1、获取Cookie：

|  |
| --- |
| request.COOKIES['key']  request.get\_signed\_cookie(key, default=RAISE\_ERROR, salt='', max\_age=None)      参数：          default: 默认值             salt: 加密盐          max\_age: 后台控制过期时间 |

2、设置Cookie：

|  |  |  |  |
| --- | --- | --- | --- |
| rep = HttpResponse(...) 或 rep ＝ render(request, ...)    rep.set\_cookie(key,value,...)  rep.set\_signed\_cookie(key,value,salt='加密盐',...)      参数：          key,              键          value='',         值          max\_age=None,     超时时间          expires=None,     超时时间(IE requires expires, so set it if hasn't been already.)          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） |  |  |  |

由于cookie保存在客户端的电脑上，所以，JavaScript和jquery也可以操作cookie。

|  |
| --- |
| <script src='/static/js/jquery.cookie.js'></script>  $.cookie("list\_pager\_num", 30,{ path: '/' }); |

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

1、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）        b. 使用        def index(request):          # 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")            request.session.set\_expiry(value)              \* 如果value是个整数，session会在些秒数后失效。              \* 如果value是个datatime或timedelta，session就会在这个时间后失效。              \* 如果value是0,用户关闭浏览器session就会失效。              \* 如果value是None,session会依赖全局session失效策略。 |

　　2、缓存Session

|  |
| --- |
| a. 配置 settings.py        SESSION\_ENGINE = 'django.contrib.sessions.backends.cache'  # 引擎      SESSION\_CACHE\_ALIAS = 'default'                            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      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        b. 使用        同上 |

3、文件Session

|  |
| --- |
| a. 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    b. 使用        同上 |

4、缓存+数据库Session

|  |
| --- |
| 数据库用于做持久化，缓存用于提高效率    a. 配置 settings.py        SESSION\_ENGINE = 'django.contrib.sessions.backends.cached\_db'        # 引擎    b. 使用        同上 |

5、加密cookie Session

|  |
| --- |
| a. 配置 settings.py        SESSION\_ENGINE = 'django.contrib.sessions.backends.signed\_cookies'   # 引擎    b. 使用        同上 |

扩展：Session用户验证

|  |
| --- |
| def login(func):      def wrap(request, \*args, \*\*kwargs):          # 如果未登陆，跳转到指定页面          if request.path == '/test/':              return redirect('http://www.baidu.com')          return func(request, \*args, \*\*kwargs)      return wrap |