DSL领域特定语言设计 ——chatbot

语法设计

词法规则

该语言的单词分为以下类型:

关键字(KEY)、数据类型符号(DTYPE)、程序块符号(BLOCK)、注释符号(NOTE)、运算符号(OP)、数字(NUM)、字符串（STRING）、界符(BOUND)、标识符(ID)

所有单词，以空格分隔

关键字 KEY

1. **Def**:
   全局变量的定义，只允许在程序入口之前定义
2. **Step**:
   模块/程序的开始，完整表示一个步骤的所有行为
3. **End_step**:
   模块/程序的结束，完整表示一个步骤的结尾
4. **Speak**:
   输出计算表达式合成一段文字，调用媒体服务器进行语音合成并播放
   简化为输出一行字符串
5. **Listen**:
   调用媒体服务器对客户说的话录音，并进行语音识别
   语音识别的结果调用“自然语言分析服务”分析客户的意愿
   简化为接受一句来自命令行的输入
6. **Exp**:
   表达式关键字，表明其后是一条表达式， 进行运算，赋值等操作
7. **Read**:
   脚本的文件读操作
8. **Write**:
   脚本的文件写操作

9. **Slience**:
   暂停,等待一段时间
10. **Switch**:
    分支语句的开始
11. **End_switch**:
    分支语句的结束
12. **Branch**:
    对客户的意愿进行分支处理，不同的意愿，跳转到不同的Step
13. **Default**:
    如果客户意愿没有相应匹配，应该跳转到哪个Step
14. **Go**:
    调用另一个程序块
15. **Exit**:
    结束对话，程序结束

数据类型符号 DTYPE

1. **Num**: 浮点数
2. **Str**: 字符串

函数块（程序块符号）BLOCK

由_（下划线）开头的符号

注释符号 NOTE

符号#
只有# 且在一行的开头，作用范围也是一行

数字 NUM

仅有0~9和.组成的数值

字符（串）STRING

以""包括的符号

界符 BOUNDER

暂无

标识符 ID

即由用户定义的变量符号，只能以小写字母组成

运算符 OP

分别有
数值运算符+ - * /
赋值符号=

语法规则

1. 只允许定义全局变量，且只支持在入口Step之前定义；

2. 全局变量也可以由文件引入；
但是不会检查该指定文件的格式是否符合规则，可能会引入错误

3. 每行必须由关键字开始，最多只能有一个关键字；

4. 标识符，程序块符号不可以重复定义；

5. 脚本第一个模块（Step）为脚本主模块，该模块执行完毕（或遇到结束关键字（Exit）脚本结束；

6. Switch语句不允许嵌套；

语句规则

1. **空语句:**
   格式：
   说明：只是一个空行
   语义：无，解释执行时跳过
2. **注释语句:**
   格式：# ANY
   说明：#后可以跟任意内容
   语义：无语义动作，解释执行时跳过
   举例：
   ```
   # 这是一行注释，这里可以有任意内容
   ```
3. **Def 语句:**
   格式：Def DTYPE ID NUM/STRING
   说明：Def后只能跟3个符号，数据类型符号，标识符，初值（数字/字符串）
   语义：定义变量，设定类型和初值
   举例：

```
Def Num money 100
# 该语句定义了一个Num类型的变量，初值为100
```

4. **Step语句**
   格式：Step BLOCK
   说明：Step后只能跟1个程序块符号
   语义：定义程序块，表明程序块入口，并且将栈的当前层次+1
   举例：

```
Step _welcome
# 该语句定义了一个程序块_welcome，并且层次+1
```

5. **End_step语句**
   格式：End_step
   说明：End_step 后不接任何符号
   语义：表明当前程序块结束，并且将栈的当前层次-1
   举例：

```
End_step
# 该语句表明当前程序块结束
```

6. **Speak语句**
   格式：Speak ID/STRING
   说明：Speak 后只能跟1个标识符或者字符串
   语义：将标识符的值输出到屏幕（命令行或重定向到文件）
   举例：

```
Speak "北邮欢迎你"
# 该语句将输出：北邮欢迎你
Speak money
# 该语句将输出标识符money的值
```

7. **Listen语句**
   格式：Listen ID
   说明：Listen 后只能跟1个标识符
   语义：接收输入（命令行/重定向文件），将标识符ID的值修改为接收的内容
   举例：

```
Listen complaint
# 该语句将输入内容保存到标识符complaint
```

8. **Exp语句**
   格式1：Exp ID = ID/STRING/NUM
   格式2：Exp ID = ID/NUM OP ID/NUM

格式3：Exp ID = ID/STRING/NUM + ID/NUM/STRING
说明：单纯的赋值表达式由3个符号组成：标识符 = 标识符/数字/字符串
运算表达式5个符号组成： 标识符 = 标识符/数字 运算符 标识符/数字（只支持数值运算）
字符串运算表达式由5个符号组成：标识符 = 标识符/字符串/数字 + 标识符/字符串/数字，即提供拼接功能
语义：根据等号左边的标识符类型确定赋值、运算、字符串拼接
举例：

```
Exp money = 100
# 该语句将标识符赋值为100
Exp money = 100 - 20
# 该语句先计算100-20的结果，得到结果80，再将其赋值给，money
Exp sentence = sentence + "再见"
# 该语句将字符串标识符sentence与"再见字符串"拼接，再赋值给原标识符
```

9. **Read语句**
格式：Read STRING/ID NUM ID
说明：Read后跟3个符号:字符串/标识符+行号(数字)+标识符，第一个参数是文件名（相对路径） 第二个参数是行号 第三个用来接受读入的信息
语义：打开文件，定位到相应行，读出一行内容，保存到相应标识符中
举例：

```
Read "rcx.txt" 2 bill
# 该语句将文件rcx.txt的第二行内容读到标识符bill中
```

10. **Write语句**
格式：Write STRING/ID STRING/ID NUM
说明：Write后跟3个符号，第一个字符串/标识符指明文件名，第二个字符串/标识符指明写入信息，第三个数字是写入方式： 0覆盖，1追加
语义：打开文件，按照指定的写入方式，写入一行数据
举例：

```
Write "rcx.txt" "20" 1
# 该语句打开文件rcx.txt，在末尾追加一行，内容为：20
```

11. **Silence语句**
格式：Silence NUM
说明：Slience 后接1个符号,数值符号，表明等待多少时间
语义：根据数值，暂停等待一段时间
举例：

```
Silence 10
# 该语句暂停等待10秒
```

12. **Switch语句**

    格式：Switch ID

    说明：Switch 后跟1个标识符，代表将用其进行判断

    语义：开启当前层次的比较标志，允许执行Branch和Default语句，将标识符的内容暂存到栈的当前层次信息中，当遇到Branch或者Default语句时进行比较、选择与跳转

    举例：

    ```
    Switch sentence
    # 该语句首先开启当前Step的sflag(置为1),并将sentence的内容保存到当前层次信息smember中
    ```

13. **End_switch语句**

    格式：End_switch

    说明：End_switch 后不接任何符号

    语义：关闭当前层次的比较标志，表明不再允许Branch和Default语句执行

    举例：

    ```
    End_switch
    # 该语句关闭当前Step的sflag(置为0)
    ```

14. **Branch语句**

    格式：Branch STRING/ID BLOCK

    说明：Branch 后跟2个符号，第一个为字符串或者标识符，第二个为程序块符号

    语义：只有在当前层次比较标志开启时才会执行该语句，将标识符/字符串内容和当前层次信息的相关内容比较，表明匹配将跳转到哪个程序块执行，若执行成功，关闭当前层次比较标志

    举例：

    ```
    Branch "投诉" _complaint
    # 该语句先判断当前层次sflag是否为1，
    # 为1且"投诉"与当前层次比较信息smember匹配，就跳转到_complaint模块，跳转成功就关闭该层sflag(置为0)
    ```

15. **Default语句**

    格式：Default BLOCK

    说明：Default 后跟1个程序块符号

    语义：只有在当前层次比较标志开启时才会执行该语句，表明无匹配时跳转到哪个块，若执行成功，关闭当前层次比较标志

    举例：

    ```
    Default _thanks
    # 该语句先判断当前层次sflag是否为1，为1就跳转到_complaint模块，跳转成功就关闭该层sflag(置为0)
    ```

16. **Go语句**

    格式：GO BLOCK

    说明：Go 后跟1个程序块符号

语义：直接跳转到该程序块

举例：

```
Go _welcome
# 直接跳转到_welcome程序块
```

17. **Exit语句**

格式：Exit

说明：Exit 后不接任何符号

语义：表明程序中止

举例：

```
Exit
# 该语句表明程序/脚本到此结束
```