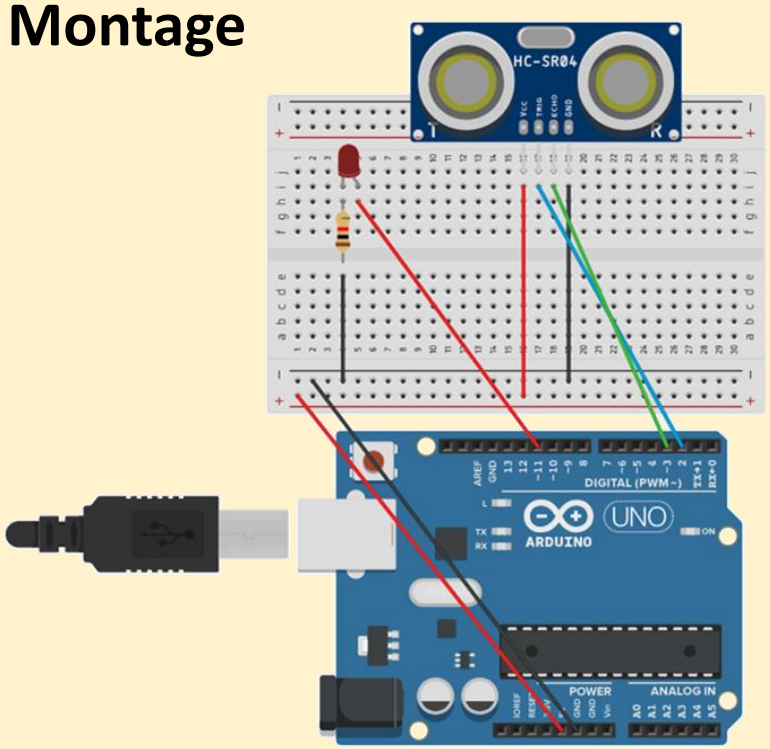
Étape 6 : Lier une LED au capteur

Objectifs:

- ❑ Conditionner l'allumage de la LED selon les valeurs recueillies par le capteur
- ❑ Changer la fréquence d'allumage de la LED

Conditionner l'allumage de la LED selon les valeurs recueillies par le capteur

Montage



Par rapport au montage précédent, on rajoute une LED connectée sur le port 11 et à la masse. On n'oublie pas la résistance pour ne pas endommager la LED.

? Exercice pratique: compléter le code ci-dessous avec les informations manquantes pour allumer la LED si la distance est inférieure à 20 cm et l'éteindre sinon.

```
1.  #define port_trigger 2   //on déclare le port de l'émetteur
2.  #define port_echo 3     //et du capteur
3.  const int LED = 11;      //puis de la LED
4.
5.  /* Vitesse du son en cm/us */
6.  float vitesse_son = 343.0 / 10000;
7.
8.  void setup() {
9.    pinMode(LED, OUTPUT);
10.   pinMode(port_trigger, OUTPUT);
11.   digitalWrite(port_trigger,LOW);
12.   pinMode(port_echo,INPUT);
13.   Serial.begin(9600);
14.  }
15.
16. void loop() {
17.   digitalWrite(port_trigger,HIGH);  //on allume l'émetteur
18.   delayMicroseconds(10);            //pendant 10 microsecondes
19.   digitalWrite(port_trigger,LOW);   //puis on l'éteint
20.   long measure = pulseIn(port_echo,HIGH); //on récupère le temps
21.   float distance_cm = measure / 2.0 * vitesse_son;
22.   Serial.println(distance_cm);
23.
24.
25.   if (distance_cm < ...) {
26.     digitalWrite(LED, HIGH);
27.   }
28.   else {
29.     digitalWrite(LED,...);
30.  }
31.   delay(500);                  //on laisse une demi-seconde pour
32.  }                               éviter de calculer trop souvent
```

Code identique à l'étape 5

Si la distance est inférieure à 20 cm, on allume la LED

Simuler un capteur de voiture

Dans un **radar de recul pour voiture**, l'alarme sonne **de plus en plus vite** pour alerter le conducteur de sa **proximité** avec l'obstacle. On souhaite donc faire clignoter la LED de plus en plus rapidement lorsque la distance diminue.

? Exercice pratique : Réfléchir à une solution pour adapter ce fonctionnement à notre circuit, en s'inspirant de ce qui précède

……
……
……
……
……
……
……
……
……
……
……

En pratique, de nombreuses solutions sont envisageables. On pourrait très bien adapter ce qui a été fait en début d'étape. On a utilisé un **seuil** (une valeur limite au-delà de laquelle le comportement du code change). On pourrait donc utiliser plusieurs seuils. Cependant, à des fins de concision dans notre code, on choisira une autre solution : la **fréquence** d'allumage de la LED sera **inversement proportionnelle** à la distance à l'obstacle.

? Question pratique : Que signifie « fréquence » ici ?

Si la distance augmente, comment varie la fréquence d'allumage de la LED ?

Ici, on ne modifiera que le void loop.
Pour influer sur la fréquence, on changera le *delay.*

```
1.  #define port_trigger 2   //on déclare le port de l'émetteur
2.  #define port_echo 3     //et du capteur
3.  const int LED = 11;      //puis de la LED
4.
5.  /* Vitesse du son en cm/us */
6.  float vitesse_son = 343.0 / 10000;
7.
8.  void setup() {
9.    pinMode(LED, OUTPUT);          //la LED est une sortie
10.   pinMode(port_trigger, OUTPUT); //l'émetteur est une sortie
11.   digitalWrite(port_trigger,LOW);   qui n'émet rien au début
12.   pinMode(port_echo,INPUT);       //le capteur est une entrée
13.  Serial.begin(9600);
14. }
15.
16. void loop() {
17.   digitalWrite(port_trigger,HIGH);  //on allume l'émetteur
18.   delayMicroseconds(10);            //pendant 10 microsecondes
19.   digitalWrite(port_trigger,LOW);   //puis on l'éteint
20.   long measure = pulseIn(port_echo,HIGH); //on récupère le temps
21.  float distance_cm = measure / 2.0 * vitesse_son;
22.  Serial.println(distance_cm);
23.  if (distance_cm > 50) {          //si la distance > 50cm
24.    digitalWrite(LED, LOW);       //on éteint la LED
25.    delay(500);                   //on patiente 500ms
26.  }
27.  else {                          //sinon
28.    digitalWrite(LED,HIGH);       //on allume la LED
29.    delay(distance_cm*20);        //on patiente
30.    digitalWrite(LED,LOW);        //on éteint la LED
31.    delay(distance_cm*20);        //on patiente
32.  }
33. }
```

Ici, le délai doit être proportionnel à la distance mais le 20 est arbitraire. Il permet de bien voir le changement de fréquence pour des distances de quelques centimètres.