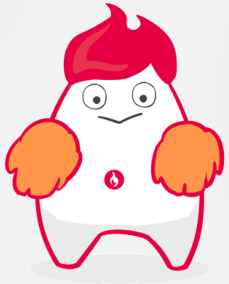
Python中的加密工具



本节课内容

- ◆hashlib包介绍
- ◆hashlib包中的常用方法
- ◆base64包介绍
- ◆base64包的常用方法

hashlib包介绍

◆难破解

◆不可逆

hashlib常用加密方法

函数名	参数	介绍	举例	返回值
md5	byte	Md5算法加密	hashlib.md5(b’ hello’)	Hash对象
sha1	byte	Sha1算法加密	hashlib.sha1(b’ hello’)	Hash对象
sha256	byte	Sha256算法加密	hashlib.sha256(b’ hello’)	Hash对象
sha512	byte	Sha512算法加密	hashlib.sha512(b’ hello’)	Hash对象

hashlib常用加密方法

生成加密字符串：

```
hashobj = hashlib.md5(b’ hello’ )
result = hashobj. hexdigest()
print(result)
>> '6b8bfd346d1dcc17e6940aa59b9c4fa6864064a8'
```

base64包介绍

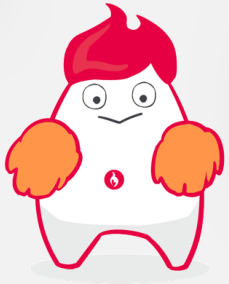
◆通用型

◆可解密

base64包常用方法

函数名	参数	介绍	举例	返回值
encodestring	Byte	进行base64加密	base64.encodestring(b’py’)	Byte
decodingstring	Byte	对base64解密	base64.decodestring(b'eGlhb211\n ’)	Byte
encodebytes	Byte	进行base64加密	base64.encodebytes(b’py’)	Byte
decodingbytes	Byte	对base64解密	base64.decodebytes(b'e	Byte

Python中的日志模块



本节课内容

- 日志的作用
- 日志的等级
- logging模块的使用

日志的作用

- 日记
- 程序行为
- 重要信息记录

日志的等级

- debug
- info
- warnning
- error
- critical

logging模块的使用

- logging.basicConfig

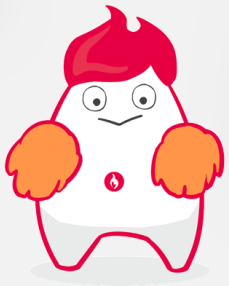
参数名	作用	举例
level	日志输出等级	level=logging.DEBUG
format	日志输出格式	
filename	存储位置	filename= ‘d://back.log’
filemode	输入模式	filemode= “w”

format具体格式

格式符	含义
%(levelname)s	日志级别名称
%(pathname)s	执行程序的路径
%(filename)s	执行程序名
%(lineno)d	日志的当前行号
%(asctime)s	打印日志的时间
%(message)s	日志信息

format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s'

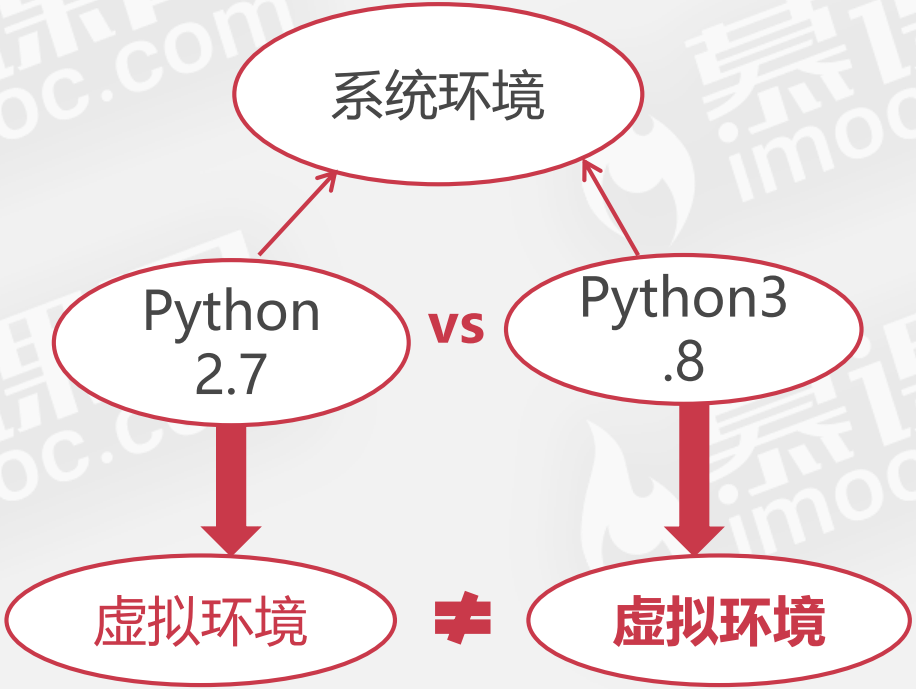
Python中的虚拟环境



本节课内容

- 认识虚拟环境
- Python中的虚拟环境工具

认识虚拟环境



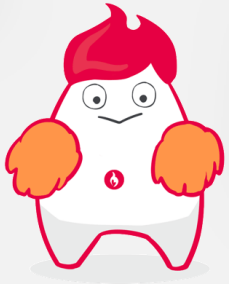
Python中的虚拟环境工具

- Virtualenv
- pyenv

virtualenv

- ◆命令行下使用
- ◆pip install virtualenv
- ◆选择目录
- ◆virtualenv -p python3 penv
- ◆./env/bin/active
- ◆deactive

Python中的内置函数总结



本节课内容

◆总结python的常用内置函数

常用函数1

函数名	参数	介绍	返回值	举例
abs	Number	返回数字绝对值	正数字	abs(-10)
all	List	判断列表内容是否全是true	Bool	all([‘’ , ‘123’])
help	object	打印对象的用法	无	help(list)
enumerate	iterable	迭代时记录索引	无	for index, item in enumerate(list)
input	Str	命令行输如消息	Str	input(‘请输出信息:’)

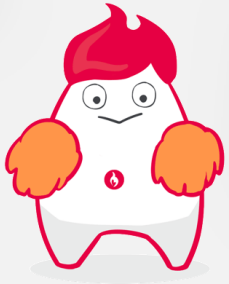
常用函数2

函数名	参数	介绍	返回值	举例
isinstance	Object, type	判断对象是否是某种类型	Bool	isinstance(‘a’ ,str)
type	Object	判断对象的类型	Str	type(10)
vars	instance	返回实例化的字典信息	dict	
dir	object	返回对象中所有可用方法和属性	List	dir(‘asd’)
hasattr	Obj, key	判断对象中是否有某个属性	Bool	hasattr(‘1’ , ‘upper’)

常用函数3

函数名	参数	介绍	返回值	举例
setattr	Obj,key,value	为实例化对象添加属性与值	无	setattr(instance, ‘run’, ‘go)
getattr	obj, key	通过对象获取属性	任何类型	getattr(obj, key)
any	Iterable	判断内容是否有true值	Bool	any([1,0,’ ’]

python的随机模块--random



本节课内容

- random.random
- random.uniform
- random.randint
- random.choice
- random.sample
- random.randrange

random.random

- 随机返回0 ~ 1之间的浮点数

```
In [12]: print(random.random())
0.5699696526701014

In [13]: print(random.random())
0.2642802810441326
```

random.uniform

- 产生一个a、b区间的随机浮点数

```
In [14]: random.uniform(1, 10)
Out[14]: 3.803371151513113

In [15]: random.uniform(1, 10)
Out[15]: 9.520559777639644
```

random.randint

- 产生一个a、b区间的随机整数

```
In [16]: random.randint(1, 10)
Out[16]: 10

In [17]: random.randint(1, 10)
Out[17]: 2
```

random.choice

- 返回对象中的一个随机元素

```
In [18]: random.choice(['a', 'b', 'c'])
Out[18]: 'b'

In [19]: random.choice('abc')
Out[19]: 'c'
```

random.sample

- 随机返回对象中指定的元素

```
In [20]: random.sample(['a', 'b', 'c'], 2)
Out[20]: ['a', 'c']

In [21]: random.sample('abc', 2)
Out[21]: ['b', 'c']
```

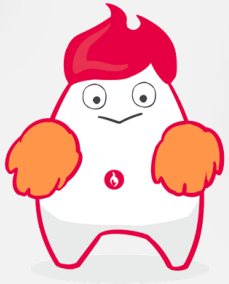
random.randrange

- 获取区间内的一个随机数

```
In [22]: random.randrange(0, 100, 1)
Out[22]: 51

In [23]: random.choice(range(0, 100, 1))
Out[23]: 45
```

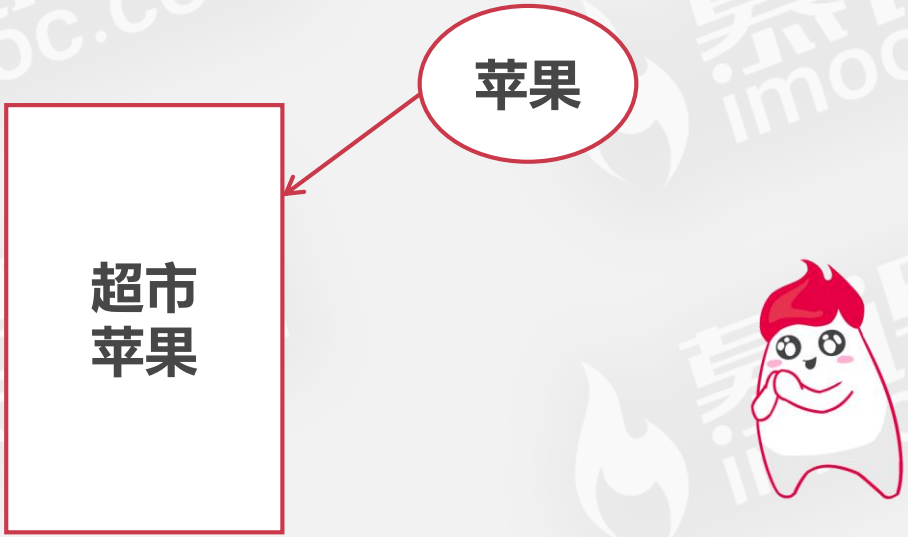
Python中的迭代器



本节课内容

- 什么是迭代器
- 如何生成迭代器
- 迭代器的用法

什么是迭代器



如何生成迭代器-iter

介绍：

生成一个迭代器对象

用法：

iter(iterable)

参数介绍：

iterable： 可迭代的数据类型

举例：

iter([1，2，3])

返回值：

<list_iterator at 0x4f3aee0>

迭代器的使用--next

介绍：

返回迭代器中数据

用法：

next(iterator)

参数介绍：

iterator：迭代器对象

举例：

```
iter_obj = iter([1,2,3])
next(iter_obj)
```

返回值：

1, ,2 ,3

StopIteration

迭代器的使用--next

```
In [21]: iter_obj = iter([1, 2, 3])

In [22]: next(iter_obj)
Out[22]: 1

In [23]: next(iter_obj)
Out[23]: 2

In [24]: next(iter_obj)
Out[24]: 3
```

```
In [25]: next(iter_obj)
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
<ipython-input-25-7e96fd445ff1> in <module>
----> 1 next(iter_obj)

StopIteration:
```

迭代器常用方法之生成迭代器

- for循环生成法—yield

```
In [27]: def test():
    ...:     for i in range(10):
    ...:         yield i
    ...:

In [28]: res = test()

In [29]: next(res)
Out[29]: 0
```

- for循环一行生成迭代器

```
In [30]: res = (i for i in [1, 2, 3])

In [31]: next(res)
Out[31]: 1
```

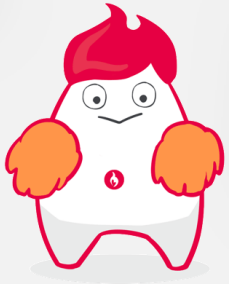
迭代器常用方法之for循环获取

- 不会报错

```
In [32]: res = (i for i in [1, 2, 3])

In [33]: for item in res:
    ...:     print(item)
    ...:
1
2
3
```

Python中的高级函数



本节课内容

- filter
- map
- reduce

filter功能

- 对循环根据过滤条件进行过滤

filter用法

用法：

filter(func, list)

参数介绍：

func: 对 list每个item进行条件过滤的定义

list： 需要过滤的列表

filter举例

举例：

```
res = filter(lambda x: x > 1, [0,1,2])
```

返回值：

```
<filter at 0x4f3af70> -> [1,2]
```

map

- 对循环根据过滤条件是否符合要求的判断

map

用法：

map(func, list)

参数介绍：

func: 对 list每个item进行条件过滤的定义

list： 需要过滤的列表

举例：

res = map(lambda x: x > 1, [0,1,2])

返回值：

<map at 0x4f3af70> -> [False, True, True]

reduce 功能

- 对循环前后两个数据进行累加

reduce 用法

用法：

map(func, list)

参数介绍：

func: 对 数据累加的函数

list： 需要处理的列表

举例：

res = reduce(lambda x,y: x + y, [0,1,2])

返回值：

数字->2

reduce的导入

- from functools import reduce