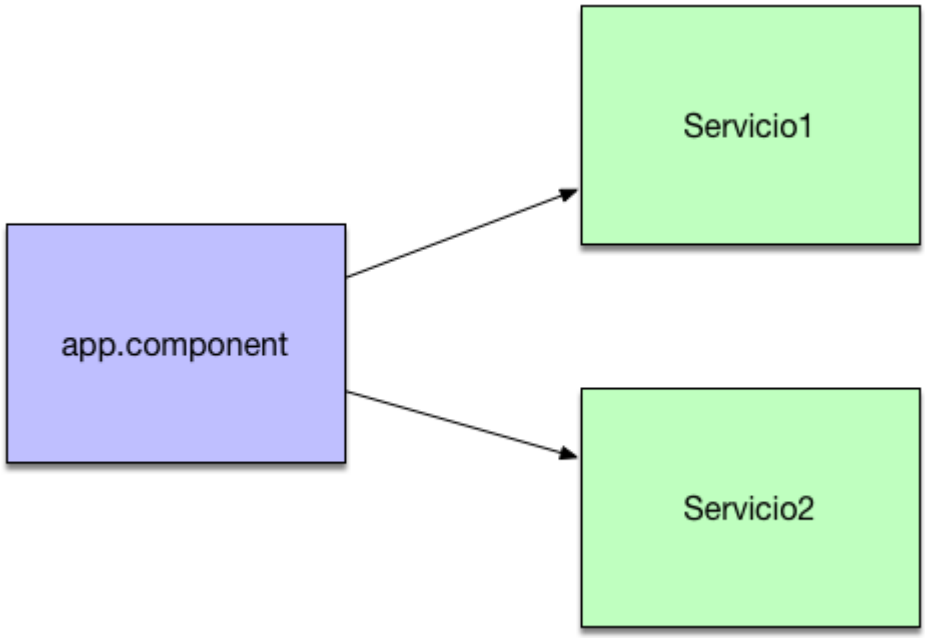
El concepto de Angular Modules es un concepto cercano al de Java Archive de Java o DLL de .NET en el cual nosotros queremos definir un grupo de funcionalidad reutilizable. Vamos a construir un ejemplo sencillo de Angular Modules. Para ello vamos a partir de un componente de Angular que utiliza un par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



Abordemos el código:

```
import { Component } from '@angular/core';
import {Servicio1Service} from "./servicio1.service"
import {Servicio2Service} from "./servicio2.service"

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
```

```
  styleUrls: ['./app.component.css']
})
export class AppComponent {


  mensaje1:string;
  mensaje2:string;

  constructor(s1:Servicio1Service,s2:Servicio2Service) {


      this.mensaje1=s1.mensaje1();
      this.mensaje2=s2.mensaje2();

  }
}
```

Veamos lo que hace la plantilla :

```
{{mensaje1}}
{{mensaje2}}
```

Veamos lo que hacen los servicios :

```
import { Injectable } from '@angular/core';
```

```
@Injectable()
export class Servicio1Service {

  constructor() { }

  mensaje1() {

    return "hola1";
  }
}


import { Injectable } from '@angular/core';

@Injectable()
export class Servicio2Service {

  constructor() { }

 mensaje2() {

   return "hola2"
 }
}
```

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.

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { HolaComponent } from './hola/hola.component';
import {Servicio1Service} from "./servicio1.service"
import {Servicio2Service} from "./servicio2.service"


@NgModule({
  declarations: [
    AppComponent,
    HolaComponent
  ],
  imports: [

    BrowserModule,

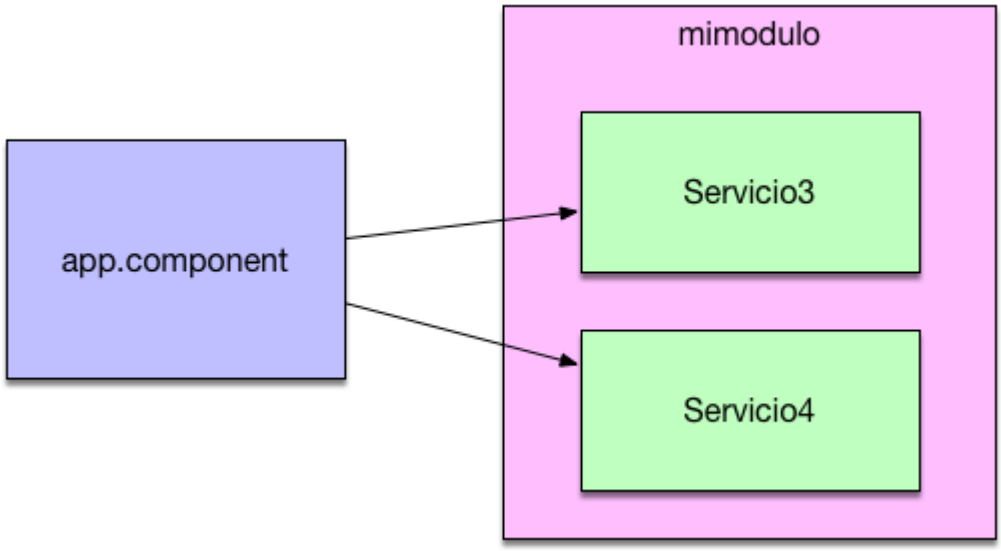
  ],
  providers: [Servicio1Service,Servicio2Service],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

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:

ng generate service mimodulo/servicio3
ng generate service mimodulo/servicio4:

Acabamos de generar dos servicios dentro de un módulo ,veamos cual es la estructura de carpeta:



MiModulo.module.ts

En esta carpeta tenemos lo 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.

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import {Servicio3Service} from "./servicio3.service"
import {Servicio4Service} from "./servicio4.service"

@NgModule({
  imports: [
    CommonModule
  ],
  declarations: [
  ],
   providers: [Servicio3Service,Servicio4Service]
})
export class MimoduloModule { }
```

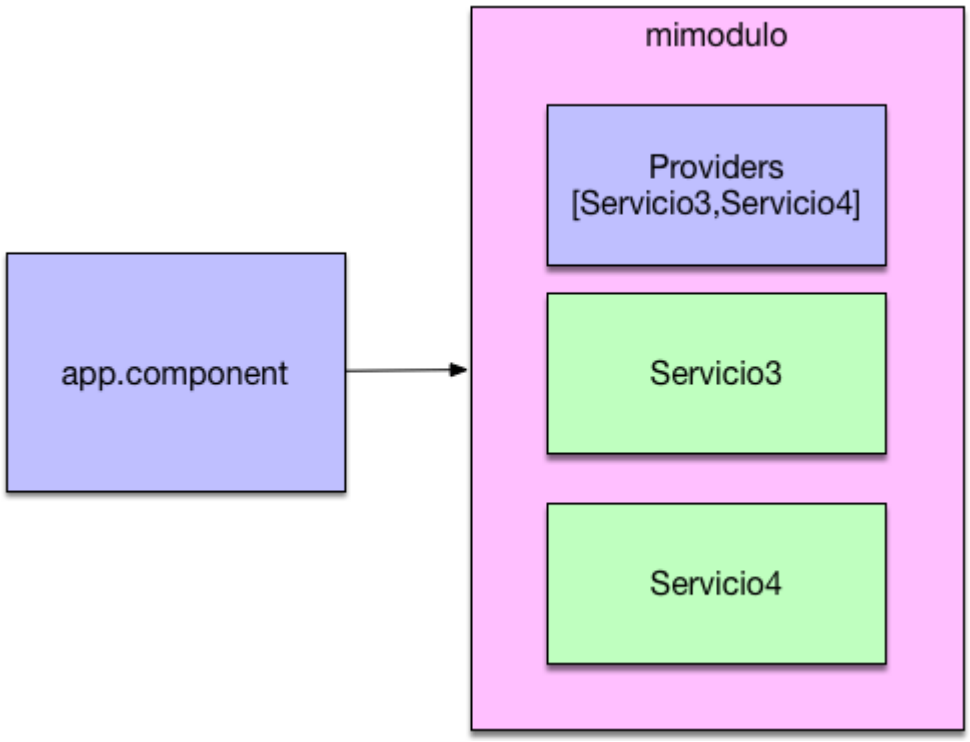
El módulo básicamente se define a través de del decorador @NgModule que contiene las siguientes secciones:

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 esta orientado a servicios así que simplemente rellenamos los providers con Servicio3 y Servicio4



Veamos el código de los proveedores**:**

```
import { Injectable } from '@angular/core';

@Injectable()
```

```
export class Servicio3Service {

  constructor() { }


  mensaje3() {

    return "hola3";
  }
}
```

```
import { Injectable } from '@angular/core';

@Injectable()
export class Servicio4Service {

  constructor() { }


  mensaje4() {

    return "hola4";
  }
}
```

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.

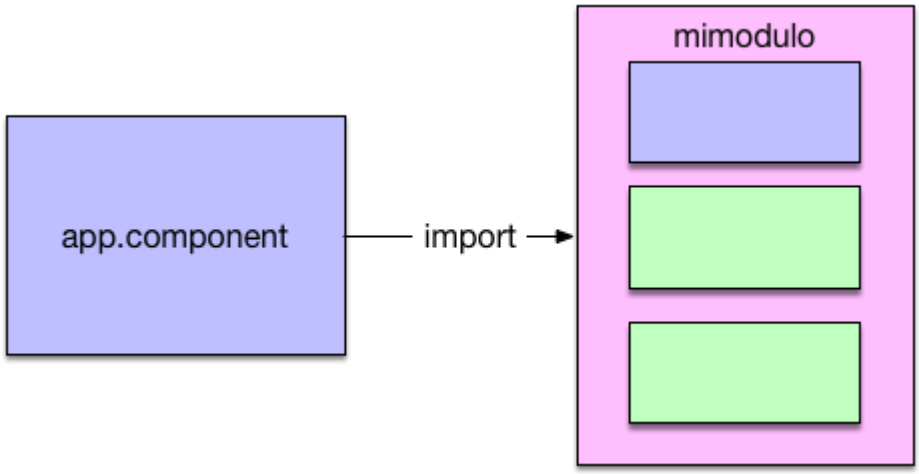
```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { HolaComponent } from './hola/hola.component';
import {Servicio1Service} from "./servicio1.service"
import {Servicio2Service} from "./servicio2.service"
import {MimoduloModule} from "./mimodulo/mimodulo.module"

@NgModule({
  declarations: [
    AppComponent,
    HolaComponent
  ],
  imports: [

    BrowserModule,
  MimoduloModule

  ],
  providers: [Servicio1Service,Servicio2Service],
  bootstrap: [AppComponent]
})
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.

```
import { Component } from '@angular/core';
import {Servicio1Service} from "./servicio1.service"
import {Servicio2Service} from "./servicio2.service"
import {Servicio3Service} from "./mimodulo/servicio3.service"
import {Servicio4Service} from "./mimodulo/servicio4.service"
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {


  mensaje1:string;
  mensaje2:string;
```

```
  mensaje3:string;
  mensaje4:string;
constructor(s1:Servicio1Service,s2:Servicio2Service,s3:Servicio3Servic
e,s4:Servicio4Service) {

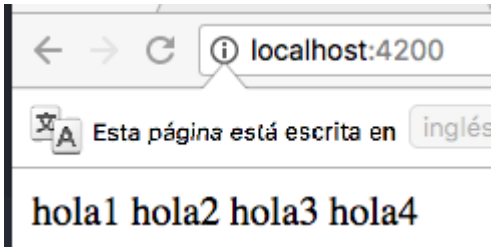

      this.mensaje1=s1.mensaje1();
      this.mensaje2=s2.mensaje2();
      this.mensaje3=s3.mensaje3();
      this.mensaje4=s4.mensaje4();

  }
}
```

Los cambios son muy sencillos , simplemente hemos añadido los dos nuevos servicios que devuelven los mensajes. Por lo tanto también cambiaremos la plantilla para tener dos mensajes más.

```
{{mensaje1}}
{{mensaje2}}
{{mensaje3}}
{{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 librerias