Session

用户会话

服务器无法识别每一次 HTTP 请求的出处（不知道来自于哪个终端），它只会接受到一个请求信号，所以就存在一个问题：将用户的响应发送给其他人，必须有一种技术来让服务器知道请求来自哪，这就是会话技术。

会话：就是客户端和服务器之间发生的一系列连续的请求和响应的过程，打开浏览器进行操作到关闭浏览器的过程。

会话状态：指服务器和浏览器在会话过程中产生的状态信息，借助于会话状态，服务器能够把属于同一次会话的一系列请求和响应关联起来。

实现会话有两种方式：

- session
- cookie

属于同一次会话的请求都有一个相同的标识符，sessionID

session 常用的方法：

String getId() 获取 sessionID

void setMaxInactiveInterval(int interval) 设置 session 的失效时间，单位为秒

int getMaxInactiveInterval() 获取当前 session 的失效时间

void invalidate() 设置 session 立即失效

void setAttribute(String key,Object value) 通过键值对的形式来存储数据

Object getAttribute(String key) 通过键获取对应的数据

void removeAttribute(String key) 通过键删除对应的数据

Cookie

Cookie 是服务端在 HTTP 响应中附带传给浏览器的一个小文本文件，一旦浏览器保存了某个 Cookie，在之后的请求和响应过程中，会将此 Cookie 来回传递，这样就可以通过 Cookie 这个载体完成客户端和服务端的数据交互。

Cookie

- 创建 Cookie

```
Cookie cookie = new Cookie("name","tom");
response.addCookie(cookie);
```

- 读取 Cookie

```
Cookie[] cookies = request.getCookies();
for (Cookie cookie:cookies){
    out.write(cookie.getName()+":"+cookie.getValue()+"<br/>");
}
```

Cookie 常用的方法

void setMaxAge(int age) 设置 Cookie 的有效时间，单位为秒

int getMaxAge() 获取 Cookie 的有效时间

String getName() 获取 Cookie 的 name

String getValue() 获取 Cookie 的 value

Session 和 Cookie 的区别

session：保存在服务器

保存的数据是 Object

会随着会话的结束而销毁

保存重要信息

cookie：保存在浏览器

保存的数据是 String

可以长期保存在浏览器中，无会话无关

保存不重要信息

存储用户信息：

session：setAttribute("name","admin") 存

getAttribute("name") 取

生命周期：服务端：只要 WEB 应用重启就销毁，客户端：只要浏览器关闭就销毁。

退出登录：session.invalidate()

cookie：response.addCookie(new Cookie(name,"admin")) 存

```
Cookie[] cookies = request.getCookies();
for (Cookie cookie:cookies){
    if(cookie.getName().equals("name")){
        out.write("欢迎回来"+cookie.getValue());
    }
}
```

取

生命周期：不随服务端的重启而销毁，客户端：默认是只要关闭浏览器就销毁，我们通过 setMaxAge() 方法设置有效期，一旦设置了有效期，则不随浏览器的关闭而销毁，而是由设置的时间来决定。

退出登录：setMaxAge(0)