BOTBLOQ: Ecosistema integral para el diseño, fabricación y programación de robots DIY

Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI)
EXPEDIENTE: IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

ACRÓNIMO DEL PROYECTO: BOTBLOQ

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD

CDTI Centro para el Desarrollo Tecnológico Industrial

UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

ENTREGABLE 5.3.2 Módulo de programación de dispositivos móviles Android para robots BOTBLOQ

RESUMEN DEL DOCUMENTO

En este documento especifican los distintos módulos de programación desarrollados para poder integrar los robots de Botbloq con dispositivos móviles.

Índice

Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

1.- Introducción

Ya en anteriores entregables e hitos se desarrolló la librería que permite generar código desde bloques, una forma eficaz y sencilla de que los alumno se centren en la lógica de la programación y no tanto en la sintaxis.
Se puede obtener el código fuente desde aquí: https://github.com/bq/blogs .
Posteriormente se dio soporte a esa librería para generar código python mediante unas reglas similares a las creadas para generar código Arduino. Y durante el hito 2 se crearon bloques de programación para interactuar con la aplicación de Bitbloq Connect.
En este hito, se han estado modificando los bloques generados para incluir las reglas de generación necesarias para que generen el código python necesario. Documentamos para que sirve cada bloque y que es lo que se ha añadido.

2.- Bloques modificados

2.1.- Emitir sonido

Este bloque sirve para que el robot pueda enviar al dispositivo móvil la orden de emitir un sonido. Tiene un listado de sonidos que van integrados en la aplicación.

Función implementada en python:

```
def emitir_sonido(server_sock, sonido ):
  server_sock.send("playSound-%s\n" % sonido)
  time.sleep(1)
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'emitir_sonido(server_sock, {SOUND})'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
emitir_sonido(server_sock, "bass")
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

2.2.- Mostrar en la pantalla

Este bloque permite mostrar el mensaje que se manda en la pantalla del propio dispositivo. Requiere de un bloque de texto tradicional.

Función implementada en python:

```
def escribe_texto(server_sock,texto):
  server_sock.send("%s\n" % texto)
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'escribe_texto(server_sock, {DATA})'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
escribe_texto(server_sock, "hola")
```

2.3.- Recibir datos por voz o texto

Este bloque hace que la programación del robot se detenga hasta que se reciben datos del dispositivo. El dispositivo tiene en el interfaz la opción de mandar datos escritos o mediante el micrófono que tenga el dispositivo.

Función implementada en python:

```
def recibe_texto(server_sock):
  data = server_sock.recv(1024)
  return data
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'recibe_texto(server_sock)'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
a3 = recibe_texto(server_sock)
```

2.4.- Encender la linterna del dispositivo

Este bloque manda la orden al dispositivo de encender la linterna (en el caso de que la tenga) y que suele estar colocada donde la cámara del dispositivo ya que suele usarse como flash en las cámaras.

Función implementada en python:

```
def enciende_linterna(server_sock):
  server_sock.send("turnonFlashlight-\n")
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'enciende_linterna(server_sock)'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
enciende_linterna(server_sock)
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

2.5.- Apagar la linterna del dispositivo

Este bloque manda la orden al dispositivo de apagar la linterna (en el caso de que la tenga) .

Función implementada en python:

```
def apaga_linterna(server_sock):
  server_sock.send("turnoffFlashlight-\n")
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'apaga_linterna(server_sock)'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
apaga_linterna(server_sock)
```

2.6.- Leer valores de aceleración

Este bloque detiene la programación del robot hasta que reciba del dispositivo los valores de la aceleración que le piden. Se pueden pedir tanto la aceleración normal, la aceleración lineal o la aceleración de la gravedad en cualquiera de los ejes X, Y o Z.

Función implementada en python:

```
def recibir_aceleracion(server_sock, message, axis):
  server_sock.send("%s%s\n" %(message, axis))
  dato = recibe_texto(server_sock)
  print("aceleration: %s" %dato)
  return dato
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'recibir_aceleracion(server_sock, {MESSAGE}, {AXIS})'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
a1 = recibir_aceleracion(server_sock, "readAccel-", "x")
```

2.7.- Leer los valores dados por el giroscopio

Este bloque detiene la programación del robot hasta que reciba del dispositivo los valores del giroscopio que tiene el dispositivo (si lo tiene, caso contrario devuelve 0 o NaN dependiendo del dispositivo). Se pueden pedir en cualquiera de los ejes X, Y o Z.

Función implementada en python:

```
def recibir_giroscopio(server_sock, axis):
  server_sock.send("readGyros-%s\n" % axis)
  dato = recibe_texto(server_sock)
  print("giroscopio: %s" %dato)
  return dato
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'recibir_giroscopio(server_sock, {AXIS})'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
a4 = recibir_giroscopio(server_sock, "x")
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

2.8.-Leer los valores del campo magnético

Este bloque detiene la programación del robot hasta que reciba del dispositivo los valores del sensor de campos magenitcos que tiene el dispositivo (si lo tiene, caso contrario devuelve 0 o NaN dependiendo del dispositivo). Se pueden pedir en cualquiera de los ejes X, Y o Z.

Función implementada en python:

```
#leer campo magnético
def recibir_campomagnetico(server_sock, axis):
   server_sock.send("readMagnetic-%s\n" % axis)
   dato = recibe_texto(server_sock)
   print("magnetic field: %s" %dato)
   return dato
```

Reglas añadidas al bloque para generar código Python:

```
python: {
      codeLines: [{
         code: 'recibir_campomagnetico(server_sock, {AXIS})'
      }]
   }
```

Código de ejemplo generado por el bloque:

```
a5 = recibir_campomagnetico(server_sock, "x")
```

2.9.- Leer los ángulos de navegación

Este bloque detiene la programación del robot hasta que reciba del dispositivo los valores de orientación del dispositivo (si lo tiene, caso contrario devuelve 0 o NaN dependiendo del dispositivo). Se pueden pedir en cualquiera de los ejes X, Y o Z y se puede pedir cualquiera de las 3 variables (Azimuth, roll o pitch).

Función implementada en python:

```
def recibir_orientacion(server_sock, variable):
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI). Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

```
  server_sock.send("readOrientation-%s\n" % variable)
  dato = recibe_texto(server_sock)
  print("variable: %s" %dato)
  return dato
```

Reglas añadidas al bloque para generar código Python:

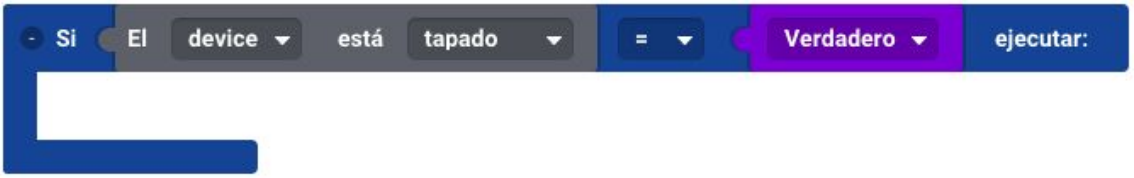
```
python: {
    codeLines: [{
      code: 'recibir_orientacion(server_sock, {AXIS})'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
a2 = recibir_orientacion(server_sock, "azimuth")
```

2.10.- Obtener el valor del sensor que indica si un dispositivo está tapado

Este bloque devuelve verdadero (true) si el dispositivo está tapado.



Función implementada en python:

```
def recibir_estacubierto(server_sock, cv):
  server_sock.send("readProx-%s\n" % cv) #Fallo
  dato = recibe_texto(server_sock)
  print("cubierto: %s" %dato)
  return dato
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'recibir_estacubierto(server_sock, {COVERED})'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
a6 = recibir_estacubierto(server_sock, "covered")
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa

2.11.- Obtener el valor de la luz del ambiente

Este bloque detiene la programación del robot hasta que reciba del dispositivo los valores del sensor de luz que tiene el dispositivo (si lo tiene, caso contrario devuelve 0 o NaN dependiendo del dispositivo).

Función implementada en python:

```
def leer_luz(server_sock):
  server_sock.send("readLight-\n")
  dato = recibe_texto(server_sock)
  print("nivel luz: %s" %dato)
  return dato
```

Reglas añadidas al bloque para generar código Python:

```
python: {
    codeLines: [{
      code: 'leer_luz(server_sock)'
    }]
  }
```

Código de ejemplo generado por el bloque:

```
a7 = leer_luz(server_sock)
```


Proyecto Financiado por el Centro de Desarrollo Tecnológico Industrial (CDTI).
Expediente IDI-20150289
Cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) a través del Programa Operativo Plurirregional de Crecimiento Inteligente 2014-2020

MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
CDTI Centro para el Desarrollo Tecnológico Industrial
UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional (FEDER)
Una manera de hacer Europa