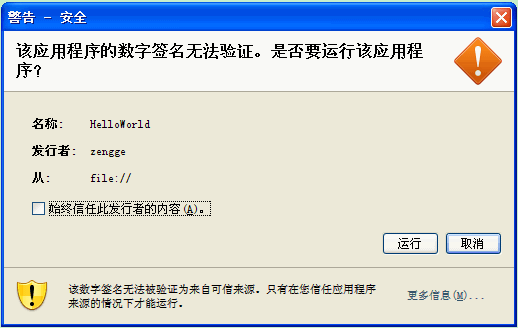
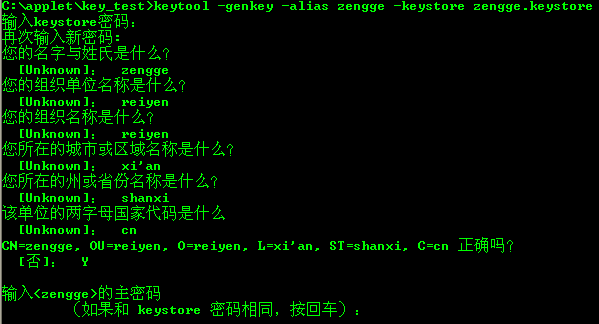
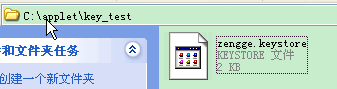
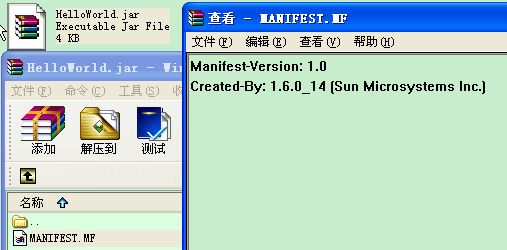
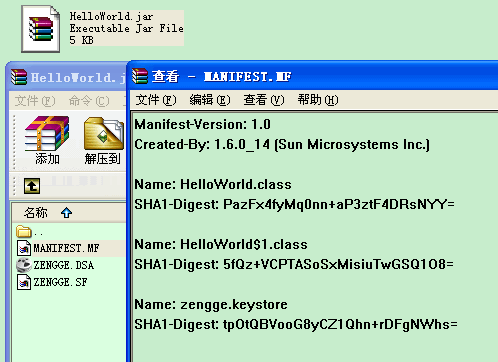
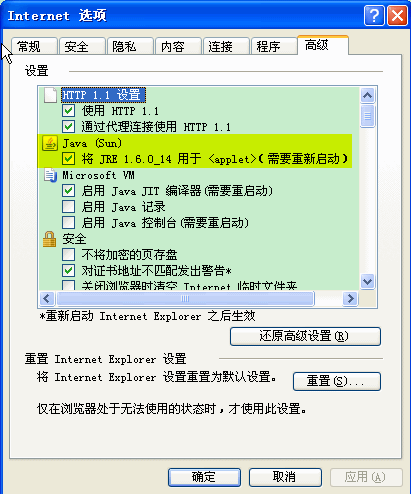
[**手把手教你做基于web的文件扫描并上传**](http://bbs.51cto.com/thread-817032-1.html) [[复制链接]](javascript:setcopy('http://bbs.51cto.com/thread-817032-1.html','%E5%A4%8D%E5%88%B6%E5%B8%96%E5%AD%90%E9%93%BE%E6%8E%A5%E5%88%B0%E5%89%AA%E8%B4%B4%E6%9D%BF');)

发表于 2011-3-14 18:40 | 来自 [51CTO网页](http://bbs.51cto.com/)

[[只看他]](http://bbs.51cto.com/viewthread.php?tid=817032&page=1&authorid=940717) **楼主**

最近碰到的问题，客户端调用本地的扫描仪，将扫描的文件上传。  
使用到的技术：applet,twain,HttpClient  
当然也碰到很多问题，因为在这周之前我都不知道什么是applet  
一.Applet操作本地资源  
基于安全方面的原因，applet是不允许操作本地资源的。但是java提供了相应的为jar包签名的机制来提升applet的权限。相信很多人都碰到过这种对话框：  
  
让用户来决定是否给applet提升权限，如果用户信任这个资源，applet将能操作本地资源。  
  
1.为applet依赖的jar包签名  
java提供了两个工具；  
keytool用于生成存放key的库  
jarsigner用于为jar进行签名  
  
首先建立一个keystore(这是在当前路径上操作):  
keytool -genkey -alias zengge -keystore zengge.keystore  
keytool -genkey表示建库  
-alias zengge是为要建立的key取一个别名  
-keystore zengge.keystore是建立一个名字叫zengge.keystore的key库，key就存在里面.  
如下图：  
  
  
这样一个keystore就建立好了,在当前目录下就多出了一个zengge.keystore的文件  
  
  
  
有了keystore之后jarsigner就可以利用存放在keystore中的key来为jar签名  
  
  
这里有两个类，先打包  
  
  
  
包里面的META-INFO里面只有一个文件，且内容为上图  
  
接下来为jar包签名：  
jarsigner -keystore zengge.keystore HelloWorld.jar zengge  
zengge.keystore是keystore的路径/名字，这里是相对路径  
HelloWorld.jar是要签名的jar包  
zengge是存放在keystore中的密钥的别名  
  
  
密码为建立keystore时的密码  
  
再来看一下签过名的jar里面的情况有什么变化：  
  
  
可以看到META-INFO里面现在是三个文件，.DSAG与.SF我想肯定是用来加解密用的，大家注意现在的MANIFEST.MF,  
可以看出为每个类添加了一个SHA签名，用它来保证，这个jar里面的内容不会被其它人修改，用户可以相信这个jar。  
  
2.将applet嵌入html  
applet是通过浏览器来运行了，可能你会问，java的东西浏览器怎么能运行呢，难道客户端也要安装java？  
实际上applet是通过嵌在浏览器中的jvm在运行，但是这个jvm是从那里来的呢？  
对于IE,大家可以看一下，下图中的java如果选中，就表示会调用本地的jre来运行applet。  
  
  
但是绝大部分情况下，客户端是不会安装java的啊，还有firefox下面没有类似的这种选项。那么要如何来解决这种问题呢？  
早期的applet都是用applet标签来嵌入html的（当然现在也可以）,例如：  
<APPLET CODE = "HelloWorld" archive ="HelloWorld.jar" JAVA\_CODEBASE = "." WIDTH = "320" HEIGHT = "240" NAME = "HelloWorld"></APPLET>  
这里的CODE表示类名（类名后可以加上.class)，archvie表示类所在的jar包，如果你有多个jar包，可以全加在archive里面，用，号分开(archive="a,jar,b.jar,c.jar"，当然这些jar要签名的还得签名)。  
如果用这种标签，在IE下，如果没选中用本地jre运行applet的话，是运行不了的,没安装插件的firefox也是不能运行的。  
  
面对这种情况，升级版的标签出现了,java提供了一个工具名字叫HTMLconverter,通过它，能将html中的applet标签转换成标准的标签，如下

**Html代码  
[url=]http://www.javaeye.com/images/icon_star.png[/url]**

1. **<object**
2. classid = "clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
3. codebase = "http://java.sun.com/update/1.6.0/jinstall-6u14-windows-i586.cab#Version=6,0,0,8"
4. WIDTH = "320"  
   HEIGHT = "240"  
   NAME = "HelloWorld"  
   **>**
5. **<PARAM**  
   NAME = CODE  
   VALUE = "HelloWorld"  
   **>**
6. **<PARAM**  
   NAME = CODEBASE  
   VALUE = "."  
   **>**
7. **<PARAM**  
   NAME = ARCHIVE  
   VALUE = "applet\_test.jar"  
   **>**
8. **<PARAM**  
   NAMENAME = NAME VALUE = "HelloWorld"  
   **>**
9. **<param**  
   name = "type"  
   value = "application/x-java-applet;version=1.6"**>**
10. **<param**  
    name = "scriptable"  
    value = "false"**>**
11. //上面是针对IE
12. //下面的embed是针对firefox
13. **<comment>**
14. **<embed**
15. type = "application/x-java-applet;version=1.6" \
16. CODE = "HelloWorld" \
17. JAVA\_CODEBASE = "." \
18. ARCHIVE = "applet\_test.jar" \
19. NAME = "HelloWorld" \
20. WIDTH = "320" \
21. HEIGHT = "240"
22. scriptable = false
23. pluginspage = "http://java.sun.com/products/plugin/index.html#download"**>**
24. **<noembed>**
25. **</noembed>**
26. **</embed>**
27. **</comment>**
28. **</object>**
29. <!--
30. **<APPLET**  
    CODE = "HelloWorld"  
    JAVA\_CODEBASE = "."  
    WIDTH = "320"  
    HEIGHT = "240"  
    NAME = "HelloWorld"**>**
31. **</APPLET>**
32. --**>**
33. <!--"END\_CONVERTED\_APPLET"-->

其中Object部分是针对IE的，embed是针对firefox的.classid与codebase都是表示相应的plugin的下载地址，如果codebase的版本高于classid将下载codebase版本的plugin。这样不管本地有没有安装java，applet都能正常运行了。  
  
现在，在applet里面己经能调用本地的资源了，那么如何驱动扫描仪呢?  
  
二.applet调用TWain驱动本地扫描仪  
TWain是一个标准，用于获取扫描仪等设备的信息，它有很多实现（基本都是收费的），这里选择的是一个开源twain产品，mmsc twain(官网 <http://www.mms-computing.co.uk/>例子很丰富）。  
只要本地安装有扫描仪驱动，twain就能找到并运行且获取到扫描的数据。  
它里面有个Scanner类，Scanner scanner = Scanner.getDevice()能获取到相应的设备。然后为scanner添加一个监听器，ScannerListener,它里面有个方法public void update(ScannerIOMetadata.Type type, ScannerIOMetadata metadata){}，第二个参数即为扫描得到的数据，而且这个方法是在扫描述的状态发生变变就会解发。具体可以看一些mmsc里面的例子。  
  
三.applet与服务器通信  
得到了扫描的数据，得把它上传到服务器。我这里用的是HttpClient(需要的jar包commons-codec-1.4.jar,commons-httpclient-3.0.jar,commons-io-1.4.jar,commons-logging-1.0.2.jar)。在使用时最好对这几个jar包都签名(我没有测试这种方式，我是将这几个jar全给解压了，最后连同我的类一起打成了一个jar，最后签名),上传代码位于update方法内即可。  
因为扫描仪有多种状态，所以要进行判断，那次才是拿到了扫描数据。

**Java代码  
[url=]http://www.javaeye.com/images/icon_star.png[/url]**

1. **public**  
   **void** update(ScannerIOMetadata.Type type, ScannerIOMetadata metadata){
2. **if**(type.equals(ScannerIOMetadata.ACQUIRED)){
3. BufferedImage image=metadata.getImage();
4. System.out.println("Have an image now!");
5. HttpClient httpClient = **new** HttpClient();
6. MultipartPostMethod mpm = **new** MultipartPostMethod("http://localhost:8086/ReiyenDMS/TestUploadServlet");
7. // MultipartPostMethod mpm = new MultipartPostMethod("http://localhost:8086/applet\_study/servlet/AppletServlet");
8. File file = **new** File("c:/upload/abc"+index+".jpg");
9. **try**{
10. ImageIO.write(image, "jpg", file);
11. index++;
12. mpm.addParameter("aFile", "haha.pdf", file);
13. httpClient.executeMethod(mpm);
14. }**catch**(Exception e){
15. e.printStackTrace();
16. }
17. }**else**  
    **if**(type.equals(ScannerIOMetadata.NEGOTIATE)){
18. ScannerDevice device=metadata.getDevice();
19. /\*
20. try{
21. device.setResolution(100);
22. // device.setRegionOfInterest(0.0,0.0,40.0,50.0); // top-left corner 40x50 mm
23. device.setRegionOfInterest(0,0,400,500); // top-left corner 400x500 pixels
24. device.setShowUserInterface(false);
25. device.setShowProgressBar(false);
26. }catch(Exception e){
27. e.printStackTrace();
28. }
29. \*/
30. }**else**  
    **if**(type.equals(ScannerIOMetadata.STATECHANGE)){
31. System.err.println(metadata.getStateStr());
32. }**else**  
    **if**(type.equals(ScannerIOMetadata.EXCEPTION)){
33. metadata.getException().printStackTrace();
34. }
35. }

完工  
中间碰到最多的问题就是，applet签名及applet布署，开始老以为applet根平时的类布署是一样的，后来发现就应该把它当成一个独立的应用来对等。  
  
中间可能有错，有问题请在家指出，谢谢  
  
效果图(点击中间的acquire就能扫描了)：  
  
  
  
  
  
有童鞋反映提供的twain站点不能用了，特加上一个twain的源码,详见符件twian.rar

**http://bbs.51cto.com/images/attachicons/rar.gif** [**twain.rar**](http://bbs.51cto.com/attachment.php?aid=194963&k=4312168a465c7b17f636fffc46192782&t=1421254973) ***(4.26 MB)***

2011-3-14 18:40, 下载次数: 414