

Práctica 3

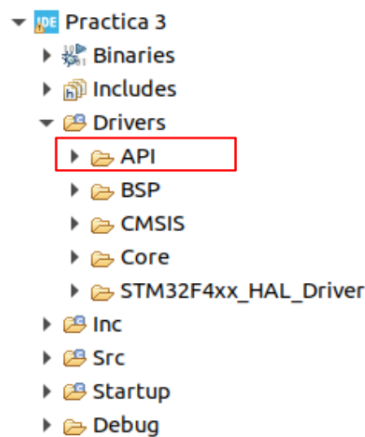
Objetivo:

Implementar un módulo de software para trabajar con retardos no bloqueantes a partir de las funciones creadas en la práctica 2.

Punto 1

Crear un nuevo proyecto como copia del proyecto realizado para la práctica 2.

Crear una carpeta **API** dentro de la carpeta Drivers en la estructura de directorios del nuevo proyecto. Crear dentro de la carpeta API, subcarpetas /Src y /Inc.



Encapsular las funciones necesarias para usar retardos no bloqueantes en un archivo fuente **API_delay.c** con su correspondiente archivo de cabecera **API_delay.h**, y ubicar estos archivos en la carpeta API creada.

En API_delay.h se deben ubicar los prototipos de las funciones y declaraciones

```
typedef uint32_t tick_t; // Qué biblioteca se debe incluir para que esto compile?
```

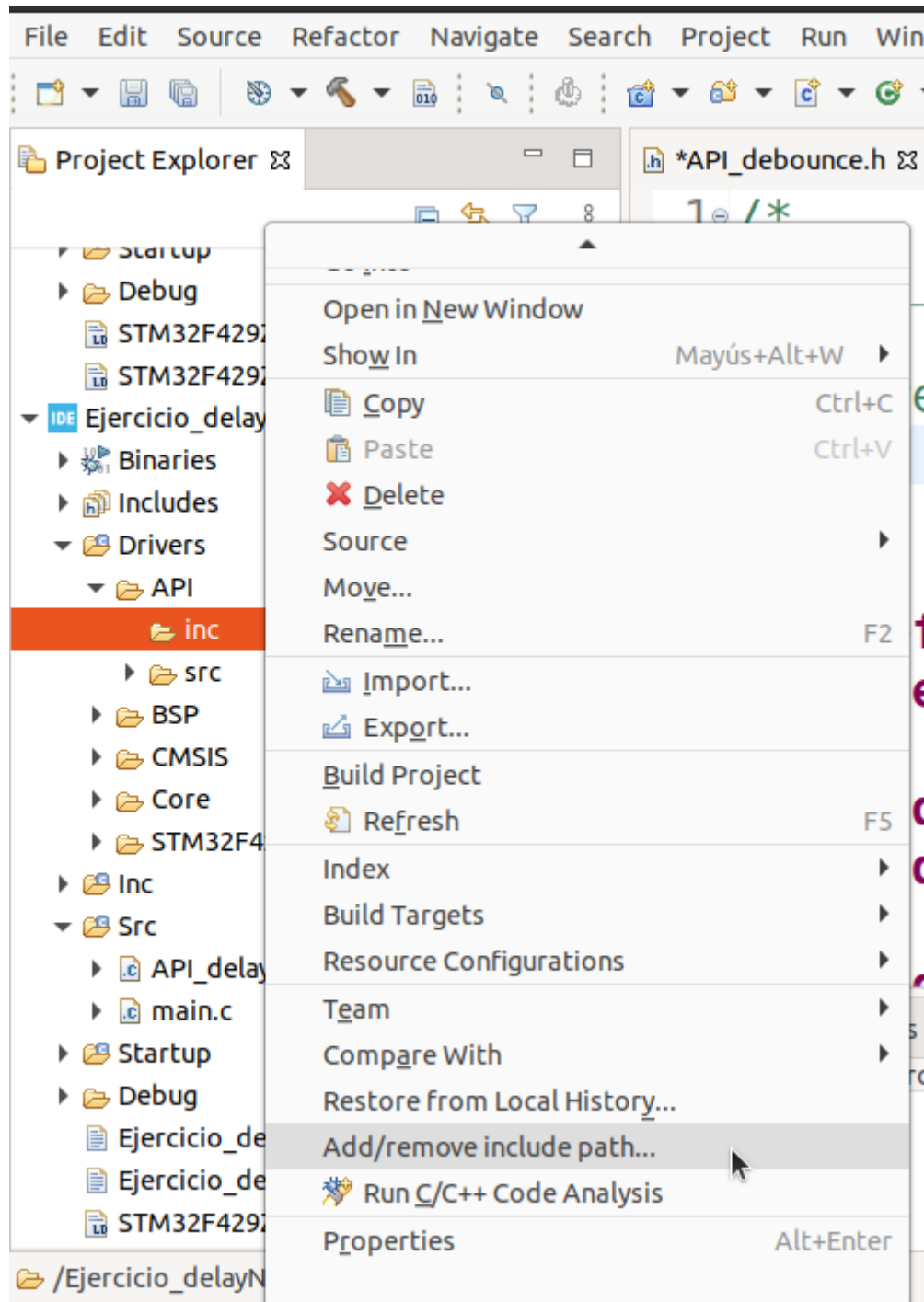
```
typedef bool bool_t;    // Qué biblioteca se debe incluir para que esto compile?
```

```
typedef struct{  
    tick_t startTime;  
    tick_t duration;  
    bool_t running;  
} delay_t;
```

```
void delayInit( delay_t * delay, tick_t duration );  
bool_t delayRead( delay_t * delay );  
void delayWrite( delay_t * delay, tick_t duration );
```

En API_delay.c se deben ubicar la implementación de todas las funciones.

nota: cuando se agregar carpetas a un proyecto de eclipse se deben incluir en el *include path* para que se incluya su contenido en la compilación. Se debe hacer clic derecho sobre la carpeta con los archivos de encabezamiento y seleccionar la opción add/remove include path.



Punto 2

Implementar un programa que utilice retardos no bloqueantes y haga titilar en forma periódica e independiente los tres leds de la placa NUCLEO-F429ZI de acuerdo a una secuencia determinada como en la práctica 1.

Cada led debe permanecer encendido 200 ms. No debe encenderse más de un led simultáneamente en ningún momento.

Para pensar luego de resolver el ejercicio:

- ¿Se pueden cambiar los tiempos de encendido de cada led fácilmente en un solo lugar del código o estos están *hardcodeados*? ¿Hay números “mágicos”?
- ¿Qué bibliotecas estándar se debieron agregar a `API_delay.h` para que el código compile? Si las funcionalidades de una API propia crecieran, habría que pensar cuál sería el mejor lugar para incluir esas bibliotecas y algunos `typedefs` que se usan en el ejercicio.
- ¿Es adecuado el control de los parámetros pasados por el usuario que se hace en las funciones implementadas? ¿Se controla que sean valores válidos? ¿Se controla que estén dentro de los rangos correctos?