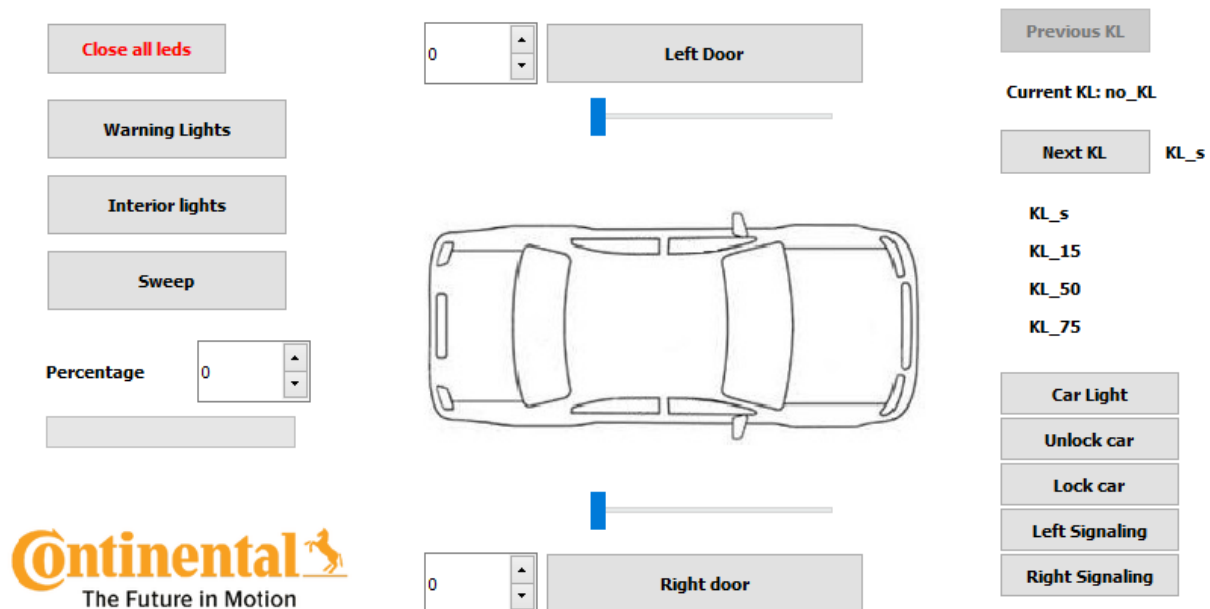


Laboratory 1 – Interior lights application

Interior lights interface



Exercise 1: Open and close one “led”

0.5p

Close all leds button must clear all the “leds” when is pressed.

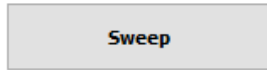
Complete the **close_all_leds** function in order to do this action.

Interior lights button must open and close 1 LED when is pressed using **set_interior_lights** function.

Complete the specific function to do this action.

Exercise 2: Sweep all leds

0.5p




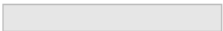
button must give “One led at a time” effect when is pressed.

MyThread_sweep needs to be used to make the state machine, and **sweep_leds** function will be used in order to switch the states emitted by thread to change “leds” state.

HINT: Use **set4leds** in order to change colors for “leds”.

Exercise 3: Control led brightness

0.5p

Percentage spin box  is scaled between 0-100, this means the led brightness percentage. The progress bar  0% is also scaled between 0-100 and must go through all the values until his value is equal with led brightness percentage, on a real led it would give the fade effect.

Progress bar value is stored in a variable and in the **valuechange** function is checked if it is bigger or smaller than the spin box value.

If progress bar value is **smaller** than the spin box value you should **fade out** from the actual brightness (meaning the progress bar percentage) to the next brightness (meaning the spin box percentage).

Change “**change_pb_down_value**” function to make this action work.

If progress bar value is **bigger** than the spin box value you should **fade in** from the actual brightness (meaning the progress bar percentage) to the next brightness (meaning the spin box percentage).

Change “**change_pb_up_value**” function to make this action work.

HINT: Use **ENTER key** after setting the value you want for the spin box.

Exercise 4: KL control

0.5p

KL is the abbreviation from 'klemme', which is the German term for connector/connection.

no_KL – all the “leds” are closed



KL_S – **I1**, grey color (is the ignition switch position #1 - accessory)

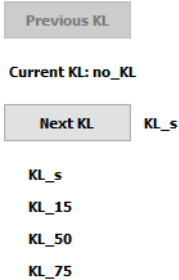
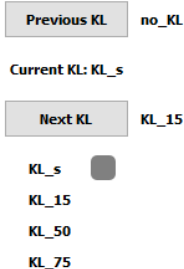
KL_15 – **I2**, green color (ignition switch position #2 – ON)

KL_50 – **I3**, red color (ignition switch position #3 – start, CRANKING)

KL_75 – **I4**, blue color (ENGINE RUNNING)

*I1,I2,I3,I4 are **set_bg_colors** arguments

KL_list is a list that contain all the KL's. Using  and  you have to go through all this list and set the following status for current KL:

Klemme	Leds state	Visual representation
no_KL	all 4 leds closed	
KL_S	just KL_S led open	

KL_15	KL_s, KL_15 leds open	<div> <div>Previous KL</div> <div>KL_s</div> </div> <div>Current KL: KL_15</div> <div> <div>Next KL</div> <div>KL_50</div> </div> <div> <div>KL_s</div> <div>KL_15</div> <div>KL_50</div> <div>KL_75</div> </div>
KL_50	KL_s, KL_15, KL_50, KL_75 leds open	<div> <div>Previous KL</div> <div>KL_15</div> </div> <div>Current KL: KL_50</div> <div> <div>Next KL</div> <div>KL_75</div> </div> <div> <div>KL_s</div> <div>KL_15</div> <div>KL_50</div> <div>KL_75</div> </div>
KL_75	KL_s, KL_15, KL_50, KL_75 leds open	<div> <div>Previous KL</div> <div>KL_50</div> </div> <div>Current KL: KL_75</div> <div> <div>Next KL</div> </div> <div> <div>KL_s</div> <div>KL_15</div> <div>KL_50</div> <div>KL_75</div> </div>

Complete **KL_lights** function to make this application work.

Description of existing functions:

set_bg_colors – setting the color of each led.

prev_kl_function - when


Previous KL

 button is pressed it changes the current KL to the previous KL.

next_kl_function – when

Next KL

 button is pressed it changes the current KL to the next KL.

set_enable – set  button to disable when current KL is no_KL and set the “Next KL” button to disable when current KL is KL_75.

Exercise 5: Obstacle detection

0.5p

Let’s start talking about the widgets that we use to make this exercise running. On the left side of the “Left Door”/”Right door” buttons there is a spin box. This spin box is actually the **obstacle**. The spin box is scaled between 0-100 (meaning the distance in cm that a door can open). Setting a value between 0-100 we fix an obstacle at the given value.



There are also 2 sliders, those sliders are also scaled between 0-100.

The “Left Door”/”Right door” button are making all the magic, because **after** we set the obstacle we just press one of these buttons to simulate an open door.

open_door_left/open_door_right are the functions handler for above buttons. When they are pressed the function should make the slider go through all the values until the obstacle value is reached, not more.

valuechange_right_slider/valuechange_left_slider are the functions handler for the sliders, when the slider is moved these functions are called. This functions should stop the slider moving to values bigger than the spin box value (obstacle).

*value **0** in the spin box is considered “**no obstacle present**”.

Complete above specified functions to make the obstacle detection work.

Info! exercises 6,7,8,9,10 use **setWarningLights** to set the color of the LEDs

Exercise 6: Warning Lights

0.5p

Warning Lights

When the button is pressed for the first time 4 LEDs will flash until the button is pressed again. If **Right Signaling** or **Left Signaling** are on, they must be switched off during the operation of the **Warning Lights**, after that it must resume their functionality.

Complete the specific functions to do this action.

Exercise 7: Left Signaling

0.5p

Left Signaling

When the button is pressed for the first time 2 LEDs will flash until the button is pressed again.

If **Warning Lights** are **on** while **Left Signaling** is actioned, warning lights should stay **on hold** while left signaling is executed and after it is stopped, warning lights must continue the execution until it is stopped.

If **Right Signaling** is on while **Left Signaling** is actioned, it must be set to **off** and continue only with the left signaling action.

Complete the specific functions to do this action.

Exercise 8: Right Signaling

0.5p

Right Signaling

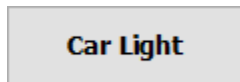
When the button is pressed for the first time 2 LEDs will flash until the button is pressed again.

If **Warning Lights** are **on** while **Right Signaling** is actioned, warning lights should stay **on hold** while right signaling is executed and after it is stopped, warning lights must continue execution until it is stopped.

If **Left Signaling** is on while **Right Signaling** is actioned, it must be set to **off** and continue only with the left signaling action.

Complete the specific functions to do this action.

Useful for next exercises:

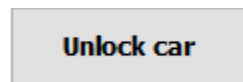


The button must turn on and off a led. Complete **carLight** function to do this. (as in **set_interior_lights** function)

*use **setcarLight** function to light the "led"

Exercise 9:

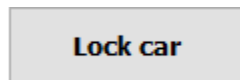
0.5p



When the button is pressed **Warning Lights** must flash for 2 times and the **Car Light** must be switched on if it is off. **Car Light** must be switched off after the care is successfully unlocked.

Exercise 10:

0.5p



When the buttons is pressed **Warning Lights** must flash 1 time and **Car Light** must be switched off if it is on.