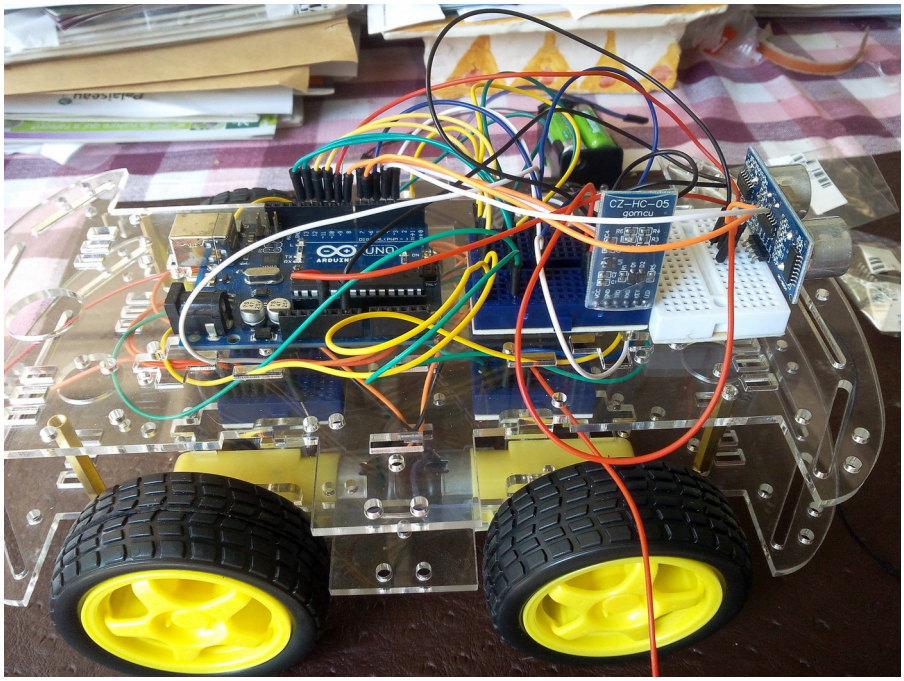
Construire sa voiture bluetooth-commandée avec arduino

I Le resultat final

Voici la voiture que vous allez être sur le point de construire



Mais pour cela il va nous falloir un peu de materiel en voici la liste :
-> un chassis avec 4 moteurs CC
-> 5 mini breadboards
-> 2 L293D
-> un speaker
-> un module bluetooth HC-05
-> un module ultrason SR-04
-> 2 blocs de 4 piles AA
-> une pile 9V ou une alimentation USB
-> des fils
-> un arduino (ici un uno)

Le code complet de la voiture et des différents composants est disponible ici :
https://github.com/maximouth/voiture_arduino

Mais avant de commencer à tout brancher, nous allons voir comment fonctionnent les differents composants de cette voiture.

II Le bluetooth

Nous allons commencer par voir le fonctionnement du module HC-05 qui est un port serie bluetooth.

Voici à quoi il ressemble :

« photos »

Il possède 6 broches, mais seulement 4 nous seront utiles ici.

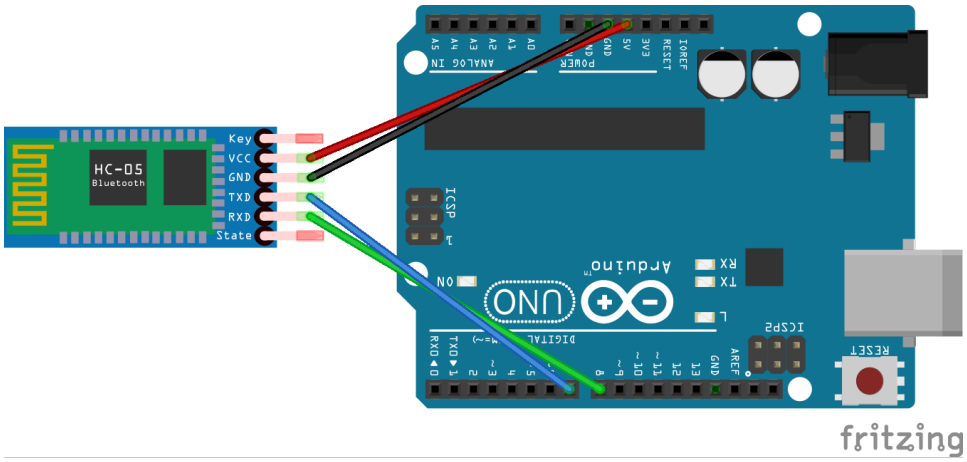
-> VCC, GND correspondent aux bornes positives et negatives de la carte.

-> Rx et Tx sont les dex broches qui servent à envoyer et recevoir les informations transmisent.

-> Key sert à configurer le module (nom,mdp...) nous ne nous ne servirons pas ici.

-> Led est utiliser pour brancher le module sur une led et ainsi observer les echanges de données, on ne s'en servira pas non plus.

Une fois cela vu nous allons nous attaquer au montage, il est assez simple à réaliser.



VCC sur 5V

GND sur GND

TXD sur la broche 7

RXD sur la broche 8

Nous allons maintenant nous attaquer au code !

Pour ce faire nous utiliserons la libraire « softwareserial » qui va nous permetttre de controler ce module. Ce sera l'unique librairie utilisée lors la création de cettre voiture.

On va commencer par cette ligne à rajouter en haut du ficher

```
#include <SoftwareSerial.h>
```

qui permet d'inclure la librairie cité au dessus.

Et maintenant nous allons creer notre objet qui va nous permettre d'utiliser le HC-05.

```
SoftwareSerial bluetooth_card(8,7);
```

8 pour l'endroit où est branché RXD et 7 pou TXD.
Notre objet btsp est donc crée et va nous permettre de dialoguer entre un objet connecté en bluetooth au HC-05 et notre arduino.

Il faut maintenant initialiser notre objet avec la ligne suivante

```
bluetooth_card.begin(9600);
```

Notre prochain objectif est de lire un caractère envoyé depuis un télephone et de l'afficher à l'ecran grace au port serie de l'arduino.
Le code va donc être maintenant écrit dans la fonction `loop()`

```
if (bluetooth_card.available()) {
data=bluetooth_card.read();
Serial.println (data) ;
delay (100) ;
}
```

Le code suivant fait :
Si il y a quelque chose à lire -> le lire , l'afficher et attendre 100 ms
Sinon rien

Nous savons maintenant comment recuperer un caractère envoyé par bluetooth et l'afficher.
Voici le code complet du programme

```
#include <SoftwareSerial.h>

char data;
SoftwareSerial bluetooth_card (8,7);

void setup() {
  bluetooth_card.begin(9600); // initialise le hc-05
  Serial.begin(9600); // initialise le port serie de l'arduino
}
```

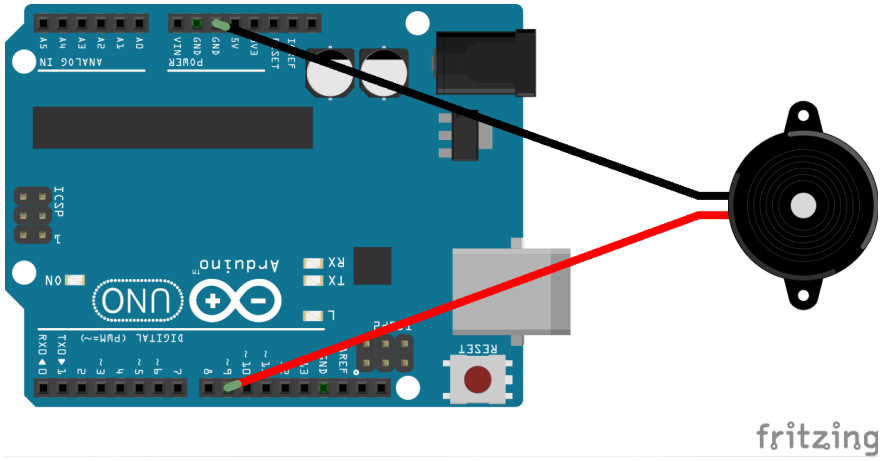
```
void loop() {

  if (bluetooth_card.available()) {
    data = bluetooth_card.read();
    Serial.println (data);
    delay (100);
  }

}
```

disponible ici :
https://github.com/maximouth/voiture_arduino/tree/master/code/bt

III Le speaker

Dans cette partie nous allons voir comment utiliser un petit speaker, très facile d'utilisation que voici
« photo »

Il ne posède que deux broches, VCC et GND,
Brancher VCC sur la broche 9 de votre arduino et GND sur GND de l'arduino.



Nous allons utiliser le pwm pour generer un signal qui va être transformer en son, en utilisant `analogWrite (9, x)`

-9 car nous avons branché le speaker sur la broche 9
- x une valeur entre 0 et 255, la valeur que vous allez mettre à x changera le son qui sortira, prenez celui qui vous conviens le mieux comme klaxon.

Nous allons tester toutes les sons possible de generer avec notre speaker dans un petit programe.
On va commencer par declarer notre speaker avec cette ligne en haut du fichier :

```
const int spk = 9;
```

puis notre x qui va servir de compteur.

```
int x = 0;
```

nous pouvons maintenant ecrire notre fonction `beep()` *qui servira a produire un son pendant 1 seconde.*

```
void beep(){
  analogWrite(spk,x); // produire un son
  // x va prendre une valeur differente à chaque tour
  delay (1000); // attendre 1 seconde
  analogWrite(spk,0); // arreter le son
}
```

Mais à quel moment la valeur de x change elle me direz vous ? Et bien dans la fonction loop, on va appeler `beep()` *puis incrementer x (modulo 256 pour le pas depasser 255) et enfin attendre 2 secondes avant de recommencer.*

Voici le code complet :

```
const int spk = 9;
int x = 0;


void setup()  {
  // declare pin 9 to be an output:
  pinMode(spk, OUTPUT);
 }

void loop()  {
  beep();
  x = (x+1)%256;
  delay (2000);
}

void beep(){
  analogWrite(spk,x);      // x va prendre une valeur differente à
chaque tour
  delay (1000);
  analogWrite(spk,0);
}
```

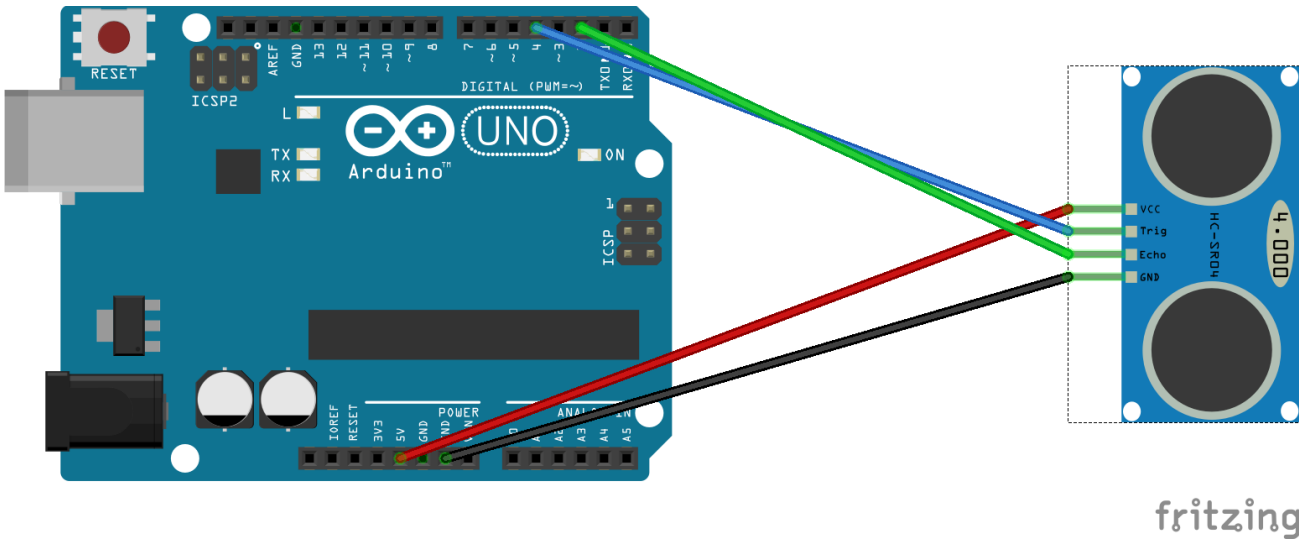
disponible ici :
https://github.com/maximouth/voiture_arduino/tree/master/code/klaxon

IV Le module ultrason

Nous allons maintenant nous interesser à notre module ultrason SR-04, et à comment recuperer la ditance entre le module et un objet.

Pour commencer voici le module en question :
« photo »

Ce module possède 4 broches, comme d'habitude VCC et GND pour les deux bornes '+' et '-' , Trig sert à commander l'émetteur du module et Echo le recepteur.
Voici de quel façon nous allons le brancher à notre arduino.



Une fois les branchements faits nous allons pouvoir commencer à l'utiliser.

Ouvrez donc un nouveau fichier, et commencer par declarer les deux broches sur lequelles sont branchées votre capteur.

```
const int trig = 4;
const int echo = 2;
```

Et dans la `setup()` *n'ouliez pas de definir si ces broches sont en entrée ou sortie sur votre carte.*
Trig etant l'émetteur, sa broche sera en sortie,

```
pinMode(trig, OUTPUT);
```

et Echo l'e recepteur nous auront donc besoin des informations recueillies, sa broche sera en entrée,

```
pinMode(echo, INPUT);
```

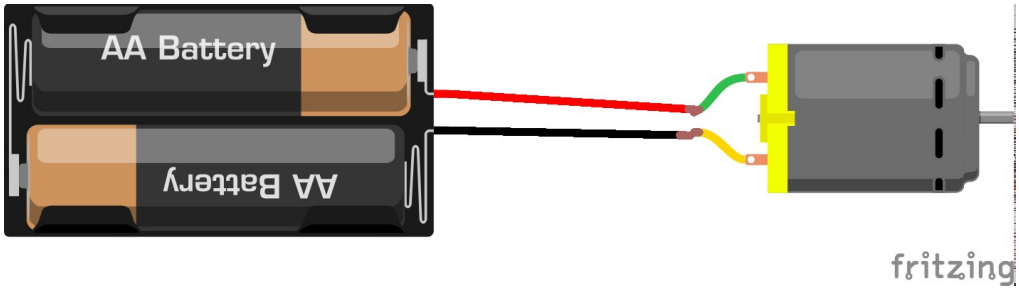
REGARDER COMMENT MARCHE EXACTEMENT LE CALCUL DE DISTANCE, ET LA FONCTION `PULSE IN()` *ET POURQUOI ON MET HIGH SUR TRIG PENDANT 10 uS*
voir cette page :
https://itechnofrance.wordpress.com/2013/03/12/utilisation-du-module-ultrason-hc-sr04-avec-larduino/

V Les Moteurs CC

Voici l'une des parties les plus difficiles, non pas à réaliser, mais à comprendre, faire tourner un moteur nimporte qui en serait capable, mais ici nous allons voir comment regler la puissance du moteur, son sens de rotation, et le plus important pour notre voiture, comment en utiliser plusieurs en même temps pour faire avancer une voiture.

Nous allons avoir besoin dans cette partie d'un peu plus de materiel qu'avant :
un moteur, un l293d et notre arduino, avant de commencer nous allons voir le fonctionnement d'un l293D, et de quelle façon l'utiliser avec l'arduino.
Le voici :
« photos »

Un moteur CC ne possède que deux broches, une pour la borne positive l'autre pour la négative d'une alimentation. Branché dans un sens le moteur tournera dans un sens, branché dans l'autre il ira dans le sens inverse.
Vous pouvez tester de cette façon en utilisant un de vos bloc de pile :



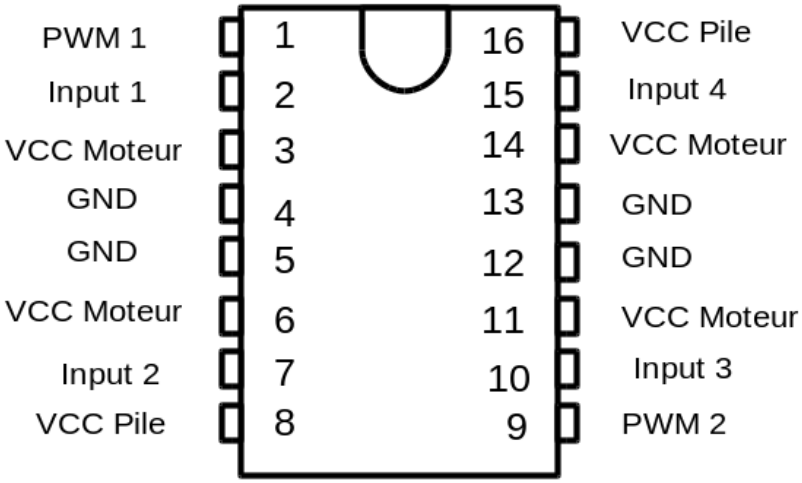
et refaire la meme chose en inversant les bornes.

Vous voyez ce n'est pas trop difficile jusque la.
Mais comment brancher le moteur avec l'arduino et les piles pour que le moteur ne fasse pas que de tourner même avec une carte non alimentée.
Le L293D me direz vous ? Et bien c'est tout à fait cela !

L'arduino ne pouvant delivrer assez de puissance pour faire fonctionner les moteurs sans risque de griller votre carte, nous sommes obligé d'utiliser un alimentation exterrieure ainsi qu'un petit composant pour relier le tout ensemble.

Et le voici sa fichetechnique pour connaitre les tensions maximales du composant ainsi que ces caracteristiques techniques :
http://www.ti.com/lit/ds/symlink/l293.pdf

ainsi que la façon de le brancher pour nous, Nous utiliserons un coté du L293D par moteur.



Nous allons maintenant expliquer comment se servir de chaques entrées.

Le PWM va nous servir à controler la vitesse à laquelle tournera le moteur, cela va marcher un peu de la même facon qu'avec notre speaker, nous allons utiliser la fonction `analogWrite(pin PWM, vitesse).`

VCC pile sert à brancher la borne plus de votre alimetation dessus et les différentes sorties GND iront sur la borne moins, eviter d'inverser ces deux bornes sur votre L293D par risque de faire enflamer le composant (si si c'est possible), et peut être votre voiture si vous ne vous en rendez pas compte.
(bon le composant brulé à été vecu, mais la voiture n'a subit aucun degats, juste une breadboard qui a un peu fondu mais rien de grave)
Les broches 3-6 et 11-14 servent à brancher les moteurs dessus, pas de sens precis,

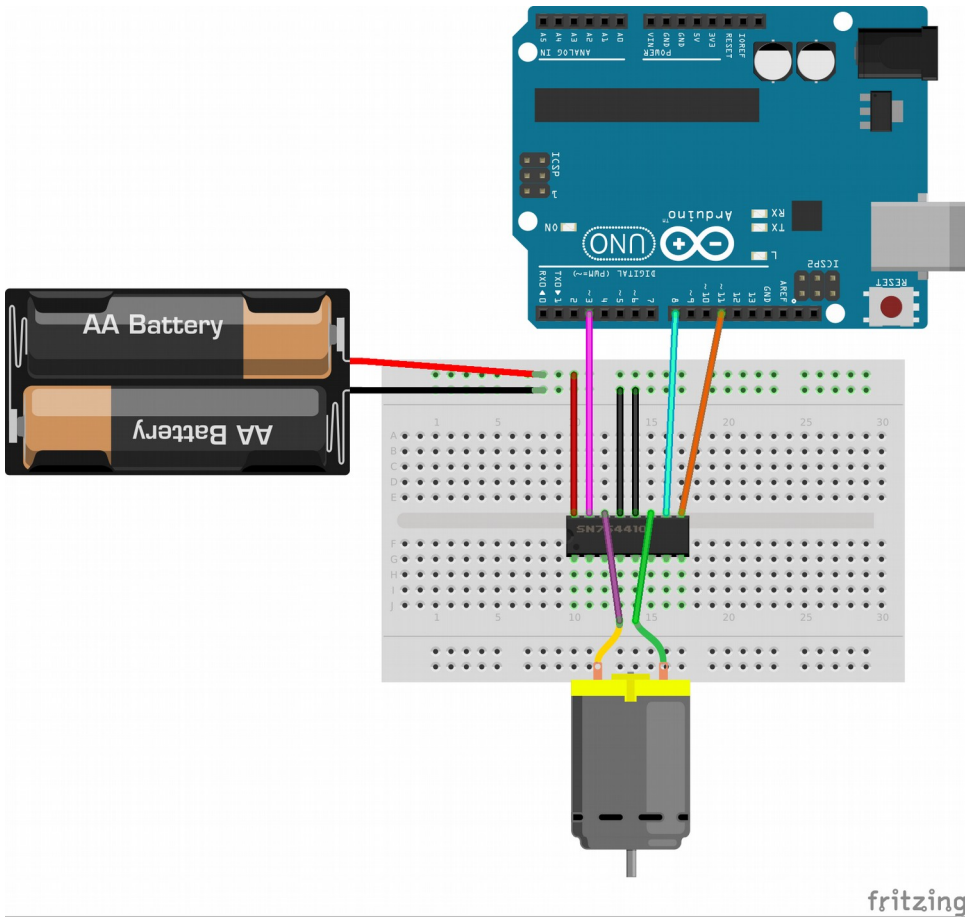
essayer juste de tous les brancher dans le même sens si vous en utilisez plusieurs cela evitera d'avoir à trop jouer avec les entrée 2-7 et 10-15

Nous allons donc voir de quelle facon nous servir de ces 4 broches, elles servent à controler le moteur, c'est à dire le faire tourner dans un sens, dans l'autre ou le stopper.

Pour cela nous allons utiliser la fonction `digitalWrite(num Input,HIGH/LOW)` *qui selon les différentes que peuvent prendre les deux inputs 2 à deux va faire quelquechose, voici les différentes possibilitée (1 -> HIGH) (0 -> LOW)*

Input 1 (2-10)	*Input 2 (7-15)*	*Effet*
0	*0*	*Frein*
0	*1*	*Tourne sens des aiguilles d'une montre*
1	*0*	*Tourne dans le sens inverse*
1	*1*	*Frein*

Nous allons maintenant pouvoir réaliser ce montage et comprendre de quelle façon l'utiliser.



Nous allons faire augmenter puis diminuer la vitesse de rotation du moteur sur toutes les valeurs possibles dans ce nouvel exempleque nous ecrirons dans un fichier motor.ino.

Pour commencer, nous allons comme d'habitude declarer nos differentes broches :

```
const int input1 = 8;
const int input2 = 3;
const int pwm = 11;
```

car la broche 8 de notre arduino est branchée sur input1, la 3 sur input2, et la 11 qui permet l'utilisation du pwm en 11.

Il va aussi nous falloir deux variables, une pour la valeur qui l'on va donner à la fonction `analogWrite()` *et une autre pour indiquer si on est entrain d'augmenter ou de reduire la vitesse de notre moteur.*

```
int x = 0;
int sens = 0; // 0 pour incrementer - 1 pour decrementer
```

Dans `setup()` *nous allons devoir declarer nos 3 broches comme broches de sortie.*

```
  pinMode(input1, OUTPUT);
  pinMode(input2, OUTPUT);
  pinMode(pwm, OUTPUT);
```

Une fois ces declarations faites, nous allons nous attaquer au gros de la fonction à l'interrieur de loop.
Voici le principe de notre fonction.

A chaque tour de boucle nous allons ecrire la nouvelle vitesse à laquelle va tourner notre moteur de cette façon `analogWrite (pwm,x);` *, puis nous occuper de changer la valeur de ce x.*

Si sens est egal à 0, c'est que l'on est entrain de monter la vitesse, du coup on augmente x de 1, on verifie si notre x n'a pas depassé 255 (valeur max) sinon on inverse le sens en le mettant à un et on modifie x pour lui donner 255 comme valeur. Puis attendre 500 ms avant de recommencer.
Si sens est egal à 1, on fait le symetrique de si sens est egal à 0.

Voici le code complet :

```
const int input1 = 8;
const int input2 = 3;
const int pwm = 11;

int x = 0;
int sens = 0; // 0 pour incrementer - 1 pour decrementer

void setup() {

  pinMode(input1, OUTPUT);
  pinMode(input2, OUTPUT);
  pinMode(pwm, OUTPUT);

}

void loop() {

  analogWrite (pwm,x);
  if (sens = 0) {
    x++;
    if (x > 255) {
      sens = 1;
      x = 255;
    }
  }
  else {
    x--;
    if (x < 1) {
      sens = 0;
      x = 1;
    }
  }

  delay (500);

}
```

disponible ici :
https://github.com/maximouth/voiture_arduino/tree/master/code/motor

VI Le montage final
VII code, video, améliorations possibles, sources