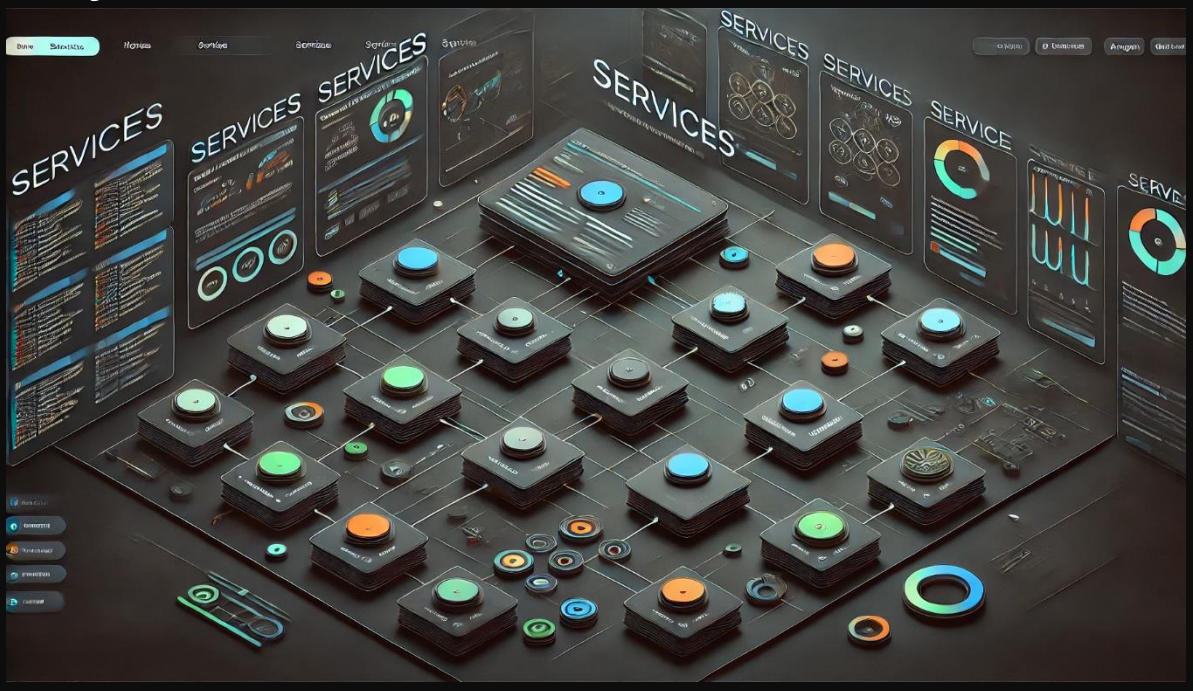
Flujo de Actualización en Tienda Online



Flujo de Actualización en Tienda Online

En esta lección vamos a estudiar a detalle el flujo de actualización de la página de Angular una vez que se detecta un cambio.

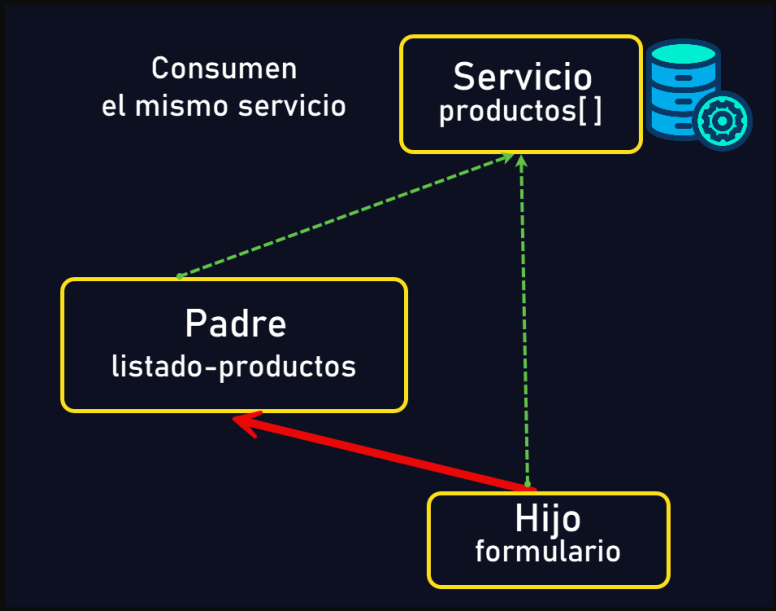
El proceso de actualización de la vista, por ejemplo, cuando agregamos un nuevo producto, es el resultado de cómo funciona la **reactividad** y el ciclo de detección de cambios en Angular. A continuación, veremos a detalle cómo sucede el proceso de actualización:

¿Qué dispara la actualización?

En el caso mencionado de agregar un nuevo producto, lo que realmente dispara la actualización de la página es el **vinculador de datos (data binding)** de Angular y el **servicio compartido**. No es necesario que el componente hijo le informe al componente padre de manera explícita, recordemos que ya no estamos emitiendo ningún mensaje desde el componente hijo de formulario al componente padre de listado de productos. Así que veamos cómo ocurre la actualización en este ejemplo:

Flujo del proceso paso a paso:

1. **El servicio**: Tienes el `ProductoService` que contiene un arreglo `productos`, el cual es compartido tanto por el componente `ListadoProductosComponent` (el padre) como por el componente `FormularioComponent` (el hijo). Este servicio actúa como un **almacenamiento compartido** entre ambos componentes.



2. **Agregar un producto (FormularioComponent)**: Cuando el usuario agrega un nuevo producto a través del formulario, se ejecuta el método `agregarProducto` del `FormularioComponent`. Este método llama al servicio `ProductoService` para agregar un nuevo producto al arreglo `productos`.

```
agregarProducto() {
  const producto = new Producto(this.descripcionInput, this.precioInput);
  this.productoService.agregarProducto(producto);  // Aquí se modifica el arreglo en el
servicio
}
```

3. **Modificación del estado en el servicio**: Al llamar al método `agregarProducto` del servicio, se modifica directamente el arreglo `productos` que está almacenado en el `ProductoService`.

```
agregarProducto(producto: Producto) {
  this.productos.push(producto);  // Se agrega el nuevo producto al array
}
```

4. **Reactividad y detección de cambios (Change Detection)**: Una vez que el arreglo `productos` del servicio se modifica, el **mecanismo de detección de cambios de Angular** entra en juego. Angular **automáticamente** detecta que el estado del servicio ha cambiado (específicamente, que el arreglo `productos` se ha actualizado) y, dado que `ListadoProductosComponent` está utilizando este arreglo a través del servicio, el componente se vuelve a renderizar con los datos actualizados.

```
ngOnInit(): void {
  this.productos = this.productoService.productos;  // Se usa la referencia al arreglo de
productos del servicio
```

```
}
```

Debido a que `productos` está vinculado a la vista a través de la directiva `@for`, Angular detecta que el arreglo ha cambiado y vuelve a renderizar la lista de productos. El ciclo de detección de cambios se asegura de que la vista refleje el estado actualizado del arreglo.

```
<ul class="list-group w-50 mx-auto">
    @for (productoElemento of productos; track productoElemento) {
    <app-producto [producto]="productoElemento" />
    }
</ul>
```

5. **Visualización del nuevo producto**: El componente `ListadoProductosComponent` muestra automáticamente el nuevo producto en la lista sin necesidad de acciones adicionales por parte del programador, ya que Angular está observando los cambios en los datos y actualiza la interfaz de usuario de forma reactiva.

¿Qué es lo que realmente causa la actualización?

Es el **servicio compartido** (`ProductoService`) el que almacena los productos, y cuando se modifica este estado, Angular automáticamente detecta el cambio y vuelve a renderizar la vista. Angular realiza esta actualización a través de su mecanismo de **detección de cambios**, lo que garantiza que cuando cualquier componente modifique los datos en un servicio, esos cambios se reflejen automáticamente en todos los lugares donde esos datos son consumidos.

Detección de cambios en Angular

Angular utiliza un mecanismo de **detección de cambios** que se ejecuta en momentos específicos del ciclo de vida de la aplicación, como:

- Cuando el usuario interactúa con la interfaz (por ejemplo, al hacer clic en un botón).
- Después de cada operación asincrónica (como respuestas HTTP o eventos del DOM).
- Después de los eventos como el `click` que llama a `agregarProducto()`.

En nuestro caso, la llamada a `agregarProducto()` activa la detección de cambios, y Angular verifica si hay algo que debe actualizarse en la interfaz de usuario. Como el arreglo `productos` se ha modificado, la vista se actualiza para reflejar estos cambios.

En resumen:

1. El **FormularioComponent** llama al método `agregarProducto()` del servicio para modificar el arreglo de productos.
2. El **ProductoService** almacena los productos compartidos y modifica su estado.
3. El **ListadoProductosComponent** accede a este arreglo en su propiedad `productos`, lo que permite que Angular detecte el cambio.
4. **Angular**, a través de su sistema de **detección de cambios**, actualiza automáticamente la vista del componente padre cuando los productos cambian en el servicio.

No hay necesidad de que el componente hijo notifique directamente al componente padre, ya que ambos comparten el estado a través del servicio.

Este es un ejemplo perfecto de cómo Angular facilita la gestión de estado a través de servicios y cómo se puede aprovechar su detección de cambios para que las vistas se actualicen automáticamente cuando los datos cambian.

Saludos!

Ing. Ubaldo Acosta

Fundador de GlobalMentoring.com.mx