Tedu.cn
达内教育

运维开发实战

NSD DEVOPS

DAY01

内容

上午	09:00 ~ 09:30	多进程编程
	09:30 ~ 10:20	
	10:30 ~ 11:20	多线程编程
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	urllib模块
	15:00 ~ 15:50	
	16:10 ~ 17:00	paramiko模块
	17:10 ~ 18:00	总结和答疑

多进程编程


多进程编程
forking工作原理
什么是forking
进程的生命周期
僵尸进程
forking编程
forking编程基本思路
解决zombie问题

forking工作原理

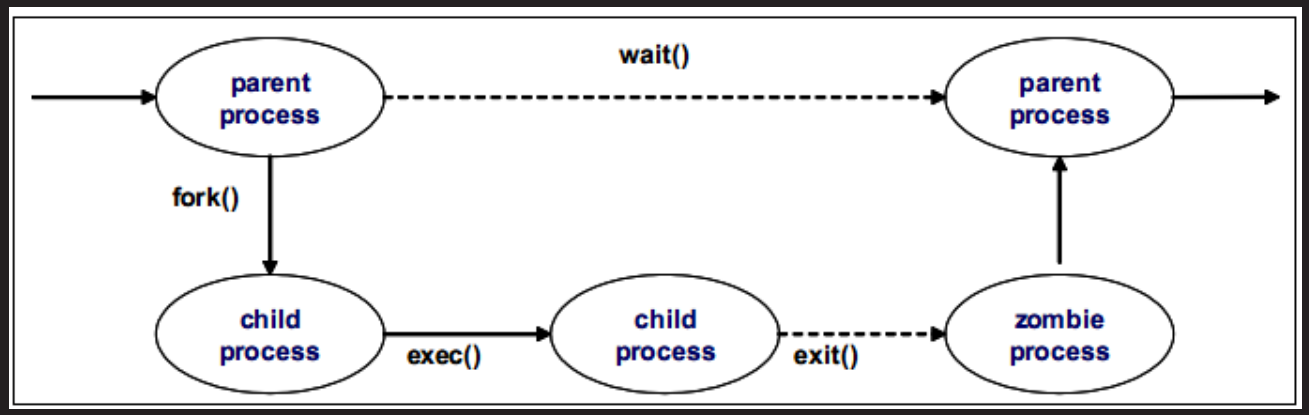
什么是forking

- fork（分岔）在Linux系统中使用非常广泛
- 当某一命令执行时，父进程（当前进程）fork出一个子进程
- 父进程将自身资源拷贝一份，命令在子进程中运行时，就具有和父进程完全一样的运行环境

知识讲解

进程的生命周期

- 父进程fork出子进程并挂起
- 子进程运行完毕后，释放大部分资源并通知父进程，这个时候，子进程被称作僵尸进程
- 父进程获知子进程结束，子进程所有资源释放



知识讲解

僵尸进程

- 僵尸进程没有任何可执行代码，也不能被调度
- 如果系统中存在过多的僵尸进程，将因为没有可用的进程号而导致系统不能产生新的进程
- 对于系统管理员来说，可以试图杀死其父进程或重启系统来消除僵尸进程

知识讲解

Tedu.cn
达内教育

forking编程

forking编程基本思路

- 需要使用os模块
- os.fork()函数实现forking功能
- python中，绝大多数的函数只返回一次，os.fork将返回两次
- 对fork()的调用，针对父进程返回子进程的PID；对于子进程，返回PID0

知识讲解

案例1：forking基础应用

- 编写一个forking脚本
  1. 在父进程中打印“In parent”然后睡眠10秒
  2. 在子进程中编写循环，循环5次，输出当系统时间，每次循环结束后睡眠1秒
  3. 父子进程结束后，分别打印“parent exit”和“child exit”

课堂练习

案例2：扫描存活主机

1. 通过ping测试主机是否可达
2. 如果ping不通，不管什么原因都认为主机不可用
3. 通过fork方式实现并发扫描

课堂练习

解决zombie问题

- 父进程通过os.wait()来得到子进程是否终止的信息
- 在子进程终止和父进程调用wait()之间的这段时间，子进程被称为zombie（僵尸）进程
- 如果子进程还没有终止，父进程先退出了，那么子进程会持续工作。系统自动将子进程的父进程设置为init进程，init将来负责清理僵尸进程

知识讲解

解决zombie问题（续1）

- python可以使用waitpid()来处理子进程
- waitpid()接受两个参数，第一个参数设置为-1，表示与wait()函数相同；第二参数如果设置为0表示挂起父进程，直到子程序退出，设置为1表示不挂起父进程
- waitpid()的返回值：如果子进程尚未结束则返回0，否则返回子进程的PID

知识讲解

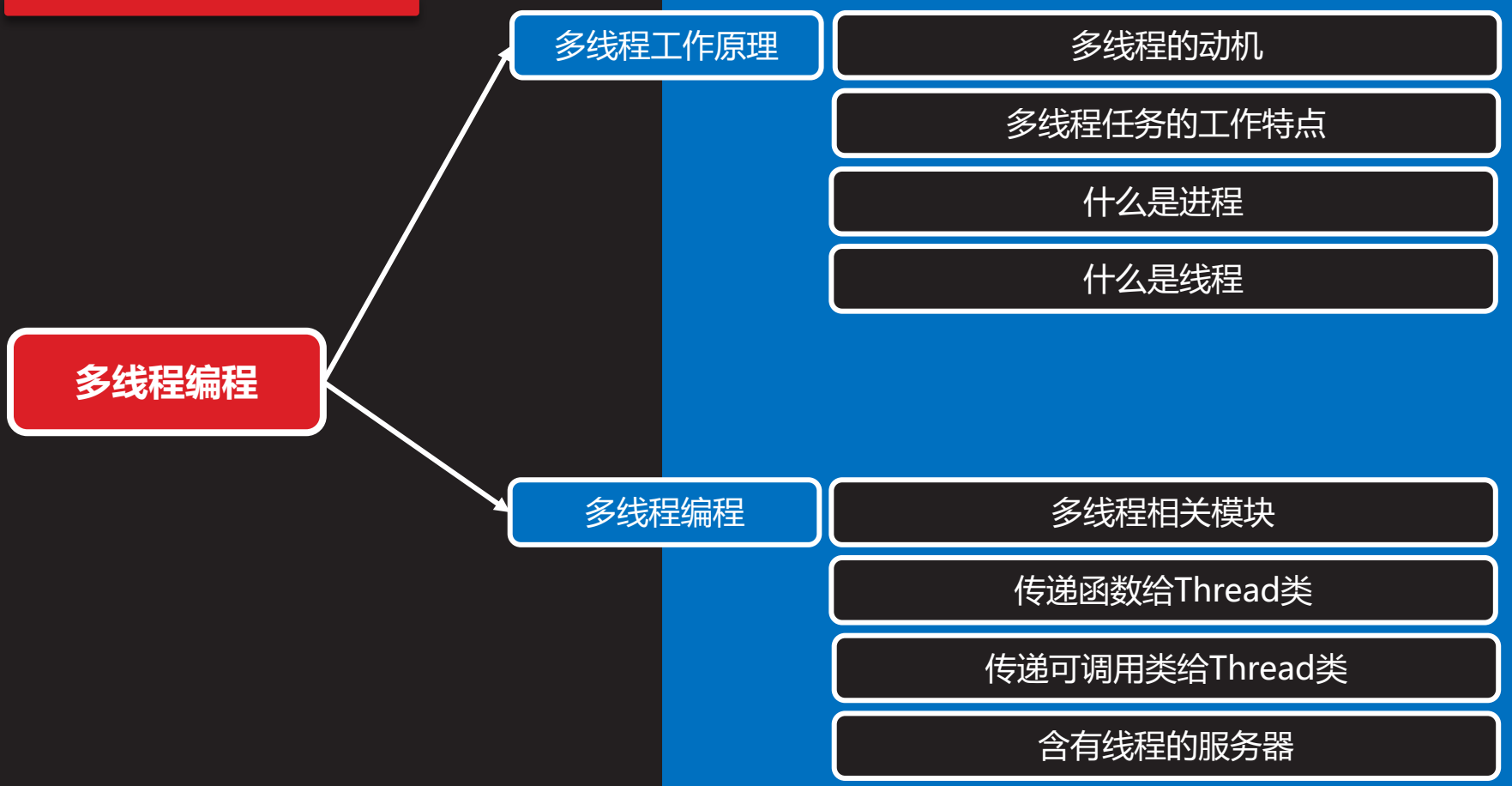
解决zombie问题（续2）

知识讲解

```
import os,  time
def reap():
    result = os.waitpid(-1, 1)
    print('Reaped child process %d' % result[0])

pid = os.fork()
if pid:
    print 'In parent. Sleeping 15s...'
    time.sleep(15)
    reap()
    time.sleep(5)
    print('parent done')
else:
    print 'In child. Sleeping 5s...'
    time.sleep(5)
    print('Child terminating.')
```

多线程编程


多线程编程
多线程工作原理
多线程的动机
多线程任务的工作特点
什么是进程
什么是线程
多线程编程
多线程相关模块
传递函数给Thread类
传递可调用类给Thread类
含有线程的服务器

Tedu.cn
达内教育

多线程工作原理

多线程的动机

- 在多线程（MT）编程出现之前，电脑程序的运行由一个执行序列组成，执行序列按顺序在主机的中央处理器（CPU）中运行
- 无论是任务本身要求顺序执行还是整个程序是由多个子任务组成，程序都是按这种方式执行的
- 即使子任务相互独立，互相无关（即，一个子任务的结果不影响其它子任务的结果）时也是这样
- 如果并行运行这些相互独立的子任务可以大幅度地提升整个任务的效率

知识讲解

多线程任务的工作特点

- 它们本质上就是异步的，需要有多个并发事务
- 各个事务的运行顺序可以是不确定的，随机的，不可预测的
- 这样的编程任务可以被分成多个执行流，每个流都有一个要完成的目标
- 根据应用的不同，这些子任务可能都要计算出一个中间结果，用于合并得到最后的结果

知识讲解

什么是进程

- 计算机程序只不过是磁盘中可执行的、二进制（或其它类型）的数据
- 进程（有时被称为重量级进程）是程序的一次执行
- 每个进程都有自己的地址空间、内存以及其它记录其运行轨迹的辅助数据
- 操作系统管理在其上运行的所有进程，并为这些进程公平地分配时间

知识讲解

什么是线程

- 线程（有时被称为轻量级进程）跟进程有些相似。不同的是，所有的线程运行在同一个进程中，共享相同的运行环境
- 一个进程中的各个线程之间共享同一片数据空间，所以线程之间可以比进程之间更方便地共享数据以及相互通讯

知识讲解

Tedu.cn
达内教育

多线程编程

多线程相关模块

- thread和threading模块允许程序员创建和管理线程
- thread模块提供了基本的线程和锁的支持，而threading提供了更高级别、功能更强的线程管理功能
- 推荐使用更高级别的threading模块

知识讲解

传递函数给Thread类

- 多线程编程有多种方法，传递函数给threading模块的Thread类是介绍的第一种方法
- Thread对象使用start()方法开始线程的执行，使用join()方法挂起程序，直到线程结束

知识讲解

传递可调用类给Thread类

- 传递可调用类给Thread类是介绍的第二种方法
- 相对于一个或几个函数来说，由于类对象里可以使用类的强大的功能，可以保存更多的信息，这种方法更为灵活

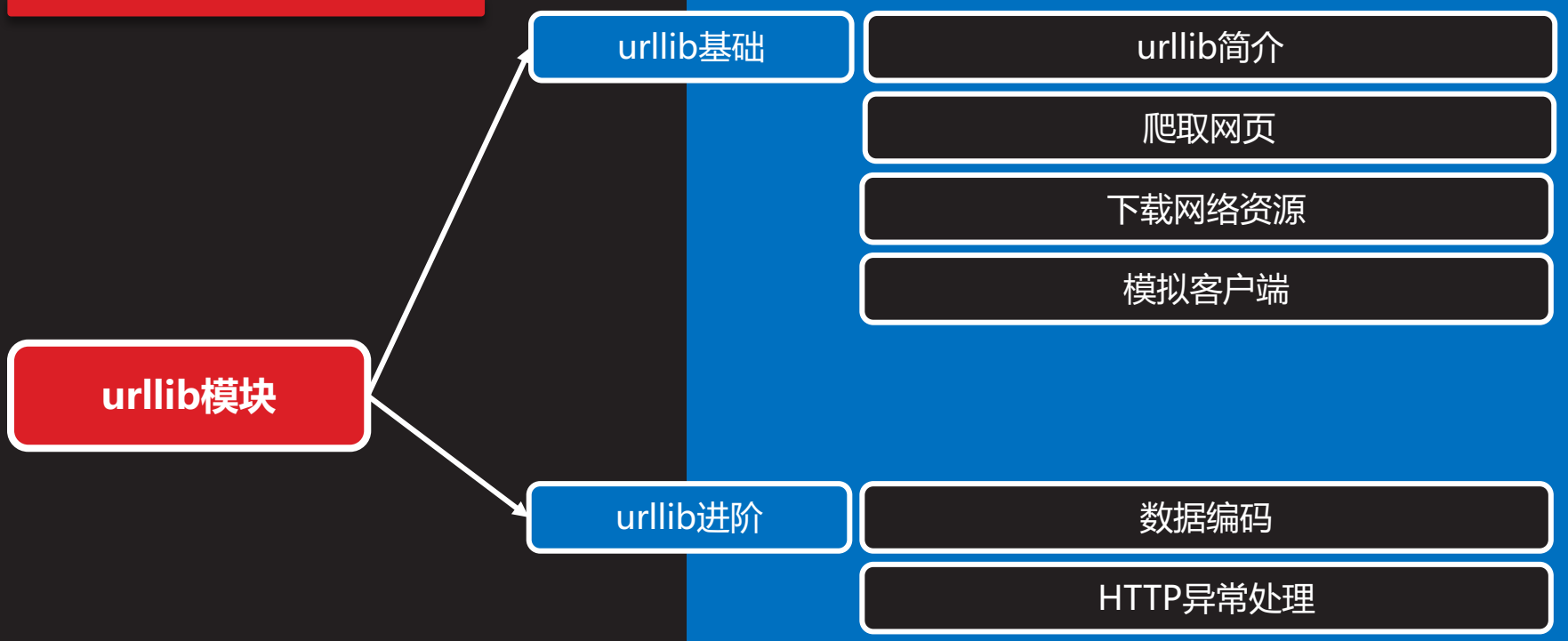
知识讲解

案例4：扫描存活主机

1. 通过ping测试主机是否可达
2. 如果ping不通，不管什么原因都认为主机不可用
3. 通过多线程方式实现并发扫描

课堂练习

urllib模块


urllib模块
urllib基础
urllib简介
爬取网页
下载网络资源
模拟客户端
urllib进阶
数据编码
HTTP异常处理

Tedu.cn
达内教育

urllib基础

urllib简介

- 在Python2版本中，有urllib和urlib2两个库可以用来实现request的发送。而在Python3中，已经不存在urllib2这个库了，统一为urllib
- urllib中包括了四个模块
  - urllib.request可以用来发送request和获取request的结果
  - urllib.error包含了urllib.request产生的异常
  - urllib.parse用来解析和处理URL
  - urllib.robotparse用来解析页面的robots.txt文件

知识讲解

爬取网页

知识讲解

- 先需要导入用到的模块：urllib.request
- 在导入了模块之后，我们需要使用urllib.request.urlopen打开并爬取一个网页
- 读取内容常见的有3种方式：
  - read()读取文件的全部内容，与readlines()不同的是，read()会把读取到的内容赋给一个字符串变量。
  - readlines()读取文件的全部内容，readlines()会把读取到的内容赋值给一个列表变量。
  - readline()读取文件的一行内容。

爬取网页（续1）

```
import urllib.request
html = urllib.request.urlopen('http://www.tedu.cn')
html.readline()
html.read(4096)
html.readlines()
```

知识讲解

案例2：爬取网页

1. 爬取的网页为http://www.tedu.cn
2. 保存的文件名为/tmp/tedu.html

课堂练习

下载网络资源

- urllib不仅可以下载网页，其他网络资源均可下载
- 有些文件比较大，需要像读取文件一样，每次读取一部分数据

```
import urllib.request
html = urllib.request.urlopen('http://172.40.50.116/python.pdf')
fobj = open('/tmp/python.pdf', 'ab')
while True:
    data = html.read(4096)
    if not data:
        break
    fobj.write(data)
fobj.close()
```

知识讲解

案例3：爬取图片

1. 将http://www.tedu.cn所有的图片下载到本地
2. 本地的目录为/tmp/images
3. 图片名与网站上图片名保持一致

课堂练习

模拟客户端

- 有些网页为了防止别人恶意采集其信息所以进行了一些反爬虫的设置，而我们又想进行爬取
- 可以设置一些Headers信息（User-Agent），模拟成浏览器去访问这些网站

知识讲解

```
import urllib.request

url='http://www.tedu.cn'
hearder={
   'User-Agent':'Mozilla/5.0 (X11; Fedora; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110
Safari/537.36'
}
html=urllib.request.Request(url,headers=header)
data=urllib.request.urlopen(request).read()
```

Tedu.cn
达内教育

urllib进阶

数据编码

知识讲解

- 一般来说，URL标准中只会允许一部分ASCII字符，比如数字、字母、部分符号等
- 而其他的一些字符，比如汉字等，>是不符合URL标准的。此时，我们需要编码。
- 如果要进行编码，可以使用urllib.request.quote()进行

```
>>> urllib.request.quote('hello world!')
'hello%20world%21'
>>> urllib.request.unquote('hello%20world%21')
'hello world!'
```

HTTP异常处理

- 如果访问的页面不存在或拒绝访问，程序将抛出异常
- 捕获异常需要导入urllib.error模块

知识讲解

```
>>> html = urllib.request.urlopen('http://172.40.50.116/a.html')
urllib.error.HTTPError: HTTP Error 404: Not Found
>>> html = urllib.request.urlopen('http://172.40.50.116/aaa')
urllib.error.HTTPError: HTTP Error 403: Forbidden
```

案例4：处理下载错误

1. 起动一个web服务
2. 在web服务器的文档目录下创建目录ban，权限设置为700
3. 编写python程序访问不存在的路径和ban目录，处理404和403错误
4. 404错误打印“无此页面”，403错误打印“无权访问”

课堂练习

paramiko模块

paramiko

安装paramiko模块

- 本地安装

```
# yum  install  -y  gcc  gcc-c++  python-devel
# tar  xzf  paramiko-1.15.4.tar.gz
# python  setup.py  install
```

- 网络安装

```
# pip install paramiko
```

知识讲解

基础使用介绍

- SSHClient
  - 创建用于连接ssh服务器的实例

```
>>> ssh = paramiko.SSHClient()
```

- paramiko.AutoAddPolicy
  - 设置自动添加主机密钥
- ssh.connect
  - 连接ssh服务器
- ssh.exec_comand
  - 在ssh服务器上执行指定命令

知识讲解

paramiko实例

- 编写用于实现ssh访问的脚本
  - 创建SSHClient实例
  - 设置添加主机密钥策略
  - 连接ssh服务器
  - 执行指定命令
  - 在shell命令行中接受用于连接远程服务器的密码以及在远程主机上执行的命令

知识讲解

案例5：多线程ssh并发访问

- 编写脚本程序
  1. 在文件中取出所有远程主机IP地址
  2. 在shell命令行中接受远程服务器IP地址文件、远程服务器密码以及在远程主机上执行的命令
  3. 通过多线程实现在所有的远程服务器上并发执行命令

课堂练习

Tedu.cn
达内教育

总结和答疑