

# Python开发入门

**NSD PYTHON1**

**DAY01**

# 内容

上午	09:00 ~ 09:30	Python概述
	09:30 ~ 10:20	环境准备
	10:30 ~ 11:20	使用git
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	Python起步
	15:00 ~ 15:50	
	16:10 ~ 17:00	
	17:10 ~ 18:00	总结和答疑



## python概述

python概述

python简介

Python起源

Python版本

Python的特点

# Python简介

## python起源

- 贵铎·范·罗萨姆 ( Guido van Rossum ) 于1989年底始创了python
- 1991年初，python发布了第一个公开发行版
- 为了更好的完成荷兰的CWI ( 国家数学和计算机科学研究院 ) 的一个研究项目而创建



# Python版本

知识讲解

- Python2.x
  - 目前所有系统默认安装的版本
- Python3.x
  - 2009年2月13日发布
  - 在语法和功能上有较大调整
  - Python的发展趋势



# Python的特点

知识讲解

- 高级：有高级的数据结构，缩短开发与代码量
- 面向对象：为数据和逻辑相分离的结构化和过程化编程添加了新的活力
- 可升级：提供了基本的开发模块，可以在它上面开发软件，实现代码的重用
- 可扩展：通过将其分离为多个文件或模块加以组织管理



# Python的特点 ( 续1 )

知识讲解

- 可移植性：python是用C写的，又由于C的可移植性，使得python可以运行在任何带有ANSI C编译器的平台上
- 易学：python关键字少、结构简单、语法清晰
- 易读：没有其他语言通常用来访问变量、定义代码块和进行模式匹配的命令式符号
- 内存管理器：内存管理是由python解释器负责的



## 环境准备

环境准备

安装与配置

获取python3源码

安装python3

设置环境变量

设置pycharm

# 安装与配置

## 获取python3源码

- 官方站点
  - <http://www.python.org>
- 选择正确的系统
- 选择正确的版本



# 安装python3

知识讲解

- 安装依赖包

```
# yum install -y gcc gcc-c++ zlib-devel openssl-devel readline-devel  
libffi-devel sqlite-devel tcl-devel tk-devel
```

- 安装python3

```
# tar xzf Python-3.6.4.tar.gz  
# cd Python-3.6.4  
# ./configure --prefix=/usr/local  
# make && make install
```



# 设置pycharm

知识讲解

- Pycharm是由JetBrains打造的一款Python IDE
- 支持的功能有：
  - 调试、语法高亮
  - Project管理、代码跳转
  - 智能提示、自动完成
  - 单元测试、版本控制
- 下载地址：  
<https://www.jetbrains.com/pycharm/download>
- 分为收费的专业版和免费的社区版



# 案例1：准备python开发环境

课堂练习

1. 下载最新版本的python3
2. 下载pycharm社区版
3. 安装python3，使其支持Tab键补全
4. 配置pycharm，使其符合自己的习惯



## 使用git

### 使用git

#### 本地操作

#### Git简介

#### 安装及配置

#### Git工作流程

#### 工作区、暂存区和版本库

#### 创建仓库

#### 添加文件到暂存区

#### 确认至仓库

#### 删除跟踪文件

#### 使用远程服务器

#### 搭建本地gitlab服务器

#### 初始化gitlab服务器

#### 添加gitlab项目

#### 创建群组

#### 创建项目

#### 创建用户

#### 用户管理



# 本地操作

## Git简介

知识讲解

- Git是一个开源的分布式版本控制系统，用于敏捷高效地处理任何或小或大的项目。
- Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。
- Git 与常用的版本控制工具 CVS, Subversion 等不同，它采用了分布式版本库的方式，不必服务器端软件支持。



# 安装及配置

知识讲解

- Git安装后需配置用户相关信息

```
[root@localhost ~]# yum install -y git
[root@localhost ~]# git config --global user.name "Mr.Zhang"
[root@localhost ~]# git config --global user.email "zhangzg@tedu.cn"
[root@localhost ~]# git config --global core.editor vim
[root@localhost ~]# git config --list
[root@localhost ~]# cat ~/.gitconfig
```



## 案例2：配置git

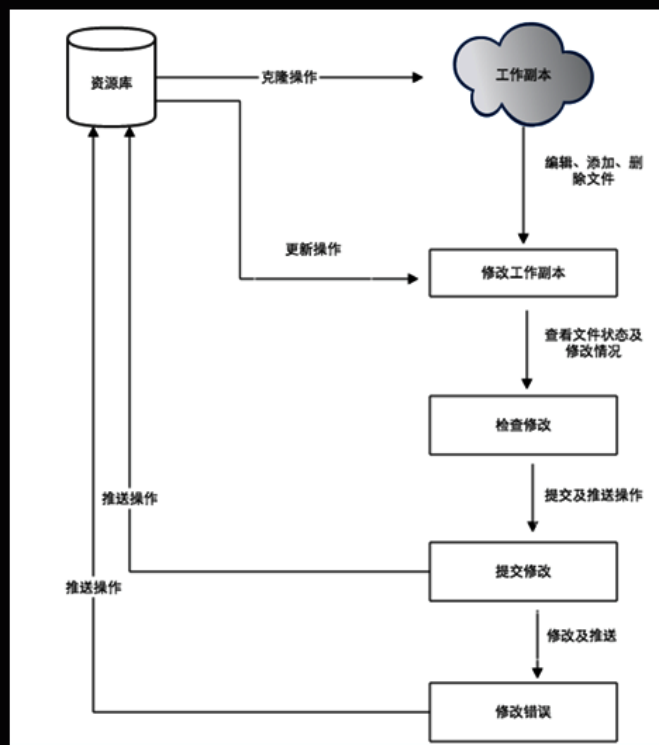
课堂练习

1. 安装git版本控制软件
2. 设置用户信息，如用户名、email等
3. 设置默认编辑器为vim
4. 查看用户配置



# Git工作流程

知识讲解



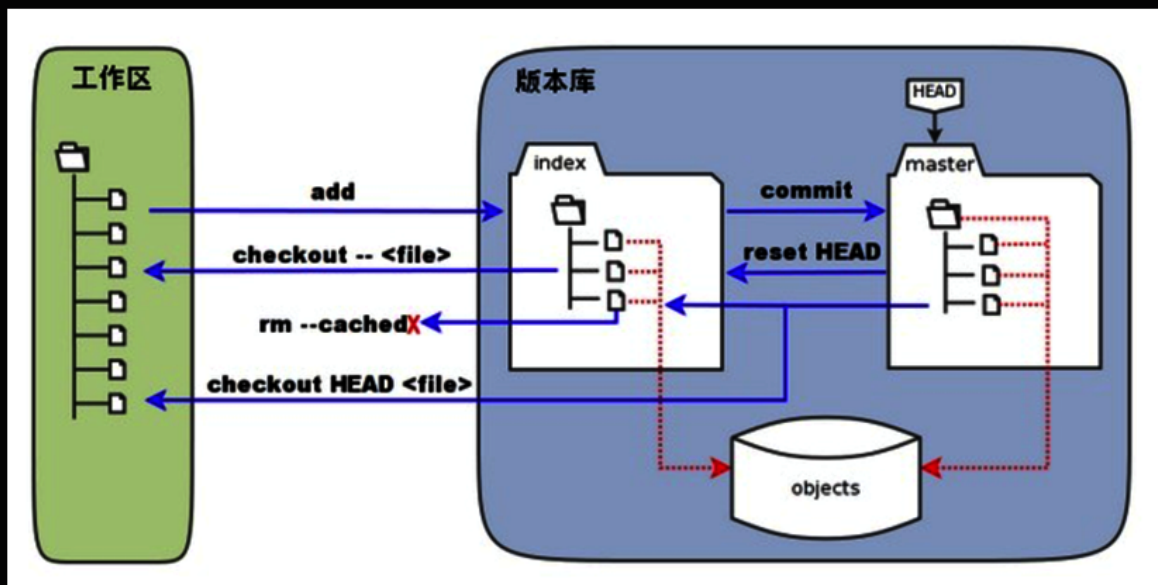
## 工作区、暂存区和版本库

知识讲解

- **工作区**：就是你在电脑里能看到的目录
- **暂存区**：英文叫stage, 或index。一般存放在 ".git目录" 下的index文件 ( .git/index ) 中，所以我们把暂存区有时也叫作索引 ( index )
- **版本库**：工作区有一个隐藏目录.git，这个不算工作区，而是Git的版本库



## 工作区、暂存区和版本库（续1）



知识讲解



## 创建仓库

- Git 使用 `git init` 命令来初始化一个 Git 仓库，Git 的很多命令都需要在 Git 的仓库中运行，所以 `git init` 是使用 Git 的第一个命令。

```
[root@localhost ~]# mkdir devops
[root@localhost ~]# cd devops/
[root@localhost devops]# git init
或
[root@localhost ~]# git init devops
```

知识讲解



## 添加文件到暂存区

知识讲解

- 添加指定文件

```
[root@localhost devops]# echo 'print("hello world!")' > hello.py  
[root@localhost devops]# git add hello.py  
[root@localhost devops]# git status
```

- 添加所有文件

```
[root@localhost devops]# cp hello.py welcome.py  
[root@localhost devops]# git add .  
[root@localhost devops]# git status -s
```



## 确认至仓库

知识讲解

- 提交之前务必先设置用户信息

```
[root@localhost devops]# git commit -m "初始化仓库"  
[root@localhost devops]# git status
```

- 添加追踪文件并提交到版本库

```
[root@localhost devops]# echo 'print("done.")' >> hello.py  
[root@localhost devops]# git commit -am "向hello.py添加新行"
```



## 删除跟踪文件

知识讲解

- 要从 Git 中移除某个文件，就必须要从已跟踪文件清单中移除，然后提交

```
[root@localhost devops]# git ls-files    //查看版本库中文件  
[root@localhost devops]# git rm welcome.py  
[root@localhost devops]# git commit -m '删除welcome.py'
```



## 案例3：git本地操作

课堂练习

1. 创建devops目录
2. 为devops创建git仓库
3. 新建文件hello.py，并将文件初始化到仓库中
4. 修改hello.py并将其更新到仓库
5. 从他库中删除hello.py



# 使用远程服务器

## 搭建本地gitlab服务器

- 导入中文版gitlab镜像

```
[root@localhost devops]# docker load < /path/to/gitlab_zh.tar
```

- 将物理主机ssh端口改为2022后，启动容器

```
[root@localhost devops]# docker run -d -h gitlab --name gitlab -p 443:443 -p 80:80 -p 22:22 --restart always -v /srv/gitlab/config:/etc/gitlab -v /srv/gitlab/logs:/var/log/gitlab -v /srv/gitlab/data gitlab_zh:latest
```

# 初始化gitlab服务器

- 密码需大于8位

知识讲解



# 初始化gitlab服务器（续1）

- 默认用户名为root

知识讲解





# 添加gitlab项目

知识讲解

- 创建群组group
  - 使用群组管理项目和人员是非常好的方式
- 创建项目project
  - 存储代码的地方，里面还包含问题列表、维基文档以及其他一些Gitlab功能
- 创建成员member
  - 添加你的团队成员或其他人员到Gitlab



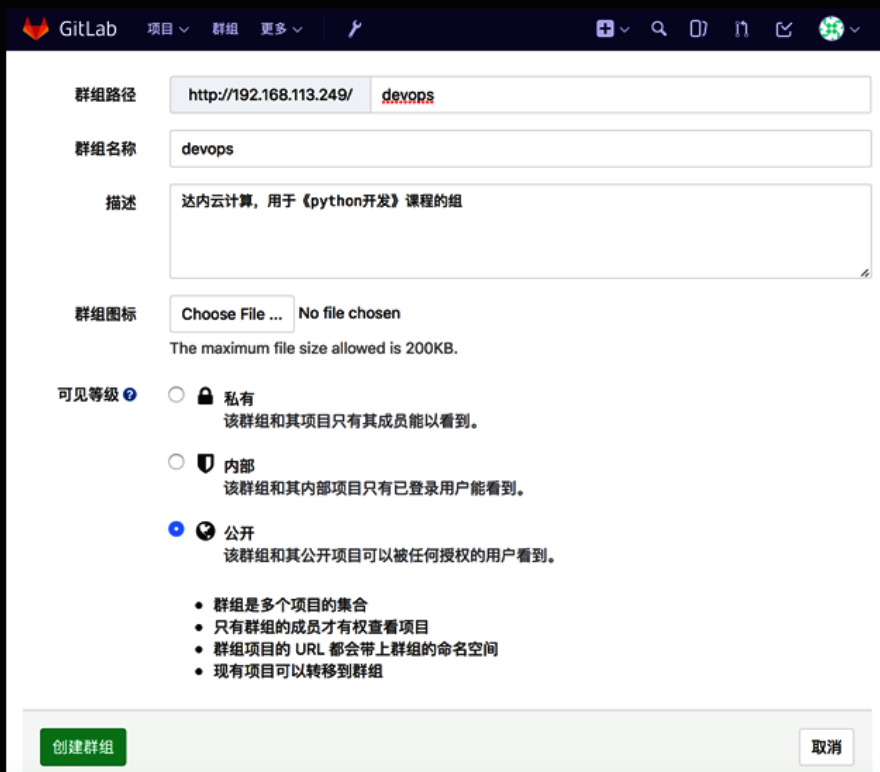
# 创建群组

知识讲解



# 创建群组（续1）

知识讲解



GitLab 项目 群组 更多

群组路径

群组名称

描述

群组图标  No file chosen  
The maximum file size allowed is 200KB.

可见等级 ☐ 私有  
该群组和其项目只有其成员能看到。

☐ 内部  
该群组和其内部项目只有已登录用户能看到。

☒ 公开  
该群组和其公开项目可以被任何授权的用户看到。

- 群组是多个项目的集合
- 只有群组的成员才有权查看项目
- 群组项目的 URL 都会带上群组的命名空间
- 现有项目可以转移到群组

# 创建项目

知识讲解



GitLab 项目 群组 更多

devops > 详细信息

群组 'devops' 创建成功。

devops  
达内云计算，用于《python开发》课程的组

按名称过滤...

群组是几个项目的集合。  
如果您在一个群组下组织项目，它的工作方式就像一个文件夹。  
您可以管理群组成员的权限并访问群组中的每个项目。

# 创建项目（续1）

知识讲解



**新建项目**

项目可以用于存储你的文件（版本库）、安排你的工作（问题列表）、以及发布你的文档（维基页面）、以及 [一些其它事情](#)。  
项目建立后，所有这些功能都会被启用。不过你可以随后禁用那些你不需要的功能。

空白项目      从模板创建      导入项目

项目路径: `http://192.168.113.249/`      devops      项目名称: `core_py`

希望将几个相关联的项目放置于同一个命名空间下? [创建群组](#)

项目描述 (可选):  
用于存储核心语法的代码

可见等级

- ☐ 私有  
项目访问权限必须明确授权给每个用户。
- ☐ 内部  
该项目允许已登录的用户访问。
- ☒ 公开  
该项目允许任何人访问。

[创建项目](#)      [取消](#)

# 创建项目（续2）

知识讲解



在账号中 **新建 SSH 公钥** 之前将无法通过 SSH 拉取或推送代码。 [不再显示](#) | [稍后提醒](#)

devops > core\_py > 详细信息

项目 'core\_py' 已创建成功。

**core\_py**

用于存储核心语法的代码

☆ 星标 0      SSH `git@gitlab:devops/core_py.git`      +      全局

当前项目的版本仓库是空的

可以通过下面的命令行推送一个已存在的版本库。

或者可以从增加 [README](#), a [许可证](#), or a [.gitignore](#) 文件开始本项目。

由于主分支(master)会被自动保护, 只有当前项目的拥有者或者您具备主程序员权限才能进行初始推送。

您可以为此项目激活 [DevOps 自动化\(测试版\)](#)。

将根据预定义的 CI/CD 配置自动构建、测试和部署应用程序。

[新建文件](#)

# 创建用户

Tedu.cn  
达内教育

管理区域 > Dashboard

**Projects: 1**

New project

**Users: 1**

New user

**Groups: 1**

New group

**统计**

派生项目数	0
问题数	0
合并请求数	0
批注数	0
代码片段数	0
SSH 密钥数	0
里程碑数	0
激活用户数	1

**特性**

注册	●
LDAP	⏻
头像	●
OmniAuth	⏻
邮件回复	⏻
容器注册表	⏻
GitLab 页面	⏻
共享 Runners	●

**组件** update asap

GitLab	10.5.4
GitLab Shell	6.0.3
GitLab Workhorse	v3.6.0
GitLab API	v4
Ruby	2.3.6p384
Rails	4.2.10
postgresql	9.6.5
GitLab Servers	

## 创建用户（续1）

Tedu.cn  
达内教育

- 创建用户后，再次编辑可设置密码

GitLab 项目 群组 更多

账号

姓名: 张志刚  
\* 必须填写

用户名: zhangzg  
\* 必须填写

电子邮箱: zhangzg@tedu.cn  
\* 必须填写

**密码**

密码: 重置链接将自动生成临时密码发送给用户。用户在第一次登录后需要强制修改密码。

**权限**

项目限制: 100000

可以创建群组: ☒

权限级别: ☒ 普通用户  
普通用户可以访问他们的群组和项目

☐ 管理员  
管理员可以访问所有组、项目和用户，并且可以管理此安装中的所有功能

## 创建用户（续2）

- root用户将新用户加入组中，并设置新用户为“主程序员”

知识讲解



## 创建用户（续3）

- 新用户初次登陆，需设置自己的密码

知识讲解



# 用户管理

- 将本地生成的公钥上传至服务器

知识讲解



## 用户管理（续1）

- 将本地仓库推送至服务器

```
[root@localhost devops]# git remote rename origin old-origin
[root@localhost devops]# git remote add origin
git@192.168.113.249:/devops/core_py.git
[root@localhost devops]# git push -u origin --all
```

- 添加新文件

```
[root@localhost devops]# echo '# this is a test' > hi.py
[root@localhost devops]# git add hi.py
[root@localhost devops]# git commit hi.py -m '新的测试'
[root@localhost devops]# git push origin master
```

## 用户管理（续2）

知识讲解

- 下载代码到本地

```
[root@localhost ~]# git clone  
git@192.168.113.249:devops/core_py.git  
[root@localhost ~]# ls -a core_py/  
. .. .git hello.py hi.py
```

- 更新代码到本地

```
[root@localhost core_py]# git pull
```



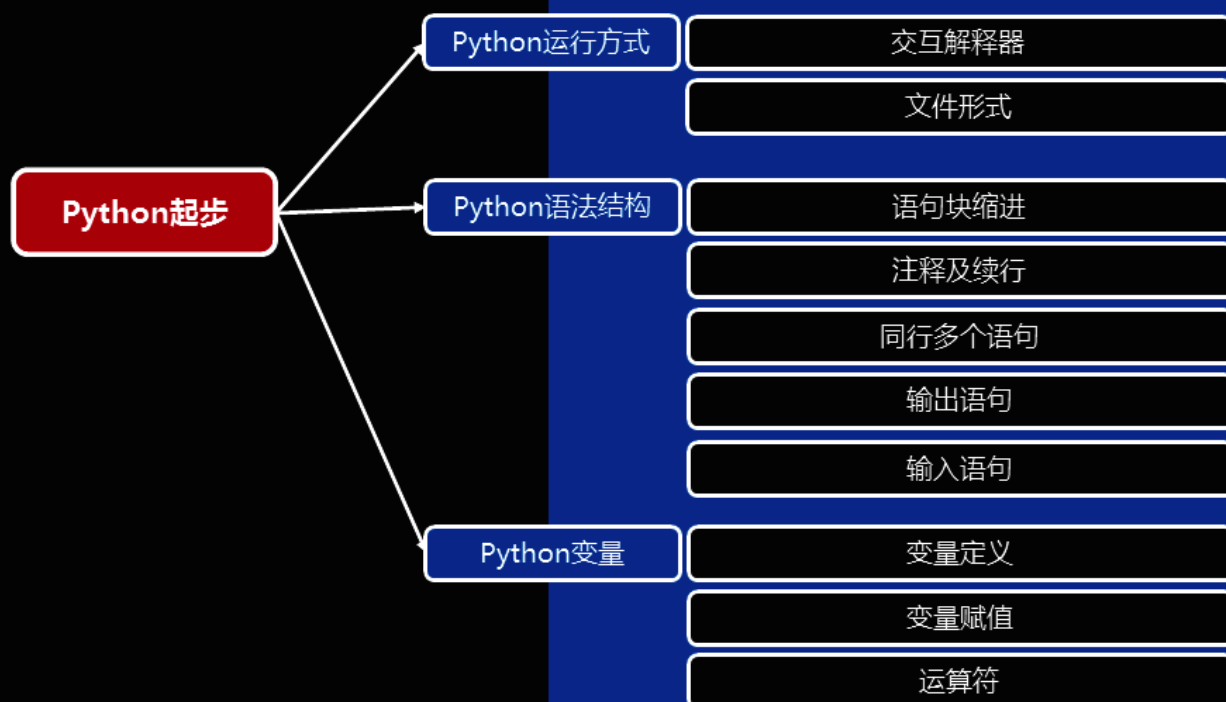
## 案例4：使用自建gitlab服务器

课堂练习

1. 通过docker搭建gitlab服务器
2. 新建群组devops
3. 新建项目core\_py
4. 新建用户，他/她在devops组中是主程序员
5. 新用户上传版本库到gitlab
6. 熟悉git远程操作方法



# Python起步



## Python运行方式



## 交互解释器

知识讲解

- 进入交互解释器

```
[root@zzghost1 bin]# python3
Python 3.6.3 (default, Oct 13 2017, 11:38:12)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-4)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- 退出交互解释器

```
>>> exit()
或
>>> ctrl + d
```



## 文件形式

知识讲解

- 明确指定解释器

```
[root@zzghost1 day01]# python3 hello.py
```

- 赋予python文件可执行权限

```
[root@zzghost1 day01]# chmod +x hello.py
[root@zzghost1 day01]# ./hello.py
```



# Python语法结构

## 语句块缩进

知识讲解

- python代码块通过缩进对齐表达代码逻辑而不是使用大括号
- 缩进表达一个语句属于哪个代码块
- 缩进风格
  - 1或2：可能不够，很难确定代码语句属于哪个块
  - 8至10：可能太多，如果代码内嵌的层次太多，就会使得代码很难阅读
  - 4个空格：非常流行，范·罗萨姆支持的风格



## 语句块缩进（续1）

知识讲解

- 缩进相同的一组语句构成一个代码块，称之代码组
- 首行以关键字开始，以冒号：结束，该行之后的一行或多行代码构成代码组
- 如果代码组只有一行，可以将其直接写在冒号后面，但是这样的写法可读性差，不推荐



## 注释及续行

知识讲解

- 首要说明的是：尽管Python是可读性最好的语言之一，这并不意味着程序员在代码中就可以不写注释
- 和很多UNIX脚本类似，python注释语句从#字符开始
- 注释可以在一行的任何地方开始，解释器会忽略掉该行#之后的所有内容
- 一行过长的语句可以使用反斜杠\分解成几行



## 同行多个语句

知识讲解

- 分号；允许你将多个语句写在同一行上
- 但是些语句不能在这行开始一个新的代码块
- 因为可读会变差，所以不推荐使用



## 输出语句

知识讲解

- 获取帮助

```
>>> help(print)
```

- 使用方式

```
>>> print('Hello World!')
>>> print('Hello' + 'World!')
>>> print('Hello', 'World!')
>>> print('Hello', 'World!', sep='***')
>>> print('Hello', 'World!', sep='***', end='')
```



# 输入语句

## 知识讲解

- 获得帮助

```
>>> help(input)
```

- 使用方式（注意，返回值一定是字符类型）

```
>>> num = input("Number: ")
```

```
Number: 20
```

```
>>> num + 10
```

```
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in <module>
```

```
TypeError: must be str, not int
```



## 案例5：模拟用户登陆

1. 创建名为login.py的程序文件
2. 程序提示用户输入用户名
3. 用户输入用户名后，打印欢迎用户

## 课堂练习



# Python变量

## 变量定义

知识讲解

- 变量名称约定
  - 第一个字符只能是大小写字母或下划线
  - 后续字符只能是大小写字母或数字或下划线
  - 区分大小写
- python是动态类型语言，即不需要预先声明变量的类型



## 变量定义（续1）

知识讲解

- 推荐采用的全名方法
  - 变量名全部采用小写字母
  - 简短、有意义
  - 多个单词间用下划线分隔
  - 变量名用名词，函数名用谓词（动词+名词）
  - 类名采用驼峰形式



## 变量赋值

知识讲解

- 变量的类型和值在赋值那一刻被初始化
- 变量赋值通过等号来执行
- python也支持增量赋值

```
>>> counter = 0
>>> name = 'bob'
>>> n += 1      #等价于n = n + 1
>>> n *= 1      #等价于n = n * 1
>>> i++
File "<stdin>", line 1
  i++
  ^
SyntaxError: invalid syntax
```



# 运算符

## 知识讲解

- 标准算术运算符

`+` `-` `*` `/` `//` `%` `**`

- 比较运算符

`<` `<=` `>` `>=` `==` `!=` `<>`

- 逻辑运算符

`and` `not` `or`



## 总结和答疑

总结和答疑

起动容器失败

问题现象

故障分析及排除



# 起动容器失败

## 问题现象

- 起动容器时，出现以下错误

Error response from daemon: driver failed programming external connectivity on endpoint gitlab (58e237fda6825cb3fe5944425b09496417c0a17ba27f1ebff6bc49429002a2a1): Error starting userland proxy: listen tcp 0.0.0.0:22: bind: address already in use.

知识讲解



# 故障分析及排除

## 知识讲解

- 原因分析
  - 在起动容器时，要将宿主机的22端口与容器的22端口进行映射
  - 提示地址已被占用
- 解决办法
  - 将宿主机ssh服务更换端口

