1.修改

1.1GUI的修改

在右方增设一个控制台，更好的向用户展示信息。

代码实现

添加已给textEdit并设置属性

```
    self.textEdit = QtWidgets.QTextEdit(self.centralwidget)
    self.textEdit.setGeometry(QtCore.QRect(314, 10,314,748))
    self.textEdit.setObjectName("textEdit")
    self.textEdit.setStyleSheet("background-color: Black ")
    self.textEdit.setReadOnly(True)#

 self.textEdit.verticalScrollBar().setValue(self.textEdit.verticalScrollBar().ma
ximum())
```

在UI类添加输出函数：

```
    def print(self,string):
        self.textEdit.append("<font color=\"#F8F8FF\">"+string+"</font> ");
```

因为是使用另一个监听线程去向UI线程输出，所以需要使用信号来实现

```
_signal=QtCore.pyqtSignal(str)          #添加信号
_signal.connect(self.print)             #绑定信号到print函数
```

通过emit()函数即可完成输出

```
self._signal.emit("Hi! How can I help?")    #调用信号的emit函数进行输出
```

本来想添加一个背景图片，但是感觉加了之后更丑了，便没有设置

```
MainWindow.setStyleSheet("#MainWindow{border-image:url(./icon/bg.jpg);}")
```

1.2代码修改

一共增加了三个类

1. LAEThread

   定义在listenAndExecuteThread.py中，继承与Thread类，通过它来新建一个线程。监听麦克风输入，并调用Recongnizer中的百度语音识别接口，结果如果成功返回，则交给Orders去处理，通过re来判断是否执行某条命令

2. Orders

通过正则表达式来判断识别结果和命令的匹配性。通过win32api来执行每条命令。

3. Recognizer

提供给外界recorder_and_rec(self,language)方法，

```
    def recorder_and_rec(self,language):
        frames_data=self.record(5,"./temp.wav")
        return self.get_result(frames_data,'wav',self.rate,language)
```

外界需要提供参数language，language为语言的代号：百度提供接口支持的语言如下：

dev_pid	语言	模型	是否有标点	备注
1537	普通话(纯中文识别)	输入法模型	有标点	支持自定义词库
1737	英语	英语模型	无标点	不支持自定义词库
1637	粤语	粤语模型	有标点	不支持自定义词库
1837	四川话	四川话模型	有标点	不支持自定义词库
1936	普通话远场	远场模型	有标点	不支持自定义词库

该方法调用record来记录n秒的语音暂存到temp.wav中，然后调用get_result获取百度接口结果。（具体实现看代码）

语音识别准确性

使用speech_recognition库的语音的识别准确性很低，只有偶然成功了一次打开音乐。主要原因是发音要求比较高，不能满足其识别的条件，其本身识别效果也不是很好。

于是使用腾讯的语音识别接口，但是他官方SDK库有问题，但是腾讯的官方库有问题，因为python版本的原因，无法成功导入SSL，因此没有成功实现腾讯语音识别接口。于是转向百度的语音识别接口，对于中文识别成功率很高，几乎每次都可以成功。