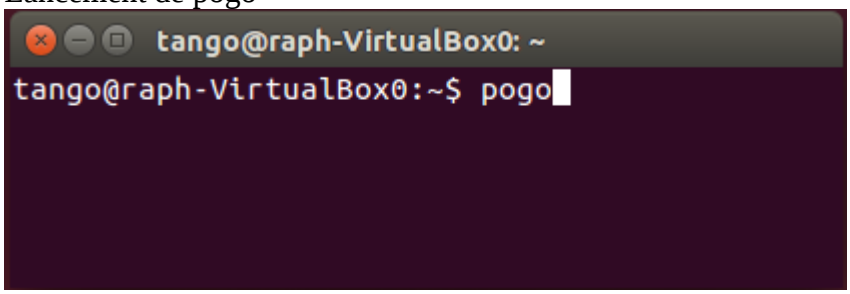
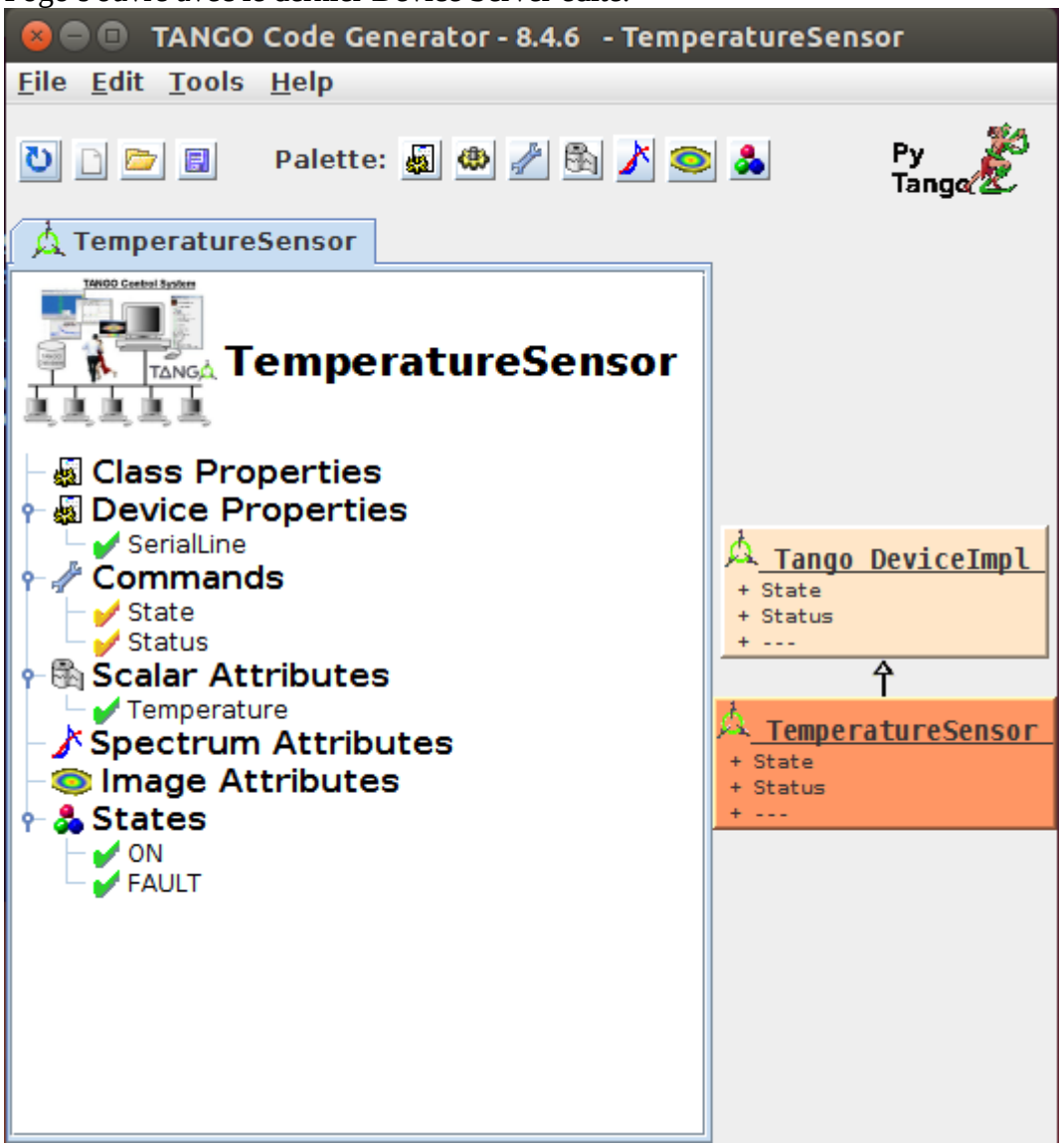
OPERATIONS REALISÉES PENDANT LE TP TEMPERATURE SERIAL

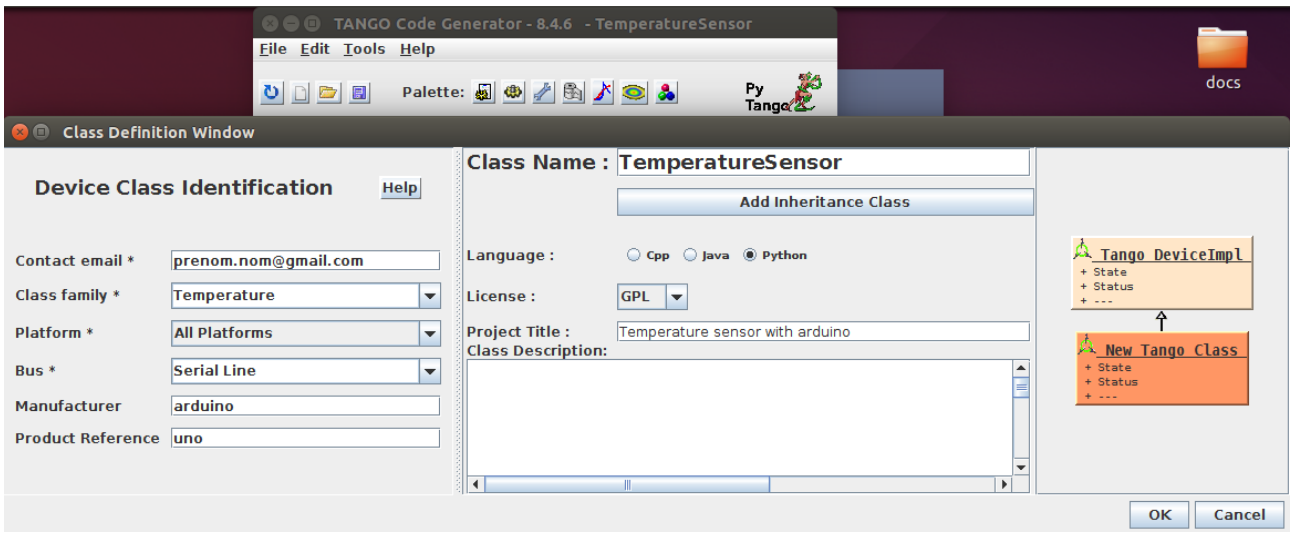
Lancement de pogo



Pogo s'ouvre avec le dernier Device Server édité.

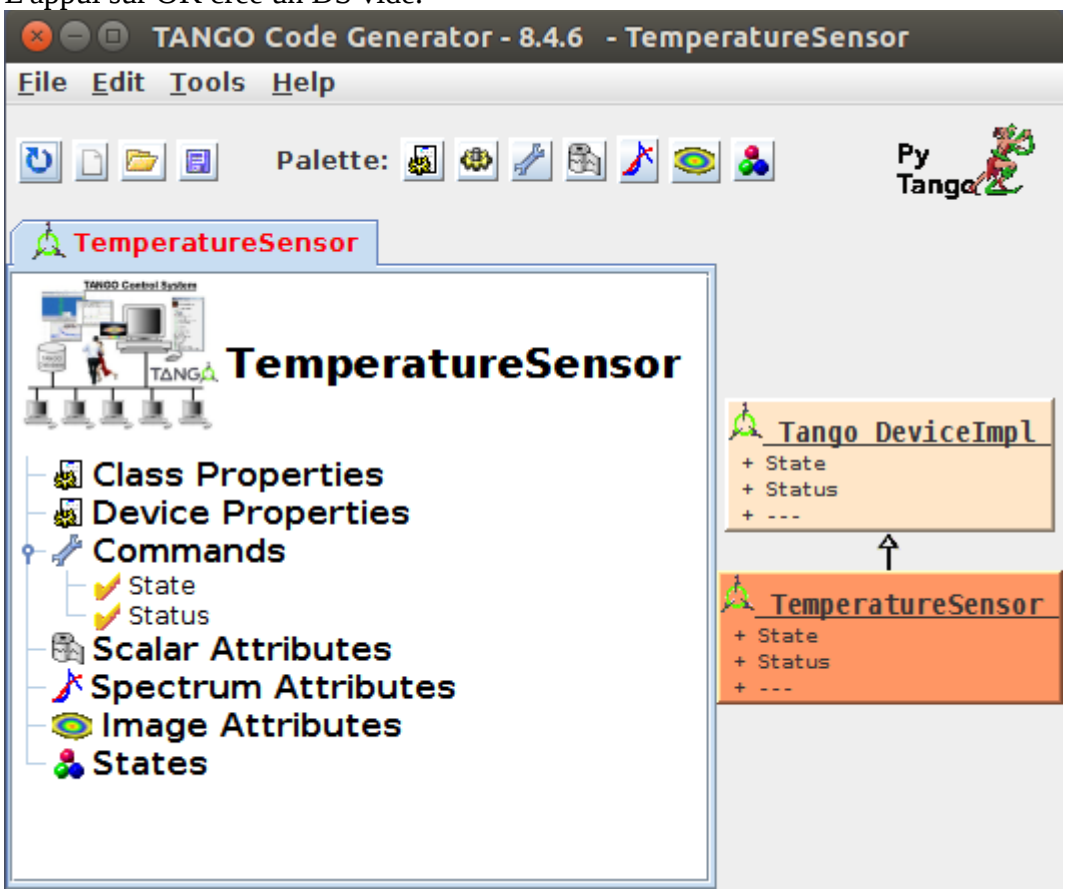


New pour la création d'un Device Server

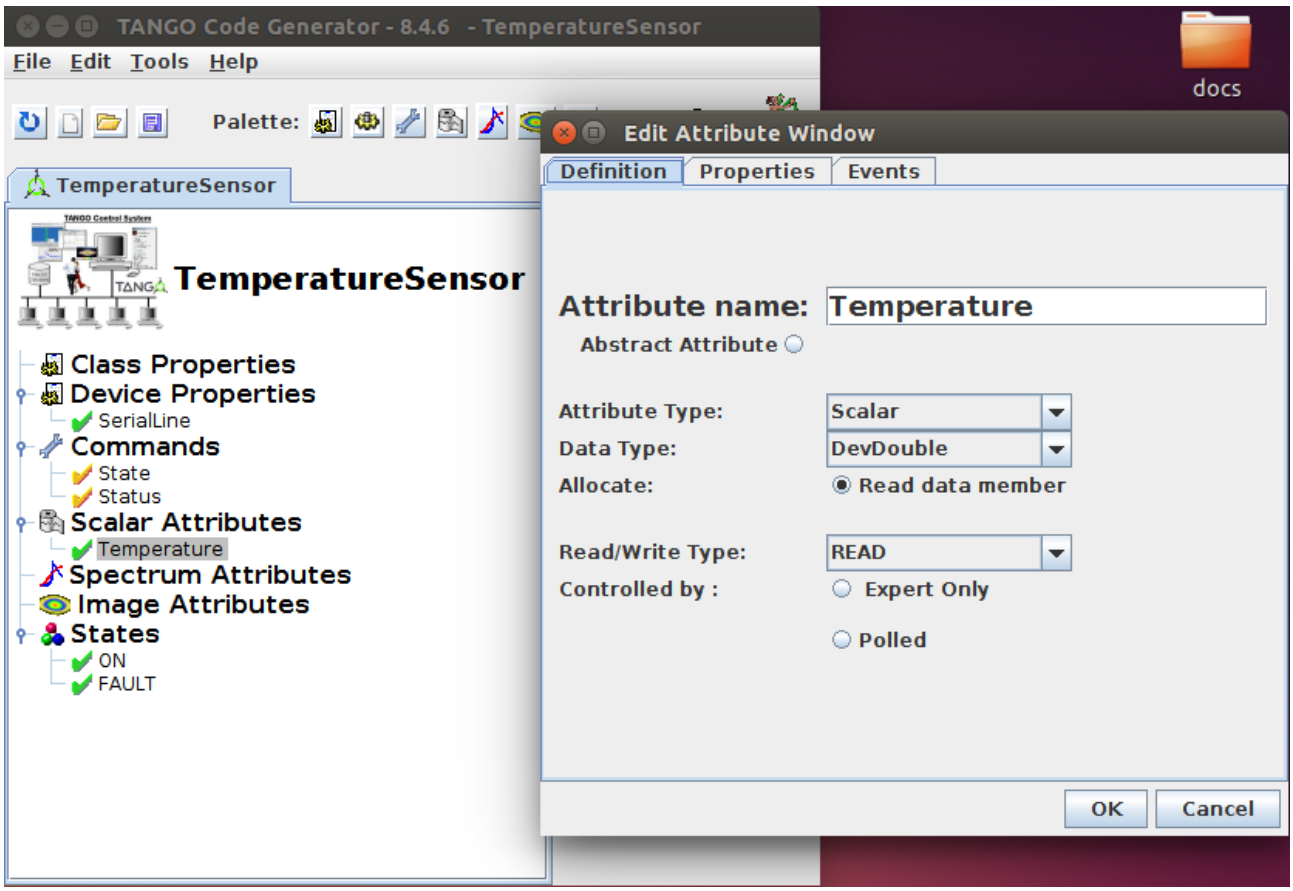


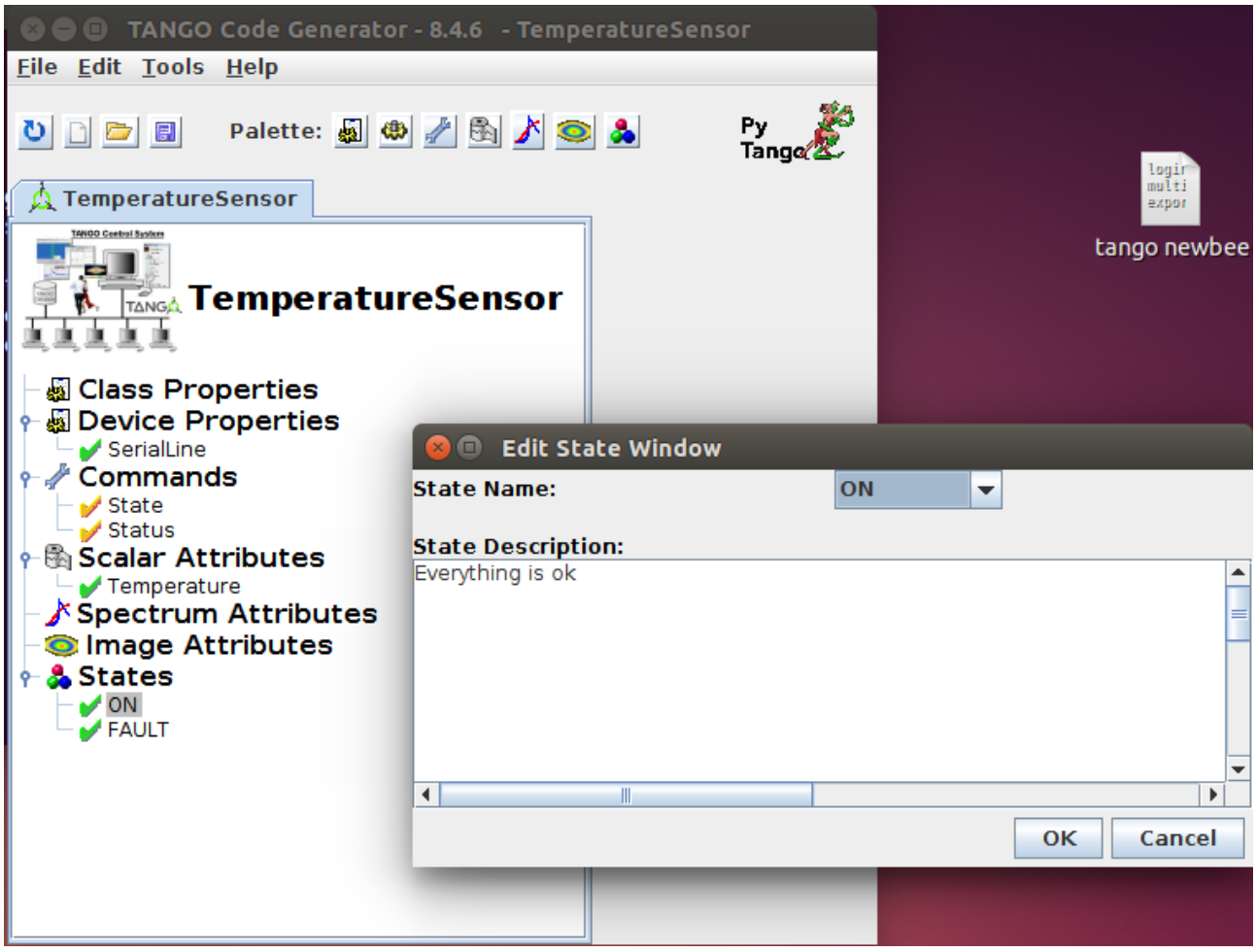
Ici on a choisi python comme langage.

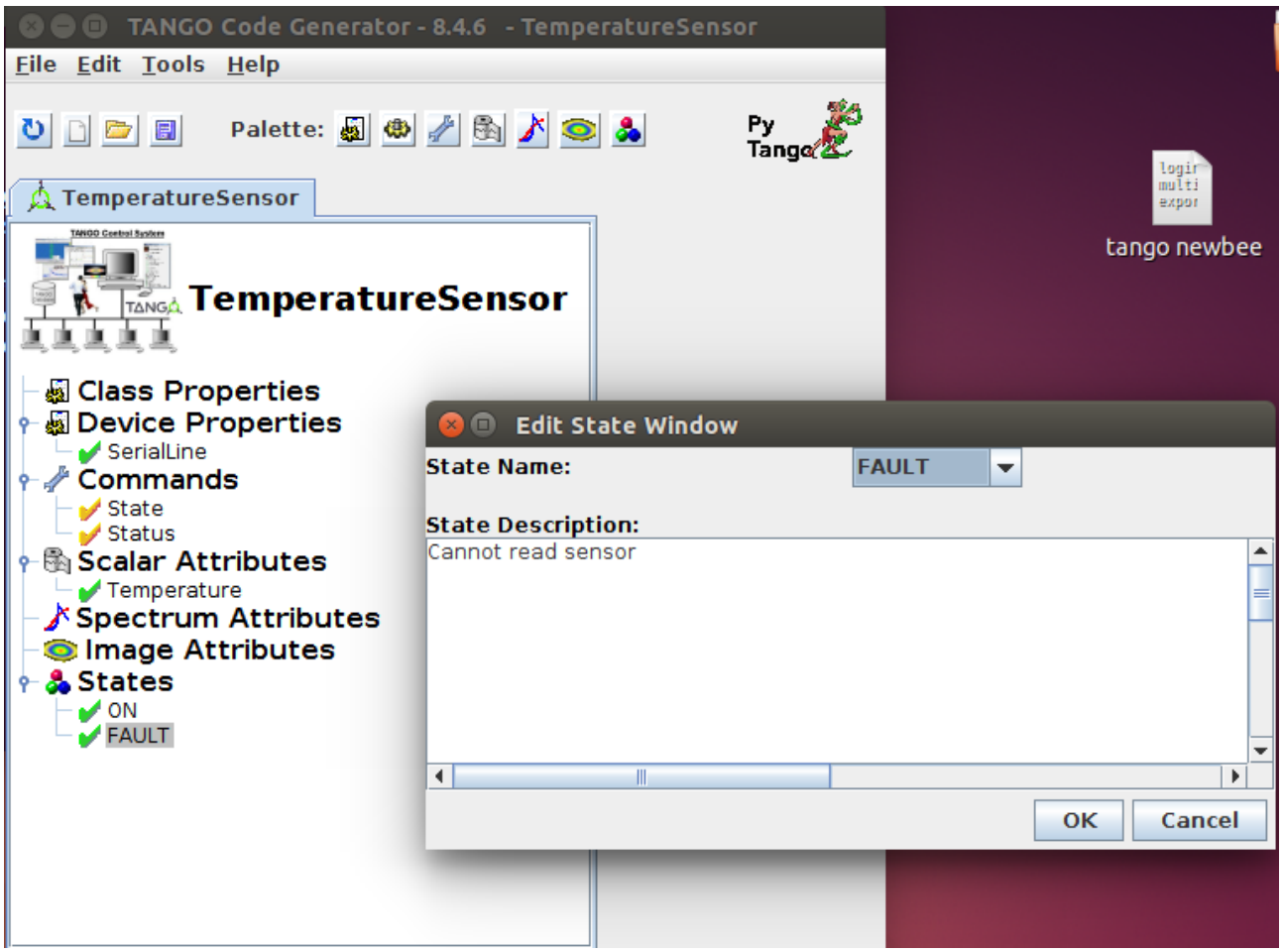
L'appui sur OK crée un DS vide.

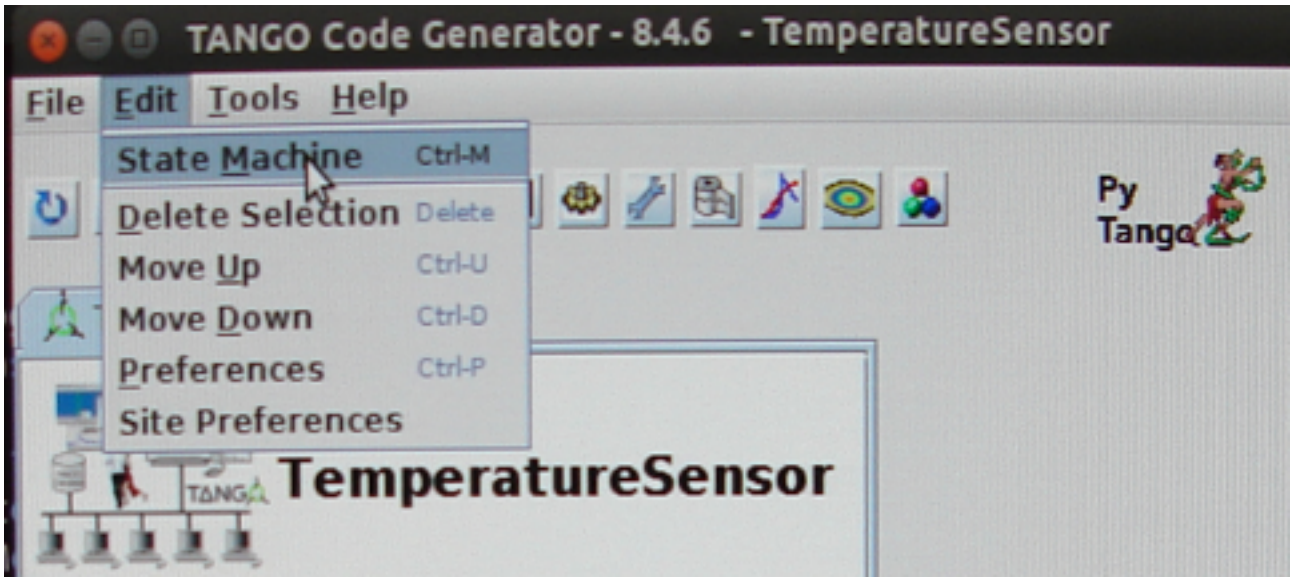


Clic droit sur les éléments de l'arborescence permet de créer les propriétés, attributs, états

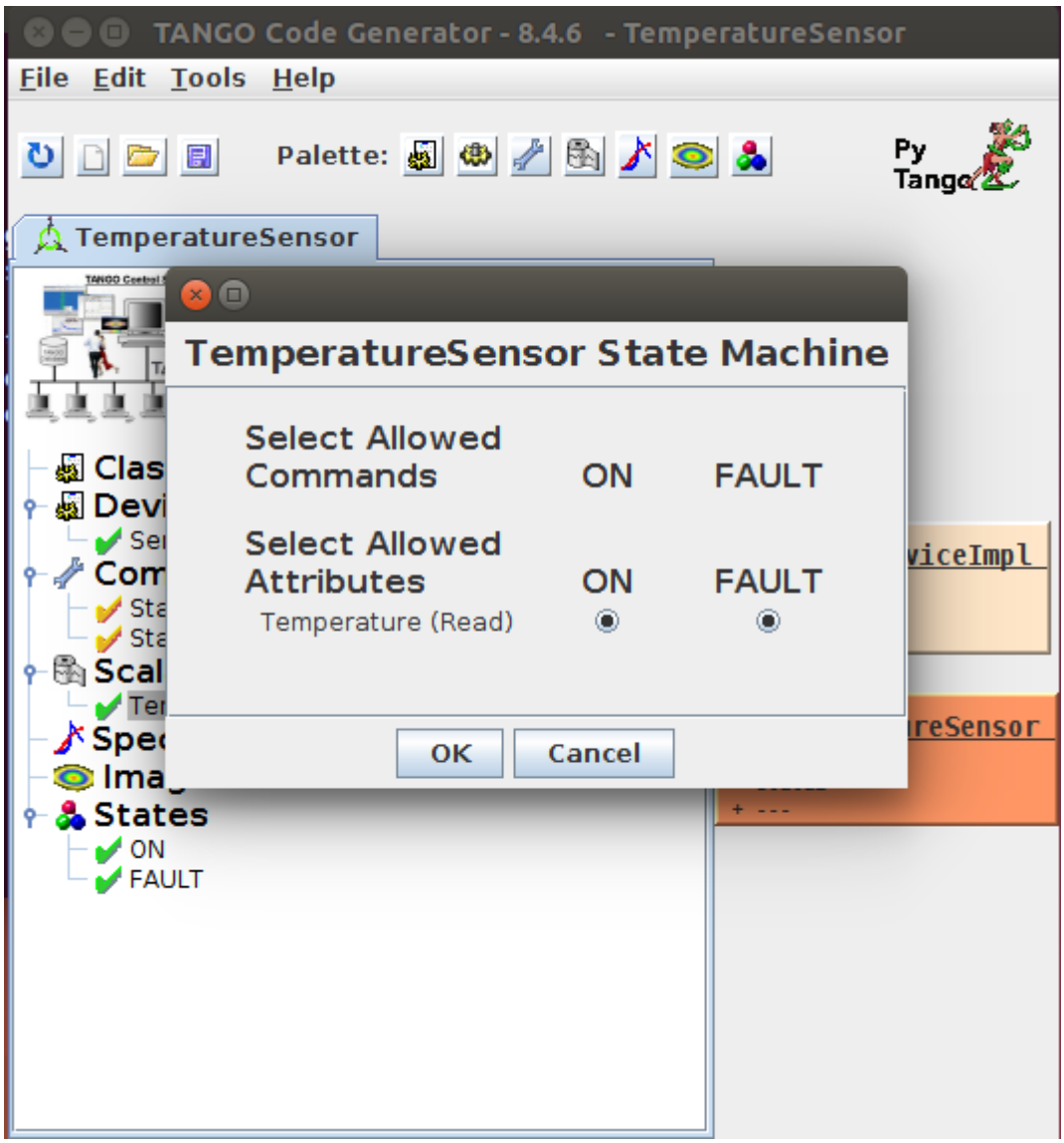

TANGO Code Generator - 8.4.6 - TemperatureSensor
File Edit Tools Help
Palette:
docs
TemperatureSensor
TemperatureSensor
Class Properties
Device Properties
SerialLine
Commands
State
Status
Scalar Attributes
Temperature
Spectrum Attributes
Image Attributes
States
ON
FAULT
Edit Attribute Window
Definition
Properties
Events
Attribute name:
Temperature
Abstract Attribute
Attribute Type:
Scalar
Data Type:
DevDouble
Allocate:
Read data member
Read/Write Type:
READ
Controlled by :
Expert Only
Polled
OK
Cancel

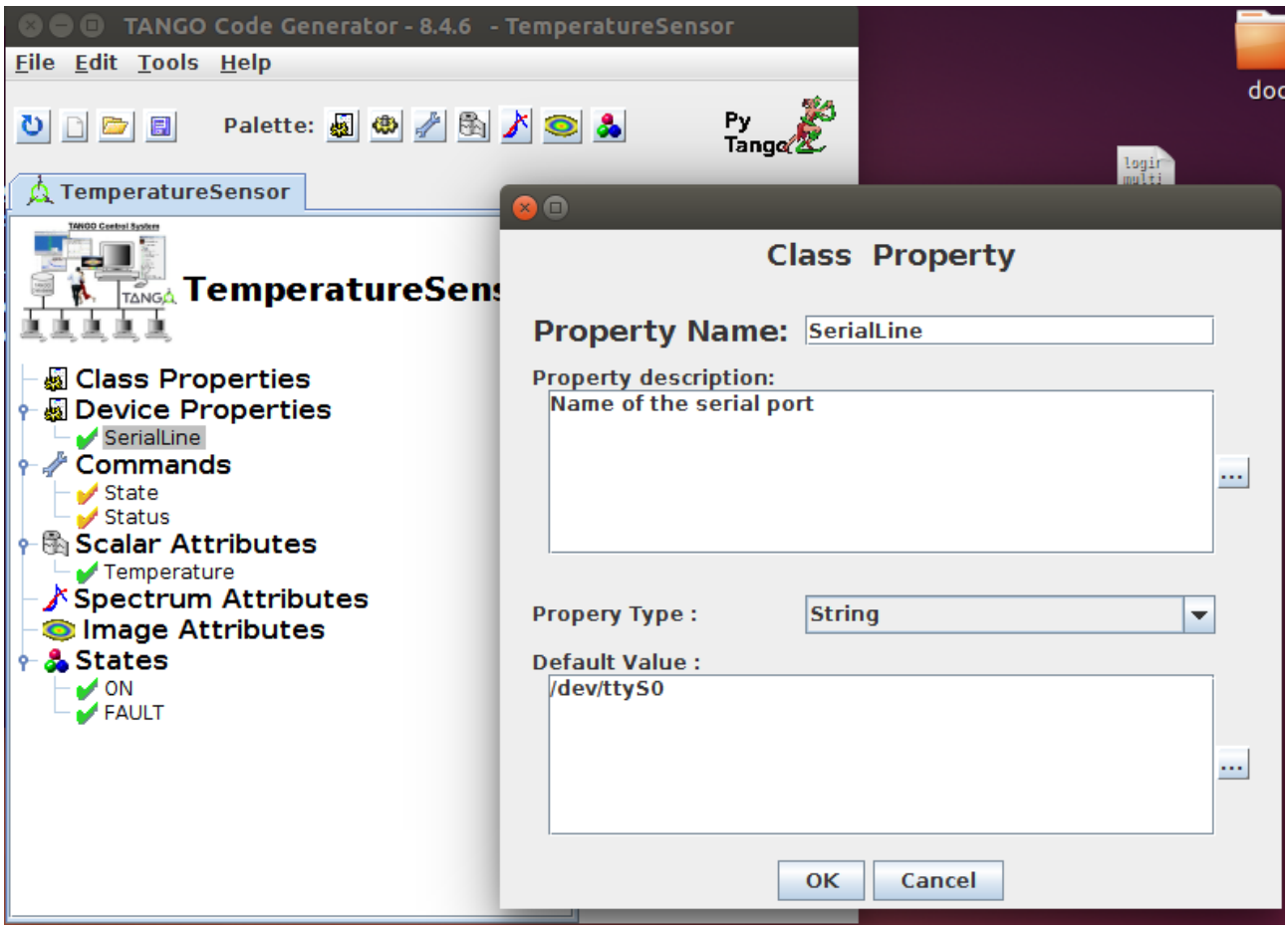

TANGO Code Generator - 8.4.6 - TemperatureSensor
File Edit Tools Help
Palette:
PyTango
TemperatureSensor
TemperatureSensor
Class Properties
Device Properties
SerialLine
Commands
State
Status
Scalar Attributes
Temperature
Spectrum Attributes
Image Attributes
States
ON
FAULT
tango newbee
Edit State Window
State Name:
ON
State Description:
Everything is ok
OK
Cancel



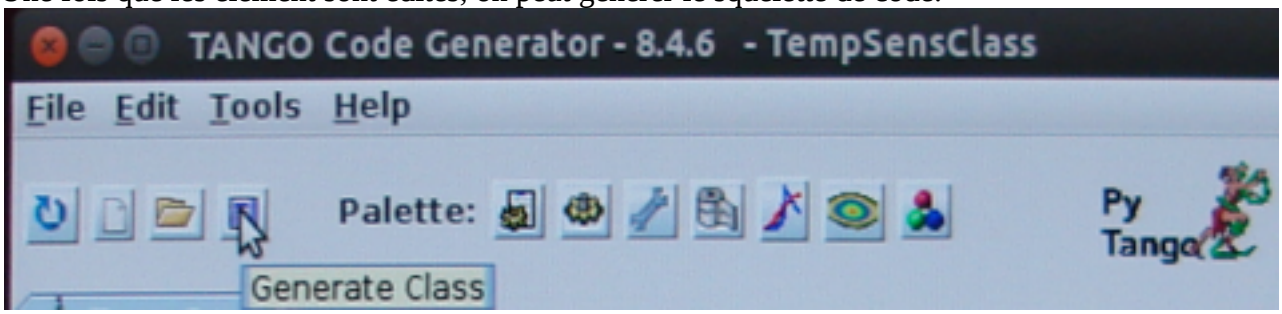


Il est possible de d'activer/désactiver la lecture des attributs en fonction des états qui ont été crées. Ici on conserve la lecture même dans le cas FAULT.

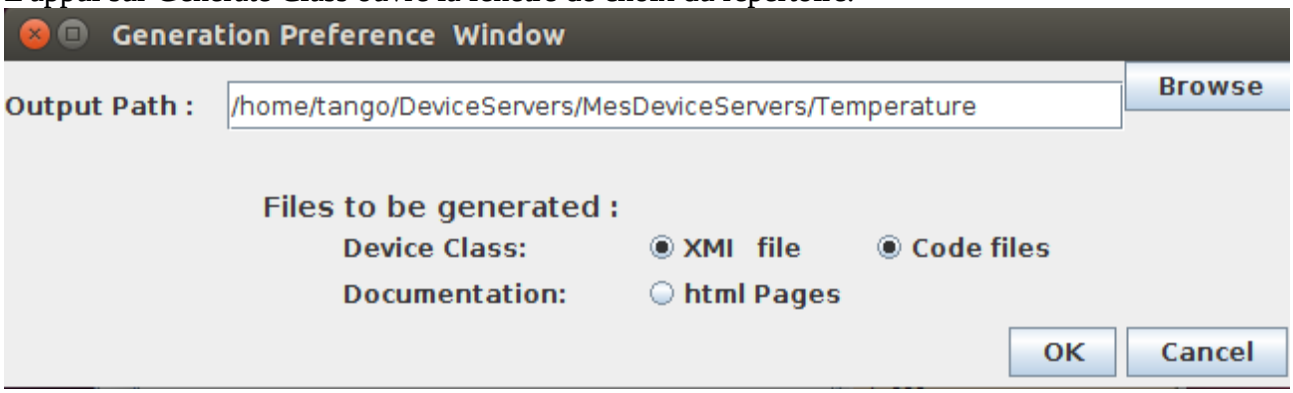

TANGO Code Generator - 8.4.6 - TemperatureSensor
File Edit Tools Help
Palette:
Py Tango
TemperatureSensor
TemperatureSensor State Machine
Select Allowed Commands
ON
FAULT
Select Allowed Attributes
ON
FAULT
Temperature (Read)
OK
Cancel
States
ON
FAULT



Une fois que les élément sont édités, on peut générer le squelette de code.



L'appui sur Generate Class ouvre la fenêtre de choix du répertoire.



Ok => création du code TemperatureSensor.py au chemin désigné.
Ce code doit être édité comme ci dessous.

Code pour l'initialisation (avant ajouts)

```
80
81      def init_device(self):
82          self.debug_stream("In init_device()")
83          self.get_device_properties(self.get_device_class())
84          self.attr_Temperature_read = 0.0
85          #----- PROTECTED REGION ID(TemperatureSensor.init_device) ENABLED START -----#
86
87          #----- PROTECTED REGION END -----#      //      TemperatureSensor.init_device
88
89      def always_executed_hook(self):
90          self.debug_stream("In always_excuted_hook()")
91          #----- PROTECTED REGION ID(TemperatureSensor.always_executed_hook) ENABLED START -----#
92
93          #----- PROTECTED REGION END -----#      //      TemperatureSensor.always_executed_hook
94
```

Après ajouts

```
82      def init_device(self):
83          self.debug_stream("In init_device()")
84          self.get_device_properties(self.get_device_class())
85          self.attr_Temperature_read = 0.0
86          #----- PROTECTED REGION ID(TemperatureSensor.init_device) ENABLED START -----#
87
88          self.set_state(PyTango.DevState.ON)
89          self.serial_proxy=PyTango.DeviceProxy(self.SerialLine)
90          #----- PROTECTED REGION END -----#      //      TemperatureSensor.init_device
91
92      def always_executed_hook(self):
93          self.debug_stream("In always_excuted_hook()")
94          #----- PROTECTED REGION ID(TemperatureSensor.always_executed_hook) ENABLED START -----#
95
96          #----- PROTECTED REGION END -----#      //      TemperatureSensor.always_executed_hook
97
```

Code pour la lecture des attributs avant ajouts

```
95      #--------------------------------------------------------------------------
96      #    TemperatureSensor read/write attribute methods
97      #--------------------------------------------------------------------------
98
99      def read_Temperature(self, attr):
100         self.debug_stream("In read_Temperature()")
101         #----- PROTECTED REGION ID(TemperatureSensor.Temperature_read) ENABLED START -----#
102         attr.set_value(self.attr_Temperature_read)
103
104         #----- PROTECTED REGION END -----#      //      TemperatureSensor.Temperature_read
105
```

Après ajouts

```
98      #--------------------------------------------------------------------------
99      #    TemperatureSensor read/write attribute methods
100     #--------------------------------------------------------------------------
101
102     def read_Temperature(self, attr):
103         self.debug_stream("In read_Temperature()")
104         #----- PROTECTED REGION ID(TemperatureSensor.Temperature_read) ENABLED START -----#
105         #self.attr_Temperature_read=random.radint(0,9)
106         #self.attr_Temperature_read=3.14
107
108         self.serial_proxy.DevSerWriteString("T")
109         answer = self.serial_proxy.DevSerReadString(2)
110         self.attr_Temperature_read = float(answer)
111         attr.set_value(self.attr_Temperature_read)
112
113         #----- PROTECTED REGION END -----#      //      TemperatureSensor.Temperature_read
114
115
```

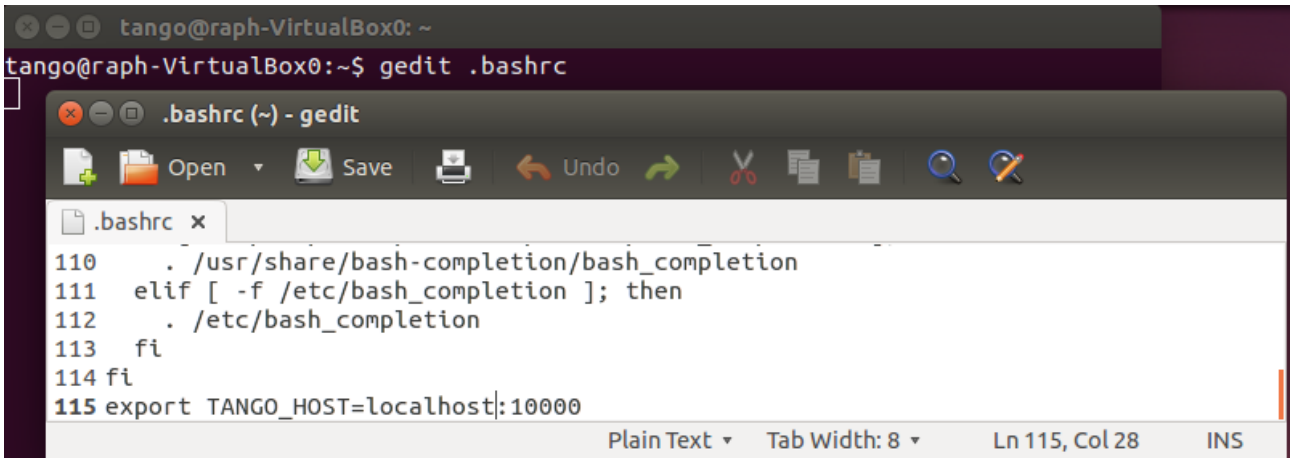
Faire la correction de syntaxe ci dessous pour être conforme au python 3.
Avant correction.

```
189         U = PyTango.Util.instance()
190         U.server_init()
191         U.server_run()
192
193     except PyTango.DevFailed,e:
194         print '-------> Received a DevFailed exception:',e
195     except Exception,e:
196         print '-------> An unforeseen exception occured....',e
197
198 if __name__ == '__main__':
199     main()
```

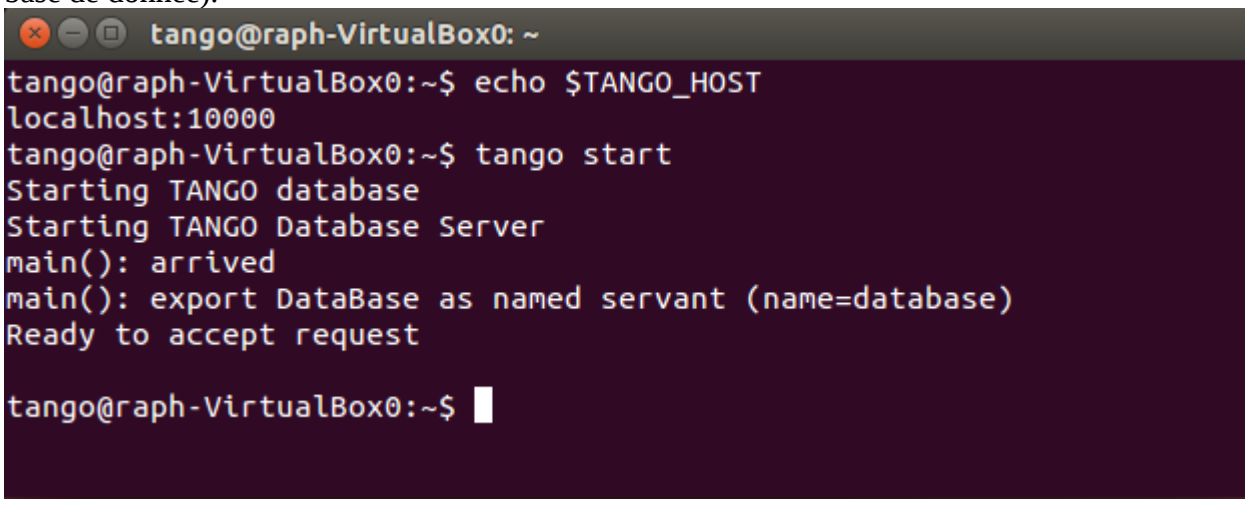
Après corrections (« as » à la place des virgules et parenthèses en plus).

```
189         U = PyTango.Util.instance()
190         U.server_init()
191         U.server_run()
192
193     except PyTango.DevFaile as e:
194         print ('-------> Received a DevFailed exception:',e)
195     except Exception as e:
196         print ('-------> An unforeseen exception occured....',e)
197
198 if __name__ == '__main__':
199     main()
```

Un PC du réseau doit héberger la base de donnée que tango va utiliser.
La variable d'environnement TANGO_HOST doit contenir l'adresse IP et le port soft de ce PC.
Le fichier bashrc peut être édité pour fixer la valeur de TANGO_HOST au démarrage d'Unbuntu.
Ici la base de donnée est dans le PC local.

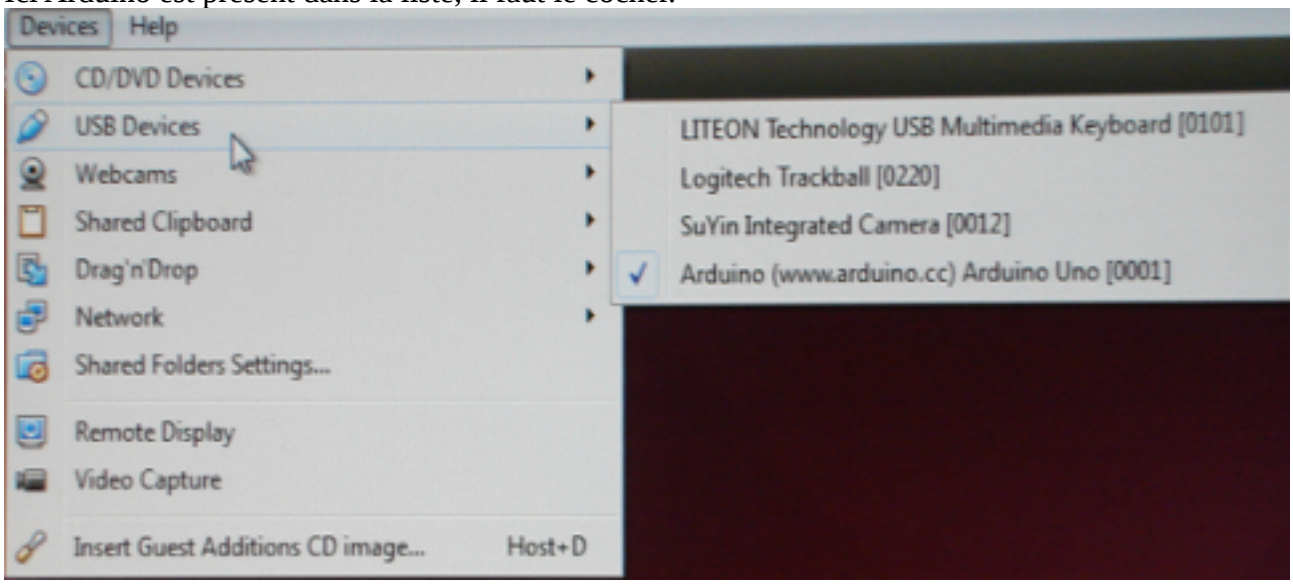


Lancement de tango par la commande tango start (à faire uniquement sur le poste qui héberge la base de donnée).

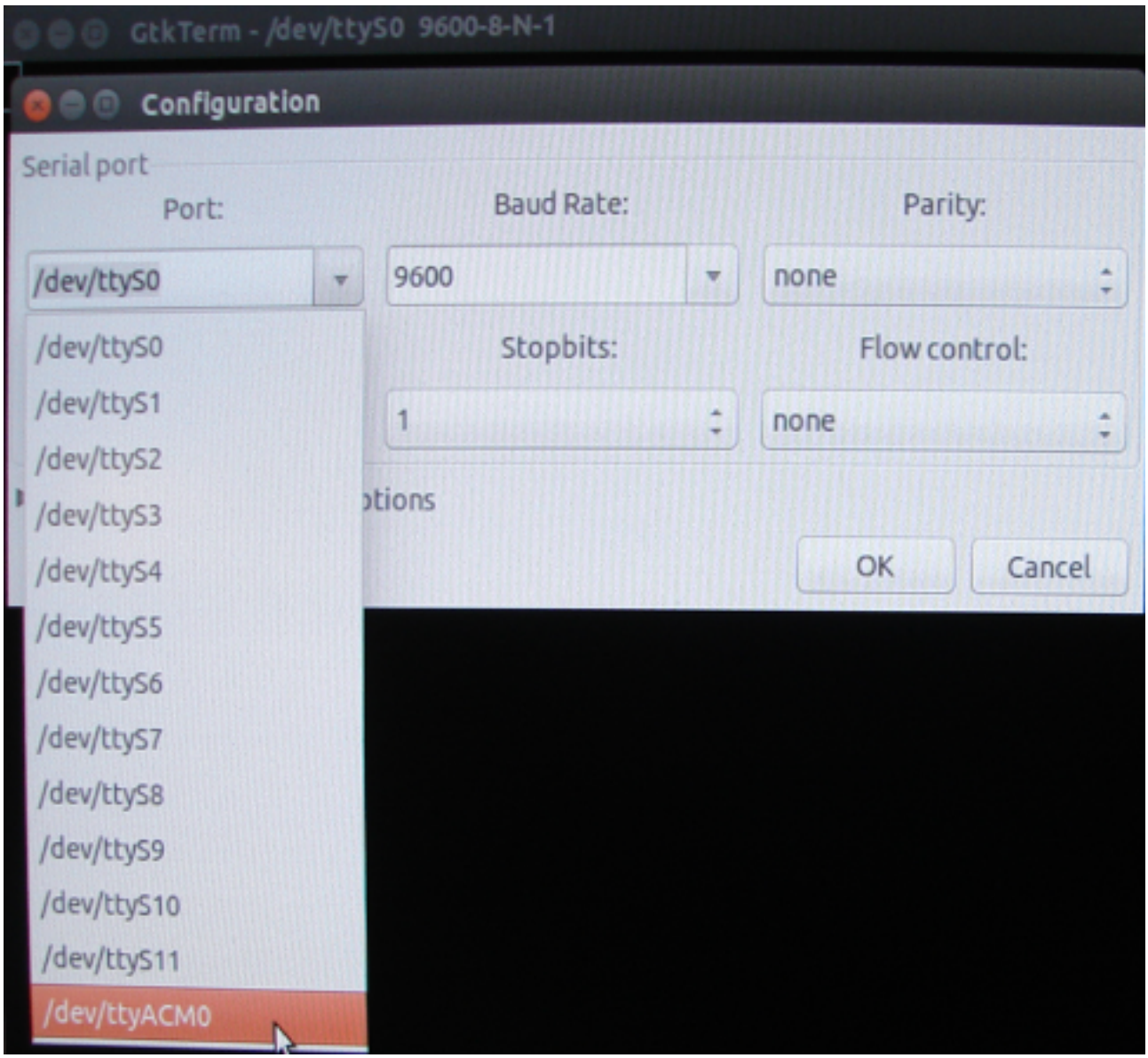


La manip nécessite l'utilisation d'un port com via USB.

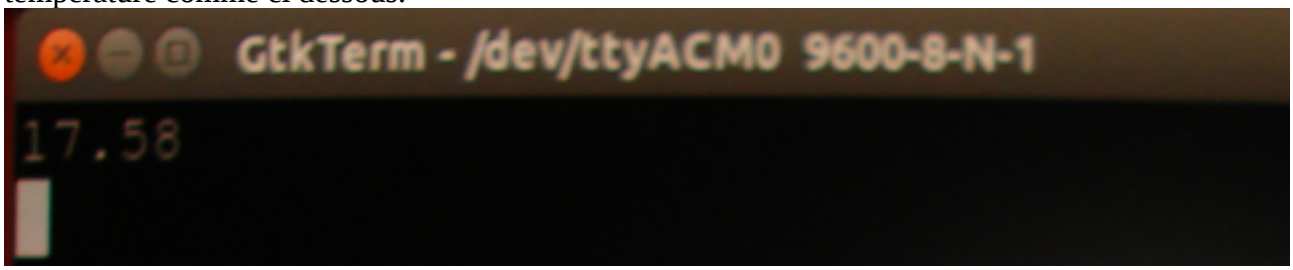
Il faut activer le port USB pour qu'il soit utilisable dans la machine virtuelle.
Ici Arduino est présent dans la liste, il faut le cocher.



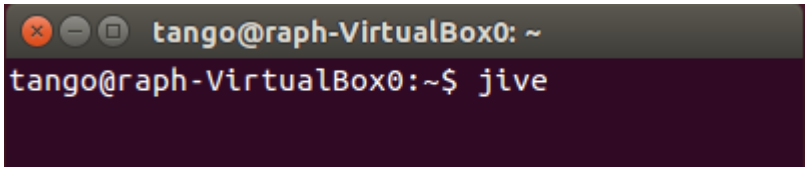
GtkTerm permet de valider la reconnaissance du périphérique et le fonctionnement de la communication.
Dans configuration, le port ttyACM0 doit être présent. Il faut le sélectionner.
Pour les autres paramètres, les valeurs par défaut conviennent (9600 bits/s, 1 stop , pas de parité).

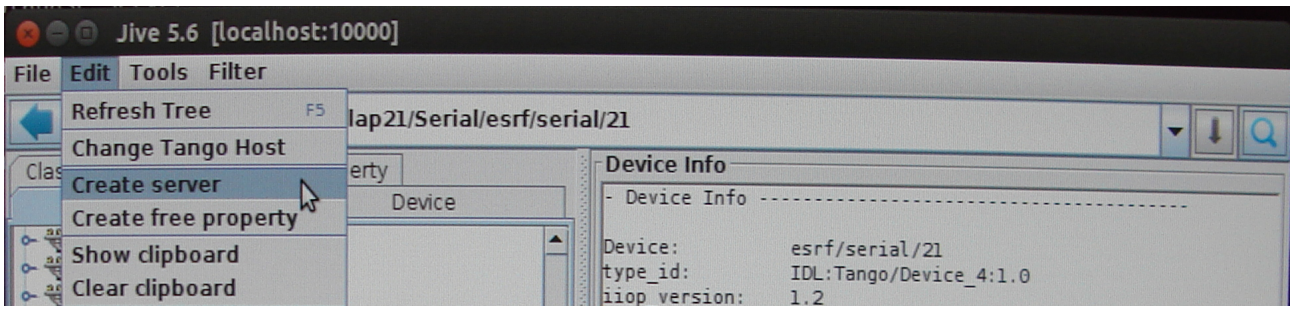


L’hyper-terminal est actif. L'arduino a été programmé pour que la réception du caractère T renvoi la température comme ci dessous.

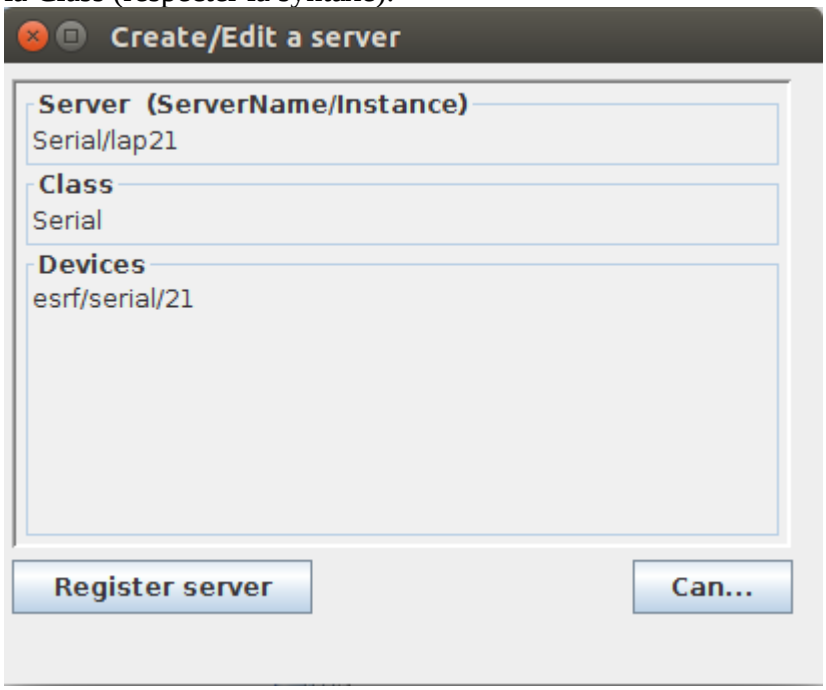


On peut créer un Device Server Serial dans la base de donnée et vérifier son fonctionnement.
Lancement de Jive.

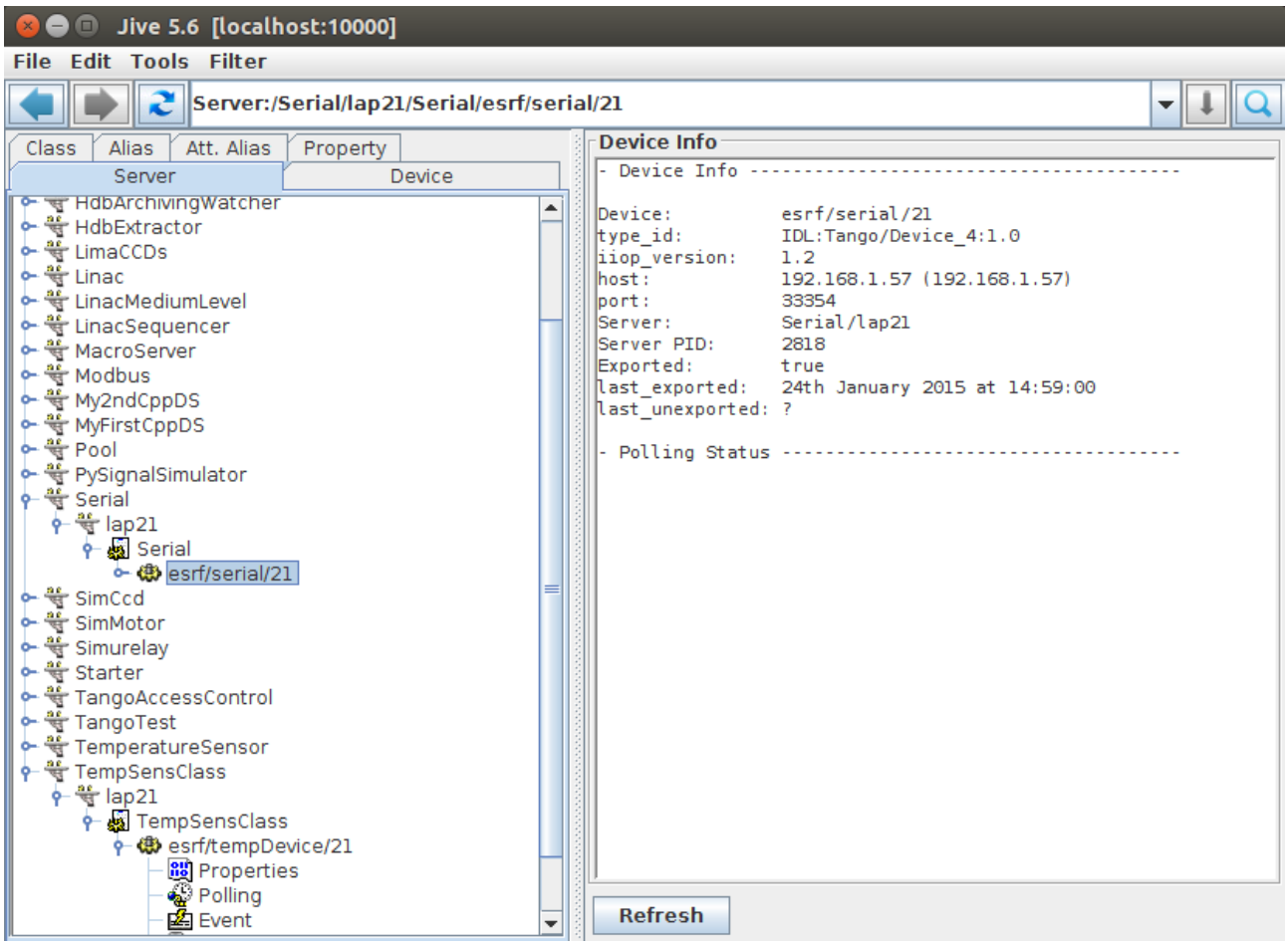




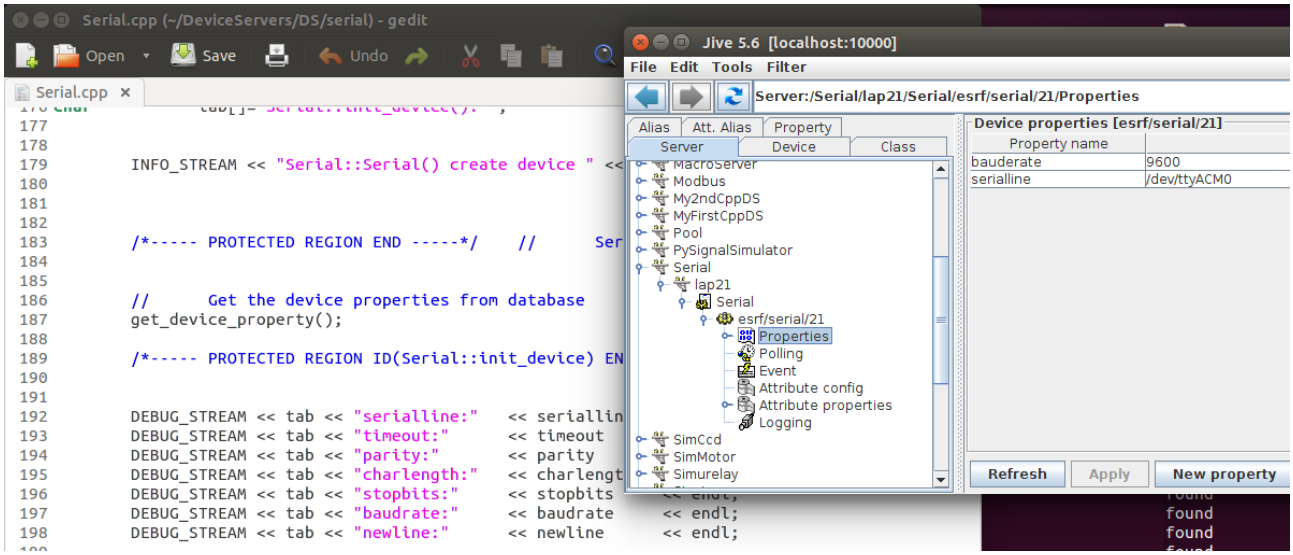
Création du Serveur en utilisant le nom de l'exécutable C++ « Serial » pour le ServerName et pour la Class (respecter la syntaxe).



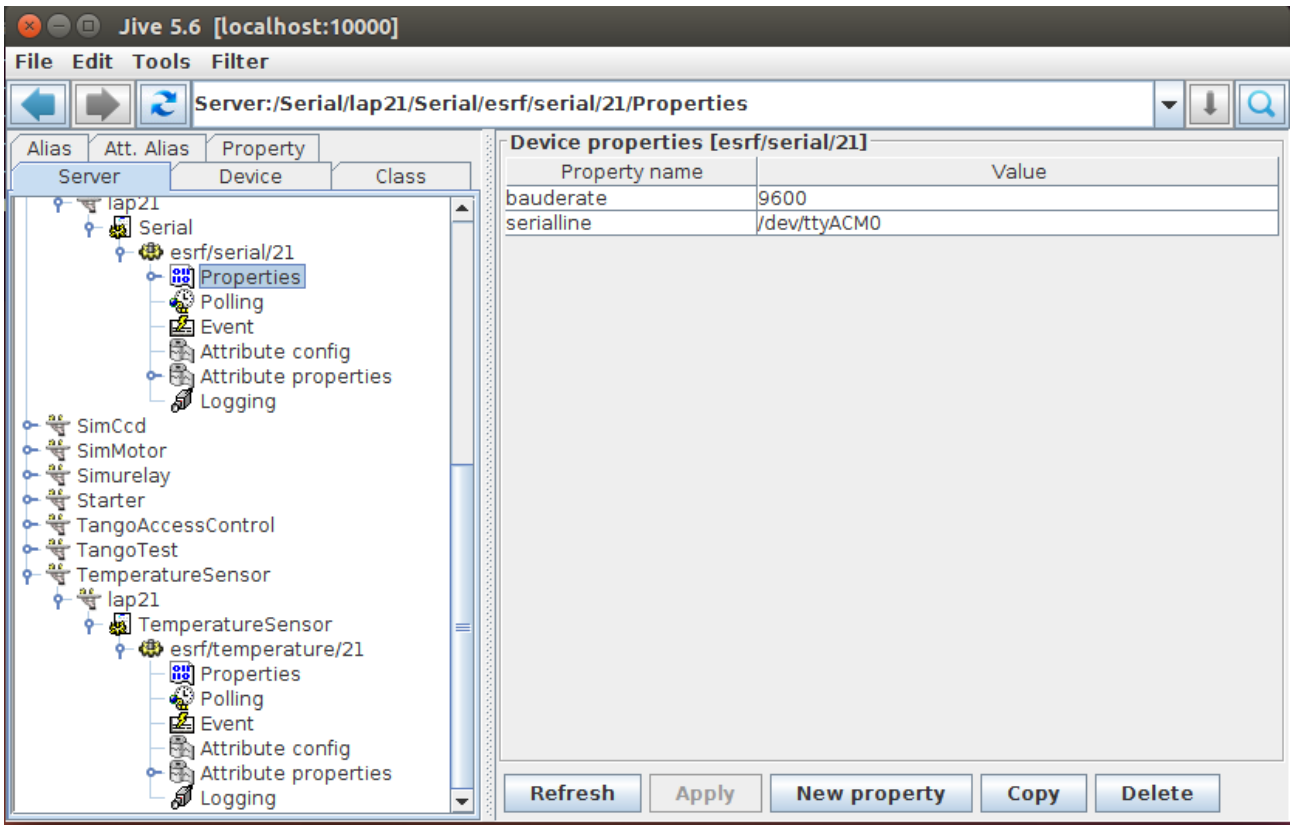
Le Device crée apparaît dans l'arborescence de jive.



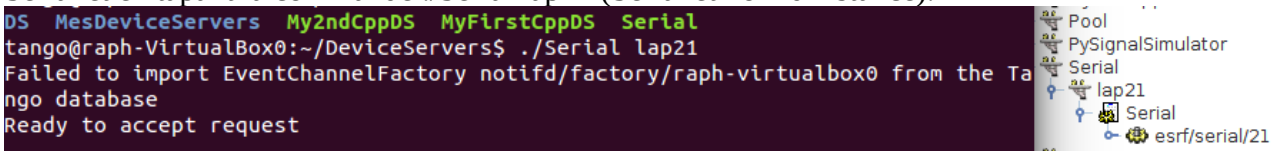
Création de deux propriétés en correspondance avec les chaines de caractère du code C++ du Device Server Serial.



Clic droit sur Properties et création de bauderate et serialline.
Editer les valeurs 9600 et /dev/ttyACM0 par saisie dans value puis appui sur Apply suivi de l'appui sur Refresh.

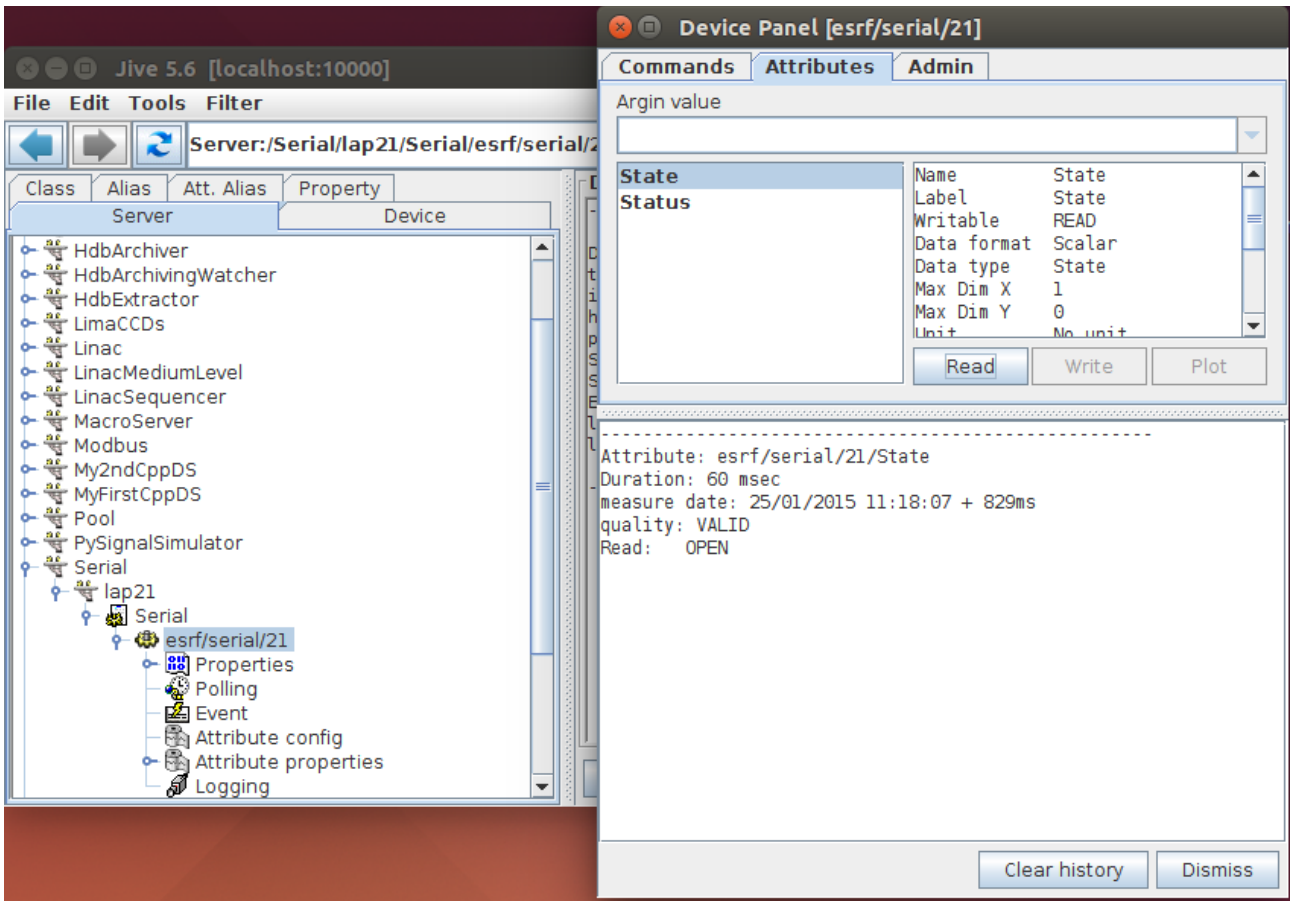


Le Device Server Serial peut être démarré via la console en se plaçant au chemin de l'exécutable Serial et en tapant la commande ./Serial lap21 (Serial et nom d'instance).

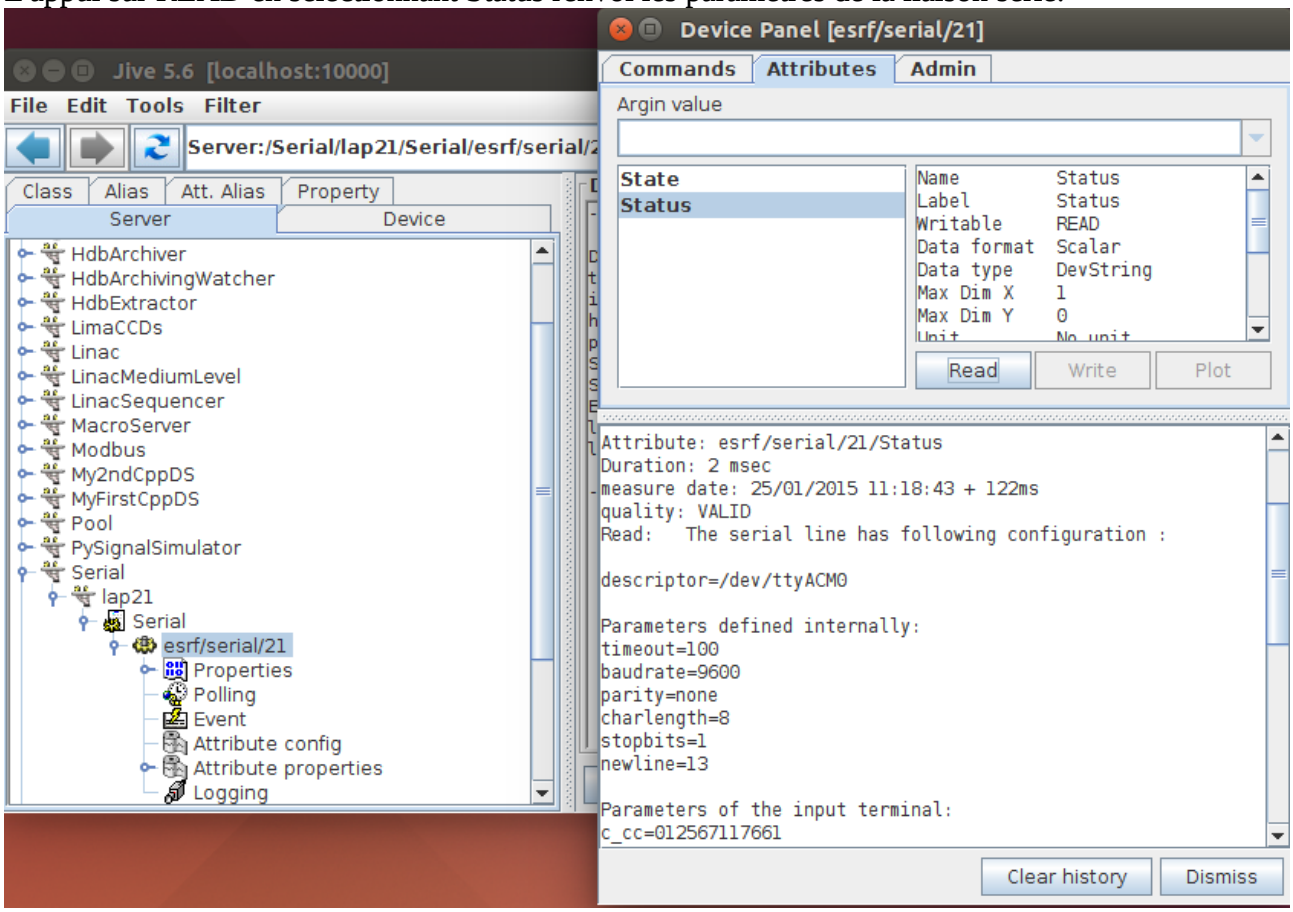


Le message d'erreur ci dessus est normal.

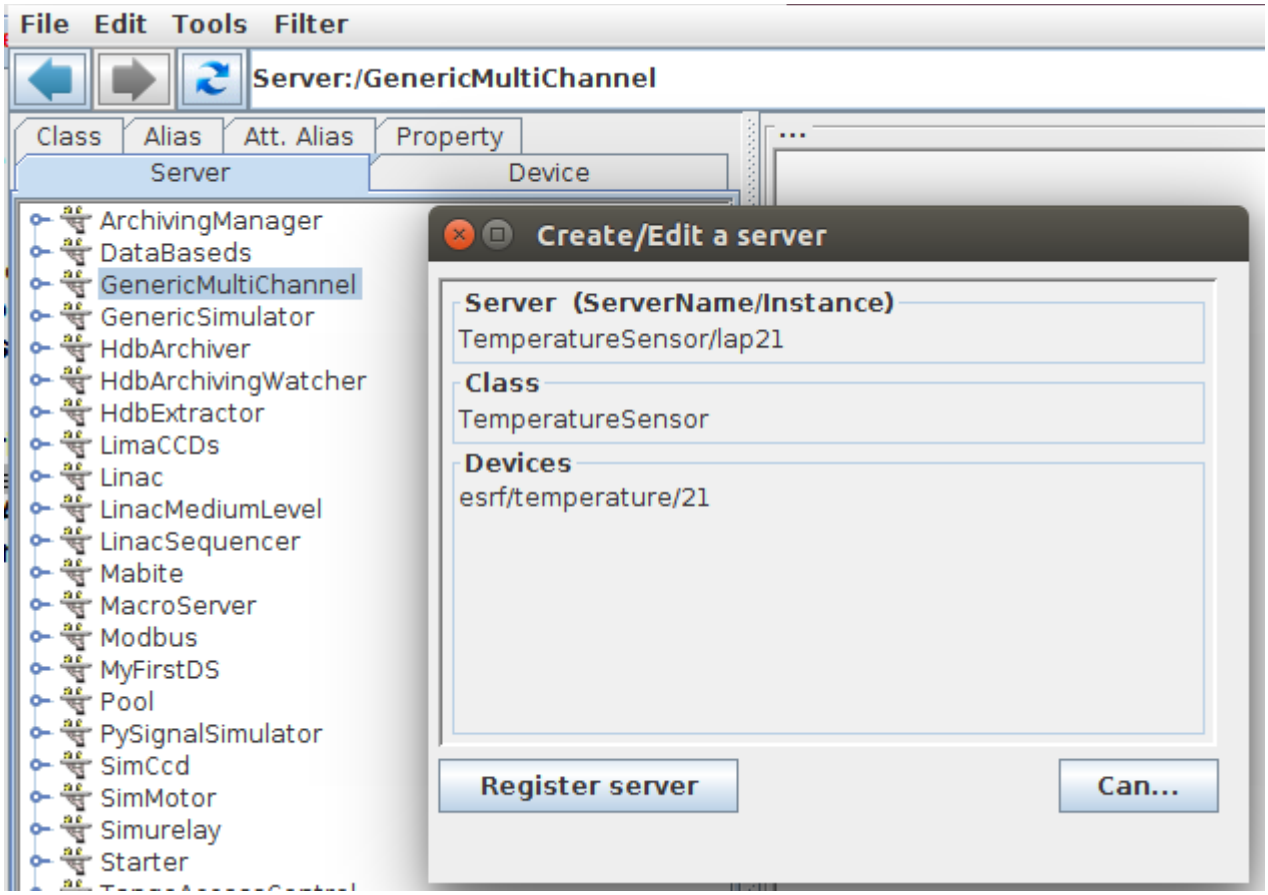
Dans jive on peut lancer par clic droit le menu TestDevice pour contrôler le fonctionnement de la liaison série.
L'appui sur Read et sélectionnant State donne la réponse VALID et OPEN.



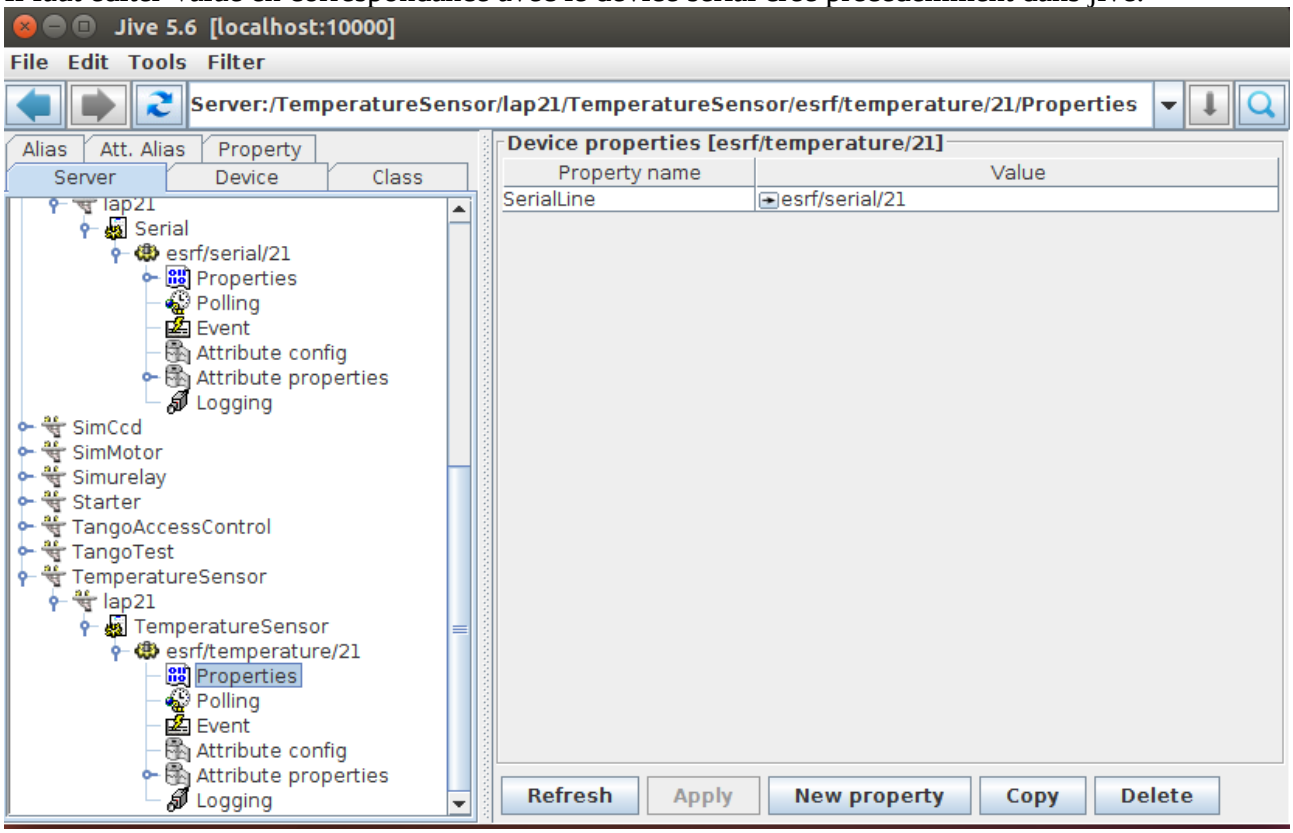
L'appui sur READ en sélectionnant Status renvoi les paramètres de la liaison série.



On peut créer le Device Server Température Sensor via Edit → Create server en respectant la syntaxe de TemperatureSensor comme le nom du fichier python.

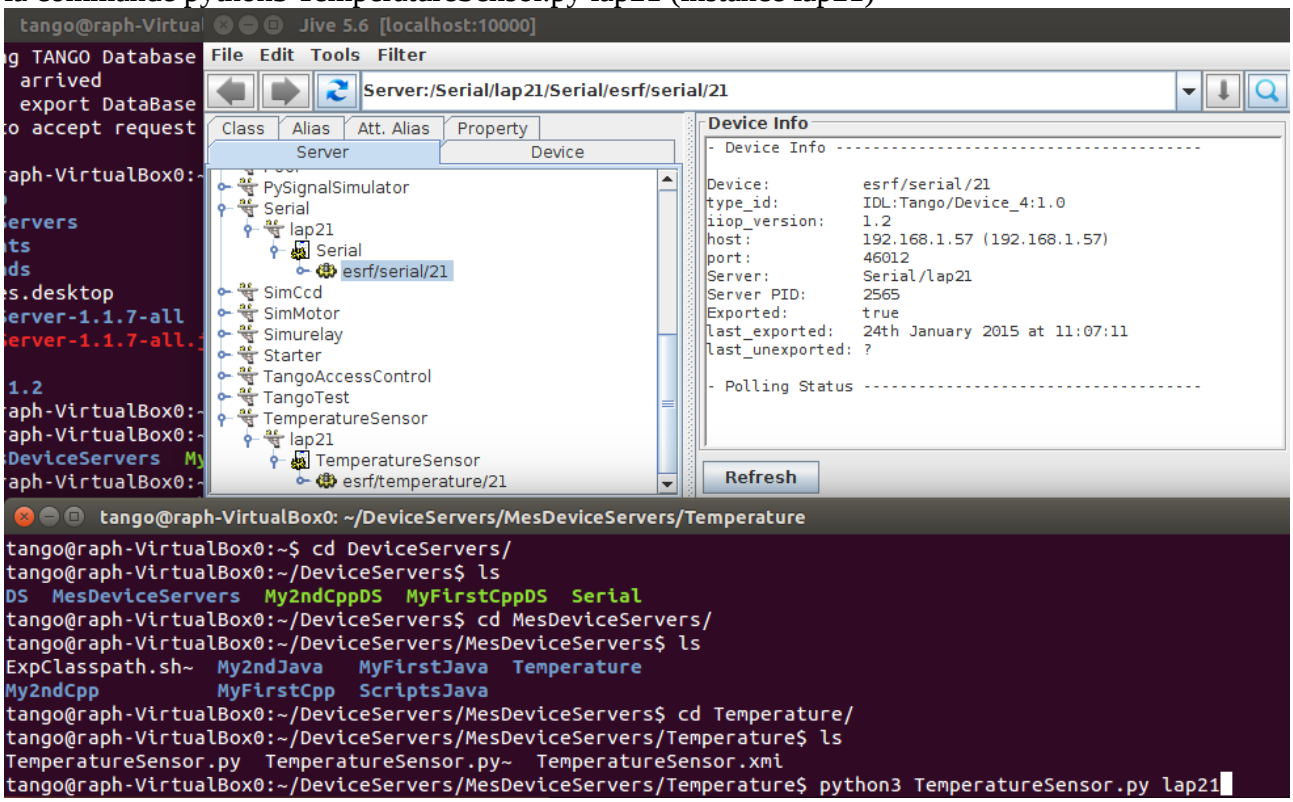


Il faut créer la propriété SerialLine en respectant la syntaxe de la propriété crée via pogo.
Il faut éditer Value en correspondance avec le device serial crée précédemment dans jive.

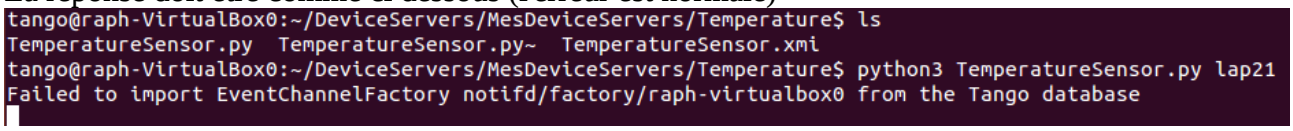


Il faut lancer le Device server en se plaçant dans le répertoire du fichier TemperatureSensor.py avec

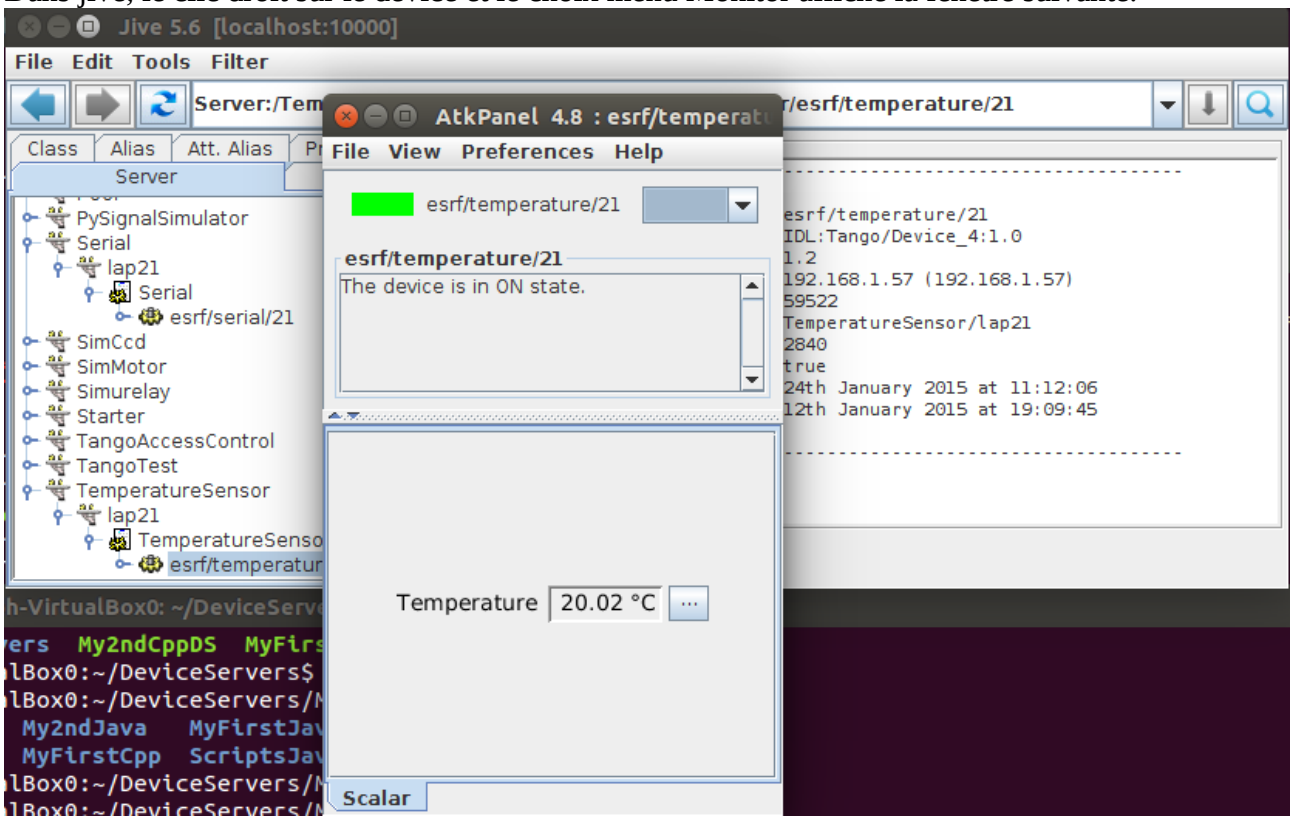
la commande python3 TemperatureSensor.py lap21 (instance lap21)



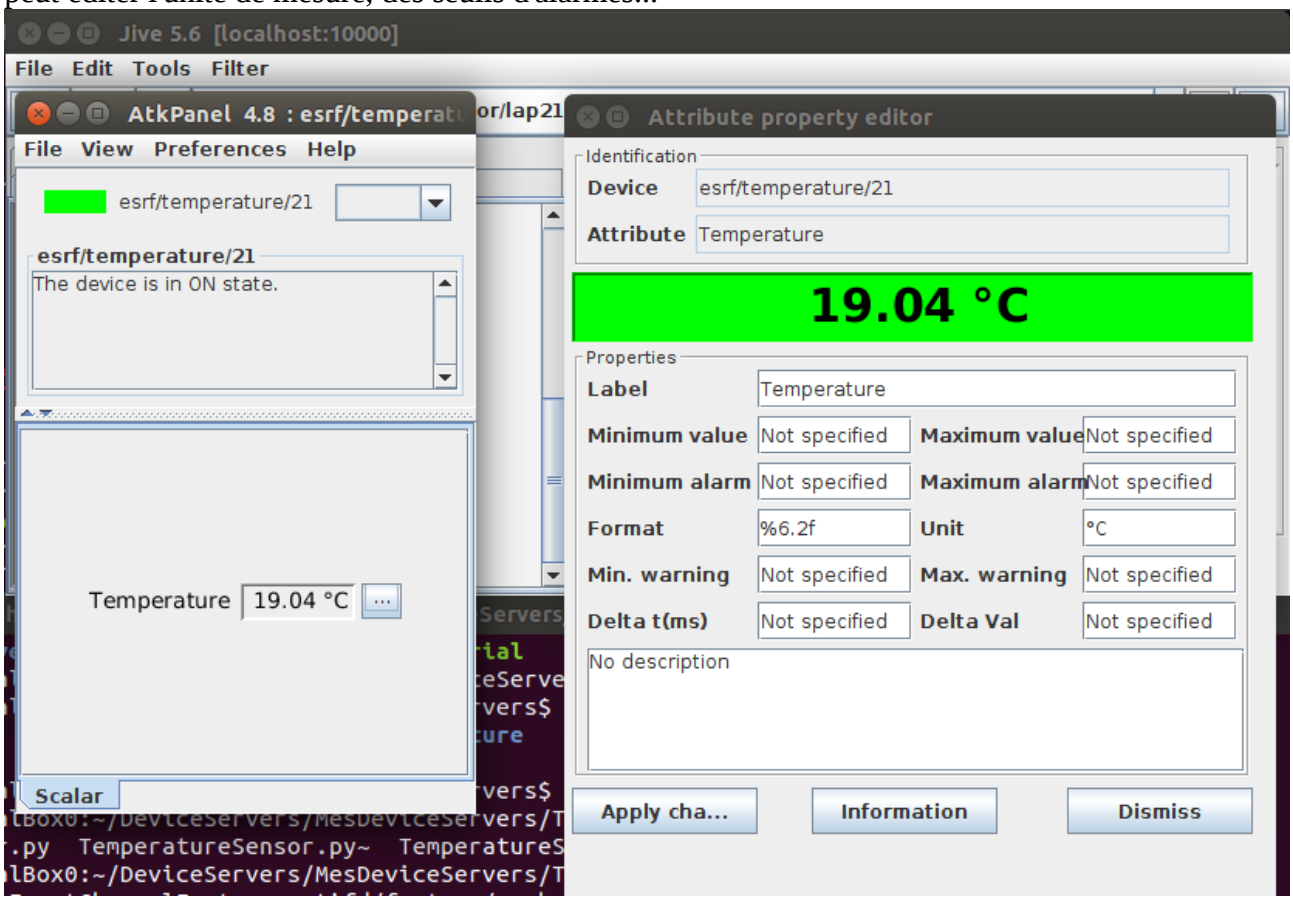
La réponse doit être comme ci dessous (l'erreur est normale)



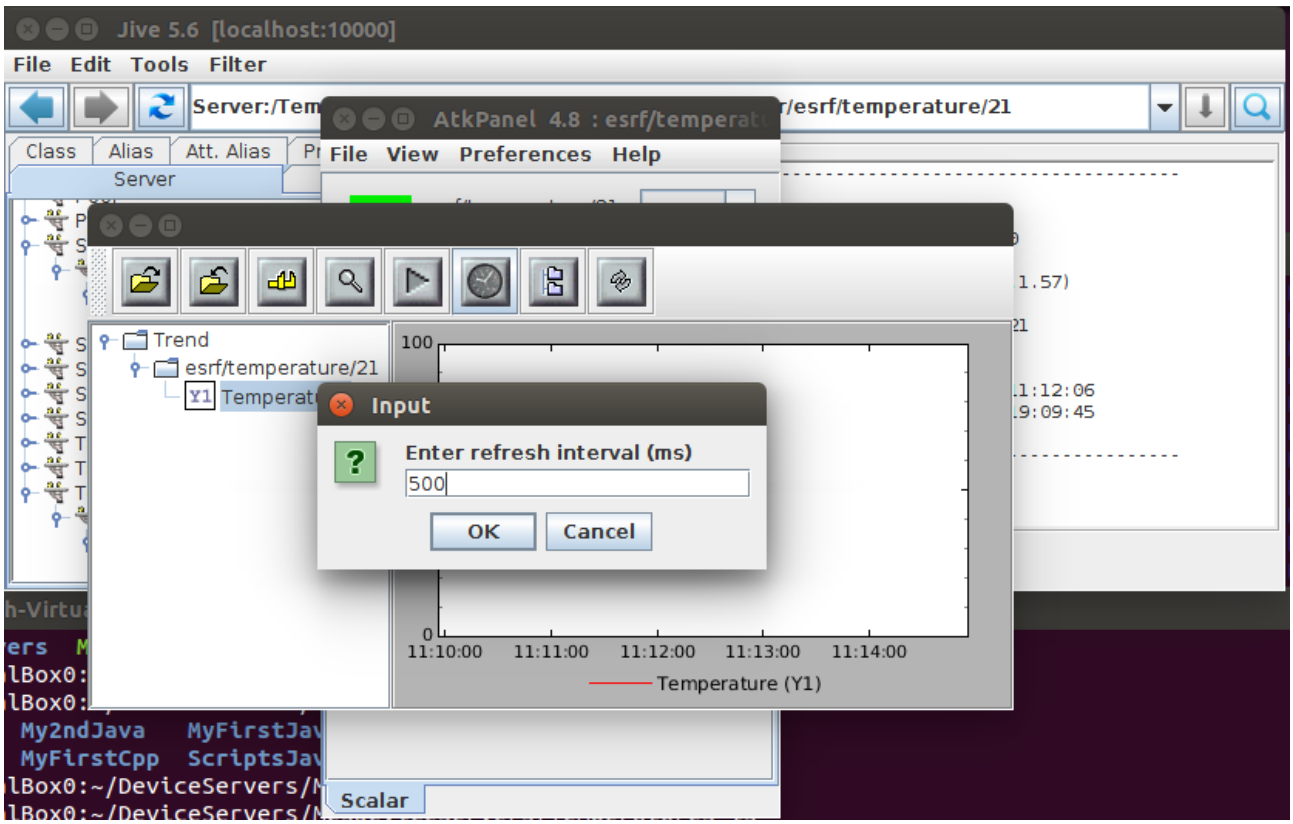
Dans jive, le clic droit sur le device et le choix menu Monitor affiche la fenêtre suivante.



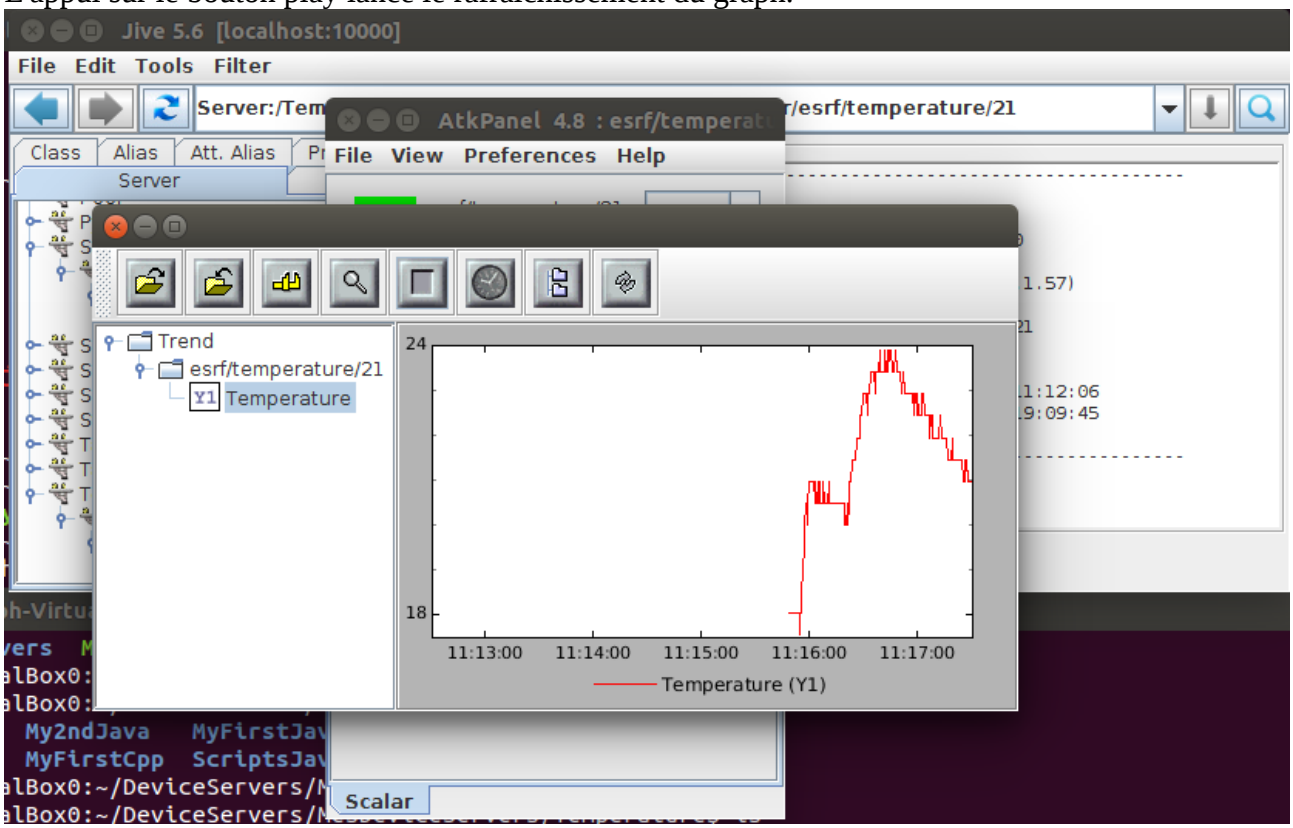
L'appui sur le bouton à coté de la valeur de température affiche la fenêtre suivante dans laquelle on peut éditer l'unité de mesure, des seuils d'alarmes...



le choix View → Numeric Trend ouvre un graph.
Le clic droit sur esrf/temperature/21 permet d'affecter cette valeur à l'axe Y1.
L'appui sur le symbole horloge permet ouvre une fenêtre qui permet de définir la périodicité de mise à jour du graph.



L'appui sur le bouton play lance le rafraîchissement du graph.



Variante sans utilisation du DS Serial, l'envoi des caractères se fait directement dans le DS TemperatureSensor

Avant

```
48
49 import PyTango
50 import sys
51 # Add additional import
52 #----- PROTECTED REGION ID(TemperatureSensor.additionnal_import) ENABLED START -----#
53
54 #----- PROTECTED REGION END -----#      //      TemperatureSensor.additionnal_import
55
```

Après

```
48
49 import PyTango
50 import sys
51 # Add additional import
52 #----- PROTECTED REGION ID(TemperatureSensor.additionnal_import) ENABLED START -----#
53
54 import serial
55
56 #----- PROTECTED REGION END -----#      //      TemperatureSensor.additionnal_import
57
```

Avant

```
80
81     def init_device(self):
82         self.debug_stream("In init_device()")
83         self.get_device_properties(self.get_device_class())
84         self.attr_Temperature_read = 0.0
85         #----- PROTECTED REGION ID(TemperatureSensor.init_device) ENABLED START -----#
86
87         #----- PROTECTED REGION END -----#      //      TemperatureSensor.init_device
88
```

Après

```
83     def init_device(self):
84         self.debug_stream("In init_device()")
85         self.get_device_properties(self.get_device_class())
86         self.attr_Temperature_read = 0.0
87         #----- PROTECTED REGION ID(TemperatureSensor.init_device) ENABLED START -----#
88
89         self.set_state(PyTango.DevState.ON)
90         self.ser=serial.Serial(self.SerialLine,9600)
91
92         #----- PROTECTED REGION END -----#      //      TemperatureSensor.init_device
93
```

Avant

```
95     #--------------------------------------------------------------------------
96     #    TemperatureSensor read/write attribute methods
97     #--------------------------------------------------------------------------
98
99     def read_Temperature(self, attr):
100        self.debug_stream("In read_Temperature()")
101        #----- PROTECTED REGION ID(TemperatureSensor.Temperature_read) ENABLED START -----#
102        attr.set_value(self.attr_Temperature_read)
103
104        #----- PROTECTED REGION END -----#      //      TemperatureSensor.Temperature_read
105
```

Après

```
100     #----------------------------------------------------------------------------
101     #    TemperatureSensor read/write attribute methods
102     #----------------------------------------------------------------------------
103
104     def read_Temperature(self, attr):
105         self.debug_stream("In read_Temperature()")
106         #----- PROTECTED REGION ID(TemperatureSensor.Temperature_read) ENABLED START -----#
107
108         self.ser.write("T".encode("utf8"))
109         answer=self.ser.readline()
110         stripped_answer=answer.rstrip()
111         self.attr_Temperature_read = float(stripped_answer)
112         attr.set_value(self.attr_Temperature_read)
113
114         #----- PROTECTED REGION END -----#      //      TemperatureSensor.Temperature_read
115
```

L'édition de la propriété est différente du 1er cas (ici on déclare directement le port serie)