Servicios e inyección de dependencias


Productos
Logging

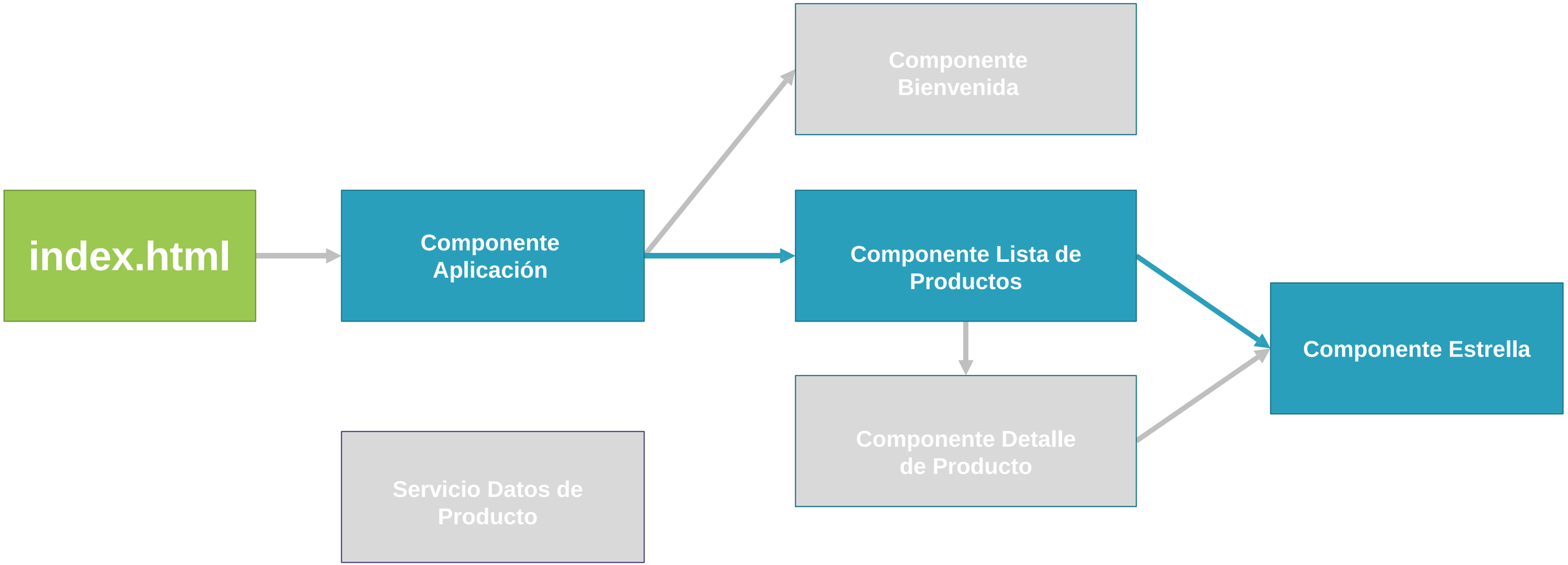
Servicio

- **Una clase con un propósito específico.**
- **Características:**
  - **Son independientes de cualquier componente en particular**
  - **Proporcionan datos o lógica compartidos entre componentes**
  - **Encapsulan interacciones externas**

Resumen

- **¿Cómo funciona?**
- **Crear un servicio**
- **Registrar el servicio**
- **Inyectar el servicio**

Arquitectura de la aplicación



¿Cómo funciona?

aquí let. no sustituye en el tiemo de compilacion (no existe aqui)

Servicio

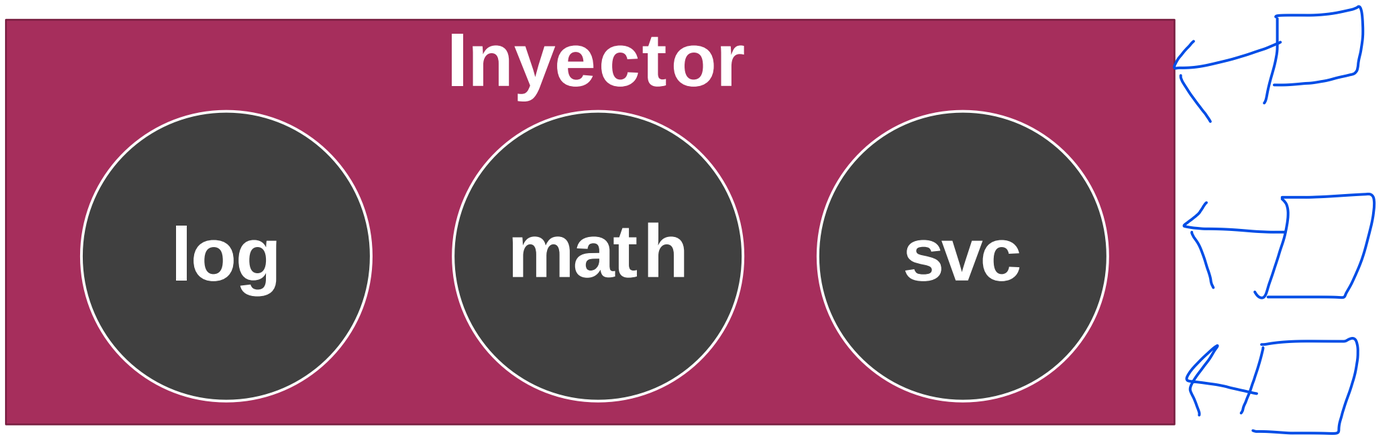
```
export class myService {}
```

Componente

```
let svc = new myService();
```

¿Cómo funciona?

singleton => Una unica instancia para todo el mundo



componentes

Servicio

```
export class myService {}
```

Componente

```
constructor(private myService) {}
```

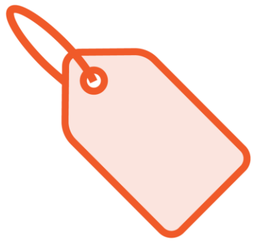
constructor se llama así

Inyección de dependencias

Un patrón de programación en el que una clase recibe las instancias de los objetos que necesita (llamadas dependencias) de una fuente externa en lugar de crearlas ella misma.

Crear un servicio

Crear la clase de servicio



Definir los metadatos mediante un decorador

Importar lo necesario

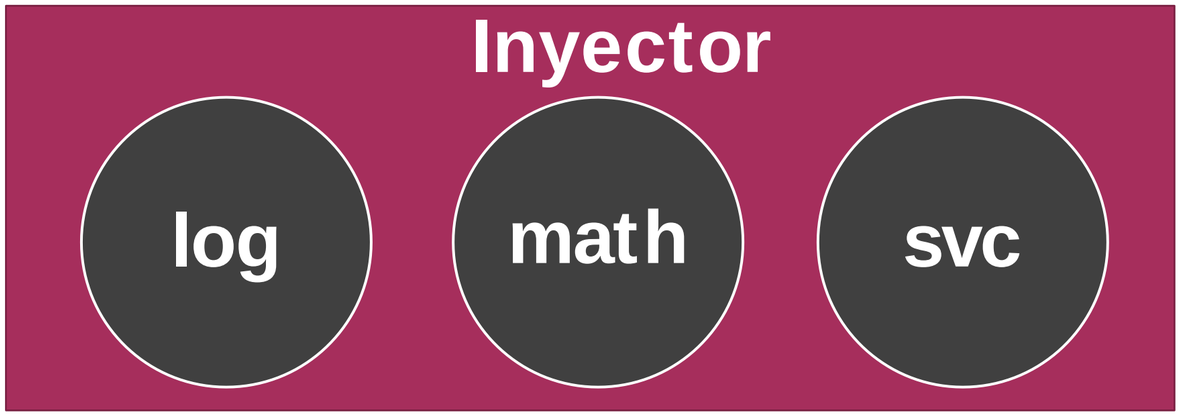
Crear un servicio

product.service.ts

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

@Injectable()
export class ProductService {

  getProducts(): IProduct[] {
  }

}
```

permite que esta clase pueda abrirse en cualquier constructor

Registro de un servicio



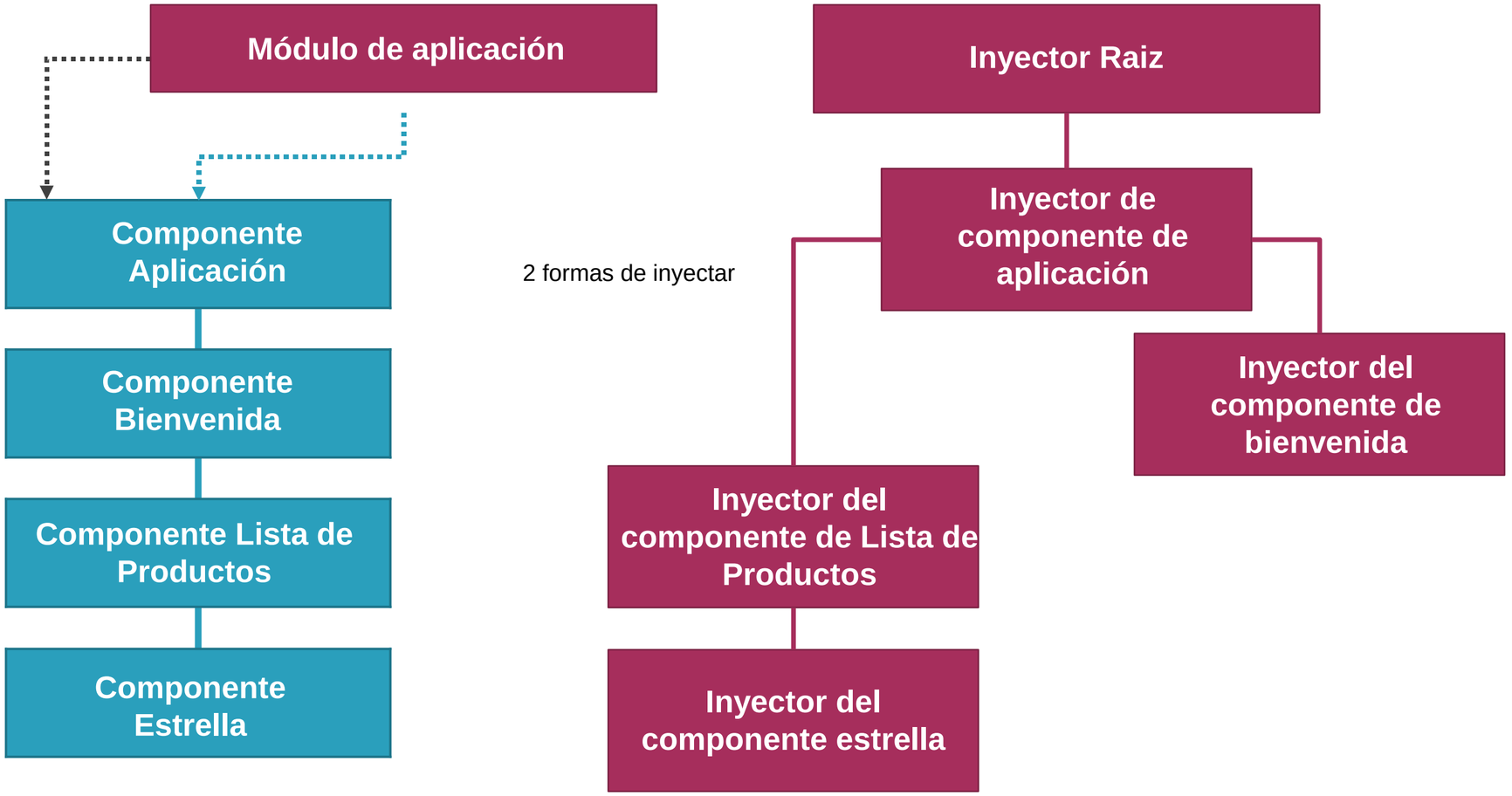
Servicio

```
export class myService {}
```

Componente

```
constructor(private myService) {}
```

Inyectores en Angular



Registrar un servicio

Inyector raiz

El servicio está disponible en toda la aplicación

Recomendado para la mayoría de los escenarios

Inyector de Componente

El servicio está disponible SOLO para ese componente y sus componentes hijos (anidados)

Aísla un servicio utilizado por un solo componente

Proporciona múltiples instancias del servicio

Registro de un servicio - Raiz de la aplicación

product.service.ts

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

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

  getProducts(): IProduct[] {
  }

}
```

product.service.ts

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

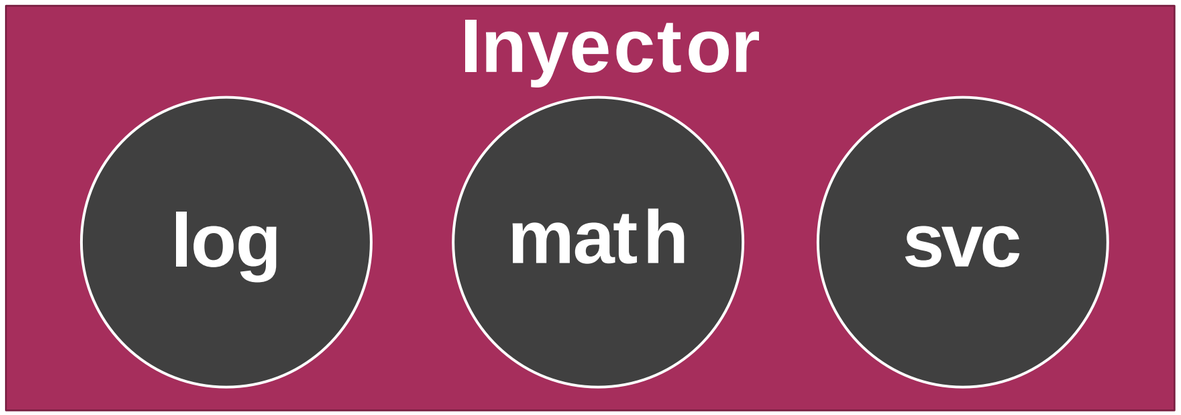
product-list.component.ts

```
@Component({
  templateUrl: './product-list.component.html',
  providers: [ProductService]
})
export class ProductListComponent { }
```

app.module.ts

```
@NgModule({
  imports: [ BrowserModule ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ],
  providers: [ProductService]
})
export class AppModule { }
```

Injecting the Service



Servicio

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

Componente

```
constructor(private myService) {}
```

Inyectando el servicio

product-list.component.ts

```
...


@Component({
  selector: 'pm-products',
  templateUrl: './product-list.component.html'
})
export class ProductListComponent {

 constructor() {
 }

}
```

Inyectando el servicio

product-list.component.ts

```
...
import { ProductService } from './product.service';

@Component({
  selector: 'pm-products',
  templateUrl: './product-list.component.html'
})
export class ProductListComponent {
 private _productService;
 constructor(productService: ProductService) {
   this._productService = productService;
 }

}
```

Inyectando el servicio

product-list.component.ts

```
...
import { ProductService } from './product.service';

@Component({
  selector: 'pm-products',
  templateUrl: './product-list.component.html'
})
export class ProductListComponent {

 constructor(private productService: ProductService) { }

}
```

Lista de comprobación de servicios: Crear un servicio

- Clase de servicio
  - Nombre específico
  - Utiliza PascalCasing
  - Añade "Service" al nombre del servicio
  - Palabra clave `export`
- Decorador de servcio
  - Utiliza Injectable
  - Prefijo con @; Sufijo con ()
- Importa lo necesario

```
import { Injectable } from '@angular/core';
@Injectable({
  providedIn: 'root'
})
export class ProductService {...}
```

Lista de comprobación de servicios: Registrar un servicio

- Selecciona el nivel adecuado en la jerarquía
  - El inyector raíz de la aplicación si el servicio se va a utilizar en toda la aplicación
  - Inyector específico del componente si este componente es el único que va a utilizar el servicio

- Decorador para el servicio: Injectable
  - Establece la propiedad `providedIn` a `'root'`

```
@Injectable({
  providedIn: 'root'
})
export class ProductService {...}
```

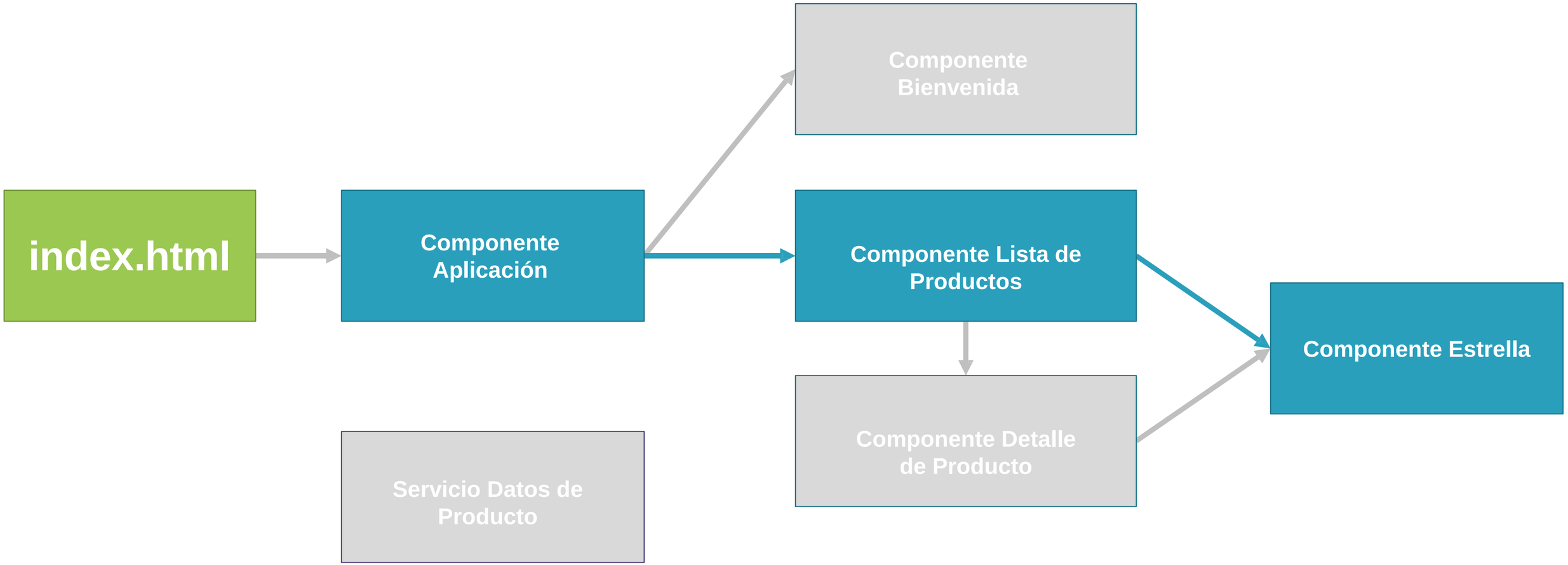
- Decorador de componente
  - Establece la propiedad `providers` para el servicio

Lista de comprobación de servicios: Inyección de dependencias

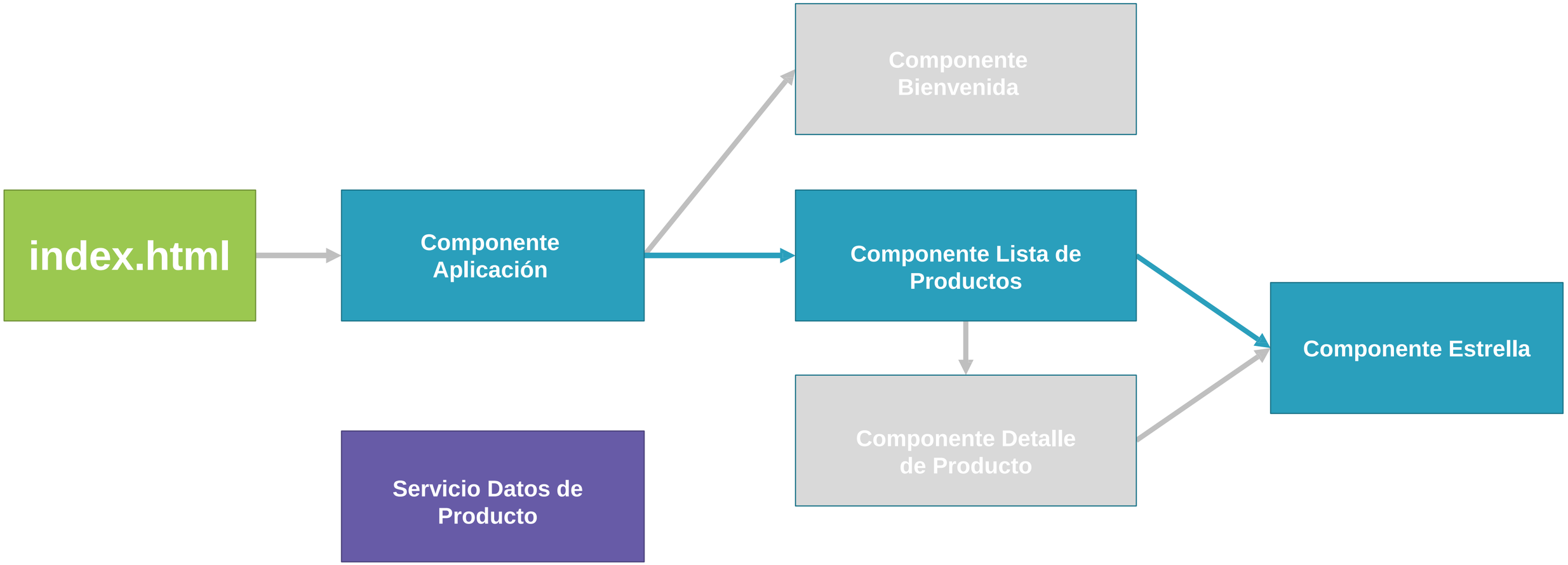
- Especificar el servicio como una dependencia
- Utiliza un parámetro en el constructor
- El servicio se inyecta cuando se instancia el componente

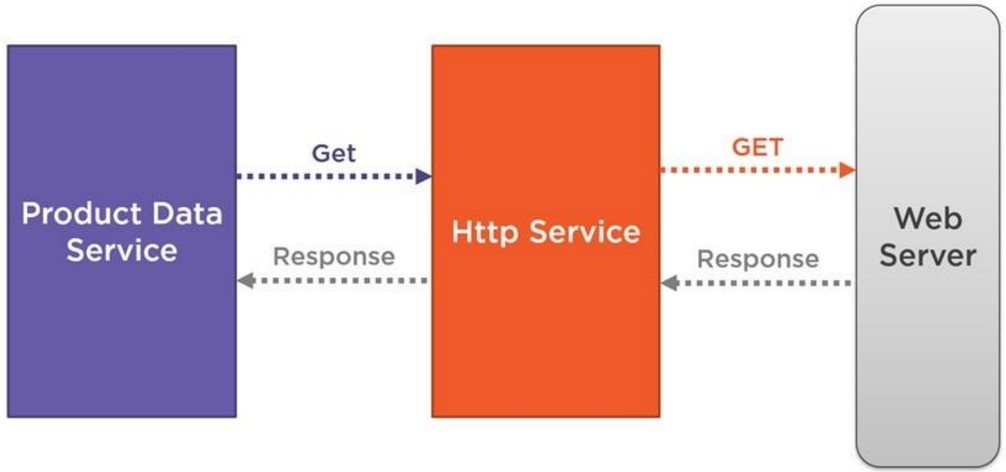
```
constructor(private productService: ProductService) { }
```

Arquitectura de la aplicación



Arquitectura de la aplicación





A continuación …

Recuperación de datos mediante HTTP