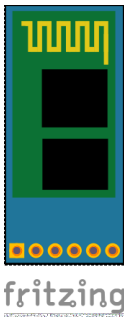
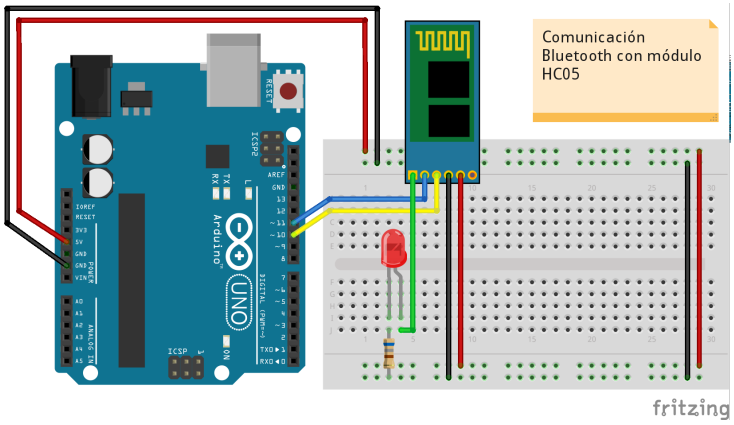
Módulo Bluetooth HC05

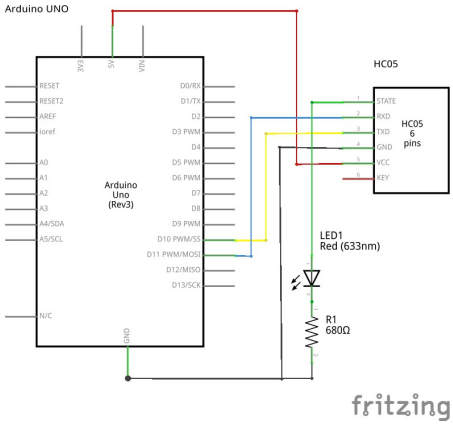


Nivel: bajo

Con este montaje se muestra la sencillez de conexionado del modulo HC05 (una vez está en su board) con el Arduino. La electrónica de la placa interfaz de nuestro HC05 posee un botón que nos permite pasar al modo envío de comandos (modo AT) y una electrónica que realiza un desplazamiento de nivel (el núcleo delHC05 funciona a 3.3v)
El pinout del módulo, visto de frente, es: STATE, RXD,TXD,GND,VCC y KEY.

STATE es un pin de salida del que podemos obtener el estado del modulo, cuando se pone a nivel alto (por defecto, depende de la configuración) es cuando está enlazado con otro dispositivo.
RXD, TXD, GND y VCC son los pines básicos de cualquier comunicación serial.
Y finalmente, KEY. Se trata de un pin con el que se puede pasar el dispositivo al modo AT (no necesario con nuestra placa)





Conexionado básico del modulo:

Conectamos STATE con un LED y una resistencia limitadora a GND, RXD con el pin 11 de Arduino, TXD con el pin 10 de Arduino, VCC con VCC, GND con GND y KEY lo dejamos sin conectar. Ahora sólo falta cargarle el programa (Bluettooth_HC0x.ino) para poderle pasar comandos desde el terminal y ver así las diferentes opciones de configuración.

Los comandos que vamos a ver son los más básicos que nos permiten personalizar y hacer configuraciones simples, además de interconectar con otros dispositivos. Todos estos comandos y muchos más, los pueden consultar en el documento anexo disponible en nuestro repositorio.

NOTA: No todos los comandos descritos están disponibles en nuestro módulo, depende del firmware que tenga instalado.

Bluetooth_HC0x.ino

```
#include <SoftwareSerial.h> //Librería que permite establecer comunicación serie en otros
pins

//Aquí conectamos los pins RXD,TDX del módulo Bluetooth.
SoftwareSerial BT(10,11); //10 RX, 11 TX.

void setup()
{
  BT.begin(9600); //Velocidad del puerto del módulo Bluetooth
  Serial.begin(9600); //Abrimos la comunicación serie con el PC y establecemos velocidad
}

void loop()
{
  if(BT.available())
  {
    Serial.write(BT.read());
  }

  if(Serial.available())
  {
     BT.write(Serial.read());
  }
}
```

Los comandos que vamos a utilizar son :

AT
Con este determinamos si estamos en modo AT

AT+VERSION
Con este la versión del firmware

AT+ADDR?
Con este averiguamos la dirección física del dispositivo

AT+NAME[?|=<param>]
Con este podemos averifguar o asignarle un nombre.

AT+ROLE[?|=<param>]
Con este podemos determinar el comportamiento: 0-> Slave, 1-> Master, 2-> Slave-loop (Por defecto: 0)

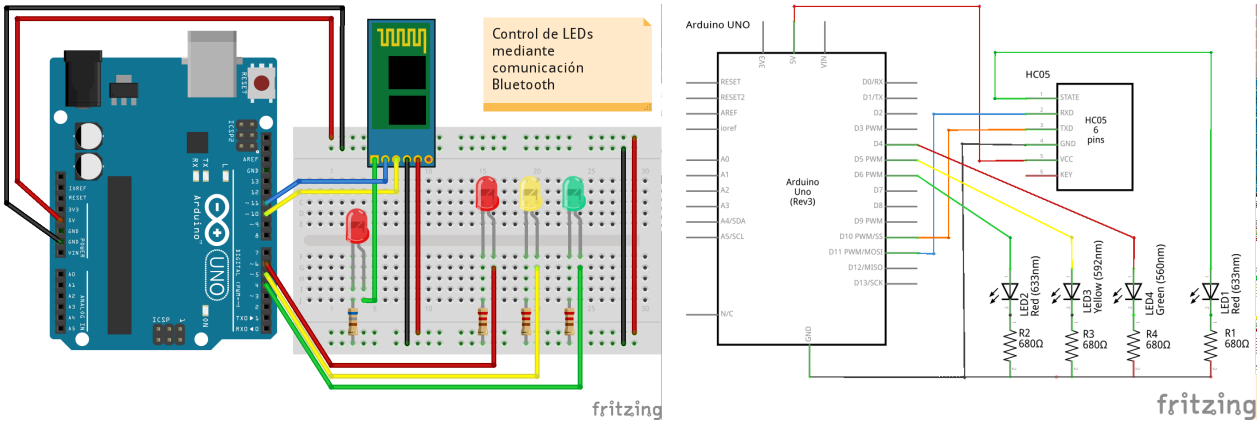
AT+PSWD[?|=param]
Con este podemos averiguar o establecer la contraseña (Por defecto: 1234)

AT+LINK=<param>
Con este vamos a conectarnos al dispositivo con la dirección indicada, el formato es xxxx,xx,xxxxxx.

NOTA: El resto de comandos los vamos a considerar PELIGROSOS, y por lo tanto se usarán bajo su responsabilidad.

Nivel: medio

Con este montaje vamos a controlar una serie de LEDs mediante comunicación entre dos módulos HC05. Uno usará el programa anterior y el segundo usará el siguiente (CtrlBTLEDs.ino)



El diagrama a conectar es el siguiente (se usa como base el diagrama anterior)

CtrlBTLEDs.ino

```
/*
  www.diymakers.es
  by A.García
  Arduino + Bluetooth
  Tutorial en: http://diymakers.es/arduino-bluetooth/
*/

#include <SoftwareSerial.h> //Librería que permite establecer comunicación serie en otros pins

//Aquí conectamos los pins RXD,TDX del módulo Bluetooth.
SoftwareSerial BT(10, 11); //10 RX, 11 TX.

int green = 4;
int yellow = 5;
int red = 6;
char cadena[255]; //Creamos un array de caracteres de 256 cposiciones
int i = 0; //Tamaño actual del array

void setup()
{
  BT.begin(9600);
  Serial.begin(9600);
  pinMode(green, OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(red, OUTPUT);
}

void loop()
{
  //Cuando haya datos disponibles
  if (BT.available())
  {
    char dato = BT.read(); //Guarda los datos carácter a carácter en la variable "dato"

    cadena[i++] = dato; //Vamos colocando cada carácter recibido en el array "cadena"

    //Cuando reciba una nueva línea (al pulsar enter en la app) entra en la función
    if (dato == '\n')
    {
      Serial.print(cadena); //Visualizamos el comando recibido en el Monitor Serial

      //GREEN LED
      if (strstr(cadena, "green on") != 0)
      {
        digitalWrite(green, HIGH);
```

```
      }
      if (strstr(cadena, "green off") != 0)
      {
        digitalWrite(green, LOW);
      }
      //YELLOW LED
      if (strstr(cadena, "yellow on") != 0)
      {
        digitalWrite(yellow, HIGH);
      }
      if (strstr(cadena, "yellow off") != 0)
      {
        digitalWrite(yellow, LOW);
      }
      //RED LED
      if (strstr(cadena, "red on") != 0)
      {
        digitalWrite(red, HIGH);
      }
      if (strstr(cadena, "red off") != 0)
      {
        digitalWrite(red, LOW);
      }
      //ALL ON
      if (strstr(cadena, "on all") != 0)
      {
        digitalWrite(green, HIGH);
        digitalWrite(yellow, HIGH);
        digitalWrite(red, HIGH);
      }
      //ALL OFF
      if (strstr(cadena, "off all") != 0)
      {
        digitalWrite(green, LOW);
        digitalWrite(yellow, LOW);
        digitalWrite(red, LOW);
      }

       BT.write("\r"); //Enviamos un retorno de carro de la app. La app ya crea una línea
nueva
      clean(); //Ejecutamos la función clean() para limpiar el array
    }
  }
}

//Limpia el array
void clean()
{
  for (int cl = 0; cl <= i; cl++)
  {
    cadena[cl] = 0;
  }
  i = 0;
}
```