

**DESARROLLO DE SOFTWARE FRONTEND III**

**MARTINEZ COSIO JOSE ALFREDO**

**Aarón Hernández García**

**Práctica con enrutamiento en Angular (Proyecto sencillo)**

**programación reactiva RxJS**

**05**

**/0**

**8**

**/202**

**5**

**Inicio programación reactiva RxJS**

**Conceptos clave utilizados**

**1. interval()**

La función interval() de RxJS crea un observable que emite valores secuenciales (0, 1, 2, …) con un intervalo de tiempo determinado. En este caso, se usará con 2000 milisegundos (2 segundos).

**2. pipe()**

pipe() permite encadenar operadores para transformar o filtrar los valores que emite un observable. Es la forma en que se “procesan” los datos antes de usarlos.

**3. map()**

El operador map() sirve para transformar los valores emitidos por un observable. En este ejercicio, transformará los índices emitidos por interval() en frases.

**4. take()**

El operador take(n) se utiliza para limitar la cantidad de emisiones que hace un observable. Es importante aquí para detener automáticamente la emisión de frases después de mostrar la última.

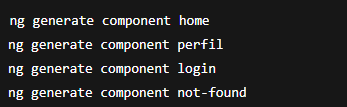
Proyecto Angular Propuesto: **“Mi Sitio Simple”**

**Proyecto Angular Propuesto: “Mi Sitio Simple”**

1. Crear el proyecto y configurar el enrutamiento



2. Crear componentes



3. Crear un guardia de ruta (AuthGuard)

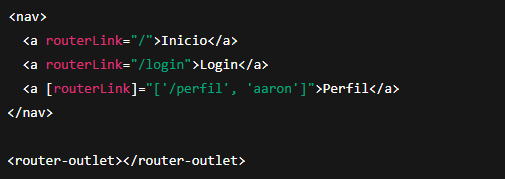


4. Configurar las rutas en app-routing.module.ts



5. Implementar navegación entre rutas (routerLink)

En app.component.html:

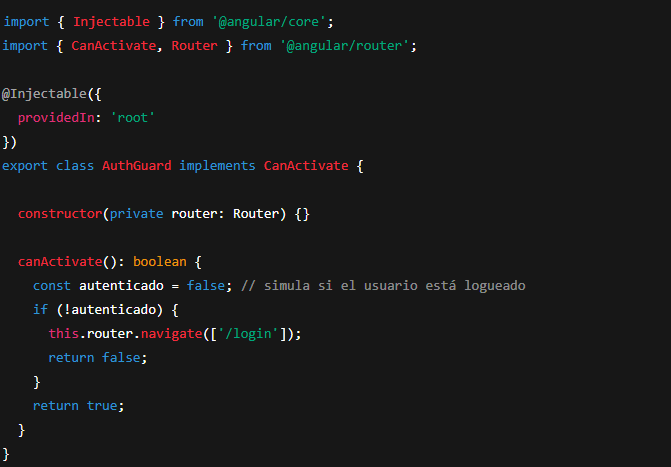


6. Usar parámetros en la ruta perfil/:usuario

En perfil.component.ts:



7. Crear el guardia auth.guard.ts



8. Conclusión del documento

El enrutamiento en Angular permite organizar la navegación de la aplicación de forma modular y escalable. El uso de rutas con parámetros facilita el paso de datos entre páginas, mientras que los guardias (guards) agregan control sobre el acceso a ciertas vistas. Estas herramientas permiten crear experiencias más seguras, dinámicas y estructuradas.