# Use Command Line Tools to Run Transformations and Jobs

Pentaho Data Integration command line tools execute PDI content from outside of the PDI Client (Spoon). Typically you would use these tools in the context of creating a script or a cron job to run the job or transformation based on some condition outside of the realm of Pentaho software.

Pan is the PDI command line tool for executing transformations.

Kitchen is the PDI command line tool for executing jobs.

Both of these programs are explained in detail below.

**Parent Topic**

- Products

**Child Topics**

- Startup script options
- Pan Options and Syntax

  Pan runs transformations, either from a PDI repository (database or enterprise), or from a local file. The syntax for the batch file and shell script are shown below. All Pan options are the same for both.
- Pan Status Codes

  When you run Pan, there are seven possible return codes that indicate the result of the operation. All of them are defined below.
- Kitchen Options and Syntax

  Kitchen runs jobs, either from a PDI repository (database or enterprise), or from a local file. The syntax for the batch file and shell script are shown below. All Kitchen options are the same for both.
- Kitchen Status Codes
- Import KJB or KTR Files From a Zip Archive
- Connect to a Repository with Command-Line Tools

  To export repository objects into XML format using command-line tools instead of exporting repository configurations from within the PDI client, use named parameters and command-line options when calling Kitchen or Pan from a command-line prompt.
- Export Content from Repositories with Command-Line Tools

  To export repository objects into XML format, using command-line tools instead of exporting repository configurations from within the PDI client, use named parameters and command-line options when calling Kitchen or Pan from a command-line prompt.

# Startup script options

Pan and Kitchen recognize the command line options in the scripts that start the PDI client: `Spoon.bat` on Windows or `Spoon.sh` on Linux. To use the following options with Pan or Kitchen, modify your startup script to include these options.

NoteThe startup script default directory is `design-tools/data-integration`. See Modify the PDI client startup script for more information,

The following table describes the command line options:

| Option | Description |
|---|---|
| **FILTER_GTK_WARNINGS** | Option to suppress GTK warnings from the output of the `spoon.sh` and `kitchen.sh` scripts. You can:<br>• Set to `true` to suppress warnings.<br>• Leave this option empty to view warnings. |
| **SKIP_WEBKITGTK_CHECK** | Option to suppress warnings that the `libwebkitgtk` file is not installed when launching the PDI client. You can:<br>• Set to `true` to suppress warnings.<br>• Leave this option empty to view warnings. |
| **KETTLE_HOME** | Option identifying the user's home directory. The directory contains configuration files, which vary depending on the user who is logged on. You can use the KETTLE_HOME variable to change the location of the files normally in the `<user home>.kettle` directory or to specify the home directory for all users on a machine. |
| **KETTLE_LOG_SIZE_LIMIT** | Option to limit the log size of transformations and jobs that do not have the log size limit property. |
| **KETTLE_JNDI_ROOT** | Option used to change the Simple JNDI path, which is the directory that contains the `jdbc.properties` file. |
| **KETTLE_DIR** | The directory where the PDI client is installed. |
| **KETTLE_REPOSITORY** | The repository that Kettle connects to when it starts. |
| **LIBPATH** | Value that is passed as the -Djava.library.path Java parameter. |
| **PENTAHO_DI_JAVA_OPTIONS** | Option to pass additional Java arguments when running Kettle. For example, you can set an option to Increase the PDI client memory limit. |

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Pan Options and Syntax

Pan runs transformations, either from a PDI repository (database or enterprise), or from a local file. The syntax for the batch file and shell script are shown below. All Pan options are the same for both.

NoteWindows systems use syntax with the forward slash (“/”) and colon (“:”). If spaces are present in the option values, use single quotes (‘) and double quotes (“”) to keep spaces together, for example, `"-param:MASTER_HOST=192.168.1.3" "-param:MASTER_PORT=8181"`

```
pan.sh -option=value arg1 arg2
```

```
pan.bat /option: value arg1 arg2
```

| Switch | Purpose |
|---|---|
| rep | Enterprise or database repository name, if you are using one |
| user | Repository username |
| pass | Repository password |
| trans | The name of the transformation (as it appears in the repository) to launch |
| dir | The repository directory that contains the transformation, including the leading slash |
| file | If you are calling a local KTR file, this is the filename, including the path if it is not in the local directory |
| level | The logging level (Basic, Detailed, Debug, Rowlevel, Error, Nothing) |
| logfile | A local filename to write log output to |
| listdir | Lists the directories in the specified repository |
| listtrans | Lists the transformations in the specified repository directory |
| listrep | Lists the available repositories |
| exprep | Exports all repository objects to one XML file |
| norep | Prevents Pan from logging into a repository. If you have set the KETTLE_REPOSITORY, KETTLE_USER, and KETTLE_PASSWORD environment variables, then this option will enable you to prevent Pan from logging into the specified repository, assuming you would like to execute a local KTR file instead. |
| safemode | Runs in safe mode, which enables extra checking |
| version | Shows the version, revision, and build date |
| param | Set a named parameter in a *name=value* format. For example: *-param:FOO=bar* |

| Switch | Purpose |
| --- | --- |
| listparam | List information about the defined named parameters in the specified transformation. |
| maxloglines | The maximum number of log lines that are kept internally by PDI. Set to 0 to keep all rows (default) |
| maxlogtimeout | The maximum age (in minutes) of a log line while being kept internally by PDI. Set to 0 to keep all rows indefinitely (default) |

```
sh pan.sh -rep=initech_pdi_repo -user=pgibbons -pass=lumburghsux -trans=TPS_
reports_2011
```

```
pan.bat /rep:initech_pdi_repo /user:pgibbons /pass:lumburghsux /trans:TPS_reports_
2011
```

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Pan Status Codes

When you run Pan, there are seven possible return codes that indicate the result of the operation. All of them are defined below.

| Status Code | Definition |
| --- | --- |
| 0 | The transformation ran without a problem. |
| 1 | Errors occurred during processing |
| 2 | An unexpected error occurred during loading / running of the transformation |
| 3 | Unable to prepare and initialize this transformation |
| 7 | The transformation couldn't be loaded from XML or the Repository |
| 8 | Error loading steps or plugins (error in loading one of the plugins mostly) |
| 9 | Command line usage printing |

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Kitchen Options and Syntax

Kitchen runs jobs, either from a PDI repository (database or enterprise), or from a local file. The syntax for the batch file and shell script are shown below. All Kitchen options are the same for both.

NoteWindows systems use syntax with the forward slash (“/”) and colon (“:”). If spaces are present in the option values, use single quotes (‘) and double quotes (“”) to keep spaces together, for example, `"-param:MASTER_HOST=192.168.1.3" "-param:MASTER_PORT=8181"`

```
kitchen.sh -option=value arg1 arg2
```

```
kitchen.bat /option: value arg1 arg2
```

| Switch | Purpose |
|---|---|
| rep | Enterprise or database repository name, if you are using one |
| user | Repository username |
| pass | Repository password |
| job | The name of the job (as it appears in the repository) to launch |
| dir | The repository directory that contains the job, including the leading slash |
| file | If you are calling a local KJB file, this is the filename, including the path if it is not in the local directory |
| level | The logging level (Basic, Detailed, Debug, Rowlevel, Error, Nothing) |
| logfile | A local filename to write log output to |
| listdir | Lists the sub-directories within the specified repository directory |
| listjob | Lists the jobs in the specified repository directory |
| listrep | Lists the available repositories |
| export | Exports all linked resources of the specified job. The argument is the name of a ZIP file. |
| norep | Prevents Kitchen from logging into a repository. If you have set the KETTLE_REPOSITORY, KETTLE_USER, and KETTLE_PASSWORD environment variables, then this option will enable you to prevent Kitchen from logging into the specified repository, assuming you would like to execute a local KTR file instead. |
| version | Shows the version, revision, and build date |
| param | Set a named parameter in a name=value format. For |

| Switch | Purpose |
| --- | --- |
| | example: -param:FOO=bar |
| listparam | List information about the defined named parameters in the specified job. |
| maxloglines | The maximum number of log lines that are kept internally by PDI. Set to 0 to keep all rows (default) |
| maxlogtimeout | The maximum age (in minutes) of a log line while being kept internally by PDI. Set to 0 to keep all rows indefinitely (default) |

```
sh kitchen.sh -rep=initech_pdi_repo -user=pgibbons -pass=lumburghsux -job=TPS_
reports_2011
```

```
kitchen.bat /rep:initech_pdi_repo /user:pgibbons /pass:lumburghsux /job:TPS_
reports_2011
```

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Kitchen Status Codes

When you run Kitchen, there are seven possible return codes that indicate the result of the operation. All of them are defined below.

| Status Code | Definition |
| --- | --- |
| 0 | The job ran without a problem. |
| 1 | Errors occurred during processing |
| 2 | An unexpected error occurred during loading or running of the job |
| 7 | The job couldn't be loaded from XML or the Repository |
| 8 | Error loading steps or plugins (error in loading one of the plugins mostly) |
| 9 | Command line usage printing |

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Import KJB or KTR Files From a Zip Archive

Both Pan and Kitchen can pull PDI content files from out of Zip files. To do this, use the ! switch, as in this example:

```
Kitchen.bat /file:"zip:file:///C:/Pentaho/PDI Examples/Sandbox/linked_executable_
job_and_transform.zip!Hourly_Stats_Job_Unix.kjb"
```

If you are using Linux or Solaris, the ! must be escaped:

```
./kitchen.sh -file:"zip:file:////home/user/pentaho/pdi-ee/my_package/linked_
executable_job_and_transform.zip\!Hourly_Stats_Job_Unix.kjb"
```

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Connect to a Repository with Command-Line Tools

To export repository objects into XML format using command-line tools instead of exporting repository configurations from within the PDI client, use named parameters and command-line options when calling Kitchen or Pan from a command-line prompt.

The following is an example command-line entry to execute an export job using Kitchen:

```
    call kitchen.bat /file:C:\Pentaho_samples\repository\repository_export.kjb
    "/param:rep_name=PDI2000" "/param:rep_user=admin" "/param:rep_
password=password"
    "/param:rep_folder=/public/dev"
    "/param:target_filename=C:\Pentaho_samples\repository\export\dev.xml"
```

| Parameter | Description |
|---|---|
| rep_folder | Repository Folder |
| rep_name | Repository Name |
| rep_password | Repository Password |
| rep_user | Repository Username |
| target_filename | Target Filename |

NoteIt is also possible to use obfuscated passwords with Encr a command line tool for encrypting strings for storage or use by PDI.

The following is an example command-line entry to execute a complete command-line call for the export in

addition to checking for errors:

```
@echo off
ECHO This an example of a batch file calling the repository_export.kjb

cd C:\Pentaho\pdi-ee-<filepath>--check--</filepath>9.3.0>\data-integration

call kitchen.bat /file:C:\Pentaho_samples\repository\repository_export.kjb
"/param:rep_name=PDI2000"
"/param:rep_user=admin" "/param:rep_password=password" "/param:rep_folder=/public/
dev"
"/param:target_filename=C:\Pentaho_samples\repository\export\dev.xml"

if errorlevel 1 goto error
echo Export finished successfull.
goto finished

:error
echo ERROR: An error occured during repository export.
:finished
REM Allow the user to read the message when testing, so having a pause
pause
```

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs

# Export Content from Repositories with Command-Line Tools

To export repository objects into XML format, using command-line tools instead of exporting repository configurations from within the PDI client, use named parameters and command-line options when calling Kitchen or Pan from a command-line prompt.

The following is an example command-line entry to execute an export job using Kitchen:

```
call kitchen.bat /file:C:\Pentaho_samples\repository\repository_export.kjb
"/param:rep_name=PDI2000" "/param:rep_user=admin" "/param:rep_password=password"
"/param:rep_folder=/public/dev"
"/param:target_filename=C:\Pentaho_samples\repository\export\dev.xml"
```

| Parameter | Description |
| --- | --- |
| rep_folder | Repository Folder |
| rep_name | Repository Name |
| rep_password | Repository Password |
| rep_user | Repository Username |
| target_filename | Target Filename |

It is also possible to use obfuscated passwords with Encr, the command line tool for encrypting strings for storage/use by PDI. The following is an example command-line entry to execute a complete command-line call for the export in addition to checking for errors:

```
@echo off
ECHO This an example of a batch file calling the repository_export.kjb

cd C:\Pentaho\pdi-ee-<filepath>--check--</filepath>9.3.0>\data-integration

call kitchen.bat /file:C:\Pentaho_samples\repository\repository_export.kjb
"/param:rep_name=PDI2000"
"/param:rep_user=admin" "/param:rep_password=password" "/param:rep_folder=/public/
dev"
"/param:target_filename=C:\Pentaho_samples\repository\export\dev.xml"

if errorlevel 1 goto error
echo Export finished successful.
goto finished

:error
echo ERROR: An error occurred during repository export.
:finished
REM Allow the user to read the message when testing, so having a pause
pause
```

**Parent Topic**

- Use Command Line Tools to Run Transformations and Jobs