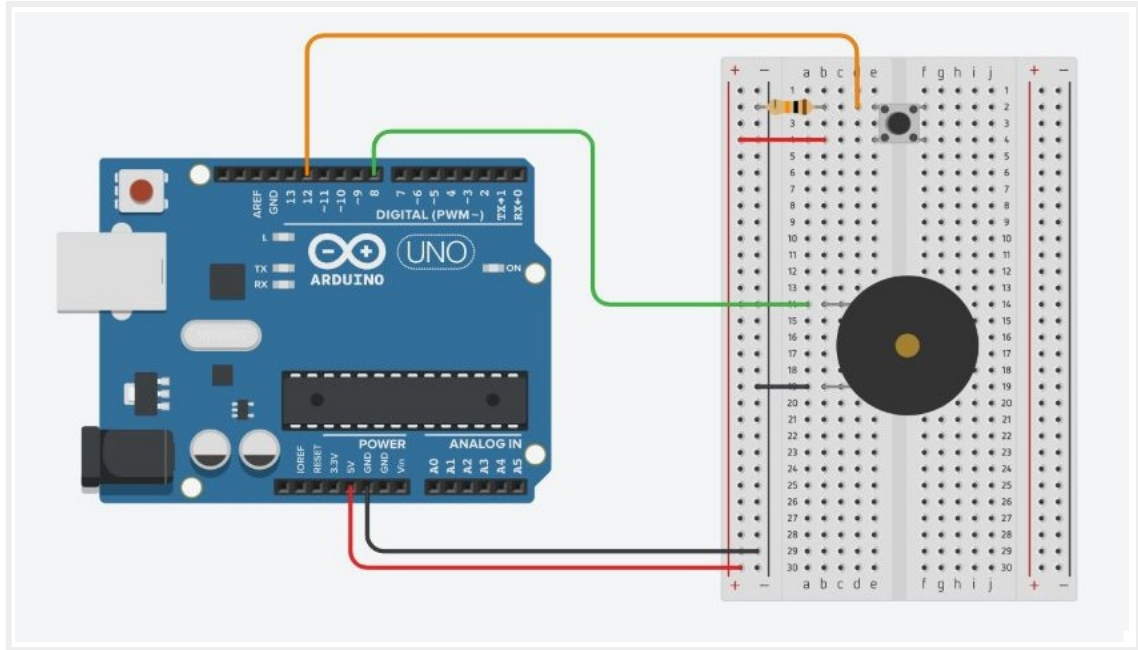


# Melody

## (Jouer une mélodie avec un Arduino)



### . Liste des composants :

- . 1 bouton poussoir
- . 1 résistance de 10 kΩ (résistance du circuit du bouton poussoir)
- . 1 buzzer
- . 1 plaque d'essai
- . Fils de connexion

### . Objectif

Dans cette activité, nous allons voir qu'il est possible de jouer une mélodie avec un Arduino.

La mélodie sera jouée en continu après un appui sur le bouton poussoir et arrêtée en appuyant de nouveau sur celui-ci.

## . Les différentes notes de musique

Chaque note est caractérisée par une fréquence fondamentale déterminée. Lorsque deux notes sont séparées d'une octave, le rapport de leur fréquence est égal à deux.

Dans la pratique, ces deux notes ont le même nom, comme par exemple le « la » de l'octave 3 et le « la » de l'octave 4, nommée couramment « la3 » et « la4 » pour éviter toute ambiguïté entre elles.

On appelle gamme l'ensemble des notes (de Do à Si) composant une octave donnée. Dans la gamme tempérée, c'est-à-dire celle utilisée dans la musique occidentale, l'octave est divisée en 12 demi-tons, ce qui correspond à 12 notes, en comptant les notes diésées (#).

Par exemple, pour l'octave 1, voici les valeurs de fréquence des 12 notes :

Note	Fréquence (Hz)
Do	65
Do# ou <u>Réb</u>	69
Ré	74
Ré# ou <u>Mib</u>	78
Mi	83
Fa	87
Fa# ou <u>Solb</u>	93
Sol	98
Sol# ou <u>Lab</u>	104
La	110
La# ou Sib	117
Si	123

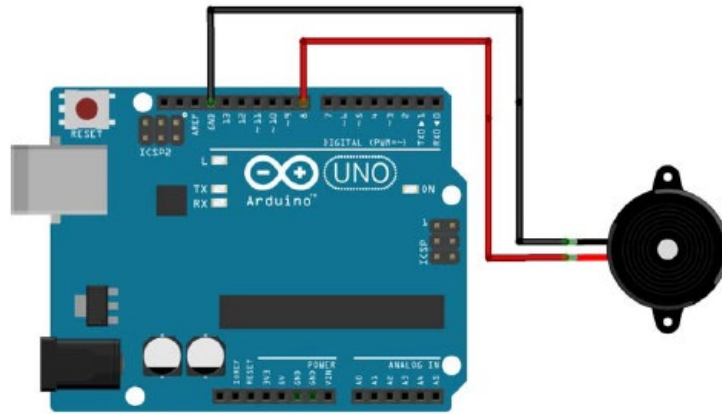
Le # signifie dièse et le b signifie bémol. Ce sont des altérations des notes de la gamme de base (Do, Ré, Mi, Fa, Sol, La, Si). Le dièse augmente la fréquence de la note et le bémol la diminue. Ainsi un La # est situé en fréquence entre le **La** et le **Si**

- En notation latine (la notation historique : do ré mi ...), on a l'habitude d'écrire que le diapason (la note qui nous sert de référence à 440 Hz) est le « **la3** ». Dans ce système, on peut se donner en repère que la première octave entièrement audible commence par le do0.
- En notation américaine (la notation scientifique : A B C ...), le « **la** » du diapason est le **A4**. C'est le système qui est utilisé en langage Arduino. Dans ce système, on a choisi de dire que le C (do) le plus grave d'un clavier de piano à 88 touches est le C1. Suivant ce repère, on a alors  
**A4 = 440 Hz.**

Voici un tableau qui référence la fréquence des notes de musique en hertz de la gamme tempérée (notation américaine en noir et notation latine en rouge):

Note \ octave	0 (-1)	1 (0)	2 (1)	3 (2)	4 (3)	5 (4)	6 (5)	7 (6)	8 (7)
<b>C</b> (Do)	16,35	32,70	65,41	130,81	261,63	523,25	1046,50	2093,00	4186,01
<b>C# – Db</b> (Do#)	17,32	34,65	69,30	138,59	277,18	554,37	1108,73	2217,46	4434,92
<b>D</b> (Ré)	18,35	36,71	73,42	146,83	293,66	587,33	1174,66	2349,32	4698,64
<b>D# – Eb</b> (Ré#)	19,45	38,89	77,78	155,56	311,13	622,25	1244,51	2489,02	4978,03
<b>E</b> (Mi)	20,60	41,20	82,41	164,81	329,63	659,26	1318,51	2637,02	5274,04
<b>F</b> (Fa)	21,83	43,65	87,31	174,61	349,23	698,46	1396,91	2793,83	5587,65
<b>F# – Gb</b> (Fa#)	23,12	46,25	92,50	185,00	369,99	739,99	1479,98	2959,96	5919,91
<b>G</b> (Sol)	24,50	49,00	98,00	196,00	392,00	783,99	1567,98	3135,96	6271,93
<b>G# – Ab</b> (Sol#)	25,96	51,91	103,83	207,65	415,30	830,61	1661,22	3322,44	6644,88
<b>A</b> (La)	27,50	55,00	110,00	220,00	440,00	880,00	1760,00	3520,00	7040,00
<b>A# – Bb</b> (La#)	29,14	58,27	116,54	233,08	466,16	932,33	1864,66	3729,31	7458,62
<b>B</b> (Si)	30,87	61,74	123,47	246,94	493,88	987,77	1975,53	3951,07	7902,13

## . Jouer une mélodie avec un Arduino



On utilisera la fonction **tone()** pour jouer les notes de musique (voir tableau des fréquences des notes) de la mélodie pendant la durée définie pour chaque note.

La durée des notes est généralement calculée en attribuant une seconde (1000 ms) à une ronde. Une blanche représente alors 500 ms (ronde/2), une noire, 250 ms (blanche/2, ou ronde /4), une croche, 125 ms (noire/2 ou ronde/8), etc...

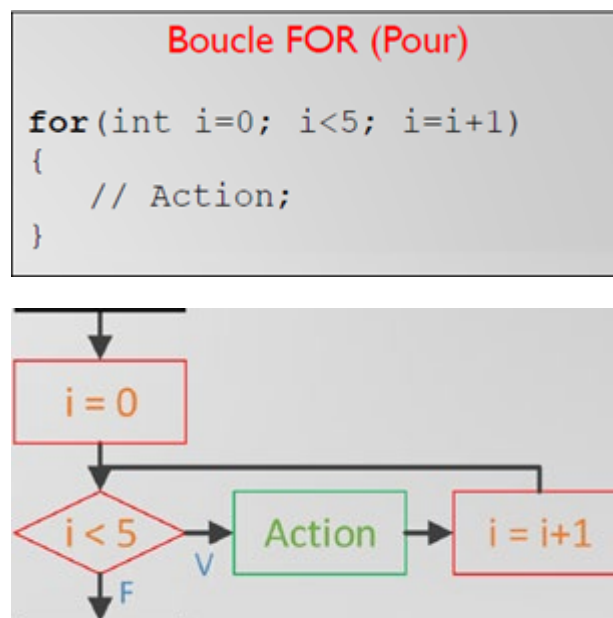
Pour bien distinguer les notes jouées par l'Arduino, il faut respecter un temps de pause entre chaque note, égal à la durée de la note + 30 % :

**Temps de pause (en ms) = Durée de la note (en ms) x 1,3**

Pour cela, le plus simple est de créer une liste des fréquences (**Freq**) des notes à jouer et une liste des durées (**Dur**) puis de demander à l'Arduino de jouer chaque élément, **i**, des listes à l'aide d'une boucle for :

```
for (int i = 0 ; i < 11 ; i++) {  
  tone (broche du piezo, Freq[i], Dur[i])  
  delay(Dur[i] x 1.3)  
}
```

. Rappel sur la programmation d'une boucle for:



## . Le programme

Le programme de l'activité pourra être modifié pour voir l'influence des variables (fréquence des notes, durée des notes, durée des silences entre les notes).

```
Melody

// Déclaration des constantes et variables

const int PinTone = 8;
const int PinButton = 12;
const int Notes[] = {262,294,330,262,294,330,349,392,330,349,392};
const int NoteDurations[] = {4,4,4,4,4,4,4,8,4,4,8};

int State = 0;
int ValButton = 0;
int OldValButton = 0;

// Initialisation des entrées et sorties

void setup() {
  pinMode (PinButton, INPUT);
}

// Fonction principale en boucle

void loop() {
  ValButton = digitalRead(PinButton);
  delay(10);
  if (ValButton == HIGH and OldValButton == LOW) {
    State = 1 - State;
  }
  if (State == 1) {
    for(int i = 0 ; i < 11 ; i++)
    {
      int NoteDuration = int(1000/NoteDurations[i]);
      tone(PinTone, Notes[i], NoteDuration);
      delay(1.3*NoteDuration);

      ValButton = digitalRead(PinButton);
      delay(10);
      if (ValButton == HIGH and OldValButton == LOW) {
        State = 1 - State;
        break;
      }
      OldValButton = ValButton;
    }
  }
  else {
    noTone(PinTone);
  }
  OldValButton = ValButton;
}
}
```

## Déroulement du programme :

– Déclaration des constantes et variables :

. **const int PinButton = 12** (constante nombre entier correspondant au n° de la broche sur laquelle le bouton poussoir est connecté)

. **const int PinTone = 8** (constante nombre entier correspondant au n° de la broche sur laquelle le buzzer est connecté)

. **const int Notes[ ]** (liste de constantes nombres entiers correspondant aux fréquences des notes à jouer)

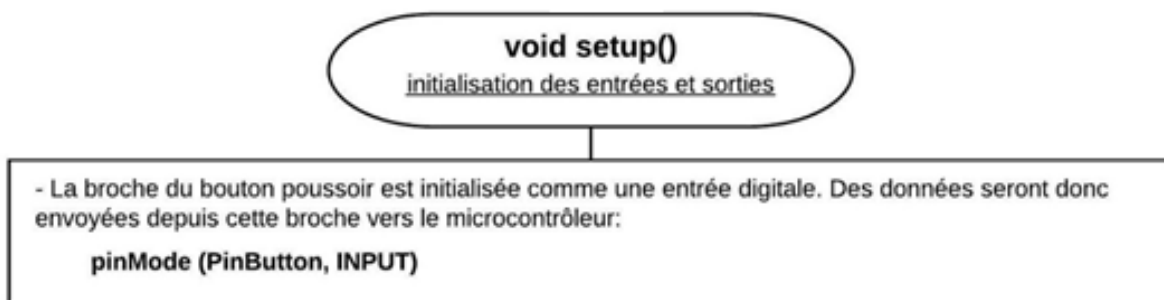
. **const int NoteDurations[ ]** (liste de constantes nombre entier correspondant aux durées des notes en fraction de seconde)

. **int State = 0** (variable nombre entier correspondant à l'action à effectuer)

. **int ValButton = 0** (variable nombre entier pour stocker la valeur de la broche du bouton poussoir)

. **int OldValButton = 0** (variable nombre entier pour stocker la valeur précédente de la broche du bouton poussoir)

– Initialisation des entrées et sorties :



– Fonction principale en boucle :

