BEAWebLogic
Portal ®

Bulkloader Guide

Contents

Using the Bulkloader

Using the Bulkloader

The BulkLoader is a command-line application that is capable of loading content and metadata into a configured virtual content repository. The BulkLoader scans the user defined file system hierarchy and loads all of the folders and files, along with their defined metadata into the specified repository. The BulkLoader supports all content types, such as `HTML`, `.jpg`, or `.gif`.

Note: You cannot use the BulkLoader to load content into a BEA repository if library services have been enabled for that repository.

This document contains information on the following subjects:

How the Bulkloader Finds Files

The following sequence describes how the BulkLoader locates files:

1. The BulkLoader starts by looking at the list of files and folders specified from the command line.

2. If no files or folder are specified, it uses only the content root specified by the `-d` option. It then loops over the list of files and folders.

3. If it finds a folder and `+recurse` is specified, then it stops.

4. If it finds a folder and recursion is turned on (the default or with `-recurse`), then the BulkLoader loops over the files and folders contained within that folder.

Note: If the file or folder is not an absolute path, then it is assumed to be relative to the content root specified by the `-d` option.

5. To determine if the BulkLoader should process a file or folder, it checks to see if the file is marked as a hidden file.

Note: If it is a hidden file (or folder) and the `+hidden` option was not specified, then the file or folder is ignored.

6. If the file or folder does not exist or is not readable by the user executing the BulkLoader, a warning is displayed and the file or folder is ignored.

7. If a folder is found then it is loaded as a hierarchy node into the virtual repository.

8. If a file is found then it is loaded as a content node into the virtual repository. The content node will be created with the node type specified in the md.properties. The primary property for the node type is where the file will be loaded to, so it must be a binary property.

9. If the loaded object is a folder and recursion is enabled, then the files and folders under the folder are retrieved by filtering against the `-match` and `-ignore` options.

Note: The `-match` and `-ignore` options only apply to files and folders not listed on the command line; in other words, they apply only to those found by recursing into a folder. The patterns specified with the `-match` and `-ignore` options (and the `-htmlPat` options, for that matter) should be DOS-style patterns: `'*'` matches any set of characters, `'?'` matches any one character. Sets of characters (for example, `[aceg]`) are not supported.

Note: Files with an extension matching the extension specified by `-mdext` (`.md.properties` by default) are always ignored.

How the Bulkloader Finds Metadata Files

1. As the BulkLoader finds files and folders, it attempts to load metadata property files. Whenever the BulkLoader encounters a folder to process, it looks for a file called dir.`<mdext>` where `<mdext>` is the extension specified by the `-mdext` option. The default filename it looks for is `dir.md.properties`. If this file exists and is readable by the user, the BulkLoader loads it as a Java-style properties file of name=value properties. If the folder is actually a subfolder entered because +recurse was not specified and the `+inheritProps` option is not specified, then the properties from `dir.md.properties` is added to the properties from the

parent folders. All files in the folder and all files in any subfolder gain these metadata properties.

2. When the BulkLoader finds a file which is to be included and loaded, it looks for a file whose name is the original filename appended with the `-mdext` extension. So, for example, if the file is called `image.gif`, the BulkLoader looks for a file called `image.gif.md.properties`. In order to load a file as a content node, the nodeType must be specified in the metadata. This may either be inherited from the folder md.properties, or specified in the file `md.properties`. The file along with the metadata to create must adhere to the rules defined in the specified nodeType. For example, if the node type specifies that a property named "author" is required, then there must be a metadata property named author with a value.

3. Next, if the file is an HTML file and the `+metaparse` option was not specified, then the BulkLoader parses the HTML, looking for `<meta>` tags and `<title>` tags. The BulkLoader determines if a file is an HTML file by using the filename patterns specified by the `-htmlPat` options. If no `-htmlPat` patterns are specified, then `*.htm` and `*.html` are used. The BulkLoader loads into the file's properties any `<meta>` tags that contain name and content values found anywhere in the file (not just in the HTML head section). Additionally, it pulls the title from the `<title></title>` and set it as "title".

4. Finally, the BulkLoader passes the file to the loadProperties method of each registered LoaderFilter (the `-filter` option). The LoaderFilter may assign additional metadata to the file. When the BulkLoader starts up, it looks for a `content\com\bea\content\loader\bulk` file in the classpath. From that, it looks for a `loader.defFilters` property. This is the colon-separated list of LoaderFilter class names the BulkLoader should always load. Unless that file is modified, the BulkLoader will load an ImageLoaderFilter, which will pull the width and height from `*.gif`, `*.jpg`, `*.png`, and `*.xbm` image files.

In summary, the BulkLoader gathers metadata for a document from the following sources (in this order):

- The parent folders `dir.md.properties` file.

- The file's folder's `dir.md.properties` file.

- The file's `.md.properties` file.

- If the file is an HTML file, then it uses `<meta>` tags.

- The list of LoaderFilters

Loading Content with the BulkLoader

To load content with the Bulkloader:

1. Make sure you have configured a repository and have created the appropriate types.

    Content Types define the available values for a given property, including whether it can contain multiple values. When you create a content type, you are creating a corresponding `md.properties` file for that piece of content.

    For example, If you create a content type for a specific piece of content, such as `logo.gif`, your metadata file is `logo.gif.md.properties`. if you create metatdata for a directory that contains several content files, your metadata file is `dir.md.properties`. The `dir.md.properties` file defines properties that can be inherited by children but can be overridden by `md.properties` files for specific content.

    In a later step, you will configure the individual properties (metadata) for your content types.

    For more information about configuring a repository and creating conent types, see the documetation for the Administration Portal.

2. Create a directory in your file system to store your loaded content.

Note: This directory should eventually contain only the content you want loaded because in a later step, you will point the Bulkloader to this directory, and it will load everything in it.

3. Add content to the directory you created in Step 2. You can load only one binary property per node, and it must be defined as the primary property. For example:
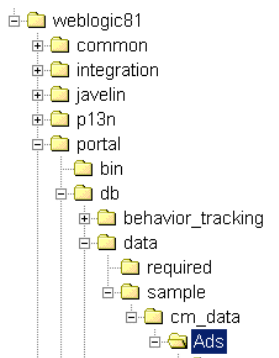
4. Map groups of content to Content types. (Create Metadata)

When you create the metadata you are mapping the properites of your content to the content type you created.

To create metadata manually:

- In a `dir.md.properties` file for a directory full of content, you simply point the file to the correct content type. For example

  a. If you created a content type called "ad", you need to open up the `dir.md.properties` file.

  b. Update it to reflect the following:

    ```
    nodeType=ad
    ```

  c. Save your changes.

- In an `md.properties` file for a specific piece of content, you will have to ad more specific metadata information. For example:

  a. For a piece of content called `logo.gif`, you need to open the corresponding `logo.gif.md.properties` file.

    ```
    height=65
    width=115
    adTargetUrl=
    adTargetContent=
    adWinClose=
    adWinTarget=
    adWinTitle=
    adClickTarget=

    adUseXhtml=
    adAltText=BEA Logo
    adMapName=
    adMap=
    adBorder=
    audience=internal
    ```

  b. Update the property values (metadata) for this piece of content.

  c. Save your changes.

5. Create the BulkLoader Script.

  a. In your WebLogic Platform 8.1 build directory, navigate to `Weblogic81b/portal/bin`, and open `load_cm_data.cmd`.

Listing 1   Bulkloader Script

```
@ECHO OFF
REM
######################################################################
REM #      (c) BEA SYSTEMS INC. All rights reserved
REM #
REM
######################################################################
SETLOCAL

SET PLATFORM_HOME=C:\bea\weblogic81
FOR %%i IN ("%PLATFORM_HOME%") DO SET PLATFORM_HOME=%%~fsi
SET PORTAL_HOME=%PLATFORM_HOME%\portal
SET P13N_HOME=%PLATFORM_HOME%\p13n

CALL %PLATFORM_HOME%\common\bin\commEnv.cmd

@rem
****************************************************************************
@rem Set any additional CLASSPATH information below
@rem
****************************************************************************
setCLASSPATH=%POINTBASE_CLASSPATH%;%WEBLOGIC_CLASSPATH%;%P13N_HOME%\lib\p1
3n_system.jar;%PORTAL_HOME%\lib\content.jar;%PORTAL_HOME%\lib\content_syst
em.jar;%CLASSPATH%

REM Set some defaults
if "%CM_DATA%"=="" set CM_DATA=..\db\data\sample\cm_data

%JAVA_HOME%\bin\java -classpath %CLASSPATH%
com.bea.content.loader.bulk.BulkLoader -verbose -repository "BEA
Repository" -application portalApp -d %CM_DATA% Ads%*

ENDLOCAL
```

b. Ensure that the Platform Home points to your WebLogic Server installation location.

```
SET PLATFORM_HOME=@PLATFORM_HOME
```

where @PLATFORM_HOME is the location of your WebLogic Server installation.

c. Point the Bulkloader to the directory you created in Step 2 and the metadata you created in Step 4.

d. Provide specific values for the following *required* items in the Bulkloader script:

Parameter in the Bulkloader Script	Description
-repository <repository name>	The name of the repository to run the loader against.
-application <app name>	The name of the application to run the loader against.
-url <wls url>	The WebLogic Server instance host where the content manager is running.
-user <principal username>	The username for the principal permitted to access the Loader EJB resource.
-password <principal password>	The password for the principal permitted to access the LoaderEJB resource.
-d <dir>	Specifies the baseDirectory to which non-absolute paths will be relative.

e. (Optional) Update the bulkloader scripts with any of the following paramemters:

Parameter in the Bulkloader Script	Description
-recurse	Recurse into directories [default].
+recurse	Don't recurse into directories.
-metaparse	Parse HTML files for META tags [default].
+metaparse	Don't parse HTML files for META tags.
-hidden	Specify to ignore hidden files and directories [default].
+hidden	Specify to include hidden files and directories.
-inheritProps	Specify to have metadata properties be inherited when recursing [default].

Parameter in the Bulkloader Script	Description
+inheritProps	Specify to have metadata properties not be inherited when recursing.
-ignoreErrors	Ignore any errors while loading a document (errors will still be reported)
+ignoreErrors	Stop processing on any error [default].
-htmlPat <pattern>	Specifies a pattern for determining which files are HTML files for determining whether to do the META tag parse. This can be specified multiple times. If none are specifiec, '*htm' and '*.html' are used.
-encoding <encoding>	Specifies the file encoding to use. See your JDK documentation for the valid encoding names. Defaults to the system's default file encoding.
-match <pattern>	Specifies a file pattern the BulkLoader should include. This can be specified multiple times. If non are specified, all files and directories are included.
-ignore <pattern>	Specifies a file pattern the BulkLoader should not include. This can be specified multiple times.
-mdext <ext>	Specifies the file name extension for metadata property files. The value should start with a ".". This defaults to .md.properties.
-filter <filter class>	Specifies the class name of a LoaderFilter to run files through. This can be specified multiple times to add to the list of LoaderFilters.
-filters	Clears the current list of LoaderFilters (including the default filters.)
--	Everything after this is considered a file or directory.

6. Save your changes to the Bulkloader script.

7. Run the script by double clicking it in your file system. Your content will be loaded.