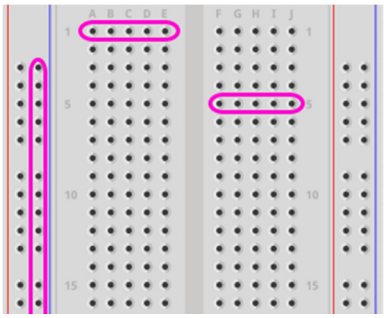
Blink med to lysdioder

Arduino-eksperiment	130110	*Stikord*	Konstanter, variabler, udtryk, for-løkke	
Version	2018-05-18 / HS	*Niveau*	Grundkursus, modul 2	*p.* 1/4

Det lærer du:

- Gør programmet mere overskueligt ved at bruge *konstanter*
- Gem et tal i en *variabel*
- Gentag en stribe kommandoer med en `for`-*løkke*

- Byg et lille kredsløb på et breadboard
- Brug en lysdiode sammen med en *formodstand*



1 – Tilslut en lysdiode på et breadboard

Breadboard

Små kredsløb opbygges let uden lodning på et *breadboard* (også kaldet et "fumlebræt"). Komponenterne stikkes blot i hullerne, som er elektrisk forbundet som vist på figuren:

- De yderste lange rækker i hver side er forbundet hele vejen.
- De nummererede rækker består af 5 forbundne huller.
  Der er ingen forbindelse hen over midten af breadboardet

De lange rækker bruges typisk til 5 V og 0 V. I denne øvelse, skal du bruge en række til 0 V.

Lysdioder og modstande

Da du brugte den "indbyggede" lysdiode på ben 13, opdagede du næppe, at den ikke er forbundet direkte mellem ben 13 og 0 V – men den er faktisk forbundet i serie med en modstand. Ellers vil der gå så høj en strøm, at lysdioden eller Arduinoen kan blive ødelagt.

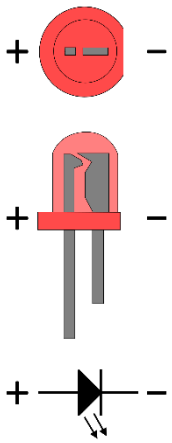
- En lysdiode skal altid forbindes i serie med en modstand.
  En værdi på 1 kΩ virker normalt fint. Farvekode: Brun, sort, rød.

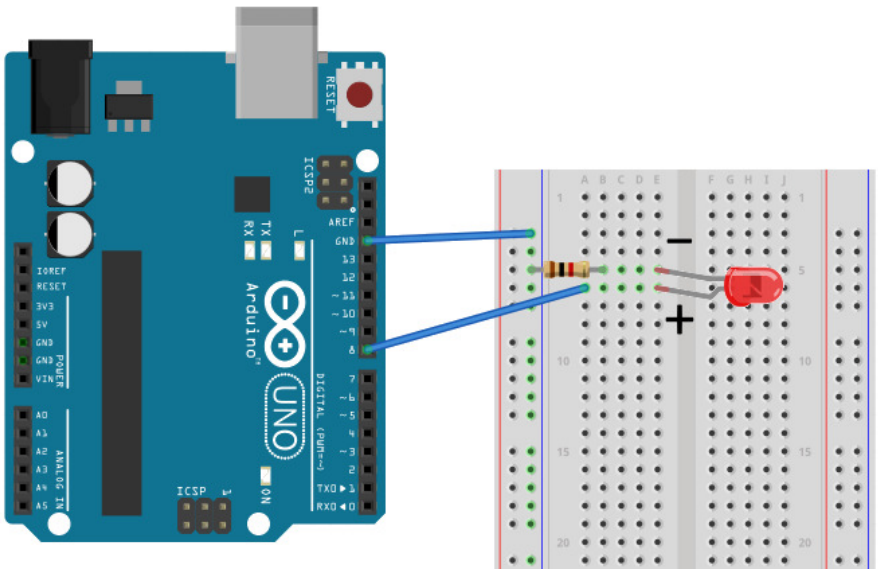
På figuren herunder ses en opstilling, hvor en lysdiode i serie med en modstand er tilsluttet ben 8 på Arduinoen. En af de lange rækker er tilsluttet 0 V: Benet GND på Arduinoen – det står for "Ground" eller "Stel".

Lysdioden virker kun, når den forbindes korrekt til plus og minus: Det længste ben skal til "plus" (benet på Arduinoen), det korteste til minus (0 V). Minussiden er også markeret med en flad kant.

- Lysdioden skal vende rigtigt for at lyse.

På næste side gennemgås programmet.





2 – Programmering med eksterne lysdioder

Du kan evt. åbne programmet **blink-LED**, som blev gemt i eksperiment 130101 (har du ikke gemt dette program, taster du bare ind som vist nedenfor).

For at finde frem til programmet, skal du først åbne mappen med navnet blink-LED, deri ligger en Arduino-fil med samme navn – vælg den.

Hver gang du vil ændre på et fungerende program, er det mest fornuftigt først *at gemme det under et nyt navn*, så du altid kan gå tilbage til den tidligere version: Vælg menuen *Fil / Gem som*. Brug navnet **Blink-2**.

Vi skal ikke længere bruge den indbyggede lysdiode men en ekstern, som er forbundet til pin 8. Ret programmet, så det kommer til at se således ud:

```
const byte pinRed = 8;

void setup() {
  pinMode(pinRed,OUTPUT);
}

void loop() {
  digitalWrite(pinRed,HIGH);
  delay(200);
  digitalWrite(pinRed,LOW);
  delay(800);
}
```

I det første program skrev vi ben-nummeret (13) ind direkte i programmet.

Her definerer vi i stedet først en *konstant* med navnet `pinRed`, som derefter bruges i resten af programmet.

Efterhånden som projekterne bliver større, kan det blive et mareridt at skulle ændre på talværdier, som står hist og pist rundt i hele programmet – det er langt sikrere at bruge konstanter, som kan vedligeholdes ét sted i programmet.

Værktøj: Konstanter

Eksempel:

```
const byte pinRed = 8;
```

Hver gang du skriver `pinRed`, indsættes værdien 8

Forklaring:

`const` ***A B*** `=` ***C***`;`

A Konstantens **type**, her bruges typen `byte`. En `byte` kan rumme tal fra 0 til 255 (inkl.)

B Konstantens **navn**. Her `pinRed`

C Konstantens **værdi**. Her 8

Tal gemmes i Arduinoens hukommelse i forskellige typer af hukommelsespladser. Disse typer bruges ved definition af både konstanter og variabler (næste afsnit)

Typen `byte` fylder (– SURPRICE! –) 1 byte. Den rummer heltallige værdier fra 0 til max. 255.

Typen `int` (engelsk "integer", dvs. heltal) fylder 2 bytes og rummer hele tal mellem -32768 og 32767.

Typen `word` fylder også 2 bytes, men talområdet går fra 0 til 65535.

Der findes flere typer end disse.

Udfordring 1

Tilslut endnu en lysdiode (f.eks. en grøn) med formodstand. Brug ben 7 på Arduinoen.

Tilføj en konstant mere til programmet – den kunne hedde `pinGreen`.

Ret `setup()`, så ben 7 også gøres til en udgang – brug konstanten

Skriv `loop()` om, så den røde lysdiode er tændt i et kvart sekund, mens den grønne er slukket – derefter skal den grønne være tændt i et kvart sekund, mens den røde er slukket. Derefter skal begge lysdioder være slukket i et sekund. (Og denne sekvens skal gentages i det uendelige.)

Der blinkes med temmelig mange lysdioder, når folk skal lære at programmere en microcontroller. Det er altid ment som et *eksempel* på, at man kan styre noget i den fysiske verden med sit program – det er kun et spørgsmål om lidt ekstra hardware, hvis man i stedet skulle tænde en 1500 W projektør. (Men den ville nok ikke have godt at af stå og blinke så hurtigt ☺.)

3 – Gentagelser (løkker)

Gentagelses-løkker bruges utrolig tit i programmer. Funktionen loop() sørger allerede for et gentage en række kommandoer i det uendelig. Men hvad nu, hvis en mindre række kommandoer skal gentages f.eks. 10 gange, hvorefter der skal ske noget helt andet?

Du kunne selvfølgelig kopiere de linjer, det drejer sig om, ti gange! Men det bliver hurtigt noget rod – og hvis antallet ikke er fast, bliver det nærmest umuligt.

Inden vi gennemgår, hvordan man laver løkker, må vi først se på et meget vigtigt begreb, nemlig *en variabel*.

Værktøj: Variabler

Eksempel:

```
byte antal = 10;
```

Sæt plads af i hukommelsen og giv den et navn (og evt. en startværdi).

Forklaring (Vi ser på 2 versioner):

1: ***A*** ***B***;
2: ***A*** ***B*** = ***C***;

A Variablens **type**, her bruges typen byte.

B Variablens **navn**, her kalder vi den for `antal`.

C Variablens **startværdi** (her 10).
Version 1 ovenfor anvender ikke en startværdi – i så fald er indholdet af hukommelsespladsen *tilfældig*, indtil variablen får en værdi senere i programmet.

Variabler

Se værktøjs-boksen til højre.

Her opretter vi en variabel med navnet `antal`, som får værdien 10 – det ligner fuldstændigt det, vi gjorde med konstanter lige før.

Forskellen er, at vi senere i programmet kan ændre en variabels værdi:

```
antal = 7;
```

Ofte vil variabler få en værdi, som beregnes ud fra andre variablers værdi. I andre tilfælde vil man måske bare gøre værdien 1 større end den var – det har man brug for så tit, at der findes en særlig skrivemåde for det. De følgende to linjer forøger begge værdien af `antal` med 1:

```
antal = antal + 1;
antal++;
```

Lighedstegnet "=" skal ikke læses "er lig med", men som "sættes lig med"; først beregnes højresiden – så tildeles denne værdi til variablen.

Synlighed af variabler

Skriver du variabel-definitionen øverst i programmet, udenfor nogen funktion, har du skabt en *global* variabel. Den kan bruges overalt og har samme værdi overalt.

Skriver du variabel-definitionen indenfor krølleparenteser {} – f.eks. inde i funktionen `setup()` – så bliver variablen "usynlig" udenfor denne programblok. Det kaldes en *lokal* variabel. Det kan have visse fordele i større programmer.

For-løkke

Nu er vi klar til at lave en `for`-løkke – se værktøjs-boksen til højre.

I eksemplet ser vi en lidt speciel type lokal variabel. Den defineres med "`int i=0`" og er synlig hele vejen frem til den afsluttende }.

Det kunne udnyttes til at lade `delay`-pausen variere. Ændres den sidste `delay(150)` til:

```
delay(150*i);
```

- så bliver pausen første gang 0 ms, anden gang 150 ms, tredje gang 300 ms, osv.

Værktøj: Gentagelses-løkke med `for`

Eksempel:

```
for (int i=0; i<5; i++) {
  digitalWrite(pinRed,HIGH);
  delay(150);
  digitalWrite(pinRed,LOW);
  delay(150);
}
```

Gentag 5 gange: Blink med rød lysdiode

Forklaring:

```
for (A; B; C) { D }
```

A Udføres **før** løkken. Her får en heltals-variabel navnet `i` og startværdien 0.

B Betingelse for at udføre ***D***. Her kontrolleres, at `i` er *mindre end* 5. (Ellers fortsættes til næste kommando efter `for`-løkken)

C Udføres **efter** hver gentagelse af ***D***. Her lægges 1 til `i`.

D Det, som skal gentages. I eksemplet blinkes én gang.

(fortsætter på næste side)

Udfordring 2

Brug en `for`-løkke til at lade den røde og den grønne lysdiode blinke hurtigt på skift 10 gange (200 ms pr. gang).

Det skal efterfølges af en pause på 1 sekund, hvor de begge er slukket.

(Og denne sekvens skal gentages i det uendelige.)

Udfordring 3

Lav `for`-løkken om, så den røde og den grønne lysdiode blinker med varierende længde: I starten skal rød blinke kort og grøn længe. Men efterhånden skal rød være tændt længere og længere, mens grøn omvendt skal være tændt i kortere og kortere tid.

Det skal efterfølges af en pause på 1 sekund, hvor de begge er slukket.

(Og denne sekvens skal gentages i det uendelige.)

4 – Ekstra opgave

Vi har hidtil brugt konstanter til at adressere benene på Arduinoen (`pinRed, pinGreen`). Det kan lige så godt gøres med en variabel. Ben-nummeret kan også findes ved beregning:

```
byte pin = 2;

pinMode(pin,OUTPUT);
pinMode(pin+4,OUTPUT);
```

Ovenstående lille programstump sætter ben 2 og ben 6 til at være udgange.

Udfordring 4 – Løbelys

Tilslut 8 lysdioder gennem formodstande til benene 2 til 9 på Arduinoen.

Brug en `for`-løkke i `setup()` til at sætte alle disse ben til at være udgange.

Brug en anden `for`-løkke til at blinke med én lysdiode ad gangen i rækkefølge. Derved ser det ud, som om en lysende prik ”løber” hen ad rækken af lysdioder.

(Og denne sekvens skal gentages i det uendelige.)