DICOM Import and Export Plugins

covers software version 1.0.0

by Thomas Hacklaender, MD, MSc (hacklaender@iftm.de)

IFTM Institut für Telematik in der Medizin GmbH

This package contains a set of plugins for the ImageJ framework (http://rsb.info.nih.gov/ij/) to import and export DICOM images. In contrast to the ImageJ build-in DICOM functionality it is possible to read and write to the local filesystem and to DICOM file-sets, i.e. a DICOMDIR. That means, that for example DICOM CD's may be read and written. As an option the imported images may contain all the DICOM metadata either as Java property-strings and/or binary data. These metadata can be used by other plugins for image calculation.

The package contains four plugins:

?? Dcm_Import: This plugin imports DICOM compatible images to ImageJ.

?? Dcm_Export: This plugin exports all ImagesJ images (besides 32-bit float) as DICOM Secondary Capture images.

?? Dcm_PropertyLister: This plugin lists the properties assigned to an ImageJ image.

?? Dcm_Inspector: This is a plugin and a stand-alone program. Its purpose is to look inside a DICOM image. It also allows to convert a DICOM image to a XML representation and to process that by a XSL processor.

The package was developed under the GNU General Public License. Library parts are under the GNU Lesser General Public License. See the source code for details.

The package is based on the *dcm4che* DICOM library which includes network and media functions also. It was developed by Gunter Zeilinger. The homepage of this project is http://sourceforge.net/projects/dcm4che/ .

Table of Contents

1 Getting Started 3
1.1 Release Notes 3
1.2 Installation 3
1.2.1 Install the JRE 1.4 3
1.2.2 Install ImageJ 3
1.2.3 Install the plugins 3
1.2.4 Modify the classpath 4
2 Description 5
2.1 General Information 5
2.2 The Property File dcmie.properties 5
2.3 DICOM Metadata as Properties of an ImagePlus 6
2.3.1 Binary Properties 6
2.3.2 String Properties 7
2.4 Accessing DICOM File-Sets 7
3 Plugins 8
3.1 Dcm_Import 8
3.2 Dcm_Export 8
3.3 Dcm_PropertyLister 11
3.4 Dcm_Inspector 11
4 References 13

1 Getting Started

1.1 Release Notes

The package was developed under Java 1.4 and ImageJ 1.26t on a PII 400 with 256 MB RAM. It requires Java 1.4 as a runtime environment.

1.2 Installation

1.2.1 Install the JRE 1.4

If you do not have a recent version of Java installed on your system, download and install the JRE from http://www.javasoft.com.

1.2.2 Install ImageJ

Download and install ImageJ as described on the ImageJ website: http://rsb.info.nih.gov/ij/ .

The directory under which you install ImageJ will be referred to as `<IMAGEJ_HOME>` in the remainder of this document.

1.2.3 Install the plugins

The package is archived in the ZIP file `dcmie.zip` . Unzip the file to `<IMAGEJ_HOME>`. You will receive the following files and directories:

File/Directory	Description
Dcm	The directory containing the four plugins.
dcmie.properties	A property file containing general settings for the plugins.
inspector.properties	A property file containing general settings for the Inspector as a stand-alone program. The properties are a subset of the `dcmie.properties`.
/doc	The directory containing documentation files: `default_metadata.properties` - An optional property file containing DICOM metadata for export. `manual.pdf` – This manual as PDF fille. `manual.rtf` – This manual as Rich Text File. `MetadataToHTML.xsl` – The build in XSL file to convert the XML representation to a HTML representation. `NativeMetadataFormat.dtd` – The DTD of the XML representation of a DICOM object.
gpl.txt	The GNU General Public License.
lgpl.txt	GNU Lesser General Public License.
/lib	The directory containing necessary library files. All libraries are developed under the GNU GPL or LGPL license. `dcm4che.jar` - The low-level DICOM library (version 1.0.3) `dcmie.jar` - The library for the Dcm-package `getopt.jar` - A library for command-line options (version 1.0.9) `links.txt` - Links to obtain additional information and/or the source code for these libraries. `log4j.jar` - The library for the logger used by dcm4che (version 1.2.3).

log4j.properties	Property file for the log4j logger used by dcm4che.
Run_ImageJ.bat	A batch file to start ImageJ in a Windows environment.
SMPTE.dcm	A sample DICOM image.
src.zip	The source code of the package.

Move the whole directory `Dcm` into the folder `<IMAGEJ_HOME>/plugins` .

When you start ImageJ you will find the newly installed plugins in the submenu `Plugins/Dcm`.

1.2.4 Modify the classpath

If you have a standard ImageJ installation your start-script should look like:

```
javaw.exe -Xms96m -Xmx96m –cp ij.jar ij.ImageJ
```

It states, that the Java VM should look into the file `ij.jar` for necessary files. For proper operation of the plugins you must insert the file **`.\lib\dcmie.jar`** into the classpath. After that the script should look like:

```
javaw.exe -Xms96m -Xmx96m –cp .\lib\dcmie.jar;ij.jar ij.ImageJ
```

It is not necessary to include the other `.jar` files in the directory `<IMAGEJ_HOME>/lib` to the classpath. This is automatically done during runtime by the `MANIFEST` in the `dcmie.jar` file.

2 Description

2.1 General Information

Normally the plugins are called using the `Plugins` menu of ImageJ. But it is also possible to call them by an internal program or macro. In such a situation it is sometimes not desirable to have a graphical user interface. Therefore, the plugins may be called with arguments. These should be POSIX conform (see http://java.sun.com/docs/books/tutorial/essential/attributes/cmdLineArgs.html).

Some of them require the specification of a file name. In this case URI's are used. For a detailed description see the API-Doc of the `URI` class in Java 1.4. For Windows-OS the absolute URI `file:/c:/user/tom/foo.txt` describes the file `C:\user\tom\foo.txt`. Relative URI's, e.g. without the `file:` schema-prefix, are relative to the user-directory, given by the system property `user.dir`. For example: If the `user.dir` is `C:\user\tom\` and the relative URI is `/abc/foo.txt` the referenced file is `C:\user\tom\abc\foo.txt`. The abbreviations "`.`" for the current and "`..`" for the upper directory are valid to form a relative URI.

2.2 The Property File `dcmie.properties`

The default property file is as follows:

```
# Some properties use a <file-uri> as a operation system independent way for
# file references. The format of a such a URI is (see also the API-Doc of the
# URI class):
# <file-uri>  For Windows-OS the absolute URI "file:/c:/user/tom/foo.txt"
#             describes the file "C:\\user\\tom\\foo.txt".
#             Relative URI's, e.g. without the "file:" schema-prefix, are
#             relativ to the user-directory, given by the system property
#             "user.dir". For example: If the user.dir is "C:\\user\\tom\\"
#             and the relative URI is "/abc/foo.txt" the referenced file is
#             "C:\\user\\tom\\abc\\foo.txt". The abbreviations "." for the
#             current and ".." for the upper directory are valid to form a
#             relative URI.


# The <file-uri> of the destination file or directory for export operations.
# Default value : "./" corresponds to <user.dir>
dcmie.export.file=./SecondaryCapture.dcm

# If the filesystem should be the default destination for export operations
# the value is true. If destination should be a DICOMDIR the value is false.
# Default value : true
dcmie.export.filesystem=true

# The <file-uri> of the property file describing general metadata valid for
# all slices in an ImagePlus. The data overwrite the metadata of the image
# and the default values.
# Default value : <null>
# dcmie.export.metadata.general = ./general.properties

# The <file-uri> of the property file describing a mask: Only attributes
# defined in this file will be included to the metadata of the exported image.
# All metadata necessary to construct the Image Information-Entity need not
# to be declared in this file. If this optionis not set or the property file
# could not be opened, all defined attributes were written to the exported image.
# Default value : <null>
# dcmie.export.metadata.mask = ./mask.properties

# True, if the writer should consider the metadata of each slice of the ImagePlus.
# In this case the image metadata overwrite the corresponding default values.
# Default value : false
dcmie.export.metadata.useimage = false
```

```
# The <file-uri> of the source file or directory for import operations.
# Default value :  "./" corresponds to <user.dir>
dcmie.import.file=./DICOMDIR

# If the filesystem should be the default source for import operations
# the value is true. If the source should be a DICOMDIR the value is false.
# Default value : true
dcmie.import.filesystem=false

# If the DcmImportPanel should display "dcmie.import.ij.*" properties.
# Default value : true
dcmie.import.ij.mode=true

# If the imported ImagePlus should be displayed the value is true.
# Default value : true
dcmie.import.ij.image.show=true

# If metadata should be appended as binary properties the ImagePlus the value
# is true. The format of the properties is:
# Key:   dcm4che.binary<br>
# Value: The value of the element as a Datset[]. Each slice in the ImagePlus
#        has one entry in the array. For an ImagePlus with only one image
#        the property is a Dataset[1].
# Note:  Slices are numbered starting with 1, while the corresponding array-
#        indices start with 0.
# Default value : false
dcmie.import.ij.metadata.binary=false

# true, if only the metadata of the first image of a stack should be included
# Default value : false
dcmie.import.ij.metadata.onlyfirst=false

# If metadata should be appended as String properties the ImagePlus the value
# is true. The format of the properties is:
# Key:            dcm4che.string.[slice_number].[tag]
# Value:          The value of the element as a String. If the value of the
#                 element is empty the string "" is returned. Note: In the
#                 current version of dcm4che (2002.5.26) elements with
#                 VR = UN, OB, OW and SQ are ignored. The library does not
#                 convert their binary values to a string-representation.
# [slice_number]: For each slice in a ImagePlus a seperate set of properties
#                 is given. The slice number starts with 1. For an ImagePlus
#                 with only one image [slice_numer] is 1.
# [tag]:          The tag-number of the element. It is a 8 character long
#                 hexadecimal number with the first 4 digits corresponding to
#                 the group-number and the last 4 digits corresponding to the
#                 element number.
# Default value : false
dcmie.import.ij.metadata.string=true
```

2.3 DICOM Metadata as Properties of an ImagePlus

As an option metadata may be assigned to an ImagePlus as properties. Keep in mind that an ImagePlus may contain multiple images as slices, but only the ImagePlus as a whole may contain properties. Two kinds of properties are defined:

2.3.1 Binary Properties

For each slice an instance of the `Dataset` class is included. The description of the class can be found in the API-documentation of the dcm4che package. All `Dataset`'s are integrated in an array of `Dataset[]` which is added to the ImagePlus. In this case the property-key is `dcm4che.binary`. Keep in mind that the array index starts at `0` and the slice numbering starts at 1. Therefore the metadata for the first slice can be found in `Dataset[0]`!

2.3.2 String Properties

The key of these properties is `dcm4che.string.<slice_number>.<tag>`.

`<slice_number>` is the decimal number of the slice the property belongs to.

`<tag>` is an 8 character long hexadecimal string describing the attribute-tag. The first 4 character are the group-number, the last 4 character the element-number. For example to reference the Instance Number of the third slide the key should be `dcm4che.string.3.00200013`.

The value of the property is always a string that should *not* be surrounded by quotes. If the attribute should not have a value keep the string empty. For attributes with multiple values (e.g. Value Multiplicity > 1) they should be separated by the string “\\”. Use always two backslashes! Only one backslash would be interpreted as a Java escape character sequence. Attributes with the Value Representations OB, OW, OF, SQ, UN and NONE are ignored.

As an example the Image Type of the first slice should be written as:

```
dcm4che.string.1.00080008 = ORIGINAL\PRIMARY
```

2.4 Accessing DICOM File-Sets

DICOM File-Sets are defined in PS 3.10 “Media Storage and File Format for Media Interchange” and PS 3.3 Annex F “Basic Directory Information Object Definition”. Keep in mind that every file-set must contain exactly one file with name “DICOMDIR” (in capital letters).

If the DICOMDIR file is in the directory `<DICOMDIR_HOME>` the Dcm_Export plugin stores a DICOM image according its PatientName, StudyID, SeriesNumber and InstanceNumber at `<DICOMDIR_HOME>/<PatientName>/<StudyID>/<SeriesNumber>/<InstanceNumber>`

The directory names have a maximum number of 8 characters containing only capital letters and numbers. If a filename is not unique the InstanceNumber is consecutively replaced by hexadecimal random numbers until the name is unique.

3 Plugins

3.1 Dcm_Import

This plugin supports the following arguments:

Option	Description
-p <file-uri>	The name of an optional property file. If this option is not present the default `<filename> = "dcmie.properties"` is chosen. If that file is not found in the `user.directory` the default values of the class `DcmieParam` are taken.
-f <file-uri>	The name of a DICOM file to import. If this option is given, the GUI is not displayed.

3.2 Dcm_Export

This plugin supports the following arguments:

Option	Description
-p <file-uri>	The name of an optional property file. If this option is not present the default `<filename> = "dcmie.properties"` is chosen. If that file is not found in the `user.directory` the default values of the class `DcmieParam` were taken.
-h	Do not show the GUI.
-t <type>	Run in test-mode. Only useful during development. See source code for details.

The Dcm_Export plugin maps ImageJ image-types according the following table to the Photometric Interpretation of the resulting DICOM image:

ImageJ Type	Photometric Interpretation
GRAY8	MONOCHROME2
GRAY16	MONOCHROME2
GRAY32	- not supported -
COLOR_256	RGB
COLOR_RGB	RGB

If the ImageJ image contains more than one slice all slices are written with the slice number as InstanceNumber and the same PatientName, StudyID and SeriesNumber. If the export is to the local filesystem, the given filename is postfixed with the 3-character long InstanceNumber starting with the number given in the metadata.

Under "normal" conditions, i.e. if the settings in the `dsmie.properties` file are not changed, a graphical user interface with a table of default values is presented. Even if you do not change these values you receive an individual and valid Secondary Capture image.

But it is also possible to customize this behaviour:

The Secondary Capture Image IOD (Information Object Definition) is defined in PS 3.3 - A.8. as follows:

IE	Module Name	Reference PS 3.3	Option
Patient IE	Patient Module	C.7.1.1	Mandatory
Study IE	General Study Module	C.7.2.1	Mandatory
Study IE	Patient Study Module	C.7.2.2	User Option
Series IE	General Series Module	C.7.3.1	Mandatory
Equipment IE	General Equipment Module	C.7.5.1	User Option
Equipment IE	SC Equipment Module	C.8.6.1	Mandatory
Image IE	General Image Module	C.7.6.1	Mandatory
Image IE	Image Pixel Module	C.7.6.3	Mandatory
Image IE	SC Image Module	C.8.6.2	Mandatory
Image IE	Overlay Plane Module	C.9.2	User Option
Image IE	Modality LUT Module	C.11.1	User Option
Image IE	VOI LUT Module	C.11.2	User Option
Image IE	SOP Common Module	C.12.1	Mandatory

The metadata of the Image IE (Information Entity) are derived from the image that should be exported directly. The user has no influence on this data under any circumstances. All other metadata may be changed by the user.

As default values all attributes are included to the metadata, as long as their type is 1 (Required w/ value) or 2 (Required w/o value). For the Equipment Modules some type 3 (Optional) attributes are included also. This leads to the following list of attributes:

Attribute Name	Property name	Value)* values are generated during runtime
SOP Class UID	dcm4che.string.1.00080016	1.2.840.10008.5.1.4.1.1.7
SOP Instance UID	dcm4che.string.1.00080018	1.2.40.0.13.1.1.192.168.0.2.20020626183926558.32773)*
Study Date	dcm4che.string.1.00080020	20020626)*
Content Date	dcm4che.string.1.00080023	20020626)*
Study Time	dcm4che.string.1.00080030	183926.598)*
Content Time	dcm4che.string.1.00080033	183926.598)*
Accession Number	dcm4che.string.1.00080050	0
Modality	dcm4che.string.1.00080060	OT
Conversion Type	dcm4che.string.1.00080064	WSD
Referring Physician's Name	dcm4che.string.1.00080090	ReferringPhysicianName
Patient's Name	dcm4che.string.1.00100010	PatientName
Patient ID	dcm4che.string.1.00100020	1025109564004
Patient's Birth Date	dcm4che.string.1.00100030	19501031
Patient's Sex	dcm4che.string.1.00100040	O
Secondary Capture Device ID	dcm4che.string.1.00181010	<no value>
Date of Secondary Capture	dcm4che.string.1.00181012	20020626)*
Time of Secondary Capture	dcm4che.string.1.00181014	183926.598)*
Secondary Capture Device Manufacturer	dcm4che.string.1.00181016	dcm4cheri
Secondary Capture Device Software Version(s)	dcm4che.string.1.00181019	1.0

Study Instance UID	dcm4che.string.1.0020000d	1.2.40.0.13.1.1.192.168.0.2.20020626183926558.32771)*
Series Instance UID	dcm4che.string.1.0020000e	1.2.40.0.13.1.1.192.168.0.2.20020626183926558.32772)*
Study ID	dcm4che.string.1.00200010	1
Series Number	dcm4che.string.1.00200011	1
Instance Number	dcm4che.string.1.00200013	1
Patient Orientation	dcm4che.string.1.00200020	<no value>
Laterality	dcm4che.string.1.00200060	<no value>

The metadata included in the Secondary Capture image are composed of different sources. The following table lists the sources and the sequence of inclusion to the resulting metadata. If not otherwise stated an attribute overwrites the value of the same attribute defined in a step before.

Step	Source	Description
1	Default metadata as described above	
2	Metadata found as a binary-property for the first slice in the image to export	Only if the property `dcmie.export.metadata.useimage = true`
3	Metadata found as string-properties for the first slice in the image to export	Only if the property `dcmie.export.metadata.useimage = true`
4	The metadata found in the property file set with the property `dcmie.export.metadata.general`	The file should contain attributes coded as string-properties (see above)
5	A mask given in the property file `dcmie.export.metadata.mask`	The file should contain attributes coded as string-properties (see above). If a file is given, only attributes listed in this file are used in the following. The value of the listed attributes has no meaning, but must be valid for that attribute.
6		The metadata are presented to the user in a table. The user may alter the values.
7a	If the ImagePlus contains only one slice	The metadata are written together with the metadata of the Image IE to the Secondary Capture image.
7b	If the ImagePlus contains multiple slices the following happens for each slice:	
8	Start with empty metadata	
9	Metadata found as a binary-property for the slice in the image to export	Only if the property `dcmie.export.metadata.useimage = true`
10	Metadata found as string-properties for the slice in the image to export	Only if the property `dcmie.export.metadata.useimage = true`
11	The metadata created in step 6	
12	For each slice the InstanceNumber is incre-mented by one, starting with the number given in step 6.	
13	A mask given in the property file `dcmie.export.metadata.mask`	The file should contain attributes coded as string-properties (see above). If a file is given, only attributes listed in this file are used in the following. The value of the listed attributes has no meaning, but must be valid for that attribute.
14		The metadata are written together with the metadata of the Image IE to the Secondary Capture image.

The user may add, delete or modify the metadata string-properties during runtime.

3.3 Dcm_PropertyLister

This plugin supports the following arguments:

Option	Description
-t <type>	Run in test-mode. Only useful during development. See source code for details.

The plugin lists all properties found in the active ImagePlus. If the key of the property starts with `dcm4che.string` its contents is listed as a string-property described above.

3.4 Dcm_Inspector

This plugin supports the following arguments:

Option	Description
-p <file-uri>	The name of an optional property file. If this option is not present the default `<filename> = "dcmie.properties"` is chosen. If that file is not found in the `user.directory` the default values of the class `DcmieParam` are taken.

The Dcm_Inspector may also be called as a stand alone program on its own Virtual Machine. The start script should than look like this:

```
javaw.exe -Xms96m -Xmx96m -cp .\lib\dcmie.jar;ij.jar;.\plugins\DcmIE
Dcm_Inspector
```

To use a different set of properties with the stand-alone version add the appropriate option:

```
javaw.exe -Xms96m -Xmx96m -cp .\lib\dcmie.jar;ij.jar;.\plugins\DcmIE
Dcm_Inspector –p inspector.properties
```

One of the options of this plugin is the conversion of a DICOM file into a XML representation. The DTD (Document Type Definition) can be found in the `doc` folder. To convert this XML representation into a HTML file there is XSL stylesheet build in, which can also be found in the `doc` folder.

The following listing shows an (not complete) example of a the XML represenation of a DICOM file:

```
<?xml version="1.0" encoding="UTF-8"?>

<dicomfile>

  <filemetainfo preamble="00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\
                         00\00\00\00\00\00\00\00\00\00\00\00\00\00\00\00">
    <elm tag="00020000" vr="UL" pos="132" name="Group Length">
      <val vm="1" len="4" data="176"/>
    </elm>
    <elm tag="00020001" vr="OB" pos="144" name="File Meta Information Version">
      <val vm="1" len="2" data="00\01"/>
    </elm>
    <elm tag="00020013" vr="SH" pos="298" name="Implementation Version Name">
      <val vm="1" len="14" data="DCM4CHE12JUN02"/>
    </elm>
  </filemetainfo>

  <dataset>
     <elm tag="00041130" vr="CS" name="File-set ID">
       <val vm="1" len="8" data="FileSet "/>
     </elm>
     <elm tag="00041220" vr="SQ" name="Directory Record Sequence">
       <seq len="4116">
         <item id="1" pos="386" len="126">
           <elm tag="00041400" vr="UL" name="Offset of the Next Directory Record">
             <val vm="1" len="4" data="0"/>
           </elm>
           <elm tag="00041410" vr="US" name="Record In-use Flag">
             <val vm="1" len="2" data="65535"/>
           </elm>
         </item>
         <item id="2" pos="520" len="226">
           <elm tag="00041400" vr="UL" name="Offset of the Next Directory Record">
              <val vm="1" len="4" data="0"/>
           </elm>
         </item>
       </seq>
     </elm>
    <elm tag="00212311" vr="UN">
      <val vm="1" len="4" data="00\00\00\00"/>
    </elm>
    <elm tag="7fe00010" vr="OW" name="Pixel Data">
      <val vm="1" len="131072" data="0000\0000\0000\0000\0000/>
    </elm>
  </dataset>

</dicomfile>
```

4 References

The ImageJ framework: http://rsb.info.nih.gov/ij/

The dcm4che project homepage: http://sourceforge.net/projects/dcm4che/

The DICOM normative text: http://www.dclunie.com/