py脚本说明

## reply\_mytext.py

**实现的是回复一条自定义的信息。这是回复的就是：我是机器人小小，谢谢！**

#coding=utf8

import requests

import itchat

# 带对象参数注册，对应消息对象将调用该方法

@itchat.msg\_register(itchat.content.TEXT, isFriendChat=True, isGroupChat=True, isMpChat=True)

def text\_reply(msg):

itchat.send(u'我是机器人小小，谢谢！',msg['FromUserName'])

# 为了让实验过程更加方便（修改程序不用多次扫码），我们使用热启动

itchat.auto\_login(hotReload=True)

itchat.run()

## 2. reply\_to\_filehelper.py

**实现的是在私聊或群聊时将一条信息发送给文件传输助手。**

#coding=utf8

import requests

import itchat

# 带对象参数注册，对应消息对象将调用该方法

@itchat.msg\_register(itchat.content.TEXT, isFriendChat=True, isGroupChat=True, isMpChat=True)

def text\_reply(msg):

# 注意实验楼环境的中文输入切换

itchat.send('%s: %s' % (msg['FromUserName'], msg['Text']), 'filehelper')

# 为了让实验过程更加方便（修改程序不用多次扫码），我们使用热启动

itchat.auto\_login(hotReload=True)

itchat.run()

## noreply\_listen\_private\_group

**实现的是将群聊、私聊的信息全部转发到文件传输助手，这样就避免了对方或群里面的人将信息撤回。不过这是将全部信息转发，而不是再有人撤回时再转发，这样会导致文件传输助手的信息很多，需要完善。**

#coding=utf8

import itchat

from itchat.content import \*

import requests

import time; # 引入time模块

@itchat.msg\_register([PICTURE, RECORDING, ATTACHMENT, VIDEO])

def download\_files(msg):

msg['Text'](msg['FileName'])

return '@%s@%s' % ({'Picture': 'img', 'Video': 'vid'}.get(msg['Type'], 'fil'), msg['FileName'])

@itchat.msg\_register(TEXT,isFriendChat=False, isGroupChat=True)

def text\_reply(msg):

# 发送给文件助手，防止对方撤回

itchat.send('%s: %s' % (msg['ActualNickName'], msg['Text']), 'filehelper')

@itchat.msg\_register(TEXT,isFriendChat=True,isGroupChat=False)

def text\_reply(msg):

# 发送给文件助手，防止对方撤回

# itchat.send('%s: %s' % (msg['ActualNickName'], msg['Text']), 'filehelper')

# 将时间戳转化为localtime

x = time.localtime(msg['CreateTime'])

createTime = time.strftime('%Y-%m-%d %H:%M:%S',x)

itchat.send('FromUserName:%s\nToUserName:%s\n\nCreateTime: %s\nText:%s' % (msg['FromUserName'],msg['ToUserName'],createTime, msg['Text']), 'filehelper')

itchat.auto\_login(hotReload=True)

itchat.run()

## 4. simple\_tuling\_reply\_private.py

**实现接入了图灵机器人，在私聊时，可以进行智能回复。当然，这是对所有私聊都进行智能回复，但群聊没有实现。**

#coding=utf8

import requests

import itchat

KEY = '4019b0eb4728489e9f537872ab18f00c'

def get\_response(msg):

# 这里我们就像在“3. 实现最简单的与图灵机器人的交互”中做的一样

# 构造了要发送给服务器的数据

apiUrl = 'http://www.tuling123.com/openapi/api'

data = {

'key' : KEY,

'info' : msg,

'userid' : 'wechat-robot',

}

try:

r = requests.post(apiUrl, data=data).json()

# 字典的get方法在字典没有'text'值的时候会返回None而不会抛出异常

return r.get('text')

# 为了防止服务器没有正常响应导致程序异常退出，这里用try-except捕获了异常

# 如果服务器没能正常交互（返回非json或无法连接），那么就会进入下面的return

except:

# 将会返回一个None

return

# 这里是我们在“1. 实现微信消息的获取”中已经用到过的同样的注册方法

@itchat.msg\_register(itchat.content.TEXT)

def tuling\_reply(msg):

# 为了保证在图灵Key出现问题的时候仍旧可以回复，这里设置一个默认回复

defaultReply = 'I received: ' + msg['Text']

# 如果图灵Key出现问题，那么reply将会是None

reply = get\_response(msg['Text'])

# a or b的意思是，如果a有内容，那么返回a，否则返回b

# 有内容一般就是指非空或者非None，你可以用`if a: print('True')`来测试

return reply or defaultReply

# 为了让实验过程更加方便（修改程序不用多次扫码），我们使用热启动

itchat.auto\_login(hotReload=True)

itchat.run()

## 5. tuling\_reply\_group.py

实现的是在群聊中，只有被@的时候才会智能地自动回复。

import itchat

from itchat.content import \*

import requests

KEY = '4019b0eb4728489e9f537872ab18f00c'

def get\_response(msg):

apiUrl = 'http://www.tuling123.com/openapi/api'

data = {

'key' : KEY,

'info' : msg,

'userid' : 'wechat-robot',

}

try:

r = requests.post(apiUrl, data=data).json()

return r.get('text')

except:

return

@itchat.msg\_register([PICTURE, RECORDING, ATTACHMENT, VIDEO])

def download\_files(msg):

msg['Text'](msg['FileName'])

return '@%s@%s' % ({'Picture': 'img', 'Video': 'vid'}.get(msg['Type'], 'fil'), msg['FileName'])

@itchat.msg\_register(TEXT, isGroupChat=True)

def text\_reply(msg):

if msg['isAt']:

defaultReply = 'I received: ' + msg['Text']

reply = get\_response(msg['Text'])

return reply or defaultReply

itchat.auto\_login(hotReload=True)

itchat.run()

## 6. tuling\_reply\_private\_group\_mytext.py

实现了更自定义的自动回复：在群聊中，只是将信息转发到文件传输助手，而在私聊中，不同的人在第一次私聊时，会先回复自定义的信息，本例中是“我是智能机器人小小，主人在闭关学习了。有急事请拨打电……”，接下来才进入了智能自动回复，这样就可以在第一次别人第一次私聊自己，但自己又不在时可以回复自定义信息，避免了智能回复的突兀。二十分钟后，如果同一个人再次私聊，也会再次先回复自定义信息。

#coding=utf8

import itchat

from itchat.content import \*

import requests

import time; # 引入time模块

KEY = '4019b0eb4728489e9f537872ab18f00c'

userDict = {}

userList = []

userFirstChatDateList = []

def get\_response(msg):

apiUrl = 'http://www.tuling123.com/openapi/api'

data = {

'key' : KEY,

'info' : msg,

'userid' : 'wechat-robot',

}

try:

r = requests.post(apiUrl, data=data).json()

return r.get('text')

except:

return

@itchat.msg\_register([PICTURE, RECORDING, ATTACHMENT, VIDEO])

def download\_files(msg):

msg['Text'](msg['FileName'])

return '@%s@%s' % ({'Picture': 'img', 'Video': 'vid'}.get(msg['Type'], 'fil'), msg['FileName'])

@itchat.msg\_register(TEXT,isFriendChat=False, isGroupChat=True)

def text\_reply(msg):

# 发送给文件助手，防止对方撤回

itchat.send('%s: %s' % (msg['ActualNickName'], msg['Text']), 'filehelper')

# 获取名字中含有特定字符的群聊，返回值为一个字典的列表

# list = itchat.search\_chatrooms(name=u'测试一下')

# for index in list:

# print index

# 如果有人@我

# if msg['isAt']:

# 发送给文件助手，防止对方撤回

# itchat.send('%s: %s' % (msg['ActualNickName'], msg['Text']), 'filehelper')

# defaultReply = u'你好！'

# reply = u'小小：'+get\_response(msg['Text'])+'\n \n'+u'PS：我是智能机器人小小，主人在闭关学习了。有急事请拨打电话18826077893，或者发送电子邮件1214914477@qq.com。不@就不会自动回复。'

# return reply or defaultReply

@itchat.msg\_register(TEXT,isFriendChat=True,isGroupChat=False)

def text\_reply(msg):

# 发送给文件助手，防止对方撤回

# itchat.send('%s: %s' % (msg['ActualNickName'], msg['Text']), 'filehelper')

# 将时间戳转化为localtime

x = time.localtime(msg['CreateTime'])

createTime = time.strftime('%Y-%m-%d %H:%M:%S',x)

itchat.send('FromUserName:%s\nToUserName:%s\n\nCreateTime: %s\nText:%s' % (msg['FromUserName'],msg['ToUserName'],createTime, msg['Text']), 'filehelper')

defaultReply = u'你好，智能机器人小小为你服务！'

defaultReply2 = u'你好，我是智能机器人小小！主人在闭关学习，一般很少查看一下微信的信息。由此带来不便，请谅解！如有急事，请致电或发短信至18826077893。谢谢！\n \n接下来由小小为你服务：小小会讲故事、讲笑话、成语接龙、做数学运算、新闻资讯、星座运势和聊天对话......'

global userDict

global userList

global userFirstChatDateList

# 获取当前时间

ticks = time.time()

# 取出最后一次聊天时的userList和userFirstChatDateList

# currentUserList = userDict['userList']

# currentUserFirstChatDateList = userDict['userFirstChatDateList']

# 先把字典中上一次聊天时间与当前时间距离超过20分钟的去掉

for index in range(len(userFirstChatDateList)):

lastChatDate = userFirstChatDateList[index]

print lastChatDate

print ticks

if (ticks - lastChatDate) >= 1200 :

# 去掉与该时间

del userFirstChatDateList[index]

# 去掉与该时间对应的user

del userList[index]

# userList该列表维护的是第一次聊天了，但是时间还没超过20分钟的。简单来说，就是直接进行智能应答的列表

# 如果在用户列表中，是同一个用户，而且不是第一次聊天，时间也还没超过20分钟的，就直接进行智能应答

if msg['FromUserName'] in userList :

reply = get\_response(msg['Text'])

return reply or defaultReply

# 如果不在用户列表中

else :

# 添加用户

userList.append(msg['FromUserName'])

# 添加时间

print ticks

userFirstChatDateList.append(ticks)

# 字典中更新用户和用户聊天时间

userDict={'userList':userList,'userFirstChatDateList':userFirstChatDateList}

# 返回默认的回复2

return defaultReply2

itchat.auto\_login(hotReload=True)

itchat.run()