

A?

Aalto-yliopisto
Sähkötekniikan
korkeakoulu

Arduinon ohjelmointi

ELEC-A4010 Sähköpaja,

27.09.2021

Olli Auranen

Mistä liikkeelle?

- Mieti mitä haluat ohjelman tekevän
- Jaa kokonaiskuva toimintojen perusteella osiin

Esimerkki: Esteitä väistelevä auto

- Miten auto liikkuu?
 - *Moottorit? Ohjaus?*
- Miten tunnistan esteen?
 - *Etäisyysanturit? Mittaus?*
- Mitä auto tekee kun tunnistan esteen?
 - *Käänny kunnes este häviää?*

Esimerkki jatkuu...

- Miten auto liikkuu?
 - Moottoriohjain?
 - *Funktio jossa kerrotaan moottoriohjaimelle mitä moottoria pyöritetään ja mihin suuntaan*
- Miten tunnistan esteen?
 - Ultraäänianturi?
 - *Funktio, joka palauttaa ultraäänianturilta etäisyyden seuraavaan esteeseen*
- Mitä auto tekee kun tunnistan esteen?
 - Käänny kunnes este häviää?
 - *Voisin yhdistää edelliset kaksi funktiota*

Arduinon ohjelmointikieli

<https://www.arduino.cc/reference/en/>

Arduinon ohjelmointikieli on käytännössä C/C++:

Ohjelmointikielien tutut rakenteet löytyvät

- Muuttujat, funktiot
- Silmukat, laskutoimitukset, ehtolauseet

Ohjelmointikieli on laiteläheinen ja käännetään ennen lähettämistä

- Osa asioista vaikuttaa kankeammilta kuin esim. Pythonissa

Kommentit

Koodin sisälle voi kirjoittaa kommentteja

- esim. miten funktio toimii

Kirjoita ihmeessä kommentteja koodista

- Arduinossa rivikommentti aloitetaan kahdella kauttaviivalla (`//kommentti`)
- Usean rivin kommentti merkkien `/*` ja `*/` väliin

```
2 |
3 | //Yhden rivin kommentti
4 |
5 | /*
6 | Usean
7 | Rivin
8 | Kommentti
9 | */
10 |
```

Sisennys

```
void loop()
{
String nimi;
if (tila == tila1)
nimi = kysy_nimi();
if (tila == tila2)
sano_moi(nimi);
}
```

```
void loop()
{
String nimi;

if (tila == tila1)
nimi = kysy_nimi();
if (tila == tila2)
sano_moi(nimi);
}
```

- Tekee koodista helppolukuista
- ”Tab”, liikuttaa oikealle
- ”Shift + Tab”, liikuttaa vasemmalle

Muuttujat

<https://www.arduino.cc/en/tutorial/variables>

`int lamputila = 5;`
tyyppi nimi arvo

Muuttujalle määritetään:

1. Tyyppi

- desimaali (**float**), kirjain (**char**), tavu (**byte**)...
- Tyyppiä voi edeltää tyyppimääre, esim.
Vakiolle **const**

2. Nimi

3. Arvo

- Täytyy olla tyyppin mukainen
- ”Teksti”, 'a', 3.0

Muuttujatyypit laskutoimituksissa

<https://www.arduino.cc/en/tutorial/variables>

Arduino ei automaattisesti muunna muuttujatyyppejä

```
int kokonaisluku = 2;
float muuttuja1 = 3/kokonaisluku;           // 1
float muuttuja2 = 3.0/2.0;                  // 1.5
float muuttuja3 = 3.0/(float)kokonaisluku; // 1.5
```

- ”(tyyppi)muuttuja”-notaatio muuntaa muuttujan tyyppiin
- Muunnos isommasta tyypestä pienempään aiheuttaa ongelmia jos muunnettava arvo on liian iso. (long -> int)

Ali-/ylivuoto

- Mikäli muuttujan arvo menee muuttujatyyppin alueen yli, ”pyörähtää muuttuja yli”
- Byte-tyyppin maksimiarvo on 255

```
byte c = 0; // c = 0
--c;      // c = 255 // sama kuin c = c - 1;
c = c + 1; // c = 0
```

Tähän käyttäytymiseen ei kannata luottaa, koska se ei ole vakio.

Huom! Tähän saattaa törmätä millis()-funktion kanssa

Funktiot

<https://www.arduino.cc/en/Reference/FunctionDeclaration>

Funktio ottaa sisäänsä parametrejä, tekee niillä jotakin ja antaa jonkinlaisen paluuarvon.

```
int funktionimi(int parametri1, int parametri2)
{
    return paluuarvo;
}
```

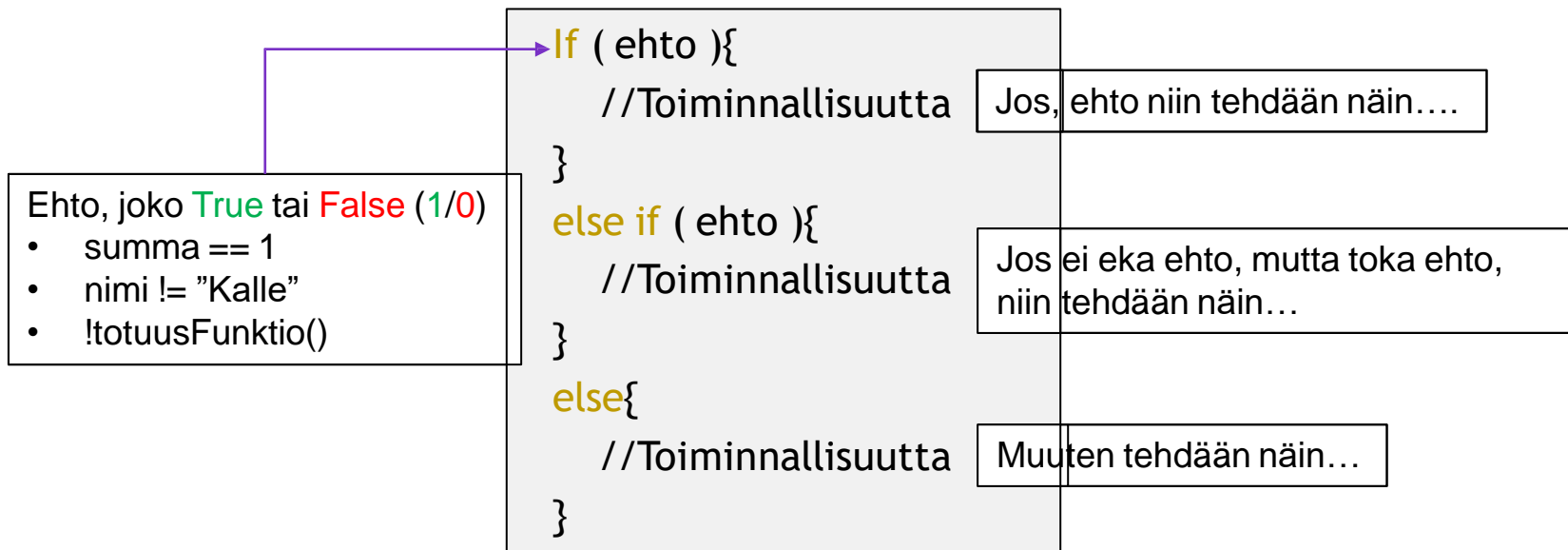
Funktiolle määritetään

1. **Paluuarvo ja tyyppi**
 - *Tyypiksi void, jos ei palauteta mitään*
2. **Funktion nimi**
3. **Parametrit ja niiden tyypit**
 - *Erotetaan pilkulla*

Funktion kutsu: `muuttuja = funktionnimi(muuttuja, muuttuja);`

Ehtolauseet

<https://www.arduino.cc/en/Tutorial/ifStatementConditional>



Taulukot

<https://www.arduino.cc/reference/en/language/variables/data-types/array/>

```
int led_pin[3] = {3, 5, 6};  
//led_pin[0] on nyt 3  
//led_pin[1] vastaavasti 5
```

- Tallennetaan useita arvoja yhden muuttujan alle
- Taulukon jäseniä kutsutaan alkioiksi
- Alkioon käsiksi syntaksilla **nimi**[**alkion nro**]
 - Indeksointi nolasta

```
led_pin[1] = 2;  
//led_pin vastaa nyt määritelmää {3, 2, 6}
```

Teksti

<https://www.arduino.cc/reference/en/language/variables/data-types/string/>

- C-kielessä ei erillistä tyyppiä tekstile, vain merkeille (char)
- Merkkijonot esitetään char-taulukkoina

```
char materiaali[] = "puu";  
// {'p', 'u', 'u', '\0'}
```

- Merkkijonon lopussa ns. nollatavu '\0'
 - Pystytään päättelemään mihin merkkijono päättyy.
 - Kolmen merkin merkkijonon tallentamiseen tarvitaan siis neljä merkkiä!

String-objekti

<https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/>

- C-kielessä tekstin käsittely kankeaa
- Arduinossa String-objekti jonka käyttö yksinkertaista

```
String string1 = "Hello There!";  
String string2 = String("General " + "Genobi!");  
String string3 = String( muuttuja tai numero );
```

[], +, +=, ==, >, >=, <, <= sekä !=
Operaattorit toimivat String-objekteilla

Globaalit ja lokaalit muuttujat

```
int a = 1;

void setup()
{
    int b = a + 2; //Toimii
}

void loop()
{
    a = b + 2; //Ei toimi, b näkyy
              //vain setup-funktiossa
}
```

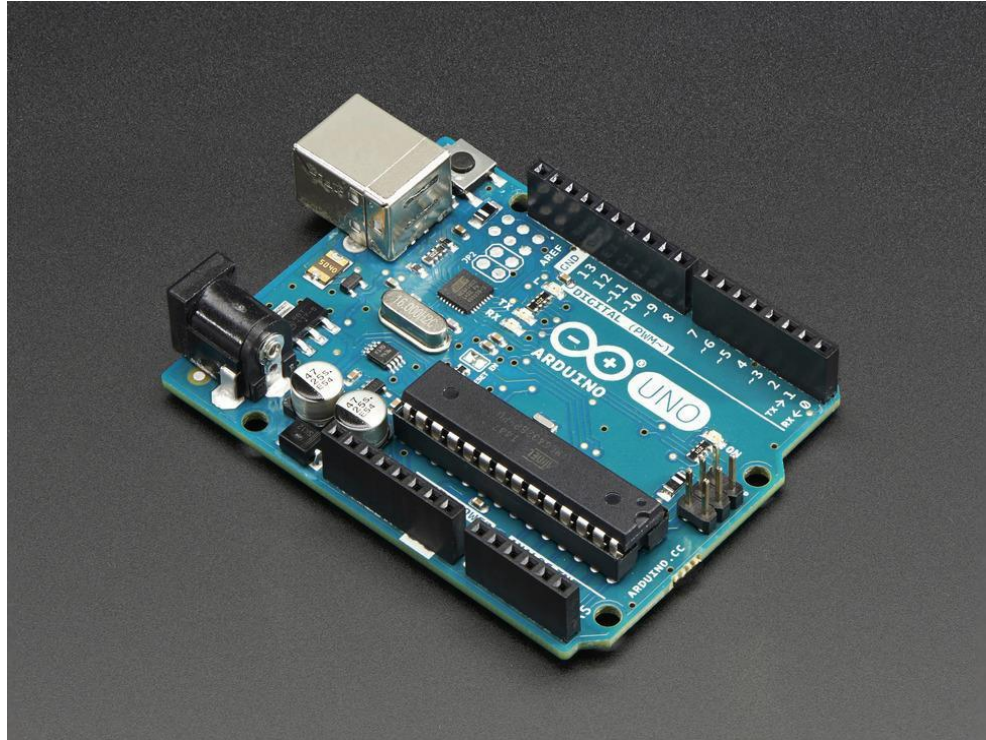
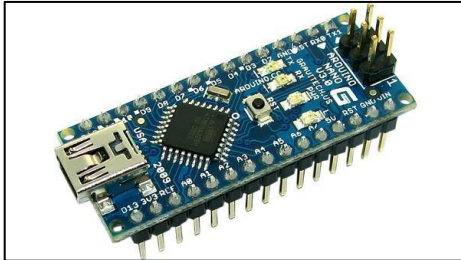
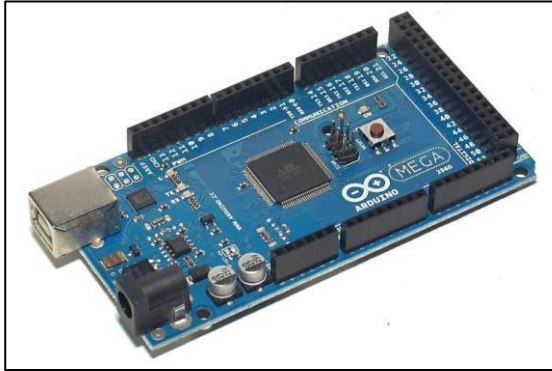
Globaali

- Voidaan käyttää kaikkialla ohjelmassa
- **Vältä mikäli mahdollista.**

Lokaali

- Voi käyttää vain niiden aaltosulkeiden (funktion) välissä jossa määritelty
- **Vaikeampi tehdä mokia**

Arduino



Arduino-ohjelman rakenne

```
int ledpin = 13;

void setup() ←
{
  pinMode(ledpin, OUTPUT);
}

void loop() { ←
  digitalWrite(ledpin, HIGH);
  delay(1000);
  digitalWrite(ledpin, LOW);
  delay(1000);
}
```

setup-funktio

- Ajetaan kerran kun Arduinoon kytketään virrat

loop-funktio

- Ajetaan uudelleen ja uudelleen kunnes Arduinoon katkaistaan virrat

digitalWrite

<https://www.arduino.cc/reference/en/language/functions/digital-io/digitalwrite/>

```
void setup()
{
  pinMode(13, OUTPUT);
}
void loop()
{
  digitalWrite(13, HIGH); //LED päällä
  delay(1000);           //Odota sekunti
  digitalWrite(13, LOW); //LED pois päältä
  delay(1000);           //Odota sekunti
}
```

digitalWrite(pinNro, arvo);

Asettaa pinin arvon **HIGH** tai **LOW**

- **LOW** eli 0 eli 0v
- **HIGH** eli 1 eli 5v

digitalRead

<https://www.arduino.cc/reference/en/language/functions/digital-io/digitalread/>

```
void setup()
{
  pinMode(3, INPUT);
}

void loop()
{
  int arvo = digitalRead(3);
}
```

digitalRead(pinNro);

Lukee pinin tilan

- **LOW** , pini kytketty maahan (**GND** eli **0v**)
- **HIGH**, pini kytketty jännitteeseen (riippuu käyttöjännitteestä **3.3v** tai **5v**)

analogRead

<https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/>

```
void setup()
{
  pinMode(A1, INPUT);
}

void loop()
{
  int arvo = analogRead(A1);
}
```

digitalRead(pinNro);

Lukee analogi pinin tilan (pinit A0, A1...)

- Analogi pinien takana 10-bit analogidigitaalimuuntaja (AD)
- Arvoja väliltä 0-5v, jotka saadaan lukuina 0-1023 (esim. 2.5v == 512)
- Kyseisiä pinejä voi käyttää myös digitaalikäyttöön

analogWrite

<https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/>

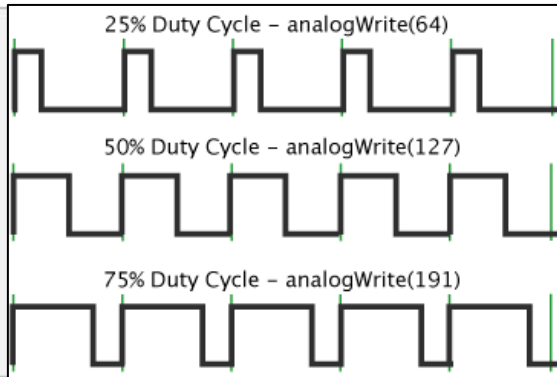
analogWrite(pinNro, arvo);

Arduino UNO:ssa ei ole DA-muunninta, joten analoginen ulostulos tehdään "keinotekoisesti"

-> PWM (pinit joissa " ~ " merkki)

- PWM, eli Arduino kytkee piniä nopeasti 0v ja 5v välillä (kanttiaalto)
- "Teho" määräytyy kauanko piniä pidetään HIGH- ja kauanko LOW-tilassa

```
void setup() {  
  pinMode(3, OUTPUT);  
}  
void loop() { //Ledi palaa "puolella teholla"  
  analogWrite(3, 127);  
}
```



Ylös-/alasetovastukset

<https://learn.sparkfun.com/tutorials/pull-up-resistors/all?print=1>

Mikäli piniä ei ole kytketty mihinkään, sanotaan että se kelluu

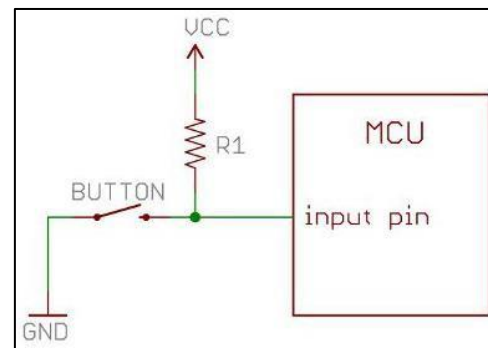
- digitalRead palauttaa satunnaisesti **HIGH** tai **LOW**

Ratkaisuna **alasetovastus**

- Kytketään pini vastuksella (n. 10- 220k Ω) **ov:iin (GND)**
- Pinin arvo pysyy nyt **LOW** kunnes se kytketään **+5v:iin**

Ylösvetovastus lähes sama asia:

- Kytketään pini vastuksella **+5v:iin**
- Pinin arvo pysyy nyt **HIGH** kunnes se kytketään **ov:iin (GND)**



Ylösvetovastuskytkentä

Sisäinen ylösvetovastus

<https://www.arduino.cc/en/Tutorial/InputPullupSerial>

```
void setup()
{
  pinMode(3, INPUT_PULLUP);
}

void loop()
{
  int nappula = digitalRead(3);
}
```

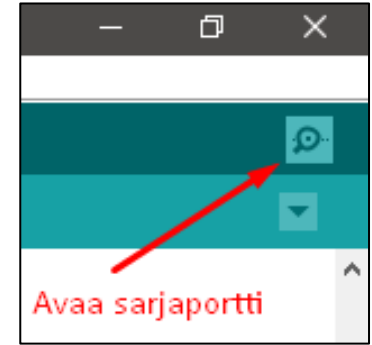
INPUT_PULLUP

- Arduinossa sisäänrakennettuna **ylösvetovastus** (Joistakin myös **alasetovastus**)
- Aktivoidaan käyttämällä **INPUT_PULLUP**
- Nappi voidaan kytkeä suoraan maan (**GND**) ja halutun pinin välille. Vastusta ei tarvita kytkennässä!

Sarjaportti

<https://www.arduino.cc/reference/en/language/functions/communication/serial/>

- Arduino keskustelelee USB:n kautta tietokoneen sarjaporttiin
- Helpoin tapa tietää mitä Arduinossa tapahtuu
 - Kun ohjelma ei toimi kuten pitäisi, kannattaa sarjaporttiin tulostaa eri muuttujien arvoja
 - Tekstin tulostaminen ennen ja jälkeen jonkin ohjelman osan auttaa selvittämään, jääkö ohjelma jumiiin johonkin kohtaan.
 - *Serial.print(muuttuja tai teksti) //Tulostaa pötköön*
 - *Serial.println(muuttuja tai teksti) //Tulostaa riveittäin*



Huom! Jos käytät sarjaporttia, pinejä **0** ja **1** ei voi enää käyttää muuhun! (Pätee Arduino UNO:oon)

Yleisiä ongelmia

- **a = b ja a == b ovat eri asioita!**
 - a = b //Sijoita b:n arvo a:han
 - a == b //Tarkista ovatko a ja b saman arvoiset
 - If (a = b) on siis useimmiten virhe!
- C-tyylisiä merkkijonoja ei voi vertailla ”==”-operaattorilla, mutta Arduinon ”String”-objekteja voi!

Linkkejä

- **Arduinon oppaat ja esimerkit**
 - <https://www.arduino.cc/en/Tutorial/HomePage?from=Main.Tutorials>
- **Arduinon kielen referenssi**
 - <https://www.arduino.cc/reference/en/>
- **Google**
 - <https://www.google.fi/>