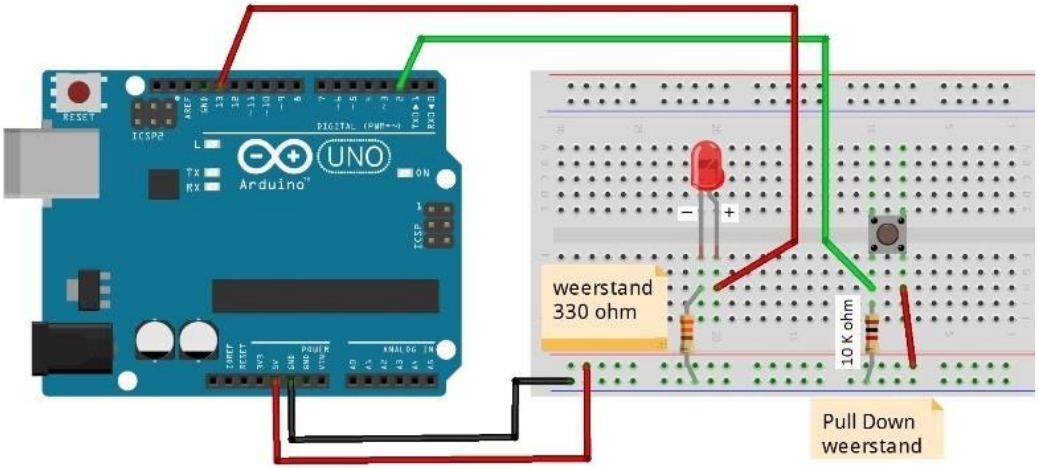
We gaan zorgen dat de arduino het verschil kent tussen een hoog (HIGH) en een laag (LOW) signaal.
We gaan eerst zorgen dat als je de schakelaar indrukt een led gaat branden daarna het omgekeerde. Als je de schakelaar indrukt gaat de led uit.

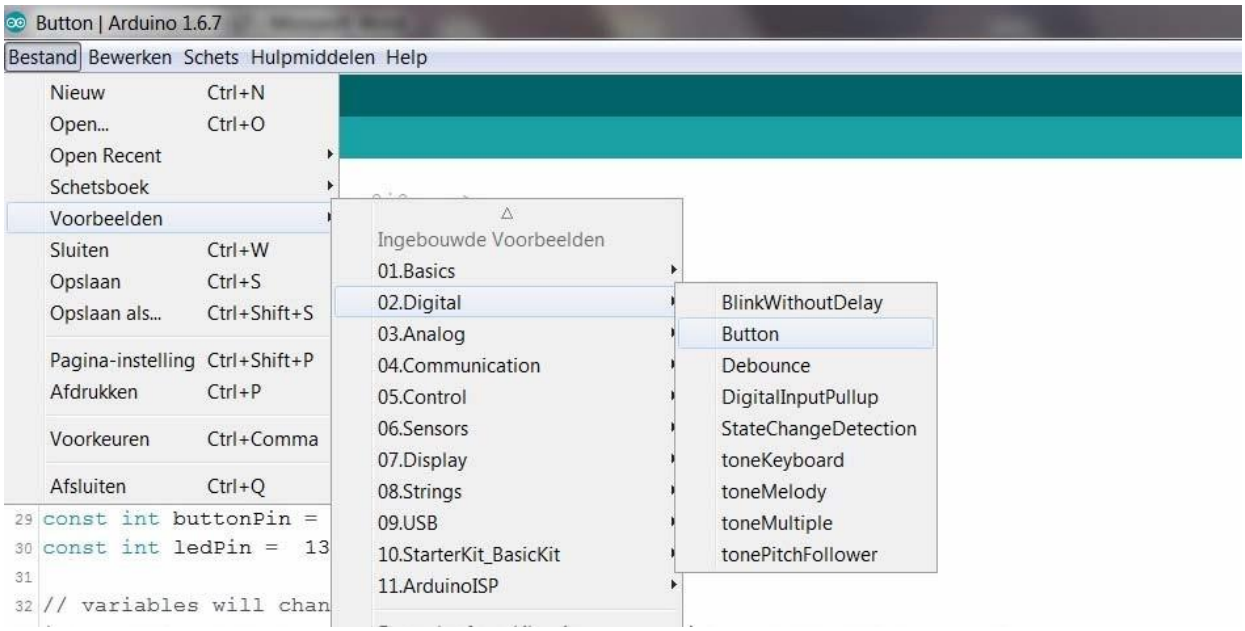
Project 3: LED met schakelaar



Verbind één kant van de schakelaar met een weerstand van 10.000 Ohm aan de – lijn.
10.000 Ohm wordt korter geschreven als 10k (want k is kilo).
De kleurencode is Bruin-Zwart-Oranje. (Want Bruin is 1, Zwart is 0 en Oranje is 3)

Resultaat:
Als de schakelaar niet is ingedrukt is die verbindingsdraad via de weerstand verbonden met de –
De weerstand 'trekt' die draad naar beneden. Dit heet een Pull-Down weerstand.
Als je de schakelaar indrukt wordt verbindingsdraad (dus ook pin2) verbonden met de +.
Dus niet ingedrukt: pin2 is *Laag*
Wel ingedrukt: pin2 is *Hoog*

We gaan nu zorgen dat Arduino het verschil ziet.
Kies voor Bestand, Voorbeelden, 02.Digital, Button



```
const int buttonPin = 2;     // the number of the pushbutton pin const
int ledPin =  13;      // the number of the LED pin

int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}

void loop() {
  buttonState = digitalRead(buttonPin);

  if (buttonState == HIGH) {
digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
}
```

Sommige regels kwamen al eerder voor.

- De schakelaar is vastgemaakt aan pin 2
- Er wordt weer gebruik gemaakt van de ingebouwde Led op pin 13. const int buttonPin = 2; // the number of the pushbutton pin const int ledPin = 13; // the number of the LED pin
- Er is een variabele en een naam ingevoerd voor de Toestand van de schakelaar. Ingedrukt is 1, niet ingedrukt is 0. Omdat dit in het programma kan veranderen kan hier niet const voor staan. int buttonState = 0; // variable for reading the pushbutton status - In de setup wordt opnieuw ledPin als Output gedefinieerd. Nieuw is dat de buttonPin als Input wordt gedefinieerd.

```
  pinMode(ledPin, OUTPUT);
pinMode(buttonPin, INPUT);
```

- In de loop wordt continue de toestand van de schakelaar gelezen met de opdracht digitalRead.

```
  buttonState = digitalRead(buttonPin);
```

En nu laten we de Arduino een beslissing nemen.
Als buttonState = LOW, (0 kan ook) dan is de schakelaar niet ingedrukt en moet de Led uitgezet.
Als buttonState = HIGH, (1 kan ook) dan is de schakelaar wel ingedrukt en moet de Led aangezet.
Die beslissing staat in een if(voorwaarde) { doe dan dit } else {doe iets anders}
Let op het verschil tussen de volgende 2 regels!
buttonState = HIGH betekent dat buttonState HIGH wórdt.
buttonState == HIGH betekent de vraag of buttonState HIGH ís.

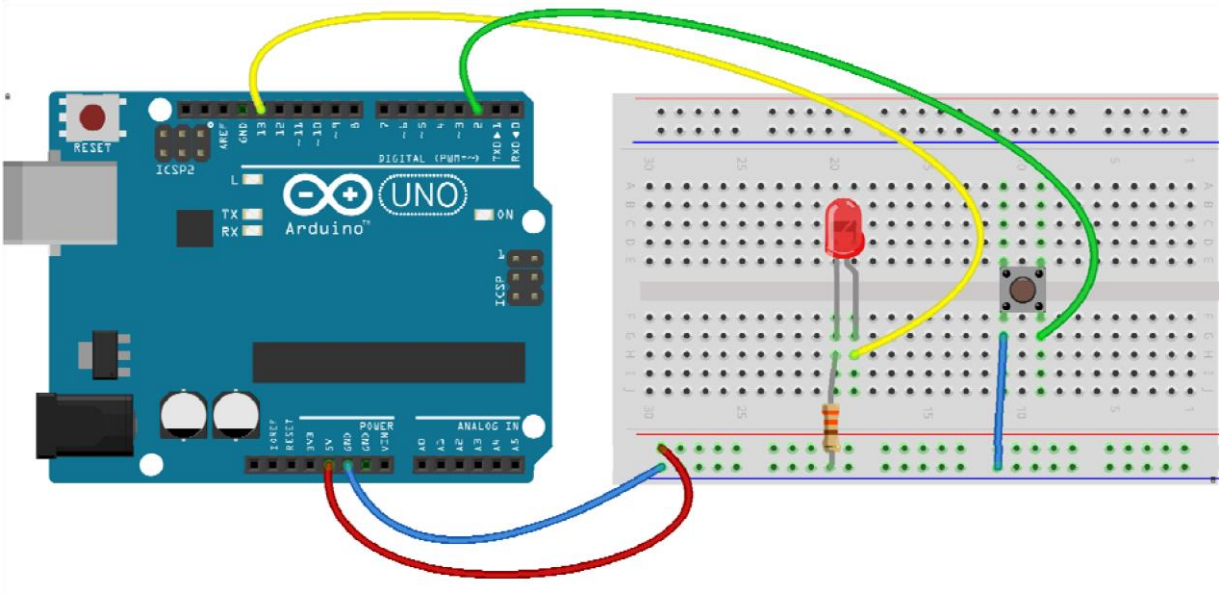
```
  if (buttonState == HIGH) {
digitalWrite(ledPin, HIGH);    // turn LED on:
  }    else {                digitalWrite(ledPin, LOW);
// turn LED off:   }
```

Na het uploaden van programma *Button* zal de Led aangaan als op de schakelaar wordt gedrukt.

Pull_Down of Pull_Up

Het gebruik van zo'n *Pull-Down* weerstand is onhandig. Het breadboard wordt er onoverzichtelijk door.
Het kan ook anders.

In plaats van een Pull-Down weerstand kunnen we net zo goed een Pull-Up weerstand gebruiken. Dan '*trekken'* we de schakelaar naar de + en bij indrukken wordt het een – Dat lijkt niets uit te maken maar… die Pull_Up weerstand zit al in de Arduino.



Verander hiervoor in het programma de regel
pinMode(buttonPin, INPUT);

in
pinMode(buttonPin, INPUT_PULLUP);

Dit "activeert" de interne Pull_Up weerstand. De schakeling wordt eenvoudiger.
Maar …..
De Led is nu AAN als je de schakelaar niet indrukt en gaat juist UIT als je 'm wel indrukt!
Dat komt omdat het indrukken van de schakelaar nu de buttonPin naar de – trekt in plaats van naar de +. Dat is makkelijk op te lossen in je programma. Verander HIGH in LOW.

```
 if (buttonState == LOW) {
digitalWrite(ledPin, HIGH);
 } else {
        digitalWrite(ledPin,
LOW);   }
```