BREVE GUÍA DE USO DE LA LIBRERÍA RPyC EN ev3dev/Python

Curso: Proyecto I
Profesor: Ricardo Valdivia

I. INSTALACIÓN

a) Instalar RPyC en el Servidor (EV3):

```
sudo easy_install3 rpyc
```

o (de forma similar)

```
sudo apt install python3-rpyc
```

Cuidado, se instalará la versión 3.3.0 de RPyC.

b) Debe tenerse cuidado de utilizar una versión anterior de **Python** en el Cliente (PC), versiones 3.4 o 3.5, por problemas de compatibilidad con la librería.

c) Instalar la misma versión de RPyC en el Cliente (PC):

```
pip install 'rpyc==3.3.0'
```

d) Opcionalmente, comprobar la versión instalada de RPyC en el Cliente (PC) y en el Servidor (EV3). Puede haber problemas si estas no son similares:

```
python -c "import rpyc; print(rpyc.__version__)"
```

II. PRUEBAS

II.1 PRUEBA BÁSICA

En el Servidor (EV3):

Cree un archivo rpyc_server.py e incluya los siguientes elementos,

a) Primero, importe las librerías `RPyC`, `ThreadedServer` y `ev3dev` (se utilizará la clase `Sound` para probar este script).

b) Ahora, defina una clase (en este caso `MyService`). A diferencia de una clase común, usted puede seleccionar que métodos exponer al Cliente. Si el método comienza con `exposed_`, este será remotamente accesible, en otro caso sólo será accesible localmente. En el ejemplo, los clientes podrán llamar al método `speak_message`.

c) Los parámetros del método `speak_message` son `self` (porque este es un método que pertenece a la clase MyService) y `msg`, el texto que dirá el robot EV3. En la función se ha utilizado el método `speak` de la clase `Sound` para indicar al robot EV3 que diga el mensaje.

d) Para exponer el método `speak_message` al mundo, es necesario iniciar un Servidor. Para esto se ha invocado al servicio `ThreadedServer`.

```
import rpyc
from rpyc.utils.server import ThreadedServer
from ev3dev.ev3 import *

class MyService(rpyc.Service):
    def exposed_speak_message(self, msg):
        Sound.speak(msg)

if __name__ == '__main__':
    s = ThreadedServer(MyService, port=18812)
    s.start()
```

e) Inicie el script en el robot EV3:

```
python3 rpyc_server.py
```

En el Cliente (PC):

Cree un archivo `rpyc_client.py` e incluya los siguientes elementos,

a) Primero, importe la librería `RPyC`.

b) Realice una conección `RPyC` indicando la dirección ip del Servidor (EV3) y definiendo el número de la puerta (debe ser la misma definida en el script del EV3).

c) Ahora, es posible llamar al servicio deseado, en este caso `speak_message`. Este servicio es expuesto como el root object de la conexión: `conn.root`.

```
import rpyc

conn = rpyc.connect('192.168.61.250', port=18812)
conn.root.speak_message('Thewer is 42')
```

d) Inicie el script en el cliente (PC):

```
python3 rpyc_client.py
```

Este ejemplo, el robot EV3 dice las famosas palabras: “The answer is 42”.

II.2 PRUEBA CON MOTORES

SI la configuración anterior funciona sin problemas, puede, ahora incorporar más métodos a su Servidor (EV3). Por ejemplo, en este caso se ha incorporado el método `move_wheel` que mueve una rueda del robot.

Asegúrese, antes de iniciar el nuevo proceso servidor, que el anterior proceso esté detenido (`sudo ps –e …`), en caso contrario deténgalo (`sudo kill –9 …`) . Si un proceso no se ha detenido recibirá un mensaje que impide la ejecución de un nuevo proceso similar.

```
import rpyc
from rpyc.utils.server import ThreadedServer
from ev3dev.ev3 import *

class MyService(rpyc.Service):
        def exposed_speak_message(self, msg):
                Sound.speak(msg)

        def exposed_move_wheel(self, out):
                m = LargeMotor(out)
                m.run_timed(time_sp=1000, speed_sp=600)

if __name__ == '__main__':
        s = ThreadedServer(MyService, port=18812)
        s.start()
```

Incorpore nuevas llamadas desde el Cliente (PC) que prueben el correcto funcionamiento del nuevo método implementado.

```
conn = rpyc.connect('192.168.61.250', port=18812)
conn.root.speak_message('Thewer is 42')
conn.root.move_weel('outB')
conn.root.move_weel('outC')
```

III. REFERENCIAS

RPyC
Transparent, Symmetric Distributed Computing
https://rpyc.readthedocs.io/en/latest/

Python-ev3dev
Working with ev3dev remotely using RPyC
https://ev3dev-lang.readthedocs.io/projects/python-ev3dev/en/stable/rpyc.html#

KEES, Lego MindStorms Projects by Kees Smith
Working with RPyC
https://ksmev3.wordpress.com/2017/06/08/working-with-rpyc/

Ev3python
RPyC
https://sites.google.com/site/ev3python/learn_ev3_python/rpyc