Projet *Aladin* :

allumer une lampe en claquant des doigts

1. Scénario pédagogique

1. Dans le programme

Périphériques d'entrée et de sortie Interface Homme-Machine (IHM)	Identifier le rôle des capteurs et actionneurs. Réaliser par programmation une IHM répondant à un cahier des charges donné.	Les activités peuvent être développées sur des objets connectés, des systèmes embarqués ou robots.

2. Objectifs

- Création d'une lampe qui s'allume par déclenchement sonore, par une chaîne capteur-actionneur réalisée progressivement.

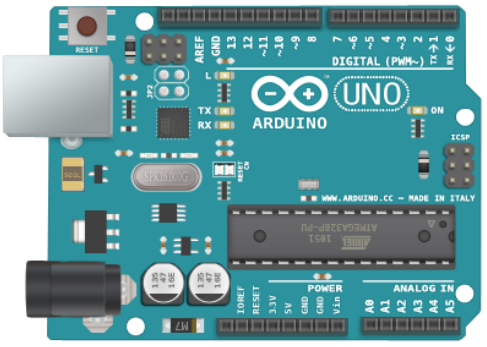
3. Modalités pédagogiques

a. **Prérequis**
Cette séquence est une séance d'ouverture générale à l'écosystème Arduino : elle ne requiert aucune expertise en câblage (grâce au shield Gravity) ni en programmation.
Les connaissances algorithmiques travaillées en Python pendant l'année sont réinvesties.

b. **Déroulement**
Deux séances de 2h, en demi-groupe.

c. **Matériel utilisé**
Cf le document materiel_PA.pdf

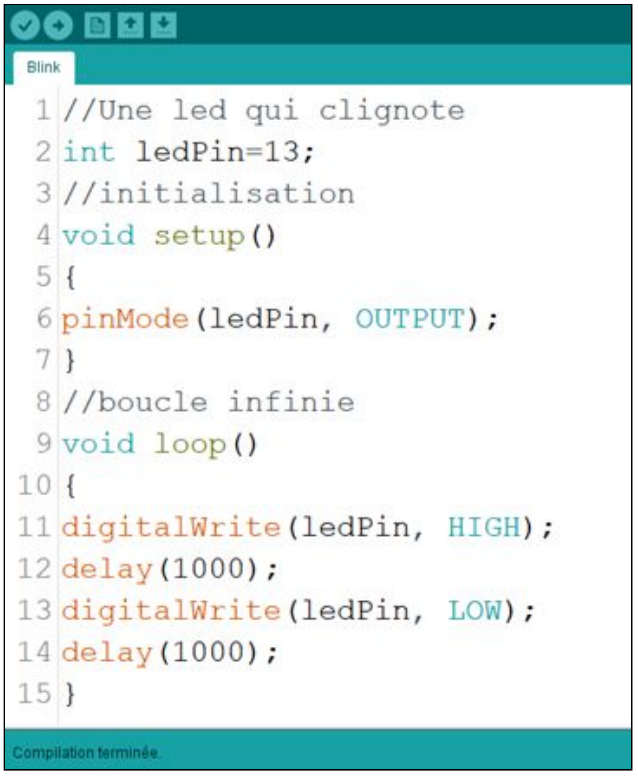
2. Présentation de la carte Arduino et de l'IDE



Voir ce document de découverte et d'approfondissement de la carte Arduino.

3. Tout premier programme : Blink

- Dans Fichier / Exemples / 01Basics, ouvrez le fichier Blink (le "Hello World" de l'Arduino !)



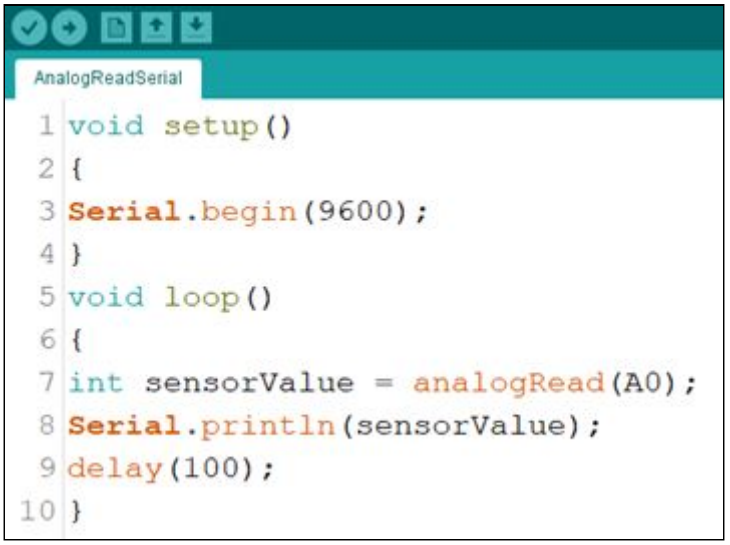
- Commentaires sur les différences apparentes entre Python et le langage Arduino (assimilable à du C / C++):
    - variables typées
    - pas d'indentation mais une ponctuation spécifique
    - programme compilé avant téléversement dans l'Arduino (l'Arduino n'a pas d'OS)

Questions :
pour répondre aux questions suivantes, il est conseillé de modifier légèrement le code (puis de re-compiler et re-téléverser) et/ou d'utiliser le site de référence d'Arduino (accessible depuis l'IDE par Aide / Référence)

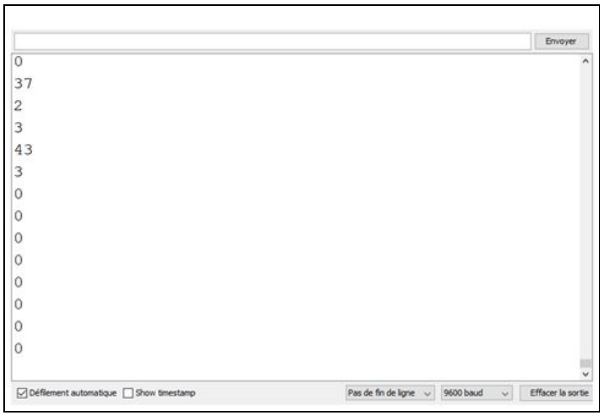
1. Quel est le rôle de la fonction **delay()** ? Quelle est l'unité du nombre passé en paramètres ?
2. Remplacez la valeur **HIGH** par le nombre 1 et observez ce qui change (ou pas) après compilation et téléversement du code. Procédez de même avec la valeur **LOW**.
3. Lorsque Arduino manipule des valeurs digitales, quelles sont les deux seules valeurs avec lesquelles il travaille ?

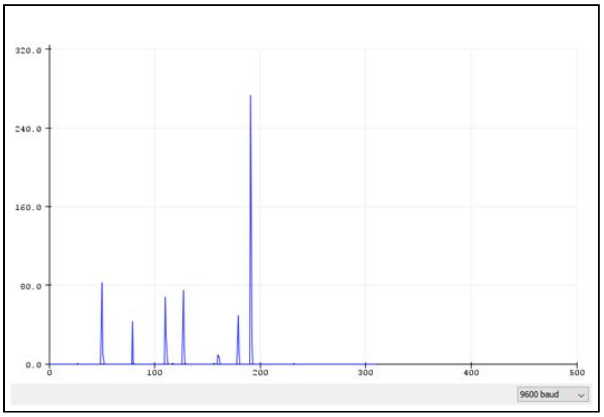
4. Premier capteur analogique

- Branchez un capteur sonore sur le pin analogique A0.
- Dans Fichier / Exemples / 01Basics, ouvrez le fichier AnalogReadSerial.



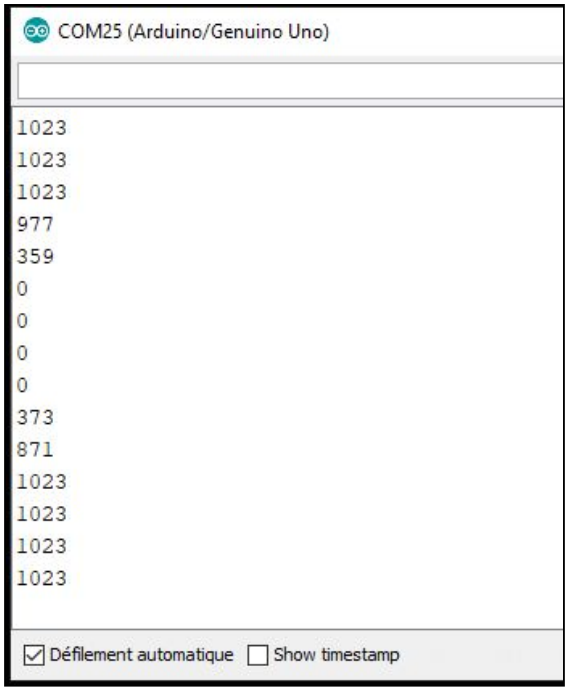
- Observez les valeurs affichées dans le Moniteur Série. Vous pouvez aussi aller dans Outils / Traceur Série

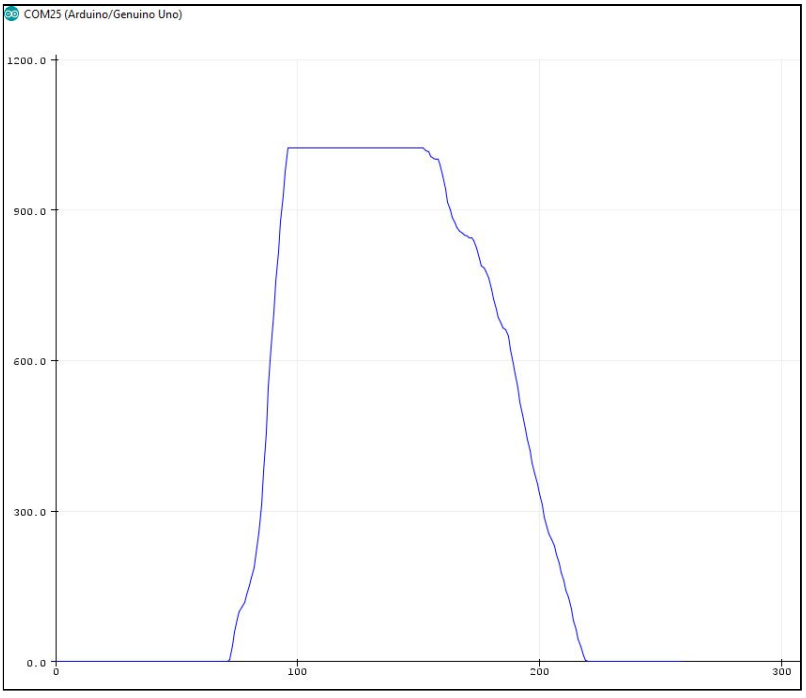




Questions :

1. Quelles sont les valeurs minimales et maximales que vous observez ?
2. La valeur de la grandeur physique captée par le capteur sonore est une valeur analogique. Elle est ensuite convertie en valeur numérique (celle affichée par le Moniteur Série) en utilisant un convertisseur 10 bits.
    a. Quel est en binaire le plus grand nombre que l'on peut écrire sur 10 bits ? Donner aussi sa valeur en notation décimale
    b. Comme constaté au 1., la plus petite valeur retournée par l'Arduino sur une entrée analogique est 0. D'après la question précédente, quelle est alors la plus grande valeur possible ?
3. Branchez un potentiomètre à la place de votre capteur sonore, et visualisez la valeur maximale qui peut être atteinte.





5. À vous ! Une Led qui se déclenche en fonction du bruit ambiant

- En combinant les deux programmes étudiés, créez un code permettant d'allumer la LED en claquant des mains à proximité du capteur sonore. La LED doit s'éteindre automatiquement 5 secondes plus tard.
  (n'oubliez pas d'utiliser la page de Référence pour avoir de l'aide sur la syntaxe…)

  Correction :

AnalogReadSerialson

```
1  //Allumer une led en tapant des mains.La led s'éteint au bout de 5s
2  int seuil=100;
3  int ledPin=13;
4
5  void setup()
6  {
7  pinMode(ledPin, OUTPUT);
8  Serial.begin(9600);
9  digitalWrite(13, LOW);
10 }
11
12 void loop()
13 {
14 int sensorValue = analogRead(A0);
15 Serial.println(sensorValue);
16 if (sensorValue>seuil)
17 {
18 digitalWrite(ledPin, HIGH);
19 delay(5000);
20 digitalWrite(ledPin, LOW);
21 }
22 }
```

5. Allumer… mais aussi éteindre

Pour l'instant, la LED s'éteint automatiquement au bout de 5 secondes. Il serait plus intéressant qu'elle s'éteigne au claquement de mains suivant…

Nous allons pour cela utiliser une variable booléenne : c'est une variable qui ne peut prendre que 2 valeurs : `true` ou `false`.

Il est possible d'inverser la valeur contenue par la variable booléenne `etat` par l'expression::

```
etat = !etat
```

Ainsi, la variable `etat` passe de `true` à `false` ou bien de `false` à `true`.

(il est possible d'écrire aussi comme en python `valeur = not(valeur)`)

- En utilisant une variable booléenne, modifiez le programme précédent afin que la lampe s'allume ou s'éteigne par un claquement de mains.

Correction :

bruit_on_off§

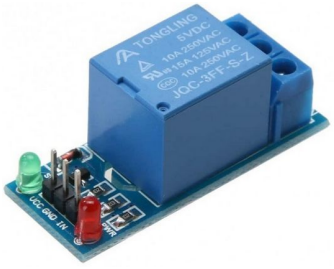
```
//On allume la led en tapant des mains.On éteint la led en tapant des mains.
boolean etat=false;
int ledPin=13;
int seuil=100;
void setup()
{
pinMode(ledPin, OUTPUT);
Serial.begin(9600);
digitalWrite(13, LOW);
}
void loop()
{
int sensorValue = analogRead(A0);
Serial.println(sensorValue);
if (sensorValue> 100)
  {
  if(etat==false)
    {
    digitalWrite(13, LOW);
    delay(100);
    }
  else
    {digitalWrite(13, HIGH);
    delay(100);
    }
  etat=!etat;
  }
}
```

5. Utilisation d'un relais

Notre dispositif permet pour l'instant d'allumer ou d'éteindre une LED alimentée par notre Arduino.

Afin de commander en toute sécurité un objet électrique domestique (une lampe, une radio, un volet roulant…) il est nécessaire de faire appel à un dispositif qui va faire le lien entre le circuit électrique de l'Arduino (5V) et le circuit électrique du dispositif à commander (220V par exemple).

Ce dispositif s'appelle un relais :



Ce relais sera branchée sur la pin 13. Lorsque celle-ci sera passée à HIGH, le relais va s'activer (on entend distinctement un "clac") afin de fermer le circuit électrique de notre lampe de bureau.

6. Comment rendre le dispositif autonome

Pour l'instant, notre dispositif est encore relié par un câble USB à l'ordinateur sur lequel nous avons tapé notre code.

Quelle rôle joue ce câble USB dans le fonctionnement de notre dispositif ?

7. Arduino dans un plus large écosystème numérique

7.1 Utilisation du moniteur Série.

Rebranchons l'Arduino à notre ordinateur : dans quelle mesure un programme externe (non Arduino) peut-il interagir avec l'Arduino et donc commander notre lampe ?
Découvrons tout d'abord un exemple avec Python.

Le programme Arduino `lampe_on_off_blink.ino` ci-dessous permet à Arduino d'«écouter» les caractères qui lui sont transmis sur son port Série.

Question :
Analysez ce programme et déterminez comment réagit le programme lorsque l'utilisateur tape successivement sur les touches "A", "E", puis "B".

7.2 Communication avec un programme en Python

Le programme Python `lampe.py` ci-dessous permet envoyer via le port Série de l'Arduino les lettres tapées au clavier par un utilisateur.

Question :
Analysez ce programme et déterminez comment réagit le programme lorsque l'utilisateur tape successivement sur les touches "A", puis "Q".

7.2 Communication avec un smartphone via Bluetooth

L'utilisation d'un shield Bluetooth sur l'Arduino permet de recevoir des informations sur le port série depuis n'importe quel dispositif Bluetooth.
On peut par exemple utiliser l'application Serial Bluetooth Terminal sur Android.

lampe-on-off-blink

```
boolean etat=false;
char data_temp='E';
char data='E';
int ledPin=13;

void blink()
{
digitalWrite(ledPin, HIGH);
delay(1000);
digitalWrite(ledPin, LOW);
delay(1000);
}

void readSerialData()
{
if(Serial.available())
  {
  switch(Serial.read())
    {
    case 'A': data_temp='A';break;
    case 'E': data_temp='E';break;
    case 'B': data_temp='B';break;
    default:break;
    }
  }
}

void changeState()
{
if (data_temp!=data)
  {
   data=data_temp;
  }
switch(data)
  {
  case 'A': {digitalWrite(ledPin, HIGH);delay(200);break;}
  case 'E': {digitalWrite(ledPin, LOW);delay(200);break;}
  case 'B': {blink();break;}
  default:break;
  }
}

void setup()
{
pinMode(ledPin, OUTPUT);
Serial.begin(9600);
}

void loop()
{
readSerialData() ;
changeState();
}
```

lampe.py - C:\Users\ybou\Desktop\lampe.py (3.6.6)

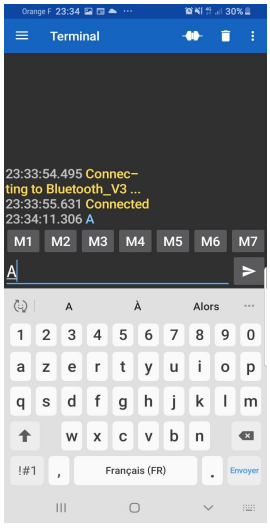
File Edit Format Run Options Window Help

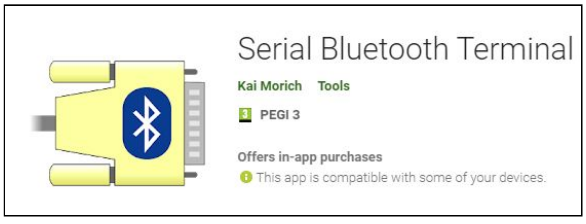
```
import serial
import time
ser = serial.Serial('COM5', 9600, timeout = 1)
time.sleep(1)
if ser.is_open:
    print("communication")
continuer=True

while continuer:
    commandToSend=input()
    if commandToSend=='Q':
        ser.write('E'.encode())
        continuer=False
    else:
        ser.write(commandToSend.encode('utf-8'))

ser.close()
```

Python 3.6.6 Shell

File Edit Shell Debug Options Window Help

```
Python 3.6.6 (v3.6.6:4cf1f54eb7, Jun 27 2018, 02:47:15) [MSC v.1900 32 bit (Inte
l)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\ybou\Desktop\lampe.py ==========
communication
A
E
B
Q
>>>
```

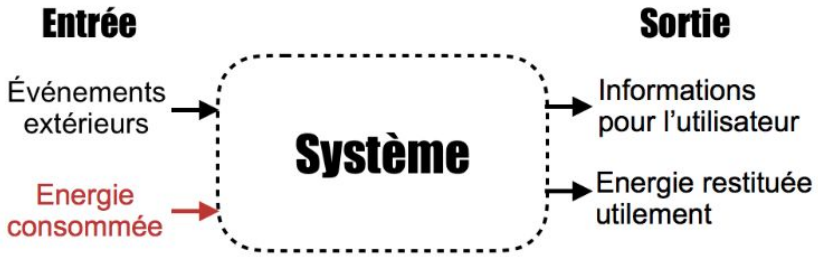




8. Synthèse

Système informatique embarqués

Les **systèmes embarqués** sont des systèmes électroniques et informatiques autonomes, souvent temps réel, spécialisés dans une **tâche** bien précise capable d'acquérir une donnée, la traiter et la communiquer. Le terme désigne aussi bien le matériel (Hardware) que le logiciel (Software) utilisé.



L'information provient soit des **IHM** soit des **capteurs**, pour contrôler automatiquement ou manuellement le fonctionnement physique par des **actionneurs** et transmettre des informations aux utilisateurs.

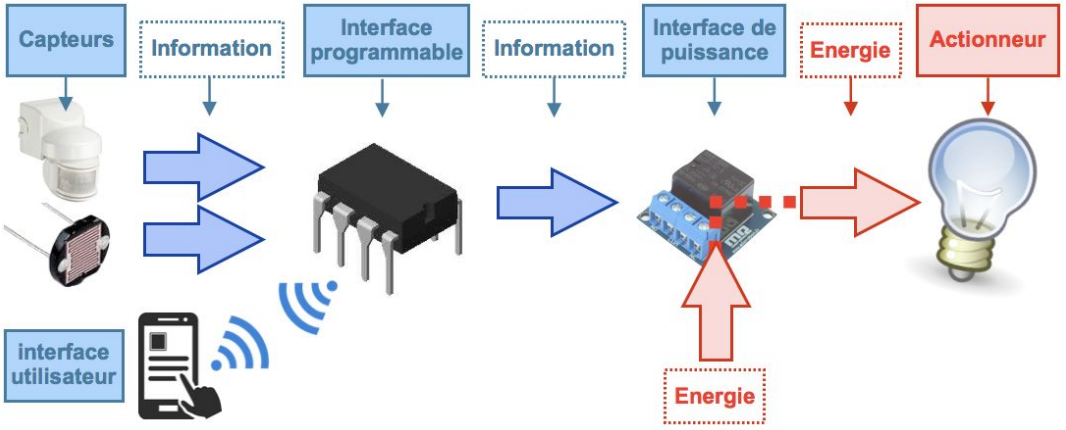
Interface Homme-Machine

Une **Interface Homme-Machine** (IHM) est une interface utilisateur permettant de connecter une personne à une machine, à un système ou à un appareil. Le flux d'informations à travers les IHM permet une **interaction continue** entre l'homme et la machine.

Les IHM peuvent prendre différentes formes : écrans directement intégrés aux machines, écrans d'ordinateur, tablettes tactiles ou smartphones, …

Structure d'un Système embarqués

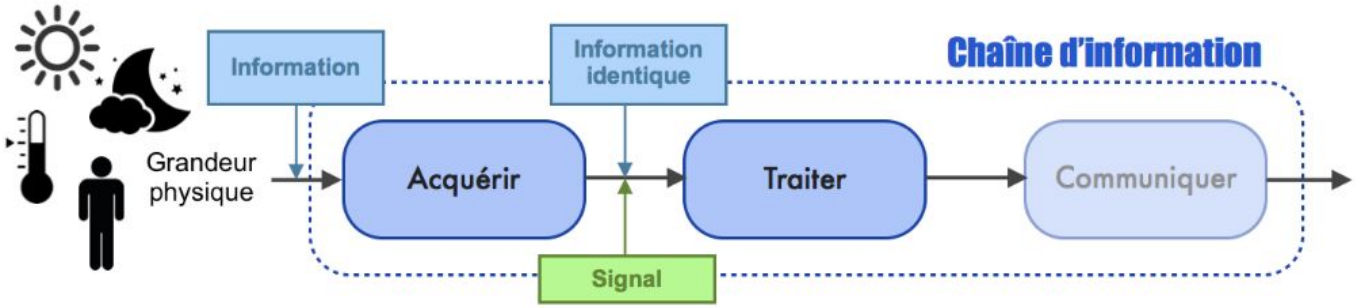
Les capteurs permettent d'**acquérir des informations** qui sont traitées par une interface programmable pour **piloter des actionneurs**. Souvent, il faut utiliser une **interface de puissance** pour distribuer l'énergie vers l'actionneur.

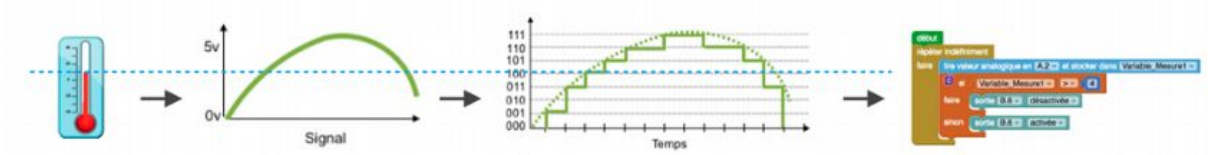


Il est aussi possible d'envoyer des informations directement depuis des interfaces utilisateur (ordinateur, appareil nomade, ...) afin de modifier en temps réel le fonctionnement du système embarqué.

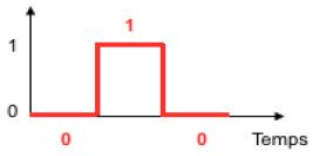
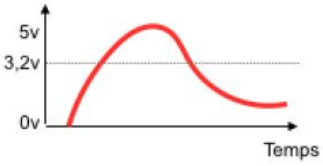
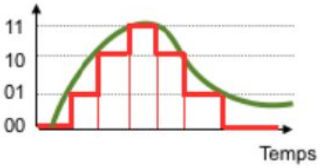
Capteurs

Un capteur permet d'acquérir une grandeur physique pour la transformer en grandeur « utilisable » par un système.





Cette information logique ou analogique acquise sera portée par un signal analogique ou numérique.

Information logique	**Information Analogique**	
2 informations : *Appuyé / Non appuyé* *Barrière IR coupée / non coupée* *Présence / Pas de présence*	plusieurs informations : *Quantité de lumière* *Direction et angle de déplacement* *Températures*	
Signal Numérique	**Signal Analogique**	**Signal Numérique**
		

Actionneurs

Les Actionneurs permettent de transformer l'énergie reçue en un phénomène physique (déplacement, dégagement de chaleur, émission de lumière …) pour répondre à un besoin donné.

Exemple : un moteur, une lampe, une DEL, un buzzer, …