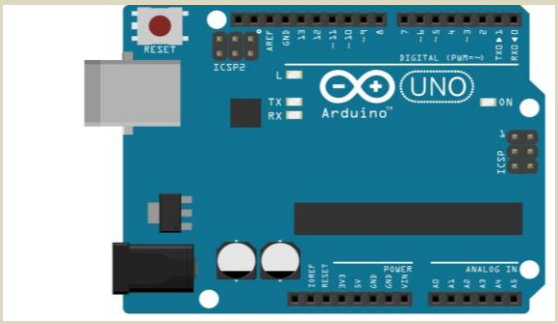
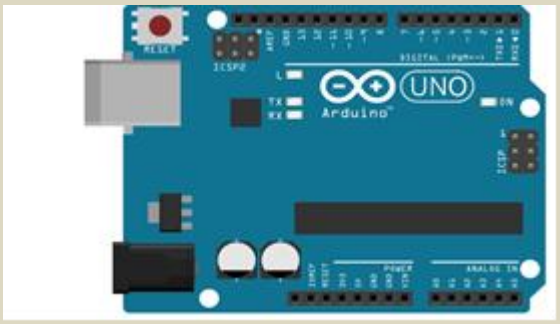
Mon Arduino veut communiquer en série

Présentation du matériel.





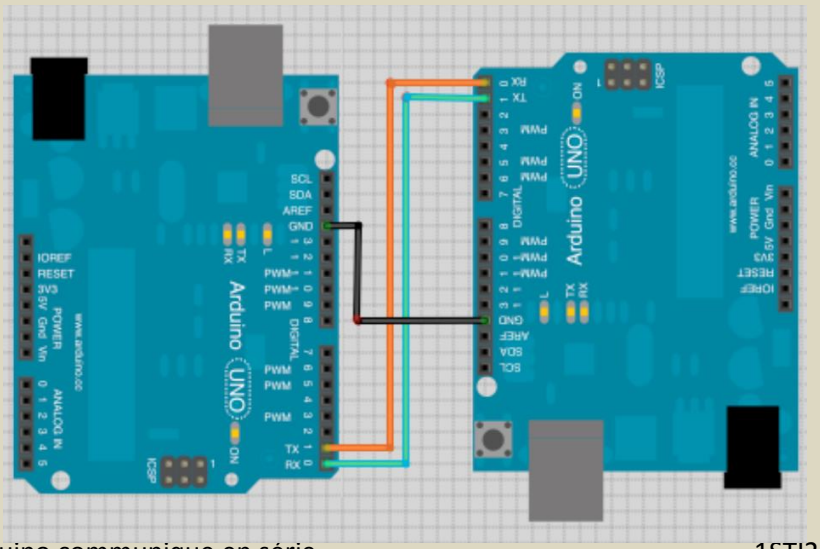
I – Généralités

- La liaison série entre la carte Arduino et l'ordinateur est établie à travers le port USB.

En fait, se port USB n'est pas utilisé avec le protocole USB, mais avec celui de la liaison série RS232.

Ceci est donc géré par la carte Arduino et il n'y a rien à paramétrer.

- Pour relier deux cartes Arduino en liaison série, il suffit de connecter les broches Tx et Rx ensemble, de cette manière :



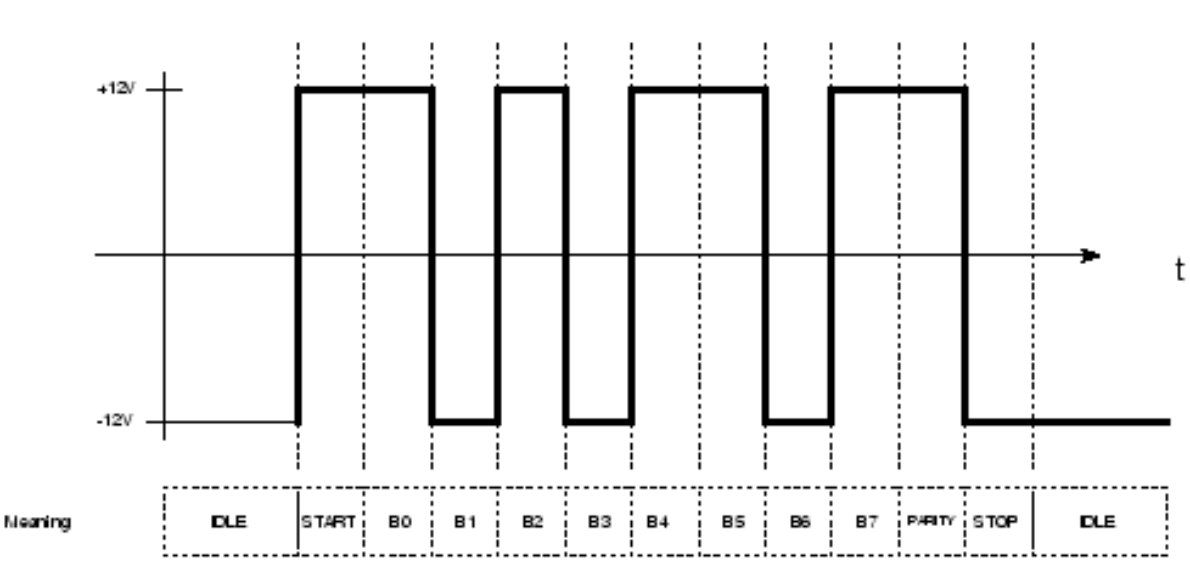
✓ Donner le rôle des broches Rx et Tx dans une liaison série ?

- l'ordinateur utilise des niveaux de -12V à +12V (de manière habituelle, mais ils sont en réalité entre -3/-24V et +3/+24V). Les niveaux "positifs" représentent un état bas (un '0' logique), alors qu'un niveau haut (le '1' logique) est représenté par les tensions négatives.

- L'arduino utilise les niveaux de tensions 0V et 5V (niveau TTL)

- La carte Arduino, plutôt que d'être branché sur un port série classique sera donc branché sur l'USB. Les niveaux seront donc toujours du 5V maximum. Ensuite, un composant intégré à Arduino se chargera de simuler une voie série et tout devient transparent pour votre ordinateur. Il vous suffit donc juste d'utiliser le câble USB et de le relier.

✓ Soit l'image ci-dessous, donnez le code transmit en binaire.



✓ Retrouver le code ASCII ainsi que le caractère envoyé.

✓ Expliquer le rôle du bit de parité.

II – Notion de classe

Dans les langages de programmation actuels, les concepteurs ont imaginé la possibilité de pouvoir rassembler des fonctions ensemble lorsqu'elles s'appliquent à une même fonctionnalité.
Tout comme une fonction, une classe aura un nom et pour distinguer une classe d'une fonction, le nom d'une classe commencera par une MAJUSCULE.
Le langage Arduino ou ses librairies comporte plusieurs classes et il faut comprendre ce concept :

- Toutes les fonctions qui gèrent la communication avec le port série USB sont rassemblées dans une classe appelée Serial,
- La classe LiquidCrystal pour la gestion d'un afficheur LCD,
- La classe Servo pour la gestion d'un servomoteur,
- Etc…

En pratique, pour utiliser une fonction d'une classe du langage Arduino, on utilisera le nom de la classe + un point + le nom de la fonction.

NomClasse.fonction1()

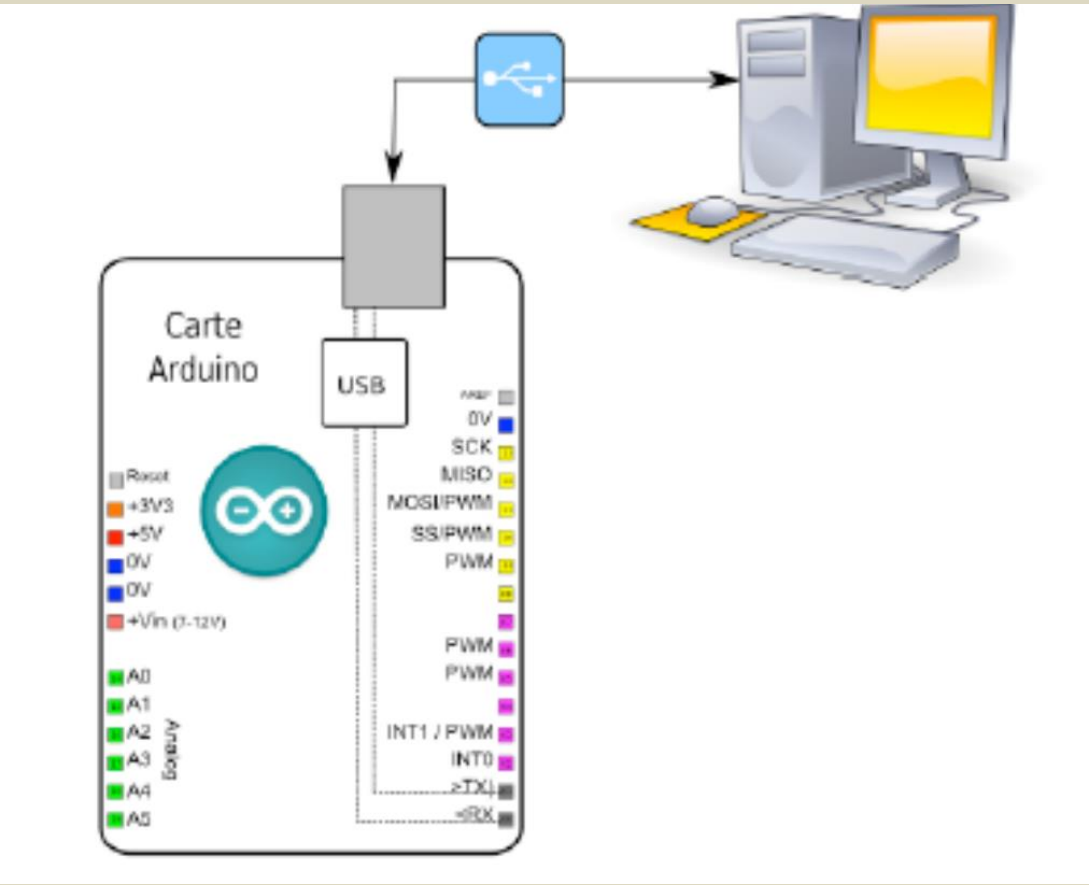
La classe Serial
Les fonctions de la classe Serial sont a nombre d'une dizaine.
Nous allons découvrir les 4 fonctions qui permettent d'écrire un programme pour afficher
des messages vers le PC.

- begin() : fonction d'initialisation de la communication USB
- print() : fonction d'affichage d'un message sans saut de ligne
- println() : fonction d'affichage d'un message avec saut de ligne
- read() : permet de lire et de renvoyer le premier caractère présent dans le port série
- available() : renvoie le nombre d'octets présents en réception sur le port série (true si caractère présent et false sinon…)
- flush() : vide la file d'attente en réception du port série.

Dans le cas de la classe Serial, on fera :

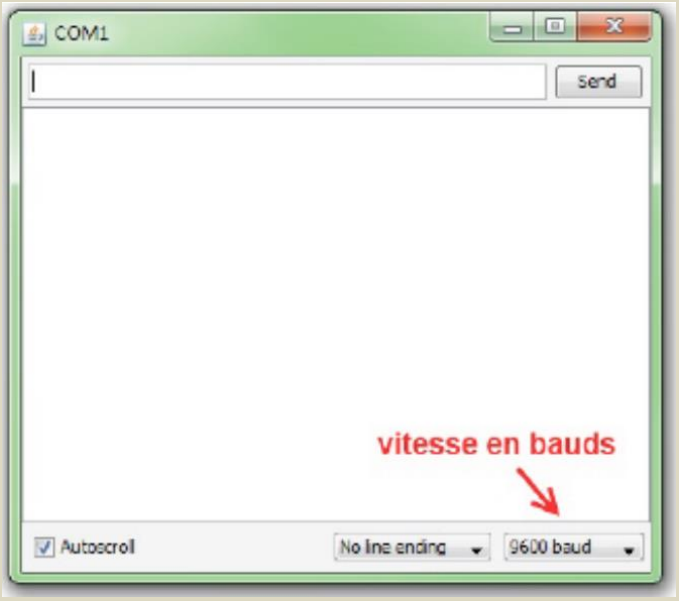
- Serial.begin() pour appeler la fonction begin(),
- Serial.print() pour appeler la fonction print(),
- Serial.println() pour appeler la fonction println(),
- Serial.read() pour appeler la fonction read(),
- Serial.available() pour appeler la fonction available()

III– Envoyer et recevoir des données.



Pour pouvoir utiliser la communication de l'ordinateur, l'environnement de développement Arduino propose de base un outil pour communiquer. Pour cela, il suffit de cliquer sur le bouton dans la barre de menu pour démarrer l'outil.

Une nouvelle fenêtre s'ouvre, c'est le terminal série :



Dans cette fenêtre, vous allez pouvoir envoyer des messages sur la liaison série de votre ordinateur (qui est émulée par l'Arduino) ; recevoir les messages que votre Arduino vous envoie ; et régler des paramètres tels que la vitesse de communication avec l'Arduino et l'autoscroll qui fait défiler le texte automatiquement.

TP1 : Programme bonjour

Ecrire un programme que vous nommerez bonjour qui va :

- Initialiser la communication à 9600 bauds,
- Afficher le message « b » dans le terminal toutes les 2 secondes , avec retour à la ligne.
- Avec un oscilloscope recupérer le code ascii de b. expliquer votre démarche.
- Afficher le message « bonjour » dans le terminal toutes les 2 secondes , avec retour à la ligne.

TP2 : Programme caractère_reçu

Ecrire un programme que vous nommerez caractère_reçu dont la structure est la suivante :

- Entête déclarative

- On déclare une variable int pour stocker en réception (code ASCII)
- On déclare une variable char pour stocker le caractère correspondant

- Fonction setup()

- on initialise la communication série avec une vitesse de 9600 bauds.

- Fonction loop()

- On écoute le port série en testant l'arrivée d'un caractère
- Si un octet est reçu, on affiche successivement :
  - Sa valeur numérique (code ASCII du caractère reçu)
  - Puis le caractère correspondant.

TP3 : Programme com_leds

Ecrire le programme com_leds qui permet de piloter 3 LEDs sur la carte Arduino, à partir de certaines touches du clavier du PC. Nous utiliserons le moniteur série pour activer les lumières.

- Principe de fonctionnement

Des caractères sont saisis à partir du moniteur série de l'interface de programmation Arduino pour allumer et éteindre les LED.

- Saisir 'R ou r' pour allumer la LED rouge,
- Saisir 'J ou j' pour allumer la LED jaune,
- Saisir 'V ou v' pour allumer la LED verte,
- Saisir 'E ou e' pour éteindre les LED.

TP5 : Programme poste_secours

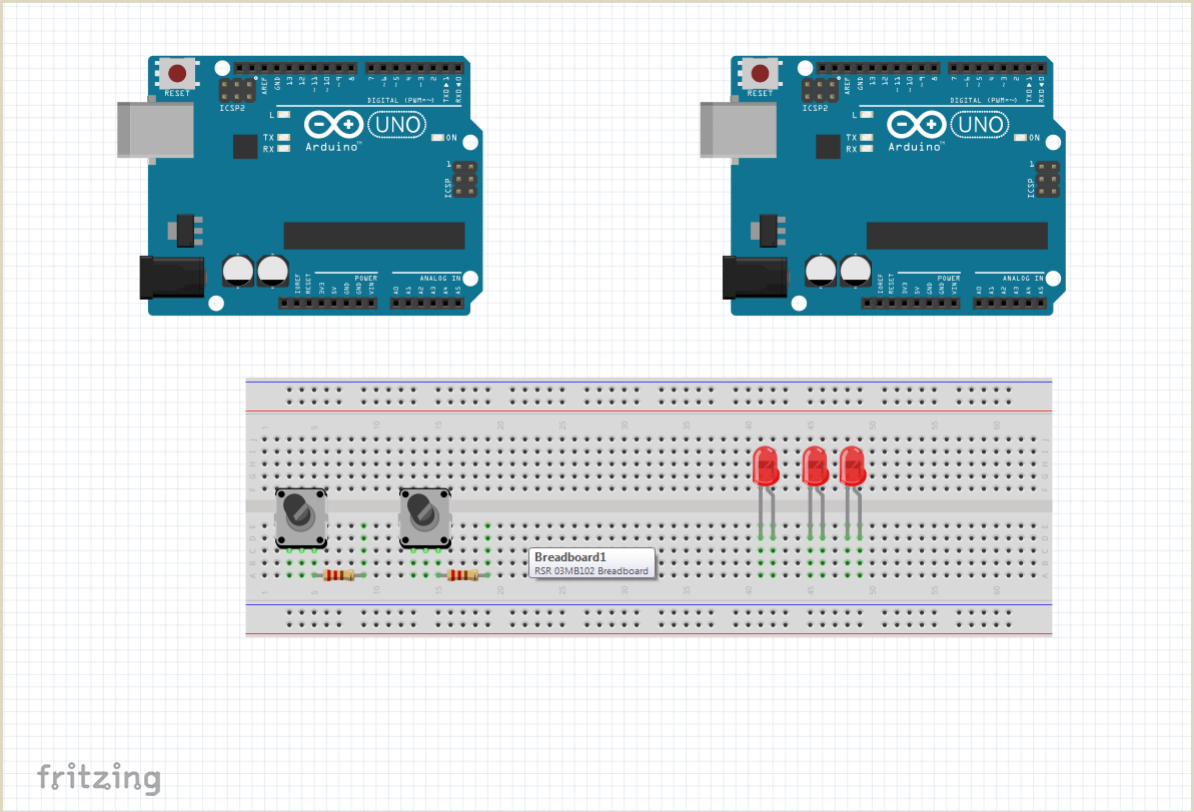
L'objectif du Tp est d'afficher un indicateur de qualité de baignade à la plage, en temps réel.
Vous avez à disposition 2 cartes arduinos , Une émetteur avec les boutons poussoirs, l'autre récepteur avec les leds, 3 Leds de trois couleurs différentes :

- LED1 (pin 13) = Rouge : pour indiquer un danger et l'interdiction de se baigner (animaux dangereux, météo…)
- LED2 (pin 12) = Orange : pour indiquer un risque pour la baignade (trop de vagues…)
- LED3 (pin 11) = Verte : pour indiquer aucun risque de baignade.

Vous utiliserez aussi 2 boutons poussoirs :

- BP1 (pin 2) pour déclencher une alarme indiquant un danger et allumer le Led rouge,
- BP2 (pin 3) pour revenir à la situation précédente après dissipation du danger.

Compléter le câblage du schéma électrique ci-dessous : (vous pouvez utiliser le logiciel Fritzing)



Proposer votre solution de programme poste_secours pour remplir le cahier des charges.

Conseils : aidez de l'algorithme suivant si vous le souhaitez.

On démarre la fonction loop

La led verte est allumé.

Si on a un appui sur le bouton SOS :

- On commence par faire clignoter la led orange pour signaler l'alarme et éteindre la led verte.
- Et on clignote tant que le sauveteur n'a pas appuyé sur le second bouton.
- Si on appui sur le second bouton la led orange reste allumé fixe.

Un autre appui sur le bouton SOS :

- On commence par faire clignoter la led rouge pour signaler l'alarme et éteindre la led orange.
- Et on clignote tant que le sauveteur n'a pas appuyé sur le second bouton.
- Si on appui sur le second bouton la led rouge reste allumé fixe.

Si on a un appui sur le bouton SOS : on revient au 1er cas.