

How to Automate Validation with Pinnacle 21 Command Line Interface and SAS®

By Amy Garrett, Pinnacle 21

Introduction

The Command Line Interface (CLI) allows users run a validation by passing a series of parameters to the command line. The CLI is an alternative to the Graphical User Interface (GUI) and allows users to have increased customization and automation options. It supports all aspects of Pinnacle 21 validation, including validation of define.xml.

The CLI is available to users of Pinnacle 21 Community as well as those with Pinnacle 21 Enterprise v3.4 or greater.

Parameters - Enterprise & Community
Standard and Version
--standard
--standard.version
Validator Source Data
--source.adam
--source.sdtm
--source.send
--source.define
Terminology and Dictionaries
--cdisc.ct.adam.version
--cdisc.ct.sdtm.version
--cdisc.ct.send.version
--meddra.version
--ndf-rt.version
--snomed.version
--unii.version
Validator Report
--report
Define.xml Designer
--source
--output

Parameters - Enterprise
--api.key
--web.host
--group
--project
--study
--datapackage
--create.always

- ▶ Most Enterprise specific parameters can be stored in a config file, pinnacle21.conf
- ▶ Parameters stored in pinnacle21.conf will be automatically passed to the CLI
- ▶ Recommendation is to store static parameters in pinnacle21.conf

Enterprise Examples

```
x
java -jar "&jarpath.\&jarfile."
--standard=sdm
--standard.version=3.2
--source.sdtm="aspera//C:\users\data\sdm\xpt"
--source.define="C:\users\data\sdm\define.xml"
--cdisc.ct.sdtm.version=2017-12-22
--unii.version=2017-11-15
--ndf-rt.version=2018-02-05
--meddra.version=19.0
--report="C:\users\reports\s101-sdm.xlsx";
```

Example 1. SDTM Validation

```
x
java -jar "&jarpath.\&jarfile."
--standard=adam
--standard.version=1.0
--source.sdtm="aspera//C:\users\data\sdm\xpt"
--source.adam="aspera//C:\users\data\adam\xpt"
--cdisc.ct.sdtm.version=2017-12-22
--cdisc.ct.adam.version=2017-09-29
--meddra.version=19.0
--report="C:\users\reports\s101-adam.xlsx";
```

Example 3. ADaM Validation

```
x
java -jar "&jarpath.\&jarfile."
--standard=sdm
--standard.version=3.2
--source="C:\users\data\sdm\define.xlsx"
--output="C:\users\data\sdm\define.xml"
--report="C:\users\reports\s101-sdm-define.xlsx"
```

Example 5. Define Conversion and Validation

Conclusion

With a little bit of set-up, the CLI can be called to run validations on a fixed schedule, after a dataset is created, or after a dataset is added to a library. Error codes may be leveraged to kick-off additional work-flows.

The CLI supports all aspects of validation, saves time, and can be used to automate workflows.

Community Examples

```
x
java -jar "&jarpath.\&jarfile."
--standard=sdm
--standard.version=3.2
--source.sdtm="C:\users\data\sdm\xpt"
--source.define="C:\users\data\sdm\define.xml"
--cdisc.ct.sdtm.version=2017-12-22
--unii.version=2017-11-15
--ndf-rt.version=2018-02-05
--meddra.version=19.0
--report=" C:\users\reports\s101-sdm.xlsx";
```

Example 2. SDTM Validation

```
x
java -jar "&jarpath.\&jarfile."
--standard=adam
--standard.version=1.0
--source.sdtm="C:\users\data\sdm\xpt"
--source.adam="C:\users\data\adam\xpt"
--cdisc.ct.sdtm.version=2017-12-22
--cdisc.ct.adam.version=2017-09-29
--meddra.version=19.0
--report=" C:\users\reports\s101-adam.xlsx";
```

Example 4. ADaM Validation

```
x
java -jar "&jarpath.\&jarfile."
--standard=sdm
--source="C:\users\data\sdm\define.xlsx"
--output="C:\users\data\sdm\define.xml "
```

```
x
java -jar "&jarpath.\&jarfile."
--standard=sdm
--source.define="C:\users\data\sdm\define.xml"
--cdisc.ct.sdtm.version=2017-12-22
--report="C:\users\reports\s101-adam.xlsx";
```

Example 6. (top) Define Conversion; xlsx to xml
(bottom) Define Validation

