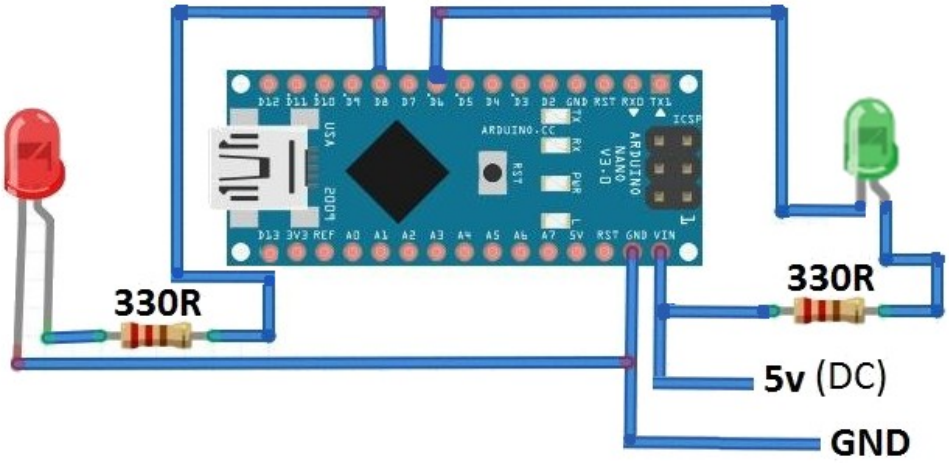
classe Led

Nous voulons commander une ou plusieurs LED, connectées à une *pin* d'une carte Arduino, en gardant la possibilité de les allumer par un niveau haut ou bas sur la pin correspondante. Pour cela, nous vous proposons de mettre en œuvre une classe c++, **Led** et de l'utiliser dans un programme de test.



Caractéristique de la LED:

- son état (**true**=allumée et **false**=éteinte)
- un entier représentant le numéro de la broche Arduino (dépend du type de carte)
- un booléen représentant le type de connexion électrique (allumée niveau haut ou bas)

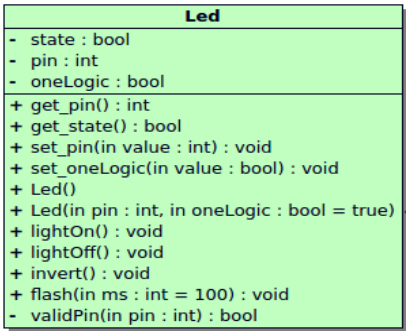
Comportement de la LED:

- une méthode **lightOn**, qui allume la LED
- une méthode **lightOff**, qui l'éteint
- une méthode **invert**, qui inverse son état
- une méthode **flash**, qui émet un flash lumineux d'une durée spécifiée en argument (millisecondes)

Fonctionnalités supplémentaires :

- Quatre accesseurs (getter/setter) :
  - **get_pin**, et set_pin pour obtenir et modifier le numéro de la broche
  - **get_state** pour obtenir l'état de la LED.
  - **set_oneLogic** pour modifier le mode de branchement électrique
- Deux constructeurs :
  - le premier, par défaut, mettra la valeur -1 dans le numéro de broche (broche indéfinie), **true** dans le mode de commande (logique "1") et **false** dans l'état (LED éteinte).
  - un autre constructeur permettra de préciser le numéro de broche et le mode de commande (logique "1" par défaut)
- Une méthode privée **validPin** qui vérifie que la pin choisie est acceptable pour la carte utilisée.

Représentation UML de la classe Led :



On demande :

- Créer et implémenter la classe Led (fichiers led.h et led.cpp)
- brancher une des LED sur la pin de votre choix.
- Ecrire un programme permettant d'instancier une Led en utilisant les caractéristiques correspondante à votre branchement, puis de la commander à partir de la liaison série ('1' allume, '0' éteint, 'f' émet un flash d'une demi seconde.
- En modifiant le câblage, vérifier que le programme fonctionne quelque soit la *pin* et la logique de commande utilisées.
- Brancher les deux autres LED, sur les pins de votre choix, puis modifier le programme pour pouvoir allumer/éteindre chacune des LED sur le terminal. (ex : 'A'/'B' pour la première, 'C'/'D' pour la deuxième et 'E'/'F' pour la troisième).
- S'il vous reste du temps, utiliser un tableau de 3 Led, à la place de trois objets distincts, ceci devrait permettre de simplifier considérablement le programme (utilisez le fait que les codes ASCII de A, B, C, D, E, F se suivent).

Exemple de programme de test :

```
#define       LED_PIN     2          // la broche de la led
#define       LOGIC       true       // la logique de commande, niveau haut ou bas
#define       FLASH_MS    500        // durée du flash
#include "Led.h"

Led led;           // instanciation de l'objet led de la classe Led

void setup(){
    // Initialise la bibliothèque Arduino et de la liaison série
    init();
    Serial.begin(115200);
    Serial.println("Programme de test !");

    led.set_pin(LED_PIN);
    led.set_oneLogic(LOGIC) ;
}

 void loop(){
      // lecture d'un caractère sur la liaison série
    char c = Serial.read();

    // changement du mode de fonctionnement
    switch (c){
    case '1': led.lightOn();  Serial.print(" on");  break;
    case '0': led.lightOff(); Serial.print(" off"); break;
    case 'f': led.flash(FLASH_MS); break ;
    default : Serial.print(" what ?") ;
    }
    Serial.println() ; Serial.print('>') ;
 }
```