```
#*-*coding:utf-8*-*
#qpy:kivy

#*-*coding:utf-8*-*
#Desarrollado por Diego Alberto Parra Garzón
#Bogotá D.C., Colombia
#noviembre 2015
#Esto es software libre con licencia GPL3
#qpy:kivy
#clase Bluetooth Arduino referencia de la api de pyjnius
#dparra@opesai.org

from jnius import autoclass

class ArduinoBluetooth:

    def obtenerCorrienteEnchufe(self, Nombre):
        conectar_dispositivo = self.AdaptadorBluetooth.getDefaultAdapter
().getBondedDevices().toArray()
        self.enchufe = None
        for dispositivo in conectar_dispositivo:
            if dispositivo.getName() == Nombre:
                self.enchufe = dispositivo.createRfcommSocketToServiceRecord
(self.UUID.fromString("00001101-0000-1000-8000-00805F9B34FB"))
                #   self.recibir = self.enchufe.getInputStream()
                self.recibir = self.Lecturabufer(self.EntradaDeDatos
(self.enchufe.getInputStream()))
                self.enviar = self.enchufe.getOutputStream()
                self.enchufe.connect()
                self.conexion = True
                print "paso la conexion en la clase"
        return self.conexion


    def Escribir(self, Mensaje, *args):
        if self.conexion == True:
            self.enviar.write(Mensaje)
            print "Mensaj enviado"
        else:
            print "Dispositivo no esta conectado"

    def LeerNUMERO(self, *args):
        CadenaDatos = ""
        if self.conexion == True:
            print "revisando conexion"
            print self.conexion
            lectura =1
            if (lectura>0):
                CadenaDatos =  float(self.recibir.readLine())
                print "Lectura correcta ", CadenaDatos
        return CadenaDatos


    def LeerCADENA(self, *args):
        CadenaDatos = ""
        if self.conexion == True:
            print "revisando conexion"
            print self.conexion
            lectura =1
            if (lectura>0):
                CadenaDatos =  str(self.recibir.readLine())
                print "Lectura correcta ", CadenaDatos
        return CadenaDatos

```

```
62      def Cerrar(self):
63          if self.conexion:
64              self.enchufe.close()
65              print "Dispositivo cerrado"
66
67      def __init__(self):
68          self.AdaptadorBluetooth = autoclass('android.bluetooth.BluetoothAdapter')
69          self.DispositivoBluetooth = autoclass('android.bluetooth.BluetoothDevice')
70          self.enchufe_Bluetooth = autoclass('android.bluetooth.BluetoothSocket')
71          self.UUID = autoclass('java.util.UUID')
72          self.Lecturabufer = autoclass('java.io.BufferedReader')
73          self.EntradaDeDatos = autoclass('java.io.InputStreamReader')
74          self.conexion = False
75
76
77      def __del__(self):
78          print "destructor de la clase ArduinoBluetooth"
79
80  #Arduino = ArduinoBluetooth()
81  #Arduino.obtenerCorrienteEnchufe("HC-05")
82  #Arduino.Escribir('1')
83  #print Arduino.Leer()
```