Manual de Usuario

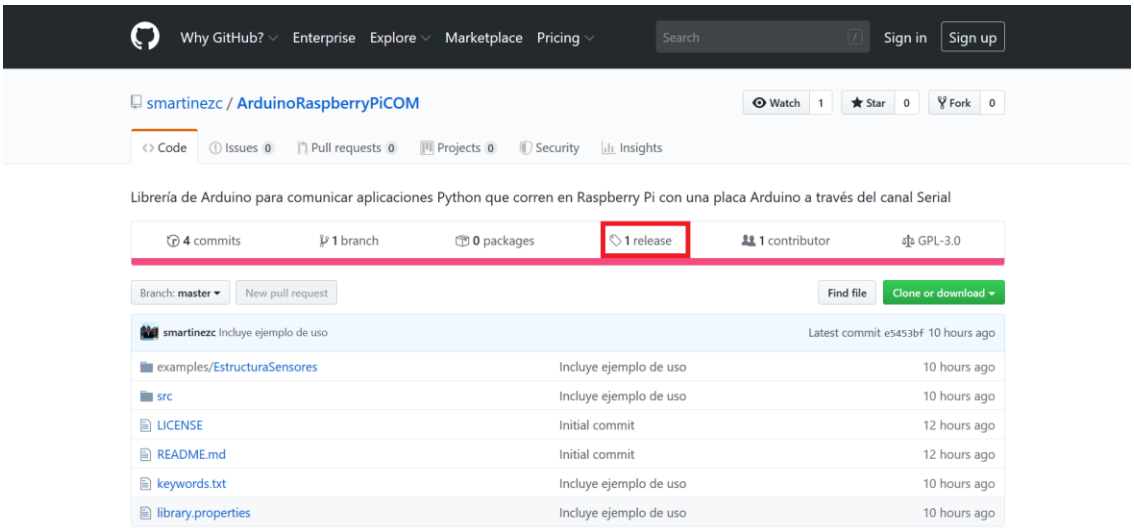
Librería ArduinoRaspberryPiCOM

(v0.6)

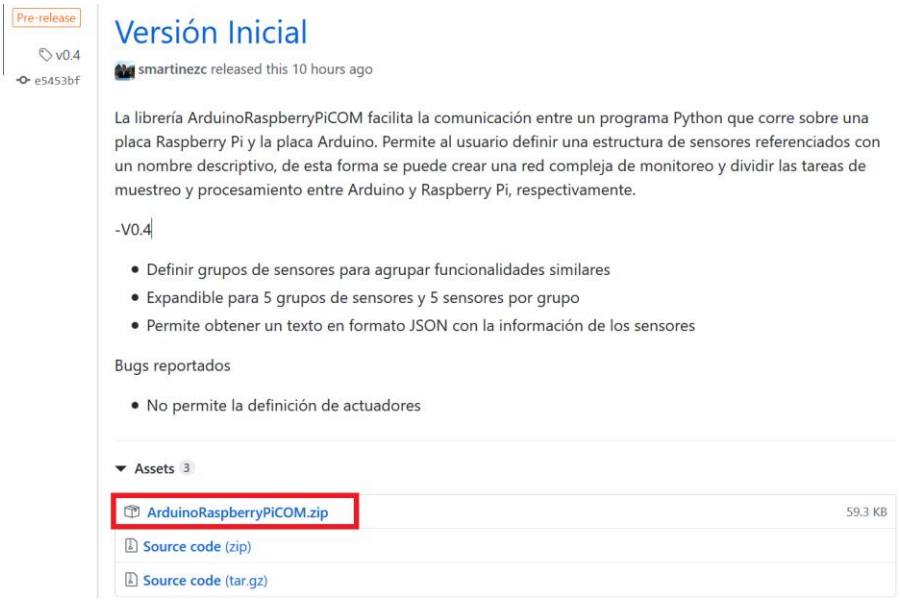
1. Guía de instalación

Puede obtener la librería de Arduino del siguiente enlace, siga los pasos mencionados para la correcta instalación: https://github.com/smartinezc/ArduinoRaspberryPiCOM

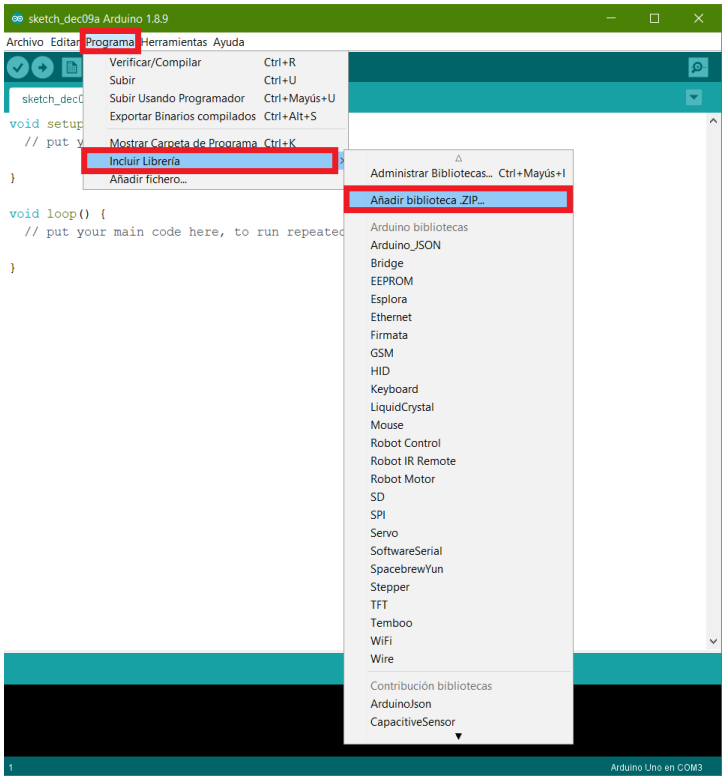
Diríjase a la pestaña de "Lanzamientos"



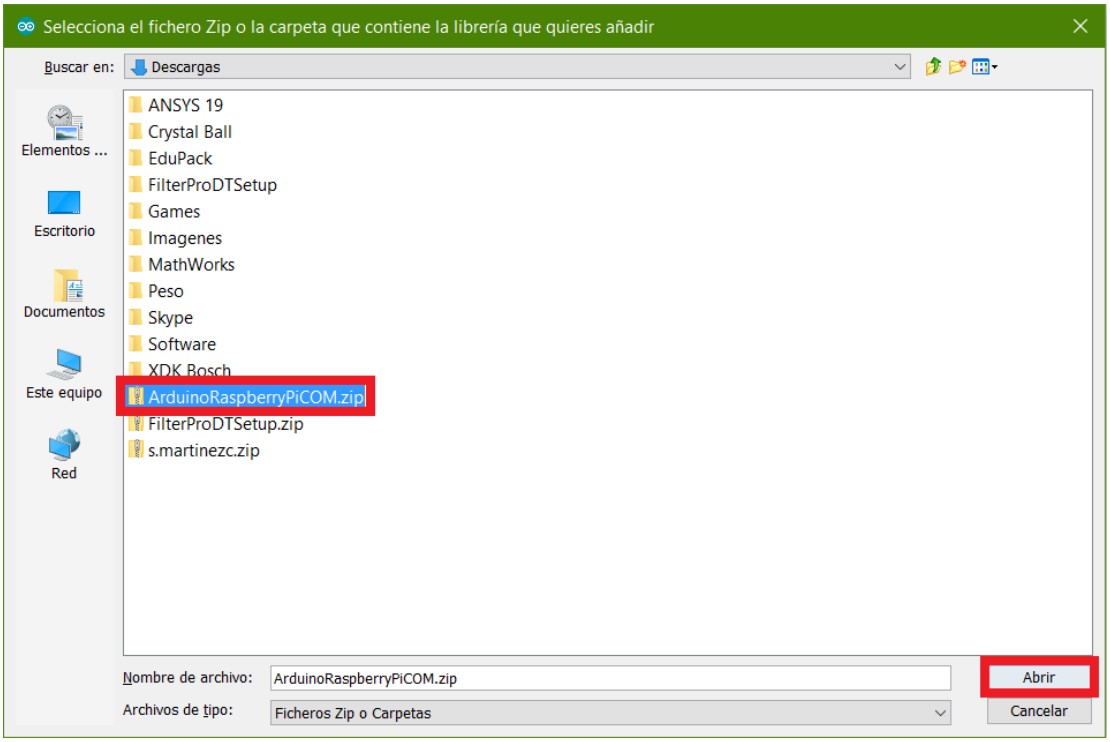
Busque la versión más reciente de la librería y descargue la carpeta .zip



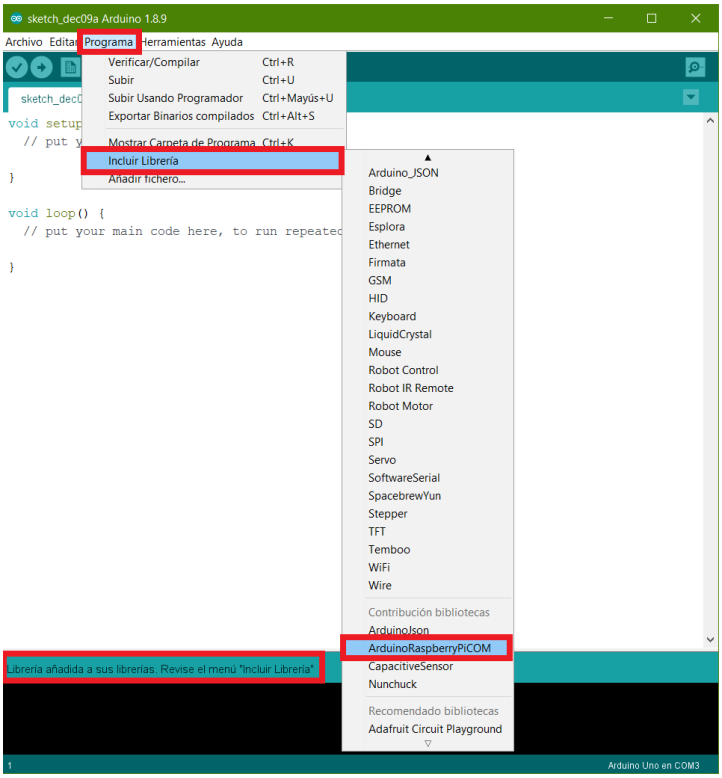
En el IDE de Arduino, seleccione la pestaña “Programa” > “Incluir Librería” > “Añadir biblioteca .ZIP…”



En la ventana de selección de archivos ubique la carpeta “ArduinoRaspberryPiCOM.zip” que descargó del enlace, luego seleccione Aceptar



La librería ha sido instalada correctamente si en el IDE de Arduino aparece el mensaje “Librería añadida a sus librerías. Revise el menú ‘Incluir librería’” y el nombre de la librería aparece en “Programa” > “Incluir Librería”



2. Uso de la librería ArduinoRaspberryPiCOM

Para familiarizarse con el uso de la librería se recomienda revisar el código de ejemplo, el cual se explica en la sección 2.1 de este documento. En la sección 2.2 se dará una definición de los Constructores y Métodos disponibles en la librería a manera de API

2.1 Código ejemplo

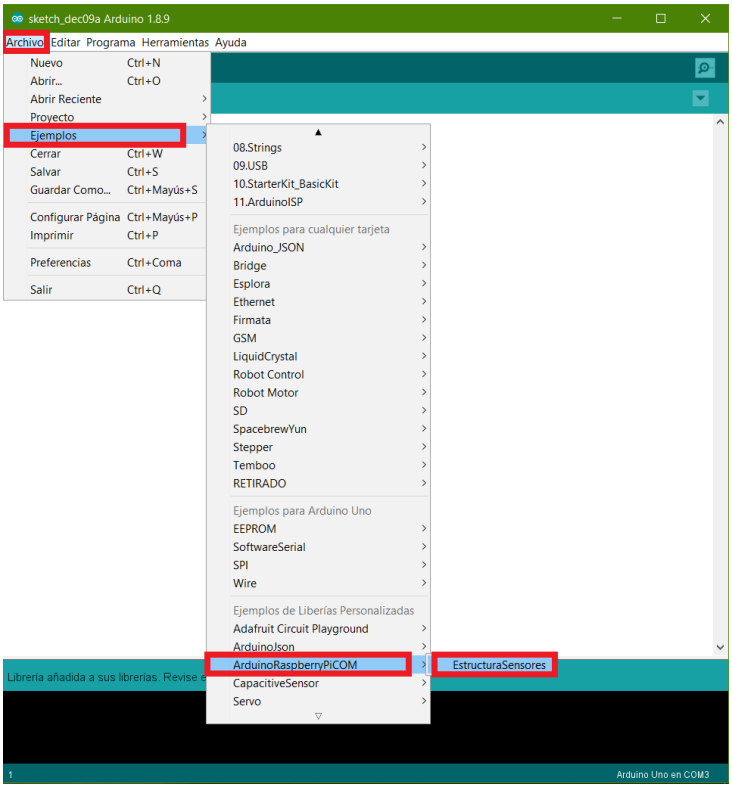
Se recomienda revisar el código ejemplo incluido en la librería para comprender el uso de las funciones definidas y la aplicación de estas. En la mayoría de los casos que se quiere emplear un Arduino para la recopilación de información obtenida de sensores y el procesamiento de estas en Raspberry Pi, el código ejemplo incluye todo lo necesario para realizar esta tarea.

En el código “EstructuraSensores.ino” se da un ejemplo del uso de la librería para obtener un texto en formato JSON con una estructura de sensores agrupados por similitudes, en el ejemplo se crea un grupo de sensores llamado “Humedad” al cual se agregan los sensores “Sensor de Humedad 1” y “Sensor de Humedad 2”, cada uno

con el pin al cual están conectados definido al inicio del código. Note que NO SE REQUIERE DEFINIR LOS PINES COMO ENTRADAS/SALIDAS, la librería lo hace.

Posteriormente se recopila la información medida por los sensores cada intervalo de tiempo definido como variable al inicio del código, se obtiene un String con el texto en formato JSON que contiene esta información y es enviado a través del canal serial (USB) hacia un Raspberry Pi conectado, el cual puede realizar cualquier procesamiento deseado sobre la información obtenida.

Para abrir y editar el código ejemplo seleccione la pestaña “Archivo” > “Ejemplos” > “ArduinoRaspberryPiCOM” > “EstructuraSensores”



2.2 Definición de Métodos disponibles

A continuación, se dará una breve descripción de los Métodos y el uso de la librería

Incluir Librería

Para incluir la librería en el código y poder hacer uso de esta debe instalar la librería (Sección 1 de este documento) y después incluir al inicio del código la siguiente línea

```
#include <ArduinoRaspberryPiCOM.h>
```

O puede hacer uso del asistente de inclusión en el IDE de Arduino desde “Programa” > “Incluir Librería” > “ArduinoRaspberryPiCOM”

Constructor

ArduinoRaspberryPiCOM arCOM = **ArduinoRaspberryPiCOM** ();

Crea una variable, llamada arCOM en este ejemplo, que representa el objeto encargado de la gestión de sensores conectados a Arduino

Métodos

A continuación, se da una breve definición de cada método disponible en la librería ArduinoRaspberryPiCOM, los parámetros obligatorios y el retorno si lo tiene

arCOM.crearGrupoSensor(String nombreGrupo, int tipoSeñal);
Crea un grupo de sensores que puede contener hasta 5 sensores, todos los sensores de este grupo deben ser del mismo tipo de señal (ANALÓGICO o DIGITAL)
nombreGrupo – **String por el que se identifica el grupo de sensores (“Humedad”)** *tipoSeñal* – **int Constante que representa el tipo de señal del sensor conectado al Arduino** (*arCOM.DIGITAL, arCOM.ANALOGO*)

arCOM.agregarSensor(String nombreSensor, String nombreGrupo, int pinSensor);
Crea un sensor dentro del grupo especificado por parámetro, se debe especificar el pin al que está asociado el sensor según el tipo de señal (ANALÓGICO o DIGITAL)
nombreSensor – **String por el que se identifica el sensor (“SenHum1”)** *nombreGrupo* – **String por el que se identifica el grupo de sensores (“Humedad”)** *pinSensor* – **int Número del pin digital o analógico al que se conectó el sensor para tomar la lectura** (0, 1, 2, … – *A0, A1, A2, …*)

arCOM.crearGrupoActuador(String nombreGrupo, int tipoSeñal);
Crea un grupo de actuadores que puede contener hasta 5 actuadores, todos los actuadores de este grupo deben ser controlados por el mismo tipo de señal (DIGITAL o PWM)
nombreGrupo **– String por el que se identifica el grupo de actuadores (“Bombas”)** *tipoSeñal* **– int Constante que representa el tipo de señal con la cual se controla el actuador conectado al Arduino** *(arCOM.DIGITAL, arCOM.PWM)*

arCOM.agregarActuador(String nombreActuador, String nombreGrupo, int pinActuador);
Crea un actuador dentro del grupo especificado por parámetro, se debe especificar el pin al que está asociado el actuador según el tipo de señal (DIGITAL o PWM)
nombreActuador **– String por el que se identifica el actuador (“Bomba de Agua”)** *nombreGrupo* **– String por el que se identifica el grupo de actuadores (“Bombas”)** *pinActuador* **– int Número del pin digital o PWM al que se conectó el actuador** *(0, 1, 2, …)*

arCOM.leerSensores();
Recorre la lista de sensores y actualiza el valor medido de éstos. Cuando se emplea el método *darJSON()* **se crea un JSON con los valores al momento de la última llamada del método** *leerSensores()*

arCOM.actualizarActuadores();
Recorre la lista de actuadores y actualiza el valor de la salida, en el pin especificado para cada uno, según la última lectura del JSON.

arCOM.darJSON();
Recorre la lista de sensores y actuadores, crea un String con formato JSON que contiene la información del valor medido por los sensores y el estado actual de los actuadores.
Retorno **– String con formato JSON con los valores medidos por los sensores y el estado actual de los actuadores.**

arCOM.leerJSON(String json,String nombreABuscar);
Recorre el JSON dado por parámetro como un String en busca del Sensor o Actuador especificado como segundo parámetro, retorna el valor de este.
json **– String con formato JSON del que se desea extraer la información** *nombreABuscar* **– String por el que se identifica el sensor o actuador que se desea buscar**
Retorno **– int que representa el valor medido por el sensor o el estado actual del actuador especificado por parámetro.**