第七讲
标准库
Lecture 7
Standard Library

明玉瑞 Yurui Ming

yrming@gmail.com

声明
Disclaimer

- 本讲义在准备过程中由于时间所限，所用材料来源并未规范标示引用来源。所引材料仅用于教学所用，作者无意侵犯原著者之知识版权，所引材料之知识版权均归原著者所有；若原著者介意之，请联系作者更正及删除。

  The time limit during the preparation of these slides incurs the situation that not all the sources of the used materials (texts or images) are properly referenced or clearly manifested. However, all materials in these slides are solely for teaching and the author is with no intention to infringe the copyright bestowed on the original authors or manufacturers. All credits go to corresponding IP holders. Please address the author for any concern for remedy including deletion.

标准库
Standard Library

- 如C++一样， Python在分发时，也提供了标准库，其包含数百个模块，这些模块提供与操作系统、解释器和 Internet 交互的工具。这些模块都经过测试，可直接用于相关应用程序的开发，大大简化了程序员的开发工作，体现了Python“包含电池”的口号。本讲通过讲解几个最常用的功能模块，体现Python易用性哲学的同时，使读者如何查找和使用标准库中的模块有初步的认识与理解。

  Just like C++, Python also provides the standard library contains hundreds of modules distributed with every copy. These modules provide tools for interacting with the operating system, interpreter, and Internet. All of these module are well tested and ready to be used to jump-start the development of different applications. To leverage the standard library can tremendously reduce the burden on programmers, according supporting Python’s “batteries included” slogan. This lecture presents selected examples demonstrating how to use the most commonly used features of some popular modules, to reflect the user-friendly philosophy of Python and to provide ABC's for check in the standard library and use the included modules.

正则表达式
Regular Expression

- 正则表达式或RegEx是表示搜索模式的字符序列。RegEx可用于检查字符串是否包含指定的搜索模式，并能根据需要，将符合特定模式的字符串内容抽取出来。

  A Regular Expression or RegEx for short, is a sequence of characters that forms a search pattern. RegEx can be used to check if a string contains the specified search pattern. It can even further extract the string content which merits the search pattern.

- 在正则表达式中，除了出现常规字符外，还有一些元字符，特殊序列，及字符集。

  In RegEx, besides the normal characters, there can be metacharacters, special sequences, as well as character set, aka characters enclosed in a pair of brackets.

- re 模块提供了一组函数，允许我们在字符串中搜索匹配项。

  The re module offers a set of functions that allows us to search a string for a match.

正则表达式
Regular Expression

- 元字符 metacharacters

Character	Description	Example
[]	A set of characters	"[a-m]"
\	Signals a special sequence (can also be used to escape special characters)	"\d"
.	Any character (except newline character)	"he..o"
^	Starts with	"^hello"
$	Ends with	"planet$"
*	Zero or more occurrences	"he.*o"
+	One or more occurrences	"he.+o"
?	Zero or one occurrences	"he.?o"
{}	Exactly the specified number of occurrences	"he.{2}o"
|	Either or	"falls|stays"
()	Capture and group	

正则表达式 Regular Expr

- 特殊序列

  special sequences

Character	Description	Example
\A	Returns a match if the specified characters are at the beginning of the string	"\AThe"
\b	Returns a match where the specified characters are at the beginning or at the end of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string")	r"\bain" r"ain\b"
\B	Returns a match where the specified characters are present, but NOT at the beginning (or at the end) of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string")	r"\Bain" r"ain\B"
\d	Returns a match where the string contains digits (numbers from 0-9)	"\d"
\D	Returns a match where the string DOES NOT contain digits	"\D"
\s	Returns a match where the string contains a white space character	"\s"
\S	Returns a match where the string DOES NOT contain a white space character	"\S"
\w	Returns a match where the string contains any word characters (characters from a to Z, digits from 0-9, and the underscore _ character)	"\w"
\W	Returns a match where the string DOES NOT contain any word characters	"\W"
\Z	Returns a match if the specified characters are at the end of the string	"Spain\Z"

正则表达式
Regular Expression

- 字符集

  character set

Set	Description
[arn]	Returns a match where one of the specified characters (`a` , `r` , or `n`) are present
[a-n]	Returns a match for any lower case character, alphabetically between `a` and `n`
[^arn]	Returns a match for any character EXCEPT `a` , `r` , and `n`
[0123]	Returns a match where any of the specified digits (`0` , `1` , `2` , or `3`) are present
[0-9]	Returns a match for any digit between `0` and `9`
[0-5][0-9]	Returns a match for any two-digit numbers from `00` and `59`
[a-zA-Z]	Returns a match for any character alphabetically between `a` and `z` , lower case OR upper case
[+]	In sets, `+` , `*` , `.` , `|` , `()` , `$` , `{}` has no special meaning, so `[+]` means: return a match for any `+` character in the string

正则表达式
Regular Expression

- 字符集

  character set

Set	Description
[arn]	Returns a match where one of the specified characters (`a` , `r` , or `n`) are present
[a-n]	Returns a match for any lower case character, alphabetically between `a` and `n`
[^arn]	Returns a match for any character EXCEPT `a` , `r` , and `n`
[0123]	Returns a match where any of the specified digits (`0` , `1` , `2` , or `3`) are present
[0-9]	Returns a match for any digit between `0` and `9`
[0-5][0-9]	Returns a match for any two-digit numbers from `00` and `59`
[a-zA-Z]	Returns a match for any character alphabetically between `a` and `z` , lower case OR upper case
[+]	In sets, `+` , `*` , `.` , `|` , `()` , `$` , `{}` has no special meaning, so `[+]` means: return a match for any `+` character in the string

正则表达式
Regular Expression

- re 模块提供的函数如下表所示：

  Function provided by the re module are as follows:

Function	Description
findall	Returns a list containing all matches
search	Returns a Match object if there is a match anywhere in the string
split	Returns a list where the string has been split at each match
sub	Replaces one or many matches with a string

正则表达式
Regular Expression

- 案例：假设你运营一个购物网站，根据用户注册所用的邮箱，利用邮箱用户名作为网站ID，并判断注册用户是否享受学生价。

  Project: suppose that you are operating an online shopping site. Based on the email address during the registering process, associate the user ID at your website with the user part of the email address, and validate whether the user qualifies a student discount or not.

- 根据需求，需要将邮箱用户名与邮箱域名均抽取出来。其中邮箱用户名作为网站ID，域名供判断用户身份。根据常识，这两部分由@号分割。因此，该表达式初步具有形式：()@()。

  According to the requirement, both the user part and domain name of the email address need to be extracted. The user part is used as user ID for the website, domain name for discount qualification. According to the structure of the email address, these two parts are separated by @ character. Therefore, the very preliminary form of the regex is ()@()

正则表达式
Regular Expression

- 假设用户名和域名的组成规则类似，均有数字、字母、横线、下划线及点号组成，则表达式可细化为：^([\da-zA-Z\-_\.]+)@([\da-zA-Z\-_\.]+)$。

  We assume the username and domain name share the same rules, that is they are consist of numbers, letters, hyphen, underscore and dot. Then the regex can be refined as ^([\da-zA-Z\-_\.]+)@([\da-zA-Z\-_\.]+)$.

- 由于我们知道学生所在的组织域名，或以edu+点号+国家代码结束，或以ac+点号+国家代码结束（由于美国国家代码默认不用提供，这里我们考虑默认需要提供的情况）。因此，域名部分可以细化为[\da-z]+\.(ac|edu)\.[a-z]{2}$

  Since we know the domain name for the organizations the student in, can either end with edu plus dot plus country code, or with ac plus dot plus country code (because the US country code is omitted by default, here we consider the case where it needs to be provided). Therefore, the domain name part can be refined as [\da-z]+\.(ac|edu)\.[a-z]{2}$.

正则表达式
Regular Expression

- 在有上面表达式的基础上，便可以根据正则表达式对用户输入的email地址进行处理。有时为了提高效率，会对表达式进行编译。同时，为了防止Python对表达式进行不必要的预转义，会用r标示字符串为正则表达式。

  Based on the regular expression, we can process the email address input by the users. Sometimes to improve the efficiency, the regex will by compiled before hand. At the same time, to prevent Python perform pre-escape of character sequence of the regex, an r will prefix the string for indication.

- 如果正则表达式能匹配成功，则返回一个匹配对象，如果失败，则返回None。如果只是检测匹配，则只需看是否返回None即可；如果需要捕获的结果，则可调用匹配对象的groups子方法，取得匹配的内容。

  If the regex can be matched successfully, then a match object is returned, otherwise it will return None. If existence of certain patterns is of interest, just check whether None is returned is sufficient. For capturing, the user needs to call groups() method of the match object to retrieved the captured content.

调试器
pdb

- 程序设计的相当一部分工作会花在调试之上。尽管现在的集成开发环境相当友好，但是pdb作为Python最初始的程序调试模块，功能强大且使用便捷。特别对于偏好在命令行进行程序开发的读者，熟练掌握pdb可以提高工作效率。

  A considerable amount of time spending on programming is actually spent on debugging. Although the current IDE provides a friendly experience for developers, pdb which is shipped as the original debugger module for Python, perfectly balances the powerfulness and convenience. For users prefer to develop in a terminal-based way, there is no doubt some mastery of pdb can increase the working efficiency.

- 通常有两种方式启动pdb，一种是直接在命令行启动，一种在是程序里面。

  There are generally two ways to start pdb. The first is directly launch it at the command line, the other is to embed it in the program.

调试器
pdb

```
$ python3 -m pdb pdb_script.py

> .../pdb_script.py(8)<module>()
-> class MyObj(object):
(Pdb)
```

```
import pdb


class MyObj:

    def __init__(self, num_loops):
        self.count = num_loops

    def go(self):
        for i in range(self.count):
            pdb.set_trace()
            print(i)
        return

if __name__ == '__main__':
    MyObj(5).go()
```

调试器
pdb

`n(ext)`	Step over
`s(tep)`	Step into
`r(eturn)`	Continue until the current function returns
`c(ontinue)`	Continue until the next breakpoint is encountered
`u(p)`	Up one level in the stack trace
`d(own)`	Down one level in the stack trace
`h(elp)` `h(elp) command`	Show help
`q(uit)`	Quit debugger

调试器

pdb

`b(reak)`	Show all breakpoints
`b(reak) line_number`	Set a breakpoint at a specific line
`b(reak) line_number, condition`	Set a breakpoint at a specific line, if condition is met
`b(reak) file:line_number`	Set a breakpoint in a file at a specific line
`b(reak) func`	Set a breakpoint at the first line of a function
`disable number`	Disable breakpoint number
`enable number`	Enable breakpoint number
`clear number`	Remove breakpoint number

调试器

pdb

`p(rint) expr` `pp expr`	Print the value of expr
`w(here)`	Print current position and stack trace
`l(ist)` `l(ist) start, end`	Print 11 lines of code around the current line
`a(rgs)`	Print the arguments of the current function

序列化
Serialization

- 将Python对象转换为一系列字节的过程称为对象序列化。序列化后的表示对象的字节流可以传输或存储，并在需要的时候重新构建以创建具有相同特征的新对象。

  The process that convert the Python object into a series of bytes is called object serialization. The byte stream representing the object can then be transmitted or stored, and later reconstructed to create a new object with the same characteristics upon requirement.

- Python的pickle模块提供了序列化的功能。注意Python的任何对象包括函数，都能被序列化，但是pickle本身不提供安全检查，即被反序列化得到的任意代码如能执行，则能执行，因此在进程间通讯用pickle时要格外小心。

  In Python the module pickle provides the functionality of serialization. Note in Python any object including functions is apt to be serialized, however, pickle offers no security guarantees. In fact, unpickling data can execute arbitrary code. Be careful when using pickle for inter-process communication.

序列化
Serialization

- 可用pickle提供的dump函数，将对象转储为字节流。

  We can use the dump function provided by pickle to encode an object or data structure as a string.

```
C:\Users\yrming\Desktop\备课\Python程序设计>cat lec_7_pickle_string.py
import pickle
import pprint

data = [{'a': 'A', 'b': 2, 'c': 3.0}]
print("original python dictionary object:")
pprint.pprint(data)

data_string = pickle.dumps(data)
print("serialized object as byte string:")
print('{!r}'.format(data_string))
C:\Users\yrming\Desktop\备课\Python程序设计>python lec_7_pickle_string.py
original python dictionary object:
[{'a': 'A', 'b': 2, 'c': 3.0}]
serialized object as byte string:
b'\x80\x03]q\x00}q\x01(X\x01\x00\x00\x00aq\x02X\x01\x00\x00\x00Aq\x03X\x01\x00
\x00\x00bq\x04K\x02X\x01\x00\x00\x00cq\x05G@\x08\x00\x00\x00\x00\x00\x00ua.'
```

序列化
Serialization

- 数据序列化后，可以写入文件、套接字、管道或其他位置。稍后，可以读取相关内容并解开数据以构造具有相同值的新对象。

  After the data is serialized, it can be written to a file, socket, pipe, or other location. Later, the content can be read back and the data unpickled to construct a new object with the same values.

```
C:\Users\yrming\Desktop\备课\Python程序设计>cat lec_7_pickle_unpickle.py
import pickle
import pprint

data1 = [{'a': 'A', 'b': 2, 'c': 3.0}]
print('representation of data1 before pickling: ')
pprint.pprint(data1)

data1_string = pickle.dumps(data1)
data2 = pickle.loads(data1_string)

print('representation of data2 after restored: ')
pprint.pprint(data2)

print('Are the identities of the two objects SAME? :', (data1 is data2))
print('Are the values of the two objects EQUAL?:', (data1 == data2))
C:\Users\yrming\Desktop\备课\Python程序设计>python lec_7_pickle_unpickle.py
representation of data1 before pickling:
[{'a': 'A', 'b': 2, 'c': 3.0}]
representation of data2 after restored:
[{'a': 'A', 'b': 2, 'c': 3.0}]
Are the identities of the two objects SAME? : False
Are the values of the two objects EQUAL?: True
```

序列化
Serialization

- 除了 dumps() 和 loading() 之外，pickle 还提供了处理类文件流的函数，可以方便地将多个对象写入一个流，然后从流中读取它们，而无需事先知道写入了多少对象或对象的大小。

  In addition to dumps() and loads(), pickle provides convenience functions for working with file-like streams. It is possible to write multiple objects to a stream, and then read them from the stream without knowing in advance how many objects are written or how big they are.

```
C:\Users\yrming\Desktop\备课\Python程序设计>cat lec_7_pickle_stream.py
import io, pprint
import pickle

class Obj:
    def __init__(self, msg):
        self._msg = msg

    def __call__(self):
        print(f"{self._msg}")

out_s = io.BytesIO() # Simulate a file.
pickle.dump(Obj('first'), out_s)
out_s.flush()
pickle.dump(Obj('second'), out_s)
out_s.flush()

# Set up a readable stream.
in_s = io.BytesIO(out_s.getvalue())
# Read the data.
while True:
    try:
        o = pickle.load(in_s)
    except EOFError:
        break
    else:
        o()
C:\Users\yrming\Desktop\备课\Python程序设计>python lec_7_pickle_stream.py
first
second
```