



Laboratorio de  
Sistemas Embebidos



**FACULTAD  
DE INGENIERIA**

Universidad de Buenos Aires

# Carrera de Especialización en Sistemas Embebidos (CESE) de la FI-UBA

## 8. Protocolos de comunicación en Sistemas Embebidos

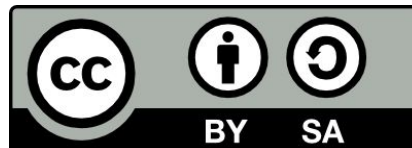
***APP Inventor - Hola Mundo con HC05 o HC06***

**Profesores:**

- Esp. Ing. Eric Pernia.
- Dr. Ing. Pablo Gómez.

## **“APP Inventor - Hola Mundo!”**

por Esp. Lic. Agustin Bassi, se distribuye bajo una Licencia  
Creative Commons Atribución-CompartirIgual 4.0  
Internacional.





# Introducción

El objetivo será crear una aplicación Android que tenga comunicación con la EDU-CIAA-NXP mediante el módulo Bluetooth HC05 o HM10.

Los materiales necesarios serán.

- EDU-CIAA-NXP.
- Módulo HC05 o HM10.
- Cables de conexión entre EDU CIAA y HC05.
- Cuenta creada en APP Inventor.
- Smartphone con Android.

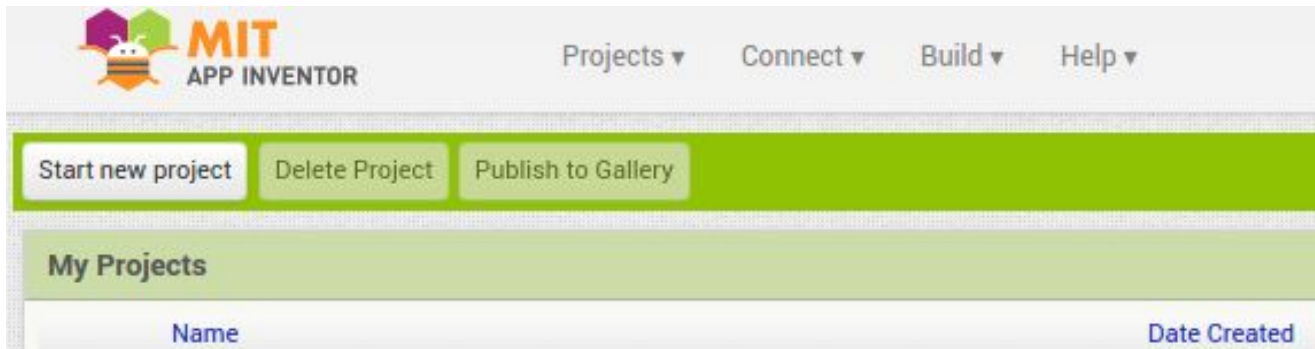
```
54 int main(void){
55     uint8_t dataBluetooth;
56
57     // Inicializar la placa
58     boardConfig();
59     // Configura la UART 232 a 9600 Baudios.
60     uartConfig(UART_232, 9600);
61     // Escribe un mensaje de bienvenida
62     uartWriteString(UART_232, "Hola mundo APP Inventor!\n\r");
63
64     while(1) {
65         // Se queda esperando que llegue informacion por bluetooth
66         if (uartReadByte(UART_232, &dataBluetooth)){
67             if (dataBluetooth == 'h'){
68                 gpioWrite(LED_B, HIGH);
69                 uartWriteString(UART_232, "LED_B encendido.\n\r");
70             } else if (dataBluetooth == 'l'){
71                 gpioWrite(LED_B, LOW);
72                 uartWriteString(UART_232, "LED_B apagado.\n\r");
73             }
74             gpioToggle(LED_R);
75             delay (1000);
76         }
77     }
78     return 0 ;
79 }
```



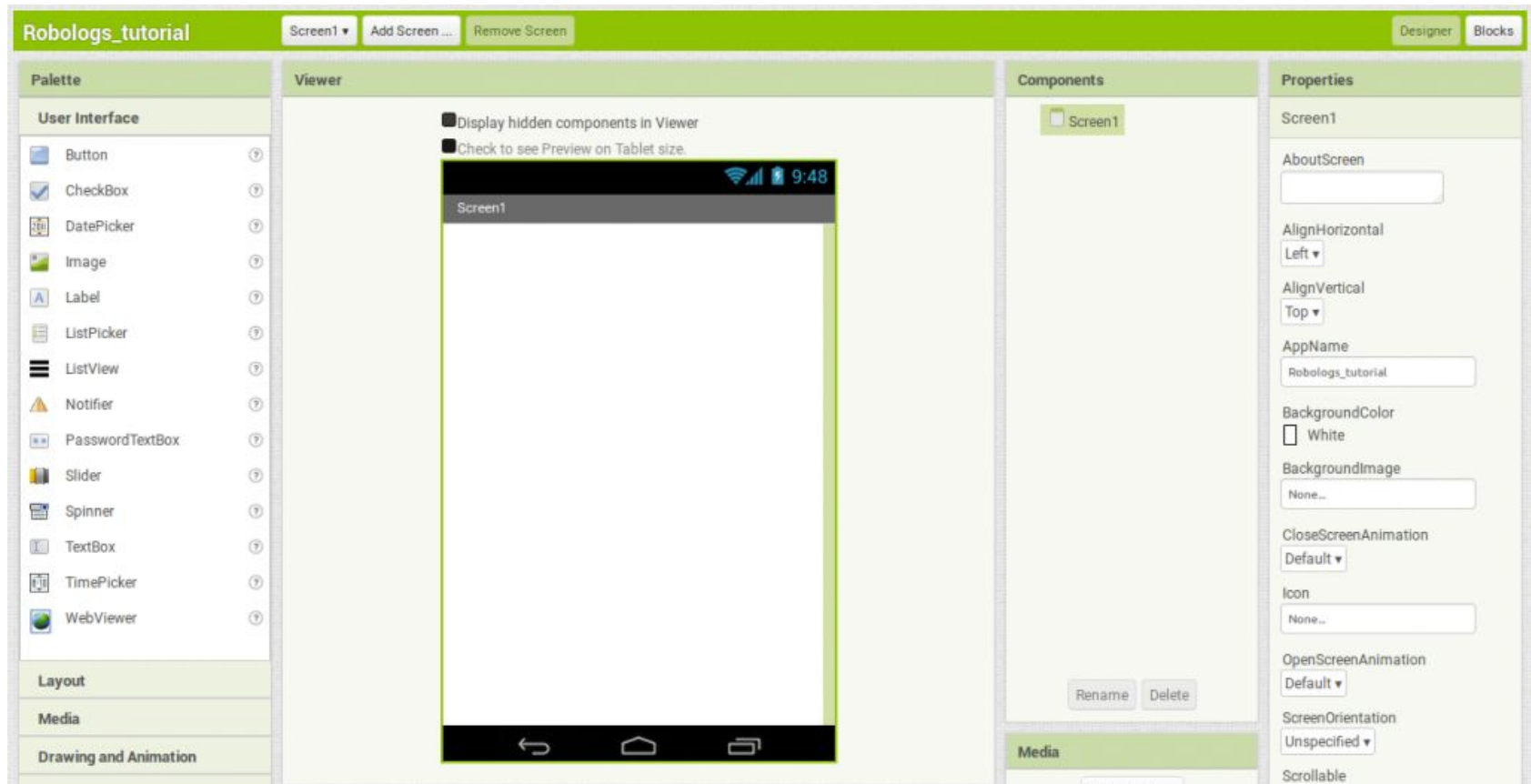
# Hola Mundo App Inventor 2!

Comenzar creando un proyecto nuevo con el botón “Comenzar nuevo proyecto”.

Ingresar como nombre Estación Metereológica.

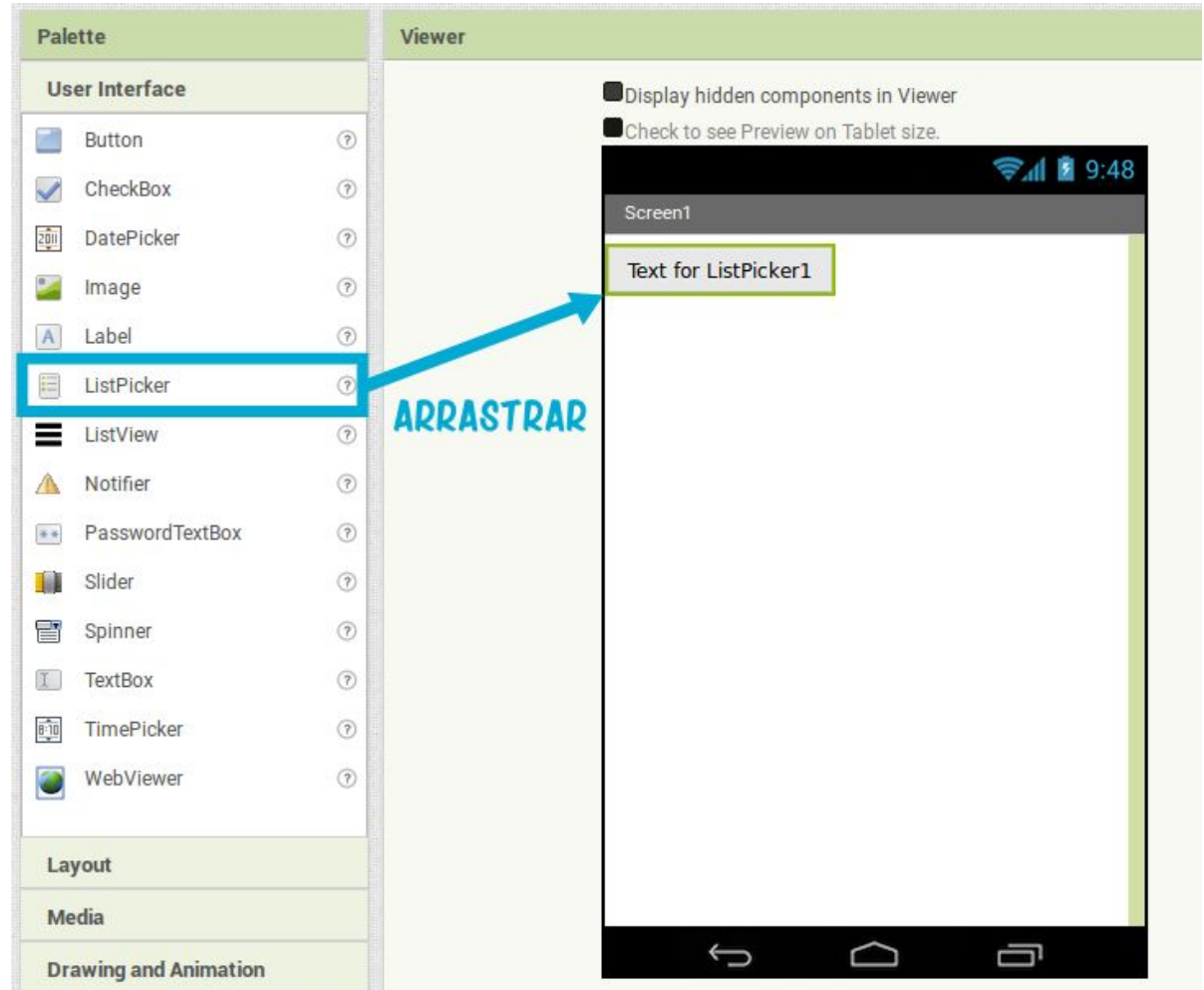


Al iniciar el proyecto, se verá una ventana como esta.



Empezar yendo a **Palette** y buscar el elemento **List Picker**.

Se trata de menú desplegable, que se programa para que muestre los dispositivos Bluetooth cercanos y se conecte a uno. Arrastrarlo hasta la pantalla.



A la derecha, en el menú **Properties**, cambiar el texto del **List Picker**. Escribir “Dispositivos”.



Volver al menú **Palette** y añadir dos elementos de tipo **Button** a la aplicación, de la misma forma que se colocó el **List Picker**.

Seleccionar cada uno de los botones y cambiarles el nombre. A uno llamar “Encender LED” y al otro “Apagar LED”. Con un botón se encenderá el LEDB y con el otro se apagará.





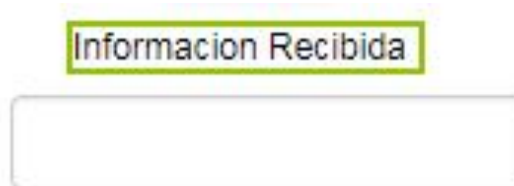


# Hola Mundo App Inventor 2!

La comunicación entre la APP y el sistema embebido debe ser bidireccional. Se deben agregar los objetos destinados a la recepción de datos.

Agregar desde el menú **Palette->User Interface** un objeto del tipo **Label** y en sus propiedades escribirle el texto “Información Recibida”.

Como último objeto visual, debajo del **Label** agregar un **TextBox**.



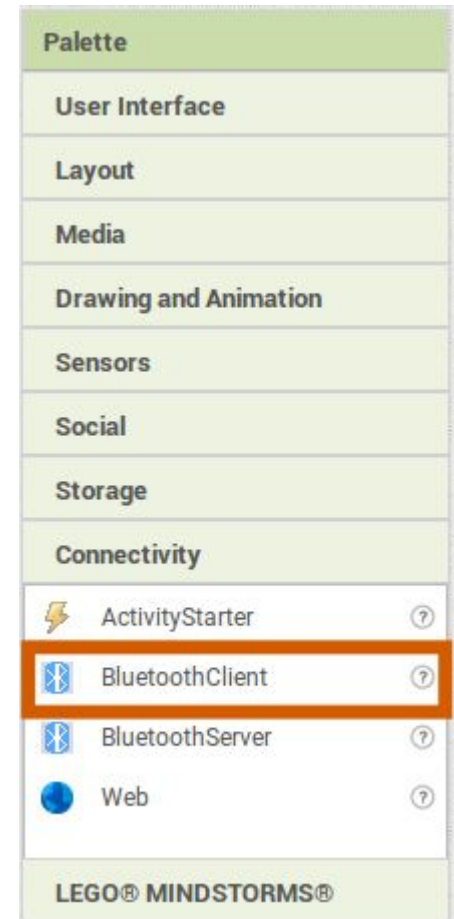


# Hola Mundo App Inventor 2!

Lo que se ha añadido hasta ahora son componentes visibles. El usuario puede verlos e interactuar con ellos. Pero hay otro tipo de componentes, llamados *non-visible components*, que sirven para activar algunas funcionalidades como el Giroscopio, la antena WiFi o Bluetooth.

Para poder conectar con la CIAA se necesita que la aplicación pueda usar la antena Bluetooth.

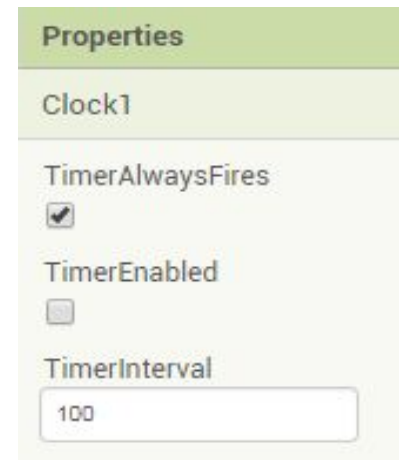
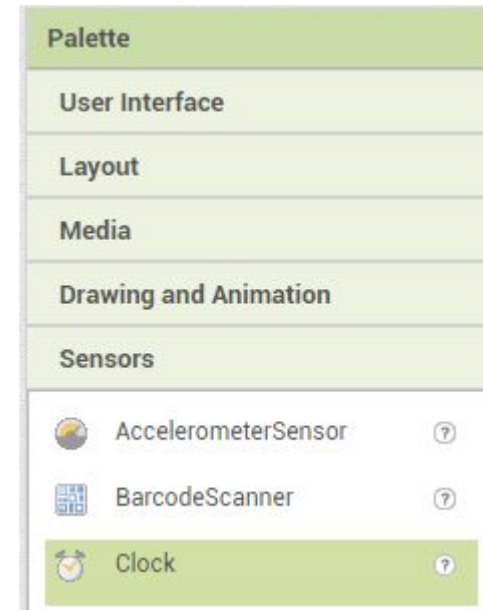
Ir a **Palette->Connectivity** y arrastrar **Bluetooth Client** a la pantalla.



El último objeto necesario a agregar es un timer que chequee si hay información disponible proveniente del sistema embebido.

Esto se realiza desde **Palette->Sensors** y arrastrando un objeto **Clock**.

En las propiedades del clock setear que se dispare cada 100 ms y que arranque deshabilitado.



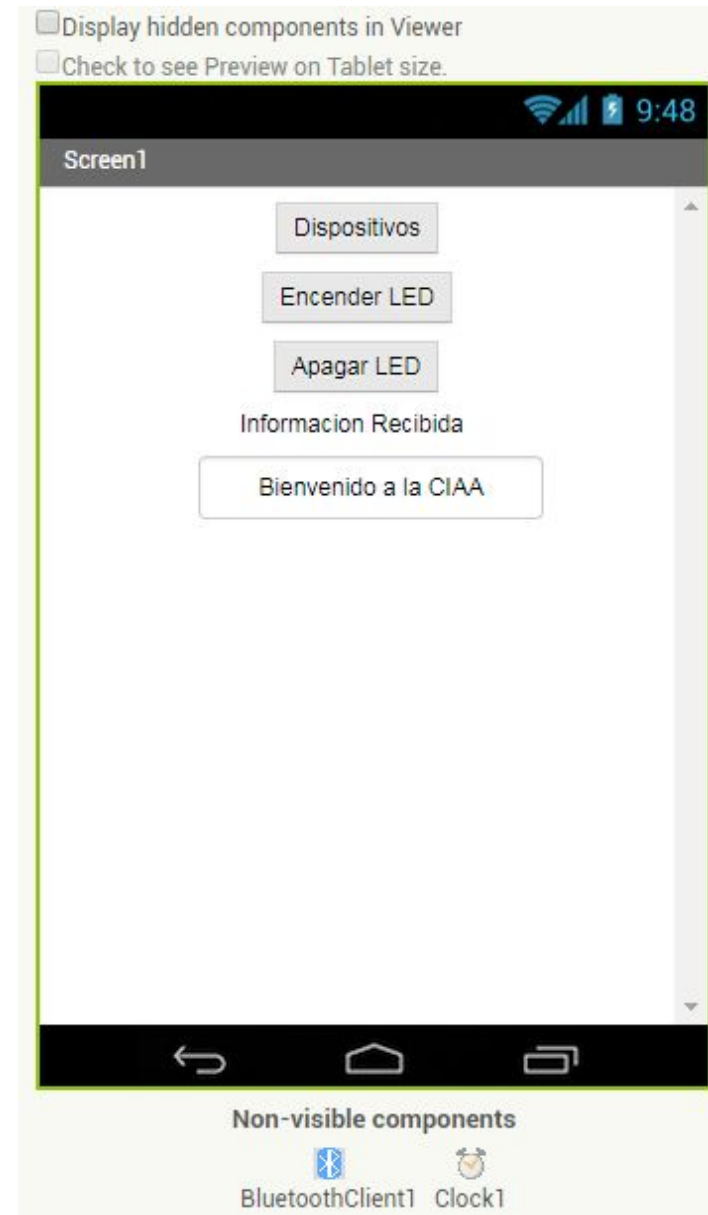


# Hola Mundo App Inventor 2!

A esta altura, la aplicación debería lucir como en la imagen.

Hasta aquí el diseño. Es hora de configurar cada uno de los componentes.

Arriba a la derecha hay dos botones juntos: **Design** y **Blocks**. El primero abre la ventana de diseño, que permite colocar todos los componentes de la aplicación. **Blocks** abre una ventana para programar los bloques.



# Hola Mundo App Inventor 2!

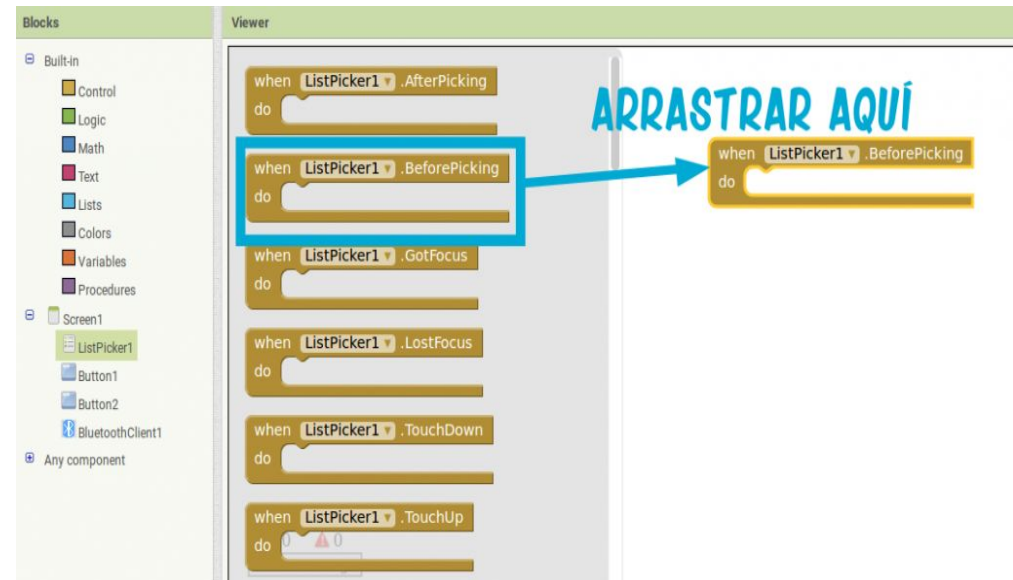
Lo primero es programar el menú desplegable. A la derecha, seleccionar **ListPicker1** y se abrirá un menú en el que aparecen los bloques relacionados con este objeto.

Hay que programar en el desplegable dos cosas: qué debe hacer *ANTES* de seleccionar una opción del menú desplegable y *DESPUÉS* de seleccionar una.

Antes de seleccionar una opción, inicializar una lista con los nombres de todos los dispositivos Bluetooth que hay cerca.

Si el móvil tiene la comunicación Bluetooth activada, **ListPicker1** deberá establecer los nombres de los dispositivos cercanos como elementos de la lista.

Dentro del menú de bloques de **ListPicker1** arrastrar. **BeforePicking** al espacio en blanco.



Ir al apartado Control y seleccionar el condicional **if**.



Después seleccionar el objeto **BluetoothClient1** y añadir un **AdressesAndNames** y **Available**.

BluetoothClient1 . AddressesAndNames

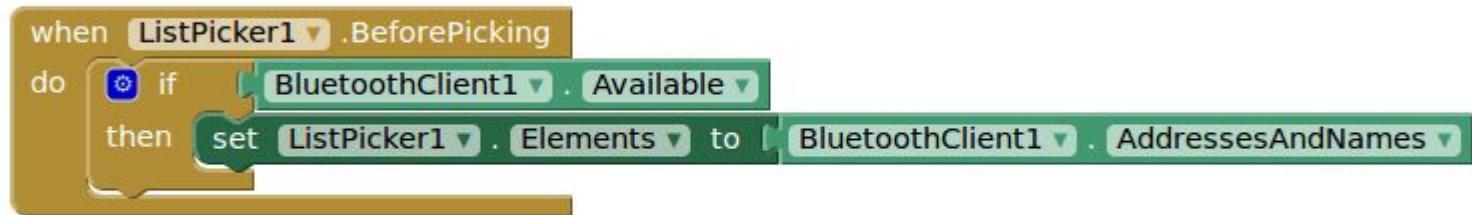
BluetoothClient1 . Available

Finalmente seleccionar ListPicker1 y añadir un **Elements To**.

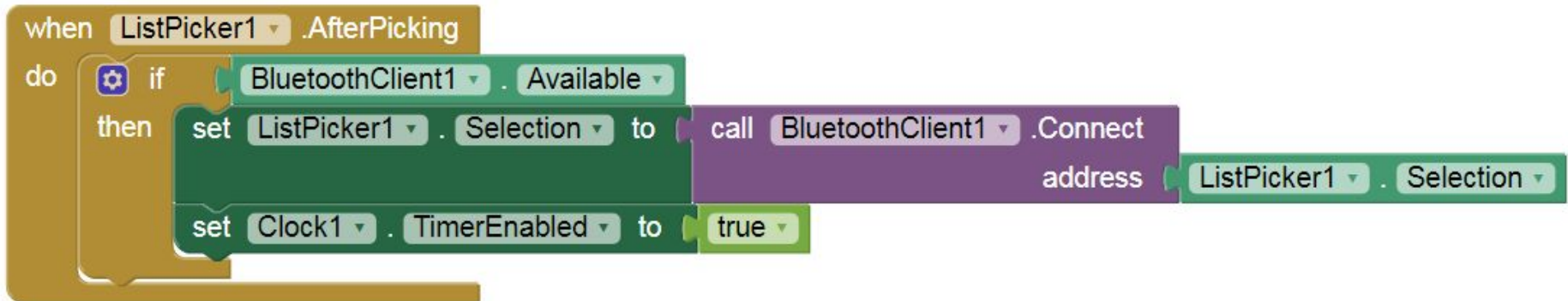
set ListPicker1 . Elements to

# Hola Mundo App Inventor 2!

Montar todos los bloques anteriores para que queden así.



Luego programar a ListPicker1 para que se conecte a la dirección que el usuario seleccione en la lista. Una vez conectado, habilitar el timer para recibir información desde el sistema embebido. El resultado final debería ser así.





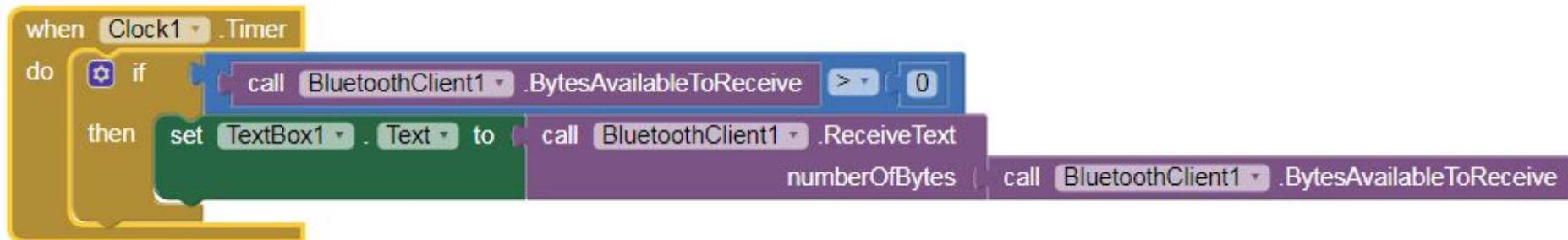
# Hola Mundo App Inventor 2!

Al inicio de la aplicación deshabilitar el campo de texto donde se va a mostrar la info recibida así no se puede escribir en ese campo.



```
when Screen1.Initialize
do
  set TextBox1.Enabled to false
```

Si la aplicación recibe un dato enviado desde el sistema embebido lo debe mostrar en el campo de texto. Para chequear si la aplicación recibió datos, es necesario chequear periódicamente mediante el timer si hay datos disponibles en el objeto de conexión **BluetoothClient1**

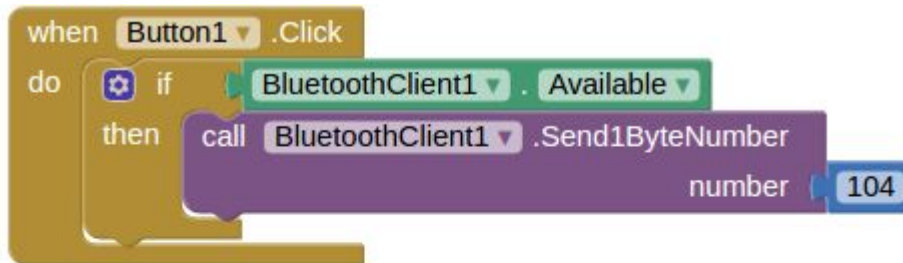


```
when Clock1.Timer
do
  if call BluetoothClient1.BytesAvailableToReceive > 0
  then
    set TextBox1.Text to call BluetoothClient1.ReceiveText
    numberOfBytes call BluetoothClient1.BytesAvailableToReceive
```

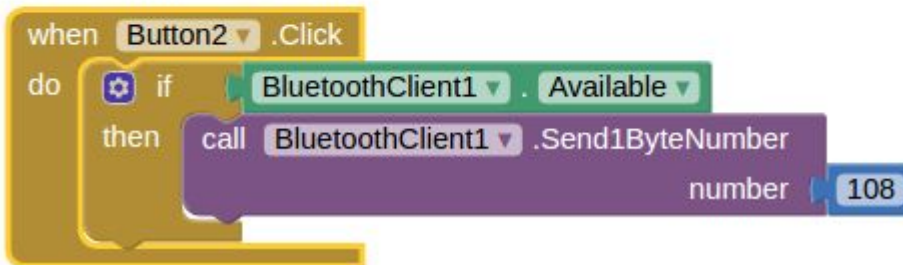


# Hola Mundo App Inventor 2!

Finalmente configurar Button1 y Button2 para que al pulsarlos se envíe un texto a la dirección Bluetooth conectado.

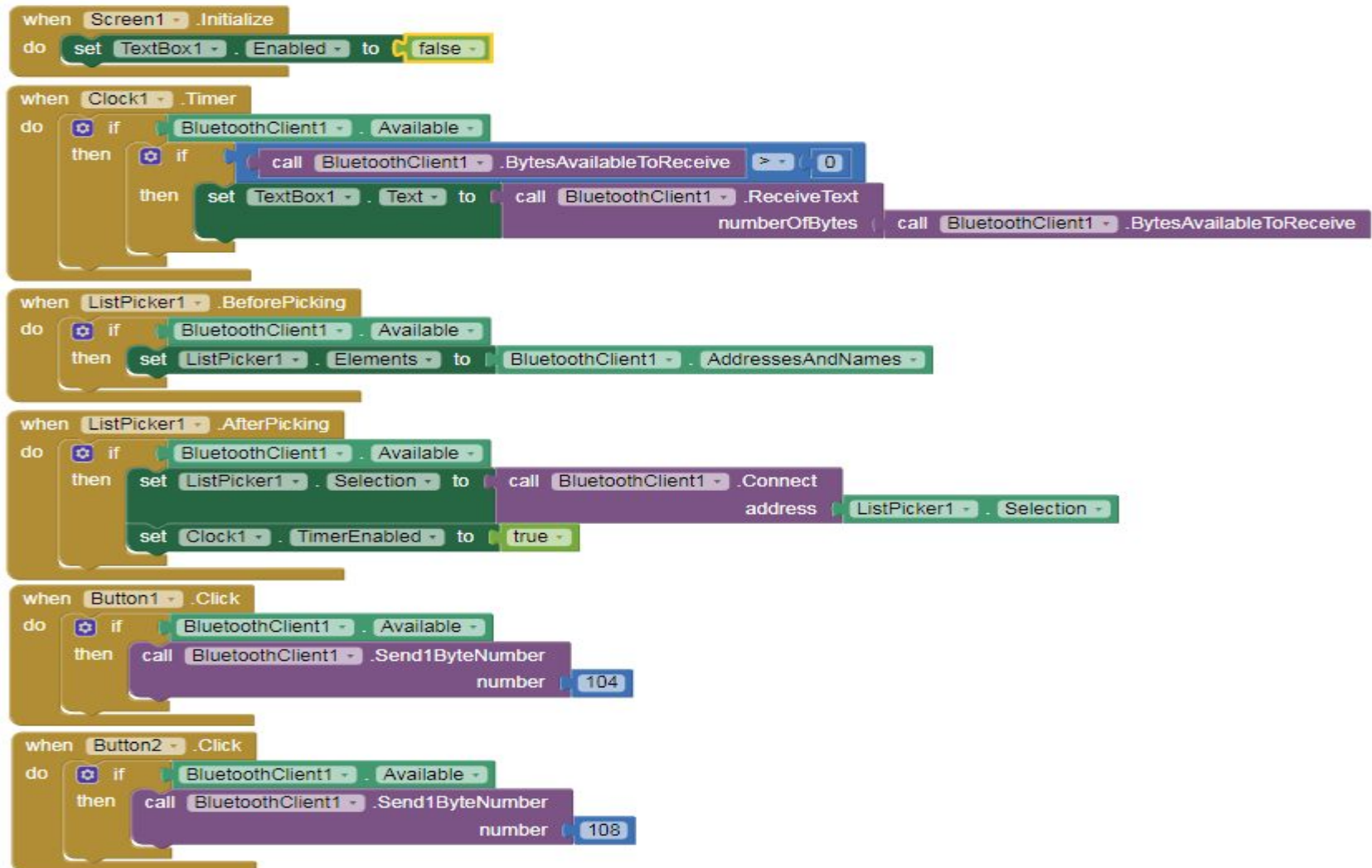


Corresponde al  
ASCII 'h'



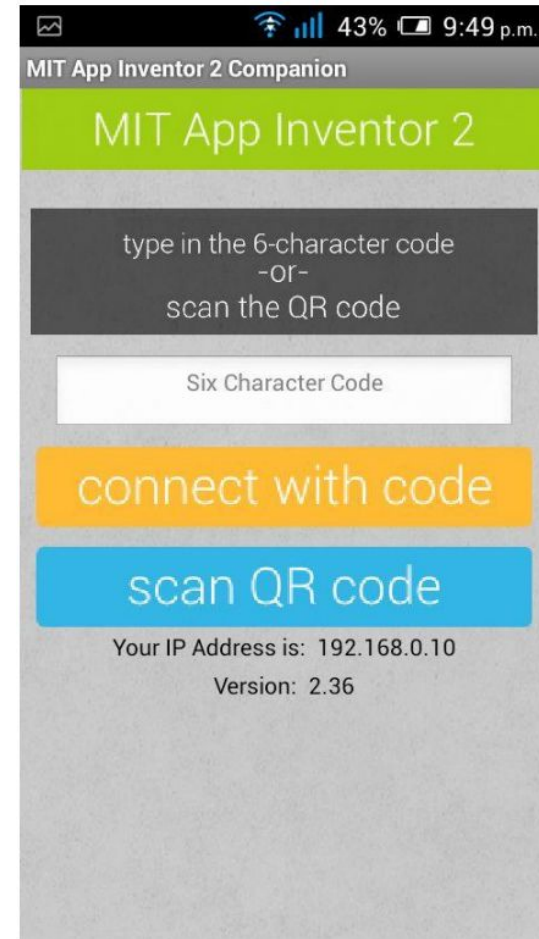
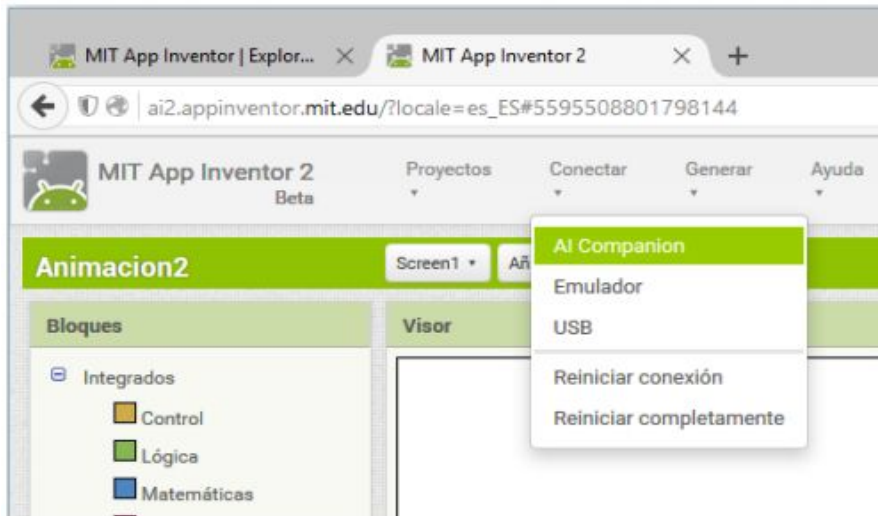
Corresponde al  
ASCII 'i'

Al final, el programa debería verse así:



# Hola Mundo App Inventor 2!

Escanear el código QR desde la APP Inventor en el smartphone y probar la aplicación.



**Recordar habilitar el Bluetooth en el smartpone antes de escanear el código QR**

# ¿Preguntas?





Laboratorio de  
Sistemas Embebidos



**FACULTAD  
DE INGENIERIA**

---

Universidad de Buenos Aires

**¡Muchas gracias!**

Consultas:

[ericpernia@gmail.com](mailto:ericpernia@gmail.com)

[elpablogomez@gmail.com](mailto:elpablogomez@gmail.com)

[sergiordj@gmail.com](mailto:sergiordj@gmail.com)

- <http://ai2.appinventor.mit.edu/>
- <http://www.bolanosdj.com.ar/MOVIL/LENGUAJES/Usando-MIT-App-Inventor-2.pdf>
- <http://codeweek.eu/resources/spain/guia-iniciacion-app-inventor.pdf>
- <https://robologs.net/2015/10/29/tutorial-de-arduino-bluetooth-y-android-2-crear-una-app-con-mit-inventor/>