Serveis

¿Qué es un servei?

És un objecte JavaScript que ens permet obtenir informació. Aparentment res de nou que entendre, seria por exemple un DAO en Java o un serveis de Java. El important d'això és que un servei mai interacciona amb la pròpia pàgina, només amb altres serveis o com un servidor de dades que pugui estar en un altre Host.

Exemples de serveis seran:

El servei $http de AngularJS. Aquest servei fa la típica trucada a AJAX a un servidor per obtenir informació d'ell. Com veiem, compleix perfectament la definició d'obtenir informació.
Un servei que es connecta a un host que ens retorna el valor del Euríbor.
Un possible servei de càlcul de Hipoteca que les dades d'una hipoteca (Import del préstec, anys, diferencial ,etc) ens calcularà quant hi ha de pagar mensualment.
Un servei que ens fes les operacions de CRUD sobre el servidor.
Un servei que transformarà els String amb una data en un objecte Date.
El servei $log d'Angular que ens permet generar un log de la nostra aplicació.
Etc.

Registrar serveis

Els serveis es registren en els mòduls, via el Module API. Aqui tenim un exemple d'utilització del Modul factory API per a registrar un servei:

```
var meuModul = angular.module('meuModul', []);
meuModul.factory('serviceId', function() {
  var ejempleNouServei;
  // factory function body that constructs ejempleNouServei
  return ejempleNouServei;
});
```

S'ha de tenir en compte que no està registrant una instància del Servei, sinó una funció de Factory que crearà aquesta instància quan el cridi.

Utilitzar serveis

La primera característica dels serveis és que tenen un nom (al igual que els controladors) i com ja vam dir en Espai de noms aquest nom és independent del mòdul en el que van ser afegits. Així que per utilitzar un servei només necessitem saber el seu nom.

Un servei senzill d'AngularJS és el servei de $log. Aquest servei simplement truca a console.log però ens abstrau per si no existeix el objecte console. $log disposa de diversos mètodes però per ara ens

quedarem amb el mètode “debug” que permet passar-li un missatge per a que es mostri per la consola del navegador.

Continuant ara amb el nostre exemple del segur mèdic anem a afegir un missatge de log quan es configura el $scope.

Cóm obtenim una instància del servei de $log en el controlador? Hem afegir un nou paràmetre a la nostre funció del controlador amb el nom del servei, que com ja hem dit en el nostre cas és $log. Si tinguéssim que sol·licitar més serveis s’afegirien com més paràmetres.

```
var app=angular.module("app",[]);

app.controller("SeguroController",function($scope,$log) {
 $scope.seguro={
  nif:"",
  nombre:"",
  ape1:"",
  edad:undefined,
  sexo:"",
  casado:false,
  numHijos:undefined,
  embarazada:false,
  coberturas: {
   oftalmologia:false,
   dental:false,
   fecundacionInVitro:false
  },
  enfermedades:{
   corazon:false,
   estomacal:false,
   rinyones:false,
   alergia:false,
   nombreAlergia:""
  },
  fechaCreacion:new Date()
 }

 $log.debug("Acabamos de crear el $scope");

});
```

Línea 3: Veiem que la funció del controlador ara té 2 paràmetres, el $scope i el $log.
Línea 28: Utilitzarem el objecte $log trucant al mètode debug.
Si mostrarem la consola de chrome (CTRL+SHIFT+J) veiem el missatge de “Acabamos de crear el $scope”

És a dir que en comptes de demanar nosaltres les instàncies dels serveis, s'afegiran com paràmetres de la nostre funció del controlador i quan AngularJS truqui al controlador ens injectarà les instàncies dels serveis que hem sol·licitat.

Veure Ejercicio1.html i Ejercicio1.js.

Opcions de serveis

ELs noms dels serveis tenen que seguir la mateixa nomenclatura que els noms de les variables. És a dir que la primera paraula té que ser en minúscula i la resta de paraules tenen que començar en majúscula (veure CamelCase).

Perquè tenen seguir la mateixa norma que les variables? Perquè quant injectem un servei ho fem sobre una variable que es diu com el servei per tant el servei es té que cridar com les variables.

Serveis propis de Angular
En angular els serveis i altres artefactes comencen sempre els seus noms per "$". La raó d'això simplement és evitar que xoqui el nom dels serveis d'Angular com els serveis creats per nosaltres mateixos o per terceres llibreries.

També hi ha noms en angular que comencen per "$$" . En aquest cas significa que són coses internes d'AngularJs i que no són públiques , per el que no haurien d'utilitzar-se ja que d'una versió a una altre podrien desaparèixer. Però en internet hi ha moltes vegades que per a fer coses que no permet angular es recorre a artefactes que comencen per "$$".

Serveis propis de AngularJS

$http

El servei $http permet fer peticions AJAX al servidor. És realment com el objecte XMLHttpRequest o el mètode ajax() de JQuery. La diferència amb aquests dos últims és que està integrat amb Angular com un servei (amb totes les avantatges d'ells comporta) però principalment perquè notifica a AngularJS que ha hagut un canvi en el model de JavaScript i actualitza la vista i la resta de dependències adequadament

El primer que tenim que fer és obtenir una referència al servei $http. Com ja hem explicat en el tema anterior, tenim que incluir-lo com paràmetre del nostre controlador per a que ens ho injecti al crear-ho.

$http accepta com paràmetre en un únic objecte anomenat config amb totes les propietats que necessita per a la petició.

Veiem ara alguna de las propietats:

method: El mètode HTTP para fer la petició. Els seus possibles valors són: GET, POST, PUT, DELETE, etc.
url: La URL de ón volem obtenir les dades.

data: Si utilitzem el mètode POST o PUT aquí posarem les dades a manar en el body de la petició HTTP
params: Un objecte que es posarà com a paràmetres de la URL.

Veure Ejercicio2.html, Ejercicio2.js i Ejercicio2.json.

$timeout

El servei de $timeout és com el mètode setTimeout() de JavaScript. La principal diferència , a part de que és un servei, és que al igual que 3.2 $http s'actualitza la vista al actualitzar el model desde $timeout.

La funció $timeout suporta 2 paràmetres:

fn: La funció a cridar quan s'acaba el timeout
tiempo: El temps en milisegons que té que passar perquè es cridi a la funció.

Veure Ejercici3.html i Ejercici3.js.

$interval

El servei de $interval és com el mètode window.setInterval() de JavaScript. Executa una funció cada cert temps.

La función $interval suporta 2 paràmetres:

fn: La funció a cridar quan s'acaba el timeout
tiempo: El temps en milisegons que té que passar per a que es cridi a la funció de forma cíclica.

Veure Ejercici4.html i Ejercici4.js.

Constant

Una `constant` és un servei al que li passarem directament el valor d'aquest servei. El seu principal característica és que es pot injectar en qualsevol lloc. Es defineix anomenant al mètode `constant` d'un mòdul. A dit mètode li passarem el nom de la constant i el seu valor.

Veure Ejercici6.html i Ejercici6.js.

Value

Un `value` és un servei al que li passem directament el valor de dit servei. Es defineix cridant al mètode `value` d'un mòdul. Aquest mètode li passarem el nom y el seu valor.

Veure Ejercici7.html i Ejercici7.js.

Service

En els dos tipus de serveis anteriors, els `constant` i els `value`. En els dos casos li indicàvem directament el valor que tenia que tenir el servei. Amb el tipus Service, li tenim que indicar una classe JavaScript i serà AngularJS el que generi internament una instància de la classe.

Veure Ejercici8.html i Ejercici8.js.

Serveis creats per nosaltres

Nosaltres també podem definir el nostres propis serveis i personalitzar-los, perquè abastin tasques que el propi AngularJS no ens serveix.

Veure Ejercici5.html i Ejercici5.js.