

搜 索

基于 OAuth 安全协议的 Java 应用编程 （转载）

[CevenCheng](#) 收藏于 2011-01-28 阅读数： 公众公开 [原文来源](#)

 [转藏到我的图书馆](#)

基于 OAuth 安全协议的 Java 应用编程

呈师, IBM






力于使网站和应用程序（统称为消费方 Consumer）能够在无须用户透露个人信息的情况下，通过 API 访问该用户在服务提供方（Service Provider）那里的受保护资源。OAuth 为 API 认证提供了一个可自由实现且通用的方法。目前互联网很多服务都提供了 OAuth 认证服务，OAuth 标准也逐渐成为开放资源授权的标准。本文主要介绍 Google Code 上提供的 OAuth Java 库来实现基于 OAuth 认证的 Java 应用，并给出使用 OAuth 方式访问 Google Data 的例子。

发布日期： 2010 年 3 月 18 日

级别： 中级

访问情况 2470 次浏览

建议： 0 ([添加评论](#))

     平均分 （共 22 个评分）

ook、Chris Messina、Larry Halff 及 David Recordon 共同发起的，目的在于为 API 访问授权提供一个安全、开放的框架。

具有以下特点：

与别的授权方式不同之处在于：OAuth 的授权不会使消费方（Consumer）触及到用户的帐号信息（如用户名与密码），消费方无需使用用户的用户名与密码就可以申请获得该用户资源的授权。

消费方都可以使用 OAuth 认证服务，任何服务提供方 (Service Provider) 都可以实现自身的 OAuth 认证服务。

消费方还是服务提供方，都很容易于理解与使用。

下图所示。

ion



[关闭](#)

Google 提供的广告

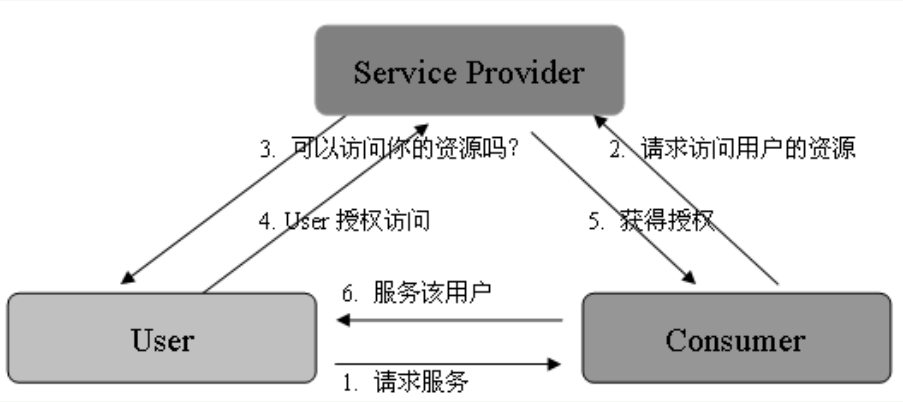
[Java下载](#)

[Java](#)

[Google Gmail](#)

热点推荐

- [书法教程：书法知识问题...](#)
- [秘制饺子蘸料「美味收藏」](#)
- [人一生要读的60篇散文...](#)
- [如何自制韩国泡菜](#)
- [中国人口在历史上的变迁...](#)
- [诗词集汇索引（2）](#)
- [一爱...](#)
- [熟人越来越多，朋友越来越...](#)
- [杨惠妍26岁登上中国首富...](#)
- [1966年中央的四次重...](#)
- [知识PK知识](#)
- [全球管理者的10大思想...](#)
- [全唐诗](#)
- [电脑使用技巧荟萃](#)
- [\[自然美景\]瞬间的魅力](#)
- [中国金银硬币欣赏\[图\]...](#)
- [你可以不美丽，但一定要...](#)
- [野外外伤急救技术](#)
- [极致影像的三个规则 最...](#)
- [是谁干掉了林彪“256...](#)
- [你想过人的一生要赚多少...](#)



如图 1 所示 OAuth 解决方案中用户、消费方及其服务提供方之间的三角关系：当用户需要 Consumer 为其提供某种服务时，该服务涉及到需要从服务提供方那里获取该用户的保护资源。OAuth 保证：只有在用户显式授权的情况下（步骤 4），消费方才可以获取该用户的资源，并用来服务于该用户。

从宏观层次来看，OAuth 按以下方式工作：

1. 消费方与不同的服务提供方建立了关系。
2. 消费方共享一个密码短语或者是公钥给服务提供方，服务提供方使用该公钥来确认消费方的身份。
3. 消费方根据服务提供方将用户重定向到登录页面。
4. 该用户登录后告诉服务提供方该消费方访问他的保护资源是没问题的。

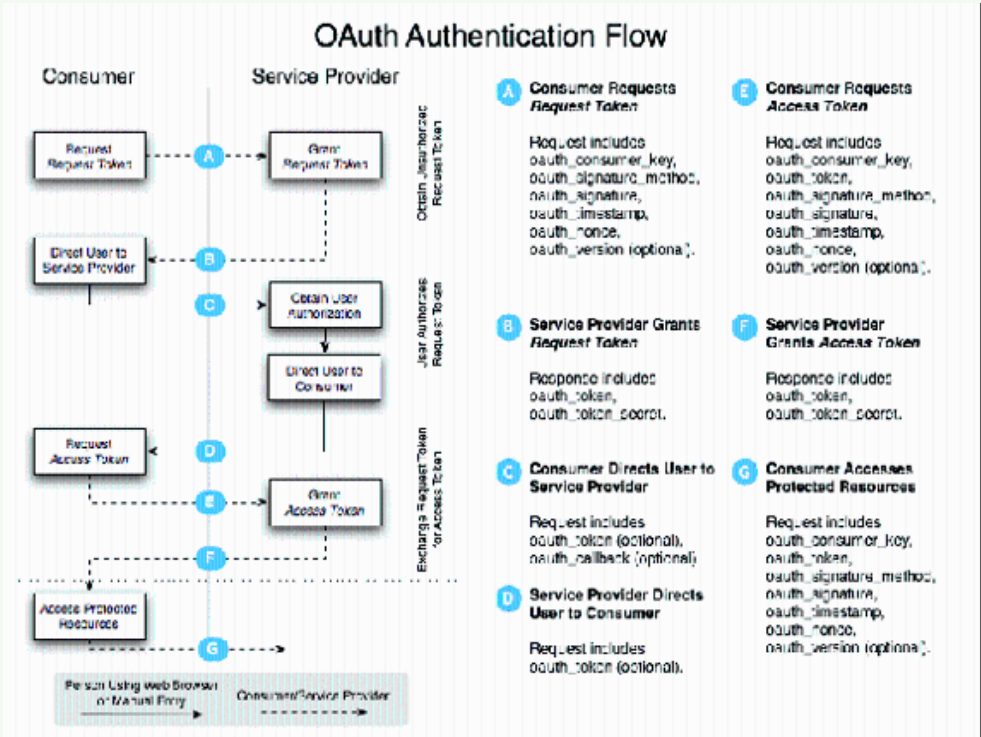


OAuth 认证授权流程

在了解 OAuth 认证流程之前，我们先来了解一下 OAuth 协议的一些基本术语定义：

- **Consumer Key**：消费方对于服务提供方的身份唯一标识。
- **Consumer Secret**：用来确认消费方对于 **Consumer Key** 的拥有关系。
- **Request Token**：获得用户授权的请求令牌，用于交换 **Access Token**。
- **Access Token**：用于获得用户在服务提供方的受保护资源。
- **Token Secret**：用来确认消费方对于令牌（**Request Token** 和 **Access Token**）的拥有关系。

图 2. OAuth 授权流程（摘自 OAuth 规范）



对于图 2 具体每一执行步骤，解释如下：

- 消费方向 OAuth 服务提供方请求未授权的 Request Token。
- OAuth 服务提供方在验证了消费方的合法请求后，向其颁发未经用户授权的 Request Token 及其相对应的 Token Secret。
- 消费方使用得到的 Request Token，通过 URL 引导用户到服务提供方那里，这一步应该是浏览器的行为。接下来，用户可以通过输入在服务提供方的用户名 / 密码信息，授权该请求。一旦授权成功，转到下一步。
- 服务提供方通过 URL 引导用户重新回到消费方那里，这一步也是浏览器的行为。
- 在获得授权的 Request Token 后，消费方使用授权的 Request Token 从服务提供方那里换取 Access Token。
- OAuth 服务提供方同意消费方的请求，并向其颁发 Access Token 及其对应的 Token Secret。
- 消费方使用上一步返回的 Access Token 访问用户授权的资源。

总的来讲，在 OAuth 的技术体系里，服务提供方需要提供如下基本的功能：

- 第 1、实现三个 Service endpoints，即：提供用于获取未授权的 Request Token 服务地址，获取用户授权的 Request Token 服务地址，以及使用授权的 Request Token 换取 Access Token 的服务地址。
- 第 2、提供基于 Form 的用户认证，以便于用户可以登录服务提供方做出授权。
- 第 3、授权的管理，比如用户可以在任何时候撤销已经做出的授权。

而对于消费方而言，需要如下的基本功能：

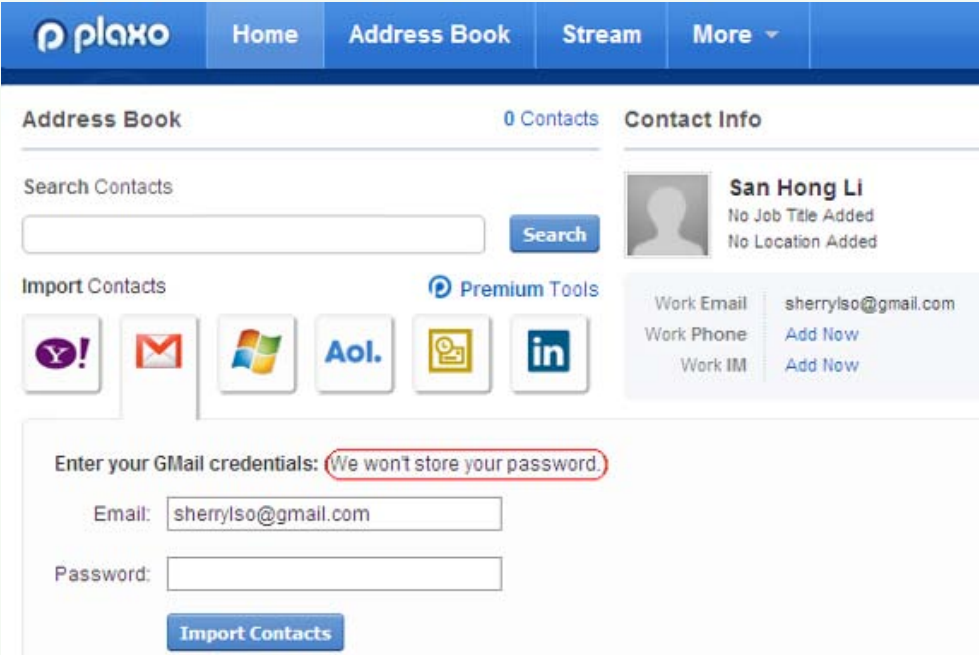
- 第 1、从服务提供方获取 Customer Key/Customer Secret。
- 第 2、提供与服务提供方之间基于 HTTP 的通信机制，以换取相关的令牌。

我们具体来看一个使用 OAuth 认证的例子。

在传统的网站应用中，如果您想在网站 A 导入网站 B 的联系人列表，需要在网站 A 输入您网站 B 的用户名、密码信息。例如，您登陆 Plaxo (<https://www.plaxo.com>)，一个联系人管理网站，当您想把 Gmail 的联系人列表导入到 Plaxo，您需要输入您的 Gmail 用户

基于 OAuth 安全协议的 Java 应用程序
名 / 密码，如图 3 所示：

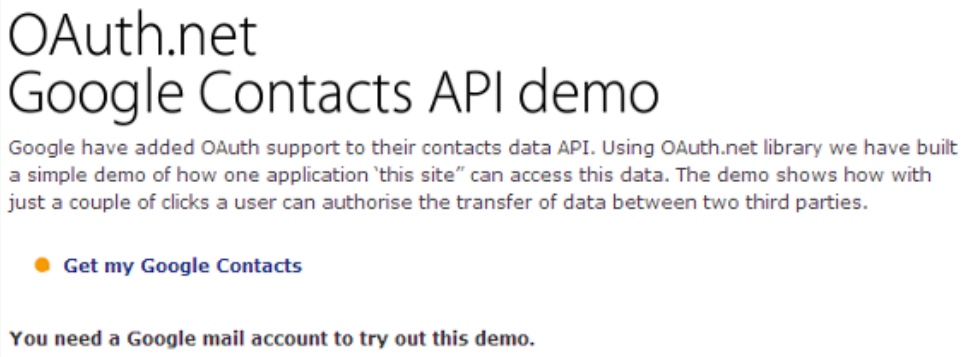
图 3. 在 Plaxo 获得 GMail 联系人



在这里，Plaxo 承诺不会保存您在 Gmail 的密码。

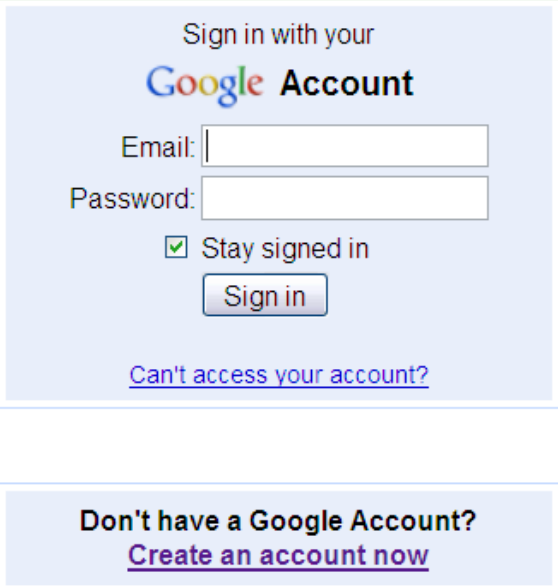
如果使用 OAuth 认证，情况是不同的，您不需要向网站 A（扮演 Consumer 角色）暴露您网站 B（扮演 Service Provider 角色）的用户名、密码信息。例如，您登录 <http://lab.madgex.com/oauth-net/googlecontacts/default.aspx> 网站，如图 4 所示：

图 4. 在 lab.madgex.com 获得 GMail 联系人



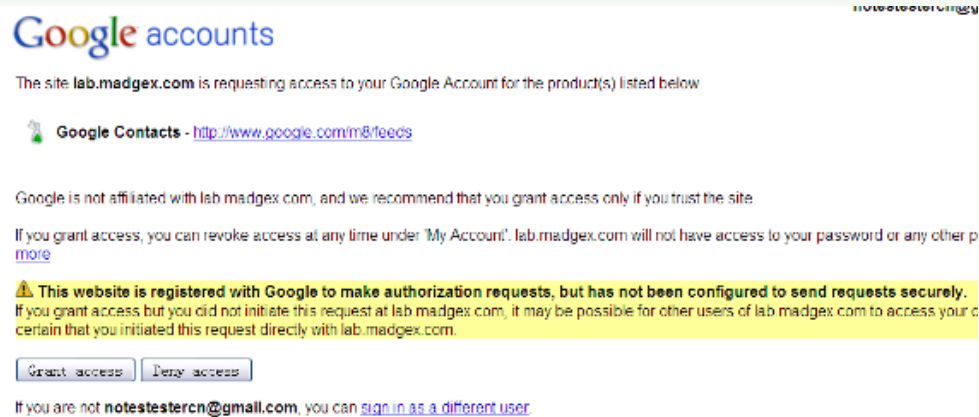
点击“Get my Google Contacts”，浏览器将会重定向到 Google，引导您登录 Google，如图 5 所示：

图 5. 登录 Google



登录成功后，将会看到图 6 的信息：

图 6. Google 对 lab.madgex.com 网站授权



在您登录 Google，点击“Grant access”，授权 lab.madgex.com 后，lab.madgex.com 就能获得您在 Google 的联系人列表。

在上面的的例子中，网站 lab.madgex.com 扮演着 Consumer 的角色，而 Google 是 Service Provider，lab.madgex.com 使用基于 OAuth 的认证方式从 Google 获得联系人列表。

下一节，本文会给出一个消费方实现的例子，通过 OAuth 机制请求 Google Service Provider 的 OAuth Access Token，并使用该 Access Token 访问用户的在 Google 上的日历信息 (Calendar)。

示例

准备工作

作为消费方，首先需要访问 <https://www.google.com/accounts/ManageDomains>，从 Google 那里获得标志我们身份的 Customer Key 及其 Customer Secret。另外，您可以生成自己的自签名 X509 数字证书，并且把证书上传给 Google，Google 将会使用证书的公钥来验证任何来自您的请求。

具体的操作步骤，请读者参考 Google 的说明文档：<http://code.google.com/apis/gdata/articles/oauth.html>。

在您完成这些工作，您将会得到 OAuth Consumer Key 及其 OAuth Consumer Secret，用于我们下面的开发工作。

如何获得 OAuth Access Token

以下的代码是基于 Google Code 上提供的 OAuth Java 库进行开发的，读者可以从 <http://oauth.googlecode.com/svn/code/java/core/> 下载获得。

- 指定 Request Token URL，User Authorization URL，以及 Access Token URL，构造 OAuthServiceProvider 对象：

```
OAuthServiceProv ider serviceProv ider = new OAuthServiceProv ider(
    "https://www.google.com/accounts/OAuthGetRequestToken",
    "https://www.google.com/accounts/OAuthAuthorizeToken",
    "https://www.google.com/accounts/OAuthGetAccessToken");
```

- 指定 Customer Key，Customer Secret 以及 OAuthServiceProvider，构造 OAuthConsumer 对象：

```
OAuthConsumer oauthConsumer = new OAuthConsumer(null
    , "www.example.com"
    , "hIsGkM+T4+90fKNesTtJq8Gs"
    , serviceProv ider);
```

- 为 OAuthConsumer 指定签名方法，以及提供您自签名 X509 数字证书的 private key。

```
oauthConsumer.setProperty(OAuth.OAUTH_SIGNATURE_METHOD, OAuth.RSA_SHA1);
oauthConsumer.setProperty(RSA_SHA1.PRIVATE_KEY, privateKey);
```

- 由 OAuthConsumer 对象生成相应的 OAuthAccessor 对象：

```
accessor = new OAuthAccessor(consumer);
```

- 指定您想要访问的 Google 服务，在这里我们使用的是 Calendar 服务：

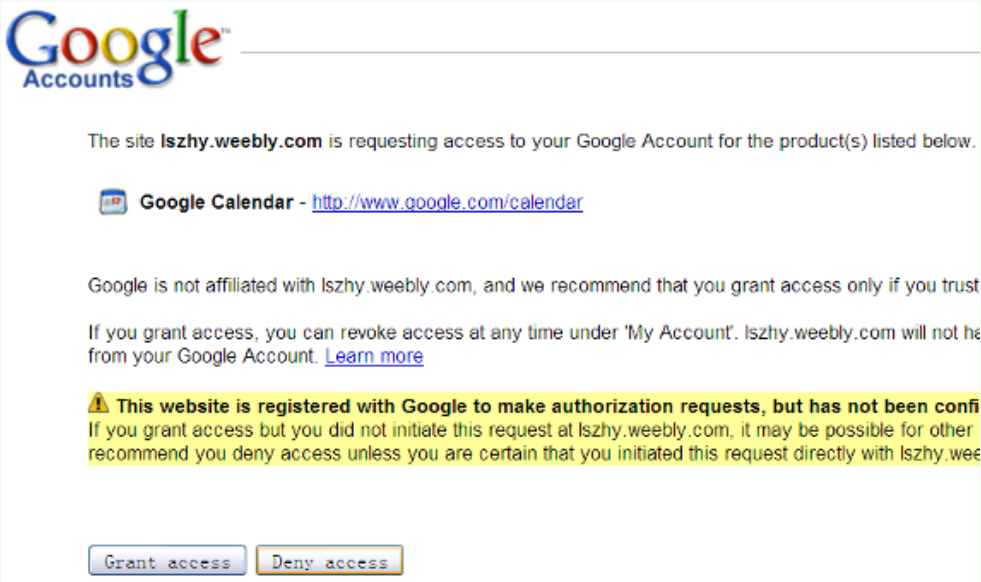
```
Collection<? extends Map.Entry> parameters
    = OAuth.newList("scope", "http://www.google.com/calendar/feeds/");
```

- 通过 OAuthClient 获得 Request Token:

```
OAuthMessage response = getOAuthClient().getRequestTokenResponse(
    accessor, null, parameters);
```

使用 Request Token，将用户重定向到授权页面，如图 7 所示：

图 7. OAuth User Authorization

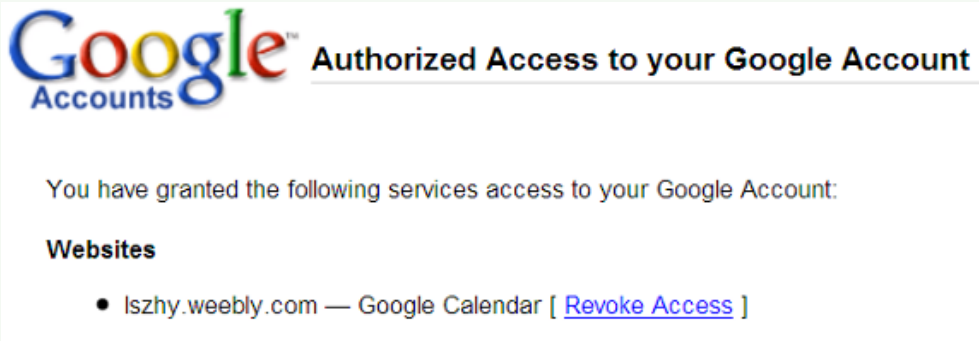


- 当用户点击“Grant access”按钮，完成授权后，再次通过 OAuthClient 获得 Access Token：

```
oauthClient.getAccessToken(accessor, null, null);
```

在上述步骤成功完成后，Access Token 将保存在 accessor 对象的 accessToken 成员变量里。查看您的 Google Account 安全管理页面，可以看到您授权的所有消费方，如图 8 所示。

图 8. Authorized OAuth Access to your Google Account



使用 OAuth Access Token 访问 Google 服务

接下来，我们使用上一节获得的 Access Token 设置 Google Service 的 OAuth 认证参数，然后从 Google Service 获取该用户的 Calendar 信息：

```
OAuthParameters para = new OAuthParameters();
para.setOAuthConsumerKey("www.example.com");
para.setOAuthToken(accessToken);
googleService.setOAuthCredentials(para, signer);
```


清单 1 是完整的示例代码，供读者参考。

清单 1. 基于 OAuth 认证的 Google Service 消费方实现

```
import java.util.Collection;
import java.util.Map;
import net.oauth.OAuth;
import net.oauth.OAuthAccessor;
import net.oauth.OAuthConsumer;
import net.oauth.client.OAuthClient;

public class DesktopClient {
    private final OAuthAccessor accessor;
    private OAuthClient oauthClient = null;
    public DesktopClient(OAuthConsumer consumer) {
        accessor = new OAuthAccessor(consumer);
    }

    public OAuthClient getOAuthClient() {
        return oauthClient;
    }

    public void setOAuthClient(OAuthClient client) {
        this.oauthClient = client;
    }

    //get the OAuth access token.
    public String getAccessToken(String httpMethod,
        Collection<? extends Map.Entry> parameters) throws Exception {
        getOAuthClient().getRequestTokenResponse(accessor, null, parameters);

        String authorizationURL = OAuth.addParameters(
            accessor.consumer.getServiceProvider.userAuthorizationURL,
            OAuth.OAUTH_TOKEN, accessor.requestToken);

        //Launch the browser and redirects user to authorization URL
        Runtime.getRuntime().exec("rundll32 url.dll,FileProtocolHandler "
            + authorizationURL);

        //wait for user's authorization
        System.out.println("Please authorize your OAuth request token. "
            + "Once that is complete, press any key to continue...");
        System.in.read();
        oauthClient.getAccessToken(accessor, null, null);
        return accessor.accessToken;
    }
}

import java.net.URL;
import java.security.KeyFactory;
import java.security.PrivateKey;
import java.security.spec.EncodedKeySpec;
import java.security.spec.PKCS8EncodedKeySpec;
import java.util.Collection;
import java.util.Map;
import com.google.gdata.client.GoogleService;
import com.google.gdata.client.authn.oauth.OAuthParameters;
import com.google.gdata.client.authn.oauth.OAuthRsaSha1Signer;
import com.google.gdata.client.authn.oauth.OAuthSigner;
import com.google.gdata.data.BaseEntry;
import com.google.gdata.data.BaseFeed;
import com.google.gdata.data.Feed;
```



```
import net.oauth.OAuth;
import net.oauth.OAuthConsumer;
import net.oauth.OAuthMessage;
import net.oauth.OAuthServiceProv ider;
import net.oauth.client.OAuthClient;
import net.oauth.client.httpclient4.HttpClient4;
import net.oauth.example.desktop.MyGoogleService;
import net.oauth.signature.OAuthSignatureMethod;
import net.oauth.signature.RSA_SHA1;

public class GoogleOAuthExample {
    //Note, use the private key of your self-signed X509 certificate.
    private static final String PRIVATE_KEY = "XXXXXXX";

    public static void main(String[] args) throws Exception {
        KeyFactory fac = KeyFactory.getInstance("RSA");
        //PRIVATE_KEY is the private key of your self-signed X509 certificate.
        EncodedKeySpec privateKeySpec = new PKCS8EncodedKeySpec(
            OAuthSignatureMethod.decodeBase64(PRIVATE_KEY));
        fac = KeyFactory.getInstance("RSA");
        PrivateKey privateKey = fac.generatePrivate(privateKeySpec);
        OAuthServiceProvider serviceProvider = new OAuthServiceProvider(
            //used for obtaining a request token
            //"https://www.google.com/accounts/OAuthGetRequestToken",
            //used for authorizing the request token
            "https://www.google.com/accounts/OAuthAuthorizeToken",
            //used for upgrading to an access token
            "https://www.google.com/accounts/OAuthGetAccessToken");

        OAuthConsumer oauthConsumer = new OAuthConsumer(null
            , "lszhy.weebly.com" //consumer key
            , "hIsGnM+T4+86fKNesUtJq7Gs" //consumer secret
            , serviceProvider);

        oauthConsumer.setProperty(OAuth.OAUTH_SIGNATURE_METHOD, OAuth.RSA_SHA1);
        oauthConsumer.setProperty(RSA_SHA1.PRIVATE_KEY, privateKey);

        DesktopClient client = new DesktopClient(oauthConsumer);
        client.setOAuthClient(new OAuthClient(new HttpClient4()));

        Collection<? extends Map.Entry> parameters =
            OAuth.newList("scope", "http://www.google.com/calendar/feeds/");

        String accessToken = client.getAccessToken(OAuthMessage.GET, parameters);

        //Make an OAuth authorized request to Google

        // Initialize the variables needed to make the request
        URL feedUrl = new URL(
            "http://www.google.com/calendar/feeds/default/allcalendars/full");

        System.out.println("Sending request to " + feedUrl.toString());
        System.out.println();

        GoogleService googleService = new GoogleService("cl", "oauth-sample-app");

        OAuthSigner signer = new OAuthRsaSha1Signer(MyGoogleService.PRIVATE_KEY);

        // Set the OAuth credentials which were obtained from the step above.
        OAuthParameters para = new OAuthParameters();
        para.setOAuthConsumerKey("lszhy.weebly.com");
        para.setOAuthToken(accessToken);
        googleService.setOAuthCredentials(para, signer);
    }
}
```

```
// Make the request to Google
BaseFeed resultFeed = googleService.getFeed(feedUrl, Feed.class);
System.out.println("Response Data: ");
System.out.println("=====");

System.out.println("|TITLE: " + resultFeed.getTitle().getPlainText());
if (resultFeed.getEntries().size() == 0) {
    System.out.println("\tNo entries found.");
} else {
    for (int i = 0; i < resultFeed.getEntries().size(); i++) {
        BaseEntry entry = (BaseEntry) resultFeed.getEntries().get(i);
        System.out.println("\t" + (i + 1) + ": "
            + entry.getTitle().getPlainText());
    }
}
System.out.println("=====");
}
```



小结

OAuth 协议作为一种开放的，基于用户登录的授权认证方式，目前互联网很多 Open API 都对 OAuth 提供了支持，这包括 Google, Yahoo, Twitter 等。本文以 Google 为例子，介绍了 Java 桌面程序如何开发 OAuth 认证应用。在开发桌面应用访问 Web 资源这样一类程序时，一般通行的步骤是：使用 OAuth 做认证，然后使用获得的 OAuth Access Token，通过 REST API 访问用户在服务提供方的资源。

事实上，目前 OAuth 正通过许多实现（包括针对 Java、C#、Objective-C、Perl、PHP 及 Ruby 语言的实现）获得巨大的动力。大部分实现都由 OAuth 项目维护并放在 Google 代码库 (<http://oauth.googlecode.com/svn/>) 上。开发者可以利用这些 OAuth 类库编写自己需要的 OAuth 应用。

参考资料

学习

- “[Using OAuth with the Google Data APIs](#)”：查看 Google 对 OAuth 的支持文档。
- “[OAuth.net](#)”：查看 OAuth 开放协议的官方网站。
- “[Google sample: OAuth Playground](#)”：查看 Google 上 OAuth 协议的示例应用。
- [技术书店](#)：浏览关于这些和其他技术主题的图书。
- [developerWorks Java 技术专区](#)：数百篇关于 Java 编程各个方面的文章。

讨论

- 加入 [developerWorks 社区](#)。
- 查看 [developerWorks 博客](#) 的最新信息。

关于作者

李三红，任职于 IBM 中国软件开发中心 Lotus 产品部门，负责 Lotus Notes 安全相关研发工作。在这之前，他一直从事网络应用开发相关工作。他感兴趣的技术领域包括：分布式对象计算、网络应用、OSGi、协作计算、Java 安全等方面。

----- 此本文来自 --[CevenCheng](#)-- 的文件夹 --[[客户端接口](#)].
上一篇: [OAuth协议简介](#)
下一篇: [OAuth授权的Java实现详解](#)

相关文章

- [同步令牌防止用户重复提交](#) 2005-11-15 [duduwolf](#)
- [SSO技术总结](#) 2006-02-20 [Joshua](#)
- [lucene的中文分词器使用指南](#) 2007-07-03 [iversion](#)
- [Jbpm流程图显示](#) 2009-09-16 [luyucs](#)
- [Lucene 中文分词](#) 2007-12-29 [jiang0723](#)
- [应用整合中SSO的技术实现](#) 2006-02-20 [Joshua](#)
- [SSO技术总结（相关文章）](#) 2007-07-15 [hendryn](#)
- [WebLogic平台的Web SSO（SAML）解决方案](#) 2006-02-20 [Joshua](#)

发表评论: [查看更多文章>>](#)

发表评论

[服务条款](#) | [设360doc为首页](#) | [留言交流](#) | [联系我们](#) | [友情链接](#)

北京六智信息技术有限公司 Copyright © 2005-2011 360doc.com , All Rights Reserved
[京ICP证090625号](#) 京ICP备05038915号 京网文【2010】0370-002号 京公网安备110105001118号

