

Documentation

Welcome to the LogViewPlus documentation. This documentation aims to give you a head start in learning how to use LogViewPlus v2.5 and was last updated on 01 November 2020 and is also available for [download as a PDF file](#).

If you are having a specific problem with LogViewPlus, please see [our F.A.Q.](#)

If you're unable to find the information you're looking for, or if you find that some sections are not clear, please don't hesitate to [contact us](#).

Getting Started

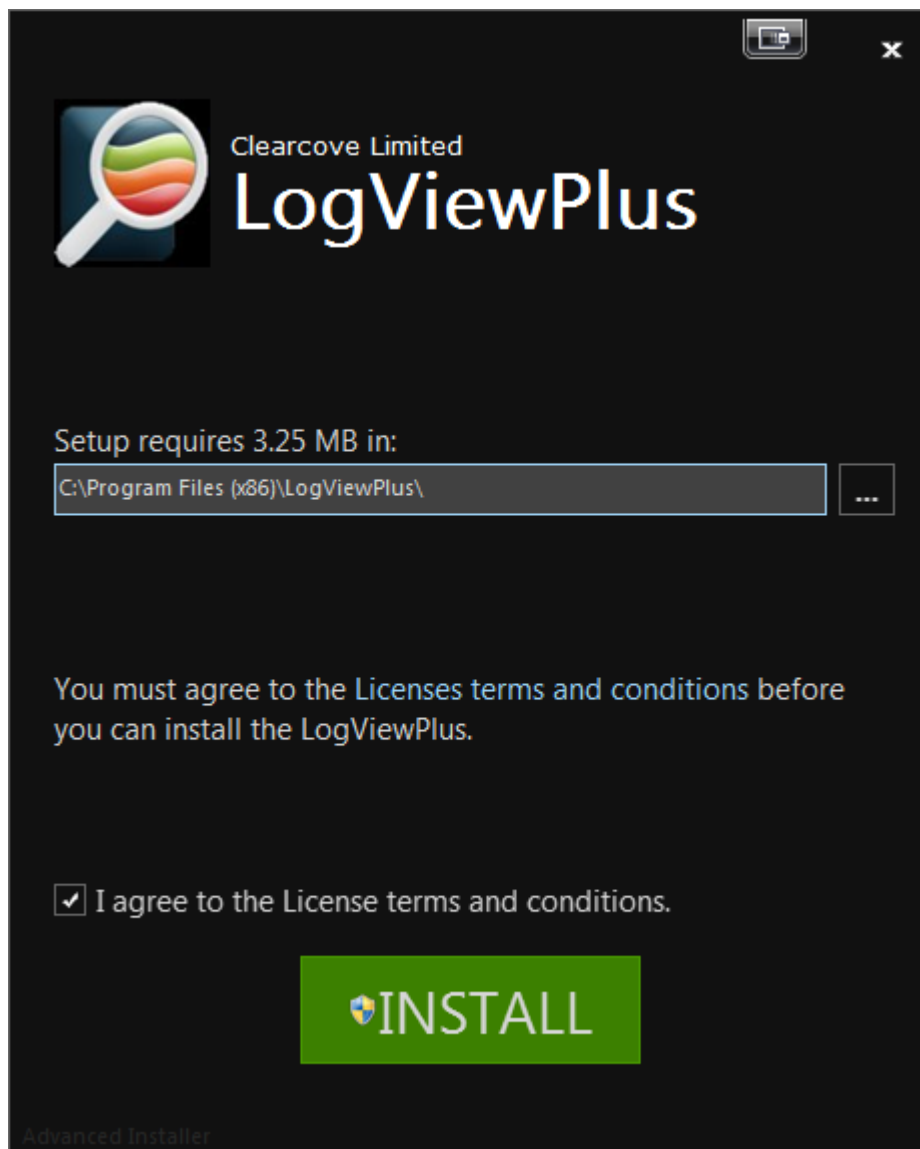
LogViewPlus is built specifically for viewing application log files. LogViewPlus works by parsing your log files and transforming the data into information. Because of this transformation, LogViewPlus can give you filtering and analysis options beyond those available in a typical text editor.

For a quick walk through of LogViewPlus check out [our videos](#) which cover basic functionality and configuration. We have also created a series of [short tutorials](#) which show you how to access and use specific features.

Installing LogViewPlus

Installing LogViewPlus is easy. You can get started by downloading and running the LogViewPlus installation executable which is located on our [download page](#). When downloading LogViewPlus, your browser may need confirmation to begin the download. For example, Internet Explorer may show a prompt at the bottom of the screen.

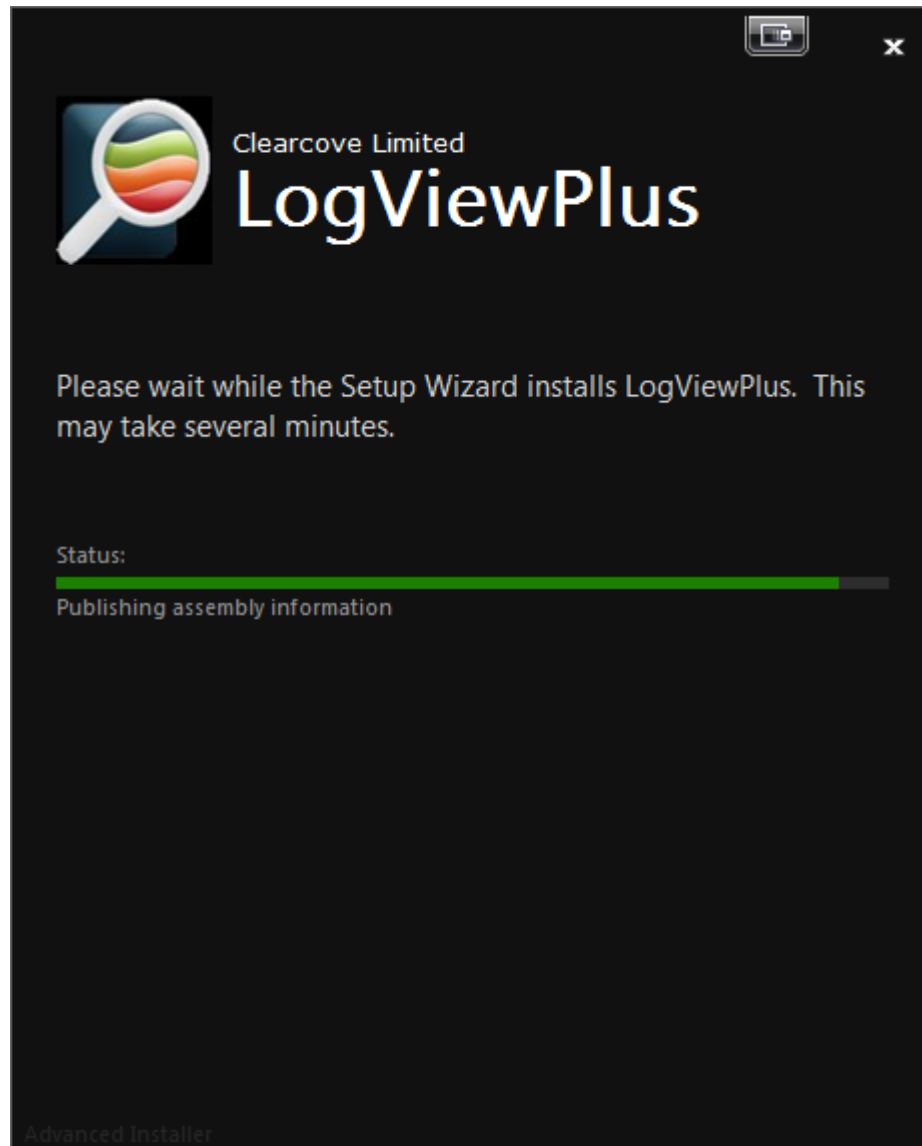
Once you have downloaded LogViewPlus, you can begin the installation process by locating the installer and double clicking it. The LogViewPlus installation will require elevated permission to execute. Once running, you will see the main installation window:



Choose the directory where you want to install LogViewPlus, agree to the [licensing terms](#) and click 'Install' to begin the installation process.

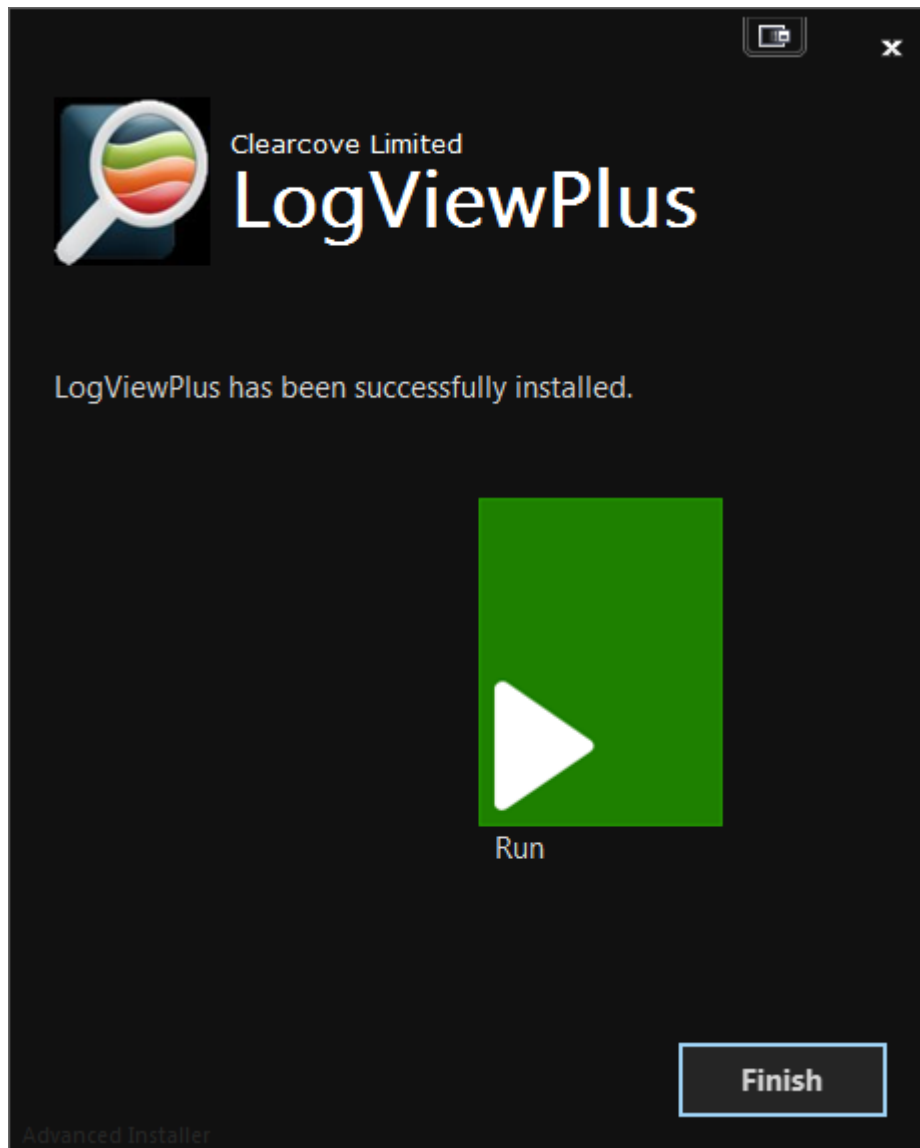
At this point, if you are not running with elevated permission, you may be prompted by the User Access Control informing you that you are making changes to your computer. You will need to agree to make these changes for the installation process to continue.

You should see a screen which notifies you of progress. Something like:



The progress screen may take several minutes as assemblies need to be registered. During this time, the progress bar may not be updating. This is normal - please be patient.

Finally, you will see a screen informing you that installation is complete.



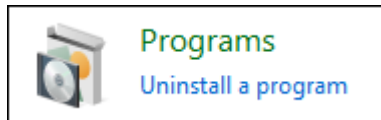
You are now ready to run LogViewPlus. It is important to note that starting LogViewPlus from the installation complete page will run LogViewPlus under the same user account as the installer. This may be problematic if you have installed LogViewPlus under a special account.

You may experience increased CPU activity for a minute or so after the installation has completed. Some libraries used by LogViewPlus may need to be compiled to target the current computer. Windows may continue processing these files after the installation process has completed.

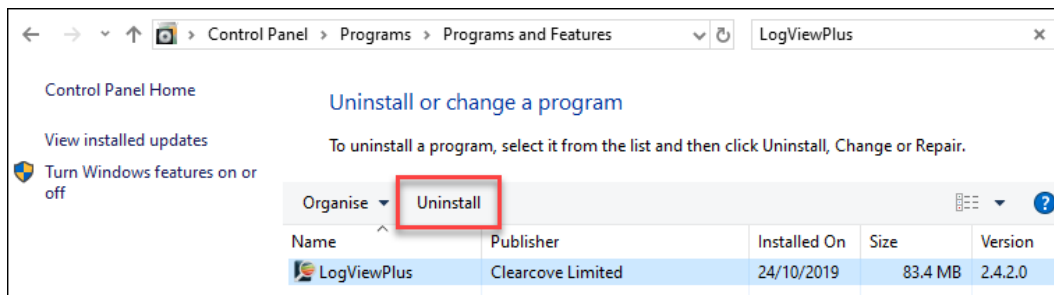
If you have any problems with installing LogViewPlus, please have a look at our [FAQ](#), or [contact us](#) for assistance.

Uninstalling LogViewPlus

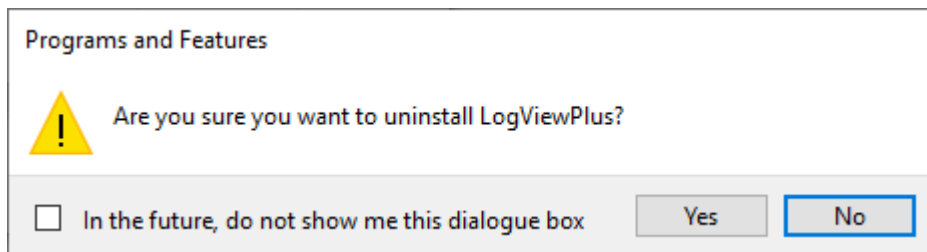
Uninstalling LogViewPlus is easy. Simply open Control Panel by clicking on the start menu and typing "Control Panel". Then select the "Uninstall Programs" option:



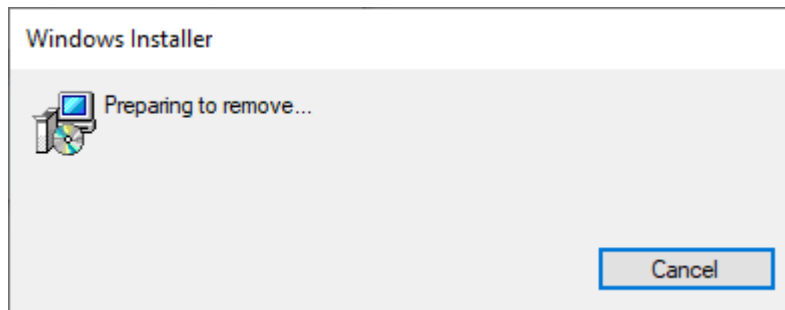
Next, find and select LogViewPlus in your list of installed programs and click Uninstall:



A message will be displayed to confirm you want to uninstall:



Once confirmed, a window will be shown which allows you to monitor the uninstall progress:



At this point, you may be prompted by the User Access Control informing you that you are making changes to your computer. You will need to agree these changes for the uninstall process to continue.

LogViewPlus will then be removed automatically without the need for further user input.

If you have any problems uninstalling LogViewPlus, please have a look at our [FAQ](#), or [contact us](#) for assistance.

First Look

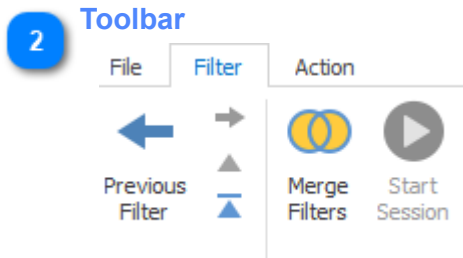
The screenshot shows the LogViewPlus application interface. Numbered callouts point to the following components:

- 1 Quick Toolbar:** Located at the top left, containing icons for file operations (open, save, print, etc.).
- 2 Toolbar:** Located below the Quick Toolbar, containing icons for filtering and session management (Previous Filter, Merge Filters, Start Session, My Filters, Text Filter, Thread Filter, Logger Filter, Date Time Filter, Log Level Filter).
- 3 Files and Views:** A tree view on the left showing the hierarchy of log files and filters. It includes folders for 'SVR_0.Alice.Client.small.log', 'WARN, ERROR, FATAL', '09:40:56.000 - 12:25:21.999', 'Thread '2'', 'Thread '8'', 'Logger 'ActiveContainer'', 'Before 12:21:06.447', 'SVR_0.Bill.Client.small.log', 'INFO', and '19374'.
- 4 File Indicator:** A small icon next to the selected log file in the Files and Views pane.
- 5 Log Entry Grid:** A table displaying log entries with columns for Time, Thread, Level, and Logger. The table shows several entries, including one with an ERROR level for Thread_13.
- 6 Toolbox Tabs:** A tabbed interface at the bottom left showing statistics for Records, Threads, Loggers, Debug, Info, Warn, Error, and Fatal. The 'All' tab is selected, showing a total of 19104 records.
- 7 Original Entry:** The main display area on the right showing the raw log entry for the selected record. It includes a timestamp, thread name, level, and the log message content.

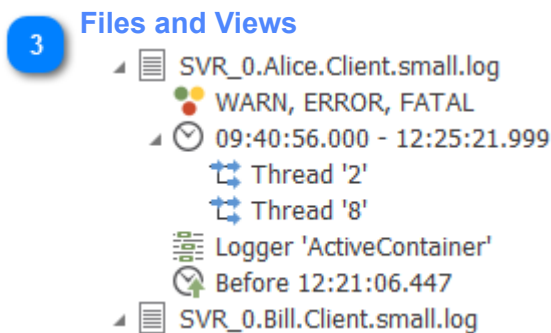
Once you open a log file in LogViewPlus the application should look similar to the above screenshot. LogViewPlus can be thought of as seven distinct areas.

1 Quick Toolbar

The quick toolbar allows easy access to common functions like opening file, merging files and managing auto-scroll.



The [application toolbar](#) contains commands and actions you can use to work with your log files.



On the far left of the screen you should see a tree list which shows all of the files and views you currently have open. You can use this list to navigate between log files and filter results.

This list represents a hierarchy of views on your log file. The element at the base of the hierarchy represents all entries in the log file. As you move away from the base element each generation will have an increasingly focused view of the current log file. Every new generation contains a subset of the log entries available in the previous generation. These filters can be thought of as search results. Double-clicking on a filter result will find the original log entry in the root log file.

Also, note that as you navigate between views LogViewPlus attempts to keep your current record in focus. This allows you to easily view records which were written before and after the current entry when moving between views.

4 File Indicator



The colors in the file indicator bar will appear by default when viewing a merged file. Each distinct color represents a different file. This makes it easy to see at a glance which log entries were written by the same log file. There is also a 'Log File Name' column where you can see the name of the file responsible for the log entry. This column is hidden by default.

The file indicator bar is also used to show bookmarked log entries where applicable.

5 Log Entry Grid

Time	Thread	Level
12:17:18.911	Thread_12	TRACE
12:17:18.999	Thread_13	INFO
12:17:19.073	Thread_10	TRACE
12:17:19.113	Thread_1	TRACE
12:17:19.255	Thread_3	INFO
12:17:19.435	Thread_14	TRACE
12:17:19.623	Thread_13	ERROR
12:17:19.743	Thread_8	INFO

The log entry grid is the heart of LogViewPlus. You can use this grid to easily view information about your log file. This grid is color-coded based on the log level. LogViewPlus supports five different primary log levels: Debug, Info, Warn, Error and Fatal. Primary log levels are immutable, but [secondary log levels can be configured](#).

6

Toolbox Tabs

	All	View
Records:	19104	19104
Threads:	14	14
Loggers:	115	115
Debug:	6012	6012
Info:	11829	11829
Warn:	986	986
Error:	277	277
Fatal:	0	0

The toolbox tab allows easy navigation between different LogViewPlus functional areas. These areas include [statistics](#), [bookmarks](#), [highlights](#), [directory monitors](#), [file transfers](#), and [notes](#).

The tab shown is the statistics grid which can be used to give you an idea of how many elements are in your log file. It's designed to give you a quick idea of how data is distributed in your log file as well as an overview of the available information.

7

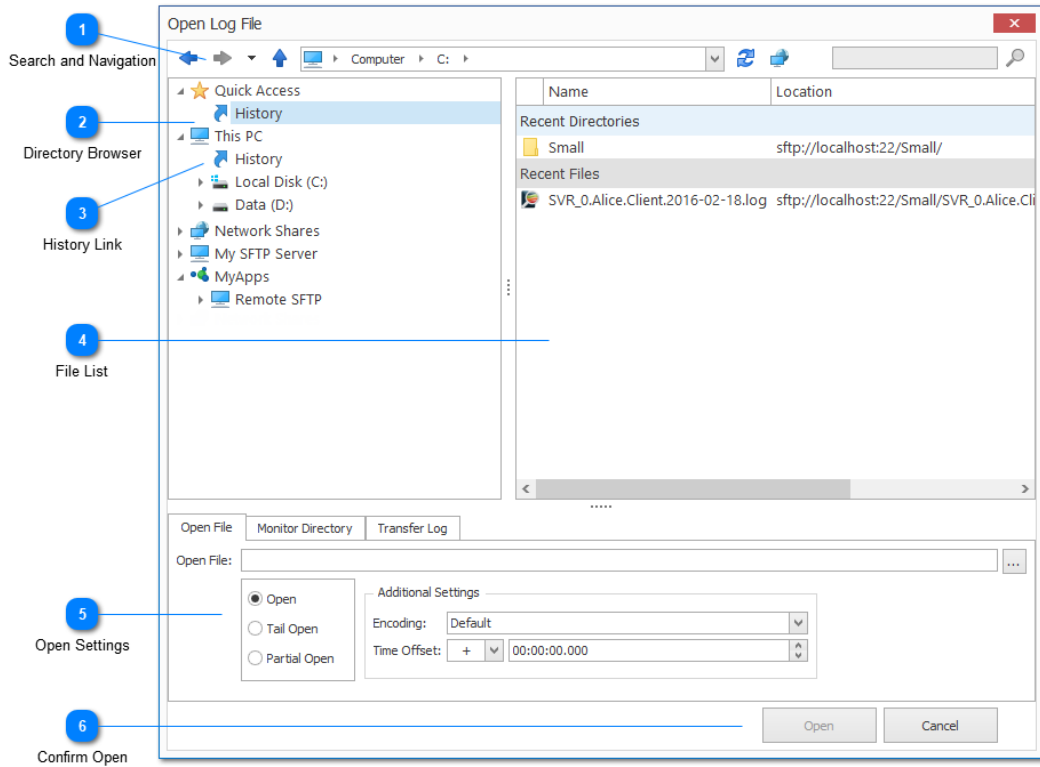
Original Entry

```
12:17:20,008 [Thread_9] INFO CopyCon
<?xml version="1.0" encoding="utf-8"
<SERIES-AND-CLASSES-CONTRACTS-DATA>
  <MERGER-SERIES-AND-CLASSES-CONTRA
    <MERGER>
      <SERIES OWNER-CIK="" SERI
        <CLASS-CONTRACT CLASS
      </SERIES>
      <SERIES OWNER-CIK="" SERI
        <CLASS-CONTRACT CLASS
      </SERIES>
    </MERGER>
  </MERGER-SERIES-AND-CLASSES-CONTR
```

At the bottom of the application you'll find a text box which contains the original log entry as it would appear in the text file. This log entry will change based on your selection in the log entry grid.

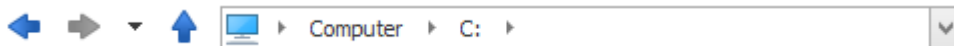
The log entry can be color coded based on pre-defined syntax highlighting rules. Right-clicking the original entry will bring up the [Log Entry Menu](#) which allows you to change the syntax highlighting style.

File Explorer



The LogViewPlus file browser allows you to quickly and easily browse your local file system as well as remote filesystems via SFTP or FTP.

1 Search and Navigation

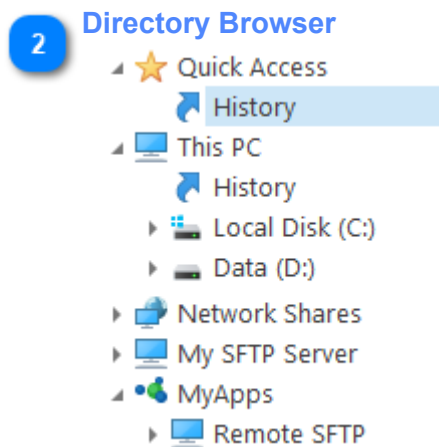


The search navigation bar can be used to navigate forward, backward, and up into a parent folder. There is also a small drop-down after the forward command which can be used to quickly navigate your browsing history.

The navigation bar switches mode depending on if the current file system is local or remote. In a local file system mode the path is tokenized (as shown) allowing quick drop-down navigation into parent folders. When the file system is remote, a

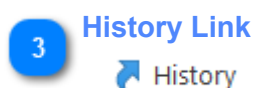
free-form text box will be used instead. This will require the full path to the target directory.

Finally, note that the navigation bar can also support commands and shortcuts - much like Windows Explorer. For example, the shortcut "%temp%" can be used to navigate to your local temp directory. The command "cmd" can be used to open a shell command window in the current target directory. The availability of commands and shortcuts will be dependent on how your Windows shell is configured.



The directory browser is used to navigate between folders. If you have configured LogViewPlus with the connection details of a remote [SFTP or FTP server](#), the server will automatically be added to the bottom of the file browser root nodes.



Network Shares will always be unpopulated when you first install LogViewPlus. This is because some networks may contain a large number of machines and quickly scanning the network to find the machine you want to access may not be possible. Instead, LogViewPlus uses an approach that requires you to know the name of the machine you want to connect to in advance. You can either explicitly add this machine name in the [file system settings](#), or you can paste the full path to the file you want to open into the open file text box. Once a local network file has been opened by LogViewPlus, the network node will be available for future access.



History links are special nodes within the directory browser. They contain a list of your file access history. History links work at two levels. Quick access history contains the last 20 items you opened regardless of where those items point to. Alternatively, computer history links contain the last 20 items you have opened on that particular computer or drive.

Please see the [history](#) node documentation for more information.

4 File List

Name	Location
Recent Directories	
 Small	sftp://localhost:22/Small/
Recent Files	
 SVR_0.Alice.Client.2016-02-18.log	sftp://localhost:22/Small/SVR_0.Alice.Cli

The File List contains a list of all the files in the current directory. In the example above, the [History List](#) is shown. This list shows recently accessed files.

5 Open Settings

Open File

Monitor Directory

Transfer Log

Open File:

☒ Open

☐ Tail Open

☐ Partial Open

Additional Settings

Encoding:

Time Offset:

The Open Settings configuration at the bottom of the screen is used to configure how LogViewPlus should open the target log file or directory. The Open Settings configuration will be [discussed in detail](#) later in this documentation.

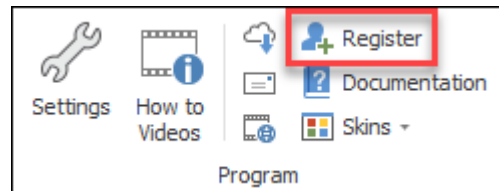
6 Confirm Open

Open

Cancel

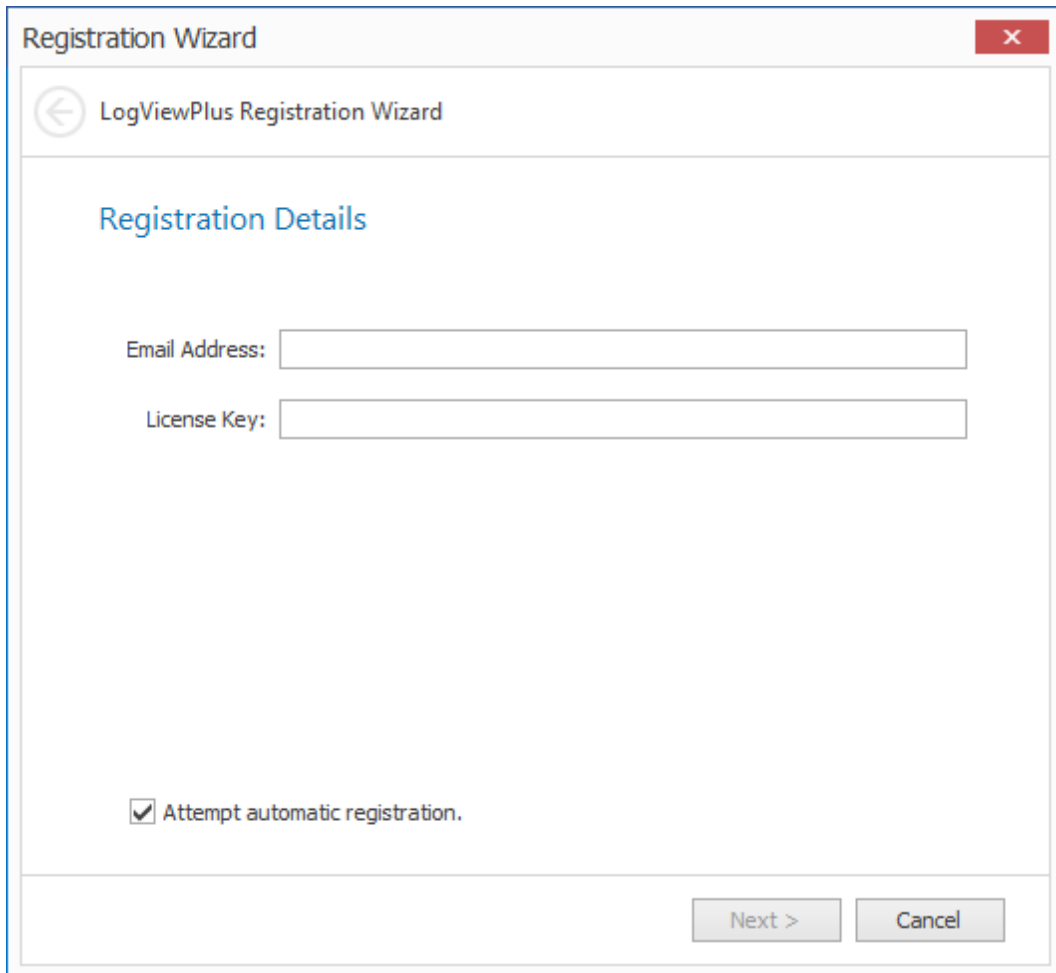
Finally, at the bottom of the Open File dialog is a command to open the currently selected log file as configured. Alternatively, you can close the Open File dialog by selecting the cancel command.

Registration



The registration process is a one-off event that you can use to register LogViewPlus. If LogViewPlus remains unregistered the software will automatically be deactivated in 30 days. An extension may be given [on request](#).

Registration Details



The screenshot shows a Windows-style dialog box titled "Registration Wizard" with a red close button in the top right corner. Inside the dialog, there is a sub-header "LogViewPlus Registration Wizard" with a back arrow icon. Below this, the section "Registration Details" is displayed. It contains two text input fields: "Email Address:" and "License Key:". At the bottom left, there is a checked checkbox labeled "Attempt automatic registration.". At the bottom right, there are two buttons: "Next >" and "Cancel".

The first step in the registration process is to enter the e-mail address and license key you received after your payment was processed. This information is sent to you from LogViewPlus support. If you experience a delay in receiving your license key, it may be worth checking your junk email folder before contacting [LogViewPlus support](#).

If you purchased an Individual License, your license key will be permanently associated with the given e-mail address. You must use this e-mail address with your license key when registering.

If you purchased a Corporate License, your license key will be associated with an email domain. If you would like to [change the email domain](#), you must do so before the first key registration. When registering, you should use your corporate domain email address.

LogViewPlus is registered per user and therefore can be installed on multiple machines without the need to purchase multiple licenses provided the user of the application is the same person.

Once you have entered your registration details you can click the next button to proceed. The registration Wizard will then attempt automatic registration. Automatic registration should be successful assuming you're not attempting to register the application from behind a firewall or proxy server. If automatic registration fails the application will proceed to manual registration. Manual registration is discussed in the next section.

Having trouble with your license key? You can [verify your license key](#) is still valid.

Manual Registration

The screenshot shows a 'Registration Wizard' window with a title bar containing a close button (X). Below the title bar is a navigation bar with a back arrow icon and the text 'Registration Details'. The main content area is titled 'Manual Registration' and contains three steps:

- Step 1:** Please go to: <https://www.logviewplus.com/register.aspx>
- Step 2:** Copy the text below and paste it into the textbox provided on the webpage.
Below this instruction is a large, empty text box with a vertical scrollbar on the right side.
- Step 3:** Paste the response generated by the web page into the textbox below:
Below this instruction is another large, empty text box with a vertical scrollbar on the right side.

At the bottom right of the window, there are two buttons: 'Next >' and 'Cancel'.

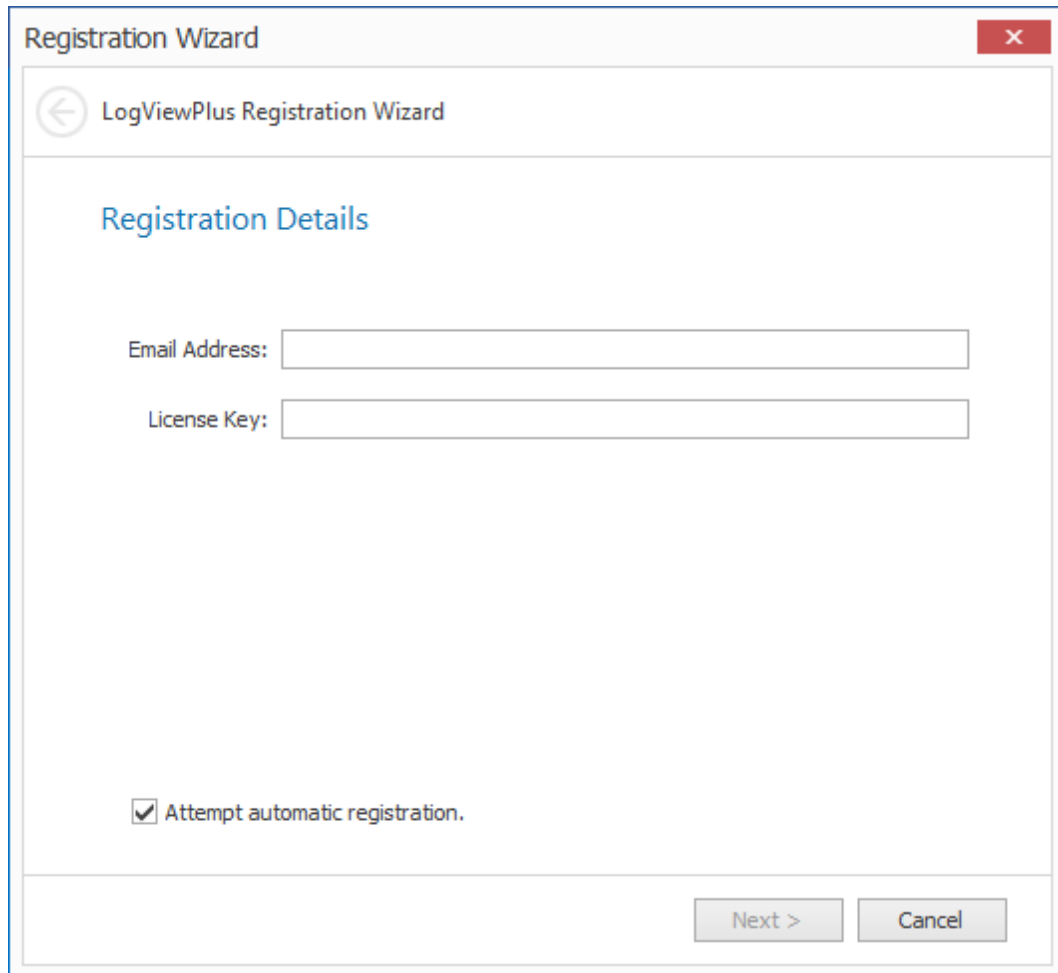
Manual registration is the process where you register LogViewPlus on a given machine by feeding data into the online registration form. This is a three-step process:

First you need to go to the LogViewPlus [registration page](#) and provide your license key.

Your license key is not contained in the e-mail you received from LogViewPlus support. Instead a license key is generated dynamically from within LogViewPlus based on the information you received from LogViewPlus support. Your license key is shown in the "Step 2" text box on the manual registration page (note that the license key has been removed from the screenshot above). Copy your license key and paste it into the text box provided on the registration webpage and press submit.

The webpage will generate the confirmation code needed by LogViewPlus to complete the registration process. Copy the confirmation code provided by the webpage and paste it into the text box provided in step 3. Then click the next button to complete the registration process.

Registration Complete



The image shows a Windows-style dialog box titled "Registration Wizard" with a red close button in the top right corner. Inside the dialog, there is a sub-header "LogViewPlus Registration Wizard" with a back arrow icon. Below this, the section "Registration Details" is displayed. It contains two input fields: "Email Address:" and "License Key:". At the bottom of the details section, there is a checked checkbox labeled "Attempt automatic registration.". At the very bottom of the dialog, there are two buttons: "Next >" and "Cancel".

Registration Wizard

LogViewPlus Registration Wizard

Registration Details

Email Address:

License Key:

☒ Attempt automatic registration.

Next > Cancel

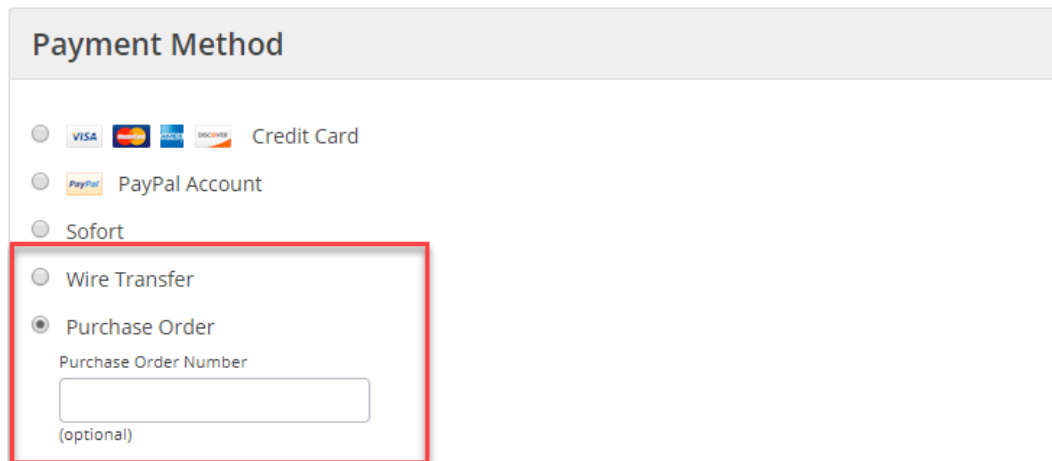
Once you've completed the LogViewPlus registration process you should see the page shown above. If you have any difficulty registering LogViewPlus please don't hesitate to [contact us](#).

Generating a Quote

FastSpring is the official reseller of LogViewPlus. If you are considering purchasing LogViewPlus, you can use their automated system to generate an invoice or quote if needed.

To generate a quote go to the LogViewPlus [purchase page](#) and select the type of license you are considering. Next, enter the quantity needed as well as your company details such as address and contact email address.

When generating a quote, it's important to select the "Wire Transfer" or "Purchase Order" option from the "Payment Method" menu as shown below. The primary difference between these two options is that "Purchase Order" will allow you to enter your purchase order number which will then be present in the final receipt.



The screenshot shows a 'Payment Method' selection form. It includes radio buttons for 'Credit Card', 'PayPal Account', 'Sofort', 'Wire Transfer', and 'Purchase Order'. The 'Purchase Order' option is selected and highlighted with a red rectangular box. Below the 'Purchase Order' option is a text input field labeled 'Purchase Order Number' with the text '(optional)' underneath it.

Next, you will be given the option of supplying a VAT ID or coupon if applicable. Finally, click the "Complete Order" button.

This will generate a quote and provide you with instructions on how to make a bank wire transfer. More importantly, you will receive an email which contains a link to an HTML version of the quote. This link will be in a format similar to: <https://sites.fastspring.com/clearcove/order/invoice/CLE000000-0000-00000>

If you would prefer a PDF quote, simply add '/pdf' to the end of the URI. For example: <https://sites.fastspring.com/clearcove/order/invoice/CLE000000-0000-00000/pdf>

Quotes generated through this process represent a non-binding intent to purchase and are valid for 45 days. Also, quotes do not need to be paid by wire transfer. Instead, you will find a link in both the email and online invoice which allows you to 'Pay Now'. This link will let you pay the invoice using a range of payment options including credit card or PayPal.

License keys will only be sent once FastSpring has received payment. If you decide to pay the invoice via wire transfer, please note that this may cause delays in receiving your license key. If you have paid by wire transfer and have not yet received your license key, we recommend waiting 3 - 5 days. If you would like us to follow-up with FastSpring please don't hesitate to [contact us](#).

Finally, please note that you are purchasing LogViewPlus from FastSpring and not Clearcove Limited. Additional information about FastSpring including VAT number, DUNS, and W-9 can be [found here](#).

Trust and Networking

LogViewPlus is [virus and spyware free](#). We do not store any of your data beyond the basic use cases described in our [privacy policy](#) - such as replying to emails and tracking purchases.

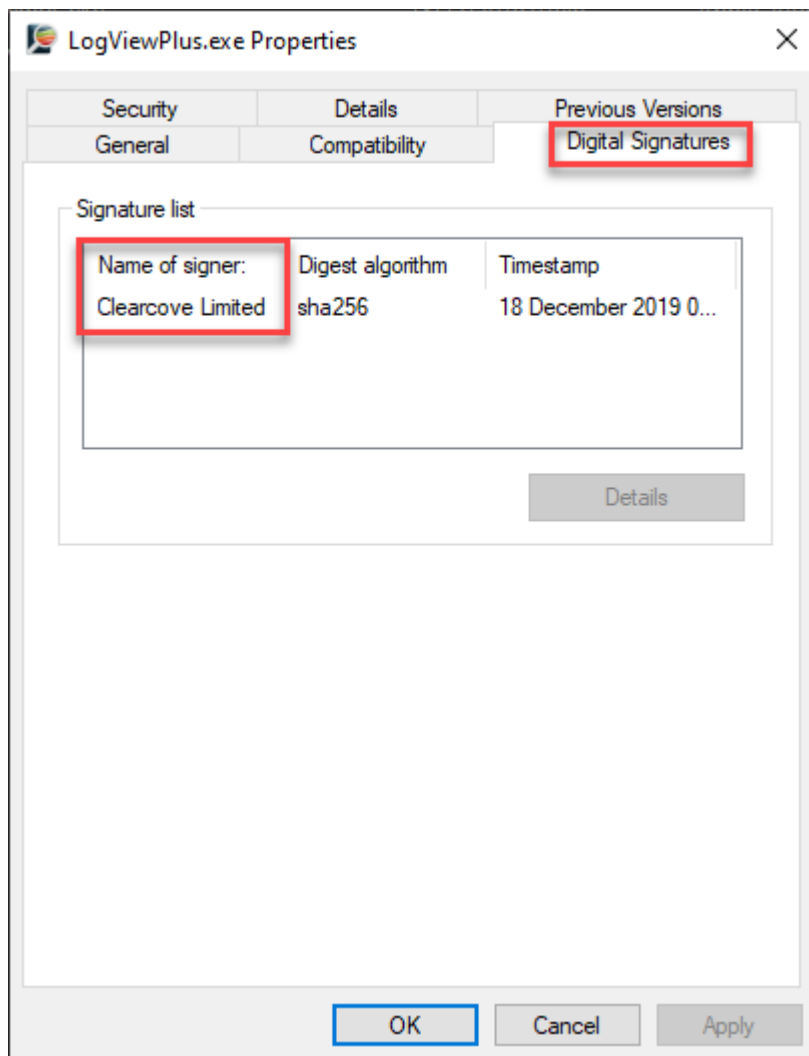
In addition to standard tools like [antivirus](#), here are three additional heuristics you can use to determine if LogViewPlus can be trusted.

1. We are accountable. [Clearcove Limited](#) is registered in England and has been doing business online since 2006. Our company number is 6030104 which you can verify with the [UK government](#). Clearcove is governed by English and European law - including GDPR.

LogViewPlus also licenses code from professional third party companies including Microsoft, DevExpress, RedGate and Rebex.

Accountability is a very important security guarantee that is not generally provided by open source software. LogViewPlus does not use open source software with the exception of the tar libraries from SharpZipLib. This code was manually reviewed before a one-time import.

2. All of our code is signed. You can verify this by opening the file properties on any of our executables. The file properties should include a "Digital Signatures" tab.



All of our executable code is signed with the name "Clearcove Limited". This signature is a cryptographically secure way of guaranteeing the code you are running is Clearcove Limited's responsibility. One advantage of a closed source application is that the chain of responsibility is crystal clear.

LogViewPlus should only be downloaded from LogViewPlus.com. If you do not see the "Clearcove Limited" signature on your executable, please [let us know](#) and DO NOT execute the software. Cracked software will be missing this signature.

3. Verify that LogViewPlus runs offline. LogViewPlus is a tool used by security professionals and is designed to run without a network connection.

You can verify LogViewPlus is running offline by checking the ports with netstat. This is done by executing two commands:

tasklist | findstr LogViewPlus.exe - Finds the process ID (PID) for the application. This command is case sensitive.

netstat -aon | findstr [PID] - Finds all ports currently used by the application.

Executing these two commands from the command line will show that, by default, LogViewPlus has only one port open:

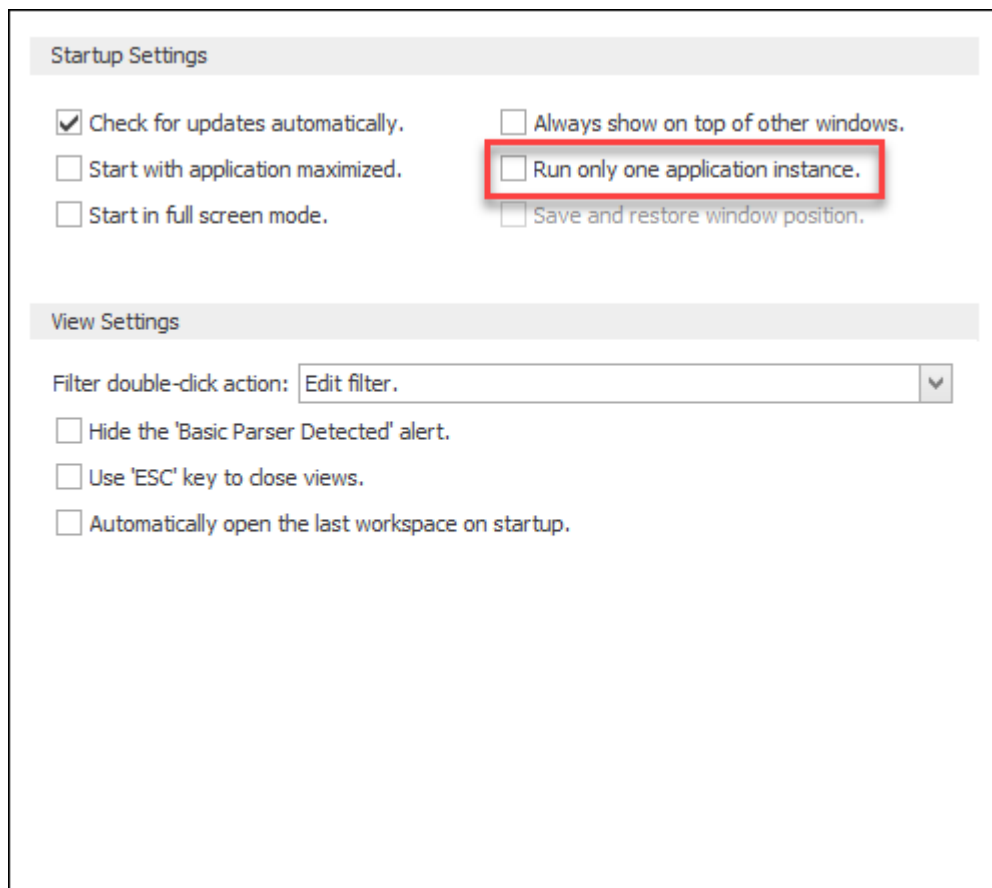
```
C:\>tasklist | findstr LogViewPlus.exe
LogViewPlus.exe           16444 Console                6      18,884 K

C:\>netstat -aon | findstr 16444
TCP        127.0.0.1:60652          0.0.0.0:0                LISTENING        16444

C:\>
```

The port is listening on the loopback address - so it is only accessible to processes running on the local machine. The port number is random and will likely change with each application restart.

This loopback port is used to run LogViewPlus in "single instance" mode. You can disable this listening port by allowing multiple application instances. This is done by deselecting the "Run only one application instance" checkbox in [Application Settings](#):



The screenshot shows the 'Startup Settings' and 'View Settings' sections of the LogViewPlus application. In the 'Startup Settings' section, the checkbox for 'Run only one application instance.' is highlighted with a red rectangle. Other settings include 'Check for updates automatically.' (checked), 'Start with application maximized.', 'Start in full screen mode.', 'Always show on top of other windows.', and 'Save and restore window position.' (all unchecked). The 'View Settings' section includes a 'Filter double-click action:' dropdown set to 'Edit filter.', and three unchecked checkboxes: 'Hide the 'Basic Parser Detected' alert.', 'Use 'ESC' key to close views.', and 'Automatically open the last workspace on startup.'

Startup Settings	
<input checked="" type="checkbox"/> Check for updates automatically.	<input type="checkbox"/> Always show on top of other windows.
<input type="checkbox"/> Start with application maximized.	<input type="checkbox"/> Run only one application instance.
<input type="checkbox"/> Start in full screen mode.	<input type="checkbox"/> Save and restore window position.

View Settings	
Filter double-click action: Edit filter.	
<input type="checkbox"/> Hide the 'Basic Parser Detected' alert.	
<input type="checkbox"/> Use 'ESC' key to close views.	
<input type="checkbox"/> Automatically open the last workspace on startup.	

After making this change, you can save your settings and restart LogViewPlus. There will now be no ports in use:

```
C:\>tasklist | findstr LogViewPlus.exe
LogViewPlus.exe           3976 Console             6      20,216 K

C:\>netstat -aon | findstr 3976

C:\>
```

The only other time that you will see ports listed for LogViewPlus is if you have taken a user action which is obviously network related. For example, monitoring a remote data source, registering, or checking for application updates. LogViewPlus may also check for

application updates shortly after start up. This check can also be disabled in [Application Settings](#).

Antivirus Analysis

LogViewPlus currently has [no outstanding issues](#) with any antivirus provider. However, LogViewPlus is frequently updated. New updates may lead to suspicion from your anti-virus software until the update is fully trusted and the trust has been propagated. In some cases, propagation may take several days and during this time your anti-virus may take longer to analyze LogViewPlus. This is expected behavior as the anti-virus has no prior knowledge of the executable. False positives may also occur during this time, but in our experience these are rare.

If your antivirus provider is blocking LogViewPlus, please [let us know](#).

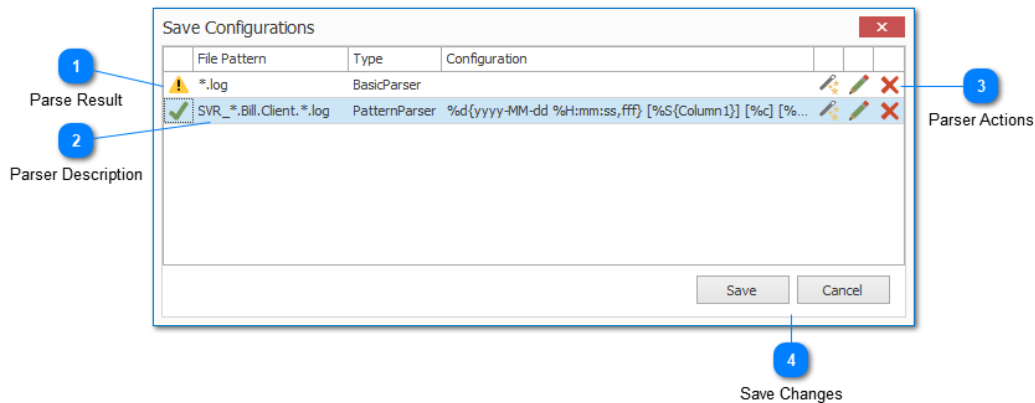
Parsing Log Files

LogViewPlus is an unusual application because it does not write log files and therefore has no pre-existing knowledge of the file format it needs to process. LogViewPlus may be able to dynamically determine the format of your log file, but for performance and data flexibility reasons you are almost always better off manually configuring your log parsers.

Log parsers and parser configuration are the subject of this section. If you are new to configuring log parsers we recommend using the [Parser Wizard](#).

If you are already familiar with log parsing, consider jumping ahead to [Analyzing Log Files](#).

Automatic Configuration



Automatic parser configuration is designed to give you a head start in creating a valid parser for your log file by dramatically lowering the learning curve. However, it's still useful to understand how [parser configurations](#) work. In particular, it's important to understand how LogViewPlus assigns data to columns.

If LogViewPlus is able to automatically generate a parser configuration it will display the dialog show above which will give you an opportunity to modify and save the generated configuration. Multiple configurations can be managed at the same time. Saved configurations can be removed in the [Parser Mappings](#) application settings.

There are a couple of things you should double check in the automatically generated configuration before saving. If you are familiar with configuring parsers, then you can easily review these settings in the [Parser Configuration Dialog](#) by clicking on the pencil icon.

First, does the filename pattern makes sense? The generated filename pattern will automatically wildcard numbers and symbols which may be detected in the log file name.

Often, this pattern can be made more generic to match a wider variety of filenames. For example, in the screenshot above the name SVR_*.log would likely be sufficient and match a wider variety of files.

Second, do the column names suggested by the generated configuration make sense? If the generated configuration suggests string columns (prefixed by %s), then generic column names will be used. For example Column1 and Column2. These names should be modified to better identify your data. It may help to find out more about the [%S specifiers](#).

Finally, the automatically generated configuration will attempt to identify the thread and logger columns based on inbuilt heuristics. This best effort approach can be problematic and you should review the decisions made by LogViewPlus to ensure they are correct.

1 Parse Result



The parse result is indicated by an icon located in the far left of the dialog. Hovering over this icon will provide a tooltip with a detailed description of the parse result. This message can also be seen by clicking on the icon.

2 Parser Description

File Pattern	Type	Configuration
*.log	BasicParser	
SVR_*.Bill.Client.*.log	PatternParser	%d{yyyy-MM-dd %H:mm:ss,fff} [%S{Column1}] [%c] [%...

A brief description of the pattern is shown including the file name pattern suggested, the type of parser used, and the configuration used for the parser. Double clicking on this description will allow you to edit the parser configuration settings in the [Parser Wizard](#).

3 Parser Actions



Three actions are available in the far right of the dialog. These actions allow you to:

1. Edit the parser configuration using the [Parser Wizard](#).
2. Edit the parser configuration using the standard [Parser Configuration Dialog](#).
3. Remove the configuration. If the configuration is removed, it cannot be saved.

4 Save Changes

Save	Cancel
------	--------

If you are happy with the parser configurations, you can save them to your [application settings](#). Saving the parser configurations will prevent this dialog from appearing the next time you open a file matching one of the file name patterns provided.

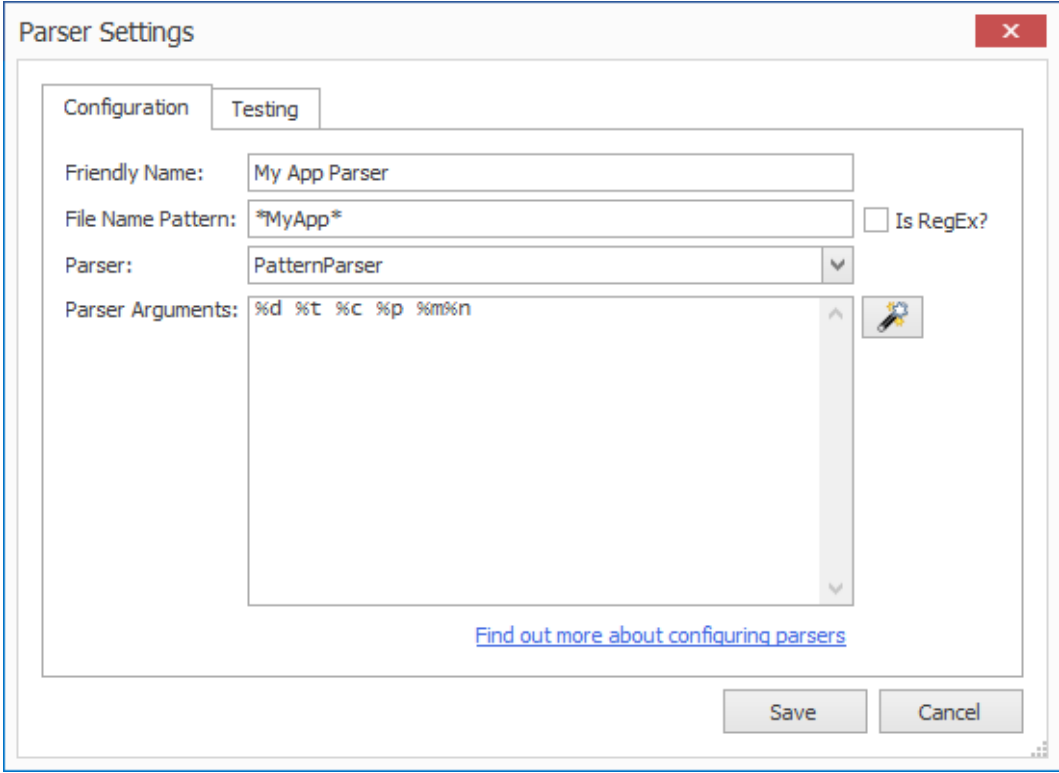
You can also cancel the changes if you do not want to save the parser configurations.

Log Parsers

LogViewPlus gathers information about your log files by parsing the log entries. It does this by identifying a log file by name and pairing it with a parser. You can associate a parser with a log file by going to Settings -> [Parser Mappings](#).

For example, say your application writes log entries to a file named 15.6.2014_MyApp_1.log. You can associate a parser to all your application log files with the pattern *MyApp*. In this example, the wildcard '*' is being used to match multiple unspecified characters. For more advanced patterns consider using a regular expression by checking the 'Is Regex' checkbox. This will allow the filename pattern text box to contain a full regular expression.

The 'Friendly Name' field will generally match the File Name Pattern, but you can give your parser a custom name to make it easier to identify in the future.



The screenshot shows the 'Parser Settings' dialog box with two tabs: 'Configuration' and 'Testing'. The 'Configuration' tab is active. It contains the following fields:

- Friendly Name:** A text box containing 'My App Parser'.
- File Name Pattern:** A text box containing '*MyApp*'. To its right is a checkbox labeled 'Is Regex?' which is currently unchecked.
- Parser:** A dropdown menu showing 'PatternParser'.
- Parser Arguments:** A text box containing '%d %t %c %p %m%n'. To its right is a small icon of a wrench and screwdriver.

At the bottom of the dialog, there is a blue hyperlink that reads 'Find out more about configuring parsers'. Below the dialog, there are 'Save' and 'Cancel' buttons.

If you want to use the same parser with multiple log files, you can use the pipe character - | - to separate file names. For example, *MyApp1*|*MyOtherApp*.

Once you have determined the pattern you want to use to identify your log files, you need to determine the appropriate parser to process your log files. By default LogViewPlus has six different parsers:

[Basic Parser](#) - This is the default parser for LogViewPlus. No configuration is necessary. It is useful when viewing log files that generally conform to the Apache Log4 standard. This parser is not appropriate for more complex log files or when speed is a concern.

[Pattern Parser](#) - For most log files, this is the parser you want to use. It is fast and provides you with the maximum amount of information. Unfortunately, it requires a bit of configuration. To learn more, start with the [Pattern Parser](#) section and then read more about the [Conversion Specifiers](#) including [Specifier Basics](#) and the [Date Specifier](#).

[JSON Parser](#) - Are your log files in a custom JSON format? If so, this is the parser for you.

[XML Parser](#) - Are your log files in a custom XML format? If so, this is the parser for you.

[Log4Xml Parser](#) - If you write log files with one of the Apache Log4 XML Appenders then this is the parser to use. No configuration is necessary. If LogViewPlus is asked to open an XML file without any further information, then this parser will be used by default.

[DSV Parser](#) - The Delimiter Separated Value Parser is for log files in a delimited format, such as comma, tab or pipe separated values. For example, if your log file is also a CSV file.

[Regex Parser](#) - The [Regular Expression Parser](#) is only recommended for users who are already familiar with Regular Expression. For most users, the [Pattern Parser](#) will be easier to configure.

Of course, if none of the provided parsers suit your needs, you can consider [writing your own log parser](#).

Basic Parser

The Basic Parser does a simple scan on every entry in your log file and analyses it on a best effort basis. The Basic Parser works best on log entries that generally conform to the Apache Log4 standard (with fields like date, level, and message).

The advantage of the Basic Parser is that it requires no configuration. The disadvantage is that it is forced to make assumptions. These assumptions make the parser slower and more error prone. For these reasons, we always recommend using the [Pattern Parser](#) where possible.

The basic parser looks at each entry and tries to identify three things:

1. The date and time the log entry was written. **This field is required.** If LogViewPlus cannot identify this field, the file will not be parsed successfully (see below).
2. The level for this log entry. For example, Debug, Info, Warn, Error, and Fatal. This field is optional.
3. The log entry message. This is a distinct field which occurs after the date and log level fields.

The Basic Parser uses a number of different techniques to identify the timestamp. However, in the event that the timestamp cannot be identified, a warning will be displayed stating that the log file was not parsed correctly. When a timestamp is ambiguous, the Basic Parser will assume an international date format instead of a US date format (i.e, dd/MM/yyyy vs MM/dd/yyyy).

The basic parser requires no configuration and will be used by default if no file mapping is found.

Pattern Parser

The Pattern Parser uses a conversion pattern to read a log file. If you are familiar with the Apache Log4 libraries, you are probably already familiar with [Pattern Layouts](#). It is basically the same in LogViewPlus. In fact...

+__ If you use a pattern layout for writing your log file, you can probably use this as your conversion pattern for parsing your log file. __+

To find out more about conversion patterns, let's start with a simple log entry:

2014-09-13 10:50:14,125 [Thread 1] INFO MyApp - My message...

Here we have a simple entry made up of five parts: date, thread, log level, logger, and the message. We need a way to tell LogViewPlus about each part of our log entry as well as how to tell when one part ends and the next begins. To do this, we are going to use patterns.

Now, back to our log entry. Let's break our log entry down piece by piece:

Field	Example	Conversion Specifier
Date	2014-09-13 10:50:14,125	%d
<i>Literal</i>	" ["	" ["
Thread	Thread 1	%t
<i>Literal</i>] "] "
Priority	INFO	%p
<i>Literal</i>	" "	" "
Logger	MyApp	%c
<i>Literal</i>	" - "	" - "
Message	My Message...	%m
<i>Literal</i>	New Line.	%n

If we put all of the [conversion specifiers](#) together we have:

%d [%t] %p %c - %m%n

If the sample log entry above matched your log entries, you could use the pattern we created to parse your log files. Unfortunately, this is probably not the case as every log file is different. You will need to figure out the conversion pattern that works for your log files and this may require a bit of trial and error. The [Parser Wizard](#) can help you create and test an appropriate conversion pattern.

Once you have decided on the appropriate [conversion pattern](#), you will need to provide that as an argument to the Pattern Parser.

Parser Settings

ConfigurationTesting

Friendly Name:My App Parser

File Name Pattern:*MyApp*

Is RegEx?

Parser:PatternParser

Parser Arguments:%d %t %c %p %m%n

Find out more about configuring parsers

SaveCancel

[Conversion patterns](#) can be a bit tricky at first, so here is a cheat sheet. For getting the most out of LogViewPlus, you only really need to know 7 different types of conversion specifiers.

Field	Specifier	Notes
Date	%d	Dates can be tricky because there are so many ways they can be represented. See Date Patterns for more.

Thread	%t	Thread names can contain spaces, so you need a non-spaced literal separating the thread from other fields. For example, brackets in "[my thread]".
Priority	%p	DEBUG, INFO, WARN, ERROR, etc...
Logger	%c	The logger responsible for writing the log entry. Very helpful when filtering.
Message	%m%n	When using the Pattern Parser, these two conversion specifiers almost always go together.
Any Word	%s	Cheat #1 - match any string without spaces.
Multiple Words	%S	Cheat #2 - match any array of strings spaces included. Note this field is upper-case.

Check out [Advanced Specifiers](#) for the full list of supported conversion specifiers.

Why are %s and %S cheats? Two reasons: First, they are specific to LogViewPlus and not supported by standard logging frameworks like Apache Log4. Second, the %s specifiers can optionally take a parameter which specifies the name of the column which should display the field. To do this, you just need to enclose the column name in curly brackets. For example: **%s{My Column Name}**. This ability to take any field and convert it into a LogViewPlus column makes these specifiers extremely powerful. The %s specifiers are the only conversion specifiers which do not have a predefined column name.

If the optional column name is not specified, then LogViewPlus will not know how to display the field. You will still be able to use LogViewPlus to search for this text, but it will not be shown in a dedicated column in the log entry grid.

You can find out more about [%s specifiers](#) by seeing them used to parse an IIS log file.

Now let's go through a few quick examples using just the specifiers listed above. For simplicity, these examples will use timestamps only. Dates are discussed in detail in the [Date Specifier](#) section.

<p>Log Entry: 07:36 My message...</p> <p>Pattern: %d %m%n</p> <p>Notes: The pipe character ' ' is used to separate date and message.</p>

Log Entry:	10:50 [Thread 1] INFO property1 - My message...
Pattern:	%d [%t] %p %s - %m%n
Notes:	The 'any word' specifier '%s' is used to skip over the unknown field 'property1'.
Log Entry:	10:50 [Thread 1] INFO ndc1 ndc2 ndc3 - My message...
Pattern:	%d [%t] %p %S - %m%n
Notes:	Skipping over multiple fields with the %S specifier.
Log Entry:	10:50 Thread 1 - My message...
Pattern:	%d %t - %m%n
Notes:	The multi-word thread name can be parsed thanks to unique literal ' - '.
Log Entry:	10:50 240 1 2 LEVEL: Info 3
Pattern:	%d %S LEVEL: %p %m%n
Notes:	Advanced literal ' LEVEL: ' used to specify a field. Everything prior to the literal is skipped with the multi-word specifier '%S'.
Log Entry:	[10:50] [notice] Apache/1.3.29 (Unix) server configured -- resuming operations
Pattern:	[%d] [%p] %s (%s) %S -- %m%n
Notes:	Good use of literals to separate strings. Ignoring non-pertinent information.

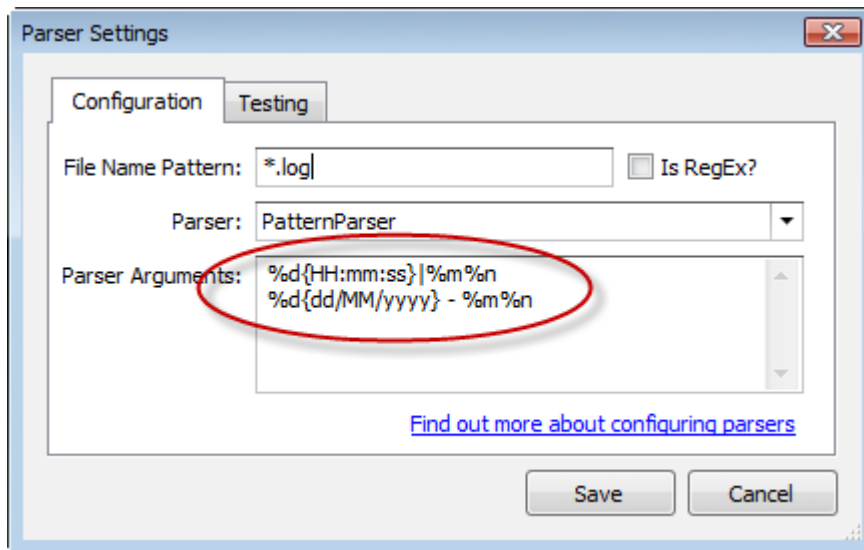
To find out more about conversion patters, please see [Conversion Specifiers](#).

Finally, please note that it is possible for a log file to have multiple patterns. To find out more about parsing log files with multiple patterns, please see [Multi Patterns](#).

Multi-Patterns

It is very common to have one logging format per log file. However, this is not a hard requirement and it is possible to have multiple logging formats in a single log file.

The [Pattern Parser](#) and [Regex Parser](#) allow the possibility of multiple conversion patterns, one per line, as can be seen in the screenshot below:



Multi-patterns cannot be configured using the [Parser Wizard](#). You must use [manual parser configuration](#) instead.

When using multiple conversion patterns, it's important to keep a few things in mind. First, using multiple patterns will always be slower than using a single pattern. This is because the parser will expect a degree of error when parsing the log file and a single log line will often be parsed multiple times.

When using multi-patterns the first pattern which successfully parses the log line will be used. Therefore, it is best to put more generic patterns which match log fields more broadly at the bottom of the pattern list. For example, '%d %m%n' would match most log entries and therefore should be placed near the bottom.

Patterns which span multiple lines, such as those which declare '\n' as part of the static text, are not supported in multi-patterns.

Finally, when using multi-patterns, it is important that the individual patterns are distinct. For example, if you were using two patterns and every log entry could be parsed by either pattern successfully then LogViewPlus will not know which pattern to use. 'Successful' in this case means that the log entry is parsed without error and does not necessarily mean that data is placed in the correct column.

JSON Parser

LogViewPlus has a built in JSON parser which is capable of analyzing your JSON log files. It does this by parsing your JSON file according to a template. A template is a sample JSON log entry that has certain fields identified with [Conversion Specifiers](#).

Let's look at a simple JSON example:

```
{
  "firstName":"John",
  "lastName":"Doe",
  "employeeId":"12345",
  "other":"ignore me",
  "dateJoined":"2014-05-16 10:50:14,125"
}
```

This is a JSON log entry with five fields: firstName, lastName, employeeId, other, and dateJoined. What we need to do is replace the field data with a [Conversion Specifier](#) that identifies the field data type. This might give us the following mapping.

JSON Field	Conversion Specifier	LogViewPlus Column
firstName	%S{First Name}	First Name
lastName	%S{Last Name}	Last Name
employeeId	%s{Employee Id}	Employee Id
other		We want to ignore this field.
dateJoined	%d	Date and Time

Therefore, we could parse this JSON log entry with the template:

```
{
  "firstName":"%S{First Name}",
  "lastName":"%S{Last Name}",
  "employeeId":"%s{Employee Id}",
  "dateJoined":"%d"
}
```

Notice that in the above template the "other" field has been ignored. To ignore a field we simply do not include it in our template. If one of the elements we were interested in had

been a child of a parent node, we would have needed to include the parent node in our template. The important thing is that the template has the full path to the target node.

Once we load this template into LogViewPlus it will appear as:

First Name	Last Name	Employee Id	Date
John	Doe	12345	16 May 2014

To do this, we just need to give LogViewPlus our parsing template as an argument for the JSON parser. We can do this in [Parser Mappings](#):

Parser Settings

Configuration Testing

Friendly Name: *.json

File Name Pattern: *.json ☐ Is RegEx?

Parser: JsonParser

Parser Arguments: {
 "firstName": "%S{First Name}",
 "lastName": "%S{Last Name}",
 "employeeId": "%s{Employee Id}",
 "dateJoined": "%d"
}

[Find out more about configuring parsers](#)

Save Cancel

Whitespace will be ignored, so we are free to format the JSON as needed.

Log files parsed with the JSON parser support automatic pretty-printing.

Finally, notice the similarities between the JSON Parser and the [XML Parser](#) discussed in the next section. Both use the concept of templates, so once you have learned one you have basically learned the other.

XML Parser

LogViewPlus has a built in XML parser which is capable of analyzing your custom XML log files. It does this by parsing your XML file according to a template. A template is a sample XML log entry that has certain fields identified with [Conversion Specifiers](#).

Let's look at a simple XML example:

```
<xml>
  <name firstName="John" lastName="Doe" />
  <employeeId>12345</employeeId>
  <other attr1="ignore">ignore</other>
  <dateJoined>2014-05-16 10:50:14,125</dateJoined>
</xml>
```

This is a XML log entry with five fields: firstName, lastName, employeeId, other, and dateJoined. What we need to do is replace the field data with a [Conversion Specifier](#) that identifies the field data type. This might give us the following mapping.

XML Field	Conversion Specifier	LogViewPlus Column
firstName	%S{First Name}	First Name
lastName	%S{Last Name}	Last Name
employeeId	%s{Employee Id}	Employee Id
other		We want to ignore this field.
dateJoined	%d	Date and Time

Therefore, we could parse this XML log entry with the template:

```
<xml>
  <name firstName="%S{First Name}" lastName="%S{Last Name}" />
  <employeeId>%s{Employee Id}</employeeId>
  <dateJoined>%d</dateJoined>
</xml>
```

Notice that in the above template the "other" field has been ignored. To ignore a field we simply do not include it in our template. If one of the elements we were interested in had been a child of a parent node, we would have needed to include the parent node in our template. The important thing is that the template has the full path to the target node.

Once we load this template into LogViewPlus it will appear as:

First Name	Last Name	Employee Id	Date
John	Doe	12345	16 May 2014

To do this, we just need to give LogViewPlus our parsing template as an argument for the XML parser. We can do this in [Parser Mappings](#):

Parser Settings

Configuration Testing

Friendly Name:

File Name Pattern: ☐ Is RegEx?

Parser:

Parser Arguments:

```
<xml>
<name firstName=\"%S{First Name}\" lastName=\"%S{Last Name}\" />
<employeeId>%s{Employee Id}</employeeId>
<dateJoined>%d</dateJoined>
</xml>
```

[Find out more about configuring parsers](#)

Save Cancel

Whitespace will be ignored, so we are free to format the XML as needed.

Log files parsed with the XML parser support automatic pretty-printing.

Finally, notice the similarities between the XML Parser and the [JSON Parser](#). Both use the concept of templates, so once you have learned one you have basically learned the other.

Log4Xml Parser

Use the Log4Xml parser if you are using the Apache Log4 Libraries (Log4Net, Log4J, Log4Php, etc...) with one of the prebuilt XML appenders. The Log4 libraries have more or less standardized in the output XML format which means that the LogViewPlus Log4Xml parser requires no configuration.

You can think of the Log4Xml parser as being exactly the same as the XML Parser, but with a prebuilt configuration that makes it easy to work with Log4 XML files.

The Log4Xml parser will be used by default when LogViewPlus is asked to open an XML file and no other parser configurations are matched. In other words, if you ask LogViewPlus to open an XML file, LogViewPlus will assume it is in the Log4 XML format unless the program is specifically told otherwise.

DSV Parser

DSV stands for [delimiter separated values](#). We created the DSV Parser primarily because we needed a "CSV" parser - a log parser that was capable of reading comma separated value (CSV) files. However, as soon as our "CSV" parser was completed, we realized that we needed another parser for tab separated files. Also, what if the customer wants to use the pipe character to separate values? Or a tilde?

The DSV parser solves these problems by reading the separating character from the parser arguments. When reading the parser arguments LogViewPlus will assume the first non-space character which is not part of a [conversion specifier](#) is the character which should be used to separate values.

For example, consider the following conversion pattern:

%d, %t, %p, %c, %m

Here, the first non-space character which is not part of a conversion specifier is a comma. So, a comma will be used as the separating character. Similarly, if we consider the pattern:

%d\t%t\t%p\t%c\t%m

We can see that the tab character '\t' is the first non-space character which is not part of a conversion specifier.

Why not just use the [Pattern Parser](#)? Why do we need a separate parser for delimiter separated values? This is a good question because certainly the Pattern Parser would be able to read and interpret the conversion patterns outlined above. However, a delimiter separated value file may have values which are surrounded by quotes and some fields may even contain the delimiter. Also, note that the above conversion patterns do not define a new line %n conversion specifier. In the case of a delimiter separated value file, the log entry is complete only when all expected fields have been read successfully. This means that fields can be multiline.

Fields that run multiple lines, and fields that contain the delimiter as part of their values must be enclosed in quotes. Quotes can be either single or double. This will be determined inline.

If the field starts with a quote character, then the same character will be the expected to close the field. To escape a quote character within a field value, you must use two quote characters back to back. For example, " (two singles) or "" (two doubles).

The above rules conform to how some programs, like Microsoft Excel, write DSV files.

For example, the following is a valid CSV entry:

"2014-05-16 10:50:14,125", Thread 1, INFO, MyLogger, "This is my ""LogMessage"", my app is running."

This log message could be parsed using the first conversion pattern given - **%d, %t, %p, %c, %m**. LogViewPlus will automatically remove all surrounding quotes, but whitespace will be preserved. In the above example, the Message field would contain:

**This is
my "LogMessage", my app is running.**

What about headers? Delimiter separated value files frequently use a header row to define the columns. For example, column headers for the log entry above might be: Date, Thread, Priority, Logger and Message. LogViewPlus will intelligently ignore column headers. They are not expected and not required. If found, they will be ignored. Delimiter separated files are always parsed according to the provided conversion pattern.

Event Log Parser

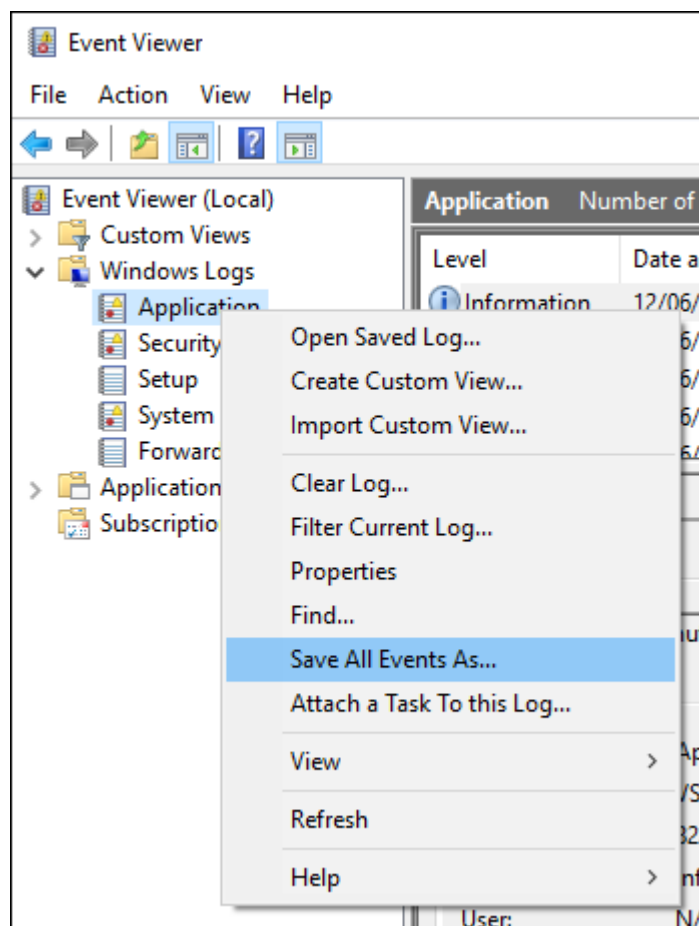
The event log parser can be used to parse *.evtx files. Evtx file parsing is based on the event log classes provided by Microsoft in the System.Diagnostics.Eventing.Reader namespace. Unfortunately, experience suggests that this library can be both slow and unreliable. For this reason, **the preferred solution for reading event log entries is via a CSV file export.**

This documentation will cover both a CSV file export as well as the event log parser. Reading log entries directly from the Windows Event Log is also supported. This is covered in the [Windows Events](#) documentation.

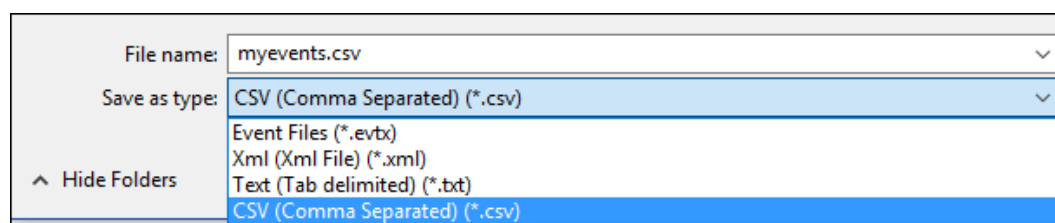
CSV File Export

To export your event log entries as a CSV file the first thing you need to do is open event viewer and select the log category that you want to export. Log entries will be exported as they appear in the log viewer grid. Changes made to column sorting will be preserved.

With your event viewer open right click on the target log category and select "Save All Events As...".

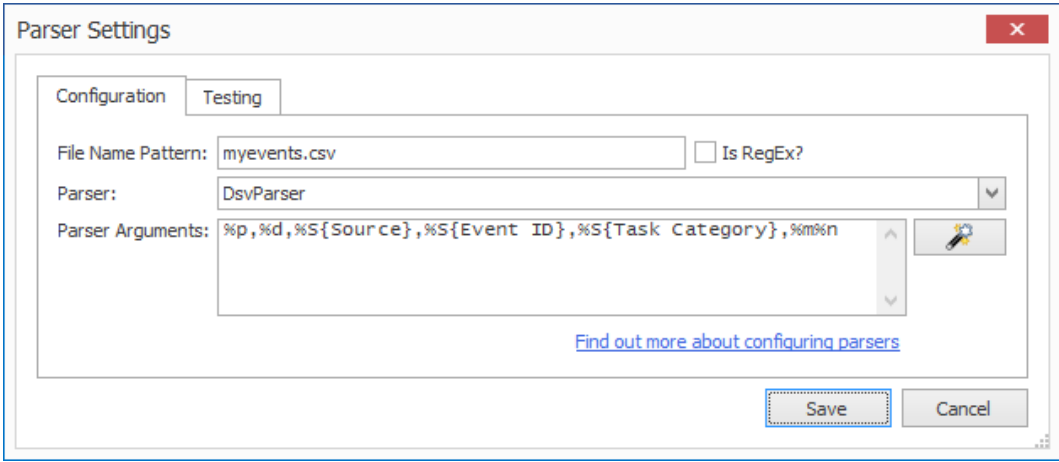


When prompted enter a name for your new CSV file and select "CSV (Comma Separated)" as the saved type.



Finally click "Save" to export the event log entries.

Before we open our new CSV log file in LogViewPlus, we need to configure the application so it can parse the CSV file. To do this go to Settings -> [Parser Mappings](#) and click 'Add'. In the [parser configuration dialog](#) enter a filename pattern which will match the file name given to your CSV file. Next, set the parser type to [DSV Parser](#) and parser arguments to: **%p,%d,%S{Source},%S{Event ID},%S{Task Category},%m%n** as shown.



The image shows the 'Parser Settings' dialog box with two tabs: 'Configuration' and 'Testing'. The 'Configuration' tab is active. It contains the following fields:

- File Name Pattern:** A text box containing 'myevents.csv'. To its right is a checkbox labeled 'Is RegEx?' which is unchecked.
- Parser:** A dropdown menu showing 'DsvParser'.
- Parser Arguments:** A text box containing '%p,%d,%S{Source},%S{Event ID},%S{Task Category},%m%n'. To its right is a small icon of a key and a lock.

Below the text boxes is a blue hyperlink that reads 'Find out more about configuring parsers'. At the bottom right of the dialog are two buttons: 'Save' and 'Cancel'.

Click Save followed by OK to save the parser settings.

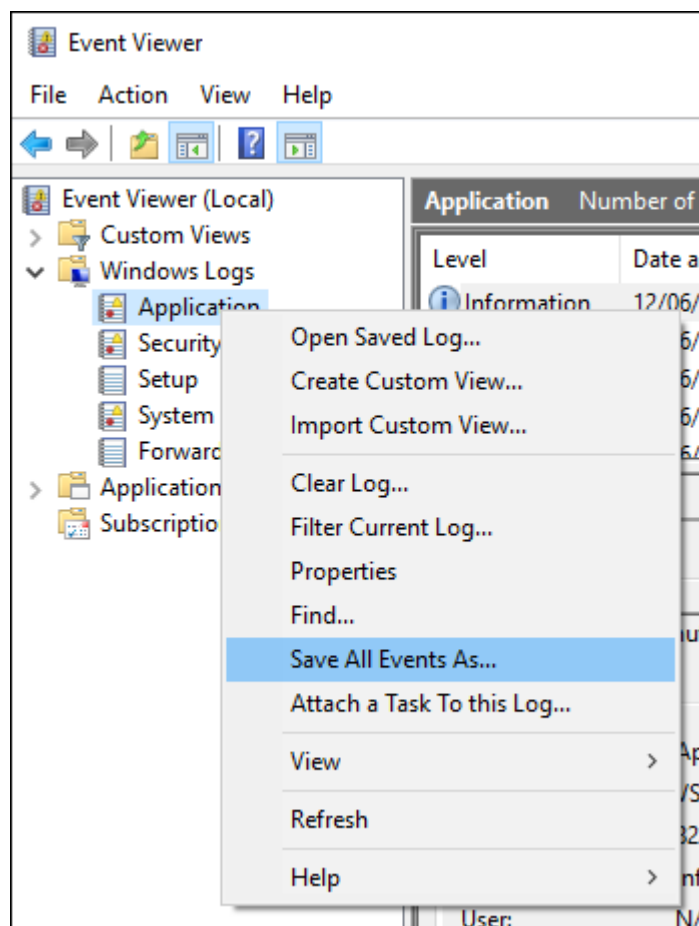
We are now ready to open the CSV export file we created earlier. Opening this file in LogViewPlus will show all of the exported events in the log entry grid. Any future CSV event log exports will need separate configuration if the filename patterns do not match.

Time	Source	Event ID	Task Cat...	Message
11:56:05.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.
11:12:58.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.
10:56:05.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.
10:12:58.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.
09:56:05.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.
09:12:58.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.
08:56:05.000	VSS	8224	None	The VSS service is shutting down due to idle timeout.

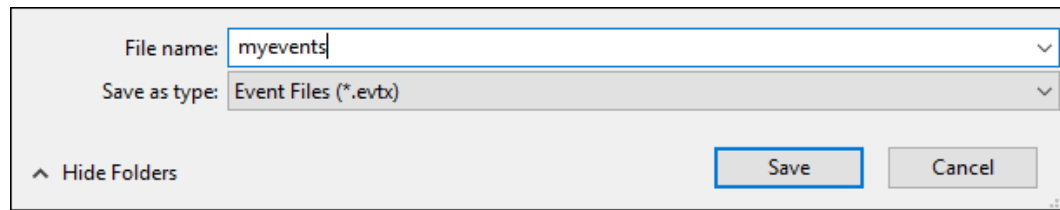
EVTX File Export

Please note that, as stated earlier, due to performance and reliability issues the preferred method for exporting event log entries is via CSV as discussed above.

To export your event log entries as an EVTX file the first thing you need to do is open event viewer and select the log category that you want to export. Next, right click on the target category and select "Save All Events As...".



When prompted enter a name for your new EVTX file and select "Event Files" as the saved type.



Finally click "Save" to export the event log entries. The exported EVTX file can be opened in LogViewPlus immediately without any further configuration.

Regex Parser

Our Regex Parser was introduced in LogViewPlus 2.3.5 to allow users who are already familiar with regular expressions a quick and easy way to configure parsers. However, the Regex Parser is not performant when compared to the [Pattern Parser](#). For this reason, we always recommend using the Pattern Parser - especially for large log files.

The idea behind the Regex Parser is to use a standard regular expression to parse the log file. In this case, the field names in the regular expression should be valid [Conversion Specifiers](#). If an unknown field is found, it will be assumed to be the name of a column and the string conversion specifier (%s) will be used.

When using conversion specifiers in the Regex Parser, you can use either the full name of the specifier or the [abbreviation](#). For example, the following regular expressions would be considered equivalent:

(?<date>.*?) - (?<priority>.*?) \[(?<thread>.*?)\] (?<logger>.*?) - (?<message>.*?)

(?<d>.*?) - (?<p>.*?) \[(?<t>.*?)\] (?<c>.*?) - (?<m>.*?)

Either of the regular expressions above could be used to parse a log like:

09 MAY 2019 10:50:14,125 - INFO [Thread_10] MyServer - My Server initializing...

All fields defined in the regular expression must be found on the same line of the log entry.

If the log entry contains a multi-line message, this will be processed automatically by the parser after the regular expression parse fails. In other words, if a line cannot be parsed, it will be assumed to be a continuation of the previous log entry's message.

The Regex Parser currently does not allow for custom date formats. Therefore, the date must be in a format which can be auto-detected by LogViewPlus. If you are able to open and parse the file using the [Basic Parser](#), then this should not be an issue. However, if you do have problems, please [contact support](#).

Conversion Specifiers

A conversion specifier is used to represent a single field within a conversion pattern. They are a special character sequence that is used by LogViewPlus to assign meaning to a given part of log entry. Conversion specifiers always begin with a percent character '%'. For example, consider the following conversion pattern:

%d, %t, %p, %c, %m

This conversion pattern contains five conversion specifiers. Conversion specifiers and their meanings will be covered in this section.

Conversion specifiers are used by five of the eight default log parsers that ship with LogViewPlus: [Pattern Parser](#), [JSON Parser](#), [XML Parser](#), [DSV Parser](#), and [Regex Parser](#).

A good understanding of conversion specifiers is important for almost all log parser configuration.

Specifier Basics

The conversion specifiers defined by LogViewPlus give you maximum flexibility when parsing your log files. At first glance, the full list of conversion specifiers can seem a little bit daunting. However, you can get the most out of LogViewPlus and parse any log file using just the seven specifiers defined below. This is your cheat sheet and learning these seven specifiers is time well spent.

Field	Specifier	Notes
Date	%d	Dates can be tricky because there are so many ways they can be represented. See Date Patterns for more.
Thread	%t	Thread names can contain spaces, so you need a non-spaced literal separating the thread from other fields. For example, brackets in "[my thread]".
Priority	%p	DEBUG, INFO, WARN, ERROR, etc...
Logger	%c	The logger responsible for writing the log entry. Very helpful when filtering.
Message	%m	The log message. When using the Pattern Parser, this specifier is almost always followed by the %n specifier.
Any Word	%s	Cheat #1 - match any string without spaces.
Multiple Words	%S	Cheat #2 - match any array of strings (spaces included). Note this field is upper-case.

Why are %s and %S cheats? Two reasons: First, they are specific to LogViewPlus and not supported by standard logging frameworks like Apache Log4. Second, the %s specifiers can optionally take a parameter which specifies the name of the column which should display the field. To do this, you just need to enclose the column name in curly brackets. For example: %s{My Column Name}. This ability to take any field and convert it into a LogViewPlus column makes these specifiers extremely powerful.

The %s specifiers are the only conversion specifiers which do not have a predefined column name. If the optional column name is not specified, then LogViewPlus will not know how to display the field. You will still be able to use LogViewPlus to search for this text, but it will not be shown in a dedicated column in the log entry grid.

You can find out more about [%s specifiers](#) by seeing them used to parse an IIS log file.

All conversion specifiers use the curly bracket syntax to define arguments - like {Arguments}. However the arguments supported by a given conversion specifier may be unique - as is the case with the %s specifier defined above. Arguments are discussed in more detail in the sections that follow.

Before moving onto the next section, it's worth reviewing the sample patterns discussed in [Pattern Parser](#). For simplicity, these examples will use timestamps only. [Date Specifiers](#) are discussed in detail in the next section.

Log Entry: 07:36|My message...

Pattern: %d|%m%n

Notes: The pipe character '|' is used to separate date and message.

Log Entry: 10:50 [Thread 1] INFO property1 - My message...

Pattern: %d [%t] %p %s - %m%n

Notes: The 'any word' specifier '%s' is used to skip over the unknown field 'property1'.

Log Entry: 10:50 [Thread 1] INFO ndc1 ndc2 ndc3 - My message...

Pattern: %d [%t] %p %S - %m%n

Notes: Skipping over multiple fields with the %S specifier.

Log Entry: 10:50 Thread 1 - My message...

Pattern: %d %t - %m%n

Notes: Multi-word thread name can be parsed thanks to unique literal ' - '.

Log Entry: 10:50|240 1 | 2 LEVEL: Info 3

Pattern: %d|%S LEVEL: %p %m%n

Notes: Advanced literal ' LEVEL: ' used to specify a field. Everything prior to the literal is skipped with the multi-word specifier '%S'.

Log Entry: [10:50] [notice] Apache/1.3.29 (Unix) server configured -- resuming operations

Pattern: [%d] [%p] %s (%s) %S -- %m%n

Notes: Good use of literals to separate strings. Ignoring non-pertinent information.

Check out [Advanced Patterns](#) for the full list of supported conversion patterns.

Date Specifier

By far, the hardest and most advanced conversion specifier is the date specifier - **%d**. It is the most complicated because it is doing the most difficult job. Date parsing is hard and it requires a keen eye for detail. For example, consider the date:

2014-03-23 13:46:45,566 +01:00 PM

LogViewPlus uses a standard date parsing technology developed by Microsoft called Date Format Strings. You can find out everything there is to know about date format strings by looking at [Microsoft's documentation](#).

Let's jump right in and break our example date down into its parts:

Field	Example	Pattern	Notes
Year	2014	yyyy	The pattern for parsing a year is 'y', and in this case we use it four times. The number of times it is used is relevant. See below for more details.
<i>Literal</i>	-	-	
Month	03	MM	Patterns are case sensitive. See "minute" below.
<i>Literal</i>	-	-	
Day	23	dd	
<i>Literal</i>	" "	" "	
Hour	13	HH	We are using a 24 hour clock. For a 12 hour clock, we would use the lower case "hh". See below for more details.
<i>Literal</i>	:	:	
Minute	46	mm	
<i>Literal</i>	:	:	
Second	45	ss	
<i>Literal</i>	,	,	
Millisecond	566	fff	
<i>Literal</i>	" "	" "	
Time zone	+01:00	zzz	

<i>Literal</i>	" "	" "
Period	PM	tt

Putting all of the pattern elements together gives us:

yyyy-MM-dd HH:mm:ss,fff zzz tt

We can use this pattern to parse our date. The final step is to pass our date conversion pattern into our conversion specifier. We do this using a special syntax of surrounding the argument in curly brackets - "{}". Our final conversion specifier is:

%d{yyyy-MM-dd HH:mm:ss,fff zzz tt}

Seems easy enough - right? The tricky thing about date format strings is that you really need to pay attention to the number of times you are using a pattern as repeating a pattern can have a very unusual effect. The tables below show the effect of using multiple patterns:

Year

Pattern	Matches	Notes
%y	9	%y would actually match '14' as well. The percent sign in this case is an indicator that a 'year number' may or may not be found in that position.
yy	09	
yyy	2009	
yyyy	2009	

Month

Pattern	Matches
%M	9
MM	09
MMM	Sep
MMMM	September

Day

Pattern	Matches
---------	---------

%d	9
dd	09
ddd	Tue
dddd	Tuesday

Hour

Pattern	Matches	Notes
%h	9	'9 AM or PM' on a twelve hour clock. If your timestamp includes a period (for example, AM / PM) then you should always use a lower case 'h'.
hh	09	
hhh	09	
hhhh	09	
%H	21	'9 PM' on a 24 hour click.
HH	21	
HHH	21	
HHHH	21	

Minute

Pattern	Matches
%m	9
mm	09
mmm	09
mmmm	09

Seconds

Pattern	Matches
%s	9
ss	09
sss	09
ssss	09

Milliseconds

Pattern	Matches
%f	9
ff	09
fff	009
ffff	0009

Time zone

Pattern	Matches	Notes
%z	+9	
zz	+09	
zzz	+09:00	
zzzz	+09:00	
ZZZ	GMT	LogViewPlus supports about 100 different three-letter-acronym time zones. This is indicated by an uppercase ZZZ.

Period

Pattern	Matches
%t	P
tt	PM
ttt	PM
tttt	PM

That is everything you need to know about parsing dates with LogViewPlus. Now, you just need to put the parts together in a way that matches the date format you use in your log files.

Here are a few examples to get you started:

Date String	Conversion Specifier
07:36:18:660761	%d{HH:mm:ss:ffffff}
2014-05-18-14.20.46.973000	%d{yyyy-MM-dd-HH.mm.ss.ffffff}

Dec 13 05:28:27	%d{MMM dd HH:mm:ss}
Sun Mar 7 16:02:00 2014	%d{ddd MMM d HH:mm:ss yyyy}
07/Mar/2004:16:06:51 -0800	%d{dd/MMM/yyyy:HH:mm:ss zzzz}
Sunday, 14 September 2014 10:50	%d{dddd, dd MMMM yyyy HH:mm}
2014-09-14T14:02:45.0174665+01:00	%d{yyyy-MM-ddTHH:mm:ss.ffffffzzzz}

Finally, if your timestamp does not include a date and your parser has not been configured to provide a date, then LogViewPlus will assume today's date. All log entries in LogViewPlus must have a date.

Using %d Without Arguments

Finally, the date specifier has a default form which does not take any arguments. You'll notice that the documentation frequently uses the date specifier without any arguments in order to simplify the discussion. When you use the date specifier without any arguments you are asking LogViewPlus to make a best effort attempt at parsing the date. To do this, it will use the same date parsing technology which is used by the basic parser. This is functionally correct and can work in many different situations. However, it has two distinct disadvantages:

1. The date and time will be parsed on a best effort basis. This means that the parsing is not guaranteed to work, and when it does work the date time value may appear incorrectly.
2. In attempting to dynamically determine the date format, LogViewPlus has to do significantly more work which may have a performance impact.

Because of these disadvantages, we recommend you specify the date format explicitly whenever possible.

Elapsed Date Times

Dates and times are occasionally represented numerically. For example, the number of milliseconds since the UNIX epoch (Jan 1, 1970). LogViewPlus supports the following built in conversion specifiers for working with numeric dates.

Conversion Specifier	Description
----------------------	-------------

<code>%d{Elapsed}</code>	Used to parse any date time object that can be represented as a long.
<code>%d{ElapsedDecimal}</code>	Used to parse any date time object that can be represented as a decimal. Fractional parts will be assumed to be number of milliseconds.
<code>%d{TAI}</code>	Used to process longs which represent International Atomic Time.

When representing times numerically, LogViewPlus will attempt to convert the number into the correct time based on the size of the number. For example, Microsoft .Net ticks are counted as the number of 100-nanosecond intervals that have elapsed since January 1, 000. This number will be larger than the current time in milliseconds measured from Jan 1, 1970.

Metadata Date Patterns

Some log entries may be written in a format which has a timestamp but not a date. In these scenarios, the date is often extracted from the log file metadata.

Metadata Date	Date Used
<code>datetoday</code>	The current date. This setting does not use log file metadata.
<code>filedate-created</code>	The date the log file was created. This information may appear incorrect if the file has been copied or moved. This setting is not currently supported for remote log files.
<code>filedate-modified</code>	The date the log file was modified. This information may appear incorrect if the file has been copied or moved. This setting is not currently supported for remote log files.
<code>filedate-namescan</code>	LogViewPlus will try to parse the file name in order to extract the date. This process will use the same date parsing technology which is built into LogViewPlus. Unfortunately, a successful date parse cannot be guaranteed and there is currently no way to instruct LogViewPlus on the date format. Therefore, the date can only be extracted on a best-effort basis.

For example, if you wanted LogViewPlus to use the log file creation date as the date for every log entry in the log file, you could use the conversion specifier:

`%d{filedate-created %H:mm:ss}`

In the above example, we have simply used the metadata date pattern 'filedate-created' as the first part of our date time pattern.

%S and %s Specifiers

To demonstrate the flexibility of the %s specifiers, we are going to look at a pattern parser which can parse an IIS log file. Parsing an IIS log file is not difficult. We are using it as a simple example of a text file with structured data which is written in a way that differs from most application log files. Also, the columns used in an IIS log file differ from those typically used in an application log file - giving us a great opportunity to show off the power of the %s specifier.

One of the nice things about parsing IIS log files is that the log files often contain a comment which details the format of the log file. A sample IIS log file format is shown below. The IIS log file format separates all fields with a single space:

**#Fields: date time s-ip cs-method cs-uri-stem cs-uri-query s-port cs-username c-ip
cs(User-Agent) cs(Referer) sc-status sc-substatus sc-win32-status time-taken**

This format might translate into a log entry like:

**2014-05-16 13:28:28 192.168.1.1 GET /dir/file.html - 80 - 192.168.1.1 Mozilla - 200 0 0
39244**

Now, how do we convert the format given into a pattern that pattern parser can understand. For starters, let's note 2 things. First, the log entry starts with the date - just like most of our application log files. Second, the log entry can be considered complete when we reach a new line. Therefore, we can use two common specifiers at the start and end of our log format - %d and %n. Knowing this, we could be a bit lazy and parse this file with a simple pattern:

%d %m%n

But that's not very helpful. We know the structure of the data, how can we extract more useful information? The key to parsing unusual patterns is the %s specifier first discussed in [specifier basics](#). Using the %s specifier we can define columns like:

%s{S-IP}

The above specifier will retrieve the data from the "s-ip" field. Knowing this, parsing the rest of the log entry becomes trivial. The above IIS log entry can be parsed with the pattern:

**%d %s{S-IP} %s{Method} %s{URI} %s{URI-Query} %s{Port} %s{Username} %s{C-IP}
%s{User-Agent} %s{Referrer} %s{Status} %s{Substatus} %s{Win32-Status} %s{Time-
Taken}%n**

This pattern may look a bit confusing at first but notice that we are only using three conversion specifiers: %d, %s and %n. The %s specifier is doing most of the work, we simply need to give LogViewPlus column names for the data.

Using this pattern we can load the IIS log file into LogViewPlus:

Date	Time	S-IP	Method	URI	URI-Q...	Port	Userna...	C-IP
16 May 2014	13:28:28.000	192.1...	GET	/dir/Sa...	-	80	-	192.1...
16 May 2014	13:28:29.000	192.1...	POST	/dir/Sa...	-	80	-	192.1...
16 May 2014	13:28:32.000	192.1...	GET	/dir/_...	-	80	-	192.1...
16 May 2014	13:28:45.000	192.1...	GET	/dir/Sa...	-	80	-	192.1...
16 May 2014	13:28:45.000	192.1...	GET	/dir/Sc...	-	80	-	192.1...
16 May 2014	13:28:45.000	192.1...	GET	/dir/Sc...	-	80	-	192.1...
16 May 2014	13:28:46.000	192.1...	GET	/dir/_...	-	80	-	192.1...
16 May 2014	13:28:48.000	192.1...	POST	/dir/Sa...	assess...	80	-	192.1...
16 May 2014	13:29:24.000	192.1...	POST	/dir/Sa...	assess...	80	-	192.1...
16 May 2014	13:32:07.000	192.1...	GET	/dir/Sa...	-	80	-	192.1...
16 May 2014	13:32:07.000	192.1...	GET	/dir/Co...	-	80	-	192.1...

Once the log file is loaded into LogViewPlus all of the normal functionality such as text searching and data filtering will work as expected.

Advanced Specifiers

All log files can be parsed using the 7 conversion specifiers described in [Specifier Basics](#). However, LogViewPlus supports over thirty different conversion specifiers. That is because we wanted to give you the maximum amount of control over how your log file is parsed.

There are two advantages of using the advanced conversion specifiers:

1. Anything parsed and understood by LogViewPlus will have its own dedicated column in the [Log Entry Grid](#).

2. If you are creating your own [custom filters](#), you will have all parsed fields available to you in a field with a dedicated name. This will make your code easier to read and support. Unparsed or generic fields (like %s) are still available, but may be harder to access and interpret.

The full list of conversion specifiers and their corresponding grid column names are shown below. Some specifiers may have an alias.

All of the conversion specifiers listed below are considered "Reserved" when using the [Regex Parser](#). When used by the Regex Parser, conversion specifiers should not be prefixed with a percent sign.

Specifier	Grid Column	Notes
%a %appdomain	AppDomain	
%aspnet-cache	Web Cache	
%aspnet-context	Web Context	
%aspnet-request	Web Request	
%aspnet-session	Web Session	
%C %class %type	Class	
%d %date	Date + Time	This field is split across two grid columns. This makes it easy to remove the date column if it is not adding value.
%ex	Exception	

%exception %throwable %rex %rexception %rthrowable %xex %xexception %xthrowable %stacktrace %stacktracedetail		
%F %file	File Name	
%highlight	N/A	This field does not affect the way a log entry is parsed. It has been included for compatibility with Apache Log4 conversion patterns.
%l %location	Location	
%L %line	Line Number	
%c %logger	Logger	
%m %msg %message	Message	
%M %method	Method Name	
%x %ndc	NDC	
%n %newline	N/A	Marker for the end of a log entry.
%%	N/A	
%p %level	Priority	
%t	Thread	

%thread		
%r %timestamp %relative	See 'Date'.	
%w %username	Username	
%utcddate	See 'Date'.	
%marker	N/A	
%replace	N/A	This field does not affect the way a log entry is read. It has been included for compatibility with Apache Log4 conversion patterns.
%sn %sequencenumber	Sequence Number	
%style	N/A	This field does not affect the way a log entry is read. It has been included for compatibility with Apache Log4 conversion patterns.
%u %identity	Identity	
%uuid	UUID	
%s	N/A	For parsing a single word. Items added as strings will not be available in the LogViewPlus grid by default. To add them to the grid, specify a column name with an argument like %s{Column_Name}. If you were to build a custom filter, you would find string data available under the LogEntry.Strings property.
%S	N/A	For parsing multiple words. See above.
%P %K %X %key %map %mdc %property %properties	N/A	These properties will not be available in the LogViewPlus grid, but will be available in the LogEntry. If you were to build a custom filter, you would find property data available under the LogEntry.Properties property.

Log File	<p>The log file where the log entry was parsed. Useful when working with merged views.</p> <p>This grid column is available on all log files. To view it, simply select the column from the 'Add Columns' command.</p>
Log Line Number	<p>The line number in the log file where this entry was found. Useful if you need to refer to the underlying text file.</p> <p>This grid column is available on all log files. To view it, simply select the column from the 'Add Columns' command.</p>

Important: Because LogViewPlus does not write a log file, it has no way to confirm that the field displayed actually matches the relevant data. For example, the %appdomain specifier simply specifies a string which will be available through the AppDomain column in the grid. LogViewPlus has no way of knowing whether this field is actually a .Net Application domain. This relationship is assumed.

Format Modifiers

Format modifiers allow you to specify the exact number of characters used to represent a field. Format modifiers come after the percent sign and before the conversion specifier.

The following table lists format modifiers and their affects:

Format Modifier	Justified	Min Width	Max Width	Comments
%20	Right	20	None	Field is left padded with spaces to be a minimum of 20 characters long.
%-10	Left	10	None	Field is right padded with spaces to be a minimum of 10 characters long.
%.15	NA	None	15	Field is truncated if it is longer than 15 characters.
%-20.30	Left	20	30	Combines both justification and truncation. In this case, right padding up to 20 spaces and truncating if necessary after 30.
%5.5	Right	5	5	In this example, we are specifying a field with a fixed width of 5. Fixed width fields can be very useful when continuous string data.

As an example, consider the following log entry.

2012-05-09 10:50:14 A Test14Start - Initializing...

There are two things to note about this log entry. First, there are 5 spaces after the 'A' that follows the date. Second, the string 'Test14Start' actually contains three pieces of information an ID, a Position, and an Action.

The above log entry can be parsed with the conversion pattern:

%d %-5c %4.4s{ID}%2.2s{POS}%5.5s{ACTION} - %m%n

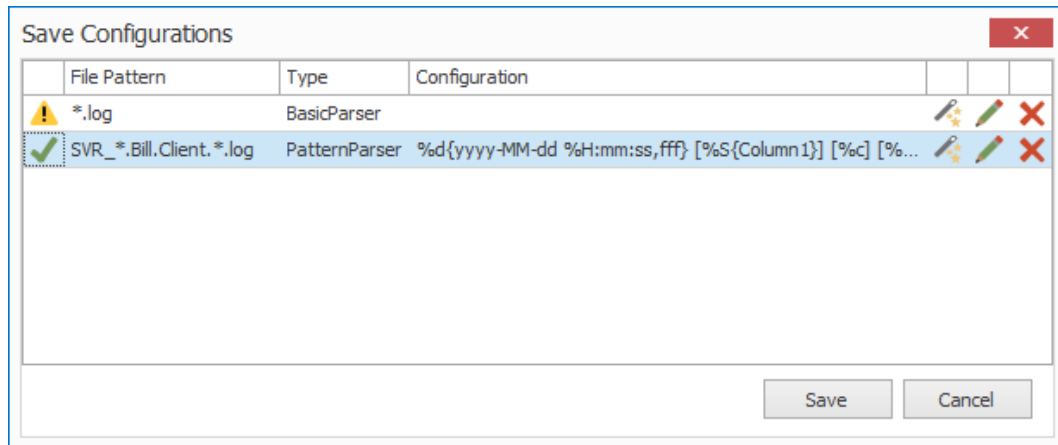
Using this conversion pattern, the log entry will be parsed as:

Format Modifier	Value	Comments
-----------------	-------	----------

%d	2012-05-09 10:50:14	The date.
%-5c	A	The extra 5 spaces are ignored. If our conversion pattern had contained two spaces between this specifier and the next, we would have had a parsing error.
%4.4s{ID}	Test	The next field is a fixed 11 characters. The first four will be the ID.
%2.2s{POS}	14	...the next two will be the Position.
%5.5s{ACTION}	Start	...and the final 5 will be the Action. Taken together, these three fields must always be exactly 11 characters long.
%m	Initializing...	The message.

You can find out more about format modifiers online. The [Apache documentation](#) is a particularly good place to start.

Parser Wizard



The first time you open a log file in LogViewPlus you may be presented with the dialog above explaining how the file was parsed. Automatically generated parser configurations are designed to be functional and may provide you with all the features you need for your current task - with zero configuration.

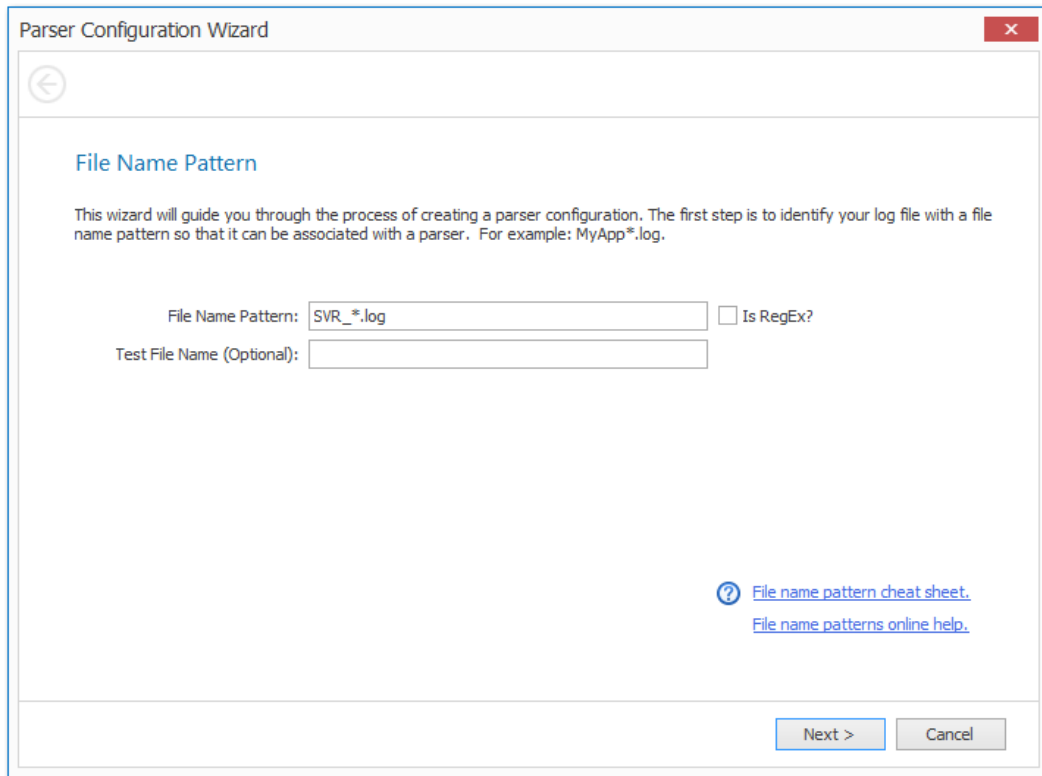
However, you may get a better experience if you setup a parser configuration manually. This is especially true for log files you open frequently.

The syntax used to configure [Log Parsers](#) manually was covered in a previous section.

For users who are new to LogViewPlus, we recommend using the Parser Wizard. The Parser Wizard can be opened by double clicking one of the provided parser descriptions. Executing the Parser Wizard in this way has the advantage of populating fields in advance which can provide a head start for new users.

The Parser Wizard can also be opened from the [Application Settings](#).

Log File Identification



The screenshot shows a 'Parser Configuration Wizard' dialog box. It has a title bar with a close button (X) in the top right corner. Inside the dialog, there is a back arrow icon in the top left. The main content area is titled 'File Name Pattern' in blue. Below the title, there is a paragraph of text: 'This wizard will guide you through the process of creating a parser configuration. The first step is to identify your log file with a file name pattern so that it can be associated with a parser. For example: MyApp*.log.' Below this text, there are two input fields. The first is labeled 'File Name Pattern:' and contains the text 'SVR_*.log'. To its right is a checkbox labeled 'Is RegEx?' which is currently unchecked. Below the first input field is a second input field labeled 'Test File Name (Optional):' which is empty. At the bottom right of the dialog, there are two buttons: 'Next >' and 'Cancel'. Above the 'Next >' button, there are two links: a question mark icon followed by 'File name pattern cheat sheet.' and 'File name patterns online help.'

When configuring a log parser via the Parser Configuration Wizard, the first thing we need to do is set our filename pattern. The filename pattern will be used to identify a log file by name and pairing it with the parser we are about to configure.

For example, say your application writes log entries to a file named SVR_15.6.2014_1.log.

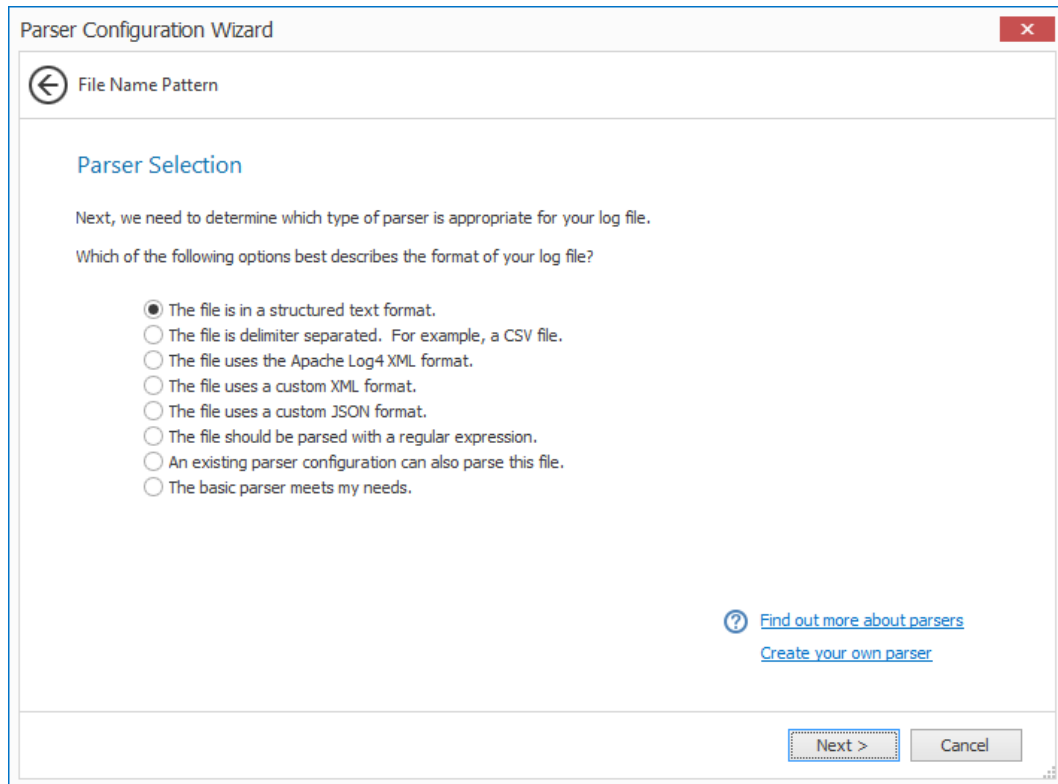
You can associate a parser to all your application log files with the pattern SVR_*.log. In this example, the wildcard '*' is being used to match multiple unspecified characters. For more advanced patterns consider using a regular expression by checking the 'Is Regex' checkbox. This will allow the filename pattern text box to contain a full regular expression.

If you want to use the same parser with multiple log files, you can use the pipe character - | - to separate file names. For example, **SVR_*.log|*MyOtherApp***.

You can optionally test the filename pattern you have provided by entering a sample file name into the 'Test File Name' text box. If you choose to test your filename pattern, a

message will appear stating whether or not the test file name is matched by the provided filename pattern.

Parser Selection



The screenshot shows a window titled "Parser Configuration Wizard" with a red close button in the top right corner. The window has a navigation bar at the top with a back arrow and the text "File Name Pattern". The main content area is titled "Parser Selection" and contains the following text: "Next, we need to determine which type of parser is appropriate for your log file." and "Which of the following options best describes the format of your log file?". Below this text is a list of seven radio button options. The first option, "The file is in a structured text format.", is selected. The other options are: "The file is delimiter separated. For example, a CSV file.", "The file uses the Apache Log4 XML format.", "The file uses a custom XML format.", "The file uses a custom JSON format.", "The file should be parsed with a regular expression.", "An existing parser configuration can also parse this file.", and "The basic parser meets my needs.". At the bottom right of the main content area, there are two links: "Find out more about parsers" and "Create your own parser". At the bottom of the window, there are two buttons: "Next >" and "Cancel".

Parser Configuration Wizard

File Name Pattern

Parser Selection

Next, we need to determine which type of parser is appropriate for your log file.

Which of the following options best describes the format of your log file?

- ☒ The file is in a structured text format.
- ☐ The file is delimiter separated. For example, a CSV file.
- ☐ The file uses the Apache Log4 XML format.
- ☐ The file uses a custom XML format.
- ☐ The file uses a custom JSON format.
- ☐ The file should be parsed with a regular expression.
- ☐ An existing parser configuration can also parse this file.
- ☐ The basic parser meets my needs.

[Find out more about parsers](#)
[Create your own parser](#)

Next > Cancel


The next step in the Parser Configuration Wizard, is to determine which parser best matches your log file data. All that you need to do here is answer the given question to the best of your ability. The Parser Wizard currently supports seven different parser types and each of the options provided correspondence to one of the seven types: [Pattern Parser](#), [DSV Parser](#), [Apache XML](#) (a zero configuration variant of the XML parser), [Xml Parser](#), [JSON Parser](#), [Regular Expression](#) and the [Basic Parser](#).

Select 'An existing parser configuration can also parse this file' when the format of the target file matches the format of an already configured parser. LogViewPlus will then allow you to select the existing parser to combine filename patterns. Please see [Combine File Patterns](#) for more information.

The final option, "The basic parser meets my needs", is a special case available only when launching the Parser Wizard from the [Automatic Configuration](#) dialog. The purpose of

this option is to allow you to explicitly use the Basic Parser and suppress the Automatic Configuration dialog in the future.

Sample Log Entry

 Parser Selection

Sample Log Entry

Please enter a sample log entry below. It may help to [open your log file in a text editor](#).

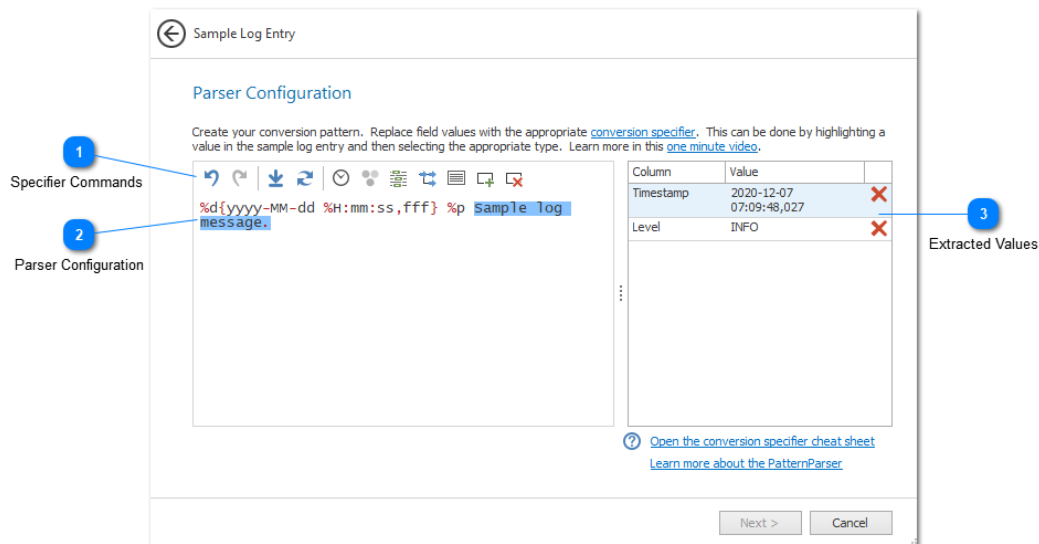
2016-12-07 07:09:48,027 INFO My example log entry.

Next >

Cancel

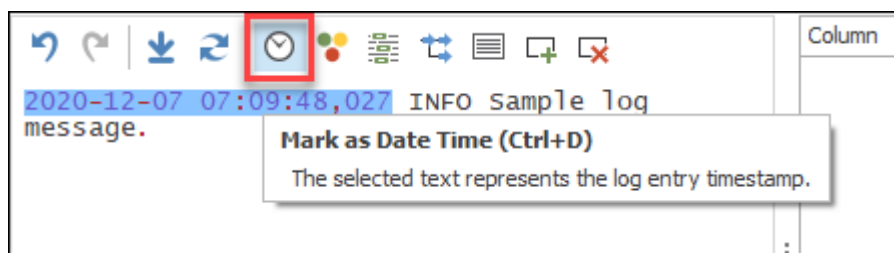
Depending on the type of parser needed, we will probably need to provide a sample log entry. The parser wizard can help you configure a parser, but it will need a sample log entry to display extracted values and verify your parser configuration is correct.

Parser Configuration



The parser configuration screen is designed to help you quickly create a pattern capable of parsing your log files.

To create a parser configuration, simply highlight a field value and then select the appropriate specifier command. For best results, we recommend moving from left to right across your sample log entry.



Once a value has been extracted, it will be replaced with the appropriate conversion specifier. The extracted value will be displayed in the Extracted Values grid along with the column name.

If a conversion specifier has already been used it may be disabled on the toolbar. Some conversion specifiers can only be used once. In the above example you can see that the

priority column has already been added and therefore this command is disabled in the context menu.

When adding a custom column through the text selection method, you will need to provide a column name. The custom column command is simply a wrapper around the [%S conversion specifier](#).

Finally, when configuring XML or JSON parsers, it is important to note that unused nodes can be removed. These parsers are only concerned with nodes that contain conversion specifiers or have children which contain conversion specifiers.

1 Specifier Commands



The specifier commands toolbar allows you to extract values from a log entry and replace them with a [Conversion Specifier](#). All toolbar commands can also be accessed on the [Parser Configuration Context Menu](#).

These commands are described in detail in the next section. You can also hover over a command to see a tool tip with a full description.

2 Parser Configuration

`%d{yyyy-MM-dd %H:mm:ss,fff} %p` Sample log message.

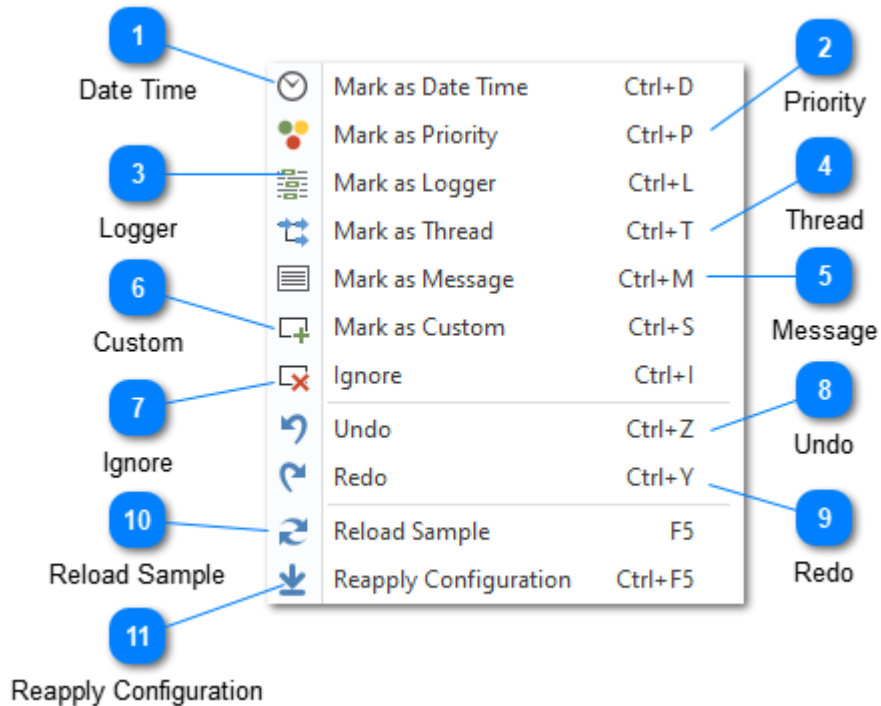
The parser configuration view shows a work in progress version of your parser configuration. If you are familiar with [Conversion Specifiers](#), you can type your parser configuration directly. However, we recommend selecting text and marking it with the specifier command toolbar. For best results, this should be done while progressing from left to right.

3 Extracted Values

Column	Value	
Timestamp	2020-12-07 07:09:48,027	✗
Level	INFO	✗

The extracted values grid displays the values that would be extracted from your sample log entry given the parser configuration provided. The column names for the values will also be displayed. Removing a value from the extracted values list will add it back to the parser configuration.


Parser Configuration Menu



The parser configuration context menu is available by right-clicking in the [Parser Configuration](#) area. It allows you to extract selected text and replace it with a [Conversion Specifier](#).

Most of the commands in the configuration menu are available only when text is selected. Some commands can only be used once.

1 Date Time

 Mark as Date Time Ctrl+D

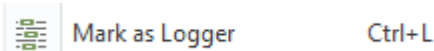
Extracts the selected text and replaces it with the date time specifier (%d). LogViewPlus will also analyze the text contained in the selection and attempt to resolve the date pattern. Having a defined date pattern will result in a faster, more accurate parse.

2 Priority



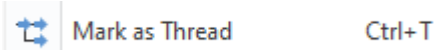
Extracts the selected text and replaces it with the priority specifier (%p). If an excess space is detected, LogViewPlus may decide to use a fixed width priority. For example, if the priority is always 5 characters, LogViewPlus may use the specifier %-5p.

3 Logger



Extracts the selected text and replaces it with the logger specifier (%c).

4 Thread



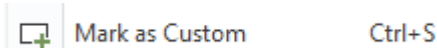
Extracts the selected text and replaces it with the thread specifier (%t).

5 Message



Extracts the selected text and replaces it with the message specifier (%m).

6 Custom



Extracts the selected text and replaces it with the [string specifier](#). Either %s or %S may be used depending on if the selected text contains spaces or otherwise has a well-defined start and end marker.

When marking the selected text as a string, you will be prompted to enter a column name. If you do not enter a column name, the field will still be parsed as a string, but the string will not be visible in the [Log Entry Grid](#).

7 Ignore



Ignore

Ctrl+I

Extracts the selected text and replaces it with the [string specifier](#). Either %s or %S may be used depending on if the selected text contains spaces or otherwise has a well-defined start and end marker. You will not be prompted to provide a column name and therefore the value will be ignored by the [Log Entry Grid](#).

8 Undo



Undo

Ctrl+Z

Reverts the previously added or removed conversion specifier.

9 Redo



Redo

Ctrl+Y

Re-applies the previously added or removed conversion specifier.

10 Reload Sample



Reload Sample

F5

Removes all parser configuration and reloads the sample log entry. This is useful when you want to start over with a new parser configuration.

11 Reapply Configuration



Reapply Configuration

Ctrl+F5

Replies the known parser configuration. This is useful when you want to start over with the existing parser configuration.

Resolve Log Entry Date

Parser Configuration Wizard

Pattern Parser Configuration

Resolve Log Entry Date

The pattern that you are using to parse log entries does not define a date. How should log entry dates be determined?

☒ Do not explicitly set a date.

☐ Use today's date.

☐ Use the log file creation date.

☐ Use the log file modified date.

☐ Extract the date from the name of the log file.

Your conversion pattern will be:

```
%d{%H:mm:ss} %t %c: %m%n
```

Next > Cancel

Some log entries may be written in a format which has a timestamp but not a date. In these scenarios, the date is often extracted from the log file metadata. The Resolve Log Entry Date configuration page provides you with a series of options that allow you to extract a date from the log file metadata. Note that this configuration page will only be shown in the event that your date specifier does not already define a date pattern.

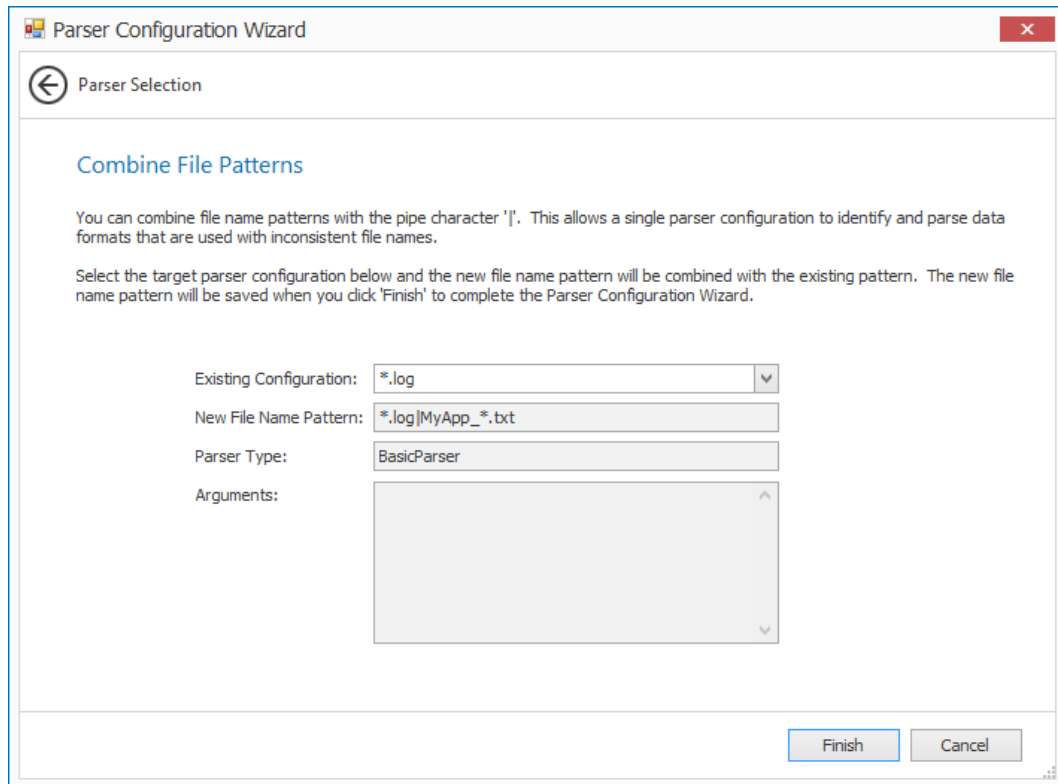
LogViewPlus supports the following metadata date extraction options:

Metadata Date	Date Used
datetoday	The current date. This setting does not use log file metadata.
filedate-created	The date the log file was created. Note that this information may not be correct if the file has been copied or moved. This setting is not currently supported for remote log files.

filedate-modified	The date the log file was modified. Note that this information may not be correct if the file has been copied or moved. This setting is not currently supported for remote log files.
filedate-namescan	LogViewPlus will try to parse the file name in order to extract the date. This process will use the same date parsing technology which is built into LogViewPlus. Unfortunately, a successful date parse cannot be guaranteed and there is currently no way to instruct LogViewPlus on the date format. Therefore, the date can only be extracted on a best-effort basis.

For more information, please see the [Date Specifier](#) documentation.

Combine File Patterns



The screenshot shows a window titled "Parser Configuration Wizard" with a red close button in the top right corner. The window has a header bar with a back arrow icon and the text "Parser Selection". Below the header, the title "Combine File Patterns" is displayed in blue. The main text area contains the following instructions: "You can combine file name patterns with the pipe character '|'. This allows a single parser configuration to identify and parse data formats that are used with inconsistent file names." and "Select the target parser configuration below and the new file name pattern will be combined with the existing pattern. The new file name pattern will be saved when you click 'Finish' to complete the Parser Configuration Wizard." Below the text, there are four input fields: "Existing Configuration:" with a dropdown menu showing "*.log"; "New File Name Pattern:" with a text box containing "*.log|MyApp_*.txt"; "Parser Type:" with a dropdown menu showing "BasicParser"; and "Arguments:" with a large empty text area. At the bottom right of the window, there are two buttons: "Finish" and "Cancel".

Parser Configuration Wizard

Parser Selection

Combine File Patterns

You can combine file name patterns with the pipe character '|'. This allows a single parser configuration to identify and parse data formats that are used with inconsistent file names.

Select the target parser configuration below and the new file name pattern will be combined with the existing pattern. The new file name pattern will be saved when you click 'Finish' to complete the Parser Configuration Wizard.

Existing Configuration: *.log

New File Name Pattern: *.log|MyApp_*.txt


Parser Type: BasicParser

Arguments:

Finish Cancel

If you want to use the same parser with multiple log files, you can use the pipe character - | - to separate file names. For example, ***.log|MyApp*.txt**. The Combine File Patterns page automates merging of file name patterns. Simply select the existing parser configuration which is capable of parsing the target log file. The filename pattern will be combined with the pattern you provided in the [Log File Identification](#) step.

Parser Testing

 Pattern Parser Configuration

Testing Your Parser

You can test your parser configuration by pasting a few sample log entries into the box provided. When you click the 'Test' button a message will appear below telling you how many records were parsed.

Testing your parser configuration is optional.

um.cs] [Calculate()] [192] [0] DEBUG Client - Application initializing.

um.cs] [Calculate()] [192] [1] DEBUG Client - Application startup successful.

<

>

✓ 2 log entries parsed successfully.

Test

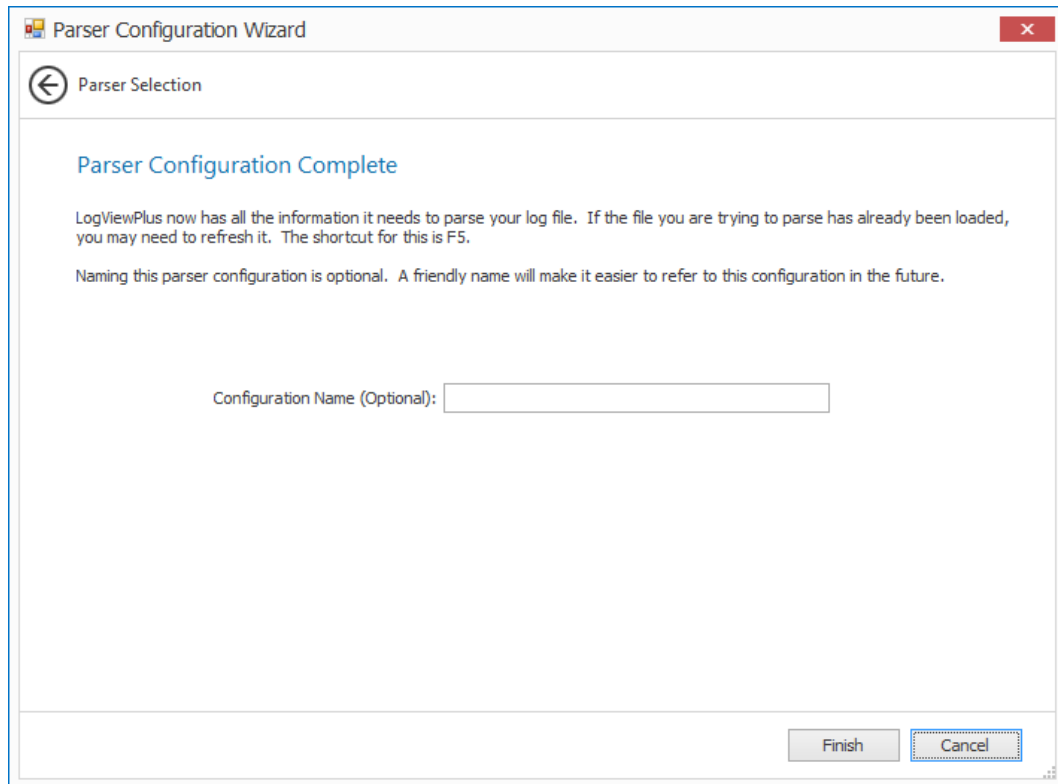
Next >

Cancel

The parser testing stage is used to confirm that you have configured your parser correctly. Simply place any number of log entries into the text area provided. Clicking the test button will then display a dialog showing how many log entries were parsed with the configuration provided in the previous screens. In the above example, a message is displayed showing that "2 log entries were parsed successfully". If an error had occurred while parsing the log entries, an error message will be displayed instead. If an incorrect number of log entries is displayed, then parser has not been configured correctly. In this case please revisit your parser configuration.

Note that parser testing is recommended but not required.

Configuration Complete



The final screen in the parser configuration wizard confirms that everything has been set up correctly. Click Finish to save your parser configuration. If you launched the parser configuration wizard from the application settings dialog you may also need to click OK again to apply your changes.

Providing a configuration name is optional. A configuration name will make it easier to find this configuration in the future.

Once you have saved your new parser configuration you may need to refresh any log files which have already been opened. To do this select the target log file and press F5.

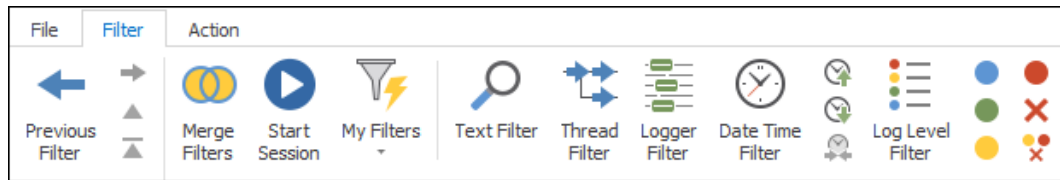
Data Stores

Log entries are not always stored in files. For example, Windows stores log entries in the Windows Event Log. Other examples of data store formats include relational databases, Syslog or UDP messages.

LogViewPlus can handle non-file based log entry stores as data stores - a new concept introduced in LogViewPlus 2.4. Data stores can be configured to appear as a file in the [LogViewPlus File Browser](#).

For more information, please see the [data stores configuration](#) topic.

Analyzing Log Files



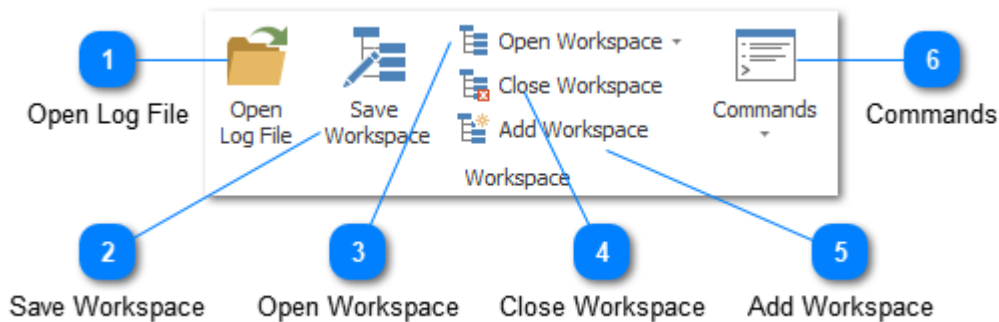
Almost all actions available in LogViewPlus can be accessed via one of the toolbars. LogViewPlus toolbar commands are covered in the following sections.

Toolbar Commands

LogViewPlus uses toolbar commands to expose features and actions. Almost all of the features available in LogViewPlus or accessible via the toolbars. Note that LogViewPlus also relies heavily on context menus. When learning LogViewPlus is useful to right-click on items you want to work with.

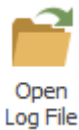
The following sections will cover the LogViewPlus toolbar commands in detail.

Workspace



The workspaces toolbar contains a number of commands to help you manage your workspace. These include commands like opening a log file, saving a workspace, managing directory monitors, and saving log entries.

1 Open Log File



The open log file command opens the LogViewPlus [file browser](#) which can be used to open a new file. The default directory used when opening a new file will be based on the previous file you have opened (if available). Note that once a file is opened it will automatically be tracked by default.

2 Save Workspace



Saves the current workspace. If you have not already saved the current workspace, you will need to provide a workspace name with the [Add Workspace](#) dialog.

Workspaces are useful when you find that you frequently open a group of files together. Note that a workspace includes files which you have opened manually, directory monitors, and any filters that you may have applied.

Files that have been opened based on rules are not part of the workspace. For example, Directory Monitors are part of the workspace while files opened by the Directory Monitor are outside of the workspace.

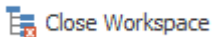
3 Open Workspace



The open workspace command allows you to open a previously saved workspace.

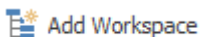
Note that files opened based on rules are not included in the workspace. The reason for this is that if you load this workspace in the future the files should be opened based on the application of the rules and not simply because they were opened before.

4 Close Workspace



Closes all files, filters, and directory monitors. All opened files and directory monitors are part of the 'current' workspace even if the workspace has not been saved.

5 Add Workspace



Adding a workspace opens the [Add Workspace](#) dialog which will allow you to save the current workspace under a new name.

6 Commands

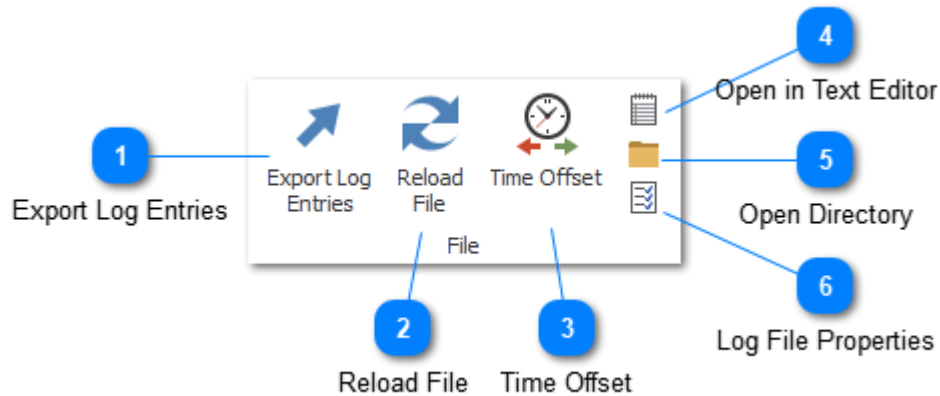


Commands



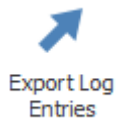
This optional toolbar command will only be visible if you have configured an [external command](#). Pre-configured external commands can be executed from this menu.

File



The file toolbar is used for managing your log files and views.

1 Export Log Entries



The save log entries command allows you to save the current view or filter as a new log file. This is helpful if you want to share your current view with somebody else who may not have LogViewPlus installed. It is also useful when you want to save the current view for later analysis.

The exported log entries can be saved in a number of different formats including CSV, HTML, or in the format in which the log entry was written. However, the recommended format is as a custom LogViewPlus CSV file with the extension *.lvp.

This format is basically CSV with a few added columns to help LogViewPlus keep track of things like bookmarks. LogViewPlus has a built-in parser for understanding files with a *.lvp extension.

2 Reload File



Reloading the log file is useful if you have previously cleared all log entries and now need to see the cleared entries. It can also be helpful when you are modifying parser configurations, or to refresh a file which is not currently tailed.

3

Time Offset



Creates a time offset for the currently selected log file. This will modify all timestamps in the currently selected log file by a user configured amount. This command is useful when you intend to merge multiple log files but the timestamps of the log files are not in sync. Please see the [time offset](#) documentation for more details.

4

Open in Text Editor



Opens the currently selected log file in your [default text editor](#).

5

Open Directory



Opens the directory which contains the currently selected log file in the [File Explorer](#).

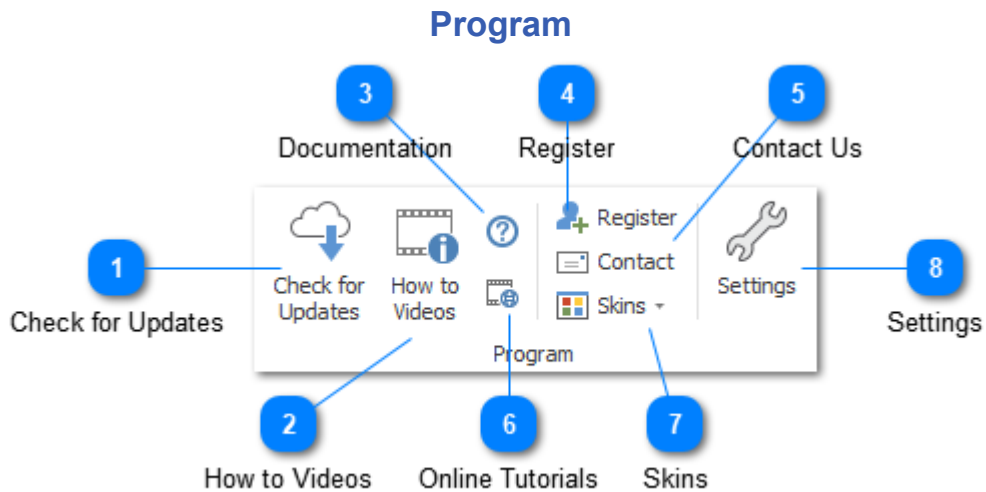
You can also hold down the CTRL key when executing this command to open the target directory in Windows Explorer. If this option is used and the log file is a remote log file (accessed via SFTP or FTP) then LogViewPlus will open the temporary directory which contains the local version of the log file.



Log File Properties



Opens the [Log File Properties](#) window for the currently selected log file.



The Program toolbar is used to manage actions related to configuring and getting help for LogViewPlus.

1 Check for Updates



Checks for updates to your currently installed LogViewPlus version. This command requires an Internet connection. In the event of connection difficulty, for example, if LogViewPlus is unable to authenticate through your proxy server, you will need to download updates from the website.

2 How to Videos



The 'How to Videos' command can be used to open a window which contains a series of links pointing to the [Quick Video Examples](#) which we have included in

the documentation. These videos are a great way to quickly get up to speed with LogViewPlus.

3

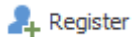
Documentation



The documentation command takes you to the [online documentation](#).

4

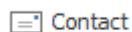
Register



The register button begins the LogViewPlus [registration process](#). Note that once this process is completed the register command will be removed.

5

Contact Us



If you have an e-mail client installed locally on your machine. This command will open an e-mail with an e-mail address and subject ready to be sent to LogViewPlus support. This command uses the "mailto:" action.

Also, note that you can always contact us by using our online [contact form](#). Please feel free to contact us if you have any questions or feedback on LogViewPlus.

6

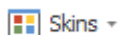
Online Tutorials



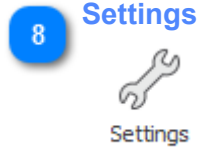
The 'online tutorials' command takes you to the videos section of the LogViewPlus website which contains a series of detailed [video tutorials](#).

7

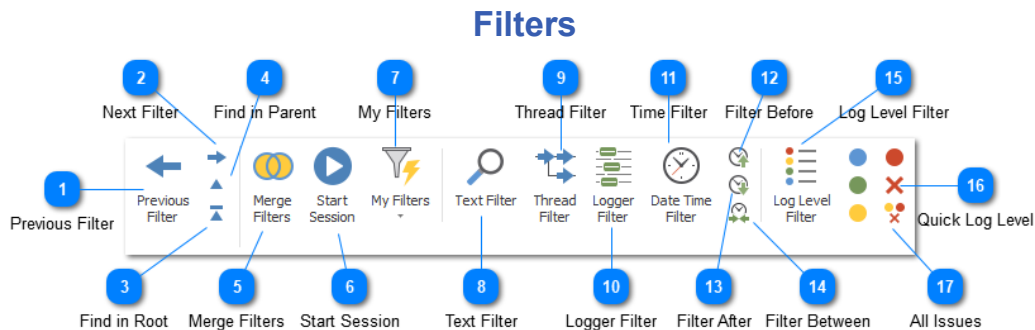
Skins



Skins command can be used to change the look and feel of LogViewPlus.



The Settings command opens the [Application Settings](#).

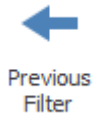


The filters toolbar is used to create new filters on the current log file or view. Filters are the primary tool used to analyze your log entries. You can create complex search criteria by either chaining filters or merging them into a single view.

When creating a new filter, it will always be created as a child of the current selection (either a Log File or a Filter). Children always contain a subset of the log entries available in the parent.

This toolbar also contains commands for navigating between filters.

1 Previous Filter



Move "back" to the previously selected filter.

2 Next Filter



Move "forward" to the previously selected filter. Note that this option will only be available if the "Previous" command has recently been used.

3

Find in Root



Finds the currently selected log entry in the original log file. This is the default action when double-clicking a log entry. Using this action as well as the "Previous" action is a great way to navigate your log files. You can quickly filter all errors, double-click an error to see the surrounding circumstances and then use the "Previous" command to move back to see your list of errors.

4

Find in Parent



Moves up the log filter hierarchy. One interesting aspect of the log filter hierarchy is that it is not possible to have a log entry available in a child filter which is not also available in the parent filter. This command will find the currently selected log entry in the parent filter.

5

Merge Filters



Merge
Filters

Displays the [Merge Filter Dialog](#) which can be used to create a new Merge Filter. Merge Filters can also be created by dragging and dropping filters, but for more complex scenarios we recommend using the Merge Filter Dialog.

6

Start Session



Start
Session

Starts or stops a logging session. Logging sessions are useful when you want to focus on log entries from the period of time starting from 'now' and ending at some point in the future. For example, if you were debugging a program, you might start

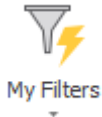
a logging session take a user action through your system and then stop a logging session. This will create a new date filter between the given start and stop times.

Logging sessions work with knowledge of the last recorded log entry. A session starts just after the last recorded entry and ends with the last recorded entry. This allows LogViewPlus to track all of the new log entries which were written over the time period regardless of time zone. Note that this may lead to confusing timespans being reported by the resulting date filter. For example, consider the case where you are tracking a log file in a different time zone where only one log entry is written during the session. You decide to track the file for ten minutes. In this case, the session filter start time will occur one millisecond after the last written log entry - which may have occurred days ago. The end time will be the timestamp associated with our single new log entry. It is therefore possible to record a 10 minute session which creates a date filter lasting for only a second. Alternatively, the timespan may cover several days. The important aspect is not the length of time elapsed, but that the filter captures all log entries written during the recording session.

Recording from the last recorded log entry allows LogViewPlus to track all new log entries written across multiple time zones when working with merged files.

A root log file can have only one active session at a time.

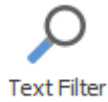
7 My Filters



The My Filter command can be used to access any custom filters you have written. This will command will only be available if you have successfully loaded a plug-in which contains a custom filter.

For more information on creating your own custom filters please see [Custom Filters](#).

8 Text Filter



Text Filter

Creates a new text filter. A text filter is a filter on the current view which shows all logs which contain the given text. You can find out more about creating [Text Filters](#).

9

Thread Filter



Thread
Filter

Creates a new thread filter. A thread filter is a filter on the current view which shows all logs written by a particular thread.

10

Logger Filter



Logger
Filter

Creates a new logger filter. A logger filter is a filter on the current view which shows all logs written by a particular logger.

11

Time Filter



Date Time
Filter

Creates a new date time filter. A date time filter is a filter on the current view which shows all logs written in a specified date range.

12

Filter Before



Reads the time of the currently selected log entry and creates a new Date Filter showing everything which occurred before that time.

13

Filter After



Reads the time of the currently selected log entry and creates a new Date Filter showing everything which occurred after that time.

14

Filter Between



When two or more log entries are selected this command will determine the start and end time of the log entries and create a new Date Filter showing everything which occurred between the selected times.

15

Log Level Filter



Log Level
Filter

Creates a new log level filter. A log level filter is a filter on the current view which shows all logs with the matching log level. You can find out more about creating [Log Level Filters](#).

16

Quick Log Level



Quick log level filters can be used to quickly filter by log entry priority. Filter options include Debug, Info, Warn, Error, Fatal, and All Issues.

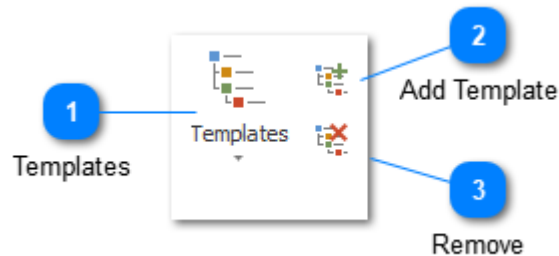
17

All Issues

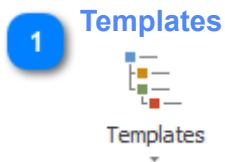


Worth of special mention within the quick log level filters is the 'All Issues' filter. This filter is a shortcut to help you quickly create a log level filter which will track all Warning, Error and Fatal log entries. This action is the same as clicking these three quick filter commands separately.

Templating



The templating toolbar allows you to add, remove, and apply templates. If you find yourself frequently applying the same filters or highlights consider creating a template instead.



The templates command opens a drop-down box with all of your available templates. Selecting a template will apply it to the current view. The command is enabled when one or more templates have been previously saved.

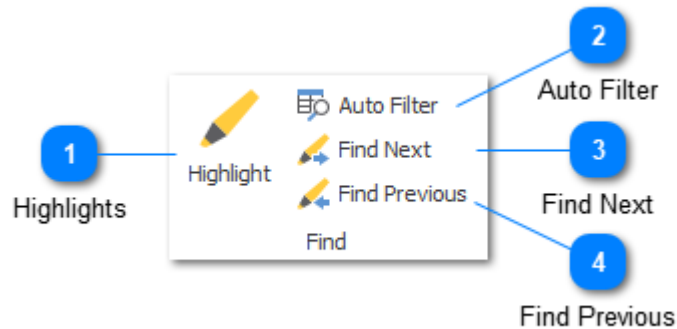


Adds the selected highlight, filter or group of filters as a new template. You will be given the option to create a template name. This command will be enabled when the current log file has a template applied.



Opens the remove template dialog box. This gives you the ability to delete a previously saved template. The command is enabled when one or more templates have been previously saved.

Find



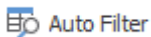
The find toolbar can be used to help you search the current view.

1 Highlights



Highlighting allows you to search for text and mark it to make it easier to find. Clicking on the Highlight command will bring up the [Highlight Manager](#) which allows you to create and delete highlights. Highlights are applied globally across all log files.

2 Auto Filter



Shows the grid search box below the column headings of the current log entry grid. This allows you to search the current log entry grid by column.

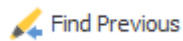
3 Find Next



Finds the next highlight. Note that any highlight will be matched. If you would like to find the next selected highlight, you can hold down the shift key while executing this command.

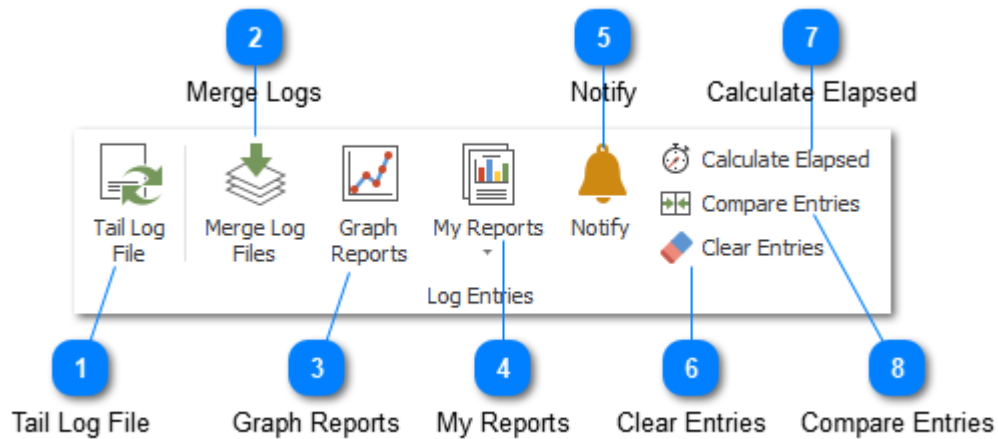


Find Previous



Finds the previous highlight. Note that any highlight will be matched. If you would like to find the previous selected highlight, you can hold down the shift key while executing this command.

Log Entries



The actions toolbar contains the list of actions you can use for the currently selected view.

1 Tail Log File

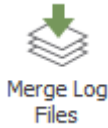


The tail log file command allows you to start or stop monitoring the current file for changes. Monitoring the file is the same as the UNIX tail command, which means new entries which are written to the file will be displayed in LogViewPlus as soon as they are written.

Auto-scroll is used to determine whether LogViewPlus should automatically scroll to new log entries as they are written. Note that auto-scrolling is only possible when a log file is in tail mode.

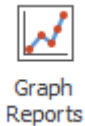
To enable auto-scroll, you simply need to select the last log entry currently in the view. An easy way to do this is to select the auto-scroll command from the [quick access toolbar](#) located in the top left corner of LogViewPlus. Alternatively, you can select the bottom of the [Navigation Bar](#).

2 Merge Logs



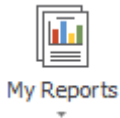
The merge logs command is used to merge two or more log files into the same view. This makes it easier to work with multiple log files which are related to the same application. Executing this command will open the [merge file selection](#) window.

3 Graph Reports



Opens a [graph based report](#) for the currently selected view.

4 My Reports



If you have configured LogViewPlus to generate custom reports, those reports will be available for execution. This command will only be visible if [custom reports](#) have been configured.

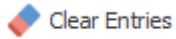
5 Notify



The notify command is used to [create an alert](#) when a new log entry is detected in a filter. This allows you to passively monitor a filter. For example, you could filter a log file for errors and be notified when a new error is detected.

6

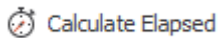
Clear Entries



The clear entries command removes all entries from the log entries grid. This is helpful if you want a fresh view on the current log file. For example, the application has recently been restarted and you are not concerned with previous log entries.

7

Calculate Elapsed



Calculates the amount of time that has elapsed between the two selected log entries.

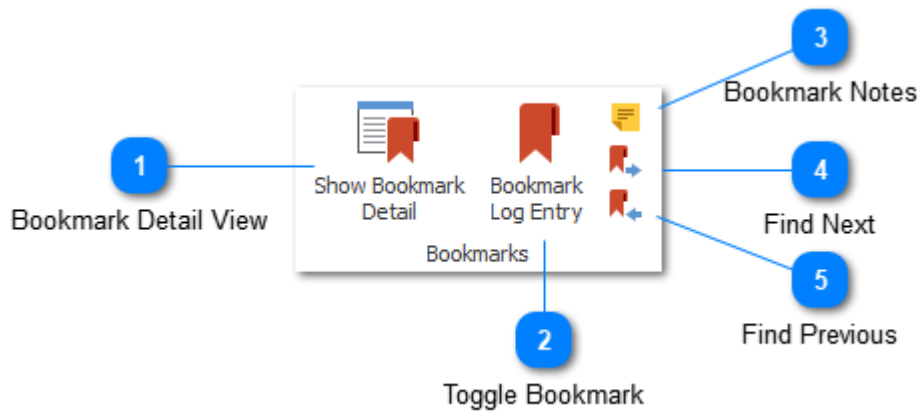
8

Compare Entries



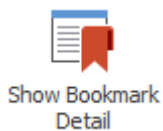
If you have [configured](#) LogViewPlus to use a compare tool (like WinMerge) the Compare Log Entries command will save the two selected log entries to separate files and open the files using your configured tool. This is helpful for quickly determining the difference between two log entries.

Bookmarks



Bookmarks can be used to quickly find a given log entry. This is useful when you are frequently moving around the log file but want to return to a known log entry. The bookmarks toolbar can be used for managing [bookmarks](#). Bookmarks also have a visual component on the log entry grid.

1 Bookmark Detail View



Opens the [Bookmark Detail View](#) which is a window that makes it easy to manage bookmarks and notes. This view can also be docked to the LogViewPlus main window.

2 Toggle Bookmark



Adds or removes a bookmark to the currently selected log entry. Note you can also add or remove bookmarks by double-clicking in the row selection column.

3

Bookmark Notes



Bookmark notes allow you to associate a given log entry with a block of free-form text. Depending on the state of the current log entry, the bookmark notes command will either convert the current bookmark, create a new bookmark, or display the already set bookmark notes.

4

Find Next



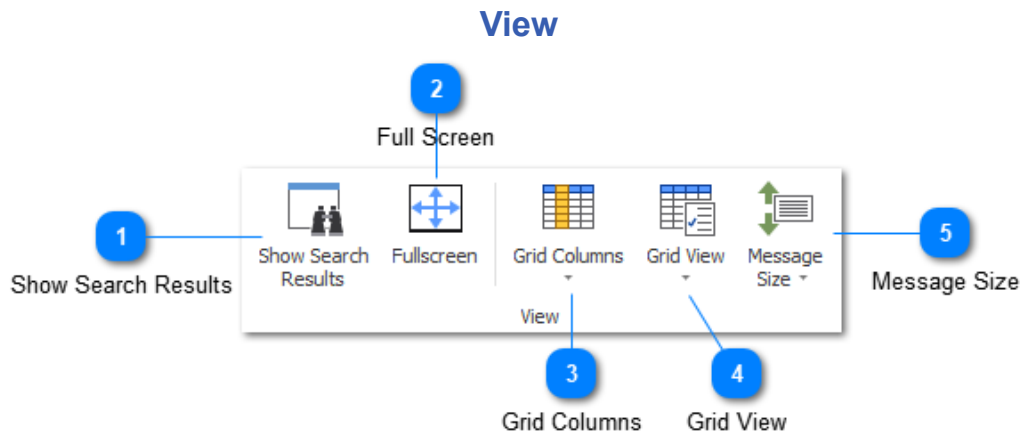
Finds the next bookmark. Note that any bookmark will be matched. If you would like to find the next selected bookmark, you can hold down the shift key while executing this command.

5

Find Previous



Finds the previous bookmark. Note that any bookmark will be matched. If you would like to find the previous selected bookmark, you can hold down the shift key while executing this command.



View commands can be used to control the display of the log entry grid.

1 Show Search Results



The show search results command can be used to display the search window. You can add a filter to the search results window by right clicking on it and selecting "Add to Search Results". Search results are discussed in detail later in this documentation.

2 Full Screen



The Full Screen command is used to toggle full-screen mode. When LogViewPlus is in full-screen mode it attempts to dedicate as much space as possible to the log entry grid and the log entry text box.

You can also toggle full-screen mode with the F11 key.

3

Grid Columns



Grid Columns

The grid columns command displays a list of columns that may be added to the current view. The columns available will change depending on the conversion pattern used when parsing the log file.

Column settings will be persisted along with the parser configuration. Modifying or recreating your parser configuration will reset your column settings.

4

Grid View



Grid View

The grid view command can be used to change the way that log entry messages are displayed within the log entry grid. Four options are available:

Show message detail: This view type can display the log entry message as multiple lines of text. Grid rows may be sized differently. This is the default view.

One log entry per grid line: This view type will display the log entry message as a single line of text. All grid rows will be the same size.

Full Message: This view type will display the log entry message in a text box underneath a grid row.

Allow Horizontal Scroll: This setting is used to turn horizontal scrolling on or off. When horizontal scrolling is turned off columns will be auto sized to fit the given area.

Grid view settings will remain active for the selected log file until the file is closed. Grid view defaults can be set in [Log Entry Display](#) settings.

5

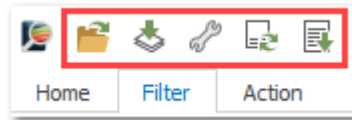
Message Size



The message size command can be used change the amount of text which is displayed in the message column of the log entry grid. This command will only be enabled if the grid view is not set to "one log entry per grid line". Message sizes range from "Very Small" to "Unlimited". We recommend that you experiment with this setting to determine the size appropriate for your log files.

Message size settings will remain active for the selected log file until the file is closed. Message size defaults can be set in [Log Entry Display](#) settings.

Quick Access



The quick access toolbar is located in the top left corner of LogViewPlus. It provides easy access to a number of commands without having to navigate menus. Note that commands available in the quick access toolbar are exactly the same as the commands available in the toolbar.

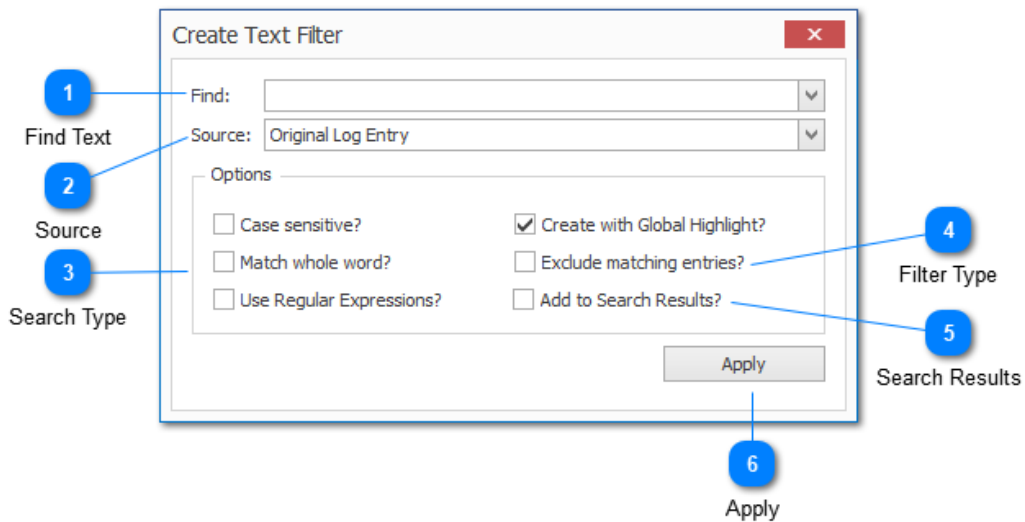
Currently, the quick access toolbar supports the following commands:

Command	Action
Open Log File	Opens the LogViewPlus file browser which can be used to open a new file.
Merge Log Files	Opens the merge file selection window.
Settings	Opens the application settings.
Tail Log File	The track changes command allows you to start or stop monitoring the current file for changes.
Auto-Scroll	Auto scroll is used to determine whether LogViewPlus should automatically scroll to new log entries. Auto-scroll is enabled by selecting the last log entry currently in the view

Toolbar Dialogs

A number of toolbar commands are used to open dialogs which request further information. These dialogs are the subject of the next section.

Text Filter



Text search filter may seem a little bit complicated when you see it for the first time. The complexity of this configuration arises due to the high degree of flexibility it provides when creating a text filter - but don't be intimidated. If you want to find text in your log files, just open the text filter, type your query, and click Apply. The default settings are appropriate for most situations. If needed, you can also [change the default settings](#).

1 Find Text

Find:

The find text box should contain the text query you want to execute. Pressing the enter key in this field will apply the current filter.

2 Source

Source:

The source drop down box contains a list of all columns currently available in the selected log file. Selecting a column will restrict this text search to that particular column. To search across all columns select the "Original Log Entry" column.

3

Search Type

Options

<input type="checkbox"/> Case sensitive?	<input checked="" type="checkbox"/> Create with Global Highlight?
<input type="checkbox"/> Match whole word?	<input type="checkbox"/> Exclude matching entries?
<input type="checkbox"/> Use Regular Expressions?	<input type="checkbox"/> Add to Search Results?

LogViewPlus supports four types of text search:

Case Sensitive: Determines the case sensitivity of the text filter. By default text searches are case insensitive.

Match Whole World: Determines whether the search should match the whole word. By default partial matches are allowed.

Use Regular Expressions: Indicates that the text provided in the find text box is a regular expression.

Create with Global Highlight: When checked a global highlight will be created when this text filter is created. This will highlight the text when it is found regardless of whether or not the filter is selected. Highlights are created by default but this can be changed in [application settings](#).

The "create as exclude filter" option is discussed separately below.

4

Filter Type

☐ Exclude matching entries?

Unlike other LogViewPlus filters, the text filters use a checkbox to set the filter type. The default filter type is inclusive. Include filters mean, "include anything that matched the filter" whereas exclude filters mean, "exclude anything that matched the filter".

5

Search Results

☐ Add to Search Results?

If selected, all log entries matching the filter will also be added to the [search results window](#). This may improve navigation.

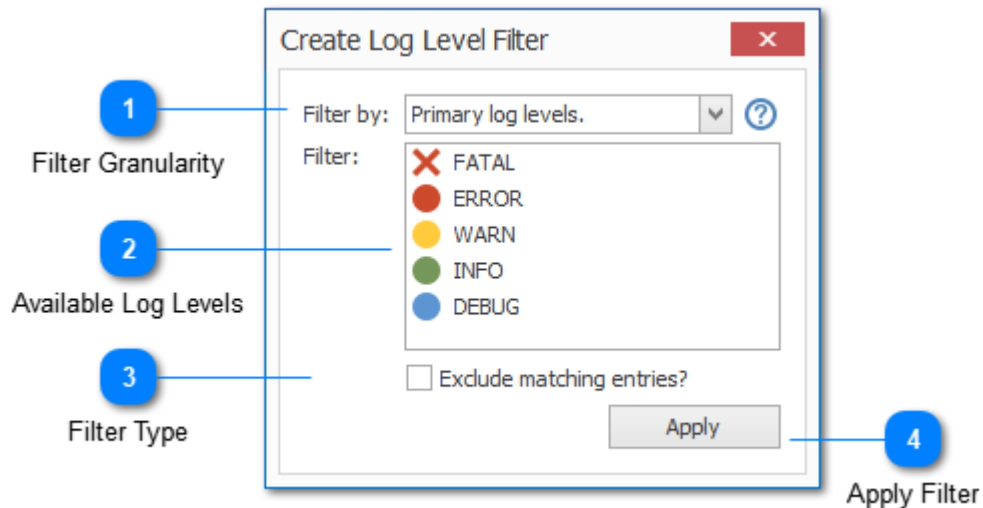
6

Apply



The apply command creates the configured filter.

Log Level Filter



The Log Level Filter configuration is more complicated than average and is worth considering individually. The complexity of this configuration arises because this filter allows you to specify the granularity of your search. For example, frequently FATAL and ALERT messages need to be treated with the same severity- you may not want to search for each log level individually.

By default, LogViewPlus supports the following log level hierarchies.

Primary	Secondary
Fatal	Alert Critical Emergency Severe
Error	
Warn	Notice
Info	
Debug	Trace Verbose Fine Finer Finest

This configuration can be modified in the [Log Levels](#) settings.

1 Filter Granularity

Filter by: Primary log levels. ▼

LogViewPlus supports filtering by log level in three different ways:

1. Primary log levels - when using this granularity level only the primary log levels will be available. Searching for 'Fatal' log levels will return any log entries which were declared as 'Alert' or 'Critical'. This is the default granularity level used by LogViewPlus. This granularity level will always be used when using the [Quick Filters](#).

2. Primary and secondary log levels - using this granularity level both the primary and secondary searches are available. When searching for a primary log level, the behavior will be as discussed above. When searching for a secondary log level, the log level will need to be matched exactly. For example, if you searched for 'Critical' and 'Debug', you would be shown 'Trace' messages if they were defined, but not 'Emergency' entries.

3. Value all log levels equally - when using this granularity level the concept of 'primary' and 'secondary' log levels is abandoned completely. In this case, the given log level will need to be matched exactly. For example, searching for 'Debug' will return only 'Debug' messages and not 'Trace' messages.

2 Available Log Levels

Filter:

- ☒ FATAL
- ☐ ERROR
- ☐ WARN
- ☐ INFO
- ☐ DEBUG

The list of log levels available for searching. You can select multiple log levels by holding down the control key when selecting. Note that the log levels list will change depending on the granularity selection. Finally, note that this list may change depending on your [Log Levels](#) settings.

3

Filter Type

☐ Exclude matching entries?

Most LogViewPlus filters allow you to specify whether they are include or exclude filters. Include filters mean, "include anything that matched the filter" whereas exclude filters mean, "exclude anything that matched the filter".

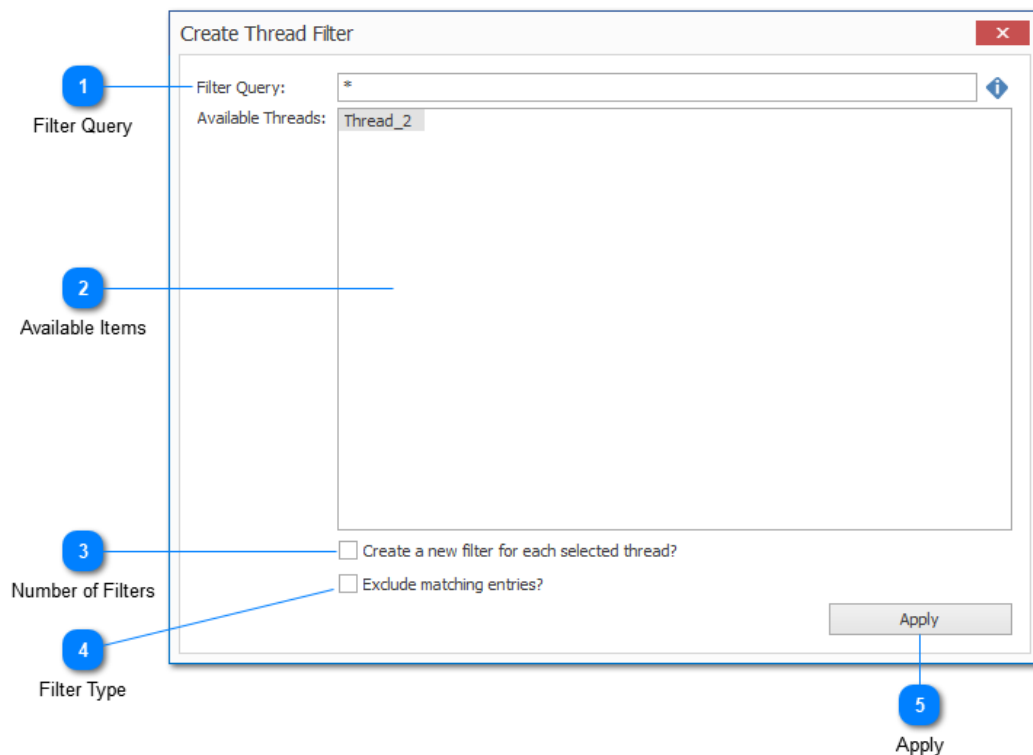
4

Apply Filter

Apply

The apply command creates the configured filter.

Value Filter



The Value Filter Dialog allows you to search within a column by selecting values. For example, the Thread and Logger [Filters](#) both use this approach to allow you to select the values which should be included in your filter. It is also possible to create multiple filters from the same dialog.

1 Filter Query

Filter Query: *

The filter query is used to define the filter. You can either manually type the filter query, or build it dynamically by selecting items from the 'Available Items' area.

If you would like to filter by data that is not currently available in the view, you can type this data manually. Alternatively, if you would like to filter by all available items as well as any new items which may appear, you can use the wildcard command '*' (as shown in the screenshot).

2

Available Items

Available Threads: Thread_2

Select items from the Available Items area to build dynamically build the filter query.

Note that the list of available items is populated by the current view. If some items appear to be missing, you may need to select a parent view.

3

Number of Filters

☐ Create a new filter for each selected thread?

By default, the filter query will be used to create a single filter. You can use this checkbox to indicate that you want to create a new filter for each matched item.

4

Filter Type

☐ Exclude matching entries?

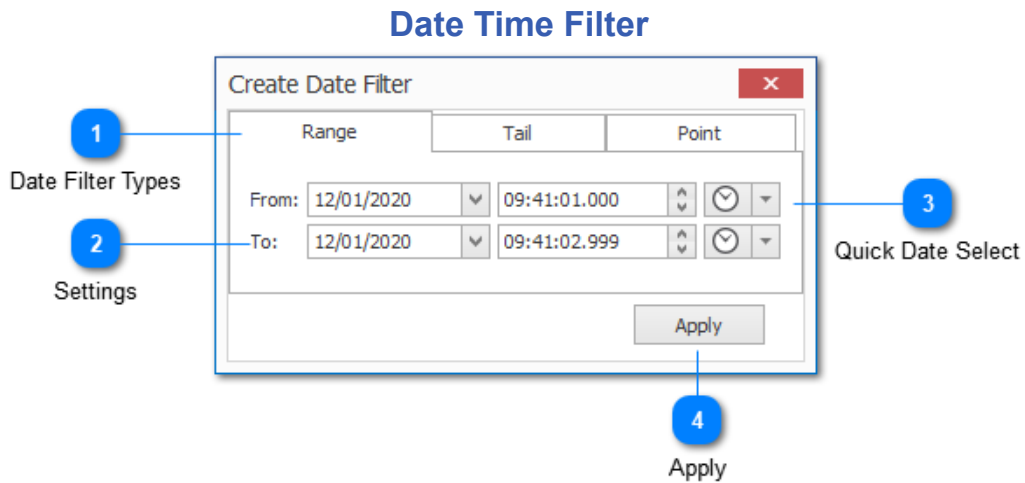
Most LogViewPlus filters allow you to specify whether they are include or exclude filters. Include filters mean, "include anything that matched the filter" whereas exclude filters mean, "exclude anything that matched the filter".

5

Apply

Apply

The apply command creates the configured filter.



Date time filters are used for creating and editing all log entry date filters. This might mean filtering log entries between two dates, or filtering log entries around a known date time.

The easiest way to create date filters is by selecting the appropriate log entries in the [Log Entries Grid](#) and then right clicking to use the [context menu](#). This will display commands that allow you to filter before, after or between the dates provided by your selection. Another easy way to create date filters is to use the [context menu](#) available from the log level [Navigation Bar](#).

While you can obviously create filters using the daytime filter form described here, we recommend creating date filters using the approaches described above. The date filter form can then be used to edit a filter and make minor changes.

1 Date Filter Types

Range	Tail	Point
-------	------	-------

The tabs at the top of the date filter window allow you to select the type of date filter you want to create. Three types are available:

Type	Description
Range	Filters log entries between the From and To dates provided.
Tail	Filters log entries created recently.

Point	Filters log entries created before or after a point in time.
-------	--

2

Settings

From:	<input type="text" value="12/01/2020"/>	<input type="button" value="v"/>	<input type="text" value="09:41:01.000"/>	<input type="button" value="^"/>	<input type="button" value="v"/>
To:	<input type="text" value="12/01/2020"/>	<input type="button" value="v"/>	<input type="text" value="09:41:02.999"/>	<input type="button" value="^"/>	<input type="button" value="v"/>

The settings displayed will depend on the filter type selected. In the example above, a range filter is selected and we are prompted to provide a From and To date.

3

Quick Date Select



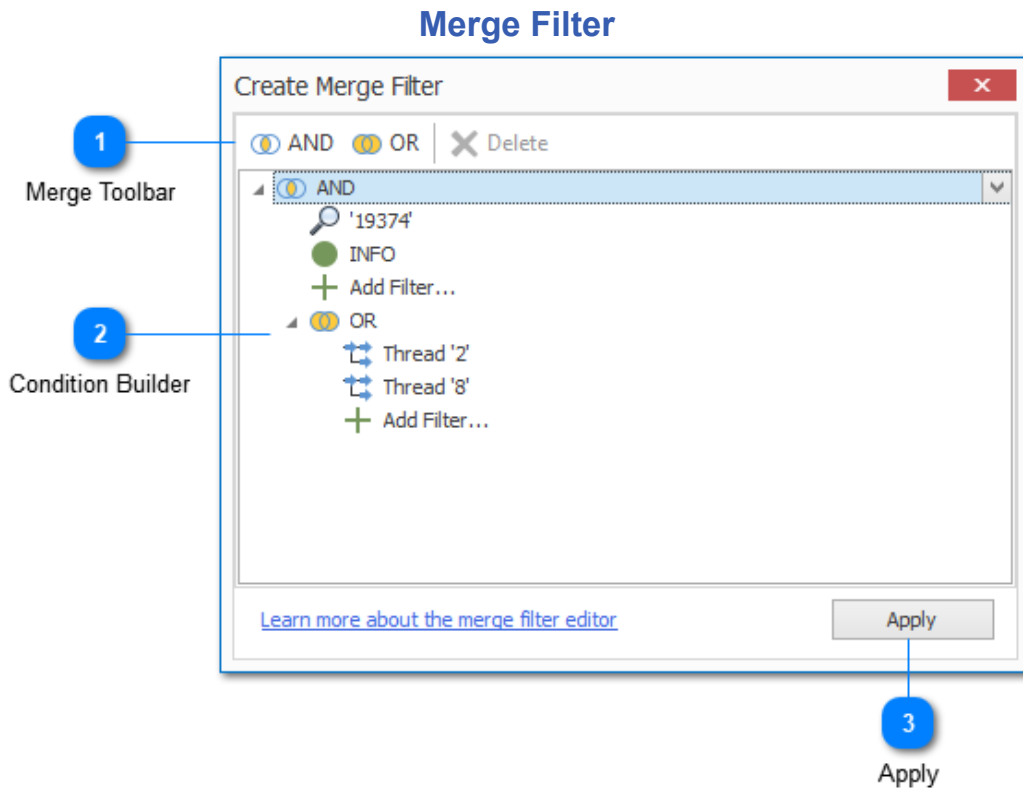
When creating a range filter, the Quick Date Select control is displayed. This control can be used to quickly input a predefined date.

The default option here is to input the date opposite the current control. For example, if the current control is the From date, the opposite control would be the To date - and vice versa. The default option can be used by simply clicking on the button. Other options are available through the pop-up menu provided.

4

Apply

The apply command creates the configured filter. The type of filter created will depend on which tab is focused.



The merge filter dialog is used to create or edit a merge filter. The dialog works exclusively with filters that have already been created.

Merges are represented as a tree where the parent node is always a conditional operation. Children can be either a pre-existing filter, or a further conditional operation. If a conditional operation is created, it must have at least two filters. Note that a single conditional can also have more than two filters.

Children of conditional operations can always be thought of as working inside parenthesis as a single operation.

For example, the merge filter above could be written as:

('19374' AND Info AND (Thread 2 OR Thread 8))

Note that existing merge filters will be included in the pre-existing filters list.

1

Merge Toolbar



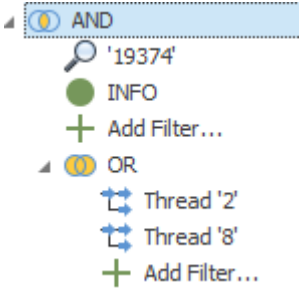
The merge toolbar is used to create or remove conditional operations from the condition builder.

The following commands are supported:

Command	Description
AND	Creates a new AND conditional.
OR	Creates a new OR conditional.
Delete	Deletes the selected conditional or filter from the current merge operation. The root conditional and 'Add Filter...' commands cannot be deleted.

2

Condition Builder



The condition builder is used to configure the merge filter. Every item in the condition builder is a drop down. For conditional operations, the drop down will contain two items - AND and OR. For filter operations the drop down will contain a list of all filters which exist in the current workspace regardless of whether those filters are available to the current log file or view.

All conditionals will always contain an 'Add Filter...' child node. This node can be used to set a new filter. After the filter is set, a new 'Add Filter...' child will be created. This allows you to set multiple filters on the same conditional.

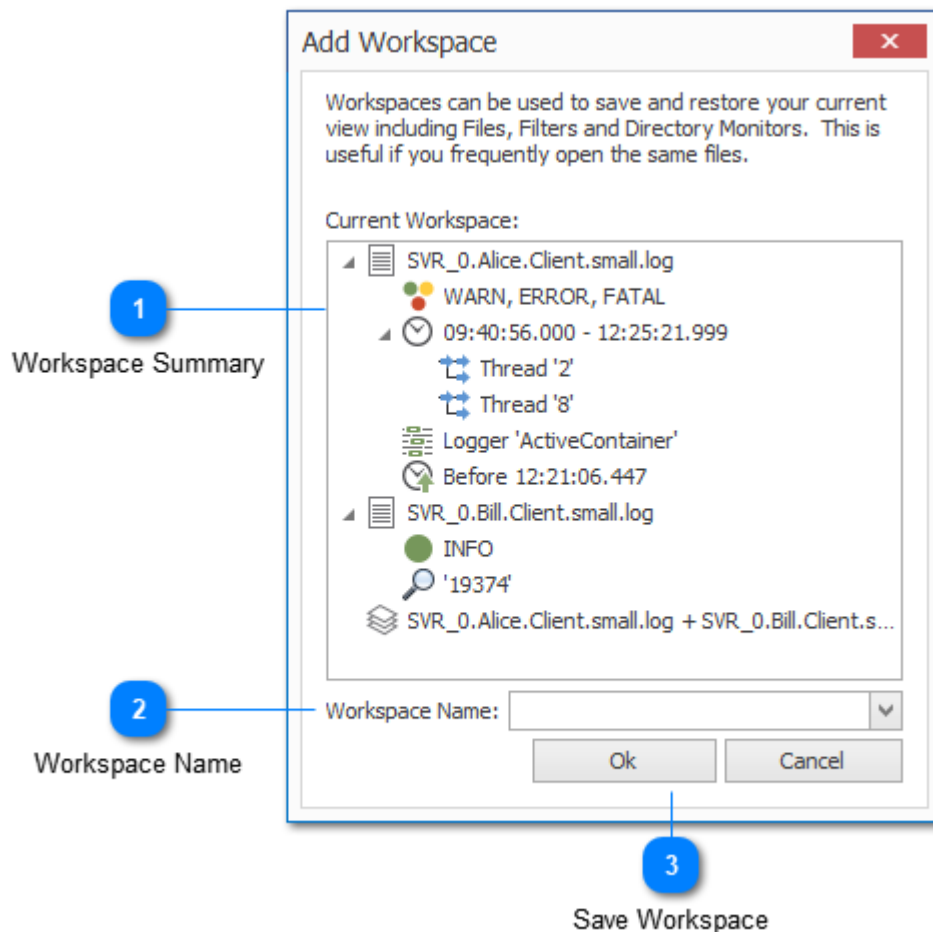
3

Apply



The apply command creates the configured filter.

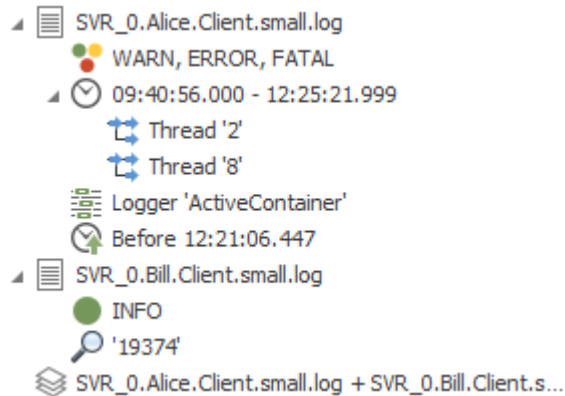
Add Workspace



The add workspace dialog is used to save the current workspace. It shows you a summary of the current workspace and allows you to set an easy to remember name which will be displayed under the workspaces pop-up.

Note that workspaces save log file locations and not the underlying data. The log file at the given location can then be re-opened in the future. If the log file is not found, it will be ignored.

1 Workspace Summary



The workspace summary shows what will be saved as part of this workspace. Workspaces can save files, filters, merged files, and directory monitors.

2 Workspace Name

Workspace Name: ▼

Choose an easy to remember name for this workspace. This name will be used to populate the drop-down box which is displayed when you access the workspaces command from the toolbar.

If you choose an existing workspace, either by using the drop-down or typing the name manually, the existing workspace will be replaced.

3 Save Workspace

Once you are done configuring your workspace you can save it or cancel the changes. Workspaces are saved across LogViewPlus sessions. Cancelled changes cannot be recovered.

Log File Properties

Properties

File Properties

Display Name: SVR_0.Alice.Client.2016-02-18.log

Size: 2.49 MB

Encoding: Windows-1252

Last Write: 12 Jan 2017 06:43:49

Last Read: 15 Jun 2017 06:37:17

Parser Settings

Name Pattern: SVR_*.log [Open Settings](#)

Parser Type: PatternParser

Post Processor: None

Arguments: %d{yyyy-MM-dd %H:mm:ss,fff} [%t] [%s] [%s] [%s] [%s]

☐ Remote Path ☒ Local Path

sftp://localhost:22/Small/SVR_0.Alice.Client.2016-02-18.log

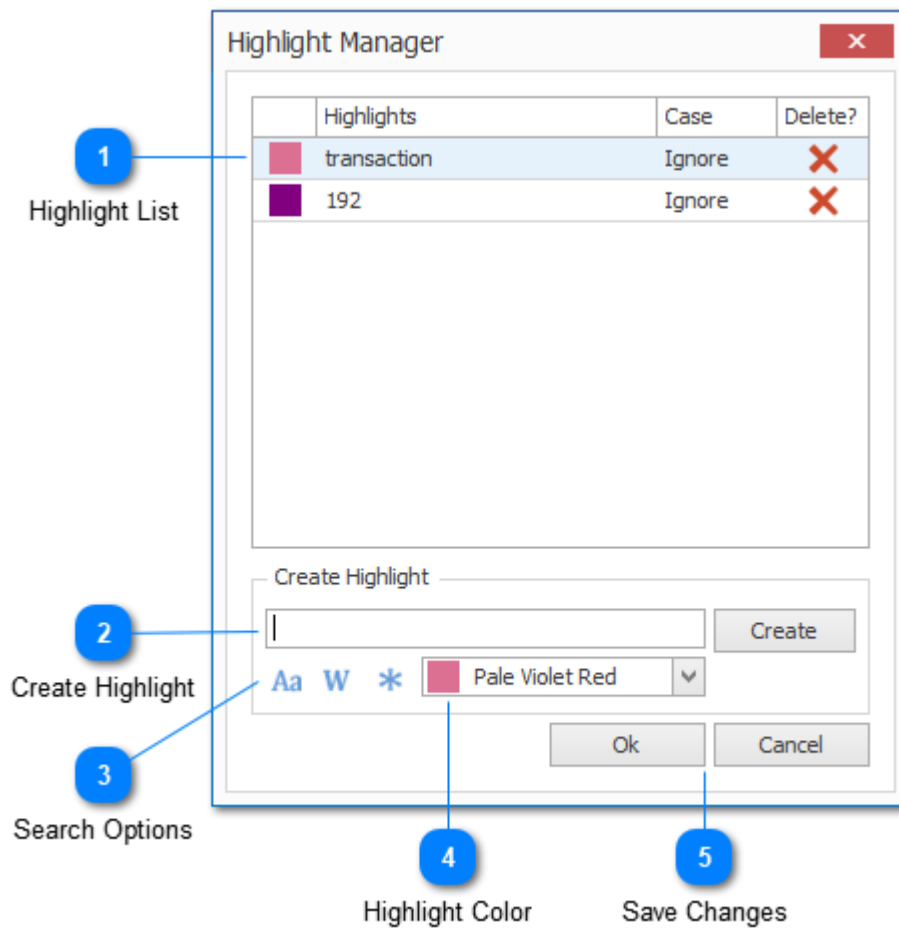
OK

The log file properties window displays a variety of information about the selected log file including file size, encoding, and details on how the log file was parsed.

The log file properties window is context-sensitive. The displayed information may change depending on whether the selected log file is local, remote, or merged.

The information displayed in the log file properties window will change depending on the type of log file selected.

Highlight Manager



Highlighting allows you to search for text and mark it to make it easier to find. The LogViewPlus highlight manager makes it easy to manage all of your text highlights.

1 Highlight List

	Highlights	Case	Delete?
	transaction	Ignore	✗
	192	Ignore	✗

The highlight list shows all current application highlights. The highlight list is editable and allows you to change the color text or case sensitivity of the given highlight. To delete highlight click the red X icon in the delete columns.

Note that when the highlight list is in dialog mode changes will only be saved when the OK button is clicked.

2 Create Highlight

A dialog box for creating a highlight. It features a single-line text input field on the left and a button labeled "Create" on the right.

The create highlight area allows you to create a new highlight. Simply type the text you want to highlight into the text box provided and click the create command.

Creating a highlight with additional settings such as search options and highlight color is discussed below.

3 Search Options

Aa W *

The search highlight options allow you to specify how the text search should be executed. The options include text case sensitivity, whole word matches, and the use of regular expressions.

For more information on text search execution options, please see the [text filter](#) documentation.

4 Highlight Color

A dialog box for selecting a highlight color. It shows a color swatch of "Pale Violet Red" next to a dropdown arrow.

The highlight color drop-down allows you to determine a color for the new highlight.

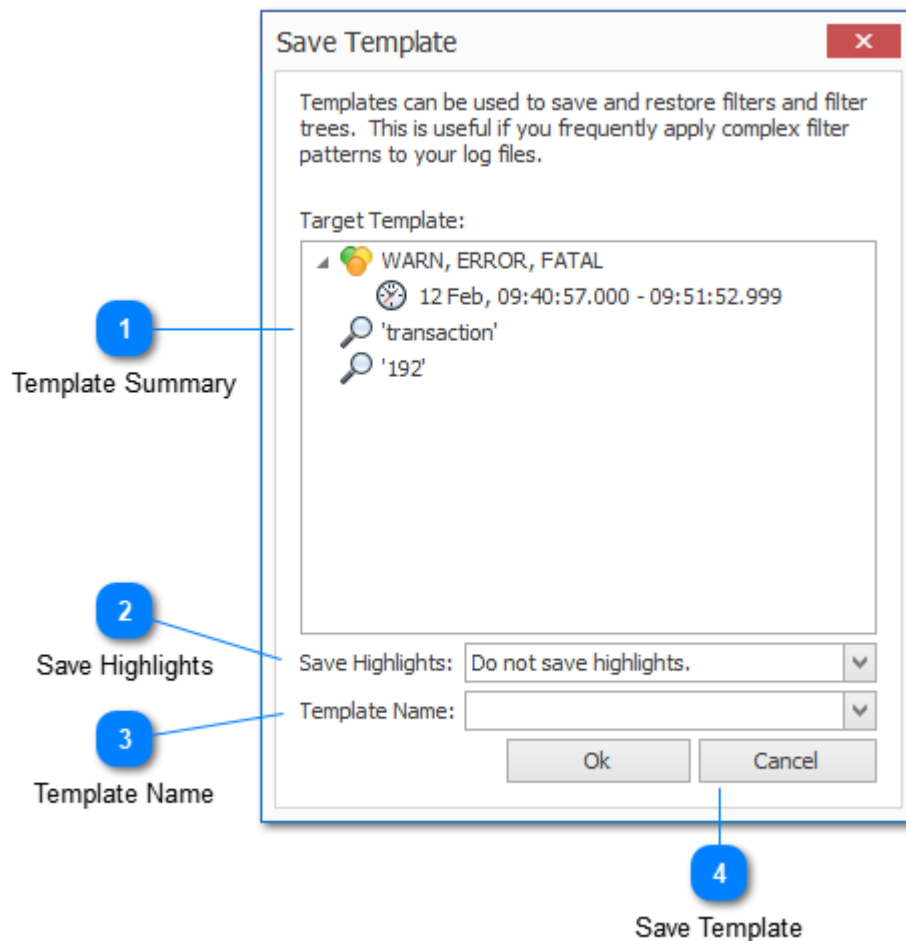
5 Save Changes

A dialog box with two buttons: "Ok" and "Cancel".

Once you are done configuring your highlights you can save it or cancel the changes. Cancelled changes cannot be recovered.

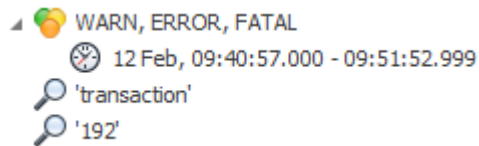
Note that the option to save changes will only be available when the highlight manager is in dialog mode. Otherwise, changes will be saved automatically.

Add Template



A template is a collection of pre-saved filters or highlights which can be applied to the current log file. Templates are useful when you find yourself frequently applying the same highlight, filter or set of filters to different log files in different LogViewPlus sessions.

1 Template Summary



The template summary shows what will be saved as part of this template. In the above example four filters will be saved.

2

Save Highlights

Save Highlights: ▼

The save highlights combo box allows you to specify how highlights should be saved in this template. Note that the template can contain any combination of filters and highlights. Highlights can be saved without a corresponding filter.

The highlight save options available are:

Do not save highlights: No highlights will be saved.

Save all user generated highlights: Only highlights explicitly created by the user will be saved. Highlights created automatically, such as through the creation of text filters, will not be saved.

Save all highlights: All highlights will be saved regardless of how they were created.

3

Template Name

Template Name: ▼

An easy to remember name for this template. This name will be used to populate the drop-down box which is displayed when you access the templates command from the toolbar.

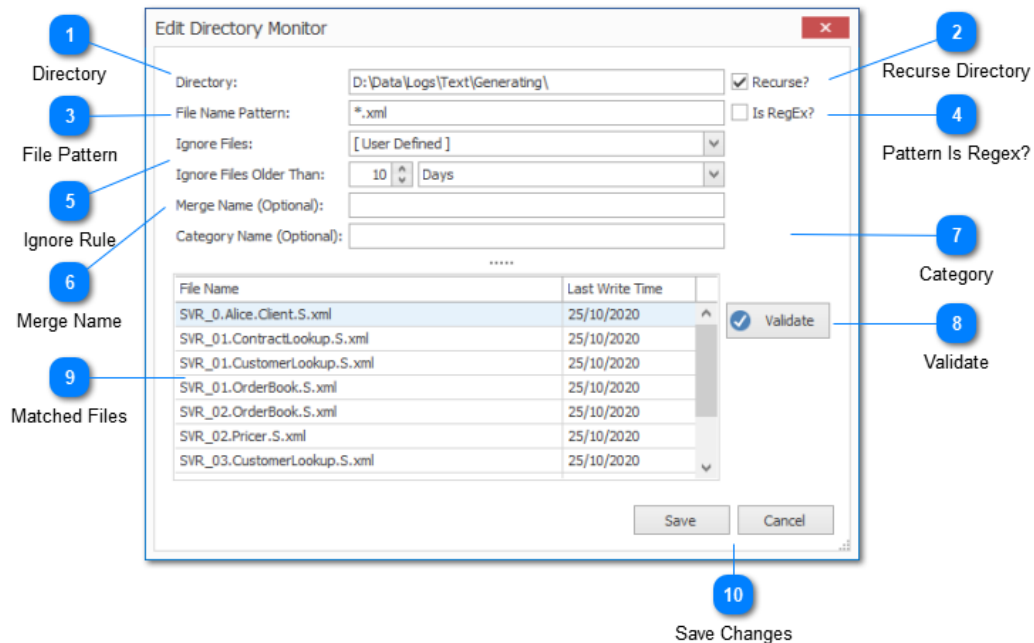
If you choose an existing template, either by using the drop-down or typing the name manually, the existing template will be replaced.

4

Save Template

Once you are done configuring your template you can save it or cancel the changes. Templates are saved across LogViewPlus sessions. Cancelled changes cannot be recovered.

Edit Directory Monitor



The directory monitor dialog is for adding and editing directory monitors. Directory monitors track a directory for new log files. If a new log file is found it can be automatically opened and merged.

1 Directory

Directory:

The full path to the directory you would like to monitor.

2 Recurse Directory

☒ Recurse?

The recurse check box is used to indicated if the given directory should be searched recursively for matching files. A recursive directory search will search the parent and all child directories. If the search is not recursive, then only the top level directory will be searched.

3

File Pattern

File Name Pattern:

The file name pattern is used to identify whether or not a new file in this directory should be opened in LogViewPlus. In the example above all XML files in the directory will be treated as new.

If you only want to open files with a ".log" extension, you could use the pattern "*.log". Note that the filename pattern can optionally be a regular expression. If you are using a regular expression you will need to check the "Is Regex" checkbox discussed below.

4

Pattern Is Regex?

☐ Is RegEx?

The "Is RegEx" checkbox is used to indicate if the provided file search pattern is a regular expression. Note that wild card searches such as "*.log" are supported in searches which do not have regular expressions enabled.

5

Ignore Rule

Ignore Files:

Ignore Files Older Than:



Days

The ignore rule allows you to ignore files which are out of date. Three possible rules are available:

Today - Matches all files modified today.

Track All Files - Matches all files regardless of age.

[User Defined] - Enables inputs to allow you to specify a custom time range. Files out of range will be ignored.

6

Merge Name

Merge Name (Optional):

If a merge name is provided, all files opened by the Directory Monitor will be added to the new merge file. This will act as a single view showing all log entries processed by the Directory Monitor.

7 Category

Category Name (Optional):

Setting a category on a directory monitor will put all files opened by the directory monitor into a folder in the workspace. Categories are useful for log file organization and will be saved as part of the workspace. For example, you may have categories like UAT and PROD in order to separate environments.

8 Validate



Once you're happy with your directory monitor configuration you can click the validate button to confirm the validity. If the directory monitor configuration is valid the matched files grid will display all files currently matched by the directory monitor. Note that your directory monitor may be valid even if no files are matched.

Note that if you change the directory monitor settings you will need to validate the settings again.

9 Matched Files

File Name	Last Write Time
SVR_0.Alice.Client.S.xml	25/10/2020
SVR_01.ContractLookup.S.xml	25/10/2020
SVR_01.CustomerLookup.S.xml	25/10/2020

The matched files grid shows you the results of your current directory monitor configuration. In the above example it shows the files that are matched by the current configuration. It may also display an error message if the directory monitor has been improperly configured.

The matched files grid will only be updated when you validate your settings.

10

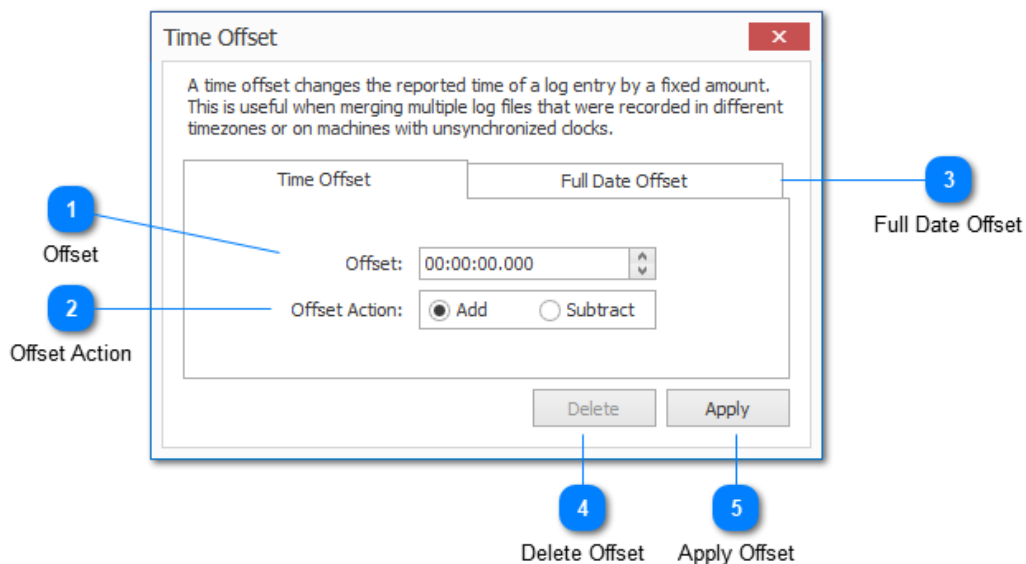
Save Changes

Save

Cancel

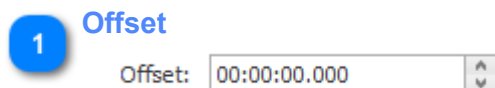
Once you are done configuring your directory monitor you can save or cancel the changes. Cancelled changes cannot be recovered.

Time Offset



The time offset dialog can be used to modify all timestamps in the currently selected log file by a user configured amount. This command is useful when you intend to merge multiple log files but the timestamps of the log files are not in sync. For example, if you need to merge multiple log files which are recorded in local time in different time zones. The command is also useful when log files recorded on different machines are a millisecond or two out of sync.

Note that this command is not available on merged files or filters.



The offset is set in hours, minutes, seconds and milliseconds.



The offset action can be either add or subtract depending on if you need to increase or decrease the time interval.

3

Full Date Offset

Full Date Offset

If you would prefer to set an offset to a specific time, you can use the Full Date Offset tab. This tab shows two full timestamps. The first is a reference point and cannot be modified. The second is the date that you would like to use instead of the reference time provided. The difference between these two timestamps will be used to calculate the time offset.

4

Delete Offset

Delete

Deletes the currently applied offset. Note that this command will be available only if the currently selected log file has an offset applied.

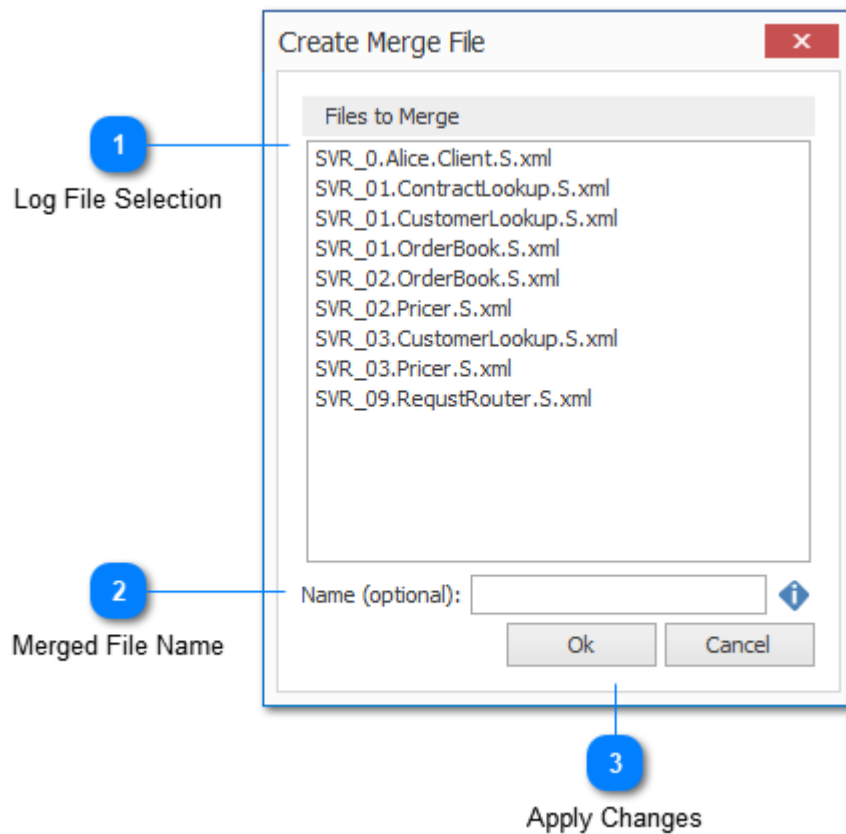
5

Apply Offset

Apply

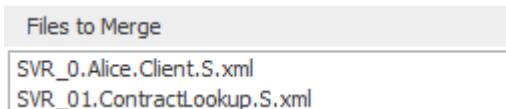
Applies the defined offset to the currently selected log file. This will cause the log file to be reloaded.

Merge File Selection



The merge file selection window is used to select multiple log files to be consolidated into a single log file. Once a merged log file has been created, it can be used just like any other log file in LogViewPlus.

1 Log File Selection



The log file selection box is used to select two or more log files to be merged into a single view. All log files currently open in LogViewPlus, including previously merged log files, will be listed as possible targets.

You can use the shortcuts CTRL+A to select all log files. If all log files are selected, you can press CTRL+A to deselect all log files.

You can also use the CTRL key with the mouse to individually select or deselect log files.

2

Merged File Name

Name (optional):

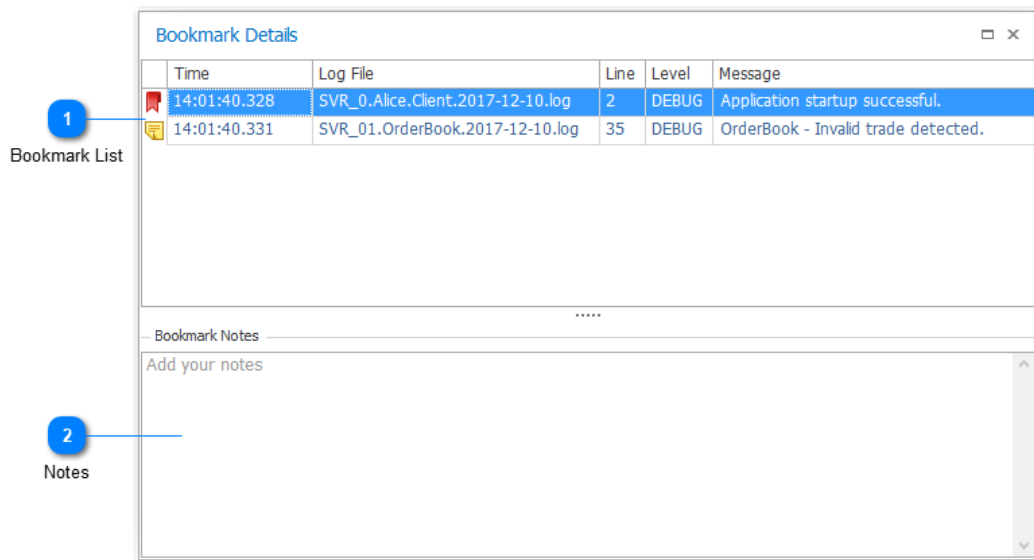
A text box is provided to give you the opportunity to name your new merged log file. If a merged log file name is not provided, one will be automatically generated based on the source log files.

3

Apply Changes

Use the 'OK' button to generate the merge file. Alternatively the 'Cancel' command can be used to return to LogViewPlus.

Bookmark Detail



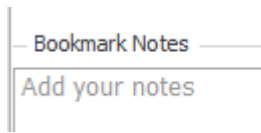
The Bookmark Detail View helps you easily manage your bookmarks and notes. To dock this window, simply drag the window over the LogViewPlus window and selected docking position. To undock, simply drag the window outside LogViewPlus. Note that the bookmark detail view will change orientation automatically based on the window layout.

1 Bookmark List

🔖	14:01:40.328	SVR_0.Alice.Client.2017-12-10.log
📄	14:01:40.331	SVR_01.OrderBook.2017-12-10.log

The bookmark list shows all available bookmarks. Double-clicking on one of the bookmarks will automatically navigate to the bookmarked log entry. Right clicking on a bookmark will bring up the [bookmark commands](#) context menu which allows simple actions like creating quick date filters, navigation, elapsed time, and bookmark management.

2 Notes



The notes area can be used to add or view any notes associated with the currently selected bookmark.

Search Results

Search Results					
WARN x		'transaction' x			
Time	Thread	Level	Logger	Message	
10:00:22.175	Thread_8	INFO	TypedRefNum	Transaction placed successfully.	
10:00:22.356	Thread_6	INFO	Splitter	Client 'Alice' transaction complete.	
10:00:28.013	Thread_2	INFO	Client	Client 'Alice' transaction initiated.	
10:00:28.384	Thread_3	TRACE	RingList	ESENT database transaction completed.	
10:00:29.154	Thread_6	INFO	ActiveContainer	Transaction details: <?xml version="1.0" encoding="utf-8" ?> <SERIES-AND-CLASSES-CONTRACTS-DATA> <MERGER-SERIES-AND-CLASSES-CONTRACTS> <MERGER> <SERIES OWNER-CIK="" SERIES-ID="" SERIES-NAME="">	
10:00:31.968	Thread_2	INFO	WaveformData	Transaction placed successfully.	
10:00:32.156	Thread_2	INFO	ClassPropDefFolder	Client 'Alice' transaction complete.	

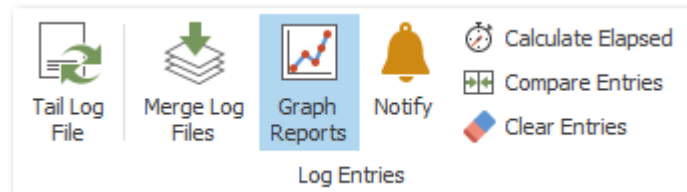
The search results window is simply another way to view a filter. This window can be docked within the LogViewPlus and this allows you to view and navigate your search results as well as another filter at the same time. This is particularly helpful if you find yourself frequently navigating back and forth between a filter and a parent view.

Double-clicking on a search result will navigate to the target log entry in the currently selected view. If the log entry does not exist in the currently selected view, the log entry will be found in the root log file instead.

To add a filter to the search results view, right click on the filter and select "Show in Search Results".

Multiple filters can be added to the search results window. Each filter will be added as a separate tab within the window.

Graph Reports



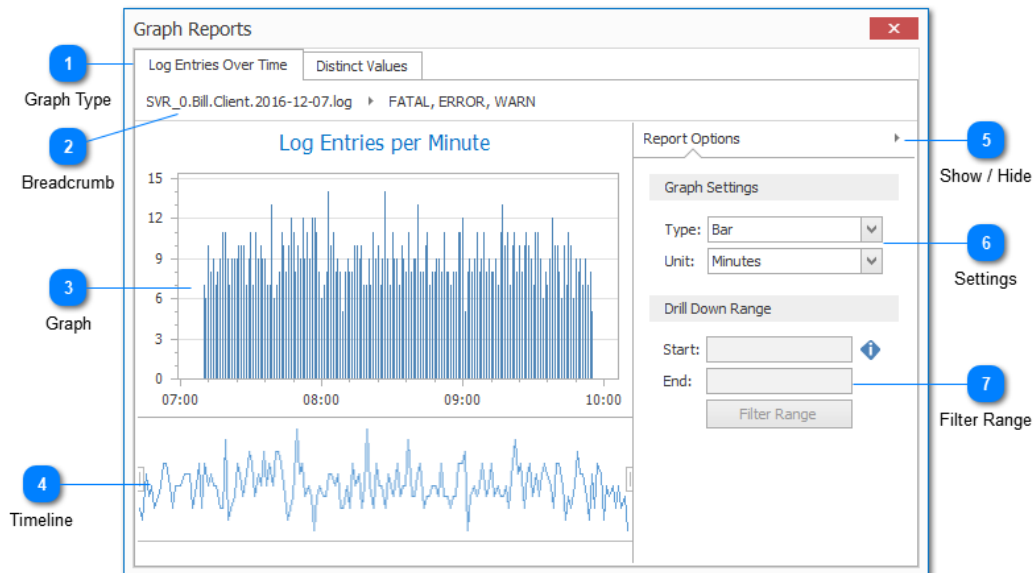
LogViewPlus can parse your log files and transform the data into information which can then be [exported](#) to a CSV file or HTML.

As you work with the information displayed in LogViewPlus you may find it helpful to see the information displayed as a graph. LogViewPlus can generate graphs based on any log entry view (any log file or filter). These graphs can give you better insight into your application behavior.

The graphs can also be used to further filter the data. Using the LogViewPlus Graph Reports to generate filters to further subdivide and analyze your log files can be a powerful tool.

Currently, LogViewPlus supports two main graph types: [Log Entries Over Time](#) and [Distinct Values](#).

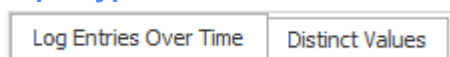
Log Entries Over Time



The log entries over time report shows the number of log entries which are contained in the current view at a given time interval. This can be used to better understand how log entries are distributed through the current view.

1

Graph Type



LogViewPlus supports two primary graph types. The first shows log entries over time. This graph is helpful for giving you a quick feel for how your log entries are distributed over time. The second graph type shows the distribution of values for individual columns. This type of graph can be useful for answering questions such as what is the ratio of debug log entries to error log entries.

2

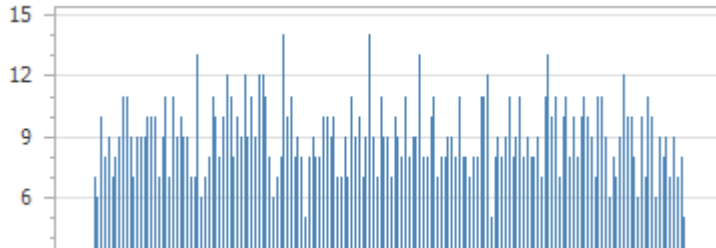
Breadcrumb



The breadcrumb navigation is used to manipulate the filter tree. Modifying the breadcrumb navigation changes the selected filter which will automatically update the report.

3

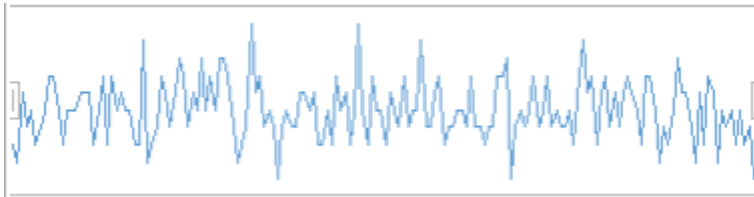
Graph



The graph area shows the currently selected graph drawn according to the graph settings provided. Note that hovering over the graph will display information about the currently selected data point. In this case, the number of log entries that were written at a particular point in time.

4

Timeline



By default the timeline of a graph will be large enough to include all log entries in a view. However, you can use the handle to the left and right of the timeline to narrow the graph down to an area that you find more interesting. The Filter Range command (discussed below) will allow you to create a new [Date Time Filter](#) from the selected timeline range. This allows you to drill into the data you find interesting.

5

Show / Hide



You can show or hide the graph settings by clicking on the report options title. This is helpful when you want to provide more room to draw the graph.

6

Settings

Type: ▼

Unit: ▼

The log entries over time graph can be displayed as either a bar or line chart. We recommend using a bar chart as line charts can sometimes be misleading if data for a particular point in time is not available. In this case, a lined can be drawn through a phantom point.

LogViewPlus will attempt to detect a time unit based on the timestamps contained in the view. However, you can also manually change the time unit.

Changes to graph settings will be applied automatically.

7

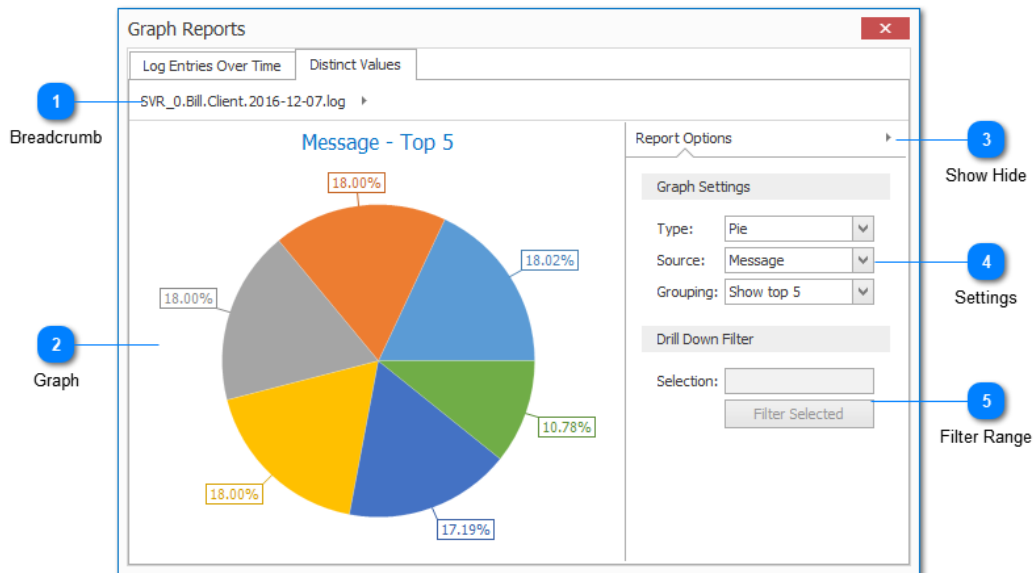
Filter Range

Start:

End:

Allows you to create a [Date Time Filter](#) for the selected timeline range (discussed above). This helps you find out more about the log entries that were created during the selected time range.

Distinct Values



The distinct values report is used to group the common values found in a given column. For example, you can use this report to find the thread or logger which produced the most log entries in the current view.

The values contained in the message column is treated a little bit differently from other columns. For messages, LogViewPlus will generate a template which attempts to extract any variable data from the core message. When creating a template, LogViewPlus will substitute out any numeric values or values contained between brackets.

Functionally, the distinct values report is very similar to the [log entries over time](#) report. The data used by the Distinct Values report is always provided by the current view.

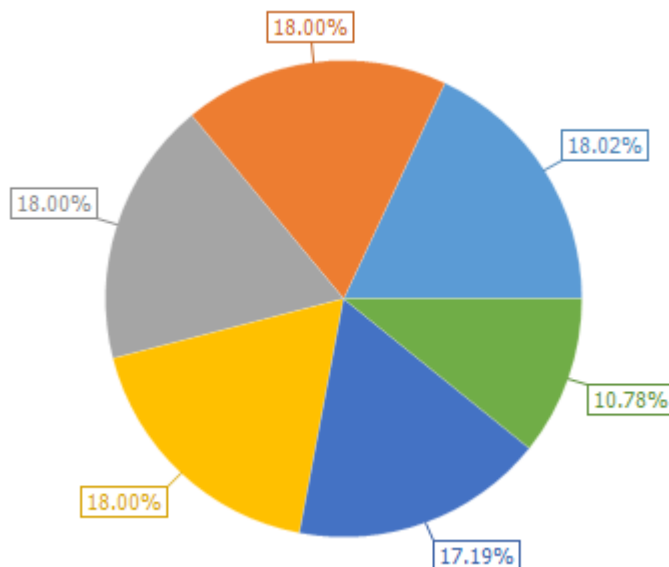
1 Breadcrumb

SVR_0.Bill.Client.2016-12-07.log

The breadcrumb navigation is used to manipulate the filter tree. Modifying the breadcrumb navigation changes the selected filter which will automatically update the report.

2

Graph



The graph area shows the currently selected graph drawn according to the graph settings provided. Hovering over the graph will display information about the currently selected data point. In this case, a description of the value identified along with statistical information regarding its commonality.

You can double-click a graph element to automatically create a filter based on the selected value.

A legend will automatically be shown in the graph if the window width allows.

Clicking on a graph element will populate the drill down filter (discussed below).

3

Show Hide



You can show or hide the graph settings by clicking on the report options title. This is helpful when you want to provide more room to draw the graph.

4 Settings

Type:	<input type="text" value="Pie"/>	▼
Source:	<input type="text" value="Message"/>	▼
Grouping:	<input type="text" value="Show top 5"/>	▼

The distinct values report can be displayed as either a pie or bar chart.

You can also change the source column to report on data from different parts of the view.

Finally, you can change the number of values to be grouped. Values which fall outside of the group number will automatically be categorized into an 'Other' category. You can filter on the 'Other' category to find more information about its constituents.

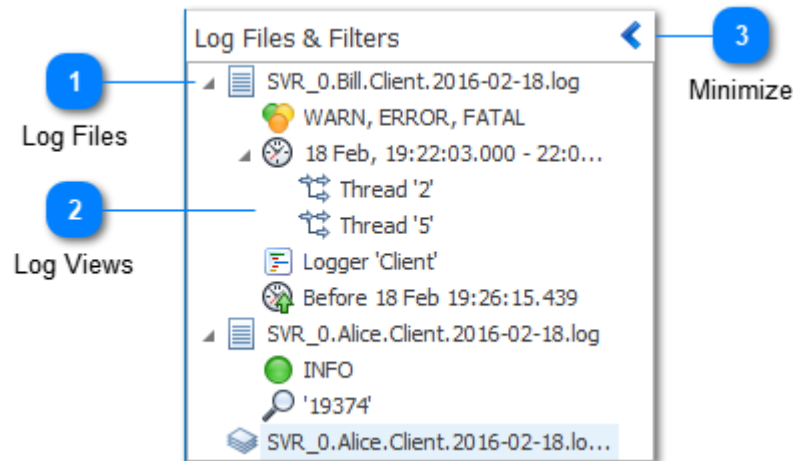
Changes to graph settings will be applied automatically.

5 Filter Range

Selection:	<input type="text"/>
<input type="button" value="Filter Selected"/>	

LogViewPlus can create a [Text Filter](#) from the selected chart element. The text filter will automatically be applied specifically to the column currently being analyzed.

File Management



On the far left of LogViewPlus window, you should see a tree list which shows you all of the files and views you currently have open. You can use this list to navigate between log files.

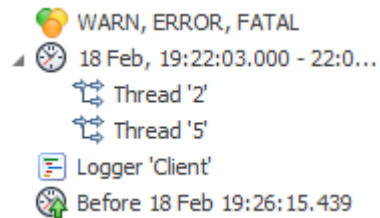
Note that this list represents a hierarchy of views on your log file. The element at the base of the hierarchy represents all entries in the log file. As you move away from the base element each generation will have an increasingly focused view of the current log file. Each new generation will have a subset of the log entries available in the previous generation. As you navigate between views LogViewPlus attempts to keep your current record in focus. This allows you to easily view records which were written before and after the current entry - even when moving between views.

1 Log Files

SVR_0.Bill.Client.2016-02-18.log

Root items represent log files. The root items always contain all log entries available the log file.

2 Log Views



Views can be thought of as child log files. They do not contain all records available in the log file. The views work by filtering out some items. Note the child views contain a subset of their parent. Therefore each successive generation narrows the records available.

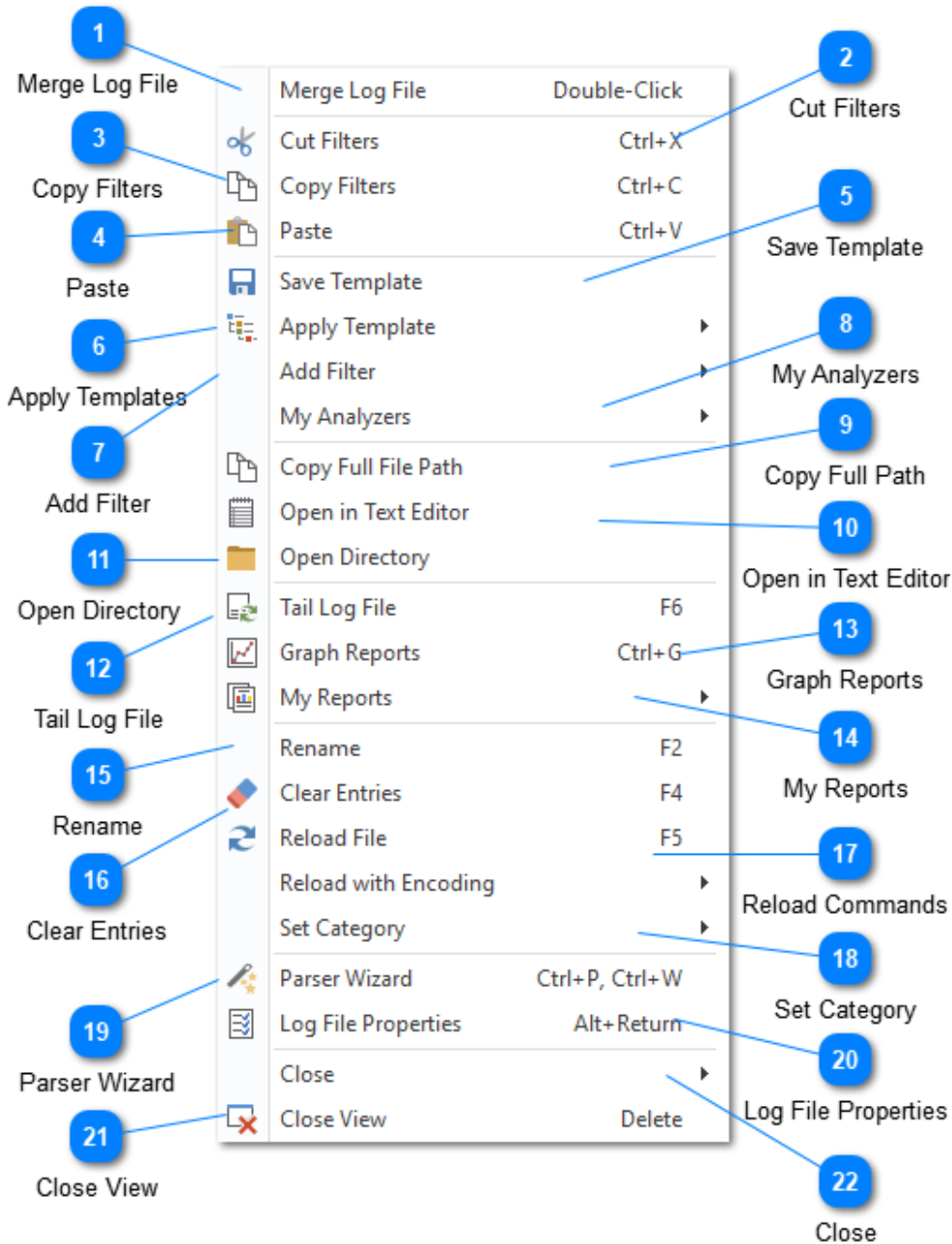
Minimize



The minimize command can be used to hide the file management tree. This dedicates more screen room to the log entry grid.

Once minimized, the file management tree can be restored either by clicking on the minimize command again or by double-clicking on the log files panel.

File Context Menu



The file context menu is available whenever you right-click on a file or merged file in LogViewPlus.

1

Merge Log File



Double-clicking on a log file will bring up the [Merge File Dialog](#) with the current log file selected. If the current file is a merge file, the dialog will have all of the constituent parts selected and the action will be to edit the merge file by adding or removing files.

2

Cut Filters



Cuts all of the filters currently set on this log file to the clipboard. If these filters are later pasted within this LogViewPlus session, the selected filters will be moved.

3

Copy Filters



Copies all of the filters currently set on this log file to the clipboard.

Filters are copied to the clipboard as plain text. This allows them to be easily shared with other LogViewPlus users.

4

Paste

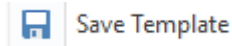


If a filter has been copied to the clipboard, the paste filter command will apply it to the currently selected log file or view.

If the clipboard contains plain text, it will be converted into a [Text Filter](#).

5

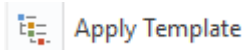
Save Template



The save template command allows you to save all of the filters which are currently applied to the log file as a [template](#). Any information associated with the filter such as notifications or notes will also be saved.

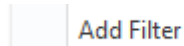
Saving filter templates helps you to quickly reapply them later.

6 Apply Templates



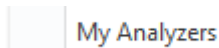
The apply templates command will show a list of all pre-saved templates. Selecting a template will apply it to the current log file.

7 Add Filter



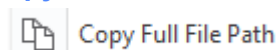
The add filter command displays a list of all filters which are currently available in the [filter toolbar](#). These commands are identical and provided here for convenience only.

8 My Analyzers



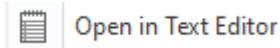
This command can be used to run a [custom analyzer](#). This command will only be available if a custom analyzer plugin has been detected by the LogViewPlus instance.

9 Copy Full Path



Copies the full path to the current log file to the clipboard.

10 Open in Text Editor



Open in Text Editor

Opens the currently selected log file in your [default text editor](#).

11

Open Directory



Open Directory

Opens the directory which contains the currently selected log file in the [File Explorer](#).

You can also hold down the CTRL key when executing this command to open the target directory in Windows Explorer. If this option is used and the log file is a remote log file (accessed via SFTP or FTP) then LogViewPlus will open the temporary directory which contains the local version of the log file.

12

Tail Log File



Tail Log File

F6

The Tail Log File command will toggle [tailing the log file](#).

13

Graph Reports



Graph Reports

Opens the [Graph Reports](#) window. This window will display the currently selected log file as a graph base report which can then be filtered.

14

My Reports



My Reports

If you have configured LogViewPlus to generate [custom reports](#), those reports will be available for execution. This command will not be visible if custom reports have not been configured.

15

Rename

 Rename F2

Renames the currently selected log file.

16

Clear Entries



 Clear Entries F4

Removes all log entries from the current log file. If the file is currently being tailed new entries will appear automatically.

The underlying log file will not be modified. No changes will be written to disk.

17

Reload Commands

 Reload File F5
 Reload with Encoding

Reloading the log file is useful if you have previously cleared all log entries and now need to see the cleared entries. It can also be helpful when you are modifying parser configurations, or to refresh a file which is not currently tailed.

The reload with encoding command allows you to reload a log file with one of the encodings provided in the [Common Encodings](#) application settings.

18


Set Category

 Set Category

Setting a category for a log file will put it into a folder in the workspace. Categories are useful for log file organization and will be saved as part of the workspace. For example, you may have categories like UAT and PROD in order to separate environments.

19

Parser Wizard

 Parser Wizard

Opens the [Parser Wizard](#) which can be used to quickly add or modify the parser configuration for the currently selected log file.

20

Log File Properties



Opens the [log file properties dialog](#).

21

Close View



Closes the current log file and any child views.

22

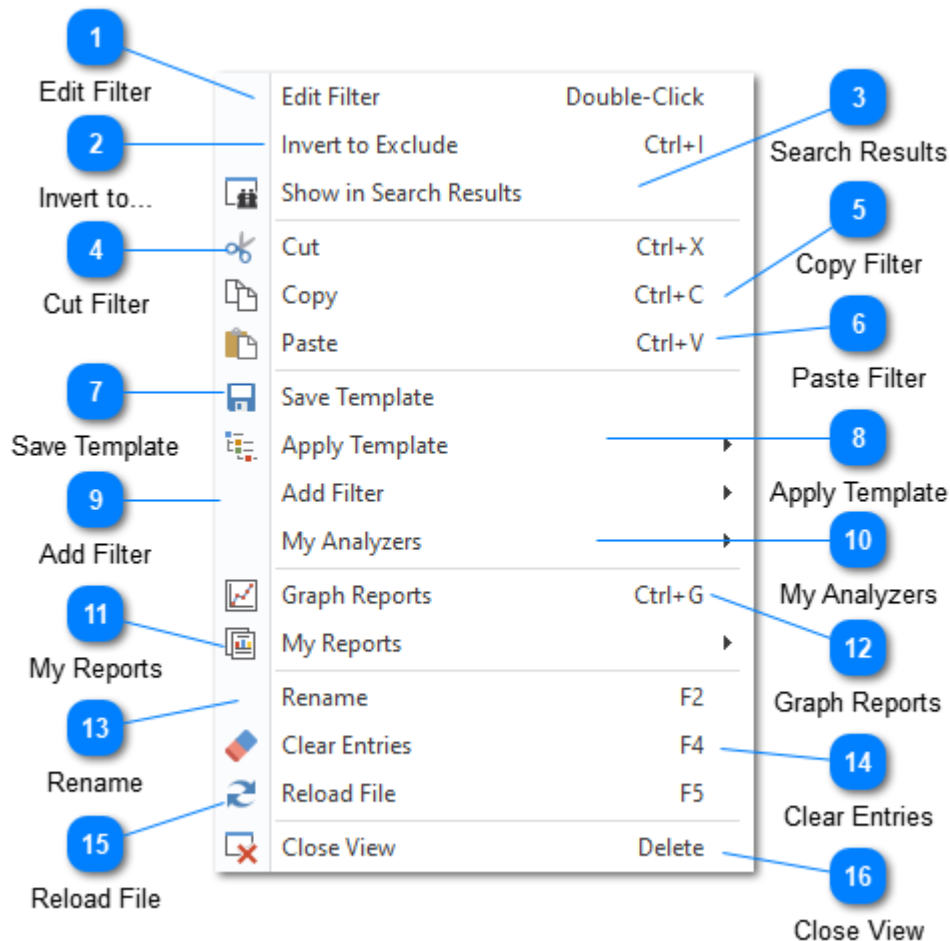
Close



The close command brings up a sub-menu of different close options:

Command	Description
Close All	Closes all log files, filters, and directory monitors.
Close All But This	Closes all log files and filters except for the selected file.
Close Source Files	Closes all source log files. Available for merge files only.
Close All Filters	Closes all filters across all log files.
Close View	Closes the current view.

Filter Context Menu



The filter context menu is available whenever you right-click on a filter in LogViewPlus.

1 Edit Filter

Edit Filter Double-Click

The edit filter command allows you to edit an existing filter. This is the same as double-clicking on the filter if the filter does not have any child filters. If the filter has children then the node will be expanded or collapsed.

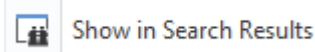
The double-click behavior is [configurable](#). The default action is to edit the filter.

2 Invert to...



The "Invert to..." command will switch the currently selected filter type to either Include or Exclude. This command will only be enabled when the target filter type supports excluding log entries.

3 Search Results



Adds the currently selected filter to the [search results](#) window.

4 Cut Filter



Cuts the currently selected filter to the clipboard. If this filter is later pasted within this LogViewPlus session, the selected filter will be moved.

5 Copy Filter



Copies the currently selected filter to the clipboard.

Filters are copied to the clipboard as plain text. This allows them to be easily shared with other LogViewPlus users.

6 Paste Filter



If a filter has been copied to the clipboard, the paste filter command will apply it to the currently selected log file or view.

If the clipboard contains plain text, it will be converted into a [Text Filter](#).

7

Save Template

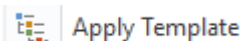


The save template command allows you to save all of the filters which are currently applied to the log file as a template. Any information associated with the filter such as notifications or notes will also be saved.

Saving filter templates helps you to quickly reapply them later.

8

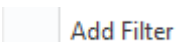
Apply Template



The apply templates command will show a list of all pre-saved templates. Selecting a template will apply it to the current filter.

9

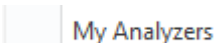
Add Filter



The add filter command displays a list of all filters which are currently available in the [filter toolbar](#). These commands are also available on the toolbar and are provided here for convenience only.

10

My Analyzers



This command can be used to run a [custom analyzer](#). This command will only be available if a custom analyzer plugin has been detected by the LogViewPlus instance.

11

My Reports



If you have configured LogViewPlus to generate [custom reports](#), those reports will be available for execution. This command will not be visible if custom reports have not been configured.

12 Graph Reports



Opens the [graph log entries](#) dialog. The screen can be used to display the currently selected filter as a graph.

13 Rename



Renames the currently selected filter.

14 Clear Entries



Removes all log entries from the current log file. If the file is currently being tailed new entries will appear automatically.

The underlying log file will not be modified. No changes will be written to disk.

15 Reload File



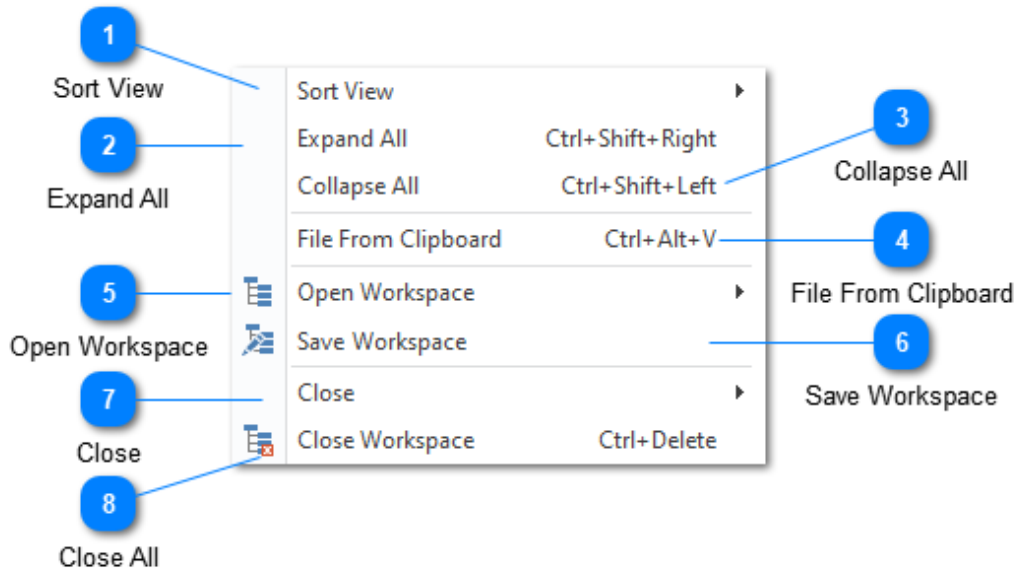
Reloading the log file is useful if you have previously cleared all log entries and now need to see the cleared entries. It can also be helpful when you are modifying parser configurations, or to refresh a file which is not currently tailed.

16 Close View



Closes the current filter and any child filters.

Workspace Menu



The workspace context menu is available whenever you right-click on an empty area in the [file management](#) control.

1 Sort View

Sort View

You can sort your workspace either alphabetically (ascending or descending) or by the order in which items were created. Sorting occurs independently at each level of the tree.

2 Expand All

Expand All Ctrl+Shift+Right


Expands all levels in the tree.

3 Collapse All

 Collapse All Ctrl+Shift+Left


Collapses all levels in the tree.

4 File From Clipboard

 File From Clipboard Ctrl+Alt+V


Writes the clipboard contents to a temporary file and then attempts to automatically parse the file. This command is useful when your log entries are extracted from another program and not yet available as a file.

5 Open Workspace

 Open Workspace


The open workspace command will show a list of all saved workspaces. Selecting a workspace will apply it to the current view.

6 Save Workspace

 Save Workspace

Saves the current workspace. If the workspace has not previously been saved, you will be prompted to provide a workspace name.

7 Close

 Close

The close command brings up a sub-menu of different close options:

Command	Description
Close All	Closes all log files, filters, and directory monitors.
Close All But This	Closes all log files and filters except for the selected file.
Close Source Files	Closes all source log files. Available for merge files only.
Close All Filters	Closes all filters across all log files.
Close View	Closes the current view.

8

Close All



Close Workspace

Ctrl+Delete

Closes all log files, filters, and directory monitors.

Log Entries Grid

The diagram shows the Log Entries Grid interface with four numbered callouts:

- 1** Grouping: Points to the header area with the text "Drag a column header here to group by that column".
- 2** Auto Filtering: Points to the first data row.
- 3** Log Entries: Points to the main grid of log entries.
- 4** Log File Indicator: Points to the left margin of the grid.

	Time	Thr...	Level	Message
	15:33:01.465	3	DEBUG	Price information set a bunch of stuff. Nee...
	15:33:01.471	3	DEBUG	Invalid trade detected.
	15:33:01.484	4	ERROR	Transaction failed.
	15:33:01.485	4	INFO	Price for trade determined.
	15:33:01.493	2	INFO	Price information set a bunch of stuff. Nee...
	15:33:01.498	5	INFO	Price information set a bunch of stuff. Nee...
	15:33:01.522	2	WARN	Price information set a bunch of stuff. Nee...
	15:33:01.534	5	INFO	Price for trade determined.
	15:33:01.552	4	DEBUG	Invalid trade detected.
	15:33:01.553	5	INFO	Price information set a bunch of stuff. Nee...
	15:33:01.594	5	INFO	Price information set a bunch of stuff. Nee...
	15:33:01.608	5	WARN	Invalid trade price detecting. Trade 2994 u...
	15:33:01.616	3	INFO	Price for trade determined.

The log entry grid is the heart of LogViewPlus. You can use this grid to easily view information about your log file. Note that this grid is color-coded based on the log level. LogViewPlus supports five different log levels: Debug, Info, Warn, Error and Fatal.

1

Grouping

Drag a column header here to group by that column

The grouping box can be used to group a particular column. For example, by thread or log level. For more information, please see [Grouping Log Entries](#).

2

Auto Filtering

Time	Thr...	Level	Message

Auto filtering executes a text search in the context of the given column. Rows which do not match the search criteria will be removed from the view.

3

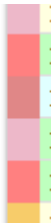
Log Entries

15:33:01.465	3	DEBUG	Price information
15:33:01.471	3	DEBUG	Invalid trade det
15:33:01.484	4	ERROR	Transaction failed
15:33:01.485	4	INFO	Price for trade d
15:33:01.493	2	INFO	Price information
15:33:01.498	5	INFO	Price information
15:33:01.522	2	WARN	Price information

The log entries list shows all log entries available on the current view.

4

Log File Indicator



This log file indicator will appear by default when viewing a merged file. Each distinct color represents a different file. This makes it easy to see at a glance which log entries were written by the same application. Of course, there is also a 'Log File Name' column where you can see the name of the file responsible for the log entry.

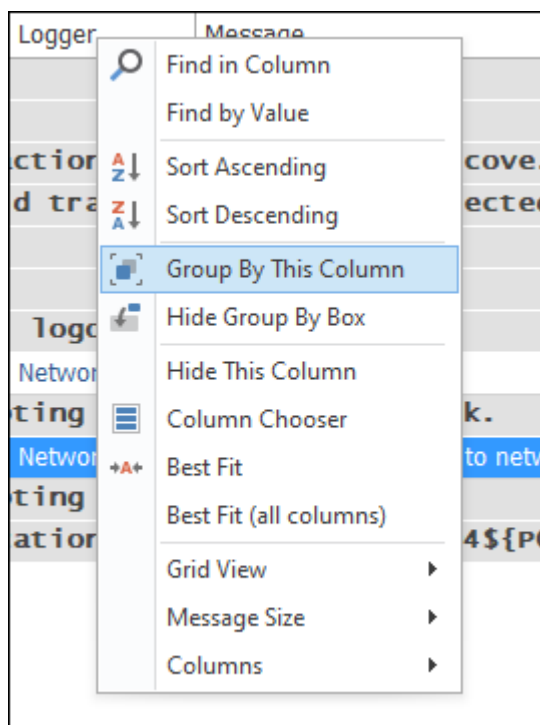
Grouping Log Entries

The screenshot shows the LogViewPlus interface with log entries grouped by level. Annotations on the left side identify key UI elements:

- 1 Group By Box:** Points to the 'Level' dropdown menu at the top left.
- 2 Group Expander:** Points to the expand/collapse arrow on the left of a group header.
- 3 Grouping Descriptions:** Points to the text within a group header (e.g., '[40502] INFO').
- 4 Log Entry:** Points to an individual log entry row.

Date	Time	Logger	Message
[40502] INFO			
[9998] ERROR			
[9508] Transaction failed. at Clearcove.Logviewer.Lc			
[490] Invalid transaction details detected. verifying			
[4889] WARN			
[4] DEBUG			
[1] client logout initiated.			
5 Apr 20...	23:11:16.356	Networks.Cli...	Client logout initiated.

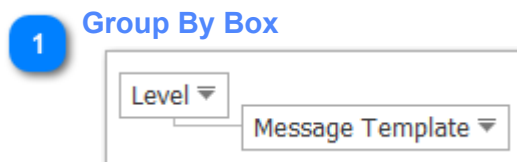
LogViewPlus gives you the ability to group log entries by column. Log entries containing common values for the selected column will be grouped together. To group log entries you can either drag a column into the "Group By" box, or right click on a column and select "Group By This Column".



Note that grouping hierarchies are supported. In the above example log entries are grouped by Level first and Message Template second. If you need to group by Message, we recommend using the Message Template column instead. The Message Template column attempts to remove any variables from the message and produce a cleaner report.

Grouping while tailing a log file is possible. However, the groups will expand as new entries are added. This will make reading the data difficult. For this reason, we recommend you disable the tail functionality while grouping.

Finally, note that grouped log entries cannot be saved. If you attempt to save a view with grouping enabled, all groups will be cleared before the file is saved.



The Group By box shows the columns that are currently grouped. You can add or remove columns to the group by box by dragging and dropping the column headers.

2

Group Expander



The arrow next to each group can be used to expand all child entries within the group. Note that a group may contain child groups or log entries.

3

Grouping Descriptions

▶	[9998] ERROR
▶	[9508] Transaction failed.
▶	[490] Invalid transaction

The grouping descriptions give you information about the current groups. This information is broken into two fields. The first field is the number of records contained within the group. This field is enclosed in square brackets. For example, "[128]" would tell us that this group contains 128 log entries.

The second field contains the value for the group. This value is the column value which will be common for all log entries in the group. In the example above, "WARN" is the log level. The two subsequent group values are common messages.

This is perhaps more clear if you consider the values in the "Group By" box while interpreting the grouping description.

The font used to display the grouping descriptions will be the same font that is used by the [log entry box](#).

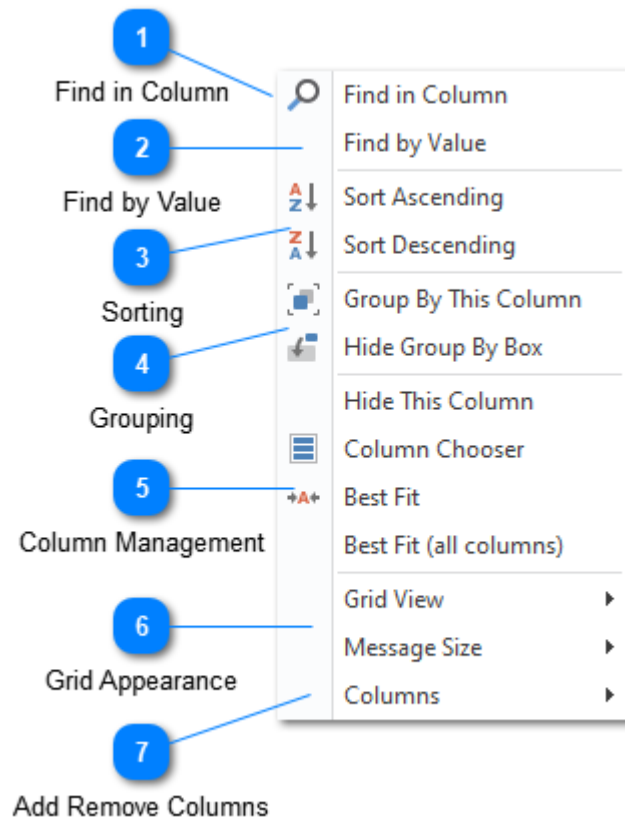
4

Log Entry

5 Apr 20...	23:11:16.356	Networks.Cli...	Client logout initiated.
-------------	--------------	-----------------	--------------------------

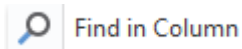
If you drill down far enough in the grouping, you will eventually come to the underlying log entries. These log entries will not display columns which are included as part of the grouping. In the example above the level and message columns are not used when displaying the log entry.

Grid Column Menu



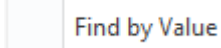
The grid menu is available by right clicking on a column header of the log entries grid.

1 Find in Column



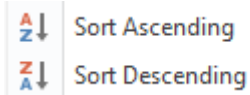
Creates a new [Text Filter](#) with the source field set to the currently selected column. This will constrain the search to only the selected column.

2 Find by Value



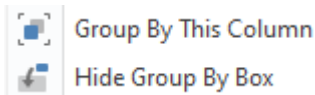
Creates a new [Value Filter](#) based on the values contained in the selected column.

3 Sorting



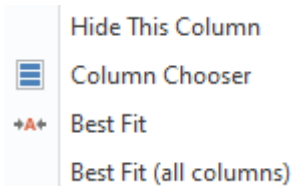
The sorting commands allow you to sort in ascending or descending order. You can also clear sorting if the selected column already has sorting applied.

4 Grouping



The grouping commands allow you to control grid grouping. You can either group by the selected column or manage whether the group by box is shown or hidden.

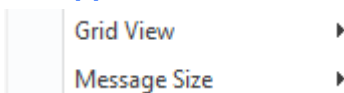
5 Column Management



The column management commands allow you to configure which columns are shown in the log entry grid.

You can use the Best Fit options to automatically manage column widths.

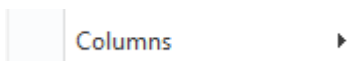
6 Grid Appearance



Grid appearance commands can be used to control the display of the log entry grid. Please see the [view toolbar](#) documentation for more information.

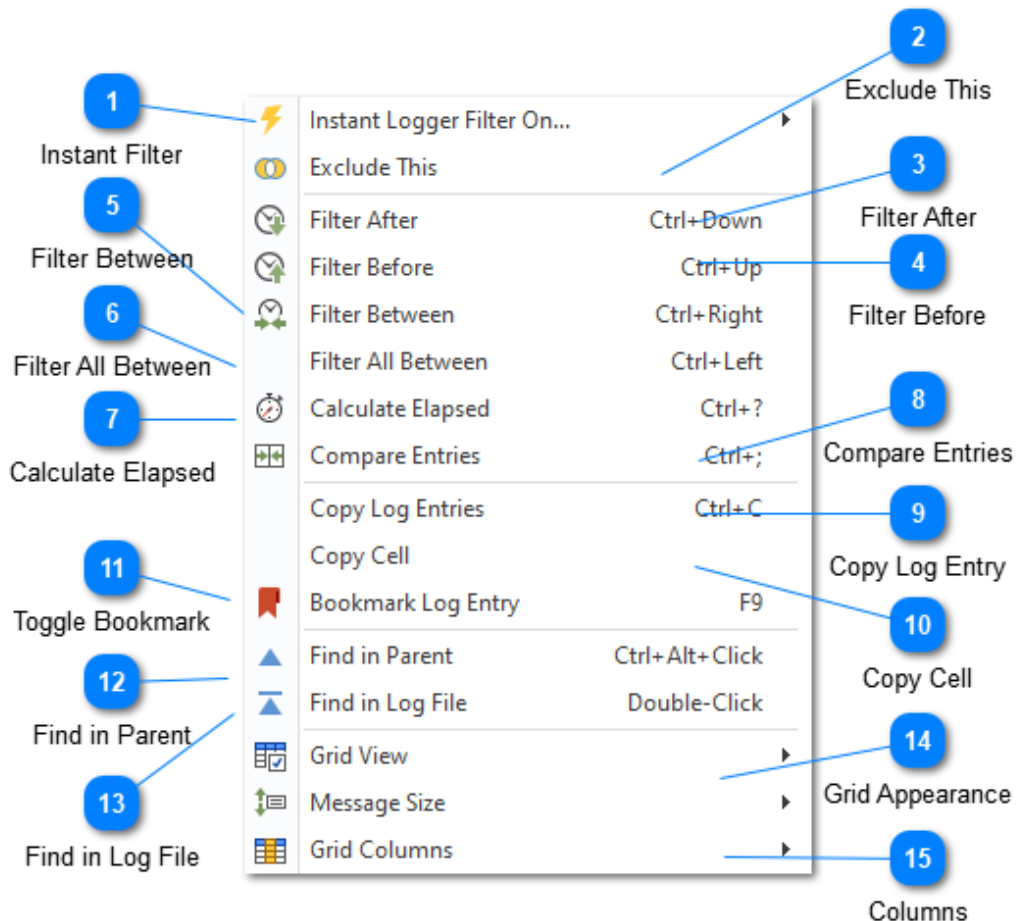
7

Add Remove Columns



The columns command can be used to quickly add or remove a column. Selecting this command will open a further set of menu items each of which represents a grid column. Currently visible columns will have a check next to them.

Grid Context Menu



The grid context menu is available by right clicking on a row in the log entries grid. Note that the actions available in the grid context menu will change depending on the column where the right-click action occurred.

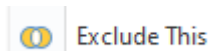
1 Instant Filter

An "instant" filter is a filter which will be created without any further configuration. In this example the type of instant filter is a thread filter. Selecting this option would

therefore create a thread filter at the level of the current log entry. For example, if the current log entry is at thread '5' then this command would create a new thread filter for thread '5'.

Note that in some cases, we may need to select where in the filter hierarchy the filter should be applied. Should we apply it to the log file? Or to the current filter? Or, perhaps, a child filter?

2 Exclude This



Exclude filters work the same as the "instant" filter described above, but the filter created will be an Exclude rather than an Include. This means the generated view will not contain the selected text in the given column.

3 Filter After



Reads the time of the currently selected log entry and creates a new Date Filter showing everything which occurred after that time.

4 Filter Before



Reads the time of the currently selected log entry and creates a new Date Filter showing everything which occurred before that time.


5 Filter Between



When two or more log entries are selected this command will determine the start and end time of the log entries and create a new Date Filter showing everything in the current view which occurred between the selected times.

6


Filter All Between

 Filter All Between Ctrl+Left

Filtering "all" between is the same as filtering between, but the "all" command will be executed against the root log file rather than the current view. Therefore, the filter generated by this command will contain all log entries in the log file between the two selected dates.

7


Calculate Elapsed

 Calculate Elapsed Ctrl+?

Calculates the amount of time that has elapsed between the two selected log entries.

8


Compare Entries

 Compare Entries Ctrl+;

If you have [configured](#) LogViewPlus to use a compare tool (like WinMerge) the Compare Log Entries command will open the two selected log entries using your configured tool. This is helpful for quickly determining the difference between two log entries.

9


Copy Log Entry

 Copy Log Entries Ctrl+C

Copies the currently selected log entry to the clipboard. Note that the copied log entry will be the full log entry as it would have appeared in a text editor.

10

Copy Cell

 Copy Cell

Copies the currently selected cell to the clipboard.

If multiple rows are selected, this command will be called "Copy Column Values". All column values in the selection will be copied to the clipboard with one value per line.

11

Toggle Bookmark



Adds or removes a bookmark to the currently selected log entry. Note you can also add or remove bookmarks by double-clicking in the row selection column.

12

Find in Parent



This command will find the currently selected log entry in the parent log file or filter.

13

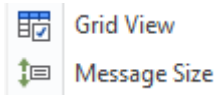
Find in Log File



This command will find the currently selected log entry in the original log file.

14

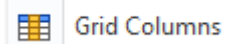
Grid Appearance



Grid appearance commands can be used to control the display of the log entry grid. Please see the [view toolbar](#) documentation for more information.

15

Columns



The columns command can be used to quickly add or remove a column. Selecting this command will open a further set of menu items each of which represents a grid column. Currently visible columns will have a check next to them.

Navigation Bar

Time	Level	Logger	Message
09:51...	INFO	MulticolumnLis...	Client 'Alice' transaction initiated.
09:51...	TRACE	LVVariant	Process start client initiated.
09:51...	TRACE	SceneGraphDis...	ESENT database transaction completed.
09:51...	INFO	KnobPanel	Network connection established.
09:51...	TRACE	DAQmxName	Executing SQL statements.
09:51...	WARN	CopyConflict	Transaction time delayed significantly. This may effect available pricing.
09:51...	ERROR	TagSet	Invalid transaction details detected. Verifying with server.
09:51...	INFO	XControlLibrary	Transaction details: <?xml version="1.0" encoding="utf-8" ?> <SERIES-AND-CLASSES-CONTRACTS-DATA> <MERGER-SERIES-AND-CLASSES-CONTRACTS>

The diagram shows a vertical navigation bar to the right of the log table. It features a color-coded bar representing log levels (red for error, orange for warn, yellow for info, green for trace). Numbered callouts point to specific features:

- 1 Quick Scroll**: Points to the top of the navigation bar.
- 2 Bookmark Indicator**: Points to a small square icon on the bar.
- 3 Log Level Indicator**: Points to the colored bar itself.
- 4 Selection Indicator**: Points to a small green square at the bottom of the bar.
- 5 Auto-scroll Indicator**: Points to a small green circle at the bottom of the bar.

The log level navigation bar is found to the right of the log entries grid. This navigation bar is designed to give you a quick view of your log file.

The idea behind the log level navigation bar is that it shows 100% of your log file. If, for example, your log file had 100 log entries and the log level navigation bar had 100 pixels, then one pixel on the navigation bar could represent one log entry. As your log file grows beyond 100 log entries, the log level navigation bar will fold log entries based on log priority.

For example, if your log file had 1000 log entries, then a pixel would represent 10 log entries. If one of those 10 log entries was an error, then the pixel would be red.

Clicking anywhere on the log level navigation bar will take you to the corresponding position in the log file. Right clicking on the log level navigation bar will display the navigation bar [context menu](#).

Note that selecting the bottom of the navigation bar will automatically select the last log entry in the view. This will enable auto-scroll behavior if the log file is currently in tail mode.

1 Quick Scroll



The gray arrow found at the top and bottom of the log level navigation bar can be used to quickly scroll to the first or last log entry in your log file. Clicking on the gray arrow found at the bottom of the navigation bar scrolls to the last log entry and enables auto scroll. Auto scroll is disabled by selecting any other log entry in the view.

2

Bookmark Indicator



The bookmark indicator is used to show the location of bookmarked log entries. Clicking on a bookmarked indicator will select the bookmarked log entry.

3

Log Level Indicator



The log level navigation bar is color coded based on log entry priority. Log entries with the priority less than warning will not be displayed. Fatal and error log entries will be displayed as red. Warnings are displayed as orange.

4

Selection Indicator



The green arrow in the log level navigation bar is used to indicate the location of the currently selected log entry.

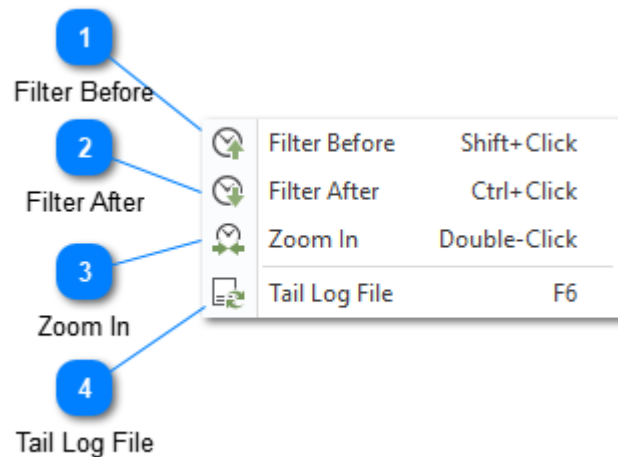
5

Auto-scroll Indicator



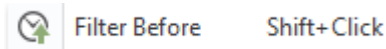
If auto-scroll is currently enabled for the log file, the bottom quick scroll icon will be replaced with a green auto-scroll icon. Auto-scroll can be disabled by selecting any log entry in the view other than the last log entry.

Navigation Bar Menu



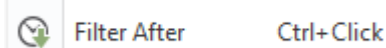
Right clicking on the log level [navigation bar](#) will display the navigation bar context menu.

1 Filter Before



Creates a date time filter which displays everything which occurred before the selected point.

2 Filter After



Creates a date time filter which displays everything which occurred after the selected point.

3 Zoom In



Zooms in on the currently selected point. In order to zoom in LogViewPlus will create a date time filter encompassing roughly 10% of the current view with the selected point in the middle of that calculated time range.

4

Tail Log File

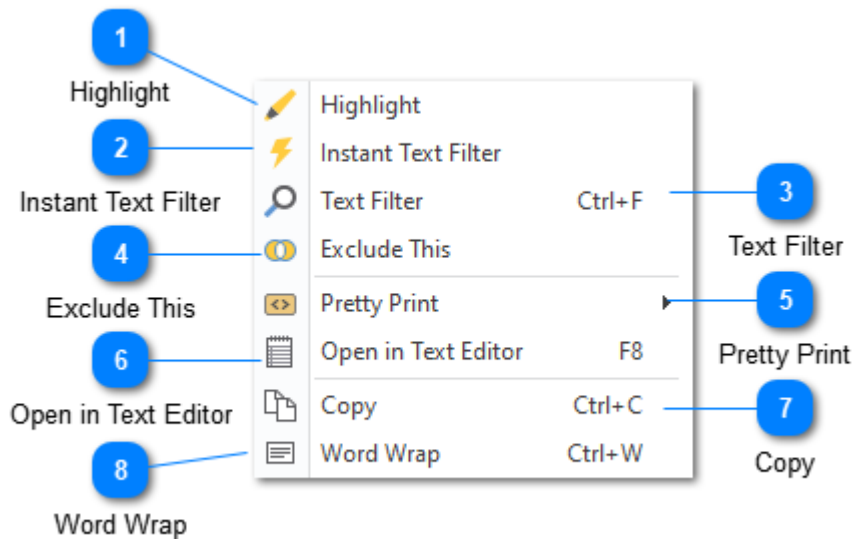


Tail Log File

F6

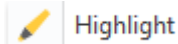
The tail log file command allows you to start or stop monitoring the current file for changes. You can also enable auto-scroll from the navigation bar by selecting the bottom of the bar. Auto-scroll can only be enabled if the log file is currently in tail mode.

Log Entry Menu



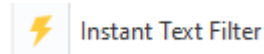
When you right-click on the log entry box which is located just below the log entry grid you will be presented with the log entry menu shown above.

1 Highlight



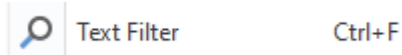
The highlight command can be used to [create a highlight](#) from the currently selected text.

2 Instant Text Filter



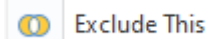
The instant text filter will immediately create a text filter from the currently selected text. Note that you will not be given an opportunity to configure the text filter the forward is created.

3 Text Filter



The text filter command will create a new text filter from the currently selected text. This command will give you the opportunity to configure the text filter before it is created.

4 Exclude This



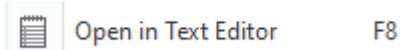
Exclude filters work the same as the "instant" filter described above, but the filter created will be an Exclude rather than an Include. This means the generated view will not contain the selected text in the given column.

5 Pretty Print



The pretty print command can be used for pretty printing and syntax highlighting of the log entry. Currently LogViewPlus supports three types of pretty printing: XML, JSON, and Symbols & Numbers. Alternatively you can also use this command to turn off pretty printing and syntax highlighting for the currently selected log entry.

6 Open in Text Editor



This command will save the current log entry to a file and open it in your [default text editor](#). The file extension for the new file will be determined based on the pretty print type. This command is useful when viewing very large, individual log entries.

For example, a log entry with XML output, or a log entry containing all system environment variables.

If multiple log lines are selected in the [Log Entries Grid](#), then multiple lines will be displayed in the log entry text area. In this case, all log entries will be saved to the new file.

7

Copy



Copy

Ctrl+C

The copy command copies the currently selected text to the Windows clipboard.

8

Word Wrap



Word Wrap

Ctrl+W

The word wrap command can be used to turn word wrap on or off. Turning off word wrap can be particularly helpful if the current log entry is formatted by default.

Notifications

Notifications can be used to create an alert when a new log entry is detected in a filter. This allows you to passively monitor a filter. For example, you could filter a log file for errors and be notified when a new error is detected.

Creating a notification is a two-step process. First you need to create a notification provider. [Notification providers](#) are used to define how you would like to be notified. Then, you need to [add a notification](#) to a filter. This will tell LogViewPlus when you would like to be notified.

Once a notification is added to a filter it will be activated when a new log entry is added to the filter.

Currently, LogViewPlus supports [popup notification](#) and [SMTP notification](#) providers.

To see event notifications in action, check out our [event notifications](#) quick video example.

Popup Provider

The screenshot shows a dialog box titled "Add Popup Alert Provider" with a close button (X) in the top right corner. Inside the dialog, there is a descriptive text: "A popup alerts is a small notification window. Popup alerts may be transient and do not require user interaction. Popup alerts are the simplest form of notification." Below this text are several input fields and controls:

- 1 Name:** A text input field labeled "Alert Name:".
- 2 Screen Location:** A dropdown menu labeled "Alert Screen Location:" with "TopLeft" selected.
- 3 Require Acknowledgement:** A checkbox labeled "Require Acknowledgement" which is currently unchecked.
- 4 Display Time:** A numeric input field labeled "Display Time (in seconds):" with the value "15" and up/down arrows.
- 5 Apply Changes:** A button labeled "Save" (highlighted with a blue border) located below the "Test", "Save", and "Cancel" buttons.

There is also a link "Find out more about popup notifications" and buttons "Test", "Save", and "Cancel" at the bottom of the dialog.

The pop-up notification provider configuration screen is used to create a new pop-up alert. Use this configuration to define how you would like a pop-up alert to be displayed.

1 Name

Alert Name:

The alert name is a friendly name used to identify this alert provider configuration. This name will be used when selecting a provider in the [add notification](#) dialog.

2 Screen Location

Alert Screen Location:

Alert screen location determines where a pop-up alert appears in your monitor. Currently, LogViewPlus supports the four corners of your monitor. These are defined as: TopLeft, BottomLeft, TopRigth, BottomRight. Note that LogViewPlus does not need to be visible or active for a pop-up to be displayed.

3 Require Acknowledgement

☐ Require Acknowledgement

If a pop-up alert requires acknowledgment it will not be closed automatically. This ensures that notifications are seen by the user.

4

Display Time

Display Time (in seconds):

If a pop-up alert does not require acknowledgment, then it will only be displayed for a short period of time before being closed automatically. The displayed time allows you to configure the amount of time that the alert pop-up should be displayed.

Note that alerts configured with a display time may not be seen by the user.

5

Apply Changes

Once you are happy with your pop-up alert configuration you can press the 'OK' button to save the configuration. Alternatively the 'Cancel' command can be used to return to the LogViewPlus application settings.

We recommend that you test the alert provider before saving your changes.

SMTP Provider

The screenshot shows a dialog box titled "Add SMTP Alert Provider" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- 1 Friendly Name:** A text input field.
- 2 Server Details:** A section containing:
 - SMTP Server Name:** A text input field.
 - Port Number:** A spinner box set to 25, followed by a checked checkbox labeled "Secure Connection with SSL".
 - User Name:** A text input field.
 - Password:** A text input field.
- 3 From Address:** A text input field.
- 4 To Address:** A text input field.
- 5 Max Emails:** A text input field set to 10, with a small up/down arrow.

Below the fields is a link: [Find out more about SMTP notifications](#).

At the bottom are three buttons: "Test" (labeled **6**), "Save" (labeled **7**), and "Cancel".

The SMTP notification provider configuration screen is used to create a new SMTP alert. To create an SMTP alert you must have permission to use an external SMTP server.

1 Friendly Name

Friendly Name:

The friendly name is used when referencing this SMTP provider alert. This name will be used when selecting a provider in the [add notification](#) dialog.

2 Server Details

SMTP Server Name:

Port Number: ☒ Secure Connection with SSL

User Name:

Password:

The server details configuration is where you will provide all details necessary to establish a connection with your SMTP server.

3

From Address

From Email Address:

The From Address will be used to mock the source of the alert email. Note that, just like any other email, you will be able to reply to the SMTP alerts. We therefore recommend using a valid From address. However, LogViewPlus does not ensure that the From address is valid.

4

To Address

To Email Addresses:

The To Address specifies where the alert should be sent. This field is required.

5

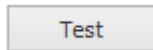
Max Emails

Max Emails Per Hour:

When setting up an SMTP provider alert it's important to keep in mind that sometimes log events happen more frequently than we initially anticipated.

The max emails per hour field allows you to specify a threshold beyond which no more emails will be sent. This is a useful feature to ensure that you do not accidentally spam the target recipient. Note however that if the same SMTP provider is used to monitor multiple filters then these filters will share a common threshold.

6 Test



The test command can be used to send an SMTP email message using the configuration provided. We always recommend testing your configuration before setting up your alerts.

7 Apply Changes



Once you are happy with your SMTP alert configuration you can press the 'OK' button to save the configuration. Alternatively the 'Cancel' command can be used to return to the LogViewPlus application settings.

Add Notification

The 'Add Notification' dialog box contains the following fields and controls:

- 1** Notification Provider: A dropdown menu showing 'Default Popup Alert'.
- 2** Notification Frequency: A dropdown menu showing '5 Seconds'.
- 3** Message Subject: A text field containing the placeholder `\${FILTER_NAME} - \${LOG_FILE_NAME}`.
- 4** Message Detail: A text area containing the placeholder `\${LOG_ENTRY_FULL}`.
- 5** Apply: A button at the bottom right of the dialog.

Additional text in the dialog includes a warning: 'A notification will be triggered when any log entry is added to the filter. To prevent multiple notifications in the event of multiple matching log entries you should set the notification frequency.' and a link: 'Find out more about notifications.'

You can add a notification to a filter by clicking on the [notify command](#) in the log entries toolbar. The notify command will bring up the add notification dialogue which allows you to configure the notification.

1

Notification Provider

Notification Provider:

The notification provider tells LogViewPlus how a notification should be delivered. Notification providers are configured in [application settings](#).

2

Notification Frequency

Notification Frequency:

Notification frequency determines how often a notification should be activated. This is important to consider if you are adding a notification to a filter which may be triggered repeatedly over a short time.

A notification can only be fired once within the notification frequency time window.

3

Message Subject

Message Subject:

`${FILTER_NAME} - ${LOG_FILE_NAME}`

The subject that should be displayed by the notification provider. The subject should be a short message. Subjects can be parameterized and using [argument templates](#).

4

Message Detail

Message Detail:

`${LOG_ENTRY_FULL}`

The message that should be displayed by the notification provider. Note that the message can be parameterized using [argument templates](#).

5

Apply

Apply

Once you are happy with your notification configuration you can click Apply to add the notification to the filter.

If a filter has a notification it will be shown with a notify icon next to it. Double-clicking on the notify icon will bring up the Edit Notification dialog. This dialog will be the same as the Add Notification dialogue discussed above with the exception that it also includes a Delete command at the bottom of the view. The Delete command can be used to remove the notification from the filter.

Command Line

LogViewPlus supports the following command line options. This help is also available by executing **logviewplus.exe /?** at the command line. Note that LogViewPlus is not placed into your system PATH by default. Therefore it is necessary to navigate to the installation directory before executing the commands. The default installation directory is C:\Program Files\LogViewPlus.

Usage: LogViewPlus [-?] [-lastworkspace] [-workspace:name] file1path file2path

Options:

-? Displays the command line usage.

-lastworkspace Opens the last viewed workspace if available. Note that LogViewPlus automatically saves your current workspace every 15 seconds.

-workspace:name Opens the workspace specified by the 'name' variable if available. If the workspace does not exist, the command will be ignored.

Examples:

LogViewPlus file1path file2path - Opens file1path and file2path. Note that full file paths are expected.

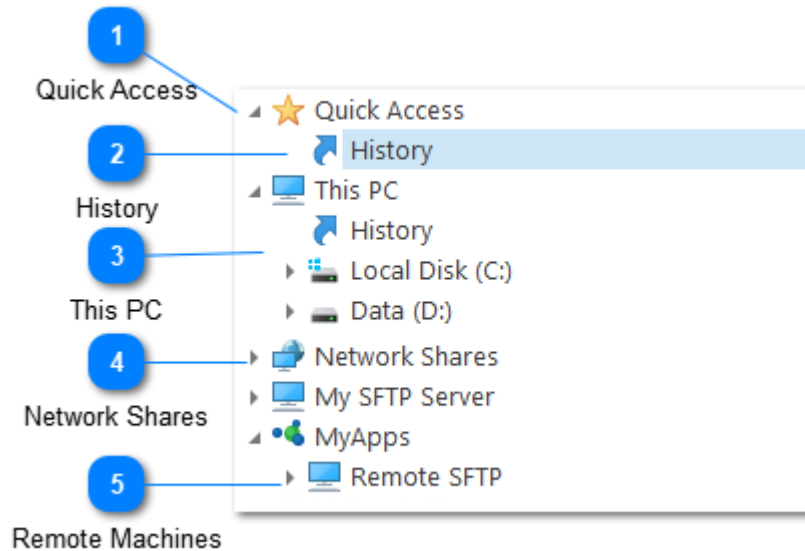
LogViewPlus "-workspace:My Worksapce" file1path - Opens the workspace named 'My Worksapce' as well as file1.

File Systems

Unlike most applications, LogViewPlus uses bespoke file system access. This allows you to quickly and easily browse your local file system as well as remote filesystems via SFTP or FTP.

Managing your filesystems is discussed in detail in the next section. If you have not already used the LogViewPlus file system to open a log file, we recommend you take a [First Look](#) to get a high level overview before continuing.

Folder Tree View



The folder tree view gives quick access to all available filesystems. Note that filesystems can be local or remote via SFTP or FTP.

1

Quick Access

★ Quick Access

The quick access node is always the first node available in the LogViewPlus file browser. This node contains shortcuts and quick links to help you navigate to frequently accessed directories and files.

To add the directory to the quick access list right-click on a directory and select "Add to Quick Access" as discussed in the [file context menu](#) documentation.

When removing an item from the quick access list, you do so with the "Delete" command. This command will remove the quick access item, but will have no effect on the underlying file system.

2

History

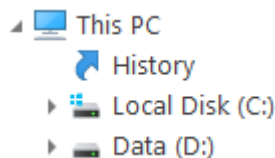
History

History links are special nodes within the directory browser. They contain a list of your file access history. History links work at two levels. Quick access history contains the last 20 items you opened regardless of where those items point to. Alternatively, computer history links contain the last 20 items you have opened on that particular computer or drive.

Please see the [history](#) node documentation for more information.

3

This PC

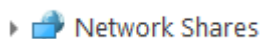


The "This PC" node contains a list of all of the drives which are available on your local machine. Browsing these drives should be very similar to using Windows Explorer.

Note that there is a separate history note for your local file system. This history node will only contain recently accessed items on your local machine.

4

Network Shares



The network shares node contains a list of all available Windows or SMB shares.

Note that Network Shares will always be unpopulated when you first install LogViewPlus. This is because some networks may contain a large number of machines and quickly scanning the network to find the machine you want to access may not be possible. Instead, LogViewPlus uses an approach that requires you to know the name of the machine you want to connect to in advance. You can either explicitly add this machine name in the [file system settings](#), or you can paste the full path to the file you want to open into the open file text box. Once a local network file has been opened by LogViewPlus, the network node will be available for future access.

5

Remote Machines

▶ Remote SFTP

Once a remote SFTP or FTP server has been added to the LogViewPlus file system configuration settings, you will be able to browse the remote server here.

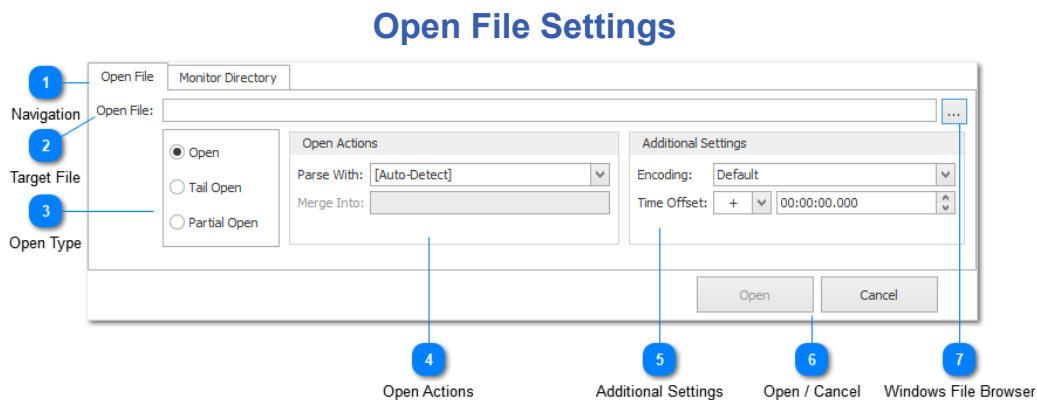
Note that the example above shows a server named "Remote SFTP" with a friendly category of "MyApps".

If you have any difficulty with your remote machine connection, you may find it helpful to view the file transfer log file located at %Temp%\LogViewPlus.

File List View

Name	Date Modified	Size	Description
OtherFormats	27/02/2016 11:52		File folder
basic.log	09/04/2017 17:15	1 KB	Application Log File
pattern_dates.log	09/04/2017 17:14	1 KB	Application Log File
rename.vbs	27/02/2016 11:52	1 KB	VBScript Script File
Small.zip	19/10/2016 06:53	182 KB	Compressed (zipped) Folder
SVR_0.Alice.Client.2016-02-18 - ...	12/01/2017 06:43	2,553 KB	Application Log File
SVR_0.Alice.Client.2016-02-18.log	12/01/2017 06:43	2,553 KB	Application Log File
SVR_0.Bill.Client.2016-02-18 .ext...	18/02/2016 19:24	2,553 KB	Application Log File
SVR_0.Bill.Client.2016-02-18.log	18/02/2016 19:24	2,553 KB	Application Log File
SVR_0.Dana.Client.2016-02-18.I...	18/02/2016 19:27	2,550 KB	Application Log File
SVR_0.Eddie.Client.2016-02-18.I...	18/02/2016 19:28	2,556 KB	Application Log File
SVR_0.Frank.Client.2016-02-18.I...	18/02/2016 19:30	2,557 KB	Application Log File
SVR_0.George.Client.2016-02-1...	18/02/2016 19:32	2,558 KB	Application Log File
SVR_0.Harry.Client.2016-02-18.I...	18/02/2016 19:33	2,558 KB	Application Log File
SVR_0.Irene.Client.2016-02-18.I...	18/02/2016 19:34	2,559 KB	Application Log File
SVR_0.Jerry.Client.2016-02-18.log	18/02/2016 19:36	2,559 KB	Application Log File
SVR_0.Kim.Client.2016-02-18.log	18/02/2016 19:37	2,554 KB	Application Log File
SVR_0.Louise.Client.2016-02-18...	18/02/2016 19:39	2,561 KB	Application Log File
SVR_0.Matt.Client.2016-02-18.log	18/02/2016 19:40	2,556 KB	Application Log File

The file list view is located to the right of the folder tree view. This control shows a list of all of the files and folders that are contained within the selected folder. Selecting one or more files will populate the [open file settings](#) automatically. You can also right-click on the file or folder to open the [file context menu](#).



At the bottom of the LogViewPlus file browser are a number of settings which provide flexibility in how log files are opened.

1 Navigation



The tabs open file settings panel were used to determine whether you are opening a log file or directory. The [transfer log](#) tab shows remote server connection logs.

2 Target File

Open File:

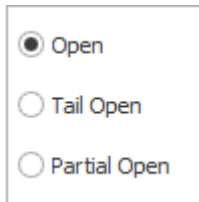
The open file text box should be automatically populated based on your file selection in the files list view. Alternatively you can paste the full path to the file you want to open.

If you enter the full file path to open a file which exists on a network share LogViewPlus will automatically add the network share to its configuration settings. From that point on the server containing your log file will be visible in your Network Shares.

LogViewPlus also supports SCP. SCP does not support directory browsing, so to open a file via SCP you will need the full URL to the file. LogViewPlus SCP URLs are always prefixed with "scp://". For example, scp://server:port/path/file.log.

3

Open Type

A dialog box titled "Open Type" with three radio button options: "Open" (selected), "Tail Open", and "Partial Open".

☒ Open

☐ Tail Open

☐ Partial Open

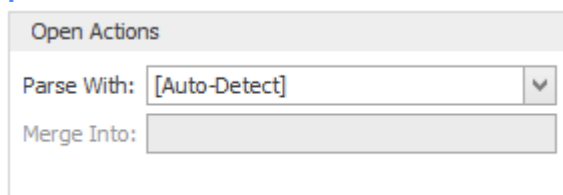
LogViewPlus provides three ways to open a log file. To simply "Open" the log file means that you will read the entire log file from start to finish.

If you are only interested in new log file entries, you may prefer to "Tail Open" the log file. This option will read a little bit from the end of the log file and then update LogViewPlus with any additional log entries.

For very [large log files](#), you may only be interested in a part of the log file. For example, if you have a 1 GB log file and are interested in a problem that occurred an hour ago, you may want to try opening the file at the 700 MB to 800 MB range. In LogViewPlus this is called a "Partial Open".

4

Open Actions

A dialog box titled "Open Actions" with two fields: "Parse With:" with a dropdown menu showing "[Auto-Detect]" and a downward arrow, and "Merge Into:" with an empty text box.

Open Actions

Parse With: [Auto-Detect] ▼

Merge Into:

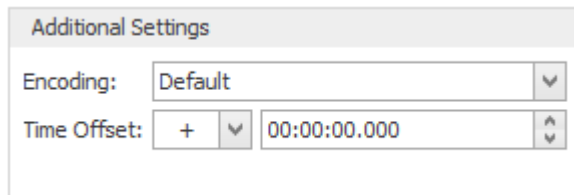
The Open Actions settings can be used to determine how the selected log files should be processed.

By default, LogViewPlus will parse the file according to the rules defined by [pattern matching the log file name](#). If a parser is manually selected the target parser will be used and the default parser will be ignored.

If more than one log file is selected the Merge Into option will be enabled. This option allows you to easily merge the selected log files without having to perform a separate action after the files are loaded.

5

Additional Settings



Regardless of how you choose to open a log file in LogViewPlus, you will be able to set two additional settings - the file encoding and a time offset.

By default LogViewPlus will use the Windows configured file encoding. This changes depending on your locale and you may need to choose a different option depending on how your log files are written. The encoding you use to read your log files should be the same encoding you use to write your log files.

The time offset can be used to modify all timestamps in the currently selected log file by a user configured amount. This command is useful when you intend to merge multiple log files but the timestamps of the log files are not in sync. For example, if you need to merge multiple log files which are recorded in local time in different time zones. The command is also useful when log files recorded on different machines are a millisecond or two out of sync. A time offset can either add or subtract time depending on if you need to increase or decrease the time interval.

6

Open / Cancel



Once you have configured your open settings, you can click the "Open" command at the bottom of the screen. Use the "Cancel" command to return to LogViewPlus.

7

Windows File Browser



LogViewPlus gives you the ability to select log files using the Windows file browser. This should be unnecessary as the LogViewPlus file browser should have access to all of your local files. Therefore this ability is only included for redundancy.

Tail Open

1 Tail Open

2 Tail Options

Opens a file and tail mode. When in tail mode only the last few entries of the log file will be read. The number of entries will be expanded as new log entries are written.

When tail opening a file, there are additional settings options which are not discussed here. For more information, please see [Open File Settings](#).

1 Tail Open

To tail open a log file, you must first set the Tail Open option.

2 Tail Options

When tail opening a log file, you need to specify how much data you want to read from the end of the log file. Regardless of how much data you read from the end of the log file, LogViewPlus will read new log entries as they are written.

Partial Open

The diagram shows a settings window for 'Partial Open'. On the left, there are three radio buttons: 'Open', 'Tail Open', and 'Partial Open'. The 'Partial Open' option is selected. A blue circle with the number '1' is placed below this group, with a line pointing to the 'Partial Open' radio button. To the right of the radio buttons is a section titled 'Partial Open Options'. It contains two dropdown menus: 'Chunk Size' set to '10 MB' and 'Open Part' set to 'Part 3 (bytes > 2...'. A blue circle with the number '2' is placed below this section, with a line pointing to the 'Open Part' dropdown.

Partial Open

Partial Open Options

Partial open is useful when dealing with large files. It allows you to open a chunk of the file without having to process all log entries.

When partial opening a file, there are additional settings options which are not discussed here. For more information, please see [Open File Settings](#).

1 Partial Open

A close-up of the radio button selection area. It shows three options: 'Open', 'Tail Open', and 'Partial Open'. The 'Partial Open' option is selected, indicated by a filled circle next to it.

To partial open a log file, you must first set the Partial Open option.

2 Partial Open Options

A close-up of the 'Partial Open Options' section. It shows two dropdown menus. The first is 'Chunk Size' with '10 MB' selected. The second is 'Open Part' with 'Part 3 (bytes > 2...' selected.

When opening a part of the file, there are two settings you need to specify. The first is the chunk size, this determines how much data you will be reading from the log file. The second is the part of the log file you are interested in opening.

For example, a 1 GB log file with the chunk size of 100 MB will have 10 parts. The same log file with a 50 MB chunk size will have 20 parts.

Note that once you have opened a part of a log file, you may then choose to open another part of the log file. Log file parts can be merged just like normal log files.

Monitor Directory

Open File	Monitor Directory	
Directory:	<input type="text" value="file:///D:/Data/Logs/Text/Large/"/>	<input checked="" type="checkbox"/> Recurse?
File Name Pattern:	<input type="text" value="*.log"/>	<input type="checkbox"/> Is RegEx?
Ignore Files:	<input type="text" value="Older Than Today"/>	...
Merge Name (Optional):	<input type="text"/>	
Category Name (Optional):	<input type="text"/>	
		<div>Match Results</div> <div>You must validate the configuration before opening ---></div> <div><input checked="" type="button" value="Validate"/></div>

Occasionally, you might find that you need to monitor an entire directory for new log files. This is particularly helpful in situations where you have rolling log files and the names of new log files are automatically generated. In these situations it's helpful to have a directory monitor which can poll a target directory and automatically open new files as needed.

For detailed instructions in how to configure a new directory monitor, please see the [Edit Directory Monitor](#) documentation.

Open Database

The screenshot shows the 'Open Database' dialog box with the following components:

- 1 Target Connection:** A text field labeled 'Connection:' containing 'Logs x'.
- 2 Open Action:** A group box containing four radio buttons: 'Tail Recent' (selected), 'Tail Open', 'Open Range', and 'Open Point'.
- 3 Open Settings:** A 'Show recent entries:' spinner set to '1000' and a 'Condition (Optional):' text field.
- 4 Execution Plan:** A large text area with the message 'You must validate the configuration before opening ---->' and a 'Planned SQL' tab.
- 5 Validation:** A 'Validate' button with a checkmark icon.

When you select a database in the LogViewPlus File Explorer, you will be presented with the configuration options shown above. These settings allow you to customize the SQL to be executed. You must test the proposed SQL before you will be able to retrieve the data.

1 Target Connection

Connection: Logs x

The target connection contains a reference to the connection string selected in the File Explorer.

2 Open Action

The image shows a close-up of the 'Open Action' group box with the following options:

- ☒ Tail Recent
- ☐ Tail Open
- ☐ Open Range
- ☐ Open Point

LogViewPlus supports a number of different database open actions. Note that, regardless of the open action type, new log entries will only be retrieved if the database connection has been configured to allow tail.

1. **Tail Recent** - allows you to retrieve the most recent log entries. The number of entries retrieved is defined by the user.
2. **Tail Open** - allows you to open all log entries after a user defined point in time.

3. **Open Point** - allows you to open all log entries between a user defined date range.

3 Open Settings

Show recent entries:

Condition (Optional):

The Open Settings will change based on the action type selected (see above).

The Condition field is always available regardless of the open action type. This field allows you to manually enter a conditional which will be included in the SQL to be executed. For example, you might restrict your dataset to only records where ColumnName = '%value1%'. Conditions provided must be valid SQL conditional for the table defined in the connection setup. Multiple conditionals can be provided if you also include the operation (AND / OR).

4 Execution Plan

Match Results

Planned SQL

You must validate the configuration before opening --->

After validation, the match results tab will show the top 5 results returned from your query as well as a count with the number of elements LogViewPlus expects to load. This is just to give you an indication of what LogViewPlus is about to execute.

The Planned SQL tab shows the SQL that will be executed against the database should you decided to open the full results.

5 Validation

✓

Validate

Once you're happy with your database query configuration you can click the validate button. This will verify that the SQL to be executed is valid and any errors will be reported to the user.

You must validate your query to enable full execution. LogViewPlus will disable opening the dataset until the query is validated.

Note that if you database query configuration you will need to revalidate.





Transfer Log

Open File	Monitor Directory	Transfer Log
<pre>06:47:10.392 sftp://localhost:22 No connection could be made because the target machine actively refused it. 06:47:10.392 sftp://localhost:22 Unable to connect to server at sftp://localhost:22. 06:47:09.344 sftp://localhost:22 Connecting... 06:47:09.346 sftp://localhost:22 Connecting to localhost:22 using sftp. 06:47:09.346 sftp://localhost:22 Assembly: Rebex.Sftp 2017 R2 for .NET 4.0-4.6 06:47:10.392 sftp://localhost:22 No connection could be made because the target machine actively refused it. 06:47:10.392 sftp://localhost:22 Unable to connect to server at sftp://localhost:22.</pre>		

The transfer log shows a running log of all remote file access attempts. This log will only be visible if the currently selected directory is a remote directory. This log can be very helpful when trying to diagnose connection issues.

A more detailed version of the transfer log is also written to disk. By default this log is located at %Temp%\LogViewPlus.

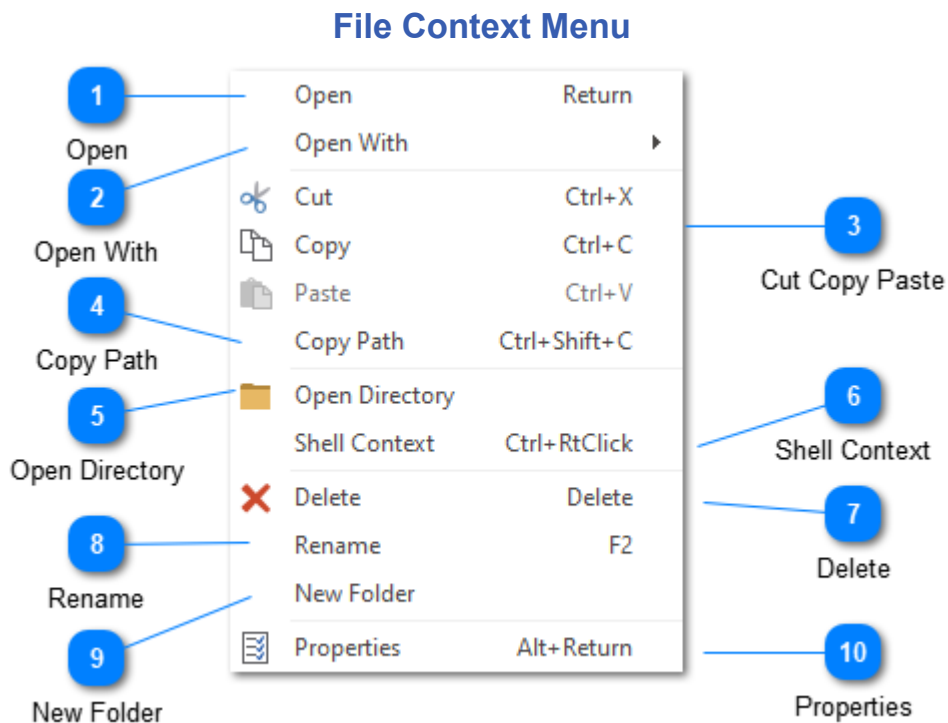
History

Name	Location	Last Opened
Recent Directories		
 Small	D:\Data\Logs\Text\Small\	05/06/2017 05:47
 Small	sftp://localhost:22/Small/	04/06/2017 12:30
Recent Files		
 SVR_0.Bill.Client.2016-02-18.log	D:\Data\Logs\Text\Small\SVR_0.Bill.Client.2016-02-18.log	05/06/2017 05:47
 SVR_0.Alice.Client.2016-02-18.log	sftp://localhost:22/Small/SVR_0.Alice.Client.2016-02-18.log	04/06/2017 12:30

When you click on a history node in the [folder tree view](#), the [file list view](#) will automatically switch to history mode. When in history mode, the file list view will scan all of the recently accessed log files on the selected computer. For each log file LogViewPlus will extract the directory which is then displayed in the recent directories category. Log files will be displayed in the recent files category. The location column is used to show the full path to the target log file or directory.

When managing your local history LogViewPlus can remember up to 20 files for each computer you access. In addition to this LogViewPlus keeps a global history of the 20 most recently accessed files across all computers. You can configure the number of recent history items stored in the [recent history](#) settings.

The global history will be displayed in the quick access history node. History for each computer will be displayed in the specific computer's history node.



The file context menu is displayed whenever you right click on a file in the LogViewPlus file browser. Note that some actions may be disabled or hidden when accessing a remote file system.

1 Open

Open Return

Opens the currently selected file in LogViewPlus. This action is the same as double-clicking on the file.


2 Open With

Open With

The Open With command can be used to open a file in another application. The alternative application can either be [configured in advance](#) or added at the time the

action is taken. If the file exists on a remote machine it will be downloaded before being opened by the target application.

3 Cut Copy Paste

	Cut	Ctrl+X
	Copy	Ctrl+C
	Paste	Ctrl+V

The cut copy and paste commands can be used to copy or move files. These commands are only available when working with the local file system.

4 Copy Path

	Copy Path	Ctrl+Shift+C
---	-----------	--------------


The copy path command can be used to copy the full path to the target file or directory. When working with remote files the full URL to the target will be copied.

5 Open Directory

	Open Directory
---	----------------

Opens the currently selected directory in Windows Explorer. This command is only available when working with local file systems.

6 Shell Context

	Shell Context	Ctrl+RtClick
---	---------------	--------------

The shell context command can be used to display the Windows Explorer context menu for the selected item. Note that if you select "Open" from the shell context menu LogViewPlus will interpret this to mean that you want to open the file in LogViewPlus as opposed to the Windows configured default application.

7 Delete

	Delete	Delete
---	--------	--------

The delete command can be used to remove the selected file or directory. When working with local file systems deleted items will be moved to your recycle bin. When working with remote file systems deletion is permanent.

8

Rename

Rename

F2

The rename command can be used to rename the selected file or folder.

9

New Folder

New Folder

The new folder command creates a new folder in the target directory.

10

Properties

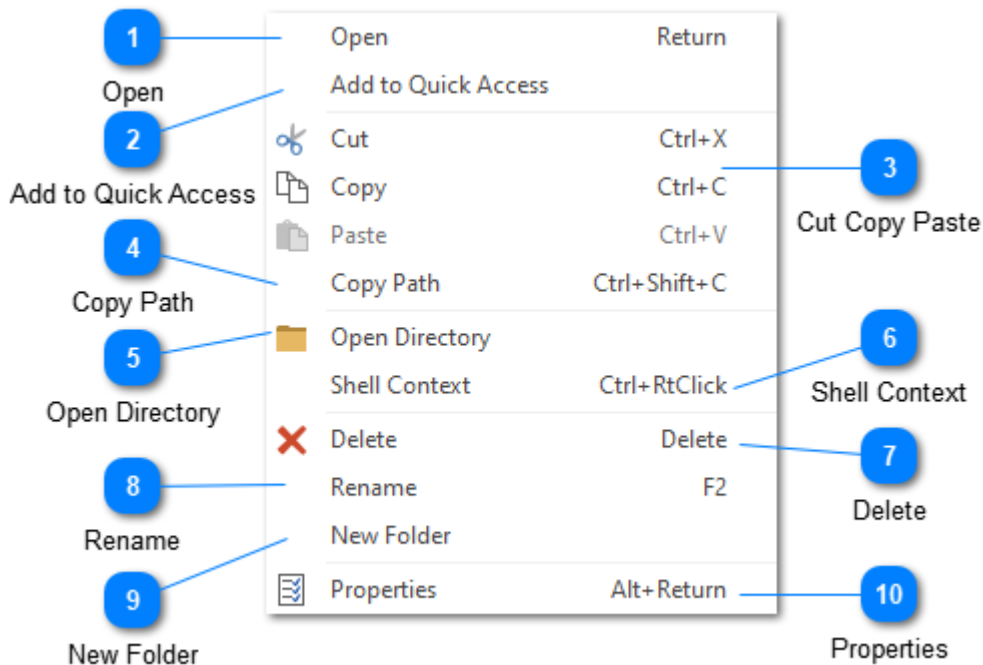


Properties

Alt+Return

The properties command will display the Windows Shell file properties for the selected file or directory. This command is currently disabled for remote file systems.

Folder Context Menu



The folder context menu is displayed whenever you right click on a folder in the LogViewPlus file browser. Note that some actions may be disabled or hidden when accessing a remote file system.

1 Open

Open Return

Opens the currently selected folder in the browser. This action is the same as clicking or on the folder.


2 Add to Quick Access

Add to Quick Access

Adding a folder to the quick access menu is useful when you frequently access the folder and you want that folder to be available regardless of your recent history

settings. Note that removing a folder from the quick access list has no impact on the underlying file system.

3 Cut Copy Paste

	Cut	Ctrl+X
	Copy	Ctrl+C
	Paste	Ctrl+V

The cut copy and paste commands can be used to copy or move folders. These commands are only available when working with the local file system.

4 Copy Path

	Copy Path	Ctrl+Shift+C
---	-----------	--------------

The copy path command can be used to copy the full path to the target directory. When working with remote file systems the full URL to the target will be copied.

5 Open Directory

	Open Directory
---	----------------

Opens the currently selected directory in Windows Explorer. This command is only available when working with local file systems.

6 Shell Context

	Shell Context	Ctrl+RtClick
---	---------------	--------------

The shell context command can be used to display the Windows Explorer context menu for the selected item. Note that if you select "Open" from the shell context menu LogViewPlus will interpret this to mean that you want to open the file in LogViewPlus as opposed to the Windows configured default application.

7 Delete

	Delete	Delete
---	--------	--------

The delete command can be used to remove the selected directory. When working with local file systems deleted items will be moved to your recycle bin. When working with remote file systems deletion is permanent.

8

Rename

Rename

F2

The rename command can be used to rename the selected folder.

9

New Folder

New Folder

The new folder command creates a new folder in the target directory.

10

Properties

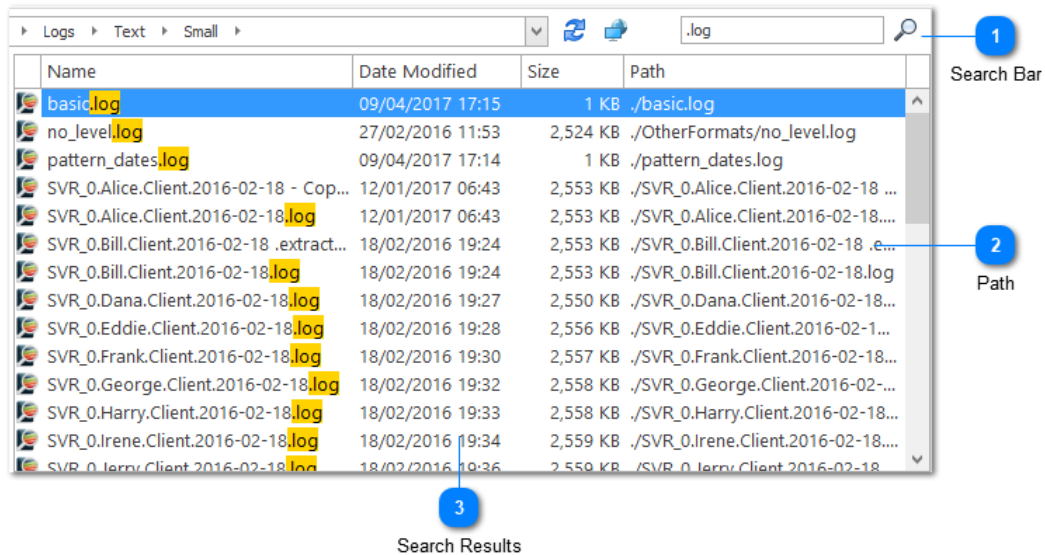


Properties

Alt+Return

The properties command will display the Windows Shell file properties for the selected directory. This command is currently disabled for remote file systems.

Searching for Files



LogViewPlus supports a basic file search by file name. Note that file contents are not accessed by the LogViewPlus search.



To execute a file search in LogViewPlus, simply enter your search text in the search bar and press return or click the apply button. Once a search is started the apply button will change to allow you to cancel the in progress search. All LogViewPlus searches are recursive starting at the currently selected directory.

Note that the asterisk character (*) cannot be used in a filename search. Searching by wildcard is implied. You can think of a search for ".log" to really mean "*.log*".







Path
./basic.log
./OtherFormats/no_level.log
./pattern_dates.log

The path column shows the path to a given search result relative to the currently selected directory.

3

Search Results


 basic.log	09/04/2017 17:15
 no_level.log	27/02/2016 11:53
 pattern_dates.log	09/04/2017 17:14
 SVR_0.Alice.Client.2016-02-18 - Cop...	12/01/2017 06:43

Search results are displayed in the log file list. Text matched by the file name search appears highlighted.

Configure File Systems

File system configuration settings are discussed in detail in the [File System Settings](#) documentation topic.


Toolbox Features

	All	View
Records:	19104	11829
Threads:	14	14
Loggers:	115	115
Debug:	6012	0
Info:	11829	11829
Warn:	986	0
Error:	277	0
Fatal:	0	0
		

The LogViewPlus toolbox is located in the bottom left corner of LogViewPlus. The toolbox contains a number of tabs. Each tab contains a different tool designed to help you get the most out of your log files.

Each of the toolbox tabs will be discussed in detail in the following sections.

Statistics Grid











	All	View
Records:	19104	11829
Threads:	14	14
Loggers:	115	115
Debug:	6012	0
Info:	11829	11829
Warn:	986	0
Error:	277	0
Fatal:	0	0
		

The statistics grid can be used to give you an idea of how many elements are in your log file. It's designed to give you a quick idea of what important information might be available in this log file. The All column represents all statistics on all log entries in the log file. The View column represent statistics on the current view.

Clicking on the column links at the top of the LogViewPlus statistics grid will open the [graph log entries dialog](#) for either the log file or the view depending on which link is clicked.

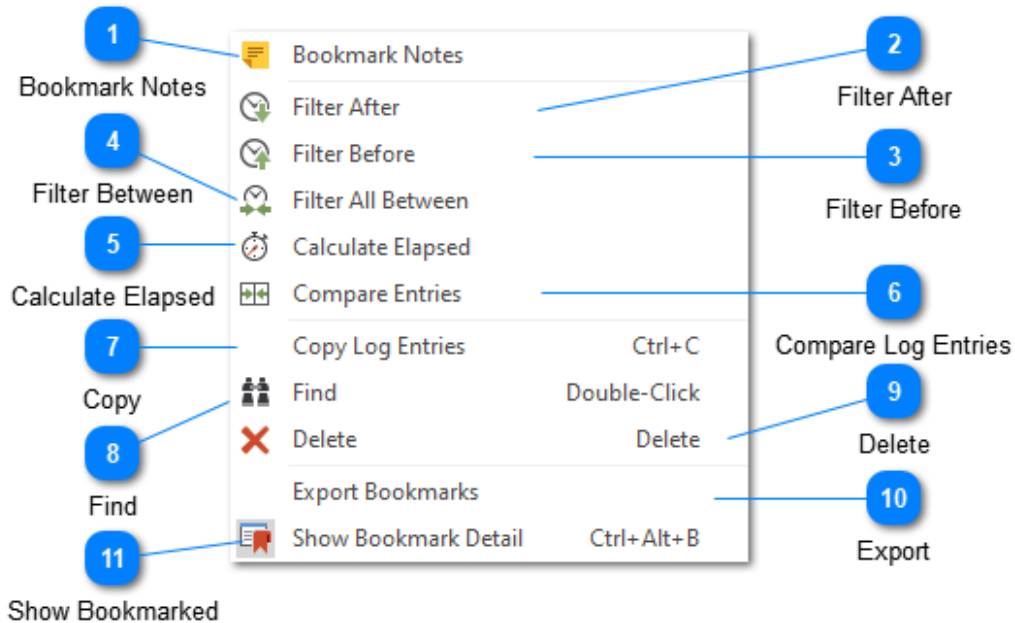
Double-clicking on a row in the statistics grid will bring up a create filter configuration screen with settings relevant to the selected statistic row.

Bookmarks

	Bookmark	Line
	SVR_0.Alice.Client.small.log	23501
	SVR_0.Alice.Client.small.log	24654
	SVR_0.Bill.Client.small.log	24672
		
		
		


The bookmarks toolbar shows all available bookmarks. Double-clicking on one of the bookmarks will automatically navigate to the bookmarked log entry. Right clicking on a bookmark will bring up the [bookmark commands](#) context menu which allows simple actions like creating quick date filters, navigation, elapsed time, and bookmark management.

Bookmark Commands



The bookmark commands context menu is available when you right-click on the bookmark in the [bookmarks toolbar](#).

1 Bookmark Notes

 Bookmark Notes

Bookmark notes allow you to associate a given log entry with a block of free-form text. Depending on the state of the current log entry, the bookmark notes command will either convert the current bookmark or display the already set bookmark notes.

2 Filter After

 Filter After

Reads the time of the bookmarked log entry and creates a new Date Filter showing everything which occurred after that time.

3

Filter Before



Filter Before

Reads the time of the bookmarked log entry and creates a new Date Filter showing everything which occurred before that time.

4

Filter Between



Filter All Between

When two or more bookmarks are selected this command will determine the start and end time of the underlying log entries and create a new Date Filter showing everything which occurred between the selected times.

5

Calculate Elapsed



Calculate Elapsed

Calculates the amount of time that has elapsed between the two selected bookmarks.

6

Compare Log Entries



Compare Entries

If you have [configured](#) LogViewPlus to use a compare tool (like WinMerge) the Compare Log Entries command will open the log entries of the two selected bookmarks using your configured tool. This is helpful for quickly determining the difference between two bookmarked log entries.

7

Copy



Copy Log Entries

Ctrl+C

The copy commands are used to copy data to the Windows clipboard. You can either copy the full log entry as it was originally written or you can copy a particular cell within the grid. Copying cells is particularly helpful if you intend to create a text filter from the cell data.

8

Find



Find

Double-Click

Finds the currently selected. This action is the same as double-clicking on the bookmark.

9

Delete



Delete

Delete

Permanently deletes the currently selected bookmark. Any notes associated with the bookmark will also be deleted.

10

Export

Export Bookmarks

Saves all current bookmarks to a log file in your Windows Temp directory. This log file is then opened in LogViewPlus using a predefined pattern. This is helpful when you want to quickly view all log entries associated with your bookmarks. It can also be helpful if you need to remember the bookmarks you have created.

11

Show Bookmarked















Show Bookmark Detail

Ctrl+Alt+B

The show bookmark detail command opens the docked bookmark view.

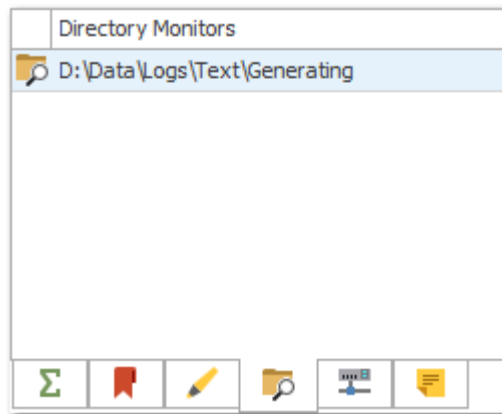
Highlights

	Highlights	Case	Delete?
	transaction	Ignore	
	192	Ignore	
<div>Create new highlight </div>			
<div>      </div>			

Highlighting allows you to search for text and mark it to make it easier to find. The LogViewPlus highlight toolbar is a scaled down version of the LogViewPlus highlight manager. Please see the LogViewPlus [highlight manager](#) documentation for more details.

Note that changes made in the highlights toolbar will be automatically saved.

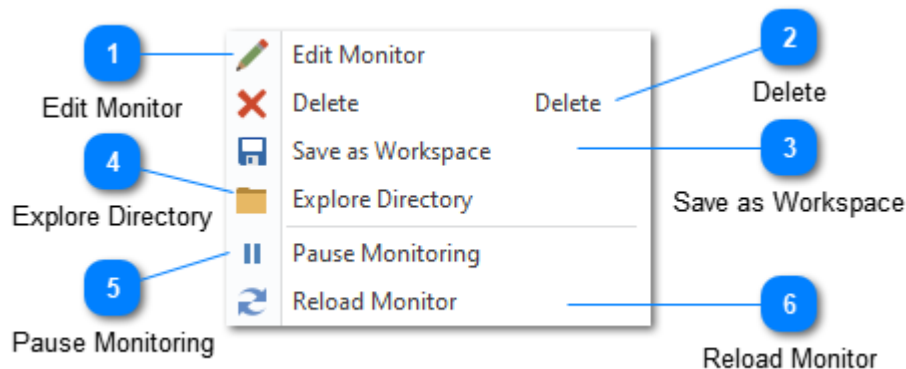
Directory Monitors



Occasionally, you might find that you need to monitor an entire directory for new log files. This is particularly helpful in situations where you have rolling log files and the names of new log files are automatically generated. In these situations it's helpful to have a directory monitor which can poll a target directory and automatically open new files as needed.

The directory monitor toolbox item shows the list of currently monitored directories. Right clicking on a directory monitor in the toolbar will bring up the [monitor commands](#) context menu which will allow you to edit and delete directory monitors.

Monitor Commands



The monitor commands context menu is displayed whenever you right click on the directory monitor in the directory monitors toolbox.

1 Edit Monitor



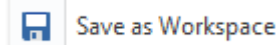
This command can be used to invoke the [edit directory monitor](#) dialog.

2 Delete



The delete command can be used to remove the current directory monitor. Any changes to the directory will be ignored after the directory monitor has been deleted.

3 Save as Workspace



The save as workspace command can be used to save the currently selected directory monitor as a workspace. Note that only the selected directory monitor will be part of the workspace. Regardless of other log files which have been opened or other directory monitors which may be running.

4

Explore Directory



Explore Directory

The explore directory command can be used to open the local directory being monitored in Windows Explorer.

5

Pause Monitoring



Pause Monitoring

The pause monitoring command can be used to pause the current directory monitor. Any changes to the directory will be ignored if the directory monitor has been paused.

6

Reload Monitor



Reload Monitor

Reloading a directory monitor will completely reapply the directory monitor settings. This is useful if you have removed some of the files initially detected by the directory monitor and would like the files to be re-added. Note that log files in LogViewPlus can only be open once. This will prevent duplicate log files from being opened by the directory monitor.

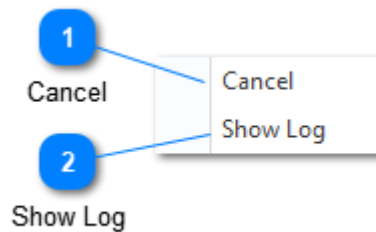
File Transfers

	Remote File Name	Progress ▲	
	SVR_0.Alice.Client....	Monitoring...	

The file transfers toolbar is used to display all of the remote files that LogViewPlus is currently tracking. Remote files will only be monitored in the event that tail is enabled for the remote file. If LogViewPlus detects that a remote file has been updated LogViewPlus will only download the updated data. This work similar to a "resume download" feature.

Right clicking on a file in the file transfers toolbar will bring up the [transfer commands](#) context menu.

Transfer Commands



The transfer commands context menu is shown whenever you right-click on the file transfers toolbox.

1 Cancel

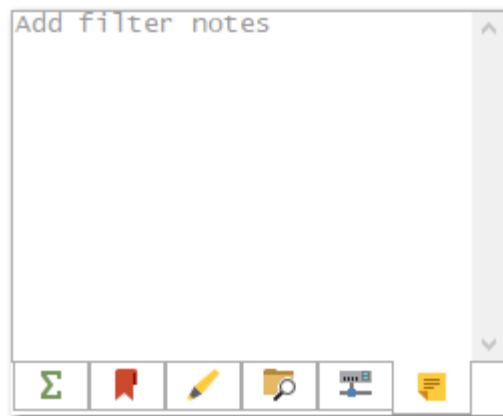
The cancel command can be used to stop monitoring a remote file. This has a similar impact as disabling tail in that it will not impact the current display of the log file but new log entries will not be added.

2 Show Log

The show log command can be used to display the LogViewPlus file transfers log. This log file contains detailed information on background file transfer operations in LogViewPlus. This log file is very useful for debugging if you are experiencing file transfer problems.

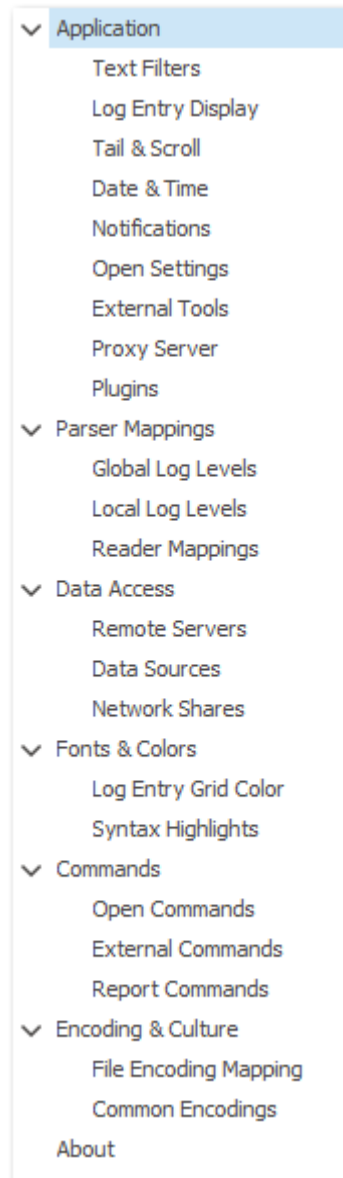
By default, this log is located at %Temp%\LogViewPlus.

Notes

A dialog box titled "Notes" with a text area labeled "Add filter notes" and a toolbar at the bottom. The toolbar contains icons for a summation symbol, a bookmark, a pencil, a magnifying glass, a document with a magnifying glass, and a speech bubble. The text area is empty and has a vertical scrollbar on the right side.

The notes toolbar is only visible if a filter is currently selected. The notes toolbar contains a free-form text box that allows you to create notes to be associated with the selected filter. Notes that have been assigned to a filter will be persisted with the filter when it is saved as [template](#). Notes will also be copied to the clipboard as part of the text which represents the filter when using the copy and paste commands.

Application Settings

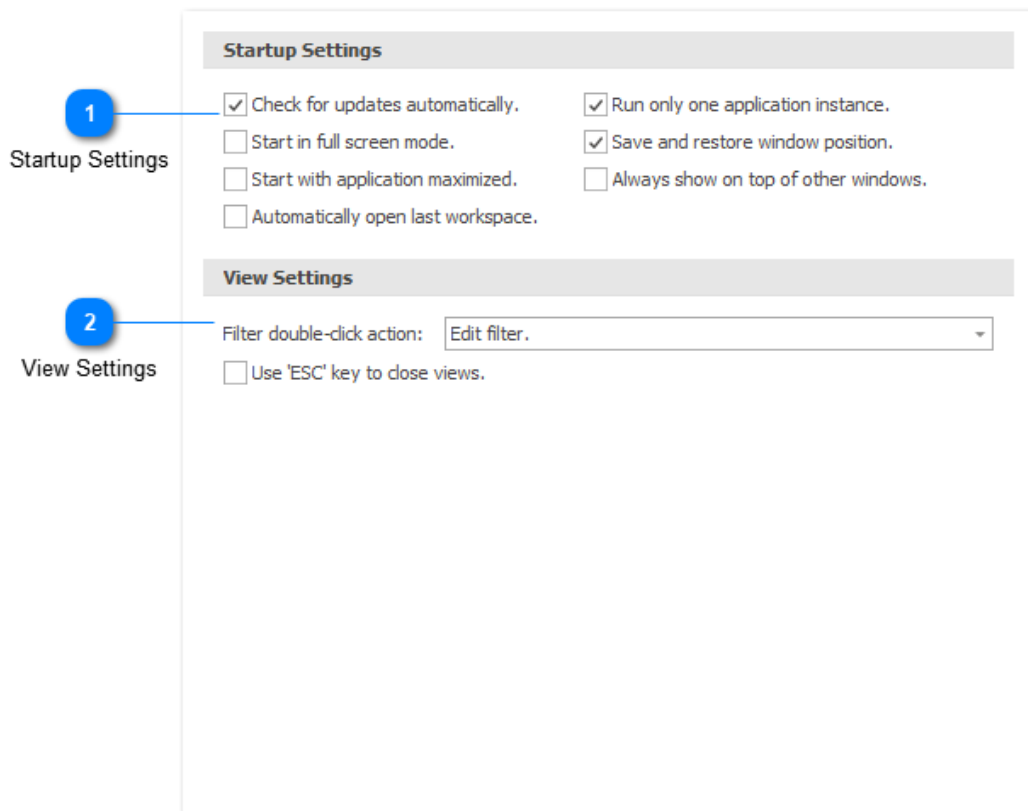


On the far left side of the settings window you will find the navigation menu. The navigation menu allows you to select the settings configuration category that you're interested in.

LogViewPlus settings are divided into seven separate sections: [Application](#), [Parser Mappings](#), [Data Access](#), [Fonts & Colors](#), [Commands](#), [Encoding & Culture](#), and [About](#).

For more information about settings management - including importing and exporting settings - please see the [Settings Management](#) documentation.

Application



The general settings section contains configuration options which affect the overall behavior of LogViewPlus.

1 Startup Settings

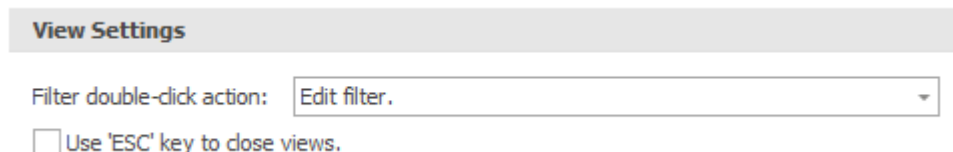
Startup Settings

- | | |
|--|--|
| <input checked="" type="checkbox"/> Check for updates automatically. | <input checked="" type="checkbox"/> Run only one application instance. |
| <input type="checkbox"/> Start in full screen mode. | <input checked="" type="checkbox"/> Save and restore window position. |
| <input type="checkbox"/> Start with application maximized. | <input type="checkbox"/> Always show on top of other windows. |
| <input type="checkbox"/> Automatically open last workspace. | |

Startup settings allow you to specify:

1. If LogViewPlus will check for updates automatically on startup. Checking for updates automatically can be disabled by setting the 'OfflineMode' flag in the application configuration file. This will also disable manual error reporting.
2. If LogViewPlus should start in full screen mode with the ribbon minimized. Also note that you can run LogViewPlus in full screen mode by pressing F11.
3. If LogViewPlus should start with the window maximized.
4. If the last known workspace should automatically be opened after LogViewPlus is started. This setting is helpful if you frequently work with the same log files.
5. If LogViewPlus should run as a single instance. Changing this option will require an application restart.
6. If LogViewPlus should save and restore to the same window position. This option will only be available if the application is running in a single instance mode because the setting does not add value if multiple windows can be opened.
7. If LogViewPlus should run as a top most window. If selected, LogViewPlus will run above other windows. Set this to true if you want LogViewPlus to always be visible.

2 View Settings



View Settings

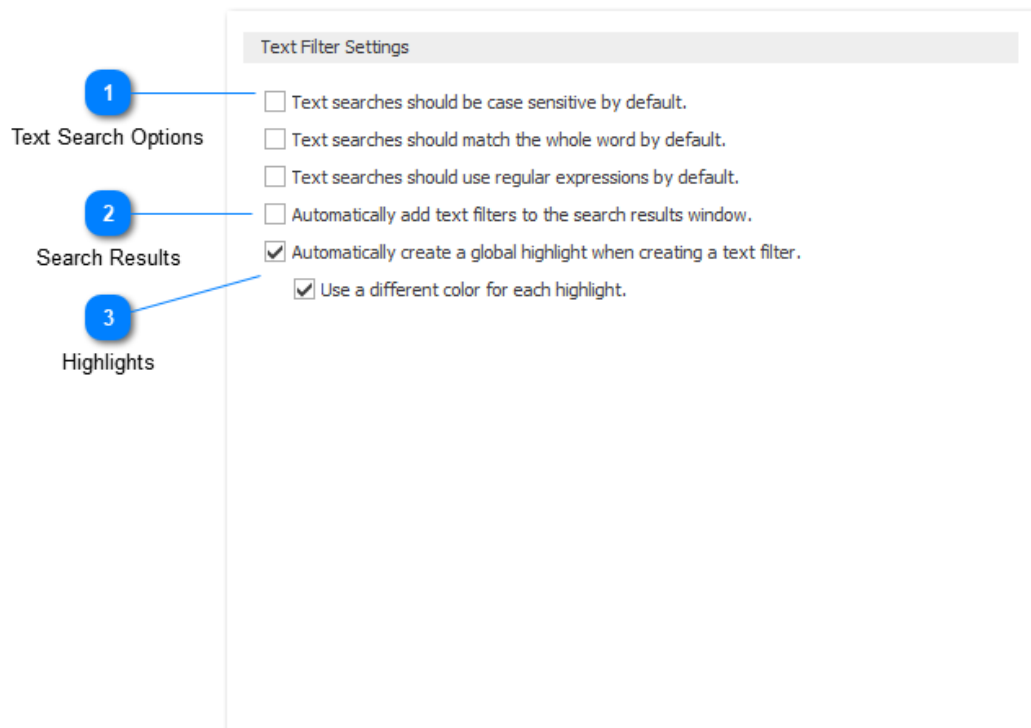
Filter double-click action: Edit filter.

☐ Use 'ESC' key to close views.

View settings allow you to specify:

1. The default action when you double click a filter. This can be set to either edit the filter or expand / collapse the tree node. If the node has no children, double-clicking will always result in an edit action.
2. If the escape key (ESC) can be used to close a view. A view can be either a filter or a log file. Note that the delete key can always be used to close a view.

Text Filters



The text filter options allow you to set the default options that should be used when creating a text filter.

1 Text Search Options

- ☐ Text searches should be case sensitive by default.
- ☐ Text searches should match the whole word by default.
- ☐ Text searches should use regular expressions by default.

The text search options allow you to specify:

1. If text searches should be case-sensitive by default. For example, should a search for "error" also match "ERROR"?
2. If text searches should match the whole word by default by default. For example, should a search for "or" also match "error".

3. If text searches should use regular expressions by default. Enable this setting if you are familiar with and frequently use regular expressions.

2

Search Results

☐ Automatically add text filters to the search results window.

If the search results checkbox is enabled then whenever you create a new text filter the results of the filter will be added to the [search results window](#). Additionally, the new filter will not be automatically selected. This will allow you to quickly search a parent filter using the contents of the search result window.

This feature allows you to search for text in a way that may feel more familiar if you are used to searching for text in applications like Notepad++ or Visual Studio.

3

Highlights

☒ Automatically create a global highlight when creating a text filter.

☒ Use a different color for each highlight.

Highlight options allow you to specify:

1. If LogViewPlus should create a global highlight whenever a text filter is created.
2. If LogViewPlus should use different colors when creating highlights. Note that you can always explicitly set a highlight color.

Log Entry Display

The screenshot shows a settings dialog titled "Log Entry Display". It is divided into two main sections: "Log Entry Grid Settings" and "Log Entry Box Settings".

Log Entry Grid Settings

- ☒ Save column settings for each parser.
- ☐ Scroll grid horizontally by default.
- Default grid view: Show message detail
- Default message size: Small
- Log level navigation size: Small
- Number of preceding rows: 10

Log Entry Box Settings

- ☒ Word-wrap by default. You can override this setting in the context menu.
- ☐ Automatically copy selected text to the clipboard.
- ☒ Show all selected log entries.

Callout 1 points to the "Log Entry Grid Settings" section, labeled "Grid Settings".

Callout 2 points to the "Log Entry Box Settings" section, labeled "Log Entry Settings".

The log entry grid settings are used to configure how the log entries should be managed and displayed in the grid.

1 Grid Settings

Log Entry Grid Settings

☒ Save column settings for each parser.

☐ Scroll grid horizontally by default.

Default grid view:

Show message detail

Default message size:

Small

Log level navigation size:

Small

Number of preceding rows:

10

The Grid Settings allow you to specify:

1. If LogViewPlus should automatically save any columns setting changes you make to a log entry grid. For example, adding or removing a column. Column settings will be associated with a parser configuration and not a particular log file. This means that future log files opened with the same parser configuration will have the same column settings. This setting requires a restart.

2. If LogViewPlus should show a horizontal scroll bar in the log entry grid. Showing a horizontal scroll bar implies that you do not want the application to auto-size the grid view columns.

3. The default grid view setting can be used to change the way that log entry messages are displayed within the log entry grid. Three options are available:

Show message detail: This view type can display the log entry message as multiple lines of text. Grid rows may be the sized differently. This is this is the default view.

One log entry per grid line: This view type will display the log entry message as a single line of text. All grid rows will be the same size.

Full Message: This view type will display the log entry message in a text box underneath a grid row.

4. The default grid message size can be used change the amount of text which is displayed in the message column of the log entry grid. This command will only be enabled if the grid view is not set to "one log entry per grid line". Message sizes range from "Very Small" to "Unlimited". Configuring the log entry grid view settings can also be done through the [grid column context menu](#).

5. The size of the log level navigation bar. The log level navigation bar is located to the right of the log entry grid and displays a summary of the log file based on log entry priority. This setting can also be used to hide the navigation bar.
6. The number of log entries to show before the selected entry. This is helpful when navigating a file. Should the focused log entry be placed at the top of the grid, or would you prefer to be able to see a few of the preceding log entries?

2 Log Entry Settings

Log Entry Box Settings

- ☒ Word-wrap by default. You can override this setting in the context menu.
- ☐ Automatically copy selected text to the clipboard.
- ☒ Show all selected log entries.

The Log Entry Settings allow you to specify:

1. If word-wrap should be on or off by default. Note that word wrap can also be turned on or off via the [Log Entry Box](#) context menu.
2. If selecting text in the log entry box will automatically copy it to the clipboard. There will be no need to press Ctrl+C or use the context menu.
3. If all log entries selected in the grid should be shown in the log entry text area, or only the focused log entry.

Tail & Scroll

Tail & Scroll Settings

☒ Automatically tail log files after opening.

☐ After initial load, sort log entries by date in ascending order.

☐ Clear all log entries when the log file is rolled or recreated.

Change tracking frequency (higher values require more CPU):

When tailing a remote file, stop monitoring after:

When tail opening a file, default to opening the last:

Tail & Scroll Settings allow you to specify:

1. If LogViewPlus will automatically tail the log file once it is opened. Disable this check box if you do not frequently [tail log files](#).
2. If your log files do not contain naturally sorted log entries, you have the option of sorting them on load. With this option selected, newer log entries will be located at the bottom of the grid. If the file is in tail mode, new log entries will always be found at the bottom of the grid - regardless of the log entry time stamp. This is to prevent 'lost' entries which are not seen by the user.

3. If LogViewPlus should remove all existing entries when a log file is rolled. This is helpful if previous log entries become irrelevant after roll.

4. The frequency LogViewPlus will use when checking a file for changes. Checking the file more frequently requires more CPU which may be an issue on low-power machines like laptops.

5. The time window in which LogViewPlus will monitor a remote log file after the file has been opened. Note that monitoring remote log files requires pulling the server for changes via SFTP or FTP.

6. The default log file tail size. When [tail opening a log file](#), you will have the option of overriding this default value.

Date & Time

The screenshot shows the 'Date & Time Settings' dialog box. It contains the following fields and callouts:

- 1 Display Formats:** Points to the 'Date Display Format' field, which contains '%d MMM yyyy'.
- 2 Assumed Time Zone:** Points to the 'When time zone is unspecified' dropdown menu, which is set to 'Assume log entries are in Local Time'.
- 3 Target Time Zone:** Points to the 'Convert timestamps to' dropdown menu, which is set to 'Do not modify log entry time'.

The dialog box also includes a title bar 'Date & Time Settings' and a descriptive text: 'The following date and time display settings will be used to display log items in the grid.'

The date and time settings allow you to control how dates and times are displayed within the LogViewPlus [Log Entry Grid](#).

1 Display Formats

This section shows the 'Date Display Format' field with the value '%d MMM yyyy' and the 'Time Display Format' field with the value 'HH:mm:ss.fff'.

The display format settings allow you to specify the format that should be used to display dates and times in the [Log Entry Grid](#). Note that the display format is independent of the log file parse format.

2 Assumed Time Zone

This section shows the 'When time zone is unspecified' dropdown menu, which is set to 'Assume log entries are in Local Time'.

Often log entries are written using a date time format that does not specify the time zone. In this case LogViewPlus must make an assumption about the underlying time zone. This setting allows you to specify what LogViewPlus should assume about timestamps which do not explicitly specify a time zone. The options available are:

Make no assumptions about time zone - The timestamp should be treated as "unspecified".

Assume log entries are in Local Time - Assume the log entries written in the same time zone as the current machine.

Assume log entries are in UTC Time - Assume the log entries were written in a standard UTC time zone.



Target Time Zone

Convert timestamps to:

Do not modify log entry time



This setting allows you to specify the display time zone for log entry timestamps. The displayed time zone can either be local or UTC. By default LogViewPlus does not modify the display time.

Notifications

LogViewPlus filters can be configured to notify you when new log entries are matched. Use the settings below to manage your notification providers.

Name	Type
test	Simple

Existing Providers

1

2

Notification Management

+ Add Clone Edit Delete

Notification settings allow you to add, edit and delete [notification providers](#).

1 Existing Providers

Name	Type
test	Simple

Existing providers are shown in the notification configuration grid. Select a provider in order to edit or delete the configuration.

2

Notification Management



The notification management commands are located at the bottom of the notification provider's configuration. These commands allow you to add, clone, edit and delete notification providers.

Open Settings

The screenshot shows a dialog box titled "Open Settings". It has two main sections: "Open Settings" and "Recent History".

Open Settings

1 ☐ When opening files based on rules, only open files with a known parser.

Recent History

2 When setting recent history tracking, keep in mind that there may be start-up performance implications to gathering file information. Particularly if your file history references remote files.

Limit file tracking to: 20 Files

Limit directory tracking to: 20 Directories

Callout 1 points to the checkbox in the "Open Settings" section. Callout 2 points to the explanatory text in the "Recent History" section.

The Open Settings view can be used to fine tune how you open files.

1 Open Settings

Open Settings

☐ When opening files based on rules, only open files with a known parser.

Currently, there is only one open setting. This setting allows you to control what happens when an unknown file is identified by a rule based open command. For example, if you are [opening a zip file](#) or using a [directory monitor](#), you can choose to ignore unknown files rather than attempting to open them.

2

Recent History

Recent History

When setting recent history tracking, keep in mind that there may be start-up performance implications to gathering file information. Particularly if your file history references remote files.

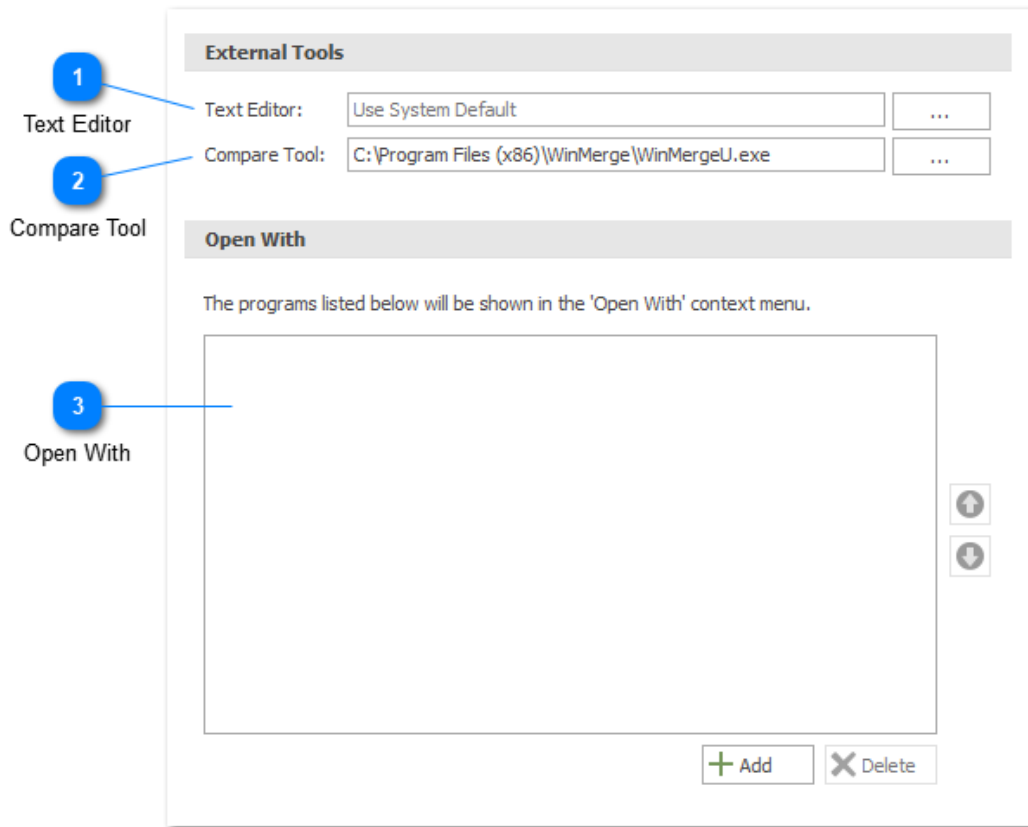
Limit file tracking to:

Limit directory tracking to:

Recent history settings can be used to manage the number of items that LogViewPlus will track when managing your log file access history. Recent history items are managed in batches of 10 up to a maximum of 50 items. By default LogViewPlus will track the most recent 20 log files.

For more information please see the file system [history](#) documentation.

External Tools



LogViewPlus attempts to make use of external tools where possible. Use the External Tools view to configure the default tools used.

1 Text Editor

Text Editor: ...

There are several features available in LogViewPlus that make use of a text editor.

If this setting is not configured, LogViewPlus will use the editor configured in Windows for opening *.txt files.

2

Compare Tool

Compare Tool:

You can optionally configure LogViewPlus to point to a local compare tool (such as WinMerge). Doing so will allow you to compare two log entries by executing the "Compare Log Entries" command on the [Log Entries](#) menu.

The tool configured must support a command line with the signature "cmd.exe Arg1 Arg2" where cmd.exe is the configured process and arg1 and arg2 are the files to be compared.

3

Open With

Open With

The 'Open With' settings allow you to configure a list of applications which will be available in the LogViewPlus [File System Context Menu](#).

Proxy Settings

The screenshot shows a 'Proxy Server' dialog box. A blue circle with the number '1' points to the 'Proxy Settings' label on the left. A blue circle with the number '2' points to the 'Test Internet Connection' button at the bottom right of the dialog box.

Proxy Server

There are four functional areas where LogViewPlus can optionally make use of a proxy server: registration, error reporting, checking for updates, and access to file systems outside of your local network (via SFTP, FTP or SCP). Configuring a proxy server is not required. Changes to this setting will require an application restart.

Proxy Type:

Hostname:

Port:

Username:

Password:

There are four functional areas where LogViewPlus can optionally make use of a proxy server: registration, error reporting, checking for updates, and access to file systems outside of your local network (via SFTP, FTP or SCP). Configuring a proxy server is not required.

1 Proxy Settings

Proxy Type:

Hostname:

Port:

Username:

Password:

Use the available proxy settings to configure your proxy server. If you are unfamiliar with how to connect to your proxy server, please contact local your system administrator.

2

Test Connection

Test Internet Connection

The test Internet connection command is used to verify that you can connect successfully to remote sites. If you have configured a proxy server, the test will attempt to use the configuration provided.

Note that this test may be executed even if no proxy server has been configured.

Plugins

You can extend LogViewPlus through the use of plugins. Several different types of plugins are possible including Custom Parsers, Custom Filters and Postprocessors.

[Find out more about creating custom plugins](#)

1 Extensibility ☒ Allow custom plugins. This setting requires an application restart.

2 Plugin List

Currently Loaded Plugins		
Class	Plugin Type	Assembly
CustomFilter.MyFilter	Filter	CustomFilter.dll
Customer.Parsers.OrderXmlParser	Parser	Customer.Parsers.dll
Customer.Parsers.OrderXmlParserV2	Parser	Customer.Parsers.dll
CustomParser.MyParser	Parser	CustomParser.dll
CustomParserXml.CustomXmlParser	Parser	CustomParserXml.dll
CustomReader.MyReader	Reader	CustomReader.dll
CustomPostProcessor.MyPostProcessor	Post Processor	CustomPostProcessor.dll
CustomPostProcessorAdvanced.MyAdvance...	Post Processor	CustomPostProcessorAdv...

The plug-ins settings menu can be used to view which plug-ins are currently running in the application. This is particularly helpful when debugging problems loading custom built extensions.

1 Extensibility

☒ Allow custom plugins.

The extensibility settings allow you to specify whether custom log filters, parsers and post processors should be allowed by the application. Changing this settings will require an application restart. Please see the [customization](#) documentation for more information.

2 Plugin List

Currently Loaded Plugins		
Class	Plugin Type	Assembly
CustomFilter.MyFilter	Filter	CustomFilter.dll
Customer.Parsers.OrderXmlParser	Parser	Customer.Parsers.dll
Customer.Parsers.OrderXmlParserV2	Parser	Customer.Parsers.dll

The plug-ins list shows all custom plug-ins which are currently running in LogViewPlus. This list will only be visible if extensibility has been enabled since application startup.

Parser Mappings

The screenshot shows the 'Parser Mappings' dialog box in LogViewPlus. It contains a table with columns: Name, Regex, Parser, and Parser Args. The table lists three parsers: 'Custom_ILogParse...', 'Log4 XML', and 'Basic Parser'. Each has an unchecked checkbox in the 'Regex' column. To the right of the table are two arrow buttons for reordering. At the bottom are four buttons: 'Parser Wizard', 'Add', 'Edit', and 'Delete'. Numbered callouts point to these elements: 1 points to the 'Parsers' label on the left; 2 points to the 'Reorder' buttons on the right; 3 points to the 'Parser Wizard' button at the bottom; and 4 points to the 'Add', 'Edit', and 'Delete' buttons at the bottom.

LogViewPlus uses the rules below to determine how a log file should be parsed. The rules are processed in order from top to bottom until all file name patterns have been tried. If no match is found, LogViewPlus will default to the BasicParser.

The Parser Wizard is recommended when creating new parser configurations.

Name	Regex	Parser	Parser Args
Custom_ILogParse...	<input type="checkbox"/>	MyParser	
Log4 XML	<input type="checkbox"/>	Log4XmlParser	
Basic Parser	<input type="checkbox"/>	BasicParser	

1 Parsers

2 Reorder

3 Parser Wizard

4 Parser Management

The 'log parser mappings' setting configuration is used to determine how a given log file will be processed. The rules for determining how a log file is processed are keyed off of the log file name. LogViewPlus will try to match the log filename using the first filename pattern. If the first file name pattern doesn't match, LogViewPlus will move its second file name pattern and so on until a match is found. Once a match is found LogViewPlus will use the parser configured for that match. In the event that no match is found, LogViewPlus will parse the file using the basic parser.

You can find out more about the [Log Parsers](#) available in LogViewPlus.

1 Parsers

Name	Regex	Parser
Custom_ILogParse...	<input type="checkbox"/>	MyParser
Log4 XML	<input type="checkbox"/>	Log4XmlParser
Basic Parser	<input type="checkbox"/>	BasicParser

The parsers grid shows the currently configured parsers. This grid is read-only.

The 'Name' column will display the file name pattern for the parser configuration if a friendly name has not been provided.

2

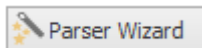
Reorder



You can use the ordering commands on the far right side of the log parser mappings configuration to specify the order in which file name patterns are processed.

3

Parser Wizard



The Parser Wizard can be used to guide you through the process of selecting and configuring a LogViewPlus parser. Please see the [Parser Wizard](#) documentation for more information.

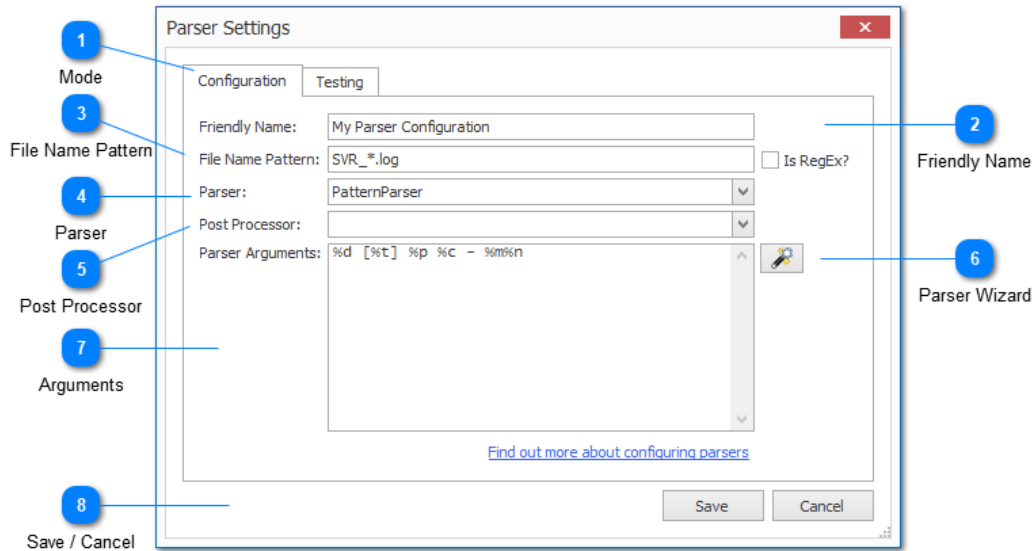
4

Parser Management



The parser management commands located at the bottom of the log parser mapping configuration allow you to add, edit and delete log parser mappings.

Parser Settings



The parser settings dialog is used to create a new parser mapping. A parser mapping is a relationship between a log file name and instructions on how the log file should be parsed. Parsing instructions include setting the parser type and providing any arguments which may be required by the parser.

You can find out more about the [Log Parsers](#) available in LogViewPlus. It is also possible to create [custom log parsers](#).

1 Mode



There are two basic functions of the parser settings dialog.

1. Creating the parser mapping.
2. Testing the mapping to confirm that it is working correctly. [Parser testing](#) is discussed in the next section.

2

Friendly Name

Friendly Name:

A friendly name can be set to more easily refer to this configuration in the future. By default, the friendly name will mirror the file name pattern.

3

File Name Pattern

File Name Pattern: ☐ Is RegEx?

Use the file name pattern text box to specify a pattern which can be used to match the filenames of your application log file. By default, the file name can contain a wildcard character - an asterisk *. This character will match one or more other characters. For example, say the name of your log file has a date suffix, something like MyApp_12.2.16.log. In this case, you could match the date part of the filename using the asterisk wildcard. Your pattern might be **MyApp_*.log**.

If you want to use the same parser with multiple log files, you can use the pipe character - | - to separate file names. For example, **SVR_*.log|*MyOtherApp***.

Usually, the asterisk wildcard is sufficient, but if you find you need more power to match your log file name, then you can use a regular expression instead. To do this, check the "Is RegEx?" checkbox to let LogViewPlus know the search needs to be performed differently.

4

Parser

Parser:

Select the parser you want to use with the given log files. Find out more about the [Log Parsers](#) available in LogViewPlus.

5

Post Processor

Post Processor:

If you have configured LogViewPlus to allow custom plugins, you may see an option to set a post processor. Post processors are discussed later in [Custom Extensions](#).

6

Parser Wizard



The [Parser Wizard](#) can be used to guide you through the process of configuring your log file parser.

7

Arguments

Parser Arguments: %d [%t] %p %c - %m%n

You can use the parser arguments text box to configure the selected parser. Note that different parsers require different configuration. Please see the relevant [Log Parser](#) section for more information about how to configure your parser.

The parser arguments text box is optional depending on the type of parser selected. Some log parsers do not need configuration and therefore will not provide a parser arguments option.

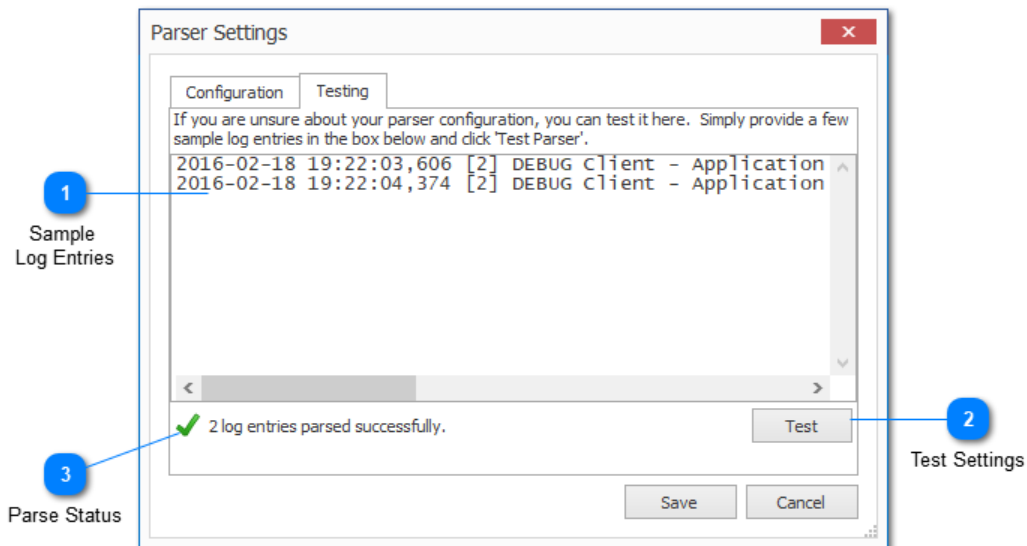
If you have written a custom parser and it needs the full path to the target log file, consider using the appropriate target [argument template](#).

8

Save / Cancel

Once you are done configuring your parser you will need to save your settings. Alternatively, if you're not happy with your parser configuration, you can cancel it.

Parser Testing



Parser configuration can be tricky and it may take a few attempts to get it right. Using the testing tab of the parser settings dialog, you can input a few sample log entries and use the current unsaved parser configuration to try to parse those log entries. If parsing is successful, LogViewPlus will display a dialog containing the number of log entries parsed.

In the example above, we expect LogViewPlus to tell us that two log entries have been successfully parsed.

Testing your log parser configuration in this way is generally easier than closing and reopening your target log file. Note that test data will be saved along with other parser configuration settings.

1 Sample Log Entries

```
2016-02-18 19:22:03,606 [2] DEBUG
2016-02-18 19:22:04,374 [2] DEBUG
```

Sample log entries can be placed in the large text area provided.

2 Test Settings

Test

Once you have entered your sample log entries, you can test your parser using the 'Test' command.

3

Parse Status

✓ 2 log entries parsed successfully.

The parse status shows the number of log entries that have been successfully parsed. If an error occurred during parsing, an error message will be displayed instead. This information will only be displayed after the sample log entries have been tested.

Global Log Levels

Log files can use many different logging levels. Some log levels like 'FINE' and 'ALERT' are rarely used. In order to simplify categorization, LogViewPlus uses 5 primary levels which cannot be modified. Each primary level may contain a list of secondary levels which will be color coded the same as their parent. Granularity of log level filtering is controlled using the 'Filter By' option in the 'Log Level Filter' dialog.

1 Primary List

2 Secondary List

3 List Management

Primary Log Level	Secondary Log Level
✗ FATAL	✗ ALERT
● ERROR	✗ CRITICAL
● WARN	✗ EMERGENCY
● INFO	✗ SEVERE
● DEBUG	

+ Add ✎ Edit ✗ Delete

Log files can use many different logging levels. Some of these log levels like 'FINE' and 'ALERT' may be rarely used. In order to simplify categorization, LogViewPlus uses 5 primary levels which cannot be modified. Each primary level may contain a list of secondary levels which will inherit behavior (such as color coding) from the parent. By default, LogViewPlus supports the following log level hierarchy.

Primary	Secondary
Fatal	Alert
	Critical
	Emergency
	Severe
Error	
Warn	Notice

Info	
Debug	Trace
	Verbose
	Fine
	Finer
	Finest

Configuring your log levels globally allows for centralized settings and easier configuration management. However, this approach presents a problem. Imagine the secondary log level "3". Some systems might see "3" as an Error while others see it as Debug. How can we configure LogViewPlus to interpret log level "3" when its meaning can change between systems? In LogViewPlus, this is achieved by configuring parser specific [Local Log Levels](#). Local log levels take precedence over global log levels.

Once your log levels have been configured, you can control filter granularity using the 'Filter By' option in the '[Log Level Filter](#)' dialog.

You can use the Global Log Level configuration settings described here to specify default log levels that will be applied when no [local log levels](#) are detected. Global log levels are also available to you in the log level filter dialog.





Changes made to global log levels will require a log file refresh.

1 Primary List

	Primary Log Level
✖	FATAL
●	ERROR
●	WARN
●	INFO
●	DEBUG

The list of primary log levels. This list cannot be modified.

2 Secondary List

	Secondary Log Level
	ALERT
	CRITICAL
	EMERGENCY
	SEVERE

The list of secondary log levels. The contents of this list may change depending on your primary log level selection. You can add new secondary log levels by clicking the 'Add' command located below this list box. Custom secondary log levels can be modified and deleted as well. Please note that the default secondary log levels cannot be modified.

3

List Management







The list management commands used to modify secondary log levels.

Local Log Levels

Local log levels apply only to a single parser or reader. If you are new to log levels in LogViewPlus, we recommend starting with [Global Log Levels](#) before creating a parser specific configuration.

Configuration Name	Log Levels
SqlServer	FATAL, ERROR {5}, WARN {4}, INFO {3}, DEBUG {1, 2}

[Find out more about local log levels.](#)

 Add  Clone  Edit  Delete

Local log level configuration lets you to specify custom log levels for a given parser or reader. This gives you a high degree of control over how log levels should be resolved.

For example, imagine the secondary log level "3". Some systems might see "3" as an Error while others see it as Debug. You can use a local log level to ensure the correct meaning based on the parser type used for the file. Local log levels take precedence over global log levels.

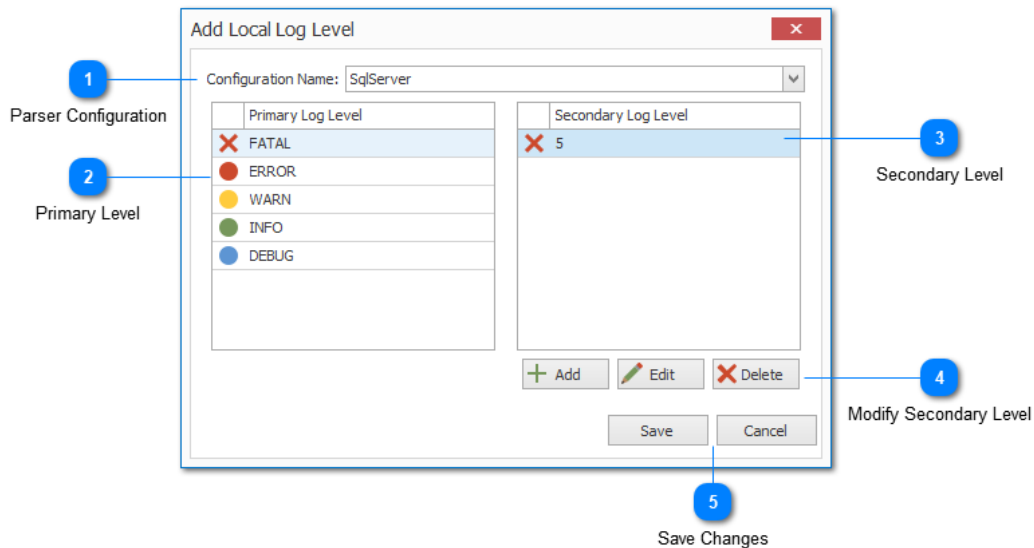
Local log levels are specific to a named parser configuration. If your target parser configuration does not have a friendly name defined, it will not be available for local log

level configuration. Application settings must be saved after changing or adding a parser configuration friendly name.

You can add or modify a Local Log Level by using the [Local Level Mapping](#) dialog.

Changes made to local log levels will require a log file refresh.

Local Level Mapping



The log level mapping dialog allows you to add or modify a [Local Log Level](#) setting.






Creating a local log level is similar to creating a [Global Log Level](#). We recommend experimenting with Global Log Levels first to understand concepts. Local log levels are the same, except that the levels will apply only to the given parser configuration. The parser configuration must be named.

1 Parser Configuration

Configuration Name:

The parser configuration associated with this local log level mapping. Only parser configurations with a friendly name will be shown here. This allows you to edit the parser configuration in the future without needing to update the local log level mapping (assuming the friendly name does not change).


2 Primary Level

	Primary Log Level
	FATAL
	ERROR
	WARN
	INFO
	DEBUG

The five primary log levels. See [Global Log Levels](#) for more information. Primary log levels cannot be modified.

3

Secondary Level

	Secondary Log Level
	5

Secondary log levels are used to identify log level and associate it with a primary. The contents of this list may change depending on your primary log level selection. You can add new secondary log levels by clicking the 'Add' command located below this list box.

When creating a local log level mapping, no secondary values will be assumed. All secondary values must be entered manually.

4

Modify Secondary Level

 Add
  Edit
  Delete

Use the commands located at the bottom of the secondary level list to create, edit or delete a log level.

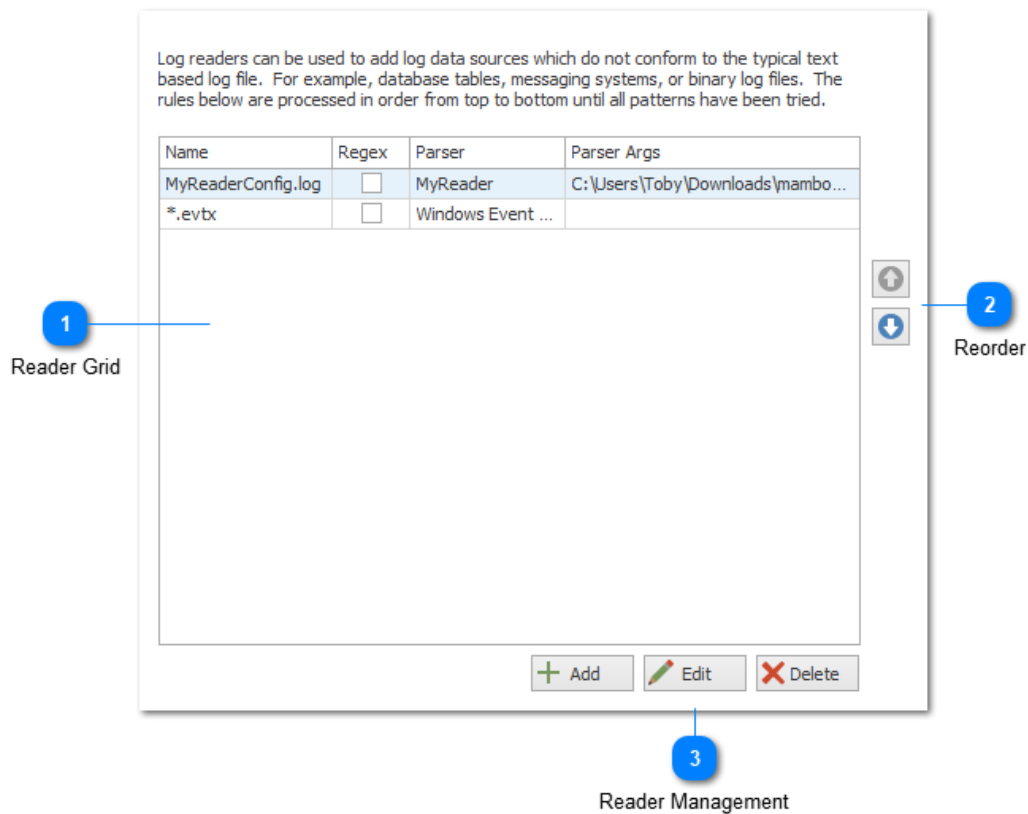
5

Save Changes

 Save
  Cancel

Once you have finished modifying the log level mapping, you will need to save your changes. If you are modifying local log levels from the application settings dialog, you will need to save changes to that dialog as well. Finally, your log files may need to be refreshed to detect the changes.

Reader Mappings



The reader mappings settings dialog shows the list of currently configured log file readers. Log file readers can be used to process log files which are not stored as text. LogViewPlus only ships with one log file reader - the Windows Event Log reader. However, LogViewPlus can be extended through the creation of [custom readers](#).

1 Reader Grid

Name	Regex	Parser
MyReaderConfig.log	<input type="checkbox"/>	MyReader

The reader grid shows the currently configured log readers. This grid is read-only.

The 'Name' column will display the file name pattern for the reader configuration if a friendly name has not been provided.

2

Reorder



You can use the ordering commands on the far right side of the log parser mappings configuration to specify the order in which file name patterns are processed.

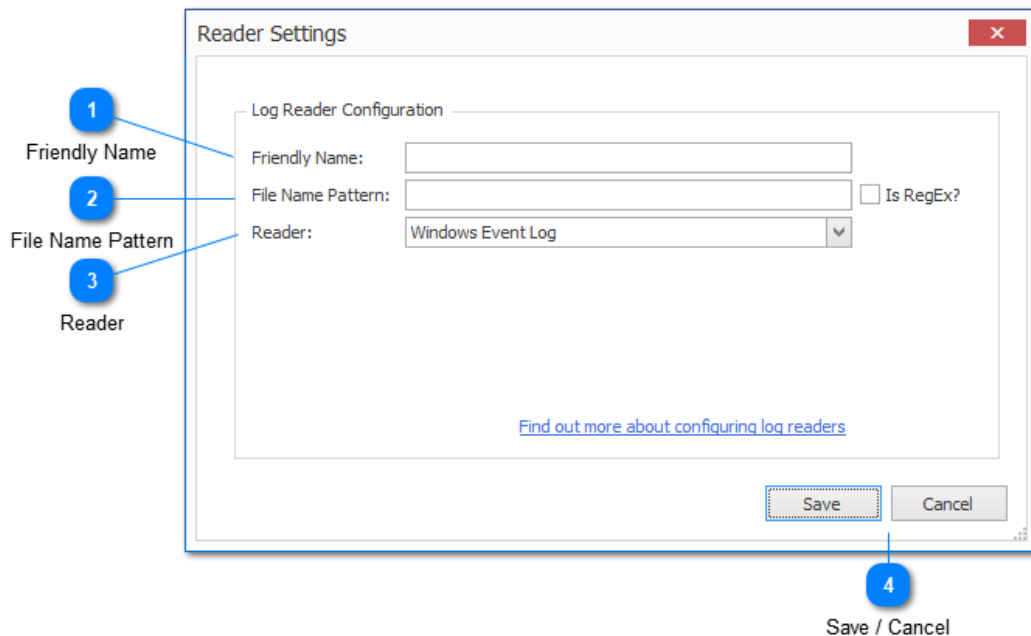
3

Reader Management



The reader management commands located at the bottom of the log reader mapping configuration allow you to add, edit and delete log reader mappings.

Reader Settings



Most log files are persisted as text. Binary log files can be read using a log file reader. By default, LogViewPlus includes a Windows Event Log reader, but you can create additional [custom log file readers](#).

Configuring a log file reader is discussed below.

1 Friendly Name

Friendly Name:

A friendly name can be set to more easily refer to this configuration in the future. By default, the friendly name will mirror the file name pattern.

2 File Name Pattern

File Name Pattern: ☐ Is RegEx?

Use the file name pattern text box to specify a pattern which can be used to match the filenames of your application log file. By default, the file name can contain a

wildcard character - an asterisk *. This character will match one or more other characters. For example, say the name of your log file has a date suffix, something like MyApp_12.2.16.log. In this case, you could match the date part of the filename using the asterisk wildcard. Your pattern might be MyApp_*.log.

Usually, the asterisk wildcard is sufficient, but if you find you need more power to match your log file name, then you can use a regular expression instead. To do this, check the "Is Regex?" checkbox to let LogViewPlus know the search needs to be performed differently.

3 Reader

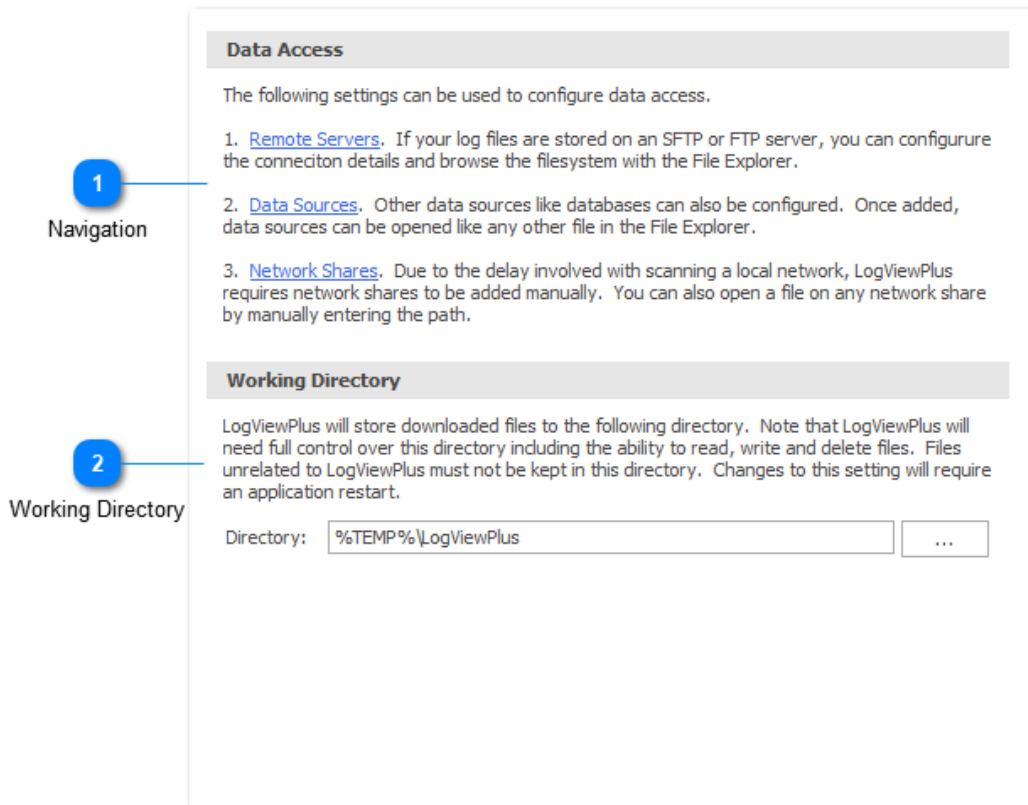
Reader: ▼

The reader you want to associate with the given filename pattern.

4 Save / Cancel

Once you are done configuring your reader you will need to save your settings. Alternatively, if you're not happy with your reader configuration, you can cancel it.

Data Access



The default data access settings view can be used to navigate to other relevant application settings or set the LogViewPlus working directory.

A detailed description of [Data Access Settings](#) is discussed latter in this documentation.

1 Navigation

1. [Remote Servers](#).
the connection detail:

A large area of the default data access settings view has been dedicated to documentation. Hyperlinks in this documentation can be used for settings navigation.

2

Working Directory

Directory:

...

The working directory is the location LogViewPlus uses for storing remote file system access log files as well as any files downloaded or monitored by LogViewPlus. By default LogViewPlus will store this information in a folder called "LogViewPlus" located in your local temp directory. You can use Windows Explorer to navigate to this directory with the path %TEMP%\LogViewPlus.

Fonts & Colors

The screenshot shows a 'Fonts & Colors' settings window. It is divided into two main sections: 'Font Settings' and 'Color Settings'. In the 'Font Settings' section, there is a recommendation to use a monospaced font. Below this, there are two rows of settings: 'Log Entry Display Font' set to 'Lucida Console' with a size of '10' and a checked 'Monospaced?' checkbox, and 'Log Entry Grid Font' set to 'Tahoma' with a size of '10'. In the 'Color Settings' section, there is a note about the impact of color settings on the 'Log Entry Grid' and 'Syntax Highlighting' colors. Below this note, the 'Application Skin' is set to 'Office White', and there is a 'Reset All Colors' button. Four numbered blue circles with arrows point to specific elements: '1' points to the 'Log Entry Display Font' dropdown, '2' points to the 'Log Entry Grid Font' dropdown, '3' points to the 'Application Skin' dropdown, and '4' points to the 'Reset All Colors' button.

Font Settings

We recommend using a monospaced font for the log entry display to ensure all characters are the same size and preserve text alignment.

Log Entry Display Font: ☒ Monospaced?

Log Entry Grid Font:

Color Settings

The color settings below impact the [Log Entry Grid](#) and [Syntax Highlighting](#) colors which can also be configured independently.

Application Skin:

The fonts and colors setting dialogue is used primarily to set application fonts as well as the default skin. Additional color configuration options such as specifying custom [grid colors](#) and [syntax highlighting colors](#) are available.

1 Log Entry Font

Log Entry Display Font: ☒ Monospaced?

The Log Entry Box setting configuration allows you to set the behavior and visual style of the Log Entry Box. The Log Entry Box is the text area shown at the bottom of the [Log Entry Grid](#).

The font chosen here will also be used for [Log Entry Grid Grouping](#).

Monospace fonts are recommended because they print all characters the same size. This is sometimes important when viewing log entries which are formatted

according to this expectation. A checkbox is provided to filter the font list to show only the monospaced fonts installed on the local machine.

2

Log Entry Grid Font

Log Entry Grid Font:

Sets the font used by the [Log Entry Grid](#).

3

Application Skin

Application Skin:

Sets the default application skin. This can also be set in the [Program Toolbar](#).

4

Reset Colors

Sets all colors used by the application back to installation defaults. This includes the application skin as well as any custom grid or syntax highlighting colors.

Log Entry Grid Color

1 Grid Color Settings

Row Colors		
Log Level	Fore Color	Back Color
Unknown	0, 0, 0	255, 255, 255
Debug	54, 96, 146	255, 255, 255
Info	0, 125, 60	255, 255, 255
Warn	225, 125, 50	255, 255, 255
Error	240, 0, 0	255, 255, 255
Fatal	255, 255, 255	190, 0, 0

2 Add Level

3 Example Grid

Example Log Entry Grid	
DEBUG	Checking time against network.
INFO	Application progressing normally.
WARN	Expected type should contain a message.
ERROR	Failed in Goo. Calling Foo. Check inner exception.
FATAL	Exception thrown from method Bar. Unable to continue.

The Log Entry Grid Color setting configuration can be used to specify how long entries of a given information level should be displayed in the log entries grid.

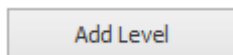
1 Grid Color Settings

Row Colors		
Log Level	Fore Color	Back Color
Unknown	0, 0, 0	255, 255, 255
Debug	54, 96, 146	255, 255, 255
Info	0, 125, 60	255, 255, 255
Warn	225, 125, 50	255, 255, 255
Error	240, 0, 0	255, 255, 255
Fatal	255, 255, 255	190, 0, 0

The color settings allows you to specify the foreground color and background color for a given log level.

By default, only primary log levels have a custom appearance. Secondary log levels will be colour coded based on their primary assignments.

2 Add Level



Allows you to add a secondary log level to the Color Settings grid. Once added, you will be able to define a custom font colour and background colour.

3 Example Grid

Example Log Entry Grid	
DEBUG	Checking time against network.
INFO	Application progressing normally.
WARN	Expected type should contain a message.
ERROR	Failed in Goo. Calling Foo. Check inner exception.
FATAL	Exception thrown from method Bar. Unable to continue.

The example grid gives you a real-time presentation of the current grid configuration settings. This allows you to see what your custom grid will look like without you having to save your changes.

Syntax Highlights

The screenshot shows the 'Syntax Highlight Options' dialog box. It has two main sections: 'Syntax Highlight Options' and 'Syntax Colors'. The 'Syntax Highlight Options' section contains two checkboxes: 'Disable automatic syntax highlighting.' and 'Changing the highlight style applies to all log entries in the log file.' The 'Syntax Colors' section contains a 'Target Syntax:' dropdown menu set to 'XML & JSON' and a table of syntax elements with their corresponding colors and RGB values. Three blue callout boxes with numbers 1, 2, and 3 point to the 'Highlight Options' section, the 'Target Syntax' dropdown, and the 'Syntax Highlights' table respectively.

1 Highlight Options

2 Target Syntax

3 Syntax Highlights

Syntax Highlight Options	
<input type="checkbox"/>	Disable automatic syntax highlighting.
<input type="checkbox"/>	Changing the highlight style applies to all log entries in the log file.

Syntax Colors	
Target Syntax:	XML & JSON
Keyword	30, 144, 255
Attribute	218, 112, 214
Symbol	178, 34, 34
Literal	184, 134, 11
Quoted	72, 61, 139
Comment	0, 128, 0

The Syntax Highlighting setting configuration can be used to the colors used when pretty printing a given syntax.

1 Highlight Options

- ☐ Disable automatic syntax highlighting.
- ☐ Changing the highlight style applies to all log entries in the log file.

Highlighting options allow you to configure:

1. If LogViewPlus should automatically pretty print log entries. This setting can also be turned on or off for a particular log entry in the [Log Entry Box](#) context menu.
2. If changing the highlight style in the [Log Entry Box](#) context menu should apply to all log entries in the log file, or just the current log entry. By default, only the current log entry will be impacted.

2

Target Syntax

Target Syntax: XML & JSON









The target syntax command can be used to specify which syntax type is currently being configured.

LogViewPlus currently supports three separate syntax types: XML, JSON, and Symbols & Numbers. Symbols & Numbers is the default syntax used when displaying most log entries.

XML and JSON are separate syntax formats. However these formats share a common syntax color configuration and therefore share a common target syntax.

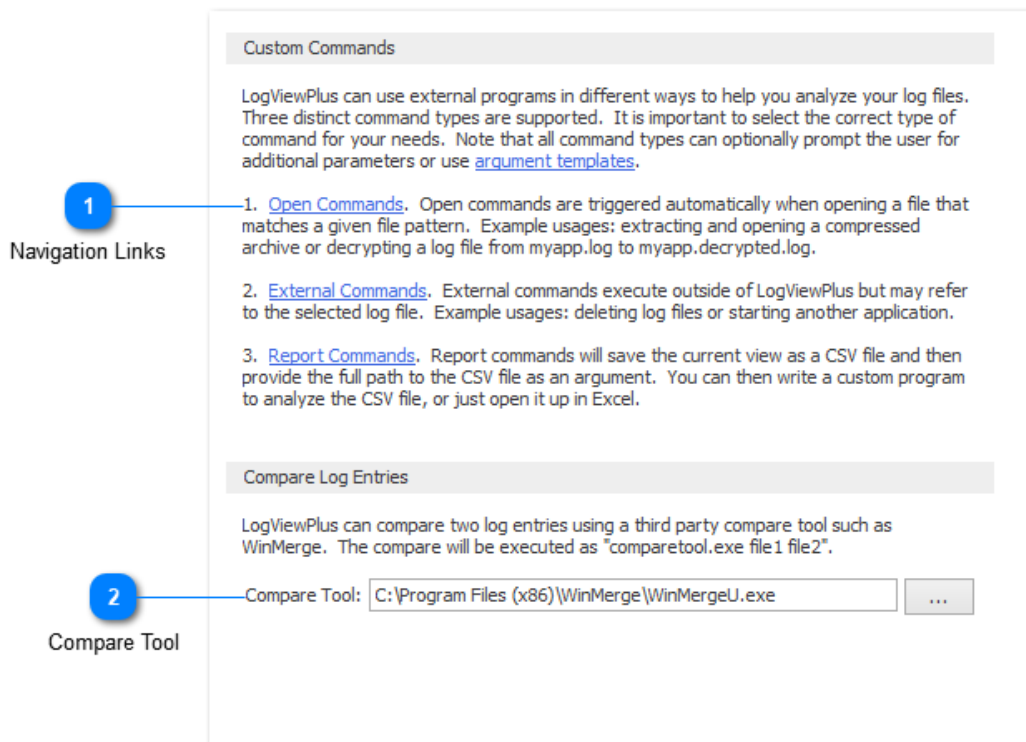
3

Syntax Highlights

Keyword	 30, 144, 255
Attribute	 218, 112, 214
Symbol	 178, 34, 34
Literal	 184, 134, 11
Quoted	 72, 61, 139
Comment	 0, 128, 0

The syntax highlights grid is used to assign a particular color to a given syntax element. The syntax elements shown will depend on the target syntax.

Commands



Commands allows you to tell LogViewPlus about a batch file you would like to execute from within LogViewPlus. LogViewPlus supports several different types of commands which can execute in a number of different scenarios depending on the use case.

1 Navigation Links

1. [Open Commands](#). Open matches a given file pattern

Navigation links are provided for your convenience when selecting the appropriate command type. The command types are also available from the settings tree when you expand the 'Commands' node.

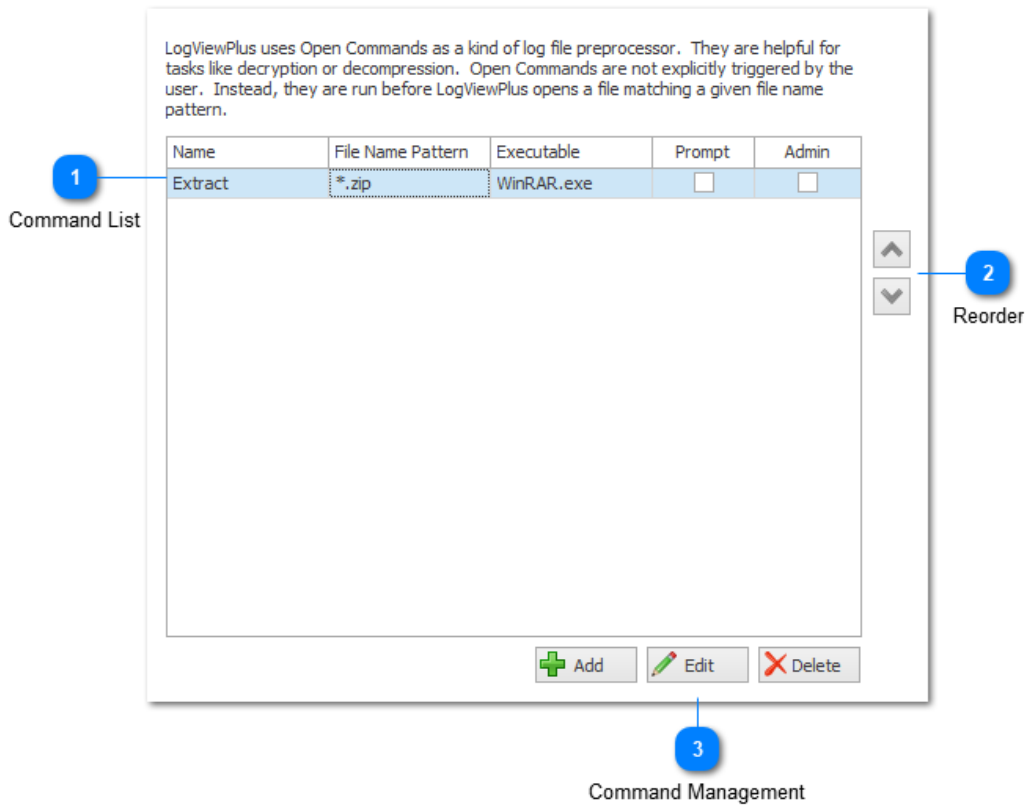
2 Compare Tool

Compare Tool:

You can optionally configure LogViewPlus to point to a local compare tool (such as WinMerge). Doing so will allow you to compare two log entries by executing the "Compare Log Entries" command on the [Log Entries](#) menu.

The tool configured must support a command line with the signature "cmd.exe Arg1 Arg2" where cmd.exe is the configured process and arg1 and arg2 are the files to be compared.

Open Commands



Open commands are used to override the default LogViewPlus open behavior. If the filename pattern of the open command is matched, then the given command will be executed before LogViewPlus opens a log file. Open commands may instruct LogViewPlus to ignore the file being opened in favor of another file or workspace.

Open commands can be useful for tasks like extracting zip files and opening the contents.

1 Command List

Name	File Name Pattern	Executable	Prompt	Admin
Extract	*.zip	WinRAR.exe	<input type="checkbox"/>	<input type="checkbox"/>

The commands grid shows the list of currently configured open commands. This grid is read-only.

2

Reorder



The ordering command can be used to change the order of the open commands. Changing the order of the commands will change the order in which the file name patterns will be scanned. LogViewPlus will execute the first open command with a matching filename pattern.

3

Command Management



The command management buttons located at the bottom of the view can be used to add, edit and delete open command configurations.

Open Settings

The screenshot shows the 'New Open Command' dialog box with the following fields and sections:

- 1 Command Settings:** Points to the 'Arguments' text area, which contains the placeholder text `${TARGET_PATH}`.
- 2 Open Trigger:** Points to the 'File Open Trigger Action' section, which includes a 'File Name Pattern' text box and an 'Is Regex?' checkbox.
- 3 Post Execution:** Points to the 'Post Execution Action' section, which includes an 'Action' dropdown menu (currently showing 'Open a different file in the target file directory') and an 'Output File Name' text box containing the placeholder `${TARGET_NAME}.extract.${TARGET_EXTENSION}`.
- 4 Save / Cancel:** Points to the 'Save' and 'Cancel' buttons at the bottom right of the dialog.

Other fields in the dialog include 'Friendly Name', 'Executable' (with a browse button), and 'Run Directory'.

Open commands are used to override the default LogViewPlus open behavior. Open commands can be configured in the open settings dialog discussed below.

1 Command Settings

The screenshot shows a configuration window with the following elements:

- Friendly Name:** A text input field.
- Executable:** A text input field with a browse button (three dots) to its right.
- Run Directory:** A text input field.
- Arguments:** A large text area containing the placeholder text `${TARGET_PATH}`. It has a vertical scrollbar on the right.
- Options:** Two checkboxes at the bottom:
 - ☐ Prompt for arguments before executing?
 - ☐ Run process with Admin credentials?

The basic command settings are:

1. **Friendly Name:** The friendly name is a display name that can be associated with this command configuration. The friendly name helps you easily identify the command.
2. **Executable:** The full path to the target executable or batch file to be run.
3. **Run Directory:** If the process you are trying to start needs to be started from a specific directory, you can optionally configure the run directory.
4. **Arguments:** Arguments are the parameters used when starting the target application. LogViewPlus also supports a small set of [optional argument templates](#).
5. **Prompt for arguments:** If the arguments used to start your target application change frequently you can optionally configure LogViewPlus to prompt you. This will give you the opportunity to change the arguments before the target application is executed.
6. **Run as Admin:** If required, you can optionally configure LogViewPlus to run the target application as Admin.

File Open Trigger Action

File Name Pattern:
☐ Is Regex?

The open trigger is the filename pattern which is used when opening a log file to determine if a given open command should be executed. By default, the file name can contain a wildcard character - an asterisk *. This character will match one or more other characters. For example, say the name of your log file has a date suffix, something like MyApp_12.2.16.log. In this case, you could match the date part of the filename using the asterisk wildcard. Your pattern might be **MyApp_*.log**.

Usually, the asterisk wildcard is sufficient, but if you find you need more power to match your log file name, then you can use a regular expression instead. To do this, check the "Is Regex?" checkbox to let LogViewPlus know the search needs to be performed differently.

3

Post Execution

Post Execution Action

Action:

Output File Name:

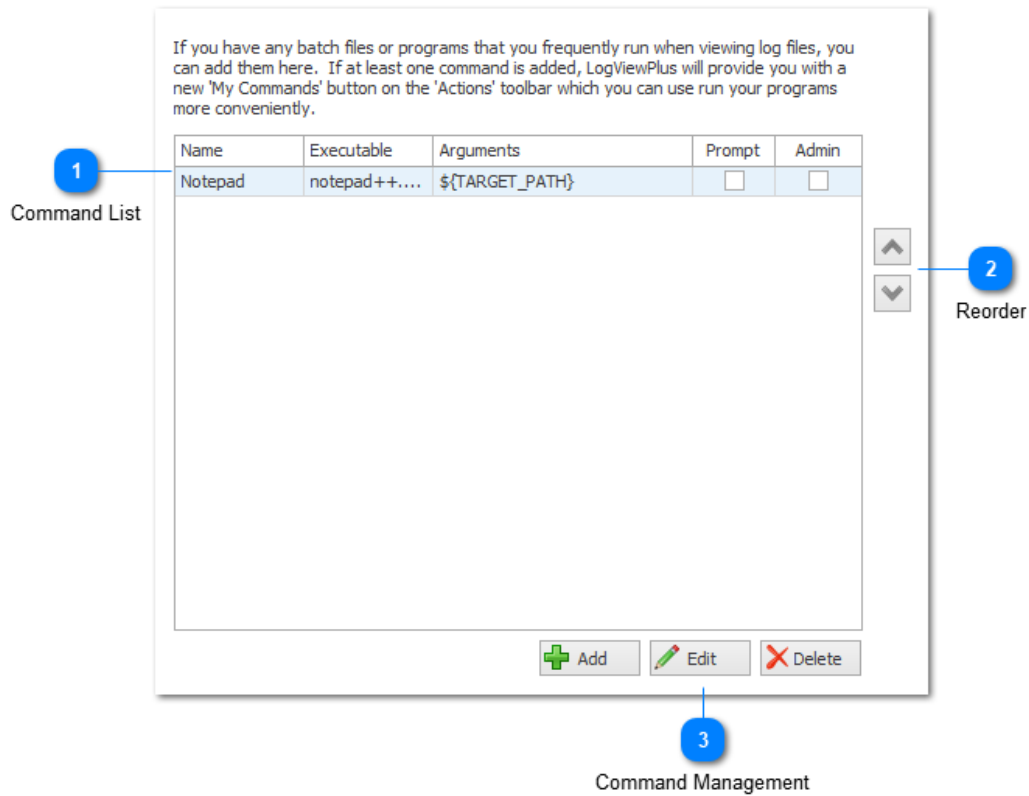
Determines the action to be taken once the target program has completed execution. You can choose to open a file in the target directory or the temp directory. Alternatively you can open a predefined workspace. LogViewPlus also supports a small set of [optional argument templates](#).

4

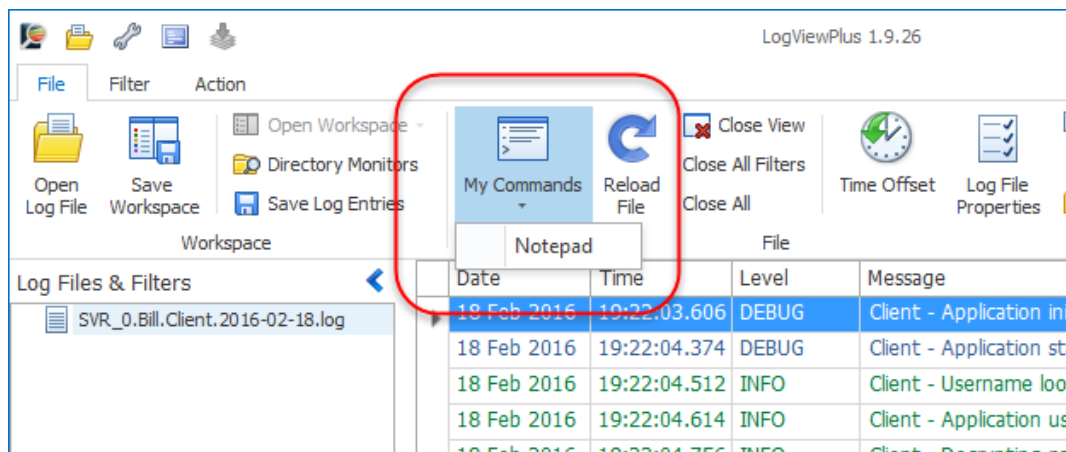
Save / Cancel

Once you are done configuring your command you will need to save your settings. Alternatively, if you're not happy with your command configuration, you can cancel it.

External Commands



Once at least one external command has been configured, you will see a 'My Commands' button listed on the toolbar under the file menu. This command will list all configured commands. When a configured command is selected, it will be executed against the currently selected log file:



External commands can be used for tasks like stopping and starting an external process or cleaning up an external directory. External commands allow you to execute a batch file without leaving LogViewPlus.

1 Command List

Name	Executable	Arguments	Prompt	Admin
Notepad	notepad++....	\${TARGET_PATH}	<input type="checkbox"/>	<input type="checkbox"/>

The commands grid shows the list of currently configured external commands. This grid is read-only.

2 Reorder



The ordering command can be used to change the display order of the external commands.

3 Command Management



The command management buttons located at the bottom of the view can be used to add, edit and delete external command configurations.

External Settings

The image shows a 'New External Command' dialog box with the following fields and options:

- Friendly Name:** A text input field.
- Executable:** A text input field with a browse button (three dots).
- Run Directory:** A text input field.
- Arguments:** A large text area for entering command arguments.
- ☐ Prompt for arguments before executing?
- ☐ Run process with Admin credentials?
- Post Execution Open Workspace:** A section containing a 'Workspace:' dropdown menu currently set to 'None'.
- Buttons:** 'Save' and 'Cancel' buttons at the bottom right.

Numbered callouts on the left side of the dialog:

- 1** Command Settings (points to the main configuration area)
- 2** Post Execution (points to the 'Post Execution Open Workspace' section)
- 3** Save / Cancel (points to the 'Save' and 'Cancel' buttons)

External commands allow you to execute a batch file without leaving LogViewPlus. External commands can be configured in the external settings dialog discussed below.

1 Command Settings

The image shows a configuration window for LogViewPlus. It contains four input fields: 'Friendly Name' (a single-line text box), 'Executable' (a single-line text box with a browse button '...' to its right), 'Run Directory' (a single-line text box), and 'Arguments' (a multi-line text area with a vertical scrollbar). Below these fields are two checkboxes: 'Prompt for arguments before executing?' and 'Run process with Admin credentials?'.

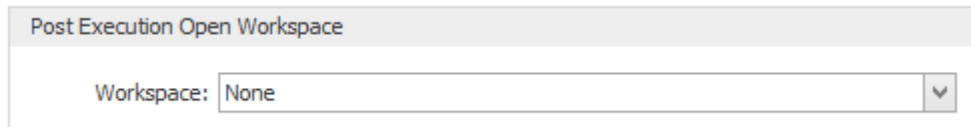
The basic command settings are:

1. **Friendly Name:** The friendly name is a display name that can be associated with this command configuration. The friendly name helps you easily identify the command.
2. **Executable:** The full path to the target executable or batch file to be run.
3. **Run Directory:** If the process you are trying to start needs to be started from a specific directory, you can optionally configure the run directory.
4. **Arguments:** Arguments are the parameters used when starting the target application. LogViewPlus also supports a small set of [optional argument templates](#).
5. **Prompt for arguments:** If the arguments used to start your target application change frequently you can optionally configure LogViewPlus to prompt you. This will give you the opportunity to change the arguments before the target application is executed.

6. **Run as Admin:** If required, you can optionally configure LogViewPlus to run the target application as Admin.

2

Post Execution

A dialog box titled "Post Execution Open Workspace". It contains a label "Workspace:" followed by a text input field containing the word "None" and a small downward-pointing arrow icon on the right side of the field.

Post Execution Open Workspace

Workspace: None

Post execution settings allow you to configure a pre-saved workspace to be opened once the external command has completed. Configuring a workspace is optional.

3

Save / Cancel

Two rectangular buttons with rounded corners. The left button is labeled "Save" and the right button is labeled "Cancel". Both buttons have a light gray background and a thin border.

Save Cancel

Once you are done configuring your command you will need to save your settings. Alternatively, if you're not happy with your command configuration, you can cancel it.

Report Commands

Report Commands will export the current view as a CSV file and then pass the CSV file path as an argument to a batch file. If at least one command is added, LogViewPlus will provide you with a new 'My Reports' button on the log file context menu.

Name	Executable	Arguments	Prompt	Admin
Excel	EXCEL.EXE	\${REPORT_PATH}	<input type="checkbox"/>	<input type="checkbox"/>

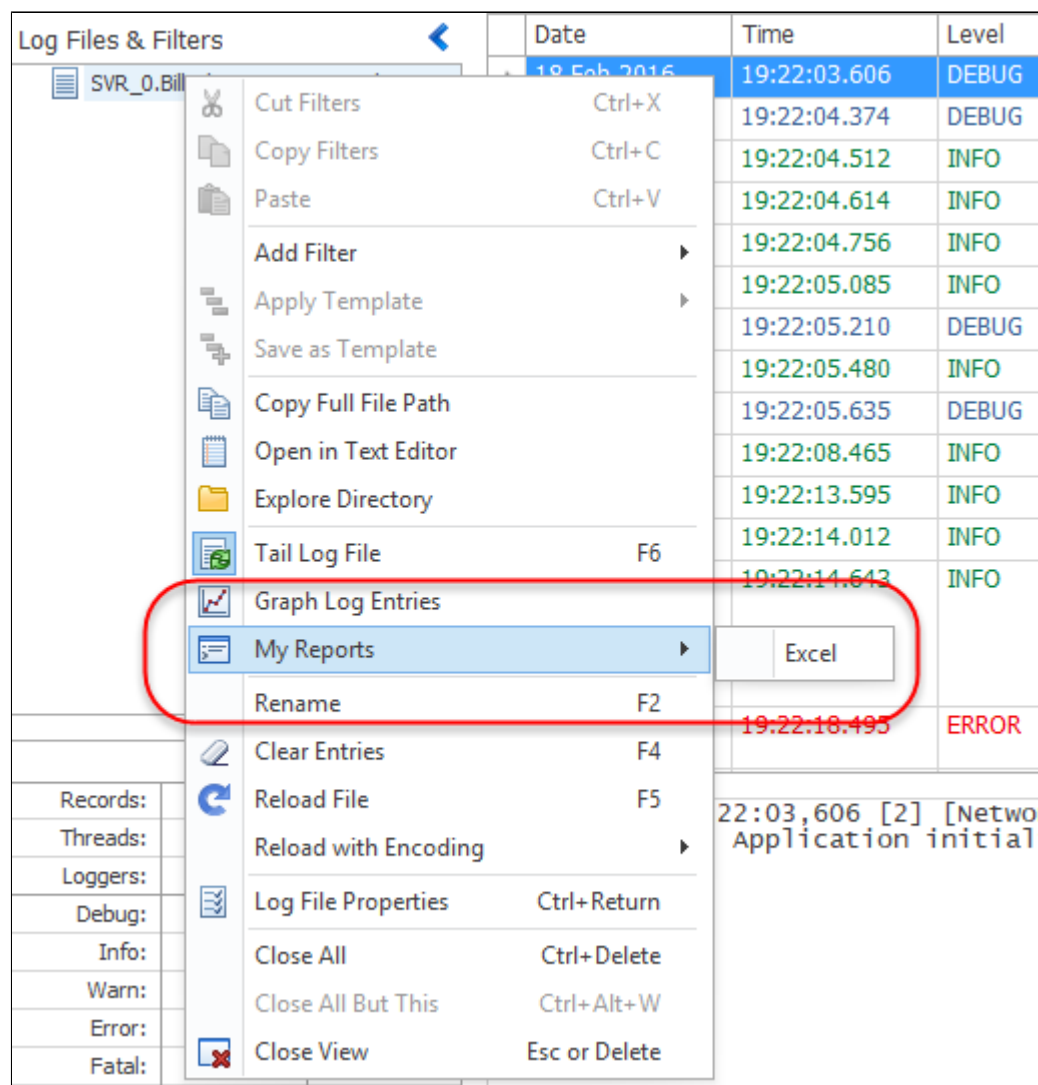
1 Command List

2 Reorder

3 Command Management

Buttons: Add, Edit, Delete

Once at least one report command has been configured, you will see a 'My Reports' button listed on log file context menu. This command will list all configured report commands. When a report command is selected, it will be executed against the currently selected log file.



Report commands are available from the log file and filter context menu.

1 Command List

Name	Executable	Arguments	Prompt	Admin
Excel	EXCEL.EXE	\${REPORT_PATH}	<input type="checkbox"/>	<input type="checkbox"/>

The commands grid shows the list of currently configured report commands. This grid is read-only.

2

Reorder



The ordering command can be used to change the display order of the report commands.

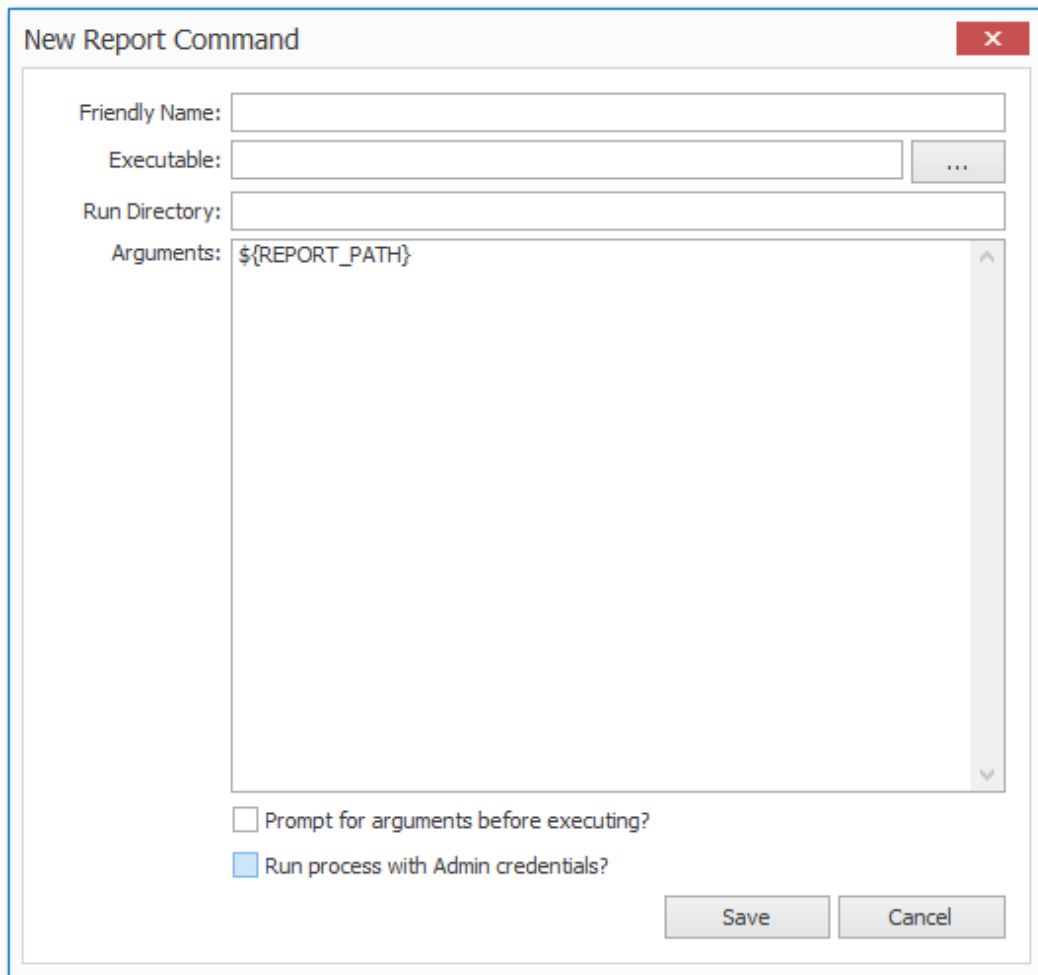
3

Command Management



The command management buttons located at the bottom of the view can be used to add, edit and delete report command configurations.

Report Settings



The image shows a Windows-style dialog box titled "New Report Command". It has a standard title bar with a close button (X) in the top right corner. The dialog contains several input fields and checkboxes. The "Friendly Name:" field is a simple text box. The "Executable:" field is a text box followed by a button with three dots (...). The "Run Directory:" field is a text box. The "Arguments:" field is a large text area containing the text "\${REPORT_PATH}" and a vertical scrollbar on the right. At the bottom, there are two checkboxes: "Prompt for arguments before executing?" (unchecked) and "Run process with Admin credentials?" (checked). Below the checkboxes are two buttons: "Save" and "Cancel".

New Report Command

Friendly Name:

Executable: ...

Run Directory:

Arguments:

☐ Prompt for arguments before executing?

☒ Run process with Admin credentials?

Save Cancel

Before executing a report command, LogViewPlus will export the currently selected log file as a CSV file. The CSV file export will be provided as an argument to the report command.

A report command might be used for tasks like opening a log file in Excel.

The report command settings are:

1. **Friendly Name:** The friendly name is a display name that can be associated with this command configuration. The friendly name helps you easily identify the command.

2. **Executable:** The full path to the target executable or batch file to be run.
3. **Run Directory:** If the process you are trying to start needs to be started from a specific directory, you can optionally configure the run directory.
4. **Arguments:** Arguments are the parameters used when starting the target application. LogViewPlus also supports a small set of [optional argument templates](#).
5. **Prompt for arguments:** If the arguments used to start your target application change frequently you can optionally configure LogViewPlus to prompt you. This will give you the opportunity to change the arguments before the target application is executed.
6. **Run as Admin:** If required, you can optionally configure LogViewPlus to run the target application as Admin.

Argument Templates

Argument templates are a set of predefined text substitution commands which can be used when providing arguments to a LogViewPlus command or parser. Argument templates are written with the syntax **\${TEMPLATE}**. When an argument template is found, LogViewPlus will replace the template with its in memory value. This allows LogViewPlus to provide arguments such as "the currently selected log file" to the target process.

LogViewPlus supports the following argument templates. Some templates may be dependent on the type of command being executed. For example the **\${REPORT_FILE}** template is specific to report command execution.

Argument templates can be divided into three groups:

Target - Represents the log file currently being opened.

\${TARGET_PATH} - The full path to the target file.

\${TARGET_DIRECTORY} - The full path of the directory containing the target file.

\${TARGET_NAME} - The file name of the target file.

\${TARGET_EXTENSION} - The file name extension of the target file. Note that the extension will not be prefixed with a period. An example extension might be "log" for any *.log file.

Report - Represents the CSV file which was produced prior to the report command execution.

\${REPORT_PATH} - The full path to the report file.

\${REPORT_DIRECTORY} - The full path of the directory containing the report file.

\${REPORT_NAME} - The file name of the report file.

\${REPORT_EXTENSION} - The file name extension of the report file. Note that the extension will not be prefixed with a period. An example extension might be "log" for any *.log file.

Current - Represents the currently selected log file.

\${CURRENT_PATH} - The full path to the currently selected log file.

\${CURRENT_DIRECTORY} - The full path of the directory containing the currently selected log file.

\${CURRENT_NAME} - The file name of the currently selected log file.

\${CURRENT_EXTENSION} - The file name extension of the currently selected log file. Note that the extension will not be prefixed with a period. An example extension might be "log" for any *.log file.

\${CURRENT_SELECTION} - This argument template will save each selected log entry as a temporary file. The full path to the temporary file will then be inserted into the template. If multiple log entries are selected, then multiple paths will be inserted into the template.

\${CURRENT_MESSAGES} - This template is similar to CURRENT_SELECTION, but rather than saving the entire log entry to a temporary file, only the log message (as parsed by LogViewPlus) will be saved to the temporary file.

Encoding & Culture

1 Default Encoding

2 Additional Cultures

Default Encoding

When opening a log file, LogViewPlus will attempt to identify the encoding by detecting a byte order mark (BOM). However, some files do not have a BOM, so a default encoding is needed. LogViewPlus uses the Windows system encoding by default.

☒ Windows default. Currently Windows-1252.

☐ UTF-8

☐ Other:

Additional Timestamp Cultures

Log file timestamps may be culture specific. Your local culture and English are supported by default. You can use the table below to include support for additional cultures. Changes to this setting require an application restart.

Id	Culture
de-DE	German (Germany)

When opening a log file, LogViewPlus will attempt to identify the file's encoding by detecting a byte order mark (BOM). However, some files do not have a BOM. In this case a default encoding will be used. LogViewPlus uses the Windows system encoding by default, but you can select a different default encoding.

1 Default Encoding

☒ Windows default. Currently Windows-1252.

☐ UTF-8

☐ Other:

The default file encoding LogViewPlus should use when opening a log file. Using the Windows default encoding is recommended.

2

Additional Cultures

Id	Culture
de-DE	German (Germany)


+ AddX Delete

By default, LogViewPlus will support your current Windows culture as well as English. These cultures can be used when parsing log files. For example, date timestamps are often written in a manner which is culture specific.

The additional cultures list can be used to expand the list of cultures used when parsing log files.

Encoding Mapping

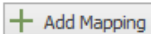
When opening a log file, LogViewPlus will use the file encoding mappings defined below to override the default encoding.

File Name Pattern	Regex	Encoding	Delete?
*.log	<input type="checkbox"/>	Windows-1252	

1 Encoding Mappings

Create New Mapping

2 File Name Pattern: ☐ Is RegEx?


3 Encoding: 

File Pattern

Encoding

Sometimes, you may want to open the log file with encoding that is not the default encoding. In this case, you can use the 'Encoding Mapping' settings to create a mapping between a given log file name and the type of encoding should be used when reading that all file.

1 Encoding Mappings

File Name Pattern	Regex	Encoding	Delete?
*.log	<input type="checkbox"/>	Windows-1252	

The encoding mapping grid contains a list of all current log file encodings. You can delete an encoding by clicking on the delete command on the far right of the grid.

2

File Pattern

File Name Pattern: ☐ Is RegEx?

Use the file name pattern text box to specify a pattern which can be used to match the filenames of your application log file. By default, the file name can contain a wildcard character - an asterisk *. This character will match one or more other characters. For example, say the name of your log file has a date suffix, something like MyApp_12.2.16.log. In this case, you could match the date part of the filename using the asterisk wildcard. Your pattern might be **MyApp_*.log**.

Usually, the asterisk wildcard is sufficient, but if you find you need more power to match your log file name, then you can use a regular expression instead. To do this, check the "Is RegEx?" checkbox to let LogViewPlus know the search needs to be performed differently.

3

Encoding

Encoding:

The encoding drop-down box can be used to select the encoding that should be applied to any files which match the given filename pattern. This drop-down box will provide a list of all encodings currently available in Windows.

Once you have selected the encoding that you want to use for your file name pattern, you can click the Add Mapping command to save your changes.

Common Encodings

This list of commonly used encodings is used when opening a log file and when reloading a log file with the "Reload File with Encoding..." context menu.

Common Encoding	Delete?
Windows Default (Windows-1252)	
utf-8	
Windows-1252	X

Add Common Encoding

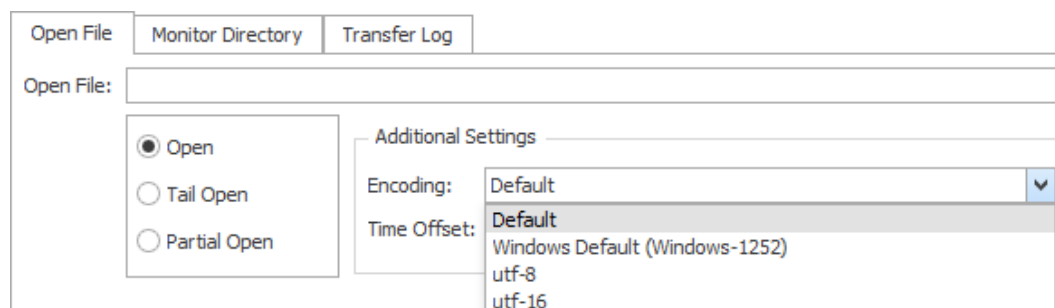
Encoding: + Add Encoding

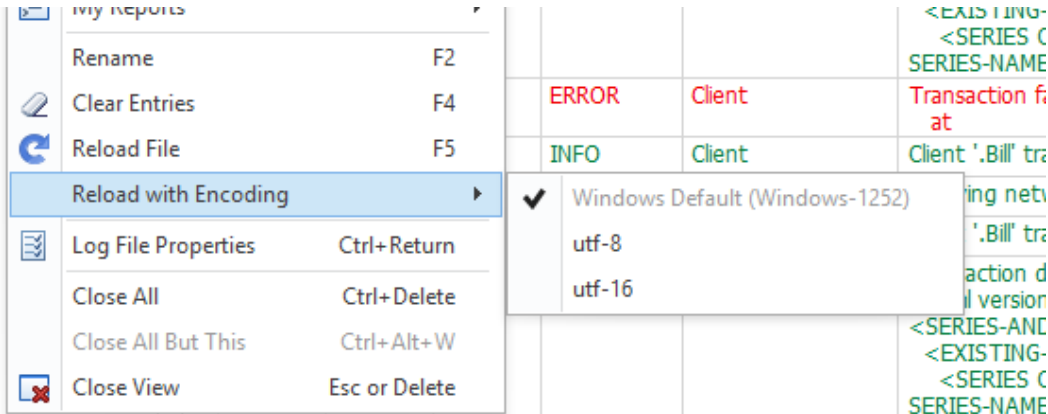
1 Available Encodings

2 Encoding

Common Encoding settings allows you to quickly set the encoding for a log file without being overloaded with encoding options. In the example screenshots, the encoding 'utf-16' has been added in addition to the defaults. Additional encodings can be removed by clicking the red 'X' icon in the 'Delete' column. Default encodings cannot be deleted.


All common encodings defined here will be listed under the 'Encoding' setting when opening log files.



Also, if you right-click on a log file, you'll notice a command which is called 'Reload with Encoding...'.


This command is useful if you have loaded a log file in the wrong encoding and want to quickly reload the file using a different encoding. The encodings available in this pop-up menu are configured in the 'Reloading Encodings' settings.

1 Available Encodings

Common Encoding	Delete?
Windows Default (Windows-1252)	
utf-8	
Windows-1252	

The available encodings grid shows a list of encodings which will currently be displayed in the 'Reload File with Encoding...' pop-up. You can delete an encoding by clicking on the delete command on the far right of the grid.

2 Encoding

Encoding:  

The encoding drop-down box can be used to select the encoding that should be displayed in the 'Reload File with Encoding...' pop-up. This drop-down box will provide a list of all encodings currently available in Windows.

Once you have selected the encoding that you want to use for your file name pattern, you can click the Add Encoding command to save your changes.

About

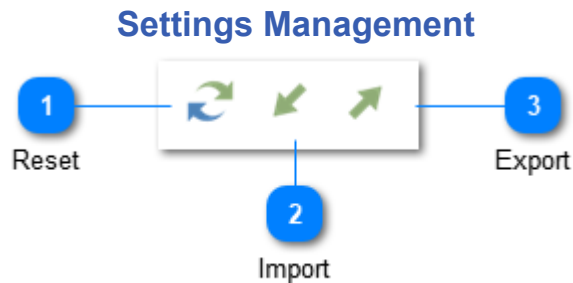
— Version —

LogViewPlus
Version: 1.9.26.0
Copyright © 2017 Clearcove Limited

— Registration —

This product is currently unregistered.
0 trial days remaining.

The About settings page gives you information about the current version of LogViewPlus as well as your registration status. If you are running a licensed version of LogViewPlus your license key will be shown here.



In the bottom left corner of the LogViewPlus Settings window are three blue commands which can be used to manage your settings: Reset, Import and Export.

1 Reset



The Reset command will delete all of your application settings and revert LogViewPlus back to the installation default settings. We recommend exporting all of your existing settings before using this command.

2 Import



The Import command can be used to import settings that have been previously saved with the 'Export' command.

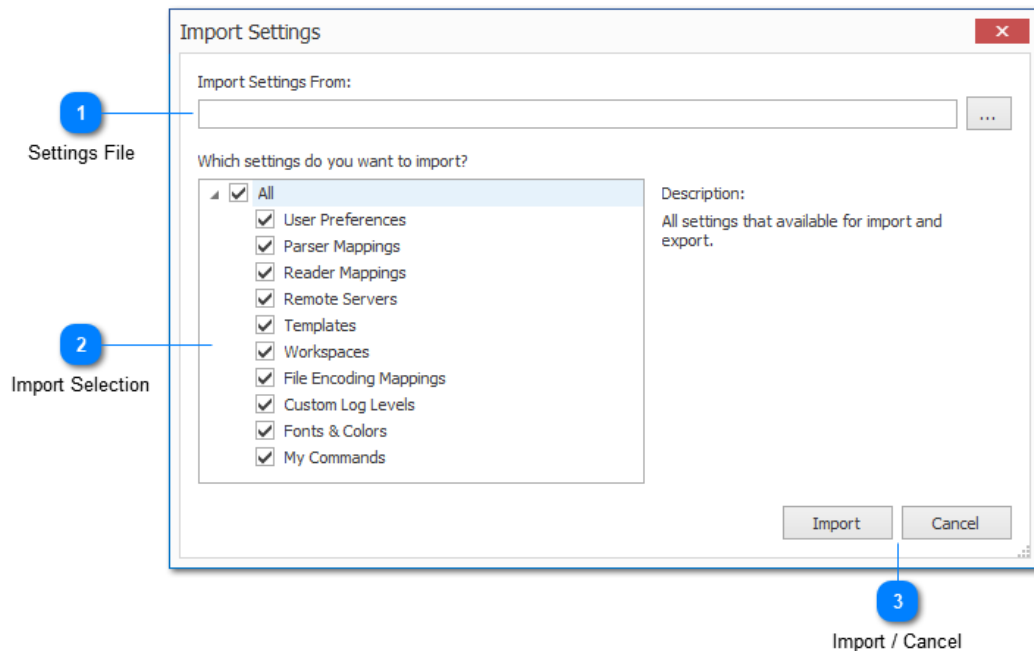
3 Export



The Export command can be used to save your current settings to an XML file.

Some elements in this XML file can be modified manually (such as parser configurations) while others are base64 encoded and should not be edited.

Import Settings



The Import Settings dialog can be used to import previously exported LogViewPlus settings.

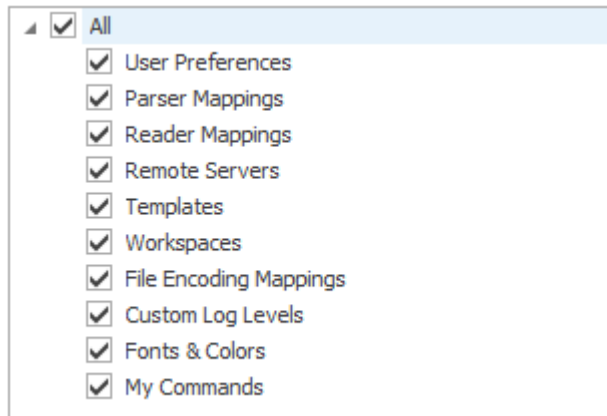
The dialog provides the ability to import settings on a granular level. This gives you more flexibility when managing your settings.

1 Settings File

Import Settings From:

The full path to the previously exported settings file. There is a command button on the far right of the window which can be used to browse for the file.

2 Import Selection



The import selection check boxes allow you to choose which settings to import. Checked categories will be completely replaced by imported settings. Categories which are unchecked will not be modified.

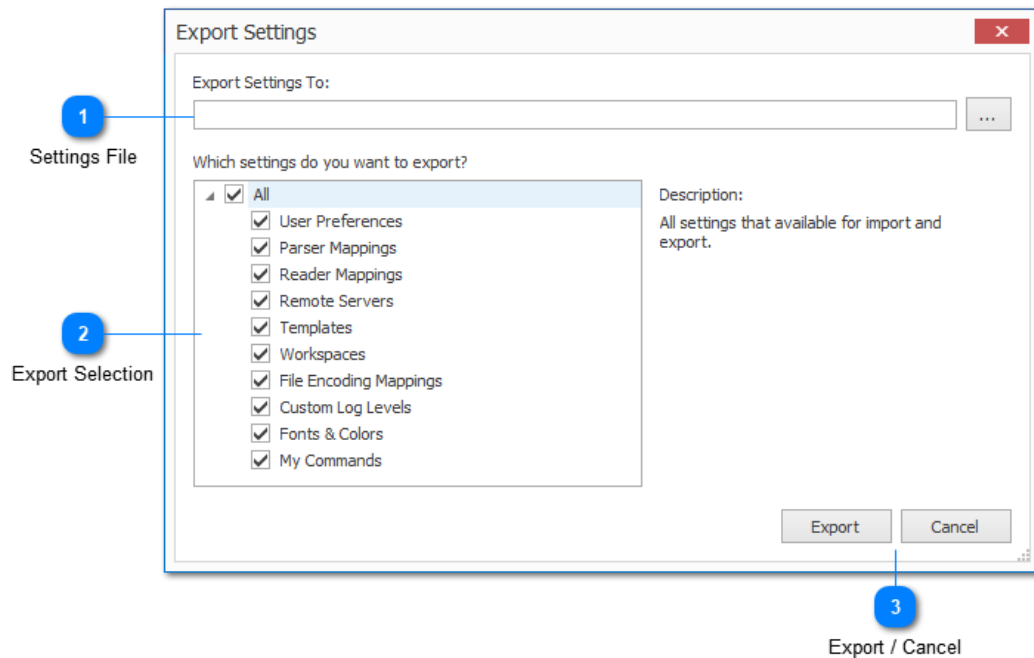
If the exported file you are trying to import does not have a particular category, then that category check box will have no impact on the import process.

3 Import / Cancel



The Import command can be used to commit the operation. Use the cancel command if you do not want to execute the import.

Export Settings



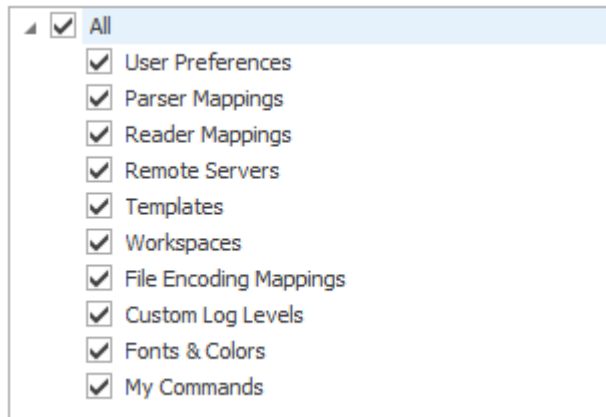
The Export Settings dialog can be used to save your LogViewPlus settings. The dialog provides the ability to export settings on a granular level. This gives you more flexibility when managing your settings or sharing settings with other users.

1 Settings File

Export Settings To:

The full path to the settings file you want to create. If this file already exists, it will be replaced on export. There is a command button on the far right of the window which can be used to browse for the file.

2 Export Selection



The export selection check boxes allow you to choose which settings to export.

Categories which are unchecked will not be exported. This is helpful when you want to share some settings with other users, but keep other settings private.

3

Export / Cancel



The Export command can be used to save your new settings file. If the file already exists, it will be replaced. Use the cancel command if you do not want to export your settings.

Data Access Settings

File system settings are used to manage the LogViewPlus file browser. This controls how you interact with file systems when using LogViewPlus.

LogViewPlus can access remote filesystems in one of four ways. SFTP, FTP, and SCP access are all managed in the [remote server's](#) configuration. Windows or SMB shares are managed in the [network shares](#) configuration.

In addition, LogViewPlus can manage a number of different [data stores](#) in the Log File Browser. These data stores are not file systems, but they appear in the LogViewPlus file system and are therefore covered in this section. LogViewPlus can support a number of different data stores including [Windows Event Logs](#), [Databases](#), [Syslog](#), [UDP](#), and [Event Tracing for Windows](#) (ETW).

Remote Servers

Connect to a remote server with SFTP or FTP.

Name ▲	Type	Server	User
My Server 1	SFtp	localhost	tester
My Server 2	SFtp	localhost	tester

1 Server List

2 Add 3 Clone 4 Edit 5 Delete

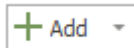
The remote server's configuration settings displays a list of all configured SFTP, FTP, SCP and HTTPS servers.

1 Server List

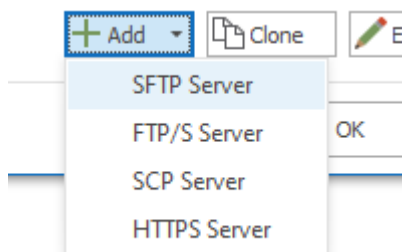
Name ▲	Type	Server
My Server 1	SFtp	localhost
My Server 2	SFtp	localhost

The server list shows all of the currently configured SFTP, FTP, and SCP servers in the order in which they will be displayed by the LogViewPlus file browser.

2 Add

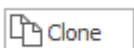


When you click to add a server you will be presented with the drop-down menu displayed below.



The add server drop-down is used to determine the type of server that you want to configure. Based on the server type selected you will be presented with either the [SFTP](#), [FTP](#), [SCP](#) or [HTTPS](#) configuration screen.

3 Clone



When cloning a server LogViewPlus will open the currently selected server configuration. You can then make changes to the server configuration and save your changes as a new server.

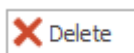
4 Edit



The edit command opens the appropriate server configuration form for the currently selected server configuration. Saving changes will overwrite the current server settings.

5

Delete



The delete command can be used to remove the currently selected server configuration.

SFTP Servers

The screenshot shows the 'Add SFTP Server' dialog box with the following numbered callouts:

- 1: Navigation (points to the 'Basic' and 'Advanced SFTP' tabs)
- 2: Friendly Name (points to the 'Friendly Name' text input field)
- 3: Friendly Category (points to the 'Friendly Category' dropdown menu)
- 4: Server Name (points to the 'Server Name' text input field)
- 5: Proxy (points to the 'Proxy' dropdown menu)
- 6: Port (points to the 'Port' text input field)
- 7: User Name (points to the 'User Name' text input field)
- 8: Password (points to the 'Password' text input field)
- 9: Show (points to the 'Show' checkbox)
- 10: Test (points to the 'Test' button)
- 11: Save / Cancel (points to the 'Save' and 'Cancel' buttons)

Configuring an SFTP server allows LogViewPlus to browse open and monitor remote log files. The remote server must have SFTP installed and running.

When connecting to an SFTP server for the first time, LogViewPlus will assume that the server's certificate fingerprint is valid and you will not be prompted to accept the certificate. If the certificate fingerprint changes on future connection attempts, you will be prompted to accept the change before your username and password is sent to the server.

1 Navigation

The image shows the 'Basic' and 'Advanced SFTP' tabs at the top of the dialog box. The 'Basic' tab is currently selected.

Navigating between basic and advanced settings is controlled by the tabs at the top of the screen. For most configuration scenarios advanced settings will not be needed.

2 Friendly Name

The image shows the 'Friendly Name' text input field, which is currently empty.

A friendly name is simply a name for the server it is easy to remember. If a friendly name is not provided, the server name will be used instead.

3

Friendly Category

Friendly Category:

A friendly category is a group name where this server can be categorized. This field is not required.

If you have previously configured a server with a friendly category, this category will be available as a drop-down option. Alternatively you can type a new name into the category text box.

4

Server Name

Server Name:

The name of the server which is hosting the SFTP process. This can either be a host name or an IP address.

5

Proxy

Proxy:

If you connect to this server through proxy, you will need to select the proxy. This option will only be available if you have preconfigured your [proxy server](#).

6

Port

Port:

The port number where the service is listening.

7

User Name

User Name:

The username that should be used to authenticate with the service.

8

Password

Password:

The password associated with the given username.

9

Show

☐ Show

Selecting the show text box will display the password in plain text. This option will only be available when adding a new server.

10

Test

Test

The test command can be used to verify the currently configured server. To do this, a connection with the server will be established. No other actions will be executed.

11

Save / Cancel

Save

Cancel

Once you have configured your server, you can use the save command to persist your changes. Once your changes have been saved the configured server will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

Advanced SFTP

1 Single Sign-On

2 SSO Domain

3 Sudo start SFTP

4 Private Key Path

5 Key Password

6 Test

7 Save / Cancel

The advanced SFTP configuration options provide alternative authentication mechanisms. This may be helpful if you are SFTP server requires advanced authentication.

1 Single Sign-On

Single Sign-On:

Single sign-on allows you to use Kerberos or NTLM authentication when connecting to the remote server. This option may be helpful if both the client and server or configured for domain trust.

2 SSO Domain

SSO Domain:

The domain used for single sign-on. This option is needed if you are using single sign-on with a username and password. In this case, you need to provide the domain associated with the user.

3

Sudo start SFTPSudo start SFTP:

A single command that can be executed post login.

The purpose of this command is to allow you to change user before the file transfer session starts. This allows you to login to the server with one account and use another account for file access. For example: `sudo su USER -c /usr/lib/sftp-server`

You will need to provide a path that is specific to your sftp-server. Common examples include: `/bin/sftp-server`, `/usr/lib/sftp-server` and `/usr/libexec/openssh/sftp-server`.

4

Private Key PathPrivate Key Path:

You can use public private key authentication for communication with the server. To do this, you must provide the full path to your private key.

5

Key PasswordKey Password:

The password for the private key provided.

6

Test

The test command can be used to verify the currently configured server. To do this, a connection with the server will be established. No other actions will be executed.

7

Save / Cancel

Once you have configured your server, you can use the save command to persist your changes. Once your changes have been saved the configured server should be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

FTP Servers

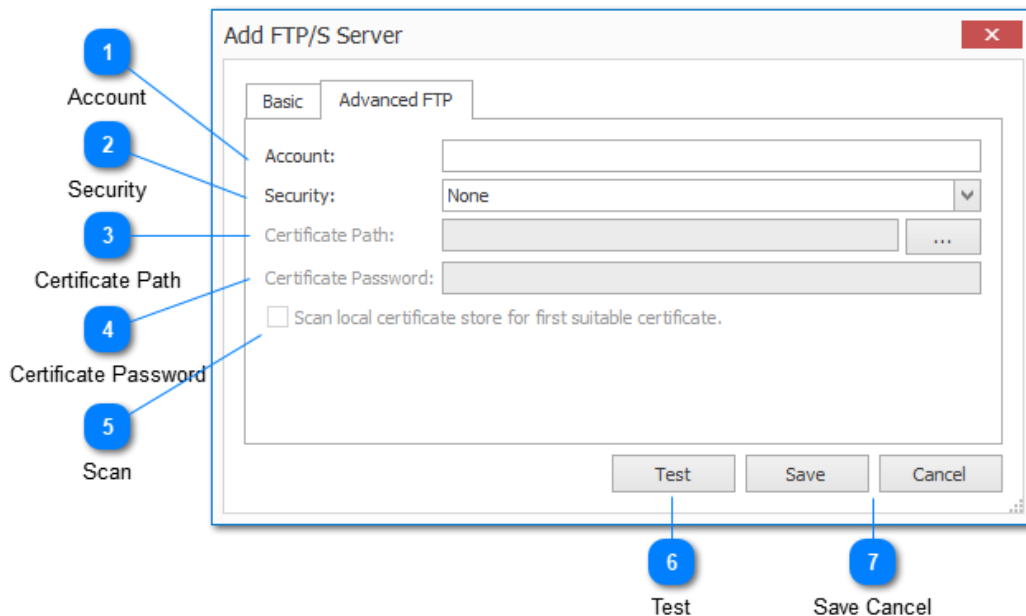
The screenshot shows a window titled "Add FTP/S Server" with a close button (X) in the top right corner. Inside the window, there are two tabs: "Basic" and "Advanced FTP". The "Basic" tab is selected. The form contains the following fields and controls:

- Friendly Name:** A text input field.
- Friendly Category:** A dropdown menu.
- Server Name:** A text input field.
- Port:** A text input field containing the value "21".
- Proxy:** A dropdown menu set to "None".
- User Name:** A text input field.
- Password:** A text input field.
- Show:** A checkbox next to the password field.
- Buttons:** "Test", "Save" (highlighted with a dashed border), and "Cancel".
- Link:** A blue hyperlink that says "Find out more about configuring servers".

Configuring an FTP server allows LogViewPlus to browse open and monitor remote log files. The remote server must have FTP installed and running.

For description of the basic fields used to configure an FTP server, please see the [SFTP server](#) documentation. For a basic configuration, there is no difference between the fields used to configure SFTP vs FTP.

Advanced FTP



The advanced FTP configuration options provide alternative authentication mechanisms. This may be helpful if you are FTP server requires advanced authentication.

1 Account

Account:

If your FTP servers may requires an account name in addition to the username and password it can be provided as an advanced setting.

2 Security

Security: ▼

Determines whether implicit or explicit security should be used in your FTPS connection. Selecting either implicit or explicit security will enable the certificate options discussed below.

3

Certificate Path

Certificate Path:

 ...

The certificate to be used for client certificate authentication.

4

Certificate Password

Certificate Password:

The password for the certificate provided.

5

Scan

☐ Scan local certificate store for first suitable certificate.

If you are using FTPS you may choose to scan the client's certificate store to find an appropriate certificate rather than using a file system certificate.

6

Test

The test command can be used to verify the currently configured server. To do this, a connection with the server will be established. No other actions will be executed.

7

Save Cancel

Once you have configured your server, you can use the save command to persist your changes. Once your changes have been saved the configured server should be immediately available in the folder tree view.

Use the "Cancel" command to return to LogViewPlus without saving your changes.

SCP Access

The screenshot shows the 'Add SCP Server' dialog box. It contains a text area with instructions, input fields for 'Server Name', 'Port', 'User Name', and 'Password', a 'Proxy' dropdown menu, and buttons for 'Test', 'Save', and 'Cancel'. Numbered callouts (1-7) point to specific elements: 1 points to the 'Server Name' label, 2 to the 'Port' label, 4 to the 'User Name' label, 5 to the 'Password' label, 3 to the 'Proxy' dropdown, 6 to the 'Test' button, and 7 to the 'Save' button.

1 Server Name

2 Port

4 User Name

5 Password

3 Proxy

6 Test

7 Save / Cancel

Find out more about configuring servers

If your remote server does not provide as SFTP or FTP access, you may still be able to access your log files via SCP. The SCP protocol is not recommended because it does not provide the ability to browse directories or [tail log files](#). However, in some situations it may still be a useful option.

In order to download a log file via SCP you will need to provide the full URL to the target file. For example: `scp://server:port/path/file.log`.

Configuring an SCP server in advance gives you the ability to protect the server authentication credentials. When you provide the URL to download a log file via SCP, LogViewPlus will look at the server and determine if SCP access for that server has been preconfigured. If so LogViewPlus will use the username and password provided when connecting to the remote server.

When connecting to an SCP server for the first time, LogViewPlus will assume that the server's certificate fingerprint is valid and you will not be prompted to accept the certificate. If the certificate fingerprint changes on future connection attempts, you will be prompted to accept the change before your username and password is sent to the server.

1

Server NameServer Name:


The target SCP server.

2

PortPort:

The port used for SSH access on the target server. The default port is usually 22.

3

ProxyProxy: 

If you connect to this server through proxy, you will need to select the proxy. This option will only be available if you have preconfigured your [proxy server](#).

4

User NameUser Name:

The username that should be used to authenticate with the service.

5

PasswordPassword:

The password associated with the given username.

6

Test

The test command can be used to verify the currently configured server. To do this, a connection to the server will be established. No other actions will be executed.

7

Save / Cancel



Once you have configured your server, you can use the save command to persist your changes. Once your changes have been saved the configured server should be immediately available in the folder tree view.

Use the "Cancel" command to return to LogViewPlus without saving your changes.

HTTPS Servers

The screenshot shows a dialog box titled "Add HTTPS Server" with a close button (X) in the top right corner. Inside the dialog, there is a text block explaining that files can be opened over HTTPS with a URL, and that configuration is only necessary if authentication is required. Below this text are four input fields: "Server Name:", "Port:" (with a spinner set to 443), "User Name:", and "Password:". To the right of the "Port:" field is a "Proxy:" dropdown menu currently set to "None". At the bottom right of the input fields is a link that says "Find out more about HTTPS connections". At the bottom center are two buttons: "Save" and "Cancel".

Numbered callouts point to the following elements:

- 1: Server Name
- 2: Port
- 3: Proxy Server
- 4: Login Credentials
- 5: Save Configuration

LogViewPlus can open log files using HTTP or HTTPS. To open a file over HTTP/S simply paste the URL into the [Open File](#) settings. Configuration is only necessary if your HTTPS server requires authentication or if a proxy server should be used.

The authentication method is chosen automatically based on the server response. Possible options include: NTLM, Kerberos, Negotiate, Digest, and Basic.

Refreshing a log file in LogViewPlus will cause the file to be re-downloaded from the server.

Authentication over HTTP is insecure and not supported. Directory browsing and tail are not currently supported over HTTP/S.

1 Server Name

Server Name:

The name of the target HTTPS server without the protocol. For example: mycompany.com or www.mycompany.com.

2 Port

Port:

The port which will be used by the HTTPS server.

3 Proxy Server

Proxy:

The [proxy server configuration](#) to use. This property will be disabled if the proxy server has not been configured. The options available are None and Default.

4 Login Credentials

User Name:
Password:

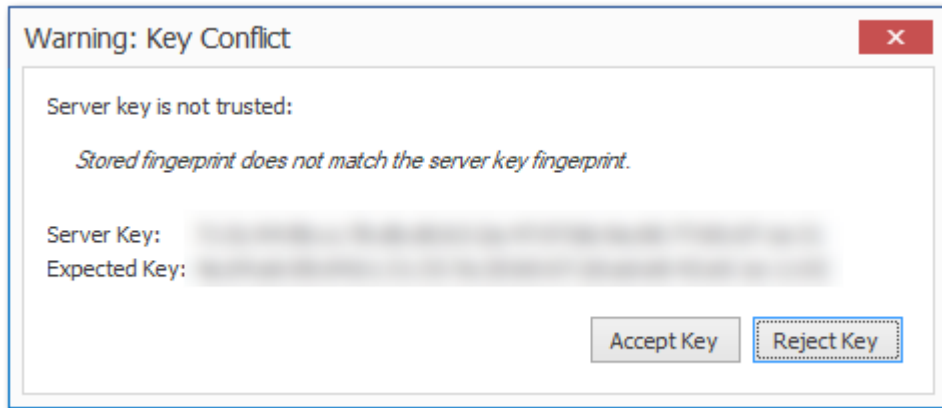
The login credentials to use if credentials are requested by the server.

5 Save Configuration

Once you have configured your server, you can use the save command to persist your changes.

Use the "Cancel" command to return to LogViewPlus without saving your changes.

Server Fingerprints



Secure servers will frequently use security fingerprints to identify themselves. When a client connects, the server presents a fingerprint and this gives the client a chance to disconnect before sending through more sensitive details like username and password.

When LogViewPlus first connects to a secure server, it has nothing to compare the fingerprint against. The default behavior in this scenario is that LogViewPlus will assume the fingerprint is valid. We take a slightly less secure approach here in favor of usability because we believe that most connections will be made behind a firewall where the initial connection can be trusted. If you would like to verify the fingerprint before connecting, we recommend using a 3rd party tool (such as Putty) immediately before the initial connection attempt. Alternatively, you can set the Lockdown property to true in %ProgramFiles%\LogViewPlus\LogViewPlus.exe.config.

Regardless of the security model used, once the initial connection has been made the fingerprint of the server will be permanently cached. On future connection attempts, the cached fingerprint will be compared against the fingerprint presented. If a discrepancy is found, the user will be presented with a dialog giving the option to accept or reject the key. If the key is rejected, then the connection will be terminated and no credentials will be sent.

Data Stores

1 Data Source List

Connect to a remote data store.

Name	Type	Connection String
DotNetEtw	Etw	{Name: "DotNetEtw", UserModeProviders: [".NET ...
localhost	WinEvent	Application, Hardware Events, Internet Explorer, Key ...
localhost	Udp	udp://localhost:28182/localhost_28182.log
localhost	Syslog	syslog://localhost:514/

2 Add 3 Clone 4 Edit 5 Delete

Beginning in LogViewPlus v2.4, we introduced support for data stores in LogViewPlus. A data store is simply any non-file based store which may contain log entries. Currently, we support the following data stores:

1. [Databases](#) - LogViewPlus currently supports SQL Server, Oracle, My SQL, PostgreSQL and SQLite databases. If you have an additional database you would like to connect to that has a .Net adapter, please [contact support](#) for assistance.
2. [Windows Events](#) - Read directly from a local or remote Windows Event Log.
3. [Syslog](#) - Reads events from a Syslog server.

4. [ETW Events](#) - Reads an [Event Tracing for Windows](#) stream.
5. [UDP](#) - Reads events from a UDP broadcast stream.

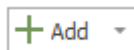
Do you have an idea for an additional data store? [Contact us](#) and let us know. We are always looking for ways to improve LogViewPlus.

1 Data Source List

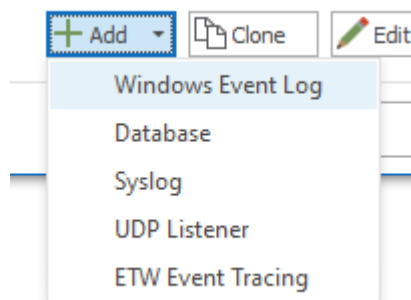
Name	Type
DotNetEtw	Etw
localhost	WinEvent
localhost	Udp
localhost	Syslog

The complete list of all known data sources. Double click on a data source to modify the connection details.

2 Add



The add data store command will display a drop down of all of the data source types currently supported by LogViewPlus.



Selecting one of these data sources will open the appropriate new configuration dialog.

3

Clone



When cloning a data source connection LogViewPlus will open the currently selected configuration. You can then make changes to the data source configuration and save your changes as a new connection.

4

Edit



The edit command opens the data source configuration form for the currently selected configuration. Saving changes will overwrite the current data source settings.

5

Delete



The delete command can be used to permanently remove the currently selected data source configuration.

Windows Events

The screenshot shows the 'Add Windows Event Connection' dialog box. It contains the following fields and controls:

- 1** Friendly Name: localhost
- 2** Friendly Category: (dropdown menu)
- 3** Server Name: localhost
- 4** Domain: (empty field)
- 5** User Name: (empty field)
- 6** Password: (empty field) with a 'Show' checkbox
- 7** Authentication Type: Negotiate (dropdown menu)
- 8** Log Sources: All Log Sources (dropdown menu)
- 9** Test, Save, Cancel buttons

Below the dialog box, there are two labels: 'Test' and 'Save / Cancel'.

LogViewPlus can use a Windows Event connection to read event log entries directly from a local or remote machine.

When reading log entries in Windows, there are two important security settings that you may need to modify.

1. To access the local Security log, LogViewPlus must be running under the administrator account.
2. If you are trying to connect to a remote server make sure that the "Remote Registry" service is running and that the user account you are trying to connect with is a member of the "Event Log Readers" group. The "Administrator" group is not needed and will not work.

1 Friendly Name

Friendly Name: localhost

A friendly name is simply a name for the server it is easy to remember. If a friendly name is not provided, the server name will be used instead.

The name provided will be used to build a URI which can be used to refer to this server. For this reason, the name provided must be URI friendly - it should not contain characters or symbols which are difficult to represent in URI form.

2

Category Name

Friendly Category:

A friendly category is a group name where this server can be categorized. This field is not required.

If you have previously configured a server with a friendly category, this category will be available as a drop-down option. Alternatively you can type a new name into the category text box.

3

Server Name

Server Name:

The name of the server which is hosting the Event Logs we want to access. This can either be a host name or an IP address.

4

Domain

Domain:

The domain of the specified user. This field will be used when connecting to remote machines.

5

User Credentials

User Name:

Password:

☐ Show

The user credentials which should be used when monitoring the event logs. This field can be left blank to use the current user credentials.

6

Authentication Type

Authentication Type:

The type of authentication to use. The values presented are adapted from the [SessionAuthentication](#) enumeration.

7 Log Sources

Log Sources:

The log sources you want to monitor.

The list of log sources is extracted dynamically from the target machine. If you do not see a log source you are interested in accessing, this may be due to permissioning issues. Make sure that LogViewPlus is running under the correct user account.

8 Test

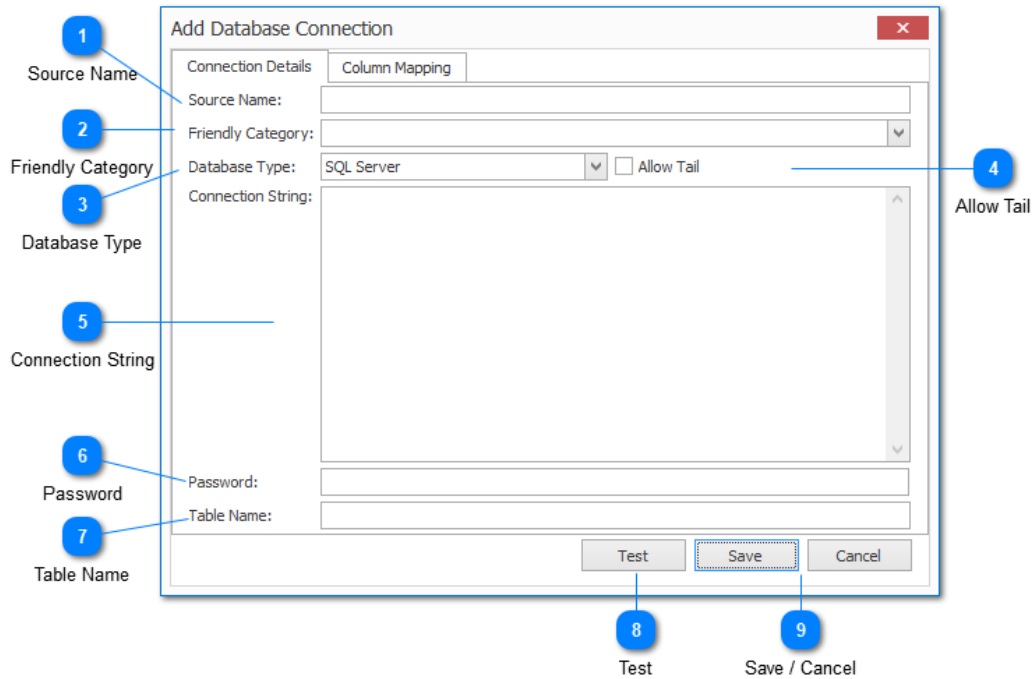
Validates and tests the provided configuration.

9 Save / Cancel

Once you have configured your server, you can use the save command to persist your changes. Once your changes have been saved the configured server will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

Databases



The database settings dialog helps you add or edit database connection details. LogViewPlus will use the details provided to read and display the log entries contains in the database.

To find out more about how database columns are mapped to LogViewPlus columns, please see [Column Mapping](#).

1 Source Name

Source Name:

This is a friendly name used to describe this database connection. If not provided, the table name will be used as a default.

2 Friendly Category

Friendly Category:

A friendly category is a group name where this database connection can be categorized. This field is not required.

If you have previously configured a server with a friendly category, this category will be available as a drop-down option. Alternatively you can type a new name into the category text box.

3

Database Type

Database Type:

The database connection type. LogViewPlus supports SQL Server, Oracle, MySQL, PostgreSQL and SQLite databases. If you have an additional database you would like to connect to that has a .Net adapter, please [contact support](#) for assistance.

If LogViewPlus cannot find the necessary libraries to load your selected database type, you may receive an error such as: "Unable to find provider to support: {providerName}."

In this scenario, you will need to add the appropriate assemblies to the %AppData%\LogViewPlus\DbProviders directory. Alternatively, the assemblies can also be placed in the %ProgramData%\LogViewPlus\DbProviders directory. Again, this step should only be necessary if the target libraries cannot be found in the GAC.

Depending on the database type selected, the following libraries may need to be placed into the plugin directory.

Database Type	Target Assembly
MySQL	MySQL.Data.dll
Oracle	Oracle.ManagedDataAccess.dll
SQLite	System.Data.SQLite.dll
PostgreSQL	Npgsql.dll

4

Allow Tail

☐ Allow Tail

By default, tail is disabled for database connections. Tailing a database table is supported, but we do not recommend it. Tailing a database table requires sorting the table by timestamp to find the latest records. This query will need to be executed approximately every second as LogViewPlus continuously checks for new updates. For large database tables, or where the table is not indexed by timestamp, this could lead to a significant performance impact on your database.

This performance impact would apply to the entire database which might impact unrelated tables and queries.

5

Connection String

Connection String:

The connection string contains the details needed to communicate with your database. Creating connection strings can be difficult if you are not familiar with the concept and, unfortunately, this is one area where our support will not be very helpful. If you are having trouble creating a valid connection string, we recommend contacting your database administrator. Alternatively, connectionstrings.com is a great resource.

If you need to encrypt aspects of your connection string, please see the Password field below.

6

Password

Password:

The password field can be used to encrypt any part of your connection string. To do this, you need to do two things:

1. Take the sensitive string out of your connection string and put it into the password field.
2. Where the string used to exist, add the template: `${Password}`

The text in the password field will be stored encrypted by LogViewPlus. When needed, it will be automatically decrypted and used to replace the `${Password}` template in the connection string.

7

Table Name

Table Name:

The name of the table that contains the log entries you want to load.

8

Test

The test command can be used to verify the currently configured database connection. To do this, a connection with the database will be established to access metadata regarding the target table.

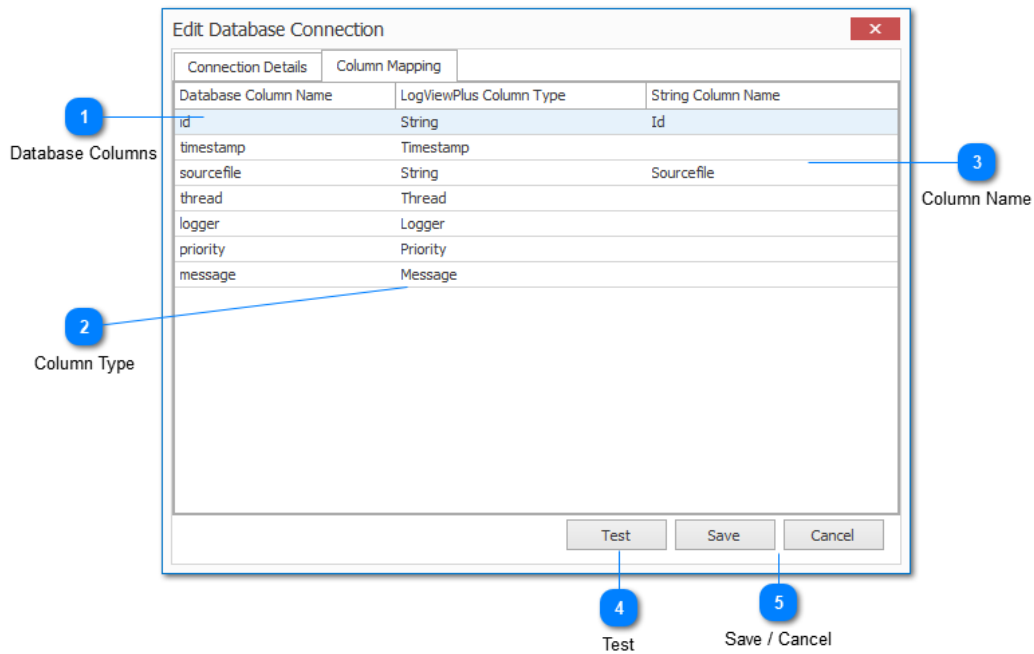
9

Save / Cancel

Once you have configured your database, you can use the save command to persist your changes. Once your changes have been saved the configured database will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

Column Mapping



The column mapping tab is used to convert database columns into LogViewPlus columns. Mapping from a database to LogViewPlus is controlled in three columns: Database Column Name, LogViewPlus Column Type, and String Column Name.

1 Database Columns

Database Column Name
id
timestamp
sourcefile
thread
logger
priority
message

This field is auto-populated by LogViewPlus based on meta-data read from the database.

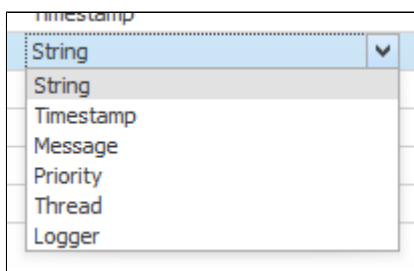
If the underlying data structure changes, you will need to reload the fields by testing your database connection. Depending on the extent of the changes to the data structure, you may need to recreate the database data source connection.

2

Column Type

LogViewPlus Column Type
String
Timestamp
String
Thread
Logger
Priority
Message

The column type field allows you to specify how the database column should be mapped in LogViewPlus. Six different LogViewPlus column types are supported as shown below:



These six column types can be divided into three categories:

1. Timestamp - This is the most column type as it defines when the log entry was created. Only one timestamp column can be set - regardless of the number of date time fields used by the database table. This field is required.

Several LogViewPlus scenarios will query or order by the Timestamp column. To improve query performance, we recommend adding an index to the Timestamp column.

2. String - This catch all property allows you to map any database field that can be represented as a string. If you specify a string column type, you should provide

a string column name. If a name is not provided, the field will not be visible in LogViewPlus.

3. Other - This category includes Thread, Logger, Priority and Message. These types map directly to the LogViewPlus types of the same name.

3 Column Name

String Column Name
Id
Sourcefile

If you specify that a column should be mapped to a LogViewPlus string, you will need to specify a column name for the field here. If a name is not provided, the field will not be visible in LogViewPlus.

4 Test

Test

The test command can be used to verify the currently configured database connection. To do this, a connection with the database will be established to access metadata regarding the target table.

5 Save / Cancel

Save	Cancel
------	--------

Once you have configured your database, you can use the save command to persist your changes. Once your changes have been saved the configured database will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

Syslog

The screenshot shows a dialog box titled "Add Syslog Connection" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- 1** Friendly Name: A text input field containing "localhost_514".
- 2** Category Name: A dropdown menu.
- 3** Server Name: A text input field containing "localhost".
- 4** Port: A numeric input field containing "514".
- 5** Protocol: A dropdown menu showing "UDP".
- 6** Test: A button.
- 7** Save / Cancel: A button.

Below the dialog, there is a link: [Find out more about configuring Syslog events](#).

LogViewPlus can use a read Syslog events from a local or remote machine.

Syslog uses a defined message format. It is not necessary to configure a parser when processing Syslog messages.

1 Friendly Name

Friendly Name: localhost_514

A friendly name is simply a name for the server it is easy to remember. If a friendly name is not provided, the server name and port number will be used instead.

The name provided will be used to build a URI which can be used to refer to this server. For this reason, the name provided must be URI friendly - it should not contain characters or symbols which are difficult to represent in URI form.

2 Category Name

Friendly Category:

A friendly category is a group name where this server can be categorized. This field is not required.

If you have previously configured a server with a friendly category, this category will be available as a drop-down option. Alternatively you can type a new name into the category text box.

3 Server Name

Server Name:

The server that is publishing the Syslog data.

4 Port

Port:

The port where the Syslog data is published.

5 Protocol

Protocol:

The protocol that should be used when listening for Syslog data. The protocol can be UDP, TCP or both.

6 Test

Validates and tests the provided configuration.

7 Save / Cancel

Once you have configured your listener, you can use the save command to persist your changes. Once your changes have been saved the configured listener will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

UDP

The screenshot shows a dialog box titled "Add UDP Listener" with a close button (X) in the top right corner. The dialog contains three input fields: "Friendly Name" with the value "localhost_1000", "Friendly Category" which is an empty dropdown menu, and "UDP Port" with the value "1000". Below these fields is a link that says "Find out more about configuring a UDP listener". At the bottom of the dialog are three buttons: "Test", "Save", and "Cancel". The "Save" button is highlighted with a dashed border. Numbered callouts (1-5) point to the following elements: 1. Friendly Name input field, 2. Friendly Category dropdown, 3. UDP Port input field, 4. Test button, and 5. Save / Cancel buttons.

LogViewPlus processes UDP logs by listening on a port and logging all data received to a local file. The local file can then be opened automatically and parsed as normal.

1 Friendly Name

Friendly Name: localhost_1000

A friendly name is simply a name for the server it is easy to remember. If a friendly name is not provided, the port number will be used instead.

The name provided will be used to build a URI which can be used to refer to this server. For this reason, the name provided must be URI friendly - it should not contain characters or symbols which are difficult to represent in URI form.

The friendly name given will be used as the file name where UDP data is logged. The name used here will therefore also be used to determine the appropriate parser configuration.

2 Category Name

Friendly Category:

A friendly category is a group name where this server can be categorized. This field is not required.

If you have previously configured a server with a friendly category, this category will be available as a drop-down option. Alternatively you can type a new name into the category text box.

3

UDP Port

UDP Port:

The UDP port where log data is published.

4

Test

Validates and tests the provided configuration.

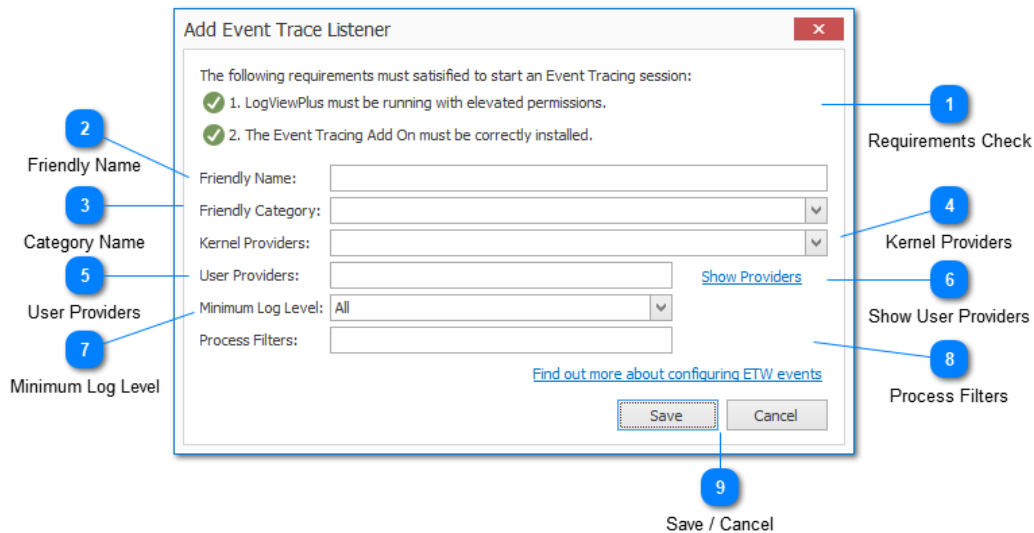
5

Save / Cancel

Once you have configured your listener, you can use the save command to persist your changes. Once your changes have been saved the configured listener will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

ETW Event Tracing



ETW event tracing is supported in LogViewPlus as an Add-On. The ETW Add-On was introduced in LogViewPlus v2.5 and is not supported in earlier versions.

To install the ETW Add-On:

1. [Download the ETW Add-On](#)
2. Extract the contents of the zip file to the %AppData%\LogViewPlus\Plugins\Etw directory. After the zip file is extracted, the new ETW directory should contain 24 items (two folders and 22 files).
3. [Enable plugins](#) and restart LogViewPlus.

LogViewPlus must be run as an administrator in order to access ETW event streams.

1 Requirements Check

The following requirements must be satisfied to start an Event Tracing session:

- ✓ 1. LogViewPlus must be running with elevated permissions.
- ✓ 2. The Event Tracing Add On must be correctly installed.

When adding a new ETW trace listener, you will be presented with a requirements check that will verify all conditions have been met for ETW trace listener execution.

Currently, there are two requirements to execute a trace listener:

1. LogViewPlus must be running with the elevated permissions. This implies either the user or the process is running as administrator.
2. The ETW Add-On should be correctly installed. Installations instructions for the ETW Add-On are discussed above.

The requirements checks are for listener execution only. The ETW listener can be configured even if no requirements have been met. The requirements check is displayed here for information purposes only.

2

Friendly Name

Friendly Name:

A friendly name is simply a name that will make this listener easier to reference.

A friendly name is required for ETW Listeners.

The name provided will be used to build a URI which can be used to refer to this listener. For this reason, the name provided must be URI friendly - it should not contain characters or symbols which are difficult to represent in URI form.

3

Category Name

Friendly Category:



A category name is a group name where this listener can be categorized. This field is not required.

If you have previously configured a server with a category, it will be available as a drop-down option. Alternatively you can type a new name into the category text box.

4

Kernel Providers

Kernel Providers:

If you are interested in monitoring kernel events, you can select them here. The list of available events is adapted from the [EVENT_TRACE_PROPERTIES](#) structure.

5

User Providers

User Providers:

The list of user mode providers to monitor. A list of user mode providers can be found by executing the 'logman' command. See the 'Show Providers' documentation below.

If you want to monitor multiple providers, the provider names should be comma separated.

6

Show User Providers

[Show Providers](#)

Executes the command "logman query providers". The [logman command](#) is used to display a list of user providers which are available on the current machine.

7

Minimum Log Level

Minimum Log Level:

The minimum log level you are interested in monitoring. If an event is detected which exceeds the minimum level, it will be discarded.

8

Process Filters

Process Filters:

A comma separated list of process names or task names. If process filters are provided and an event is received with a process or task name that is not in the list of known process filters, the event will be discarded.

This setting allows you to monitor events for a specific process only.

9

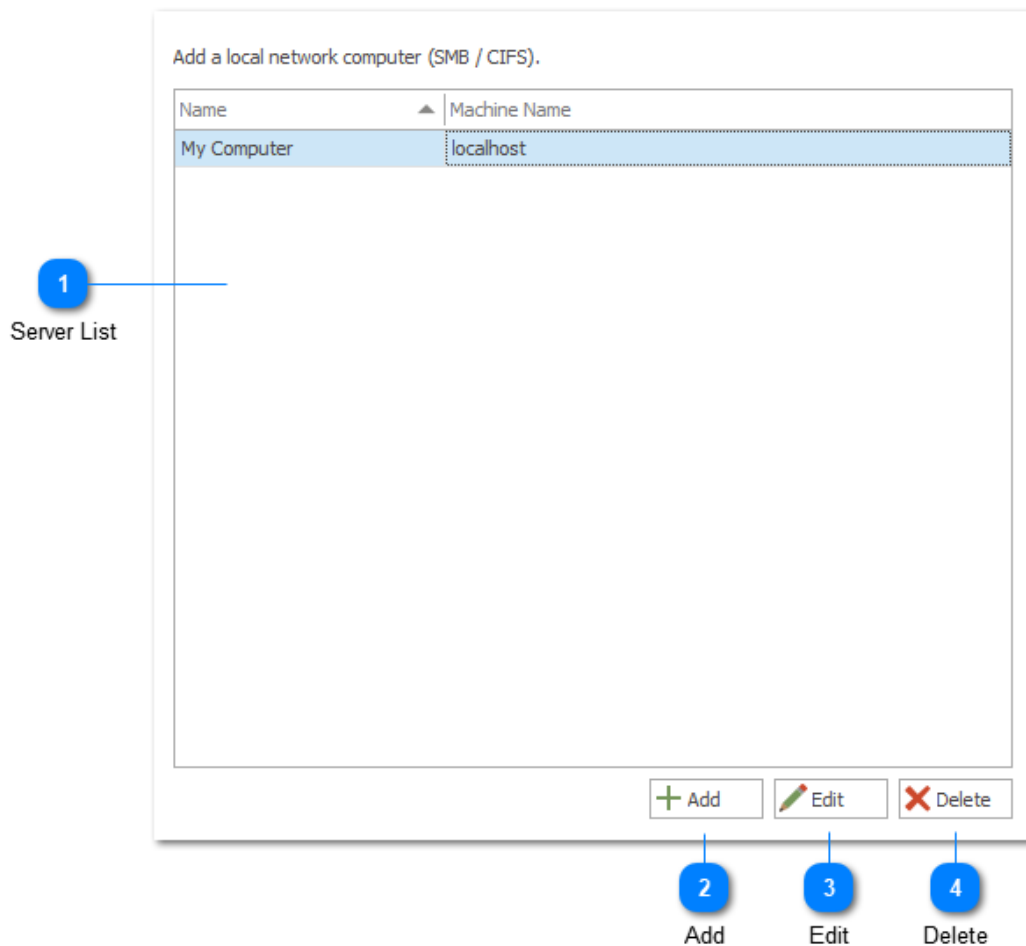
Save / Cancel



Once you have configured your listener, you can use the save command to persist your changes. Once your changes have been saved the configured listener will be immediately available in the [folder tree view](#).

Use the "Cancel" command to return to LogViewPlus without saving your changes.

Network Shares



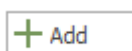
Network Shares will always be unpopulated when you first install LogViewPlus. This is because some networks may contain a large number of machines and quickly scanning the network to find the machine you want to access may not be possible. Instead, LogViewPlus uses an approach that requires you to know the name of the machine you want to connect to in advance. You can either explicitly add this machine name in the file system settings, or you can paste the full path to the file you want to open into the open file text box. Once a local network file has been opened by LogViewPlus, the network node will be available for future access.

1 Server List

Name	Machine Name
My Computer	localhost

The server list contains all of the network shares which have been previously configured.

2 Add



Creates a new network share and adds it to the configured server list.

3 Edit



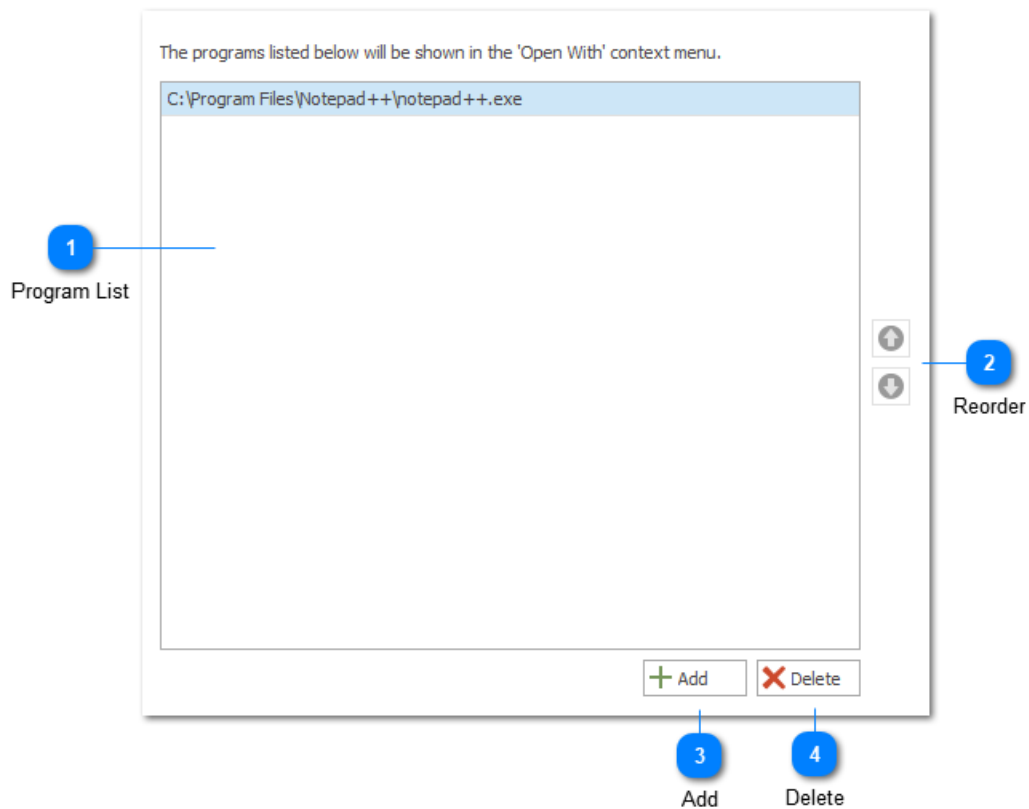
Allows you to edit the currently selected network share.

4 Delete



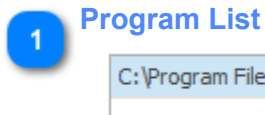
Permanently deletes the currently selected network share.

Open With Applications



The open applications settings allow you to add or remove the full path to applications which are installed on your local machine. Once an application has been added it will be visible in the "Open With" context menu. Please see the [file context menu](#) documentation for more information.

The Open With Applications configuration setting is only available from the File System Settings.



The program list contains all of the target open applications which have been previously configured.

2

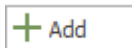
Reorder



The reorder commands on the right of the screen can be used to move the target open applications up or down in the configured server list. This controls the display order in the LogViewPlus file browser context menu.

3

Add



Allows you to browse to a new target open application and add it to the program list.

4

Delete



Permanently removes the selected target open application from the program list.

Static File System Settings

It is possible to configure LogViewPlus with server or data source configurations that cannot be modified by the end user. This option can be helpful when:

1. You need to keep access credentials protected from end users.
2. You have complex or shared environment settings and you would like to decrease the time needed for configuration.

Static file system settings cannot be exported or reset. You also will not be able to change the user credentials without replacing the static configuration file.

Having a static file system will not prevent the end user from creating or modifying new server configurations. From the end users perspective - LogViewPlus will work normally. Pre-configured servers will simply appear in the [LogViewPlus File Explorer](#) as though there were part of the standard file system.

Creating and implementing static file system settings is straightforward. All that we need to do is create a file systems settings file with our desired configuration and then copy that file to %ProgramData%\LogViewPlus\filesystem.default.dat.

You can create a static file system by following the instructions below:

1. Open LogViewPlus and [reset your application settings](#). You can back up your settings first if necessary.
2. Modify your configuration to add any needed [Remote Servers](#), [Data Stores](#) or [Network Shares](#).
3. Test your configuration. Once the settings have been made static, they may be difficult to change.
4. Move the file %AppData%\LogViewPlus\filesystem.dat to %ProgramData%\LogViewPlus\filesystem.default.dat. Notice that the file name is changed as part of this process.

If the configuration needs to be implemented for multiple users, simply copy file %ProgramData%\LogViewPlus\filesystem.default.dat to all of the target machines.

Custom Extensions

LogViewPlus currently supports several different kinds of custom plug-ins including: [filters](#), [parsers](#), [post-processors](#), [readers](#), [configuration](#), and [analyzers](#). We aim to make LogViewPlus as extensible as we can because we know that some of the best feature ideas are too niche to be widely applicable. Have an idea for an extension idea but not sure how to implement it? Please feel free to [contact us](#).

If you are new to LogViewPlus extensions, we recommend [getting started](#) with our [sample code](#).

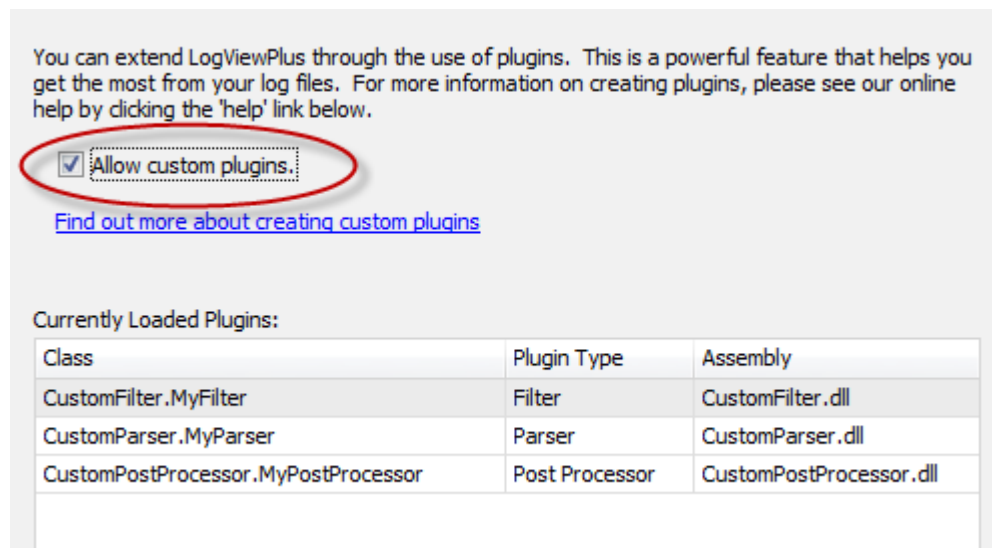
Running the Samples

When learning to create custom filters and parsers for LogViewPlus it is helpful to begin with the [sample code projects](#). There are four projects included in the sample code - a custom filter, custom parser, custom post processor and a custom reader. All of these implementations are kept intentionally simple. They serve an instructive rather than functional purpose.

The purpose of this tutorial is to get you up and running with the sample projects using Visual Studio.

To get started please [download the sample projects](#) and extract the zip file into a working directory. Open LogViewPlus_Samples.sln in Visual Studio. Once the solution is opened you should notice two projects: CustomFilter and CustomParser.

Before we build projects there's a few things we need to do. First open LogViewPlus and go to Settings -> Application -> Plugins and ensure that the "allow custom plugins" option is enabled.



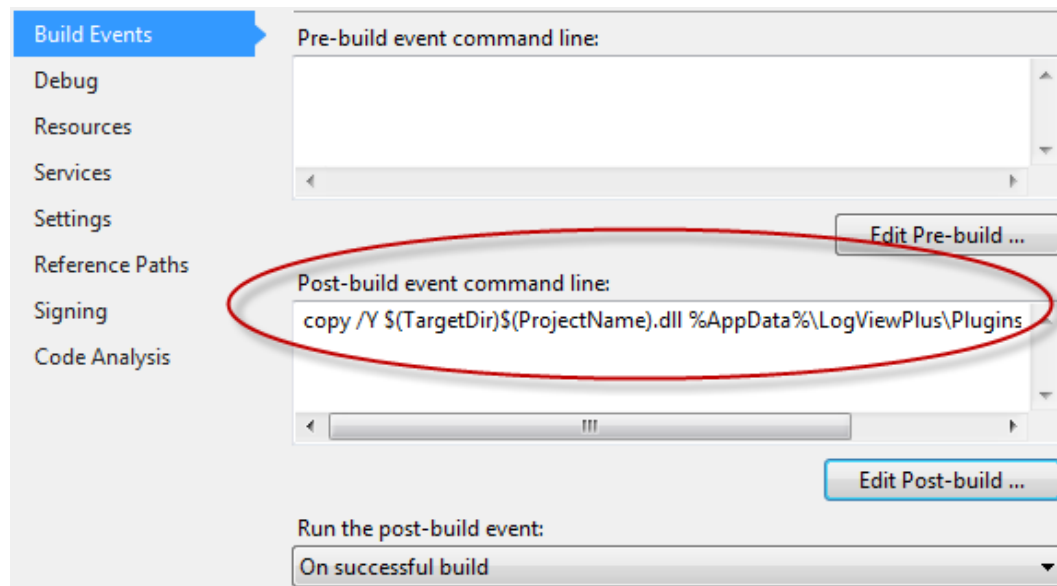
This is a security feature. LogViewPlus will only check for extensions if the above configuration is applied. Once enabled please close all running instances of LogViewPlus.

Returning to Visual Studio there are a few things we need to check before we build our project.

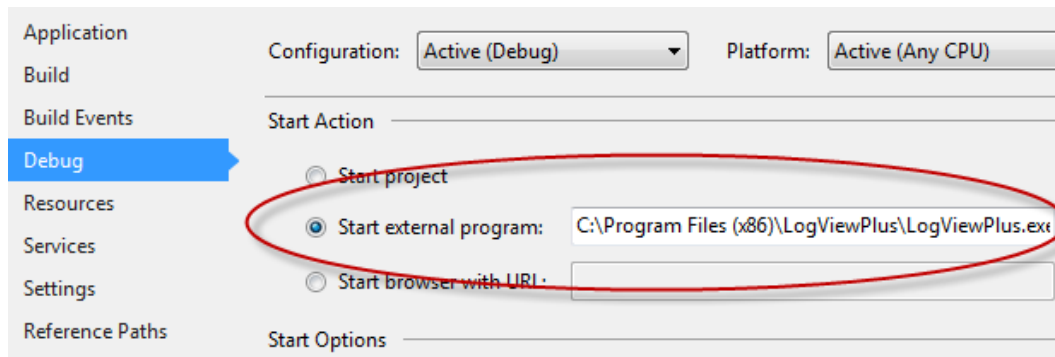
First, we recommend replacing Clearcove.LogViewer.Common.dll and Clearcove.LogViewer.Common.xml with the file included with your LogViewPlus installation.

A version of these files is included with the sample code to ensure the samples build correctly, but this version may be out of date. These files can be found in the installation directory %ProgramFiles%\LogViewPlus. They should replace the files in the "Libs" directory of the extracted sample code.

Next, for each project go to Properties -> Build Events. Both projects should have build events that copy the output of the build to the %AppData%\LogViewPlus\Plugins directory.



We also need to check that Properties -> Debug -> Start Action is set to run an external program. The plug-ins need to run inside of a LogViewPlus instance, so this should point to your LogViewPlus install directory.



We are now ready to build our solution. When you build the solution both of these projects will be built and the output would be saved to %AppData%\LogViewPlus\Plugins. This is the default directory where LogViewPlus expects plug-ins to be installed. When you run your project, Visual Studio will automatically start LogViewPlus in a debugger and you will be able to debug your plug-ins as expected.

Once LogViewPlus starts you will be able to confirm that your plug-ins are loaded successfully by going to Settings -> Application -> Plugins. Two plug-ins should be listed similar to the screenshot below.

You can extend LogViewPlus through the use of plugins. This is a powerful feature that helps you get the most from your log files. For more information on creating plugins, please see our online help by clicking the 'help' link below.

Currently Loaded Plugins:

Class	Plugin Type	Assembly
CustomFilter.MyFilter	Filter	CustomFilter.dll
CustomParser.MyParser	Parser	CustomParser.dll

[Find out more about creating custom plugins](#)

Please see the following sections for more information on creating [custom filters](#) and [custom parsers](#).

Microsoft .Net Versioning

Finally, note that LogViewPlus 2.5 and greater target .Net 4.7.2. Earlier versions target .Net 4.5.2.

The LogViewPlus Common library is required for all LogViewPlus plugins types. This library targets .Net 4.5.2. across all LogViewPlus versions to maintain backward compatibility. As discussed above, always recommend using the Clearcove.LogViewer.Common.dll library that installs with LogViewPlus when building plugins.

The sample code referenced here was updated with the 2.5 release to target .Net 4.7.2. If you are targeting an earlier version of LogViewPlus, you will need to change the target framework.

Custom Filters

When learning to create custom filters for LogViewPlus it is helpful to begin with the [sample code projects](#). This tutorial will assume that you have downloaded and run the sample projects successfully. Please see [running the samples](#) for more information.

Custom filters are useful when you need to filter your log file in a way that is not inherently supported by LogViewPlus.

Our custom filter implementation will be responsible for enhancing the LogViewPlus text search functionality. The existing LogViewPlus text search functionality works by matching the whole term. For example, if you search for "hello world" the search will not match "hello my world". Our custom filter implementation will work slightly differently. Any spaces in the search term will be interpreted as a separator for multiple search terms. Using this approach, "hello world" will match any string with the words "hello" and "world" anywhere in the string. Both search terms will be required.

If we look at the code included in the CustomFilter project and remove of the comments, we can see the custom filter implementation is very straightforward.

```
public class MyFilter : ILogFilter
{
    private string[] _searchTerms = null;
    public string Arguments { get; private set; }
    public string Name { get { return Arguments.Replace(" ", " & "); } }

    public void Initialize(string arguments)
    {
        Arguments = arguments;
        _searchTerms = arguments.Split(' ');
    }

    public bool Show(LogEntry logEntry)
    {
        foreach(var term in _searchTerms)
            if (logEntry.Message.IndexOf(term) < 0)
                return false;

        return true;
    }
}
```


All we need to do is create a class which extends the ILogFilter interface:

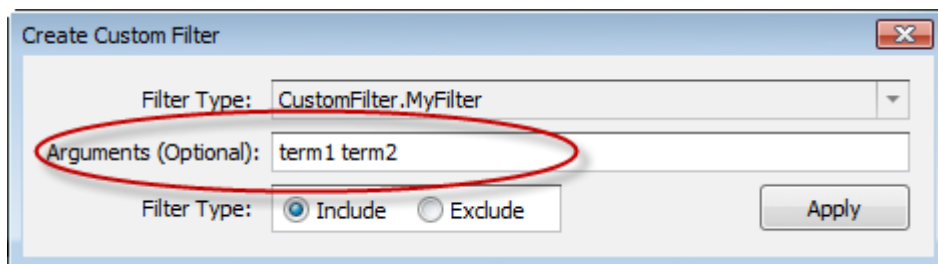
```
public interface ILogFilter
{
    string Arguments { get; }
    string Name { get; }
    void Initialize(string arguments);
    bool Show(LogEntry logEntry);
}
```

This interface has four parts:

Arguments - This getter provides access to the arguments used to initialize the filter. This is important when persisting the filter as the arguments will be needed later to reinitialize a new filter.

Name - This is the name of the filter. This name will be displayed to the user in the [file management](#) list tree. In our example filter we are replacing spaces with ampersands. This is an attempt to give the user a better understanding of how our filter works.

Initialize - A custom filter is initialized with a string that is provided by the user when creating the filter. Providing an initialization string to a custom filter is optional as some filters may not need initialization.



Show - The Show method is responsible for showing or hiding a given log entry. It is responsible for answering a simple question: should this log entry be displayed to the user?

Custom filters may be include or exclude type filters. This functionality is implemented outside of your custom filter. The show method will always assume that your filter is written as an include filter.

As shown above, creating a custom filter for LogViewPlus is relatively simple. If you have any questions or comments please feel free to [contact us](#).

Custom Parsers

When learning to create custom parsers for LogViewPlus it is helpful to begin with the [sample code projects](#). This tutorial will assume that you have downloaded and run the sample projects successfully. Please see [running the samples](#) for more information.

Custom parsers are helpful when your log files are stored as text data, but the text data cannot be parsed easily by one of the installed LogViewPlus parsers. For example, if your log files used a proprietary text-based data format.

Our custom parser implementation will be responsible for parsing a very simple log file which should be written in the format `%d{yyyy-MM-dd HH:mm:ss,fff} - %m%n`. Note that LogViewPlus is already capable of parsing log files in this format. Therefore our custom parser will provide no functional benefit. However, it would likely have a performance benefit - although this is not been tested.

If we look at the code included in the CustomParser project and remove of the comments, we can see the custom parser implementation is straightforward.


```

public class MyParser : ILogParser
{
    public string GetDescription()
    {
        return "My Custom Parser";
    }

    public void Initialize(string arguments)
    { // Not used.
    }

    public bool IsLogEntry(string logLine)
    {
        if (string.IsNullOrEmpty(logLine))
            return false;

        return logLine.IndexOf(" - ", 0) >= 0;
    }

    public ParseResult Parse(string logLine, LogEntry newEntry)
    {
        int pos = logLine.IndexOf(" - ");
        if(pos <= 0)
            return ParseResult.Fail;

        var date = logLine.Substring(0, pos);
        var msg = logLine.Substring(pos + 3);

        newEntry.Date = DateTime.ParseExact(date, "yyyy-MM-dd HH:mm:ss,fff",
            CultureInfo.CurrentCulture);
        newEntry.Message = msg;
        return ParseResult.Success;
    }
}

```

All we need to do is create a class which extends the ILogParser interface:

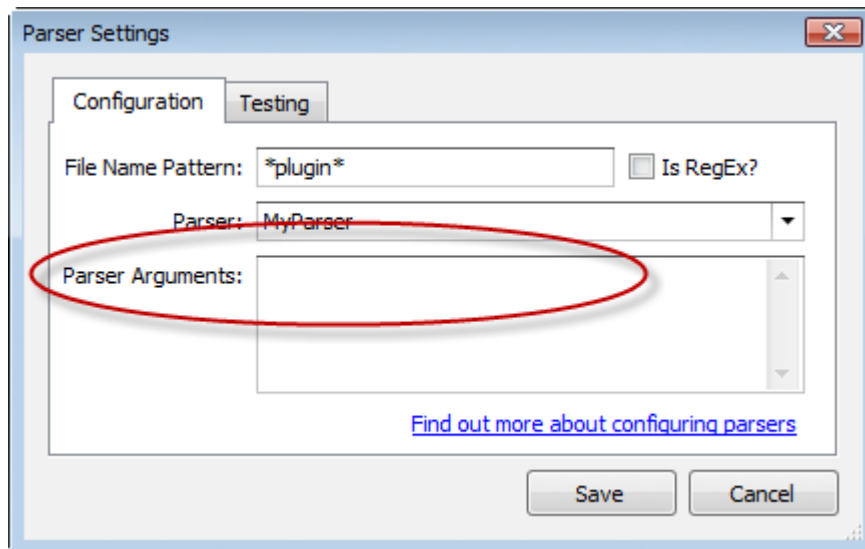

```
public interface ILogParser
{
    string GetDescription();
    void Initialize(string arguments);
    bool IsLogEntry(string logLine);
    ParseResult Parse(string logLine, LogEntry newEntry);
}

public enum ParseResult
{
    Success,
    Fail,
    ContinueRead,
    EndRead
}
```

This interface has four parts:

GetDescription - This method will return a description for our parser. This description will be displayed to the user in the tooltip that is displayed when the user hovers over a log file.

Initialize - A custom parser is initialized with a string that is provided by the user when creating the parser. Providing a initialization string to a custom filter is optional as some parsers may not need initialization.



IsLogEntry - Given a line of text, this method is responsible for determining whether or not the line is the beginning of a new log entry. Our (overly simplistic) example simply checks whether the line has the text " - " in it. While this is fine for our example it probably wouldn't work in a production system because a long line may have the text " - " unexpectedly as part of its log message. It would be better to check both the date and the static text.

Parse - This method is responsible for parsing the given log line and saving the result into the provided log entry. Note the log entry may span multiple lines. You can control the way the log entry is parsed by managing your parse result. A parse result could have one of four values:

Parse Result	Description
Success	<p>Success is a signal to the parser controller that you have been able to identify the start of a new log entry. The next few lines in the log file may be part of the same entry. In order to determine whether the next line in the log file is part of the same log entry, LogViewPlus will call the IsLogEntry method previously discussed. If the IsLogEntry method returns false, the line will be appended to the message field of this log entry. If IsLogEntry returns true, the parse method will be called again for the same log line.</p> <p>This is the basic processing loop for parsing a log file.</p>

Fail	Fail is a signal to the parser controller that the given log file line does not represent the beginning of a log entry. This may be the case, for example, if you are partially opening a log file and have started reading from the middle of the file. LogViewPlus may try to recover by ignoring the failure for a period of time and attempting to continue processing the file.
ContinueRead	ContinueRead is a signal to the parser controller that we do not yet have enough information to process this log entry and will need (at a minimum) the next log line. This field is used, for example, by the XML parser because XML log entries may span multiple lines before the entry can be parsed.
EndRead	EndRead is a signal to the parser controller to terminate processing of this log entry and start over with the next entry. It is helpful when reading from the middle of a log file when you know where your log entry begins and ends. This field is used, for example, by the Log4XmlParser to signal that the current entry should be discarded and we should start fresh on the next line.
An exception is thrown	<p>The result of throwing an exception from within the Parse method is dependent on what LogViewPlus is trying to do. If LogViewPlus is trying to do something complicated like a partial read - exceptions may be expected. In this case parsing may continue.</p> <p>However, generally speaking, the exception will be raised and the application will crash. You should therefore try to catch and handle exceptions internally.</p>

As you can see, creating a custom parser for LogViewPlus is relatively simple. If you have any questions or comments please feel free to [contact us](#).

Post Processors

When learning to create custom post processors for LogViewPlus it is helpful to begin with the [sample code projects](#). This tutorial will assume that you have downloaded and run the sample projects successfully. Please see [running the samples](#) for more information.

The existing parsers in LogViewPlus are very powerful, but in some scenarios, you may find you want to modify the output slightly. That's where custom post processors can help.

A custom post processor can be used to modify a parser's output before it is sent to LogViewPlus for display and analysis.

Let's say, for example, that we have a log file that uses custom priority levels. For example, consider the following log entries:

```
09:58:11,704 [75] DEBUG SyncEventService - Successfully created Event
09:58:11,704 [73] DEBUG SyncEventService - Successfully created Event
09:58:11,736 [75] Performance PerformanceWriter - Started Process entries
09:58:11,751 [81] Progress PerformanceWriter - Setting work completed
09:58:11,704 [73] DEBUG SyncEventService - Successfully created Event
```

Looking at these five log entries we can see three distinct log message priorities: debug, performance, and progress. The "performance" and "progress" priorities are non-standard and will not be understood by LogViewPlus. So, we are going to write custom post processor which transforms these invalid priorities into loggers which can be filtered by LogViewPlus.

Note that LogViewPlus now supports custom [Log Levels](#). Custom log levels would be a better way to implement this functionality, but this discussion is still valid as an example.

If we look at the code included in the CustomPostProcessor project and remove the comments, we can see the custom post processor implementation is straightforward.


```

public class MyPostProcessor : ILogPostProcessor
{
    public void Modify(LogEntry newEntry)
    {
        var level = newEntry.Priority;
        switch (level.ToLower())
        {
            case "progress":
            case "performance":
            {
                newEntry.Logger = level;
                newEntry.Priority = "INFO";
                break;
            }
        }
    }
}

```

Creating a post processor is as simple as implementing the ILogPostProcessor interface. This interface defines one method *Modify(LogEntry entry)*.

```

public interface ILogPostProcessor
{
    void Modify(LogEntry newEntry);
}

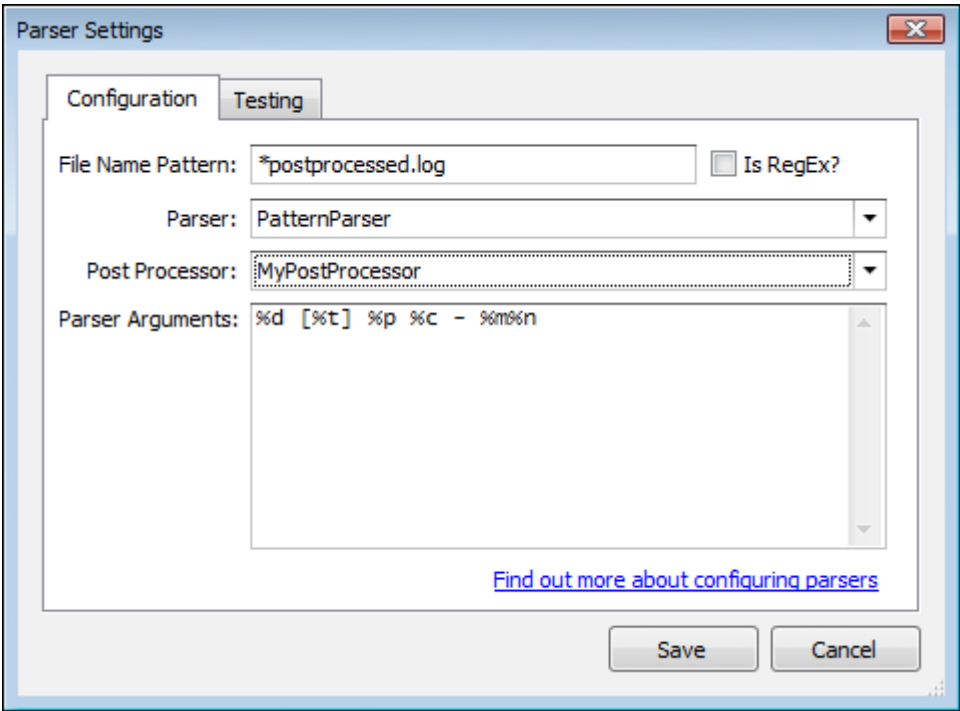
```

The Modify method can be used to modify the parsed log entry in any way you see fit. This method is called immediately after the entry is parsed, but before the log entry is given to LogViewPlus for display.

Post processors are reused across multiple new log entries. Therefore your custom post processor should not maintain any state.

Once we have compiled our custom post processor and added it into the LogViewPlus process (following the instructions outlined in [running the samples](#)), we still need to associate our post processor with a given log file. To do this, we can use the parser settings dialog. If LogViewPlus detects the possibility of a custom post processor, it will add a

new field into the parser settings dialog. The new "post processor" field will allow you to associate a post processor with a parser and log file as shown below.



Once we have associated the post processor with a log file parser all we need to do is refresh the log file to see that our custom post processor is performing as expected.

Time	Thread	Level	Logger	Message
09:58:11.736	73	INFO	Performance	Started Proc
09:58:11.751	81	INFO	Progress	Setting work

As you can see, creating a custom post processor for LogViewPlus is relatively simple. If you have any questions or comments please feel free to [contact us](#).

Custom Readers

When learning to create custom readers for LogViewPlus it is helpful to begin with the [sample code projects](#). This tutorial will assume that you have downloaded and run the sample projects successfully. Please see [running the samples](#) for more information.

Custom readers are the most advanced LogViewPlus extension type. Custom readers can be used when you need to import data into LogViewPlus from a data source that is not a text file. A custom reader might be used to access the database, read information off the network, or decode the data stored in a custom binary format (for example, a Protobuf log file).

Most custom readers will be environment specific. In an effort to simplify the deployment of our custom reader, we have decided to simply create a new log entry on a timer tick. This example is not very helpful in a real-world scenario but it does effectively show how to create a custom reader while simplifying the deployment and learning curve.

If we look at the code included in the CustomReader project and remove all of the comments, we can see the custom reader implementation is divided into three parts. First, there is the code which generates the log entries. This code is relatively straight forward.

```
private volatile int _entriesProcessed = 0;
private readonly Timer _timer = null;
private readonly List<LogEntry> _cache = new List<LogEntry>();

public MyReader()
{
    _timer = new Timer(OnTimerTick);
}

private void OnTimerTick(object state)
{
    lock (_cache)
    {
        var entry = new LogEntry();
        entry.Date = DateTime.Now;
        entry.Priority = "INFO";
        entry.Logger = "MyLogger";
        entry.Message = ++_entriesProcessed + ": " + (Arguments ?? "No arguments.");

        _cache.Add(entry);
    }
}
```

Our class has three private variables:

_entriesProcessed - This integer will keep track of the number of log entries which have been created. This number may be useful in certain debugging scenarios.

_timer - The timer is used to simulate new log entries coming into the system. Note that in the above example our timer has not yet been started.

_cache - Our cache is a list of log entries that have been created. We will need to manage our cache as part of our ILogReader implementation. Some aspects of our ILogReader implementation may be called on a separate thread. Therefore we need to control access to the cache with a lock.

The main method in our custom log reader implementation is the OnTimerTick event. This is the event that will generate our new log entries. We can see from the example above that our log entries should have the columns date, priority, logger, and message. Our custom log reader implementation is creating a simple log entry which echoes the reader configuration back to the user.

The next method in our custom log reader implementation is the GetSupportedTypes method.

```
public List<FieldColumnInfo> GetSupportedTypes()
{
    var retval = new List<FieldColumnInfo>();
    retval.Add(new FieldColumnInfo(ElementType.Date, true));
    retval.Add(new FieldColumnInfo(ElementType.Priority, true));
    retval.Add(new FieldColumnInfo(ElementType.Logger, true));
    retval.Add(new FieldColumnInfo(ElementType.Message, true));
    return retval;
}
```

The GetSupportedTypes method is required when we are implementing the IColumnManagement interface. This method is responsible for returning the column definitions for our data set. Implementation of this interface is not strictly required, however if we do not implement this interface we will not be able to see any columns for our parsed log entries. Therefore, most real-world examples will require an implementation.

The remaining methods in our CustomReader implement the ILogReader interface. It is important to note that the ILogReader interface is optimized for the scenario where we are reading log files in batches to allow for improved performance.


```

public bool AllowProgressTracking { get { return false; } }
public long ProgressPosition { get { return 0; } }
public int LineNumber { get { return _entriesProcessed; } }
public bool AllowTail { get { return true; } }
public string Arguments { get; private set; }

public bool AcceptsConfiguration()
{
    return true;
}

public void Initialize(string arguments)
{
    Arguments = arguments;
    return;
}

public List<string> GetWarnings()
{
    return null;
}

public bool AtEndOfFile(long fileSize)
{
    lock (_cache)
    {
        return _cache.Count == 0;
    }
}

public List<LogEntry> NextBatch()
{
    lock (_cache)
    {
        var list = _cache.ToList();
        _cache.Clear();
        return list;
    }
}

public bool HasNextBatch(long fileSize)
{
    return false;
}

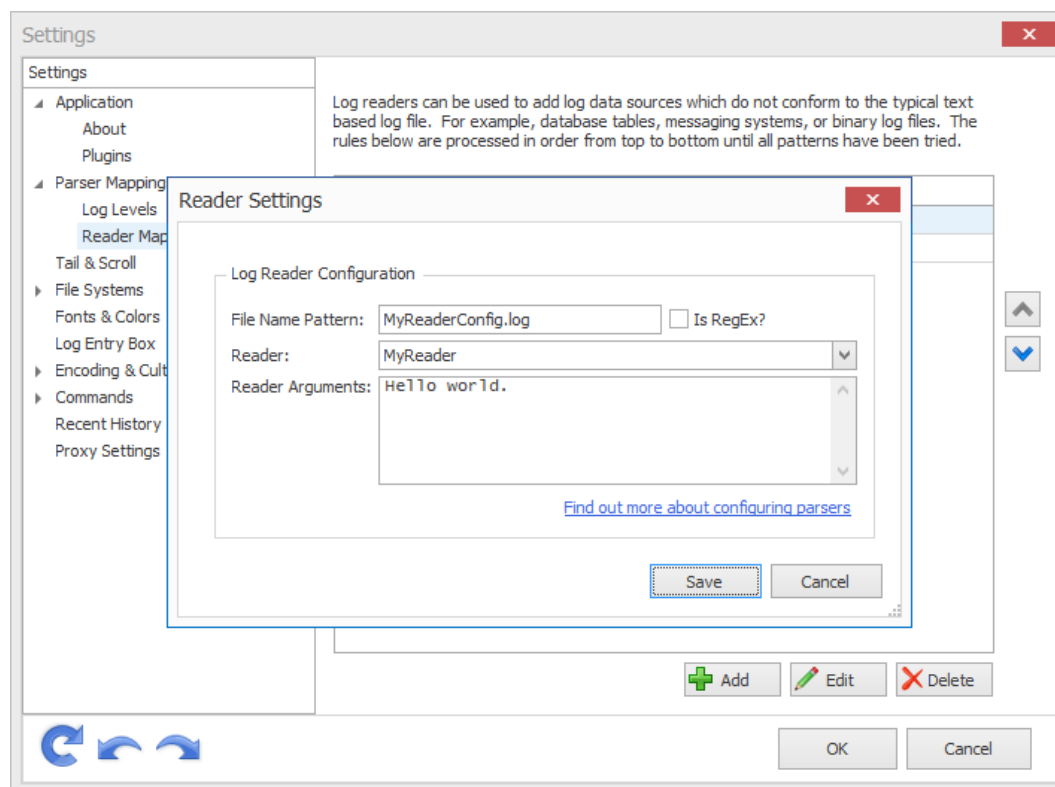
public List<LogEntry> InitializeRead(FileInfo file, long start, long stop)
{
    var txt = File.ReadAllText(file.FullName);
    int tickInterval = int.Parse(txt);
    _timer.Change(0, tickInterval);
    return null;
}

```

There are four features that the ILogReader interface will allow us to control.

Feature	Methods	Notes
Progress bars	AllowProgressTracking ProgressPosition LineNumber	<p>LogViewPlus can use either a normal progress bar or a marquee progress bar when loading a new log file. Use a marquee progress bar when the total number of log entries is unknown. To use a marquee progress bar set the AllowProgressTracking field to false.</p> <p>The LogViewPlus progress bar will only be shown when opening our log file.</p>
Argument initialization	AcceptsConfiguration Initialize GetWarnings Arguments	If our custom reader needs to be configured (AcceptsConfiguration), we can call Initialize with a list of arguments followed by GetWarnings to get a list of messages to be displayed to the user.
Batch processing	AtEndOfFile NextBatch HasNextBatch	When a tail file event occurs in LogViewPlus we will check if we are at the end of the log file. If not we will get the next batch of log entries and immediately check if further entries are pending with HasNextBatch. If HasNextBatch returns false we will wait for the next tail file event in LogViewPlus.
Reader Initialization	InitializeRead	The InitializeRead method is responsible for initializing our log reader based on the given file. In our case, the given file contains configuration that we need to execute our log reader. Note that this configuration file will be the file selected by the end user when trying to run our log reader. Using a configuration file as the target "log file" when opening a non-file based log reader is the recommended approach. Using this approach our log file access history will be managed automatically.

Once we have built our custom log reader and copied the resulting binaries into the LogViewPlus Plugins directory, we are ready to test. To do this start LogViewPlus, open Application Settings -> Reader Mappings and add the configuration shown.



After you have saved your settings, open the MyReaderConfig.log file which is included in the sample code distribution. Assuming tail is enabled you should see a new log entry appear approximately every second.

Log Files & Filters		Date	Time	Level	Logger	Message
MyReaderConfig.log		25 Jun 2017	10:10:52.112	INFO	MyLogger	1: Hello world.
		25 Jun 2017	10:10:53.119	INFO	MyLogger	2: Hello world.
		25 Jun 2017	10:10:54.120	INFO	MyLogger	3: Hello world.
		25 Jun 2017	10:10:55.120	INFO	MyLogger	4: Hello world.
		25 Jun 2017	10:10:56.122	INFO	MyLogger	5: Hello world.
		25 Jun 2017	10:10:57.134	INFO	MyLogger	6: Hello world.
		25 Jun 2017	10:10:58.134	INFO	MyLogger	7: Hello world.
		25 Jun 2017	10:10:59.135	INFO	MyLogger	8: Hello world.
		25 Jun 2017	10:11:00.150	INFO	MyLogger	9: Hello world.
		25 Jun 2017	10:11:01.149	INFO	MyLogger	10: Hello world.
		25 Jun 2017	10:11:02.152	INFO	MyLogger	11: Hello world.

If you were to temporarily disable tail and then re-enable it, all of the log entries missed during the time interval will be shown.

Custom Analyzers

When learning to create custom analyzers for LogViewPlus it is helpful to begin with the [sample code projects](#). This tutorial will assume that you have downloaded and run the sample projects successfully. Please see [running the samples](#) for more information.

Beginning in LogViewPlus 2.3.5, we introduced the ILogAnalyzer interface. This interface is both really simple and incredibly powerful. It takes an IWin32Window parent object as well as a list of all of the LogEntries found in the current view. This lets you execute a custom analysis on a set of log entries and potentially display your analysis in a custom reporting window.

```
public interface ILogAnalyzer
{
    void Analyze(object ownerWindow, IReadOnlyList<LogEntry> logEntries);
}
```

As discussed, the API has two parts:

OwnerWindow - An IWin32Window object which can be used to display a child form. The type is set as object for simplicity, but the object can be safely cast if needed.

IReadOnlyList<LogEntry> - The full list of all log entries shown in the current view. The data received by your implementation will be dependent on which view is used to execute the analyzer.

This interface will always be called from the UI thread. This allows you freely create UI components and control the user experience by blocking if necessary.

Our [sample projects](#) contain several examples of using the ICustomConfiguration interface.

For our discussion, we will focus on the implementation contained in MyParser.cs. This implementation simply calls into a new OpenFileDialog instance - this keeps the GUI aspects of this configuration simple.

MyParser implements the following ICustomConfiguration.Configure method:


```

public void Analyze(object ownerWindow, IReadOnlyList<LogEntry> logEntries)
{
    var owner = (IWin32Window)ownerWindow;
    if (logEntries == null || logEntries.Count == 0)
    {
        MessageBox.Show(owner, "No log entries found.", "No Entries",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        return;
    }

    var first = logEntries.First();
    var last = logEntries.Last();

    LogEntry largest;
    var a1 = CalculateAverageElapsed(logEntries, out largest);

    var a2 = CalculateAverageElapsed(first, first.FindNext(LookupSource.CurrentFilter, null));
    var a3 = CalculateAverageElapsed(last, last.FindPrevious(LookupSource.CurrentFilter, null));

    var result = MessageBox.Show(owner, $"Average time between log entries:" +
        "\r\n({a1})\r\n({a2})\r\n({a3})\r\n\r\n" +
        "Would you like to show the largest value?",
        "Average Elapsed", MessageBoxButtons.YesNo,
        MessageBoxIcon.Question, MessageBoxDefaultButton.Button1);

    if (result == DialogResult.Yes)
    {
        ((ILogViewer)owner).FindLogEntry(largest);
    }
}

```

Here we cast our ownerWindow to an IWin32Window and use this as the parent for a MessageBox which just displays some basic statistics.

One interesting thing to note about the code sample above is that it uses the FindNext / FindPrevious methods to navigate the current view. These methods could also be used to navigate the parent filter, or the source log file. As this is just an example, all three calculation methods produce the same result.

Beginning with LogViewPlus 2.4.30, you can also cast the IWin32Window owner object into an ILogViewer for additional functionality. For example, in the latest sample code, we use the ILogViewer interface to optionally find and select a given log entry. This functionality is currently only available in ILogAnalyzer implementations. Other custom extension types are not supported.

Implementing custom configuration in LogViewPlus is relatively simple. If you have any questions or comments please feel free to [contact us](#).

Custom Configuration

When learning to create custom configuration in LogViewPlus it is helpful to begin with the [sample code projects](#). This tutorial will assume that you have downloaded and run the sample projects successfully. Please see [running the samples](#) for more information.

Occasionally, when you are developing custom plugins for LogViewPlus you may find that the built in configuration options are insufficient. Beginning in LogViewPlus 2.3.8, we introduced the ICustomConfiguration interface. Implementing this interface on your custom [filter](#), [parser](#) or [reader](#) will allow you to execute your own configuration code on the LogViewPlus UI thread.

The ICustomConfiguration interface is very simple. It takes an IWin32Window parent object as well as a string representing the existing configuration. Using this, you can show a WinForms dialog to modify or create the desired configuration. Once the necessary changes have been made, the new configuration should be returned from the method as a string. LogViewPlus will then manage configuration persistence in exactly the same way as any other filter, parser or reader. Your custom object will receive the configuration string as a parameter to the Initialize method.

```
public interface ICustomConfiguration
{
    string Configure(object parentWindow, string configuration);
}
```

The API has three parts:

ParentWindow - An IWin32Window object which can be used to display a child form. The type is set as object for simplicity, but the object can be safely cast if needed.

Configuration - The string used to represent the target configuration or Null if a new configuration is requested.

Return String - The configuration string that should ultimately be passed to your target filter, parser or reader. We recommend a base64 string for advanced configuration.

This interface will always be called from the UI thread. This allows you freely create UI components and control the user experience by blocking if necessary.

Our [sample projects](#) contain several examples of using the ICustomConfiguration interface.

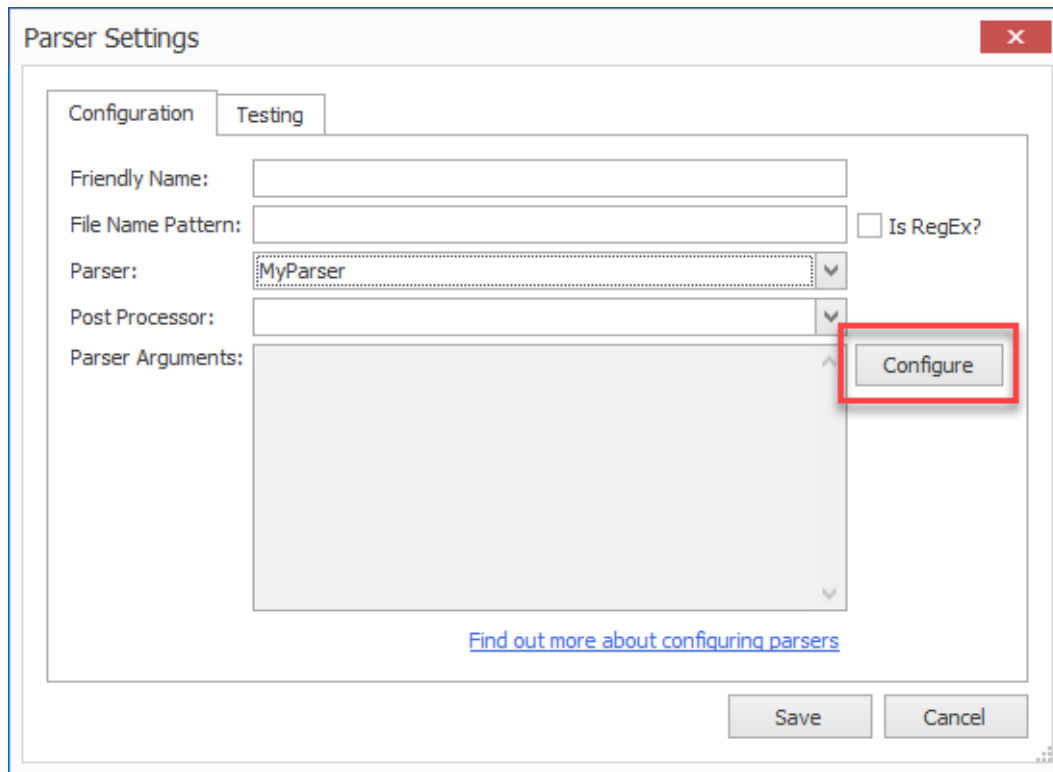
For our discussion, we will focus on the implementation contained in MyParser.cs. This implementation simply calls into a new OpenFileDialog instance - this keeps the GUI aspects of this configuration simple.

MyParser implements the following ICustomConfiguration.Configure method:

```
public string Configure(object parentWindow, string configuration)
{
    var parent = (IWin32Window) parentWindow;
    var fileBrowser = new OpenFileDialog();
    var rslt = fileBrowser.ShowDialog(parent);
    if (rslt != DialogResult.OK)
        return configuration ?? "Default Configuration";
    return fileBrowser.FileName;
}
```

Here, we cast our parentWindow to an IWin32Window and use this as the parent for a new OpenFileDialog. The file name selected by the dialog is passed back as a configuration result. Alternatively, if the configuration dialog is cancelled, we can return the received configuration or a default value.

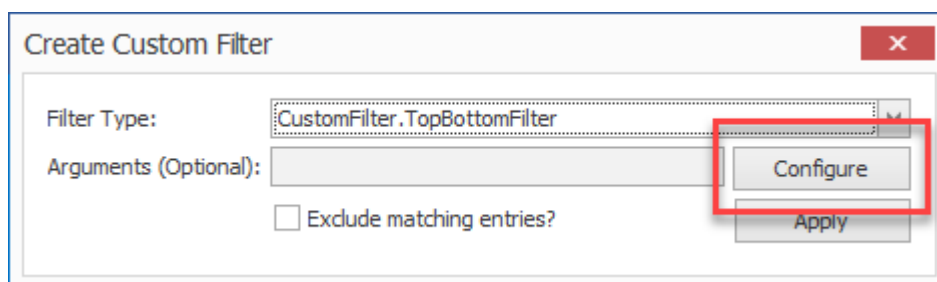
When our ICustomConfiguration interface is detected on a custom parser or reader you will find a new configuration command in the standard parser configuration dialog:



The 'Parser Settings' dialog box features a 'Configuration' tab and a 'Testing' tab. The 'Configuration' tab is active, showing fields for 'Friendly Name', 'File Name Pattern', 'Parser' (set to 'MyParser'), 'Post Processor', and 'Parser Arguments'. A 'Configure' button is highlighted with a red rectangle next to the 'Parser Arguments' field. A link 'Find out more about configuring parsers' is at the bottom. 'Save' and 'Cancel' buttons are at the bottom right.

Clicking on this command will execute the custom configuration. When custom configuration is set, the target must be configured using the configuration provided. The parser arguments box will remain disabled. This area will only be used for argument display.

If the `ICustomConfiguration` interface is detected on a custom filter then you will find a new configuration command in the custom filter configuration dialog:



The 'Create Custom Filter' dialog box shows 'Filter Type' set to 'CustomFilter.TopBottomFilter' and 'Arguments (Optional)' as a text field. A 'Configure' button is highlighted with a red rectangle next to the 'Arguments (Optional)' field. An 'Apply' button is below it. An 'Exclude matching entries?' checkbox is also present.

When custom configuration is set, the target must be configured using the configuration provided. The filter arguments box will remain disabled. This area will only be used for argument display.

Implementing custom configuration in LogViewPlus is relatively straight forward. If you have any questions or comments please feel free to [contact us](#).

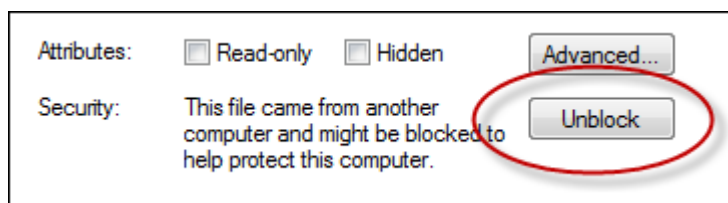
F.A.Q

+toc+

LogViewPlus is not installing correctly.

There are a few things you can try here:

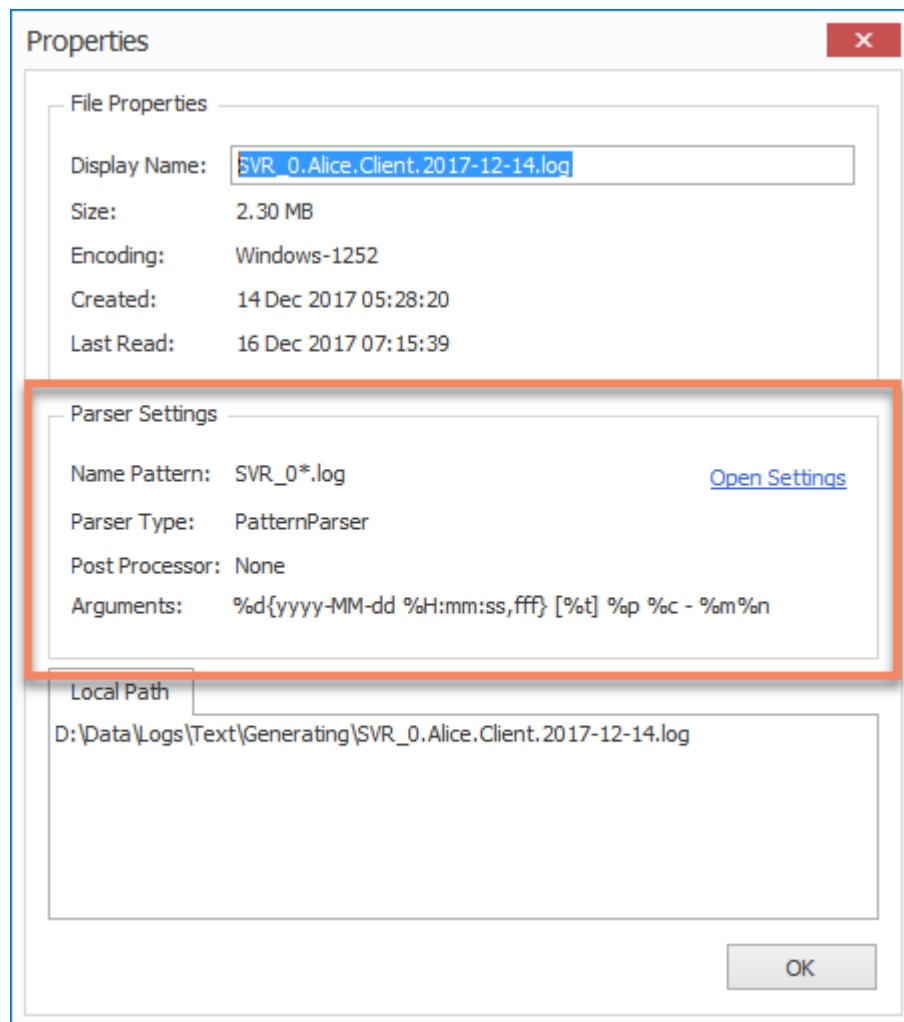
1. It is possible that the LogViewPlus installer is running into permission issues on your local machine. Right click on the installer and select "Run as Administrator".
2. Maybe your antivirus is interfering with the LogViewPlus installation? Consider disabling it before running the installer.
3. If you have recently uninstalled LogViewPlus and you are now trying to reinstall the program, it is possible a file is in use and this is preventing the installation. Restart Windows and try running the installer again.
4. If you are running on Vista, you may need to unblock the setup file for it to run correctly. Right click on the setup.exe file and go to Properties -> General. There may be a security section with an 'Unblock' option as shown in the screenshot below. The file must be unblocked to run correctly.



If you continue to have problems, please don't hesitate to [contact us](#).

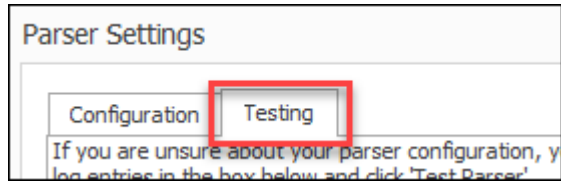
My log file is not being parsed correctly, what can I do?

The first thing to try is to right click on the file and go to "Log File Properties". This will open a dialog which tells you how the file was parsed.



Are you using the correct parser? Are the arguments correct? If the parser settings are not what you expected, it may be because there is a problem with the order of your parsers. See [parser mappings](#) for more information.

If you are using the parser you expected, the problem may be with the parser itself. Did you try testing the parser?

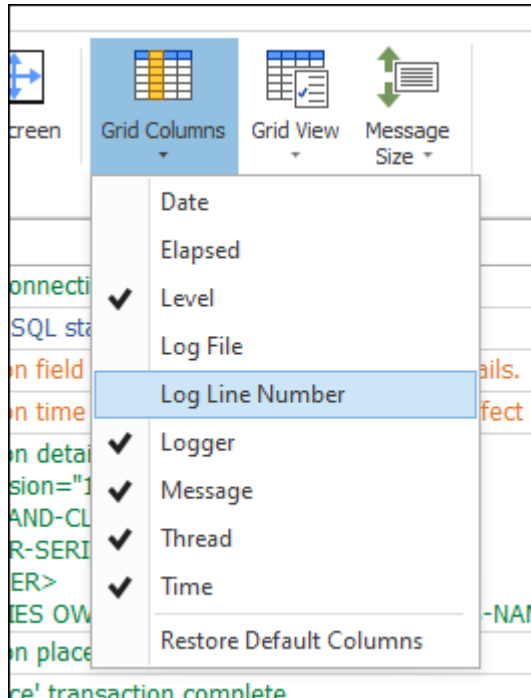


In the parser settings window, there is a tab you can use to enter sample entries. Paste the sample entries and use the 'Test' button provided. This allows you to quickly test your parser configuration settings. For more information, check out the documentation on [configuring parsers](#).

If you are still unable to configure your parser, please [contact us](#). It would be helpful if you could provide a few sample log entries as well as any details on how you were attempting to configure the parser.

How do I know if LogViewPlus is parsing all of the log entries in my log file?

If you are worried about this, there is an easy way to double check. Click on the 'Grid Columns' command and select the 'Log Line Numbers' command to add log file line numbers to your log file grid.



This will add a line number to each log entry. The line number represents the position where the log entry was found in the log file. You can now open the log file in a text editor which supports line numbers (like Notepad++) and verify the entry has been read correctly.

An error is occurring in LogViewPlus and I am unable to submit the details automatically.

The automatic error submission feature of LogViewPlus requires a working Internet connection. If LogViewPlus is unable to acquire an Internet connection, for example because of a proxy server requesting authentication, it will be unable to automatically submit the error details.

Instead, please [send us](#) the error.report file located in %AppData%\LogViewPlus.

I recently changed my settings and now LogViewPlus won't start.

You may have accidentally corrupted your settings. You can reset your settings by deleting the settings.dat file from %AppData%\LogViewPlus. Unfortunately, this will delete all of your settings. Any configuration you have saved such as parser mappings will need to be re-entered.

If you have not already reported this error to us, please [send us](#) the error.report file located in %AppData%\LogViewPlus so we can correct the underlying error. It would also be helpful if you could send us the corrupted settings.dat file so we can take a closer look.

I opened a log file successfully, but the log entries don't look right.

Which parser did you use when opening the file? You can find out by hovering over the log file.

If you used the [Pattern Parser](#), please double check your [conversion pattern](#). If you think your conversion pattern is correct, this is may be a bug. Please [send us](#) your log file along with the pattern you used to parse the file.

If you used the [Basic Parser](#), this is expected behavior. The Basic Parser is a best effort parser and may not be able to parse the file correctly. Consider using the Pattern Parser instead. Also, please [send us](#) your log file. The more test data we have the more we can tweak the Basic Parser to better suit your needs.

When I check for updates automatically, the new version is not detected.

There are two reasons why checking for updates with in LogViewPlus might fail to return the latest version. The first reason is that you may be behind a proxy server which is preventing LogViewPlus from accessing the Internet. If you believe this is the problem, you can configure your proxy server.

The other reason LogViewPlus may be failing to detect the latest version is because we have not yet made the latest version available for automatic download. We frequently release the latest version to new users only in order to improve quality and application stability before a release is pushed to everyone.

Authorized Resellers

Please note that ComponentSource is an authorized reseller of LogViewPlus. You can find more information on the [ComponentSource LogViewPlus](#) home page.