

保密级别： 公开

报告版本： v2.0

页数： 26

网页轻阅读系统

部署集成指南

编制人： 李欣
审核人： 刘丹
批准人：
日期：

修改历史记录

序号	版本号	内容	编制\日期	审核\日期	批准\日期
1	0.1	初稿			
2	V1.0				

目录

1、文档说明.....	4
2、安装部署.....	5
2.1 部署准备及环境要求.....	5
2.1.1 获取部署文件.....	5
2.2 JDK 版本.....	5
2.2.1 Windows 环境.....	5
2.2.2 Linux 环境.....	5
2.3 中间件.....	6
2.3.1 中间件部署.....	6
2.3.2 修改中间件端口.....	6
2.4 操作系统.....	7
2.5 硬件要求.....	7
2.5.1 硬盘.....	7
2.5.2 内存.....	7
2.5.3 CPU.....	7
2.6 快速部署和访问.....	8
2.6.1 win 环境.....	8
2.6.2 Linux 环境.....	8
2.6.3 专用机环境.....	8
3、显示及调用.....	9
3.1 显示.....	9
3.1.1 铺满整个页面.....	9
3.1.1 页面引用显示.....	9
3.2 调用.....	10
3.2.1 通过文档标识访问.....	10
3.2.2 通过 URL 访问.....	10
4、接口实现.....	11
4.1 引用 jar 包.....	11
4.2 实现自定义接口.....	11
4.2.1 使用说明.....	12
4.2.2 接口描述.....	12
4.3 预览 OFD 文件.....	13
方法描述.....	13
方法定义.....	13
方法示例.....	14
方法约束.....	15
4.4 保存 OFD 文件.....	15
方法描述.....	15
方法定义.....	15
方法示例.....	16
方法约束.....	17
4.5 不同用户访问权限.....	17
方法描述.....	17

方法定义.....	17
方法示例.....	17
方法约束.....	18
4.6 添加预览水印.....	18
方法描述.....	18
方法定义.....	18
方法示例.....	19
方法约束.....	19
4.7 将签名导出为图片.....	20
方法描述.....	20
方法定义.....	20
方法示例.....	20
方法约束.....	21
5、接口部署.....	21
5.1 导出接口.....	21
5.2 修改配置.....	21
5.3 重启轻阅读服务.....	22
6、服务注册.....	22
7、常用配置.....	22
7.1 配置默认访问 OFD 文件.....	22
7.2 开启用户自定义权限.....	23
7.3 隐藏页面工具栏.....	23
7.4 默认访问文件路径.....	23
7.5 自动清理缓存.....	23
8、特殊配置.....	23
8.1 自定义工具栏按钮.....	23
8.2 设置默认缩放模式.....	25
8.3 设置默认视图模式.....	26
8.4 设置预加载.....	26
9、解决方案.....	26

1、文档说明

本指南仅用于指导数科网页轻阅服务的部署及集成，如有超出本文档范围的技术问题请联系数科技术人员。

2、安装部署

2.1 部署准备及环境要求

2.1.1 获取部署文件

- 从数科获取网页轻阅读部署包，将 war 包的名字改成 web-reader.war

2.2 JDK 版本

- 1) 此平台需适配 1.7.50 及以上 JDK 版本。
- 2) 如与其它服务共同部署并且服务器上为低版本的 jdk 时，可指定单独的 JDK 环境。

2.2.1 Windows 环境

在 tomcat/bin/catalina.bat 文件和 tomcat/bin/setclasspath.bat 文件开头添加以下两句。

```
set JAVA_HOME=D:/java/jdk1.7.0_18
set JRE_HOME=D:/java/jdk1.7.0_18/jre
```

2.2.2 Linux 环境

在 tomcat/bin/catalina.sh 文件和 tomcat/bin/setclasspath.sh 文件开头添加以下两句：

```
export JAVA_HOME=/usr/local/java/jdk1.7.0_18
export JRE_HOME=/usr/local/java/jdk1.7.0_18/jre
```

2.3 中间件

- 中间件可适用常见中间件，一般由用户方提供，如无特殊要求，可使用 Tomcat7.0 及以上版本。
- 其它中间件金蝶、东方通，中创等

2.3.1 中间件部署

- 中间件部署，此处以中创为例，访问管理工具 <http://ip:8060>，点击应用管理-部署-选择文件夹部署或者 war 包部署，在弹出的框里选择需要部署的应用，选到应用根路径（即应用名）即可。



2.3.2 修改中间件端口

- 浏览器访问 <http://ip:8060> 管理工具地址，进入管理工具，点击配置管理-server-config-网络配置-网络监听-http-listener-1，修改端口值，保存重启生效。



2.4 操作系统

数科可适配常见的系统环境。

- Windows 环境，winserver2008\winserver2012 等
- Linux 环境，Centos_x64\中标麒麟龙芯\银河麒麟飞腾等

注意：Linux 系统需要安装图形化界面

2.5 硬件要求

2.5.1 硬盘

- 如本机也作为 ofd 文件存储的服务器，建议至少当前文档存量加 10 年增量值大小
- 如本机只负责运行服务，不作存储用，建议 1T 大小。

2.5.2 内存

- 建议大于等于 16G

2.5.3 CPU

- 视用户的并发要求定，建议至少八核。

2.6 快速部署和访问

注意：此快速访问方式仅用于演示和测试，生产环境调用建议用接口调用方式。

2.6.1 win 环境

- 将 web-reader.war 放到 tomcat 的 webapps 文件夹下面。
- 进入 tomcat/bin 文件夹, 双击 startup.bat 文件, 启动轻阅读服务。
- 创建 D:/ofd 目录, 放入 1.ofd 示例文件。
- 手动修改如下配置项

tomcat\webapps\web-reader\WEB-INF\AIOCfg\service.ini 的
directory = D:/ofd,

- 打开浏览器, 访问 <http://ip:port/web-reader/reader?file=1.ofd>,
可看到轻阅读的界面并浏览 D:/ofd/1.ofd 文件。

注意: 不能改变 service.ini 编码格式, 默认为 UTF-8, 修改配置文件后需要重启轻阅读服务

2.6.2 Linux 环境

- 将 web-reader.war 放到 tomcat 的 webapps 文件夹下面。
- 进入 tomcat/bin 文件夹, 运行 ./startup.sh 脚本文件, 启动轻阅读服务。
- 创建 /opt/ofd 目录, 放入 1.ofd 示例文件。
- 手动修改如下配置项

tomcat\webapps\web-reader\WEB-INF\AIOCfg\service.ini 的
directory = /opt/ofd/

- 打开浏览器, 访问 <http://ip:port/web-reader/reader?file=1.ofd>,
可看到轻阅读的界面并浏览 /opt/ofd/1.ofd 文件。

注意: 不能改变 service.ini 编码格式, 默认为 UTF-8, 修改配置文件后需要重启轻阅读服务

2.6.3 专用机环境

- 首先将 web-reader.war 在普通机中 (对应专用机系统和架构) 发布后打成安装包, 如中标龙芯普通机打成 rpm 包, 在中标龙芯专用机以安装包

的方式安装后再进行部署。

- 安装完成在/opt 目录生成 suwellreader 文件夹，在中间件中以文件夹部署的方式进行部署。
- 创建/opt/ofd/ 目录，放入 1.ofd 示例文件。
- 打开浏览器，访问 <http://ip:port/web-reader/reader?file=1.ofd>，可看到轻阅读的界面并浏览/opt/ofd/1.ofd 文件。

注意:不能改变 service.ini 编码格式，默认为 UTF-8，修改配置文件后需要重启轻阅读服务

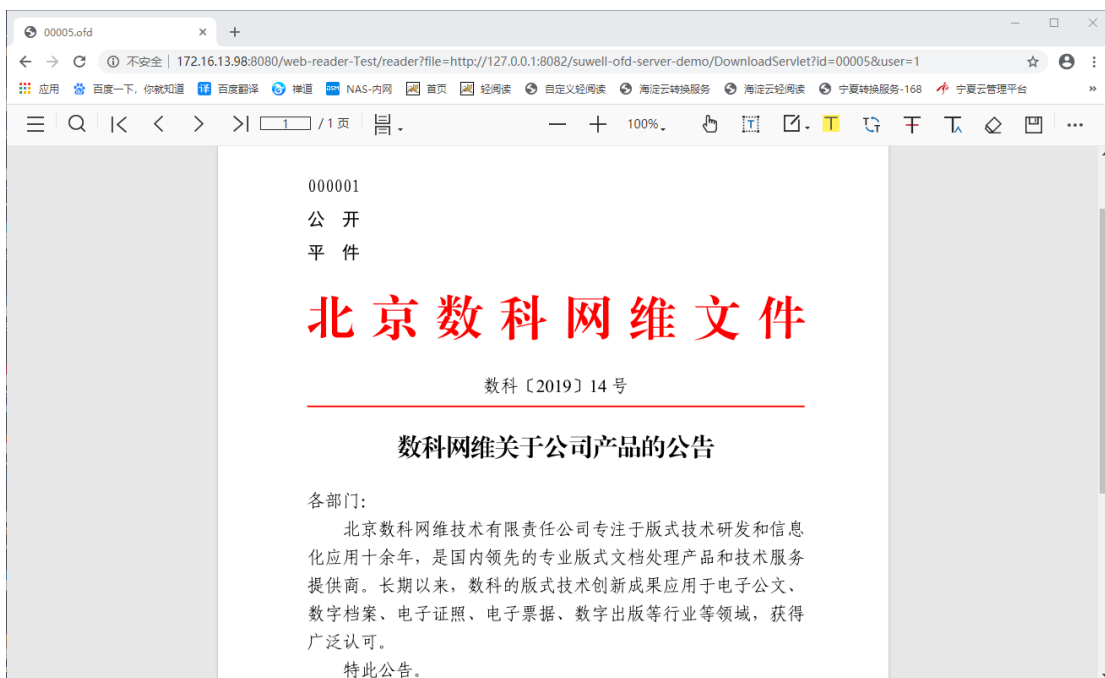
3、显示及调用

3.1 显示

3.1.1 铺满整个页面

轻阅读默认通过新页签显示，直接访问轻阅读地址则弹出新界面，并且铺满整个页面显示。

【调用示例】：<http://ip:port/web-reader/reader?file=id>



3.1.1 页面引用显示

如想在其它页面中引用显示，建议通过 iframe 框架实现，修改 width 和 height 适应页面。

【调用示例】：

```
<iframe src="http://127.0.0.1:8080/reader/reader?file=22.ofd"
style="position:relative;width:70%;height:80%"/>
```

3.2 调用

3.2.1 通过文档标识访问

【使用方法】：访问链接

<http://localhost:8080/convert-issuer/reader?file=fileID&user=userID>

【参数说明】：

file*：预打开的文档的唯一标识，此参数必需。file 可设置成文件名、文件 id、文件路径等。建议用 fileID。

user：预打开文档的用户 ID，此参数可选。在有权限控制需求时选需要传此参数。

【访问路径示例】：<http://localhost:8080/web-reader/reader?file=00001>

【返回值】：如文件能正确获取，直接回显文档内容。

【特别提示】：推荐此访问方式。但此调用方式必须配合接口实现（参考第四章接口实现）。

3.2.2 通过 URL 访问

【使用方法】：访问链接

<http://localhost:8080/convert-issuer/reader?file=url&user=0001>

【参数说明】：

file*：预打开的文档的 url 地址（此地址单独放在浏览器地址栏使用，可正确下载文件）。

user：预打开文档的用户 ID，此参数可选。在有权限控制需求时选需要传此参数。

【访问路径示例】：

<http://localhost:8080/web-reader/reader?file=http://172.16.14.43:8080/suwell-ofd-server-demo/DownloadServlet?id=00001>

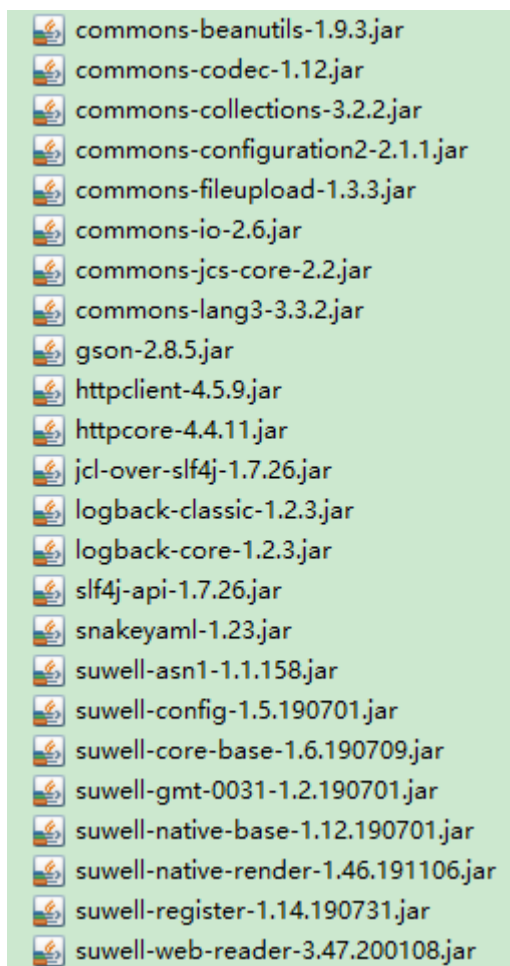
【返回值】：如文件能正确获取，直接回显文档内容。

【特别提示】：此预览方法由于无法回写保存，所以无法使用签批、文本框批注、修订等编辑功能。

4、接口实现

4.1 引用 jar 包

以下 jar 包为调用轻阅读依赖的基础 jar 包，这些 jar 包以包含基础在 test-demo 项目中，不同时期发放的 jar 包小版本号可能不同，请以实际收到的资料为准。



4.2 实现自定义接口

OFDResource 文件的资源接口，实现此接口可以满足用户的基本阅览和操作需求。

OFDStorage 数据回写接口，如用户对文件加了签批，需要实现此接口将签批内容回写到文件。

OFDMarkable 水印添加接口，实现此接口可满足用户添加水印的需求。

OFDPersist 数据持久化接口，如用户对文件加了签名，需要实现此接口将签名路径持久化保存到文件。

4.2.1 使用说明

导入项目 test-demo。根据具体业务需求自定义实现轻阅读接口。

4.2.2 接口描述

参见代码样例：test-demo

获取文档资源

实现类：com.suwell.reader.resource.OFDResource;

实现类说明：用于获取预览文件信息和文件权限设置

返回资源权限接口

```
Permission check(java.lang.String id, java.util.Map<java.lang.String,?> args)
```

获取资源数据接口

```
Result fetch(java.lang.String id, long version)
```

获取资源信息接口

```
Info info(java.lang.String id)
```

数据回写

实现类：com.suwell.reader.resource.OFDStorage;

实现类说明：用于手写签批场景。返回文件版本号与签批后的文件。

增加版本号

```
next(java.lang.String id, long current)
```

保存数据接口

```
save(java.lang.String id, long version, java.io.InputStream ofd,  
      java.util.Map<java.lang.String,?> data)
```

文件水印

实现类：com.suwell.reader.resource.OFDMarkable;

实现类说明：用于实现添加运行时、打印水印。

添加水印接口

```
mark (OFDMarkable.Type type, String id, Map<String, ?> data)
```

手机端签名

实现类：com.suwell.reader.resource.OFDPersist;

实现类说明：用于手机端添加签名和删除签名

查询数据接口

```
query(java.lang.String type,java.lang.String user,  
      java.util.Map<java.lang.String,?> extra)
```

添加数据接口

```
add(T data,java.lang.String user,java.util.Map<java.lang.String,?> extra)
```

删除数据接口

```
delete(java.lang.String type,java.lang.String id,java.lang.String user,  
       java.util.Map<java.lang.String,?> extra)
```

4.3 预览 OFD 文件

方法描述

实现该功能需引用轻阅读所有 jar 包，实现 fetch 和 info 方法获取资源文件与信息。将 OFD 渲染以 Html 形式展现。

方法定义

接口 1:获取资源数据接口

```
Result fetch(java.lang.String id, long version)
```

参数说明详见下表：

序号	参数名称	参数类型	参数含义	内容举例
1	id	字符型	资源标识	00001
2	version	长整型	资源版本	此处未使用，通过唯一的资源 ID 获取

返回值说明详见下表：

序号	返回类型	返回值含义
1	Result	资源数据

接口 2:获取资源信息接口

```
Info info(java.lang.String id)
```

参数说明详见下表：

序号	参数名称	参数类型	参数含义	内容举例
1	id	字符型	资源标识	00001

返回值说明详见下表：

序号	返回类型	返回值含义
1	info	资源信息

方法示例

```
//java 代码调用
//参考示例: com.suwell.reader.SimpleOFDResource
//功能说明: 预览 OFD 文件
public Result fetch(String arg0, long arg1) throws IOException {
    // 获取资源信息, 返回资源路径
    // System.out.println("-----获取到文件路径为-----");
    if (arg0.startsWith("http:") || arg0.startsWith("https")) {
        File dir = Util.baseDir(arg0);
        dir = new File(dir, String.valueOf(arg1));
        return new Result(new File(dir, DATA));
    } else {
        String path = this.getPath(arg0);
        // System.out.println("path" + path);
        // 获取到文件路径
        File f = new File(path);
        if (f.isDirectory()) {
            return null;
        }
        Result res = new Result(new File(path));
        // 返回一个流文件
        return res;
    }
}

public Info info(String arg0) {
    if (arg0.startsWith("http:") || arg0.startsWith("https")) {
        return this.download(arg0);
    } else {
        String path = this.getPath(arg0);
        // 获取资源信息, 返回资源路
        File file = new File(path);
        // System.out.println("-----文档信息
        -----");
        // System.out.println("文档名称" + file.getName());
        // System.out.println("文档大小" + file.length() / 1024 + "KB");
        // System.out.println("文档版本预设: " + (long) 1);
    }
}
```

```
        Info info = new Info(arg0, file.length(), (long) 1);
        return info;
    }
}
```

方法约束

无。

4.4 保存 OFD 文件

方法描述

实现该功能需引用轻阅读所有 jar 包，实现 save 方法将签批和批注保存至 OFD 文件，并可返回批注和签批信息。

方法定义

接口 1:下一个版本接口

```
long next(java.lang.String id, long current)
```

接口 2:写入数据接口

```
boolean save(java.lang.String id,
             long version,
             java.io.InputStream ofd,
             java.util.Map<java.lang.String,?> data)
             throws java.io.IOException
```

参数说明详见下表：

序号	参数名称	参数类型	参数含义	内容举例
1	id	字符型	资源标识	0001
2	version	长整型	资源版本	通过唯一的资源 ID 获取
3	ofd	InputStream 输入流	待保存的 ofd 文件	ofd

DEMO				
4	data	Map	批注或签批信息	<p>新增批注: {json=[{index=0, data={added=[{boundary={x=114. 03423,width=60.57143, y=71.7315,height=19.58454}, remark= 测试批注, id=347}], modified=null, deleted=null}}], user=1}</p> <p>修改批注: {json=[{index=0, data={added=null, modified=[{remark= 测试批注 99999,id=347}], deleted=null}}], user=1}</p> <p>新增签批 {json=[{index=0, data={added=[{boundary={x=75. 57884,width=86.301, y=26.981312, height=82.38795}, remark=null,id=347}], modified=null, deleted=null}}], user=1}</p>

返回值说明详见下表:

序号	返回类型	返回值含义
1	Result	资源数据

序号	参数名称	参数类型	参数含义	内容举例
1	id	字符型	资源标识	Info info = new Info(arg0, file.length(), (long) 1);

返回值说明详见下表:

序号	返回类型	返回值含义
1	info	资源信息

方法示例

```
//java 代码调用
//参考示例: com.suwell.reader.SimpleOFDResource
//功能说明: 保存 OFD 文件
// 文件下一个版本号, 签批
@Override
public long next(String arg0, long arg1) {
    // System.out.println("签批后文件显示版本号-----");
}
```



```
        return this.version(System.currentTimeMillis());
    }
    // 保存时要调用
    @Override
    public boolean save(String id, long version, InputStream ofd, Map<String, ?> data)
    throws IOException {
        File dir = new File(root, STORE + this.store(id));
        File file = new File(Util.mkdir(dir), this.name(version) + this.saveSuffix());
        FileUtils.copyInputStreamToFile(ofd, file);
        return true;
    }
}
```

方法约束

无。

4.5 不同用户访问权限

方法描述

实现该功能需引用轻阅读所有 jar 包，调用 Check 方法设置用户权限，不同用户是否有权限预览或编辑 CFD 文件。需要开启用户自定义权限，详情请看文档 5.2 开启权限功能。

方法定义

Permission check(String arg0, Map<String, ?> arg1)

参数说明详见下表：

序号	参数名称	参数类型	参数含义	内容举例
1	arg0	字符型	资源标识	00001
2	arg1	Map 型	访问文件所有信息（用户，版本，文件名，页数，分辨率等）。	user=01, _b=3.2.0, _d=00001, _=1584693503003, index=-1, version=-1

返回值说明详见下表：

序号	返回类型	返回值含义
1	permission	权限信息

方法示例

```
//java 代码调用
//参考示例: com.suwell.reader.SimpleOFDResource
//功能说明: 设置预览 OFD 文件中权限
public Permission check(String arg0, Map<String, ?> arg1) {
    System.out.println("-----进入权限控制-----");
    Permission per = new Permission(true, true, true);
    if (arg1.get(ARG_USER).equals("1")) {
        per = new Permission(true, true, true, true, true);// 显示, 复制, 下载, 打
        印, 回写
    } else if (arg1.get(ARG_USER).equals("2")) {
        per = new Permission(true, true, true, true, false);// 显示, 复制, 下载,
        打印, 回写
    } else if (arg1.get(ARG_USER).equals("3")) {
        per = new Permission(true, true, true, false, false);// 显示, 复制, 下载,
        打印, 回写
    } else if (arg1.get(ARG_USER).equals("4")) {
        per = new Permission(true, true, false, false, false);// 显示, 复制, 下载,
        打印, 回写
    } else if (arg1.get(ARG_USER).equals("5")) {
        per = new Permission(true, false, false, true, true);// 显示, 复制, 下载,
        打印, 回写
    }
    return per;
}
```

方法约束

无。

4. 6 添加预览水印

方法描述

实现该功能需引用轻阅读所有 jar 包, 调用 Mark 方法将运行时水印添加到 OFD 文件上。可以通过函数判断是否在打印时添加水印。

方法定义

```
mark (OFDMarkable.Type type, String id, Map<String, ?> data)
```

参数说明详见下表:

序号	参数名称	参数类型	参数含义	内容举例
1	text	字符型	水印信息	new TextMark("打印水印", new Font("SimSun", Font.PLAIN, 20),
2	font	Font 型	字体样式	
3	color	Boolean 型	颜色	

4	rule	Boolean 型	添加位置 (某一页的 某一个位 置)	
5	place	Boolean 型	添加规则 (某一页)	
6	angle	Int 型	旋转角度	
7	target	Boolean 型	水印渲染方 式	
8	isPrint	Boolean 型	是否打印	boolean isPrint = (boolean) data.get(OFDMarkable.DATA_ IS_PRINT);
9	type	OFDMarkabl e.Type	水印类型	此处未使用
10	Id	字符型	资源标识	此处未使用

返回值说明详见下表：

序号	返回类型	返回值含义
1	mark	水印信息

方法示例

```
//java 代码调用
//参考示例: com.suwell.reader.SimpleOFDResource
//功能说明: 设置预览 OFD 文件水印
public Mark mark(Type type, String id, Map<String, ?> data) {
    // 打印时判断是否添加水印,
    boolean isPrint = (boolean) data.get(OFDMarkable.DATA_IS_PRINT);
    if (isPrint) {
        // "打印水印" //水印内容
        // new Font("SimSun", Font.PLAIN, 20) //字体样式
        // randomColor 颜色
        // Rule.all 添加位置 (某一页的某一个位置)
        // Place.random 添加规则 (某一页)
        // 0 旋转角度
        // Target.image 水印渲染方式
        return new TextMark("打印水印", new Font("SimSun", Font.PLAIN, 20),
            randomColor, Rule.all, Place.topCenter, 0,
            Target.image);
    } else {
        return null;
    }
}
```

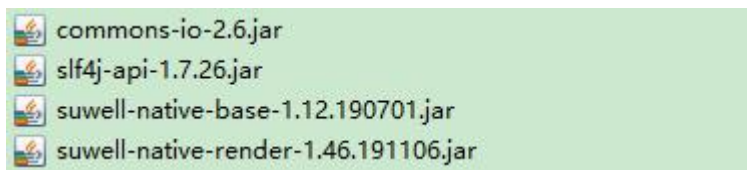
方法约束

无。

4.7 将签名导出为图片

方法描述

实现该功能需引用轻阅读部分 jar 包,调用 filter 接口获取签名信息。将 OFD 文件中的轻阅读签名导出为图片。此功能不需要导出为 jar 包进行接口部署,将依赖 jar 包拷贝到项目中可直接使用。



导出签名依赖 jar 包

方法定义

pain.filter(arg0)

参数说明详见下表:

序号	参数名称	参数类型	参数含义	内容举例
1	arg0	OFDDocument .OFDPage	签批页信息	pain.filter(page);

返回值说明详见下表:

序号	返回类型	返回值含义
1		

方法示例

```
//java 代码调用
//参考示例: com.suwell.reader.SigntoImage
//功能说明: OFD 文件签名导出为图片
// 轻阅读签批后的文件
File ofd = new File("D:/ofd/store4/1583302113582");
// 检测文件是否存在
System.err.println(ofd.exists());
try {
    // 打开文件
    OFDDocument doc = OFDDocument.open(ofd);
    // 获取签批页信息
    OFDDocument.OFDPage page = doc.getPageAt(0);
    Painter pain = new Painter();
    // 获取签名列表
```

```
List<OFDAnnotation> list = pain.filter(page);
int i = 0;
// 将签名转成图片
for (OFDAnnotation annotation : list) {
    BufferedImage bi = pain.toImage(annotation, 150);
    if (bi == null) {
        System.out.println("没有签名");
    } else {
        // 保存图片
        ImageIO.write(bi, "png", new File("D:/ofd/image/" + i++ +
".png"));
    }
}
doc.close();
System.out.println("生成结束");
} catch (RenderException e) {
    // TODO: handle exception
    e.printStackTrace();
} catch (IOException e) {
    // TODO: handle exception
    e.printStackTrace();
}
```

方法约束

无。

5、接口部署

5.1 导出接口

实现接口后生成的 test-demo.jar 放到 web-reader \WEB-INF \lib\内。导出时注意去掉除 lib 以及.project .classpath 项。

5.2 修改配置

实现接口后需要修改 web-reader\WEB-INF\AIOCfg\service.ini，将实现类的全路径赋值给 resource.provider 如
resource.provider=com.test1.SimpleOFDResource1

```

service.ini
1 #轻阅读配置
2 [webreader]
3
4 # 源数据提供者,默认为空,采用SPI方式自动加载.如果指定,则只使用它
5 resource_provider = com.suwell.resource.SimpleOFDResource
6
7 # 清空多久以前不活动的缓存,单位为分钟;小于等于0则不清除
8 result_cache_clear = 60
9 # 每个文档最多保留的版本;小于等于0则保留全部版本;只有版本号大于0才会触发此事件;注
10 result_cache_count = 3
11 # 是否开启native日志
12 sw_native_log = false
13 #临时文件生成路径
14 result_temp = /WEB-INF/temp
15 # png, jpg, bmp, webp
16 result_image_type = png
17 # 缓存文本是否压缩
18 result_compression = true
19 # 是否在图片中绘制注释
20 result_draw_annotation = true

```

注意:不能改变 service.ini 编码格式,默认为 UTF-8。

5.3 重启轻阅读服务

6、服务注册

- 在浏览器地址栏输入 <http://ip:port/web-reader/lic>, 进入注册页面。
- 将获取的机器码发送给数科技术支持人员。
- 点击选择文件将支持人员回复的注册文件上传。
- 点击【注册】, 注册成功会提示注册具体时间。

7、常用配置

配置文件路径: web-reader\WEB-INF\AIOfcg\service.ini

7.1 配置默认访问 OFD 文件

功能说明: 设置轻阅读访问文件方式

```

2 [webreader]
3
4 # 源数据提供者,默认为空,采用SPI方式自动加载.如果指定,则只使用它
5 #resource_provider = com.suwell.resource.SimpleOFDResource
6
7 # 清空多久以前不活动的缓存,单位为分钟;小于等于0则不清除
8 result_cache_clear = 60

```

默认禁用 (最前面加 ' #' 号), 实现接口后需要修改 web-reader\WEB-INF\AIOfcg\service.ini, 将实现类的全路径赋值给 resource.provider 如 resource.provider=com.test1.SimpleOFDResource

7.2 开启用户自定义权限

功能说明：设置用户是否可以自定义权限

```
# 是否开启用户权限检查
permission_check = false
```

备注：如果要想实现某些特定用户是否可以使用部分功能，例如打印，下载等。讲permission_check值改为true

7.3 隐藏页面工具栏

功能说明：隐藏轻阅读页面工具栏，true为显示

```
# 是否显示页面菜单
page_toolbar = false
```

7.4 默认访问文件路径

功能说明：设置轻阅读默认访问本地路径

```
directory = D:/ofd
```

7.5 自动清理缓存

功能说明：设置轻阅读默认清理缓存文件

```
# 清空多久以前不活动的缓存,单位为分钟;小于等于0则不清除
result_cache_clear = 60
```

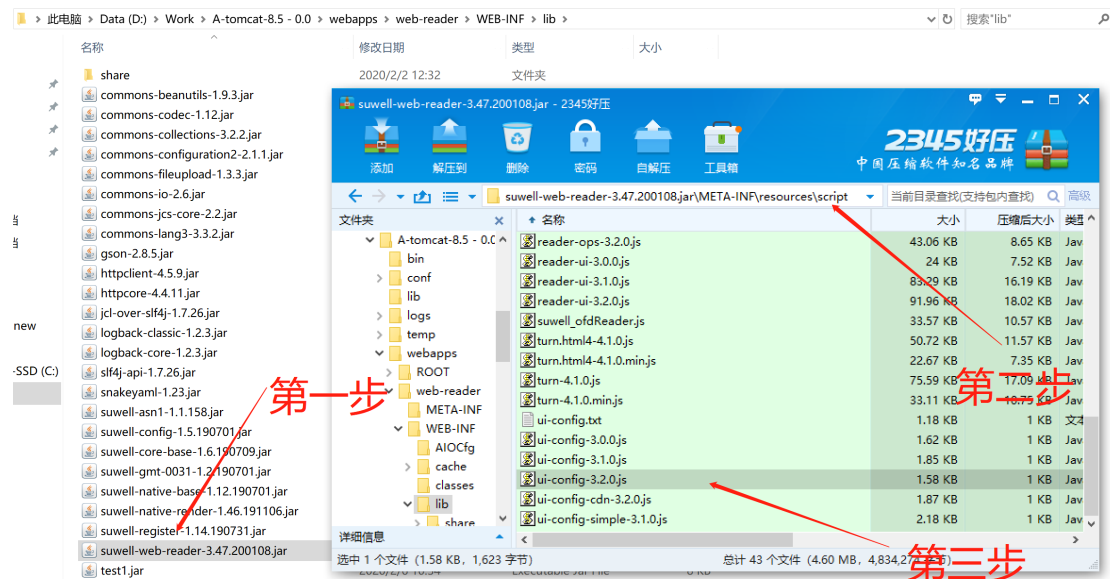
8、特殊配置

8.1 自定义工具栏按钮

功能说明：自定义轻阅读页面顶端工具栏按钮

配置文件路径：

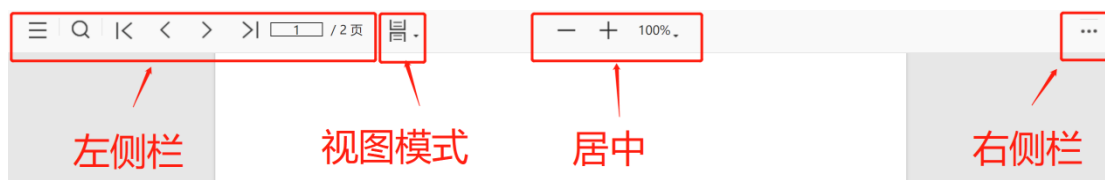
web-reader\WEB-INF\lib\suwell-web-reader*.jar\META-INF\resources\script\ui-config-3.2.0.js。



配置文件详细信息

解决方案：如需隐藏图三中工具栏按钮，对应表一删除 ui-config-3.2.0.js 中的参数如下图。

```
max: {
  // 左侧栏
  left: ['expand', '|', 'search', '|', 'textMode', 'imageMode', 'firstPage', 'prevPage', 'nextPage', 'lastPage', 'pageValue', 'pageCount',
  // 居中
  center: ['zoomOut', 'zoomIn', 'zoomSelect'],
  // 视图模式
  right: ['singleSerial', 'doubleSerial', 'double'],
  // 右侧栏
  dropdown: ['fullscreen', 'download', 'print', '|', 'about'],
  edit: ['|', 'hand', 'textSelect', 'painter', /*itext*/, /*revision-add*/, 'revision-replace', 'revision-del', 'revision-insert', 'erase', '']
}
```



功能	标识
折叠	expand
搜索	search
第一页	firstPage
上一页	prevPage
下一页	nextPage
最后页	lastPage
当前页码	pageValue
总页码	pageCount
单页连续	zoomOut
双页连续	zoomIn
翻页模式	zoomSelect
放大	zoomOut
缩小	zoomIn

自适应	zoomSelect
手型工具	hand
文本选择	textSelect
签批	painter
高亮	revision-add
标记为替换	revision-replace
标记为删除	revision-del
标记为插入	revision-insert
橡皮	erase
保存	save
全屏	fullscreen
下载	download
打印	print
关于	about

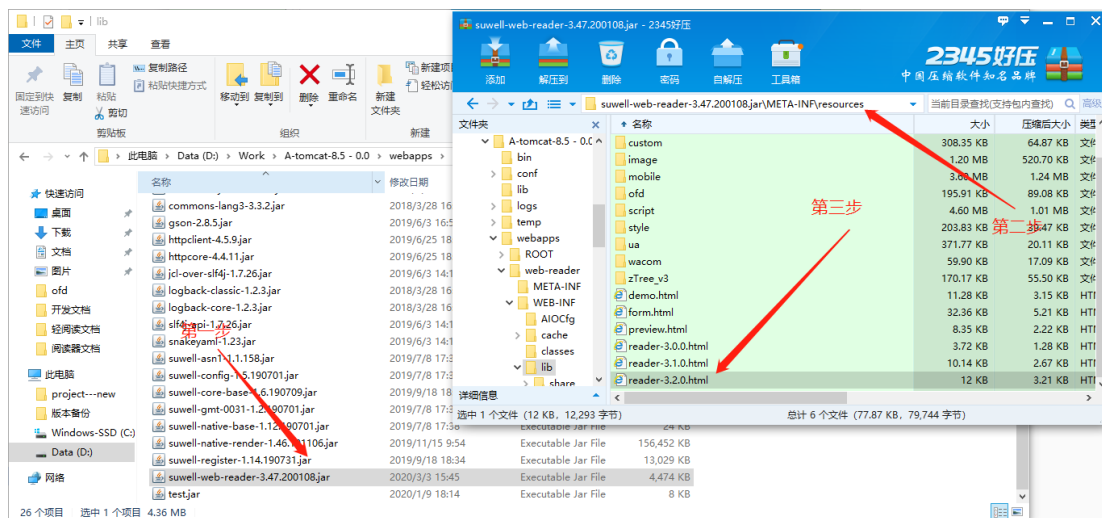
表一

8.2 设置默认缩放模式

功能说明：自定义初始化页面缩放模式

配置文件路径：

web-reader\WEB-INF\lib\suwell-web-reader*.jar\META-INF\resources\reader-3.2.0.html



配置文件详细信息

解决方案：如默认显示适合高度，在 zoom: data.zoom 后追加 ||'|' 如默认显示适合宽度，在 zoom: data.zoom 后追加 ||'|<>'。如默认显示缩放为 150%，在 zoom: data.zoom 后追加 ||'|1.5'

```

if (data.file && core) {
    var reader = new core.body({
        container: 'component',
        zoom: data.zoom,
        viewMode: data.viewMode,
        layoutMode: data.layoutMode
    }), namespace = '.index';

```

8.3 设置默认视图模式

功能说明：自定义初始化页面视图模式

配置文件路径：

web-reader\WEB-INF\lib\suwell-web-reader*.jar\META-INF\resources\reader-3.2.0.html。

配置文件详细信息

解决方案：初始化为单页连续，如默认显示双页连续，将 layoutMode: data.layoutMode 改为 layoutMode: data.layoutMode||'=='，如默认显示翻页模式，将 layoutMode: data.layoutMode 改为 layoutMode: B.LayoutMode.Double。

```

if (data.file && core) {
    var reader = new core.body({
        container: 'component',
        zoom: data.zoom||'1.5',
        viewMode: data.viewMode,
        layoutMode: data.layoutMode
    }), namespace = '.index';

```

8.4 设置预加载

功能说明：在打开轻阅读预览 OFD 前生成缓存文件，提高访问速度。但是会占用大量服务器资源，服务器硬盘小不建议使用。

解决方案：预览文件时将预览路径

http://127.0.0.1/reader/reader?file=1.ofd 改为

http://127.0.0.1/reader/reader/ready?file=1.ofd。也可以同时预加载多个文件，将预览路径改为

http://127.0.0.1/reader/reader/ready?file=1.ofd&file=2.ofd。

9、解决方案

9.1 移动端访问数字被当成电话号码呈蓝色链接

解决方案：访问页面加下面的设置

```
<meta name="format-detection" content="telephone=no"/>
```

9.2 兼容各版本的 IE

解决方案：访问页面加下面的设置

```
<meta http-equiv="X-UA-Compatible" content="IE=10;IE=9;IE=8;IE=EDGE">
```

9.3 URL 有特殊符号的问题解决方案

打开文件时，可能出现路径正确但无法打开文件的情况。是因为 url 中的路径需要先 encode 操作，否则浏览器会自行处理，导致轻阅读获取到的 url 不全，无法打开文件。例如 url 链接中有&符号，需要先转义为%26。

解决方法示例：

Jsp 集成时，链接的写法：

```
var v = encodeURIComponent('http://ip:port/test/1.ofd');
var reader=" http://ip:port/web-reader/reader?file=" +v;
$("#f").attr("src",reader);
<iframe id=" f" src=" " style="width:500px;height:500px"/>
```

9.4 URL 有中文会乱码的问题解决方案

解决方案：

1: 前台加 var b=encodeURIComponent(url);

```
<html>
```

```
<body>
```

```
<a id="a" href="">链接测试</a>
```

```
</body>
```

```
<script>
```

```
var url="http://ip:port/web-reader/reader?file=中文.ofd&user=01"
```

```
var b=encodeURIComponent(url);
```

```
document.getElementById("a").setAttribute("href",b);
```

```
console.log(b);
```

```
</script>
```

```
</html>
```

2: 中间件加编码格式设置 URIEncoding="UTF-8" permission.check = true

9.5 无法打开远程文件

2. url 路径的代码中没有 http 头中的 content-length，所以无法生成缓存文件。也就无法打开文件。轻阅读在下载该文件时会获取 http 头文件中的 content-length 长度，用于生成缓存，如果该值设置的不对会影响文件打开，建议设置为文件长度，或者为-1。

解决方案示例：

在下载文件流的代码中加语句：

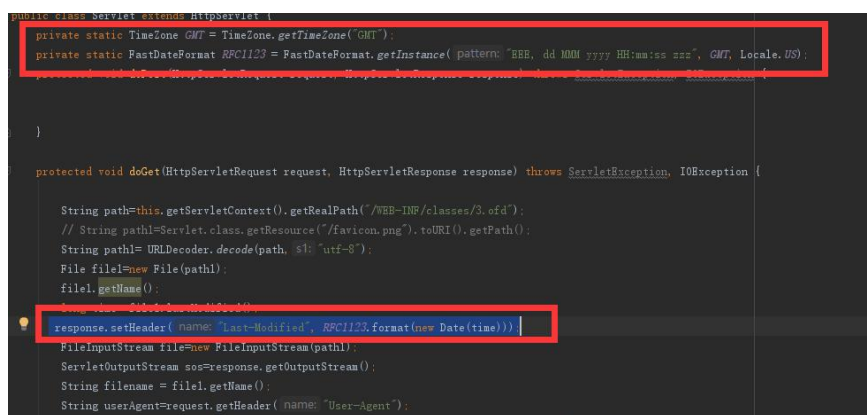
```
response.setContentLength(-1);
```

- 此方式打开文件，缓存无法刷新问题

9.6 缓存清理设置在下载链接的代码里设置 Last-Modified 用于重新打开文件时生成新的缓存。

- 在下载文件流的代码中加语句：

```
private static TimeZone GMT=TimeZone.getTimeZone("GMT");  
private static FastDateFormat RFC1123=FastDateFormat.getInstance("EEE,dd MMM  
yyyy HH:mm:ss zzz",GMT,Locale.US);  
response.setHeader("Last-Modified", RFC1123.format(new Date(time)));
```



```
public class Servlet extends HttpServlet {  
    private static TimeZone GMT = TimeZone.getTimeZone("GMT");  
    private static FastDateFormat RFC1123 = FastDateFormat.getInstance(pattern: "EEE, dd MMM yyyy HH:mm:ss zzz", GMT, Locale.US);  
    // ...  
}  
  
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
    String path=this.getServletContext().getRealPath("/WEB-INF/classes/3ofd");  
    // String path1=Servlet.class.getResource("/favicon.png").toURI().getPath();  
    String path1= URLDecoder.decode(path, "utf-8");  
    File file1=new File(path1);  
    file1.getName();  
    response.setHeader("Last-Modified", RFC1123.format(new Date(time)));  
    FileInputStream file=new FileInputStream(path1);  
    ServletOutputStream sos=response.getOutputStream();  
    String filename = file1.getName();  
    String userAgent=request.getHeader( name: "User-Agent");  
    // ...  
}
```