- FC Barcelona, Barcelona, 1895-09-08
- Bayer Munich, Munich, 1923-05-20
- Real Madrid, Madrid, 1901-03-25
- Paris Saint Germaine, Paris, 1956-09-12
- Real Asturia FC, Oviedo, 1961-12-31

Explicación:

Generamos un controler (equiposCtrl), que nos permite conectar con la tabla de equipos y a partir de la función ng-repeat obtenemos un listado de los equipos.

Ejercicio 9

Extiende el ejercicio número 7 para obtener el listado de jugadores del API REST Spring. En este ejercicio, debes mostrar la información de los jugadores en formato tabla, especificando una columna para cada atributo del jugador.

FC Barcelona	Barcelona	1895-09-08
Bayer Munich	Munich	1923-05-20
Real Madrid	Madrid	1901-03-25
Paris Sant Germane	Paris	1956-09-12
Real Asturia FC	Oviedo	1961-12-31

Explicación: en este ejercicio se muestra en modo tabla el contenido de equipos.

Ejercicio10:

Mejora la tabla del ejercicio anterior utilizando estilos CSS. Te puedes basar en el código de ejemplo proporcionado, y puedes proponer tus propios estilos.

FC Barcelona	Barcelona	1895-09-08
Bayer Munich	Munich	1923-05-20
Real Madrid	Madrid	1901-03-25
Paris Sant Germane	Paris	1956-09-12
Real Asturia FC	Oviedo	1961-12-31

Ejercicio11:

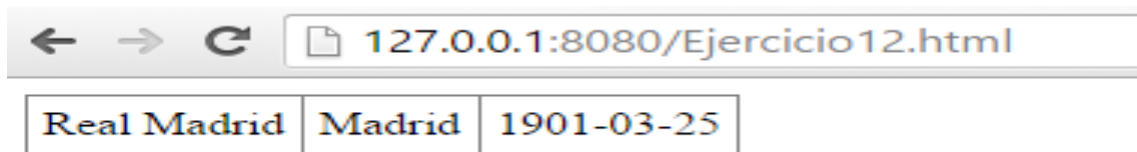
En este ejercicio, debes utilizar la nueva directiva ng-if para aplicar diversos estilos a las columnas en función de que la fila sea par o impar.

FC Barcelona	Barcelona	1895-09-08
Bayer Munich	Munich	1923-05-20
Real Madrid	Madrid	1901-03-25
Paris Sant Germane	Paris	1956-09-12
Real Asturia FC	Oviedo	1961-12-31

Explicación: con la directiva ng-if asignamos de manera par o impar el CSS en la tabla.

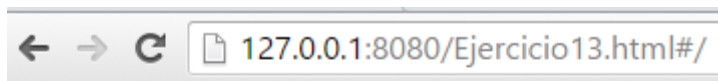
Ejercicio12:

Con ng-show podemos hacer que de un contenido que se va a mostrar por pantalla, es el que cumple la condición que hemos puesto en la directiva ng-show. En este ejemplo, he puesto que se muestren aquellos equipos que su localidad sea Madrid.



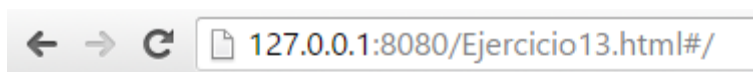
Ejercicio 13:

Con la opción filter en ng-repeat, nos permite filtrar la lista de equipos a partir de los caracteres que vamos escribiendo en el input y en la lista van quedando los nombres que va coincidiendo.



Introduce una letra para filtrar:

- FC Barcelona, Barcelona
- Bayer Munich, Munich
- Real Madrid, Madrid
- Paris Saint Germaine, Paris
- Real Asturia FC, Oviedo
- Lugo Club de Futbol, Lugo
- FC Valencia, Valencia
- FC Alicante, Alicante

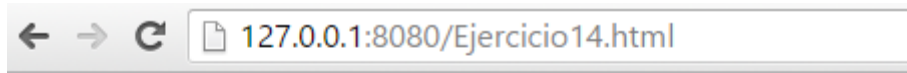


Introduce una letra para filtrar:

- FC Barcelona, Barcelona
- Bayer Munich, Munich
- Lugo Club de Futbol, Lugo

Ejercicio 14:

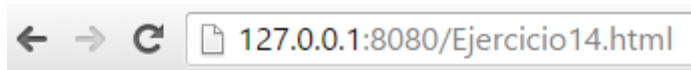
Con la directiva ng-click y la función orderorderByMe de Srping, podemos hacer que al hacer clic en la cabecera de cada columna, esta se ordene alfabéticamente en este caso.



Pulsa los headers para ordenar los campos

Nombre equipo	Localidad
FC Barcelona	Barcelona
Real Madrid	Madrid
Bayer Munich	Munich
Real Asturia FC	Oviedo
Paris Sant Germane	Paris

Ordenamos por nombre equipo:

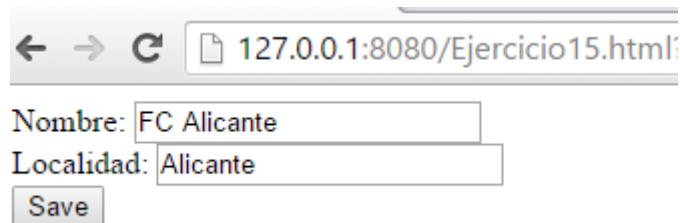


Pulsa los headers para ordenar los campos

Nombre equipo	Localidad
Bayer Munich	Munich
FC Barcelona	Barcelona
Paris Sant Germane	Paris
Real Asturia FC	Oviedo
Real Madrid	Madrid

Ejercicio 15:

Esta directiva nos permite crear dos campos input que nos permiten crear nuevos registros de equipos en la base de datos. He definido los protocolos de seguridad e incorporado las directivas necesarias de Spring generadas con JHipster para poder crear registros nuevos en la base de datos desde esta directiva.



Nombre:

Localidad:



127.0.0.1:8080/Ejercicio15.html

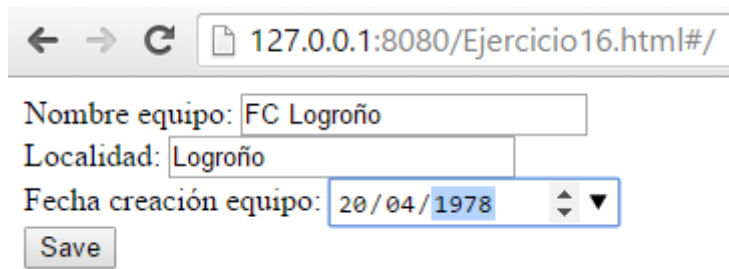
Pulsa los headers para ordenar los campos

Nombre equipo	Localidad
FC Barcelona	Barcelona
Bayer Munich	Munich
Real Madrid	Madrid
Paris Saint Germaine	Paris
Real Asturia FC	Oviedo
Lugo Club de Futbol	Lugo
FC Valencia	Valencia
FC Alicante	Alicante

Ejercicio 16:

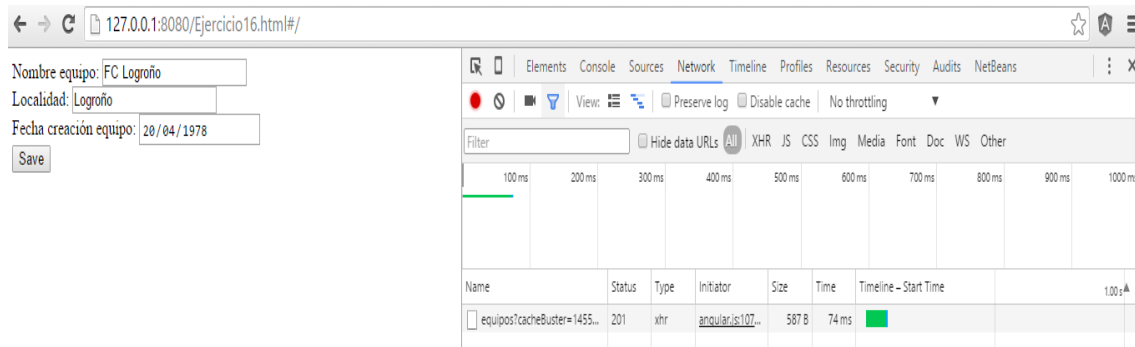
En este ejercicio se trata de poder crear nuevos equipos en la base de datos y comprobar que se crean en la tabla equipos. Primero repasé el código y lo adapté para poder registrar los equipos en mi base de datos. Después al comprobar el código, encontré que el `<script src="scripts/components/util/data-util.service.js"></script>` estaba dos veces y generaba un error. Al eliminar una de las líneas duplicadas, ejecuté el código, realicé una alta nueva de un equipo.

Introduzco un equipo nuevo:



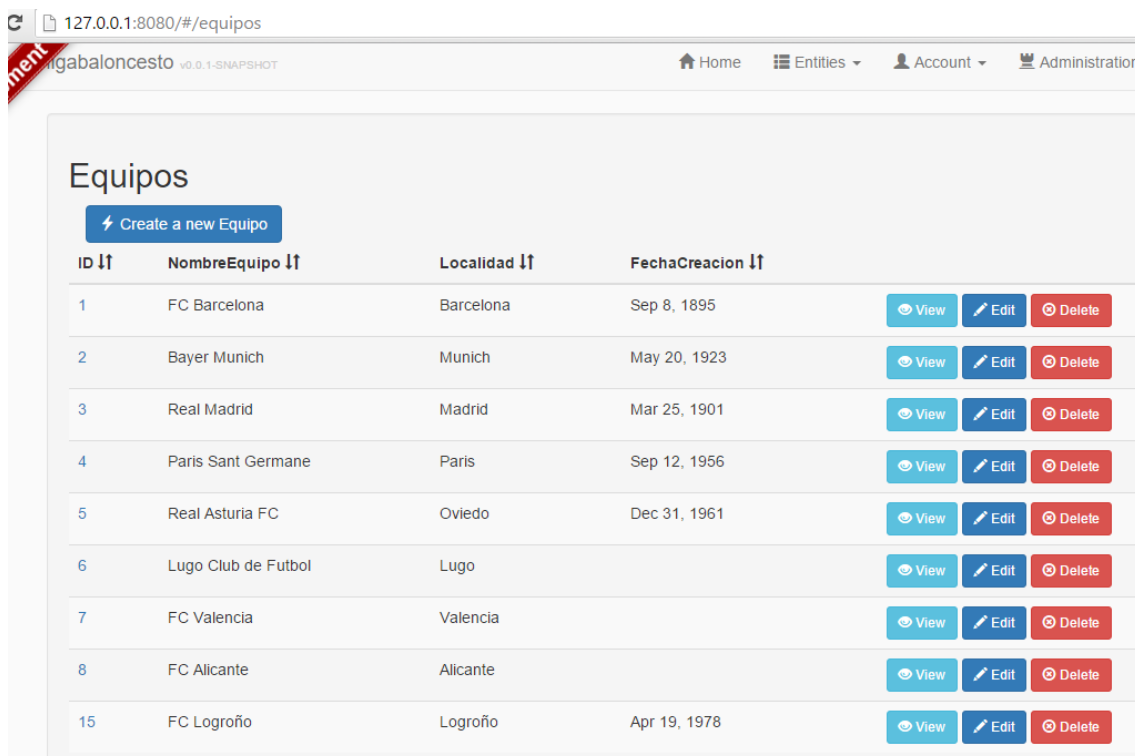
A web browser window showing a form to add a new team. The form has three input fields: 'Nombre equipo:' with the value 'FC Logroño', 'Localidad:' with the value 'Logroño', and 'Fecha creación equipo:' with a date picker set to '20/04/1978'. Below the fields is a 'Save' button.

Cuando lo grabo, compruebo que ha comunicación con la base de datos y genera trafico de datos hacia ella, enviando la información a la tabla equipos.



A screenshot of a web browser with the Chrome DevTools Network tab open. The browser address bar shows '127.0.0.1:8080/Ejercicio16.html#/'. The form from the previous image is visible in the background. The Network tab shows a list of requests, with one request selected: 'equipos?cacheBuster=1455...'. The request details show a status of 201, type of xhr, and a size of 587 B. The timeline shows the request was made at 74 ms.

Entro en el proyecto y compruebo que se ha creado la nueva entrada.



A screenshot of a web application showing a list of teams. The page has a header with navigation links: Home, Entities, Account, and Administration. The main content area is titled 'Equipos' and contains a table with the following columns: ID, NombreEquipo, Localidad, and FechaCreacion. The table lists 15 teams, with the last team being 'FC Logroño' created on 'Apr 19, 1978'. Each row has 'View', 'Edit', and 'Delete' buttons.

ID	NombreEquipo	Localidad	FechaCreacion	
1	FC Barcelona	Barcelona	Sep 8, 1895	View Edit Delete
2	Bayer Munich	Munich	May 20, 1923	View Edit Delete
3	Real Madrid	Madrid	Mar 25, 1901	View Edit Delete
4	Paris Sant Germaine	Paris	Sep 12, 1956	View Edit Delete
5	Real Asturia FC	Oviedo	Dec 31, 1961	View Edit Delete
6	Lugo Club de Futbol	Lugo		View Edit Delete
7	FC Valencia	Valencia		View Edit Delete
8	FC Alicante	Alicante		View Edit Delete
15	FC Logroño	Logroño	Apr 19, 1978	View Edit Delete