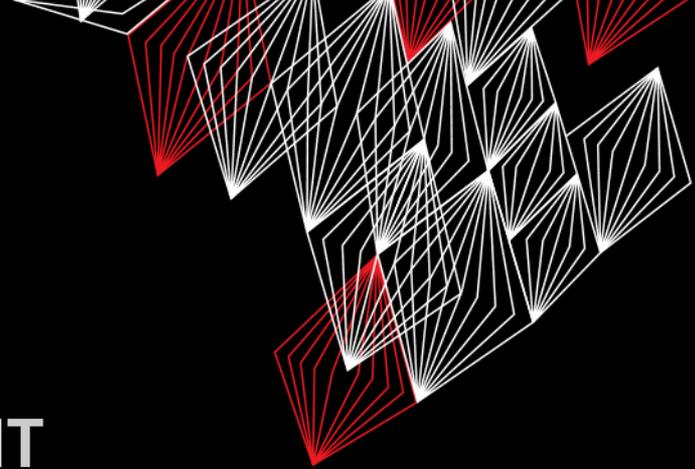


UNIVERSITY OF TWENTE.



# APPLICATION DEVELOPMENT

## LECTURE 7: ARDUINO EXAMPLES

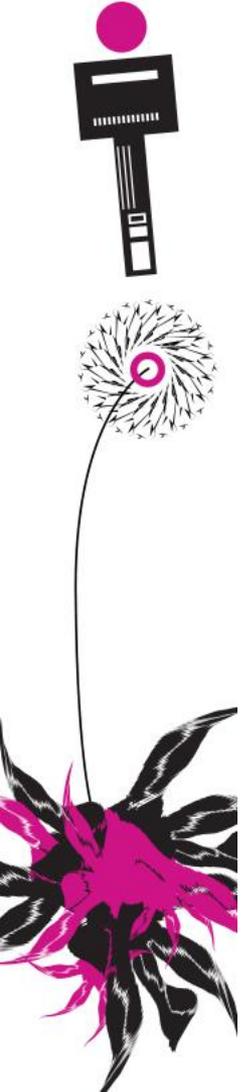
```
class AppDev {
```



```
}
```



Part of **SmartProducts**



# INTRODUCTION

## APPLICATION DEVELOPMENT

- Arduino examples
- Project: report
- Assignment

slides @ [vanslooten.com/appdev](https://vanslooten.com/appdev)

UNIVERSITY OF TWENTE.



Fjodor van Slooten  
W241 (*Horst-wing West*)  
f.vanslooten@utwente.nl



Next week: no lecture, but we are available for help (chat & forum)

# ASSIGNMENT 5

---

```
/**  
 * JavaDoc  
 */
```

- Add comments
- JavaDoc: at start of each class and at start of each method

```
/**  
 * Main drawing panel of the game. Contains all game elements.  
 * @author F. van Slooten  
 */  
public class DrawingPanel extends JPanel {  
  
    /**  
     * Draws & manages all elements and contains some of the game logic.  
     */  
    @Override  
    protected void paintComponent(Graphics g) {
```

- array
- bool
- boolean
- byte
- char
- double
- float
- int
- long
- short
- size\_t
- string
- String()
- unsigned char
- unsigned int
- unsigned long
- void
- word

# INTEGERS: HIGH NUMBERS

## ARDUINO DATA TYPES AND CONSTANTS

- int: 16-bit (2-byte) value: range of -32,768 to 32,767
- **unsigned int** (cannot be negative!): 0 – 65536
- long: (4-byte) -2,147,483,648 to 2,147,483,647
- **unsigned long**: 0 – 4,294,967,296

long integer

### Constants

- HIGH | LOW
- INPUT | OUTPUT | INPUT\_PULLUP
- LED\_BUILTIN
- true | false

### Floating Point Constants

### Integer Constants

### Conversion

- (unsigned int)
- (unsigned long)
- byte()
- char()
- float()
- int()
- long()
- word()

```
int i = 32767;
i = i + 1;
// what is value of i??
```

# TURN ON/OFF

- Add an on/off button to your prototype
- Do not show anything on display, until start button is pressed:

Initialize display, but do not show anything yet (it remains black: looks like it is off)

Wait until button is pressed

```
#define START_BTN 4

void setup() {
  Serial.begin(9600);
  pinMode(START_BTN, INPUT_PULLUP); // start button, without resistor (internal pull-up)
  display.begin();
  display.setPowerSave(0);
  display.setFont(u8x8_font_pxplusibmcgathin_f);

  // wait until button pressed to proceed:
  // (this while statement halts the program, until the button is pressed)
  while(digitalRead(START_BTN)==HIGH) ; // note the empty statement: ';' does nothing,
  // thus waits for START_BTN to be pressed (becomes LOW)
}
```

Take care: mount button properly!

No resistor: use internal pull-up resistor



# COUNT DOWN TIMER

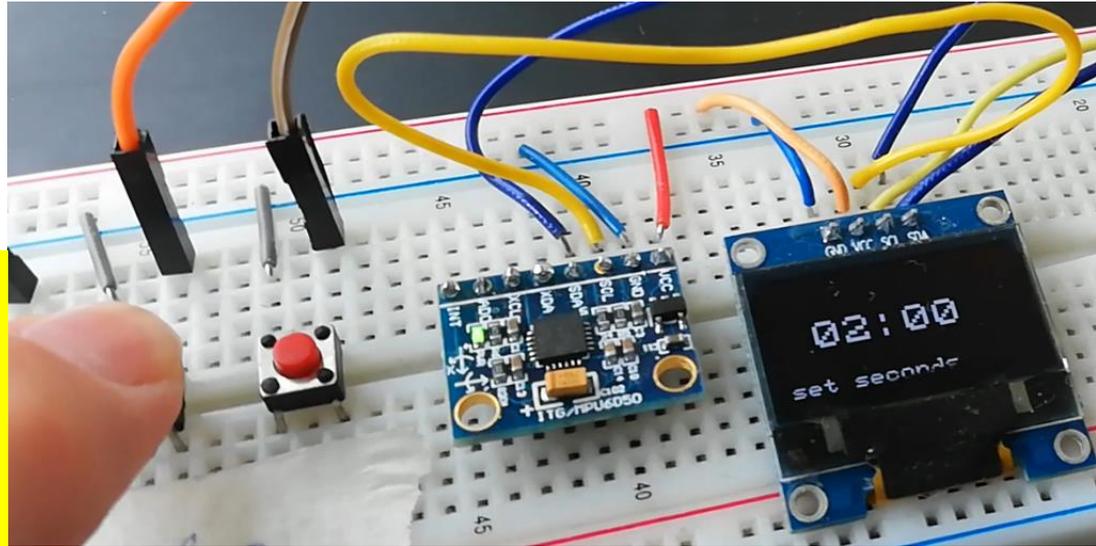
- Display a count down timer
- Either as a single number, or as time counting down

```
int count = 90;

void loop() {
  if (count >= 0) { // if counter not finished
    int min = count / 60;
    int sec = count % 60;
    sprintf(buf, "%02d:%02d", min, sec);
    display.draw2x2String(2, 3, buf);
    count = count - 1; // decrease counter
  }
  else {
    display.drawString(0, 7, "finished");
  }
  delay(1000); // wait one second
}
```

Display time as  
MM:SS

Do something  
when counter  
finished



 [youtu.be/eDKBeUZpoys](https://youtu.be/eDKBeUZpoys)

Includes advanced version which can also  
set the time of the timer and start over

[More info: Countdown timers and  
executing tasks in parallel on an Arduino](#)

# EXECUTING TASKS IN PARALLEL

- Avoid `delay()`
- Use `millis()` to count milliseconds

[Blog article “Countdown timers and executing tasks in parallel on an Arduino”](#)

```
void loop() {
    unsigned long currentMillis = millis();

    // task 1 - blink time on display every 300ms when setting it:
    if (currentMillis - previousMillis2 > 300 ) { // 300ms passed?
        previousMillis2 = currentMillis; // save the last time
        blink = !blink;
        // ... display blinking numbers when setting the time ...
    }

    // task 2 - update countdown timer every second
    if (currentMillis - previousMillis > interval) { // interval passed?
        previousMillis = currentMillis; // save the last time
        // ... display changing countdown timer on the display ...
    }

    // task 3 - check for pressed buttons
    for (int i = 0; i < NUMBUTTONS; i++) {
        // Update the Bounce instance:
        buttons[i].update();
        if ( buttons[i].fell() ) { // a button was pressed
            // ... process actions for buttons ...
        }
    } // end for
}
```

# OLED DISPLAY GRAPHICS

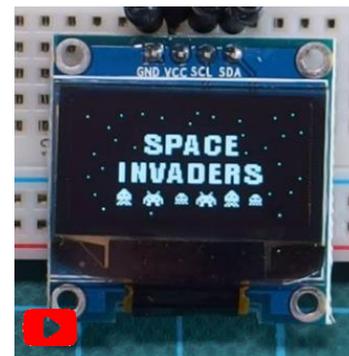
- [u8g2 / u8x8 library](#)



## [Using an OLED Display with Arduino:](#)

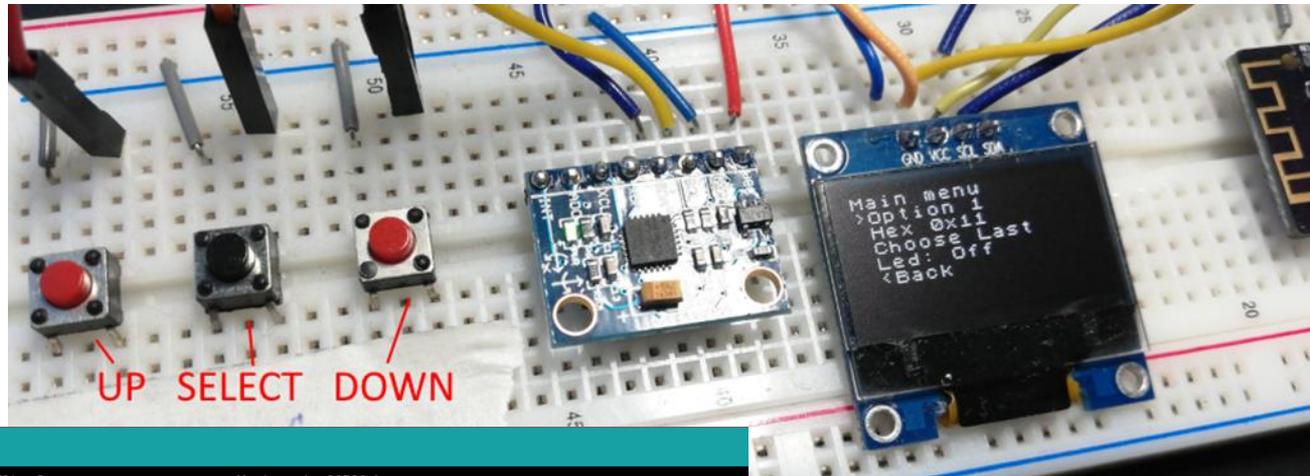
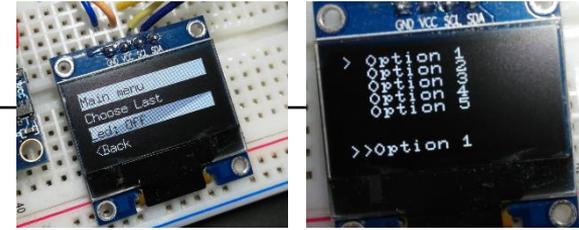
### More info

- [u8x8 library reference](#)
- [Adafruit GFX Graphics Library & Monochrome OLED Breakouts](#)
- [Extensive tutorial with info on drawing graphics and display bitmap images.](#)
- [Another extensive tutorial "Interface OLED Graphic Display Module with Arduino"](#)
- [Pixel Art on OLED Display](#)



# ARDUINO: MENU SYSTEM

- [Blog article “How to make a menu”](#)
- OLED display + UP/DOWN/SELECT button
- 3 examples, from simple to advanced



Drawback of advanced systems like ArduinoMenu: uses lots of memory...

Done Saving

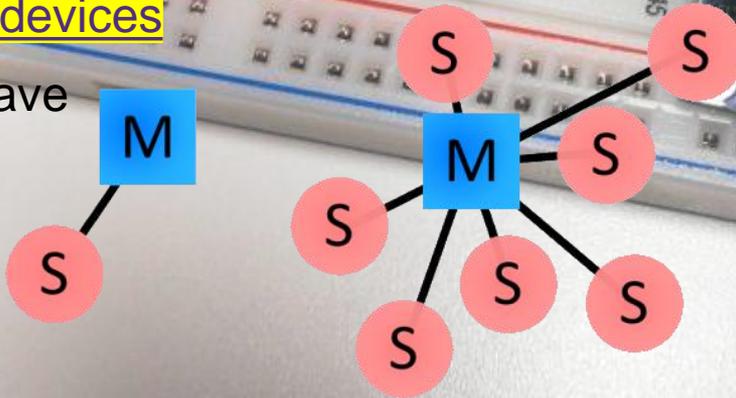
Sketch uses 26674 bytes (86% of program storage space. Maximum is 30720 bytes.  
Global variables use 1480 bytes (72% of dynamic memory, leaving 568 bytes for local variables. Maximum is 2048 bytes.

# BLE: TWO WAY COMMUNICATION

[Previous lecture: run Arduino on battery power \(slide 24\)](#)

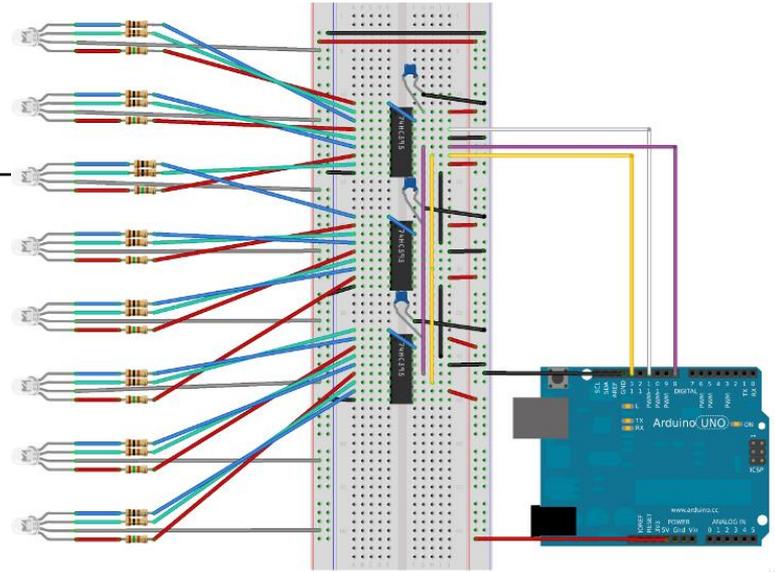
- How to setup Bluetooth communication between two Nano BLE devices

- Master - slave



# MULTIPLE RGB LEDs

- Connecting multiple 'traditional' RGB-leds requires 3 pins per LED, so that adds up quickly
- Alternative 1: use shift registers
- Alternative 2: addressable RGB strips, NeoPixels, or single addressable LEDs
- These can be chained
- Need only **1 pin** of the Arduino to control

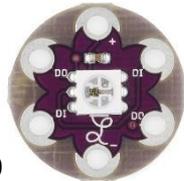


[Tutorial using 74HC595 shift registers](#)



€2.60

[NeoPixel Ring 8](#)



€1.30

[LilyPad Pixel](#)



€4.50 [WS2812B 8 RGB LEDs on a PCB](#)



€9.00

[WS2812B Digital 5050 RGB LED Strip - 30 LEDs 1m](#)

# GAME: PLAYERS... DESIGN CLASS PLAYER

---

- [Example project](#) with Controller which maintains a list of players
- Adding players
- Simulate scanned badges by random generation of uid's (for quick testing)

[Download example Arduino project](#)

```
void Userinterface::checkSensors() {  
    // scan for card  
    // uid scanned (here simulated random uid):  
    char letters[] = "abcdef0123456789";  
    String uid = "";  
    for (int i=0; i<12; i++) uid += letters[random(0, sizeof(letters))];  
    // add uid to list:  
    controller->add(uid, false, "red"); // add player with uid, thief=false, color  
    delay(5000); // wait 5 seconds  
}
```

```
class Player {  
private:  
    // class variables:  
    String uid, color;  
    bool thief;  
  
public:  
    // constructor:  
    Player(String u, bool t, String c);  
  
    // methods:  
    String getUID();  
    bool isThief();  
};
```

# REPORT

## HAND-IN SOFTWARE

Deadline report  
June 22th 17:00

Eg. example of  
mapping & class-  
diagrams/UML

Eg. flow-charts

[Week 10 | Report, prototype, manual](#)  
At the begin of week ten (Monday, June 22, 17:00), each group needs to hand in:  
- a report in which the developed product is described, together with the design rationale that underpins the product  
- a video demonstration of the group's prototype  
- a construction manual that allows others to build the technical part of the prototype.

In report:

- Design of software (requirements, class design, pseudo code, charts)
- Design rationale: **why...?** did you use/program/make software in this way? What would be different in real product?
- Appendix (digital, as part of zip-file):
  - Source code of all software (Arduino/C++; Eclipse; ... other)
  - Source code must be documented by using comments as you learned
  - Document external parts (used from online sources/libraries etc.): how did you use them?
  - Be clear on authors, refer to used sources/libraries/examples!

[How to hand-in as zip-file is explained here](#)

Hand-in **one** file: you must combine all files & folders into one zip-file!

# ASSIGNMENT #7

Next week: no lecture, but we are available for help (chat & forum)

- Free assignment
- You can do anything you like/wish/...
- E.g. a piece of code you must/want to write for the project
- A topic you would like to learn more about
- No inspiration? Sample assignments available
- [More info here](#)

There is an exam in the schedule at June 24<sup>th</sup>, but that will not be used! (**there is no exam**)

Slides, assignments etc @ [vanslooten.com/appdev](https://vanslooten.com/appdev)

This afternoon: **projects questions get priority**, questions about assignments might not be possible!