Pharmacogenomic Insights (PGXI) Utilities Developer Manual

For Pharmacogenomic Insights Utilities version 1.2

Table of Contents

Introduction

This manual provides a comprehensive overview of the PGx Insights Utilities command-line program. The program is designed to process genomic data files by applying filters, processing files through an API, generating reports, and consolidating data with demographic information.

Prerequisites

Before using the program, ensure that:

- If a previous version of the application is installed, you use Add or Remove Programs to remove the previous version.
- You install the program by using the PgxUtilsInstaller.msi in the Install directory of the distribution disk.

Overview of Commands

The program supports the following commands: •

- filterfile: Filters a single VCF input file using a specified filter file.
- filterfolder: Applies filtering to all files within a specified folder, with an optional polling interval.
- processfiles: Processes one or more input files via the PGx Insights API.
- processfolder: Processes all files in a folder through the API, with support for polling the input folder.
- reportfromfiles: Generates reports from API-processed files using a report template.
- consolidatefiles: Consolidates test results with demographic data.
- reportfromjson: Creates reports from JSON files using report templates from a designated folder.

Command Descriptions and Usage

1. filterfile Command

Description: Filters a single input file using a filter file and outputs the result to a specified folder.

Options:

- --infile: Specifies the input file to be processed.
- --filterfile: Specifies the filter file used for processing.
- --outfolder: Specifies the folder where the filtered file will be saved.

Usage Example:

filterfile --infile "C:\Data\input.vcf" --filterfile "C:\Data\filter.txt" --outfolder "C:\Data\Filtered"

2. filterfolder Command

Description: Applies filtering to all files in a specified folder using a given filter file. A polling interval can be set to continuously monitor the folder.

Options:

- --infolder: The folder containing input files.
- --filterfile: The filter file to apply.
- --outfolder: The folder where filtered files will be saved.
- --pollinginterval: The interval (in seconds) at which the folder is polled for new or changed files.

Usage Example:

filterfolder --infolder "C:\Data\InputFolder" --filterfile "C:\Data\filter.txt" --outfolder "C:\Data\Filtered" --pollinginterval 10

3. processfiles Command

Description: Processes one or more files using an API. This command requires a configuration file for authentication and supports multiple output file types.

Options:

- --infiles: A list of input files to process.
- --configfile: The configuration file used for API authentication.
- --outtypes: Specifies one or more output file types (e.g., CSV, JSON).
- --outfolder: The folder where the processed output files will be stored.

Usage Example:

processfiles --infiles "C:\Data\file1.vcf" "C:\Data\file2.vcf" --configfile "C:\Config\auth.json" -outtypes csv json --outfolder "C:\Data\Processed"

4. processfolder Command

Description: Processes all files within a given folder using an API. Similar to **processfiles**, but designed for bulk folder processing and supports polling for changes.

Options:

- --infolder: The folder containing input files.
- --configfile: The configuration file for API authentication.
- --outtypes: One or more output file types.
- --outfolder: The destination folder for the processed files.
- --pollinginterval: The polling interval (in seconds) for monitoring the input folder.

Usage Example:

processfolder --infolder "C:\Data\InputFolder" --configfile "C:\Config\auth.json" --outtypes csv json --outfolder "C:\Data\Processed" --pollinginterval 10

5. reportfromfiles Command

Description: Generates reports from files processed via the API, using a specified report template.
Options:

- --infiles: A list of input files for which reports will be generated.
- --configfile: The configuration file for API authentication.
- --templatefile: The report template file to use.
- --outfolder: The folder where the generated reports will be saved.

Usage Example:

reportfromfiles --infiles "C:\Data\file1.vcf" "C:\Data\file2.vcf" --configfile "C:\Config\auth.json" --templatefile "C:\Templates\report.docx" --outfolder "C:\Reports"

6. consolidatefiles Command

Description: Merges test results with demographic data to produce a consolidated output.
Options:

- --infolder: The folder containing the test result files.
- --demographicsfile: The file containing demographic data.
- --outfolder: The folder for the consolidated output.
- --pollinginterval: The polling interval (in seconds) for monitoring the input folder.

Usage Example:

consolidatefiles --infolder "C:\Data\TestResults" --demographicsfile "C:\Data\demographics.csv" --outfolder "C:\Data\Consolidated" --pollinginterval 10

7. reportfromjson Command

Description: Creates reports based on JSON files using report templates located in a specified template folder.
Options:

- --infolder: The folder containing JSON input files.
- --templatefolder: The folder containing the report templates.
- --outfolder: The folder where the generated reports will be stored.
- --pollinginterval: The polling interval (in seconds) for monitoring the input folder.

Usage Example:

reportfromjson --infolder "C:\Data\JsonFiles" --templatefolder "C:\Templates" --outfolder "C:\Reports" --pollinginterval 10

8. reportfromconsolidatedJson Command

Description: Creates reports based on JSON files that had been merged with patient demographics. Note that the commands are the same as 'reportfromjson' however the default locations for where the consolidated insight files are different from the original command.

Options:

- --infolder: The folder containing the consolidated JSON files. Default .\Consolidatedfiles
- --templatefolder: The folder containing the report template that is to be used. Default: .\Reports
- --outfolder: The location which the reports generated will be placed. Default: .\FinalReports.
- --pollinginterval: The number of seconds to pause while polling the input folder for new consolidated JSON files.
- --configfile: The configuration file for required authentication. Default: .\Config

Usage Example: PGxIUtils reportfromjson --infolder ".\ConsolidatedFiles" --templatefolder ".\Reports" --outfolder ".\FinalReports –configile ".\Config\PGxI_Config.json

9. consolidateHl7Files Command

Description: Merges patient demographic information from Hl7 files in the folder 'HL7 Orders' with existing JSON files in the folder 'InsightsFiles'. Replaces existing JSON files with new Consolidated JSON files in the 'Insights Files' folder.

Options:

- --infolder: The folder that contains the hl7 files. Default folder is .\HL7Orders
- --templatefolder: The folder that contains the report templates. Default is .\Reports
- --outfolder: The folder for the output files. Default is .\FinalReports
- --pollinginterval: The number of seconds to pause while polling the input folder.
- --configfile: The configuration file for required authentication. Default is .\Config

Usage Example: PgxIUtils consolidatefiles --infolder ".\insightsfiles" --hl7orders ".\HL7Orders" --outfolder ".\ConsolidatedFiles" --configfile ".\Config\PGxI_Config.json"

10. scaffoldFolder

Description: Creates a .vcf overlay that defines all of the mutations that are understood and can be useful for PGx purposes. This overlay is created for mutations that actually exist in the genetic results.

Options:

- --infolder: The folder for your input files. Default: .\Datafolder
- --outfolder: the folder where your processed vcf files
- --scaffoldfile: The file that defines what variants are in the tested sample. Default: .\Config\NewScaffold.vcf
- --pollinginterval: The number of seconds to pause while polling the input folder. Default: 1

Usage Example: PGxIUtils.exe scaffoldfolder –infolder .\DataFolder –scaffoldfile ".\Config\NewScaffold.vcf" –outfolder .\inputfolder

Additional Information

Error Handling

The program is built with robust error handling. A custom exception handler catches any runtime errors during command execution, displays a user-friendly error message on the console, and sets the exit code to 1. This behavior helps ensure that any issues are clearly communicated and that the program terminates gracefully.

Example Commands

Below are some example command invocations.

Filtering a Single File:

filterfile --infile "C:\Data\input.vcf" --filterfile "C:\Data\filter.txt" --outfolder "C:\Data\Filtered"

Processing Multiple Files:

processfiles --infiles "C:\Data\file1.vcf" "C:\Data\file2.vcf" --configfile "C:\Config\auth.json" -

outtypes csv json --outfolder "C:\Data\Processed" **Generating a Report:**

reportfromfiles --infiles "C:\Data\file1.vcf" "C:\Data\file2.vcf" --configfile "C:\Config\auth.json" -
templatefile "C:\Templates\report.docx" --outfolder "C:\Reports"

Contact and Support

For additional information, troubleshooting, or support, please consult the detailed documentation provided with the PGx Insights Utilities package or contact the support team at: Email: support@ts-bioinformatics@qiagen.com

Document Revision History

Revision	Description
May 2025	Initial release for Pharmacogenomic Insights Web Application version 1.0
September 2025	Updated with new command information and their associated options for version 1.0