How to use:

After adding the nodes to your catkin src folder, compiling it and starting roscore, follow the next commands.

```
user@localhost:~$ rosrun xela_server xConf.pyc
```
Configure the sensors prior to the first use.

```
user@localhost:~$ rosrun xela_server xServer.pyc
```
And in new tab/terminal window:

```
user@localhost:~$ rosrun xela_sensors xSensorService.pyc
```
To read the data from sensors, either subscribe to **/xServerPub** topic *(constant stream)* or call **xServ(XYZ) (sensor) (taxel)** (ex. rosservice call /xServXY 1 2) *(single query)*

Prerequisites:

Primary requirement is to run the code with Python 2.7 as the files have been pre-compiled and therefore give an error if run in different Python version. (You might see following error: RuntimeError: Bad magic number in .pyc file)

The following packages are required to run the sensor service and tools:

1) Tkinter
2) numpy
3) matplotlib
4) easygui
5) python-can

There are also optional packages that help to make the debugging easier by coloring the output. The packages for this would be:

1) coloredlogs
2) verboselogs

Nodes:

xela_sensors

A node providing publisher for sensor data (with requests)

this runs service that responds to queries

```
user@localhost:~$ rosservice call /xServXY 1 2
values: [16439, 16647]
user@localhost:~$ rosservice call /xServXYZ 2 6
values: [16451, 16517, 35901]
user@localhost:~$ rosservice call /xServX 2 1
value: 16681
user@localhost:~$ rosservice call /xServY 2 2
value: 16721
user@localhost:~$ rosservice call /xServZ 2 3
value: 37009
user@localhost:~$ rosservice call /xServStream 1
xyz: [1: [16457, 16553, 32057], 2: [16775, 16958, 31886] ]
```

xela_server

A node running server for all sensors. Required for xela_sensors node to operate.

xela_sim

A node including the service to connect to turtlebot gazebo simulation. It will use the 1st sensor (4x4 config) as controller. The code can also be seen as an example of usage

Example usage:

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

import rospy

from xela_sensors.srv import XelaSensorXYZ

import sys

rospy.init_node('use_service')

#wait the service to be advertised, otherwise the service use will fail
rospy.wait_for_service('xela_sensors')

#setup a local proxy for the service (we will ask for X,Y and Z data)
srv=rospy.ServiceProxy('xela_sensors',XelaSensorXYZ)

#use the service and send it a value. In this case, I am sending sensor: 1 and taxel: 3
service_example=srv(1,3)

#print the result from the service
print(service_example)
```