Day 02: Exercises

Task 02_1_CPP_BlinkyDelayNotBlocking

Expand project: CPP_Blinky_02_01 or setup a new project.

Class NoneBlockSystemTickDelay:
Use the SystemTick timer to implement a software delay that does not block. (see exercise 1_5)

Class STM32H7Led:
Use the HAL library to control the LEDs in STM32H7Led

Caution we need 2 constructors for STM32H7Led:

- One without initialization parameters.
- One with initialization parameters!

Class BlinkingLed:

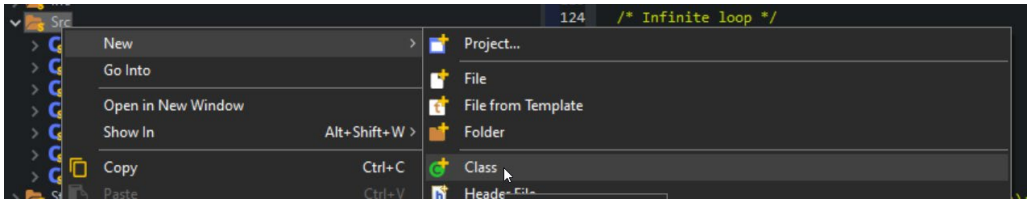
Initialised with a Led and realises blinking on every call by using the inherited logic of STM32H7Led and NoneBlockSystemTickDelay.

Main:

Instantiated the three LEDs with 50% duty cycles and the times: LED1 every 250ms.

LED2 every 500ms, LED3 every 1000ms.

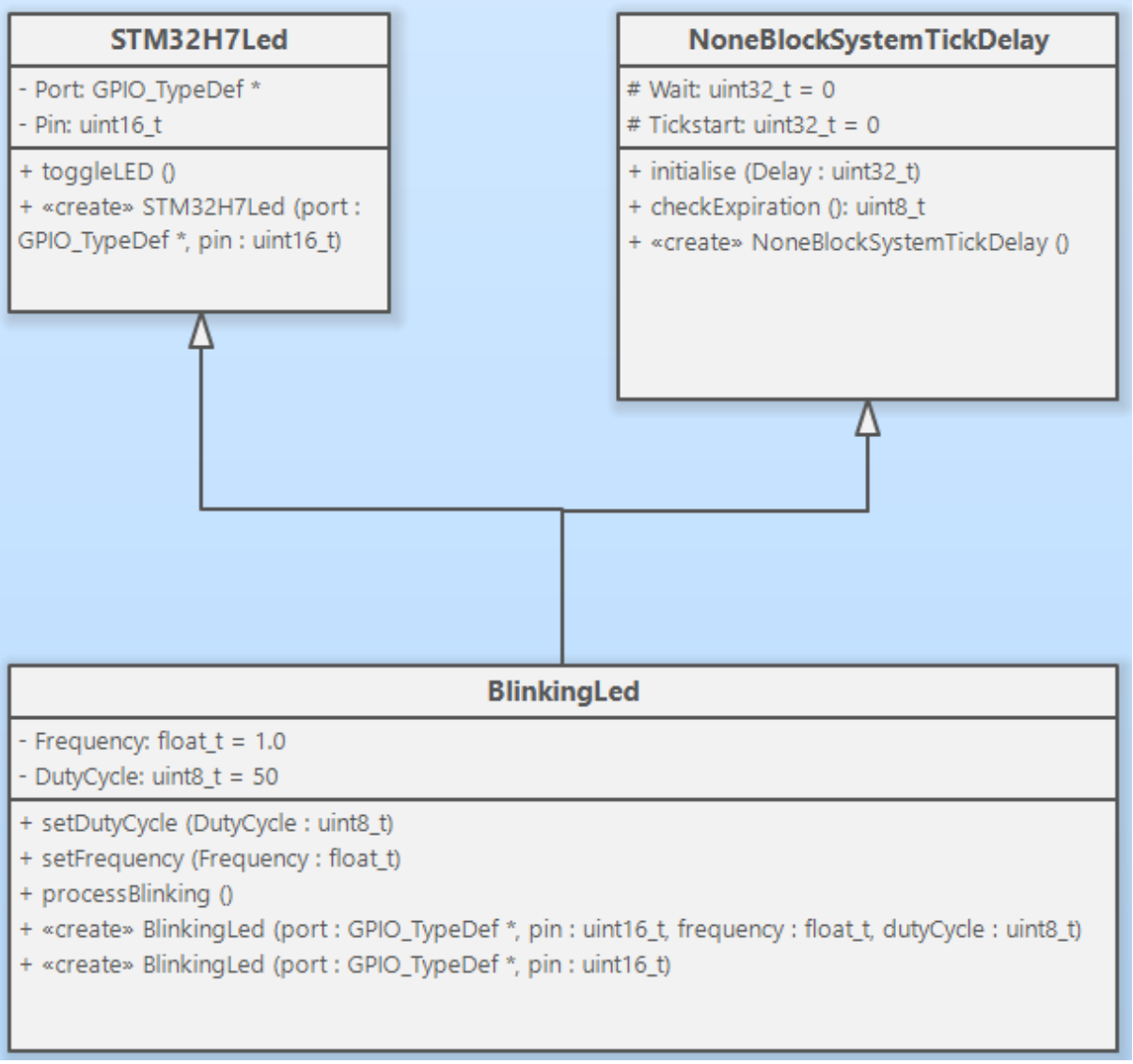
Make use of the wizard for creating a class and setter/getters (setters for BlinkinLed)!



Setter and getters: select Header-File, ALT+SHIFT+S, then “Generate Getter Setters…”

ON next page are the classes defined.

Optional: implement the return value of NoneBlocSysTickDelay::checkExpiration as bool

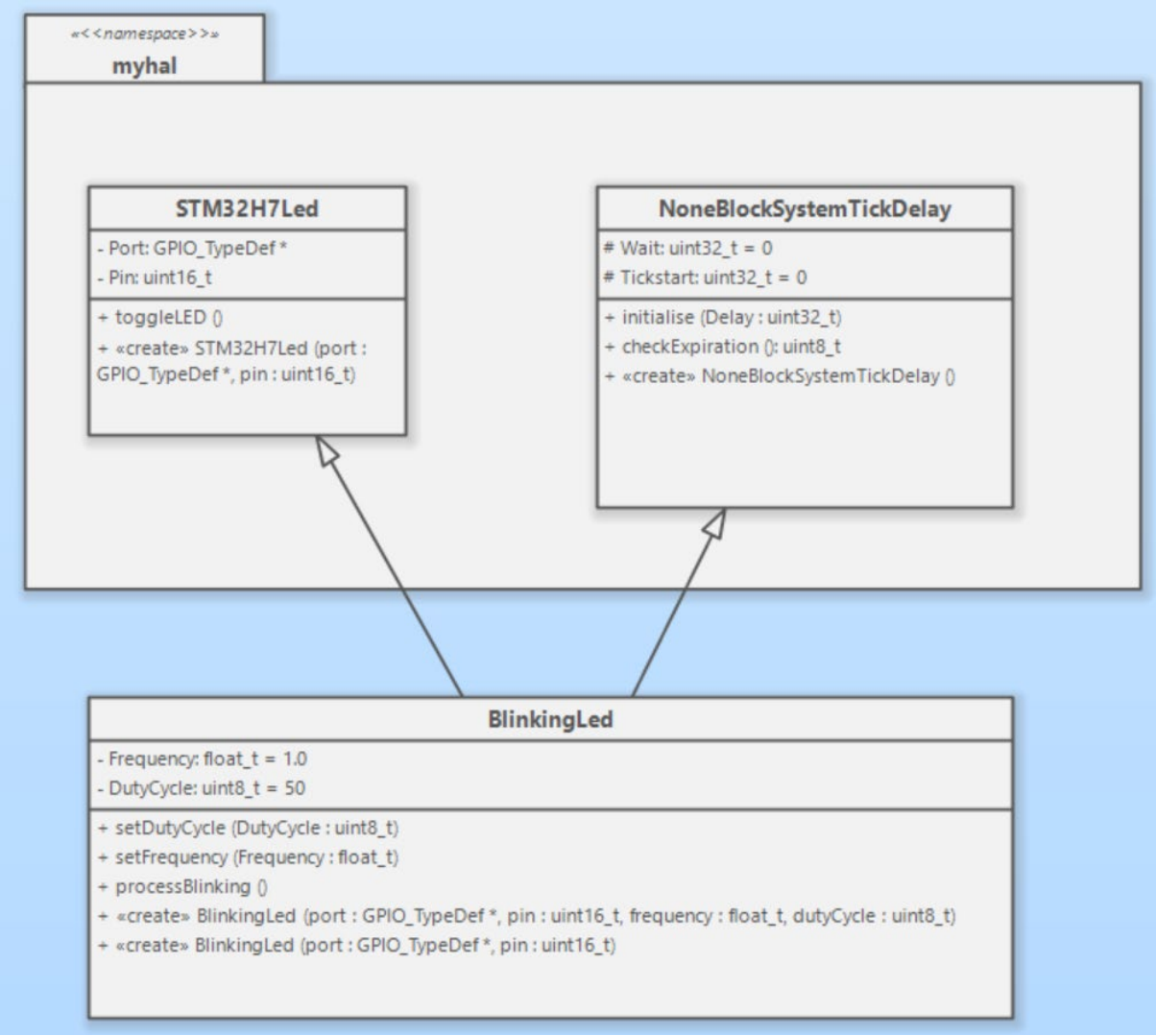

STM32H7Led
- Port: GPIO_TypeDef *
- Pin: uint16_t
+ toggleLED ()
+ «create» STM32H7Led (port : GPIO_TypeDef *, pin : uint16_t)
NoneBlockSystemTickDelay
Wait: uint32_t = 0
Tickstart: uint32_t = 0
+ initialise (Delay : uint32_t)
+ checkExpiration (): uint8_t
+ «create» NoneBlockSystemTickDelay ()
BlinkingLed
- Frequency: float_t = 1.0
- DutyCycle: uint8_t = 50
+ setDutyCycle (DutyCycle : uint8_t)
+ setFrequency (Frequency : float_t)
+ processBlinking ()
+ «create» BlinkingLed (port : GPIO_TypeDef *, pin : uint16_t, frequency : float_t, dutyCycle : uint8_t)
+ «create» BlinkingLed (port : GPIO_TypeDef *, pin : uint16_t)

Task 02_2_CPP_BlinkyNamespace

Expand project CPP_Blinky_02_01 to CPP_Blinky_02_02

Define a namespace: “myhal”

And include STM32H7Led and NoneBlockSystemTickDelay into this namespace.

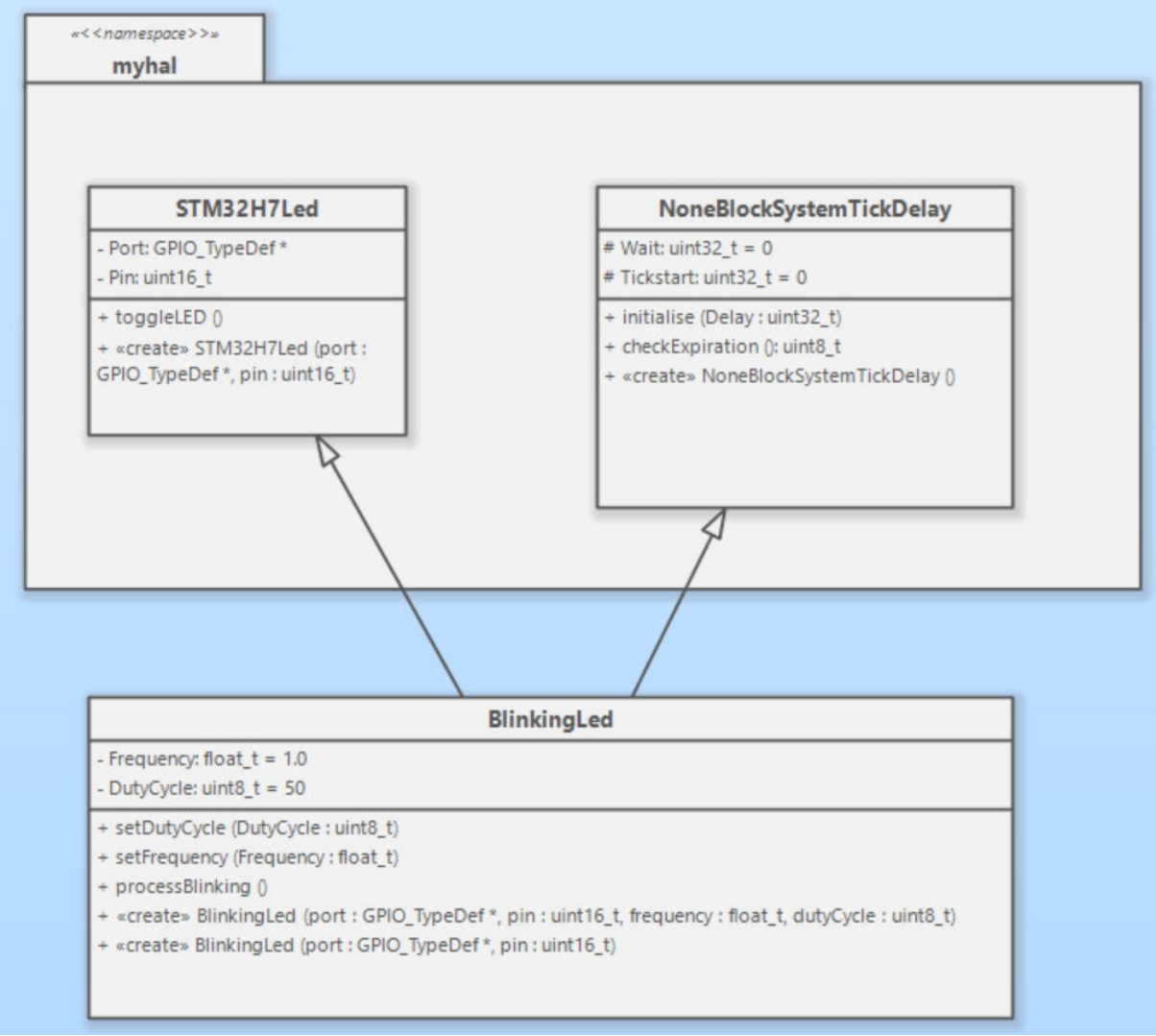


Task 02_3_CPP_BlinkyConst

Expand project: CPP_Blinky_02_02 to CPP_Blinky_02_03

Try to use:

Const, consteval and constexpr as often as possible on the methods and attributes!



Task 02_4_CPP_Template

Create a new project: CPP_Template_02_04.

Create a Array of int_16 with 6 Elements defined randomly.

Create a template function which calculates the average value of an array.

Pass the array to the template function and store the average value in a variable.

Task 02_5_CPP_Template

Extend the project: Template _02_04 to CPP_ Template _02_05

Use the sort algorithm of std and sort only the last 3 Elements in the array.