# DESARROLLO DE APLICACIONES CON ANGULAR

Angular

# ANGULAR

- Servicios
  - Son clases que encapsulan funcionalidad.
  - Permiten aislar la fuente de los datos de su uso.
  - Permiten contener información compartida por todos los componentes.
  - Por defecto funcionan como *singleton*.
  - Desde los componentes se utilizan inyectando la dependencia.

# ANGULAR

- Servicios
  - Creación:
    - ng generate service ruta/nombre_servicio

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

@Injectable({
  providedIn: 'root'
})
export class ProductosServicesService {

constructor() { }
}
```

# ANGULAR

- Servicios
  - Creación:
    - De manera básica, el servicio funciona como una clase normal, con atributos y métodos convencionales.

```
games:Game[];
addGame(title:string, cover:string){
    this.games.push(new Game(title,cover));
}
```

# ANGULAR

- Servicios
  - Uso:
    - El servicio debe inyectarse en el componente que lo usa a través del constructor:

```
constructor(private gameService:GamesServiceService) {}
```

    - A partir de este momento se dispone de una referente al servicio (gameService) como atributo de la clase:

```
public agregar():void {
    this.gameService.addGame("Batman","Batman.jpg");
}
```

# ANGULAR

- Servicios
  - De múltiples instancias (cada instancia tiene su propio servicio con sus datos):
    - Se elimina la referencia a providedIn:'root'

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

    - En los componentes, en la declaración, se agrega como providers los servicios:

```
@Component({
   selector:....,
   templateUrl:....,
   styleUrls:...,
   providers:[NombreClaseServicio]
})
```

# ANGULAR

- Servicios
  - Funcionalidad:
    - Se puede implementar mediante acceso a atributos visibles o llamadas a métodos síncronos convencionales.
    - Se puede implementar mediante "promesas":

```
getNombrePromise(sufijo:string, espera:number):Promise<string> {
    const promise = new Promise<string>((resolve,reject)=> {
      let nombreModificado = `${this.nombre}${sufijo}`;
      setTimeout(function(){
        resolve(nombreModificado);
      },espera);

    });
    return promise;
  }
```

# ANGULAR

- Servicios
  - Funcionalidad: consumo de 'promesas'
    - Inyección del servicio en el componente cliente.
    - `constructor(private servicio1:Servicio1Service) { }`
    - Mediante then-catch

```
this.servicio1.getNombrePromise("**",5000)
      .then(nombreRecibido => this.servicio1Ok(nombreRecibido))
      .catch(error => this.servicio1Ko(error));
```

    - Mediante async y await

```
async nombreFuncion() {
   try {
                    this.nombreServicio1Async = await this.servicio1.getNombrePromise("---",10000);
   } catch (error) {
                    console.log(error);
   }
}
```

# ANGULAR

- Servicios
  - Funcionalidad:
    - Se puede implementar mediante “observables”:

```
findAll(): Observable<Game[]> {
    return this.http.get<Game[]>(URL);
}
```

    - Las llamadas desde los componentes se realizan mediante suscripción:

```
this.gameService.findAll().subscribe(games => {
        this.allGames = games;
        this.games = games;
 });
```