Tedu.cn
达内教育

Python开发进阶

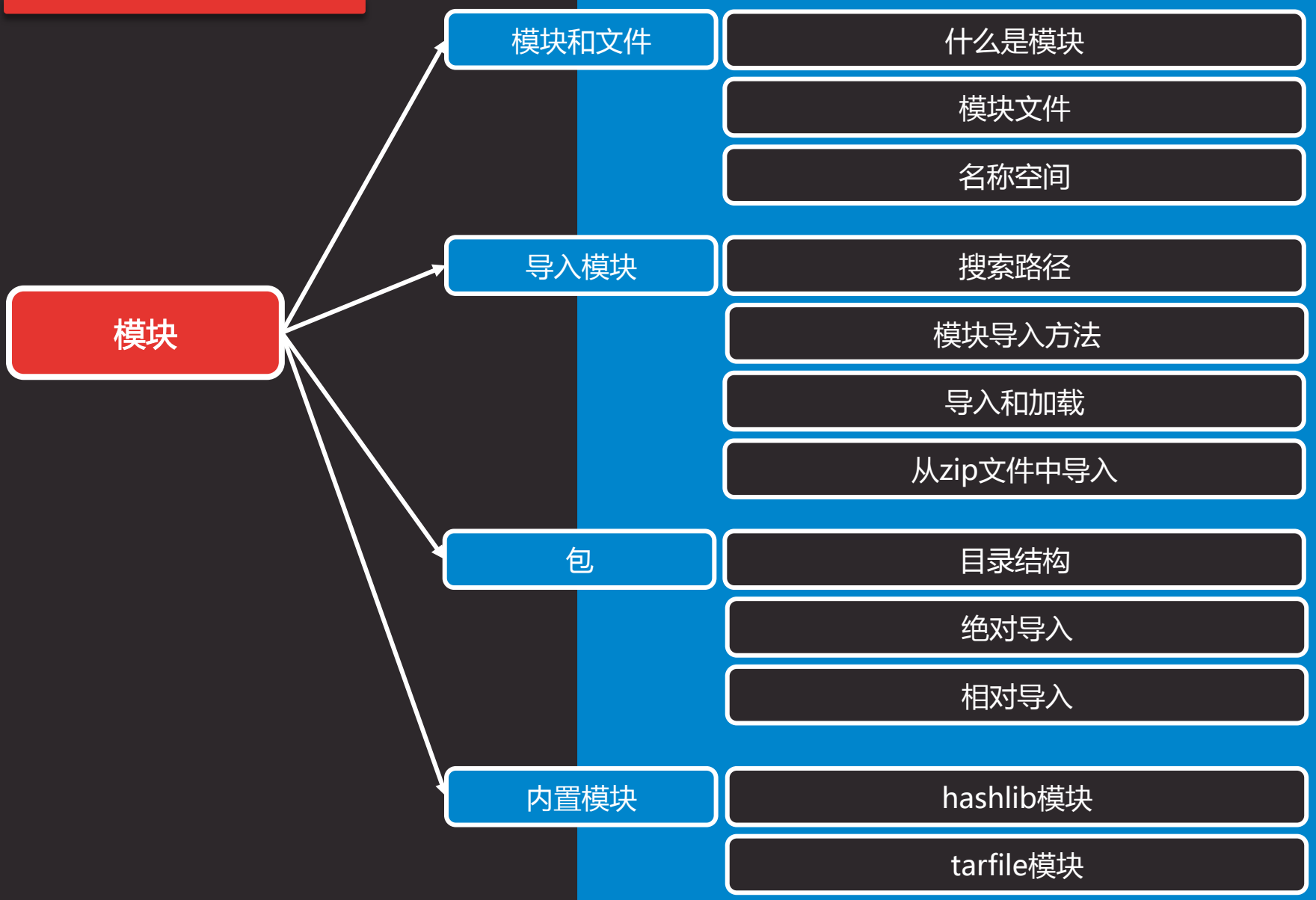
NSD PYTHON2

DAY03

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	模块
	10:30 ~ 11:20	
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	OOP基础
	15:00 ~ 15:50	OOP进阶
	16:10 ~ 17:00	
	17:10 ~ 18:00	总结和答疑

模块


模块
模块和文件
什么是模块
模块文件
名称空间
导入模块
搜索路径
模块导入方法
导入和加载
从zip文件中导入
包
目录结构
绝对导入
相对导入
内置模块
hashlib模块
tarfile模块

Tedu.cn
达内教育

模块和文件

什么是模块

- 模块支持从逻辑上组织python代码
- 当代码量变得相当大的时候, 最好把代码分成一些有组织的代码段
- 代码片段相互间有一定的联系，可能是一个包含数据成员和方法的类，也可能是一组相关但彼此独立的操作函数
- 这些代码段是共享的，所以python允许“调入”一个模块，允许使用其他模块的属性来利用之前的工作成果，实现代码重用

知识讲解

模块文件

- 说模块是按照逻辑来组织python代码的方法，文件是物理层上组织模块的方法
- 一个文件被看作是一个独立模块，一个模块也可以被看作是一个文件
- 模块的文件名就是模块的名字加上扩展名.py

知识讲解

名称空间

知识讲解

- 名称空间就是一个从名称到对象的关系映射集合
- 给定一个模块名之后，只可能有一个模块被导入到python解释器中,所以在不同模块间不会出现名称交叉现象
- 每个模块都定义了它自己的唯一的名称空间

```
>>> import foo
>>> import bar
>>> print(foo.hi)        #调用foo模中的hi变量
hello
>>> print(bar.hi)        #调用bar模块中的hi变量
greet
```

导入模块

搜索路径

- 模块的导入需要一个叫做“路径搜索”的过程
- python在文件系统“预定义区域”中查找要调用的模块
- 搜索路径在sys.path中定义

```
>>> import sys
>>> print(sys.path)
['', '/usr/local/lib/python36.zip', '/usr/local/lib/python3.6', '/usr/local/lib/python3.6/lib-dynload', '/usr/local/lib/python3.6/site-packages']
```

知识讲解

模块导入方法

- 使用import导入模块
- 可以在一行导入多个模块，但是可读性会下降
- 可以只导入模块的某些属性
- 导入模块时，可以为模块取别名

```
>>> import time, os, sys
>>> from random import choice
>>> import pickle as p
```

知识讲解

导入和加载

- 当导入模块时，模块的顶层代码会被执行
- 一个模块不管被导入（import）多少次，只会被加载（load）一次

```
[root@py01 ~]# cat foo.py
hi = 'hello'
print(hi)
[root@py01 ~]# python3
>>> import foo
Hello                    #第一次导入，执行print语句
>>> import foo           #再次导入，print语句不再执行
```

知识讲解

从zip文件中导入

- 在2.3版中，python加入了从ZIP归档文件导入模块的功能
- 如果搜索路径中存在一个包含python模块(.py、.pyc、或.pyo文件)的.zip文件，导入时会把ZIP文件当作目录处理

```
#导入sys模块，在搜索路径中加入相应的zip文件
>>> import sys
>>> sys.path.append('/root/pymodule.zip')
>>> import foo    #导入pymodule.zip压缩文件中的foo模块
```

知识讲解

包

目录结构

- 包是一个有层次的文件目录结构，为平坦的名称空间加入有层次的组织结构
- 允许程序员把有联系的模块组合到一起
- 包目录下必须有一个__init__.py文件

```
phone/
        __init__.py
        common_util.py
         voicedata/
                    __init__.py
                    post.py
```


知识讲解

绝对导入

- 包的使用越来越广泛，很多情况下导入子包会导致和真正的标准库模块发生冲突
- 因此，所有的导入现在都被认为是绝对的，也就是说这些名字必须通过python路径（sys.path或PYTHONPATH）来访问

知识讲解

相对导入

- 绝对导入特性使得程序员失去了import的自由，为此出现了相对导入
- 因为import语句总是绝对导入的，所以相对导入只应用于from-import语句

```
[root@py01 ~]#  ls -R phone/
phone/:
common_util.py  __init__.py  voicedata
phone/voicedata:
__init__.py  post.py

[root@py01 ~]# cat phone/voicedata/post.py
from .. import common_util
```

知识讲解

Tedu.cn
达内教育

内置模块

hashlib模块

- hashlib用来替换md5和sha模块，并使他们的API一致，专门提供hash算法
- 包括md5、sha1、sha224、sha256、sha384、sha512，使用非常简单、方便

```
>>> import hashlib
>>> m = hashlib.md5()
>>> m.update('hello world!')
>>> m.hexdigest()
'fc3ff98e8c6a0d3087d515c0473f8677'
```

知识讲解

tarfile模块

- tarfile模块允许创建、访问tar文件
- 同时支持gzip、bzip2格式

```
[root@py01 home]# ls /home/demo/
install.log  mima
[root@py01 home]# python
>>> import tarfile
>>> tar = tarfile.open('/home/demo.tar.gz', 'w:gz')
>>> tar.add('demo')
>>> tar.close()
```

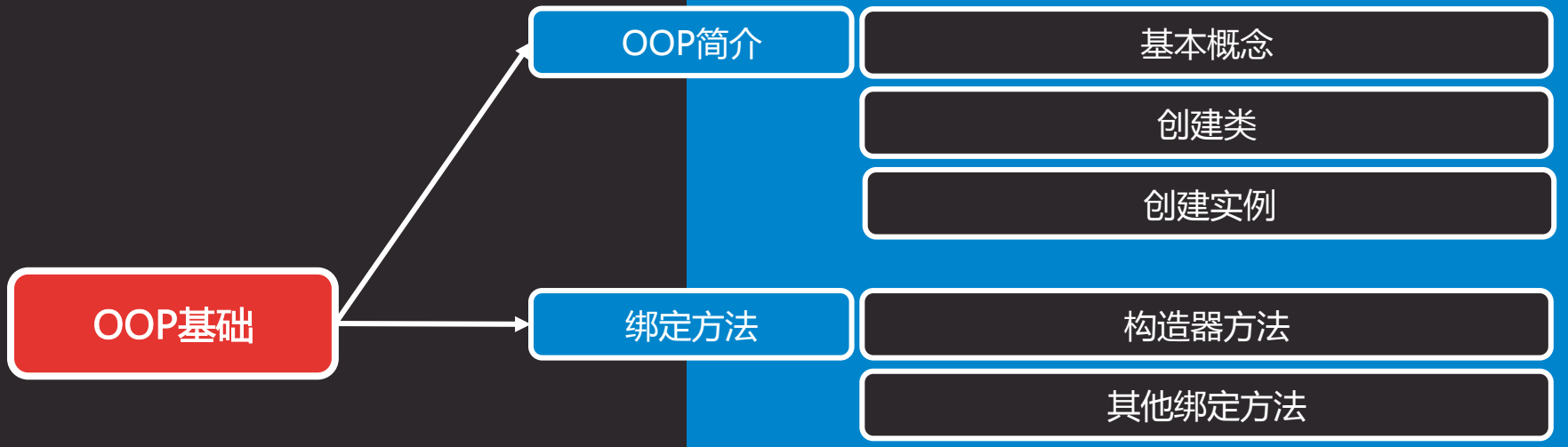
知识讲解

案例1：备份程序

1. 需要支持完全和增量备份
2. 周一执行完全备份
3. 其他时间执行增量备份
4. 备份文件需要打包为tar文件并使用gzip格式压缩

课堂练习

OOP基础



Tedu.cn
达内教育

OOP简介

基本概念

知识讲解

- 类(Class)：用来描述具有相同的属性和方法的对象的集合。它定义了该集合中每个对象所共有的属性和方法。对象是类的实例。
- 实例化：创建一个类的实例，类的具体对象。
- 方法：类中定义的函数。
- 对象：通过类定义的数据结构实例。对象包括两个数据成员（类变量和实例变量）和方法。

创建类

- 使用 class 语句来创建一个新类，class 之后为类的名称并以冒号结尾
- 类名建议使用驼峰形式

```
class BearToy:
    pass
```

知识讲解

创建实例

- 类是蓝图，实例是根据蓝图创建出来的具体对象

```
tidy = BearToy()
```

知识讲解

绑定方法

构造器方法

- 当实例化类的对象是，构造器方法默认自动调用
- 实例本身作为第一个参数，传递给self

知识讲解

```
class BearToy:
    __init__(self, size, color):
        self.size = size
        slef.color = color

if __name__ == '__main__':
    tidy = BearToy('small', 'orange')
```

其他绑定方法

- 类中定义的方法需要绑定在具体的实例，由实例调用
- 实例方法需要明确调用

知识讲解

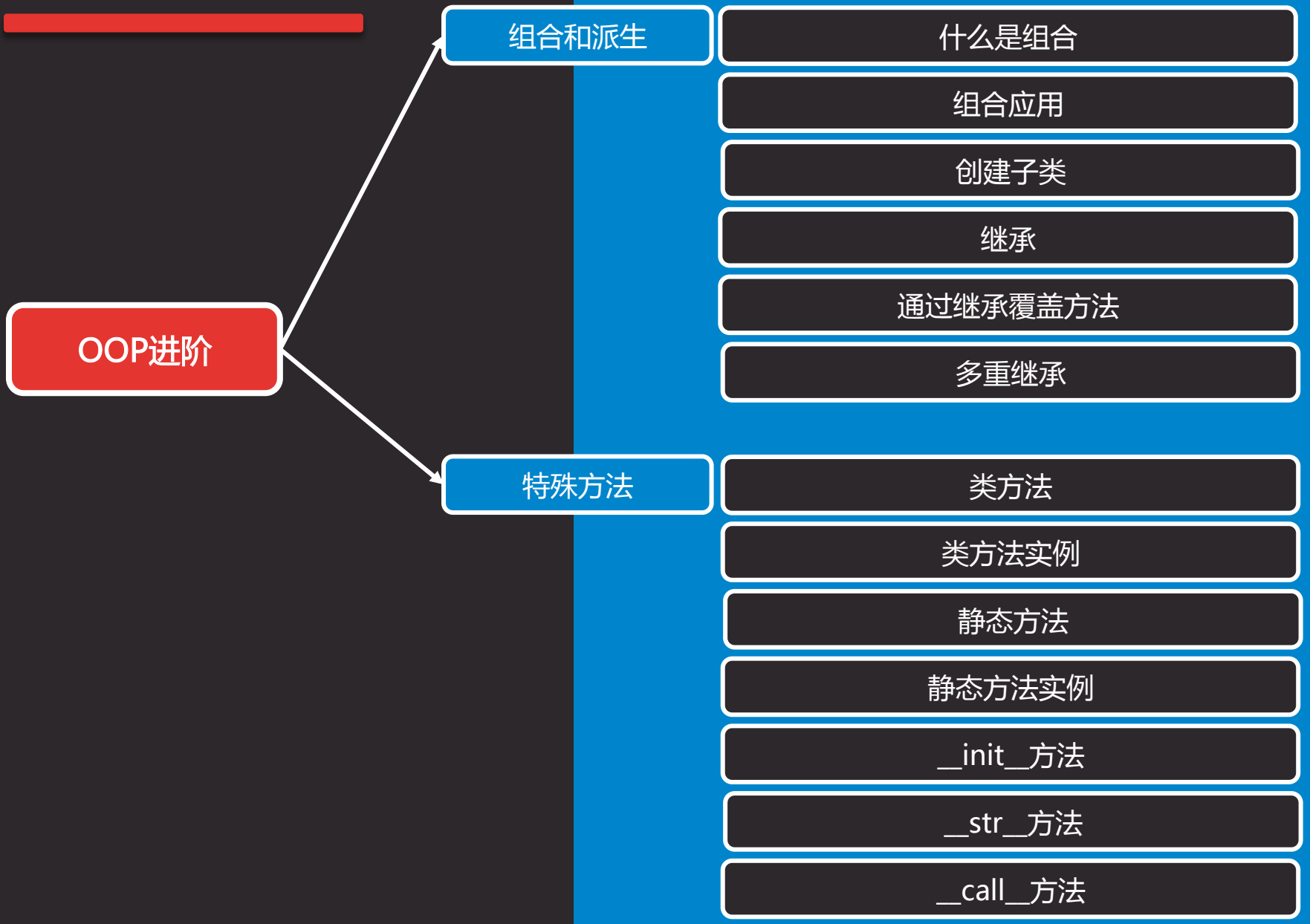
```
class BearToy:
    def __init__(self, size, color):
        self.size = size
        self.color = color

    def speak(self):
        print('hahaha')

if __name__ == '__main__':
    tidy = BearToy('small', 'orange')
    tidy.speak()
```

案例2：编写酒店类

1. 用于计算住宿开销
2. 酒店有会员卡可以打九折
3. 每天早餐15元
4. 根据住宿天数返加总费用

课堂练习

OOP进阶


OOP进阶
组合和派生
什么是组合
组合应用
创建子类
继承
通过继承覆盖方法
多重继承
特殊方法
类方法
类方法实例
静态方法
静态方法实例
__init__方法
__str__方法
__call__方法

Tedu.cn
达内教育

组合和派生

什么是组合

- 类被定义后，目标就是要把它当成一个模块来使用，并把这些对象嵌入到你的代码中去
- 组合就是让不同的类混合并加入到其它类中来增加功能和代码重用性
- 可以在一个大点的类中创建其它类的实例，实现一些其它属性和方法来增强对原来的类对象

知识讲解

组合应用

- 两个类明显不同
- 一个类是另一个类的组件

知识讲解

```
class Manufacture:
    def __init__(self, phone, email):
        self.phone = phone
        self.email = email

class BearToy:
    def __init__(self, size, color, phone, email):
        self.size = size
        self.color = color
        self.vendor = Manufacture(phone, email)
```

创建子类

知识讲解

- 当类之间有显著的不同，并且较小的类是较大的类所需要的组件时组合表现得很好；但当设计“相同的类但有一些不同的功能”时，派生就是一个更加合理的选择了
- OOP 的更强大方面之一是能够使用一个已经定义好的类，扩展它或者对其进行修改，而不会影响系统中使用现存类的其它代码片段
- OOD（面向对象设计）允许类特征在子孙类或子类中进行继承

创建子类（续1）

- 创建子类只需要在圆括号中写明从哪个父类继承即可

```
class BearToy:
    def __init__(self, size, color):
        self.size = size
        self.color = color
    … …

class NewBearToy:
    pass
```

知识讲解

继承

- 继承描述了基类的属性如何“遗传”给派生类
- 子类可以继承它的基类的任何属性，不管是数据属性还是方法

```
class BearToy:
    def __init__(self, size, color):
        self.size = size
        self.color = color
    ... ...

class NewBearToy:
    pass

if __name__ == '__main__':
    tidy = NewBearToy('small', 'orange')
    tidy.speak()
```

知识讲解

通过继承覆盖方法

- 如果子类中有和父类同名的方法，父类方法将被覆盖
- 如果需要访问父类的方法，则要调用一个未绑定的父类方法，明确给出子类的实例

```
class BearToy:
    def __init__(self, size, color, phone, email):
        self.size = size
        self.color = color
        self.vendor = Manufacture(phone, email)
    … …

class NewBearToy(BearToy):
    def __init__(self, size, color, phone, email, date):
        super(NewBearToy, self).__init__(size, color, phone, email)
        self.date = date
```

知识讲解

多重继承

- python允许多重继承，即一个类可以是多个父类的子类，子类可以拥有所有父类的属性

```
>>> class A:
...     def foo(self):
...         print('foo method’)
>>> class B:
...     def bar(self):
...         print('bar method’)
>>> class C(A, B):
...     pass
>>> c = C()
>>> c.foo()
foo method
>>> c.bar()
bar method
```

知识讲解

Tedu.cn
达内教育

特殊方法

类方法

- 使用classmethod装饰器定义
- 第一个参数cls表示类本身

```
class Date:
    def __init__(self, year, month, date):
        self.year = year
        self.month = month
        self.date = date

if __name__ == '__main__':
    d1 = Date(2018, 1, 1)
```

知识讲解

类方法实例

知识讲解

```
class Date:
    def __init__(self, year, month, date):
        self.year = year
        self.month = month
        self.date = date

    @classmethod
    def create_date(cls, string_date):
        year, month, date = map(int, string_date.split('-'))
        instance = cls(year, month, date)
        return instance
if __name__ == '__main__':
    d2 = Date.create_date('2018-05-04’)
```

静态方法

- 基本上就是一个函数
- 在语法上就像一个方法
- 没有访问对象和它的字段或方法
- 使用staticmethod装饰器定义

知识讲解

静态方法实例

知识讲解

```
class Date:
    def __init__(self, year, month, date):
        self.year = year
        self.month = month
        self.date = date

    @staticmethod
    def is_date_valid(string_date):
        year, month, date = map(int, string_date.split('-'))
        return 1 <= date <= 31 and 1 <= month <= 12 and year <
3999

if __name__ == '__main__':
    print(Date.is_date_valid('2018-02-04'))
    print(Date.is_date_valid('2018-22-04'))
```

__init__方法

- 实例化类实例时默认会调用的方法

```
class BearToy:
    __init__(self, size, color):
        self.size = size
        slef.color = color

if __name__ == '__main__':
    tidy = BearToy('small', 'orange')
```

知识讲解

__str__方法

- 打印/显示实例时调用方法
- 返回字符串

```
class BearToy:
    __init__(self, size, color):
        self.size = size
        slef.color = color

    def __str__(self):
        return '<Bear: %s %s>' % (self.size, self.color)

if __name__ == '__main__':
    tidy = BearToy('small', 'orange')
    print(tidy)
```

知识讲解

__call__方法

- 用于创建可调用的实例

```
class BearToy:
    __init__(self, size, color):
        self.size = size
        slef.color = color

    def __call__(self):
        print('I am a %s bear' % self.size)

if __name__ == '__main__':
    tidy = BearToy('small', 'orange')
    print(tidy)
```

知识讲解

案例3：出版商程序

1. 为出版商编写一个Book类
2. Book类有书名、作者、页数等属性
3. 打印实例时，输出书名
4. 调用实例时，显示该书由哪个作者编写

课堂练习

Tedu.cn
达内教育

总结和答疑