Actions dans le temps – 26 mai 2022 / GL

La **pause** permet de figer l'exécution du programme
Utiliser l'exemple fourni avec l'IDE : File – Examples – 01.Basics – Blink
Il utilise :

- la LED présente sur la carte appelée **`LED_BUILTIN`**
- **`digitalWrite(..., 0 ou 1)`** pour activer le signal électrique (0 ou 1) sur la sortie **`LED_BUILTIN`**
- **`delay(1000)`** pour attendre 1000 millisecondes = 1 seconde
  on parle d'une attente active pendant laquelle le processeur n'est pas disponible pour l'utilisateur

Le code ci-dessous produit le même résultat (allumer – éteindre LED chaque seconde) tout en permettant d'exécuter des instructions lors des pauses → **on parle d'attente passive**

Etudier la fonction millis() https://www.arduino.cc/reference/en/language/functions/time/millis/ et
ce code http://gelit.ch/Arduino/millis.ino

Il est possible de démarrer **plusieurs fonctions** : fonctionA chaque seconde, fonctionB toutes les 2 secondes, …

Mesurer un **interval de temps** avec http://gelit.ch/Arduino/interval.ino
Résultat : **`Begin -  Ellapsed Time en ms = 128`**

Modifier la valeur de N avec **`N<50000;`** pour constater que le résultat est bien divisé par 2

Supprimer la fonction digitalRead → **`for (int N=1; N<50000; N++) {}`** pour constater que l'interval < 1 ms

Remplacer millis() par micros() pour observer un interval de 33 µs (microseconde)
1 seconde équivaut à 1000 ms (millisecond) et 1000000 = 10^6 µs (microseconde)

Conclusion :

- Lors de son exécution, chaque instruction – fonction et structures de contrôle consomme du temps
- Ce temps est heureusement négligeable à l'échelle humaine (lire le clavier – écrire à l'écran)

L'élément temporel est TOUJOURS présent et doit être bien compris lors de l'exécution d'un programme informatique
Nous savons mesurer le temps consommé par une(des) instructions, fonctions ou structures de contrôle
Ce programme informatique manipule des **données** (stockées dans des **variables**) **qui apparaissent et disparaissent !!!**

Cet exemple illustre la créarion d'une variable **locale** à la procédure dont la durée de vie est limitée !!!
Utiliser http://gelit.ch/Arduino/var1.ino

```
void setup() {
  Serial.begin(115200); Serial.println("Variable locale");
}

void loop() {
  int N;    // Cette variable est créée
  Serial.println(N);  // Une variable locale est initialisée avec 0
  N++; // équivalent à N = N+1;
  Serial.println(N); // Une variable locale reste LOCALE !!!
  delay(1000);
};  // La variable N est SUPPRIMEE
```

Utilisons une variable **GLOBALE** dans http://gelit.ch/Arduino/var2.ino

```
int N; // Variable globale au programme est créée

void setup() {
  Serial.begin(115200); Serial.println("Variable globale");
  Serial.println(N);  // Une variable est initialisée avec 0
}

void loop() {
  N++;
  Serial.println(N); // la variable est GLOBALE !!!
  delay(1000);
}
// La variable N est SUPPRIMEE à la fermeture du programme
```