flask

1.Flask框架的优势?

```
Flask自由、灵活，可扩展性强，透明可控，第三方库的选择面广，
开发时可以结合最流行最强大的Python库，
```

2.Flask框架依赖组件

```
依赖jinja2模板引擎
依赖werkzurg协议
```

3.Flask蓝图的作用

```
 blueprint把实现不同功能的module分开.也就是把一个大的App分割成各自实现不同功能的module.
 在一个blueprint中可以调用另一个blueprint的视图函数，但要加相应的blueprint名.
```

4.列举使用的Flask第三方组件?

```
# Flask组件
    flask-session  session放在redis
    flask-SQLAlchemy 如django里的ORM操作
    flask-migrate  数据库迁移
    flask-script  自定义命令
    blinker  信号-触发信号
# 第三方组件
    Wtforms 快速创建前端标签、文本校验
    dbutile     创建数据库连接池
    gevnet-websocket 实现websocket
# 自定义Flask组件
    自定义auth认证
    参考flask-login组件
```

5.简述Flask上下文管理流程?

```
falsk上下文管理可以分为三个阶段:
   1、请求上文 ->
    当请求进来,第一件事就是要把当前这个请求在我服务器上的线程开辟一个空间(线程对应的空间,必须含有stack对用一个列表存放ctx(request/session),具体-->:
        将request，session封装在 RequestContext类中
        app，g封装在AppContext类中
        并通过LocalStack将requestcontext和appcontext放入Local类中
        在local类中,以线程ID号作为key的字典
   2、请求下文:
        通过localproxy--->偏函数--->localstack--->local取值
   3、'请求响应时': -->要将上下文管理中的数据清除
        先执行save.session()再各自执行pop(),将local中的数据清除
```

6.Flask中的g的作用?

在flask中，有一个专门用来存储用户信息的g对象，g的全称的为global。
当请求进来将g和current_app封装为一个APPContext类，
再通过LocalStack将Appcontext放入Local中，取值时通过偏函数在LocalStack、local中取值；
响应时将local中的g数据删除

g与session的区别
session对象是可以跨request的，只要session还未失效，不同的request的请求会获取到同一个session，但是g对象不是，g对象不需要管过期时间，请求一次就g对象就改变了一次，或者重新赋值了一次。那么g对象该如何使用呢？看代码。

7.Flask中上下文管理主要涉及到了那些相关的类？并描述类主要作用?

```
RequestContext  #封装进来的请求（赋值给ctx）request.session
AppContext      #封装app_ctx
LocalStack      #将local对象中的数据维护成一个栈（先进后出）
Local           #保存请求上下文对象和app上下文对象
```

8.为什么要Flask把Local对象中的的值stack 维护成一个列表?

因为通过维护成列表，可以实现一个栈的数据结构，进栈出栈时只取一个数据，巧妙的简化了问题。
还有，在多app应用时，可以实现数据隔离；列表里不会加数据，而是会生成一个新的列表
local是一个字典，字典里key（stack）是唯一标识，value是一个列表

9.Flask中多app应用是怎么完成?

请求进来时，可以根据URL的不同，交给不同的APP处理。蓝图也可以实现。

```
    #app1 = Flask('app01')
    #app2 = Flask('app02')
    #@app1.route('/index')
    #@app2.route('/index2')
```

源码中在DispatcherMiddleware类里调用app2.__call__，
  原理其实就是URL分割，然后将请求分发给指定的app。
之后app也按单app的流程走。就是从app.__call__走。

10.在Flask中实现WebSocket需要什么组件?

```
 gevent-websocket
```

11.wtforms组件的作用?

快速创建前端标签、文本校验；如django的ModelForm

12.Flask框架默认session处理机制?

基于cookie实现，存储在 客户端的cookie中

必须设置默认的key才能生效

13.解释Flask框架中的Local对象和threading.local对象的区别?

有些应用使用的是greenlet协程，这种情况下无法保证协程之间数据的隔离，因为不同的协程可以在同一个线程当中。
即使使用的是线程，WSGI应用也无法保证每个http请求使用的都是不同的线程，因为后一个http请求可能使用的是之前的http请求的线程，这样的话存储于thread local中的数据可能是之前残留的数据。

为了解决上述问题，Werkzeug开发了自己的local对象，这也是为什么我们需要Werkzeug的local对象

14.Flask中 blinker 是什么?

flask中的信号blinker
信号主要是让开发者可是在flask请求过程中定制一些行为。
或者说flask在列表里面预留了几个空列表，在里面存东西。
简言之，信号允许某个'发送者'通知'接收者'有事情发生了

15.SQLAlchemy中的 session和scoped_session 的区别?

scoped_session封装了两个值 Session 和 registry,registry加括号就执行了ThreadLocalRegistry的__call__方法,如果

当前本地线程中有session就返回session,没有就将session添加到了本地线程
scoped_session为每个线程创建一个session,确保了线程安全

16.SQLAlchemy如何执行原生SQL?

```
# 总的数据sql
        sql_str = "select * from (" \
                  "select *, 0 as ord from sale_goods where goods_type = 'sale-goods' and
delete = false and total_count > 0 and deadline > now()" \
                  "union " \
                  "select *, 1 as ord from sale_goods where goods_type = 'sale-goods' and
delete = false and (total_count <= 0 or deadline <= now())" \
                  ") sale_goods_mall order by ord asc, c_time desc limit %d offset %d;" %
(limit, limit * (pages - 1))

        # 在售的数据sql
        in_sale_sql = "select *, 0 as ord from sale_goods where goods_type = 'sale-goods'
and delete = false and total_count > 0 and deadline > now();"

        # sqlalchemy执行sql
        data_query = db.session.execute(sql_str)    ################
        in_sale_query = db.session.execute(in_sale_sql)##########
```

17.ORM的实现原理?

⑴映射类：它的作用是描述数据库表的结构，表中的字段在类中被描述成属性，将来就可以实现把表中的记录映射成为该类的对象。
⑵映射文件：它的作用是指定数据库表和映射类之间的关系，包括映射类和数据库表的对应关系、表字段和类属性类型的对应关系以及表字段和类属性名称的对应关系等。
⑶数据库配置文件：它的作用是指定与数据库连接时需要的连接信息，比如连接哪中数据库、登录用户名、登录密码以及连接字符串等。

18.DBUtils模块的作用?

19.SQLAchemy中如何为表设置引擎和字符编码?

20.SQLAchemy中如何设置联合唯一索引?