Lycée Felix Le Dantec
Lannion

Arduino et Processing

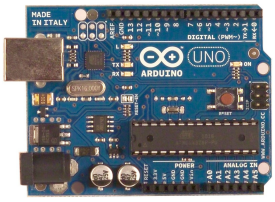
TSTI2D
TP10 - 4h

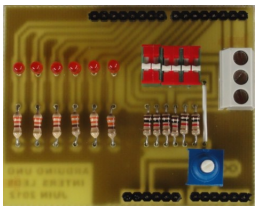
NOM :	CLASSE :
PRÉNOM :	

Condition :	• travail en binôme ; durée 4 heures
Matériel :	• un ordinateur sous Ubuntu avec les logiciels Arduino et Processing • une carte Arduino Uno, un shield interrupteurs et leds, un capteur de température TMP36
Documents :	• le sujet du TP • les cours sur Processing et Arduino

L'interface graphique programmable Processing, véritable "couteau suisse logiciel", écrite en Java (multiplateforme donc...) par le MIT, libre et opensource, donne la possibilité de créer des interfaces graphiques côté PC pour communiquer avec le système Arduino. On peut ainsi commander à la souris ou au clavier la carte Arduino ou bien afficher sur le PC des graphiques à partir des données reçues depuis la carte Arduino. Processing permet également de réaliser de la capture vidéo, du traitement d'image, des applications réseau serveur ou client, de la lecture de son, de la reconnaissance vocale...

On utilise dans cette première partie la maquette Arduino Uno et la maquette shield interrupteurs et leds.





Pour rappel, cette carte shield est composée de 6 leds, 6 interrupteurs, un potentiomètre et un bornier connecté à la liaison série de l'Arduino. Le brochage de la carte Shield est le suivant :

broche	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	A0
nom	I2	I3	I4	I5	I6	I7	led8	led9	led10	led11	led12	led13	pot

I = Interrupteurs pot = potentiomètre

1. Installer les bibliothèques sous Processing

Nous allons utiliser deux bibliothèques pour Processing.

- ✔ Pour installer cette bibliothèque, démarrez Processing, passez en mode Java. Cliquez sur *Outils* > Ajouter un outil... > *Libraries* ... .
- ✔ Recherchez la bibliothèque *Arduino (Firmata)* puis installez-la.
- ✔ De la même façon recherchez puis installez la bibliothèque *controlP5*. Fermez la fenêtre.
- ✔ Si vous travaillez sous Linux, utilisez Nautilus, l'explorateur de fichiers, pour renommer le fichier /sketchbook/libraries/arduino/library/Arduino.jar en arduino.jar (remplacez le A majuscule par un a minuscule).
- ✔ Fermez Processing.

2. Télécharger le firmware dans l'Arduino

Pour que notre maquette Arduino Uno communique avec Processing, il nous faut installer le firmware StandardFirmata.

- ✔ Démarrez le logiciel Arduino. Ouvrez le fichier exemple Firmata > Standardfirmata. Téléchargez ce fichier dans l'Arduino. Fermer le logiciel Arduino.

Votre Arduino est prêt à communiquer avec Processing.

3. Commander les leds à partir de Processing

1.1 Allumer une led

L'objectif est d'allumer la led D13 à partir de Processing. On donne le programme sous Processing :

```
/* prog1_allumeD13.pde */

import processing.serial.*;
import cc.arduino.*;

Arduino arduino;

void setup()
{
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  arduino.pinMode(13, Arduino.OUTPUT);
}

void draw()
{
  arduino.digitalWrite(13, Arduino.HIGH);
}
```

✔ Redémarrez Processing et copiez-collez ce programme sous le nom *prog1_allumeD13.pde.* Testez.

✔ Retrouvez dans ce programme les éléments suivants :
- mise à 1 de la sortie 13 de l'arduino
- déclaration d'un objet arduino de type Arduino
- déclaration de la broche 13 de la maquette arduino en sortie

1.2 Commander une led à partir du clavier

L'objectif est ici d'allumer la led D10 du shield lorsqu'on appuie sur la touche '1' du clavier et de l'éteindre lors de l'appui sur la touche '0'.

✔ Modifiez le programme précédent. Enregistrez le fichier sous *prog2_ledetclavier.pde*. Testez.

Validation prof :

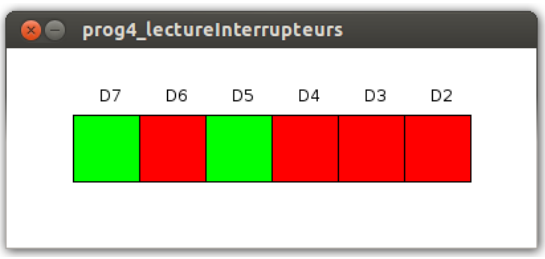
✔ Modifiez votre programme afin de commander la led D10 du shield avec une variation de la luminosité à partir des touches 0 à 9 du PC. L'éclairement varie en fonction du numéro de la touche appuyée : 0=led éteinte ; 9=led allumée pleine puissance. Enregistrez le fichier sous *prog3_variationD10.pde*. Testez.

Validation prof :

1.3 Lire les états des interrupteurs

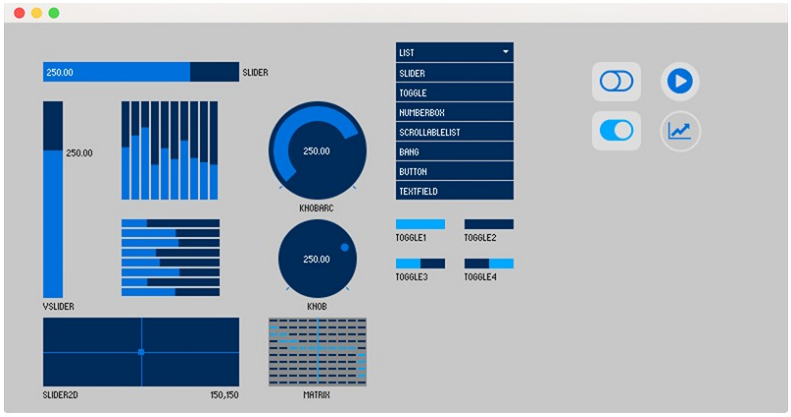
L'objectif est d'afficher dans la fenêtre de Processing les états des 6 interrupteurs de la maquette shield.

✔ Modifiez votre programme afin de réaliser cet affichage. La fenêtre sous Processing, de taille 400x150 pixels, ressemblera à celle ci-dessous : un carré vert pour un état haut, un carré rouge pour un état bas. Enregistrez le fichier sous *prog4_lectureInterrupteurs.pde*. Testez.



Validation prof :

Nous allons utiliser maintenant pour la gestion de l'interface utilisateur la bibliothèque ControlP5 que vous avez installé. Cette bibliothèque nous propose des boutons, des cases à cocher, des champs textes, des sliders, etc.



Quelques éléments disponibles dans la bibliothèque ControlP5

1.4 Commander une led à partir d'un bouton graphique

L'objectif est ici d'allumer la led D10 du shield lorsqu'on appuie sur un bouton affiché à l'écran. On donne un programme partiel qui crée un bouton et affiche son état dans la console de Processing.

```
/* prog5_boutonD10.pde */

import processing.serial.*;
import cc.arduino.*;
import controlP5.*;

Arduino arduino;
ControlP5 controlP5;

void setup()
{
  size(400, 350);
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  arduino.pinMode(10, Arduino.OUTPUT);
  controlP5 = new ControlP5(this);
  controlP5.addToggle("led10", false, 100, 100, 30, 20);
}

void draw()
{
  background(255);
}

void led10(int valeur)
{
  println("valeur de led10 : "+valeur);
}
```

déclaration d'un bouton *led10*

la fonction *led10* est automatiquement appelée lors d'un appui sur le bouton *led10*

✔ Retrouver dans le programme les éléments suivants :
- création d'une fenêtre de 400x350 pixels ;
- création d'un bouton de type Toggle (= bascule, 2 états) ;
- écriture de l'état du bouton sur la console de Processing ;

✔ Copiez-collez ce programme et nommez-le *prog5_boutonD10.pde*. Testez.

✔ Modifiez le programme afin de commander la led D10 du shield à partir du bouton de type Toggle « led10 ». Enregistrez le fichier sous *prog6_boutoncommandeD10.pde*. Testez.

Validation prof :

✔ Modifiez le programme afin de commander les 6 leds de la carte shield à partir de 6 boutons de type Toggle placés sur votre interface graphique. Enregistrez le fichier sous *prog7_6boutons.pde*. Testez.

Validation prof :

1.5 Commander une led à partir d'un slider

L'objectif est ici d'allumer la led D9 de l'Arduino à l'aide d'un slider affiché à l'écran.

✔ Ouvrez le fichier exemple ControlP5slider et enregistrez ce fichier sous le nom *prog8_slider.pde*. Testez.

✔ Modifiez le programme afin d'obtenir le slider ci-dessus pour commander la led D9. Testez.

Validation prof :

✔ Modifiez le programme afin d'ajouter 2 sliders pour commander les leds D10 et D11 de la carte shield. Enregistrez le fichier sous *prog9_3sliders.pde*. Testez.

Validation prof :

4. Afficher la température de la salle dans une interface graphique

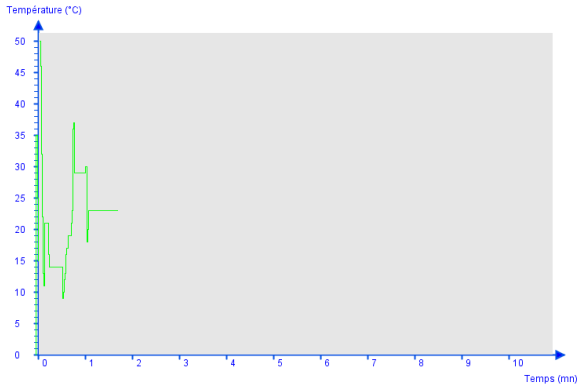
✔ Retirez soigneusement le shield interrupteurs et leds. Câblez sur l'entrée analogique de l'Arduino un capteur de température TMP36.

✔ En utilisant le travail précédent et le TP sur la mesure de température, écrivez un programme sous Processing qui permet d'afficher sur la fenêtre graphique du PC la température de la salle. Enregistrez le fichier sous *prog10_affichTemp.pde*. Testez.

Validation prof :

5. Tracer l'évolution de la température de la salle dans une interface graphique

On souhaite maintenant tracer l'évolution de la température de la salle sous la forme d'une courbe. On vous donne le programme incomplet.



✔ Ouvrez le fichier *prog11_courbeTemp.pde* disponible sur le réseau. Complétez le fichier en y ajoutant la lecture de la température dans la variable temp au début de la fonction draw(). Testez.

Validation prof :

✔ Ajoutez à l'interface graphique l'affichage des températures min et max relevées. Enregistrez le fichier sous *prog12_tempMinMax.pde*. Testez.

Validation prof :