链滴

Flask 工厂函数方式创建 APP

作者：YYJeffrey

原文链接：https://ld246.com/article/1576726303409

来源网站：链滴

使用工厂函数方式创建APP，更加符合规范，并且部署和测试也更加方便，灵活性也更加的高。一般厂函数创建APP首先需要先加载配置，然后初始化扩展，最后返回程序实例。

```
# __init__.py
import ...

db = SQLAlchemy()
cors = CORS()


def create_app(config_name=None):
    # 加载配置
    if config_name is None:
        config_name = os.getenv("FLASK_CONFIG", "dev")

    app = Flask(__name__)
    app.config.from_object(config_choice.get(config_name))

    register_logging(app)    # 加载日志处理器
    register_extensions(app)    # 注册扩展
    register_api(app)        # 注册API或者蓝图
    register_errors(app)    # 注册错误处理
    register_commands(app)    # 注册click或script命令
    register_shell_context(app)    # 注册shell上下文
    register_template_context(app)    # 注册模板上下文

    return app


def register_logging(app):
    app.logger.setLevel(logging.INFO)
    formatter = logging.Formatter("%(asctime)s - %(name)s - %(levelname)s - %(message)s")
```

原文链接：Flask 工厂函数方式创建 APP

```
    file_hanlder = RotatingFileHandler("logs/app.log", maxBytes=10 * 1024 * 1024, backupCou
t=10)
    file_hanlder.setFormatter(formatter)
    file_hanlder.setLevel(logging.INFO)

    if not app.debug:
        app.logger.addHandler(file_hanlder)


def register_extensions(app):
    db.init_app(app)  # 初始化db
    cors.init_app(app)  # 初始化跨域模块
    ...

def register_api(app):
    app.register_blueprint(index)
    app.register_blueprint(admin)
    ...


def register_errors(app):
    @app.errorhandler(400)
    def bad_request(e):
        return render_template("errors/400.html"), 400


def register_commands(app):
    pass


def register_shell_context(app):
    pass


def register_template_context(app):
    pass
```