

EL MÓDULO BLUETOOTH HC-05



Arduino y la conexión BlueTooth

Version: 07-09-17

OBJETIVOS

- Presentar el módulo HC-05.
- Describir las ventajas y diferencias con el módulo HC-06.
- Presentar más posibilidades con los comandos AT.
- Presentar un montaje y programa que nos permite entrar directamente en modo de configuración de comandos AT.

MATERIAL REQUERIDO.

	Arduino UNO o equivalente.
	Algunos cables de protoboard, preferiblemente Dupont macho/hembra.
	Un módulo BlueTooth HC-05 , tiene 6 pines.

EL MÓDULO BLUETOOTH HC-05

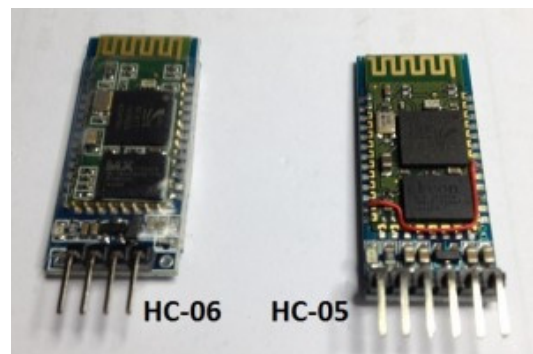
En la sesión anterior presentamos algunos conceptos básicos sobre las redes Bluetooth y las funciones que se suelen realizar con ellos. Presentamos el concepto de Master y Slave y configuramos el modulo HC-06 que solo podía trabajar en el modo Slave.

En esta sesión, vamos a utilizar su hermano mayor el modulo HC-05, que puede configurarse tanto como Master que como Slave, y que además dispone de bastante más parámetros de configuración y capacidades de interrogación.

El aspecto externo es bastante similar y la mejor manera de diferenciarlos es por los pines el soporte, 4 para el HC-06 y 6 pines para el HC-05.

No hay diferencias hardware entre ambos modelos, pero sí que hay diferencias muy importantes en el firmware (La programación interna del módulo).

Además, mientras que el HC-06 entra en modo de programación en cuanto lo enciendes y mientras no haya nadie conectado por Bluetooth, el HC-05 es ligeramente más complicado de colocar en modo comandos y requiere una cierta manera de arrancado, concretamente requiere que el pin KEY, (Que no estaba conectado el caso del HC-06) este en HIGH cuando encendemos el modulo.



He visto bastante información he Internet que recomendaba unos ciertos procesos para arrancar el modulo que siempre me han parecido confusos y tienden a despistar a los que se acercan al tema por primera vez.

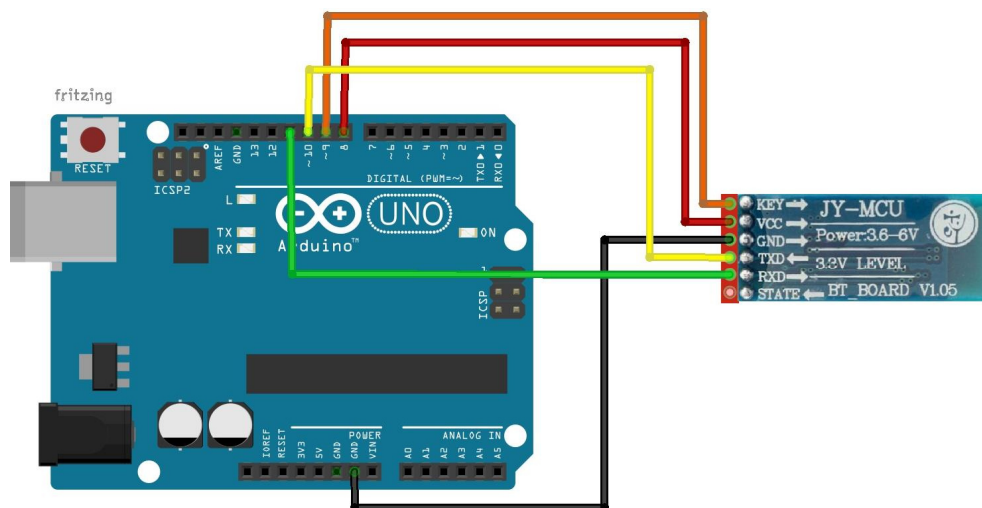
Por eso hemos decidido implementar este procedimiento en un Sketch de Arduino, para que podamos correrlo cuando necesitemos programar el módulo.

El procedimiento normal con estos módulos, suele ser conectarlos, ver la configuración y reprogramarlos con nuestras preferencias. Después mantendrá la programación hasta que decidamos cambiarla.

Pero parece que el proceso da bastante guerra, así que vamos a ver si podemos ayudar.

CONEXIÓN DEL BLUETOOTH HC-05 AL ARDUINO

La conexión es muy sencilla aunque requiere algún cable más que el modulo HC-06, y algún cambio:



En primer lugar, para que el HC-05 entre en modo comandos AT, requiere que cuando se enciende el modulo, el pin KEY este HIGH. Por eso hemos conectado la tensión Vcc del módulo BlueTooth al pin 8 de nuestro Arduino.

El consumo del módulo es mínimo y nuestro Arduino es capaz de alimentarlo sin problemas, por eso el modulo se encenderá cuando pongamos HIGH en el pin 9. Esto nos permitirá poner en HIGH el pin digital 8, al iniciar nuestro programa y después levantar el pin 8, de este modo cuando arranque entrara sin más en el modo de comandos AT.

El resto de los pines se conectan de forma similar a lo que hicimos en la sesión anterior. Txd y Rxd se deben conectar cruzados con los pines de comunicación de Arduino, que usaremos mediante la librería software Serial.

El pin State refleja, supuestamente, la situación en la que se encuentra el modulo y por ahora no vamos a utilizarlo.

PROGRAMA DE CONTROL

NO OLVIDE VER ENSAYOS REALIZADOS POR EL DOCENTE MAS ADELANTE

Así pues, no tenemos nada nuevo en el programa excepto que alimentaremos el modulo desde el pin digital 8, para forzarle a entrar en el modo comandos AT. Nuestro programa quedaría así:

```
#include <SoftwareSerial.h>

SoftwareSerial BT1(10, 11); // RX | TX

void setup()

{ pinMode(8, OUTPUT);          // Al poner en HIGH forzaremos el modo AT

  pinMode(9, OUTPUT);          // cuando se alimente de aqui
```

```

digitalWrite(9, HIGH);

delay (500) ;           // Espera antes de encender el modulo

Serial.begin(9600);

Serial.println("Levantando el modulo HC-06");

digitalWrite (8, HIGH); //Enciende el modulo

Serial.println("Esperando comandos AT:");

BT1.begin(57600);

}

void loop()

{ if (BT1.available())

    Serial.write(BT1.read());

    if (Serial.available())

        BT1.write(Serial.read());

}

```

Con este procedimiento garantizamos que el modulo HC-05 entra solo en modo AT comandos, y que nuestra consola nos va a servir para programarlo, y confío en que nos evite todas las complicaciones.

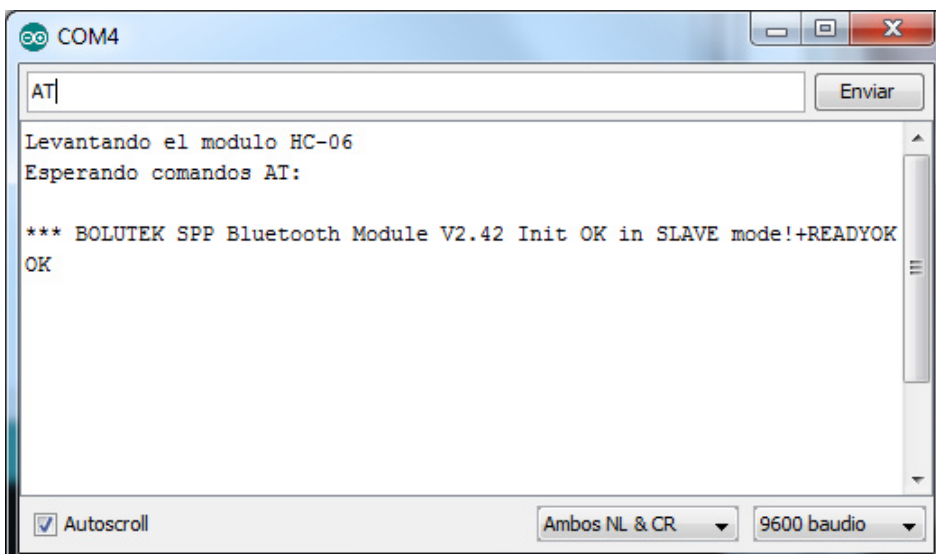
Mi HC-05 me envía a la consola un mensaje de Status en cuanto arranca en el modo AT, si es vuestro caso deberíais ver un mensaje en la consola. Si no es así, o veis signos raros, suele ser indicador de que la velocidad de comunicación entre Arduino y el modulo es incorrecta.

Id probando velocidades hasta que veáis algún mensaje correcto. Enviar un AT siempre debería recibir una respuesta de OK.

PROGRAMANDO EL HC-05 CON COMANDOS AT

(NO DEJAR DE VER ENSAYOS DEL DOCENTE, LOS MODULOS PUEDEN DIFERIR)

Confío en que si hemos seguido los pasos anteriores cuando abras la consola deberías recibir un mensaje del módulo directamente, y si no, basta con que escribáis AT Intro, recibiréis un OK como muestra la captura de pantalla:



El mensaje que obtengáis no tiene por qué ser el mismo e incluso puede que no tengáis ningún mensaje en absoluto, pero es importante que os aseguréis de que tenéis seleccionado Ambos NL&CR para terminar las líneas.

El modulo HC-6, finaliza los comandos por tiempo y por eso en la sesión anterior usamos GetLine para conseguir líneas completas. Pero el HC-05, no hace esto sino que espera que la línea acabe educadamente en \r\n y entonces ejecuta el comando.

- *Aparentemente el número de modificaciones de software con la que se venden estos módulos es bastante elevada y por ello no es fácil garantizar nada con ellos.*
- *Aparentemente todos los módulos que he probado aceptan ordenes similares, pero no todos los módulos aceptan todas las ordenes,(Supongo que dependiendo del Firmware).*

Cuando menos si escribís en mayúsculas AT e intro, deberías recibir una respuesta de OK en la consola. Y si es así podemos seguir viendo más comandos.

AT+VERSION, Requiere la versión del Firmware. En la captura de arriba muestra el resultado

AT+NAME, Requiere que nos informe del nombre que tiene asignado el modulo. Debería devolvernos un mensaje del tipo +NAME=HC-05, indicando que se llama HC-05.

- *El modulo HC-06 se podía renombrar exactamente así, pero no podíamos preguntarle qué nombre tenía asignado. Solo cambiarlo.*

AT+NAMEXXXX, programa el nombre que queremos presentar cuando alguien nos localice:

AT+NAMECharly

AT+BAUD, nos permite solicitar la velocidad a la que está programado el modulo para hablar con Arduino, y AT+BAUDX, Fija la velocidad de comunicación entre el modulo y la consola de acuerdo a la siguiente tabla:

1 configura	1200bps
2 configura	2400bps
3 configura	4800bps
4 configura	9600bps (Default)
5 configura	19200bps
6 configura	38400bps
7 configura	57600bps
8 configura	115200bps

Ejemplo: AT+BAUD7 configura la comunicación a 57600 baudios

Si enviamos AT+BAUD, la respuesta es de la misma forma: BAUD=7 para indicar 9600.

AT+PIN, Solicita el PIN actual y en la consola veris: PIN=1234 o similar.

AT+PINXXXX, configura el número de identificación personal, que se requerirá para establecer la vinculación

AT+PIN4516, establece 4516 como PIN.

- Recordad que el PIN es el número de identificación personal, que usaremos al conectarnos al módulo, pues exigirá conocer la contraseña.
- El PIN es de 4 dígitos siempre

Básicamente estos son los mismos comandos que aceptaba el modulo HC-06 pero el HC-05 acepta bastante más.

AT+ROLE Nos informa de si está configurado como Maestro 1, o como esclavo 0.

ROLE=0 (Esclavo)

AT+ROLE1 Configura el modulo como Master.

AT+ROLE0 Configura el modulo como Slave.

- *He visto muchos ejemplos por internet que indican que se debe usar un = para asignar valores, por ejemplo AT+ROLE=1, o que para pedir información de la velocidad hay que hacer AT+BAUD?, También he visto que varias páginas aseguran que para cambiar el PIN hay que hacer AT+PSSWD.*
- *Yo no he sido capaz de que nada de eso me funcione así, y sin embargo en mi modulo funciona la sintaxis que os escribo en los ejemplos. Tengo que suponer que hay diferentes Firmwares que modifican la sintaxis porque no puedo creer que haya tantas nomenclaturas erróneas corriendo por la red.*
- *Así que tendréis que probar cual es la que os funciona a vosotros.*

No tendría demasiado sentido revisar aquí todos los posibles comandos AT, y por eso he preferido mostrar media docena de ellos, los más típicos, para que podáis experimentar y configurar el modulo.

Aquí tenéis una lista con algunos comandos a los que más o menos he encontrado sentido (Porque hay montones que no tengo ni idea para que sirvan).

AT COMMAND LISTING	
COMMAND	FUNCTION
AT	Test UART Connection
AT+RESET	Reset Device
AT+VERSION	Query firmware version
AT+ORGL	Restore settings to Factory Defaults
AT+ADDR	Query Device Bluetooth Address
AT+NAME	Query/Set Device Name
AT+RNAME	Query Remote Bluetooth Device's
AT+ROLE	Query/Set Device Role
AT+CLASS	Query/Set Class of Device CoD
AT+IAC	Query/Set Inquire Access Code
AT+INQM	Query/Set Inquire Access Mode
AT+PSWDAT+PIN	Query/Set Pairing Passkey
AT+UART	Query/Set UART parameter
AT+CMODE	Query/Set Connection Mode
AT+BIND	Query/Set Binding Bluetooth Address
AT+POLAR	Query/Set LED Output Polarity
AT+PIO	Set/Reset a User I/O pin

Cuando hayamos programado el modulo como deseemos, podemos retirar la conexión del pin rotulado como KEY y el HC-05 está listo para trabajar normalmente sin aceptar ya comandos AT. En el caso ensayado dice EN

ATENCIÓN:

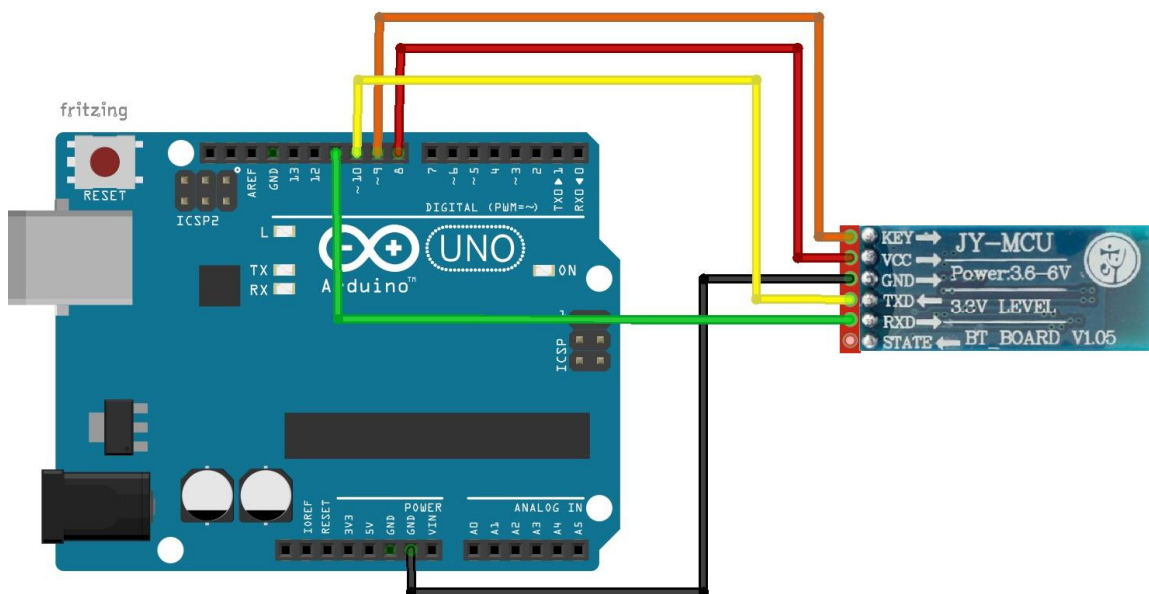
Es importante comprender, que en cuanto conectemos algún dispositivo a nuestro modulo Bluetooth HC-05, la luz roja dejará de parpadear y automáticamente saldremos del modo de comandos AT para entrar en el de transporte de información, es decir, cualquier cosa que le enviemos, incluyendo comandos AT, se consideraran texto plano que se reenviará al otro extremos de la conexión Bluetooth.

Recordar que solo podemos enviar comandos AT mientras no estemos conectados

RESUMEN DE LA SESIÓN

- Hemos presentado el modulo BlueTooth HC-05.
- Hemos visto las diferencias con el módulo HC-06 y que a diferencia de este, se puede configurar como maestro o como esclavo.
- También hemos visto que acepta bastantes mas ordenes en comandos AT.
- Hemos presentado un montaje que fuerza a entrar en el modo AT al módulo cuando arranca. Una vez programado podemos soltar el pin KEY y el modulo estará listo para el servicio.

ENSAYOS REALIZADOS POR EL DOCENTE Circuito empleado



Este es el programa que funciona

```
#include <SoftwareSerial.h>

SoftwareSerial BT1(10, 11); // RX | TX para no usar los pines especificos RX TX
void setup()
{ pinMode(8, OUTPUT); // Al poner en HIGH forzaremos el modo AT
  pinMode(9, OUTPUT); // cuando se alimente de aqui
  digitalWrite(9, HIGH);
  delay (500); // Espera antes de encender el modulo
  Serial.begin(9600);
  Serial.println("Levantando el modulo HC-05");
  digitalWrite (8, HIGH); //Enciende el modulo
  Serial.println("Esperando comandos AT:");
  // BT1.begin(57600); Original en programa
```

```

    BT1.begin(38400); //Si no cambiamos esta velocidad NO FUNCIONA(DJB)
  }

void loop()
{ if (BT1.available())
  Serial.write(BT1.read());
  if (Serial.available())
    BT1.write(Serial.read());
  }
}

```

Es importante aclarar este cambio:

// BT1.begin(57600); Original en programa

BT1.begin(38400); //Si no cambiamos esta velocidad NO FUNCIONA(DJB)

Se ejecutaran una serie de comandos en el monitor serial y se esperara la respuesta del modulo. (ver archivo [istd016A.pdf](#)). Notar como ciertos comandos en cierta sintaxis no funcionaron.

Comando AT	Respuesta del modulo HC-05
AT	OK
AT+VERSION Requiere la versión del Firmware	+VERSION:hc01.comV2.1 OK
AT+NAME Requiere que nos informe del nombre que tiene asignado el modulo	+NAME:HC-05 OK
AT+BAUD nos permite solicitar la velocidad a la que está programado el modulo para hablar con Arduino	ERROR:(0)
AT+NAMEXXXX programa el nombre que queremos presentar cuando alguien nos localice: Ej: AT+NAMEHC-05DJB	ERROR:(0)
AT+NAME=nombredeseado Permite cambia el nombre Ej: AT+NAME=ITeadStudio Cambia el nombre Cuando le pido el nombre AT+NAME Ej AT+NAME=HC05_DJB AT+NAME	OK +NAME:ITeadStudio OK OK +NAME:HC05_DJB OK
AT+PIN Solicita el PIN actual y en la consola veris: PIN=1234 o similar.	ERROR:(0)

AT+ROLE Nos informa de si está configurado como Maestro 1, o como esclavo 0.	+ROLE:0 OK
AT+PSWD? Check PIN code AT+PSWD=<Param> Cambiar el código PIN Ej: AT+PSWD=1963	+PSWD:1234 OK OK
AT+PSWD?	+PSWD:1963 OK

TERMINADA LA CONFIGURACION, AHORA VEREMOS COMO CONECTARNOS CON ESTOS MODULOS

PERO ANTES TENGAMOS EN CUENTA LO SIGUIENTE

El modulo que se ensayo admite conectar VCC a 5V (Power 3,6v a 6v), por lo tanto es posible conectar a los 5 v de Arduino (que es lo que se uso) e incluso se probó que funciona con la salida de 3,3 del Arduino.

Con esto tenemos resuelto el “encender” el módulo, ahora lo que nos hace falta es poder comunicarnos con él.

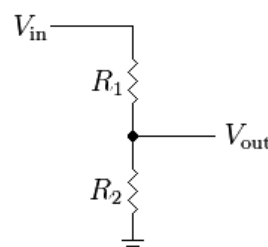
Para ello conectaremos dos pines digitales del Arduino (podemos utilizar el RX y TX que tenemos en los pines 0 y 1 respectivamente) con los pines de TX y RX del módulo, pero aquí nos encontramos con el problema de que nuestro Arduino está trabando a 5V y el módulo bluetooth en cuanto a señales trabaja a 3.3V (LEVEL 3,3v), y, aunque hay gente que los ha conectado directamente y asegura que le funciona perfectamente yo prefiero no arriesgarme y hacer una adaptación de niveles, dado que es muy simple.



En primer lugar tenemos la salida del módulo (TX, pin 1) en ella tendremos un nivel de 0 para el nivel lógico 0 y aproximadamente 3.3V para el nivel 1. Como según las características del ATmega328P el V_{ih} es $0.6V_{cc}$, y nuestro V_{cc} es de 5V, tendremos que V_{ih} es aproximadamente de 3V y por tanto **es perfectamente válido conectar el TX del HC-05 con el RX del Arduino.**

Algo totalmente diferente es la salida del Arduino hacia la entrada (RX, pin 2) del módulo. En el datasheet del HC-05 no he visto que indique que las entradas sean tolerantes a 5V, y por otro **lado en el datasheet del ATmega328P indica que V_{oh} es, como mínimo, de 4.2V**, por lo que, por precaución, **deberíamos adaptar los niveles.** Hay muchos métodos para hacer la adaptación de niveles, aunque, para mí, en este caso la solución más fácil es realizar un divisor resistivo, de forma que solo le entreguemos una parte de la tensión que esté dentro de sus márgenes de tolerancia al módulo bluetooth.

Para poder utilizar un divisor resistivo con señales lo primero que tenemos que tener en cuenta es a quien vamos a conectar la salida del divisor, ya que si tiene una impedancia en el mismo orden de magnitud (o inferior) que la resistencia de salida el paralelo de ambas hará que los cálculos que hayamos realizado estén mal. Para este paso nos iremos al datasheet del módulo y vemos que la entrada RX es de tipo CMOS con, esto significa que tendrá una



impedancia de entrada alta.

Llegados a este punto está claro queremos realizar un divisor de tensión que tenga una entrada de 5V y una salida de 3.3V, con una impedancia de salida que no sea muy alta. La disposición sera así:

La fórmula a utilizar es: $V_o = V_i \cdot R_2 / (R_1 + R_2)$

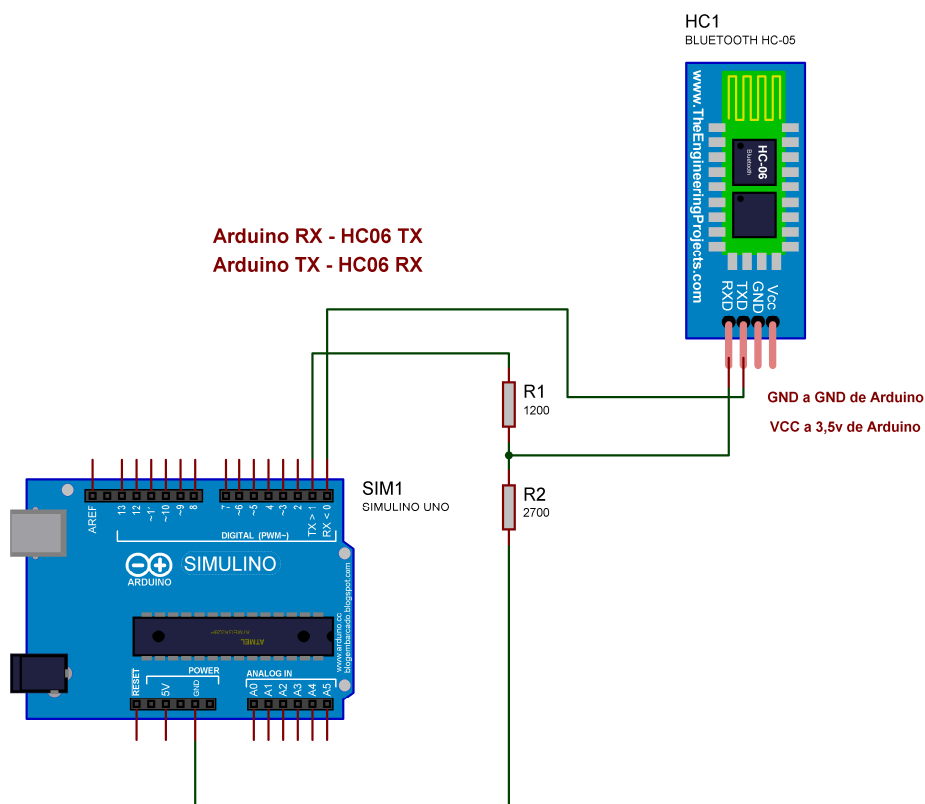
Cuando no es muy relevante la resistencia que hay que poner en una conexión uno de los valores típicos es 10KΩ, por lo que empezaremos por ese valor como resistencia de salida del divisor y vemos si nos sale algo razonable.

Si tomamos como valores $R_2 = 10K$, $V_i = 5V$, $V_o = 3.3V$ y despejamos R_1 , tendremos:

$$R_1 = (V_i / V_o - 1) \cdot R_2$$

Donde R_1 nos da un valor 5151.515Ω de forma que llevándolo a valores estándar podremos poner una resistencia de 4K7Ω o de 5K6Ω, en el primer caso tendremos una salida de 3.4V y en el segundo de 3.2V. Y eso es todo lo necesario para conectar el módulo, otro día veremos como comunicarnos con él.

Nota: Se ha probado con $R_2 = 2,7K$ y $R_1 = 1,2K$ dándonos una tensión aproximada de de 3,3 v



La conexión sería la siguiente entre el modulo Bluetooth HC 05 (valido también para el HC-06) y la placa Arduino:

Arduino 5v – HC05 VCC
Arduino GND – HC05 GND
Arduino TX – HC05 RX
Arduino RX – HC05 TX

Primer Ejemplo: El programa que se uso y funcionó es el siguiente: (PrimerBlue)

```
void setup()
{
  Serial.begin(9600); //Iniciar el serial
  pinMode(5, OUTPUT); //Establecer el pin 13 como salida
}

void loop()
{
```

```

if(Serial.available()>=1)
{
  char entrada = Serial.read(); //Leer un caracter

  if(entrada == 'h' or entrada == 'H') //Si es 'H', encender el LED
  {
    digitalWrite(5, HIGH);
    Serial.println("LED encendido");
  }

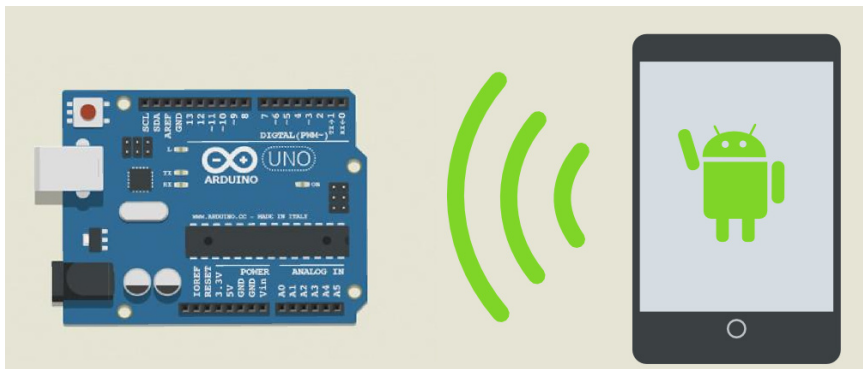
  else if(entrada == 'l' or entrada == 'L') //Si es 'L', apagar el LED
  {
    digitalWrite(5, LOW);
    Serial.println("LED apagado");
  }

  else if(entrada == 'i' or entrada == 'I') //Si es 'I', mostrar un mensaje de ayuda
  {
    Serial.println("Comandos:n (i) - abrir esta listan (h)- encender ledn (l) - apagar ledn");
  }
}
}

```

Recuerde desconectar momentáneamente el modulo Bluetooth cuando va a cargar el programa en el Arduino

Creación de la APP en Android para conectarse a Arduino por medio del modulo Bluetooth **PrimerBlue**



Vamos a programar una aplicación para Android con MIT App Inventor 2, un aplicativo web que nos permitirá construir aplicaciones rápidamente sin tener que programar (concepto relativo, ya que hay que saber algo de programación). Controlaremos un LED, construyéndonos una interface a medida.

El programa MIT App Inventor 2 es muy simple. Hay dos ventanas: una en la que se crea el diseño de la aplicación (arrastrando elementos de un menú tales como botones, barras y colocándolos en la pantalla del móvil) y la segunda para programar mediante bloques.

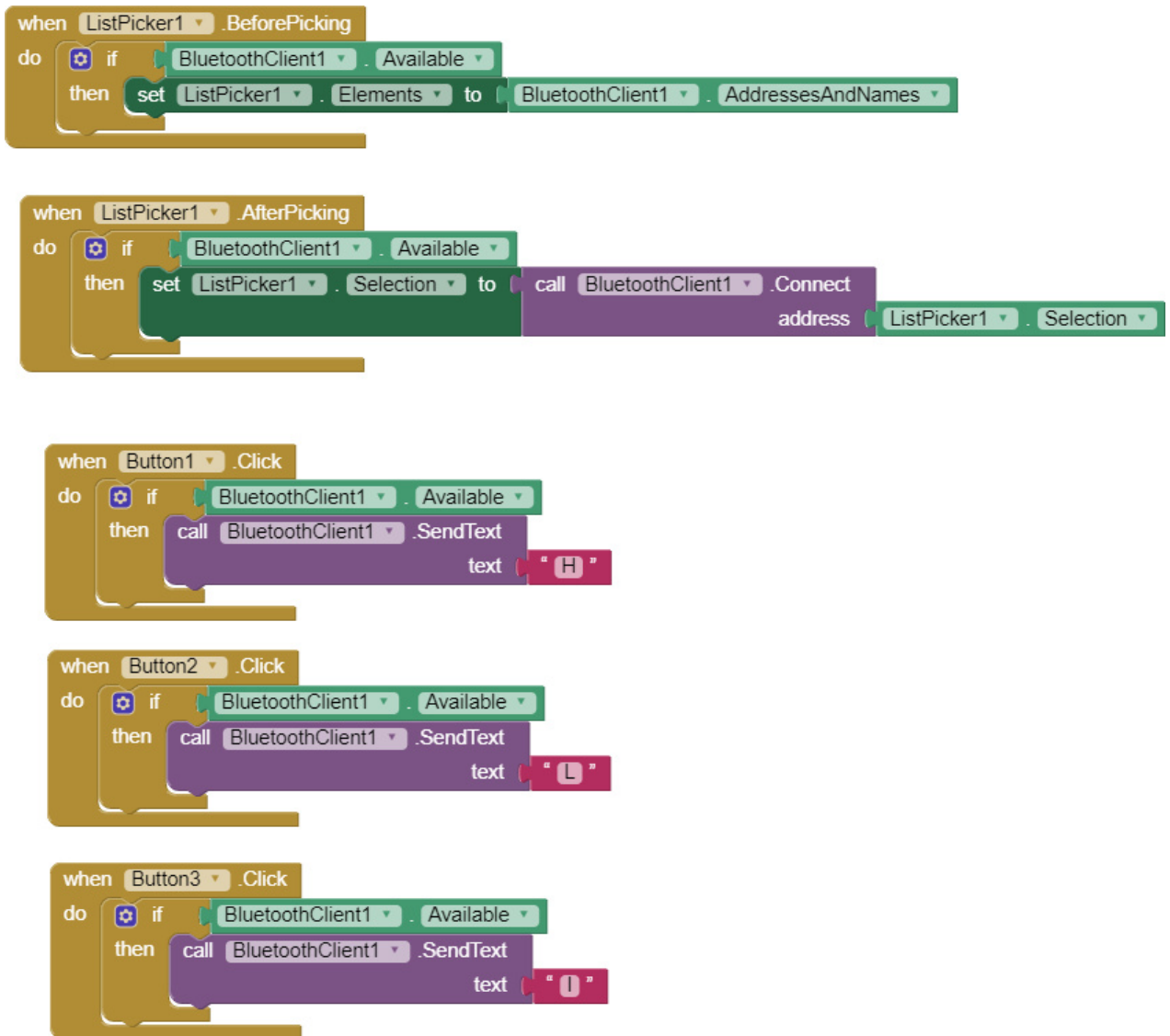
Asumiendo que ya tiene experiencia previa con el MIT App Inventor 2 (de lo contrario hay muchos tutoriales en Internet), los objetos que necesita serian los siguientes:

- List Picker
- Button1
- Button2
- Button3
- Bluetooth Client (de Palette->Connectivity)

La pantalla de la APP quedaría como se ve en la figura.



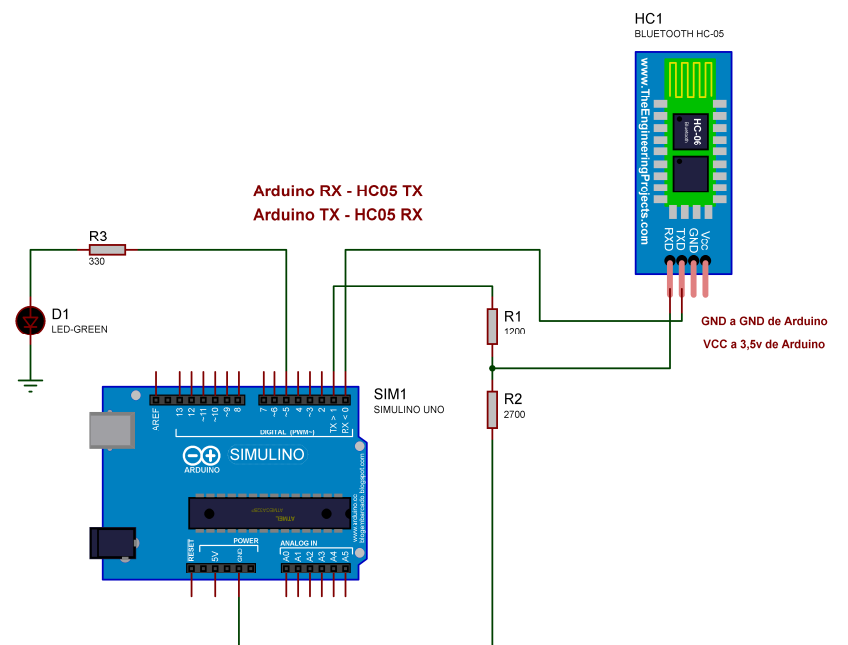
La estructura de bloques así:



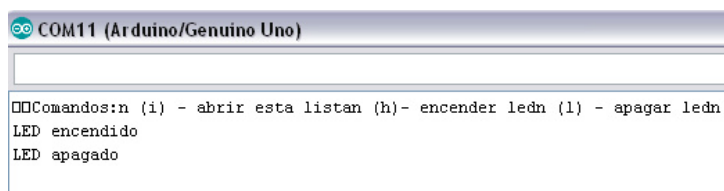
Luego de cargar la aplicación en nuestro telefono o tablet, alimentamos nuestro circuito que se detalla en la figura siguiente:

El LED del modulo Bluetooth parpadeara rápidamente, indicando que espera conexión entrante.

Encendemos el Bluetooth del movil, ejecutamos la APP, buscamos con el primer botón las conexiones cercanas, seleccionamos la de nuestro modulo (Es importante recordar el nombre que le dimos en la configuración del mismo). El LED parpadeara pero a mas baja velocidad.



Cada vez que presionemos algunos de los botones, podemos observar en el Monitor Serial.



Segundo ejemplo **SegunBlue**

Se desea realizar un nuevo programa para Arduino y para el Movil que permita encender el LED pero mediante un código específico. Por lo cual reformularemos ambos.

SegunBlue

```
void setup()
{
  Serial.begin(9600); //Iniciar el serial
  pinMode(5, OUTPUT); //Establecer el pin 5 como salida
}

void loop()
{
  if(Serial.available()>=1)
  {

    //Delay para favorecer la lectura de caracteres

    delay(22);

    //Se crea una variable que servirá como buffer
    String bufferString = "";

    /*
     * Se le indica a Arduino que mientras haya datos
     * disponibles para ser leídos en el puerto serie
     * se mantenga concatenando los caracteres en la
     * variable bufferString
     */

    while (Serial.available()>0) {
      bufferString += (char)Serial.read();
    }

    long entrada = bufferString.toInt(); //Se carga lo leído en la variable entrada

    Serial.println(entrada); //Muestro lo que entre

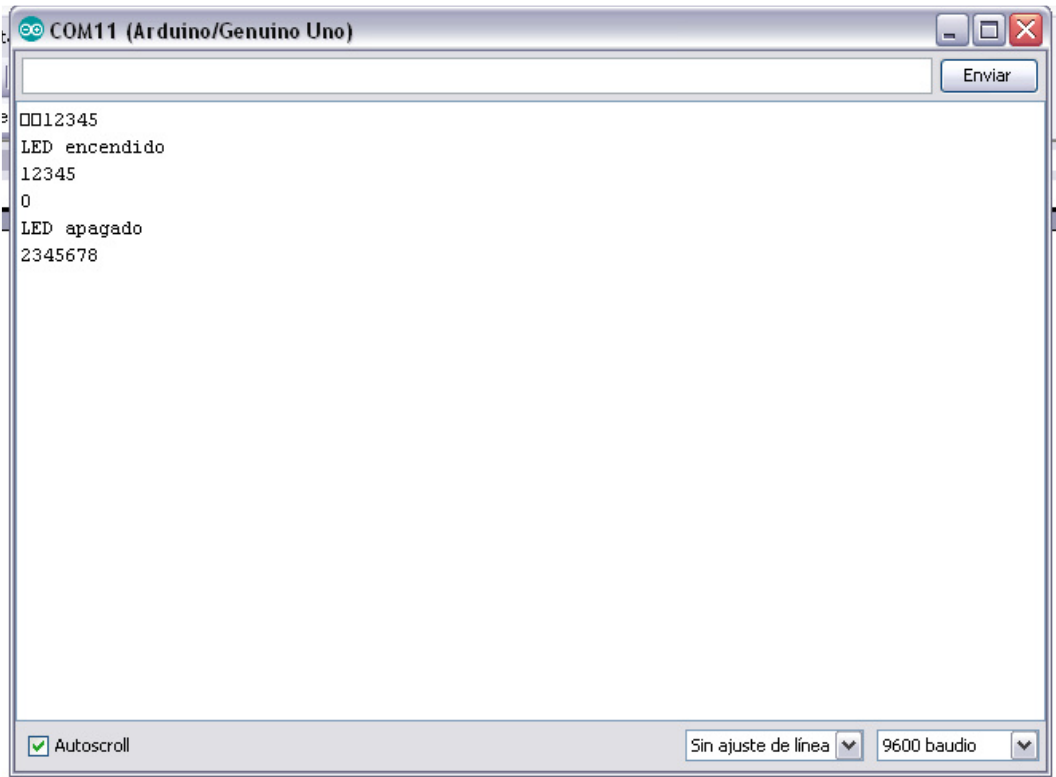
    if(entrada == 12345) //Si es 'codigo', encender el LED
    {
      digitalWrite(5, HIGH);
      Serial.println("LED encendido");
      Serial.println(entrada);
    }

    else if(entrada == 0) //Si es '0', apagar el LED
    {
      digitalWrite(5, LOW);
      Serial.println("LED apagado");
    }

  }
}
```

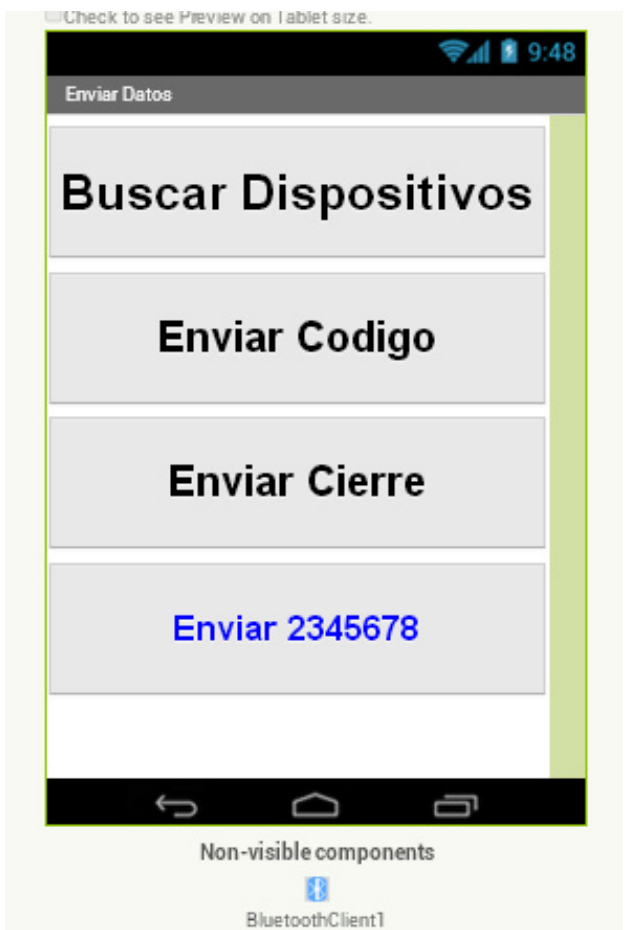
Recuerde desconectar momentáneamente el modulo Bluetooth cuando va a cargar el programa en el Arduino

Este programa permite hacer el seguimiento por el Monitor Serie de Arduino:



Creación de la APP en Android para conectarse a Arduino por medio del modulo Bluetooth **SegunBlue**

Envía un código que reconoce Arduino y enciende LED. Se podría utilizar para activar otros mecanismos e inclusive ampliar a varios. Un botón envía código (12345) que Arduino reconoce y enciende LED, otro código (00000) apaga LED y dado que el programa utilizado en Arduino contempla el seguimiento por Monitor Seria se plantea el envío de otro código que no ejecuta ninguna tarea (2345678).



```

when ListPicker1 .BeforePicking
do
  if BluetoothClient1 . Available
  then set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do
  if BluetoothClient1 . Available
  then set ListPicker1 . Selection to call BluetoothClient1 .Connect
  address ListPicker1 . Selection

when Button1 .Click
do
  if BluetoothClient1 . Available
  then call BluetoothClient1 .SendText
  text " 12345 "

when Button2 .Click
do
  if BluetoothClient1 . Available
  then call BluetoothClient1 .SendText
  text " 00000 "

when Button3 .Click
do
  if BluetoothClient1 . Available
  then call BluetoothClient1 .SendText
  text " 2345678 "

```

Tercer ejemplo **TercerBlue**

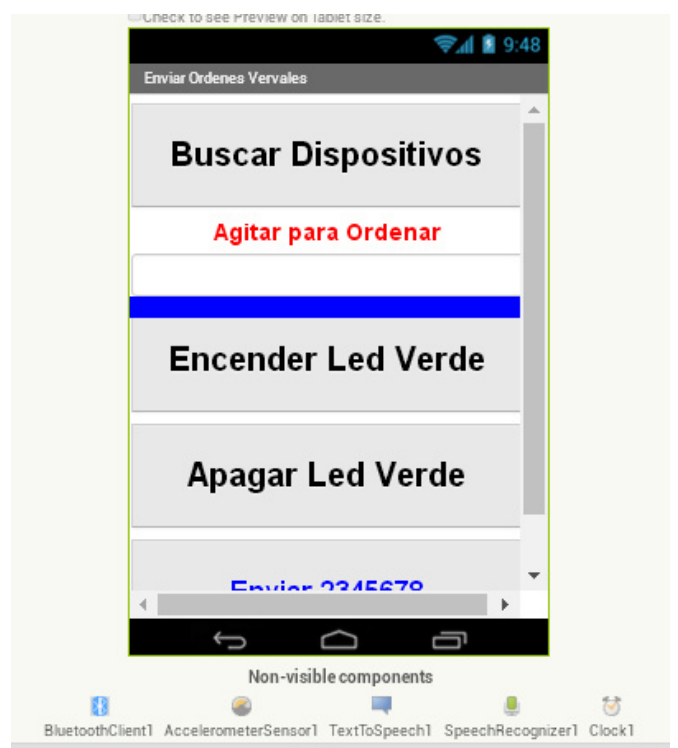
Se desea realizar una nueva versión del programa para Arduino y para el Movil que incluya las funciones del **SegunBlue**, pero que tambien incluya ordenes verbales, que se podran dar luego de agitar el movil (uso del acelerometro).

Las órdenes son:

- **Apagar LED verde**
- **Encender LED verde**
- **Desconectar (cierra el enlace Bluetooth)**

Tener en cuenta que para el reconocimiento de voz y el uso del sintetizador de voz es necesario una conexión a Internet, de lo contrario no se podran usar dichas funciones.

El programa para Arduino sigue siendo el anterior, solo cambia la APP del movil.



```

when ListPicker1 .BeforePicking
do
  if BluetoothClient1 .Available
  then
    set ListPicker1 .Elements to BluetoothClient1 .AddressesAndNames

when ListPicker1 .AfterPicking
do
  if BluetoothClient1 .Available
  then
    set ListPicker1 .Selector to call BluetoothClient1 .Connect
    address ListPicker1 .Selection

when Button1 .Click
do
  if BluetoothClient1 .Available
  then
    call BluetoothClient1 .SendText
    text " 2345 "

when Button2 .Click
do
  if BluetoothClient1 .Available
  then
    call BluetoothClient1 .SendText
    text " 00000 "

when Button3 .Click
do
  if BluetoothClient1 .Available
  then
    call BluetoothClient1 .SendText
    text " 2345078 "

when AccelerometerSensor1 .Shaking
do
  call TextToSpeech1 .Speak
  message " Hola, que deseas hacer "
  call SpeechRecognizer1 .GetText

when SpeechRecognizer1 .AfterGettingText
  result
do
  set OrdenRecibida .Text to get result
  if OrdenRecibida .Text = " encender led verde "
  then
    call TextToSpeech1 .Speak
    message " Encendiendo led verde "
    call BluetoothClient1 .SendText
    text " 12345 "
  else
    if OrdenRecibida .Text = " Apagar led verde "
    then
      call TextToSpeech1 .Speak
      message " Apagando led verde "
      call BluetoothClient1 .SendText
      text " 0 "
    else if OrdenRecibida .Text = " desconectar "
    then
      call TextToSpeech1 .Speak
      message " Desconectando enlace, que tenga buen dia. "
      call BluetoothClient1 .Disconnect
    else
      call TextToSpeech1 .Speak
      message " No entendo, repita por favor "
  end if
end if
end do

```


Cambiar el uso de los Pines RX y TX por otros Pines

A través de la librería SoftwareSerial se pueden cambiar los pines RX y TX a otros pines para establecer la comunicación con el Modulo Bluetooth ya que utilizaremos el puerto serie RX TX para comunicación con la PC, ya sea para cargar o depurar el programa o comunicarse via PC con Arduino. Esta librería viene con el IDE de Arduino, solo debemos incluirla en nuestro programa.

```
//A través de la librería SoftwareSerial
//se pueden cambiar los pines RX y TX a otros pines
//para establecer la comunicación con el Modulo Bluetooth
//ya que utilizaremos el puerto serie RX TX para comunicacion
//con la PC, ya sea para cargar o depurar el programa o comunicarse
//via PC con Arduino

#include <SoftwareSerial.h>
SoftwareSerial BT(2,3); // Cambio RX | TX para conectar Modulo Bluetooth
                        //en pines 2 y 3 (yo elijo llamarlo BT)

long bps=9600; // Como comodidad para definir la velocidad de la comunicacion

void setup()
{
  Serial.begin(bps); //Iniciar el serial para Monitor serial
  BT.begin(bps); //Iniciar serial para Modulo BT

  pinMode(5, OUTPUT); //Establecer el pin 5 como salida
}

void loop()
{
  if(BT.available()>=1) // Me refiero a la comunicacion con Modulo BT
  {

    //Delay para favorecer la lectura de caracteres

    delay(22);

    //Se crea una variable que servirá como buffer
    String bufferString = "";

    /*
     * Se le indica a Arduino que mientras haya datos
     * disponibles para ser leídos en el puerto serie
     * se mantenga concatenando los caracteres en la
     * variable bufferString
     */

    while (BT.available()>0) {
      bufferString += (char)BT.read();
    }

    long entrada = bufferString.toInt(); //Se carga lo leído en la variable entrada
```

```
Serial.println(entrada);//Muestro contenido de variable entrada

if(entrada == 12345) //Si es 'codigo', encender el LED
{
  digitalWrite(5, HIGH);
  Serial.println("LED encendido");
  Serial.println(entrada);
}

else if(entrada == 0) //Si es 'L', apagar el LED
{
  digitalWrite(5, LOW);
  Serial.println("LED apagado");

}

}
}
```

Circuito con los cambios de pines.

