Python语言的介绍

为什么要学Python

- Python十分简洁，上手轻松
- Python是最好的网站开发语言之一
- Python是开发效率极高的语言

Python的版本

- Python主要版本：Python 2 与Python 3
- Python 3 目前最新版本为3.8
- Python 2 已经停止更新

Python的特点

- 简单优雅的语言，容易理解
- Python是开源的，全世界程序员都在为之添砖加瓦
- Python适用于短平快的日常任务

Python的优缺点

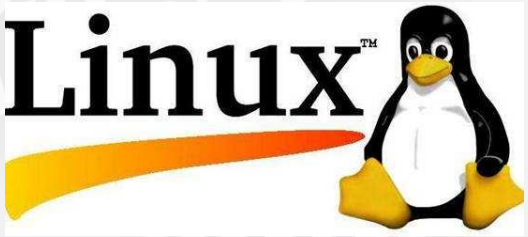
优点	缺点
简单、易学	运行速度慢
免费、开源	国内刚刚起步
丰富的库、可扩展性	中文资料匮乏

环境搭建

本节课内容

- 常见的操作系统介绍
- 本课使用的系统环境
- Python3.8的下载与安装
- 开发工具IDE与Pycharm
- Pycharm的下载与安装

常见操作系统



本课程使用操作系统

下载Python

- 官网 https://www.python.org/
- 下载Python3.8.1

安装Python

IDE与PyCharm

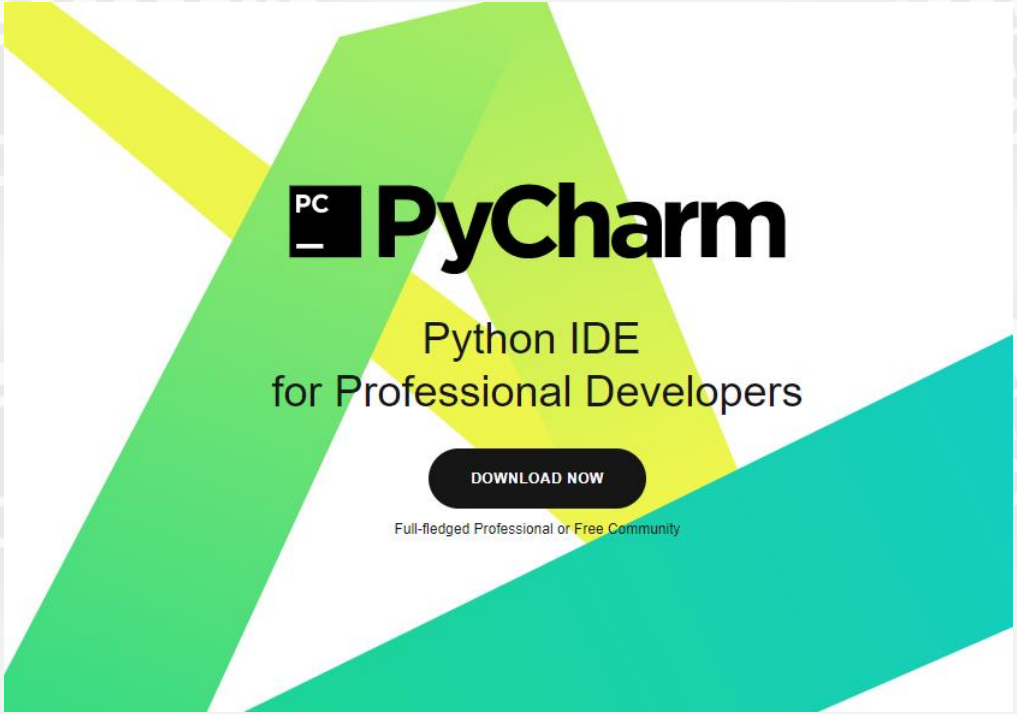
- 集成开发环境（Integrated Development Environment）
- PyCharm是Python最好的IDE之一
- Pycharm是JetBrains公司的产品，具备跨平台特性

下载PyCharm

- 官网：https://www.jetbrains.com/pycharm/download

- 下载PyCharm（免费版）

安装PyCharm


PC
PyCharm
Python IDE
for Professional Developers
DOWNLOAD NOW
Full-fledged Professional or Free Community

初次使用PyCharm注意事项

- 版本不同导致设置修改后不能立即生效
- 先关闭再启动，设置生效

编写一个完整的Python脚本

本节课内容

- Python脚本的格式
- Python脚本内部的结构

Python脚本的格式

脚本名　脚本格式

hello.py

Python脚本的结构

```
# coding:utf-8

import os

print(os.getcwd())
print('hello world')
print(3.14)
```

脚本头

脚本体

引用部分

业务部分

Python脚本的执行

Python解释器

被解释器执行的python脚本

python hello.py

```
(env3.8) → python脚本的结构 python hello.py
/Users/zhangdewei/muke/python_base/lessions/lession02/ python脚本的结构
hello world!
```

Python的头部注释

本节课内容

- 什么是Python头部注释
- Python头部注释的作用
- 头注释的结构
- 常用的头部注释

什么是头注释

- 写在Python脚本第一行的用 # 号表示的信息就是头注释

```
1 #  coding: utf-8
```

头注释的作用

- 头注释并不是为代码而服务，更多是被系统所调用

头部注释的结构

注释符号　　注释的内容

```
#  coding:utf-8
```

常见头注释介绍

- 国内很常用

  `# coding:utf-8` 定义 coding 则告诉系统脚本是何编码格式

- 目前很少使用

  `#!/usr/bin/env` 定义#！,会去找指定路径下的python解释器

Python的导入位置

本节课内容

- 导入的是什么
- 为什么需要导入
- 为什么要放在头注释下边
- 初识导入语法

导入的是什么

- 导入是将Python的一些功能函数放到当前的脚本中使用
- 不导入的功能无法直接在当前脚本使用（除了python自带的内置函数）

为什么要导入

b脚本想借用a脚本里的功能用一用，就需要把a脚本中的功能导入到b脚本中

为什么是在头注释下边

初识导入语法

内置导入函数

```
import os
```

被导入的模块

Python程序的执行顺序

本节课内容

- 如何执行
- 什么是python的内置函数
- 第一个python的内置函数 print的使用

如何执行

- 自上而下，逐行执行

```
# coding:utf-8

print('这是第一行程序')
print('这是第二行程序')
print('...')
print('这是最后一行了!')
```

什么是内置函数

第一个内置函数--print

- print 在百度翻译上的解释为： 打印
- 将你希望的信息显示（打印）在控制台（cmd）上的函数
- print 可以将Python代码中几乎所有程序都可以打印出来

print用法（函数的初探）

函数名（执行函数）　参数体，函数执行的必要数据，有了数据才能执行函数

```
print(object, end=‘’)
```

打印的信息，多信息用逗号隔开　　结尾符，平时不用，默认是换行

函数的相逢

- 第一次接触函数
- 会不断接触内置函数
- 函数的章节中，会学习函数的创建与执行等，敬请期待

Python代码中的注释

本节课内容

- 什么是注释
- 为什么要用注释
- 注释的三种用法

什么是注释

- 在代码中，不会被Python直接运行的语句

为什么要用注释

```
print(‘哈哈你好！’) # 看到这行代码的人，注意了，这是一个测试代码
```

注释的三种用法

- #

```
# 看到这行代码的人，注意了，这是一个测试代码
```

- 三引号

```
"""
这是三引号注释的第一种
双引号形式，我们可以随意换行
"""
```

- 单引号

```
'''
这是三引号注释的第二种
单引号形式，功能与双引
号完全一致
'''
```

Python脚本执行的入口

本节课内容

- 什么是脚本的入口
- 脚本入口的写法
- 缩进
- 是否一定需要脚本入口

什么是代码执行的入口

- 赛车要想进入赛道一定有一个入口

  程序的执行也需要这么一个入口

- 一般我们称代码执行的入口叫做主函数 main

代码脚本的写法

__name__ == '__main__'

双下划线 双下划线 双下划线 双下划线

假如你叫小明.py, 在朋友眼中，你是小明(__name__ == '小明')；
在你自己眼里，你是你自己（__name__ == '__main__'）

缩进

```
# coding:utf-8

import os


if __name__ == '__main__':
    print(os.getcwd())
```

缩进
1个tab

```
print('我回到了1级代码块')
```

是否一定需要脚本入口

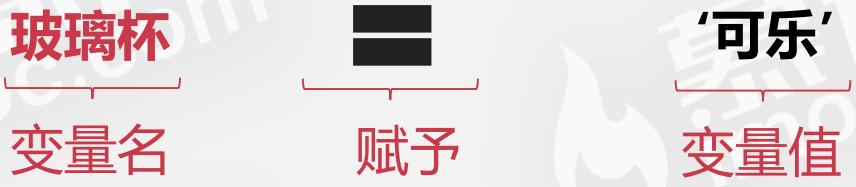
并不是

变量

本节课内容

- 什么是变量
- 变量住哪里
- 变量名的规则

什么是变量

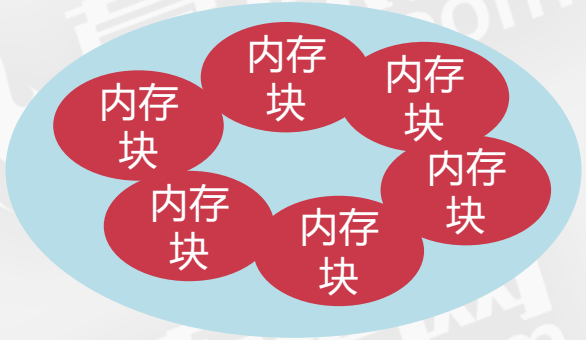


- 赋值语句
- 通过给变量名赋值可以进行值的改变
- 变量名可以不变，而变量值可以随时改变
- 一个拥有变量值的变量名---变量

变量存在哪里

◆变量存在于我们电脑的内存里

◆每个变量被定义后存入一个内存块



变量名的规则

- 组成：必须是数字，字母，下划线组成
- 长度：任意长度，但不建议太长，20字符以内
- 要求：开头必须是字母
- 注意：区分大小写、有意义

变量名举例

username

create_table

_create_table

OpenLesson ?

01work X

get&create X

abc0101 X

你好 X

Python中的关键字

本节课内容

- 什么是关键字
- 关键字与变量名的区别
- Python中的常见关键字

什么是关键字

- Python内部自带的用于处理业务逻辑的特殊单词

- 变量名绝不能使用关键字来命名

关键字与变量名的区别

- 变量名用于给变量赋值使用，而关键字用于业务逻辑处理

关键字的分类

- 强关键字
- 弱关键字

Python中的常见关键字

关键字	含义
True	布尔类型，表示真
False	布尔类型，表示否
def	函数定义
if	逻辑中的如果
elif	逻辑中的或者如果
else	逻辑中的否则
try	异常语句的开启

关键字	含义
is	判断变量是否是某个类的实例
not	逻辑运算，非的操作
or	逻辑运算，或的操作
pass	无意义，站位字符
raise	主动抛出异常
in	判断变量是否在序列中
while	While循环语句
with	简化python语句
yield	从循环或函数依次返回数据
import	导入语句，可与from共用