Developing, Compiling, and Running a Non-Web Data Source Crawler Plug-in with IBM Content Analytics with Enterprise Search 3.0

Vijai Gandikota
April 2013
# Contents

Introduction ................................................................................................................................. 1
Understanding the code of the crawler ...................................................................................... 1
The data for this example .............................................................................................................. 2
Compiling and using the sample crawler plug-in ....................................................................... 3
The sample plug-in code .............................................................................................................. 5
Removing content and submitting only the metadata ................................................................ 6
Modifying the content .................................................................................................................. 6
Compiling the plug-in code ........................................................................................................... 7
   The Java Compiler ...................................................................................................................... 7
   Classpath .................................................................................................................................... 7
   Compilation ............................................................................................................................... 7
   Archival ...................................................................................................................................... 8
   Verification ............................................................................................................................... 9
Adding the plug-in to the crawler configuration in the administration console ....................... 9
   Start Content Analytics with Enterprise Search .......................................................................... 9
   Creating a collection .................................................................................................................. 10
   Creating and configuring a crawler ............................................................................................ 16
      To use or not use the package ................................................................................................. 18
Running the crawler with the plug-in .......................................................................................... 22
   Reading the RDS to verify the crawled data ............................................................................. 23
      RDS Read Usage ..................................................................................................................... 23
      RDS Dump .............................................................................................................................. 24
   Understanding what was crawled .............................................................................................. 26
Parse and Index ............................................................................................................................. 27
   DumpIndex ............................................................................................................................... 28
      Examining the DumpIndex ....................................................................................................... 28
Creating an Index Field and Field Mapping .............................................................................. 30
Using the Plug-inLogger to log messages from the Plug-in ....................................................... 34
Troubleshooting .......................................................................................................................... 36
**Introduction**

A data source crawler plug-in is a Java application associated with a data source crawler (a non-web crawler). Examples of such crawlers are Windows file system crawler, Seed list crawler, etc.

Using a data source crawler plug-in, you can create, modify or delete the content, metadata, or security tokens.

For more information about the API used in a crawler plug-in, see the following location on the machine where IBM Content Analytics with Enterprise Search 3.0 is installed.

```
ES_INSTALL_ROOT/docs/api/crawler
```

In this article, we cover the plug-ins for non-web data source crawlers only. We do not cover the enhanced crawler framework plug-ins (for FileNet P8, SharePoint, and Agent for Windows file systems crawlers).

To use a crawler plug-in, you must:

1. Create a Java class
2. Compile it, including the necessary libraries in the classpath
3. Archive it as a .jar file
4. Configure a data source crawler to use this plug-in
5. Start the crawler
6. View the log files to troubleshoot any problems with the crawler plug-in

The Java class of the plug-in is called for each document that is crawled. The crawler then passes the document identifier, the security tokens, the metadata, and the content of the document. The plug-in Java class can then modify these items and return them to the crawler. You can add or modify the security tokens but not the native ACLs collected by the crawler.

**Understanding the code of the crawler**

The default constructor instantiates the object of the plug-in class. When the plug-in code is executed, the following methods are called once:

1. Init
2. isMetadataUsed()
3. isContentUsed()
4. term()
5. updateDocument()

When the plug-in terminates, the term() method is called.

You must implement the updateDocument() method to add, modify, or delete metadata, security tokens, and content.
The data for this example

In this example, the crawler crawls a folder that contains two text documents, TextDoc1.txt and TextDoc2.txt, which have the content Text 1 and Text 2 respectively.

The security permissions are the default for the administrative user of the Windows operating system.
No additional metadata has been added to the documents. Therefore the crawler extracts the title as the file name.

Note that the URI of the document cannot be modified by using a crawler plug-in. However, you can modify the other metadata such as the title.

Compiling and using the sample crawler plug-in
Access the information center and search for the term “plug-in”. Access the link for “Sample plug-in application for non-web crawlers”.

3
Create a folder:

C:\plug-in

Copy the code from the above page and save it in a file called MyCrawlerPlug-in.java.

The plug-in class name and the name of the file should be the same.

Remove this line:

```java
package sample;
```

If this line is not removed, then the crawler will throw an exception that the class is not found. Alternatively, specify the crawler plug-in class as sample.MyCrawlerPlug-in in the crawler configuration.
The sample plug-in code
Let us now examine some sections of the code.

If you modify both metadata and security tokens, make sure that the isMetadataUsed() method returns true. If it only modifies security tokens, ensure it returns false.

Let’s examine the updateDocument() method.

The following three statements retrieve the URI string, security tokens, and field metadata:

```java
String url = crawledData.getURI();
String securityTokens = crawledData.getSecurityTokens();
List metadataList = crawledData.getMetadataList();
```

If a document that meets certain criteria is to be rejected, add the logic in this section:

```java
// This sample always returns updated document.
if (false) {
    return null;
}
```

For example you could check for a metadata field value. If found, this if block should return null.

The following section creates the security token and sets it in the crawledData object:

```java
String newToken = "SampleToken";
String newSecurityTokens = securityTokens + "," + newToken;
crawledData.setSecurityTokens(newSecurityTokens);
```

The following section creates a new Metadata field:

```java
// update metadata (for sample)
FieldMetadata newFieldMetaData = new FieldMetadata("copyright", "IBM");
metadataList.add(newFieldMetaData);
crawledData.setMetadataList(metadataList);
```

The following statement sets the mime type to text/plain:

```java
content.setMimeType("text/plain");
```

Also note that in the sample plug-in code, we do not really read the content and we replace it with new text, "The new content of plain text ", in the following section:

```java
String newText = "The new content of plain text ";
BufferedWriter br = new BufferedWriter(new OutputStreamWriter(outputStream, "UTF-8"));
br.write(newText);
```

Instead of replacing the content, you can add to it or modify it.
Then we submit the content to the crawledData object and return the crawledData object to the crawler.

crawledData.submitContent(content);
    return crawledData;

Removing content and submitting only the metadata
Many times we encounter documents having binary content that are not supported by the parser or the stellent parser. Yet, to be able to provide the document in the search results is a requirement. The way to achieve this is by crawling all the metadata and the security tokens but removing the content.

We can use the crawler plug-in and not submit the content while submitting only the metadata and security tokens.

To do this, set the content submitted to null as shown below:

    // Submit change for the content.
    crawledData.submitContent(null);
    return crawledData;

Modifying the content
Import the following:

import java.io.BufferedReader;
import java.io.InputStreamReader;

Add the following statements in the updateDocument method under the comment //read the content:

    // read the content
    BufferedReader breader = new BufferedReader(new InputStreamReader(in));
    if((inputText = breader.readLine()) == null){
        inputText="";
    }

Modify the statements under the comment // write content to OutputStream:

    // write content to OutputStream
    String newText = inputText + " " + "The new content of plain text ";
Compiling the plug-in code

The Java Compiler
Download and install Java 1.6 (jdk1.6.0_45) from the Oracle website. Currently, this is the latest version of Java supported by Content Analytics with Enterprise Search for the plug-in compilation and archival. This is the last version also available at the Java 1.6 level.

Once you have downloaded Java 1.6, ensure either that the PATH environment variable is pointing to the bin folder containing the executables or to the executables from that folder so that the executable of the correct version is used.

Classpath
You must include the path to the dscrawler.jar file in the classpath during compilation as shown below. The typical location of dscrawler.jar is in the lib folder under ES_INSTALL_ROOT.

```
C:\>echo %ES_INSTALL_ROOT%
C:\Program Files\IBM\es
C:\>cd %ES_INSTALL_ROOT%\lib
C:\Program Files\IBM\es\lib>dir dscrawler.jar
  Volume Serial Number is 03F-BF83
  Directory of C:\Program Files\IBM\es\lib
  03/07/2013  11:46 AM   532,305 dscrawler.jar
      1 File(s)    532,305 bytes
      0 Dir(s)   19,622,023,164 bytes free
C:\Program Files\IBM\es\lib>
```

Verify that this file exists at the path you are specifying.

Compilation
Now compile the Java program:

```
C:\plug-in>"C:\Program Files\Java\jdk1.6.0_45\bin\javac" -cp "C:\Program Files\ibm\es\lib\dscrawler.jar" MyCrawlerPlug-in.java
```
If you see any errors during compilation, revisit the code and correct any modifications you may have done that caused the problems.

**Archival**

Next make a .jar file for this .class file:

```
C:\plug-in>"C:\Program Files\Java\jdk1.6.0_45\bin\jar" -cvf MyCrawlerPlug-in.jar MyCrawlerPlug-in.class
```

Check the directory to make sure that the .jar file is created.
Verification
Also verify that the .jar file includes Manifest.xml and MyCrawlerPlug-in.class. You can do this by “un-jarring” the .jar file.

Adding the plug-in to the crawler configuration in the administration console

Start Content Analytics with Enterprise Search
Start all of the sessions of IBM Content Analytics with Enterprise Search 3.0 by running this command:

`esadmin system startall`
Creating a collection

Access the administration console (ESAdmin) and log in with the default administrator user ID and password (this user ID is configured when you install Content Analytics with Enterprise Search).
Click the Collections tab and then click the **Create Collection** button.

In the **Collection name** field, enter “Plug-inCollection”.

For **Collection type**, select **Enterprise search collection**.
Under **Advanced options**, enter “To use the crawler plug-in for data source crawlers” in the **Description** field.

Under **Collection security**, select **Enable security for the collection**. This is necessary to be able to use the security tokens and the Native ACLs extracted by the crawler.
Scroll down and in the field for **Collection ID**, select the radio button for **Custom ID** and enter **plug-in** in the text box for the collection ID value.
Scroll down and click **OK** to create the collection.
The collection that you created is displayed in the Enterprise Search Collections section of the Collections tab:
Creating and configuring a crawler

Click on the **Crawlers** section and click on the + sign to create a crawler.

On the Create a Crawler page, select **Windows file system** in the drop-down for the **Crawler type** field.

Click **Next** to go to the Windows Crawler Properties page.

Enter “WFS Crawler With Plug-in” in the **Crawler name** field
Click **Next** to go to the Select Windows Subdirectories to Crawl page.
To use or not use the package

If you specify the package name “sample” in the code, specify the plug-in class as sample.MyCrawlerPlug-in in the crawler configuration screen, as shown in the following image:

You can select the check box to run in unfenced mode (that is, the plug-in process runs inside the crawler process). This mode might help improve the plug-in performance. However, if the plug-in runs into an unrecoverable problem, it will result in the termination of the entire crawler process.

If you want the crawler plug-in to also run on documents that you have specified in the exclusion list of file extensions, select that option in the drop-down for excluded document content.
Click **Next** to go to the Select Windows Subdirectories to Crawl page. Select the folder to crawl and click **Next**.
Click **Finish** to configure the crawler.
Observe that the Parse and Index session is started and waiting for documents. In this article we will also observe the raw data store (RDS) and the Index so that we can view the items added/modified by the plug-in. The output of the crawler is the RDS. The output of the Parse and Index session is the Index.

If the Parse and Index session runs continuously, it will immediately consume the documents from the RDS and put them in the index. So, click on the red box next to the Parse and Index session to stop it. This way we can complete a crawl, collect the RDS Read, and then proceed to Parse and Index the documents. You can leave the search session running.
Running the crawler with the plug-in
Now click on the green Start button of the crawler to start the Windows file system crawler.
Reading the RDS to verify the crawled data
At the command prompt, enter the following command to do the RDS read:

```
esadmin rds read -cid plug-in -metadata -httpcode -content> RDSDump.txt
```

**RDS Read Usage**
To know more about the options to use with the esadmin rds command, run `esadmin rds help`.

```
esadmin rds help at the command line
```

FFQU0310W This tool 'esadmin rds' provides diagnostic information that can help IBM Software Support troubleshoot problems. It is strongly recommended that you use this tool only under the guidance of IBM Software Support. FFQC5392I Usage: esadmin rds read -cid collectionID [-options]

These commands must be run on the index server.
The 'rds' command reads the raw data source for a collection. The raw data source contains all the documents that were crawled by the crawlers for this collection. The collection parser reads from the raw data source, tokenizing and applying linguistic analysis on each document.

Options:
- `-if-url string`: dump only documents whose URL contains the given string
- `-if-httpcode code`: dump only those documents whose HTTP code contains the given code
- `-if-flags flags`: dump only documents whose flags contains the given integer value
- `-file string`: send the output to a file
- `-info`: print details about the raw data source storage
- `-url`: print the URL of each document
- `-httpcode`: print the HTTP code for each document
- `-flags`: print the flags for each document
RDS Dump
Open the RDSDump.txt file and review its contents:

URL: file:///C:/tocrawl/text+files/TextDoc2.txt
HTTP Code: 2000
Metadata: <?xml version='1.0' encoding='UTF-8'?>
<Metadata Language="en">
  <CommonMetadata Datasource="WinFS" StaticScoreRef="0"
    CrawlerId="plugin.WIN_50605" CrawlspaceId="t1367092881385"
    DatasourceName="WFS Crawler With Plugin" IsCustomDatasource="NO"
    Date="1363966052">
    <HasSeparateContent ContentType="text/plain" Charset="UTF-8"
      Language="en" Truncated="NO">YES</HasSeparateContent>
    <SecurityACLs Public="NO">,SampleToken</SecurityACLs>
    <NativeACLs>
      <Impersonate>YES</Impersonate>
      <Domain>IBM-C4FEB0165E1</Domain>
      <Layer Level="1">
        <Allow>
          <Entry>Users</Entry>
          <Entry>Administrators</Entry>
          <Entry>SYSTEM</Entry>
          <Entry>Authenticated Users</Entry>
        </Allow>
      </Layer>
      <Layer Level="2">
        <Allow>
          <Entry>Users</Entry>
          <Entry>Administrators</Entry>
          <Entry>SYSTEM</Entry>
          <Entry>Authenticated Users</Entry>
        </Allow>
      </Layer>
    </NativeACLs>
  </CommonMetadata>
  <DatasourceSpecificMetadata>
    <Field FieldName="__$Directory$__" Searchable="YES" FieldSearchable="NO"
      Metadata="NO" ParametricSearch="NO" ResolveConflict="MetadataPreferred"
      IsContent="NO" Implicit="YES" PluginCreated="NO" ExactMatch="NO"
      Sortable="NO">C:\tocrawl\text files</Field>
    <Field FieldName="__$FileName$__" Searchable="YES" FieldSearchable="NO"
      Metadata="NO" ParametricSearch="NO" ResolveConflict="MetadataPreferred"
      IsContent="NO" Implicit="YES" PluginCreated="NO" ExactMatch="NO"
      Sortable="NO">TextDoc2.txt</Field>
    <Field FieldName="__$Extension$__" Searchable="YES" FieldSearchable="NO"
      Metadata="NO" ParametricSearch="NO" ResolveConflict="MetadataPreferred"
      IsContent="NO" Implicit="YES" PluginCreated="NO" ExactMatch="NO"
      Sortable="NO">.txt</Field>
    <Field FieldName="__$ModifiedDate$__" Searchable="YES"
      FieldSearchable="NO" Metadata="NO" ParametricSearch="NO"
      ResolveConflict="MetadataPreferred" IsContent="NO" Implicit="YES"
Understanding what was crawled
Observe that there are two documents. HTTP Code: 2000 signifies a successful crawl.

Observe the following metadata values:

Language="en"
Datasource="WinFS"
CrawlerId="plug-in.WIN_50605"
DatasourceName="WFS Crawler With Plug-in"

Notice that the following entry:

ContentType="text/plain"

Specifies the mime type that was set.

Notice that the NativeACLs section cannot be changed:

```
<NativeACLs>
  <Impersonate>YES</Impersonate>
  <Domain>IBM-C4FEB0165E1</Domain>
  <Layer Level="1">
    <Allow>
      <Entry>Users</Entry>
      <Entry>Administrators</Entry>
      <Entry>SYSTEM</Entry>
      <Entry>Authenticated Users</Entry>
    </Allow>
  </Layer>
</NativeACLs>
```
Immediately prior to that section is the Security ACLs section, which shows that the security token “SampleToken” was inserted:

<SecurityACLs Public="NO">,SampleToken</SecurityACLs>

In the DatasourceSpecificMetadata section, the copyright field has been created. We will examine later how to make sure that this field appears in the search results. This shows that the copyright field was created with the value IBM.

<Field FieldName="copyright" Searchable="YES" FieldSearchable="NO" Metadata="NO" ParametricSearch="NO" ResolveConflict="MetadataPreferred" IsContent="YES" Implicit="NO" PluginCreated="YES" ExactMatch="NO" Sortable="NO">IBM</Field>

Parse and Index

Now, start the Parse and Index session by clicking on the Start button next to the Parse and Index session for the collection:
**DumpIndex**

Once the Parse and Index session completes and shows that two documents were processed, run the following command at the command prompt:

```
C:\plug-in>DumpIndex plug-in --uri --fields --code --indextokens > dumpindex.txt
```

**Examining the DumpIndex**

Found 2 documents:

**ID: 0**

**URI:** file:///C:/tocrawl/text+files/TextDoc2.txt  
**Code:** 200  
**Index Tokens:**

- $plain:
  - the@0[B]
  - the@0[L]
  - new@1[B]
  - new@1[L]
  - content@2[B]
  - content@2[L]
  - of@3[B]
  - of@3[L]
  - plain@4[B]
  - plain@4[L]
  - text@5[B]
  - text@5[L]
  - s\2@5[S]
  - textdoc2@6[O]
  - txt@7[O]

- $#%date: 
  - @-1[G] <0,51,4c,78,64>

- $directory:
  - c:\\tocrawl\\text files@0[O]

- $filename:
  - textdoc2.txt@0[O]

- $extension:
  - .txt@0[O]

- $#%modifieddate: 
  - @-1[G] <0,51,4c,78,64>

- $#%filesize: 
  - @-1[G] <0,6>

- $title:
  - textdoc2@0[O]
  - txt@1[O]

- $url: 
  - file@0[O]
  - @0[O]
  - /@2[O]
  - /@3[O]
  - /@4[O]
  - c@5[O]
  - @6[O]
  - /@7[O]
Observe that the tokens from the content have been taken and put into the index. Also observe that a list of fields was created. However, notice that the field copyright does not exist. The reason is that before being able to use the field for search, we must first create an index field.

**Creating an Index Field and Field Mapping**

Click on the pencil icon in the Parse and Index session and click on **Index fields**.
Click the **Create Index Field** button.
On the Create an Index Field page, enter the values as shown in the following image:

The name of the field must be the same as the field that you are creating with the plug-in. Click OK.
Click **Return**. Click on the Edit icon again, and click on **Index field mappings for extracted metadata**.

On the page to Map Extracted Metadata to Index Fields, click the **Add Mapping** button.

Type **copyright** in the **Metadata field name** field. In the drop-down for **Field name**, select the copyright field that was created earlier. Click **OK** and return to the Collections page.
Start the crawler session. Click on the Monitor (eye) icon and then click the **Start a full crawl** button.

Next, either start the Parse and Index session or wait for the indexing to complete if it is already started.

Now run the DumpIndex command and examine the output.

This time you can see the copyright field in the DumpIndex. Because we made the field returnable and field searchable, you can see it in the list of fields in the search application for each document in the results. You can also search for documents based on the value that appears in this field by using the syntax for doing fielded search.

**Using the Plug-inLogger to log messages from the Plug-in**

The Plug-inLogger is a class that provides the capabilities to include log statements in the Content Analytics with Enterprise Search logs from the plug-in. To use the Plug-inLogger, specify the following statement in the import statements:

```java
import com.ibm.es.crawler.plugin.logging.PluginLogger;
```

Add the following statements after the start of the class declaration:

```java
/** Logger */
    private static final PluginLogger logger;
    static {
        PluginLogger.init(PluginLogger.LOGTYPE_OSS, PluginLogger.LOGLEVEL_INFO);
        logger = PluginLogger.getInstance();
    }
/** End Logger **/```
In the updateDocument section, add the following statements to output test logging statements of the type INFO, WARN and ERROR:

```java
/* Testing Logging Statements*/
logger.info("This is info.");
logger.warn("This is warning.");
logger.error("This is an error.");
/* End Testing Logging Statements */
```

These three statements for info, warn, and error appear in the collection log as shown below:

```
W  FFQD2801W 2013/04/27 23:02:05.619 CDT plug-in plug-in.WIN_50605.crawlerplug-in FFQD2801W A warning was generated from the crawler plug-in. Message: This is a warning message.
E  FFQD2800E 2013/04/27 23:02:05.681 CDT plug-in plug-in.WIN_50605.crawlerplug-in FFQD2800E An error was generated from the crawler plug-in. Message: This is an error message.
```

To change this setting and allow INFO messages also to appear, click Actions > Logging > Configure log file options as shown below.

Click on the drop-down for Type of information to log and select All messages. Select the same option for the Type of information to trace for audit.
Stop and restart the crawler session and look for the informational messages now in the collection log.

**Troubleshooting**

If any error or exception occurs after you try to start the crawler, access the collection log in the logs folder or the crawler log in the audit folder.

1. Here is an example from the crawler audit log. In this example, the exception points to the version of Java used for compilation and to jar the class file.

```plaintext
I     FFQX1208I  2013/04/24 19:23:09.437 CDT FFQX1208I DSCrawlerAPI.init() was called.
I     FFQX1208I  2013/04/24 19:23:09.468 CDT FFQX1208I DSCrawlerAPI.init(): the destination of the controller node = IBM-C4FEB0165E1
I     FFQX1208I  2013/04/24 19:23:09.484 CDT FFQX1208I DSCrawlerAPI.init(): crawler temporary file is set for this session. C:\Program Files\IBM\es\esadmin\tmp\plug-in.WIN_14094
I     FFQX1208I  2013/04/24 19:23:09.546 CDT FFQX1208I DSCrawlerAPI.start() was called.
I     FFQX1208I  2013/04/24 19:23:16.171 CDT FFQX1208I Valut config handler was initialized with C:\Program Files\IBM\es\esadmin\config\vault.xml
```
I  FFQX1208I  2013/04/24 19:23:17.015 CDT  FFQX1208I Valut configuration has been applied successfully.
I  FFQX1208I  2013/04/24 19:23:18.171 CDT  FFQX1208I schema:ESADMIN conId:0

E  FFQQ0277E  2013/04/24 19:23:25.187 CDT  FFQQ0277E An exception was caught with the detail 'com.ibm.es.crawler.utilities.CrawlerException: FFQD3027E An error occurred for the crawler.' and a stack trace of 'com.ibm.es.crawler.utilities.CrawlerException:
 at com.ibm.es.crawler.CrawlerManager.initialize(CrawlerManager.java:637)
 at com.ibm.es.crawler.CrawlerManager.initialize(CrawlerManager.java:294)
 at com.ibm.es.crawler.api.DSCrawlerAPI.start(DSCrawlerAPI.java:397)
 at com.ibm.es.control.communication.server.ComponentBaseW.startComponentInternal(ComponentBaseW.java:1439)
 at com.ibm.es.control.communication.server.ComponentBaseW.startComponent(ComponentBaseW.java:1396)
 at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
 at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:60)
 at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:37)
 at java.lang.reflect.Method.invoke(Method.java:611)
 at com.ibm.es.control.communication.server.ComponentBaseW.handleApiRequest(ComponentBaseW.java:308)
 at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
 at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:60)
 at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:37)
 at java.lang.reflect.Method.invoke(Method.java:611)
 at com.ibm.es.ccl.sessionwrapper.CallThread.run(CallThread.java:119)
 at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:452)
 at java.util.concurrent.FutureTask.run(FutureTask.java:149)
 at java.lang.Thread.run(Thread.java:736)
Caused by: java.lang.UnsupportedClassVersionError: JVMCFRE003 bad major version; class=MyCrawlerPlug-in, offset=6
 at java.lang.ClassLoader.defineClassImpl(Native Method)
 at java.lang.ClassLoader.defineClass(ClassLoader.java:275)
 at java.net.URLClassLoader.defineClass(URLClassLoader.java:540)
 at java.net.URLClassLoader.access$300(URLClassLoader.java:79)
 at java.net.URLClassLoader$ClassFinder.run(URLClassLoader.java:1038)
 at java.security.AccessController.doPrivileged(AccessController.java:284)
 at java.net.URLClassLoader.findClass(URLClassLoader.java:429)
 at java.lang.ClassLoader.loadClass(ClassLoader.java:413)
 at com.ibm.es.crawler.loader.DSClassLoaderManager.setClassInfo(DSClassLoaderManager.java:135)
 at com.ibm.es.crawler.loader.DSClassLoaderManager.setClassInfo(DSClassLoaderManager.java:91)
 at com.ibm.es.crawler.CrawlerManager.initialize(CrawlerManager.java:513)
 ... 21 more

2. Here is an example from the collection log. This example shows that the use of the package “sample” in the code without specifying it in the class name in the crawler configuration results in this exception because the class could not be found.

E  FFQQ0277E  2013/04/24 20:39:33.515 CDT  FFQQ0277E An exception was caught with the detail 'java.lang.NoClassDefFoundError: MyCrawlerPlug-in (wrong name: sample/MyCrawlerPlug-in)’ and a stack trace of 'java.lang.NoClassDefFoundError: MyCrawlerPlug-in (wrong name: sample/MyCrawlerPlug-in)'
 at java.lang.ClassLoader.defineClassImpl(Native Method)
 at java.lang.ClassLoader.defineClass(ClassLoader.java:275)
3. Look for the following information in the crawler plug-in audit log in the logs/audit folder. This shows the steps that occur when the plug-in is initialized and the plug-in object is instantiated.

<table>
<thead>
<tr>
<th>Code</th>
<th>Timestamp</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFQX0704I</td>
<td>2013/04/27 20:09:57.494 CDT</td>
<td>Initialize session: plug-in.WIN_50605.crawlerplug-in</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.572 CDT</td>
<td>called start</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.588 CDT</td>
<td>exit start</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.650 CDT</td>
<td>in setClassInformation className:MyCrawlerPlug-in clsPath:C:\plug-in\MyCrawlerPlug-in.jar</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.666 CDT</td>
<td>out setClassInformation rc:0</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.775 CDT</td>
<td>in isMetadataUsed</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.791 CDT</td>
<td>out isMetadataUsed</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:57.994 CDT</td>
<td>in getPort</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:58.056 CDT</td>
<td>requestPort:0</td>
</tr>
<tr>
<td>FFQX1208I</td>
<td>2013/04/27 20:09:58.072 CDT</td>
<td>out getPort rc:3627</td>
</tr>
</tbody>
</table>