L'utilisation d'un bouton poussoir, et plus généralement d'un contacteur, va vous amener à gérer non plus les sorties numériques, mais les **entrées** de la carte Arduino. Nous allons donc voir comment elles fonctionnent, comment on les programme, comment on en récupère les informations (il s'agit pour le moment de récupérer les informations des entrées numériques).

Les entrées numériques

Nous avons utilisé jusqu'à présent les sorties numériques : soit l'Arduino est en position haute (HIGH) et il envoie du +5V, soit il est en position basse, et il est à 0V, donc au ground.
Et bien les entrées numériques fonctionnent sur le même principe :

- Soit elles reçoivent du +5V ou du 0V.

Pour les **sorties** nous utilisions la commande : **digitalWrite(pin, état)**, qui est donc une commande d'écriture : write=écrire
Pour les **entrées**, nous utiliserons la commande : **digitalRead(pin)**, qui vous l'aurez peut-être deviné est une commande de lecture : read=lire.
Mais avant toute chose il faut comprendre une notion importante de l'Arduino : un pin est **soit en entrée, soit en sortie, mais pas les deux**. Il est donc important de bien définir si le pin va se comporter en entrée ou en sortie !
C'est ce que nous pouvons faire grâce à la commande que nous avons utilisée :

pinMode(pin, mode);

- **Mode OUTPUT :**

pinMode(pin,OUTPUT);
Pour indiquer à la carte que le pin doit être en mode **écriture**, c'est-à-dire qu'il peut envoyer ou non du courant. C'est donc une **sortie**.

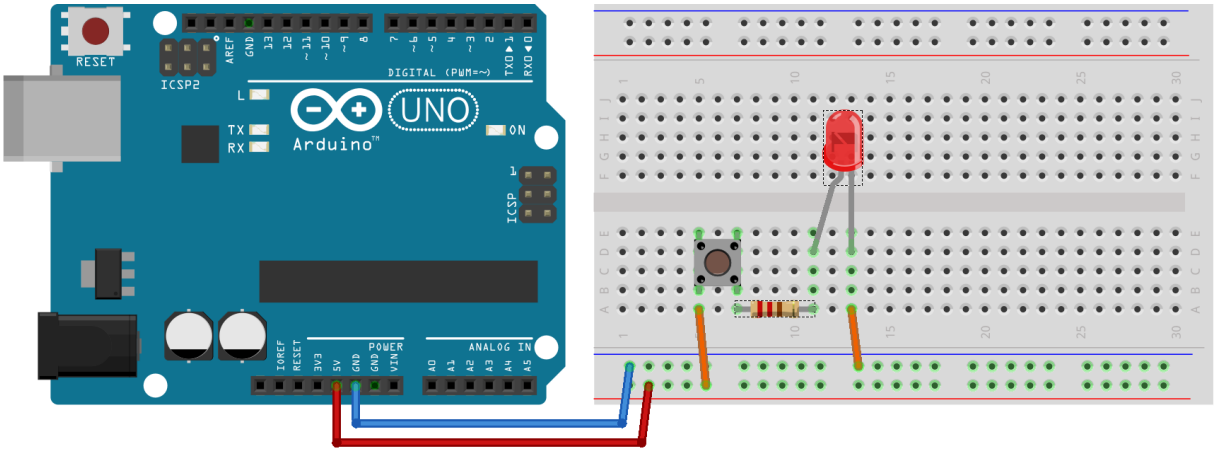
- **Mode INPUT :**

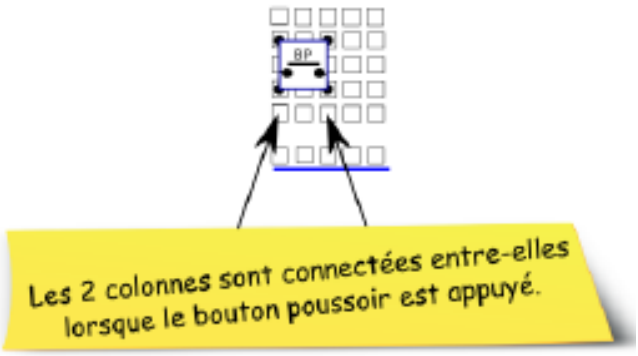
pinMode(pin,INPUT);
Pour indiquer que le pin est en mode **lecture**. **Il ne va donc pas piloter du courant, mais être à l'écoute du courant qui va lui arriver.**

Le bouton poussoir

Le principe de ce bouton est que lorsque l'on appuie, le courant passe, et lorsque l'on relâche et bien... le courant ne passe plus !
Contrairement à un interrupteur, il ne garde pas la position (il faut garder le doigt dessus pour qu'il fasse contact).

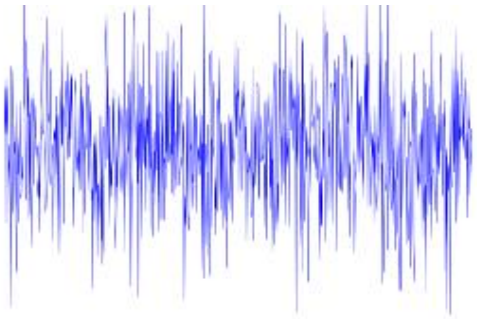
Tester le montage suivant :





Attention, même s'il a 4 pattes, le bouton poussoir est un dipôle. En fait les pattes sont reliées deux par deux. Le montage au-dessus permet d'ailleurs de bien repérer les pattes reliées ou non entre elles avant d'aller plus loin. **Le montage fonctionne si la LED s'allume lorsqu'on appuie sur le poussoir. Si elle s'allume en permanence, il faut tourner le bouton d'un quart de tour.**

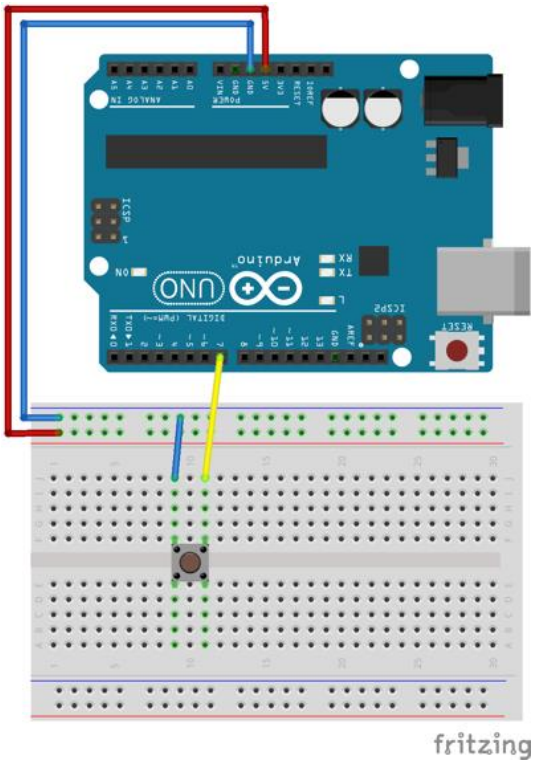
Une broche numérique utilisée en entrée va permettre de « lire » l'état de la broche, ainsi lorsque l'on connecte un bouton poussoir entre la broche en entrée et le 0 V (ground), si le bouton est appuyé, le contact s'établi et la broche sera connectée au 0v (ground), tandis que si le bouton est relâché, le contact n'est pas établi et la broche ne sera pas connectée au 0V.



Oui, mais dans ce cas : dans quel état sera la broche numérique en entrée ?
En effet, lorsque le bouton poussoir est relâchée, la broche numérique en entrée restera non connectée et lorsque l'arduino lira son état, quel sera-t-il ?
En fait dans cet état la broche va se comporter comme une petite antenne et va osciller en permanence au gré des interférences électromagnétiques.
On obtiendrait quelque chose comme l'image ci-contre.

Pour contourner ce problème, on « attache » au +5V (ou au 0V) la broche numérique en entrée à l'aide d'une résistance assez élevée (10 kΩ). **On appelle cela le « rappel au plus »**

Cependant, la carte Arduino propose par défaut un mode qui permet d'activer une résistance de 20 KΩ qui est dans la carte pour en faire un rappel au plus (pull-up).



Il faut indiquer à l'Arduino d'activer cette résistance, on utilise la commande suivante :

```
pinMode(pin,INPUT_PULLUP);
```

Ce qui pour le montage de l'image ci-contre deviendrait :

```
pinMode(7,INPUT_PULLUP);
```

Tester le programme suivant
Expliquer le rôle de la variable etatBouton ? Quelle valeur peut-elle prendre ?

```
int pinBouton;
void setup()
{
   Serial.begin(9600);
   pinBouton=7;
   pinMode(pinBouton,INPUT_PULLUP);
}
void loop()
{
   boolean etatBouton=digitalRead(pinBouton);
   Serial.println(etatBouton);
}
```

Exercice 1 : Allumage d'une LED

Ajoutez une LED (n'oubliez-pas la résistance) au montage précédent.
La LED doit être reliée au connecteur 8.
Votre programme reprend le précédent et doit permettre en plus l'allumage de la LED lorsque l'on appuie sur le bouton poussoir et son extinction lorsque l'on relâche ce même bouton.

```
if (test)
{
   //code à exécuter
}
else if //autre test
{
   //autre code
}
else
{
   //code exécuté si aucun test n'est vérifié
}
```

Aide : La variable booléenne etatBouton permet de récupérer la lecture de l'état du bouton poussoir.
La création d'une condition permettra d'allumer ou d'éteindre la LED en fonction de la valeur de la variable etatBouton.

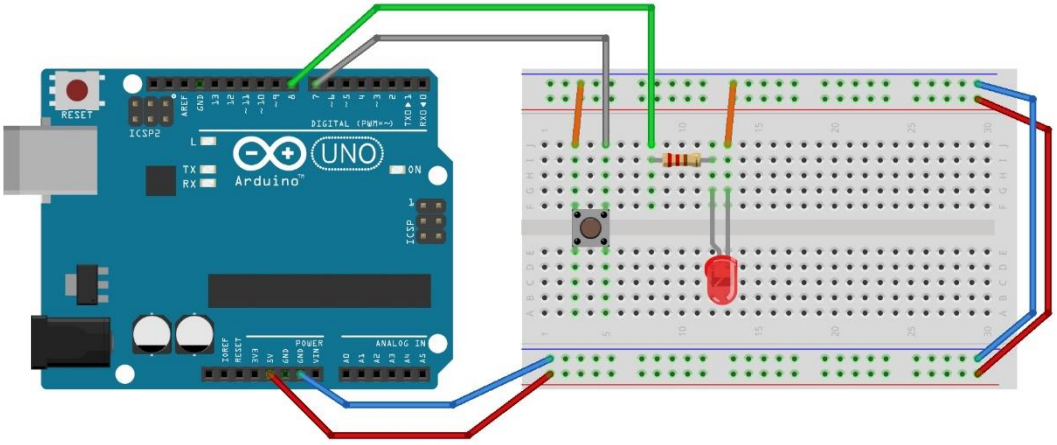
Si vous avez trouvé, bravo…
Vous pouvez modifier votre programme sachant que :
il est possible, pour éviter la succession des deux conditions "if" d'utiliser une méthode de tests combinés.

Observez le code ci-contre et modifiez votre programme en conséquence.

Exercice 2 : « interrupteur »

Reprenez le même montage que précédemment mais cette fois :

- ***Quand on appuie une fois sur le bouton, la LED s'allume (et reste allumée) ;***
- ***Lorsqu'on appuie à nouveau, la LED s'éteint (et reste éteinte).***



Exercice 3 : Mini-orgue

Dans ce programme, vous allez déclencher la génération d'une note lors de l'appui sur un bouton poussoir donné. Il faudra utiliser :

- 3 boutons poussoirs configuré en INPUT_PULLUP
- 1 piezo buzzer (cf. image ci-contre)

Concernant ce dernier, la patte indiquée + se branche sur un des connecteurs de l'arduino (sauf le 3 et le 11 pour des raisons d'interférence de timer..), celui-ci sera configuré en **sortie (comme pour la LED dans le programme précédent : pinMode…)**, l'autre se branche sur le ground (-)
Pour produire une note, vous devrez utiliser la fonction **tone**. Elle s'écrit de la manière suivante :
tone (pin du piezo , fréquence de la note à jouer) ;

Exemple : j'ai créé une variable int PIEZO=10 (10 correspondant au numéro du pin ou est branché la patte + du piezo).
Pour faire jouer un DO (dont la fréquence est 262), j'écris : **tone(PIEZO, 262) ;**

DO : fréquence 262
RE : fréquence 294
MI : fréquence 330