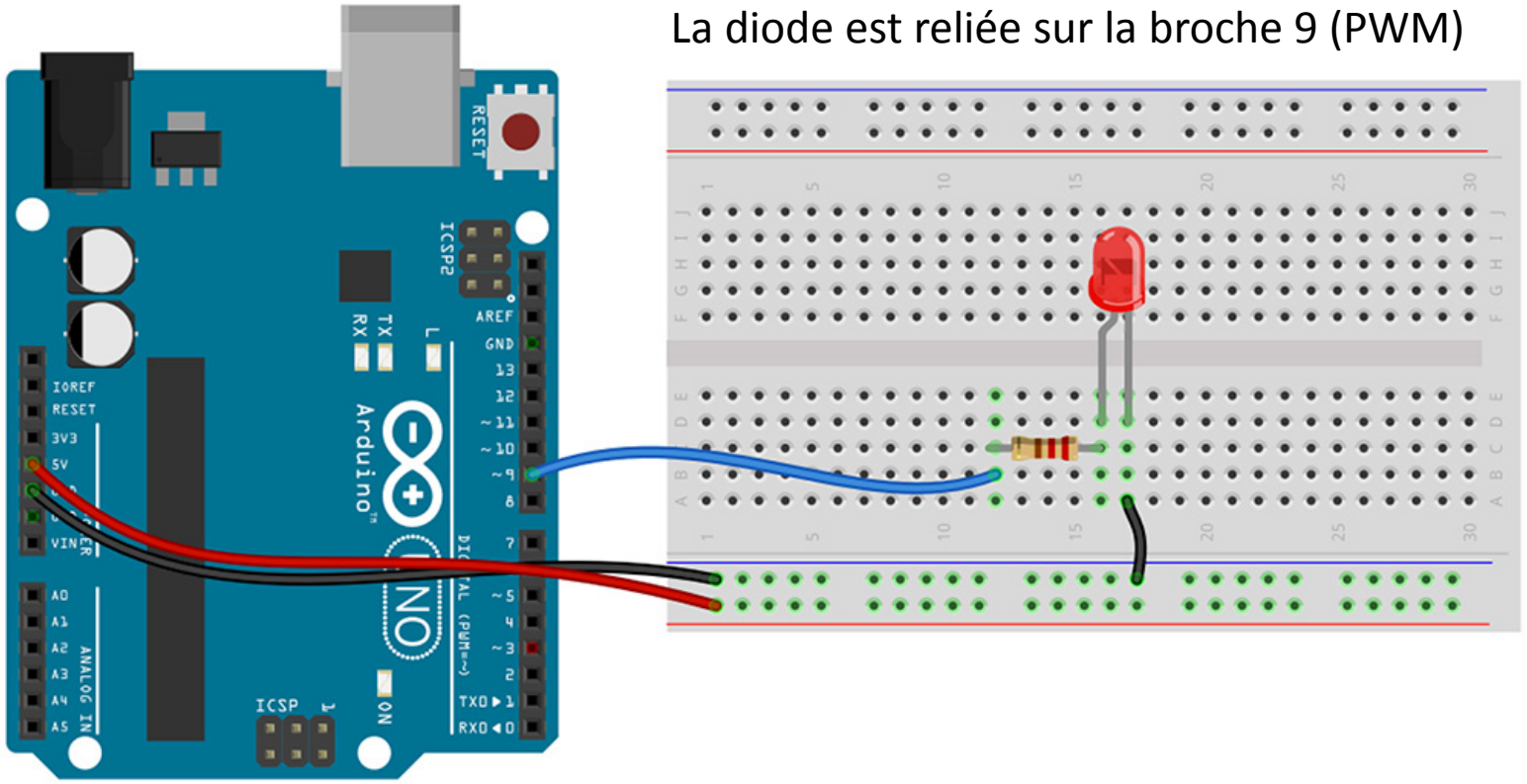
Initiation Arduino #2

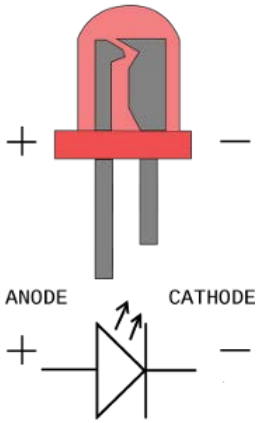
Club astro QF – Pascal ANDRE mars 2016

Objectif

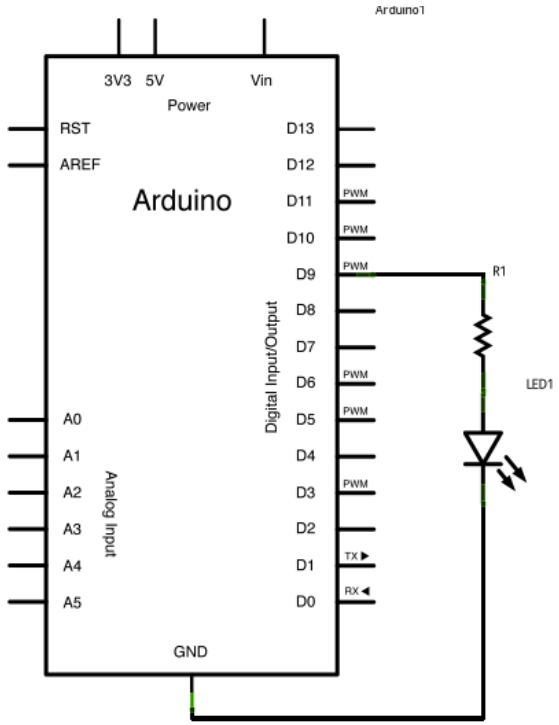
- Comprendre les E/S Digitales (suite) : fonctionnement du mode PWM (Pulse Width Modulation) broches N° 3,5,6, 9,10,11 repérées par « pwm » ou ~
- Réaliser un montage breadbord avec une diode led dont on fera varier l'intensité en jouant sur la modulation de son temps d'alimentation
- Rappel sur la loi d'ohm et le code couleur des résistances
- Comprendre et modifier le code arduino

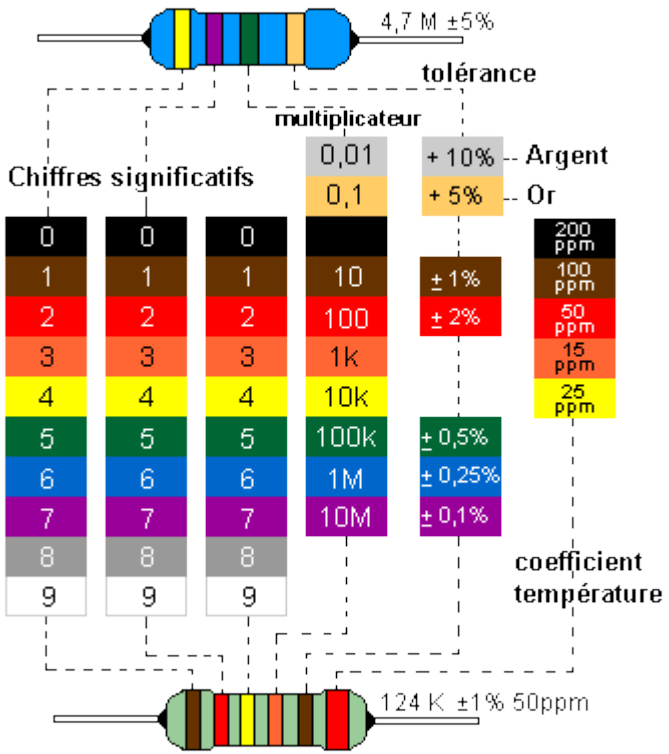
La diode est reliée sur la broche 9 (PWM)

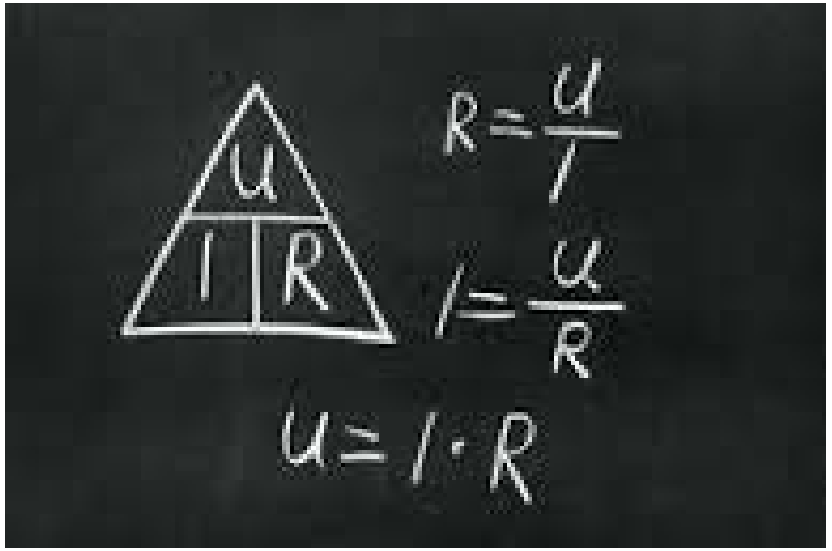




+ = patte la plus longue

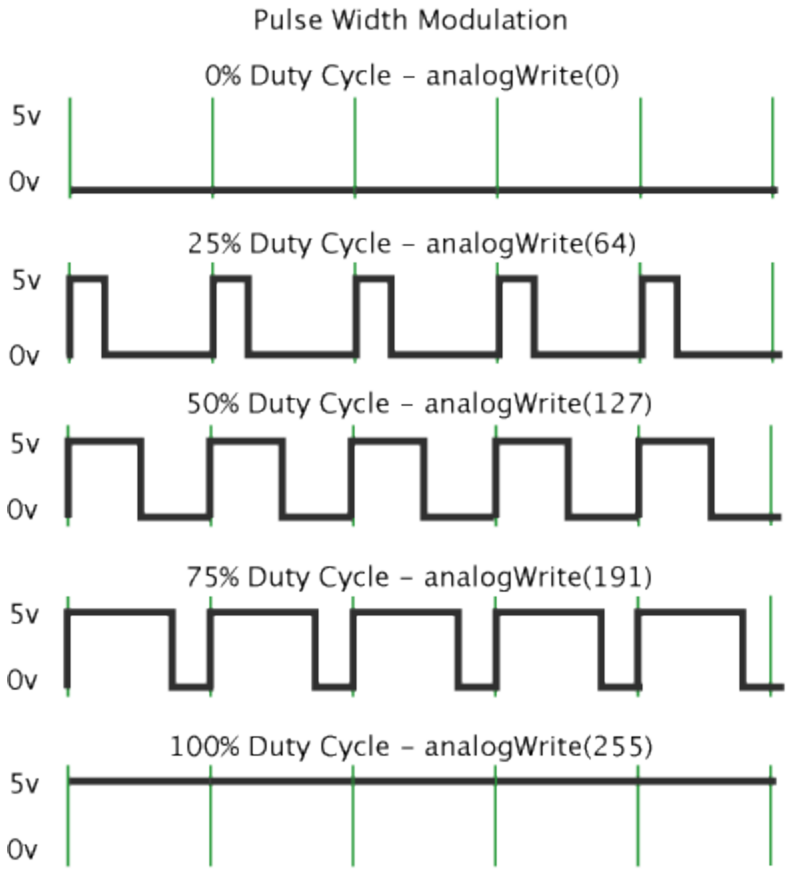






La valeur de la résistance montée sur la breadbord : rouge rouge marron or = 220 ohms +/-5%

Le principe du PWM



à l'aide d'une impulsion rapide, on va faire varier la proportion de temps où la LED est allumée, ce qui donnera un effet d'allure analogique de variation de la luminosité de la LED

Pourquoi 255 ? = 1 + 2+4+8+16+32+64+128 = 1
Octet = « 11111111 » en binaire

Le code correspondant et son explication
Examples > Basics> Fade

```
/*
Variation de luminosité d'une LED

Cet exemple montre comment faire varier la luminosité
sur la broche 9 en utilisant l'instruction analogWrite()


Cet exemple est dans le domaine public
Traduction française par X. HINAULT - www.mon-club-elec.fr

 */

int luminosite = 0;   // % de temps où la LED est allumée
int variation = 5;    // intervalle de variation de la luminosité

void setup()  {
  // configure la broche 9 en SORTIE
  pinMode(9, OUTPUT);
}

void loop()  {
  // applique une impulsion de largeur correspondant à la luminosité sur la broche 9
  analogWrite(9, luminosite);

  // modifie la luminosité pour le passage suivant dans la boucle loop()
  luminosite = luminosite + variation;

  // inverse le sens de variation de la luminosité quand on atteint les valeurs extremes 0 ou 255
  if (luminosite == 0 || luminosite == 255) {
    variation = -variation ;
  }
  // pause de 30 millisecondes pour voir l'effet de variation
  delay(30);
}
```

Au niveau de la fonction setup ()
A part la déclaration de la broche 9 utilisée en sortie, il n'y a rien à faire de spécial dans la fonction setup().
pinMode(9, OUTPUT);
Au niveau de la fonction loop()
L'instruction analogWrite() utilisée dans la fonction princpale loop() requiert deux arguments :
le premier indiquant la broche utilisée pour générer l'impulsion,
le second indiquant la valeur PWM à utiliser pour générer l'impulsion (0 pour état HAUT pendant 0% du temps et 255 pour état HAUT pendant 100% du temps).
Pour obtenir la variation de la luminosité de la LED, il est nécessaire d'augmenter progressivement la valeur PWM utilisée avec l'instruction analogWrite, de 0 (0% du temps au niveau HAUT) à 255 (100% du temps au niveau HAUT), puis de revenir à la valeur 0 et de recommencer le cycle. Dans le programme ci-dessous, la valeur PWM est fixée en utilisant une variable appelée **luminosite**. A chaque passage de la boucle loop(), cette variable est augmentée (ou incrémentée) par la valeur de la variable **variation** (qui fixe le "cran" de variation de la valeur PWM).
Lorsque la variable **luminosite** arrive à sa valeur extrême (soit 0 ou 255), la variable **variation** est changée en valeur opposée (négative ou positive). En d'autres termes, si la variable **variation** vaut 5, elle est fixée à -5. Si elle vaut -5, elle est fixée à +5. Ainsi, au passage suivant de la boucle loop(), cela inverse le sens de variation de la variable **luminosite**.
L'instruction analogWrite() va changer la valeur PWM très rapidement : on utilisera donc une instruction delay() à la fin du programme pour fixer la vitesse de variation de la luminosité.
Essayer de changer la valeur utilisée pour l'instruction delay() et observer ce qui se passe.

Effet : La luminosité de la LED varie crescendo puis decrescendo et boucle ainsi de suite.

Une Variante avec l'instruction for (initialization; condition; incrementation) { //instruction(s)à exécuter; }

```
int ledPin = 9;    // LED connectée à la broche 9

void setup() {
 pinMode(9, OUTPUT);
 }

void loop() {
  // augmente l'intensité lumineuse par incrément de 5:
  for (int fadeValue = 0 ; fadeValue <= 255; fadeValue += 5) {


    analogWrite (ledPin, fadeValue);


  // attendre 30millisec

    delay(30);
  }

   for (int fadeValue = 255 ; fadeValue >= 0; fadeValue -= 5) {
      analogWrite(ledPin, fadeValue);
      delay(30);
   }
}
```

L'instruction **for** est utilisée pour répéter l'exécution d'un bloc d'instructions regroupées entre des accolades. Un compteur incrémental est habituellement utilisé pour incrémenter et finir la boucle. L'instruction **for** est très utile pour toutes les opérations répétitives et est souvent utilisées en association avec des tableaux de variables pour agir sur un ensemble de données ou broches.

Exemple condensé for (int i=0; i <= 255; i++){ // instruc}

C’est fini pour aujourd’hui

- Prochaines séances : lire l’état d’un bouton poussoir, visualiser une valeur analogique (ex potentiomètre).