导入模块时，python会到这些位置搜索：

1、sys.path定义的路径

2、环境变量PYTHONPATH定义的路径

# export PYTHONPATH=/path/to/module

导入模块的方法

1、在一行导入多个模块

>>> import time, os, pickle # 不推荐，推荐每行导入一个模块

2、导入模块中的一部分功能

>>> from random import choice, randint # 常用

3、导入模块时，起别名

>>> import pickle as p # 不太常用

pycharm可以点code -> optimize import可以优化导入模块（没用到的模块将会被

移除，导入的模块按字母顺序排序）

在python中，目录可以当成特殊的模块，叫作包。导入mods目录中的hello.py文件可以写作

# cat mods/hello.py

hi = 'hello world'

>>> import mods.hello

>>> mods.hello.hi

>>> from mods.hello import hi

>>> hi

>>> from mods import hello

>>> hello.hi

注意，如果使用python2版本，目录下必须有一个\_\_init\_\_.py的文件，否则不能成为包

加密：

对称加密：加解密使用相同的算法和密钥，DES/3DES/AES

非对称加密：加解密使用不同的密钥，RSA/DSA

单向加密：加密只能向一个方向进行，不能通过结果反推回原始数据，MD5/SHA

单向加密用途：

1、存储加密密码

2、校验文件的完整性

http://www.cmd5.com/

hashlib: 可用于计算md5值

>>> import hashlib

>>> m = hashlib.md5(b'123456')

>>> m.hexdigest()

'e10adc3949ba59abbe56e057f20f883e'

>>> with open('/etc/passwd', 'rb') as fobj:

... data = fobj.read()

...

>>> m = hashlib.md5(data)

>>> m.hexdigest()

tarfile：实现压缩、解压缩，可以调用gzip、bzip2等

>>> import tarfile

>>> import os

>>> tar = tarfile.open('/tmp/security.tar.gz', 'w:gz')

>>> os.chdir('/etc')

>>> tar.add('security')

>>> tar.close()

>>> os.mkdir('/tmp/demo')

>>> tar = tarfile.open('/tmp/security.tar.gz', 'r:gz')

>>> tar.extractall(path='/tmp/demo')

>>> tar.close()

完全备份：

1、把整个目录打压缩

2、计算每个文件的md5值

{ 'access.conf': '234a8c832',

'abc.txt': '258202392'

}

增量备份：

1、通过现在文件的md5值，判断哪些文件是新增加的，哪些文件有变化

{ 'access.conf': '234a8c832',

'abc.txt': '7392784923',

'xyz.txt': '789234789a3cbd',

}

2、把新增的、有变化的压缩

os.walk的使用：

递归列出所有的文件

>>> list(os.walk('/tmp/demo/security'))

[(路径, 路径中的目录列表, 路径中的文件列表), (路径, 路径中的目录列表, 路径中的文件列表)]

>>> for path, folders, files in os.walk('/tmp/demo/security'):

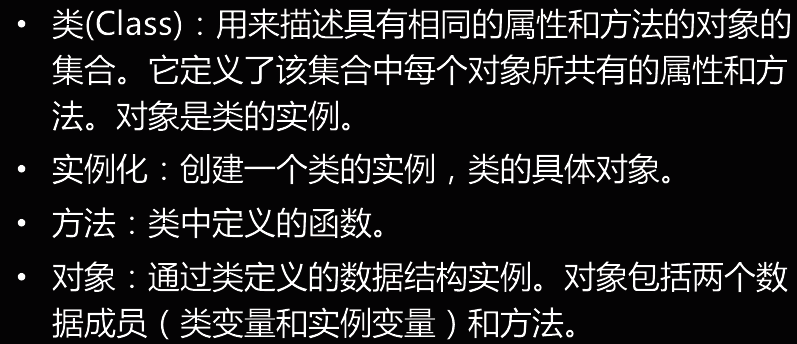
... for file in files:

... os.path.join(path, file)

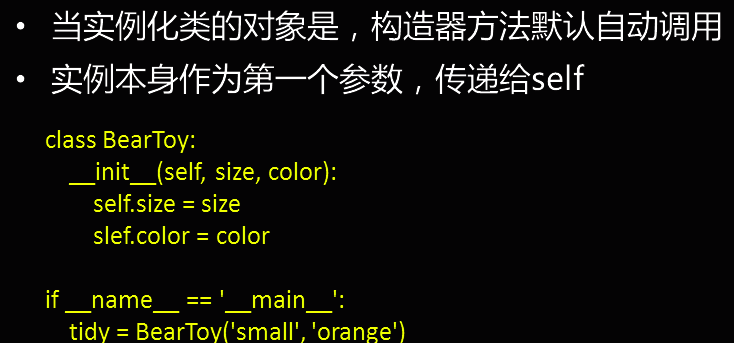
游戏人物：

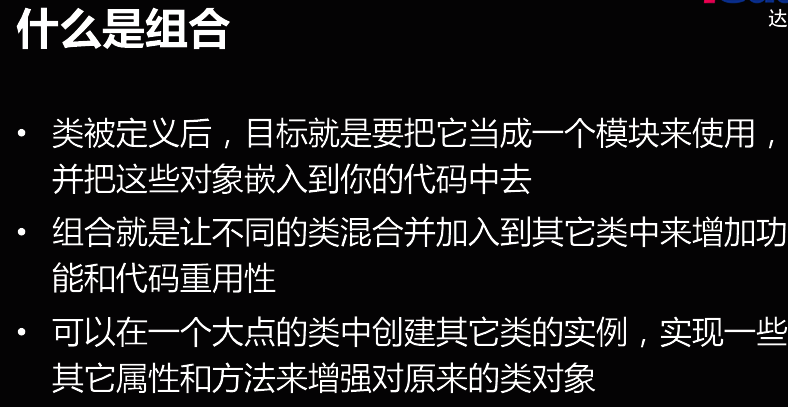
名字、职业、性别、装备、武器、位置

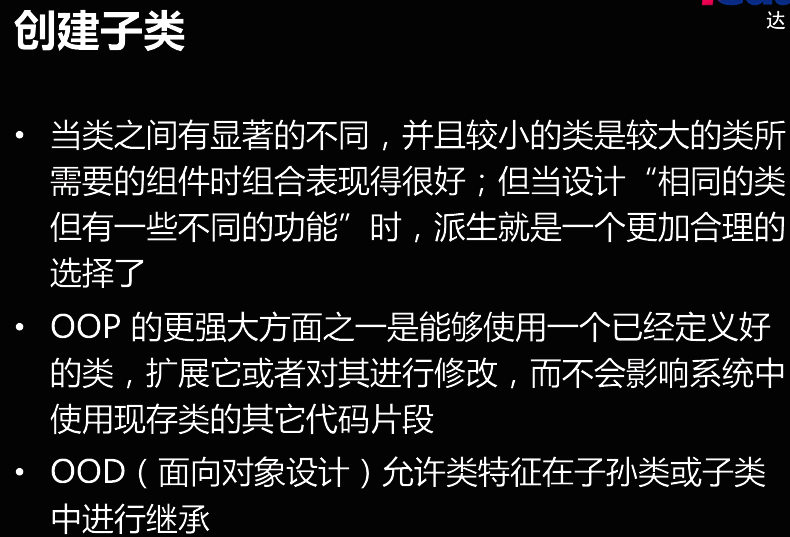
跳、走、攻击











**方法就是类中的函数**

OOP：面向对象的编程，实现属性和行为的融合统一

class ZhanShi:

name = xxx

gender = xxx

def run():

move xxx yyyy

athena = ZhanShi(name=athena, gender=female)

athena.run()

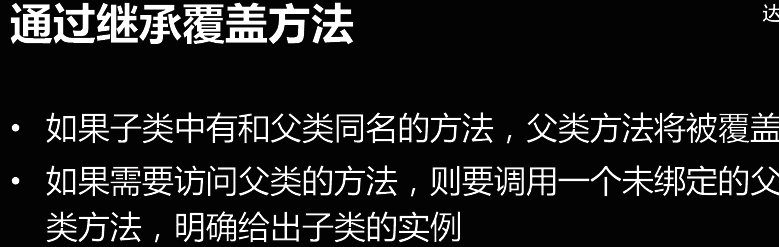
lvbu = ZhanShi(name=lvbj, gender=male)

组合：把一个类的对象作为另一个类的组件，叫作组合

**class** BOOK:  
 **def** \_\_init\_\_(self,name,author):  
 self.name = name  
 self.author = author  
  
 **def** \_\_str\_\_(self):  
 **return '《%s》'** % self.name  
  
 **def** \_\_call\_\_(self):  
 print(**'%s是%s写的'** % (self.name,self.author))  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 m = BOOK(**'core python'**,**'wesley'**)  
 print(m)  
 m()

**这题的重点是str 和 call，他并没有调用函数，所以print返回函数的时候要用str定义。**

**直接应用的时候，有一个call的返回值，我这边call没有返回值，所以有两个数，一个是print的输出，还有一个是没有return输出None**



**如果要调用父类必须要在子类重指定号父类的方法。如果你是要直接使用子类是不需要指定父类的方法。**

**这是调用子类的情况下，在子类中没有指定父类的方法。**

**class** Foo:  
 **def** hello(self):  
 print(**'hello world'**)  
  
**class** Bar:  
 **def** welcome(self):  
 print(**'how are you'**)  
  
**class** FooBar(Foo,Bar):  
 **pass  
  
if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 mc1 = FooBar()  
 mc1.welcome()  
 mc1.hello()

**这是子类变量（函数）有所增加的情况，需要指定一下父类的方法。Self.size也算是一个函数吗，不太懂，但是PPT写的是调用方法。。。。,没有指定的情况可没有新增加的变量。**

**class** Vendor:  
 **def** \_\_init\_\_(self,phone):  
 self.phone = phone  
  
**class** BearToy:  
 **def** \_\_init\_\_(self,size,color):  
 self.size = size  
 self.color = color  
 self.ventor = Vendor(**'400-820-8820'**)  
  
 **def** sing(self):  
 print(**'My color is %s,Lalala'** % self.color )  
  
**class** NewBearToy(BearToy):  
 **def** \_\_init\_\_(self,name,size,color):  
  **BearToy.\_\_init\_\_(self,size,color)** self.color = color  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 bear2 = NewBearToy(**'bear3'**,**'small'**,**'red'**)  
 print(bear2.color)  
 bear2.sing()

**类方法**

**类class中的方法（函数）不能直接应用，必须要，加上类方法@classmethod 才能够引用。**

类方法和组合，

