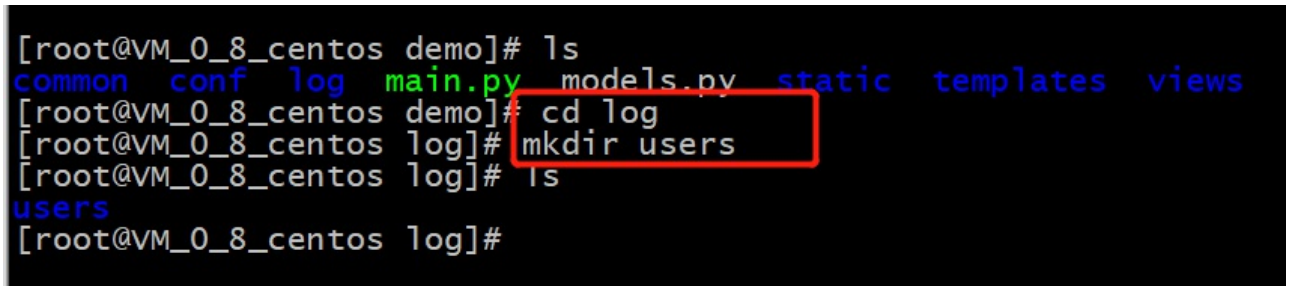
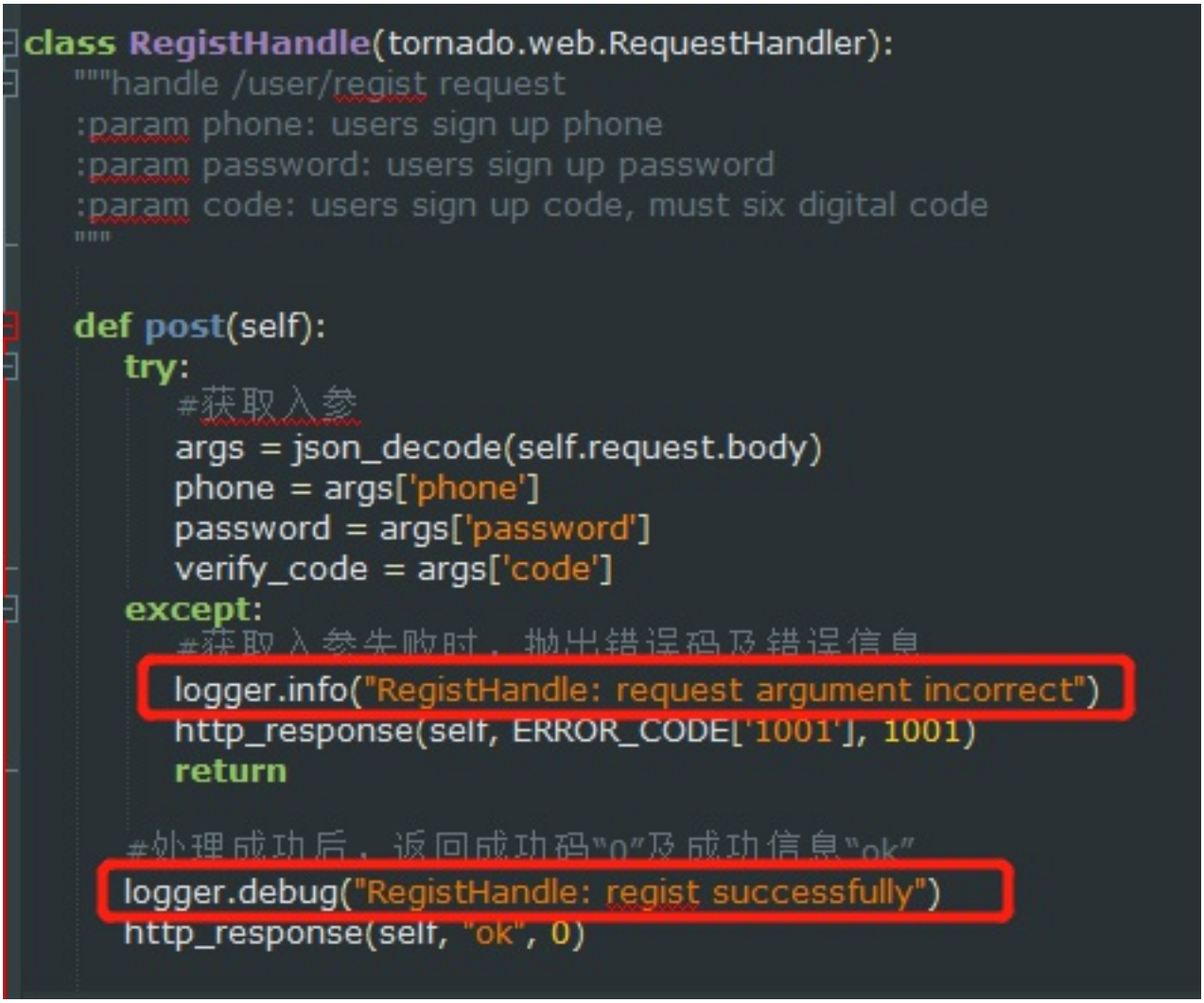
为用户处理模块增加 log 管理

作为一个程序员，log 管理几乎是必备技能，本小节将在原来代码的基础上，增加 log 管理，以方便调试。进入 log 目录，并创建 users 目录。



进入 users_views.py，导入 logging 模块，并指定 log 目录文件（log/users/users.log），指定 log 级别（DEBUG）和 log 保留方式（这里设定按天保存，保留 30 天的 log 记录），并在处理方法中加入对应的 log 信息。



users_views.py 的完整代码如下：

```
#! /usr/bin/python3
# -*- coding:utf-8 -*-

import tornado.web
from tornado.escape import json_decode
import logging
from logging.handlers import
TimedRotatingFileHandler

#从commons中导入http_response方法
from common.commons import (
    http_response,
```

```
)

#从配置文件中导入错误码
from conf.base import (
    ERROR_CODE,
)


########## Configure logging #############
logFilePath = "log/users/users.log"
logger = logging.getLogger("Users")
logger.setLevel(logging.DEBUG)
handler = TimedRotatingFileHandler(logFilePath,
                                   when="D",
                                   interval=1,

backupCount=30)
formatter = logging.Formatter('%(asctime)s \
%(filename)s[line:%(lineno)d] %(levelname)s %
(message)s',)
handler.suffix = "%Y%m%d"
handler.setFormatter(formatter)
logger.addHandler(handler)


class RegistHandle(tornado.web.RequestHandler):
    """handle /user/regist request
    :param phone: users sign up phone
    :param password: users sign up password
    :param code: users sign up code, must six
digital code
    """
```

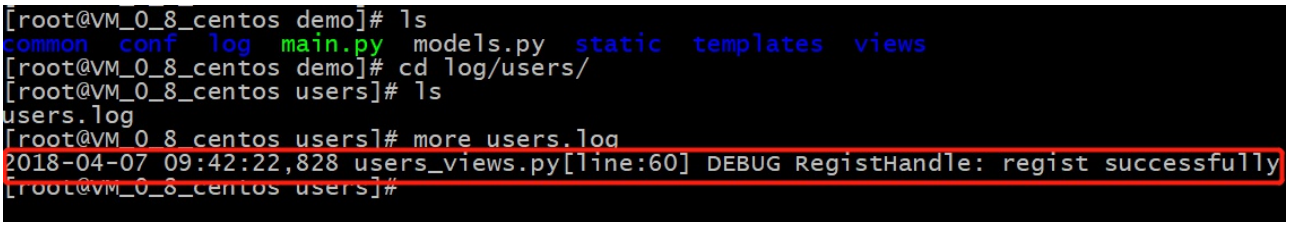
```
    def post(self):
        try:
            #获取入参
            args = json_decode(self.request.body)
            phone = args['phone']
            password = args['password']
            verify_code = args['code']
        except:
            #获取入参失败时，抛出错误码及错误信息
            logger.info("RegistHandle: request
argument incorrect")
            http_response(self,
ERROR_CODE['1001'], 1001)
            return

        #处理成功后，返回成功码“0”及成功信息“ok”
        logger.debug("RegistHandle: regist
successfully")
        http_response(self, ERROR_CODE['0'], 0)
```

再次执行 main.py

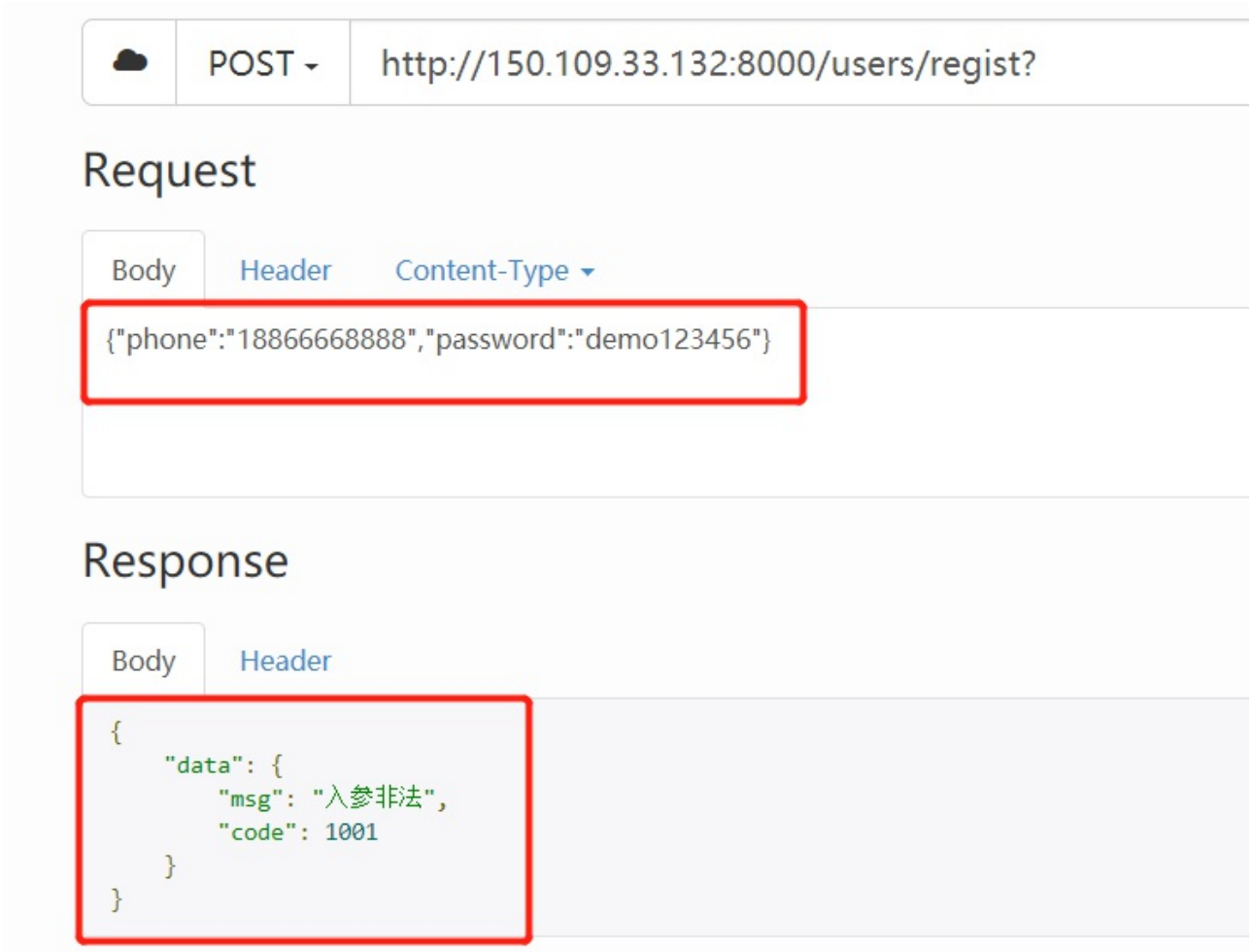
```
[root@VM_0_8_centos demo]#
[root@VM_0_8_centos demo]# ls
common  conf  log  main.py  models.py  static  templates  views
[root@VM_0_8_centos demo]# ./main.py
Tornado server is ready for service
[D 180407 09:25:24 users_views:60] RegistHandle: regist successfully
[I 180407 09:25:24 web:2106] 200 POST /users/regist (101.105.54.156) 1.15ms
```

HTTP 客户端发起正确的注册请求

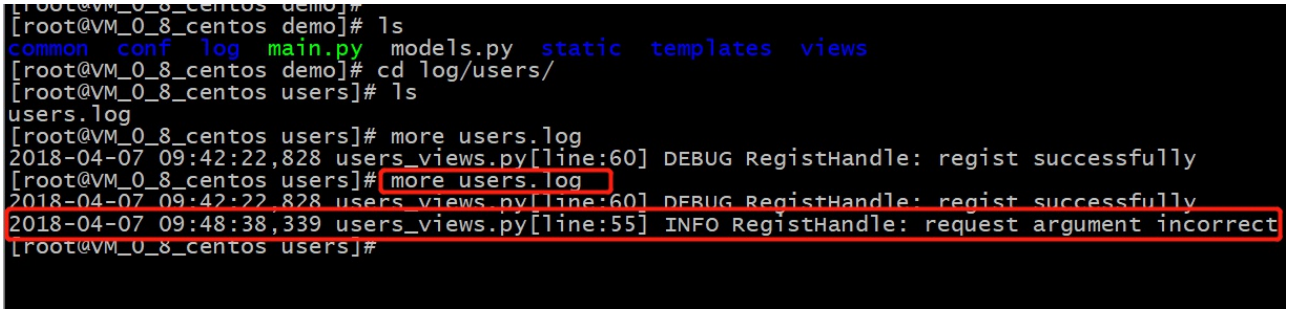
查看 log/users/users.log



在日志文件中，日志的格式包含时间、文件名、打印代码行数、log 级别和自定义 log 信息。这些信息足以满足问题定位及排错。在前面的配置信息中，我们定义的 log 级别是 DEBUG，下面看看入参出错时，报的 INFO 日志。



查看 log/users/users.log



这里看到日志文件中多出了一行日志，级别为 INFO。前面我们也提到，我们定义了日志文件的记录保留，本小册由于是新建讲解项目，还无法直接查看日志保留记录。这里贴出之前项目的记录，可以看到历史保留文件是以天为后缀的，当天的文件还是在 users.log 中。

名称	大小	压缩后大小	类型
..			本地磁盘
users.log	201	201	文本文档
users.log.20170922	740	740	20170922 文件
users.log.20170923	201	201	20170923 文件
users.log.20170925	169	169	20170925 文件
users.log.20170927	488	488	20170927 文件
users.log.20170929	201	201	20170929 文件
users.log.20170930	402	402	20170930 文件
users.log.20171003	201	201	20171003 文件
users.log.20171005	1,561	1,561	20171005 文件
users.log.20171006	941	941	20171006 文件
users.log.20171007	2,265	2,265	20171007 文件
users.log.20171008	1,260	1,260	20171008 文件
users.log.20171009	603	603	20171009 文件
users.log.20171011	571	571	20171011 文件
users.log.20171014	169	169	20171014 文件
users.log.20171016	201	201	20171016 文件

代码下载

到目前为止，服务器端代码如下：
[demo7 (https://github.com/Jawish185/demo7.git)](https://github.com/Jawish185/demo7.git)

小结

本小节简单介绍了日志服务在服务器端开发中的应用，开发者可以自定义 log 级别及其历史保留记录。开发者可以根据自己的喜好及习惯，去定义具体的级别和信息。下一小节，我们将讲解如何利用 ORM 的方式和数据库打交道，并将用户注册信息写入数据库中。