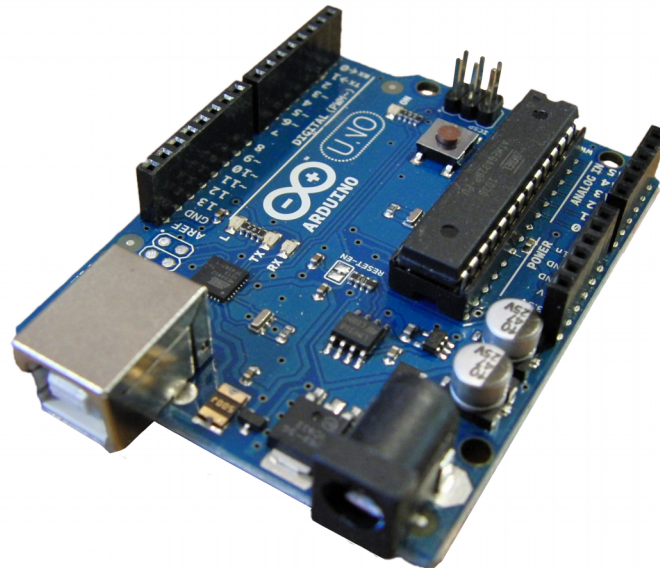


Functies 2

© 2015 Richel Bilderbeek

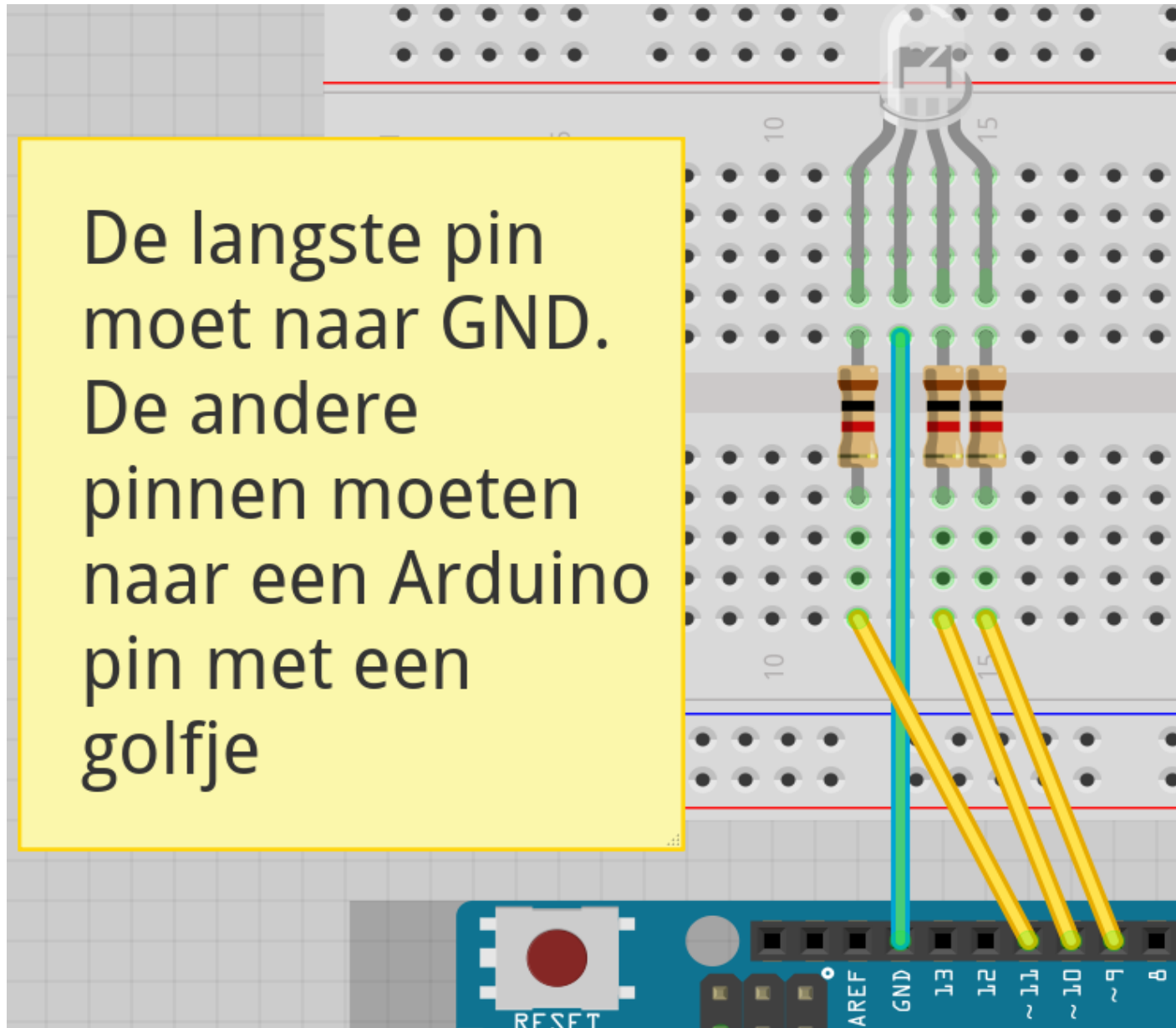


Probleem

- Je programmeercode is moeilijk te lezen
- Je programmeercode herhaalt zich

Voorbeeld

De langste pin moet naar GND.
De andere pinnen moeten naar een Arduino pin met een golfje



Setup

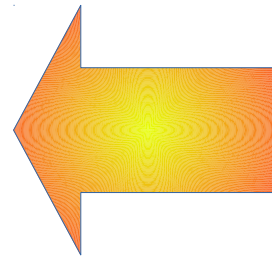
```
void setup() {  
    pinMode(9, OUTPUT) ;  
    pinMode(10, OUTPUT) ;  
    pinMode(11, OUTPUT) ;  
}
```

Loop, moeilijk leesbaar

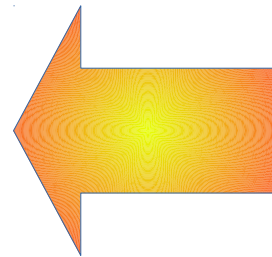
```
void loop() {  
    digitalWrite(9,HIGH);  
    digitalWrite(10,LOW);  
    digitalWrite(11,LOW);  
    delay(1000);  
    digitalWrite(9,HIGH);  
    digitalWrite(10,HIGH);  
    digitalWrite(11,LOW);  
    delay(1000);  
    //Nog meer code hier  
}
```

Loop, moeilijk leesbaar

```
void loop() {  
    digitalWrite(9,HIGH);  
    digitalWrite(10,LOW);  
    digitalWrite(11,LOW);  
    delay(1000);  
    digitalWrite(9,HIGH);  
    digitalWrite(10,HIGH);  
    digitalWrite(11,LOW);  
    delay(1000);  
    //Nog meer code hier  
}
```



Maak de
RGB LED
rood



Maak de
RGB LED
geel

Loop gemakkelijk leesbaar

```
void MaakLedRood() {  
    digitalWrite(9,HIGH);  
    digitalWrite(10,LOW);  
    digitalWrite(11,LOW);  
}
```



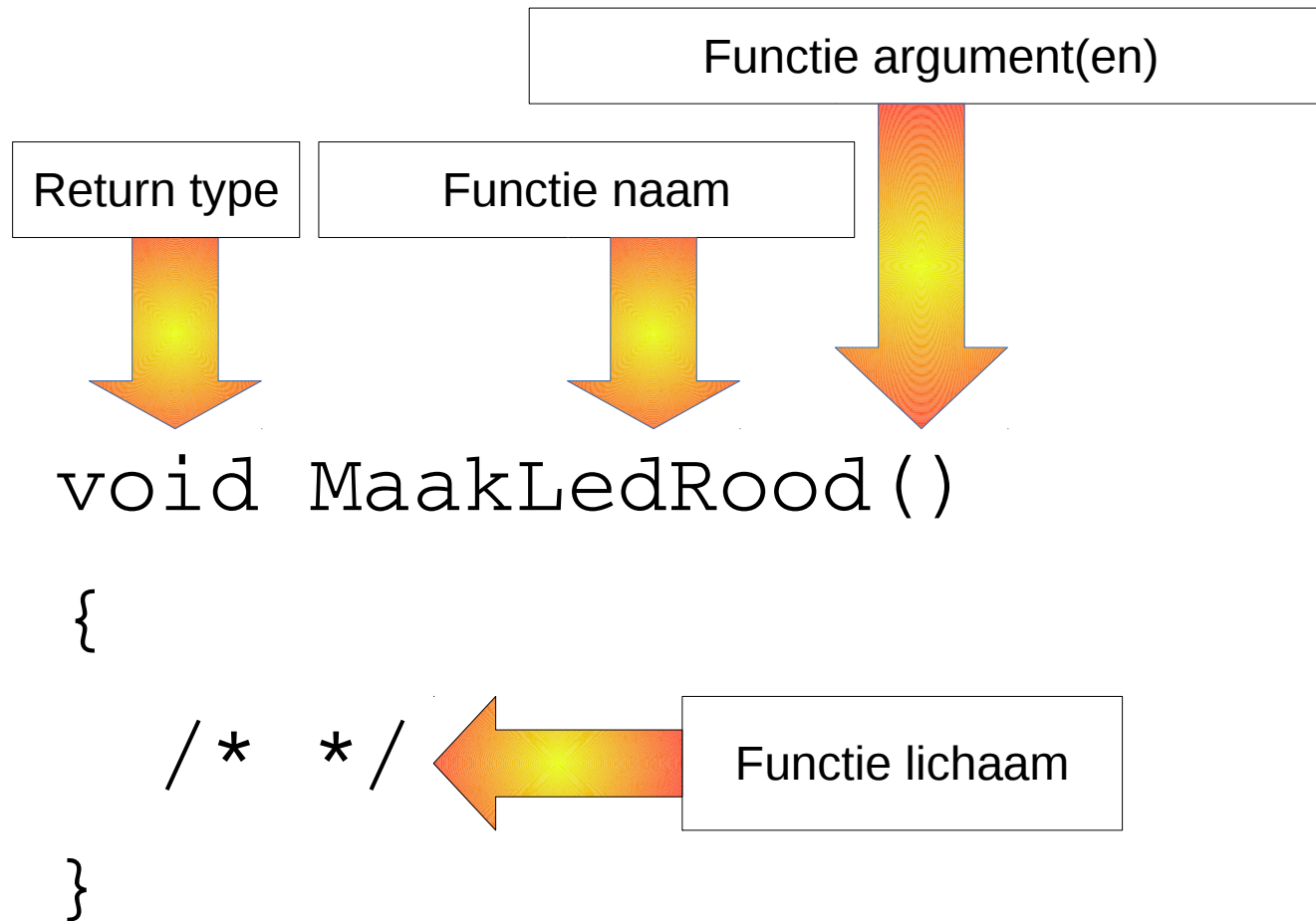
Definitie van
de 'MaakLedRood'
functie

```
void loop() {  
    MaakLedRood();  
    delay(1000);  
    MaakLedGeel();  
    delay(1000);  
    //Nog meer code hier  
}
```



Aanroep van de
'MaakLedRood' functie

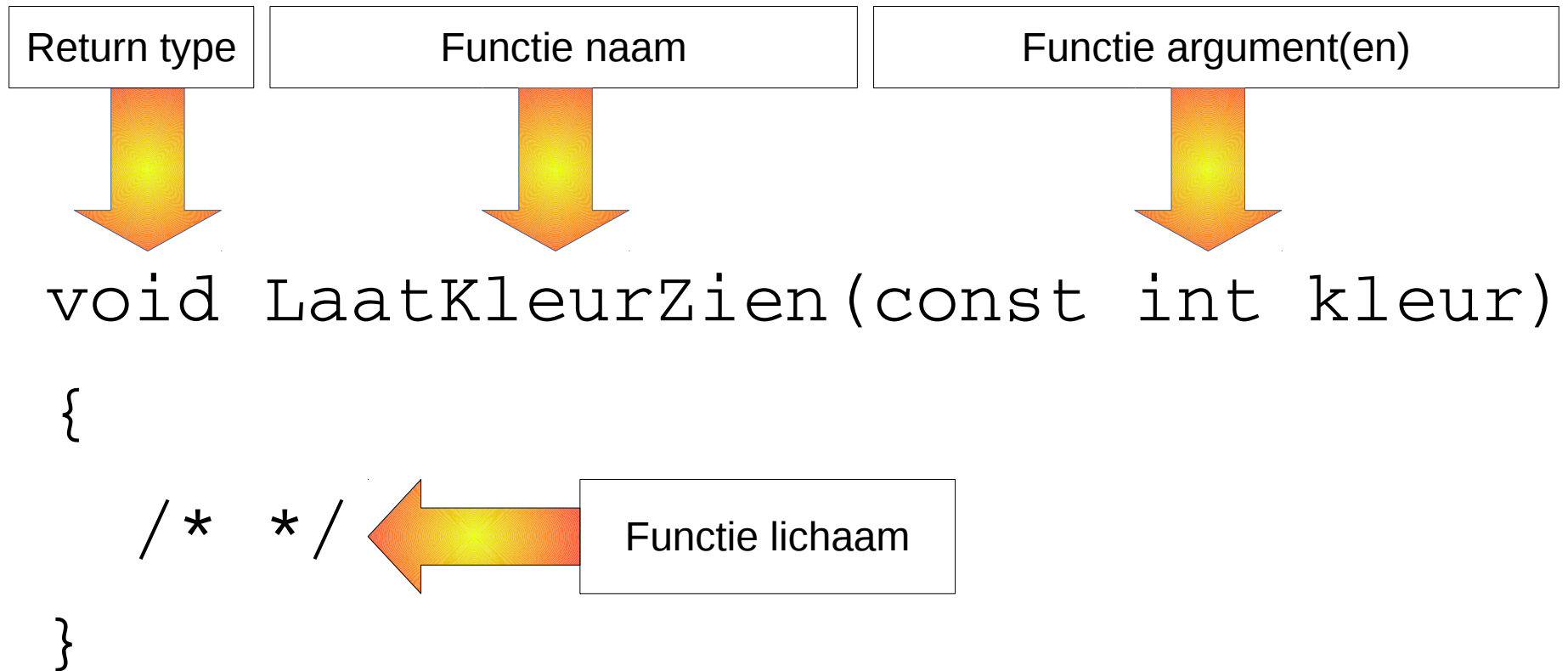
Functie zonder argumenten



Probleem

- Nu is er voor elke kleur een aparte functie
- Kan dit niet in een functie?

Functie met argumenten



Functie met argumenten

```
void LaatKleurZien(const int kleur)
{
    switch (kleur) {
        case 0: MaakLedRood(); break;
        case 1: MaakLedOranje(); break;
    }
}
```

```
void loop() { LaatKleurZien(0); }
```

Functie met argumenten

```
void LaatKleurZien(const String& kleur)
{
    if (kleur == "Rood") {
        MaakLedRood();
    } else if (kleur == "Oranje") {
        MaakLedOranje();
    }
}
```

```
void loop() { LaatKleurZien("Rood"); }
```

Functie met argumenten

```
enum Kleur { rood, oranje };
```

```
void LaatKleurZien(const Kleur kleur)
{
    switch (kleur) {
        case rood: MaakLedRood(); break;
        case oranje: MaakLedOranje(); break;
    }
}
```

```
void loop() { LaatKleurZien(rood); }
```

Meer over functies

- Een functie kan meerdere argumenten hebben, deze worden gescheiden met komma's

```
void KnipperInMorse(  
    const int led_pin,  
    const char letter  
) { /* */ }
```

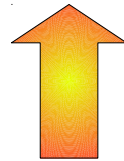
```
void loop() { KnipperInMorse(13, 'R'); }
```

Meer over argumenten

- Je kunt op meer manieren argumenten aan een functie
- Vuistregel: gebruik '&' als data type met een hoofdletter begint (oftewel: een klasse is)

```
void f(const int i) { /* */ }
```


```
void g(const String& i) { /* */ }
```



Tips

- De naam van een functie
 - Begint met een hoofdletter (of kleine letter)
 - Begint met een werkwoord
 - Is in CamelCase
- Een functie moet een ding goed kunnen
 - Wel: MaakLedRood
 - Niet: MaakLedRoodEnWacht
- Binnen functies kunnen je andere functies aanroepen, bijvoorbeeld 'MaakLedRood' binnen loop

Moeilijkere functies

- Functies kunnen ook een waarde teruggeven, zoals analogRead, digitalRead
 -  Functies3