**附加的关键字参数直接在Cookie.Morsel上设置。有关可用属性 ，请参见**

[https://docs.python.org/2/library/cookie.html#Cookie.Morsel](https://docs.python.org/2/library/cookie.html" \l "Cookie.Morsel)

**设置cookie**

RequestHandler**.**set\_cookie(self, name, value, domain=None, expires=None,path=**"/"**,expires\_days=None, \*\*kwargs)

name=cookie名称,value=cookie值,path=路径,expires=过期秒数,expires\_days=过期天数

httponly=防止XSS攻击, max\_age=优先级高于expires

RequestHandler**.**get\_cookie(self, name, default=None)

**设置安全cookie**

RequestHandler**.**set\_secure\_cookie(self, name, value, expires\_days=30, version=None, \*\*kwargs)

RequestHandler.get\_secure\_cookie(self, name, value=None, max\_age\_days=31, min\_version=None)

**删除cookie**

RequestHandler**.**clear\_cookie(self, name, path=**"/"**, domain=None):

RequestHandler**.**clear\_all\_cookies(self, path=**"/"**, domain=None):

**class** CookieHandler(tornado.web.RequestHandler):  
 **def** get(self):

self.write(**'Hello Tornado'**)

self.set\_cookie(**'aaa'**,**'111'**) #设置cookie  
 **print** self.get\_cookie(**'aaa'**) #获取cookie

self.set\_secure\_cookie(**'aaa'**,**'111'**) #设置安全cookie,需在settings设置cookie\_secret  
 **print** self.get\_secure\_cookie(**'aaa'**) #获取安全cookie

self.clear\_cookie(**'aaa'**) #删除cookie  
self.clear\_all\_cookies() #删除所有cookie

handlers = [ *#handlers路由映射* (**r'/cookie'**,CookieHandler),  
]

settings = {

"cookie\_secret": "bZJc2sWbQLKos6GkHn/VB9oXwQt8S0R0kRvJ5/xJ89E="

}

传递给Application构造函数的cookie\_secret值应该是唯一的随机字符串。

import base64, uuid  
print base64.b64encode(uuid.uuid4().bytes + uuid.uuid4().bytes)

"bZJc2sWbQLKos6GkHn/VB9oXwQt8S0R0kRvJ5/xJ89E="

然而，Tornado的安全cookies仍然容易被窃听。攻击者可能会通过脚本或浏览器插件截获cookies，或者干脆窃听未加密的网络数据。记住cookie值是签名的而不是加密的。恶意程序能够读取已存储的cookies，并且可以传输他们的数据到任意服务器，或者通过发送没有修改的数据给应用伪造请求。httponly和secure属性可以帮助我们防范这种攻击。

为了开启这些功能，你可以向set\_cookie和set\_secure\_cookie方法传递关键字参数。比如，一个安全的HTTP-only cookie（不是Tornado的签名cookie）可以调用self.set\_cookie('foo', 'bar', httponly=True, secure=True)发送。

1.domain

是cookie所在的域，默认为请求的地址，

如网址为http://192.168.201.130:8000/cookie，那么domain默认为192.168.201.130:8000

domain属性可以使多个web服务器共享cookie。domain属性的默认值是创建cookie的网页所在服务器的主机名。不能将一个cookie的域设置成服务器所在的域之外的域。

news.baidu.com的服务器能够读取www.baidu.com设置的cookie值。如果www.baidu.com的页面创建的cookie把自己的path属性设置为“/”，把domain属性设置成“.baidu.com”，

那么所有位于www.baidu.com，以及位于baidu.com域的其他服务器上的网页都可以访问这个cookie。

2.path

表示cookie所在的目录路径，默认为/，就是根目录。

在同一个服务器上有目录如下：/hello，  / hello/aa， / hello/bb

设置第一个cookie1的path值为/hello，hello路径下的所有页面都可以访问到cookie1

设置第一个 cookie2的path值为 /hello/aa，/hello和/hello/bb的子页面不能访问cookie2

cookie能让其路径下的所有子页面访问。

浏览器会将domain和path都相同的cookie保存在一个文件里，当访问服务器的时候就会自动提交

3、HttpOnly防止XSS攻击

cookie设置HttpOnly属性，通过js脚本（document.cookie）无法读取到cookie信息，能有效的防止XSS攻击

4、secure

布尔值,传输类型（安全传输false/不安全传输true），默认是不安全传输

当设置为true时，表示创建的 Cookie 会被以安全的形式向服务器传输，也就是只能在 HTTPS 连接中被浏览器传递到服务器端进行会话验证，如果是 HTTP 连接则不会传递该信息，所以不会被窃取到Cookie 的具体内容。

5、expires

绝对失效时间，时间格式只支持GTM标准时间，所有浏览器都支持

默认情况下cookie的生存时间到退出浏览器时结束，可以设置expires为未来的一个失效时间。

6、max\_age

相对失效时间，格式是数字，单位秒

max\_age和expires两个都设置了，max-age的优先级高于expires。

注意问题

1、Cookie数量和长度的限制

各个浏览器的限制不同IE7+和Firefox最大限制为50条，chrome和Safari无限制，IE6-最大限制20条。

且所有浏览器限制每个cookie长度不能超过4KB，否则会被截掉。浏览器cookie个数不要超过50条，大小不能超过4KB

2、安全性问题

客服端每次访问服务端都会提交cookie，cookie容易被劫持，劫持者原样发cookie就可以冒用其他用户，

cookie绝不能保存敏感信息

cookie过程

1、服务器向客户端发送cookie

2、浏览器将cookie保存

3、之后每次http请求浏览器都会将cookie发送给服务器端