```
#!/usr/bin/env python

"""
Random Value Property with Sleep

This application is a server of analog value objects that return a random
number when the present value is read.  This version has an additional
'sleep' time that slows down its performance.
"""

import threading
import eglobalsShared
import eglobalsCMM
import os
import random
import time

from bacpypes.debugging import bacpypes_debugging, ModuleLogger
from bacpypes.consolelogging import ConfigArgumentParser

from bacpypes.core import run

from bacpypes.primitivedata import Real
from bacpypes.object import AnalogValueObject, Property, register_object_type
from bacpypes.errors import ExecutionError

from bacpypes.app import BIPSimpleApplication
from bacpypes.local.device import LocalDeviceObject

# some debugging
_debug = 1
_log = ModuleLogger(globals())

# settings
SLEEP_TIME = float(os.getenv('SLEEP_TIME', 0.1))
RANDOM_OBJECT_COUNT = int(os.getenv('RANDOM_OBJECT_COUNT', 10))

# globals
args = None

#
#   RandomValueProperty
#

class RandomValueProperty(Property):

    def __init__(self, identifier):
        if _debug: RandomValueProperty._debug("__init__ %r", identifier)
        Property.__init__(self, identifier, Real, default=0.0, optional=True,
        mutable=False)
        print("BACnet: init ID: %r" %identifier )

    def ReadProperty(self, obj, arrayIndex=None):
        if _debug: RandomValueProperty._debug("ReadProperty %r arrayIndex=%r", obj,
        arrayIndex)
        value = 0.0
        try:
            type, instance = obj.objectIdentifier
            #print("BACnet: type=%r" %(type))
            #print("BACnet: objname=%s" %(obj.objectName))
            #print("BACnet: instance=%r" %(instance))
            if obj is not None:
                if( obj.objectName == "Outdoor Ambient Temp" ):
                    #print("BACnet: ReadProperty: obj rel Temp: %r" %obj.objectName )
                    value = eglobalsShared.masterAmbient
                elif( obj.objectName == "Outdoor Relative Humidity" ):
                    #print("BACnet: ReadProperty: obj rel hum: %r" %obj.objectName )
```

```
66                      value = eglobalsShared.masterRH
67                  elif( obj.objectName == "System Water Pressure" ):
68                      #print("BACnet: ReadProperty: obj water pressure: %r"
                        %obj.objectName )
69                      value = eglobalsShared.waterPressure
70                  elif( obj.objectName == "Sensor Data Storage Rate" ):
71                      #print("BACnet: ReadProperty: obj data storage rate : %r"
                        %obj.objectName )
72                      value = eglobalsShared.waterPressure
73                  elif( obj.objectName == "Internal excitation voltage" ):
74                      #print("BACnet: ReadProperty: obj data excitation voltage: %r"
                        %obj.objectName )
75                      value = eglobalsShared.masterRefVoltage
76                  else:
77                      for chan in range( 6 ):
78                          if( obj.objectName == "Channel %d Condenser Liquid Temp" %chan ):
79                              #print("BACnet: ReadProperty: obj data Condenser Liquid
                                Temp voltage: %r" \
80                                  #%obj.objectName )
81                              value = eglobalsShared.condenserLiqTemp[chan]
82                          elif( obj.objectName == "Channel %d Condenser Air Temp" %chan ):
83                              #print("BACnet: ReadProperty: obj data Condenser Air Temp
                                voltage: %r" \
84                                  #%obj.objectName )
85                              value = eglobalsShared.condenserAirTemp[chan]
86                          elif( obj.objectName == "Channel %d Compressor Power" %chan ):
87                              #print("BACnet: ReadProperty: obj data Compressor Power:
                                %r" \
88                                  #%obj.objectName )
89                              value = eglobalsShared.compressorPower[chan]
90                          elif( obj.objectName == "Channel %d Cumulative Spray Time"
                            %chan ):
91                              #print("BACnet: ReadProperty: obj data Cumulative Spray
                                Time: %r" \
92                                  #%obj.objectName )
93                              if( len(eglobalsShared.cumSprayTime) > chan ):
94                                  value = eglobalsShared.cumSprayTime[chan]
95                          #else:
96                              #print("BACnet: Could not find objName=%s" %obj.objectName )
97
98          except Exception, e:
99              print("BACnet: ReadProperty except %s" %e)
100
101         if arrayIndex is not None:
102             print("BACnet: ReadProperty: arrayIndex: %r" %arrayIndex )
103         global args
104         #print("BACnet: value=%f" %value)
105
106         # access an array
107         if arrayIndex is not None:
108             raise ExecutionError(errorClass='property', errorCode='propertyIsNotAnArray')
109
110         # sleep a little
111         time.sleep(args.sleep)
112
113         #if( arrayIndex == 1 ):
114             #print("BACnet: index =1")
115
116         # return a random value
117         #value = random.random() * 100.0
118         if _debug: RandomValueProperty._debug("    - value: %r", value)
119
120         return value
121
122     def WriteProperty(self, obj, value, arrayIndex=None, priority=None, direct=False):
123         print("TEC WriteProperty: value:")
124         print(value)
```

```
125             print(obj)
126             if _debug: RandomValueProperty._debug("WriteProperty %r %r arrayIndex=%r
                priority=%r direct=%r", obj, value, arrayIndex, priority, direct)
127             raise ExecutionError(errorClass='property', errorCode='writeAccessDenied')
128
129     bacpypes_debugging(RandomValueProperty)
130
131     #
132     #   Random Value Object Type
133     #
134
135     class RandomAnalogValueObject(AnalogValueObject):
136
137         properties = [
138             RandomValueProperty('presentValue'),
139             ]
140
141         def __init__(self, **kwargs):
142             if _debug: RandomAnalogValueObject._debug("__init__ %r", kwargs)
143             AnalogValueObject.__init__(self, **kwargs)
144
145     bacpypes_debugging(RandomAnalogValueObject)
146     print("RandomAnalogValueObject line 143")
147     register_object_type(RandomAnalogValueObject)
148
149     class BACnetService(threading.Thread):
150         # Bacnet Client service.
151
152         def __init__(self, eventObj=None):
153
154             threading.Thread.__init__(self)
155             threading.Thread.setDaemon(self, True)
156
157             self.evtObj = None
158             if( eventObj != None ):
159                 self.evtObj = eventObj
160
161             global args
162
163             # parse the command line arguments
164             parser = ConfigArgumentParser(description=__doc__)
165
166             # add an option to override the sleep time
167             parser.add_argument('--sleep', type=float,
168                 help="sleep before returning the value",
169                 default=SLEEP_TIME,
170                 )
171
172             # parse the command line arguments
173             args = parser.parse_args()
174
175             if _debug: _log.debug("initialization")
176             if _debug: _log.debug("    - args: %r", args)
177
178             # make a device object
179             this_device = LocalDeviceObject(ini=args.ini)
180             if _debug: _log.debug("    - this_device: %r", this_device)
181
182             ipv4 = os.popen('ip addr show eth0 | grep "\<inet\>" | awk \'{ print $2 }\' |
                awk -F "/" \'{ print $1 }\'').read().strip()
183             print("IP address test: %s" %ipv4 )
184
185             # make a sample application
186             this_application = BIPSimpleApplication(this_device, args.ini.address)
187
188             Peak_ObjectIDs = []
189             Peak_ObjectNames = []
```

```
        Peak_ObjectUnits = []

        Peak_ObjectIDs.append('analogValue' )
        Peak_ObjectNames.append('Outdoor Relative Humidity' )
        Peak_ObjectUnits.append('percentRelativeHumidity')

        Peak_ObjectIDs.append('analogValue' )
        Peak_ObjectNames.append('Outdoor Ambient Temp' )
        Peak_ObjectUnits.append('degreesFahrenheit')

        Peak_ObjectIDs.append('analogValue' )
        Peak_ObjectNames.append('System Water Pressure' )
        Peak_ObjectUnits.append('poundsForcePerSquareInch')

        Peak_ObjectIDs.append('analogValue' )
        Peak_ObjectNames.append('Sensor Data Storage Rate' )
        Peak_ObjectUnits.append('seconds')

        Peak_ObjectIDs.append('analogValue' )
        Peak_ObjectNames.append('Internal excitation voltage' )
        Peak_ObjectUnits.append('volts')

        # Now add Channel specific measurements. There should be 2
        for chan in range( 0, len(eglobalsShared.chanParamsDicts) ):
            vchan = eglobalsShared.chanVchanList[chan]
            print("BACnet: adding chan %d names" %vchan)
            instance = ((chan+1) * 10)

            Peak_ObjectIDs.append('analogValue' )
            Peak_ObjectNames.append('Channel %d Condenser Liquid Temp'%vchan )
            Peak_ObjectUnits.append('degreesFahrenheit')

            Peak_ObjectIDs.append('analogValue' )
            Peak_ObjectNames.append('Channel %d Condenser Air Temp'%vchan)
            Peak_ObjectUnits.append('degreesFahrenheit')

            Peak_ObjectIDs.append('analogValue' )
            Peak_ObjectNames.append('Channel %d Compressor Power' %vchan )
            Peak_ObjectUnits.append('watts')

            Peak_ObjectIDs.append('analogValue' )
            Peak_ObjectNames.append('Channel %d Cumulative Spray Time' %vchan )
            Peak_ObjectUnits.append('Seconds')

        # make some random input objects
        for i in range(len(Peak_ObjectIDs)):
            ravo = RandomAnalogValueObject(
                objectIdentifier=(Peak_ObjectIDs[i], i),
                objectName=Peak_ObjectNames[i], )
            _log.debug("    - ravo: %r", ravo)
            this_application.add_object(ravo)

        # make sure they are all there
        _log.debug("    - object list: %r", this_device.objectList)


        # TEC new May 2022
        print("RandomAnalogValueObject line 244")
        #register_object_type(RandomAnalogValueObject)
        # get the services supported
        services_supported = this_application.get_services_supported()
        if _debug: _log.debug("    - services_supported: %r", services_supported)
        print("services_supported:")
        print(services_supported)
        print(services_supported.value)

        # let the device object know. permission problem
```

```
257             #this_device.protocolServicesSupported = services_supported.value
258 
259             print("Bacnet thread init over")
260 
261         def run(self): # thread
262             _log.debug("running")
263 
264             run() # bacnet
265 
266             _log.debug("fini")
267             print("Bacnet thread run over")
268 
269 
270     def main():
271         bacnet = BACnetService()
272         if( 1 or ( eglobalsCMM.usingBACnet == 1 ) ):
273             bacnet.start()
274             print "BACnet Started"
275             while True:
276                 time.sleep(1.0)
277 
278     if __name__ == '__main__':
279         main()
280 
281 
```