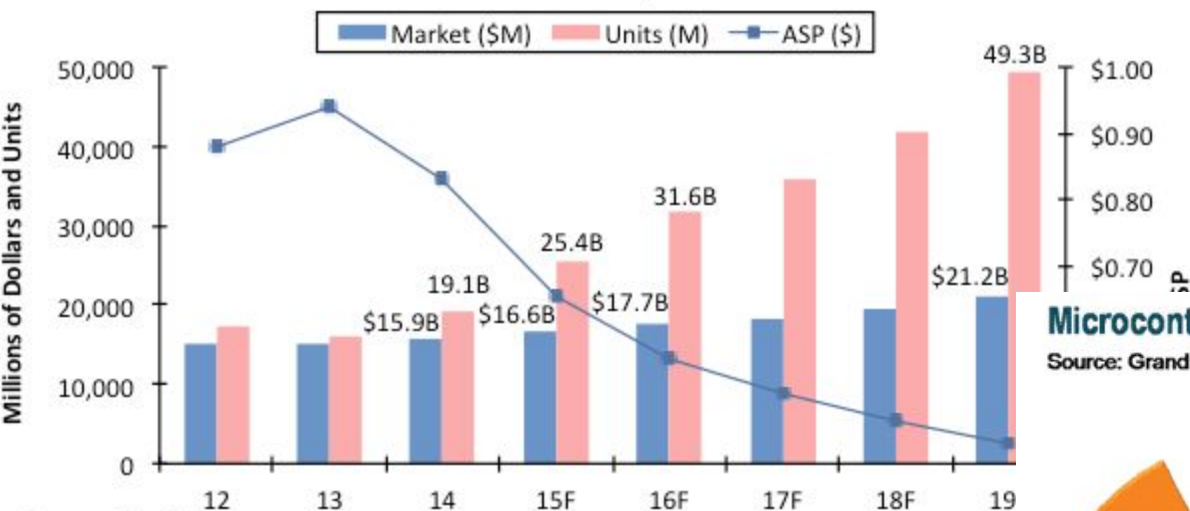


Informatique S1

découverte du langage c/c++ avec arduino

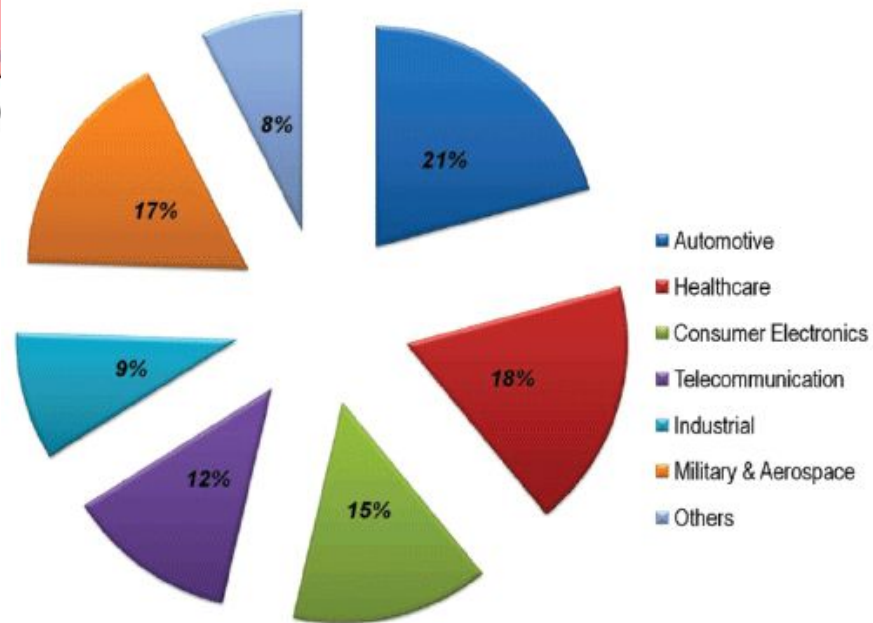
MCU Market History and Forecast



Source: IC Insights

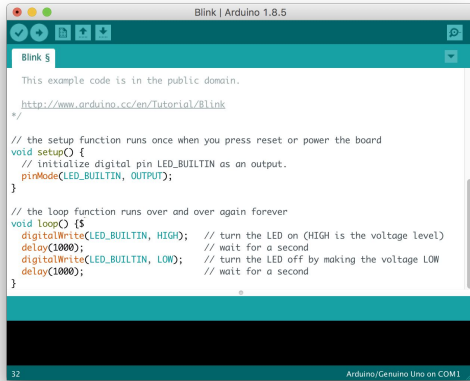
Microcontroller System market share by application, 2013

Source: Grand View Research, Inc



Arduino ?

Logiciel/EDI (Software / IDE)



```
Blink

This example code is in the public domain.

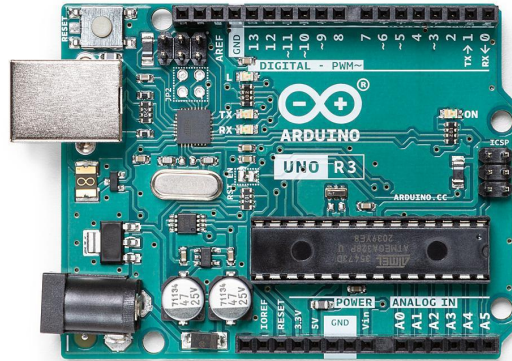
* http://www.arduino.cc/en/Tutorial/Blink */

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}

32 Arduino/Genuino Uno on COM1
```

Matériel (Hardware)



Fonctions (Language)

LANGUAGE

FUNCTIONS

VARIABLES

STRUCTURE

LIBRARIES

IoT CLOUD API

GLOSSARY

Arduino Reference text is licensed under a Creative Commons Attribution-ShareAlike license. Anything that can be improved, suggest corrections via documentation via

on how to use Github? everything you need to read this tutorial.

Language Reference

Arduino programming language can be divided in three main parts: functions (variables and constants), and structure.

Functions

For controlling the Arduino board and performing computations.

Digital I/O

digitalRead()
digitalWrite()
pinMode()

Analog I/O

analogRead()
analogReference()
analogWrite()

Zero, Due & MKR Family

analogReadResolution()
analogWriteResolution()

Advanced I/O

noTone()

Characters

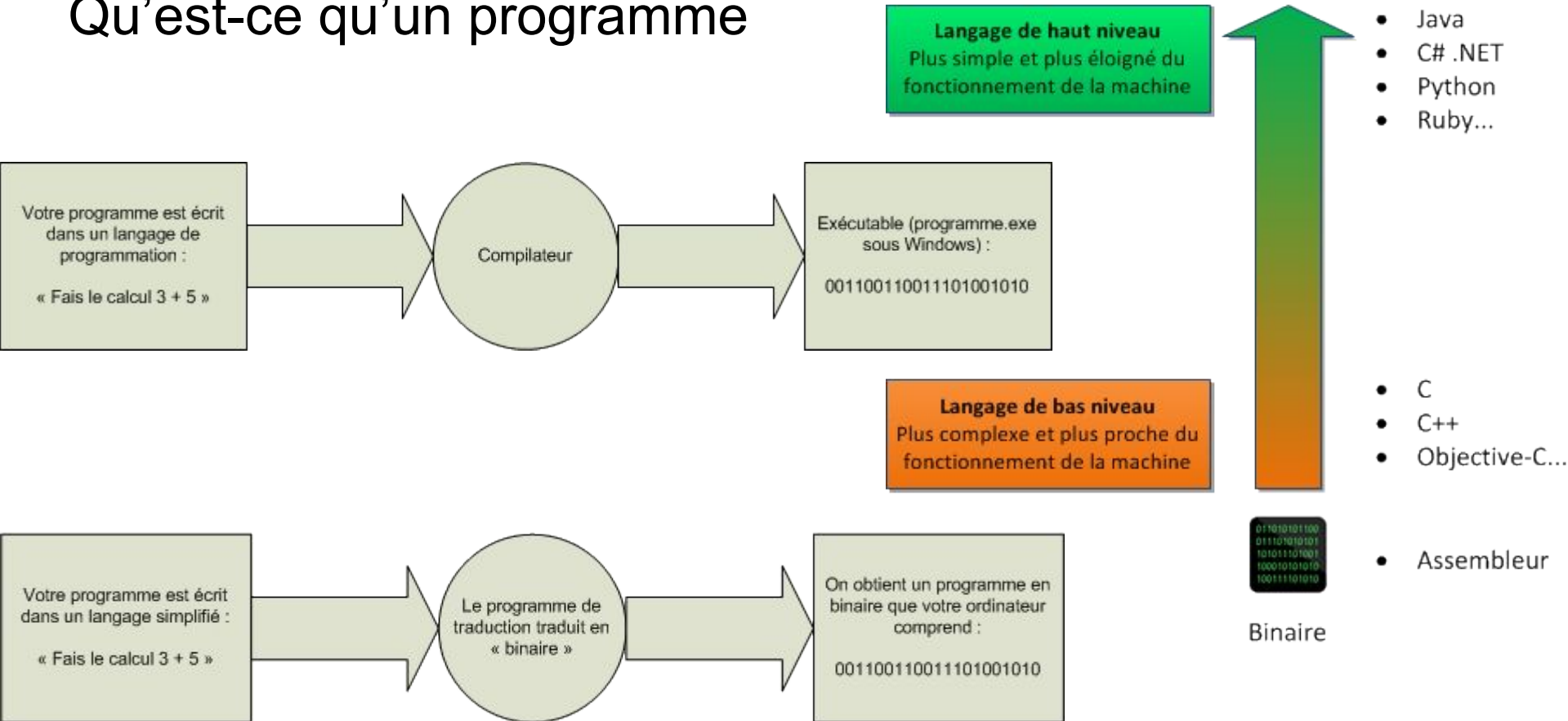
isAlpha()
isAlphanumeric()
isAscii()
isControl()
isDigit()
isGraph()
isHexadecimalDigit()
isLowerCase()
isPrintable()
isPunct()
isSpace()
isUpperCase()
isWhitespace()

Random Numbers

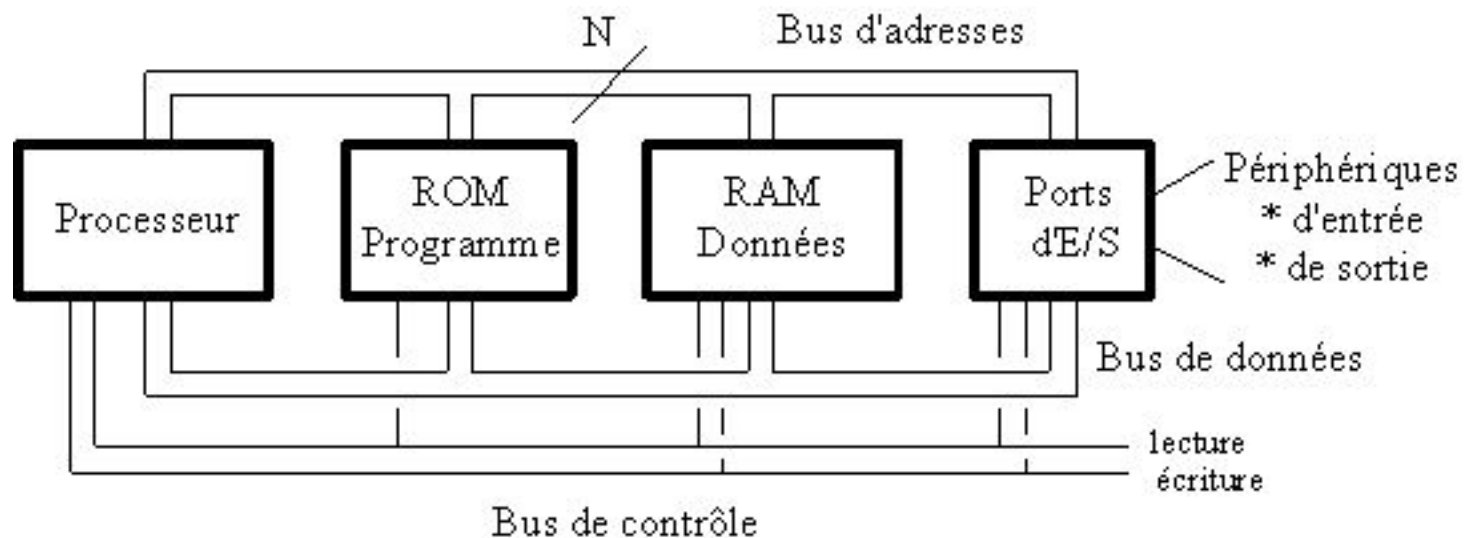
random()
randomSeed()



Qu'est-ce qu'un programme



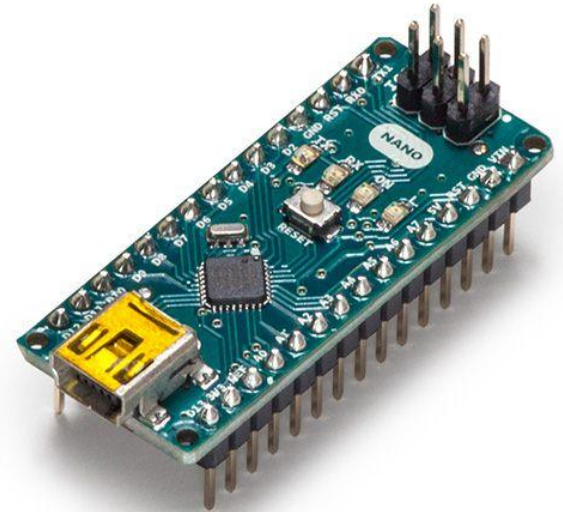
Architecture d'un microcontrôleur



La carte Arduino Nano

Microcontrôleur Atmel ATMEGA328 8 bits

- 2Ko SRAM, 1Ko EEPROM, 32Ko Flash
- Quartz 16 Mhz
- Port micro USB
- 14 lignes E/S numériques (GPIO)
 - ◆ dont 6 sorties analogiques (PWM)
 - ◆ dont 8 entrées analogiques

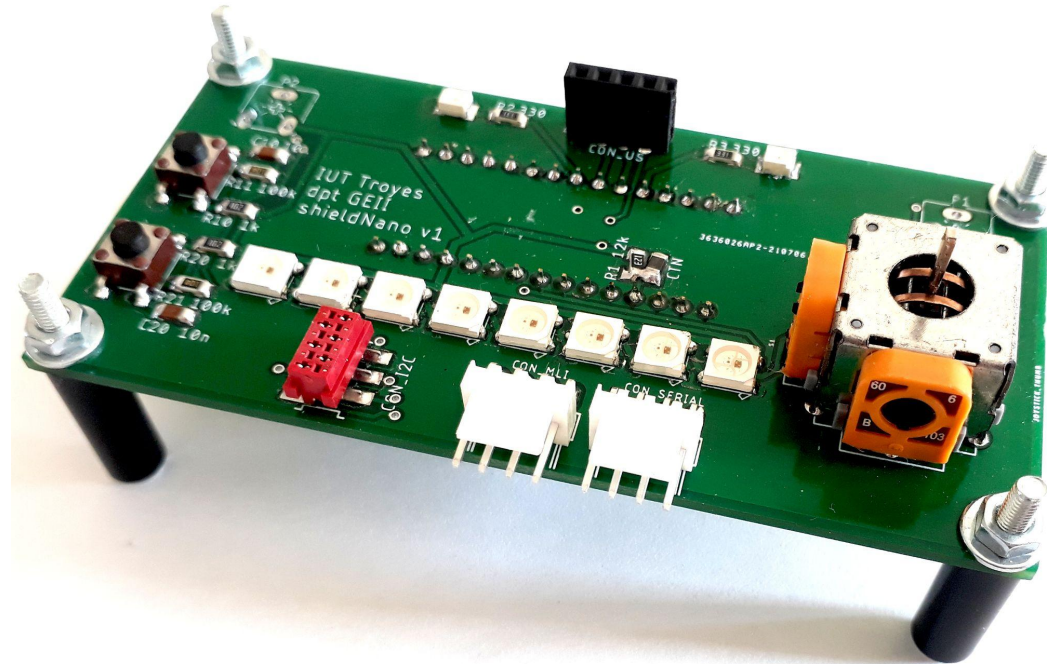


Shield Arduino

→ carte d'extension

→ pour la nôtre :

- ◆ 2 bps
- ◆ 2 leds
- ◆ 8 leds "numériques" 3 couleurs
- ◆ 4 connecteurs d'extension
 - afficheur
 - moteurs
 - ...
- ◆ capteur de température
- ◆ joystick ou potentiomètres



Objectif : saé après la toussaint

- étude et fabrication d'un shield pour carte arduino Nano
- conception du programme
- thème : multimètre
 - ◆ voltmètre
 - ◆ ohmmètre
 - ◆ affichage de la valeur
 - ◆ évolution possible vers mesure de capacité et inductance

Structure d'un programme arduino

```
#include <stdio.h>

int main()
{
    setup();
    while(1)
    {
        loop();
    }
}

void setup()
{
}

void loop()
{
}
```

Search... 🔍

Brick

Sensors

Motors

Music

Loops

Logic

Variables

Math

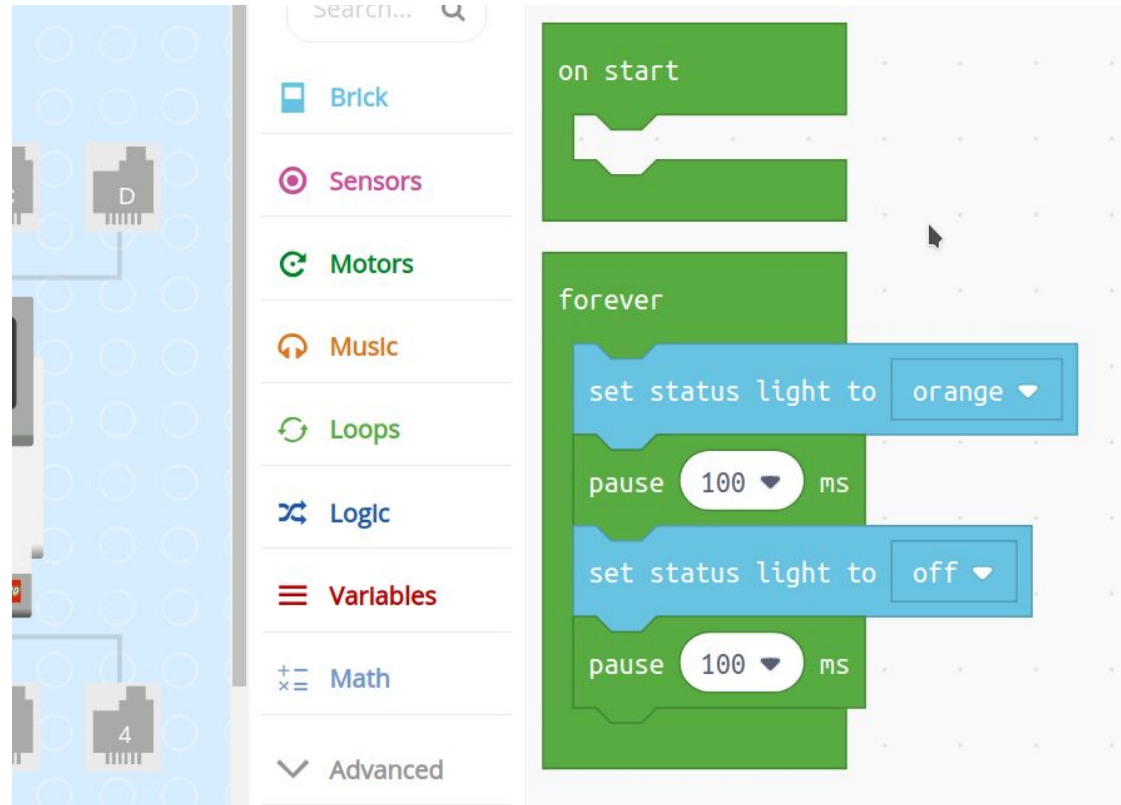
Advanced

on start

forever

Structure d'un programme arduino

```
void setup()  
{  
  pinMode(3, OUTPUT);  
}  
  
void loop()  
{  
  digitalWrite(3, 1);  
  delay(100);  
  digitalWrite(3, 0);  
  delay(100);  
}
```



The image displays the Arduino IDE interface. On the left, a wiring diagram shows two digital pins connected to LEDs. The top pin is labeled 'D' and the bottom pin is labeled '4'. In the center, a search bar is visible above a vertical list of block categories: Brick, Sensors, Motors, Music, Loops, Logic, Variables, Math, and Advanced. On the right, a Scratch-style block-based program is shown. It starts with an 'on start' block, followed by a 'forever' loop. Inside the loop, the sequence of blocks is: 'set status light to orange', 'pause 100 ms', 'set status light to off', and 'pause 100 ms'.

Manipulation des broches (Pins / GPIO)

- les broches sont configurables en entrée (Input) ou sortie (OUTPUT)
 - `pinMode(2, OUTPUT); // instruction pour déclarer la broche 2 en sortie`
 - `pinMode(3, INPUT); // instruction pour déclarer la broche 3 en entrée`
- Pour modifier l'état d'une sortie binaire on utilise la fonction `digitalWrite`
 - `digitalWrite(2, 1); // positionne la sortie à l'état 1 (tension de 5V selon l'alimentation)`
 - `digitalWrite(2, 0); // positionne la sortie à l'état 0 (tension de 0V)`
- Pour observer l'état d'une entrée binaire on utilise la fonction `digitalRead`
 - on déclare une variable : `int valEntree;`
 - on lit la valeur : `valEntree=digitalRead(3); // valEntree prendra la valeur 0 ou 1`

Variables

- Il est nécessaire de déclarer les variables avant de pouvoir les utiliser
- On doit faire un choix sur le type de variable
- quelques types pour démarrer :

type :	char	int	unsigned int	float
description :	les caractères	nombres entiers [-32 767 ; +32 767]	nombres entiers [0 ; 65535]	type flottant, nombre réel
exemple :	char c; c = 'a' ;	int i; int j; i = 5 * -3 ; j = i + 3 ;	unsigned int x ; x = 3 * 2 ;	float a ; float b = -0.2 ; a = 5.1 * b ;

Structures de contrôle : actions conditionnelles

```
int etatEntree;  
void setup()  
{  
  pinMode(3,OUTPUT);  
  pinMode(2,INPUT);  
}  
void loop()  
{  
  etatEntree = digitalRead(2);  
  if (etatEntree==0)  
  {  
    digitalWrite(3,1);  
  }  
  else  
  {  
    digitalWrite(3,0);  
  }  
}
```

Search... 🔍

- Brick
- Sensors**
- Motors
- Music
- Loops
- Logic
- Variables
- Math
- Advanced

The diagram shows a Scratch-style block structure for a color sensor. It starts with an 'on start' block containing a 'wait' block. This is followed by a 'forever' loop block. Inside the loop, there is an 'if' block with the condition 'is color sensor 3 detected'. The 'then' branch of the 'if' block contains a 'set status light to orange' block. The 'else' branch contains a 'set status light to off' block. The 'if' block has a '+' sign on its left side, indicating it is expanded.

Structures de contrôle : actions conditionnelles

```
if (condition1 )
{
    action1;
}
else if ( condition2 )
{
    action2;
}
else if ( condition3 )
{
    action3;
}
else
{
    actionParDefaut;
}
```

Opérateur	Signification	Exemple
==	égalité	if (a == 3) if (c=='a')
!=	inégalité	if (a != b)
<	Strictement inférieur à	if (d < 10)
<=	inférieur ou égal à	if (d <= -5)
>	Strictement supérieur à	if (a > d)
>=	Supérieur ou égal à	if (b >= -3)
!	négation	if (!(a==5))

Opérateur logique	Signification	Exemple
&&	ET	if ((a==3) && (b==5))
	OU	if ((a>2) (b < 5))

Autres fonctions arduino

- fonctions pour les tensions “analogiques”
 - ◆ `analogRead`
 - ◆ `analogWrite`
- affichage de messages sur le pc : liaison série
 - ◆ `Serial.begin`
 - ◆ `Serial.print`
 - ◆ `Serial.println`
- programmation événementielle : interruption
 - ◆ `attachInterrupt`

Langage c/c++ - quelques règles de syntaxe

- Attention à la casse (majuscule / minuscule)
- camelCase convention (ex variable : valeurDeMonEntree)
- Les instructions se terminent par un point-virgule
- Indentation du code (alignement du code par des tabulations)

```
int main()  
{  
    int variable;    // variable de type int  
    int Variable;   // autre variable de type int  
    float VARIABLE; // une autre de type float  
    return 0;  
}
```

```
int main()  
{  
    // commentaire  
    int x,y;  
    for (x=0;x<10;x++)  
    {  
        y=2*x;  
        ...  
    }  
    return 0;  
}
```