**Objectif** : ***Mesurer une température en °C à l'aide d'un capteur GROVE et l’afficher à l’aide d’un afficheur LCD* GROVE.**

**A partir du *dossier ressource* ci-dessous, réaliser le TP de mesure de température à l'aide d'un capteur GROVE et d’un afficheur LCD GROVE.**

**Matériel nécessaire** :

● 1 **carte Arduino Uno**,

● 1 **Shield Grove**

● 1 **capteur de température grove** **v1.0**

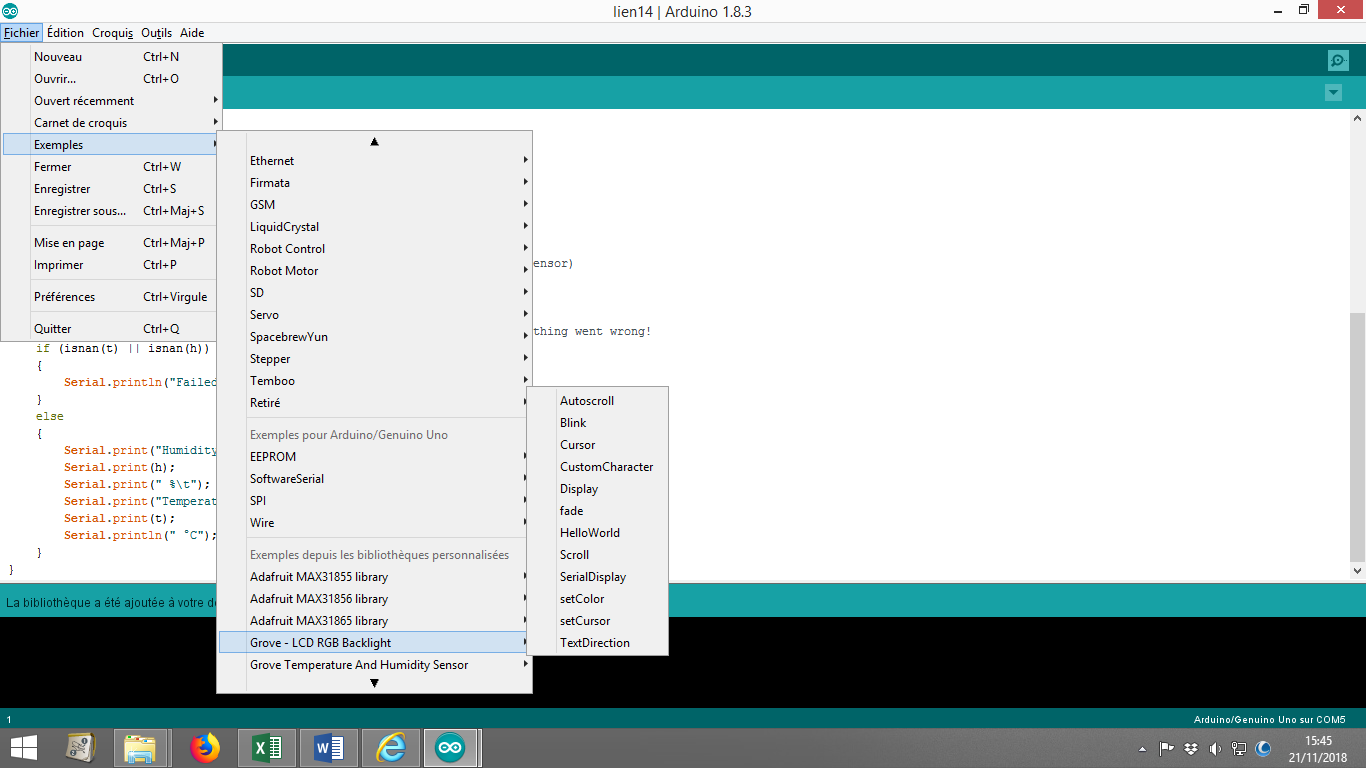
● 1 **afficheur LCD Backlight grove v2.0** **connecté au bus I2C**

● La librairie **math.h**

● La librairie **Grove-LCD RGB Backlight** disponible à partir du lien ci-dessous :

[**http://wiki.seeedstudio.com/Grove-LCD\_RGB\_Backlight/**](http://wiki.seeedstudio.com/Grove-LCD_RGB_Backlight/)

**Enregistrer le fichier zip dans le répertoire librairies, installer la librairie et tous les sketchs suivants seront à votre disposition**



 **Présentation du capteur de température** :

● Le capteur de **température Grove** **v1.0**

Ce capteur utilise une thermistance sous forme de valeur de résistance, qui est alors utilisée pour altérer **Vcc**. En fonction de la température mesurée (entre **-40°C** et **+ 125°C**), il délivre **un signal analogique** de **0 à 5 Vcc**. La résistance de la thermistance diminue quand la température augmente. Dans le sketch ci-dessous, ce capteur de température est relié en **A0**.

**Dans Arduino sur les entrées analogiques.**

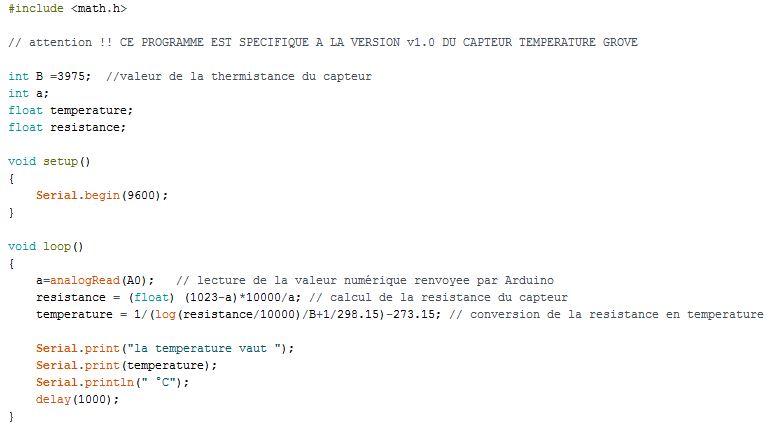
**CAN de 10 bits** qui génère une valeur numérique de **0 à 1023**

**En entrée d’Arduino :**

Signal analogique sous forme d’une tension de **0V à 5V**

**Dans le programme :** Valeur lue par l’instruction analogRead()

**Sketch Arduino pour la version v1.0 :**



**Présentation de l’afficheur LCD Backlight grove** **:**

L’afficheur **LCD Backlight grove** intègre un **afficheur LCD alphanumérique 2 x 16 caractères** avec rétro-éclairage **RGB**. L’afficheur doit être raccordé à une platine **Arduino** au moyen d'un **shield grove**.

Le module est destiné à être piloté via un **port I2C** (***communication série sur 2 fils***) d’**Arduino**. Il faudra donc inclure au programme la librairie **Wire.h.**

Caractéristiques de l’afficheur :

* Modèle **2 x 16 caractères**
* Rétro-éclairage pilotable pour réduire la consommation
* Rétro-éclairage **RGB** permettant de choisir la couleur de fond.

Exploitation de l’afficheur :

Pour utiliser ce module vous devez inclure la bibliothèque **rgb\_lcd.h**

**Détails des fonctions de la bibliothèque rgb\_lcd.h**

**lcd.begin(x,y);**

Définition de l’écran : x colonnes et y lignes

**lcd.print(data); ou lcd.print(data,BASE);**

Affiche un "texte", une variable, une donnée à l’écran.

BASE (optionnel): base dans lesquelles seront affichés les nombres:

BIN pour binaire (base 2),

DEC pour décimal (base 10),

OCT pour octal (base 8),

HEX pour hexadécimal (base 16).

**lcd.blink();**

Afficher le curseur clignotant à l'écran.

**lcd.noblink();**

Désactive le clignotement du curseur.

**lcd.cursor();**

Affiche le curseur de l'écran.

**lcd.noCursor();**

Masque le curseur de l'écran.

**lcd.display();**

Active l'écran LCD après qu'il a été éteint avec noDisplay( ) . Cela permettra de restaurer le texte (et le curseur) qui était sur l'écran.

**lcd.noDisplay();**

Eteint l'écran LCD sans perdre le texte actuellement affiché.

**lcd.setCursor(x,y);**

x = 0 et y = 0 le curseur est en haut à gauche (lcd.home() ;)

x = 16 et y = 1 le curseur est en bas à droite

**lcd.home();**

Positionne le curseur dans le coin supérieur gauche de l'écran LCD.

**lcd.clear();**

Efface l'écran LCD et positionne le curseur dans le coin supérieur gauche.

**lcd.autoscroll();**

Déplace le texte d'un espace vers la gauche à chaque fois une lettre est ajoutée.

Active le défilement automatique de l'écran LCD. Cela provoque la sortie de chaque caractère à l'écran pour pousser caractères précédents en charge par un espace. Si la direction de texte actuel est de gauche à droite (par défaut), l'écran défile vers la gauche ; si la direction actuelle est de droite à gauche, l'écran défile vers la droite. Cela a pour effet de sortir chaque nouveau caractère au même endroit sur l'écran LCD.

**lcd.noAutoscroll();**

Désactive le défilement automatique de l'écran LCD.

**lcd.scrollDisplayLeft();**

Fait défiler le contenu de l'affichage (texte et le curseur) un espace vers la gauche.

**lcd.scrollDisplayRight();**

Fait défiler le contenu de l'affichage (texte et le curseur) un espace vers la droite.

**lcd.leftToRight();**

Règle la direction pour que le texte s’écrive à l'écran LCD de gauche à droite (la valeur par défaut). Cela signifie que les caractères suivants écrits à l'écran vont de gauche à droite, mais n’affecte pas le texte précédemment sorti.

**lcd.rightToLeft();**

Règle la direction pour que le texte s’écrive à l'écran de droite à gauche (la valeur par défaut est de gauche à droite). Cela signifie que les caractères suivants écris à l'écran vont de droite à gauche, mais n’affecte pas le texte précédemment sorti.

**lcd.write(data) ;**

Écrit un caractère à l'écran LCD.

Voir exemple avec createChar(num, data) à la page suivante.

**lcd.createChar(num, data);**

Créez un caractère personnalisé (glyphe) pour une utilisation sur l'écran LCD. Jusqu'à huit caractères de 5x8 pixels sont supportés (numérotés de 0 à 7). L'apparence de chaque caractère personnalisé est spécifiée par un ensemble de huit octets, un pour chaque ligne. Les cinq bits les moins significatifs de chaque octet déterminent les pixels de cette ligne. Pour afficher un caractère personnalisé sur l'écran, write() son numéro.

Voir l'exemple ci-dessous pour l’affichage d’un **cœur** :

byte heart[8] = {

0b00000,

0b01010,

0b11111,

0b11111,

0b11111,

0b01110,

0b00100,

0b00000

};

void setup() {

lcd.createChar(0, heart);

lcd.begin(16, 2);

cd.write(byte(0));

}

void loop() {}

**lcd.setRGB(color\_r,color\_g,color\_b);**

color\_r : valeur entière de l’intensité la couleur rouge de 0 (absent) à 255 (maxi).

color\_g : valeur entière de l’intensité la couleur verte de 0 (absent) à 255 (maxi).

Color\_b : valeur entière de l’intensité la couleur bleue de 0 (absent) à 255 (maxi).

**Exemples :**

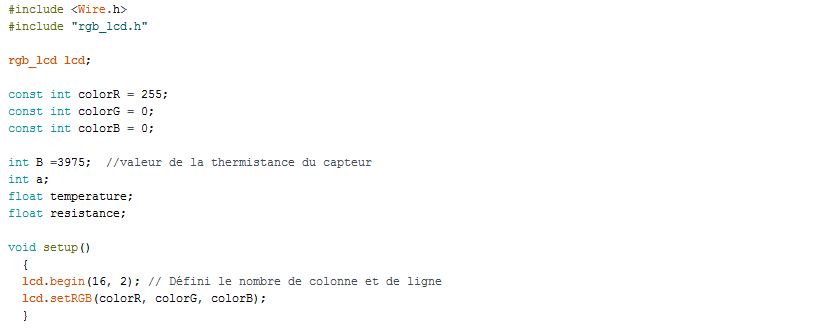
lcd.setRGB(0,0,0); // écran éteint

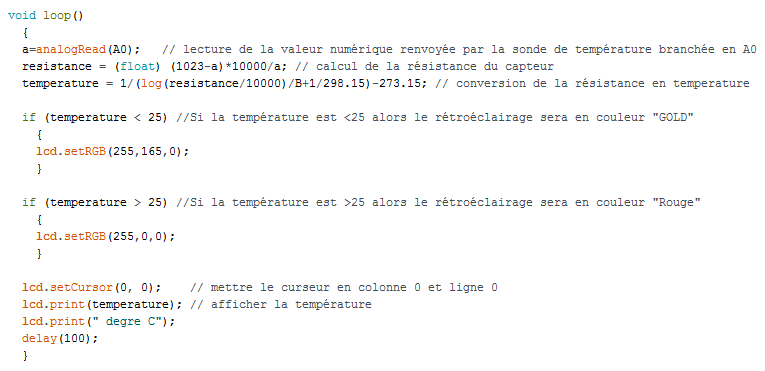
lcd.setRGB(255,255,255); // écran blanc (éclairage maxi)

lcd.setRGB(255,0,255); // écran magenta (éclairage maxi)

lcd.setRGB(255,255,0); // écran jaune (éclairage maxi)

**Sketch qui affiche le retour du Température grove v1.0 sur le LCD Backlight grove v2.0**





|  |
| --- |
|  |
| **sur arduino** |

**Tinkercad :**

|  |
| --- |
| **#include <Adafruit\_LiquidCrystal.h>**  **int seconds = 0;**  **int a;**  **Adafruit\_LiquidCrystal lcd\_1(0);**  **void setup()**  **{**  **lcd\_1.begin(16, 2);**  **}**  **void loop()**  **{**  **a= analogRead(A0); // reading the digital value returned by the temperature probe connected to 20 resistance (float) (1023-a)\*10000/a: // calculation of the resistance of the temperature sensor 1/(log(resistance/10000)/B +1/298.15)-273.15: // converting resistance into temperature**  **float volt = a \* 5.0;**  **volt = volt / 1024.0;**    **float t = (volt - 0.5) \* 100 ;**  **if (t < 100) //If the temperature is <25 then the backlight will be in "GOLD" color**  **{**  **//lcd\_1.setRGB(255,165,0);**  **}**  **if (t > 100) //S1 the temperature is >25 then the backlighting will be in “Red” color**  **{**  **//lcd\_1.setRGB(255, 0,01);**  **}**  **lcd\_1.setCursor(0, 0); // set the cursor in column 0 and row**  **lcd\_1.print(t); // display the temperature**  **lcd\_1.print(" degres celcus");**  **delay (100);**  **}** |