1 Introduction

The Kz (Kz is abbreviation of KoZo (小僧) which means apprentice it's labeled to Hosokawa-san in recent years). Python class library was developed to intend to use for manipulating robots or equipments over Ethernet much easily. This class library can be used with Python version 3. Unfortunately, Python 2 can't be used due to difference of multi-threading behaviors.

Most RORZE's robot is operated by Etnernet connection with host facilities so socket programming is very important skills for software development. But socket programming is not easy for many beginner programmers and engineers who uses programming language for making a tool for hardware development. Although a programming of socket is not easy but we need to through the way to get the goal of project.

List 1-1 shows typical client socket sample in C language.

The code is complex and long even need to much adjust for running with multi-thread operation.

In Kz class library, you need to write only few codes to talk with remote host. See below. List 1-2 is sample program of client socket application with using Kz class library in Python.

```
# -*- coding:utf-8 -*-
from KzClient import KzClient

class myClient(KzClient):
    def receiveEvt(self,d):
        rd = d.split(",")       # Convert string into array by comma.
        n = int(rd[-1])+1       # Retrieve last element in array and increment number.
        self.sendData("test," + str(n) + "\r\n")    # Send back data incremented to server.
        print(d)

cc = myClient()                 # Create instance of myClient class.
cc.connect("localhost",5000)    # Connet to localhost with port number 5000.
cc.sendData("test,1\r\n")       # Send data to server as initiation.

# This is a main loop.
# To prevent program termination, this infinite loop is necessary.
while True:
    pass
```

By using of this class library, code for background operations is hide in the class inside and programmer can concentrate their most critical parts which controling the robot behavior.

2 Basic architecture

Kz class library is adopting the object oriented theory. Server and client is declared as class and it's instantiated when application is running. These object is harmonizing to realize appropriate functionality with messaging and delegation. Fig-1 is diagram for basic structure of the Kz class library. It's snapshot of application running. Each objects invokes delegation at the appropriate timing so user can concentrate their most critical development on the project.

3 Classes

The Kz class library consists of 3 classes in 2 Python files. See table.

KzListener	KzServer.py
KzDefaultServer	KzServer.py
KzClient	KzClient.py

3.1 class KzClient

KzClient class connects to server and read/send data from/to remote host. This class makes thread for reading a response from a remote host. The KzClient class is base class so it's need to subclassing to implement user's unique behaviors.

3.1.1 def _init_(self)

Initializer for class instance. This method will be called automatically when the class is instantiated. Programmer must not call this method directly. In this method, instance variables using in this class inside are initialized. Programmer can customize this method by overridding, and initializes valiables declared by user. Don't forget to call the super class mtehod after initialization of user custom variables.

3.1.2 def connect(self,adrs,port)

Connects to specified remote host.

Parameters

adrs URL or IP address for target, with string representative

port port number for target host

Return value

No return value.

Discussion

The connect method connects socket to specified remote host. The method creates thread for reading data from remote host after connection has established normally. If the method failed to connect to remote host, method throws exception.

3.1.3 def sendData(self,d)

Send data to remote host.

Parameters

d Data for sending to remote host (string representation)

Return value

No return value.

Discussion

The sendData method give you the functionality of sending a data string representative to remote host. The parameter d should be terminated with CR/LF code.

This method may be called from multiple methods or codes in entire source project so programmer must wrap the code sending data with the block object to ensure coherency of data structures in application. The method calls the socket's send function directly, not pushes the data into cues.

3.1.4 def receiveEvt(self,d)

Class calls this method when the reading of data from the remote host was finished.

Parameters

d Data which has received from the remote host (string representation)

Return value

No return value.

Discussion

Class calls this method when the reading of data from the remote host was finished. The parameter d is string received delimited with CR/LF code. User can override this method with subclassing and can modify operation which is executed at time of data received. This method is called from thread so if the method needs to call other method or codes, programmer needs to take care of ensuring coherency of data structure.

3.1.5 def rcvLoop(self)

A loop for reading the data from the socket.

Parameters

No parameter.

Return value

No return value.

Discussion

This method is infinite loop (thread) for reading a data from the specified socket inside the class instance. The method reads a data from the buffer of socket and merge it until receiving the CR/LF code. The string which has built by merging a character until meeting to a CR/LF (as received data) will passed to the receiveEvt method. In normally, this method hasn't need to override in subclass.

3.2 class KzListener

KzListener class is used for in process of server application. The class gives you the functionality for making a socket and binding it to specified IP and port number then waits connection from clients. In normally, server application realizes functionality with combination of Listener class and default Server class. Programmer must customize server class to realize unique functionality of application.

3.2.1 def _init_(self)

Initializer for class instance. This method will be called automatically when the class is instantiated. Programmer must not call this method directly. In this method, instance variables using in this class inside are initialized. Programmer can customize this method by overriding, and initializes valiables declared by user. Don't forget to call the super after initialization of user custom variables.

3.2.2 def startListen(self,quemax)

Starts listening the connection from clients and accept it when client has requested to connect.

Parameters

quemax Maximum transaction number on listening process

Return value

No return value.

Discussion

This method starts listening the connection requests from the clients and accept it if possible. Once accepted the request, the method calls the reqServer method to get appropriate class instance for KzDefaultServer (or subclass). After getting the instance of server, class starts receiver thread and call Main method as thread too. Do not call this method directly.

3.2.3 def addServer(self,d)

(Depricated) This method is no longer supported.

3.2.4 def reqServer(self)

Class calls this method when the class needs to get instance of server class.

Parameters

No parameter.

Return value

Returns instance of server class. If there is not enough resource to make instance, return None.

Discussion

Class calls this method when the class needs to get instance of server class. Programmer should make instance of server class and return it. If the instance is not be created in some reasons, return None but when class received None, it will throw exception and program will be terminated. In subclass of KzListener, at least this method need to be overriden.

3.2.5 def runListener(self,quemax)

Starts listening a connection request from clients.

Parameters

quemax Maximum transaction number on listening process

Return value

No return value.

Discussion

This method starts listening a requests from the clients and accept it if possible. Once accepted the request, the method creates server process and start it.

3.3 class KzDefaultServer

KzDefaultServer is abstract class for server functionality. This class has basic read/send method which access to client. In normally, this class will be subclassed and method is overridden for implementing an application specific behavior.

3.3.1 def runRcvLoop(self)

This method starts thread for reading data from the socket.

Parameters

No parameter.

Return value

No return value.

Discussion

This method is called at the time of client has connected successfully, by Listener class automatically. The method creates and starts thread for reading data from the socket by paralellized with main thread. This method is invoked from class inside automatically, so do not call this method directly from user's code.

3.3.2 def setSocket(self,d)

This method starts thread for reading data from the socket.

Parameters

d Socket for conversation with client

Return value

No return value.

Discussion

This method is accessor for reference of the socket which can be used for accessing to client connected. The method is invoked by class internally at the time of establishing a connection with client. User code should not invokes this method directly.

3.3.3 def runThread(self)

Thread for reading data from the client.

Parameters

No parameter

Return value

No return value.

Discussion

This method is thread for reading data from the client connected currently. The method is invoked by the runRcvLoop method and retrieve a data from the reading buffer of socket then throw data to receiver event method when data reading has finished.

3.3.4 def receiveEvt(self,d)

An event (delegate) method for operation of data from the client.

Parameters

d Data from the client (string representation)

Return value

No return value.

Discussion

Class delegates an operation when reading a data from the client has finished. In normally, user code overrides this method in subclass.

Parameter d contains string which was sent from client. The string does not have end character such as CR/LF. User code can implement user's custom operation in this method by overriding this method in subclass of KzDefaultServer class.

3.3.5 def sendData(self,d)

Send data (string representation) to the client.

Parameters

d Data to send to the client

Return value

No return value

Discussion

Send string data to the client currently connected. The string can not contain the wide-character such as Japanese Kanji. If user code try to send wide-character, this method throws an exception. An exception does not catch in this method, so eventually application will be terminated with error. This method is executed as synchronized so execution of next code will be blocked until sending become finished.

3.3.6 def subMainCaller(self)

This method starts apparently user main function.

Parameters

No parameter

Return value

No return value.

Discussion

This method invoke thread to start the user main method which can be used for customization by user.

3.3.7 def Main(self)

The main method for implementation of custom server operation.

Parameters

No parameter

Return value

No return value.

Discussion

This method is one of important functionality on developing the server application with using this class library. In case of C language, program will be started from a main function. Like this, server application on this class library will be started at Main method. User can write the user's own sequences and programs on this main method to customize server behavior required.

For example, most robot operation software have some sequence of program execution such as command and so on. By coding the pair of a command sending and confirming the response in this method, your application can operate robot along with user requirements.

3.4 Multiple instances for server

In normally, server serves multiple clients at the same time. It allows multiple connection from the clients. To realize this behavior, this class library uses object oriented architecture over the whole of library.

[Oh! very sorry... Documentation is undergoing yet.]