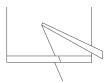
Bring ideas to life

VIA University College



## From Design to C Part I

ESW1

## From Design to C

How to come from a Design (UML) to C-code

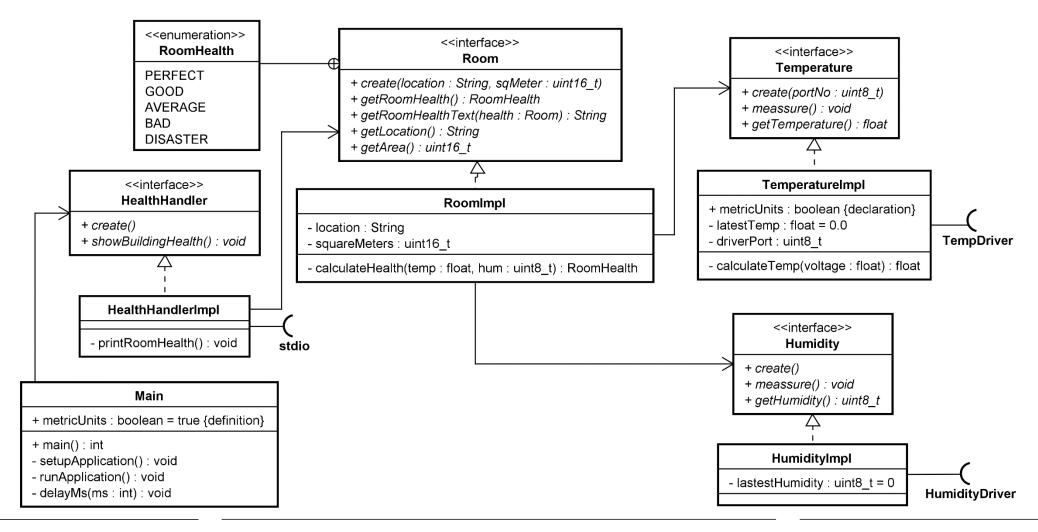
The following example design is for a small application that measures and shows the environment (health) in a single room (living room) in a building

The health is based on a temperature- and a humidity-measuring in the room.

The health should be shown once per second

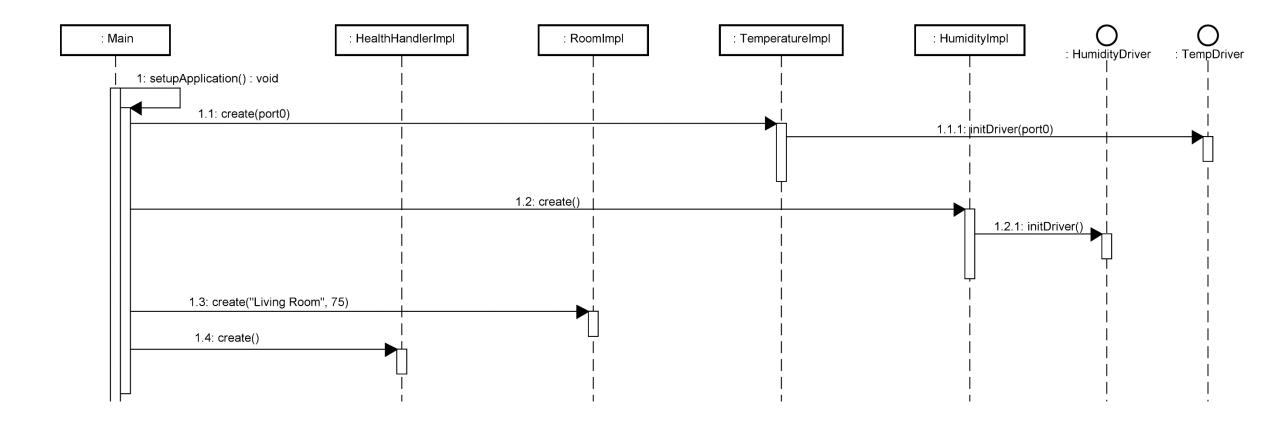
The sensors are all simulated in this example

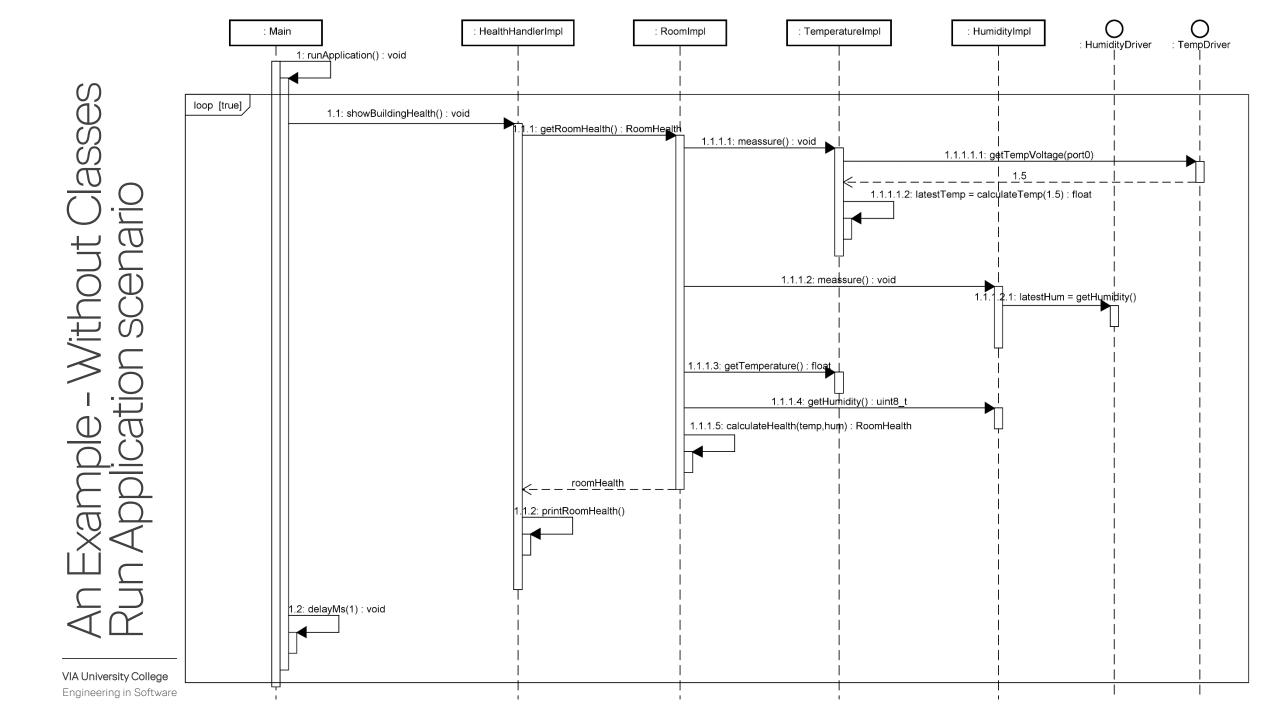


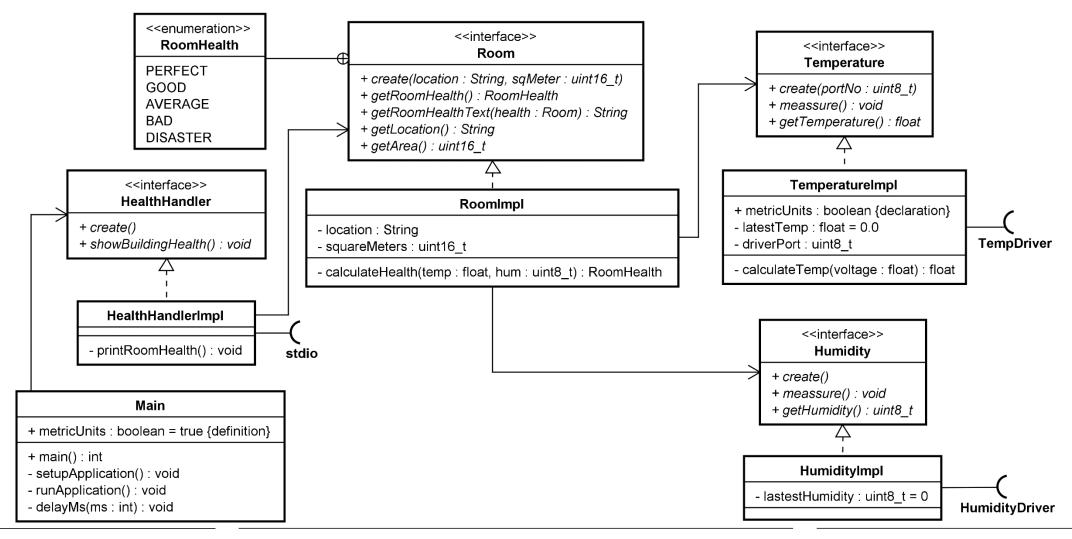


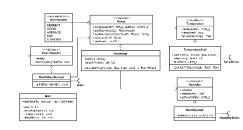
3

## An Example Without Classes – Setup Application scenario









### temperature.h

```
#pragma once

<interface>>
    Temperature

+ create(portNo : uint8_t)
+ meassure() : void
+ getTemperature() : float

#pragma once
#include <stdint.h>

void temperature_create(uint8_t portNo);
void temperature_meassure(void);
float temperature_getTemperature(void);
```

Why are the prefix temperature used?

- A kind of namespace to make the names unique in the application
  - Use the name of the interface as the prefix

<<interface>>
Temperature

- + create(portNo : uint8\_t)
- + meassure(): void
- + getTemperature() : float

<u>Д</u>

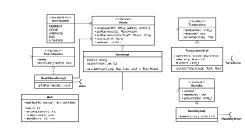
#### **TemperatureImpl**

- + metricUnits : boolean {declaration}
- latestTemp : float = 0.0
- driverPort : uint8\_t
- calculateTemp(voltage : float) : float

Transpared

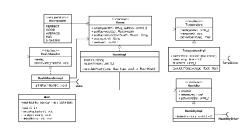
Frequency

\_\_\_\_( TempDriver



#### temperature.c

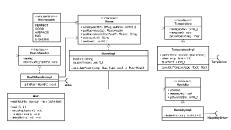
```
#include "temperature.h"
                                       #include <tempDriver.h>
        <<interface>>
        Temperature
    + create(portNo : uint8 t)
                                       extern bool temperature_metricUnits; // Declaration
    + meassure(): void
    + getTemperature(): fleat
                                       static float _latestTemp = 0.0; static means module (file) scope
                                       static uint8_t _driverPort;
                                                                                  Kind of private
      TemperatureImpl
+ metricUnits : boolean {declaration}
                                     static float calculateTemp(float voltage) {
- latestTemp : float = 0.0
                                          return 15.0+(voltage * 5.0); // dummy calc
                             TempOriver
- driverPort : uint8 t-
- calculateTemp(voltage : float) : float
                                       void temperature create(uint8 t portNo) {
                                           driverPort = portNo;
                                           temperatureDriver initDriver(portNo);
   VIA University College
                               From Design to C F
    Engineering in Software Technology
```

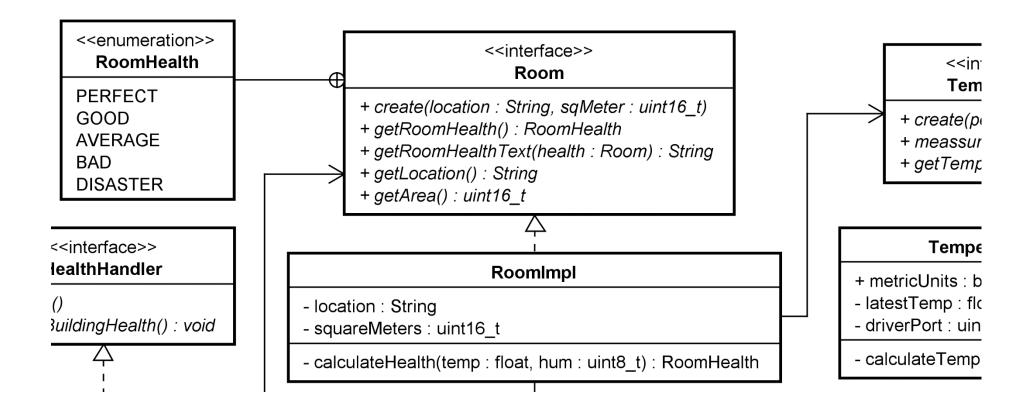


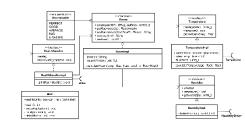
### temperature.c-continued

```
<<interface>>
           Temperature
     + create(portNo : uint8 t)
     + meassure(): void
     + getTemperature() : float
         TemperatureImpl
+ metricUnits : boolean {declaration}
- latestTemp : float = 0.0
                                         TempDriver
- driverPort : uint8 t
- calculateTemp(voltage : float) : float
```

```
void temperature_meassure(void) {
   _latestTemp =
   _calculateTemp(temperatureDriver_getVoltage());
float temperature_getTemperature(void){
   float _tmp = _latestTemp;
   if (temperature metricUnits) {
      tmp *= 0.2; // dummy conversion
   return _tmp;
```

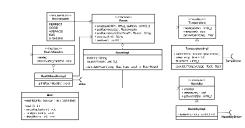






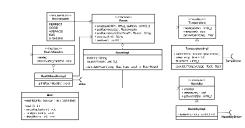
#### room.h

```
#pragma once
                                                                     #include <stdint.h>
                                                                     typedef enum {
 <<enumeration>>
  RoomHealth
                                 Room
                                                                           PFRFFCT
                                                                Tem
 PERFECT
                      + create(location : String, sqMeter : uint16 t)
 GOOD
                                                             + create(p
                      + getRoomHealth() : RoomHealth
                                                                            , GOOD
 AVERAGE
                                                             + meassur
                      + getRoomHealthText(health: Room): String
                                                             + getTemr.
                      + getLocation(): String
 DISASTER
                                                                            , AVERAGE
                      + getArea() : uint16_t
                                                                           , BAD
                                                               Tempe
<<interface>>
lealthHandler
                               RoomImpl
                                                            etricUnits : b
                                                                           ,DISASTER
                                                          - latestTemp : flc
                  - location : String
BuildingHealth(): void
                                                          driverNort : uin
                  - squareMeters : uint16 t
                                                                      }room roomHealth t;
                                                            Iculate 1 mp
                  · calculateHealth(temp : float, hum : uint8_t) : Roon Heal
                                                                     void room_create(char* location, uint16_t sqMeter);
                                                                     room roomHealth t room getRoomHealth(void);
                                                                     char* room_getRoomHealthText(room_roomHealth_t health);
                                                                     char* room getLocation(void);
                                                                      uint16 t room getArea(void);
```



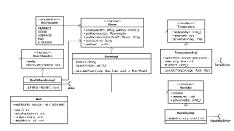
#### room.c

```
#include <string.h>
                                          #include "room.h"
                                          #include "temperature.h"
           <<interface>>
            Room
                                       #include "humidity.h"
   + create(location ; String, sqMeter : uint16_t)
   + getRoomHealth RoomHealth
   + getRoomHealthTex (health : Room) : St
   + getLocation() : String
                                          static char location[30] = { 0 };
   + getArea() : uint16 t
                                      static uint16 t squareMeters;
           RoomlmpL
                                  metricUnits : b
- location : String
                                  latestTemp : flo
- squareMeters : uint16
                                          static room_roomHealth_t _calculateHealth(float temp, uint8_t hum)
calculateHealth(temp : float, hum : uint
                                               return (uint16_t)(temp * hum) % 5;// dummy calculation;
                                     + create()
                                          void room create(char* location, uint16 t sqMeter) {
                                              // Why does it look like this??
                                               strncpy( location, location, sizeof( location) - 1);
    VIA University College
                                     From Des
                                               squareMeters = sqMeter;
    Engineering in Software Technology
```

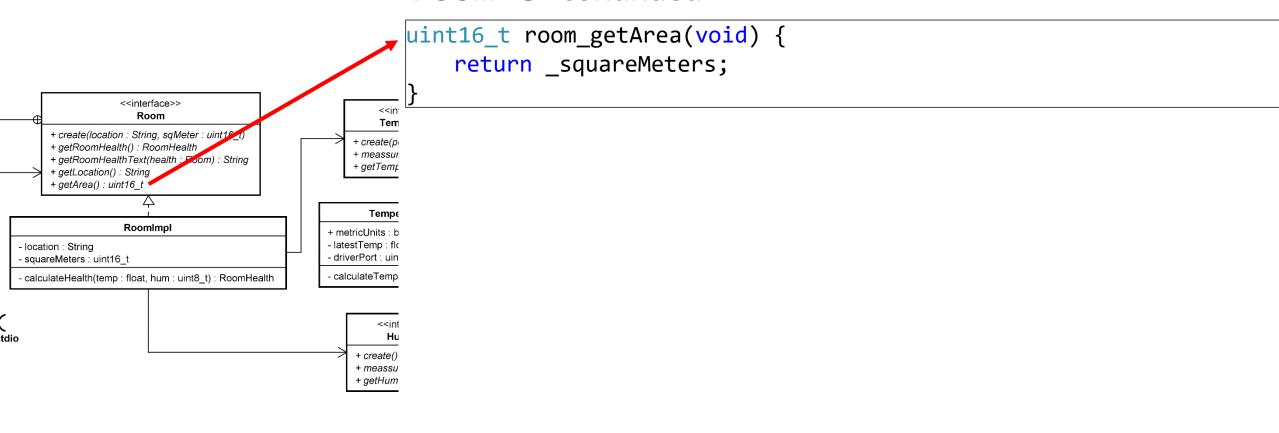


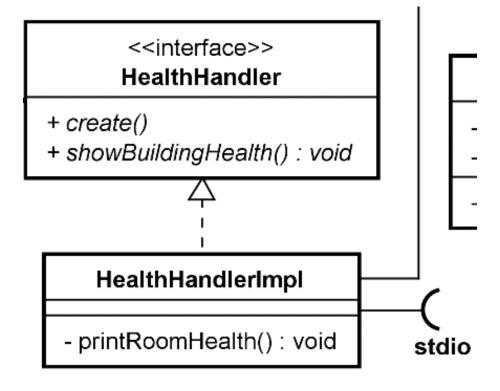
#### room. C - continued

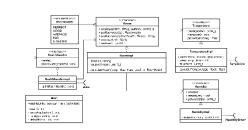
```
room roomHealth t room getRoomHealth(void) {
                                                   temperature meassure();
                                                   humidity_meassure();
            <<interface>>
                                          Tem
   + create(location : String, sqMeter .
                                        + create(p
   + getRoomHealth() : RoomHealth
                                                   return calculateHealth(temperature getTemperature(),
                                        + meassu
   + getRoomHealthText(health : Room) : String
                                        + getTemr.
   + getLocation() : String
                                                   humidity getHumidity());
    + getArea() : uint16_t
                                         Tempe
            RoomImpl
                                       etricUnits : b
                                     - latest temp : flc
- location : String
- squareMeters : uint16 t
                                              char* room getRoomHealthText(room roomHealth t health) {
- calculateHealth(temp : float, hum : uint8_t) : RoomNealth
                                                   static const char* health text[] = { "PERFECT", "GOOD",
                                                    "AVERAGE", "BAD", "DISASTER" };
                                          <<int
                                                   return health text[health];
                                        + create()
                                        + getHum
                                              char* room getLocation(void) {
                                                   return _location;
     VIA University College
                                        From Des
     Engineering in Software Technology
```

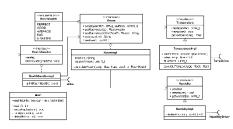


#### room. C - continued







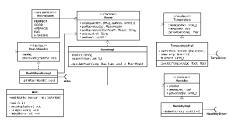


#### healthHandler.h

```
#pragma once
void healthHandler_create(void);
void healthHandler_showBuildingHealth(void);
```

```
<<interface>>
HealthHandler
+ create()
+ showBuildingHealth() : void
```

17



#### healthHandler.c

```
RoomHealth
                                              Ro
                                 + create(location : Stri #include < stdio.h>
       PERFECT
       GOOD
                                 + getRoomHealth(): 🖊
                                                  #include "healtHandler.h"
       AVERAGE
                                 + getRoomHealthText
       BAD
                                 + getLocation #include "room.h"
       DISASTER
                                 + getArea . wht16 t
                                                  static void _printRoomHealth(void) {
     <<interface>>
     HealthHandler
                                                      printf("%s: area: %d m2 Health: %s\n",
+ create()
                            - location String
                                                      room_getLocation(), room_getArea(),
+ showBuildingHealth(): voice
                            - square Meters: uint16 t
                                                      room getRoomHealthText(room getRoomHealth()));
                            - calculateHealth(temp: float,
   HealthHandlerImpl
                                                  void healthHandler_create(void) {
 - printRoomHealth(): void
                        stdio
                                                      // do something
                                                  void healthHandler showBuildingHealth(void) {
                                                      printRoomHealth();
   VIA University College
                                From Design to C Part I - Ib Havn, iha
   Engineering in Software Technology
```

# | Value | Park |

#### Main

+ metricUnits : boolean = true {definition}

+ main(): int

- setupApplication(): void

- runApplication(): void

- delayMs(ms:int): void

## An Example Without Classes main.c

```
<<interrace>>
               HealthHandler
        + create()
        + showBuildingHealth(): void
             HealthHandlerImpl
          - printRoomHealth(): void
                  Main
+ metricUnits : boolean = true {definition}
+ main(): int
- setupApplication(): void
- runApplication(): void
- delayMs(ms: int): void
```

```
#include <time.h>
       #include <stdbool.h>
       #include "temperature.h"
                                      To be able to initialise
       #include "humidity.h"
                                      these modules
       #include "room.h"
       #include "healtHandler.h"
       bool temperature_metricUnits = true; // Definition
       static void _delayMs(int milliseconds)
           long pause;
           clock_t now, then;
           pause = milliseconds * (CLOCKS_PER_SEC / 1000);
           now = then = clock();
           while ((now - then) < pause)</pre>
             now = clock();
From Design to C Part
```

<<interrace>>

HealthHandler

+ showBuildingHealth(): void

**HealthHandlerImpl** 

- printRoomHealth(): void

Main

+ metricUnits : boolean = true {definition}

+ create()

#### main.c-continued

```
TOTAL CONTROL OF THE PROPERTY OF THE PROPERTY
```

```
static void _setupApplication(void) {
   temperature create(0);
   humidity create();
   room create("Living Room", 75);
   healthHandler create();
static void runApplication(void) {
   while (1) {
       healthHandler_showBuildingHealth();
       delayMs(1000);
int main(void) {
   _setupApplication();
   _runApplication();
   return 0;
```

+ main() : int-

setupApplication(): voidrunApplication(): void

- delayMs(ms: int): void

## Exercise - Going from Design to C-code

1. Design a small system using UML (Class- and Sequence-Diagrams)

The criteria's for the design:

- The system must be small like the example I have shown today
- The system does not need to give completely meaning
- The system must only use a multiplicity of one between the classes
- 2. Check that your design can handle the wanted functionality
  - Check it by using your Sequence Diagrams

From Design to C Part I - Ib Havn, iha@via.dk

3. Implement the design in C