

Tutorial: Debugging Documentum REST Services

This doc shows some tips and practices for debugging your Documentum REST Services instance (seeing [IIG RESTful Web Services](#)).

Check The REST Build Version

It's for sure that you could check the REST build version from the WAR file or JAR file's manifest files. But one other convenient way is to get the information from Product Info Resource.

```
{
  name: "documentum-rest-services-product-info",
  - properties: {
    product: "EMC Documentum REST Services",
    product_version: "7.0.3141.5926",
    major: "7.0",
    minor: "3141",
    build_number: "5926",
    revision_number: "27182818284"
  },
  - links: [
    - {
      rel: "self",
      href: "http://demo/dctm-rest/product-info.json"
    }
  ]
}
```



Check The Error Response

For a failed REST request invocation, the server returns an error response in XML or JSON representation, according to the content negotiation. This error representation usually contains the error code, error message and details of the error which shall give you some indication.

```
{
  "status" : 409,
  "code" : "E_CREATE_FOLDER_PATH_EXISTS",
  "message" : "The folder name already exists.",
  "details" : "(DM_FOLDER_E_PATH_EXISTS) Cannot save (or link) 'My New Folder' folder with path name '/acme01/My New Folder' because one already exists."
}
```

Use The Browser As Your Tester

There is nothing Documentum REST Services does that you cannot test right there using your browser. For a read-only resource, you could just input the resource URI in the address bar to test the result. For the non-GET HTTP methods, you could use the browser's develop plugins to test the REST API. In addition, all modern web browsers have built-in web debugging tools which help you to diagnose the REST invocation issues.

Internet Explorer 11

The screenshot displays the Internet Explorer 11 interface. The main content area shows an XML document with the following structure:

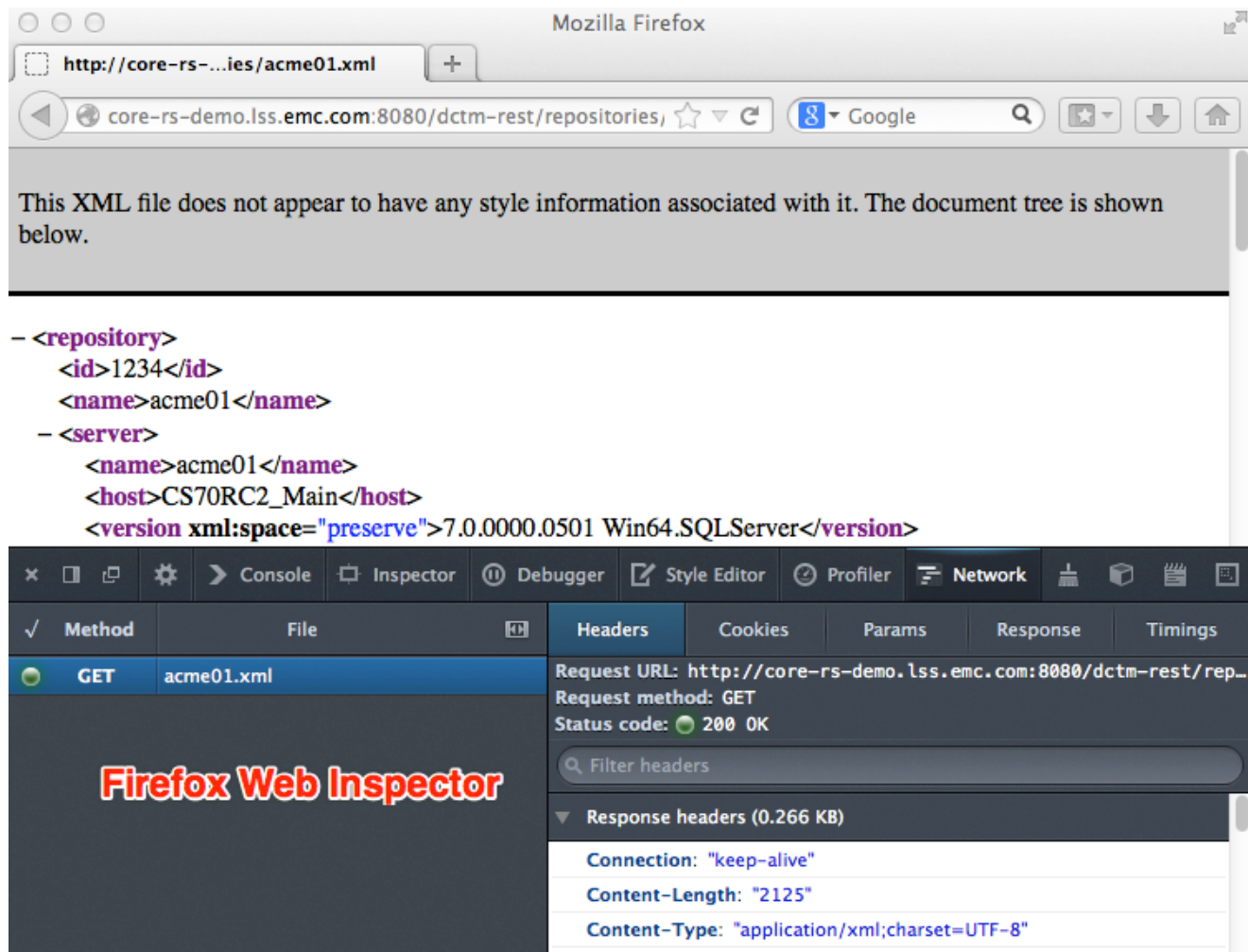
```
<?xml version="1.0" encoding="UTF-8" ?>
- <repository xmlns="http://identifiers.emc.com/vocab/documentum"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <id>1234</id>
  <name>acme01</name>
  - <server>
    <name>acme01</name>
    <host>CS70RC2_Main</host>
    <version xml:space="preserve">7.0.0000.0501 Win64.SQLServer</version>
    <docbroker>CS70RC2_Main</docbroker>
  </server>
  - <links>
    <link rel="self" href="http://core-rs-demo.lss.emc.com:8080/dctm-
      rest/repositories/acme01.xml" />
  </links>
</repository>
```

The bottom pane shows the Network and Developer tools. The Network pane has a 'SUMMARY' tab with the following data:

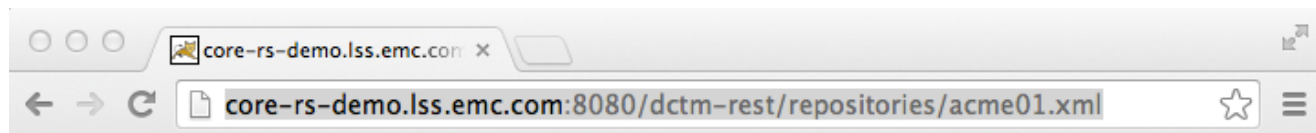
URL	Protocol	Method	Result	Type	Received	Taken
http://core-rs-demo.lss.emc.com:8080/dctm-rest/repositories/acme01.xml	HTTP	GET	200	application/xml	2.22 KB	78 ms
/favicon.ico	HTTP	GET	404	text/html	150 B	16 ms

The Developer tools pane is open, showing the 'Inspect element' button. The status bar at the bottom indicates: Items: 2, Sent: 1.71 KB (1,754 bytes), Received: 2.37 KB (2,425 bytes).

Firefox 23



Chrome 32



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<repository xmlns="http://identifiers.emc.com/vocab/documentum"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <id>1234</id>
  <name>acme01</name>
  <server>
    <name>acme01</name>
    <host>CS70RC2_Main</host>
    <version xml:space="preserve">7.0.0000.0501 Win64.SQLServer</version>
    <docbroker>CS70RC2_Main</docbroker>
  </server>
  <links>
    <link rel="self" href="http://core-rs-demo.lss.emc.com:8080/dctm-
```

Chrome web inspector

Elements | **Network** | Sources | Timeline | Profiles | Resources | Audits | Console

Name Path

acme01.xml /dctm-rest/repositories

1 requests | 2.2 KB transferred ...

Headers | Preview | Response | Cookies | Timing

Request URL: http://core-rs-demo.lss.emc.com:8080/dctm-rest/repositories/acme01.xml

Request Method: GET

Status Code: 200 OK

Request Headers view source

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Encoding: gzip, deflate, sdch

Accept-Language: en,zh-CN;q=0.8,zh;q=0.6

Authorization: Basic ZG1hZG1pbjpwYXNzd29yZA==

Connection: keep-alive

Safari 6.1

The screenshot shows a Safari 6.1 browser window displaying an XML file from `core-rs-demo.lss.emc.com:8080/dctm-rest/repositories/acme01.xml`. The XML content is as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<repository xmlns="http://identifiers.emc.com/vocab/documentum"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <id>1234</id>
  <name>acme01</name>
  <server>
    <name>acme01</name>
    <host>CS70RC2_Main</host>
    <preserve>7.0.0000.0501 Win64.SQLServer</version>
    <docbroker>CS70RC2_Main</docbroker>
  </server>
</repository>
```

Below the browser window, the Safari Web Inspector is open, showing the resource `acme01.xml` and its details:

Type	
MIME Type	application/xml
Resource Type	Other
Location	
Full URL	http://core-rs-demo.lss.emc.com:8080/dctm-rest/repositories/acme01.xml
Scheme	http
Host	core-rs-demo.lss.emc.com
Port	8080
Path	/dctm-rest/repositories/acme01.xml
Filename	acme01.xml
Request & Response	
Method	—

Understand The Server Logs

The REST server logging is leverage the log4j framework. By default, INFO level logging is enabled for the package "com.emc.documentum.rest", so once the REST server is started up, there are already useful information showing in the log file.

[illegible]

All REST runtime properties and the authentication mode is logged into the file which may help you to verify your deployment configurations.

Further more, to get the DEBUG or TRACE level logs, you'll need to set the corresponding logging level on the package "com.emc.documentum.rest". Optionally, you may need to enable DEBUG level logging for the dependent packages such as DFC and Spring Framework.

Dump The Server Messages

You could also dump the REST request and response messages (including their HTTP headers) in the server log file by customizing the **rest-api-runtime.properties** file.

```
# The default value is false.  
  
rest.message.logging.enabled=true  
  
# The default value is 1048567.  
  
# rest.message.logging.buffer=
```

Besides that, the package "com.emc.documentum.rest.log" needs to be set as DEBUG level in **log4j.properties**.

[Learn More About Documentum REST Services >>](#)