链滴

Solo 登录状态和会话

作者：88250

原文链接：https://ld246.com/article/1538708370634

来源网站：链滴

本文是《Solo 从设计到实现》的一个章节，该系列文章将介绍 Solo 这款 Java 博客系统是如何从无有的，希望大家能通过它对 Solo 从设计到实现有个直观地了解、能为想参与贡献的人介绍清楚项目也希望能为给~~重复发明~~重新定义博客系统的人做个参考
eart

Solo 的用户登录状态是依赖浏览器客户端 Cookie 的。服务端依赖容器提供的会话，如果会话没有过，则直接使用，如果过期则通过 Cookie 进行验证，验证通过的话就再次创建会话。

方法 Solos#getCurrentUser：

```
for (int i = 0; i < cookies.length; i++) {
    final Cookie cookie = cookies[i];
    if (!COOKIE_NAME.equals(cookie.getName())) {
        continue;
    }

    final String value = Crypts.decryptByAES(cookie.getValue(), COOKIE_SECRET);
    final JSONObject cookieJSONObject = new JSONObject(value);

    final String userId = cookieJSONObject.optString(Keys.OBJECT_ID);
    if (StringUtils.isBlank(userId)) {
        break;
    }

    JSONObject user = userRepository.get(userId);
    if (null == user) {
        break;
    }

    final String userPassword = user.optString(User.USER_PASSWORD);
    final String token = cookieJSONObject.optString(Keys.TOKEN);
    final String hashPassword = StringUtils.substringBeforeLast(token, ":");
    if (userPassword.equals(hashPassword)) {
        login(request, response, user);

        return currentUser(request);
    }
}
```

方法 Solos#login：

```
final JSONObject cookieJSONObject = new JSONObject();
cookieJSONObject.put(Keys.OBJECT_ID, user.optString(Keys.OBJECT_ID));
cookieJSONObject.put(User.USER_PASSWORD, user.optString(User.USER_PASSWORD));

final String random = RandomStringUtils.random(16);
cookieJSONObject.put(Keys.TOKEN, user.optString(User.USER_PASSWORD) + ":" + random);

final String cookieValue = Crypts.encryptByAES(cookieJSONObject.toString(), COOKIE_SECRET
;
final Cookie cookie = new Cookie(COOKIE_NAME, cookieValue);
cookie.setPath("/");
cookie.setMaxAge(COOKIE_EXPIRY);
cookie.setHttpOnly(COOKIE_HTTP_ONLY);
```

```
response.addCookie(cookie);
```

可以看出，Cookie 通过 AES 加密后写回客户端，保证了一定的安全性。需要注意的是，密钥是配置在 latke.props 中的（cookieSecret），如果不配置则每次启动时生成随机密钥。

具体判断是否的点是通过 AOP 拦截器实现的，ConsoleAuthAdvice.java 和 ConsoleAdminAuthAdvice.java 分别用于判断是否登录以及是否是管理员登录。

```
@Override
public void doAdvice(final HTTPRequestContext context, final Map<String, Object> args) thr
ws RequestProcessAdviceException {
    final HttpServletRequest request = context.getRequest();
    if (!Solos.isAdminLoggedIn(request)) {
        final JSONObject exception401 = new JSONObject();
        exception401.put(Keys.MSG, "Unauthorized to request [" + request.getRequestURI() + "]"
;
        exception401.put(Keys.STATUS_CODE, HttpServletResponse.SC_UNAUTHORIZED);

        throw new RequestProcessAdviceException(exception401);
    }
}
```

对了，在大多数情况下，我们不应该把 request 传到 service 层，因为这样做的话 service 层就变成**状态**的了。这里的“状态”主要是用户会话状态。当然，这也不是绝对的，有的场景在服务层是需要取用户状态的：

- 根据 Cookie 服务统计服务，避免浏览数重复计数
- 判断登录状态的服务，服务端会话不存在时需要用 Cookie 重登