1 cookie和session的作用：

cookie和session都记录了客户端的某种状态，用来跟踪用户访问网站的整个回话。两者最大的区别是cookie的信息是存放在浏览器客户端的，而session是存放在服务器端的.

1、设置Cookies

response.set\_cookie("cookie\_key","value")

2、获取Cookies

value = request.COOKIES["cookie\_key"]

3、删除Cookies

response.delete\_cookie("cookie\_key",path="/",domain=name)

4、检测Cookies

if "cookie\_name" is request.COOKIES :

5、rep.set\_signed\_cookie(key,value,salt='加密盐',...)

参数：

key, 键

value='', 值

max\_age=None, 超时时间，表示多少秒数之后失效

expires=None, 超时时间，表示失效的时间点。支持datetime 和 time.time

path='/', Cookie生效的路径，/ 表示根路径，特殊的：跟路径的cookie可以被任何url的页面访问

domain=None, Cookie生效的域名

secure=False, https传输

httponly=False 只能http协议传输，无法被JavaScript获取（不是绝对，底层抓包可以获取到也可以被覆盖）

Django中默认支持Session，其内部提供了5种类型的Session供开发者使用：

* 数据库（默认）
* 缓存
* 文件
* 缓存+数据库
* 加密cookie
* **数据库session**
* Django默认支持Session，并且默认是将Session数据存储在数据库中，即：django\_session 表中。
* a. 配置 settings.py
* SESSION\_ENGINE = 'django.contrib.sessions.backends.db' # 引擎（默认）
* SESSION\_COOKIE\_NAME ＝ "sessionid" # Session的cookie保存在浏览器上时的key，即：sessionid＝随机字符串（默认）
* SESSION\_COOKIE\_PATH ＝ "/" # Session的cookie保存的路径（默认）
* SESSION\_COOKIE\_DOMAIN = None # Session的cookie保存的域名（默认）
* SESSION\_COOKIE\_SECURE = False # 是否Https传输cookie（默认）
* SESSION\_COOKIE\_HTTPONLY = True # 是否Session的cookie只支持http传输（默认）
* SESSION\_COOKIE\_AGE = 1209600 # Session的cookie失效日期（2周）（默认）
* SESSION\_EXPIRE\_AT\_BROWSER\_CLOSE = False # 是否关闭浏览器使得Session过期（默认）
* SESSION\_SAVE\_EVERY\_REQUEST = False # 是否每次请求都保存Session，默认修改之后才保存（默认）

# 获取、设置、删除Session中数据

request.session['k1']

request.session.get('k1',None)

request.session['k1'] = 123

request.session.setdefault('k1',123) # 存在则不设置

del request.session['k1']

# 所有 键、值、键值对

request.session.keys()

request.session.values()

request.session.items()

request.session.iterkeys()

request.session.itervalues()

request.session.iteritems()

# 用户session的随机字符串

request.session.session\_key

# 将所有Session失效日期小于当前日期的数据删除

request.session.clear\_expired()

# 检查 用户session的随机字符串 在数据库中是否

request.session.exists("session\_key")

# 删除当前用户的所有Session数据

request.session.delete("session\_key")

缓存session

SESSION\_ENGINE = 'django.contrib.sessions.backends.cache' # 引擎

SESSION\_CACHE\_ALIAS = 'default' # 使用的缓存别名（默认内存缓存，也可以是memcache），此处别名依赖缓存的设置

SESSION\_COOKIE\_NAME ＝ "sessionid" # Session的cookie保存在浏览器上时的key，即：sessionid＝随机字符串

SESSION\_COOKIE\_PATH ＝ "/" # Session的cookie保存的路径

SESSION\_COOKIE\_DOMAIN = None # Session的cookie保存的域名

SESSION\_COOKIE\_SECURE = False # 是否Https传输cookie

SESSION\_COOKIE\_HTTPONLY = True # 是否Session的cookie只支持http传输

SESSION\_COOKIE\_AGE = 1209600 # Session的cookie失效日期（2周）

SESSION\_EXPIRE\_AT\_BROWSER\_CLOSE = False # 是否关闭浏览器使得Session过期

SESSION\_SAVE\_EVERY\_REQUEST = False # 是否每次请求都保存Session，默认修改之后才保存

### 文件session

配置 settings.py

SESSION\_ENGINE = 'django.contrib.sessions.backends.file' # 引擎

SESSION\_FILE\_PATH = None # 缓存文件路径，如果为None，则使用tempfile模块获取一个临时地址tempfile.gettempdir() # 如：/var/folders/d3/j9tj0gz93dg06bmwxmhh6\_xm0000gn/T

SESSION\_COOKIE\_NAME ＝ "sessionid" # Session的cookie保存在浏览器上时的key，即：sessionid＝随机字符串

SESSION\_COOKIE\_PATH ＝ "/" # Session的cookie保存的路径

SESSION\_COOKIE\_DOMAIN = None # Session的cookie保存的域名

SESSION\_COOKIE\_SECURE = False # 是否Https传输cookie

SESSION\_COOKIE\_HTTPONLY = True # 是否Session的cookie只支持http传输

SESSION\_COOKIE\_AGE = 1209600 # Session的cookie失效日期（2周）

SESSION\_EXPIRE\_AT\_BROWSER\_CLOSE = False # 是否关闭浏览器使得Session过期

SESSION\_SAVE\_EVERY\_REQUEST = False # 是否每次请求都保存Session，默认修改之后才保存

缓存+数据库Session

数据库用于做持久化，缓存用于提高效率

复制代码

a. 配置 settings.py

SESSION\_ENGINE = 'django.contrib.sessions.backends.cached\_db' # 引擎

加密cookie Session

a. 配置 settings.py

SESSION\_ENGINE = 'django.contrib.sessions.backends.signed\_cookies' # 引擎