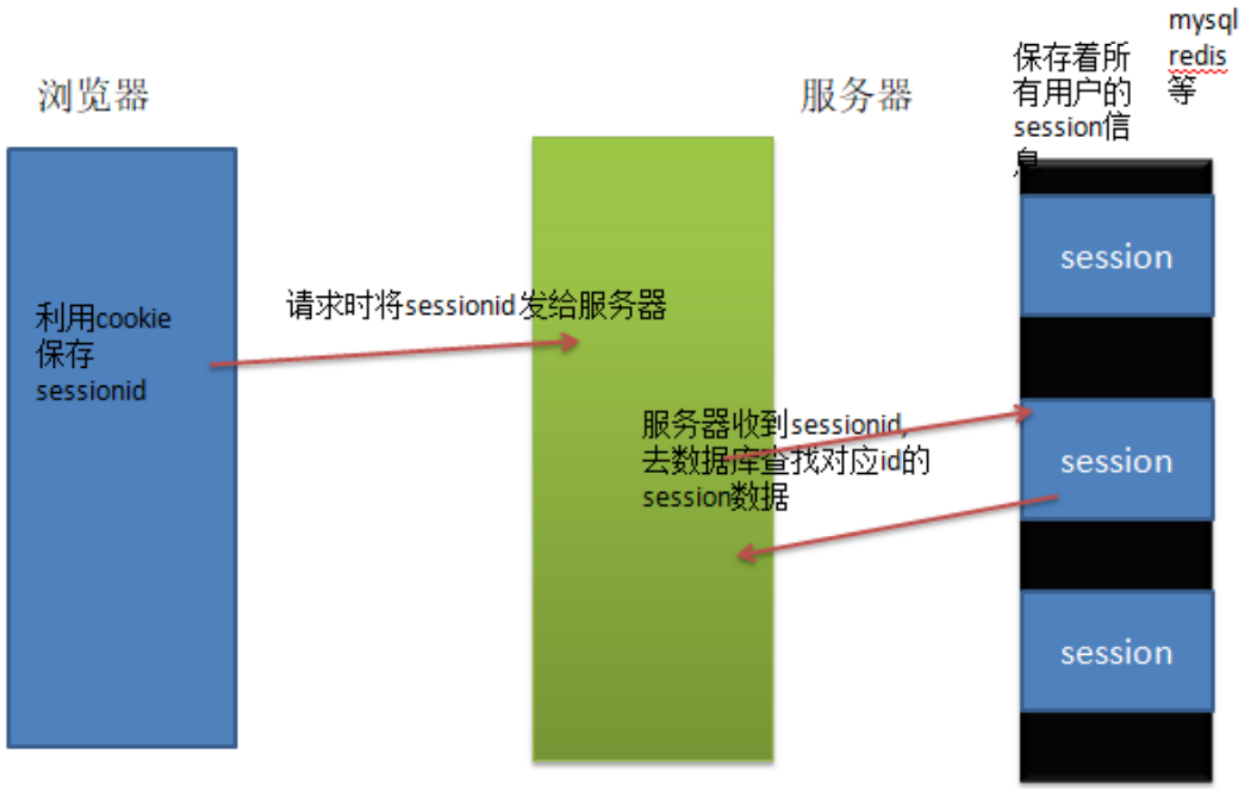
1.Session简介

与Cookie不同，Session（会话）数据存储在服务器上。会话是客户端登录到服务器并注销服务器的时间间隔。需要在该会话中保存的数据会存储在服务器上的临时目录中。

为每个客户端的会话分配会话ID。会话数据存储在cookie的顶部，服务器以加密方式对其进行签名。对于此加密，Flask应用程序需要一个定义的SECRET_KEY。

Flask session的默认存储方式是将整个数据加密后存储在cookie中，无后端存储



2.设置session

```
# Flask中需要使用session必须先配置
SECRET_KEY app.config['SECRET_KEY'] =
os.urandom(24)
```

或者

```
自己输入一个字符串
app.config['SECRET_KEY'] = 'sdfsdfs&&^%dsdf*/*$#'


----------
import os
os.urandom(n) # 返回n个字节的加密的随机字符串
```

视图函数中使用session跟python字典类似使用key获取值，或者使用get方法。

记得导包from flask import session

```
# -*- encoding: utf-8 -*-
# 导入Flask类
import os
from flask import Flask, session

# Flask 接收一个参数 name ，
# 导入模块的目录， flask以这个目录为基础，寻找静态文件目录static和模板目录
templates
app = Flask(__name__)

app.config['SECRET_KEY'] = 'qweqwr^%%&123?><\][ddd'


@app.route('/set_session')
def set_session():
    # 设置session 字典形式
    session['name'] = 'python'
    session['password'] = '123456'
    return 'session'


@app.route('/get_session')
def get_session():
    # 获取session 采用[key]方式取值，如果key不存在会报错。
    name = session['name']
    print(name)
    # 获取session 采用get方式取值，如果key不存在返回None，
    pwd = session.get('haha')
    print(pwd)
    return 'name:%s ' % name


if __name__ == '__main__':
```

```
    # Flask 应用程序实例的方法run启动web服务器
    app.run(debug=True)
```

3.删除session

可以直接使用session.pop('key',None):
session.pop('name',None)

如果要删除session中所有数据使用：clear():
session.clear()

```
@app.route('/del_session')
def del_session():
    # 删除session的某个键值对,返回删除的session值on，如果不存在可以设置一个默认值。
    result = session.pop('name', None)
    # 清除整个session.
    session.clear()
    return '删除session %s ' % result
```

4.设置session超时时间

Flask的默认session利用了Werkzeug的SecureCookie，把信息做序列化(pickle)后编码(base64)，放到cookie里了。

过期时间是通过cookie的过期时间实现的。

为了防止cookie内容被篡改，session会自动打上一个叫session的hash串，这个串是经过session内容、SECRET_KEY计算出来的，看得出，这种设计虽然不能保证session里的内容不泄露，但至少防止了不被篡改。

过期时间是这样来设置:

```
from datetime import timedelta

session.permanent = True  # flask里，仅仅是设置app.permanent_session_lifetime 还不能起作用，还需要指定session.permanent设置生效
# session的过期时间才是permanent_session_lifetime 所指定的，不然的话，要等到浏览器关闭，这个session才会过期

app.permanent_session_lifetime = timedelta(minutes=5) # 这个参数的作用是设置session的过期时间
# seconds 秒;   minutes分钟

session['key'] = value
```