|  |
| --- |
| 技术选型与项目实现分析文档 |
| "Terra快乐分享"实现分析  创建日期：2013/02/01  创建人：梁志峰 |

修订历史：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 修订者 | 修订日期 | 备注 | 审批日期 |
| V1.0 | 梁志峰 | 2012/03/01 | 创建原始版本 |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

引言 3

1．1技术选型 3

难点需求 4

2. 0 登录后自动跳回登录前页面

2. 1文件上传 4

2. 1 文件下载

2. 2文件去重 与数据库设计 5

2. 3 分享与转载 与数据库设计 6

2. 4 RESTful WebService接口设计

2. 5 HDFS资源安全问题

2. 6 权限

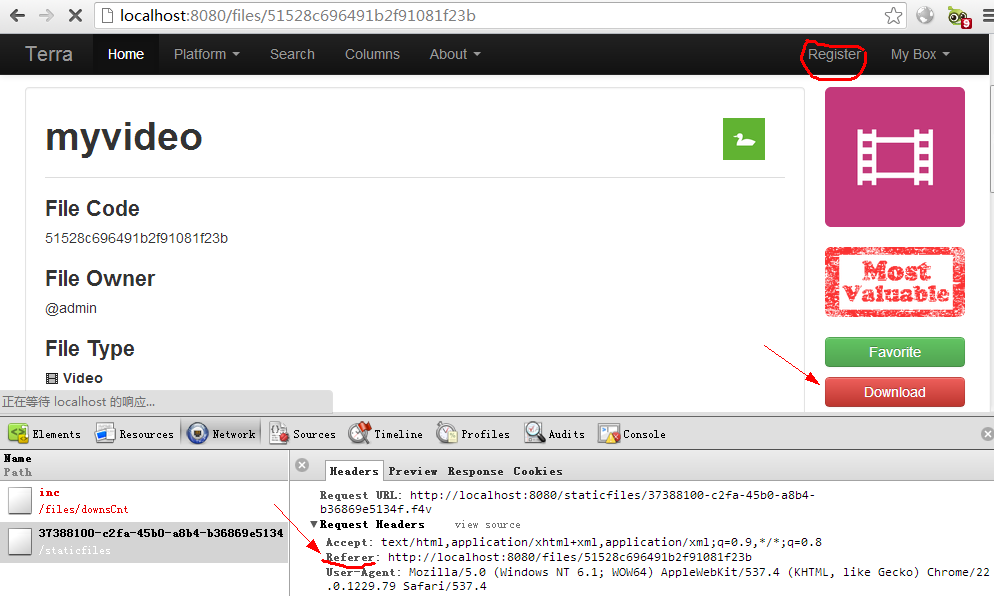
3.1基于JSON的REST Service 风格的设计

# 功能需求

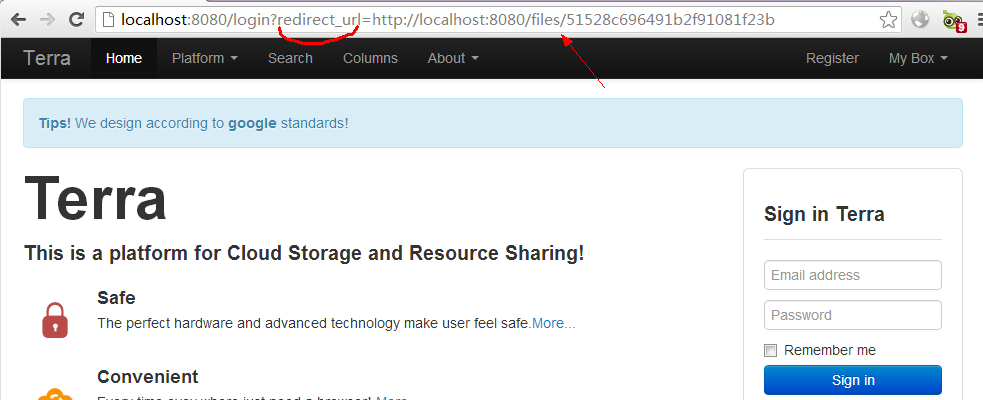
## 2．0 登录后自动跳回登录前页面

原理: 利用Referer+URL参数实现

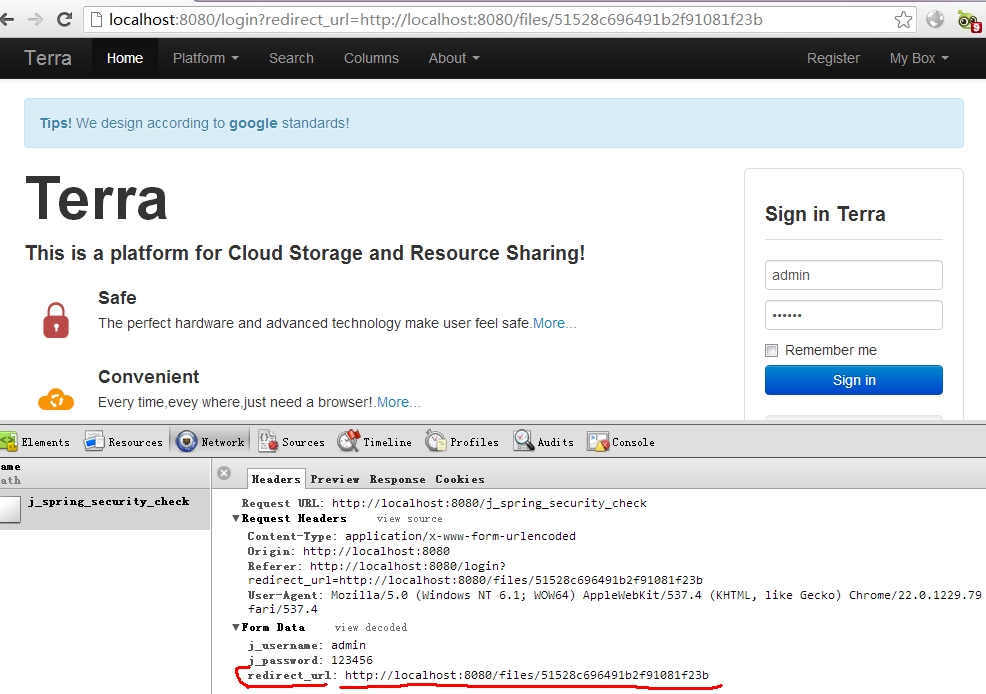
1:用户未登录,试图点击下载(下载前需要登录),点击Download时,发出的请求会带上Referer头

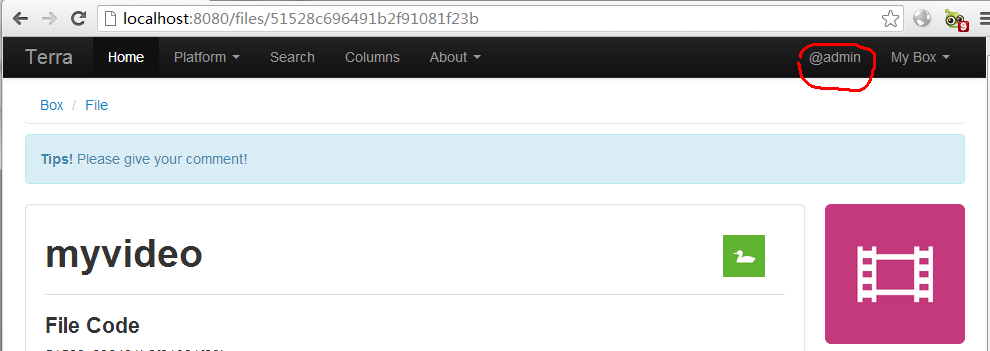


2: 重定向到登录页面时将此Referer头里的URL以URL参数方式带到登录页面,如:



3:登录时提交表单的时候,将direct\_url一起发送到服务器



4:登录后,跳转回登录前的页面  


## 2．1 文件上传

1. 浏览器方面,使用Ajax(XMLHttpRequest2)来异步上传文件
2. 由于文件上传会长时间占用服务器Servlet容器的线程,如果服务器收到过多的文件上传的请求,将会使容器的线程池的线程

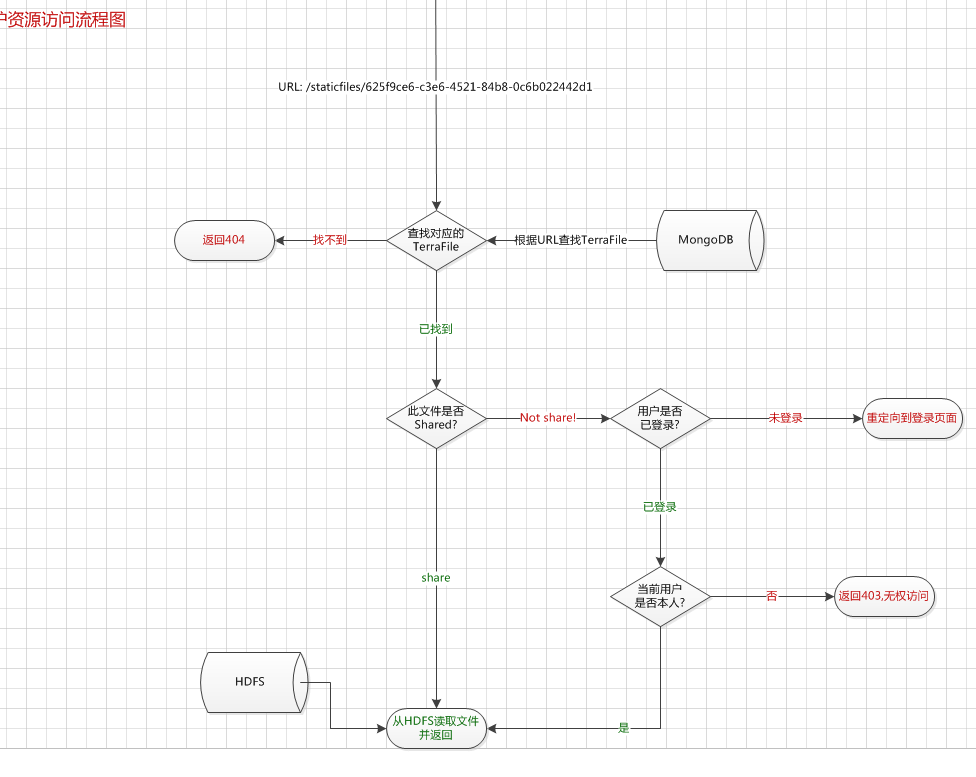
迅速耗尽,使得服务器无法处理新到来的请求,哪怕是普通页面的请求

1. 这里可以使用Servlet3.0 的AsynServlet特性: 当收到文件上传的请求后,处理完逻辑的处理后,新建一个后台线程来

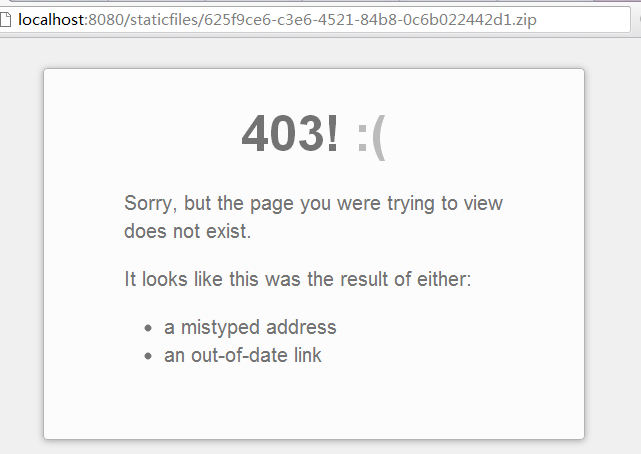
处理文件上传到HDFS的操作,保持输出流的连接,然后Servlet线程返回到容器,当该后台线程真正完成文件上传的

操作后,再将结果输出到该输出流.请求完成!

## 2．5 HDFS资源安全问题



无权限访问时返回403页面:



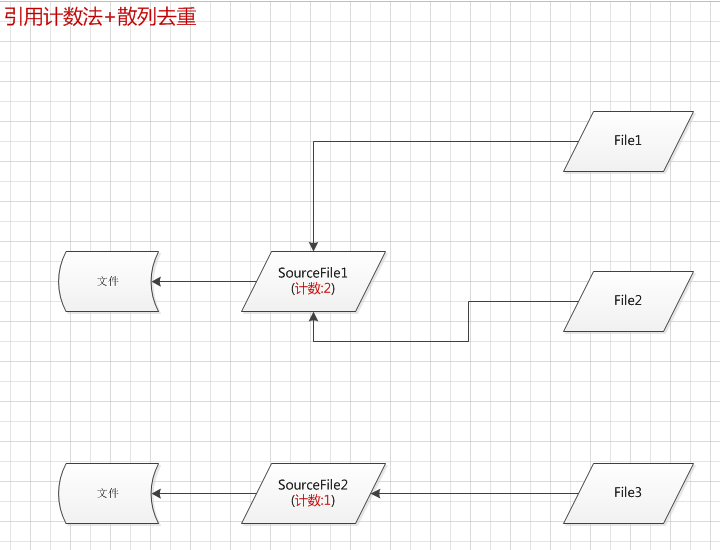
## HDFS文件去重

背景:

1. 文件去重是IO&CPU密集型操作,为了避免对正常业务造成影响,故选取凌晨02:00~04:00 业务较为空闲的时段进行.
2. 对新上传的文件进行MD5计算,然后将此值更新到文件的元数据表,如果发现MD5一致的文件,如果经过进一步比较

2个文件是否一致,如果一致,则将文件的元数据表的的HDFS路径指向同一份文件,并使文件引用计数+1,然后次重复

的文件删除.

2 

删除TerraFile的时候,若存在其它文件也在引用此文件资源,则更新: 使此资源的引用计数减1; 否则(没有其它文件引用此资源),直接删除此文件资源.

## 基于JSON的REST WebService 风格设计

为了使项目实现更好的模块化,更好了复用各模块的功能,故在实现各功能的时候参考了基于JSON的REST 的WebService来设计:

1. 灵活的登录方式

* URL: <http://localhost:8080/j_spring_security_check>
* 请求方式: POST
* 输入参数:

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **是否必须** | **类型** | **描述** |
| j\_username | 必须 | string | 用户名 |
| j\_password | 必须 | string | 密码 |

* 请求示例:

**POST /j\_spring\_security\_check HTTP/1.1**

**Host: localhost:8080**

**Content-Type: application/x-www-form-urlencoded**

**j\_username=admin&j\_password=123456**

* 正确的返回示例: (登录成功,状态码:200)

**HTTP/1.1 200 OK**

**Set-Cookie: JSESSIONID=D5657D72F1078A01DC5A5C0931D24BCE; Path=/; HttpOnly**

**Location: /myspace/file-list**

* 错误的返回示例: (密码错误,状态码:401)

**HTTP/1.1 401 Unauthorized**

**Server: Apache-Coyote/1.1**

**Transfer-Encoding: chunked**

**Date: Mon, 01 Apr 2013 02:44:17 GMT**

**{"reason":"密码错误"}**

* 错误的返回示例: (用户不存在,状态码:404)

**HTTP/1.1 404 Not Found**

**Server: Apache-Coyote/1.1**

**Transfer-Encoding: chunked**

**Date: Mon, 01 Apr 2013 02:34:06 GMT**

**{"reason":"用户名不存在"}**

1. 批量文件删除

* URL: <http://localhost:8080/files/delete>
* 请求方式: POST
* 输入参数:

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **是否必须** | **类型** | **描述** |
| files | 必须 | string | 要删除的文件的ID列表 |

* 请求示例:

**POST /files/delete HTTP/1.1**

**Host: localhost:8080**

**Accept: application/json, text/javascript, \*/\*; q=0.01**

**Content-Type: application/x-www-form-urlencoded; charset=UTF-8**

**Cookie: JSESSIONID=7E5966D49D6FC4AA738927AD59587713**

**files=5157f65dcaff3f5d76868e14,2C5157f50bcaff575e6302dbeb,2C5157f28dcaff5ee23a2ed364**

* 正确的返回示例: (删除成功,状态码:200)

**HTTP/1.1 200 OK**

**Server: Apache-Coyote/1.1**

**Content-Type: application/json;charset=UTF-8**

**Transfer-Encoding: chunked**

**Date: Mon, 01 Apr 2013 02:47:41 GMT**

**{"status":"SUCCESS","cnt":3}**