

Blog sobre Java EE

Estás aquí: [Inicio](#)/[JavaScript Frameworks](#)/[Angular](#)/Angular Modules y el uso de servicios

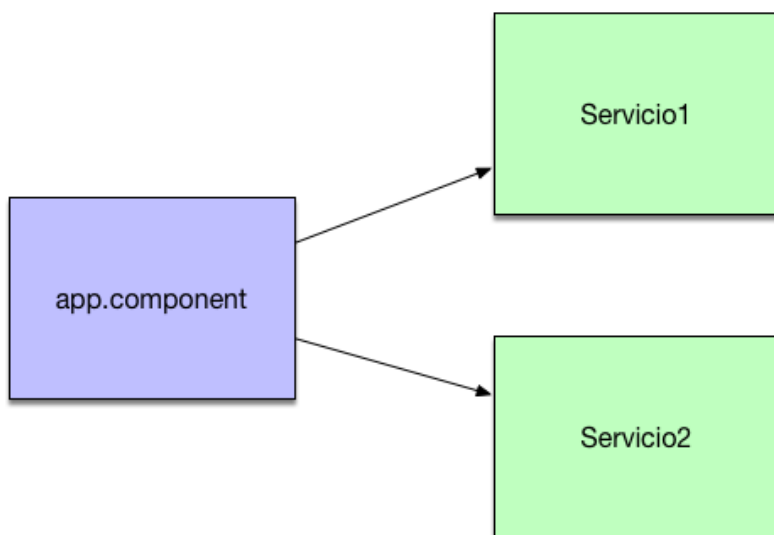
Angular Modules y el uso de servicios

9 abril, 2018 por [Cecilio Álvarez Caules](#) — [Deja un comentario](#)

Hosting para aplicaciones JAVA - Garantía de reembolso 30 días.

Incluye servidor de aplicaciones dedicado: Tomcat, Glassfish, Payara o WildFly. [anw.e](#)

El concepto de **Angular Modules** es un concepto cercano al de Java Archive de Java o DLL de .NET en el cual nosotros queremos definir un grupo de funcionalidad **reutilizable**. Vamos a construir un ejemplo sencillo de Angular Modules. Para ello vamos a partir de un componente de Angular que utiliza un par de servicios.



Abordemos el código:

```
1 import { Component } from '@angular/core';
2 import {Servicio1Service} from "../servicio1.service"
3 import {Servicio2Service} from "../servicio2.service"
4
5 @Component({
```

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)
[ACEPTAR](#)

```

16
17     constructor(s1:Servicio1Service,s2:Servicio2Service) {
18
19
20         this.mensaje1=s1.mensaje1();
21         this.mensaje2=s2.mensaje2();
22
23     }
24 }

```

Veamos lo que hace la plantilla :

```

1  {{mensaje1}}
2  {{mensaje2}}

```

Veamos lo que hacen los servicios :

```

1  import { Injectable } from '@angular/core';
2
3  @Injectable()
4  export class Servicio1Service {
5
6      constructor() { }
7
8      mensaje1() {
9
10         return "hola1";
11     }
12 }

```

```

1  import { Injectable } from '@angular/core';
2
3  @Injectable()
4  export class Servicio2Service {
5
6      constructor() { }
7
8      mensaje2() {
9
10         return "hola2"
11     }
12 }

```

Como podemos ver se trata de dos servicios sencillos . Para que Angular los pueda usar necesitaremos ir al **app.module.ts** y darlos de alta como providers.

```

1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { AppComponent } from './app.component';
4  import { HolaComponent } from './hola/hola.component';
5  import {Servicio1Service} from "./servicio1.service"
6  import {Servicio2Service} from "./servicio2.service"

```

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)[ACEPTAR](#)

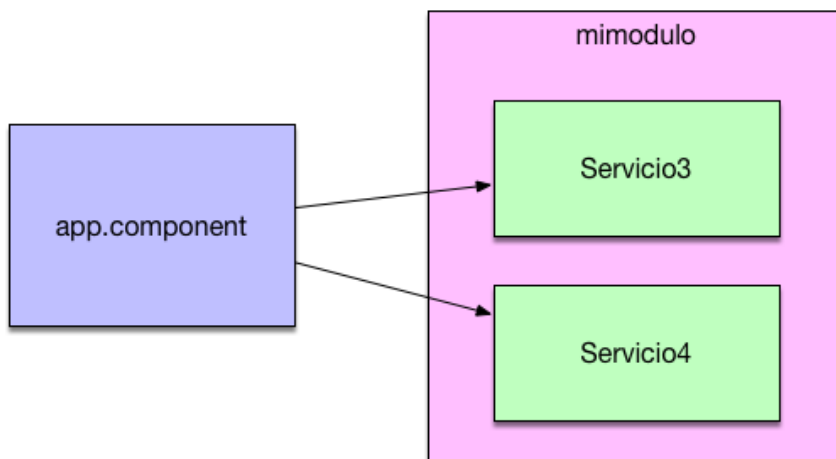
```
17  
18 ],  
19 providers: [Servicio1Service,Servicio2Service],  
20 bootstrap: [AppComponent]  
21 })  
22 export class AppModule { }
```

Si cargamos el componente podremos ver los mensajes en la pantalla:



Angular Modules

Vamos a hacer lo mismo **pero construyendo varios servicios que se encuentren dentro de un módulo.**



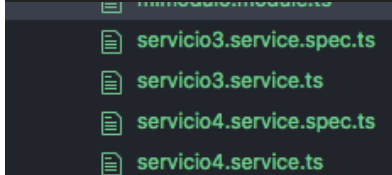
El primer paso es usar Angular Cli y ejecutar el siguiente comando:

ng generate module mimodulo

Este comando nos generará una nueva carpeta en la cual podemos comenzar a ubicar servicios para ello utilizaremos Angular Cli:

carpeta

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)[ACEPTAR](#)

Mimodulo.module.ts

En esta carpeta tenemos los servicios y los ficheros de spec para realización de Test. Ahora bien, tenemos también el fichero `mimodulo.module.ts` que define la estructura del módulo, veamos su contenido.

```
1  import { NgModule } from '@angular/core';
2  import { CommonModule } from '@angular/common';
3  import { Servicio3Service } from './servicio3.service';
4  import { Servicio4Service } from './servicio4.service';
5
6  @NgModule({
7    imports: [
8      CommonModule
9    ],
10   declarations: [
11   ],
12   providers: [Servicio3Service, Servicio4Service]
13 })
14 export class MimoduloModule { }
```

El módulo básicamente se define a través del decorador `@NgModule` que contiene las siguientes secciones:

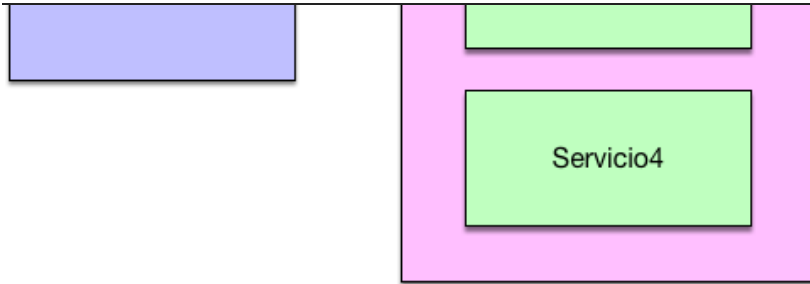
imports: Se encarga de importar otros módulos que sean necesarios en nuestro módulo

declarations: Define los componentes y directivas que pertenecen a este módulo

providers: Declara los servicios que otros componentes pueden usar si utilizan este módulo.

En nuestro caso nuestro módulo está orientado a servicios así que simplemente rellenamos los **providers con Servicio3 y Servicio4**

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)
[ACEPTAR](#)


Veamos el código de los proveedores:

```

1  import { Injectable } from '@angular/core';
2
3  @Injectable()
4  export class Servicio3Service {
5
6      constructor() { }
7
8
9      mensaje3() {
10
11          return "hola3";
12      }
13  }

1  import { Injectable } from '@angular/core';
2
3  @Injectable()
4  export class Servicio4Service {
5
6      constructor() { }
7
8
9      mensaje4() {
10
11          return "hola4";
12      }
13  }
  
```

En principio son iguales a los anteriores . Vamos a usar el módulo en nuestro componente. Para ello tendremos que ir al app.module.ts e importar el módulo.

```

1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { AppComponent } from './app.component';
4  import { HolaComponent } from './hola/hola.component';
5  import { Servicio1Service } from './servicio1.service';
6  import { Servicio2Service } from './servicio2.service';
7  import { MimoduloModule } from './mimodulo/mimodulo.module';
8
  
```

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

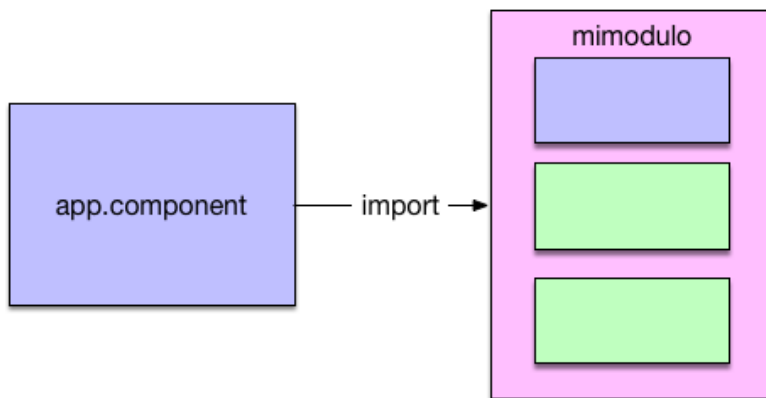
[plugin cookies](#)
[ACEPTAR](#)

```

19     },
20     providers: [Servicio1Service, Servicio2Service],
21     bootstrap: [AppComponent]
22   })
23   export class AppModule { }

```

Acabamos de registrar un módulo para nuestro componente principal



Es importante observar que ya no tenemos la necesidad **de inyectar como proveedores el Servicio3Service y el Servicio4Services** esas inyecciones son realizadas por el módulo . Es momento de ver como queda el código que hace referencia al componente.

```

1  import { Component } from '@angular/core';
2  import {Servicio1Service} from "../servicio1.service"
3  import {Servicio2Service} from "../servicio2.service"
4  import {Servicio3Service} from "../mimodulo/servicio3.service"
5  import {Servicio4Service} from "../mimodulo/servicio4.service"
6  @Component({
7    selector: 'app-root',
8    templateUrl: './app.component.html',
9    styleUrls: ['./app.component.css']
10 })
11 export class AppComponent {
12
13   mensaje1:string;
14   mensaje2:string;
15   mensaje3:string;
16   mensaje4:string;
17
18   constructor(s1:Servicio1Service,s2:Servicio2Service,s3:Servicio3Service,s
19
20
21
22     this.mensaje1=s1.mensaje1();
23     this.mensaje2=s2.mensaje2();
24     this.mensaje3=s3.mensaje3();
25     this.mensaje4=s4.mensaje4();

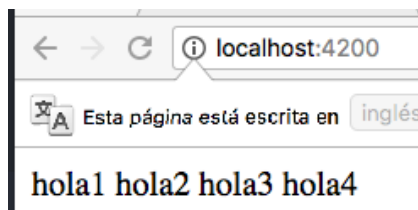
```

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)[ACEPTAR](#)

```
1 {{mensaje1}}  
2 {{mensaje2}}  
3 {{mensaje3}}  
4 {{mensaje4}}
```

Si ejecutamos ahora el programa veremos que salen 4 mensajes:



Acabamos de construir nuestro primer ejemplo con módulos:

1. [Angular async pipe y observables](#)
2. [Angular 5 Hello World y su funcionamiento](#)
3. [Angular ngFor la directiva y sus opciones](#)
4. [React vs Angular 2 , frameworks vs librerías](#)



0
COMPARTIR

SAP Hybris (v) **Tour de puesta a punto para el RGPD**

Barcelona – 24 de abril


[Registro](#)

Archivada en: [Angular](#)

Etiquetada con: [angular](#)

Leave a Reply

Be the First to Comment!

 [Subscribe](#) ▼

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[ACEPTAR](#)

[plugin cookies](#)

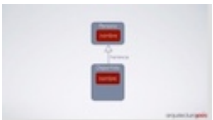
BUSCAR

Buscar en este sitio ...



Mis Cursos de Java Gratuitos

Java Herencia



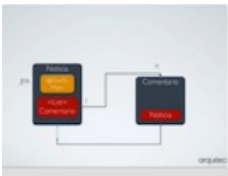
Java JDBC



Servlets



Introduccion JPA



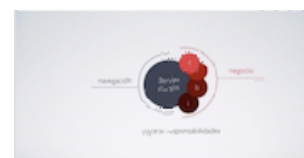
plugin cookies

ACCEPTAR

3949 71 10 0010



Java Web



Pack Java Core



Arquitectura Java Solida con Spring



Spring REST Client con RestTemplates

Angular 5 Hello World y su funcionamiento

PostMan App con Spring REST

Arquitecturas REST y sus niveles

Java 9 Modules y el concepto de modularidad

Java Interfaces y el concepto de simplicidad

Spring Boot Properties utilizzando @Value

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)[ACEPTAR](#)

LO MAS LEIDO

[¿Qué es Spring Boot?](#)[10 Características que me gustan de TypeScript](#)[Java Constructores this\(\) y super\(\)](#)[Usando Java Session en aplicaciones web](#)[Java Iterator vs ForEach](#)[Introducción a Servicios REST](#)[¿Cuales son las certificaciones Java?](#)[Java Override y encapsulación](#)[Ejemplo de Java Singleton \(Patrones y ClassLoaders\)](#)[¿Qué es Gradle?](#)[Arquitecturas REST y sus niveles](#)[REST JSON y Java](#)[Usando el patron factory](#)[Ejemplo de JPA , Introducción \(I\)](#)[Uso de Java Generics \(I\)](#)[Mis Libros](#)[¿Qué es un Microservicio?](#)[Comparando java == vs equals](#)[Spring MVC Configuración \(I\)](#)[Spring Security JDBC y su configuracion](#)

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información.

[plugin cookies](#)[ACEPTAR](#)
