### 一、Python常用的模块

Re正则表达式

re.findall()

re.match()

re.research()

re.find()

os

os.getchwd()返回当前目录

os.chmod()更改权限

os.mkdir()

os.remove()

os.path.basename(path)返回文件名

os.path.dirname()返回目录名

os.path.listdir()返回当前目录下所有的文件或文件夹列表

os.path.split() 将文件夹和文件名拆开

os.path.join(a,b) 将文件夹和文件名合并

os.path.abspath('.') 返回当前目录

os.path.dirname(os.path.abspath('.')) 返回当前目录的上级目录名

os.path.abspath(\_\_file\_\_) 返回当前文件的路径，包含文件自身

time

time.time() 获取时间戳

time.localtime() 获取一段很长的时间

time.strftime(‘’”,time.localtime()) 格式化时间

logging

getLogger()

Handler

Formatter

每个Logger可以附加多个Handler。接下来我们就来介绍一些常用的Handler：  
1) logging.StreamHandler  
使用这个Handler可以向类似与sys.stdout或者sys.stderr的任何文件对象(file object)输出信息。它的构造函数是：  
StreamHandler([strm])  
其中strm参数是一个文件对象。默认是sys.stderr

2) logging.FileHandler  
和StreamHandler类似，用于向一个文件输出日志信息。不过FileHandler会帮你打开这个文件。它的构造函数是：  
FileHandler(filename[,mode])  
filename是文件名，必须指定一个文件名。  
mode是文件的打开方式。参见Python内置函数open()的用法。默认是’a'，即添加到文件末尾。

3) logging.handlers.RotatingFileHandler  
这个Handler类似于上面的FileHandler，但是它可以管理文件大小。当文件达到一定大小之后，它会自动将当前日志文件改名，然后创建 一个新的同名日志文件继续输出。比如日志文件是chat.log。当chat.log达到指定的大小之后，RotatingFileHandler自动把 文件改名为chat.log.1。不过，如果chat.log.1已经存在，会先把chat.log.1重命名为chat.log.2。。。最后重新创建 chat.log，继续输出日志信息。它的构造函数是：  
RotatingFileHandler( filename[, mode[, maxBytes[, backupCount]]])  
其中filename和mode两个参数和FileHandler一样。  
maxBytes用于指定日志文件的最大文件大小。如果maxBytes为0，意味着日志文件可以无限大，这时上面描述的重命名过程就不会发生。  
backupCount用于指定保留的备份文件的个数。比如，如果指定为2，当上面描述的重命名过程发生时，原有的chat.log.2并不会被更名，而是被删除。

4) logging.handlers.TimedRotatingFileHandler  
这个Handler和RotatingFileHandler类似，不过，它没有通过判断文件大小来决定何时重新创建日志文件，而是间隔一定时间就 自动创建新的日志文件。重命名的过程与RotatingFileHandler类似，不过新的文件不是附加数字，而是当前时间。它的构造函数是：  
TimedRotatingFileHandler( filename [,when [,interval [,backupCount]]])  
其中filename参数和backupCount参数和RotatingFileHandler具有相同的意义。  
interval是时间间隔。  
when参数是一个字符串。表示时间间隔的单位，不区分大小写。它有以下取值：  
S 秒  
M 分  
H 小时  
D 天  
W 每星期（interval==0时代表星期一）  
midnight 每天凌晨

configparser读取配置文件

config = configparser.ConfigParser()

file\_path = 目录 + “xxx.ini”

config.read(file\_path)

config.get(xxx,xxx)

**import** os  
  
*#解析配置文件*config = configparser.ConfigParser()  
print(os.getcwd())  
*#获取配置文件路径*config\_path = os.path.dirname(os.getcwd()) + **"/conf/conf.ini"***#读取配置文件*config.read(config\_path,encoding=**'utf8'**)  
*#通过section，然后获取下面的键*url = config.get(**'address'**,**'url'**)  
*#给section添加键值对*config.set(**'address'**,**'name'**,**"lihe"**)  
*#把新加的键值对写入配置文件中***with** open(config\_path,**'w'**) **as** f:  
 config.write(f)  
  
print(url)

requests

requests常用方法：  
**import** requests  
response = requests.get(**"https://www.baidu.com"**)  
response.request.body *#获取请求体*response.request.headers *#获取请求首部*print(response.text) *#获取响应文本*print(response.content) *#获取响应文本，二进制格式*print(response.status\_code)

Tkinter

思路：

添加一个TK（）对象，然后选择组件，最后是组件布局。

pack、grid 和 place 均用于管理同在一个父组件下的所有组件的布局，其中：

* pack 是按添加顺序排列组件
* grid 是按行/列形式排列组件
* place 则允许程序员指定组件的大小和位置

### 二、python中的知识点总结

1.空对象、值为零的任何数字或者Null对象None的布尔值都是False.下面对象的布尔值是False。

None False 所有的值为零的数 空字符串”” 空列表[] 空元组() 空字典{}

2.连接操作符 +

字符串join()连接字符串

列表中extend() 是把新列表加到了原有列表后面，+操作符是创建了一个新的列表

列表方法：

pop() sort() reverse() insert bremove

更新列表： 通过直接赋值，或者append()

删除列表中的元素：del list[索引] 或者remove(元素值) pop()删除最后一个

删除列表对象：del listName

列表解析 [ i \* 2 for I in range(10) ]

Reverse() sort()不会返回对象，会修改列表自身

Reversed() sorted() 有返回值，会生成新的对象

字典的方法：

类属性与实例属性

类属性需要使用类来修改。 实例共享的

实例属性只对自身起作用

类方法与实例方法