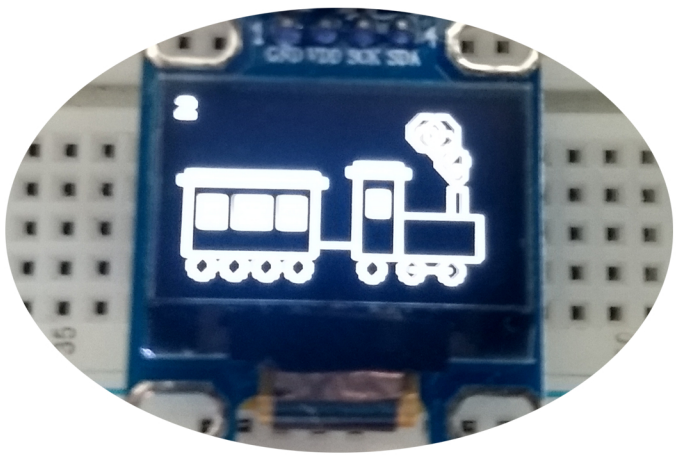
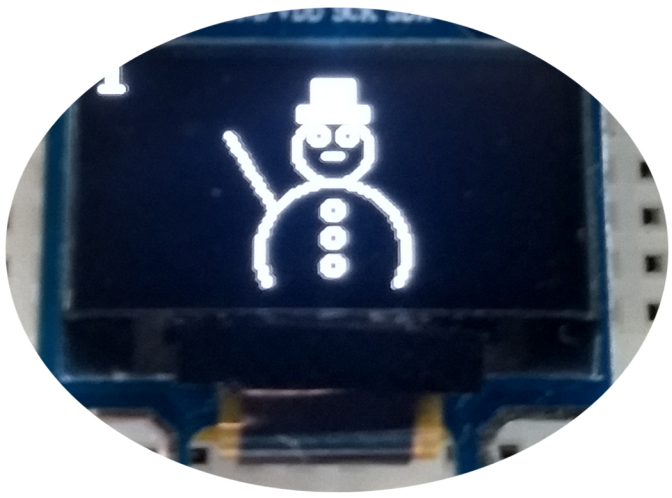
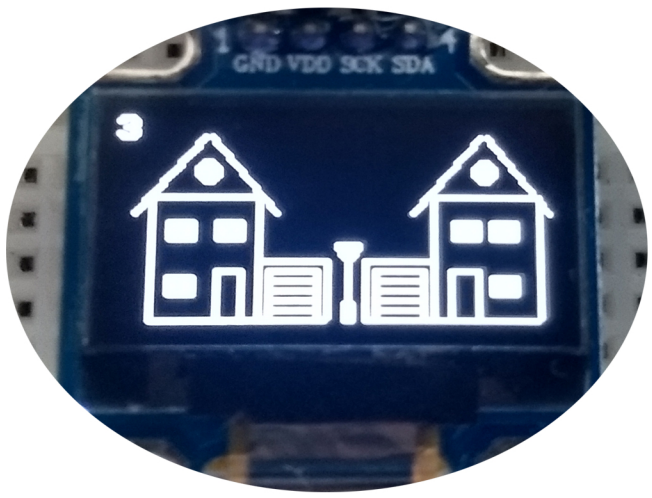
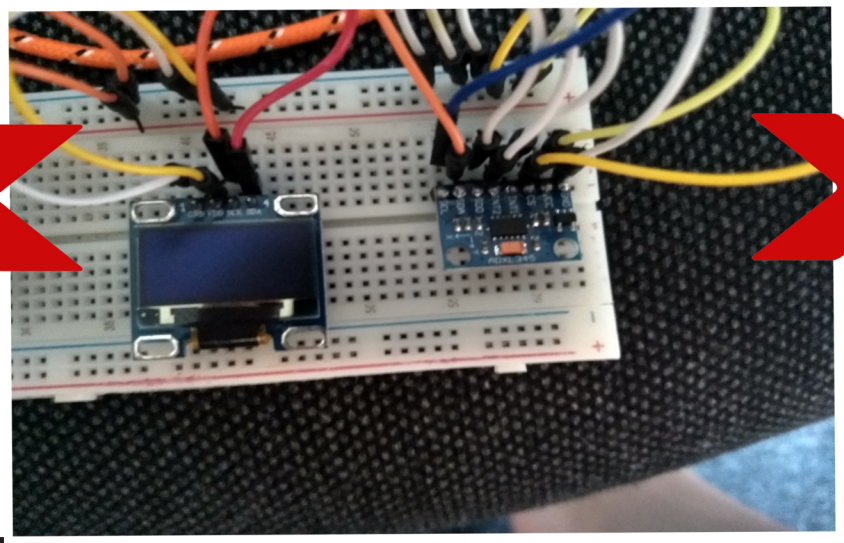
Acceslideratör









```
void images::snowman(){
    display.clear();
    display.flush();

    auto font          = hwlib::font_default_8x8();
    auto display_font       = hwlib::terminal_from( display, font );

    display_font
        << "\f" << "1"
        << hwlib::flush;

    hwlib::circle top_circle({64, 24}, 10);
    hwlib::circle bot_circle({64, 54}, 20);

    top_circle.draw(display);
    bot_circle.draw(display);

    hwlib::circle eyeL_circle({60,20}, 2);
    hwlib::circle eyeR_circle({68,20}, 2);
    eyeL_circle.draw(display);
    eyeR_circle.draw(display);

    hwlib::circle button1_circle({64,42}, 2);
    hwlib::circle button2_circle({64,50}, 2);
    hwlib::circle button3_circle({64,58}, 2);

    button1_circle.draw(display);
    button2_circle.draw(display);
    button3_circle.draw(display);

    filled_rectangle top_hat( display, 59, 4, 69, 15);
    filled_rectangle bottom_hat( display, 55, 12, 73, 15);
    filled_rectangle mouth( display,62, 26, 66, 26);
    top_hat.print();
    bottom_hat.print();
    mouth.print();

    hwlib::line left_arm({35,20}, {48,42});
    left_arm.draw(display);

    display.flush();
}
```

Features:

3 verwisselbare pixelarts op een 128x64 oled scherm.

Op basis van acceleratie wisselen van pixelart

Op basis van knoppen wisselen van pixelart

```
class adxl345{
    private:
        /// the i2c channel
        hwlib::i2c_bus & bus;
        uint8_t address_base;
        uint8_t address_read;
        uint8_t address_write;

    public:
        /// construct with both read and write adress
        adxl345( hwlib::i2c_bus & bus, uint8_t address_base =  0x53, uint8_t address_read = 0xA6 , uint8_t address_write = 0xA7):
            bus( bus ),
            address_base( address_base ),
            address_read( address_read ),
            address_write( address_write )

    {
        writeReg(ADXL345_ACT_INACT_CTL, 0x00);
        writeReg(ADXL345_THRESH_ACT, 0x00);
        writeReg(ADXL345_THRESH_INACT, 0xFF);
        writeReg(ADXL345_TIME_INACT, 0x00);
    }

    void writeReg( uint8_t reg, uint8_t value );
    uint8_t read8( uint8_t reg );
    int16_t read16( uint8_t reg );
    void powerOn();
    void setRangeSetting(int val = 2);
    void setBitRate(int val =100);
    void setAxisOffset(int x, int y, int z);
    void setInterrupts(bool b[8]);
    void setTreshTap(unsigned int val);
    void setTapDetection(bool b[4]);
    void setTapDuration(unsigned int val);
    void setLatent(unsigned int val);
    void setWindow(unsigned int val);
    void setTreshAct(unsigned int val);
    void setTreshInact(unsigned int val);
    void setInactTime(unsigned int val);
    void setActInactCtl(bool b[8]);
    void setFreeFallThresh(unsigned int val);
    void setFreeFalltime(unsigned int val);
};

#endif // ADXL345_HPP
```

github: https://github.com/RickvMourik/ipass.git