**4.2多环境配置·从配置对象中加载**

上文中三种常见的加载配置的方式，这里只演示第一种，也是我们这个工程化使用的方式。

# 新增config.py文件

from urllib import parse  
  
  
class BaseConfig(ｏｂｊｅｃｔ):  
 """  
 基础配置  
 """  
 DEBUG = True # 调试模式  
 APP\_AUTHOR = "mldong" # 作者  
 WHITE\_LIST = ['/user/get', '/user/list', '/ex/test', '/ex/other', "/db/test", "/config/test"] # 权限白名单  
 DB\_HOST = "localhost" # 数据库ip  
 DB\_PORT = "3306" # 数据库端口  
 DB\_NAME = "flaskapp" # 数据库名称  
 DB\_USER = "root" # 数据库用户  
 DB\_PASSWORD = "root" # 数据库密码  
 # 防止密码中有特殊字符，需要使用parse.quote\_plus进行转义  
 SQLALCHEMY\_DATABASE\_URI = f'mysql+pymysql://{DB\_USER}:{parse.quote\_plus(DB\_PASSWORD)}@{DB\_HOST}:{DB\_PORT}/{DB\_NAME}?charset=utf8mb4'  
 SQLALCHEMY\_TRACK\_MODIFICATIONS = True  
 SQLALCHEMY\_RECORD\_QUERIES = True  
 SQLALCHEMY\_ENGINE\_OPTIONS = {  
 'pool\_size': 5,  
 'pool\_timeout': 90,  
 'pool\_recycle': 7200,  
 'max\_overflow': 1024  
 }

# 修改app.py文件

from flask import Flask, request  
  
from config import BaseConfig  
from controllers.role\_controller import role  
from controllers.user\_controller import user  
from models import db  
  
app = Flask(\_\_name\_\_)  
  
# 注册用户模块  
app.register\_blueprint(user)  
# 注册角色模块  
app.register\_blueprint(role)  
# 从配置对象中加载  
app.config.from\_ｏｂｊｅｃｔ(BaseConfig)  
  
# 初始化db  
db.init\_app(app)  
  
  
@app.before\_request  
def auth():  
  
 # 简单处理一下，非白名单路由，提示登录  
 if request.path not in app.config['WHITE\_LIST']:  
 return "请先登录"  
  
  
@app.route("/db/test")  
def db\_test():  
 cursor = db.session.execute('select \* from t\_user')  
 result = cursor.fetchall()  
 if len(result) > 0:  
 u = result[0]  
 return f"id:{u.id},user\_name:{u.user\_name}"  
 return "无记录"  
  
  
@app.route("/config/test")  
def config\_test():  
 """  
 配置测试  
 :return:  
 """  
 return f"APP\_AUTHOR:{app.config['APP\_AUTHOR']}"  
  
  
@app.errorhandler(404)  
def error(e):  
 """  
 404异常处理  
 :param e:  
 :return:  
 """  
 return "请求地址不存在"  
  
  
@app.errorhandler(Exception)  
def error(e):  
 """  
 其他异常处理  
 :param e: 异常  
 :return:  
 """  
 return str(e)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(host="0.0.0.0", port=5000)

# 运行Flask服务

flask run

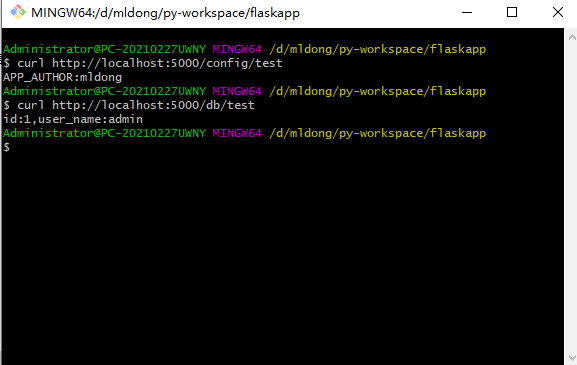
# 使用curl命令访问

## 访问/config/test

curl http://localhost:5000/config/test

## 访问/db/test

curl http://localhost:5000/db/test



# 使用Postman接口测试工具访问

略