



## **Keyword Report Tool v.0.1.0**

**for the Planetary Data System**



# Table of Contents

---

<b>1</b>	<b>Keyword Report Tool Guide</b>	
1.1	Overview .....	1
1.2	Release Notes .....	2
1.3	Installation .....	3
1.4	Operation .....	5
1.5	Appendix A - UNIX Setup Options .....	14
1.6	Appendix B - Windows Setup Options .....	15



## 1.1 Overview

---

### About Keyword Report Tool

The Keyword Report Tool (KRTool) is intended to search for and report a list of statements, in Planetary Data System (PDS) labeled data products, that match a user-supplied list of keywords.

Please send comments, change requests and bug reports to the [PDS Operator](mailto:pds_operator@jpl.nasa.gov) at [pds\\_operator@jpl.nasa.gov](mailto:pds_operator@jpl.nasa.gov).

## 1.2 Release Notes

---

### Release Notes

The purpose of this section is to provide a description of a Keyword Report Tool release including any impact that the new or modified capabilities will have on the Discipline Nodes or the PDS user community. A somewhat itemized list of changes for each release can be found on the [Release Changes](#) page. If viewing this document in PDF form, see the appendix for details.

### Release 0.1.0

This is the first release of the Keyword Report Tool, intended for beta testing. This release includes the capability to generate a report of specified keywords found in a specified set of labels.

## 1.3 Installation

---

### Installation

This section describes how to install the Keyword Report Tool (KRTool) contained in the *krtool* package. The following topics can be found in this section:

- [System Requirements](#)
- [Unpacking the KRTool Package](#)

### System Requirements

The Keyword Report Tool was developed using Java and will run on any platform with a supported Java Runtime Environment (JRE). The tool was specifically developed under Sun Java version 1.5, so the tool will execute correctly under versions 1.5 or 1.6.

Since the tool was developed using Sun's Java, this is the preferred Java environment for operation. The Sun Java package can be obtained from the [Sun Java](#) web site. Other Java environments are relatively compatible with Sun's Java.

### Unpacking the KRTool Package

Download the *krtool* package from the [Tools & Documentation](#) web page. The binary distribution is available in identical zip or tar/gzip packages. Unpack the selected binary distribution file with one of the following commands:

```
% unzip krtool-0.1.0-bin.zip
or
% tar -xvzf krtool-0.1.0-bin.tar.gz
```

Note: Depending on the platform, the native version of *tar* may produce an error when attempting to unpack the distribution file because many of the file paths are greater than 100 characters. If available, the GNU version of *tar* will resolve this problem. If that is not available or cannot be installed, the zipped package will work just fine in a UNIX environment.

The commands above result in the creation of the *krtool-0.1.0* directory with the following directory structure:

- **README.txt**

A README file directing the user to the available documentation for the project.

- **LICENSE.txt**

The copyright notice from the [California Institute of Technology](#) detailing the restrictions regarding the use and distribution of this software. Although the license is strictly worded, the software has been classified as Technology and Software Publicly Available (TSPA) and is available for *anyone* to download and use.

- **bin/**

This directory contains batch and shell scripts for executing the tool.

- **doc/**

This document directory contains a local web site with the Keyword Report Tool Guide, javadoc, unit test results and other configuration management related information. Just point your favorite browser to the *index.html* file in this directory.

- **lib/**

This directory contains the dependent jar files for the tool along with the executable jar file (krtool-0.1.0.jar) containing the Keyword Report Tool software.



## 1.4 Operation

---

### Operation

The function of the Keyword Report Tool (KRTool) is to find data elements, supplied by the user, in a PDS compliant (or "valid") label.

This section describes how to use KRTool in order to find these desired data elements in a PDS label. The following topics can be found in this section:

- [Tool Setup](#)
- [Tool Execution](#)

Note: The command-line examples in this section have been broken into multiple lines for readability. The commands should be reassembled into a single line prior to execution.

### Tool Setup

In order to execute KRTool, the user's environment must first be configured appropriately. This section describes how to setup the user environment on UNIX-based and Windows machines.

#### UNIX-Based Setup

This section details the environment setup for UNIX-based machines. The preferred method is to specify the shell script, *KRTool*, on the command-line. Setting the *PATH* environment variable to the location of the script, enables the shell script to be executed from any location on the user's machine.

The following command demonstrates how to set the *PATH* environment variable, by appending to its current setting:

```
% setenv PATH ${PATH}:%HOME/krtool-0.1.0/bin
```

The tool can now be executed via the shell script as demonstrated in the following example:

```
% KRTool <command-line arguments>
```

Additional methods for setting up a UNIX-based environment can be found in the [UNIX Setup Options](#) section. If viewing this document in PDF form, see the appendix for details.

## Windows Setup

This section details the environment setup for Windows machines. The preferred method is to specify the batch file, *KRTool.bat*, on the command-line. Setting the *PATH* environment variable to the location of the file, enables the batch file to be executed from any location on the user's machine.

The following command demonstrates how to set the *PATH* environment variable, by appending to its current setting:

```
C:\> set PATH = %PATH%;C:\krtool-0.1.0\bin
```

The tool can now be executed via the batch file as demonstrated in the following example:

```
C:\> KRTool <command-line arguments>
```

Additional methods for setting up a Windows environment can be found in the [Windows Setup Options](#) section. If viewing this document in PDF form, see the appendix for details.

## Tool Execution

KRTool can be executed in various ways. This section describes how to run the tool, as well as its behaviors and caveats.

### Command-Line Options

The following table contains command-line options available to KRTool:

Flag	Description
-t, --target <labels,URLs,dirs>	Explicitly specify the targets (label files, directories, and URLs). Targets can also be specified implicitly (example: KRTool label.lbl). For more details on target specification, see the <a href="#">Specifying Targets</a> section.
-k, --keywords <identifiers>	Specify keywords to find in a PDS label. This is a required flag.
-r, --report-file <file>	Specify the report file name. Default is to write to the standard out if this flag option is not specified on the command-line.

Flag	Description
-c, --config <file>	Specify a configuration file to set the default values.
-L, --local	Do not perform directory recursion when given a target directory.
-e, --regexp <expressions>	Specify file patterns to look for in a target directory. Separate each pattern with a comma character. Each pattern should be surrounded in quotes ("*.LBL", "*.FMT") to avoid having the system shell mistakenly interpreting them (This has been seen on non-Windows systems). Pattern matching is case-insensitive in Windows, but case-sensitive for other systems.
-X, --ignore-file <expressions>	Specify file patterns to ignore in a target directory. Separate each pattern with a comma. Each pattern should be surrounded in quotes ("*.TXT", "*.TAB") to avoid having the system shell mistakenly interpreting them (This has been seen on non-Windows systems). Pattern matching is case-insensitive in Windows, but case-sensitive for other systems.
-D, --ignore-dir <expressions>	Specify directory patterns to ignore in a target directory. Separate each pattern with a comma. Each pattern should be surrounded in quotes ("LABEL", "DOC") to avoid having the system shell mistakenly interpreting them (This has been seen on non-Windows systems). Pattern matching is case-insensitive in Windows, but case-sensitive for other systems.
-h, --help	Display KRTTool usage.
-V, --version	Display KRTTool version.

## Running KRTTool

This section demonstrates some of the ways that the tool can be run using the command-line option flags:

- Searching for a Single Keyword
- Searching for Multiple Keywords
- Writing a Human-Readable Report to File
- Searching in Files with a Specific File Pattern
- Ignoring Files with a Specific File Pattern
- Ignoring Sub-Directories while Traversing a Target Directory
- Changing Tool Behaviors Using a Configuration File

### Searching for a Single Keyword

The following command demonstrates how to search for a single keyword, *DATA\_SET\_ID*, in a PDS data product label:

```
% KRTTool LABEL.LBL -k DATA_SET_ID
```

### ***Searching for Multiple Keywords***

The following command demonstrates how to search for multiple keywords, *DATA\_SET\_ID* and *TARGET\_NAME*, in a PDS data product label:

```
% KRTool LABEL.LBL -k DATA_SET_ID, TARGET_NAME
```

Before a search for keywords can be performed, the file being passed into the tool must be a PDS compliant label. If there is a syntactical error or if the tool determines that the file is not a PDS label, the user will be notified in the output report. The PDS Validation Tool (VTool) can then be used to find out more details about the errors found in the file.

### ***Writing a Human-Readable Report to File***

In the first two examples above, the output report is written to standard out. The following command demonstrates how to write the report to a file named *report.txt*:

```
% KRTool LABEL.LBL -k DATA_SET_ID -r report.txt
```

### ***Searching in Files with a Specific File Pattern***

The following command demonstrates how to search for keywords, *DATA\_SET\_ID* and *TARGET\_NAME*, in a target directory *\$HOME/DIR*. Additionally, the search will only be performed on file names that end in ".LBL" or begins with the letters "MER":

```
% KRTool $HOME/DIR -k DATA_SET_ID, TARGET_NAME -e "*.LBL", "MER*"
```

### ***Ignoring Files with a Specific File Pattern***

The following command demonstrates how to search for keywords, *DATA\_SET\_ID* and *TARGET\_NAME*, in a target directory *\$HOME/DIR*. Additionally, the search will not be performed in any file names that end in ".IMG" or ".TAB".

```
% KRTool $HOME/DIR -k DATA_SET_ID, TARGET_NAME -X "*.IMG", "*.TAB"
```

### ***Ignoring Sub-Directories while Traversing a Target Directory***

By default, the tool will recursively traverse down a directory tree when a target directory is specified.

The *local* flag can be used to disable directory recursion. The following command demonstrates how to search for keywords, *DATA\_SET\_ID* and *TARGET\_NAME*, in a directory *\$HOME/DIR*, where directory recursion is turned off.

```
% KRTool $HOME/DIR -k DATA_SET_ID, TARGET_NAME -L
```

The *ignore directories* flag can also be used to tell the tools which sub-directories to ignore while traversing the target directory. The following command demonstrates how to search for keywords, *DATA\_SET\_ID* and *TARGET\_NAME*, in a target directory *\$HOME/DIR*. Additionally, the search will not be performed in any directories named "EXTRAS" or "LABEL".

```
% KRTool $HOME/DIR -k DATA_SET_ID, TARGET_NAME -D "EXTRAS", "LABEL"
```

### ***Changing Tool Behaviors Using a Configuration File***

A configuration file can be passed to the tool to change the default behaviors of the tool. This provides a way to use the tool with a single flag. For more details on how to setup the configuration file see the [Using a Configuration File](#) section.

The following command demonstrates how to run the tool using a configuration file:

```
% KRTool -c config.cfg
```

### **Specifying Targets**

Targets are validated in the order in which they are specified on the command-line. They can be specified implicitly and explicitly.

To specify targets implicitly, it is best to specify them first on the command-line before any other option flags.

The following command demonstrates how to search for keyword *DATA\_SET\_ID* in a single data product label, specified implicitly:

```
% KRTool LABEL.LBL -k DATA_SET_ID
```

The following command demonstrates how to search for keyword *DATA\_SET\_ID* in multiple data product labels, both specified implicitly:

```
% KRTool LABEL.LBL, $HOME/DIR -k DATA_SET_ID
```

**Implicit targets should not be specified after option flags that allow multiple arguments (see example below). Unexpected results will occur.**

```
% KRTool -k DATA_SET_ID LABEL.LBL
```

In this example, KRTool will inadvertently treat the implicit target *LABEL.LBL* as a keyword to search for in a file.

Targets can be specified both implicitly and explicitly at the same time. Targets specified implicitly are validated first, followed by those that are specified explicitly with the target flag.

The following command demonstrates how to search for keyword *DATA\_SET\_ID* in multiple data product labels, specified both implicitly and explicitly.

```
% KRTool LABEL1.LBL, LABEL2.LBL -k DATA_SET_ID -t LABEL3.LBL, $HOME/DIR
```

In this example, *LABEL1.LBL* and *LABEL2.LBL* will be searched first, then *LABEL3.LBL* and the labels in *\$HOME/DIR* will be searched next.

### Using a Configuration File

A configuration file is used to set the default behaviors of the tool. It consists of a text file made up of keyword/value pairs. The configuration file follows the syntax of the stream parsed by the Java `Properties.load(java.io.InputStream)` method.

Blank lines and lines which begin with the hash character "#" are ignored.

Values may be separated on different lines if a backslash is placed at the end of the line that continues below.

Escape sequences for special characters like a line feed, a tabulation or a unicode character, are allowed in the

values and are specified in the same notation as those used in Java strings (e.g. \n, \t, \r).

Since backslashes (\) have special meanings in a configuration file, keyword values that contain this character will not be interpreted properly by KRTool even if it is surrounded by quotes. A common example would be a Windows path name (e.g. c:\VTT\_EN\_1-1\target). Use the forward slash character instead (c:/VTT\_EN\_1-1/target) or escape the backslash character (c:\\VTT\_EN\_1-1\\target).

Note: Any flag specified on the command-line takes precedence over any equivalent settings placed in the configuration file.

The following table contains valid keywords that can be specified in the configuration file:

Property Key	Associated Flag	Valid Value(s)
krtool.target	-t	Specify labels, directories, and URLs
krtool.keywords	-k	Specify keywords to look for in PDS data product labels.
krtool.report	-r	Specify the report file name (do not specify this property key if wanting to write to standard out)
krtool.local	-L	Set to 'true' to turn off directory recursion. Set to 'false' or do not specify this property key to enable directory recursion.
krtool.regexp	-e	Specify file patterns to search for in a target directory.
krtool.ignorefile	-X	Specify file patterns to ignore in a target directory.
krtool.ignoredir	-D	Specify directory patterns to ignore when traversing a target directory.

The following example demonstrates how to set a configuration file:

```
# This is a KRTool configuration file

krtool.target    = ./TEST_DIR
krtool.keywords  = DATA_SET_ID
krtool.report    = report.txt
krtool.regexp    = ".*.LBL"
```

This is equivalent to running the tool with the following flag options:

```
-t ./TEST_DIR -k DATA_SET_ID -r report.txt -e ".*.LBL"
```

The following example demonstrates how to set a configuration file with multiple values for a property key:

```
# This is a KRTool configuration file with multiple values

krtool.target    = TEST.LBL, ./TEST_DIR
krtool.keywords  = DATA_SET_ID, TARGET_NAME
krtool.local     = true
krtool.regexp    = "*.LBL", "*.FMT"
```

This is equivalent to running the tool with the following flags:

```
-t TEST.LBL, ./TEST_DIR -k DATA_SET_ID, TARGET_NAME -L -e "*.LBL", "*.FMT"
```

The following example demonstrates how to set a configuration file with multiple values that span across multiple lines:

```
# This is a KRTool configuration file with multiple values

krtool.target    = TEST.LBL, \
                  ./TEST_DIR
krtool.keywords  = DATA_SET_ID, \
                  TARGET_NAME
krtool.local     = true
krtool.regexp    = "*.LBL", \
                  "*.FMT"
```

The following example demonstrates how to override a setting in the configuration file.

Suppose the configuration file *config.cfg* is defined as follows:

```
# This is a KRTool configuration file

krtool.target    = ./TEST_DIR
krtool.keywords  = DATA_SET_ID
krtool.regexp    = "*.LBL"
```

If a search is desired on all files in *TEST\_DIR* instead of just files that end in ".LBL" as defined in the configuration file, then the following command demonstrates how to perform this behavior change:



```
% KRTool -c config.cfg -e ""
```

## 1.5 Appendix A - UNIX Setup Options

---

### UNIX Setup Options

This section details a couple of options for setting up a UNIX environment for launching KRTool.

#### Specify the CLASSPATH on the Command-Line

An alternative method to setting the *CLASSPATH* variable with all of the tool's dependent JAR files is to specify the *java.ext.dirs* Java property on the command-line when running the tool each time. This is done by passing the property via the Java "-D" flag as demonstrated in the following example:

```
% java -Djava.ext.dirs=$HOME/krtool-0.1.0/lib \  
gov.nasa.pds.krtool.KRTool <command-line arguments>
```

#### Specify the JAR on the Command-Line

Another alternative method is to specify the executable JAR file on the command-line when running the tool each time. This is done by passing the JAR file specification via the Java "-jar" flag as demonstrated in the following example:

```
% java -jar \  
$HOME/krtool-0.1.0/lib/krtool-0.1.0.jar <command-line arguments>
```

## 1.6 Appendix B - Windows Setup Options

---

### Windows Setup Options

This section details a couple of options for setting up a Windows environment for launching KRTool.

#### Specify the CLASSPATH on the Command-Line

An alternative method to setting the *CLASSPATH* variable with all of the tool's dependent JAR files is to specify the *java.ext.dirs* Java property on the command-line when running the tool each time. This is done by passing the property via the Java "-D" flag as demonstrated in the following example:

```
C:\> java -Djava.ext.dirs=c:\krtool-0.1.0\lib \
gov.nasa.pds.krtool.KRTool <command-line arguments>
```

#### Specify the JAR on the Command-Line

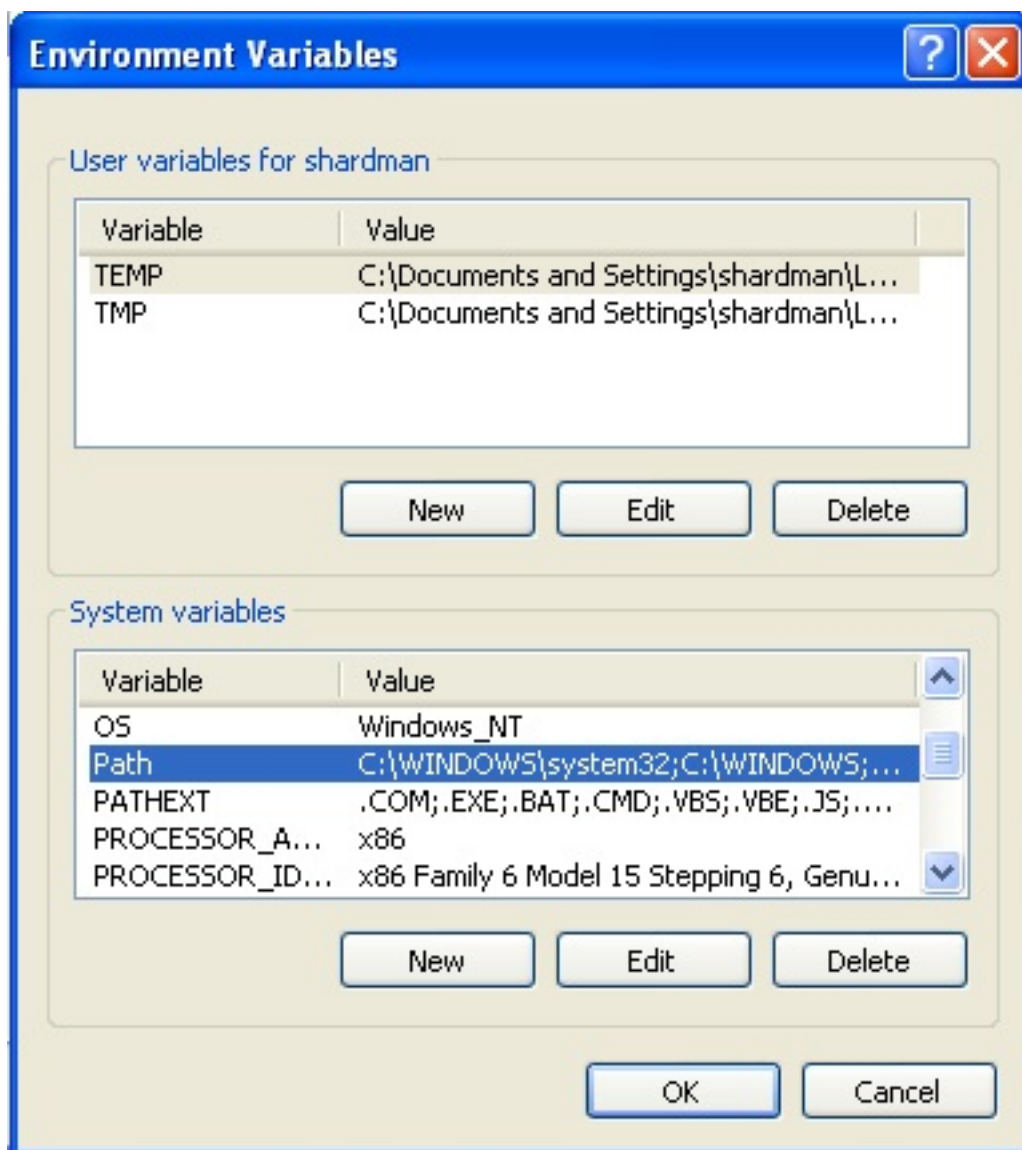
Another alternative method is to specify the executable JAR file on the command-line when running the tool each time. This is done by passing the JAR file specification via the Java "-jar" flag as demonstrated in the following example:

```
C:\> java -jar \
c:\krtool-0.1.0\lib\krtool-0.1.0.jar <command-line arguments>
```

#### Specify the Path in the Control Panel

The method for setting the executable path permanently for KRTool is to set the *Path* environment variable via the control panel as follows:

- Right-click on *My Computer* icon on your desktop and select the *Properties* menu item.
- Navigate to the *Advanced* tab and select the *Environment Variables* button. At this point, you should now see a window like the one below:



- Highlight the *Path* variable in the System Variables list and select the Edit button.
- Append to the current contents of the variable, the path to the *bin* directory within *krtool* package. Separate the package path from the current contents of the variable with a semicolon.
- Select the *OK* button when you are finished editing the *Path* variable, then select the *OK* button at the Environment Variables window to apply the changes.

Note: If you already have a DOS window open, you will need to close and re-open the window for the *Path* changes to take effect.