FBot V2.0版本使用说明文档

更新时间：2017年09月01日14:53:53

1. 环境配置：

* 安装Python3.5环境(推荐Python3.5.3);
* 安装“TYBotSDK2-2.0.14-py3-none-any.whl”:

**pip install TYBotSDK2-2.0.14-py3-none-any.whl**

1. 在代码中引入包，如下：

from TYBotSDK2.FBot.fbotV2 import FBot

1. 构建并使用FBot对象：

FBot初始化参数说明：

class FBot:  
 def \_\_init\_\_(self, connect\_to\_socks={}, loop\_event\_callbacks={}, listen\_socks=[],listen\_accpet\_callback=None,  
 config\_file="config/config.ini", report\_status=False, report\_interval=60, comment=""):  
 *'''  
 功能Bot初始化对象* ***:param*** *connect\_to\_socks: 需要连接的套管对象集合，字典格式{"连接端的惟一ID":创建连接对应的指令值}* ***:param*** *loop\_event\_callbacks: 需要在每次主轮询时执行的函数回调，一般用于数据库检测操作* ***:param*** *listen\_socks: 需要监听的套管对象ID集合，List格式[ID]* ***:param*** *config\_file: 配置文件路径，默认为"./config/config.ini"* ***:param*** *report\_status: 是否向节点状态监测节点发送节点状态信息:True-发送当前节点状态信息，False-不发送当前节点状态信息，默认为False* ***:param*** *report\_interval: 状态汇报时间间隔，单位秒，默认60秒* ***:param*** *comment: 节点描述/备注，用于描述节点功能说明，默认为""  
 '''*

Brain端【示例代码-brain\_side.py】：

第一步：新建FBot对象。

bot=FBot(listen\_socks=[900,1100],listen\_accpet\_callback=listen\_accept\_callback,report\_status=True,comment="任务大脑调度")

listen\_socks：大脑端需要构建的监听端口集合，List类型，每一个数字ID代表一个端口，在业务中代表一个类型的指令ID；

listen\_accpet\_callback：在监听管道接收到客户端连接时的回调函数，改回调函数会以字典形式传回接收到的新的连接客户端的相关参数值，具体包含如下：

listen\_sock\_id:接收到连接的sock id

peer\_name:连接进来的客户端的ID名

peer\_port:连接进来的客户端的通信端口

client\_index:连接进来的客户端的索引ID

report\_status：是否汇报节点状态到状态展示节点，True-汇报，False-不汇报；

comment：通信节点备注描述，主要用于节点状态展示用；

第二步：绑定指令回调处理函数。

bot.add\_listen\_callbacks( 901,protocol\_state\_B)  
bot.add\_listen\_callbacks( 903,protocol\_state\_D)  
bot.add\_listen\_callbacks( 1101, protocol\_state\_B2)  
bot.add\_listen\_callbacks( 1103, protocol\_state\_D2)  
bot.add\_listen\_callbacks(1105, protocol\_state\_E2)

指令支持自定义多级指令回调，可添加多种指令以及指令之间的应答，逻辑可由使用者灵活控制。

绑定指令回调处理函数说明

def add\_listen\_callbacks(self, cmd, callback):  
 *'''  
 添加服务端连接数据交互回调处理函数* ***:param*** *cmd: 回调函数对应的指令ID* ***:param*** *callback: 回调函数* ***:return****: 无  
 '''*

Callback原型形式：

def callback(peerName,value, param1,param2,is\_last\_package):  
 *'''  
 回调处理函数* ***:param*** *peerName: 回调任务远端节点名称* ***:param*** *value: 参数Value,内容由用户根据协议自定义* ***:param*** *param1: 参数Value,内容由用户根据协议自定义* ***:param*** *param2: 参数Value,内容由用户根据协议自定义* ***:param*** *is\_last\_package: 是否是最后一个数据包（处理的数据包可能是一批，也可能是单个）* ***:return****: 处理完成后需要发送应答的数据包或数据包List集合  
 '''*

第三步：运行FBot功能实例。

bot.run()

FBot端【示例代码-func\_bot\_add\_side.py】：

第一步：新建FBot对象。

bot = FBot(connect\_to\_socks={"BrainA": [900, 1100]}, report\_status=True, comment="执行加法计算")

connect\_to\_socks:功能FBot需要连接的Brain端参数，字典结构{Brain端ID名：[Brain端端口1，Brain端口2...]}

其他参数同Brain端。

第二步：绑定指令回调处理函数。

bot.add\_connect\_to\_callbacks("BrainA", 900, 901, protocol\_state\_A, True)  
bot.add\_connect\_to\_callbacks("BrainA", 900, 902, protocol\_state\_C, False)  
  
bot.add\_connect\_to\_callbacks("BrainA", 1100, 1101, protocol\_state\_A2, True)  
bot.add\_connect\_to\_callbacks("BrainA", 1100, 1102, protocol\_state\_C2, False)  
bot.add\_connect\_to\_callbacks("BrainA", 1100, 1104, protocol\_state\_E2, False)

指令支持自定义多级指令回调，可添加多种指令以及指令之间的应答，逻辑可由使用者灵活控制。

绑定指令回调处理函数说明

def add\_connect\_to\_callbacks(self, sock\_id,sock\_port, cmd\_id, callback,is\_auto\_callback=False):  
 *'''  
 添加客户端连接数据交互回调处理函数* ***:param*** *sock\_id: 对应的套管的客户端的唯一ID名* ***:param*** *sock\_port: 对应的连接套管的端口号* ***:param*** *cmd\_id: 回调对应的协议ID* ***:param*** *callback: 回调函数* ***:param*** *is\_auto\_callback: 是否为主动发送回调,主动发送的回调会定时自动触发. True-是主动触发回调, False-不是主动触发回调* ***:return****: 无  
 '''*

第三步：运行FBot功能实例。

bot.run()

完整例子参考：FBotV2Examples。