



Viz Trio User Guide

Version 4.5



Viz Trio



Copyright ©2025 Vizrt. All rights reserved.

No part of this software, documentation or publication may be reproduced, transcribed, stored in a retrieval system, translated into any language, computer language, or transmitted in any form or by any means, electronically, mechanically, magnetically, optically, chemically, photocopied, manually, or otherwise, without prior written permission from Vizrt. Vizrt specifically retains title to all Vizrt software. This software is supplied under a license agreement and may only be installed, used or copied in accordance to that agreement.

Disclaimer

Vizrt provides this publication “as is” without warranty of any kind, either expressed or implied. This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt’s policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time. Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

Antivirus Considerations

Vizrt advises customers to use an AV solution that allows for custom exclusions and granular performance tuning to prevent unnecessary interference with our products. If interference is encountered:

- **Real-Time Scanning:** Keep it enabled, but exclude any performance-sensitive operations involving Vizrt-specific folders, files, and processes. For example:
 - C:\Program Files\[Product Name]
 - C:\ProgramData\[Product Name]
 - Any custom directory where [Product Name] stores data, and any specific process related to [Product Name].
- **Risk Acknowledgment:** Excluding certain folders/processes may improve performance, but also create an attack vector.
- **Scan Scheduling:** Run full system scans during off-peak hours.
- **False Positives:** If behavior-based detection flags a false positive, mark that executable as a trusted application.

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Created on

2025/11/28

Contents

1	Introduction.....	9
1.1	What is Viz Trio?	9
1.2	System Overview - Basic Setup	9
1.3	System Overview - Advanced Setup	10
1.4	Related Documents	10
1.5	Feedback and Suggestions	10
1.6	Support.....	10
2	Installing a Viz Trio System	11
2.1	Requirements.....	12
2.1.1	Prerequisites.....	12
2.1.2	General Requirements	12
2.1.3	Hardware Requirements.....	12
2.1.4	Software Requirements	13
2.1.5	Media Sequencer.....	13
2.1.6	Viz Pilot	13
2.1.7	Viz Trio Keyboard	13
2.2	Installation and Upgrade.....	15
2.2.1	Installation.....	15
2.2.2	Upgrade	15
3	Configuration.....	21
3.1	Hardware and License	22
3.1.1	Hardware Configuration	22
3.1.2	Connecting to Media Sequencer	22
3.1.3	Connecting to Viz Engine	23
3.1.4	License	23
3.1.5	Ports.....	24
3.2	Viz Trio OneBox	25
3.2.1	Viz Trio Configuration.....	25
3.2.2	Viz Trio OneBox Configuration.....	25
3.3	Configuration Window	26
3.3.1	User Interface	27
3.3.2	Graphic Hub.....	37
3.3.3	Viz Pilot Database	39
3.3.4	Import and Export Settings.....	41

3.3.5	Output.....	42
3.3.6	Connectivity.....	54
3.4	Command Line Parameters	68
3.4.1	Adding a Command Line Parameter	68
3.4.2	Viz Engine Startup Command Line Parameters.....	70
4	Datacenter	72
4.1	Overview.....	72
4.2	Requirements.....	73
4.3	Getting Started.....	73
4.4	Configuring Viz Engine.....	73
4.5	Prepare Scenes for Data Input in Viz Artist.....	74
4.6	Configuring Datacenter	75
4.6.1	Connect Data to Fields in Scenes	76
5	Playout Interface	78
5.1	Interface Overview.....	78
5.2	Main Menu	79
5.2.1	File.....	79
5.2.2	Page	80
5.2.3	Playout.....	80
5.2.4	View.....	82
5.2.5	Tools.....	83
5.2.6	Help.....	83
5.3	Modes	84
5.3.1	Viz Artist Mode	84
5.3.2	On Air Mode	84
5.3.3	Slave Mode.....	85
5.3.4	Show Modes.....	85
5.4	Undo and Redo	88
5.5	Tab Fields Window.....	89
5.5.1	Adding and Editing Custom Values	89
5.6	Status Bar	91
5.6.1	Active Tasks	91
5.6.2	Configuring a Profile	92
5.6.3	Changing a Profile	93
5.6.4	Changing Channel Assignment.....	93
5.6.5	Checking Memory Usage.....	93

5.7	Edit Show Script.....	94
5.8	Snapshot	95
5.9	Render Videoclip	96
5.10	Import Scenes	97
5.10.1	Import Recursively	98
5.10.2	Import Scenes with Toggle or Scroller Plugin.....	98
5.11	Search Media	99
5.11.1	Media Context Menu.....	100
5.11.2	Search and Filter Options	100
5.11.3	Order Metadata Fields for Video and Images.....	101
5.12	Import Viz Engine Archive.....	102
5.13	Graphics Preview	103
5.13.1	Real-time Rendering of Graphics and Video	103
5.13.2	Adaptive Storytelling.....	103
5.13.3	Floating or Moving the Preview Window.....	107
5.13.4	Connection Status.....	108
5.13.5	OnPreview Script for a Graphical Element.....	109
5.14	Video Preview.....	110
5.14.1	Video Control Buttons and Timeline Display	110
5.15	TimeCode Monitor	111
5.15.1	TC Monitor	111
5.15.2	TimeCode Monitor Options.....	111
5.15.3	Enable the TimeCode Monitor	112
5.15.4	Disable the TimeCode Monitor	112
5.15.5	Monitor a Video Time Code.....	112
5.16	Page Editor.....	113
5.16.1	Edit a Page	113
5.16.2	Text	118
5.16.3	Database Linking.....	122
5.16.4	Image Property Editor.....	125
5.16.5	Transformation Properties	126
5.16.6	Tables.....	127
5.16.7	Clock	129
5.16.8	Maps	130
5.17	Create New Scroll.....	134
5.17.1	Create New Scroll Editor	134
5.17.2	Scroll Elements Editor.....	135

5.17.3	Scroll Configuration	136
5.17.4	Scroll Live Controls	136
5.17.5	Scroll Control.....	137
5.17.6	Element Spacing.....	137
5.17.7	Easepoint Editor	138
5.17.8	Working with Scrolls.....	139
5.18	Template List	140
5.18.1	Searching for Templates.....	140
5.18.2	Template Context Menu.....	141
5.18.3	Columns.....	143
5.18.4	Combination Templates	143
5.18.5	Creating a Combination Template	143
5.19	Page List	144
5.19.1	Page Content Filling.....	144
5.19.2	Page List Context Menu.....	145
5.19.3	Page List Columns	146
5.19.4	Concepts	148
5.19.5	Templates and Groups.....	149
5.19.6	Arm and Fire	151
5.20	Toolbar	154
5.20.1	Show Directories	154
5.20.2	Add Page List View	161
5.20.3	Creating Playlists.....	162
5.20.4	Show Properties	162
5.20.5	Cleanup Channels	167
5.20.6	Initializing Channels.....	167
5.20.7	Show Concept.....	168
5.20.8	Callup Page.....	168
5.21	Playlist Modes	169
5.21.1	Activate or Deactivate a Playlist	169
5.21.2	Take Pages On-Air in a Carousel from a Show Playlist	169
5.21.3	Show or Hide Columns in the Playlist	170
5.22	Field Linking and Feed Browsing in Viz Trio	178
5.22.1	Workflow.....	178
5.22.2	Overview	178
5.22.3	Technical.....	179
5.22.4	Field Linking	180

5.22.5	Feed Browsing	184
5.23	Timeline Editor.....	187
5.23.1	Timeline Editor Functions.....	188
5.23.2	Working with the Timeline Editor.....	191
5.23.3	Troubleshooting and Known Limitations	192
6	Macro Commands and Events	193
6.1	Events	194
6.1.1	Template Event Callbacks	194
6.1.2	Show Event Callbacks	195
6.1.3	Using the OnDelete Script Event	196
6.2	Macro Commands.....	197
6.2.1	Viz Trio Macro Language	197
6.2.2	Working with Macro Commands	198
7	Scripting.....	254
7.1	Viz Trio Scripting.....	255
7.1.1	Script Directory.....	255
7.1.2	Script Editor.....	255
7.1.3	Script Backup	259
7.1.4	Script Errors.....	259
7.2	Viz Template Wizard Scripting	260
7.2.1	Dynamically Adding Components	260
7.2.2	Set and Get Component Values.....	260
7.2.3	Set and Get Show Values	260
8	Appendix	262
8.1	Enabling Windows Crash Dumps	263
8.2	Logging.....	264
8.2.1	Viz Trio Log Files	265
8.2.2	Viz Trio Error Messages	266
8.2.3	Viz Engine and Media Sequencer Log Files	267
8.3	Rapid Import Requirements.....	269
8.3.1	General.....	269
8.3.2	Logging	269
8.4	Preview Server	270
8.5	Cherry Keyboard	271
8.5.1	Editing Keys (green)	271
8.5.2	Navigation Keys (white)	272
8.5.3	Program Channel Keys (red)	273

8.5.4	Preview Channel Keys (blue)	273
8.5.5	Program and Preview Channel Keys (red and blue)	274
8.6	Glossary	275
8.7	Viz Vectar Integration	276
8.7.1	Macro Calls to Integrated Systems	276
8.8	Troubleshooting Excel Database Connections	278

1 Introduction

Welcome to the Viz Trio User Guide.

1.1 What is Viz Trio?

Viz Trio contains all the features of a typical CG (Character Generator) system and more. Viz Engine is the graphics rendering output system and Media Sequencer controls the playout of media elements.

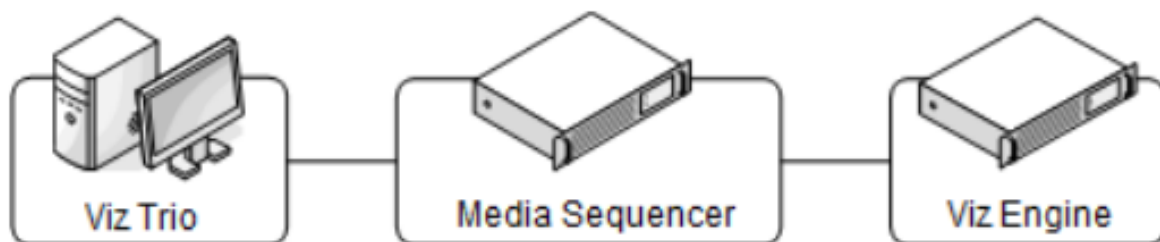
With Viz Trio you can:

- Trigger graphical elements stored as pages in a show structure, with each page utilizing a unique call-up code.
- Fill in data fields for graphics.
- Quickly import or reimport new graphics into a show, no need for a template redesign, everything happens automatically.
- Use Viz Trio as a clip player with a timeline editor.
- Customize the Viz Trio interface through scripting and macro commands.
- Use Viz Trio as a graphics playout server, integrating from a third party system through macro commands over TCP/IP.
- Let data update live through Datacenter without any scripting or advanced design.

Advanced features let you connect to multiple newsroom systems, do seamless context switches on graphics (Transition Logic), use specialized editors that can change almost any property of a graphic, produce on-the-fly graphics with a built-in design tool, and more.

Note: The Viz Trio client can run on any computer that has a network connection to Media Sequencer. For the local preview to function properly, the computer must have a good Nvidia GPU.

1.2 System Overview - Basic Setup



Info: The system overview above shows a basic setup. Viz Engine is typically used for local preview.

- The Viz Trio operator creates pages, most often from pre-made templates. A page is an instance of a template that can be customized with data values, like sports results or election data.
- When the page is ready, the Viz Trio operator can easily send pages with the necessary information to air.
- Viz Trio automatically uses Media Sequencer to trigger final rendering of the output by the Viz Engine.

 **Note:** Although Viz Trio, Media Sequencer and Viz Engine can be installed and operated on a single machine, they are mostly installed on separate servers for performance and security reasons.

- Viz Trio can also interface with other systems for news feed and media resources, for instance MAM systems such as Viz One for video and media resources and Viz Pilot.
- Social Media feeds and news can be interfaced and ingested in the workflow with the Viz Social solution.

1.3 System Overview - Advanced Setup

- An advanced setup with newsroom and video integration involves several other Vizrt products such as a Vizrt MAM system, Viz Pilot, Viz Gateway and third-party systems such as a newsroom and database system.
- Viz Gateway can be used to connect Viz Trio to most Newsroom systems using the MOS protocol.

 **Note:** Director from Viz Pilot can also be used as the control application.

1.4 Related Documents

- [Viz Artist User Guide](#): How to create standalone and transition logic scenes.
- [Viz Pilot User Guide](#): How to create a playlist in Viz Pilot that can be monitored in Viz Trio, and use the Newsroom Component to create newsroom data elements for a newsroom system playlist.
- [Viz Engine Administrator Guide](#): How to configure your Viz Engines.
- [Viz One Administrator Guide](#): How to configure your Viz One system.
- [Viz One - Studio User Guide](#): How to work with your Viz One system.

1.5 Feedback and Suggestions

We encourage suggestions and feedback about our products and documentation. To give feedback and/or suggestions, please contact your local Vizrt customer support team at www.vizrt.com.

1.6 Support

Support is available at the [Vizrt Support Portal](#).

For more information about all Vizrt products, visit:

- www.vizrt.com
- [Vizrt Documentation Center](#)
- [Vizrt Training Center](#)
- [Vizrt Forum](#)

See Also

- [Glossary](#)

2 Installing a Viz Trio System

The following sections describe the steps required to have a multi-client Viz Trio setup with a remote Media Sequencer connected to one or several Viz Engine output renderers.

 **Info:** Always [check online](#) for the latest version of the release notes regarding supported versions, hardware requirements and last-minute updates.

This section covers the following topics:

- [Requirements](#)
- [Installation and Upgrade](#)

2.1 Requirements

This section covers the following topics:

- [Prerequisites](#)
- [General Requirements](#)
- [Hardware Requirements](#)
- [Software Requirements](#)
- [Media Sequencer](#)
- [Viz Pilot](#)
- [Viz Trio Keyboard](#)

2.1.1 Prerequisites

Before installing a Viz Trio system, make sure that the correct hardware and latest software is available. All Vizrt software required is accessible from Vizrt's FTP server at download.vizrt.com. Contact your local Vizrt representative for credentials.

2.1.2 General Requirements

There are some general requirements for any Vizrt system to run, which apply when setting up a complete system with integration to other Vizrt and third party software:

- All machines should be part of the same domain.
- Users of the Vizrt machines should ideally be separated into at least two groups - administrators and designers/operators.
- Most machines running desktop applications must be logged in with sufficient privileges to run Vizrt programs, while services by default do not require users to be logged in.
- Vizrt servers must have static IP addresses.
- Vizrt recommends that customers who use remote shares for storing data use UNC (Universal Naming Convention) paths directly in the configuration, rather than mapped drives. For example: `\\vosstore\images`.
- Vizrt has license restrictions on all Viz Engine and Viz Artist instances. To have an output of Vizrt generated graphics (preview and program channels), a WIBU license is required.



Caution: Third party systems that provide Vizrt systems with files must only use Microsoft Windows operating system compatible characters in file names.

2.1.3 Hardware Requirements

In general, Viz Trio hardware requirement should match the hardware requirement for a preview version of the Viz Engine. See [Viz Engine Release Notes](#) for supported hardware.



Note: For more information on hardware specifications, please contact your local Vizrt customer support team.

2.1.4 Software Requirements

Recommended Versions

Please see the Release Notes.

Viz Engine

Viz Engine is the output service (renderer) for Viz Trio, and is a separate installer. A Viz Engine is required in order for Viz Trio to preview graphics. Check that Viz Engine is installed, configured and working before installing and using Viz Trio.

After installing Viz Engine, make sure to also install any additional plug-ins, such as Viz Datapool. The plug-ins required vary depending on the scene design and integrations.

To install Viz Engine, refer to the instructions in the [Viz Engine Administrator Guide](#).

2.1.5 Media Sequencer

Media Sequencer is in most cases installed on a server acting as a middle tier between Viz Trio and Viz Engine.

Installing Media Sequencer

1. Run the installer and follow the directions given by the installation wizard.
2. *Optional:* Install the Oracle 10g Runtime Client.
 - Restart Media Sequencer after installing the database client.
 - A database client is needed when connected to Viz Gateway for accessing the Viz Pilot database.

2.1.6 Viz Pilot

In order to edit Viz Pilot Edge elements in Viz Trio, Media Sequencer must be configured to connect to a Pilot Data Server. This server hosts Viz Pilot Edge, which is opened by Viz Trio in a web browser window.

To configure Media Sequencer to use a Pilot Data Server, go to **File > Configuration > Pilot Database** in Viz Trio and open the configuration page.

2.1.7 Viz Trio Keyboard

To install the Viz Trio keyboard, the keyboard's USB connector must be connected to the computer's USB port. The computer should be able to detect and automatically install required keyboard drivers.

 **Note:** To learn more about the keyboard's technical specifications, visit <http://www.devlin.co.uk/keyboards/semicust.html> DevlinGroup - Devlin Electronics Limited.

Importing the Keyboard Mapping File

1. Open Viz Trio and click the **Show properties** button.
2. In the **Show properties** window that opens, click the **Keyboard** button.
3. In the **Macros for Current Show** window that opens, click **Import**.

4. In the **Import keyboard shortcuts** dialog that opens, click the folder button to browse for and import the *keyboard* file (extension *.kbd).
5. *Optional:* Select to merge the existing keyboard shortcuts with the new ones.
6. Click **Import**.

See Also

- [Viz Pilot Database](#)

2.2 Installation and Upgrade

- [Installation](#)
- [Upgrade](#)
 - [Updating Viz Trio 4.x](#)
 - [Upgrading to Viz Trio 4.x from 3.x](#)
 - [Modifying VTW Templates](#)
 - [VTW template verification on Trio 3.x](#)

2.2.1 Installation

1. Make sure you have the correct hardware and software platform.
2. Ensure that Viz Engine is installed and that you have a working local Graphic Hub and Media Sequencer or have network credentials to the Graphic Hub and Media Sequencer.
3. Download the latest Viz Trio installer and Release Notes with FTP from download.vizrt.com/products/VizTrio/LatestVersion/.
4. Read the Release Notes carefully. They may contain important last minute information.
5. Double-click the downloaded installer and follow the installation instructions.
6. Visit docs.vizrt.com to learn more about the software or press F1 / Help > View Help to browse the local help.

**Note:**

- You can only have one Viz Trio installed on a computer.
- Any installed previous version of Viz Trio installed is overwritten by the new Viz Trio installer.
- The Installer attempts to preserve previous data/configuration settings.

2.2.2 Upgrade

Follow the upgrade path instructions below to ensure that Viz Trio client, settings, profiles and shows are retained [at](#) upgrade.

Updating Viz Trio 4.x

- Always back up the Media Sequencer file *default.xml* file before upgrading.

Upgrading to Viz Trio 4.x from 3.x

- Always back up the Media Sequencer file *default.xml* file before upgrading.

Major system architectural improvements were introduced with Viz Trio 4.x.

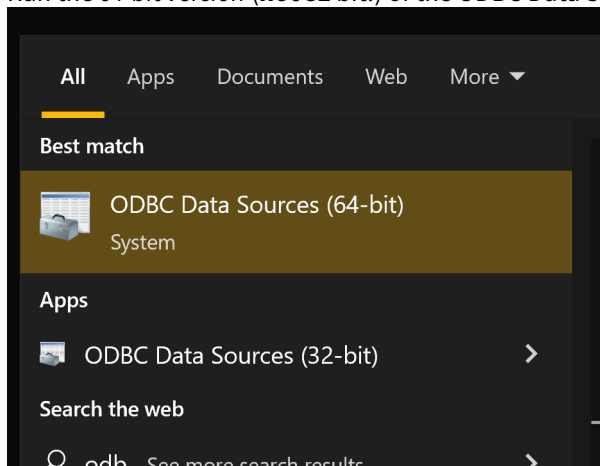
- All versions of Viz Trio that were released prior to version 4.0 are built with 32 bits. From 4.0 the product is built only for 64-bit Windows and is a 64bit application.
- Viz Trio 4.0+ can only operate with 64-bit software, incompatible 32-bit software can no longer function with Viz Trio.

Borland Database Engine Limitations

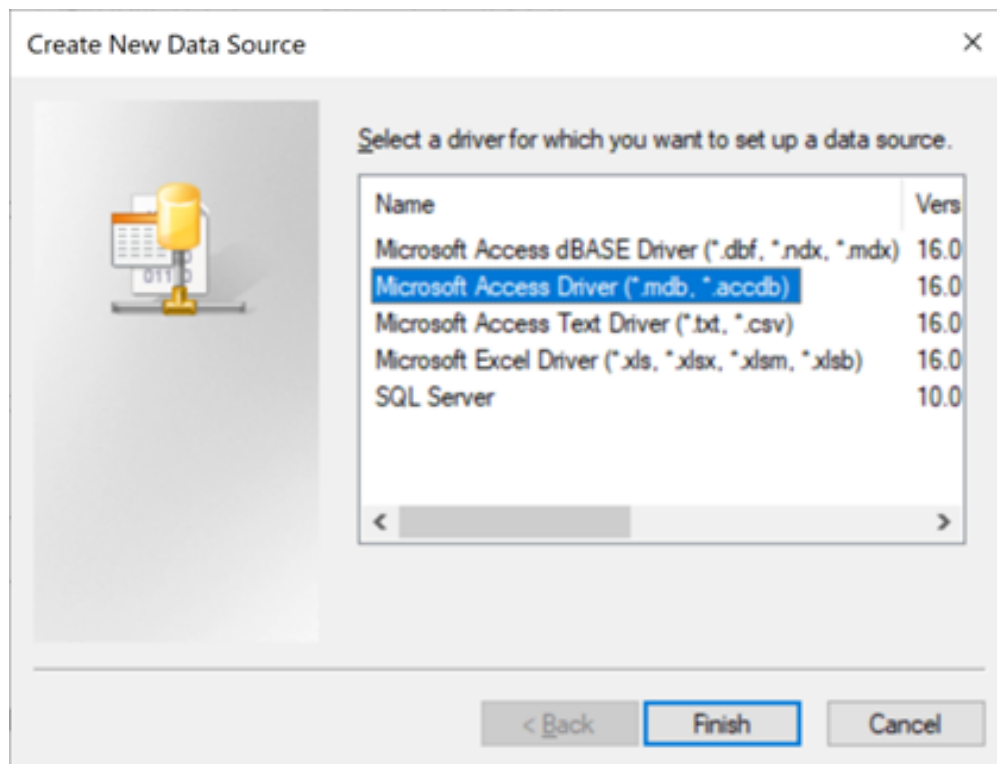
The Borland Database Engine, used in various VTW templates is 32-bit software that is no longer supported. To use existing VTW templates with Trio 4.x, these VTW templates must first be modified to instead use ODBC database drivers, with no dependency on the Borland Database Engine.

Follow the procedure below.

1. Freely download the **64-bit** version of *Microsoft Access Database Engine 2016 Redistributable* from Microsoft.
2. Install on the local computer.
3. You will now set up an ODBC connection to the Microsoft Access database, to register your database in the system
 - a. Run the *64-bit version (not 32 bit!)* of the ODBC Data Sources application.

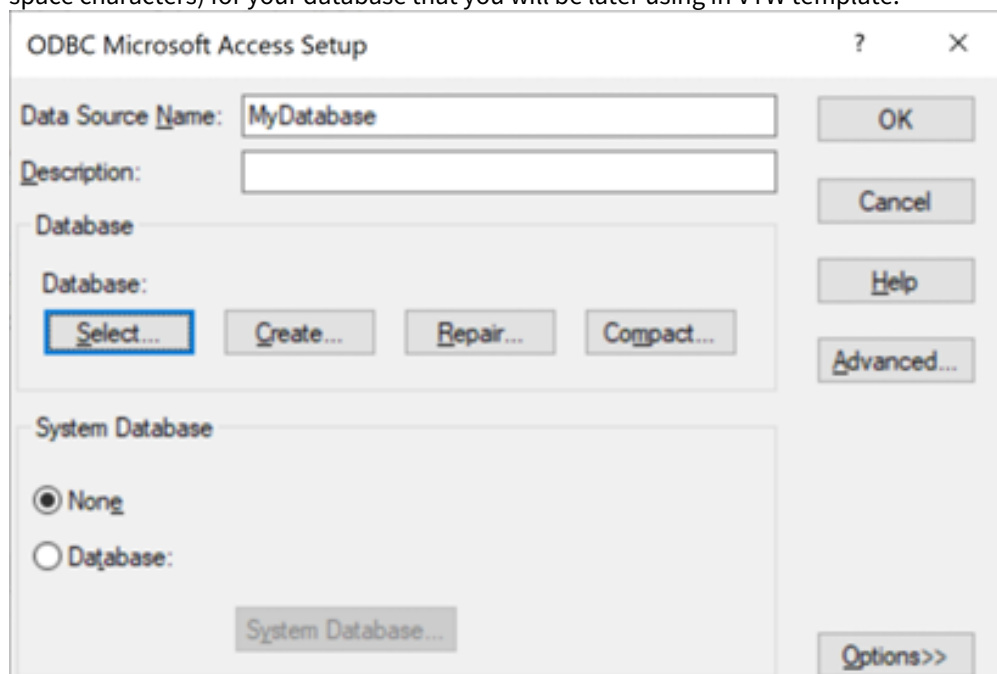


- b. Once ODBC Data Sources (64-bit) is running
- c. Select tab **User DSN**
- d. Click **Add**
- e. In **Create New Data Source** select the driver that matches your database type of your database. For example *Microsoft Access Driver*:

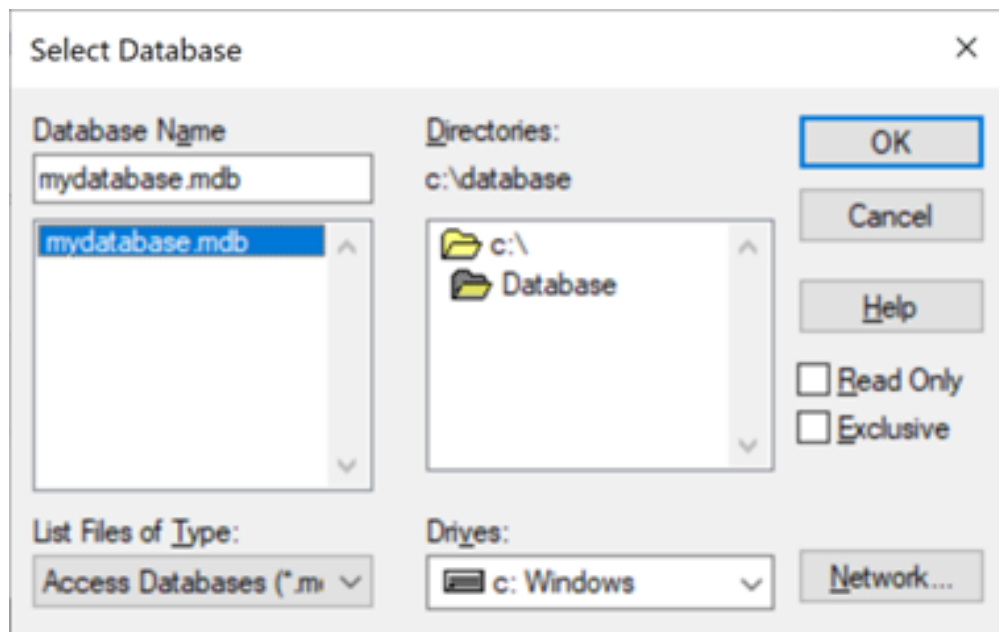


f. Click **Finish**.

g. In the **ODBC <dBase type> Setup** menu that appears provide an alias name (use a name without space characters) for your database that you will be later using in VTW template:



h. Click **Select**, then select the database file in filesystem:



- i. Click **OK**.
Your database is now registered in the system.
- j. In the ODBC Microsoft Access Setup dialog box, click **OK**.

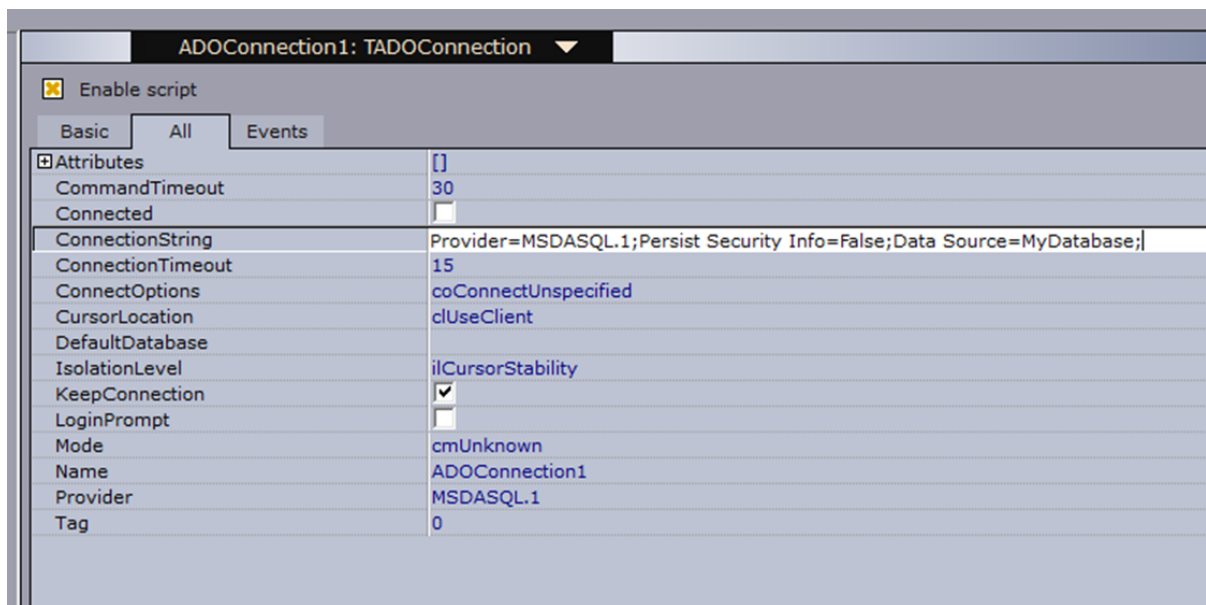
Modifying VTW Templates

Info:

- There is a lot of information about how to build connection string over the internet. This is generic in regard to ODBC database access technology
- You may actually use a connection string that points to the exact location of database file, so you do not need to register your database in the system.
However, this is a trial and error method, it is highly recommended to register a database alias as the default method.

To modify VTW templates

1. Start the **Template Wizard** application and open your VTW template.
2. Navigate to **Database** page and on the Component palette put two new components on a form:
3. ADO Connection
4. ADO Query
For example, their names are *ADOConnection1* and *ADOQuery1*
5. Assign **Connection property** of *ADOQuery1* to *ADOConnection1*
This makes the *query* component use the *connection* component.
6. For the property of *ADOConnection1*, clear the check value of **LoginPrompt**
(otherwise *username* and *password* will be requested for the Trio session, even if they are not used).
7. Set the **ConnectionString** property of *ADOConnection1* component to refer to the database you want to use.
For example, assuming sample screenshots above:
Provider=MSDASQL.1;Persist Security Info=False;Data Source=MyDatabase;



8. For those properties that are the same between this new *ADOQuery1* component and your existing Query component (let's say it has name *Query1*), for the *ADOQuery1* component set same values as in *Query1* component. This could be, for example, the **SQL** property.
9. (Example) If the existing *Query1* component has some event handlers code, create the same event handlers for *ADOQuery1* and copy the code there.
10. Delete your existing *Query1* component so only the *ADOQuery1* component remains.
11. Using Find and Replace, modify the script so it uses *ADOQuery1* where *Query1* was previously named.
12. Once you have deleted the *Query1* component, delete its event handlers code.
13. (Optional) You can control your *ADOConnection1* connection separately from queries.
 - a. You can open the connection once (*ADOConnection1.Connected* = *True*) and close it whenever you want.
 - b. When the connection is open, setting the *ADOQuery Active* property to *True* or *False* does not affect the connection state.
 - c. When the connection is closed, when set to **Active**, the first *ADOQuery* will also open the connection.
 - d. Closing the connection makes all possible *ADOQuery* instances inactive (closed) if they were active.
14. Save the template with a new name (to differentiate it from the old one).
15. Start Trio 4.x+
16. Open your show.
17. Open **Show properties**.
18. Click the ... icon next to the Active VTW template setting.
19. In the **VTW Templates** dialog box, select the existing VTW template and delete it.
20. Add and select new template that you've created at step 13.
21. Click **OK**.
22. Click **OK**.

The old template is replaced by the new one and is ready to use.

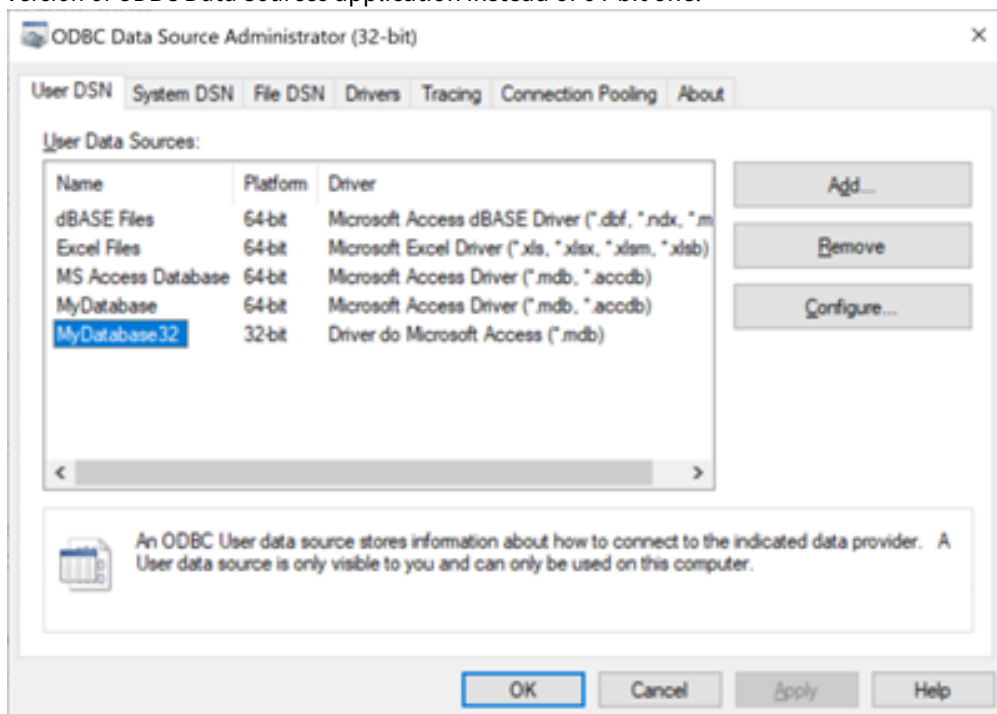
VTW template verification on Trio 3.x

To test upgraded VTW template using Trio 3.x

Technically you don't need to upgrade from Trio 3.x to upgrade your VTW templates from using Borland Database Engine to ODBC.

This route requires a planned switching to Trio 4.x+ which must avoid using BDE, while still using Trio 3.x.

- For this route, you follow the same upgrade steps as above, but register your database with the **32-bit** version of ODBC Data Sources application instead of 64-bit one.



Important usage Notes

- Be aware that registering database in 64-bit version of ODBC Data Sources application is crucial in regard of using the VTW template in Trio 4.x+.
- The 64-bit version of the ODBC Data Sources application allows creating, configuring and removal of only 64-bit connections.
The 32-bit version of the ODBC Data Sources application allows creating, configuring and removal of only 32-bit connections.

3 Configuration

Connect Viz Trio to Media Sequencer and Viz Engine, and set up Viz Trio for playout.

 **Note:** Before starting the Viz Trio client, make sure that Media Sequencer, Viz Engine and Graphic Hub have been installed, and are running and configured.

This section covers the following topics:

- [Hardware and License](#)
- [Viz Trio OneBox](#)
- [Configuration Window](#)
- [Command Line Parameters](#)

3.1 Hardware and License

Viz Trio needs a connection to Media Sequencer and a local Viz Engine in order to start successfully.

This section covers the following topics:

- [Hardware Configuration](#)
- [Connecting to Media Sequencer](#)
- [Connecting to Viz Engine](#)
- [License](#)
- [Ports](#)

3.1.1 Hardware Configuration

Preview/Program

Each Viz Trio system requires two standard desktop PCs for operation. One for the Viz Trio client, the second for the renderer, Viz Engine.

Viz Trio OneBox

You can also run a complete Viz Trio system, including the Viz Engine renderer, on a single PC (desktop or rack mountable) with all the features of a conventional Viz Trio setup. This setup requires two powerful graphics cards to ensure graphics quality and rendering speed. Both the VGA preview and final program signals (playout) can be viewed on the same machine.

 **Tip:** The single computer setup is suitable when you have limited space, such as in mobile environments (OB vans and remote broadcasts) and small studios.

3.1.2 Connecting to Media Sequencer

Parameters in the program target path determine the connection a Viz Trio client has to Media Sequencer (MSE).

To connect to a specified Media Sequencer

If no *mse* target path parameter is set, Viz Trio defaults to a *local* Media Sequencer.

- Right-click the Viz Trio desktop shortcut on the Windows desktop and set a parameter for the *-mse* setting in the **Target** path.

 **Example:** `C:\Program Files (x86)\vizrt\Viz Trio\trio.exe -mse localhost -control -lang en -loglevel 5`

3.1.3 Connecting to Viz Engine

Viz Trio needs a local Viz Engine for previews and to import scenes. Viz Trio automatically detects and runs Viz Engine.

Info:

- The same version of Viz Engine must be installed on all machines.
- All machines must connect to the same Graphic Hub.

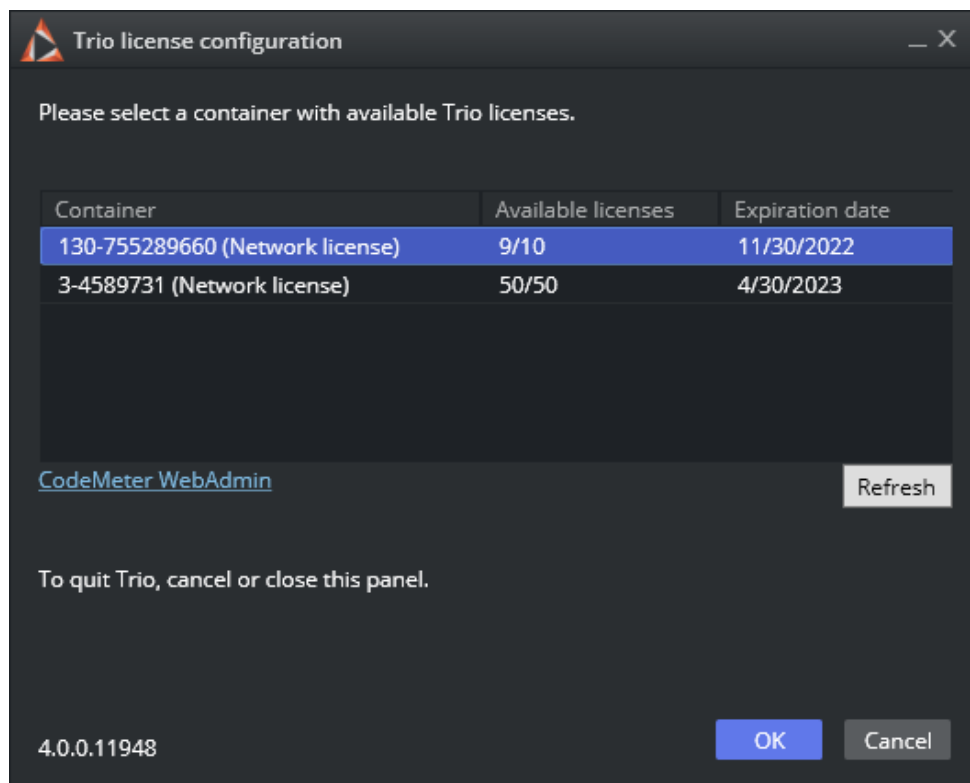
Please check [Viz Engine startup command line parameters](#).

3.1.4 License

The local Viz Engine, Viz Trio and Datacenter must be licensed through CodeMeter from WIBU systems. The CodeMeter runtime service must be running on the local Viz Trio host. CodeMeter is bundled with the local preview Viz Engine and the Viz Trio client installer. When retrieving the license from a dedicated License server, it must be configured in the Vizrt Licensing Service (see the *Client Configuration* page in the *License Server Setup and Administration* section of the [Vizrt Licensing Administrator Guide](#)) or the CodeMeter WebAdmin.

To manage license containers

When Viz Trio starts up for the first time, a dialog box prompts you to select among available license containers, locally and from the configured license server(s) on the network.



- Select a container and click **OK**. The container is remembered the next time you start Viz Trio.

- To switch container, go to **Help > Trio Licenses** to open this dialog box again.
- If there are no containers available, you may need to configure a licenser server. Click **CodeMeter WebAdmin** to open the configuration page.
- Click **Refresh** to retrieve available license containers.

3.1.5 Ports

Component	Protocol	Port	Description
<i>trio.exe</i>	TCP	6200	Used to execute macro commands of the Viz Trio client over a socket connection.
<i>trionle.exe</i>	TCP	6210	Used by the Graphics Plugin for NLE to utilize Viz Trio for effect editing.

3.2 Viz Trio OneBox

3.2.1 Viz Trio Configuration

Traditionally, each Viz Trio system required two standard desktop PCs to operate: one for the Viz Trio client and one for its renderer Viz Engine.

3.2.2 Viz Trio OneBox Configuration

You can also run a complete Viz Trio system, including the Viz Engine Program Output, on a single standard PC (virtualized, desktop or rack-mountable) with all of the features of a conventional Viz Trio setup. Two powerful graphic cards ensure the same graphics quality and rendering speed. Both the VGA preview and final program signals (playout) can be viewed on the same machine.

For the initial setup, it is recommended that you follow the configuration guide described in the Viz Engine documentation for Trio Box CG Mode.

 **Note:** The *-onebox* Trio startup parameter is a mandatory field for Viz Trio OneBox setups. The parameter *-force_gpu_count* is deprecated.

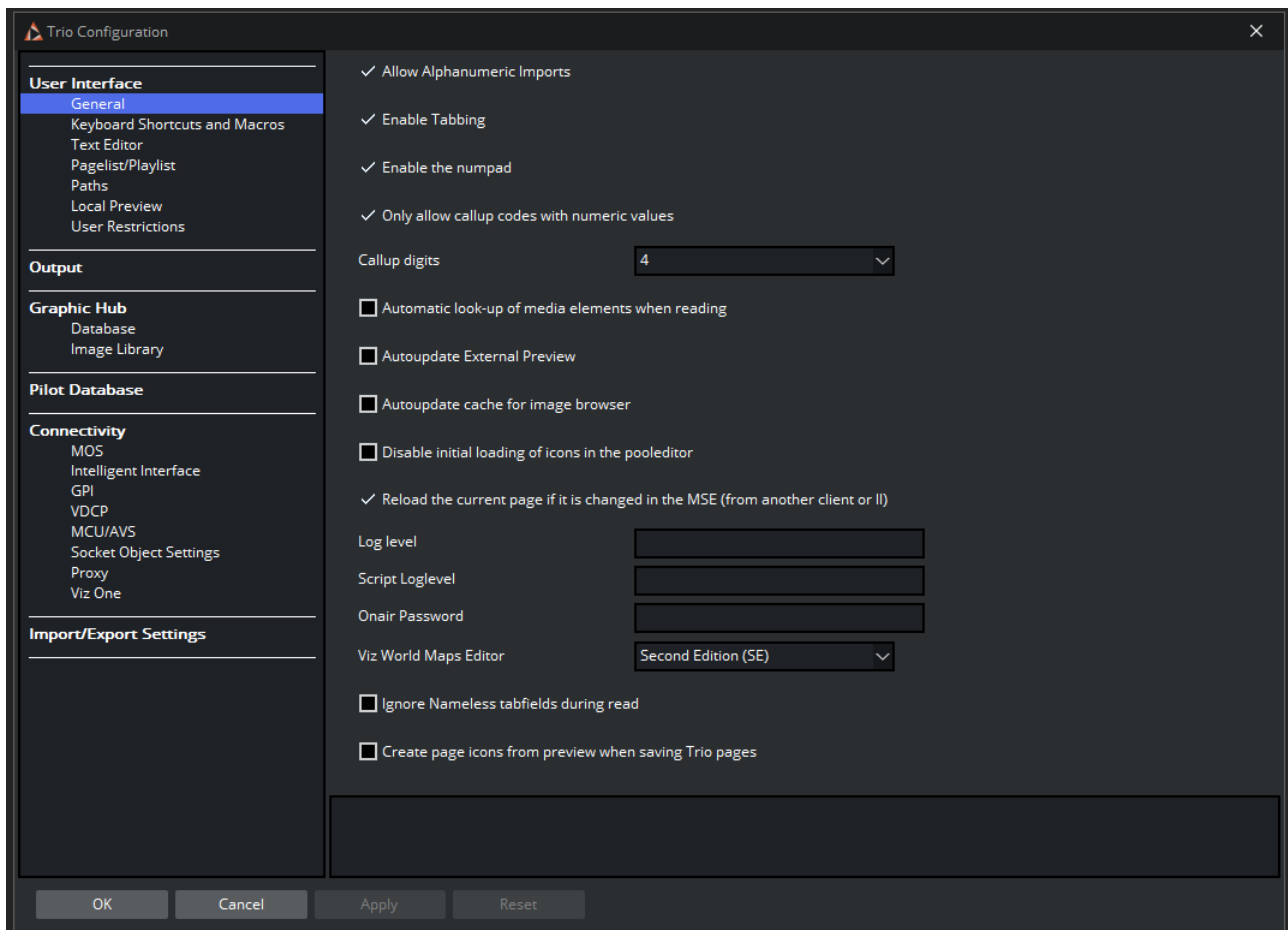
Viz Trio setups with more than one Graphical Processing Unit (GPU) may run both preview and program channels locally. The first time Viz Trio starts using the *-onebox* Viz Trio startup parameter, it automatically adds a Viz Engine host named *LocalProgramChannel* with host *localhost:6800*.

 **Note:** The *LocalProgramChannel* host is only created once, unless Media Sequencer is reset.

 **Tip:** The single PC setup is suitable when you virtualize Viz Trio, or have limited space, such as in mobile environments (OB vans and remote broadcasts) and small studios.

3.3 Configuration Window

Select **File > Configuration** to show the configuration window:



- **OK:** Applies any changes, and closes the window.
- **Cancel:** Closes the window.
- **Apply:** Applies any changes without closing the window.
- **Reset:** Resets all changes made, unless they are applied locally.

Select the menu item you want to display or change from the list on the left. Use the panel on the right to review or change setting values:

- [User Interface](#)
- [Graphic Hub](#)
- [Viz Pilot Database](#)
- [Import and Export Settings](#)
- [Output](#)
- [Connectivity](#)

3.3.1 User Interface

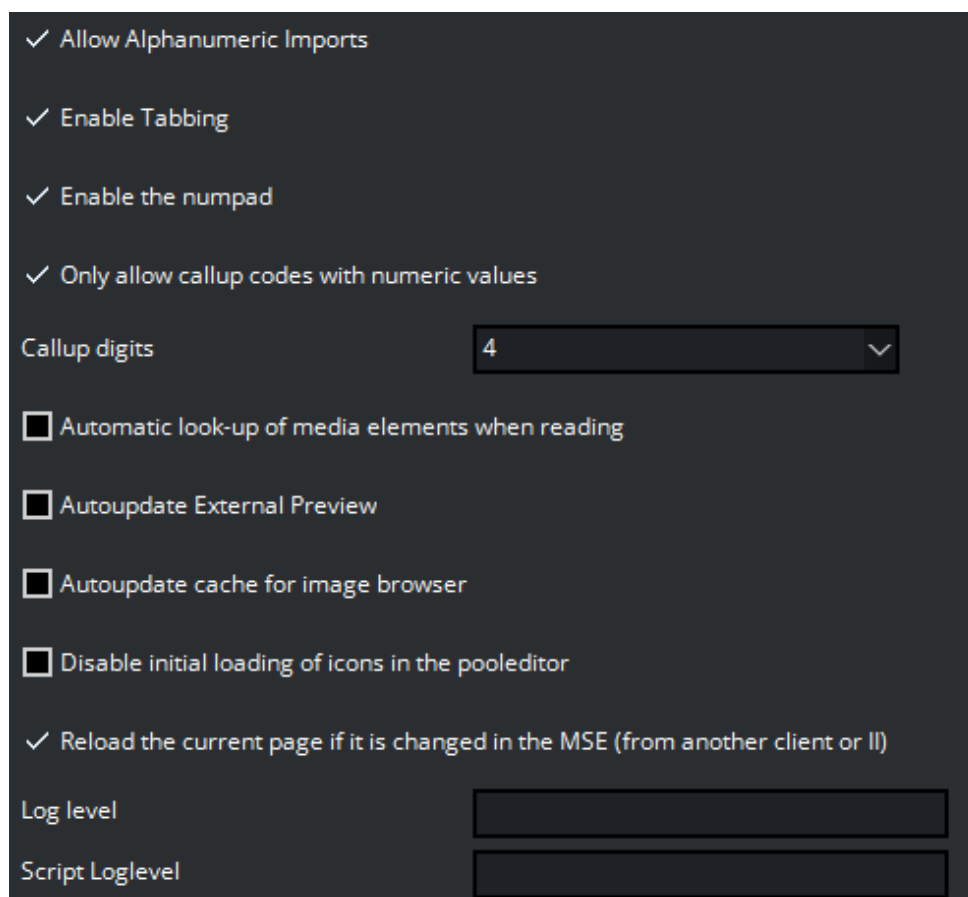
The settings below are specific to the local Viz Trio client and do not affect any other Viz Trio clients using the same Media Sequencer.

This section covers the following topics:

- [General](#)
- [Keyboard Shortcuts and Macros](#)
- [Text Editor](#)
- [Page List/Playlist](#)
- [Paths](#)
- [Local Preview](#)
- [User Restrictions](#)

General

To view the General configuration settings, click **User Interface > General** in the Viz Trio Configuration.



The screenshot shows the 'General' configuration settings for the Viz Trio client. The settings are as follows:

- ☒ Allow Alphanumeric Imports
- ☒ Enable Tabbing
- ☒ Enable the numpad
- ☒ Only allow callup codes with numeric values
- Callup digits:
- ☐ Automatic look-up of media elements when reading
- ☐ Autoupdate External Preview
- ☐ Autoupdate cache for image browser
- ☐ Disable initial loading of icons in the pooleditor
- ☒ Reload the current page if it is changed in the MSE (from another client or II)
- Log level:
- Script Loglevel:

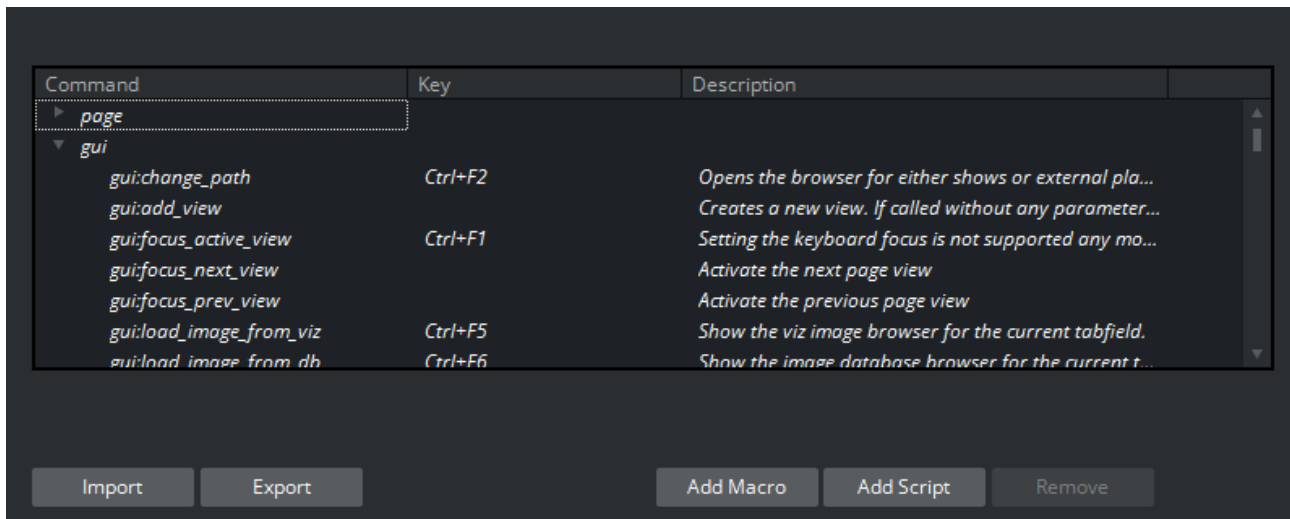
- **Allow Alphanumeric Imports:** Enable or disable import of scenes with alphanumeric names. When disabled, only scenes with numeric names are allowed.
- **Enable Tabbing:** When enabled, tabbing iterates through the tab-fields of the element currently read.

- **Enable the numpad:** When enabled, the numpad types directly into the callup field. Holding down the **CTRL** button disables this function.
- **Only allow callup codes with numeric values:** When checked, this setting only allows callup codes with numeric values when a page is read or saved. When unchecked, this setting allows all values.
- **Callup digits:** Sets the width of the two callup code edit fields on top of the Viz Trio main window. Selecting 8, for example, produces a wider code field, while selecting 2 produces a narrow code field. This setting has no impact on page name length.
- **Automatic look-up of media elements when reading:** When enabled, this option automatically displays the video in the video search area when a video element is read. Note that this slows down the reading of the element.
- **Autoupdate External Preview:** When selected, the external preview channel is automatically updated when a page is edited in the Viz Trio client. When not selected, the external preview channel only updates its preview when the page is saved.
- **Autoupdate cache for image browser:** When enabled (default), the Viz Trio client updates the local image browser cache automatically and adds any images that are new and updates those that have changed. With very large image trees this update process might slow down the system response. To achieve better system response when the image tree is very large, disable this option.
- **Disable initial loading of icons in the pooleditor:** When enabled, this setting prevents the Viz pool browser from being loaded the first time a tab field, with for example Viz geometries or images, is selected. The icons are loaded when clicking in the browser or changing the path.
- **Reload the current page if it is changed on the MSE (from another client or II):** If this setting is on, the currently read page is reloaded (read again) if it changes on the server.
- **Loglevel:** Defines what kind of log messages should be logged. Possible levels are: 0, 1, 2, 5, and 9. For more information, see [Logging](#) in Viz Trio.
- **Script Loglevel:** Sets which log level to use when running scripts. It defaults to -1 , which is no logging. So if a script triggers a viz command, the script log level must be set to 9 to see the Viz Engine command in the log file.

Keyboard Shortcuts and Macros

To view the Keyboard Shortcuts and Macros configuration settings, click **User Interface > Keyboard Shortcuts and Macros** in the Viz Trio Configuration.

Keyboard Shortcuts and Macro Settings



In this section, keyboard shortcuts can be defined for various operations. These shortcuts are global, which means they are valid for all page folders. Local keyboard shortcuts can be specified per show by adding keyboard shortcuts and macros under [Show Properties](#).

- **Import and Export:** The Import and Export function are used to import and export shortcut settings to and from an XML-file for backup, and reuse of typical configurations.

 **Note:** The keyboard file has a *.kbd file extension.

- **Add Macro:** Allows an operator to write a Viz Trio macro command, and link it to a shortcut key.
- **Add Script:** Allows an operator to write a Visual Basic script, and link it to a shortcut key. If a key combination is already in use, you are given the option of overriding it. This leaves the other command without an assigned keyboard shortcut.

Macros and scripts that are custom made can be removed using the Remove button. Select the custom made shortcut and click Remove.

 **Note:** No warning is issued when performing this operation.

 **IMPORTANT!** Avoid using **SHIFT** as part of a command as this disables the possibility to use the assigned key when writing characters (for example, in upper case).

Usage hints for the default keyboard bindings:

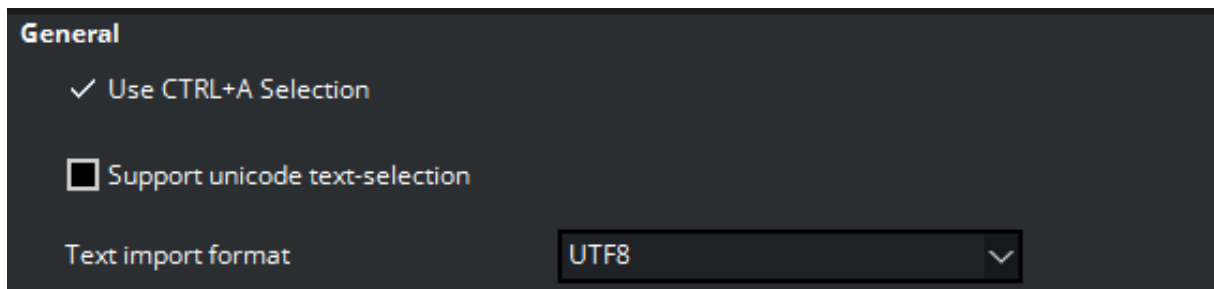
- **CTRL + N** and **CTRL + P** are default bound to *tabfield:next_property* and *tabfield:previous_property*. They can be used to navigate between several tab-field properties under one tab-field, for example when working with text and kerning.
- **CTRL + TAB** and **CTRL + SHIFT + TAB** can be used to move keyboard focus around inside the tab-field editors.
- **TAB** and **SHIFT + TAB** can be used to move keyboard focus around as if the page editor is *not* active or *no page is read*.

Text Editor

To view the Text Editor configuration settings, click **User Interface > Text Editor** in the Viz Trio Configuration.

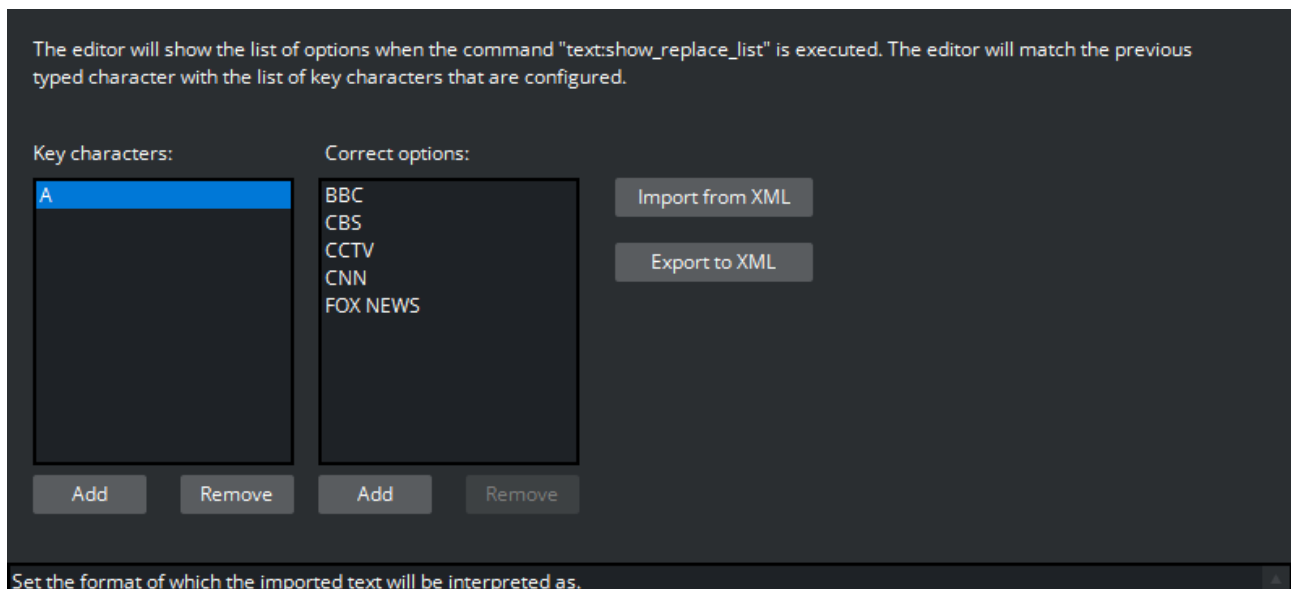
The text editor contains general settings related to text formats and how to replace key characters.

General



- **Use CTRL + A Selection:** When deselected, the normal key combination for selecting all text, **CTRL + A** is disabled.
- **Support Unicode text-selection:** Use this setting to enable or disable support for Unicode text-selection. This option is used if the default language for non-Unicode programs in Windows is set to a Unicode language. For example: If copying text from a text editor that by default does not support Unicode language into Viz Trio. The representation of text would not be correct.
- **Text import format:** This option allows a character set to be specified for text imported and interpreted by Viz Trio. By default this is set to UTF-8.

Word Replace



Word replace allows key characters to have replace words defined. The list of replacement options becomes available in the text editor when running the `text:show_replace_list` command, see the [Keyboard Shortcuts and Macros](#) section on how to define keyboard shortcuts.

The text editor matches the typed character with the list of key characters configured, and displays the corresponding replacement list. The word correction function is case sensitive, making it possible to define replacement options for both lower and upper case letters. The list of word correction options is local.

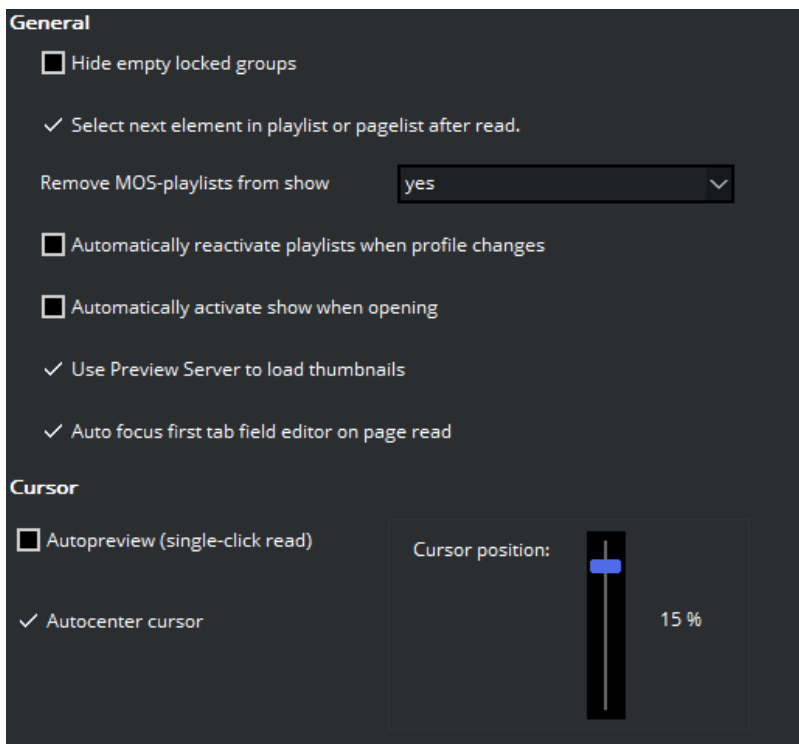
- **Key characters:** Shows a list of defined replacement key characters.
- **Correct options:** Shows a list of a key character's corresponding replacement options.
- **Add:** To add *Correct options*, a key character must be added first. To add *Correct options*, select the key character, and click the associated Add button.
- **Remove:** Select a list item to remove, and click the associated Remove button.
- **Folder:** Imports a predefined list exported to an XML file.
- **Save:** Exports the predefined list to an XML file.

Example

```
<entry name="customcharacters">
  <entry name="A">
    <entry name="CBS"></entry>
    <entry name="CNN"></entry>
    <entry name="BBC"></entry>
    <entry name="ITN"></entry>
    <entry name="NBC"></entry>
    <entry name="NRK"></entry>
    <entry name="YLE"></entry>
    <entry name="SVT"></entry>
    <entry name="CCTV"></entry>
    <entry name="TV2"></entry>
    <entry name="ZDF"></entry>
    <entry name="RTL"></entry>
  </entry>
</entry>
```

Page List/Playlist

To view the Page List/Playlist configuration settings, click **User Interface > Page List/Playlist** in the Viz Trio Configuration.



- **Hide empty locked groups:** This setting prevents locked groups with no pages from being displayed in the page list.
- **Select next element in playlist after read:** When enabled, the next element in the playlist is automatically selected after a read operation.
- **Remove MOS-playlist from show:** Removes newsroom playlists that are monitored using the MOS protocol when the show is closed. Setting this to Yes also affects other clients connected to the same Media Sequencer. The available options are: Yes, No and Ask.
- **Automatically reactivate playlists when profile changes:** If enabled, active playlists are reactivated when the current profile is changed.
- **Automatically activate show when opening:** If selected, automatically activate the show when opened. Please be aware that this option, if used, could put a high load on Media Sequencer.
- **Use Preview Server to load thumbnails:** If selected, use [Preview Server](#) to load thumbnails.
- **Auto focus first tab field editor on page read:** When enabled, automatically switches focus from the page list or playlist, to the page field of the selected page.
- **Autopreview (single-click read):** When enabled, a page is read as soon as it is selected.
- **Autocenter cursor:** With this function enabled, the cursor has a static position when the page list is longer than the Page List window. In these situations the window is provided with a scrollbar on the right side. While enabled, as long as some pages are hidden in the direction the list is browsed, the cursor remains static and the list items move instead. When the last item is revealed, the cursor starts to move again.

Paths

To view the Paths configuration settings, open **File > Configuration > User Interface > Paths** in the Viz Trio Configuration.

This section describes the directories settings for several functions. All settings are local to the specific Viz Trio client.

Paths Configuration Settings

Avi Path		...
Dif Path		...
Image Path		...
Audio Path		...
Script Path		...
Trio Logfile Path		...
MSE As-Run Logfile	c:\temp\log\log.txt	
Shared Temp Path		...
Help Path		...

- **Avi Path:** Sets the default location when browsing for AVI files.
- **Dif Path:** Sets the default location when browsing for DIF files.
- **Image Path:** Sets the default location when browsing for Image files.
- **Audio Path:** Sets the default location when browsing for Audio files.
- **Script Path:** Sets the default directory to file-based scripts that may be assigned to a Viz Trio show and/or template.
- **Trio Logfile Path:** Sets the path to the directory where the Viz Trio client should store log files. For more information, see [Changing the Log File Path](#).
- **MSE As-Run Logfile:** Sets the path to Media Sequencer *as-run* log file. This log file offers a Media Sequencer logging mechanism for all elements that have been taken on air. The as-run log file is an additional debugging tool which a Viz Trio administrator may want to enable to verify that the communication between Media Sequencer and the playout engine is working properly. The file name may be given with either relative or absolute path.

 **Note:** This is a global setting, affecting all clients connecting to the specified Media Sequencer. Also note that the as-run log file ends up on Media Sequencer's host, not on the local Viz Trio machine. Read more about the as-run log file in Media Sequencer's documentation at http://<mse-host>:8580/mse_manual.html#as-run-log.

- **Shared Temp Folder:** Used for temporary files when importing or exporting Viz Trio shows on an external Viz Engine. This must be a network share that is both accessible for the client and the external Viz Engine.
- **Help Path:** Sets the path for the main help file. Help opens when pressing **F1** or when clicking *View Help* under **Help** in the main menu.

Local Preview

To view the Local Preview configuration settings, click **User Interface > Local Preview** in the Viz Trio Configuration.

Local Preview Configuration Settings

- **Viz Engine connection timeout:** Sets the number of milliseconds until the local Viz Engine renderer times out. Changes to this setting requires a restart of Viz Trio.
- **Font encoding:** Sets the font encoding format of text sent to the local preview. This setting overrides the local Viz preview and requires a restart of Viz Trio to take effect.

Note: This does not have an effect on remote Viz Engines; hence, the font options for Viz must be set accordingly. Also note that the Viz Trio interface only supports the font encoding set for the operating system under the *Regional and Language Options* settings (i.e. support for complex script, right-to-left and East Asian languages).

- **Single layer transition logic preview:** Enable previewing of single-layer pages in the background scene. If this setting is disabled, only Combo pages are previewed in the background scene.
- **Use animation when previewing transitions logic elements:** If this setting is off, the transition logic animation is skipped in the local preview. This requires that each foreground scene defines a "pilot1" tag on a director. By default, this setting is on. In the *Local Preview Resolution* section the aspect ratio for the local preview window can be set, and the systems refresh rate defined (normally 50 for PAL and 60 for NTSC).

User Restrictions

To view the User Restrictions configuration settings, click **User Interface > User Restrictions** in the Viz Trio Configuration.

User Restriction Settings

Do not allow users to:		
Deny	Description	Corresponding Command
▼ ☐	Scripts	
☐	Edit scripts	gui:show_scripteditor
▼ ☐	Post Rendering	
☐	Enter new hostnames in post render view	
▼ ☒	Shows	
☒	Browse viz directories when changing shows	
☐	Browse external playlists	
☐	Create or rename shows	
☐	Access show properties	gui:show_settings
☐	Open the window for importing templates into a show	
☐	Delete all pages in a show	show:delete_all_pages
▼ ☒	Page	
☐	Takeout Pages	page:takeout
☐	Direct Takeout Pages	page:direct_takeout
☐	Direct Takeout Pages from a Playlist	playlist:direct_take_out_selected
☐	Delete Templates	show:delete_templates
☐	Save Templates	page:save
☒	Save values to linked databases	
☐	Rename pages	show:rename_page
▼ ☐	Trio	
☐	Cleanup External Renderers	trio:cleanup_renderers
☐	Call Viz Trio Commands	gui:show_triocommands
☐	Switch to Viz Artist	gui:artist_mode
▼ ☐	Scroll Editor	
☐	Create new scroll templates	gui:show_scroll_template_creator

The User Restrictions section lets an administrator deny the operator from performing certain actions in Viz Trio.

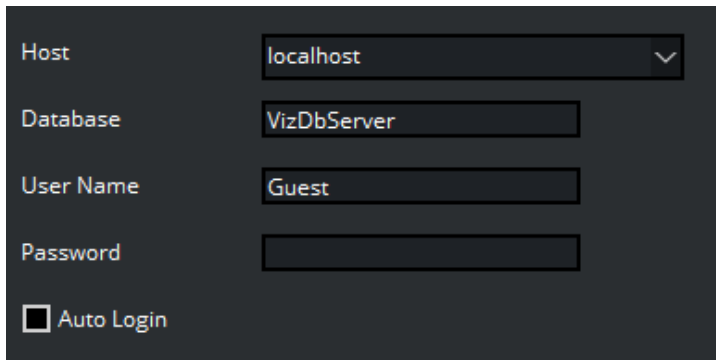
- **Page:**
 - **Delete Templates:** When checked, prevents the operator from deleting templates from the show. Command: `show:delete_templates`.
 - **Direct Takeout Pages:** When checked, prevents the operator from issuing a direct take out command on pages in a show. This also disables the context menu option for the show. Command: `page:direct_takeout`.

- **Direct Takeout Pages from a Playlist:** When checked, prevents the operator from issuing a direct take out command on pages in a playlist. Command: `playlist:direct_take_out_selected`.
- **Rename pages:** When checked, it is not possible to rename saved pages in the page list.
- **Save values to linked databases:** When checked (default), prevents the operator from saving values back to the database.
- **Takeout Pages:** When checked, prevents the operator from using the Take Out command. This also disables the Take Out button in the Page editor. Command: `page:takeout`.
- **Save Templates:** When checked, the user cannot save modifications to a template. Command: `page:save`.
- **Post Rendering:**
 - **Enter new hostnames in post render view:** When checked, prevents the operator from adding new hostnames to the Render Videoclip editor.
- **Scripts:**
 - **Edit scripts:** When checked, prevents the operator from editing the script using the Edit Script button in the Page editor. Command: `gui:show_scripteditor`.
- **Scroll Editor:**
 - **Create new scroll templates:** When checked, prevents the operator from creating new scroll templates. Command: `gui_show_scroll_template_creator`.
- **Shows:**
 - **Access Show Properties:** When checked, prevents the operator from accessing the Show Properties. Command: `gui:show_settings`.
 - **Browse external playlists:** When checked, disables the [Playlists](#) tab under [Toolbar](#), and consequently preventing the operator from browsing for playlists on Media Sequencer.
 - **Browse viz directories when changing shows:** When checked, disables the [Viz Directories](#) tab under [Toolbar](#), and consequently preventing the operator from browsing the Viz directory to set a show path.
 - **Create or rename shows:** When checked, prevents the operator from creating new [Shows](#) or renaming existing shows.
 - **Delete all pages in a show:** When checked, prevents the operator from selecting and deleting all pages in the show in one operation. Normal page by page and page group deletion is still possible.
 - **Open the window for importing templates into a show:** When checked, prevents the operator from importing new scenes from Graphic Hub (Viz 3.x) or the data root (Viz 2.x).
- **Trio:**
 - **Call Viz Trio Commands:** When checked, prevents the operator from manually calling Viz Trio Commands. This does not prevent scripts from doing the same. Command: `gui:show_triocommands`. See [Working with Macro Commands](#).
 - **Cleanup External Renderers:** When checked, prevents the operator from issuing a cleanup renderer command. Command: `trio_cleanup_renderers`.
 - **Switch to Viz Artist:** When checked, prevents the operator from switching to Viz Artist mode. Command: `gui:artist_mode`.

3.3.2 Graphic Hub

To view the Graphic Hub configuration settings, click **Graphic Hub** in the Viz Trio Configuration.

Graphic Hub Configuration Settings



Host: localhost

Database: VizDbServer

User Name: Guest

Password:

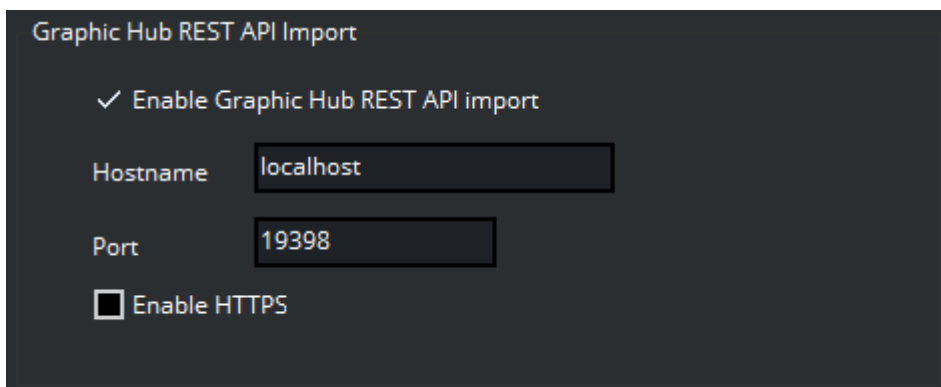
☐ Auto Login

These are the configuration settings to log on to Graphic Hub. All settings are local to the specific Viz Trio client.

- **Host:** Sets the hostname or IP address for the Viz DB server.
- **Database:** Sets the name of the Graphic Hub database.
- **User Name:** Sets the user name used for logging on to the database.
- **Password:** Sets the password used for logging on to the database.
- **Auto Login:** Enables Viz Trio to log on automatically at startup.

Note: If the local Viz Engine's Auto Login is enabled, this option is disabled.

Graphic Hub REST API Import



Graphic Hub REST API Import

☒ Enable Graphic Hub REST API import

Hostname: localhost

Port: 19398

☐ Enable HTTPS

These are the configuration settings to enable and configure scene import using Graphic Hub REST API.

- **Enable Graphic Hub REST API Import:** Enables default usage of Graphic Hub REST API Import when importing scenes. Will fallback to normal import if not possible to import with REST API.
- **Hostname:** Sets the hostname of the Graphic Hub REST API service.
- **Port:** Sets the Port of the Graphic Hub REST API service.
- **Enable HTTPS:** Enables usage of HTTPS protocol in Graphic Hub REST API Import.

Image Library

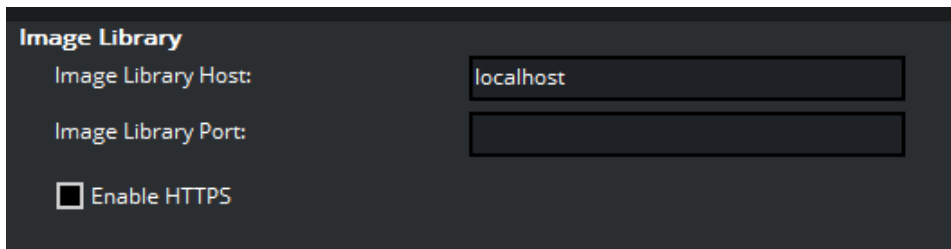
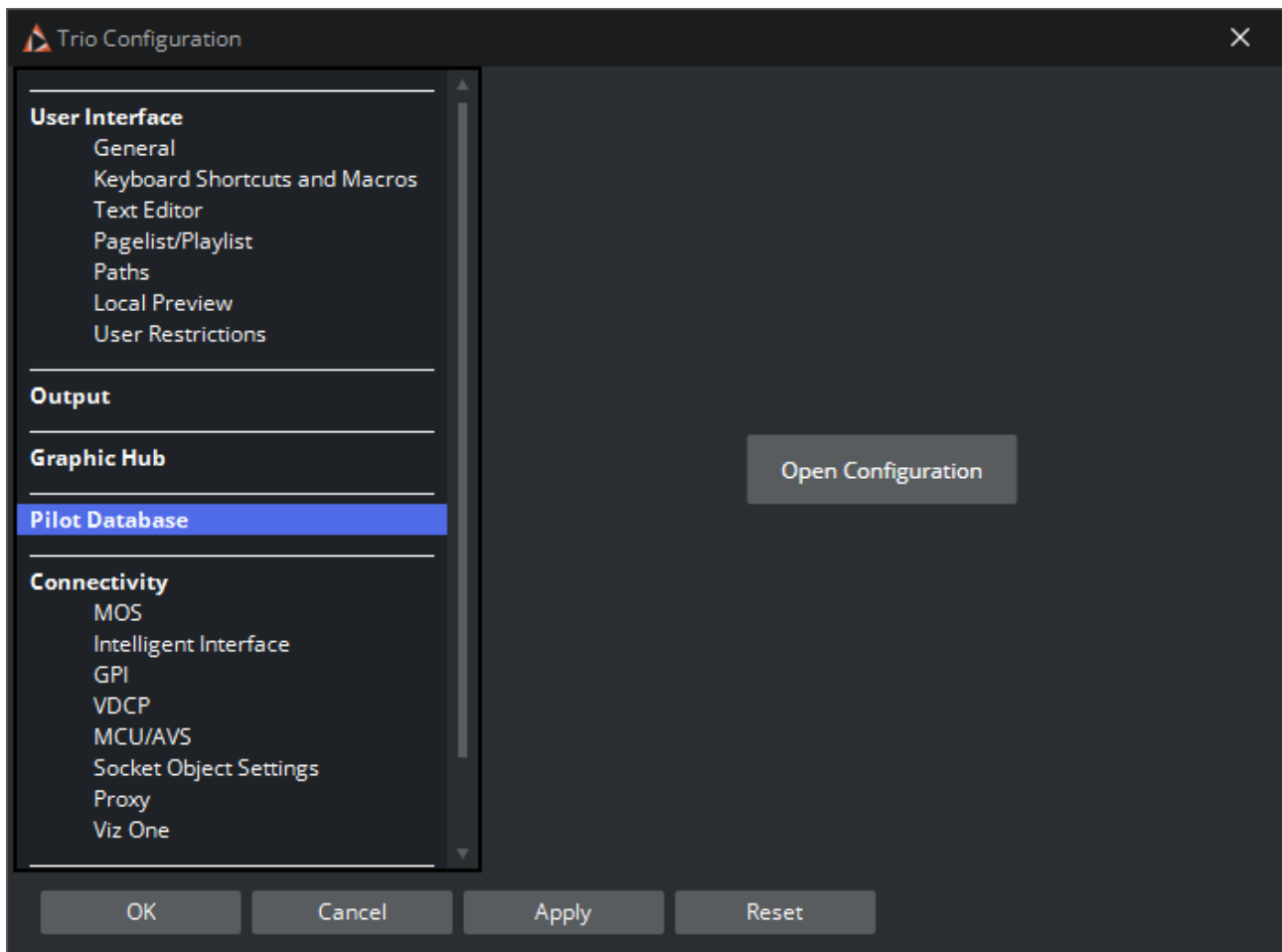
A dark-themed configuration panel titled "Image Library". It contains three settings: "Image Library Host:" with a text input field containing "localhost", "Image Library Port:" with an empty text input field, and an unchecked checkbox labeled "Enable HTTPS".

Image Library	
Image Library Host:	localhost
Image Library Port:	
☐ Enable HTTPS	

These are the configuration settings to enable the Graphic Hub Image Library Extension.

- **Host:** Sets the hostname of the Graphic Hub Image Library REST API service.
- **Port:** Sets the Port of the Graphic Hub Image Library REST API service.
- **Enable HTTPS:** Enables usage of HTTPS protocol in Graphic Hub Image Library Extension.

3.3.3 Viz Pilot Database



New from Viz Trio 4.0 is that the integration with the Viz Pilot Database is configured through the configuration web page hosted and served by Media Sequencer. You can find it in the address: <http://localhost:8580/app/pilotdbconfig/pilotdbconfig.html>, where the localhost is the host running Media Sequencer. When clicking the **Open Configuration** button, Viz Trio opens a web browser window showing the Media Sequencer web configuration for Pilot database:

Pilot Database Configuration (MSE: localhost)

☒ Pilot Data Server

Host :

Port :

☐ Oracle

Connection String :

Schema :

☒ Immediately remove deleted elements

Pilot Data Server

- **Host:** Sets the host name or IP of Pilot Data Server.
- **Port:** Sets the port of Pilot Data Server.

Oracle

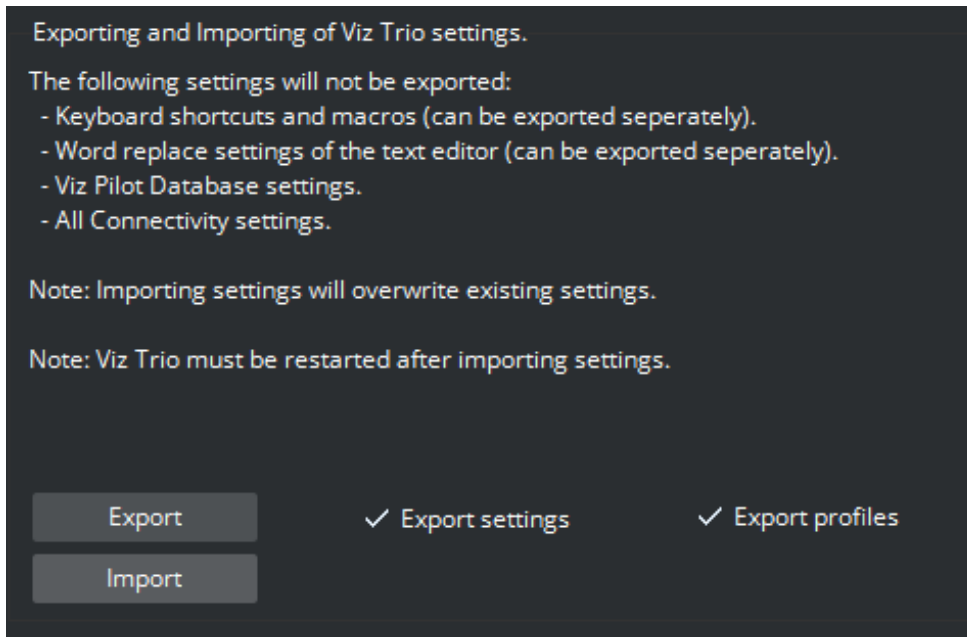
- **Connection String:** To configure a connection to the database (<username>/<password>@<hostname>/<SID>).
- **Schema:** Enter the name of the Oracle schema (default schema: *PILOT*).

Note: The database schema name must always be written in upper case.

3.3.4 Import and Export Settings

To view the Import/Export configuration settings, click **Import/Export Settings** in the Viz Trio Configuration.

Import and Export Configuration Settings



This function lets you import and export the configuration settings from and to an XML-file, letting you apply customized settings to Viz Trio clients without manually configuring the settings each time.

Export settings and **profile** configurations to an XML file. The following settings *cannot* be exported:

- [Keyboard Shortcuts and Macros](#) settings (can be exported separately).
- Word replace settings under the [Text Editor](#) settings (can be exported separately).
- [Viz Pilot Database](#) settings.
- All [Connectivity](#) settings.

Import Viz Trio settings and profile configurations from an XML file.

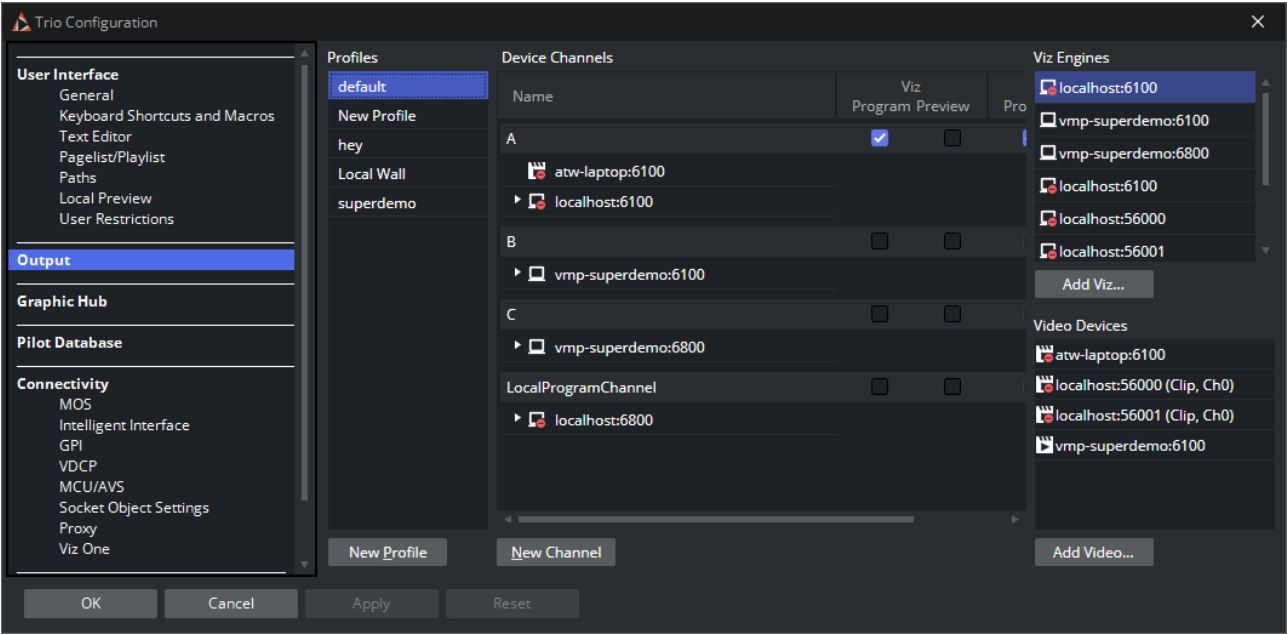


Note: Import overwrites existing settings and profiles. Viz Trio must be restarted for the new configurations to take effect.

3.3.5 Output

To view the Output configuration settings, click **Output** in the Viz Trio Configuration.

Output Configuration Settings



A key feature is the ability to create profiles for different purposes. For example, [Profile Setups](#) can be created where different channels have the program and preview function. This makes it fast and easy to switch between different output settings.

Profiles can be defined with different program and preview channels in the Output channels section. It is most common to define a studio with channels mapped to renderer machines (Viz Engine hosts).

Viz Trio supports four methods to resolve which channel is the local program channel. If an unsupported method is used, the [Connection Status](#) button does not appear in the [Status Bar](#). Consequently, the operator isn't able to control the local program channel status.

Resolving the Local Program Channel Name

Method	Comment
localhost	For the program channel; remember to add the port 6800.
127.*.*.*	Not only 127.0.0.1 refers to localhost.
Hostname	Name of computer without domain.

Method	Comment
IP	The computer's IP. If the computer has multiple IPs (e.g. it has multiple network cards), it may still fail.

✖ IMPORTANT! Although the computer may have several names (for example in /etc/hosts), only the NetBIOS name can be used (the name obtained when running the "hostname" command). It is not possible to add the domain (for example <hostname>.domain.internal).

This section covers the following topics:

- [Program and Preview Channels](#)
- [Profile Setups](#)
- [Forked Execution](#)
- [Working With the Profile Configuration](#)

Program and Preview Channels

Local preview is Viz Trio's embedded render window in the lower right corner, which always shows the last page read. Local preview displays graphics as WYSIWYG (what you see is what you get) when editing a page. Text and images change immediately as text and images are added to the page. Select which elements to edit using point and click directly in the render window.

External preview (optional) is a separate Viz Engine renderer with its own reference monitor that displays a true preview. External preview shows how the program channel looks before a page is taken to air.

Program is a separate Viz Engine renderer that renders the content that is taken on-air.

Profile Setups

Profile	Program	Preview
Main	A (Viz 1)	B (Viz 2)
Backup	B (Viz 2)	

Two-channel Setup

A typical profile setup is a two-channel setup, program and preview, with one render engine assigned to each channel. A more advanced setup is the use of forked execution. As the term suggests, forked execution is a way to configure a single output channel to contain multiple render engines.

Profiles are used to create different setups, for example in a backup configuration where a switch from a main renderer to a backup is needed. If two output renderers are named A and B, where A is program and B is preview, a profile named "Main" has channel A=program and channel B=preview.

If the renderer that is acting as program (A) in the "Main" profile fails, a profile named "Backup" could have channel B=Program.

GPI

Control payout through a General Purpose Input (GPI), for instance through hardware such as a vision mixer.

When GPI is enabled, the external cursor (the GPI system's cursor) is displayed in any client that is using the same profile as the external system. A typical setup would be that one Viz Trio client is in the same profile as the GPI system, and functions as a "prepare station" for the producer at the vision mixer desk. Data elements are then made ready and displayed on a preview visible to the producer, and the elements are then triggered from the vision mixer.

 **Note:** This configuration requires a separate "GPI" profile that is not used by other control application clients. Other clients can be in other profiles and produce content to the same output channels. However, they need to be on other transition layers or on another Viz layer so they don't interfere with the graphics controlled by the external system.

A channel can be designated as a Program or Preview channel by selecting the appropriate check box. The program and preview channels are reciprocally exclusive, only one channel can be set to program, and only one channel can be set to preview. For example, if A is set as program and B is set to preview and C is set to program, A is no longer set as program.

When configured to use video in graphics from a video server (for example Viz One), it is currently recommended to use IP addresses for both Viz and Video channels in order to visualize the transfer progress correctly. Hence, a hostname on the Viz channel works, but not on the Video channel. They must also match the video configuration for both program and preview.

The Video device configuration is used when configured to trigger video elements in a video server setup.

 **IMPORTANT!** Current limitations require that both graphics and video channels use an IP address.

Forked Execution

Forked execution can be used with standalone and transition logic scenes, and replaces the execution of visible containers.

This section covers the following topics:

- [Standalone scenes](#)
- [Transition logic scenes](#)
- [Visible containers](#)

Standalone Scenes

Forked execution supports standalone scenes by executing the same graphics with different concepts on two or more render engines. Concepts are defined per channel when [Working With the Profile Configuration](#) tool.

As an option you can also use this setup to handle fail situations by having the same graphics concept rendered on both engines.

Transition Logic Scenes

As with [Standalone scenes](#), forked execution supports setting concepts for Transition Logic scenes. In addition Transition Logic scenes, by defining channels with different render engine setups, can show different states of the

same scene on a per engine basis. All states are synchronized for all engines (at all times) in order to achieve an artifact-free and smooth morphing of the graphics from one state to the other.

For example, if you have three render engines you can set up a range of channels with different combinations of render engines per channel.

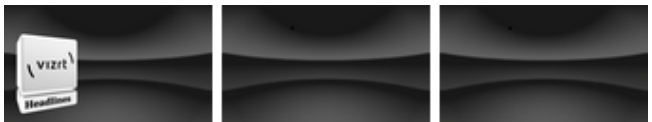
Channel	Viz Engines
A	1,2,3
B	1,3
C	1
D	2,3
E	2

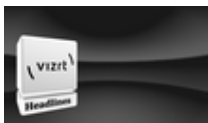
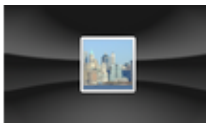
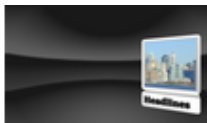
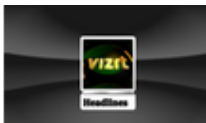
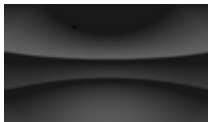
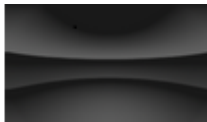
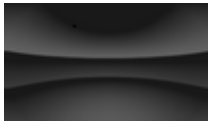
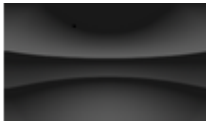
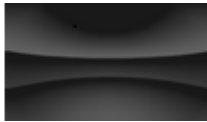
For example, if a scene has four layers, each layer can be controlled separately from the other layers (see table below). By setting a state per layer you can achieve a varied output depending on the channel used and how that channel is configured in terms of render engines (see table above).

- Layer 1: Shows and hides a geometry (for example, a cube).
- Layer 2: Shows and hides some text.
- Layer 3: Positions the geometries.
- Layer 4: Animates the layer 1 geometry by showing the next image or a logo.

States for Layer 1	States for Layer 2	States for Layer 3	States for Layer 4
Show cube	Show text	Position left	Next image
Hide cube	Hide text	Position right	Show logo
<ignore>	<ignore>	Position center	<ignore>
		<ignore>	

With the scene layers and configured channels above you can have the following output on each of the three engines:

Channel	Layer State	Output on Viz Engine 1, 2 and 3
Channel C(Viz 1)	Show cube Pos left Show text Show logo	

Channel	Layer State	Output on Viz Engine 1, 2 and 3
Channel D(Viz 2,3)	Show cubePos rightShow TextShow Logo	  
Channel E(Viz 2)	Pos centerHide text	  
Channel A(Viz 1,2,3)	Next image	  
Channel A(Viz 1,2,3)	Next image	  
Channel A(Viz 1,2,3)	Show textNext image	  
Channel A(Viz 1,2,3)	Show logo	  
Channel B(Viz 1,3)	Hide cube	  
Channel E(Viz 2)	Hide cube	  

Visible Containers

Although it is considered deprecated, a behavior similar to that of forked execution is supported. By designing a standalone scene where each root container is a variant of the other, you can configure each render engine to render specific containers (by name). In effect, a scene with two or more root containers can have one or more containers assigned and rendered visible by a single render engine.

 **Note:** Due to potential performance issues when using large textures (e.g. HD), this option is no longer recommended. Concept and variant design of standalone and transition logic scenes is therefore the recommended design convention.

Working with the Profile Configuration

This section covers the following topics

- **Profile Configuration:**
 - [Open Profile Configuration](#)
 - [Add a New Profile](#)
 - [Rename a Profile](#)
 - [Delete a Profile](#)
- **Channel Configuration:**
 - [Add an Output Channel to the Channels List](#)
 - [Rename an Output Channel in the Channels List](#)
 - [Remove an Output Channel from the Channels List](#)
 - [Add an Output Device to the Channels List](#)
 - [Remove an Output Device from the Channels List](#)
 - [Add a Concept Override for a Channel Output Device](#)
- **Output Device Configuration:**
 - [Add a Viz Engine](#)
 - [Add a Video Device](#)
 - [Edit a Viz Engine or Video Device](#)
 - [Delete a Viz Engine or Video Device](#)
 - [Enable Scene Transitions](#)
 - [Enable Still Preview](#)

Profile Configuration

Open Profile Configuration

1. Click the **Config** button (upper left), and then select **Output**.
2. Right-click the profile on the status bar (lower-left) and select **Profile Configuration...** from the menu.

Add a New Profile

- In the Profile Configuration window, click **New profile**, enter a new unique profile name in the text field and press **Enter**.

Rename a Profile

1. Right-click the profile and select **Edit Profile Name** or simply double-click the profile and enter a new name.
2. When finished editing the name, press **Enter** or click outside the Profiles list.

Delete a Profile

- Right-click the profile and select **Delete Profile**, or simply select it and press the **Delete** button.

Channel Configuration

Add an Output Channel to the Channels List

Name	Viz Program Preview	Video Program Preview
A	☐	☐
Drop here		

- Click **New channel** or drag and drop a Viz Engine or video device to the Channels list.

Rename an Output Channel in the Channels List

- Right-click the channel and select **Edit Channel Name** from the menu or double-click the name.

Remove an Output Channel from the Channels List

Name	Viz Program Preview	Video Program Preview
Channel	☒	☐
▶ VizEngine1	☐	☐
▶ VizEngine2	☐	☐

- Right-click the channel entry and select **Delete channel** from the menu or select the channel and press the Delete button.

Add an Output Device to the Channels List

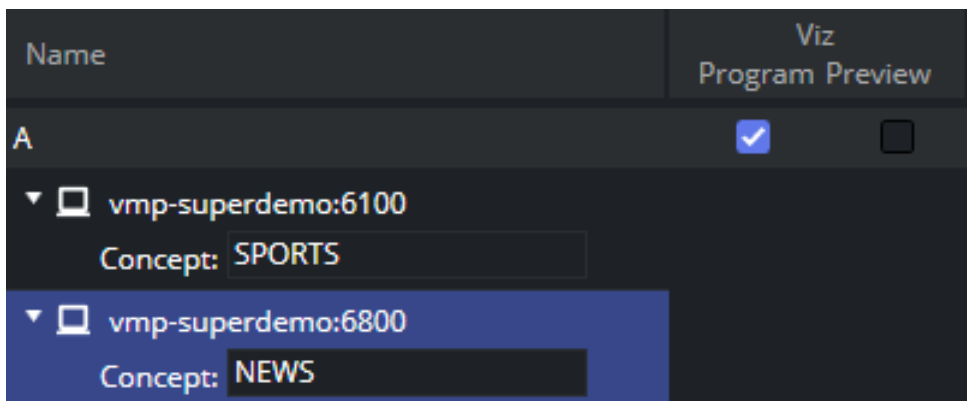
- Simply drag and drop a Viz Engine or video device onto the channel in the Channels list, or select it and select **Add to profile** (for creating a new channel) or **Add to selected channel** from the menu.

Remove an Output Device from the Channels List

- Right-click the Viz Engine or video device and select **Delete Output** from the menu or simply select it and press **Delete**.

Add a Concept Override for a Channel Output Device

It is possible to override the concept on Viz Engine output level. This concept override takes precedence over concepts set in the show or on the playout elements.

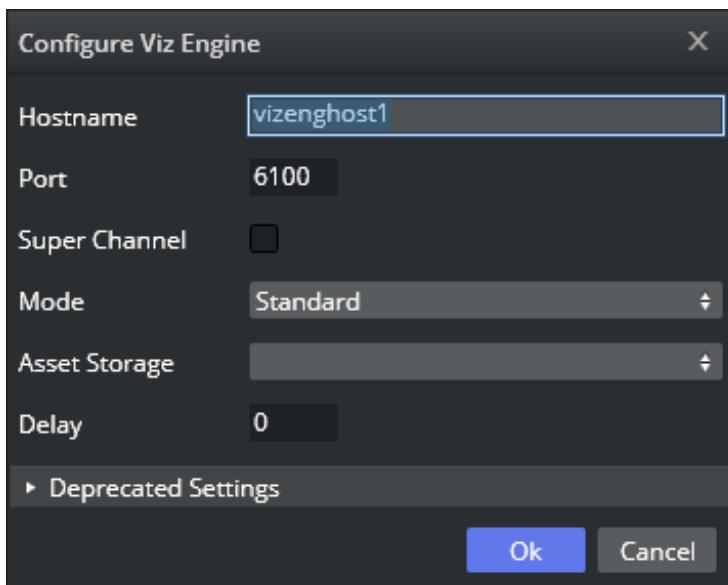


1. **Expand** the channels output device and append the concept name. This overrides any concepts set elsewhere.
2. Click **OK**.

Warning: Concept names are case sensitive.

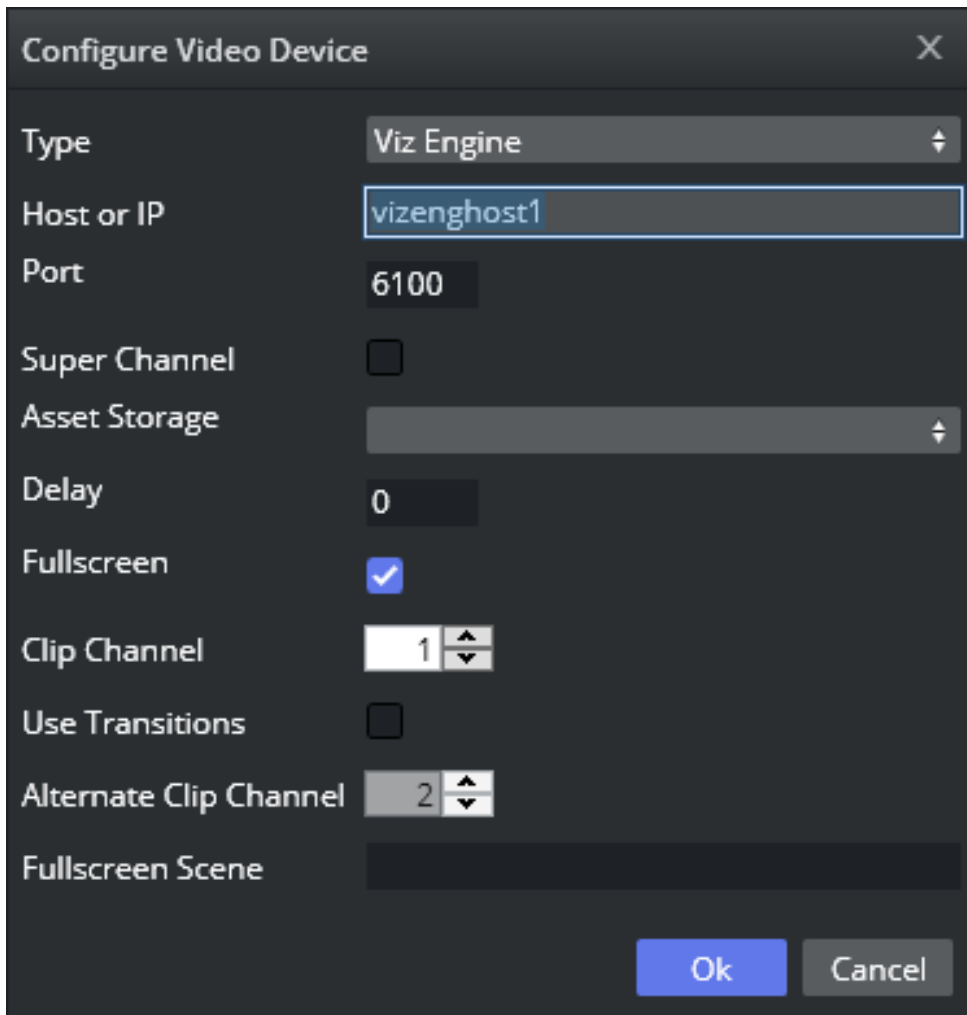
Output Device Configuration

Add a Viz Engine



1. Click **Add Viz...** in the [Profile Setups](#) window to open the **Configure Viz Engine** dialog box.
2. Enter the Hostname.
3. Default port for Viz Engine is 6100 or 56000 for Super Channels.
4. Optional: Select Mode.
5. Select whether the video device is a regular or controlled through a Viz Artist **Super Channel** plugin.
6. Set a **delay** (optional).
7. Optional: Select an Asset Storage location. Asset Storage lists available Viz Engine clip transfer and playout destinations.
8. Click **OK**. A status indicator shows if the renderer is on-air.

Add a Video Device



The image shows a 'Configure Video Device' dialog box with the following fields and controls:

- Type:** A dropdown menu set to 'Viz Engine'.
- Host or IP:** A text input field containing 'vizenghost1'.
- Port:** A text input field containing '6100'.
- Super Channel:** An unchecked checkbox.
- Asset Storage:** A dropdown menu.
- Delay:** A text input field containing '0'.
- Fullscreen:** A checked checkbox.
- Clip Channel:** A spinner control set to '1'.
- Use Transitions:** An unchecked checkbox.
- Alternate Clip Channel:** A spinner control set to '2'.
- Fullscreen Scene:** An empty text input field.
- Buttons:** 'Ok' and 'Cancel' buttons at the bottom right.

1. Click the **Add Video...** button in the **Profile Setups** window to open the **Configure Video Device** dialog box.
2. Select the video server **Type**.
3. Type the **Host or IP** address.
4. Enter the **Port** number (optional).
5. Select whether the video device is a regular or controlled through a Viz Artist **Super Channel** plugin.
6. Set a **delay** (optional).
7. Select a publishing point from the **Asset Storage** list, so that stand-alone clips and clips used in pages or data elements are transferred to the right location (the specified Viz Engine) for playout.
8. If the **Fullscreen** check box is selected, you can define additional values (**Use Transitions** and **Fullscreen Scene**) in order to provide transition effects from one video to another.

 **Note:** The fullscreen values are only available for Viz Engines, not MVCP video devices.

 **IMPORTANT!** Viz Trio by default sets the **Fullscreen Scene** in Media Sequencer to **Vizrt/System/transitions** when you switch on **Use Transitions**. Using this default, clip channel must be 1 and the alternate clip channel must be 2. Only when using other transition scenes can these settings be changed.

9. Select the **Clip Channel** to use.

 **Note:** Viz Engine from 3.6 support up to 16 clip channels which can be used for playout. A clip channel might be unavailable for various reasons: for example, if it's configured to be inactive in Viz Config or if the license only covers a limited number of clip channels.

10. **Alternate Clip Channel:** When using videoclip transitions the clip from **Clip Channel** transitions to the clip from **Alternate Clip Channel** and the next clip transitions from **Alternate Clip Channel** back to the **Clip Channel**. The transition clip channel is only enabled in the configuration window if **Use Transitions** is enabled.

11. Click **OK**. A status indicator shows if the video device is online.

 **IMPORTANT!** Media Sequencer up to and including version 2.0.0 imposes the following limitations: Clip channels 3-16 are not supported when using *fullscreen* clips. Clip channels 2-16 are not supported when using *transitions*. Using the *alternate* clip channel setting is not supported. These restrictions are lifted starting with Media Sequencer version 2.0.1 and higher.

 **Tip:** You can use the *default* profile to transfer video from Vizrt's MAM systems (unless you have manually configured a profile in both systems).

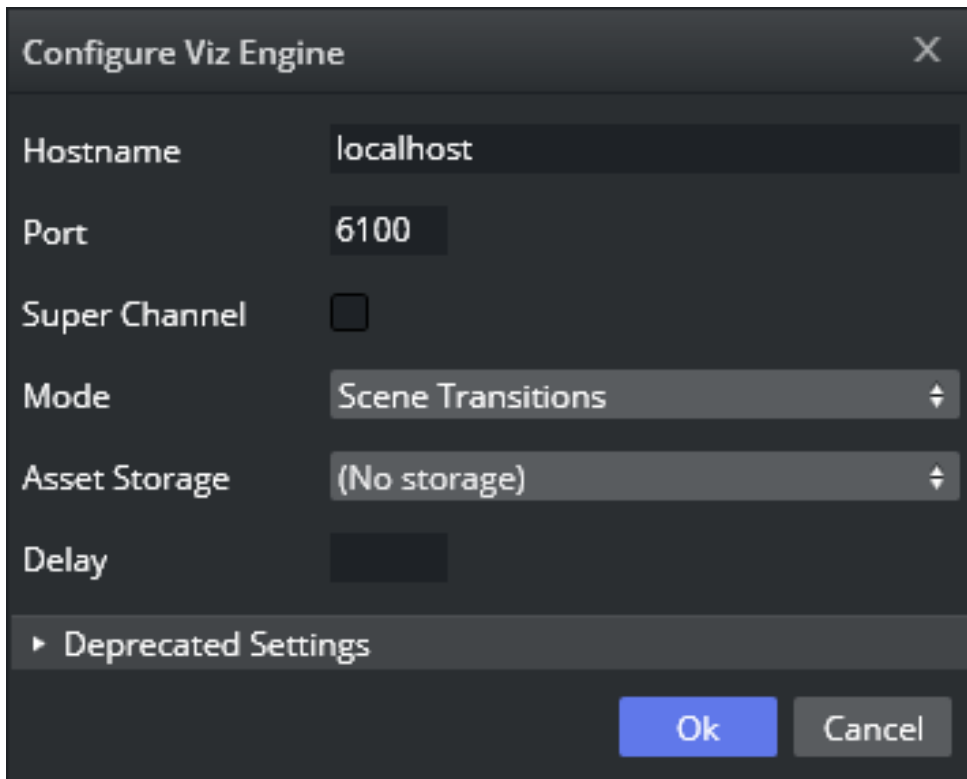
Edit a Viz Engine or Video Device

- Right-click the device and select **Edit** from the menu or simply **double-click** the device.

Delete a Viz Engine or Video Device

- Right-click the device and select **Delete** from the menu or simply select it and press the **Delete button**.

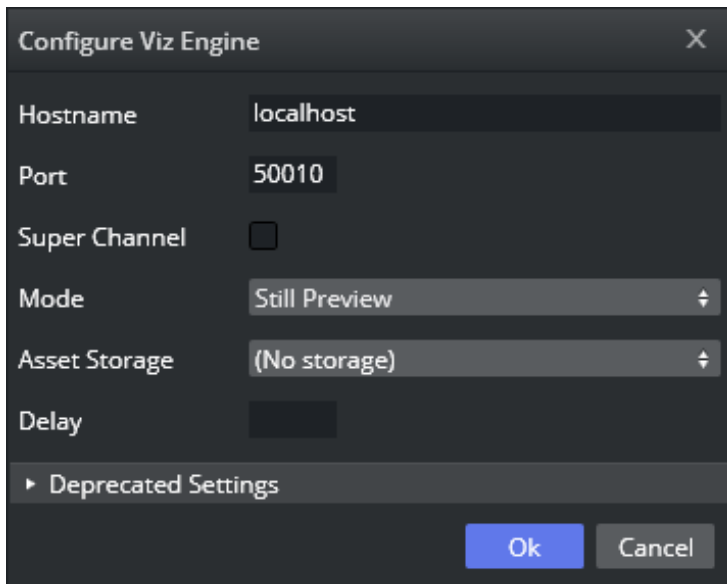
Enable Scene Transitions



1. Configure the Viz Engine settings as seen in how [add a Viz Engine](#).
2. Set the **Mode** to **Scene Transitions**.
 - **Scene Transitions:** Allows the renderer to copy (or snapshot) the scenes to create a transition effect between them.
3. Open the [Show Properties](#), and set the Transition Path. See also [Transition Effects](#).
4. Click **OK**.
5. Add the program renderer to the program channel.

 **Note:** The program channel must be configured and on-air in order for the effects to be visible.

Enable Still Preview



1. Create two channels, one for program and one for preview.
2. Configure the same render engine twice as seen in how [add a Viz Engine](#).
3. For the second render engine set the **Mode** to **Still Preview**.
 - **Still Preview:** Allows you to use the program output to get a still preview. To achieve this the Media Sequencer creates a copy of the scene being read and sends commands to your program output renderer asking for a snapshot of the scene while the current scene on air is being rendered. Note that the port used for still preview is **50010**.
4. Click **Ok**.
5. Add the program renderer to the program channel and the still preview renderer to the preview channel.

⚠ Warning: This setup requires sufficient ring buffer on the program renderer in order to prevent the rendered still preview from causing the scene on air to drop frames. This setting is therefore deprecated.

3.3.6 Connectivity

Configure protocols with which Viz Trio interfaces:

- [MOS Protocol](#)
- [Intelligent Interface \(IIF\)](#)
- [General Purpose Input \(GPI\)](#)
- [Video Disk Communication Protocol \(VDCP\)](#)
- [MCU/AVS](#)
- [Socket Object Settings](#)
- [Proxy Settings](#)
- [Viz One Configuration](#)

MOS Protocol

To view MOS configuration settings, click **Connectivity > MOS** in the Viz Trio Configuration.

MOS Newsroom Integration

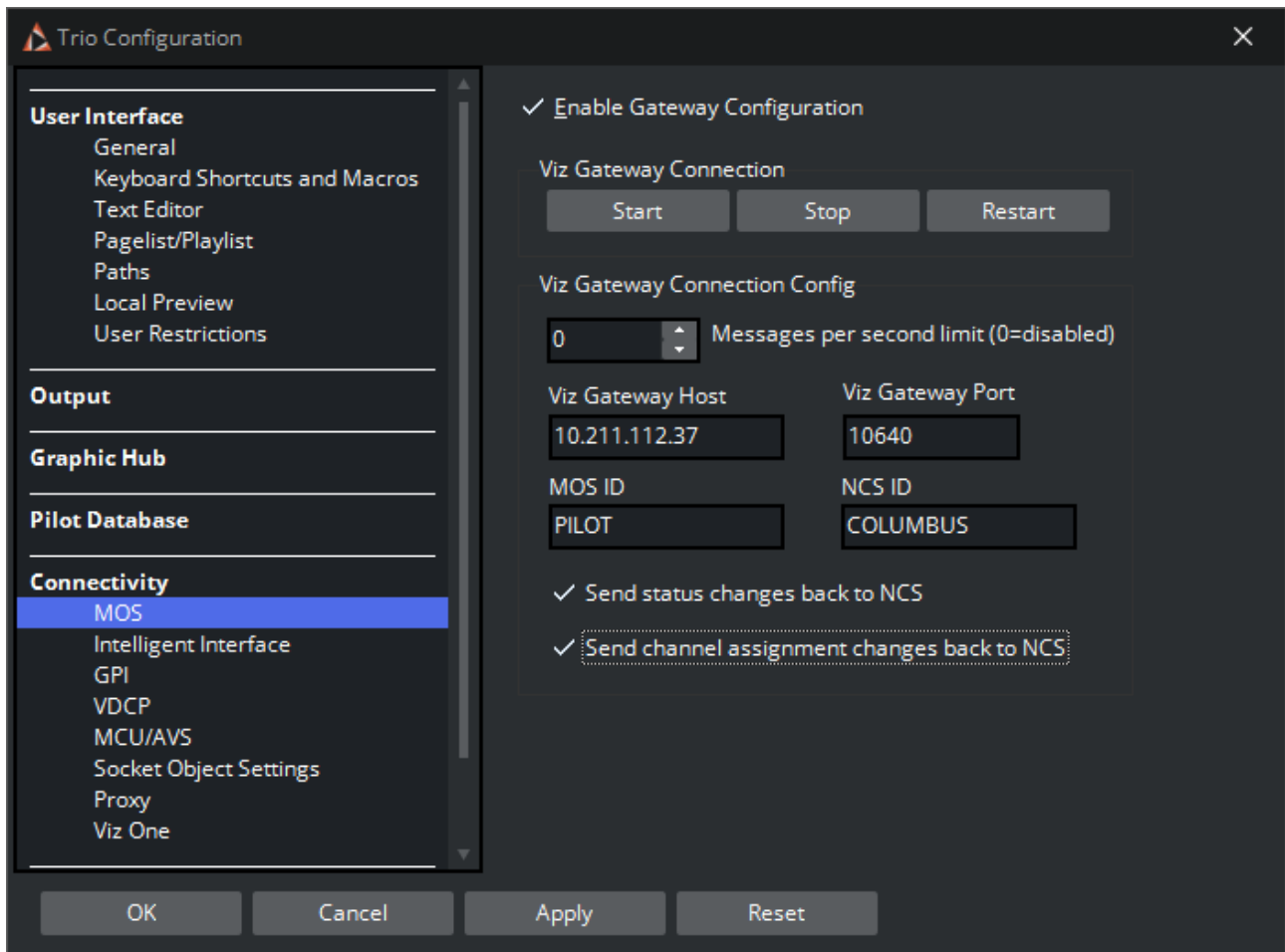
A Viz Gateway (MOS) connection allows Viz Trio to monitor rundowns from any Newsroom Computer System (NCS) that supports the Media Object Server (MOS) protocol.

Newsroom Computer Systems (NCS) that support the MOS protocol can deliver newsroom playlists to Viz Trio through the Viz Gateway and Viz Pilot's database. This means MOS integration requires a Viz Gateway, an Oracle database and Viz Pilot's Newsroom Component.

To establish a MOS connection, the NCS and Viz Gateway must be pre-configured with an NCS and MOS ID. These ID's are set in the newsroom system.

 **Note:** Most newsroom systems support the Media Object Server (MOS) protocol and/or the [Intelligent Interface](#) (IIF) protocols (Avid iNEWS ControlAir).

Viz Gateway and MOS Configuration Settings



- **Enable Gateway Configuration:** When enabled, an icon is also visible in the [Status Bar](#).
- **Viz Gateway Connection:** The controls in the Viz Gateway Connection section allows a start, stop, and restart action to be performed on a Viz Gateway connection.
 - **Start:** Starts the Viz Gateway connection.
 - **Stop:** Stops the Viz Gateway connection.
 - **Restart:** Restarts the Viz Gateway connection.
- **Viz Gateway Connection Config:** Define Viz Gateway connection settings.
 - **Messages per second limit:** Sets how many messages per second are sent from Viz Gateway to the NCS. This is done to prevent flooding of the NCS. 0 disables the message throttling.
 - **Viz Gateway Host:** IP address of the Viz Gateway host.
 - **Viz Gateway Port:** Connection port for the Viz Gateway. The default port is 10640.
 - **MOS ID:** Enter the ID of the connecting MOS device.
 - **NCS ID:** Specify the ID of the newsroom control system.
- **Send status changes back to NCS:** Sends new status changes to the NCS.
- **Send channel assignment changes back to NCS:** If enabled, instructs Media Sequencer to update channel assignments back to the Newsroom system. The next time the MOS playlist gets updated, the changed channel assignment stays at the changed element in Viz Trio and is not overwritten by the previous channel

value.

The database settings for the [Viz Pilot Database](#) on Media Sequencer, must be established in order for the Viz Gateway integration to work.

After applying the new settings, check that Viz Trio's status bar shows a green status indicator for Viz Gateway (G).

Note: MOS and NCS ID are not required for Viz Gateway versions 2 and newer.

Intelligent Interface (IIF)

To view the Intelligent Interface configuration settings, click **Connectivity > Intelligent Interface** in the Viz Trio Configuration.

Intelligent Interface Settings

Port:	COM3
Baudrate:	9600
Verbose:	no
Show path:	
Profile:	default
Encoding:	ISO-8859-1
Loglevel:	
Space means empty:	no
II Status:	uninitialized (no port)

The intelligent interface (IIF) configuration is used to set the parameters the automation system uses to connect to Media Sequencer over the intelligent interface protocol. It also defines which Viz Pilot playlist or Viz Trio show the automation system can control and on which output profile.

Media Sequencer listens to a communications port and acts as the Character Generator (CG) device side of the intelligent interface. Media Sequencer expects the other side of the communications port to be connected to a newsroom system or similar system that knows how to talk one of the supported dialects of intelligent interface.

Media Sequencer has two primary tasks:

- Receive callup data from a newsroom system and store it in the schedule.
- Trigger actions based on simulated keyboard commands it receives.

Properties and Parameters

- **Port:** Sets the serial communications port to be used.
- **Baudrate:** Sets the appropriate baud rate for the connection.
- **Verbose:** Makes the messages from the Intelligent Interface driver get displayed in the Media Sequencer console. This feature is useful when setting up the system.
- **Show path:** Sets the path of the playlist or show.

- **Profile:** Sets the appropriate profile to be used.
- **Encoding:** Sets the appropriate font encoding for the connection.
- **Loglevel:** Sets the [Media Sequencer Logging](#) for Media Sequencer.
- **Space means empty:** If set to **Yes**, the graphics template's default text is replaced with a space when creating an element. If set to **No**, the graphics element shows the default text unless it is manually changed and saved with no text.

General Purpose Input (GPI)

To view the GPI configuration settings, click **Connectivity > GPI** in the Viz Trio Configuration.

GPI action ▲	Server/Client	Server Command	Macro Command	Profile	Description
DH0	Server				
DH1	Server	next		default	goto next element
DH2	Server	previous		default	goto previous element
DH3	Server	take_current		default	take current element
DH4	Client		gui:open_shows	default	Open shows
DH5	Server				
DH6	Server				
DH7	Server				
DL0	Server				
DL1	Server				

The GPI settings allows Media Sequencer to be configured to handle GPI commands. The commands can be handled by Media Sequencer itself (Server Command) or forwarded to the Viz Trio client (Macro Command).

Note: Both server and client side commands are profile specific (see [Output](#)), meaning that the profile determines which Viz Trio client and potentially which Viz Engines should execute a client and/or server command. It is therefore very important to assign the correct profile for each GPI action as they refer to unique profiles configured per client.

For example: A Viz Trio client has its own profile with two Viz Engines (program and preview) and receives a client command from a GPI on Media Sequencer. The command executes some logic on the Viz Trio client that issues commands to the program renderer. If the correct profile is not set, such commands might end up on the wrong Viz Trio client and potentially on the wrong program renderer. The same would happen if the GPI action was defined as a server command. However, it would then trigger commands from Media Sequencer directly to the Viz Engine, and not through the Viz Trio client.

GPI Settings:

- **Show advanced:** Displays all the settings in the GPI Settings frame.
- **Box Type:** Select the type of GPI box that is being configured. The Box Type can be set to SRC-8, SRC-8 III or SeaLevel.
- **Port:** Sets the port that the GPI box is connected to. The Port can be set to COM1-COM17, or None.
- **Base Entry:** This is the node in the Media Sequencer's data structure where the systems look for the GPI actions. The base entry is by default set to /sys/gpi.
- **Baudrate:** Sets the maximum rate of bits per second (bps) that you want data to be transmitted through this port. The Baud rate can be set to 110-921600. It is recommended to use the highest rate that is supported by the computer or device that is being used.
- **Stopbits:** Sets the interval (bps) for when characters should be transmitted. Stop bits can be set to 1, 1.5, or 2.
- **Databits:** Sets the number of data bits that should be used for each transmitted and received character. The communicating computer or device must have the same setting. The number of data bits can be set to 5, 6, 7 or 8.
- **Parity:** Changes the type of error checking that is used for the selected port. The communicating computer or device must have the same setting. The parity can be set to:
 - **Even:** A parity bit may be added to make the number of 1's in the data bits even. This enables error checking.
 - **Odd:** A parity bit may be added to make the number of 1's in the data bits odd. This enables error checking.
 - **None:** No parity bit is added to the data bits sent from this port. This disables error checking.
 - **Mark:** A parity bit set to 0 is added.
 - **Space:** A parity bit set to 1 is added.
- **Flowcontrol:** Changes how the flow of data is controlled. The flow control can be set to:
 - **None:** No control of data flow.
 - **XonXoff:** Standard method of controlling the flow of data between two modems. XonXoff flow control is sometimes referred to as software handshaking.
 - **Hardware:** Standard method of controlling the flow of data between a computer and a serial device. Hardware flow control is sometimes referred to as hardware handshaking.
- **Verbose:** If enabled, the Media Sequencer's GPI handler outputs log information. This information is useful for debugging.
- **Reversed Input Order:** Note that this check box is only available if Box Type is set to SRC-8. If enabled, the signal line that originally triggered GPI action DL0/DH0 triggers GPI action DL7/DH7, the signal line that originally triggered GPI action DL1/DH1 triggers GPI action DL6/DH6, and so on.
- **GPI action:** Shows a list of the available GPI actions.
- **Server/Client:** Double click to open a drop-down list box in every row, where the selected GPI action should apply to either Media Sequencer (Server option) or the local Viz Trio client (Client option).

 **Note:** The server and client actions are reciprocally exclusive.

- **Server Command:** Double click to show a drop-down list box in every row, where the action to be performed on this GPI line can be selected.

GPI action ▲	Server/Client	Server Command	Macro Command
DH6	Server	take_and_advance ▼	
DH7	Server		
DL0	Server		
DL1	Server	take_and_advance	
DL2	Server	take_current	
DL3	Server	next	
DL4	Server	previous	
DL5	Server	continue	
DL6	Server	out_current	
DL7	Server		

Server commands are GPI actions that apply to Media Sequencer. When right-clicking an item in a [Create Playlist](#) or [Playlist Modes](#), a context menu opens. In this menu, select Set External Cursor. A red arrow appears next to the selected element in the playlist, which indicates that this is the current GPI cursor. The server commands can be set to:

- **advance_and_take:** The cursor shifts to the next element in the playlist, and then runs the Start operation.
- **take_and_advance:** Runs the Start operation on the current element, and then shifts to the next element in the playlist.
- **take_current:** Runs the Start operation on the current element (the element with the cursor).
- **next:** Shifts to the next element in the playlist.
- **previous:** Shifts to the previous element in the playlist.
- **continue:** Runs the Continue operation on the current element.
- **out_current:** Runs the Take Out operation on the current element.
- **Macro Command:** Macro commands are silent GPI actions. Clicking the ellipsis (...) button opens the Add Command window.
- **Profile:** Sets the profile to be used for the GPI action. This profile must match the profile set for the playlist that is to be triggered by the GPI actions. The drop-down list shows the profiles configured on Media Sequencer.
- **Description:** Shows the description of the GPI action, as it was specified in the Add Command window.

Assign a Server Command

1. Select a GPI action, and select **Server** in the Server/Client column.
2. Select a **Server Command**.
3. Select a **Profile**.
4. Click **Apply**.

Assign a Client Command

1. Select a GPI action, and select **Client** in the Server/Client column.
2. Select or create a **Macro Command**.
3. Select a **Profile**.
4. Click **Apply**.

Add a Macro Command

1. Select the **Macro Command** column, and click the small ellipse (...) button to open the Add Command dialog box.
2. Enter the command in the **Command** text field, or alternatively click the ellipse (...) button to open the [Predefined Functions](#) window.

Video Disk Communication Protocol (VDCP)

To view the VDCP configuration settings, click **Connectivity > VDCP** in the Viz Trio Configuration.

The Video Disk Control Protocol (VDCP) configuration allows a VDCP connection for Media Sequencer to be established in order to externally control a Viz Pilot playlist or Viz Trio show.

With VDCP, Media Sequencer acts like a server that controls the graphics through the VDCP protocol. It sets up a serial connection, and on the other end of the connection typically a video controller is placed. Over this connection VDCP commands are sent, and in this way the video controller is able to control a playlist/show.

The configuration of Media Sequencer is twofold. There are the general VDCP settings, and there is the configuration for which playlist to control.

The VDCP protocol defines recommended serial settings, but it is also possible to configure VDCP over TCP.

The configuration consists of the general VDCP settings, and the selection of which playlist to control.

- **VDCP handler:** This displays the currently selected VDCP handler. You can **change**, **add** or **delete** multiple VDCP handlers.
- **Enable/Disable handler:** The handlers can only be modified when they are disabled.

- **Connection:** Selects whether the VDCP handler communicates over serial port or TCP.
- **Port:** Select the appropriate COM port or TCP port for the selected VDCP handler.
- **Profile:** Select the profile to use. The profile defines the Viz Engines and video devices where the command is to be executed. In the VDCP handler, the configured playlist must be active on the configured profile. (See also *Profile Configuration* in the *Viz Pilot User Guide*).
- **Element identifier:** Use either the element's **name** or **description** as the VDCP ID. The Name is a unique identifier, assigned to each element by the Media Sequencer, whereas the Description is the textual description of the data element, as entered when the item was created in Director/Viz Trio.
- **Use 8 byte IDs:** Allow the use of 8 character IDs. For more information about 8 byte IDs and the Media Sequencer VDCP plugin see the "8 characters identifiers" section in the *Media Sequencer User Guide*.
- **Targets:** Select the Director playlists or Viz Trio shows which targets are Primary and Fallback. When the vdcpl_client handler receives a command over the VDCP protocol, it iterates over the **primary target** collection, looking for the element with the specified ID. If a matching item is found, the command is executed on it. If no match is found in the primary target collection, or if there is no primary target collection specified, then it searches the **fallback target** collection. If there are still no matches, then an error is returned over the VDCP protocol, and it is also reported in the Media Sequencer logs. Click the buttons next to the primary and fallback target slots to select a playlist or show. Double click a slot to clear it.
A 'Path not found' warning, or a missing playlist/show icon, indicates that the selected playlist or show is invalid (eg. playlist has been deleted)
- **Target Candidates:** Suggests some of the playlists and shows that are available. If you do not find the one you want, use the buttons next to the Primary and Fallback target slots.
Drag a candidate to a target slot or double click one to make it the Primary target.

MCU/AVS

To view the MCU/AVS configuration settings, click **Connectivity > MCU/AVS** in the Viz Trio Configuration.

MCU/AVS Configuration Settings

MCU/AVS Settings

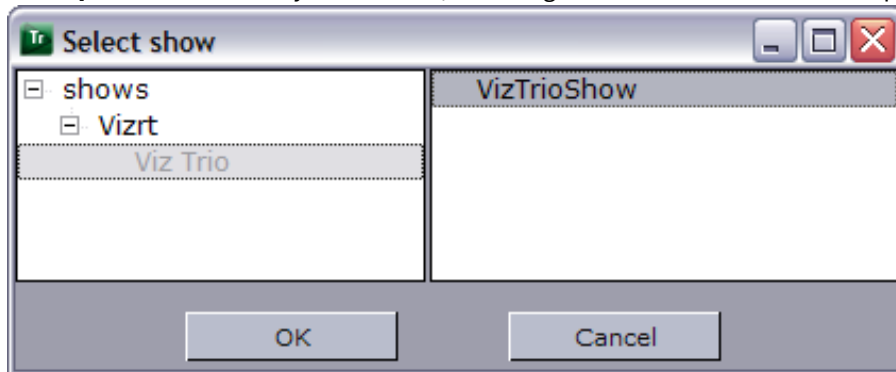
Port:	COM2	▼
Baudrate:	110	▼
Show path:	trioshow	...
Range begin:	1000	
Range end:	2000	
Offset:	100	
Verbose:	yes	▼
Replace delimiter:		
Super-delimiter:	/	

☐ Ignore incorrect events

☐ Unconditional deletes

This configuration is primarily used to configure settings for a Newstar (News*) newsroom system connection.

- **Port:** Select the appropriate Com port for the communication.
- **Baudrate:** Select the appropriate baud rate.
- **Show path:** Base directory for the MCU/AVS integration. Newsroom stories are placed here.



- **Range begin and Range end:** Pages received from the newsroom system are numbered within this range.
 - Example: If Range Begin is 1000 and Range End is 5000 and Offset is 100. Then Callup pages in story 1 are numbered 1000, 1001, 1002, and so on. Callup pages in story 2 are numbered 1100, 1101, 1102, and so on.
- **Offset:** Sets the offset value that is used to separate pages from different stories.
- **Verbose:** Enables notification messages from the MCU/AVS driver. The messages are shown on the Media Sequencer console. This feature is useful when setting up the system.
- **Replace delimiter:** Replaces a character with the super-delimiter. For example if set to '|'. slashes '/' can be put into tab-fields by entering '|'.

- **Super-delimiter:** Delimiter used in News* to separate tab-fields. Standard value is '/'.
- **Ignore incorrect events:** Incomplete scripts from News* send invalid protocol data. Typically template specifications and supers could be missing. If this option is set to "Yes", then the system accepts incomplete/incorrect story descriptions. Should be set to "Yes" in most cases.
- **Unconditional deletes:** Removed in version 1.10. On older versions of Viz Trio, this option should be enabled. This means that a transfer of messages from Newstar starts with deleting existing messages within the specified range. To maintain a proper state handling this method must be used.

Socket Object Settings

To view the Socket Object configuration settings, click **Connectivity > Socket Object Settings** in the Viz Trio Configuration.

Socket Object Configuration Settings

The Socket Object Settings section allows for socket connections to be defined for the Viz Trio client. It can either act as client or a server in a socket connection. A serial port connection which uses the same command set can also be set up.

- **Socket type:**
 - **Client socket:** When Viz Trio is set up to be a client socket it connects to the specified server socket. Specify a host name and a port for the machine that runs the server socket.
 - **Server socket:** When Viz Trio is set up to be a server socket it waits for a client socket connection. Specify the port it should listen on.
 - **Serial port (Com port):** The serial port connections work in principle as a socket connection, but instead of using TCP/IP it uses a null modem cable. The command-set is the same.
- **Text encoding:** Select a text encoding for the connection.
- **Serial port config:** If a serial port connection is to be set up the COM port number and the data baud rate must be specified here.

- **Autoconnect:** With this option set, a server socket or a serial port connection is open/active at all times, but a client socket is active when data is sent. If this option is not checked, a "connect_socket" command must be sent before sending the data.
- **Socket Host:** The socket server's host name must be set here if the Viz Trio client is to function as a client socket.
- **Socket Port:** For client and server socket a port number must be specified here.
- **Data separator (escaped)**- Specifies a data separator for incoming data.
 - Receiving data separator triggers the `OnSocketDataReceived` event in Scripts. If unprintable ASCII characters are used as data separators, special character sequences called "escape codes" are needed. All escape codes start with a backslash. For details, see [Escape Sequences](#).
 - When the socket object receives the data separator, the `OnSocketDataReceived` event is triggered, and the data received since the last separator is issued (not including the separator itself). If Data separator has no value, the `OnSocketDataReceived` is called continuously as long as data is received.

Proxy Settings

To view the Proxy configuration settings, click **Connectivity > Proxy** in the Viz Trio Configuration.

Proxy Configuration Settings

✓ Enabled This will cause all data that is sent over the port to be treated as Viz Artist commands

Listen on port: 6500

Viz location (port host): 6100

The proxy function is mostly relevant when Viz Trio and Viz Engine are set up with a direct network link to be controlled by an external control application. A dedicated network card on the Viz Trio client together with a crossed network cable directly to Viz is the normal way to make such a link.

All data sent over the configured listener port is treated as Viz commands and forwarded by the Viz Trio client to Viz. With the proxy function enabled, the data sent over the configured listener port is treated as Viz commands by the Viz Trio client. Viz Trio does this by acting as a proxy transmitting commands back and forth between Viz and the external control application.

- **Enabled:** Enables the proxy feature.
- **Listen on port:** Sets the listener port for Viz Trio that the sender uses.
- **Viz location (port host):** Sets the listener port for Viz Engine and the hostname (default is 6100 and localhost).

Viz One Configuration

To view the Viz One connectivity settings, click **Connectivity > Media Engine** in the Viz Trio Configuration.

Viz One Connectivity Settings

Viz One

✓ Enable the Viz One handler

Service document URI:

http://vizionehost:80/thirdparty/

When the service document URI of the asset management system is changed, remember to manually change the Asset Storages for each video handler in the Output section.

Search Viz One using the following credentials

The credentials are just stored for this client PC.

Username:

admin

Password:

•••••

Preview Server

Host:

previewhost1

Port:

54000

Media Sequencer communicates with Viz One when show and playlist elements that contain Viz One elements residing on Viz One are initialized for playout.

- **Enable the Viz One handler:** If selected, enables the Viz One handler.
- **Service document URI:** Defines the Viz One instance to use.



Note: When the service document URI of the Viz One is changed, remember to manually change the Viz One storages for each video handler in the [Output](#) section of the configuration. A Viz One storage points to a Viz Engine where the Viz One files are sent for playout. For more information, see [To add a video device](#).

It is possible to define the Viz One server instance based on either an IP address or hostname. It is recommended to use hostname. Host comparisons in the system are generally done by string comparison, not by lookup. This is why *either* IP or hostname must be selected. This also means that you should not mix hostnames and fully qualified domain names; *http://vizione01.vizrt.com/thirdparty_* is not the same as *_http://vizione01/thirdparty_*, even if it is possible to ping both. The system allows using both *_http* and *https*, although *http* is recommended. Whatever

options being selected (IP vs. hostname, hostname vs. domain name, http vs. https), choose one or the other, and stick with it throughout the entire setup process.

Note: Viz One version 5.3 and later uses `/api` and not `/thirdparty`.

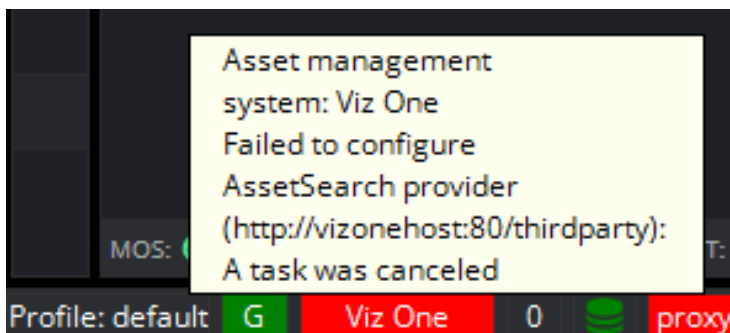
- **Username:** Searches Viz One using the following Viz One username.
- **Password:** Searches Viz One using the following Viz One password.
- **Host:** Specify the host of [Preview Server](#). This service is used to generate preview graphics in the timeline editor. If the *ZeroConf* service is installed and running, you can also select servers on the local network.
- **Port:** Specify the port of [Preview Server](#), defaults to 54000 if left empty.

Configure a Viz One Connection

1. Make sure that the **Enable the Viz One handler** check box is selected.
 2. Type the hostname of the relevant Viz One in the **Uri to service document** box.
 3. Enter working login credentials in the **Username** and **Password** boxes.
- If the Viz One configuration is successful, a green Viz One icon appears in the [Status Bar](#).

Viz One Error Message

If the configuration is not working properly, the Viz One icon is red. Scenarios where the icon is red can for example, be a Viz One-search that fails, or a more serious error, a non-responsive Viz One with no video playout. In such serious situations where Viz One is unavailable, an error message is displayed, stating that videos that have already been transferred to Viz One still work, but new videos are not transferred.



Configure Viz One Server Time-out Handling

In the event of a network failure or other unforeseen events the Viz Trio client can be instructed by scripting to switch to a backup Viz One server. In order to prepare for this you should:

1. Use the macro commands `settings:set_mam_service_document_uri [uri [username password]]` and `settings:set_media_search_credentials username password` to instruct the switch to an alternate Viz One server.
2. Update the active configuration (also updates the Media Sequencer VDOM) with the new publishing point using the command: `channelcontrol:set_asset_storage <host-name> <storage-name>`.
3. Make sure that the timeout value for the Media Sequencer VDOM value `mam/get_publishing_points_timeout` is appropriate for your network environment. By default it is 10000 (ms), i.e. 10 seconds. If the value is too low, the `settings:set_mam_service_document_uri` command could possibly return with error without making the required change due to perhaps network glitches. Note that you have to restart your Viz Trio client for a

change to this parameter to take effect.

In order to get or set the timeout value you can:

- Get the current value with: *settings:get_setting mam/get_publishing_points_timeout*.
- Set the timeout with: *settings:set_setting mam/get_publishing_points_timeout [VALUE IN MS]*. To allow for (max) 15 seconds : *settings:set_setting mam/get_publishing_points_timeout 15000*.

A basic switchover-script example:

```
settings:set_mam_service_document_uri http://MyNewVizOneServer/api
settings:set_media_search_credentials MyUserName MyPassword
channelcontrol:set_asset_storage MyNewVizOneServer The Video server
```

3.4 Command Line Parameters

Use the command line parameters below to customize startup.

- [Adding a Command Line Parameter](#)
- [Viz Engine Startup Command Line Parameters](#)

3.4.1 Adding a Command Line Parameter

Right-click the program shortcut, and edit the program target path.

Example: "C:\Program Files\Vizrt\Viz Trio\trio.exe" -mse localhost -control -lang en -loglevel 100 -vizparams "-u1 -M -n" -t "Viz Trio %v - MSE: %h - Show: %s - GH: %gh_host || %gh_db || %gh_user || %gh_url"

Parameter	Value Syntax	Description
-attach_to_viz		Attaches to a running viz.exe without asking the user to terminate it. If a Viz Engine process is already running locally, and this process started with the parameters required for Viz Trio (-u1 -M), this command line option omits the notification "viz.exe is already running. Terminate?". The viz.exe process continues and attaches without asking the user to confirm. If there is no Viz Engine process running, Viz Trio starts up its own process as before, regardless of whether the command is present. This command is a flag only and has no parameter value.
-control		Lets you configure Viz Trio.
-folder	show path	Causes the client to start in the specified show.
-force_gpu_count	integer	DEPRECATED.
-local_program_channel_port	integer	Overrides the configured port number set for the local program channel. This is useful if a different port than the default 6800 port needs to be set.
-logfile-path	directory	Folder where log files are stored.
-loglevel	integer	Controls how much is logged to the log file.

Parameter	Value Syntax	Description
-macro-port	integer	Overrides the port of the macro command server. The default port is 6200.
-mse	hostname	Media Sequencer hostname.
-nle-mode		Start Viz Trio in NLE mode.
-no-nle-mode		Start an NLE-compiled Viz Trio in normal (non-NLE) mode.
-onebox		Launching Viz Trio with the <i>-onebox</i> parameter, launches a preview instance of Viz Engine and a program instance of Viz Engine on the output port 6800. If the computer has two GPUs, it creates a profile in Media Sequencer with a program channel. Note that Viz Trio does not stop the program engine when Viz Trio shuts down.
-scriptloglevel	integer	Controls how much is logged when executing scripts.
-socket		Use socket (TreeTalk) for Media Sequencer communication.
-t	quoted string	Redefines the main window title of Viz Trio. Example: <pre>-t "Viz Trio version %v with current show being %s. The Graphic Hub host is %gh_host and the database is %gh_db"</pre> In the quoted -t parameter, the following strings can be used: • %v - version • %h - Media Sequencer host • %s - current show • %gh_host - host name of the Graphic Hub (the nameserver) • %gh_db - the name of the Graphic hub database name (usually VizDb) • %gh_user - the current GH user • %gh_url - the GH REST URL Default, if -t not specified: "Viz Trio %v - MSE: %h - Show: %s"
-usersetting	setting name	Uses something other than the hostname as an identifier for user settings.

Parameter	Value Syntax	Description
-viz-console-delay	<time in seconds>	Starts Viz Trio independently of the Viz Engine process, enabling use of Viz Trio before Viz Engine is running. Note that at startup the Viz Engine console is displayed in this mode, which can be used to debug Viz Engine start-up problems.
-vizdb	host:db:user:pw	Configures the Viz Engine 3 database login for the local preview engine.
-vizparams	quoted string	Extra parameters to pass on when starting the local viz engine.
-vizpreview	hostname:port:protocol	Connects to an external Viz Engine for preview.
-viz-startup-timeout	integer	Specifies the time-out to use during start-up of Viz Trio when communicating with Viz Engine. The default time-out value is set to 30 seconds. Time-outs during start-up often leaves Viz Trio unusable. After Viz Trio has successfully started up, the time-out setting in the Local Preview is used.

3.4.2 Viz Engine Startup Command Line Parameters

When Viz Engine instances are run manually, Viz Trio is compatible with the engine instances only if :

- Preview instance of Viz Engine is run using the shortcut parameter : "-u1 -M".
- Program instance of Viz Engine is run using the shortcut parameter : "-u2 -n".

Parameter	Description
-u1	One GPU (the default one) is used.
-y	Viz Engine runs with GUI.
-n	Viz Engine runs in OnAir mode, it could be used with or without -y, but not with -M.
-M	Viz Engine runs in Viz Trio Artist Mode. This overrides the -n parameter.

Parameter	Description
-e	OBSOLETE, used in previous versions of Viz Trio, but now ignored by the Viz Engine.

4 Datacenter

The Datacenter component allows you to connect data live sources to fields in the graphics without any scripting or special knowledge, and connect incoming live data fields in scenes. It is ideal to connect to, for instance, a scoreboard on a venue to clock-and-score graphics.



This section covers the following topics:

- [Overview](#)
- [Requirements](#)
- [Getting Started](#)
- [Configuring Viz Engine](#)
- [Prepare Scenes for Data Input in Viz Artist](#)
- [Configuring Datacenter](#)
 - [Connect Data to Fields in Scenes](#)

4.1 Overview

From Datacenter version 2.0, the component is not bundled with Viz Trio anymore, but provided as a standalone installer. Please see the documentation for [Datacenter](#).

Datacenter can run on any host on the network, and as a service on the Viz Trio machine. The user interface can be opened in a web browser in Viz Trio by going to **Tools > Datacenter**. The user interface can also be opened explicitly in a web browser by entering <http://localhost:5700/>.

4.2 Requirements

Datacenter live input requires **Viz Engine 4.4.1** or newer.

Control Plugins supported (these plugins can be fed with live data from Datacenter):

- ControlText
 - ControlParameter
 - ControllImage
-

4.3 Getting Started

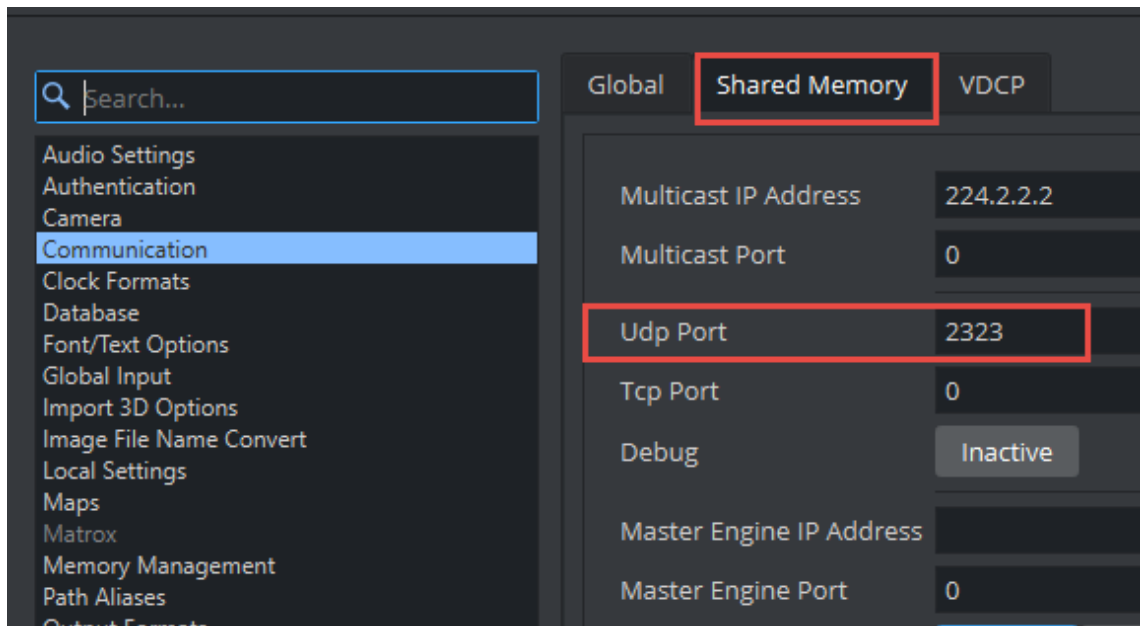
Getting started with live updated graphics in 9 easy steps:

1. In order for Viz Engine to receive live data, open configuration and enter a port for incoming UDP Shared Memory input under **Communication > Shared Memory > Udp port**.
 2. In Viz Artist, open the scenes that contains one or more fields that should be controlled with live data from the Datacenter.
 3. For each field, locate the Control plugin and check **Use shared memory placeholders**.
 4. **Save** the scene.
 5. In Viz Trio, **import the scenes** as templates into a show.
 6. Go to **Tools > Datacenter**.
 7. In Datacenter, configure one or more Viz Engines as **receivers** of the live data.
 8. Configure one or more data sets as **input**.
 9. Each data set contains a list of key/value-pairs. In Viz Trio, in the Field Value for the fields being fed with live data, use the notation `${key|default value}`. Now the "key" is replaced automatically by the value of the key in Datacenter.
-

4.4 Configuring Viz Engine

Each Viz Engine receives live data from Datacenter, and it must be configured to listen to Shared Memory input over UDP. This should also be done for the local preview Viz Engine on the Viz Trio machine.

1. Open The Viz Engine configuration.
2. Go to Communication > Shared Memory > Udp Port.
3. Enter any port number that is available on your network.



4. Save the configuration and quit. Your Viz Engine is now ready for Datacenter live input.

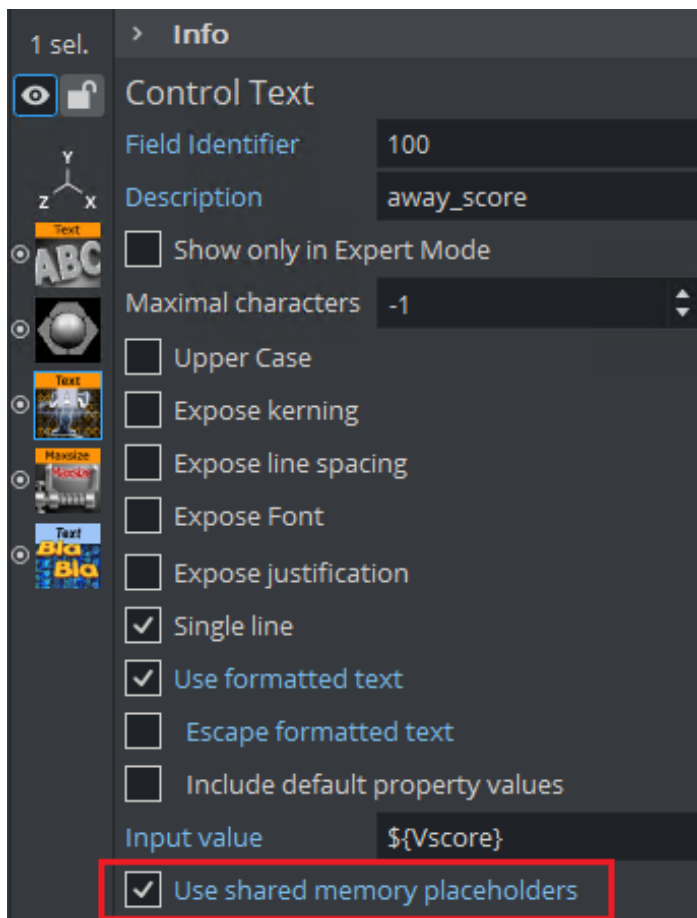
4.5 Prepare Scenes for Data Input in Viz Artist

The scenes that should receive live data from Datacenter, must enable **Use shared memory placeholders** for the control plugins that should be updated with data. The following control plugins are supported:

- ControlText
- ControlParameter
- ControllImage

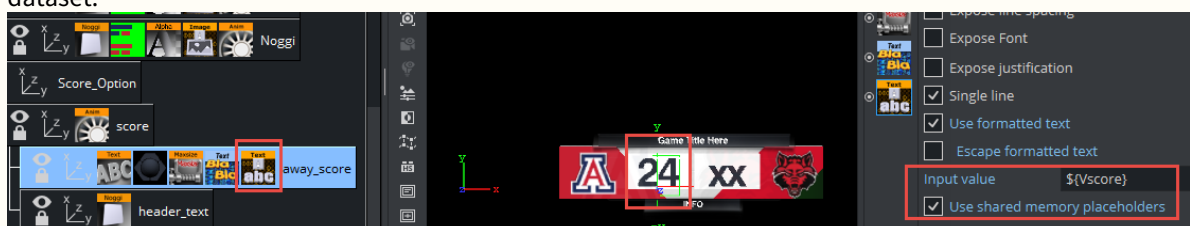
To enable live data input from Datacenter:

1. Open the scene in **Viz Artist**.
2. Locate the control plugin for the field that should receive live input.
3. Check **Use shared memory placeholders**. Note that this enables listening for shared memory in the System (Global) memory map.



4. Save the scene.

Note: If you already have a Datacenter running and have opened the SHM UDP listener port to the local preview engine, you may test the input value from Datacenter by entering the shared memory placeholder syntax in the Input value field, for instance `${Vscore}`. This requires a key called `Vscore` in a Datacenter dataset.



4.6 Configuring Datacenter

Please check the [Datacenter](#) documentation.

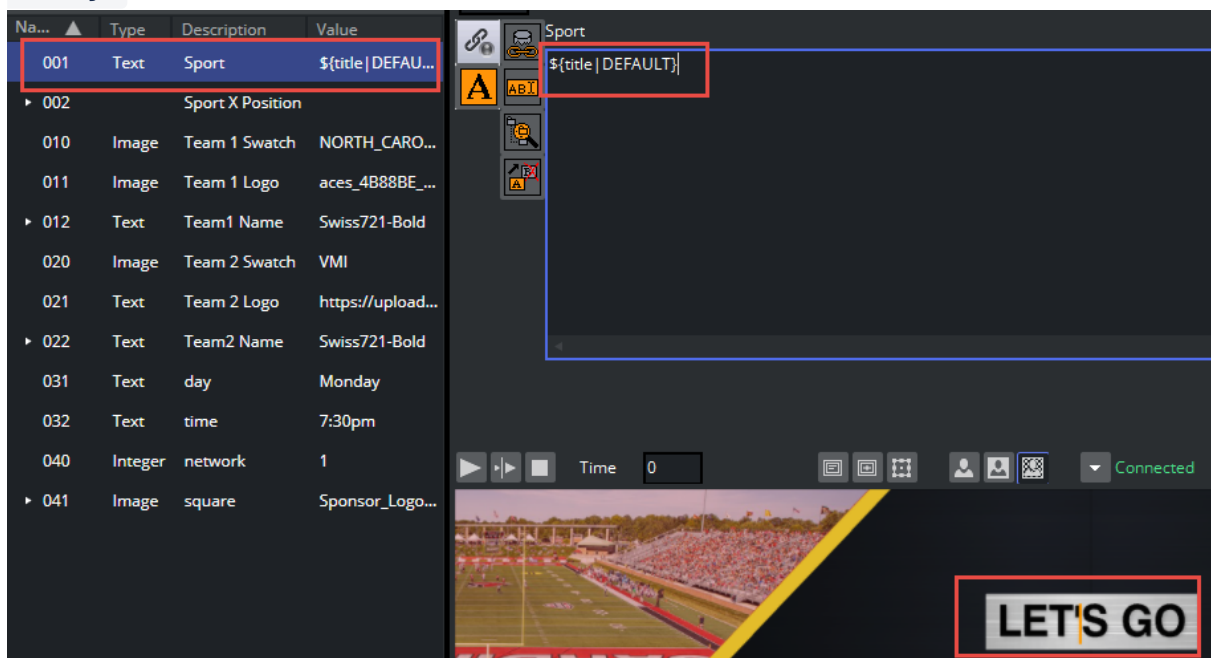
4.6.1 Connect Data to Fields in Scenes

After specifying datasets as input to the global key-value map, and specifying Viz Engines as receivers, you should specify where exactly the values appear in the scenes.

Each field has to be enabled for **Use shared memory placeholders** in the control object. See *Prepare Scenes for Data Input in Viz Artist*. If the scenes were already imported into Viz Trio, the templates need to be reimported after the scene has changed.

To connect one or more keys from Datacenter to a field:

1. Open the template in Viz Trio.
2. Locate the relevant field in the Tab Fields list.
3. Go to **Tools > Viz Artist** and enable shared memory placeholders for the control field, and reimport the template in Viz Trio.
4. In the input value, enter the following notation to use a value of a key in Datacenter: `${key | default value}`.



The key must be defined in one of the datasets in the Datacenter.

5. After the pipe, a default value can be entered (in case the key is missing or the data source is broken). It is also possible to build up a field value from multiple keys and text, for example: Current time is: `${clock | 00:00:00}` and we are in `${qtrtext | 1st}` quarter.

ControllImage

When enabling Shared memory placeholders for a ControllImage plugin in Viz Artist, note that this field is converted to a ControlText when imported or reimported into Viz Trio.

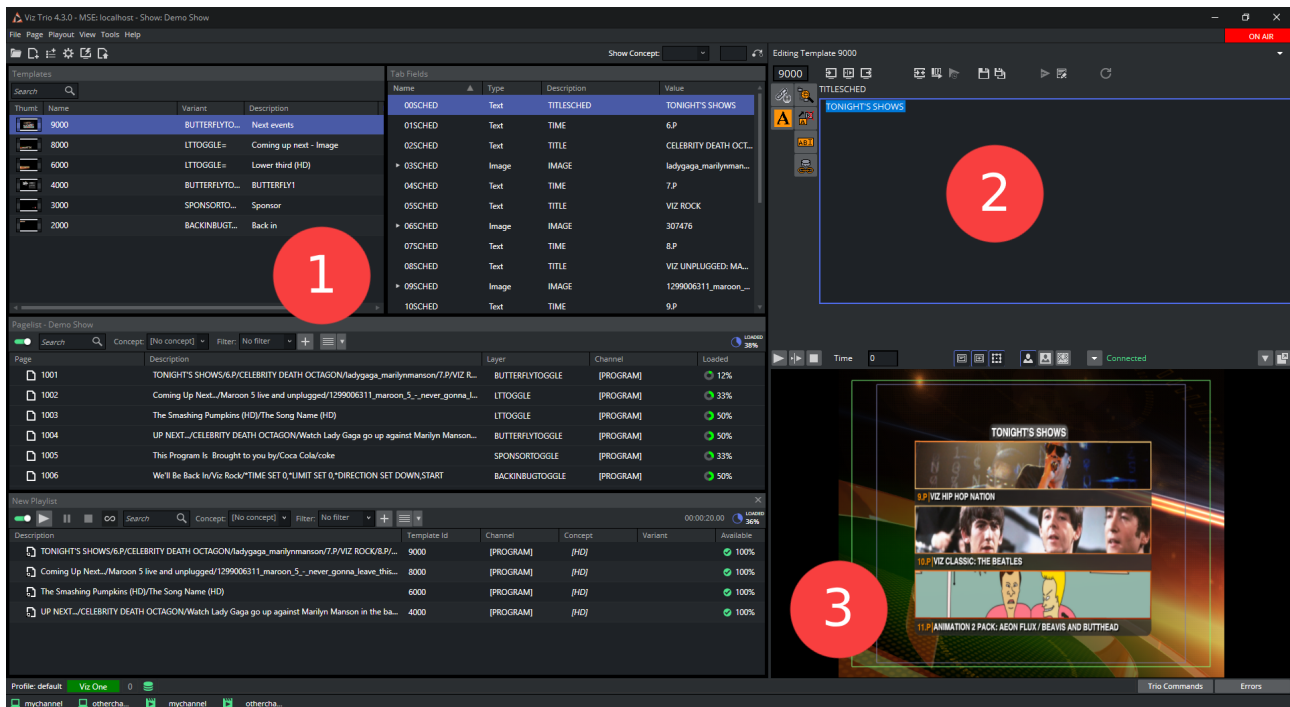
The control field can then only be controlled by entering the image path, either a Viz path or an URL. For example, the field value can contain a value from Datacenter being the full URL of an image, or the Viz scene path can be constructed with a path to the folder and a value representing the image name: `IMAGE*/0_Tool_Kit/GenericLogos/A/${teamname}`. The full path, is the path plus the value of the Datacenter key "teamname".

Tab Fields			
Name	Type	Description ▲	Value
04-assists	Text	assists	8.9
07-field4	Text	field	49
08-finals	Text	Headline	3
09-headshot	Image	headshot	basket3
01-league	Text	League	Basketball
10-logo	Text	logo	IMAGE*/0_Tool_Kit/GenericLogos/A/\${teamname}

5 Playout Interface

This section describes the playout interface.

5.1 Interface Overview



Viz Trio has three main areas (**1,2,3** in the figure above) and a top main menu and status lines with options at the bottom of the main window.

- At the top, the main menu is grouped into sections **File**, **Page**, **Playout**, **View**, **Tools** and **Help**.
- **[1]** On the left section of the screen you select and work with Templates, Pages, Pageview and edit tab values. Tab values are data entry points in templates and pages you can edit before the finished pages are played out.
- **[2]** The top right area contains editors and inspectors to view, select and change various properties for the objects you are working with.
- **[3]** The **preview window** is in the bottom right area. Viz Trio 3 and newer versions allow you to detach the preview window so that you can have it on a second monitor, for example.



Note: Viz Template Wizard pages can also be detached from the main Viz Trio window. From Viz Trio 4, the windows on the left side are dockable and detachable.

- At the bottom, status bars display status information.

5.2 Main Menu

The main menu at the top of the main window provides easy access to functions and workflows:

File Page Payout View Tools Help

A brief description of the main menu sections is listed below. For certain menu items, a more detailed explanation can be found later in this chapter.

5.2.1 File

Option	Function
Open Show...	Show a directory panel to open a TrioShow.
Open External Playlist...	Show a directory panel to open an external playlist.
New Page View...	Add a new pageview.
New Show Playlist...	Add a new playlist.
Configuration	Show the Configuration window where you configure Viz Trio.
Show Properties	View and configure properties for a TrioShow.
Import Viz Engine Archive...	Import Viz Engine Archive (.via) files from disk and re-import any affected templates in the current show.
Import Show Archive...	Import a TrioShow(.trioshow file) from disk.
Export Show Archive...	Export Show to disk as .trioshow file.
Export Selected Pages Archive...	Export Show with only selected pages including associated items, see Export Selected Pages Archive .
Slave Mode	Set Viz Trio in Slave Mode, see Slave Mode .
Quit	Quit Viz Trio.

5.2.2 Page

Option	Function
Save	Save the current page you're working on.
Save As...	Save the current page you're working on with a new name.
Rename...	Rename the current page you're working on.
Change Template...	Change the template for a page.
Refresh Linked Data	Asynchronously refresh the data for a page linked to a feed.
Reimport	Reimport the template design for a page.
Delete	Delete the currently selected page. You are asked for confirmation.
Delete All	Delete all pages. You'll be asked to confirm.
Save to XML...	Save page to an XML-file on disk.
Load from XML...	Load page from an XML-file on disk.
Print/Save Snapshot...	Take a Snapshot, which can be printed or saved to disk.
Move to Number...	Move page to a specific number.
Move with Offset...	Move page with offset.
Copy to Number...	Copy page to a specific number.
Copy with Offset...	Copy page with offset.

5.2.3 Playout

Option	Function
Take	Performs a direct take on the selected page.
Pause	Pauses the current element.

Option	Function
Continue	Continues the animation on the selected page, or any page that is loaded in the same transition logic layer as the selected page.
Out	Takes out the selected page, or any page that is loaded in the same transition logic layer as the selected page. Hard cut.
Cue	Prepares the clip for playout so the first frame is ready in the player.
Cue Append	Prepares the clip for playout. The clip starts automatically when the current clip ends.
Take and Read Next	Takes the current element and reads the next one.
Cleanup Channels	Cleans up the renderers (Viz Engines) currently in use by the open show. Asks for confirmation. This frees up renderer memory and real-time capacity. See Cleanup Channels .
Initialize Show on Channels	Loads the selected show on the program and preview renderers. See Initialize Channels .
Initialize Show on Local Preview	Loads the selected show on the Preview renderers. See Initialize Channels .
Update on Program	Updates on Program renderer without reloading or rerunning animations. Only for Transition Logic scenes.
Update on External Preview	Updates on External Preview renderer without reloading or rerunning animations. Only for Transition Logic scenes.
Clear Program	Takes out the scene on Program renderer, but does not clean up the renderer (scenes remain in memory).
Clear external Preview	Takes out the scene on Preview renderer, but does not clean up the renderer (scenes remain in memory).
Start Scroll	See Scroll Control .
Stop Scroll	See Scroll Control .
Continue Scroll	See Scroll Control .

Option	Function
Arm Current	See Arm and Fire .
Arm Selected	See Arm and Fire .
Unarm Current	See Arm and Fire .
Unarm All	See Arm and Fire .
Fire All	See Arm and Fire .

5.2.4 View

Option	Function
Expert Mode	Toggle. Enable Viz Trio to operate in expert mode. Works in conjunction with “Show Only in Expert Mode” from the Control Plugin options.
Tabfields	Toggle. Enable tabfields.
Commands	Show the command window. In this window you can execute Viz Trio macros and commands using a command-line (cli) interface.
Errors	Show the error message window.
Running Carousels	Show the task window where all running tasks in the active profile are displayed.
Refresh Thumbnails	Force refresh of all thumbnails.
Restore Default Layout	Stacks and docks the main windows (Templates, Page List and Tab Fields) and cascades all other window in undocked mode.
Lock Layout	Prevents accidental undocking of window. It is still possible to resize and dock floating windows.

5.2.5 Tools

Option	Function
Viz Artist	Switch to Viz Artist. Viz Trio keeps running. While in Artist, click the Viz Trio button to return to Viz Trio again. See Viz Artist Mode .
TimeCode Monitor	Start the TimeCode Monitor. See TimeCode Monitor .
Datacenter	Start the Datacenter application if installed on localhost. See Datacenter in Viz Trio.

5.2.6 Help

Option	Function
View Help	Show help and documentation.
Legal Notices	Information about third-party code libraries.
Trio Licensing	Opens the Viz Trio license dialog box.
About...	Displays which Viz Trio version and build that has been installed on your machine (essential information to attach when reporting issues to Vizrt), Credits and Legal Notices.

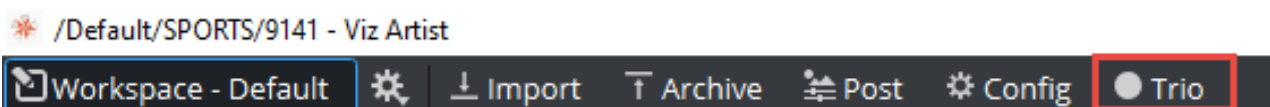
5.3 Modes

This section covers the following topics:

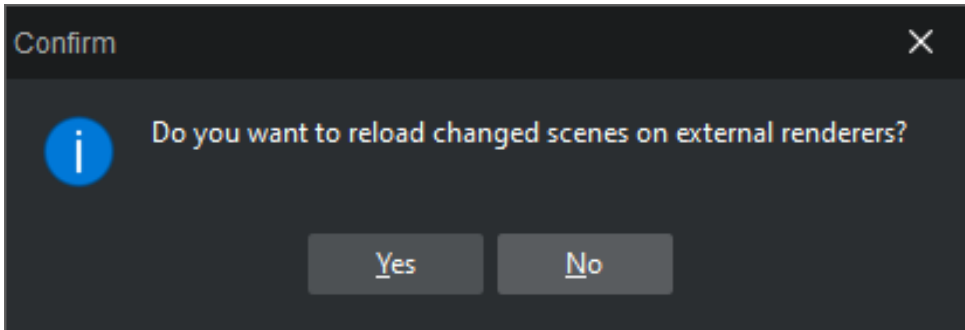
- [Viz Artist Mode](#)
- [On Air Mode](#)
- [Slave Mode](#)
- [Show Modes](#)
 - [Context Enabled Shows](#)
 - [Concept Naming Conventions](#)
 - [User-defined Contexts](#)

5.3.1 Viz Artist Mode

Viz Trio templates are based on scenes created in Viz Artist. From the main menu, click **Tools > Viz Artist** to start Viz Artist, opening the template's scene that is currently loaded in Viz Trio's preview window. You can edit the scene, and save the changes in Viz Artist.



When in Viz Artist mode, Artist displays a button labelled **Trio** in the upper right corner. Click **Trio** to return to Viz Trio.



When you are back in playout mode, the confirmation dialog above asks you whether the scene should be reloaded on external renderers, like the external preview engine.

The template is re-imported automatically.

5.3.2 On Air Mode



The top bar contains the button above, which takes Viz Trio on and off-air. **OnAir** is *red* in on-air mode. When Viz Trio is in OnAir mode, normal playout operations to the Program engine(s) are available. When Off Air, it is only possible to read in the local preview engine.

5.3.3 Slave Mode

- Select the main menu switch **File > Slave Mode** to enable/disable slave mode.
- Slave mode means that any other Viz Trio client connected to the same Media Sequencer, which is *not* in slave mode, is in effect master.
- A master triggers the slaves to change the show folder when they change. This typically makes sense when the commands to change the folder are issued from a newsroom system, the newsroom system is then master.
- To exit slave mode, click the Main menu option **File > Slave Mode** again.

5.3.4 Show Modes

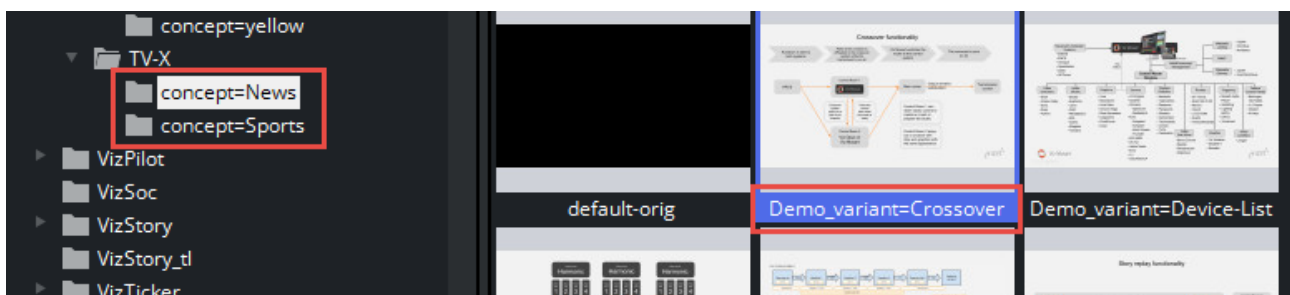
Prior to version 2.11, Viz Trio supported two show modes: traditional shows where templates and pages were part of the same view, and context enabled shows where templates and pages are split into two views. From version 2.11, Viz Trio only supports Context Enabled Shows.

Context Enabled Shows

A context-enabled show has the possibility to switch concepts and variants within a show. For example, a News concept could be switched to a Sports concept. Within that concept there could be variants of a specific scene (such as lower and top third).

Graphics for all these variations must be made, but once they are available and imported into Viz Trio, they can be switched and played out at the touch of a button.

Within a specific concept, all variants of a scene are shown as a single template in Viz Trio's [Template List](#).



For a show to successfully use context-enabled scenes in Viz Trio, the folders and scene names in Viz must follow a certain naming convention, giving them properties and values for Viz Trio to recognize and use.

Concept Naming Conventions

Type	Property		Name	Example
Concept (folder)	concept	=	<conceptname>	concept=<conceptname> concept=News concept=Sport concept=Weather
Context (folder)User-defined	<context>	=	<contextname>	<context>=<contextname> aspect=16x9 aspect=4x3 platform=HD platform=SD
Variant (file)	_variant	=	<variantname>	<scene>_variant=<variantname> banner_variant=Lower banner_variant=Top chart_variant=Graph chart_variant=Line adds_variant=Adds adds_variant=NoAdds weather_variant=3DayForecast weather_variant=1DayForecast

 **IMPORTANT!** It's important to add an underscore "_" to separate the scene name from the variant's keyword in the names of variants.

User-defined Contexts

Viz Trio supports the creation of user-defined contexts. The number of user-defined contexts is unlimited. However, user-defined contexts are only controllable through the Viz Trio command-line editor. The examples below show Viz Trio commands used with the user-defined contexts *Aspect* and *Platform*:

```
show:set_context_variable Aspect 16x9 show:set_context_variable Platform HD
show:set_context_variable X Y
```



Tip: User-defined contexts can be controlled using Viz Trio commands, which in turn can be used to create user-defined macro commands for use with scripts and shortcut keys.

See Also

- [Macro Commands and Events](#), and in particular the *show* commands.

5.4 Undo and Redo

The global functions *trio:undo* and *trio:redo* are mapped to **CTRL + Z** (undo) and **CTRL + Y** (redo). The undo only applies for deleted playlist entries.

5.5 Tab Fields Window

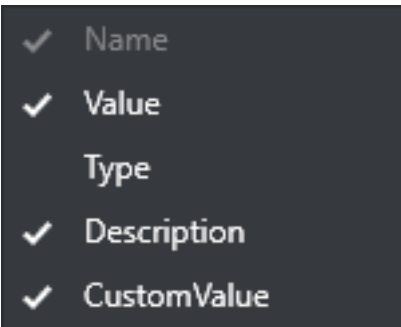
Tab Fields			
Name	Description	Value	Custom
101	Team 1 Swatch	ED_WHITE	
▼ 100	Team 1 Logo	ED_White_Logo	
image_position	Texture Position		
image_scaling	Texture Scaling		
201	Team 2 Swatch	PARKVIEW_ALT3	
► 200	Team 2 Logo	PBS_Logo_(W&R)_FIX	
005	Event Swatch	CST	
102	Team 1 Seed	8	
103	Team 1 Name	E.D. WHITE	
104	Team 1 Note	18-12	
202	Team 1 Seed	1	
203	Team 1 Name	PARKVIEW BAPTIST	

- The Tab Fields panel shows the tab fields for the page or template currently loaded. Click on a tab field to display the editor for an element.
- When a tab field for graphics elements is selected, a bounding box highlights the object in the preview window.
- When a tab field for video elements is selected, the [Search Media](#) editor is opened. A page containing video elements automatically searches for the video and previews it (see the [Video Preview](#) section).
- The tab field window has a context menu for collapsing or expanding all nodes. For more information on how to use custom properties, see the [Tab Field Variables](#) section.

5.5.1 Adding and Editing Custom Values

There are two methods for editing the custom values that correspond to a tab field. However, in order to add custom values, the Custom column must first be activated as follows:

1. Navigate the cursor to the **Tab Fields** panel in the bottom left corner of the main window.
2. Right click in the top menu.
3. Click **CustomValue**.



The Custom column now appears in the tab field panel. Now that the custom column is enabled, it is possible to edit or add a tab field's custom value using one of two methods. The first is via the **Set custom property** dialogue box and the second is with a macro command. The two methods are described below.

Set Custom Property Box

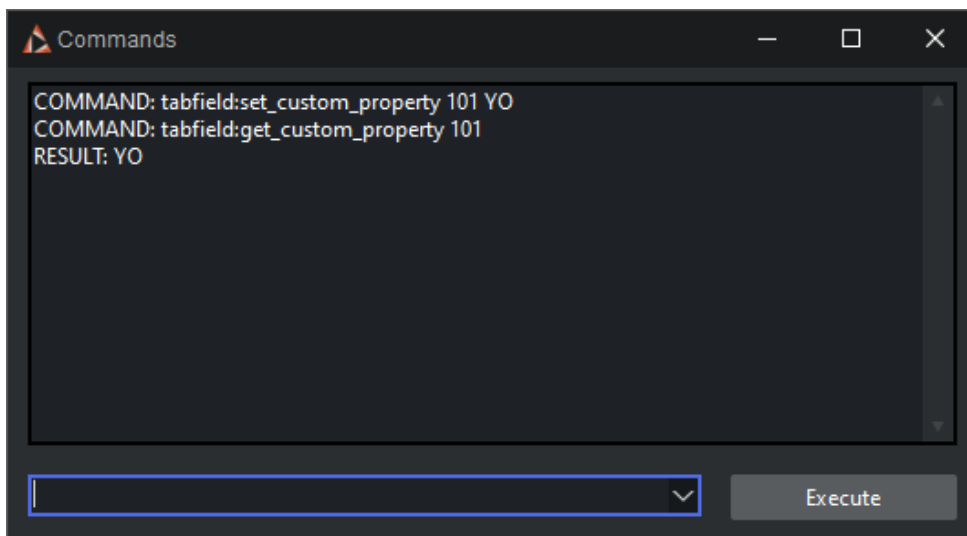
Tab Fields			
Name	Description	Value	Custom
101	Team 1 Swatch	ED WHITE	YO
▼ 100	Team 1 Logo	ED White Logo	

1. Right-click the custom value cell and select **Set custom property**.
2. Edit the text in the field that appears.
3. Click **OK** to confirm.

Macro command

1. Read up the page to which you want to set a custom tab field value.
2. Open the Viz Trio Commands window.
3. Send the command `tabfield:set_custom_property 101 YO`.

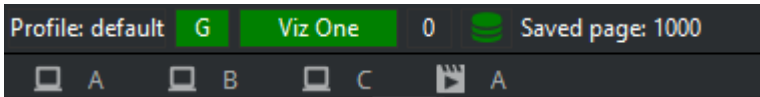
The first argument is the tab field name, and the second is the value to set. The custom value can be retrieved in a script by calling the macro command `tabfield:get_custom_value 101`.



Note: Values can be assigned and edited for top-level nodes only. This means that custom values can be added for tab fields but not the tab field properties that appear below them.

5.6 Status Bar

General status information and the Media Sequencer's connection status to devices such as Viz Engine and Viz Gateway are shown in the lower left corner of the interface:



The following icons can be displayed:

- **Profile:** Shows the current Viz Trio profile.

Note: This is not always the same as the active profile in Media Sequencer. The current profile is a private client setting, while the active profile is a Media Sequencer state where the MSE actively monitors resources in the page list and playlists.

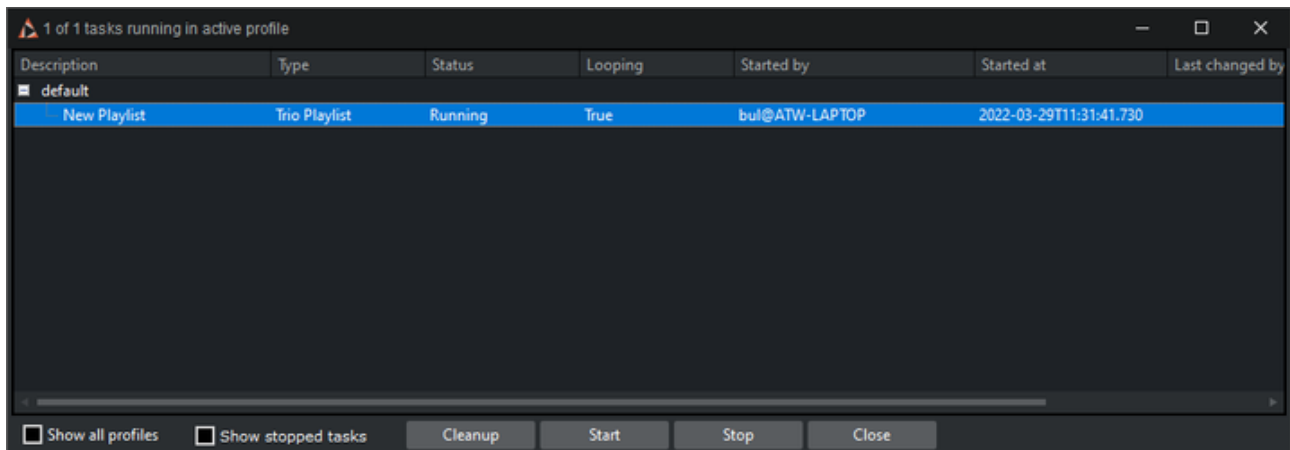
- **G:** Shows Media Sequencer's connection status to Viz Gateway. The icon indicates that a newsroom system, or several, is available. If no connection is established, the symbol is not visible. A Viz Gateway connection is used to retrieve and control playout of [Create Playlist](#). Double-click the icon to edit the Viz Gateway connection parameters (see the [MOS](#) section).
- **Viz One:** Shows Media Sequencer's connection status to the MAM system. The icon implies that one or more MAM systems, such as Viz One, are connected. A MAM connection is used to send shows and playlists, containing references to media such as video, to the MAM system. Double-click the icon to edit the [Proxy](#) connection. The indicator icon is not shown if a MAM system is not configured.
- **Active Tasks (0-n):** Indicates the number of playlist tasks that are active for the current profile(s). For details, see the [Active Tasks](#) section.
- **Database:** The green cylinder shows Media Sequencer's connection status to the Viz Pilot database. If no connection is established, the symbol is not visible.
- **Messages:** Shows messages to and from the system. For example, error messages and commands sent to Media Sequencer. Error messages are collected in the Error log. The log can be viewed directly from the status bar by clicking the Errors button (a small red error count label is displayed on the button if errors occur).
- **Channel:** Displays the channel status for all channels configured for the currently selected profile.

This section covers the following topics:

- [Active Tasks](#)
- [Configuring a Profile](#)
- [Changing a Profile](#)
- [Changing Channel Assignment](#)
- [Checking Memory Usage](#)

5.6.1 Active Tasks

Double-click the Active Tasks icon in the status bar to open the Active Tasks window.

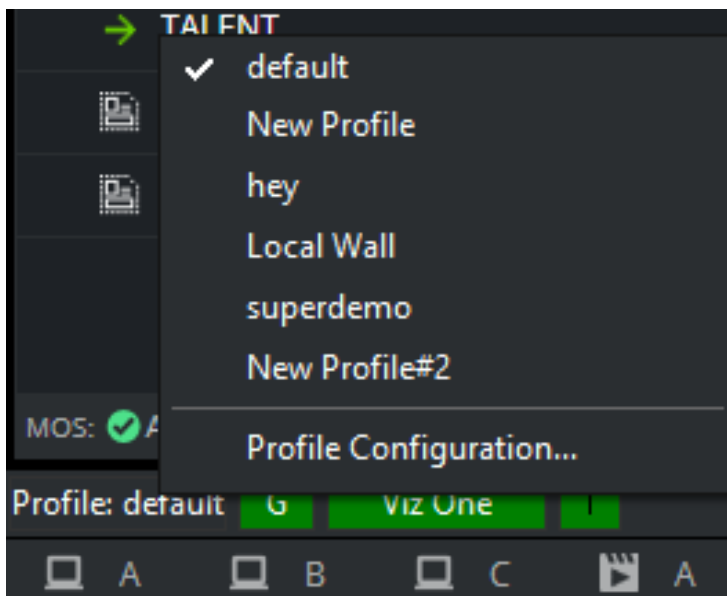


The Active Tasks window displays show-specific playlists that are running in active profiles, showing active tasks running in other profiles as well as different show playlists within a single profile.

This window contains the following options:

- **Show all profiles:** Shows all profiles with Show Playlists.
- **Show stopped tasks:** Shows all stopped tasks.
- **Cleanup:** Removes all information about stopped tasks.
- **Start:** Starts the selected Show Playlist.
- **Stop:** Stops the selected Show Playlist.
- **Close:** Closes the Active Tasks window.

5.6.2 Configuring a Profile



Right-click the current profile (displayed on the status bar) and select Profile Configuration (see [Profile Setups](#)).

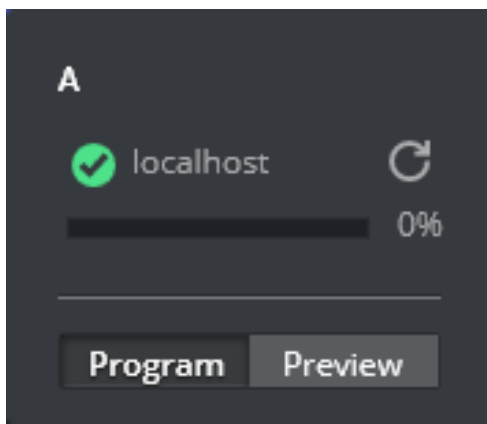
Note: You cannot change profiles unless you have the *-control* command line parameter set on startup.

5.6.3 Changing a Profile

Right-click the currently active profile (displayed on the status bar) and select the profile you want to use.

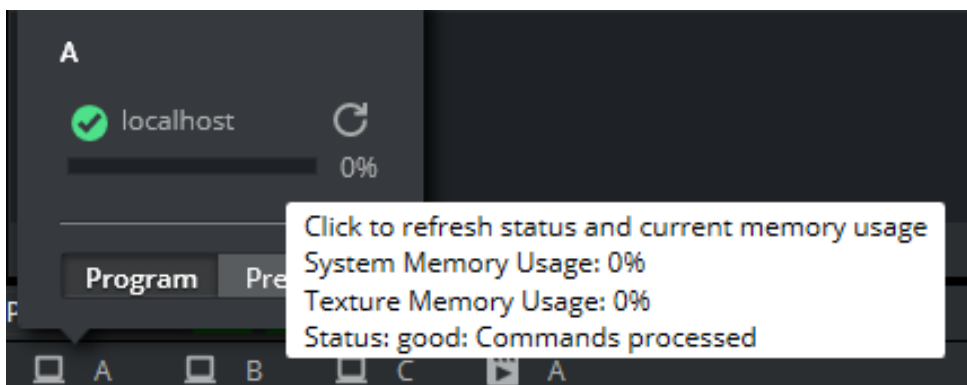
Note: Changing the profile in Viz Trio only changes the current Viz Trio profile (this is a client setting). If the configuration setting **Pagelist/Playlist > Automatically activate playlist when profile changes** is enabled, the profile which is being changed to, is also activated on Media Sequencer. Media Sequencer then actively starts monitoring resources for the page list or playlist.

5.6.4 Changing Channel Assignment



Right-click a channel and assign it as a **Program** or **Preview** channel.

5.6.5 Checking Memory Usage



Right-click a channel and hover your cursor over one of the bars to see the total usage.

See Also

- [Configuration Window](#)
- [Playlist Modes](#)

5.7 Edit Show Script

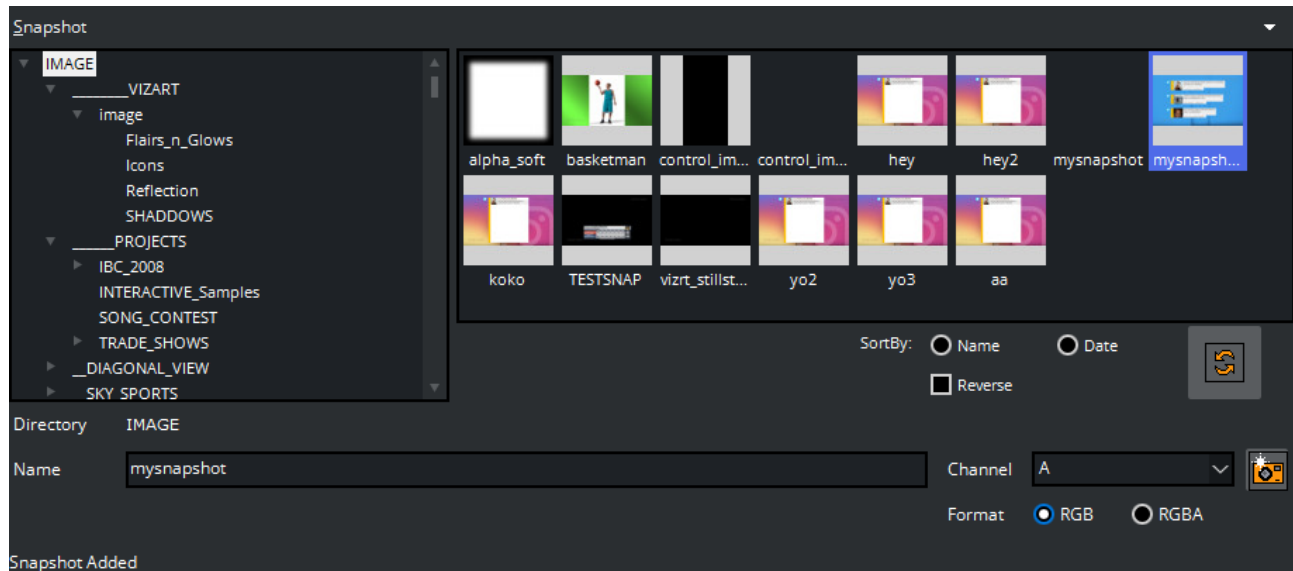
Select **Edit Show Script** from the [Page Editor](#) drop-down menu to open the **Choose Show Script** window, where you can edit the script assigned to the show.



Tip: The **Choose Show Script** window can also be opened by selecting **Script > Edit Show Script** from the [Template List](#) context menu, or by clicking the **Show Script** browse button in the [Show Properties](#) window.

5.8 Snapshot

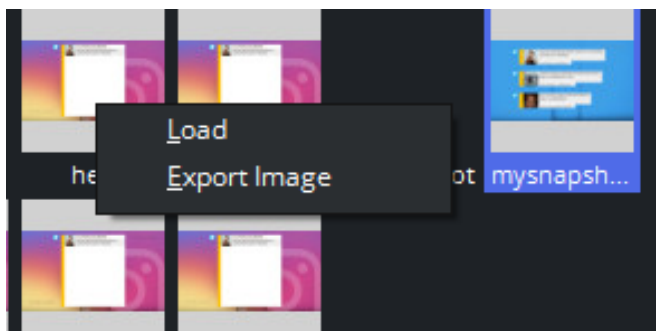
Select **Snapshot** from the [Page Editor](#) drop-down menu to open a window where a snapshot of the current view on the on-air renderer can be taken.



- Specify the channel to take a snapshot from, and select the preferred format.
- Browse the Image database to select a folder to store the snapshot.
- Images can be sorted by **Name** or **Date**. Select the **Reverse** check-box to reverse the sorting. To refresh the content of the selected folder, click the **Refresh** button.
- Then, enter a name for the image and press the **Take Snapshot**  button. A thumbnail of the snapshot is visible in the selected folder.

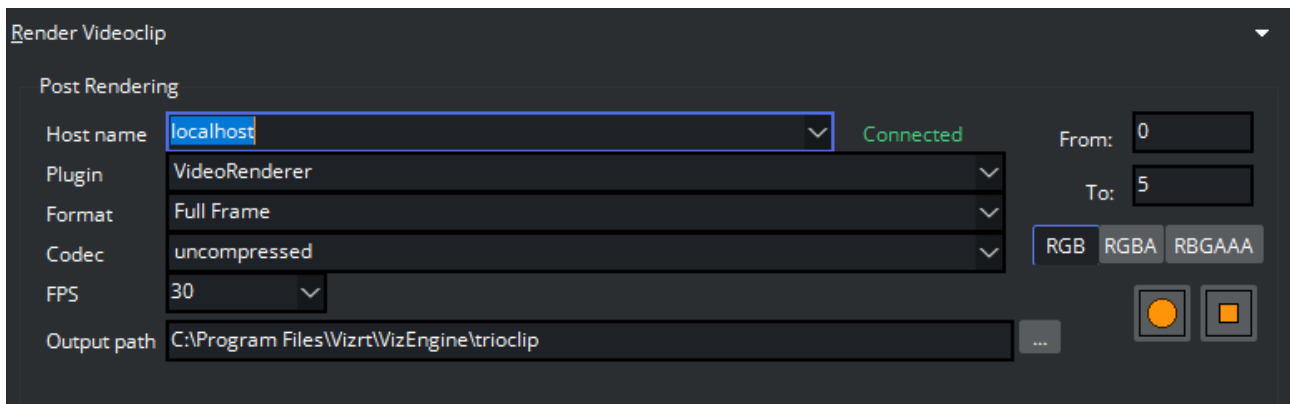
 **Note:** Snapshots can only be deleted from Viz Artist.

Right click an image to open the context menu:



- **Load:** Loads the image into the preview renderer.
- **Export Image:** Saves the image to disk.

5.9 Render Videoclip



The **Render Videoclip** option exposes an editor for post rendering pages to a device specified in Viz Artist, such as an AVI or a tape recorder.

Post rendering is used to create images or video files of graphical scenes. The files can be used for playout on Viz Engine. Selecting a video plug-in creates one file, however, selecting an image plug-in renders an image according to the configured frame rate. For example: rendering a scene for ten seconds results in 250 images if the frame rate is 25 frames per second (fps).

Rendered data elements can be full screen graphics or graphics with Alpha values such as lower thirds and over the shoulder graphics.

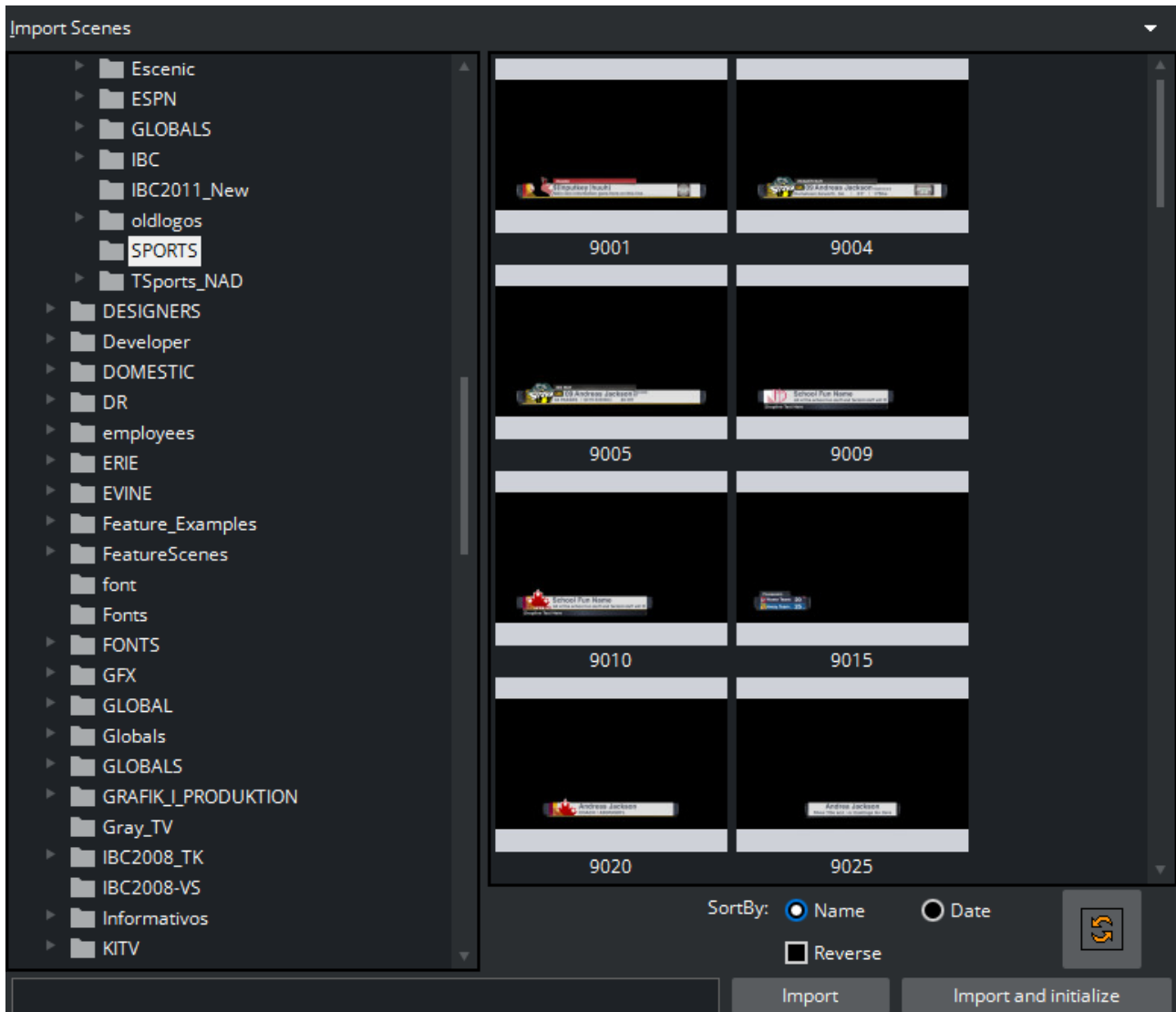
Note: Only scene-based pages can be post rendered.

The settings in the post render frame stay the same after restarting Viz Trio.

- **Host name:** Sets the Viz machine that post renders the scene. It is also possible to define the port number (`<host-name> : <port-number>`).
- **Plugin:** Sets the renderer device.
- **Format:** Sets the format. Available options are full frame, fields top, fields bottom, full frame/interlaced top, full frame/interlaced bottom and full frame skip. The options vary depending on the Plugin selected.
- **Codec:** Sets the codec to be used.
- **FPS:** Sets frame per second. Available options are 30, 60, 29.97 and 59.94.
- **Output path:** Sets the output path. A corresponding browse button is available when localhost is selected as host. As Viz Trio cannot browse the file system on other machines, manually make sure that the output path is valid if defining the path on another machine. A full or relative path can be added. If no path is given, the file is stored in the Viz program folder on the rendering machine.
- **From:** Sets the start time in seconds.
- **To:** Sets the end time in seconds.
- **RGB:** Sets the pixel format to RGB.
- **RGBA:** Sets the pixel format to RGBA which includes the alpha channel (blending/transparency).
- **Record button (circle):** Starts the rendering process.
- **Stop button (square):** Stops the rendering process before the configured stop time.

5.10 Import Scenes

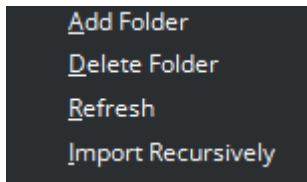
Select **Import Scenes** in the drop-down menu to open a window where Viz Artist scenes can be selected and enabled for Viz Trio.



Select scenes to import into the template list and click **Import**. You can select multiple scenes and import scenes from any scene folder. To quickly make the scenes ready for playout, click **Import and Initialize**, this loads all scenes on the program and preview renderer (Viz Engine).

Sort scenes by **Name** or **Date**, **Reverse** sorting preferences or **Refresh** content.

5.10.1 Import Recursively



Import Recursively lets you import a whole folder or a folder tree structure, and it does not initialize the show on the renderers.

Right-click the folder in the scene tree and select **Import Recursively**.

 **Caution:** If a scene has the same name as an existing template in the current show, an attempt is made to merge it with the existing template (for example, scene 1000 with template 1000).

5.10.2 Import Scenes with Toggle or Scroller Plugin

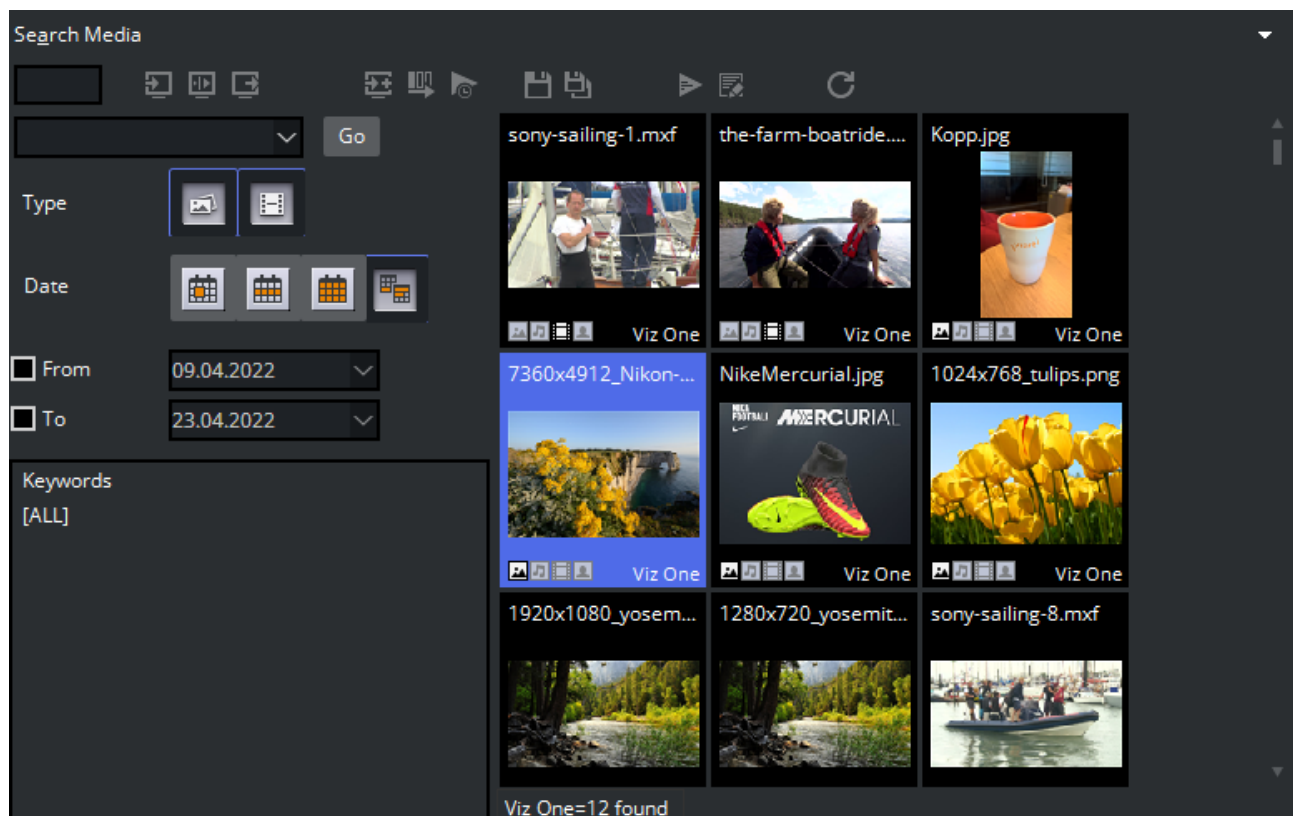
A warning appears when you attempt to import scenes with a Toggle or Scroller plugin, since such scenes usually do not need to be imported. However, if you are using a Toggle or Scroller plugin in scenes that are not Transition Logic or ticker-related, you have the option of importing the scenes as templates.

5.11 Search Media

Select **Search Media** from the top right drop-down menu. Vizrt's MAM system Viz One stores videos and stills. The Search Media editor lets you search Viz One MAM system for images and video clips.

Note: Viz Trio does not support the use of audio files.

Media items can be dragged and dropped into a show, show playlist, playlist, template or page (for example, full screen or part of the graphics), where they can be double-clicked to open the **Search Media** editor.



The left pane shows the [Search and Filter Options](#) and a list of categories, while the right pane displays the search result.

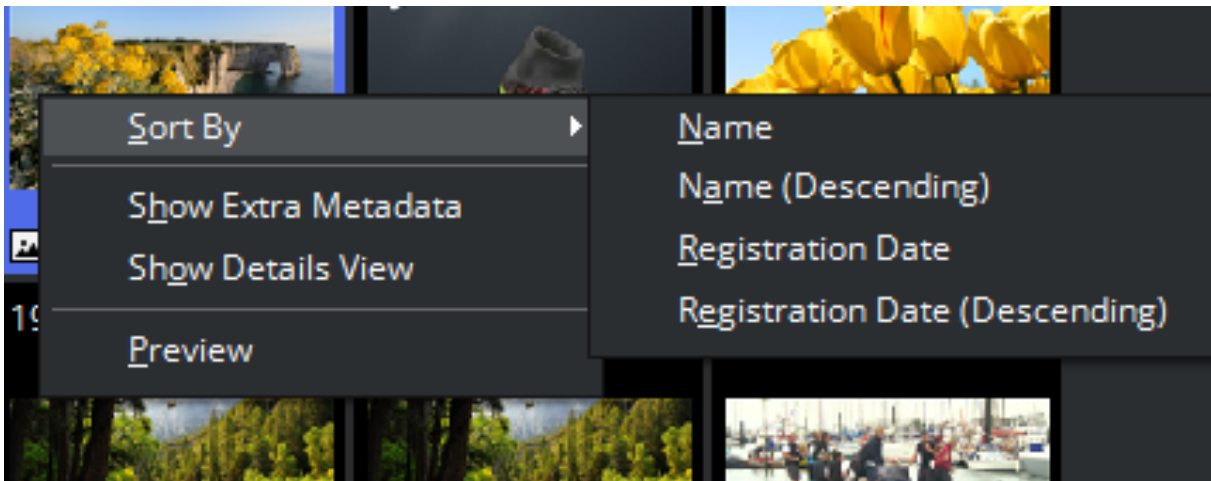
Tip: The number of fetched media files can be specified in the [General](#) section of Viz Trio Configuration. Scroll down to the bottom of the list to fetch additional media files.

When the media search frame is shown and you have many hits only the first N hits are shown. When scrolling down in the result list, the next N hits are fetched. There is no more button. The next chunk of hits get automatically searched when scrolling down to the bottom.

This section covers the following topics:

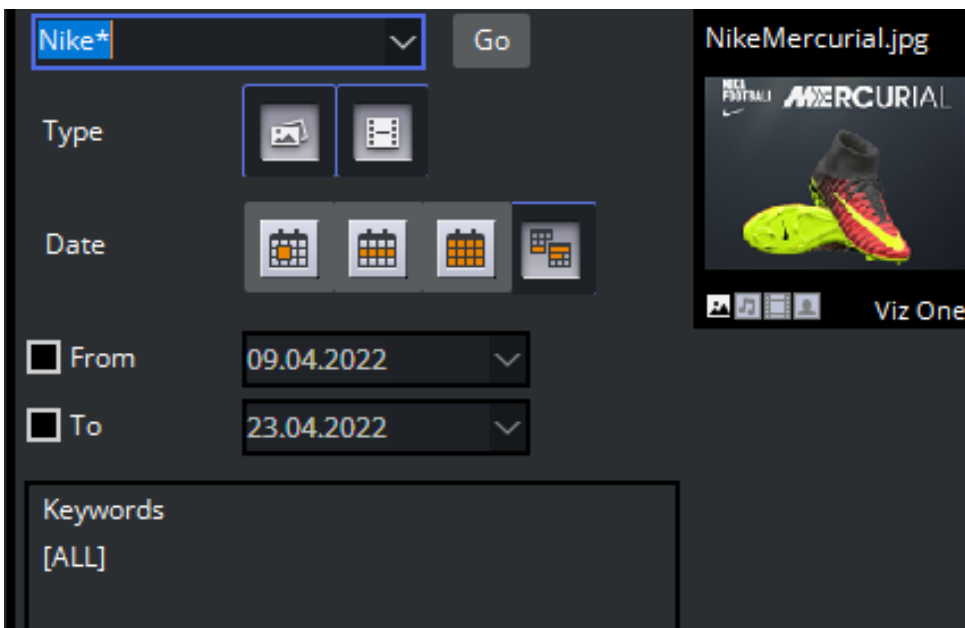
- [Media Context Menu](#)
- [Search and Filter Options](#)
- [Order Metadata Fields for Video and Images](#)

5.11.1 Media Context Menu



- **Sort By:** Displays a sub menu with sort options.
 - **Name/Name (Descending):** Sorts by name in ascending and descending order.
 - **Registration Date / Registration Date (Descending):** Sorts by registration date in ascending and descending order.
- **Show Extra Metadata:** Switches the media icons to display meta data such as complete filename, creation date, and clip length and so on.
- **Show Details View:** Opens a side panel with data about the selected media asset.
- **Preview:** Previews images and clips by opening the default browser with the full resolution image, or a low resolution variant for clips.

5.11.2 Search and Filter Options

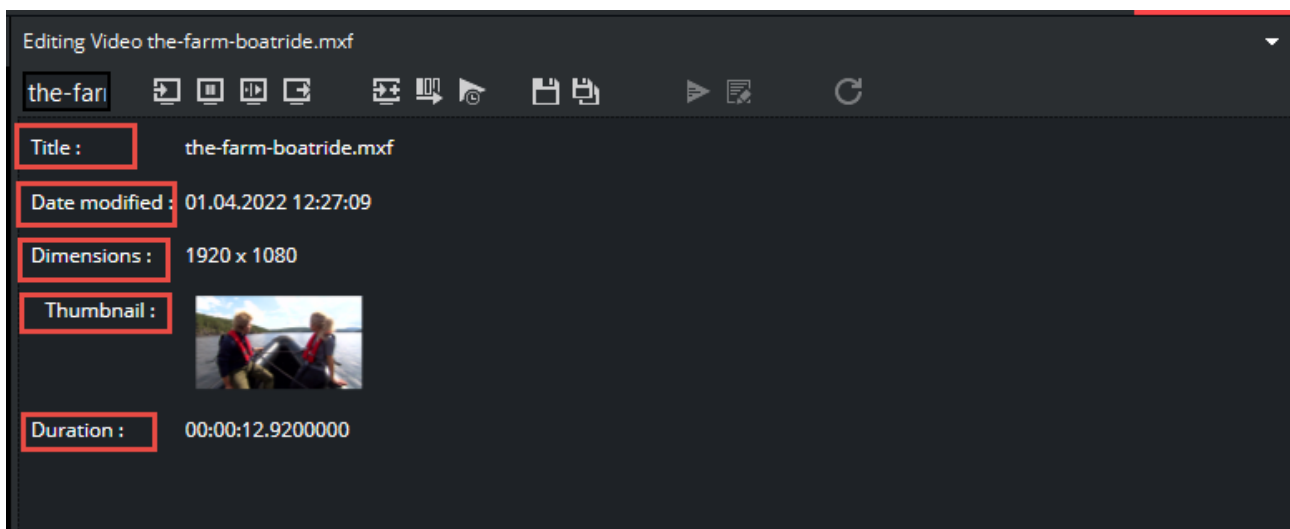


- **Search field:** Combo box for entering a search criterion. Previously entered search criteria are remembered per session.
- **Filter:** Enables/disables the image or video filters.
- **From/To:** Filters the search result based on From and To registration dates.
- **Keywords:** Filters the search result based on keywords.

Note: The search string is literal. For example as shown above, a search for *Nike* does not return any results. Adding or prepending the search string with an asterisk denotes free text search. Searching for *Nike** or **Merc** returns the asset named *NikeMercurial*.

5.11.3 Order Metadata Fields for Video and Images

When opening a full screen clip, fields in the metadata view can be dragged and dropped to change their order. This is typically used to display important fields at the top of the list.

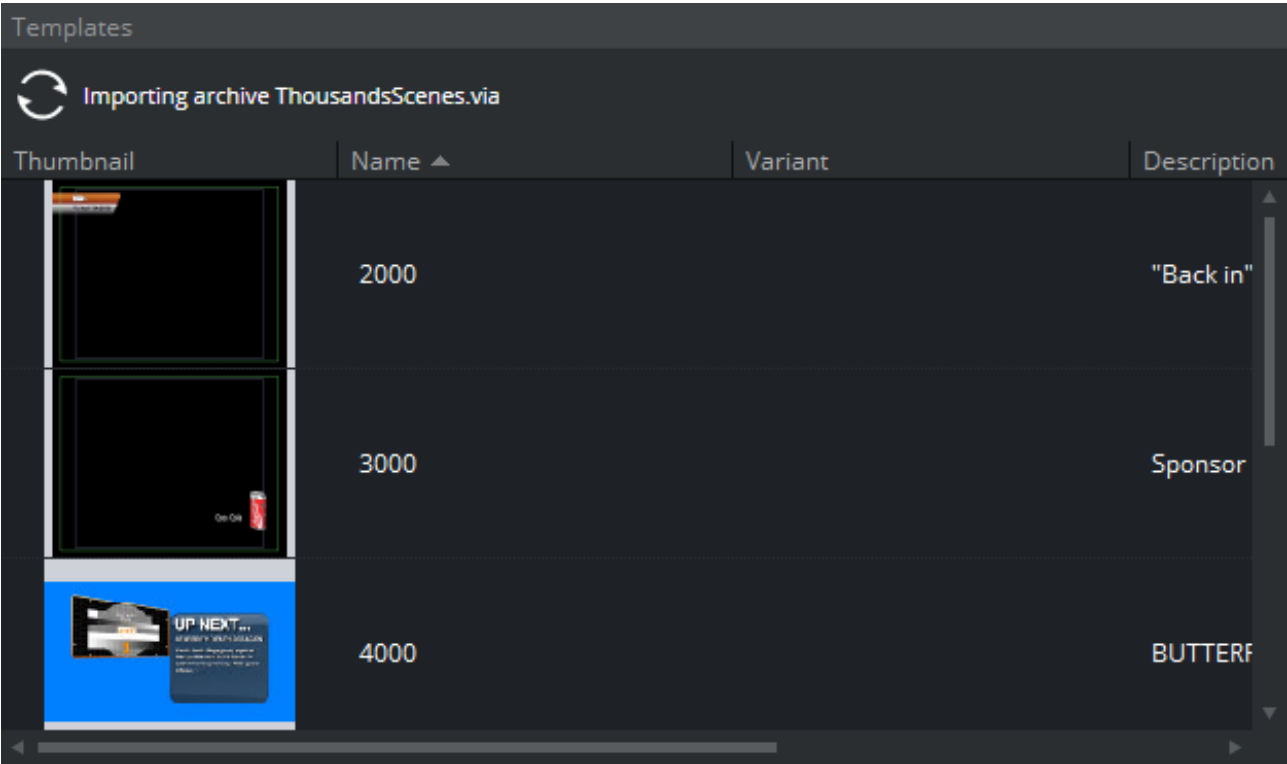


5.12 Import Viz Engine Archive

This option imports the scenes found in the selected Viz Engine Archive (.via) files into the Graphic Hub. Imported scenes that are already in use by the current Viz Trio Show, automatically trigger a re-import of the corresponding template in Viz Trio.

Click **File > Import Viz Engine Archive....** to select Viz Engine Archive (.via) files from disk. This process can also be triggered through the macro command `gui:import_viz_archive.`

If multiple .via files are selected, a search is first made for any potential conflicts between the archives.

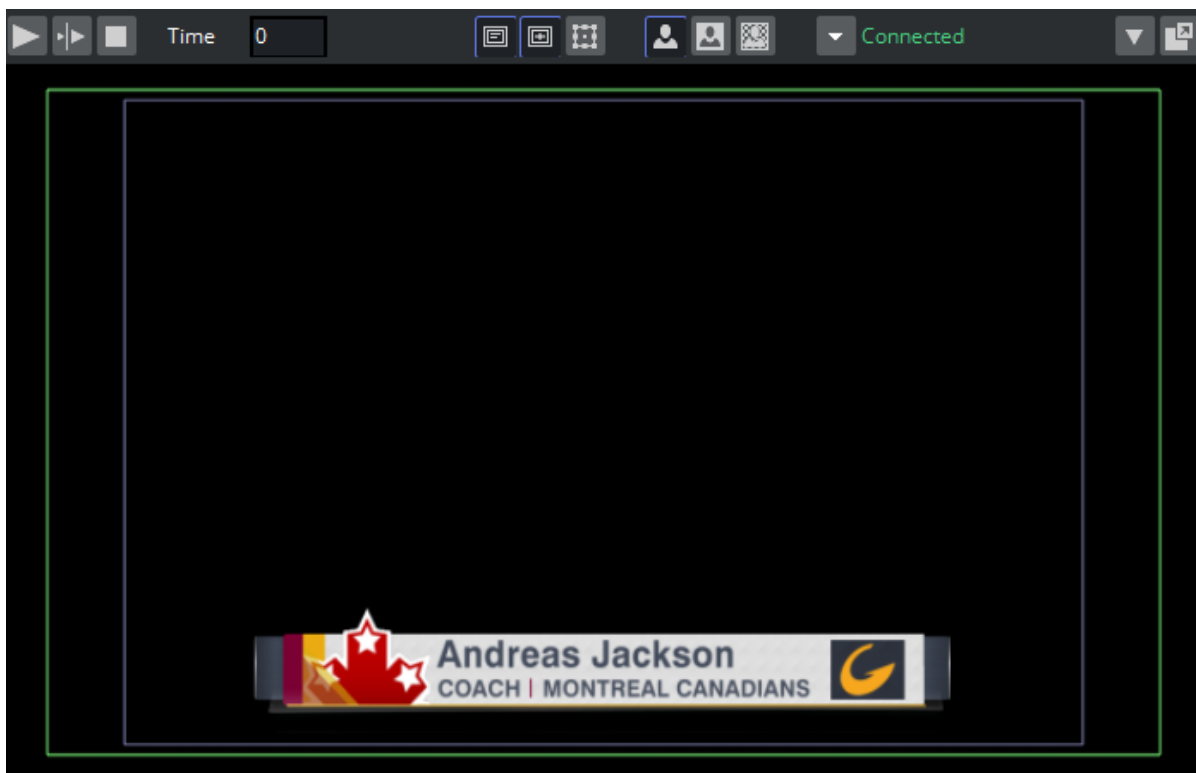


5.13 Graphics Preview

This section covers the following topics:

- [Real-time Rendering of Graphics and Video](#)
- [Adaptive Storytelling](#)
- [Floating or Moving the Preview Window](#)
- [Connection Status](#)
- [OnPreview Script for a Graphical Element](#)

5.13.1 Real-time Rendering of Graphics and Video



The local Preview window displays a representation of the graphics and video.

- The defined title and safe area and bounding boxes are displayed. Bounding boxes are related to the graphics scene's editable elements (see [Tab Fields Window](#)).
- Timeline for scrubbing full screen video preview.
- Control interactive scenes by clicking the scene inside the local preview window. You can then adjust an offside line in Viz Trio's local preview by reading a template referencing an interactive scene, for example.

5.13.2 Adaptive Storytelling

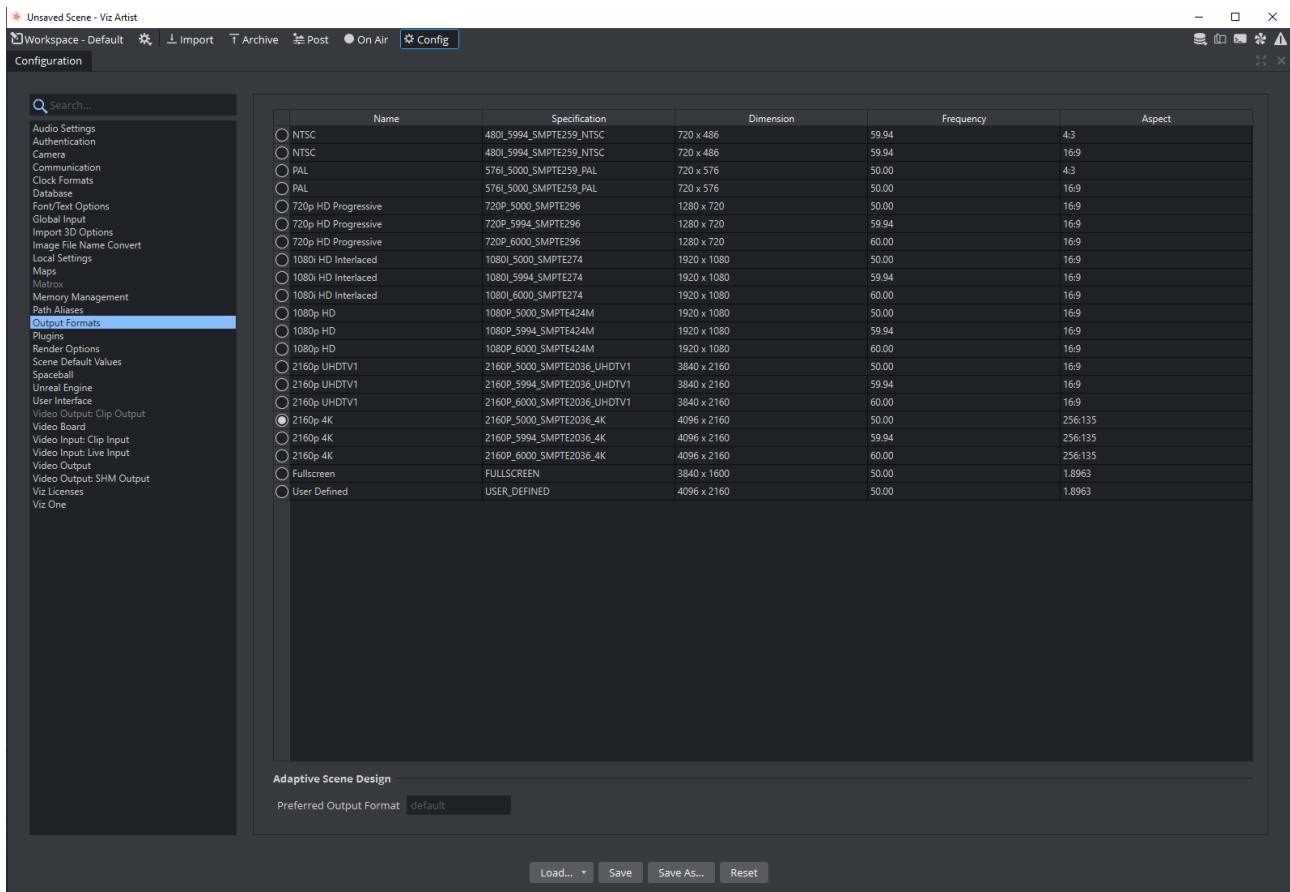
The adaptive storytelling allows you to preview scenes with different formats that were previously set in Viz Artist when designing the scene.

Info: Adaptive Storytelling can only be used with scenes based on the Viz Engine renderer, and not the Classic renderer.

When working with adaptive formats, it is recommended that the naming convention used across all Viz products and scenes is consistent, for instance *portrait* and *square*.

Before importing a scene in Viz Trio, setting the adaptive format for the output engines can be done directly in the user interface of Viz Artist or in Viz Engine.

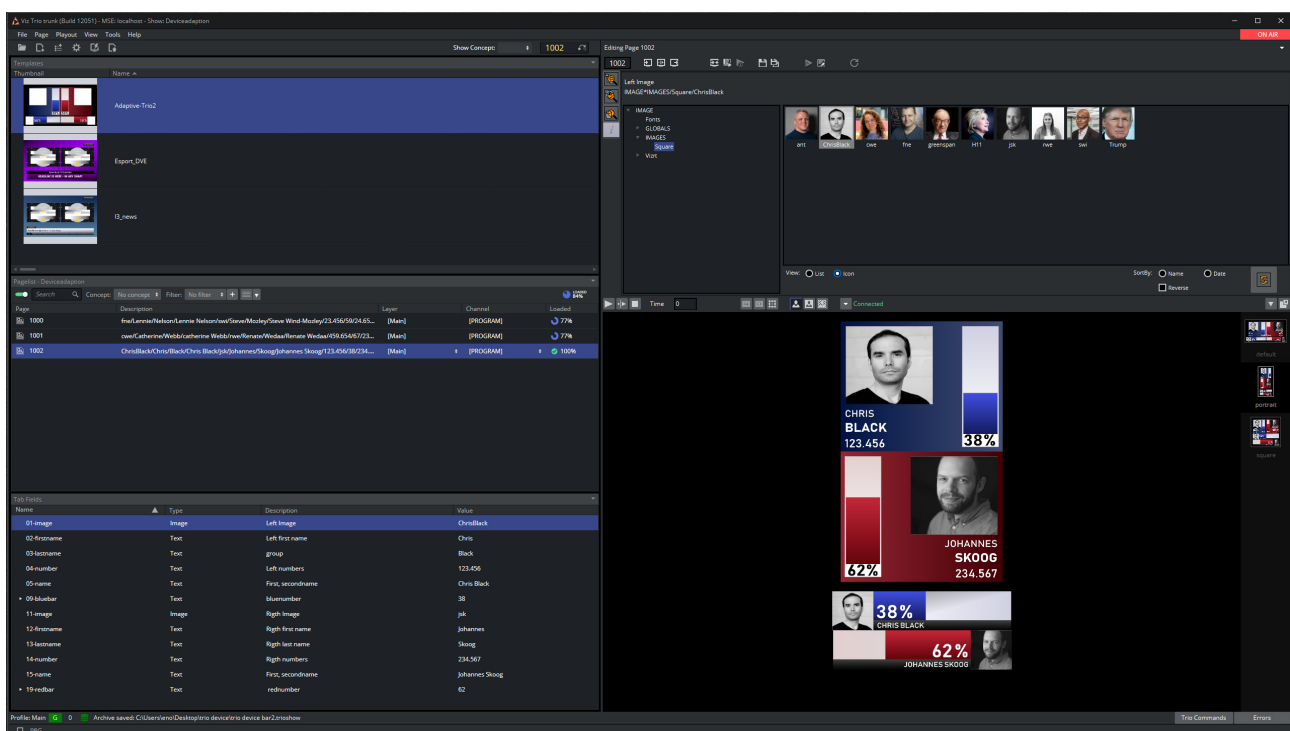
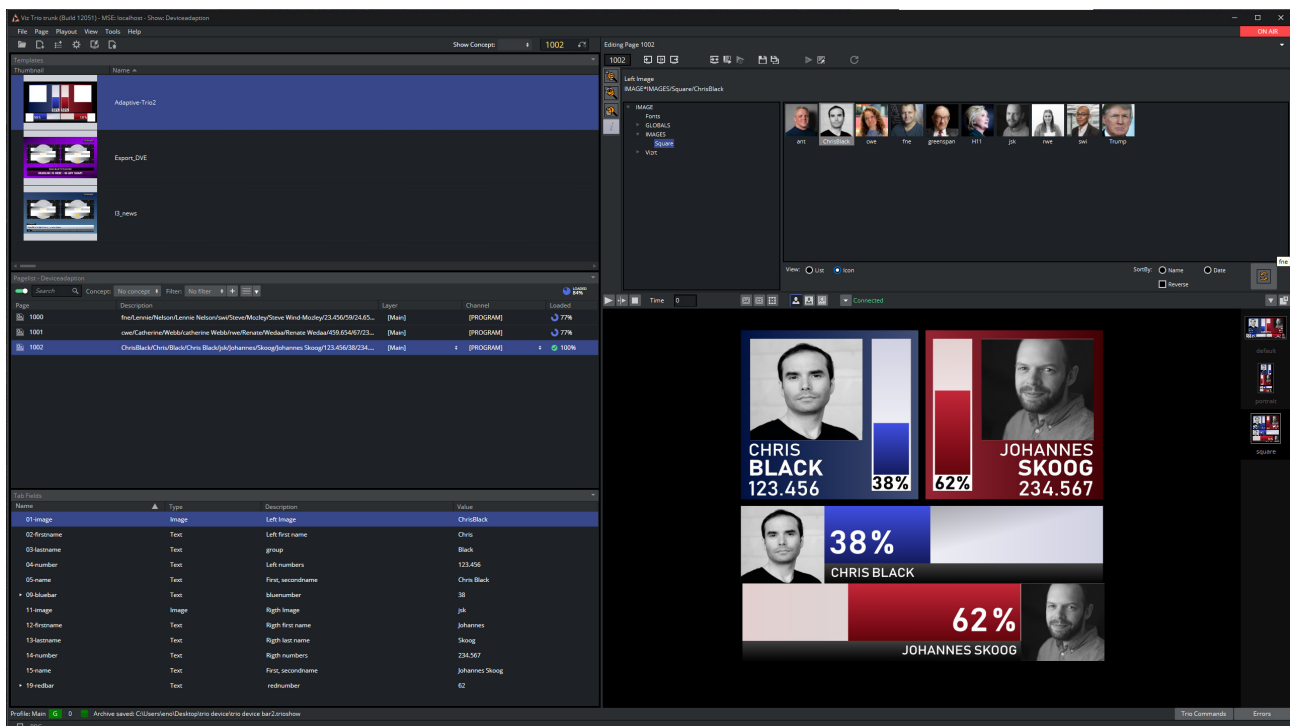
To set it in *Viz Artist*, open **Configuration** and select the **Output Formats** to set the **Preferred Output Format**:



To set the adaptive format in *Viz Engine*, please see the **Output Format** in the [Viz Engine Administration Guide](#).

Warning: All engines used for preview and playout must have a Flexbox license.

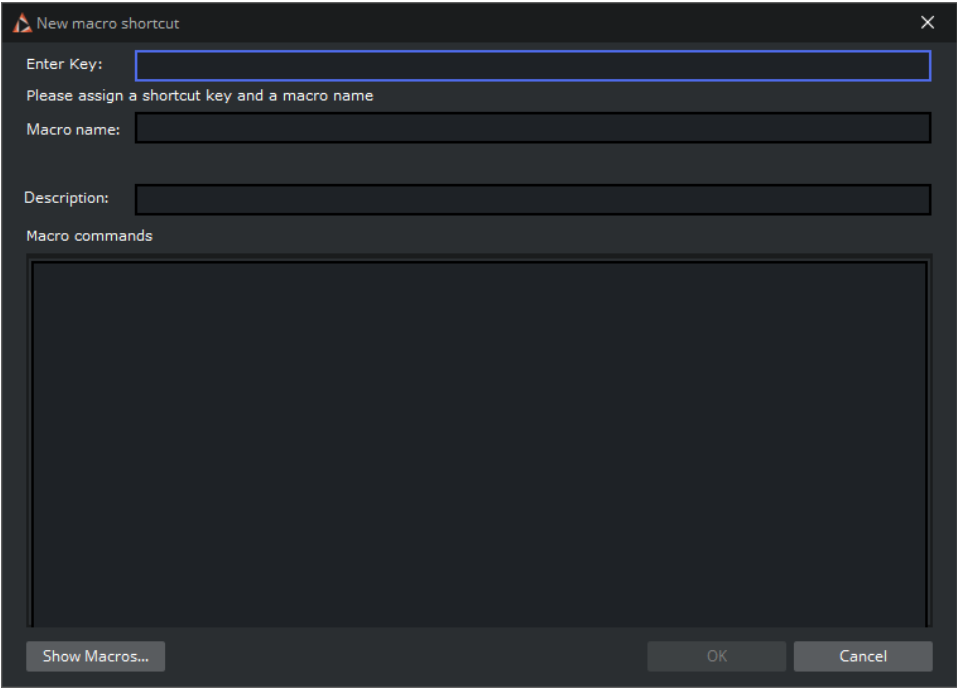
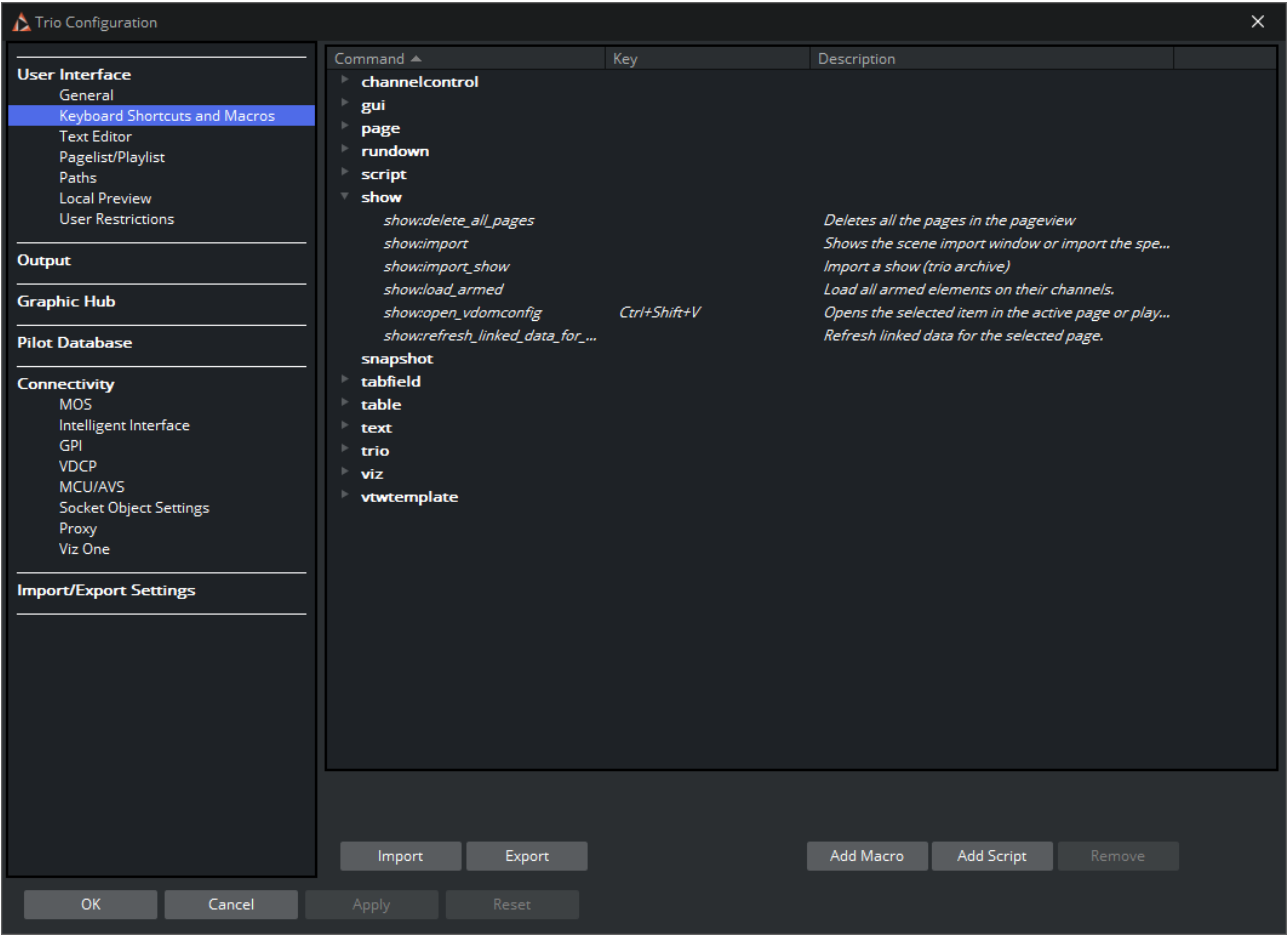
After setting the config file, open Viz Trio and import a scene. The watermark for the preview versions is shown, and the different formats for this scene are listed as thumbnails to the right of the preview.



In the examples above, you can see the preview of the default, portrait and square formats.

To switch between previews of the different formats, either click the thumbnails to the right of the preview, or use the Viz Trio macros.

For each adaptive format, you can set different macros. Open the Viz Trio configuration, select **Keyboard Shortcuts and Macros** and click **Add Macro**:



- **Enter Key:** Combination of keys for the macro, for example SHIFT+5.

- **Macro name:** Name of the macro.
- **Description:** Small description of the macro.
- **Macro commands:** Set of commands to be executed.

 **Note:** The macros available are **set_format**, **set_next_format** and **set_previous_format**.

For more information, please check the [Macro Commands](#) section.

 **Note:** Transition Logic is currently not supported in Adaptive Storytelling.

5.13.3 Floating or Moving the Preview Window

Use the **Floating preview** button at the upper right corner of the preview window pane to un-dock the preview window.

- If the preview window is docked (attached) to the Viz Trio main window, clicking the pin-button detaches the Preview window.
- If the preview-window is un-docked from the main Viz Trio window, clicking the pin-button docks the preview window to the main Viz Trio window.

Graphics Control Buttons

At the top of the preview window:



- **Play:** Starts the scene animation currently loaded in the local preview.
- **Continue:** Continues the scene animation currently loaded if it contains stop points.
- **Stop:** Stops the scene animation.
- **Time Slider:** Lets you scrub the timeline by holding in and moving your cursor in the value field. The cursor changes to an arrow to indicate that the time value can be changed.

The middle section of the toolbar:



- Show or hide the Title Area in the Preview window.
- Show or hide the Safe Area in the Preview window.
- Show or hide the Bounding Box for the tab field currently selected in the Preview window.

Preview Modes RGB, Key and RGB with Background Image



The three buttons can be used to see the previewed graphics in three different modes.

- The left button displays the graphics in an RGB mode with a default background color (black).
- The middle button displays the graphics in Key mode, showing how the key signal looks like.

- The right button displays an RGB preview with a background image, illustrating how the graphics appear on top of a video.

The respective preview modes are shown below:

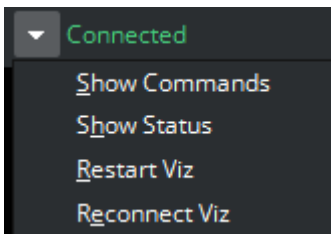


Info: To set a preview background image, go to **File > Show Properties > Preview background image** and select an image.

5.13.4 Connection Status

Connected

The **Connection** status of the local Viz renderer is shown in the upper right corner of the Preview window. Click the **arrow** to display a context menu.



These functions are mainly for testing and debugging purposes. The context menu can show these menu items:

- **Show Commands:** Opens the command shell for the local Viz Engine renderer.
- **Show Status:** Shows a status box for the local Viz Engine renderer.
- **Restart Viz:** Restarts the local Viz Engine running on the same computer as the Viz Trio client. This menu item is only active for local preview.
- **Reconnect Viz:** Allows Viz Trio to reconnect to a local preview engine after an engine running the local preview has been shut down. This option is visible regardless of whether an external preview is running.
- **Reconnects to the external Viz Engine:** Only the local Viz Trio client reconnects to the remote Viz Engine. If Viz Trio has been started with an *external* Viz Previewer (and only then), the following menu choice appears in the context menu:
- **Set OnAir:** Sets the external Viz Engine in OnAir Mode, so for example, Viz Artist's Design Mode exists and functions as a renderer engine.

Note: In this context, External Viz Previewer should not be confused with External Viz Preview settings in the *profile*. An External Viz Previewer is another way of making an external Viz Engine act as a local Viz preview. You can configure the previewer using the Viz Trio **startup parameter** `-vizpreview your-vizpreview-hostname-here`.

5.13.5 OnPreview Script for a Graphical Element

This feature supports triggering an OnPreview event in a script for a scene. The event is only triggered for the local preview in Viz Trio.

OnPreview scripts can be created using a whole scene or a container.

Creating a Scene Script:

1. Open a scene in Viz Artist.
2. Go to **Scene Settings > Script** and select **Edit** (if it is not already showing in the script editor).
3. Enter a script in order to write a unique ID for the scene, so that it can later be identified in the console output:

```
sub OnPreview(doPreview as Integer) PrintIn "OnPreview in test1 scene script: "
& doPreview end sub
```

4. Press **Compile & Run** below the script editor.
5. Save the scene.

Creating a Container Script

1. Add a group container to the root of the scene.
2. Go to **Built Ins > Container Plugins** and select the **Global** folder.
3. Drag the **Script** plugin onto the container/group. A script editor opens.
4. Enter a script that writes a unique id for the scene that can be identified in the console output later.

```
sub OnPreview(doPreview as Integer) printIn "OnPreview in test1 container
script: " & doPreview end sub
```

5. Press **Compile & Run**.
6. Save the scene.

Follow the steps below in Viz Trio and open the local preview command window to check that the OnPreview event is triggered:

1. Import a scene (Transition Logic scenes, Combo Templates, Concept/Variant scenes, TrioScroll scenes, Stillstore image scene etc.) with an onPreview script into Viz Trio. See how to make an OnPreview script above.
2. Open the local preview Viz engine console output window from **Connect > Show Command**.
3. Read the imported template and do a page take.
The page editor is used for editing pages. Depending on the template or page edited, page editor can display a wide range of specialized editors. Pages are edited using a wide range of editors: for example text, database and scroll editors. When pages have editable elements, it is possible to jump from element to element by pressing the **Tab** key. It is also possible to select elements by clicking them in the local preview window. The page editor is located in the upper right corner of the UI and is shown by default when starting Viz Trio.
4. Save the page.
5. Read the page by double-clicking it or issuing a Viz Trio command. The local Viz engine console then triggers an OnPreview script event.

5.14 Video Preview

In addition to [Graphics Preview](#), the local preview window can also show a low resolution version of full screen video clips, which are added directly to the show or playlist.

5.14.1 Video Control Buttons and Timeline Display



Located below the preview window:

- **Play:** Starts and stops the video clip currently loaded in local preview. If a Mark in point has been set, the clip jumps back to the Mark in point. Clips play automatically when loaded in Preview.
- **Pause:** Freezes and unfreezes the video clip currently loaded in local preview.
- **Stop:** Stops and rewinds the video clip currently loaded in local preview. If a Mark in point has been set, the clip jumps back to the Mark in point.
- **Timeline:** It has a marker and you can to set Mark in and out points. The clip length is displayed in hours, minutes, seconds and milliseconds. If a Mark in and out point has been set, the length of the clip shows the time of the clip marked by the in and out points.
Use the Mark in and out points to create new time code references for Viz Engine. Save the video clip and Viz Engine only plays out the frames between the Mark in and out points. Use **Save As** to create new and edited versions of the original clip.

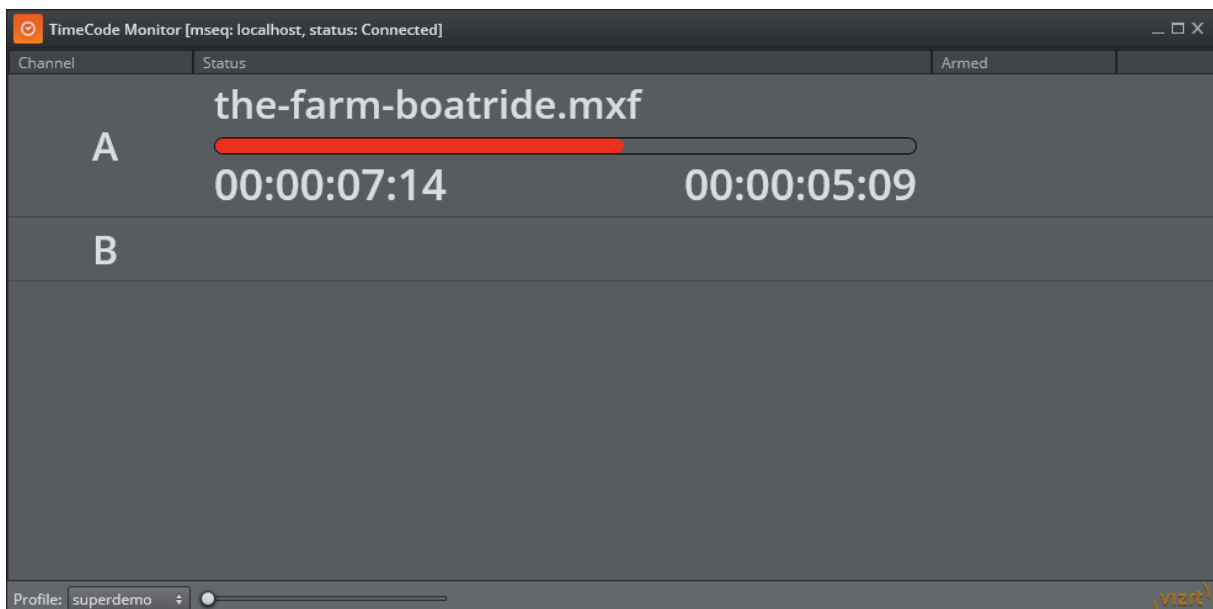
5.15 TimeCode Monitor

The TimeCode Monitor (TC Monitor) shows the progress of the current video clip on a specific channel.

This section covers the following topics:

- [TC Monitor](#)
- [TimeCode Monitor Options](#)
- [Enable the TimeCode Monitor](#)
- [Disable the TimeCode Monitor](#)
- [Monitor a Video Time Code](#)

5.15.1 TC Monitor



Note: The TimeCode Monitor keeps track of the time code and the video server channels. The system must therefore be configured to be integrated with a video server. In order to monitor video clips embedded in graphics, the render engine must also be defined as a video channel (see [Output](#)).

5.15.2 TimeCode Monitor Options

The TimeCode Monitor displays three columns: Channel, Status and Armed.

- **Channel:** Channel name. Video channels are displayed as light grey rows, with progress information in the **Status** column. Non-video channels are shown as dark grey rows without status.
- **Status:** Name of the current item, followed by any cued item, displayed above the progress bar. Current Location time and Remaining time are shown below the progress bar. The progress bar changes color to indicate its status:
 - *Green:* Video playing.
 - *Red:* Video playing, less than 10 seconds remaining.

- *Blue*: Video finished.
- **Armed**: Displays the armed state of an element on a channel.

List rows can be shown or hidden using the context menu on each item:

- **Hide**: Hide the selected channel.
- **Show All**: Show all channels.

At the bottom of the window, **zoom** lets you adjust display size. **Profile** displays the current profile name, and contains a list of other profiles to choose from.

The Timecode Monitor opens in full screen by default. Press **ESC** to exit full screen.

5.15.3 Enable the TimeCode Monitor

To activate the TimeCode Monitor, go to **Tools > TimeCode Monitor** in the main menu.

Alternatively, use the command:

```
gui: show_timecode_monitor
```

5.15.4 Disable the TimeCode Monitor

To disable the TimeCode Monitor, close the window or use the command:

```
gui: hide_timecode_monitor
```

5.15.5 Monitor a Video Time Code

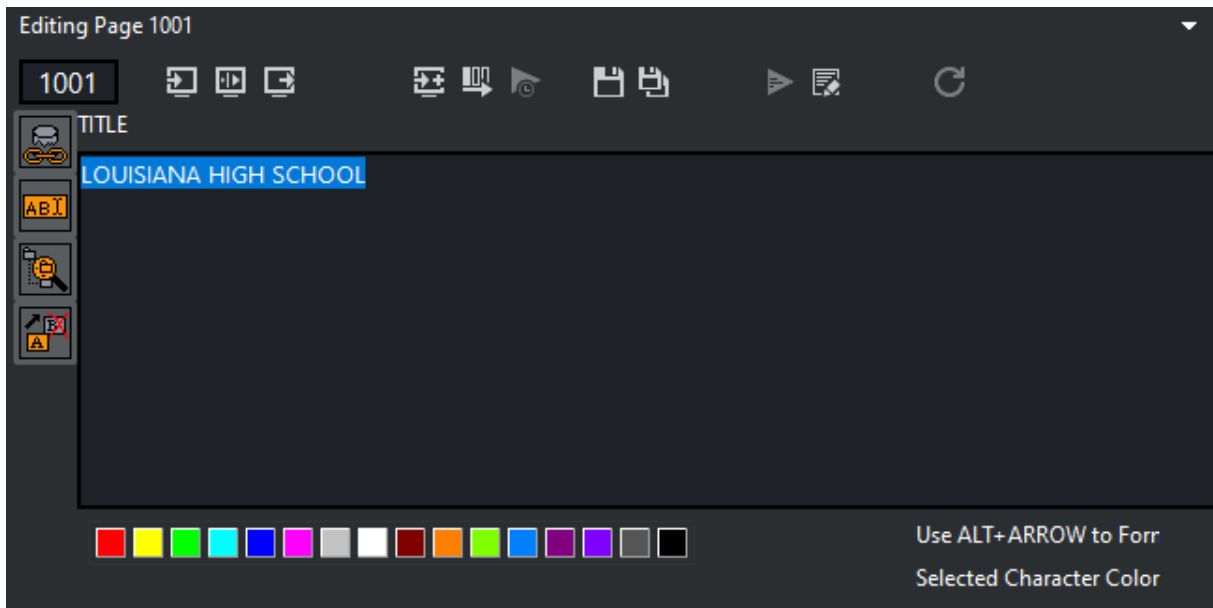
1. Make sure the TimeCode Monitor is activated, see [Enabling the TimeCode Monitor](#).
2. Double-click a video or composite element (graphics and video combined).
3. Take the video on the video output channel. The progress of the video is displayed in both the TC Monitor, and the inline TC Monitor above the local Preview panel.

5.16 Page Editor

Depending on the template or page edited, the page editor can display a wide range of specialized editors, including text, database and scroll editors.

When pages have editable elements, it's possible to navigate between elements by pressing **TAB**, or to select elements by clicking them in the local preview window.

The page editor is located in the upper right corner of the UI and is displayed by default when you start Viz Trio.



5.16.1 Edit a Page

The following example uses a page to edit common text properties:

1. **Double-click** or **Read** a page to open it in the page editor.
2. Click the **Kerning** button, or press the [Cherry Keyboard](#)'s **Kern** key.
3. Click the **Text** button to go back to text editing mode, or press the keyboard's **Text__** key.
4. Click the **Take** button to play out the page on-air, or press the **Take** key on the keyboard.
5. Click the **Save** or **Save As** button, or press the **Save** or **Save As** key on the keyboard.

Note: It is not necessary to save a page before playing it out on air. However, a template cannot be played out before it has been saved as a page.

This section covers the following topics:

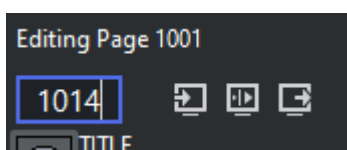
- [Edit a Page](#)
 - [Controls](#)
 - [Callup Code](#)
 - [Field Linking](#)
 - [Options in the Field-Linking Panel](#)
 - [Payout Buttons](#)
 - [Save Buttons](#)

- Script Buttons
- Refresh Button
- Variants
- Create a Page
- Select a Variant from the Page Editor
- Text
 - Text Editor Context Menu
 - Word Replacement
 - Edit a Text Item
 - Edit Text Color
 - Edit Text Kerning
 - Edit Text Position
 - Edit Text Rotation
 - Edit Text Scaling
 - Edit Text Transparency (Alpha)
 - Replacing a Word
- Database Linking
 - Database Link Editor Common Properties
 - Database Link Editor for Scalar Linking
 - Database Link Editor for Table Linking
 - Microsoft Excel and Unicode Support
- Image Property Editor
- Transformation Properties
 - Setting Transformation Properties
- Tables
 - Populate Tables Using Macro Commands
- Clock
- Maps
 - Maps Classic Client
 - Maps Second Edition Client
 - Maps Lite Edition (inline)
 - Control World Plugin Interface

Controls

The main functionality of the editor window consists of the layout, script, save and transition effect buttons. In addition, a range of editors can be used to edit the different properties of a template or page.

Callup Code

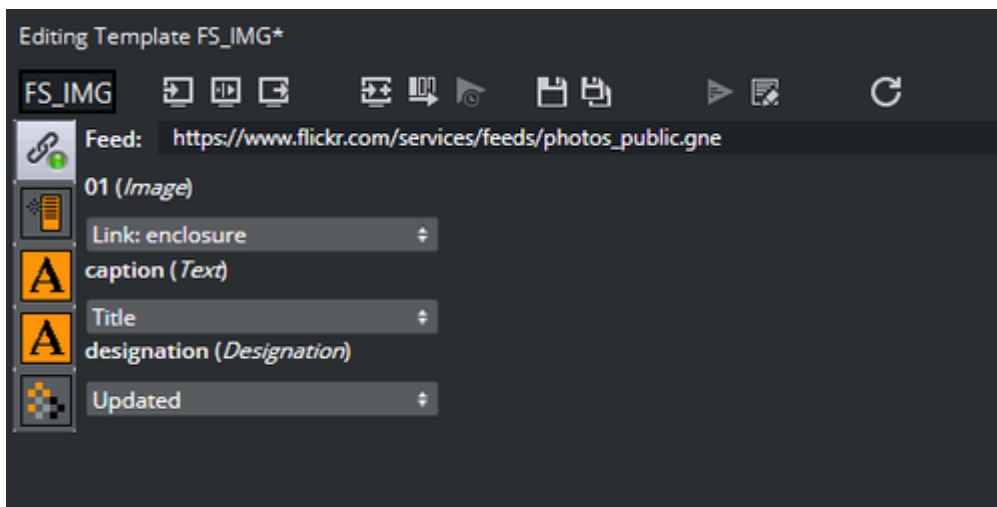


The callup code field is used to save callup codes for pages. The callup code can be set when saving a template as a page for the first time, or when saving a page as a new page.

Field Linking



When working with templates (not pages), the field linking button is shown in the properties area of the page editor. The button opens a panel, where it is possible to define a feed URI and to define how tab field properties in the template are linked to properties in the external feed.



Feed URIs can link to Twitter, Flickr, and so on. After the feed has been defined, a new property icon (the **select values button**) appears in the page editor: .

Clicking the **select values button** displays a panel where you can browse for items from the specified feed, which can be used as tab field values. The list of feed items can be displayed and sorted in various ways, and a filter can be added to narrow down the search.

Options in the Field-Linking Panel

- Check the **Use structured content** checkbox to automatically map tab field values to values in the VDF payload in the selected atom entry. VDF is the Viz Data Format, an ATOM-feed specification. Only check this option if the feed URI is known to be a valid VDF-formatted feed.
- Check the **Auto refresh on read** checkbox to automatically refresh all tab field values when the template is read.

Playout Buttons



Control buttons for page playout are located above the page editor.

 **Note:** Only pages (data instances of templates) can be taken on air.

The **Take** button takes the page that is currently displayed in the local preview On Air, if the client is set in on-air mode.

The **Continue** button continues the animation in the page currently loaded if it contains any stop points.

The **Take Out** button takes out the page that is in the preview window. If **Take Out** is clicked when a page that is not on-air is in the preview window, any element that is in the same layer as the page currently displayed in the preview, is taken out.

The **Take + Read Next** button takes the page currently read, and reads the next one in the page list.

The **Cue** button cues the page specified as parameter or the selected page if the parameter is empty.

The **Cue append** button cues the video after the currently playing video. This button is only enabled for video elements.

The **Pause** button pauses the video. This button is only enabled for video elements.

Save Buttons



When working with a template in the page editor, the **Save** and **Save as** buttons are available. Clicking **Save**, saves all modifications to the template: feed linking information, feed browser layout, and default tab-field values. Clicking the **Save as** button creates a new page based on the template. The page's callup code is by default set to be the highest callup code (number) in the page list + one (1). The page's callup code is shown in the [Callup Code](#) field. Press **ENTER** to save the new page.

 **Note:** The **Save** button is currently disabled for Viz Pilot elements read in Viz Trio. The **Save as** button works for Viz Trio and Viz Pilot templates.

When working with a page in the page editor, the **Save** and **Save as** buttons are available. Clicking the **Save** page button saves all modifications to the page without any changes to the callup code. Clicking the **Save as** button creates a new page. The page's callup code is by default set as the highest callup code (number) in the page list + one (1). The page's callup code is shown in the callup code field. Press **ENTER** to save the new page.

Script Buttons



The **Execute script** button executes a template's script (see [Script](#) macro) and perform a syntax check. The show script is not executed unless the template script calls procedures or functions within the show script. For the button to work, the script that is to be executed must have an *OnUserClick* function wrapped around it.

```
Sub OnUserClick () ... End Sub
```

The **Edit script** button opens the [Script Directory](#). The editor by default opens the assigned script.

Refresh Button



For pages with linked tab-fields ([Field Linking](#)), the user may want to press this button to refresh the data of all tab-fields in the page. A typical use case is for weather or financial data.

The button is only enabled when at least one tab-field is linked to a feed which provides self links. While refreshing, the icon on the button changes to a cross. Press the button again to cancel the refresh operation.

On a template level, data is automatically refreshed upon read.

The following script commands are related to the refresh procedure:

- `page:refresh_linked_data` : Refreshes the linked data in a blocking way (useful for scripts that manipulate the refreshed data afterwards).
- `page:refresh_linked_data_async` : Refreshes the linked data in a non-blocking way so that the GUI stays responsive.
- `page:cancel_refresh_linked_data` : Cancels a refresh started with `page:refresh_linked_data_async`.

Variants

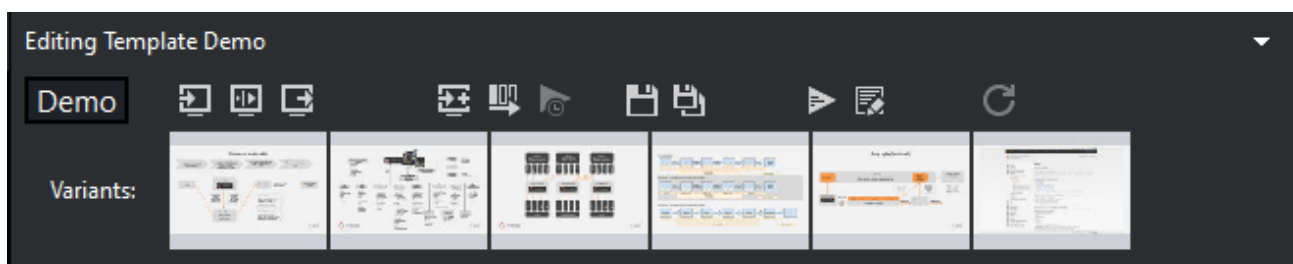
Template variants can be switched directly from the [Template List](#) or from the page editor. As with concepts, variants can also be switched using a macro command in a user-defined macro or a script.

Create a Page

1. Double-click a template from the [Template List](#) to open the template in the page editor.
2. Edit and save it as a page with a new [Callup Code](#).

Info: It is possible to create Viz Trio pages from Viz Pilot templates.

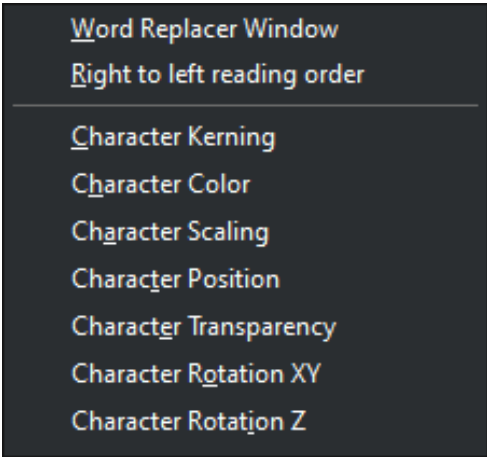
Select a Variant from the Page Editor



1. Open the template or page in the Page Editor.
2. Click on the thumbnail image to select a variant.

5.16.2 Text

Text Editor Context Menu



The text editor is the main editor for most users. Here, it's possible to do a number of operations if the scene's text editing properties are exposed, such as setting text color or adjusting the position and scale of the text. In order to enable character formatting, the scene's Control Text plugin's **Use formatted text property** must be enabled (on). This is done by the Viz Artist designer. The following can be formatted if enabled:

Mode	Command	Procedures
Color	text:set_color_mode	Editing Text Color
Alpha	text:set_alpha_mode	Editing Text Transparency (Alpha)
Kerning	text:set_kerning_mode	Editing Text Kerning
Position	text:set_position_mode	Editing Text Position
Replace character	text:show:replace_list	Replacing a Word
Rotation X, Y and Z	text:set_rotate_xy_mode text:set_rotate_z_mode	Editing Text Rotation
Scaling	text:set_scale_mode	Editing Text Scaling

The desired text manipulation mode can be assigned to a shortcut key or used in scripts. See the [User Interface](#) section for how to assign shortcut keys. The active mode is indicated in the lower right corner of the Page Editor's window. The default mode is Color editing. For more information on text commands, see [Macro Commands and Events](#).



Tip: Text commands can be used in scripts or as assigned keyboard shortcuts to switch between text manipulation modes.

In some situations, if text effects are used in graphics there might be conflicts with editing of parameters such as scaling, alpha and position. For example: if the text effect controls the scaling of text, it's not possible to change the scaling of the fonts from the text editor. The text effect's control over that parameter, override any changes made in Viz Trio. Text effects are Viz Artist plugins such as TFX Color, Scale and Rotation that are used by scene designers.

Word Replacement

The Word Replacement function allows the user to substitute a typed character with a word from a list of predefined substitution options.

This list becomes available in the text editor when you press an assigned shortcut key for the `text:show_replace_list` command, see the [Keyboard Shortcuts and Macros](#) section. The editor matches a typed character with the list of key characters that are configured.

This section covers the following topics:

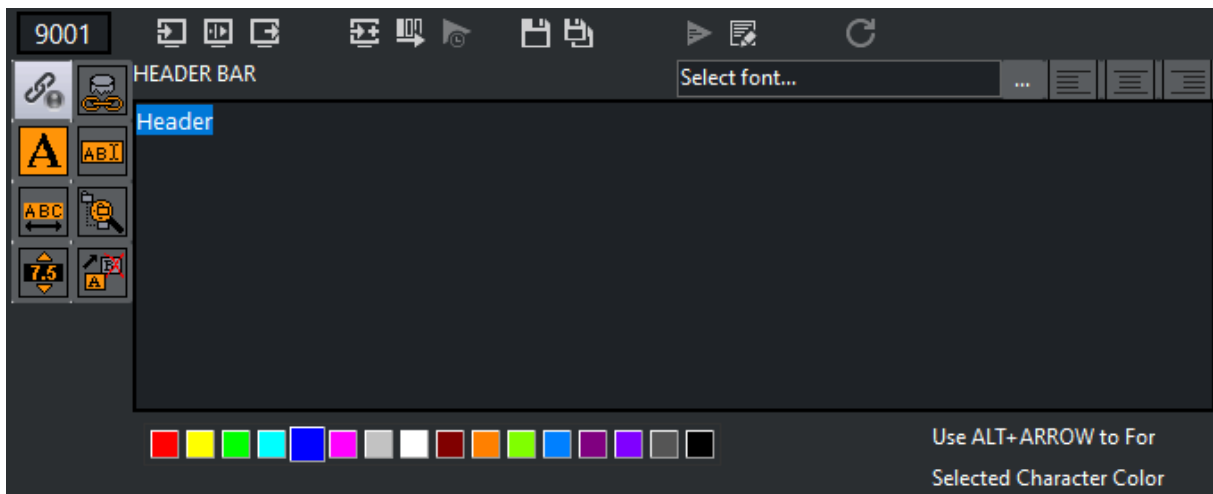
- [Edit a Text Item](#)
- [Edit Text Color](#)
- [Edit Text Kerning](#)
- [Edit Text Position](#)
- [Edit Text Rotation](#)
- [Edit Text Scaling](#)
- [Edit Text Transparency \(Alpha\)](#)
- [Replacing a Word](#)

Edit a Text Item



1. Open the appropriate template or page, and select a text element in the Tab Fields or Preview window.
2. To edit the next item, press the **Tab** key or select the item from the Tab Fields list or right-click the graphical elements in the Preview window.
3. To perform character-specific changes, right-click to open the text editor's context menu, and select one of the options.

Edit Text Color

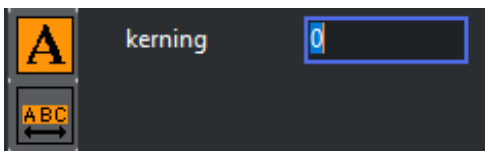


1. Open the appropriate template or page, and select a text element in the Tab Fields or Preview window.
2. Right-click the text editor, and from the [Text editor context menu](#) select **Character Color**.
3. To change the color of a character or selected characters, click a color in the color palette, or hold the **ALT** key down while using the right and left arrow keys to switch between the available colors.

✓ **Tip:** Color can also be set using the command `text:set_color_mode` when enabled by the designer.

See [Show Properties](#) on how to customize the color palette.

Edit Text Kerning



1. Right-click the text editor and select **Character Kerning** from the [Text editor context menu](#).
2. Select one or more characters and hold down the **ALT** key while using the right and left arrow keys to change the kerning, or
3. Click the **Kerning for whole text** button.
4. Click the **Text editing mode** button to switch back to the text editor.

✓ **Tip:** Character kerning can also be set using the command `text:set_kerning_mode` when enabled by the designer.

⚠ **Note:** The Control Text plugin Expose kerning property enables the Page Editor's kerning button.

Edit Text Position

1. Right-click the text editor, and from the [Text editor context menu](#) select **Character Position**.

2. Select characters.
3. Hold the **ALT** key down while using the arrow keys.

✓ **Tip:** The position can also be set using the command `text:set_position_mode` when enabled by the designer.

Edit Text Rotation

1. Right-click the text editor, and from the [Text editor context menu](#) select **Character Rotation XY *or Character Rotation Y***.
2. Select a character or multiple characters.
3. Hold the **ALT** key down while using the arrow keys.

✓ **Tip:** Rotation can also be set using the commands `text:set_rotate_xy_mode` or `text:set_rotate_z_mode` when enabled by the designer.

Edit Text Scaling

1. Right-click the text editor, and from the [Text editor context menu](#) select **Character Scaling**.
2. Select a character or multiple characters.
3. Hold the **ALT** key down while using the arrow keys.

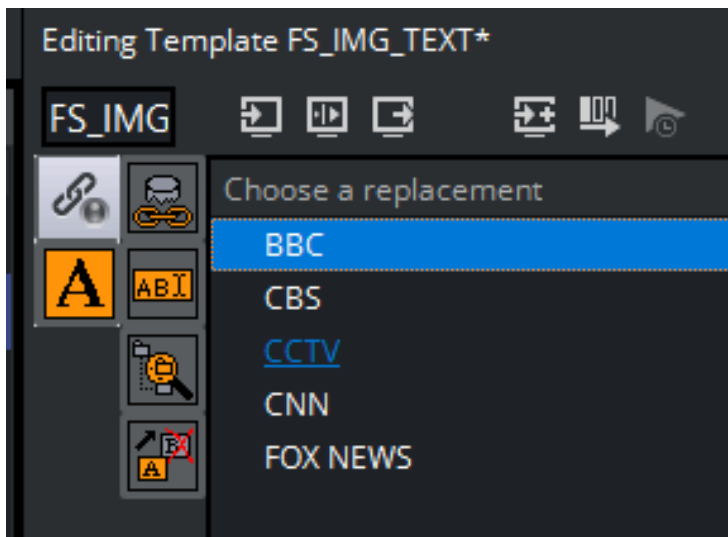
✓ **Tip:** Scaling can be also set using the command `text:set_scale_mode` when enabled by the designer.

Edit Text Transparency (Alpha)

1. Right-click the text editor and select **Character Transparency** from the [Text editor context menu](#).
2. Select characters.
3. Hold the **ALT** key down while using the arrow keys.

✓ **Tip:** Transparency can be also set using the command `text:set_alpha_mode` when enabled by the designer.

Replacing a Word



1. **Type** a character in the text editor. Make sure the cursor is placed after the character that is replaced.
2. Click the word replacement button or right-click the text editor and select **Word Replacer Window**.
3. Select a replacement word from the list and click **OK**, or press the **Enter** key to return to the text editor.
4. To close the word replacement window without changes click **Cancel**, or press the **ESC** key to quit the operation and return to the text editor.

5.16.3 Database Linking

In order to automate the way a page is filled with content, a page's tab field can be linked at an external data source such as a database or Microsoft Excel spreadsheet (see [Show Properties](#) and [To create a new database connection](#)).

There are two variants of the database link editor. The type of editor depends on the type of control plugins used in the scene. Table plugins such as Control List and Control Chart have their own table editor. Click the Database link button below:



Database link: Opens one of the database link editors so you can link data to a page.



Save to database: Saves the updated data back to the data source.

This section covers the following topics:

- [Database Link Editor Common Properties](#)
- [Database Link Editor for Scalar Linking](#)
- [Database Link Editor for Table Linking](#)
- [Microsoft Excel and Unicode Support](#)

Database Link Editor Common Properties

The Data Link editors common properties are:

- **Link to database:** Enables the database link.

- **Database connection:** Sets the connection to a selected database link defined in the [Show Properties](#) Databases settings.
- **Table:** Sets the database table for the lookup.
- **Refresh:** Refreshes the dataset.
- **Custom SQL:** Displays by default the resulting SQL query to get the value (scalar or table) from the database. The query can be overridden to perform enhanced queries. This is mostly useful for table properties, for example to restrict the number of rows or apply a different sorting for the rows.
- **Apply:** Applies the Custom SQL query.

Database Link Editor for Scalar Linking

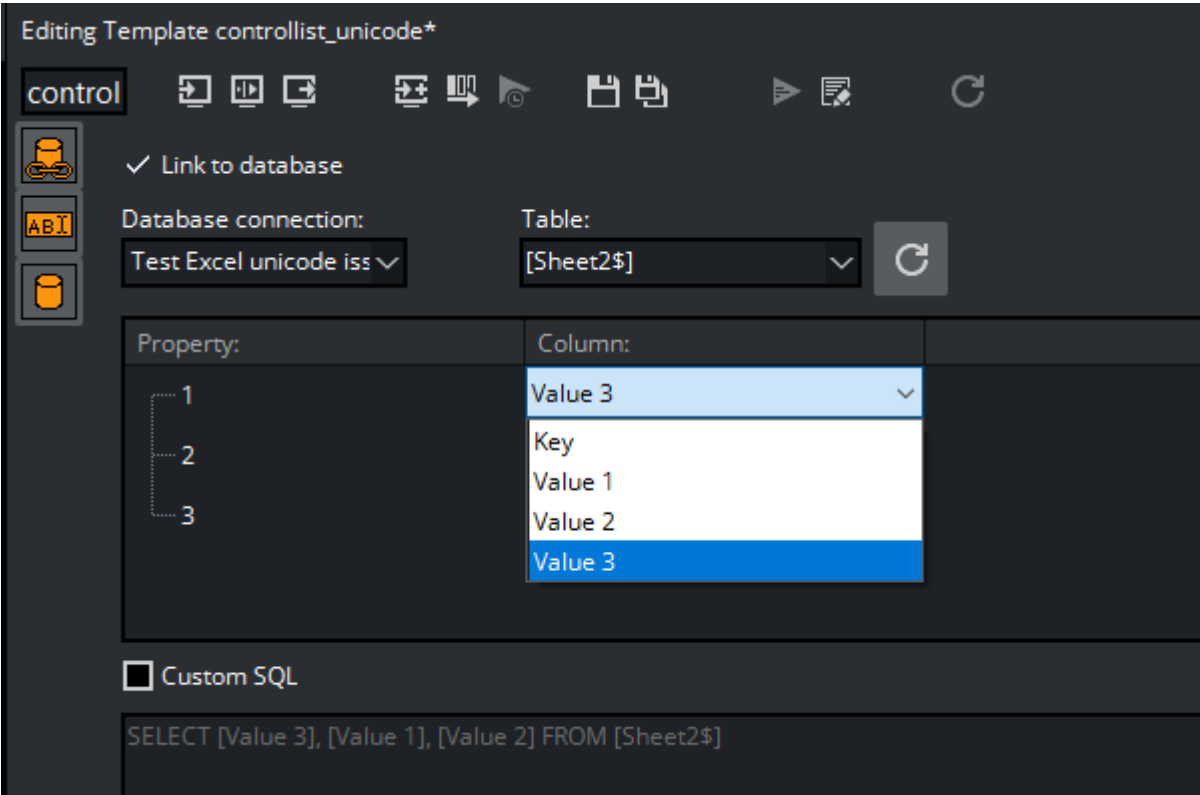
The database link editor used for text properties (scalar linking) has the following properties:

- **Select column:** Sets the Microsoft Excel spreadsheet column, from which the text value is selected.
- **Key column has value:** Specifies the key value to match with the lookup column, which is specified in the Databases settings (see [Show Properties](#)). This mechanism selects the row of the value to use. To restrict which rows from a Microsoft Excel spreadsheet are inserted, a custom SQL query can be specified in the text box at the bottom of the database link editor (below is a Microsoft Excel spreadsheet example).

```
SELECT Value, Color FROM [Sheet1$] where Key > 2
```

Note: Setting both Column and Key determines both column and row of the scalar value for the text property.

Database Link Editor for Table Linking



The database link editor used for table linking has the following properties:

- **Property:** The scenes table property that is linked with the column of the data source.
- **Column:** The data source column that is linked with the property of the scene. Press F2 in the column to change the source column.

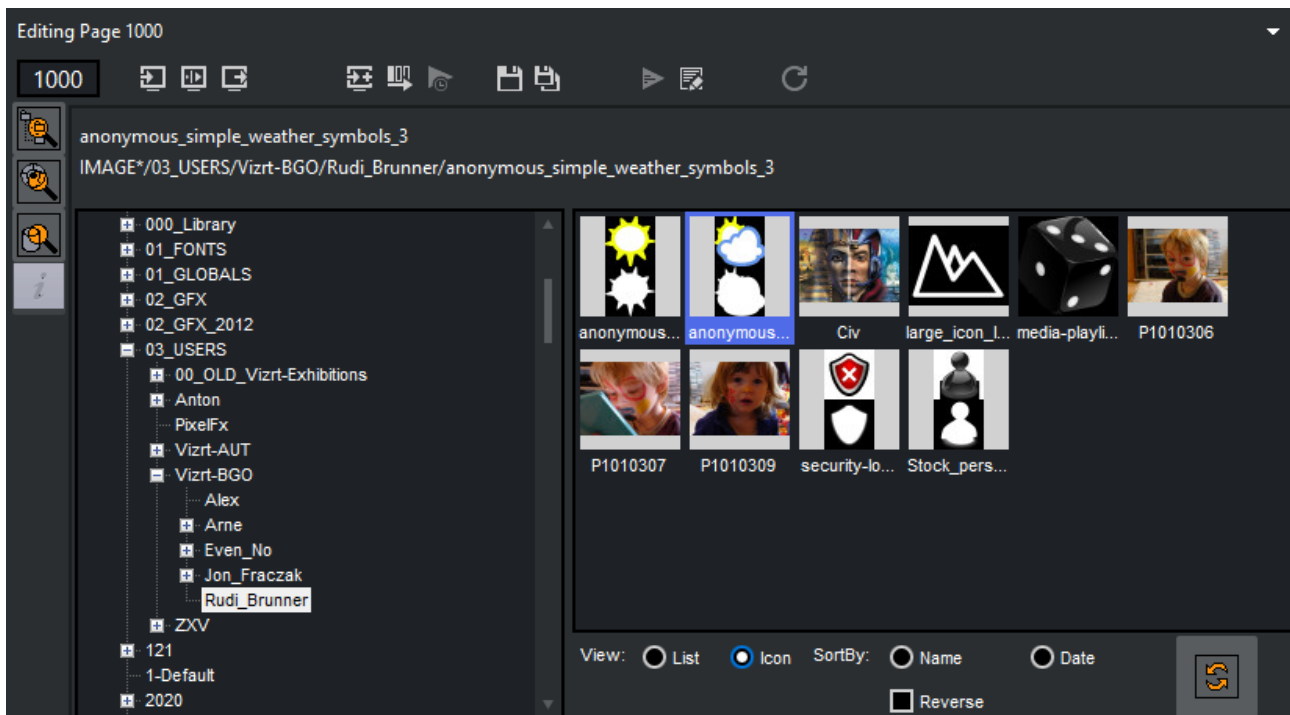
Note: Mapping a tab-field property with a data source immediately updates the graphics.

Microsoft Excel and Unicode Support

Viz Trio supports the use of Microsoft Excel spreadsheets to provide templates with data.

Info: Unicode characters can be used in Microsoft Excel table names and in cell values of a table, but not in column names.

5.16.4 Image Property Editor



The Image Property Editor appears within the Page Editor (as shown above) when selecting an image or item in a page's tab field. When the template contains an image you can load an image from file, Graphic Hub, Viz Object Store or a Vizrt MAM system, if the properties are exposed.



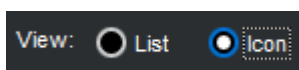
Browse File: Opens a standard Windows file browser.



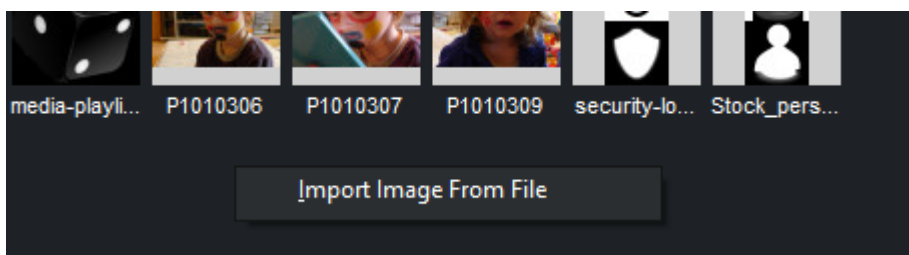
Browse Viz Artist: Opens a browser in the Page Editor for browsing images on the Viz Engine. It's possible to sort by name and date. For convenience both sorting actions can be reversed and refreshed.



Browse: Opens the [Search Media](#) pane for browsing images, video clips and person information. Note that additional Browse buttons appear if other image repositories are used.



List and icon view: Allows users to view images imported into the image editor as either small thumbnails or in a list displaying only the name of the file. Items are arranged alphabetically in both cases.



Upload multiple images at a time: Right click in the image property editor and select **Import Image From File**. Hold either the Shift or Ctrl key and left click multiple images to import or Ctrl + A to select and import all images in a folder.

Click Open and the images appear in the editor ready to be imported into a page.

5.16.5 Transformation Properties

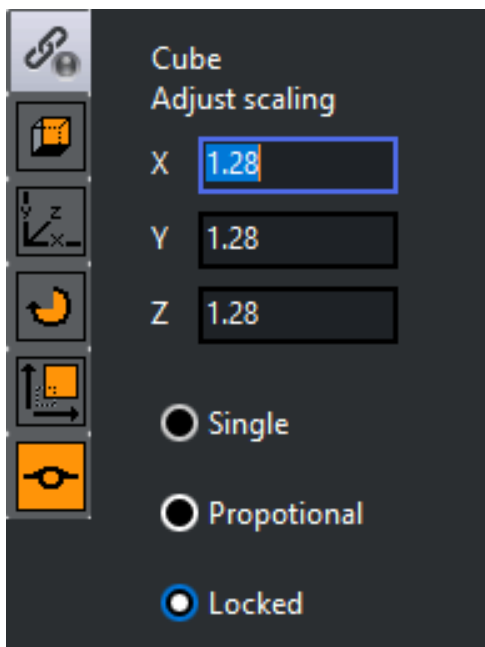
It is possible for scene designers to expose transformation properties for a tab field, as well as an active or inactive toggle. The following properties can be edited if enabled. They are displayed as buttons on the left side of the editor, from top down:

- Position
- Rotation
- Scaling
- Active



Clicking the Position, Rotation or Scale buttons opens an editor that can be used to set transformation properties.

Setting Transformation Properties



1. Enter the new values with the keyboard, or
2. Left-click the value field and drag in horizontal directions. Move between the fields using the **CTRL + TAB** keys.
3. To return to the previous selection mode, click the top button.

Note: The selection mode icon varies according to the type of object exposed, such as image, text and so on. The screenshot above shows a Geom.

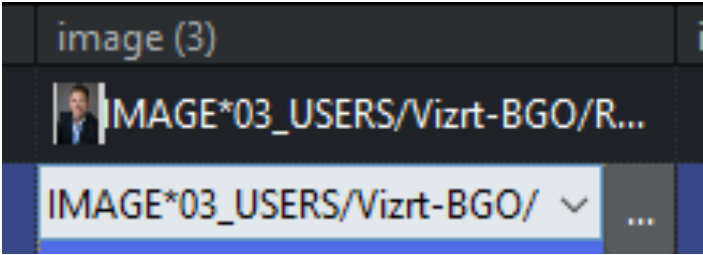
5.16.6 Tables

Editing Page 1001

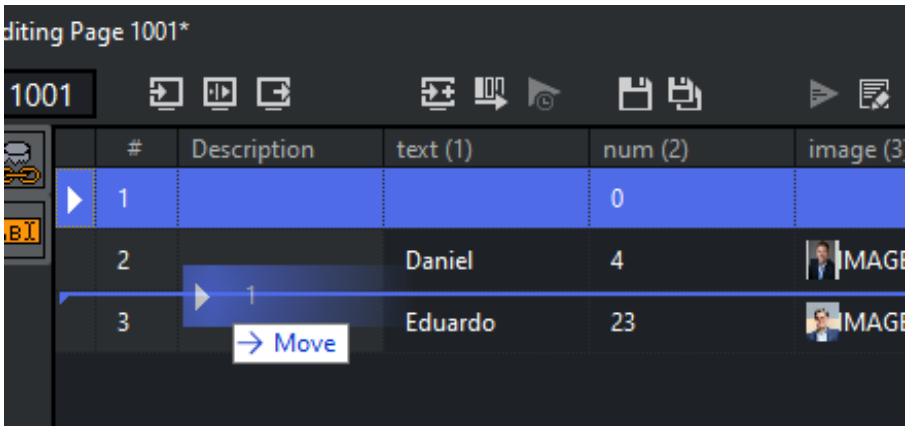
#	Description	text (1)	num (2)	image (3)	image (Texture Scaling) (3.image_scaling)
1		Daniel	4	IMAGE*03_USERS/Vizrt-BGO/R...	1.6 1.6
2		Eduardo	23	IMAGE*03_USERS/Vizrt-BGO/R...	1.000000 1.000000

If a scene contains a Control List plugin, a table editor is shown for that tab field when the scene is edited in Viz Trio. The cells contain different editors depending on the data type they display. Changes made in the table editor are instantly updated in the preview window. In addition to text fields, the table editor can show the following special editors in the table cells:

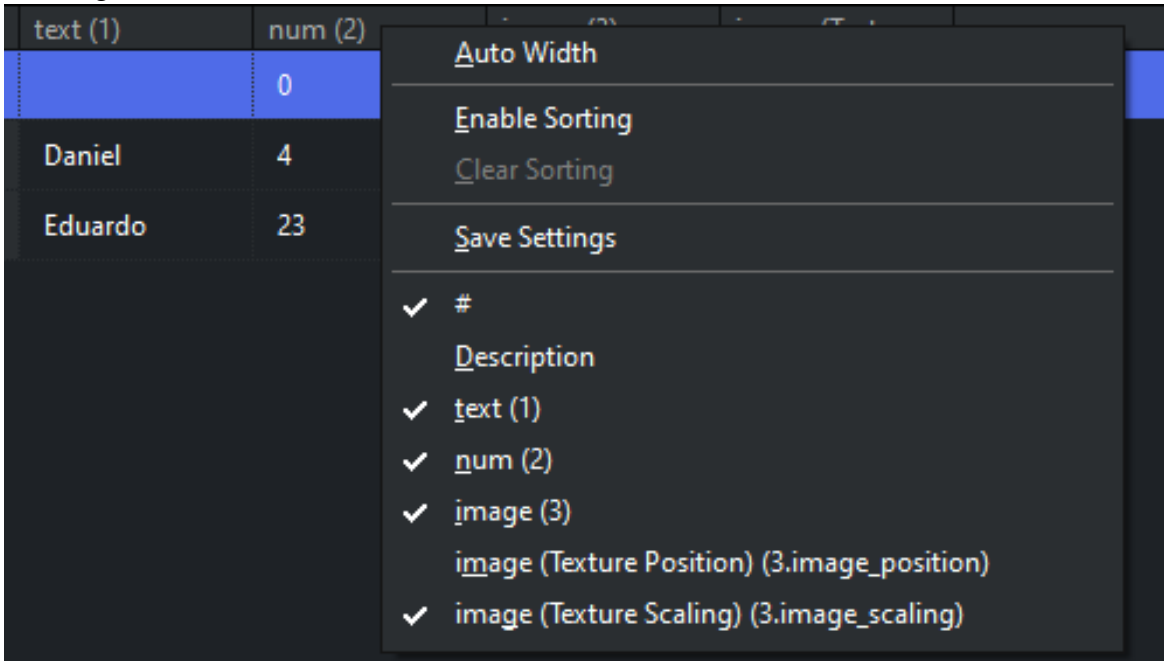
- **Image Cells:** The drop-list shows all images currently available for a specific column. This provides a quick way of choosing images when many cells are to show the same image. The browse button to the right opens the image pool.



- **Number Cells:** Scenes with exposed numbers, with and without decimals, feature a small text editor with spin button functionality. Click the spin buttons or use arrow keys, or enter numbers directly on the keyboard to change value.
- **Checkbox Cells:** For Boolean values, such as hide and show, a checkbox is displayed.
- **Table Keyboard Navigation:** To move between the cells, use the arrow keys on the keyboard.
- **Sorting:** The table can be sorted manually by clicking on the header bar of the column to sort by. Rows can be rearranged manually by drag and drop. Click on the row once and hold the mouse button to drag and drop it to its new location.



- **Customizing a Table View:** Right-click on the table header bar, to make a context menu appear. Columns can be hidden or shown by selecting or unselecting the column. The table editor context menu contains the following columns:



- **#:** Shows the row number (#).
- **Description:** Shows the description field. Descriptions can only be edited in Viz Artist by the designer.
- **Auto Width:** When disabled the column width can be set manually.
- **Enable Sorting:** Enables manual sorting of the table.
- **Clear Sorting:** Clears any sorting and take the table back to its initial state.
- **Save Settings:** Saves the column setting for an active template.

Right clicking a row displays this context menu:

- **Insert Row (Ins):** Inserts a single row.
- **Insert N Rows ...:** Opens a small input dialog for multiple row insertion. The number of rows you can add is limited by the scene design.
- **Edit Row:** Edit the row data.

- **Delete Selected Rows (DEL):** Deletes selected rows. For multiple select use **CTRL + Left mouse key** or **SHIFT + Arrow keys** up or down.
- **Select All (CTRL + A):** Selects all rows.
- **Cut (CTRL + X):** Cuts rows out of the table. Press **CTRL + V** to re-add the row(s).
- **Copy (CTRL + C):** Copies rows from the table. Press **CTRL + V** to add the new row(s).
- **Paste (CTRL + V):** Adds the cut or copied rows to the table.

Note: The number of rows that can be inserted or deleted depends on the configuration of the plugin. See the Chart and List plugins in the Viz Artist User Guide.

Populate Tables Using Macro Commands

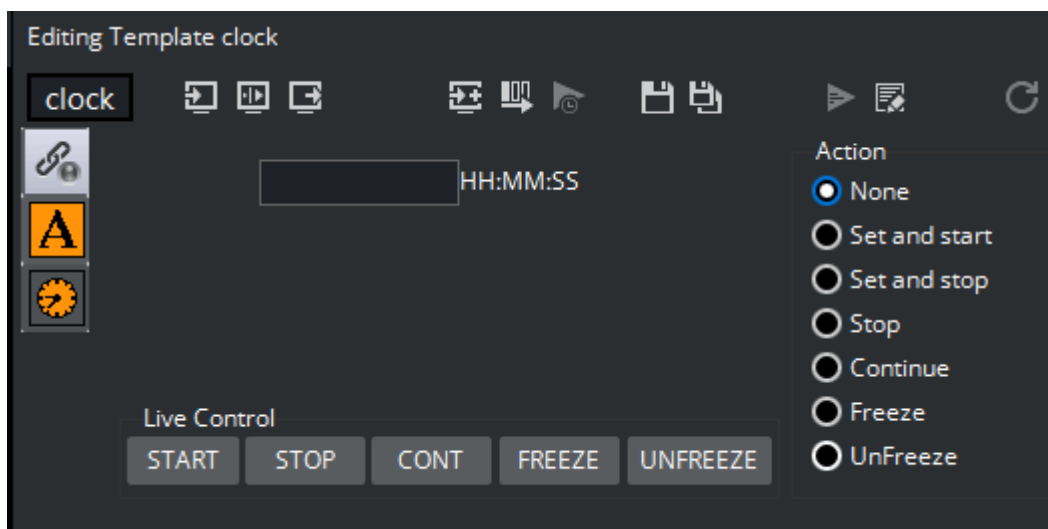
You can use Viz Trio's macro language to populate a table with data through a script, or through an external application that connects via a socket connection. To populate a table, the macro `table:set_cell_value` is used. The macro takes 4 arguments:

- The field identifiers for the **table** (string) and the **column** (string), the table's **row** number (integer: starts at 0), and the **data** to be inserted.
- The example shown below inserts the text string test, in row 4, in the column with id 1, in the table with id 1.

```
table:set_cell_value 1 1 3 test
```

5.16.7 Clock

If the template contains a clock object that is made editable with the Control Clock plugin, a clock editor opens when tabbing to the clock's tab field.



The clock editor has two main functions:

1. **Normal editor:** Specify what should happen when the page is taken on-air.
2. **Live Control:** With a page on-air, a clocks counting can be started, stopped, continued, frozen, or unfrozen.

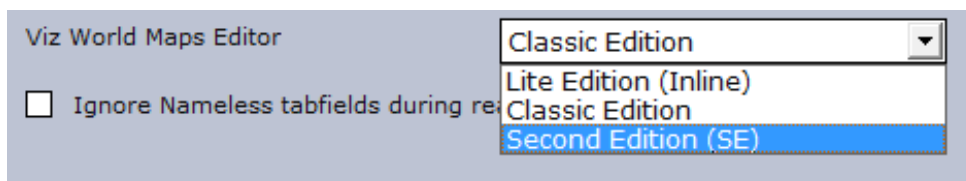
The clock editor's properties are:

- **Start At:** Sets the starting time for the clock.
- **Stop At:** If enabled, a stop time for the clock can be set in the template.
- **Up/Down:** If enabled, the direction the clock should run can be chosen in the template.
- **Action:** Select what the clock is to do when the page is taken on-air. Either:
 - **None:** The time is just set and the clock must be started with the **Live Control Start** button.
 - **Set and start:** The time is set and the clock is started.
 - **Set and stop:** The time is set and the clock is stopped.
 - **Stop:** The clock is stopped when the page is taken on-air. This can be used to create a shortcut that stops the clock. Just create a page that stops the clock and link it to a key shortcut.
 - **Continue:** Similar use as with stop.
 - **Freeze:** The clock is frozen when the page is taken on-air.
 - **Unfreeze:** The clock is unfrozen when the page is taken on-air.

5.16.8 Maps

Viz World Client is available to Viz Trio in both a simplified and a full version. The simplified version is enabled in the configuration window, see [General](#) settings under the [User Interface](#) section. The full version launches the selected Map client, either in Classic Edition or the new Second Edition (SE).

Select the preferred Map client in **File > Configuration > General > Viz World Maps Editor:**



Note: Make sure the Viz Maps Client is installed on the Viz Trio client and make sure the Viz Engines (both local preview and On-Air render Engines) connect to the same Viz World Server. To configure the Viz World Server address in *Viz Engine* go to **Config > Maps** on the Viz Engine server.

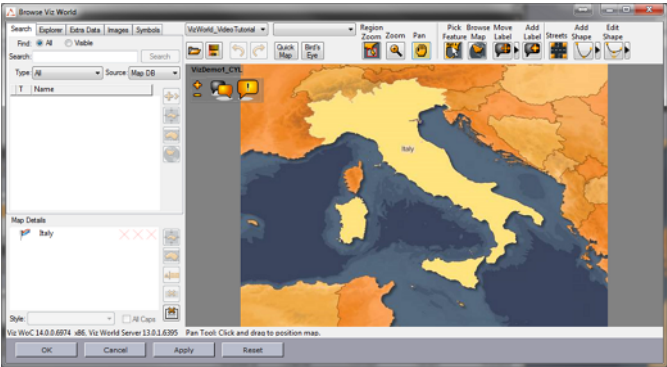


Edit Map: If the map's tab-field property has more than one property exposed, clicking this button opens the corresponding editors (for example text and text kerning).



Browse Viz World: Opens the full version of the Viz World Client. To use the full version, disable the small version. You can also click the **Select Map** button to launch the selected Map client.

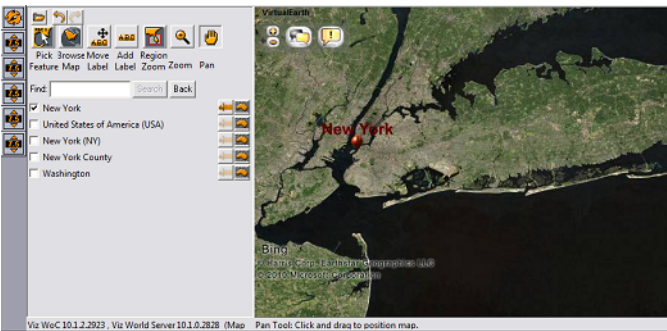
Maps Classic Client



Maps Second Edition Client



Maps Lite Edition (inline)



Control World Plugin Interface

In addition to the Viz World Client editor, Viz World also provides an easy to use control interface when using the Control World plugin. The control is a replacement for the Control Map plugin with much more options and on-the-fly feedback from Viz.

Note: The two controls are not compatible. The new control does not work with Viz Pilot, and is only a part of Viz 3.3 and later.

The control can expose different fields based on the container it resides on. When tabbing to the control (in a navigator scene), the camera jumps to the map location and all feedback (exact camera position) is immediate.

- **Longitude Offset:** Sets the longitude offset.
- **Latitude Offset:** Sets the latitude offset.
- **Distance:** Sets the distance.
- **Distance Offset (%):** Sets the distance offset in percentage.
- **Pan:** Sets the pan.
- **Tilt:** Sets the tilt.
- **Hop Duration:** Sets the hop duration.
- **Terrain Height Scale Mode:** Sets the scale mode of the terrain height.
- **Terrain Height Scale:** Sets the scale of the terrain height.
- **Select Map (button):** Opens the Viz World Editor. If the control is configured to use the simple Viz World Editor, Viz Trio must be configured to do the same.

See Also

- [General](#) settings to switch to simple Viz World Editor.
- [Viz Artist User Guide](#) for information on Control World.
- *Viz World Client and Server User Guide*
- [Field Linking and Feed Browsing in Viz Trio](#)
- [Creating a New Database Connection](#)
- The [dblink](#) macro commands for scripting.
- [Scripting](#)
- [Setting the Transition Effects Path](#)

- [Adding Transition Effects Using the Page Editor](#)

5.17 Create New Scroll

Creating a scroll allows the user in an easy way to create credit lists or any other scroll, either vertical, horizontal, with ease points and so on. When creating a scroll from scratch, Viz Trio automatically creates a scroll scene on Viz which the user can redesign.

Note: New scrolls must be created using the scroll function.

This section covers the following topics:

- [Create New Scroll Editor](#)
- [Scroll Elements Editor](#)
- [Scroll Configuration](#)
- [Scroll Live Controls](#)
- [Scroll Control](#)
- [Element Spacing](#)
- [Easepoint Editor](#)
- [Working with Scrolls](#)

5.17.1 Create New Scroll Editor

Create New Scroll

Create New Scroll Template:

Template Name: CreditsRoll

Scene Directory: Feature_Examples ...

SCENE*Feature_Examples/CreditsRoll

Template description: Viz Trio test scroll

Scroll Area Width: 500

Scroll Area Height: 375

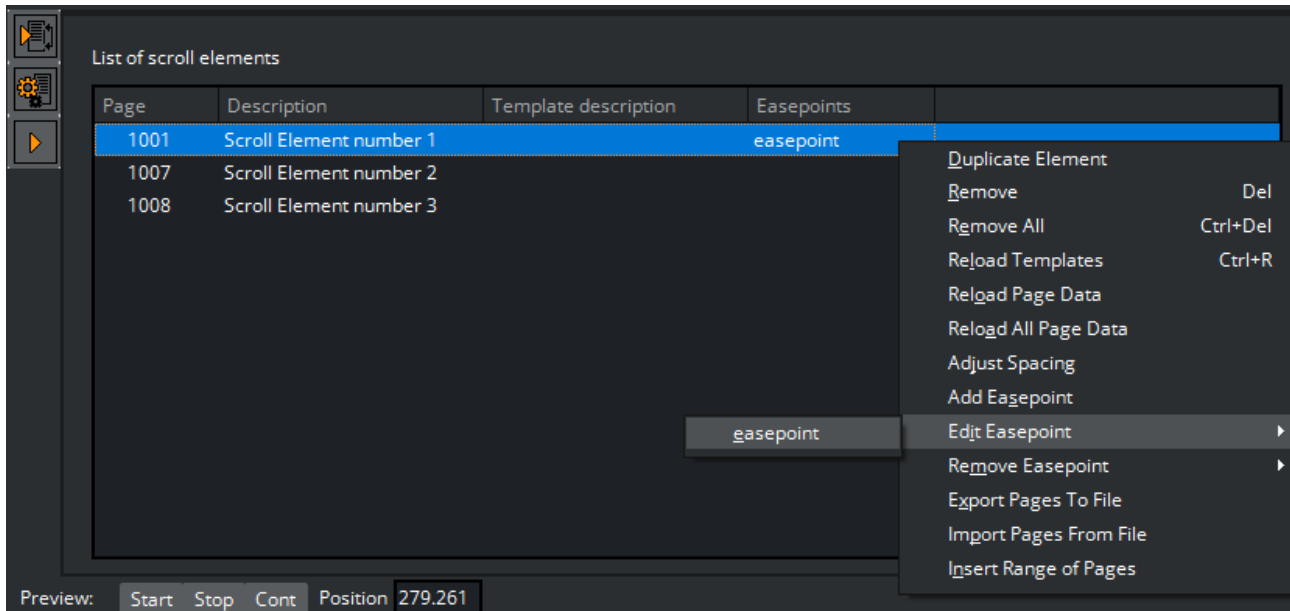
☒ Show Scroll Area

Save Template

- **Template Name:** Enter a name for the template.
- **Scene Directory:** Specify the path to a scene in which to save the template, or select a show by pressing the Browse button.
- **Template description:** Enter a description for the template.
- **Scroll Area Width:** Sets the width of the scroll area.
- **Scroll Area Height:** Sets the height of the scroll area.
- **Show Scroll Area:** Enable this option to show the scroll area in the preview window.

- **Save Template:** Saves the template in the specified scene directory.
The Scroll Elements editor opens automatically when saving or opening an already saved scroll template. A scroll is created by compiling a set of templates or pages.

5.17.2 Scroll Elements Editor



Right-click on any element in the scroll list and a menu appears where actions can be performed and parameter settings defined.

The Scroll list context menu options are:

- **Duplicate Element:** Makes a copy of the selected scroll element and adds it to the scroll list.
- **Remove:** Removes the selected element from the scroll list.
- **Remove All:** Removes all elements from the scroll list.
- **Reload Templates:** Reloads the original templates.
- **Reload Page Data:** Reloads the original page data of the selected scroll element.
- **Reload All Page Data:** Reloads the original page data of all the elements in the scroll list.
- **Adjust Spacing:** Opens the [Element Spacing](#) editor which allows the user to adjust the distance before and/or after the selected scroll element.
- **Add Easepoint:** Opens the [Easepoint Editor](#) which allows the user to add an easepoint to the selected scroll element.
- **Edit Easepoint:** Opens the create new scroll editor to adjust the easepoint settings of the selected scroll element.
- **Remove Easepoint:** Removes the ease point from the selected scroll element.
- **Export Pages To File:** Exports all scroll elements in the currently selected scroll to an XML-file.
- **Import Pages From File:** Imports a set of scroll elements from an XML-file to the scroll. Note that this option erases the scroll elements currently in the scroll.
- **Insert Range of Pages:** This option allows the user to insert one or more pages in the scroll. Specify the range of pages to be inserted in the appearing dialog.

5.17.3 Scroll Configuration



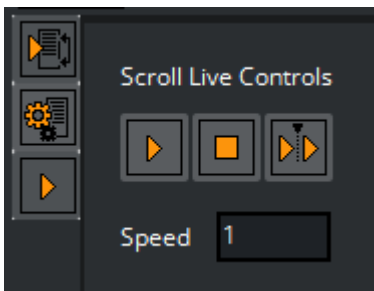
Click the **Scroll Config** button to open the scroll main property page.

- **Direction:** Sets the direction of the scroll.
- **Spacing:** Adjusts the distance between the pages in the scroll.
- **Alignment:** Aligns the elements of the scroller to each other. The bounding box of the elements is used for aligning the objects.
- **Speed:** Sets the speed of the scroll.
- **Total Time:** Shows the total scroll time in seconds.
- **Start Offset:** Set an offset position from where the scroll should start.
- **Loop:** If enabled, the scroll elements are looped.
- **Auto-start on take:** If enabled, the scroll automatically starts on take.
- **Lock Total Time:** Lock the total time the scroll is running. This option distributes the time evenly for all pages. Pages with more or less content are not taken into account. See the [Scroll Elements Editor](#) section for how to add and edit ease points.

5.17.4 Scroll Live Controls



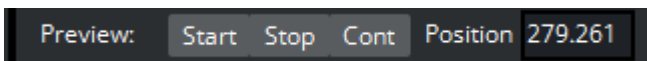
Clicking the Scroll Live Controls button, opens the Scroll Live Controls sub page, which controls the start, stop, and continue actions of a scroll on-air.



To activate the Scroll Live Controls, click the **Take** button in the Scroll Editor. When active, the Live Control buttons are green and can be used as follows:

- **Start:** Takes and starts the scroll that is currently displayed in the local preview on-air, assuming that the client is set in on-air mode.
- **Stop:** Stops the scroll.
- **Continue:** Resumes the scroll.
- **Speed:** Adjusts the speed of the scroll while on-air.

5.17.5 Scroll Control



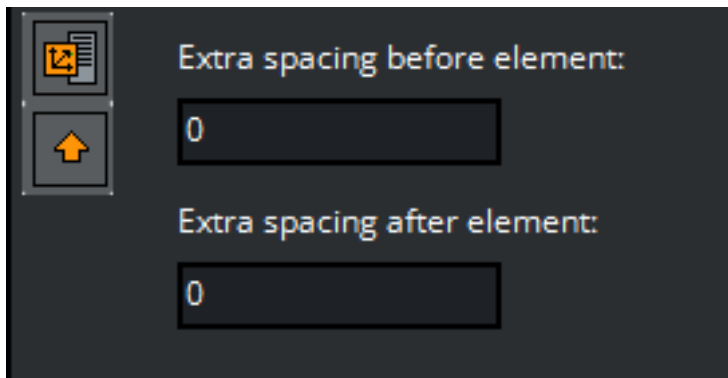
The **Start**, **Stop** and **Continue** buttons in the lower left corner of the Scroll Elements editor enables a preview of the scroll in the local preview renderer window.

- **Start:** Starts the current scroll in the local preview window.
- **Stop:** Stops the scroll.
- **Cont:** Resumes the scroll.
- **Position:** Use the Position slider to manually run a preview of the scroll. Click the value field, keep the button pressed and move the mouse horizontally. The cursor changes to an arrow to indicate that the position value can be changed.

The Scroll Elements editor has two sub-pages, [Scroll Configuration](#) and [Scroll Live Controls](#). Set parameters for the scroll using the [Scroll Configuration](#) editor, and control the scroll while on-air through the [Scroll Live Controls](#) editor.

5.17.6 Element Spacing

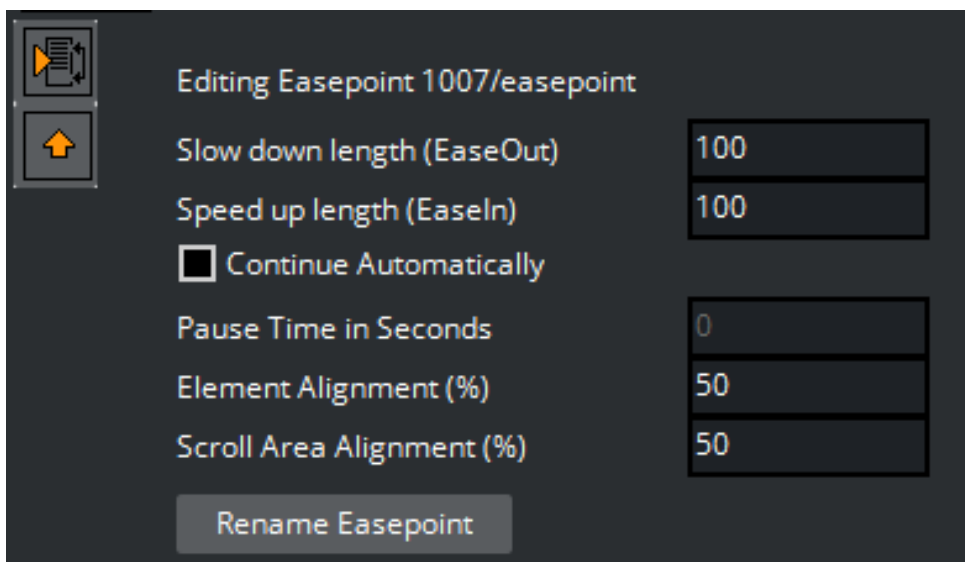
Open the adjust element spacing editor from the context menu of the scroll elements:



- **Extra spacing before element:** Sets the extra distance before the element.
- **Extra spacing after element:** Adjusts the extra distance after the element.

5.17.7 Easepoint Editor

The Easepoint editor allows ease point parameters to be set for a scroll element.



- **Slow down length (EaseOut):** Specifies the slow down length of an ease point.
- **Speed up length (EaseIn):** Sets the speed-up length of an ease point.
- **Continue Automatically:** When enabled, the scroll continues automatically after the specified pause-time.
- **Pause Time in Seconds:** Sets the pause-time in seconds.
- **Element Alignment (%):** Aligns the bounding box of an element relative to the position of an ease point.
 - The default element alignment is 50%, which aligns the center point of the element's bounding box with the ease point.
 - When creating rolls, selecting an element alignment of 0% aligns the bottom of the element's bounding box with the ease point, while 100% adjusts the bounding box's top with the ease point.
 - When creating crawls, an element of 0% aligns the left side of the bounding box with the ease point, while 100% adjusts the bounding box's right side to the ease point.
- **Scroll Area Alignment (%):** Adjusts the position of an ease point relative to the screen's center point. Scroll area alignment is defined as a percentage of the scroll area.

- The default scroll area alignment is 50%, which aligns the ease point with the center point of the screen.
- For rolls a scroll area alignment of 0% aligns the ease point with the scroll area's lower side, while 100% adjusts it to the upper side.
- For crawls, a scroll area alignment of 0% corresponds to the scroll area's left side, while 100% aligns it at the right side.
- When in ease point edit mode, the current position of the ease point is indicated by a line in the local preview window.
- **Rename Easepoint:** Click the button to rename the ease point.

5.17.8 Working with Scrolls

Creating a Scroll

1. Select **Create New Scroll** from the editor's drop-list menu.
2. Enter a **Template Name**.
3. Select a **Scene Directory**.
4. Enter a **Template description**.
5. Set the **Scroll Area Width** and **Height**.
6. Check the **Show Scroll Area** to show the scroll area in the preview window.
7. Drag and drop templates and/or pages into the [Scroll Elements Editor](#).

 **Note:** Since different templates are required for different scroll elements, the graphics designer should create templates for all scroll elements required.

Configuring a Scroll

1. Click the **Scroll Configuration** button to adjust layout and animation speed parameters.
2. Adjust spacing before or after a single scroll element using the [Element Spacing](#) editor.
3. Add and configure easepoints using the [Easepoint Editor](#) editor.
4. Click **Save (CTRL+S)** to save.

Editing a Scroll Element

1. Double-click a scroll element in the [Scroll Elements Editor](#) list to open the editor.
2. Click the **Read Parent Page** (arrow) button to switch back to the [Scroll Elements Editor](#).

Copying an Element in the Scroll Elements List

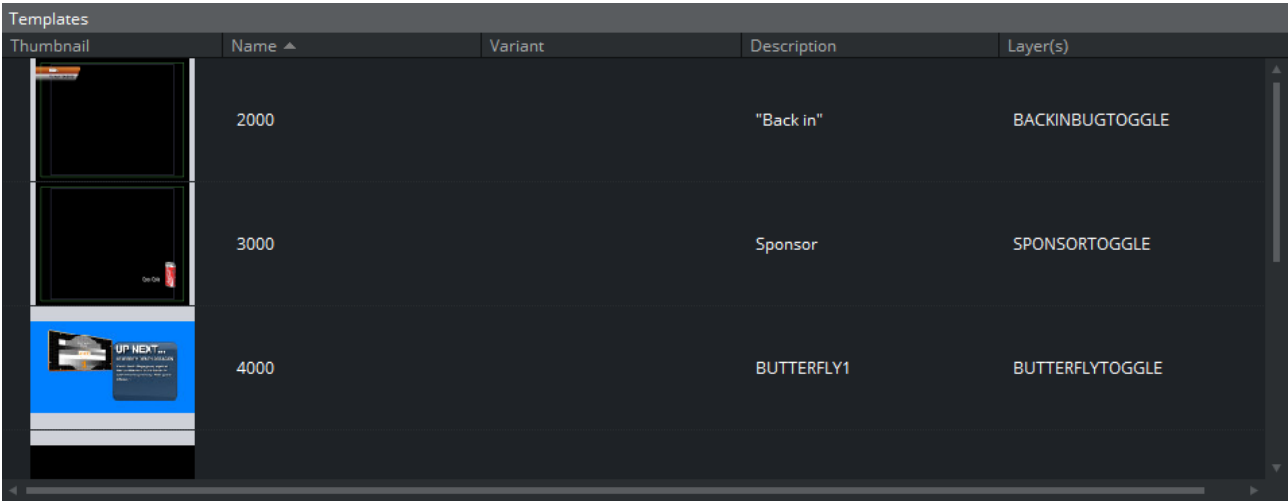
1. Right-click the list of scroll elements, and select **Duplicate Element**, or
2. Drag and drop an element into the scroll list while pressing the **CTRL** button.

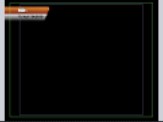
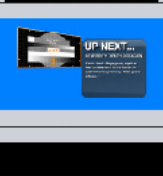
Testing a Scroll

- Click the [Scroll Control](#) buttons in the lower left corner of the scroll, or scrub the **Position** value with the mouse or arrow keys on the keyboard.

5.18 Template List

The Template List contains only templates, which are representations of one or more Viz Artist scenes with default scene data. The Page List contains pages which are instances of templates with specific page related data. Template variants can be switched from the [Page Editor](#). As with concepts, variants can also be switched using a macro command in a user-defined macro or script.



Thumbnail	Name ▲	Variant	Description	Layer(s)
	2000		"Back in"	BACKINBUGTOGGLE
	3000		Sponsor	SPONSORTOGGLE
	4000		BUTTERFLY1	BUTTERFLYTOGGLE

This section contains information on the following topics:

- [Searching for Templates](#)
- [Template Context Menu](#)
- [Columns](#)
- [Combination Templates](#)
- [Creating a Combination Template](#)

5.18.1 Searching for Templates

- Click or set the UI focus to the Templates list.
- Press **Ctrl + F** to show the search toolbar.
- While typing, the selected item jumps to the first description match.

Clock

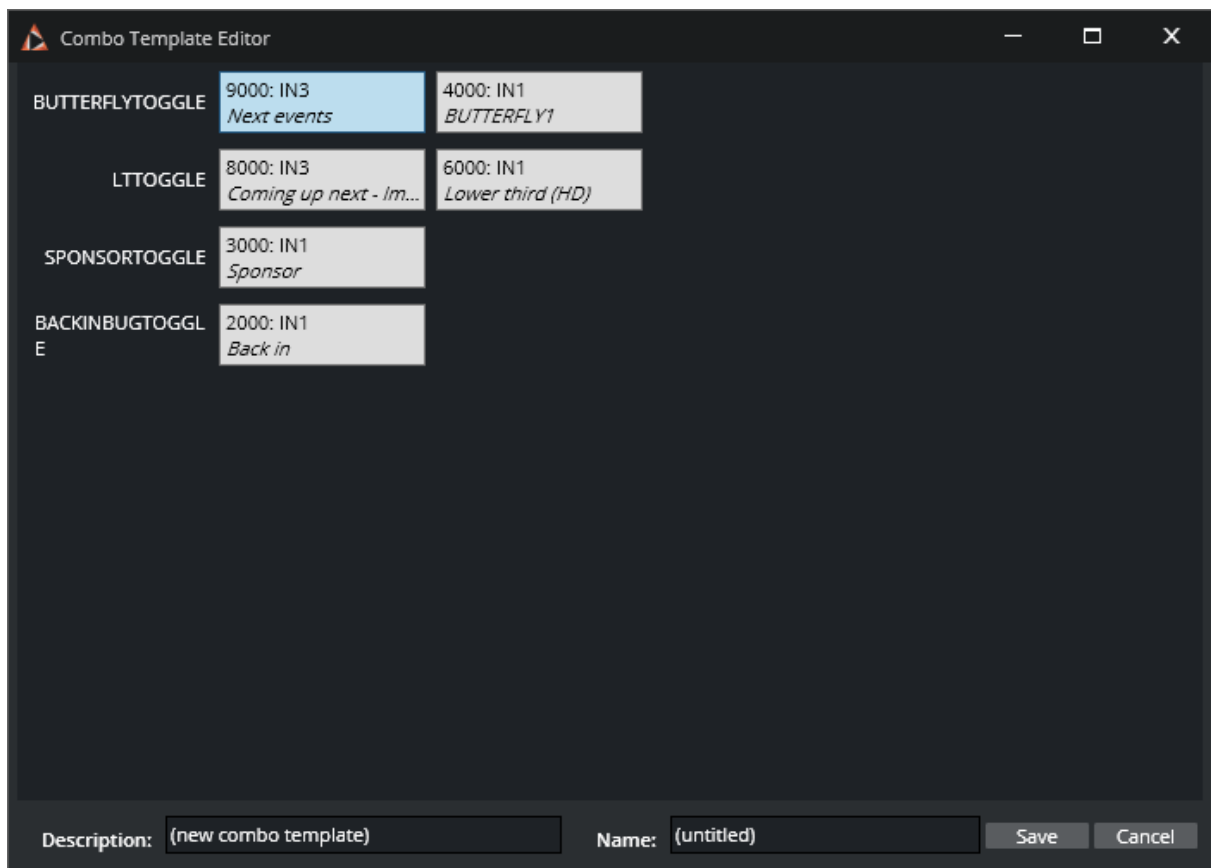
Thumbnail	Name	Variant	Description	Layer(s)
Default 	3000		Lower 3rd: w/Image & 3D Obj	
Default 	1593	OTS_R_image=		OTS_R_image
	default_scratch_onair_backup			
	Clock out	4393	clock=	Clock out

5.18.2 Template Context Menu

Select Thumbnail...
Create ComboTemplate
View Alternatives...
Reimport
Script
Delete

The Template List contains a context menu to create combinations of templates, assigning scripts and so on.

- **Select Thumbnail...:** Select which thumbnail to show for **Combo Templates**.
- **Create ComboTemplate:** Opens the Combo Template Editor, where it is possible to create a combination template that contains all selected templates. This is only relevant for transition logic templates. All templates that are to be merged into [Combination Templates](#) must be in different transition logic layers.

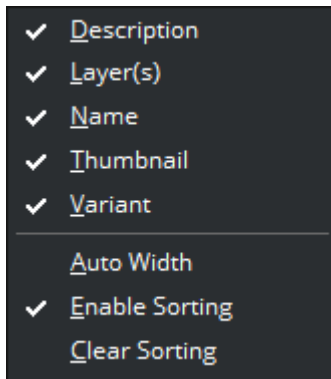


- **View Alternatives...:** Opens a dialog for viewing scene information associated with a template, such as concept, variant, user-defined contexts (platform) and scene path.
- **Reimport:** Updates any scenes that have been changed.
- **Script:** Includes all script-related options:
 - **Edit:** Opens the **script editor** window, where the script assigned to the template can be edited.
 - **Assign New:** Creates a new blank script. The script can be saved to a local or shared repository, or to Media Sequencer.
 - **Assign existing script from file:** Clicking the **Browse...** button opens a file browser on the local computer. This menu also includes a list of the names of the scripts already assigned to the selected template.
 - **Assign existing script from Media Sequencer:** Selects a script stored on the Media Sequencer.
 - **Clear:** Clears the assigned script.

 **Note:** To edit the show script, see the [Scripting](#) section.

- **Delete:** Deletes single or multiple templates from the Template List.

5.18.3 Columns



- **Description:** Shows the contents of the template tab fields.
- **Layer(s):** Shows which layer, [FRONT], [MAIN], [BACK], the scene belongs to, or the Transition Logic layer.
- **Name:** Shows the name of the template.
- **Thumbnail:** Shows the thumbnail image of the scene.
- **Variant:** Shows the variants of the template. For example lower, top and full screen.
- **Auto Width:** Adjusts the column width automatically.
- **Enable Sorting:** Toggles the ability to sort by column on/off.
- **Clear Sorting:** Clears any sorting performed by the user when clicking the column headers.

5.18.4 Combination Templates

A combination template can only be created using transition logic templates. In order to create a combination template, the templates must be in different layers (see **Layer** information in the **Template List** or **Page List** column).

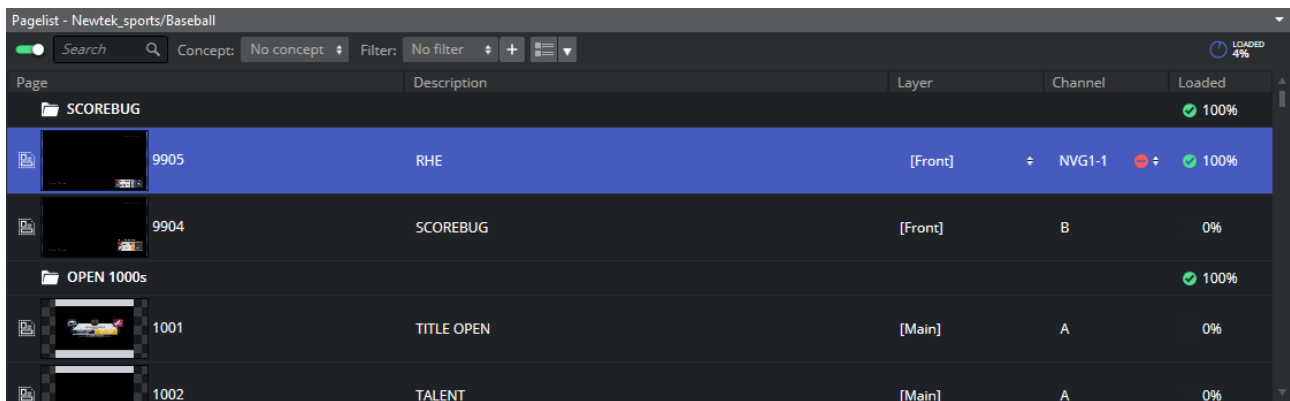
5.18.5 Creating a Combination Template

1. Right-click a transition logic template in the Template List.
2. From the context menu that appears, click **Create ComboTemplate**.
3. In the Combo Template Editor that opens:
 - Select the layers/states that should be activated for this combo template (only one state per layer can be selected).
 - Enter a **Description**.
 - Enter a **Name** (no spaces are permitted in the combo template name).
 - Click **Save**.

 **Note:** Any number of layers can be included into the combo template, but only one state from each layer.

5.19 Page List

The Page List contains a list of accessible pages. Pages are created from templates. Templates are, in turn, normally created by importing scenes created with Viz Artist. You can think of a template as a blueprint while pages are instances or real objects created from templates. You can create as many pages as you like from one template, and each of the pages you create is unique with its own page ID.



A page can be played out from a page list or from a show playlist. The page list can play its pages one by one. A show playlist has the option of creating a carousel, playing the items like a scheduled playlist. It can also be looped.

Pages are edited using the [Page Editor Controls](#) and its available editors. Most page editors are made available to the operator by the scene designer through exposed scene properties within the scene. Others, like a database connection, are made available through the [Show Properties](#) window or by using more advanced features such as [Macro Commands and Events](#).

This chapter contains the following sections:

- [Page Content Filling](#)
- [Page List Context Menu](#)
- [Page List Columns](#)
- [Concepts](#)
- [Templates and Groups](#)
- [Arm and Fire](#)

5.19.1 Page Content Filling

Filling a page with content is a central operation in Viz Trio. In order to create a page, the user must first create a show and import scenes to that show. A template is then converted into a page, and once the page is created it can be populated with media from Viz One.

To create a page and fill it with content, follow this procedure:

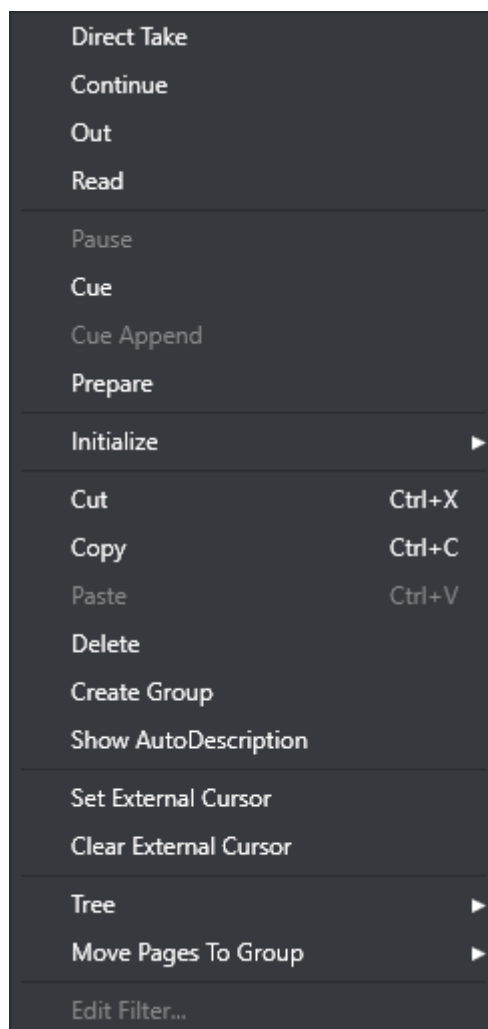
- Create a show under Show Directories from **File > Open Show**. Alternatively, access the Show Directories menu by clicking the Change Show icon.
- From here, you can import a show. If there is no show to import, click the **Create show** icon.
- A field opens in the Shows panel. Give the new show a title and click **OK**.
- From the drop down menu in the Page Editor, select Import Scenes. Select the desired scenes from the Graphic Hub and import them by clicking **Import** at the bottom of the panel.
- The scene now appears in the Template List.

- Now save the template as a page element. Double click the desired template in the Template List.
- In the Page Editor, click **Save As** (the icon with two floppy disks).
- The page number title field is highlighted. Accept the title or edit it and press **Enter**.
The new page now appears in the Page list. Pages can be searched for and filtered by page number from the Page list. From here, fill up the page with, for example, images and video by dragging and dropping media from Viz One into the Page Editor. It's also possible to take the element in a page to air from the Page list, or add pages to different playlists.

 **Note:** Composite elements cannot be displayed from the page list.

5.19.2 Page List Context Menu

Right-clicking an element in the page list activates the context menu. Select a menu item to perform an action or define a setting for a page. Clicking an image icon opens a drop-list of available template variants.

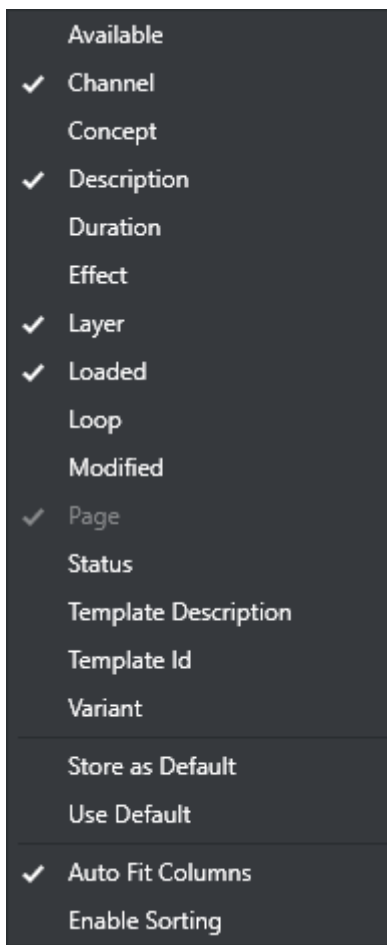


 **Note:** The menu options **Create Combo Template** and **Script** are part of the [Template List](#)'s context menu.

- **Direct Take:** Performs a direct take on the selected page.
- **Continue:** Continues the animation on the selected page, or any that is loaded in the same transition logic layer as the selected page.
- **Out:** Takes out the selected page, or any that is loaded in the same transition logic layer as the selected page.
- **Read:** Performs a read on the selected template or page.
- **Pause:** Pauses the current element.
- **Cue:** Prepares the clip for playout so the *first frame is ready* in the player.
- **Cue Append:** Prepares the clip for playout. The clip *starts automatically* when the current clip ends.
- **Prepare:** Prepares the clip in the pending player without affecting the current clip, so it is ready to be played out. The clip does *not start automatically* after the current clip. Requires Viz Engine 3.3 or above.
- **Initialize:** Loads the selected page(s) on the program and preview renderer.
- **Cut:** Cuts an element to the pasteboard (shortcut: *Ctrl+x*).
- **Copy:** Copies an element to the pasteboard (shortcut: *Ctrl+c*).
- **Paste:** Pastes in the current pasteboard element (shortcut: *Ctrl+v*).
- **Delete:** Deletes the current selected element.
- **Create Group:** Creates a new group.
- **Show AutoDescription:** Switch. Shows the Auto Description for item.
- **Set external cursor:** Sets the global cursor to the selected template or page. This setting is related to GPI or Automation setups.
- **Clear external cursor:** Clears the current global cursor.
- **Tree:** The Tree option contains sub-menus for selecting, expanding, collapsing, and moving data elements.
 - **Select All:** Selects all data elements in the playlist.
 - **Select None:** Removes all selections.
 - **Expand All:** Expands all nodes in the tree, revealing the grouped data elements.
 - **Collapse All:** Collapses all nodes in the tree, hiding the grouped data elements.
 - **Hide Empty Groups:** Hides empty groups. This setting only affects the current playlist. In order to make this setting global for all playlists, enable the **Store as Default** option in the playlist column context menu.
 - **Wrap Text:** Wraps all text properties of the element, adjusting the row height accordingly.
 - **Font...:** Opens the Font Chooser to set a different font for the playlist.
 - **Show Row Lines:** Switch. Select to show Row Lines.
 - **Show Alternating Row Colors:** Switch to show alternating row colors.
- **Move Pages To Group:** Moves the selected pages to the group selected in the sub menu of this menu item.
- **Edit Filter:** If using a filter, select this to display the **Edit Filter** menu window where you can edit a filter.

5.19.3 Page List Columns

Select the columns that should be visible in the page list. Right-click one of the existing page list columns, and the Page List columns context menu appears:



Choose the columns to add or remove.

- **Available:** Shows the availability of the element as a percentage.
- **Channel:** Selects the playout channel on which the graphics are rendered.
- **Concept:** Shows which concepts are associated with the data element.
- **Description:** Shows by default the text entered in the editable tab fields (or the element name, if the element is a stand-alone media file). Can be edited inline. Only a page's Description field can be changed.
- **Duration:** Shows the duration of the element.
- **Effect:** Opens the Choose Effect dialog when clicking the browse button.



Note: Transition effects cannot be used for transition logic scenes.

- **Layer:** Shows the layer information. This can be [FRONT], [MAIN] and [BACK].
- **Loaded:** Shows (in percent) load status of the element.
- **Loop:** Loop enables a loop mode for scenes or a full screen video clip on the video channel. Note that a full screen video clip plays once by default; however, video clips in graphics loop by default. A complete playlist (see control bar) or video clip may be looped; however, it is of course not recommended to set more than one of these modes to loop at once. Looping in a page list is aligned with the looping in a timeline editor. However, when "Reload the current page if it is changed in the MSE" is disabled from **Configuration>>General** then looping between page list and timeline editor is asynchronous.

- **Modified:** Displays the time when the element (graphic, still image, or video) was modified/created. This feature is mostly used for sorting, and note that sorting must be enabled in the page list context menu in order for this to work.
- **Page:** Shows the page ID that usually is a numeric callup code. Can be edited inline.
- **Status:** Shows the status of the selected page. Page status indicators are Finished, Unfinished and None. **None** means: no certain page status indicated. Viz Trio handles these pages regularly. **Finished** means: the page content is edited to completion and page should not be changed. Viz Trio prohibits editing pages with this status in the page editor. **Unfinished** means the page needs further editing before playout. This is a simple reminder to the journalist to finish editing the page before playout. Also see the macro commands page in the [Macro Commands and Events](#) section.
- **Template Description:** Shows the template's description; usually what kind of template it is, for example lower third, bug and full screen. Descriptions are predefined by the scene designer.
- **Template Id:** A page references a template, thus showing the template's ID. The ID is usually a numeric callup code.

 **Note:** Template and Page IDs are usually numeric. However, Alphanumeric IDs can be allowed. You must then enable **Allow Alphanumeric Imports**, see the General section.

- **Variant:** Shows the template variant text. To view a list of variants with thumbnails, left-click the text field.
- **Store as Default:** Stores the current selected column values as Defaults.
- **Use Default:** Uses the stored column Defaults.
- **Auto Fit Columns:** sizes columns automatically.
- **Enable Sorting:** disabled by default. The sort feature is specific per playlist, so if sorting is enabled on one playlist or show, it can still be disabled for another Playlist or show.

5.19.4 Concepts

Concepts allow you to switch between complete sets of designs. For example, from news at six to news at ten, or even from broadcast to online.

The Concept can be set in four places, as the first overrides the second, which overrides the third and so forth. Concepts that are higher in the list take precedence over Concepts lower in the list.

This is the Concepts priority list and they can be set for:

1. The *playout device* that overrides all concepts set elsewhere, including elements that are explicitly set. This Concept can be added to a specific Output (Viz Engine) in a Channel in a Profile.
2. The *individual elements* in your playlist (not the page list).
3. Setting the *page list or playlist* Concept overrides elements in the given list, but not the elements that are explicitly set. Concepts for the template list, playlist and page view are affected by the concept setting for the page list. However, if the playlist has set its own concept, it is not overridden by the concept setting for the page list.
4. The *show concept* overrides concept settings for all elements in the template list, page lists, page views and playlists, but not for elements that are explicitly set.

5.19.5 Templates and Groups

Change Template

For standalone scene templates, it is possible to [change a template](#) that a page is using without having to retype its content. This works under the following conditions:

- The tab fields in the original, and the new template, must be of the same type. If tab field 1 is of type text in the original template, the changed template must have text in tab field 1 as well. This also applies images.
- Be aware of restrictions or differences between the two templates such as character limits. A property exposed on the source template might not be exposed in the target. As a general rule, the templates should be similar in their logical structure.
- The total number of tab-fields should be the same:
 - If the source template has three bullet points (exposed as three tab-fields) and the target has four, for instance, the first three are filled with existing values and the fourth has a default value.
 - If the source template has four bullet points and the target only has three, the last bullet isn't shown. If no new text has been saved or added, it is possible to change back and the original template is restored.

Changing a Template

1. Select the pages for which a template is to be changed (use CTRL or SHIFT to multi-select).
2. Right-click the page(s), and select **Change template...** from the context menu.
3. Enter the new template name/number.

 **Note:** The name of the template is case sensitive.

4. Click **OK**.

Changing a Template Using a Macro Command

1. Assign the macro command `page:change_template` (string templateName) to a keyboard shortcut.
2. Press the keyboard shortcut assigned to the `change_template` command to perform the change.

 **Note:** The name of the template is case sensitive.

3. Click **OK**.

Creating Groups

It is possible to create groups in a page list to organize pages. To create a group, right click in the page list and select **Create group**. A new group appears with the default name `new_group`. Click on the name once to edit, or right click and choose edit/name.

To add pages to the group, drag and drop elements onto the right side of the group element. A page added to a group is added as a child node of the group when it is dropped onto the group. To move pages out of a group, drag and drop it outside the group.

Note: Pages cannot act as groups for other pages.

Note: Stories imported from an Avid NewStar newsroom system appear automatically as groups (one group per story). This is similar to a MOS playlist, and is built using the traditional show type. Hence, it has no support for context switching (concept and variants).

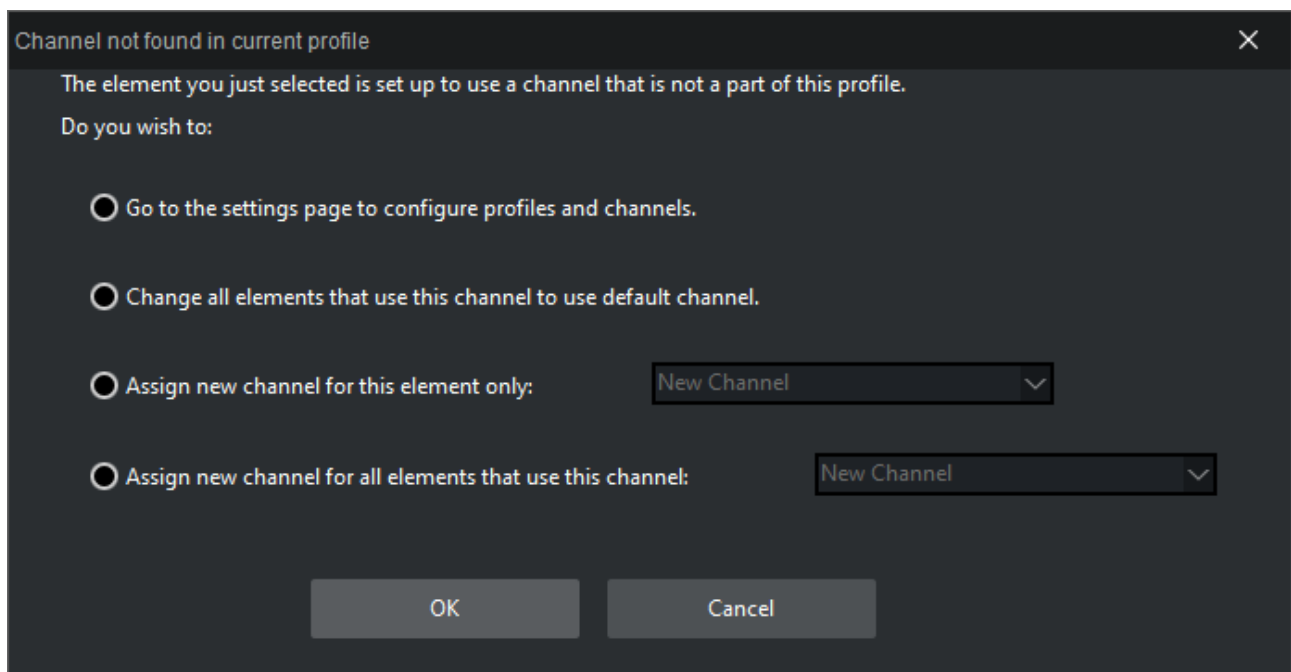
Copying and Pasting Groups

It is possible to copy groups using the copy (**CTRL + C**) and paste (**CTRL + V**) commands in the same show. The pages within the pasted group are given new unique names.

Manually Sorting Pages

Disable sorting in the page list to move pages up and down by drag and drop. This can be used to create a sequenced playlist, for example.

Channel Not Found



The channel not found dialog box opens if a channel has been set for a page (or template) that does not exist in the current profile. This occurs if a page has been configured to use a specific channel (e.g. C) and that channel has been deleted, for example.

The easiest way to avoid this scenario is to always use the default assignment [PROGRAM] as this refers to the status of the channel, and not the channel name.

Triggers the pages on-air.

5.19.6 Arm and Fire

With the Arm and Fire feature it is possible to play out two or more different elements on two different channels simultaneously. By using the commands `page:arm` and `page:arm_current` several elements can be cued up on their respective channel before firing them: channel A, C and D - take. The commands `page:fire` and `page:fire_all` execute actions (take, continue, out, and so on) on all relevant channels that are armed. `page:unarm`, `page:unarm_all` and `page:unarm_current` clears channels from being executed.

This section describes the following related functionalities:

- [Reading a page](#)
- [Taking Pages On-air](#)
- [Deleting a Page](#)
- [Saving to XML](#)
- [Loading from XML](#)
- [Adding Transition Effects Using the Page List](#)

Reading a Page

1. Select a page from the page list.
2. Double-click the page in the page list, or
3. Right-click the Page and from the appearing context menu select **Read**, or
4. Press the Read key on the [Cherry Keyboard](#).

 **Note:** When reading the page, it is displayed in the preview window and opened for editing in the [Page Editor](#).

Taking Pages On-air

1. Select a page from the page list.
2. **Read** the page.
3. Click the **OnAir** button in the upper right corner to set Viz Trio in on-air mode.
4. Click the **Take** button or the **Take** key on the keyboard.
5. Click the **Continue** button to continue the animation if the scene has stop-points.
6. Click the **Take Out** button or **Take Out** key on the keyboard to take the page off air.

Deleting a Page

1. Select and right-click a page from the page list and from the appearing context menu select **Delete page**, or
2. Press the **DELETE** key on the keyboard.

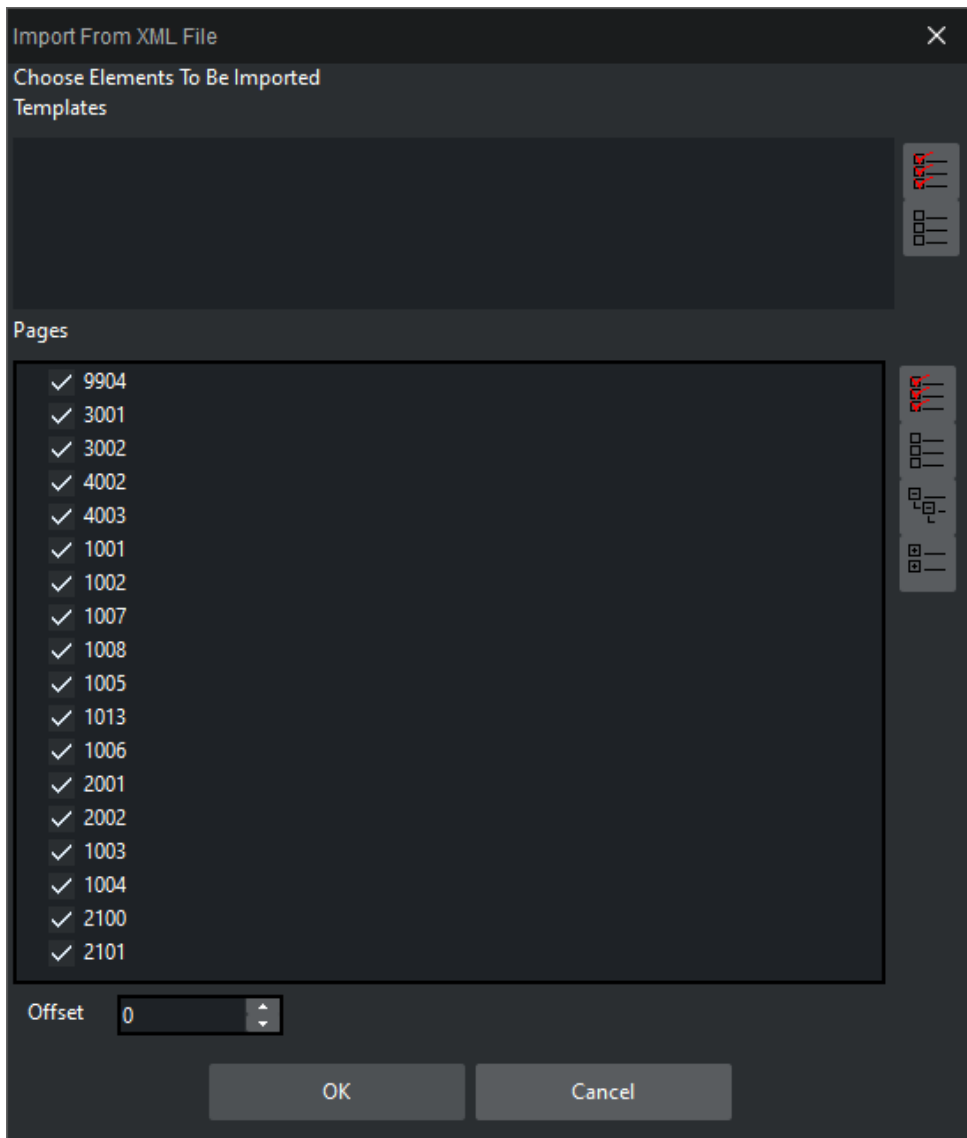
 **Note:** You can multi-select pages by holding in the SHIFT or CTRL button and selecting the range or individual pages to be deleted.

Saving to XML

To save a page to an XML-file on disk, use **File > Export Selected Pages Archive...**

For more information about this option, see [Export Selected Pages Archive](#).

Loading from XML



1. From the main menu click **Page > Load from XML...** option.
2. Choose the pages to be imported from the list.
 - Click **Check All** to select all pages.
 - Click **Uncheck All** to deselect all pages.
3. *Optional:* Set an Offset for the imported page numbers such that the imports do not conflict with existing page numbers.

Adding Transition Effects Using the Page List

1. Right-click a show's page list column header, and from the appearing context menu select **Effect**. This displays the Effect and Effect Icon columns.
2. Click the **ellipse** button in the **Effect** column and from the appearing dialog box select a transition effect.

See Also

- Keyboard Shortcuts and Macros
- [Show Properties](#)
- [Transition Effects](#)

5.20 Toolbar

The toolbar gives you access to existing shows, Viz Pilot playlists, newsroom playlists and lets you create new shows and show playlists.

Note: There are differences between a show, a show playlist and an external Pilot or MOS playlist, especially in the way they are managed and monitored. This chapter describes some of these differences.



The Show Control view contains several buttons that in turn opens different options. In short they can be described as such:

- **Change Show:** Displays the Show Directories dialog box.
- **Add Page View:** Opens the Add Page List View dialog box.
- **Create Playlist:** Opens the Create Playlist dialog box.
- **Show Properties:** Opens the Show Properties window.
- **Cleanup Renderers:** Cleans up all data on the renderers (see the Cleanup Channels section).
- **Initialize:** Initializes the show on the renderers (see the Initialize Channels section).
- **Show Concept:** Displays the show's concept, for example Sport or News.
- **Callup Code:** Shows the callup code for the next page to be read from the page list.
- **Read Page:** Reads the page that has the callup code shown in the Page To Read window.

This section covers the following topics:

- [Show Directories](#)
- [Add Page List View](#)
- [Creating Playlists](#)
- [Show Properties](#)
- [Cleanup Channels](#)
- [Initializing Channels](#)
- [Show Concept](#)
- [Callup Page](#)

5.20.1 Show Directories



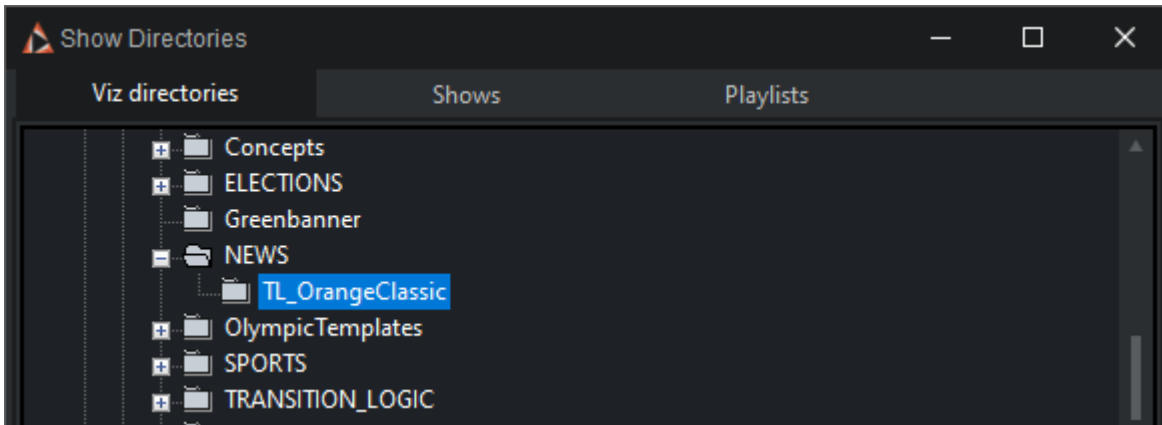
Clicking the **Change Show** button displays the Show Directories dialog box.

A [Shows](#) tab and a [Playlists](#) tab are displayed by default. However, a [Viz Directories](#) view is also available for compatibility with previous Viz Trio versions.

Viz Directories

The Show logic is the recommended way of organizing pages. However, it's also possible to set the path by choosing a Viz Artist folder for each show. To do this, choose the **Viz directories** tab below the **Show Directories** heading.

Note: The Viz directories tab is disabled by default. Open **Configuration**, and in the **User Interface/User Restrictions** section, enable **Browse viz directories when changing shows**.

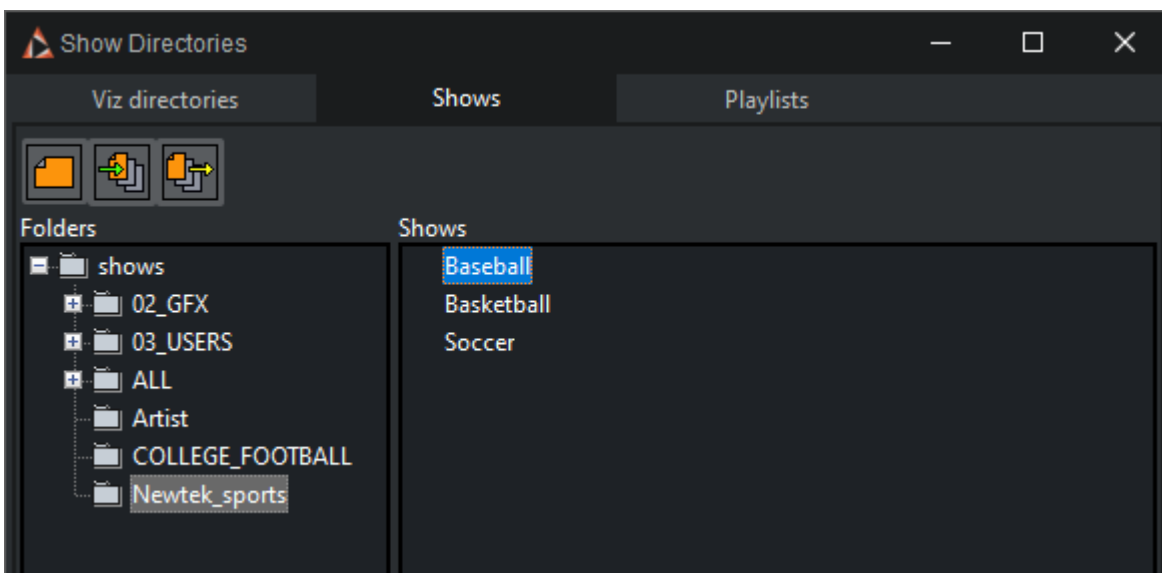


When the Viz Artist folder is set and pages are imported, a show folder with the same path and name is created in the Shows view. To keep using the Viz Artist directory workflow, where each show is tied to a Viz Artist folder, simply ignore the Shows view.

It is possible to switch at any time to the Shows view logic. However, these shows can only be accessed using the Shows view, once show folders with paths that differ from the Viz Artist scene data structure have been created or if there is more than one show per folder.

Shows

A show folder is used to organize shows that belong to a certain show or production. Scenes from the whole Viz Artist scene tree can be imported into a show. From version 2.11, Viz Trio only supports [Context Enabled Shows](#). You can also click [Create Playlist](#) to play out pages in a carousel-like manner. All scenes added to a show are added to the Templates list.



The **Folders** and **Shows** panes contain context menus that provide options for show management.



The buttons are used to create new shows, and to import and export shows. The **Import** and **Export Show** buttons let you export and import shows with all Viz Artist archives, Viz Trio pages, page views, local macros and key bindings, database setups, script units and so on.

Context Menu

Folders

- **Add folder:** Creates a new show folder at any level in the tree structure.
- **Rename folder:** Renames any folder at any level in the tree structure.
- **Delete folder:** Deletes any folder at any level in the tree structure.



Caution: Make sure to export or delete all shows and sub-folders before deleting a root folder.

Shows

- **Rename Show:** Renames an existing show.
- **Create Show:** Creates a new show.
- **Delete Show:** Deletes an existing show.
- **Export Show:** Exports an existing show. Selecting this option opens the [Export Show](#) window.



Note: It is possible to move a show to a different Show folder using drag and drop. Select a show and drag and drop it into the new folder location.

Create Show

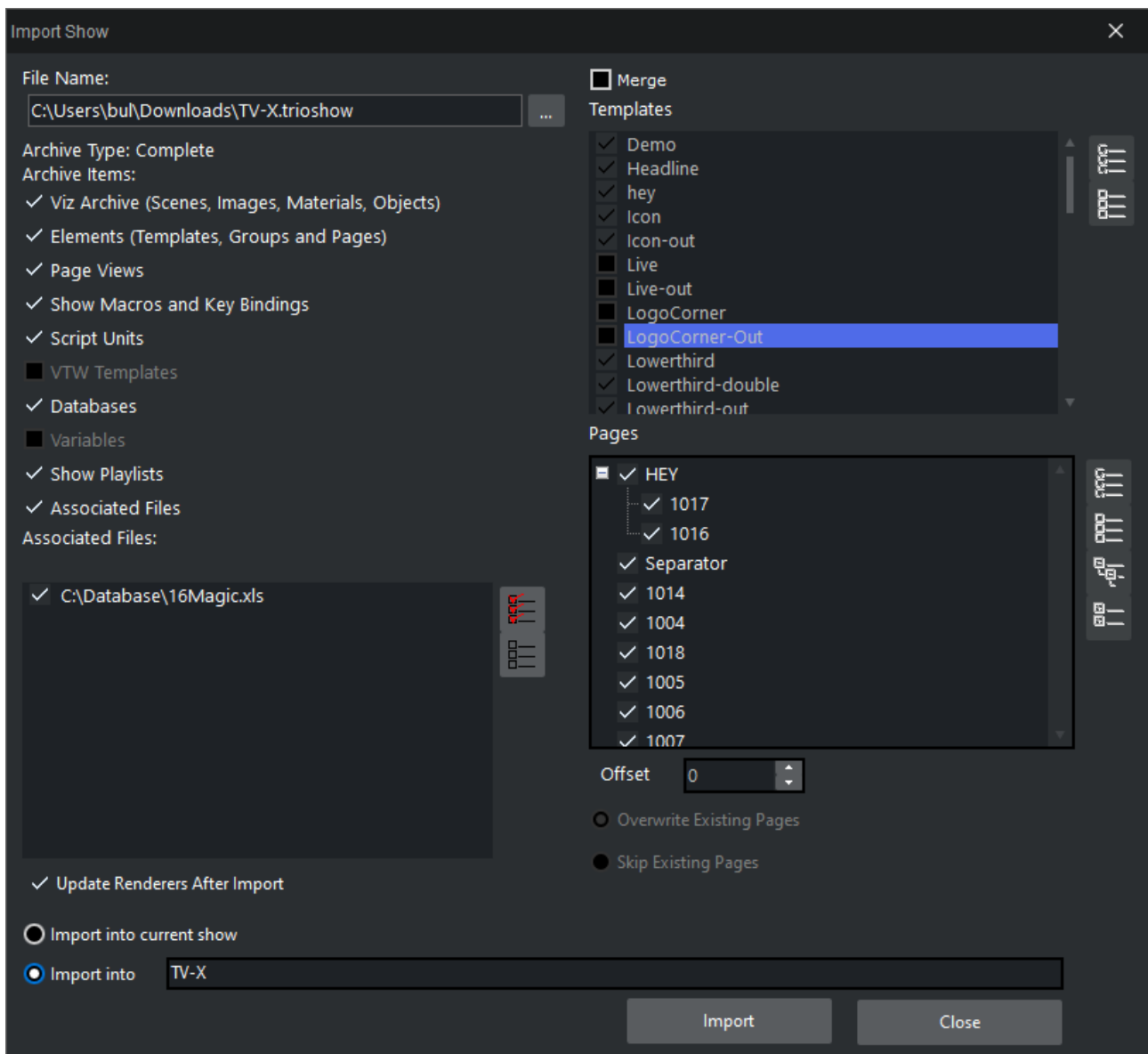


Click the Create Show button to create a new show in the current show directory.

Import Show



Clicking the **Import Show** button opens the Import Show dialog.



When importing a non context-enabled show that was created in a previous Viz Trio version, a dialog appears during import, asking if the show should be converted into a context-enabled show type. When confirming this question, the show is automatically converted to context-enabled upon import.

- **File name:** Sets the path to the show to be imported. When clicking the browse button, a browse dialog opens allowing a show file (*.trioshow*) to be selected.
- **Archive Items:** For details, see the [Export Show](#) section.
- **Update Renderers After Import:** Reloads all scenes on the renderers.
- **Import into current show:** Imports templates and pages into the current show.
- **Import into:** Imports templates and pages into a specified, suggested or new show.
- **Merge:** When checked, this option merges templates and pages into the existing show.
- **Templates:** Lists all templates in the file.
 - **Select all (button):** Selects all templates.
 - **Deselect all (button):** Deselects all templates.

- **Pages:** Lists all in the file.
 - **Select all (button):** Selects all pages.
 - **Deselect all (button):** Deselects all pages.
 - **Expand all (button):** Expands all nodes.
 - **Collapse all (button):** Collapses all nodes.
- **Offset:** Sets the numeric value used for pages that are imported with offset values. If a page has callup code 1000, and the offset is 100, the new code is 1100.

 **Note:** Only pages can be merged with an offset.

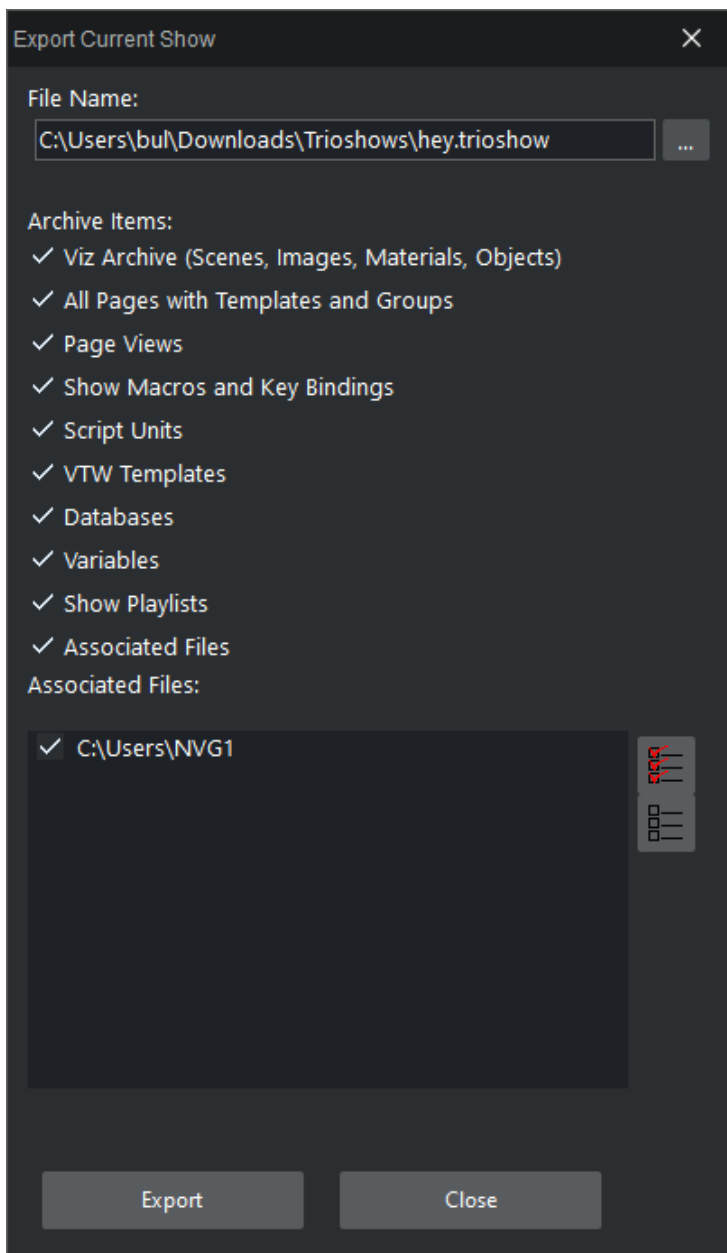
- **Overwrite Existing Pages:** If the callup codes already exist in the show, the pages are overwritten by the imported pages.
- **Skip Existing Pages:** If callup codes already exist in the show, the pages are not overwritten by the imported pages.

 **Note:** The default is to import all elements present in the show archive.

Export Show



Clicking the **Export show** button opens the Export Current Show dialog.



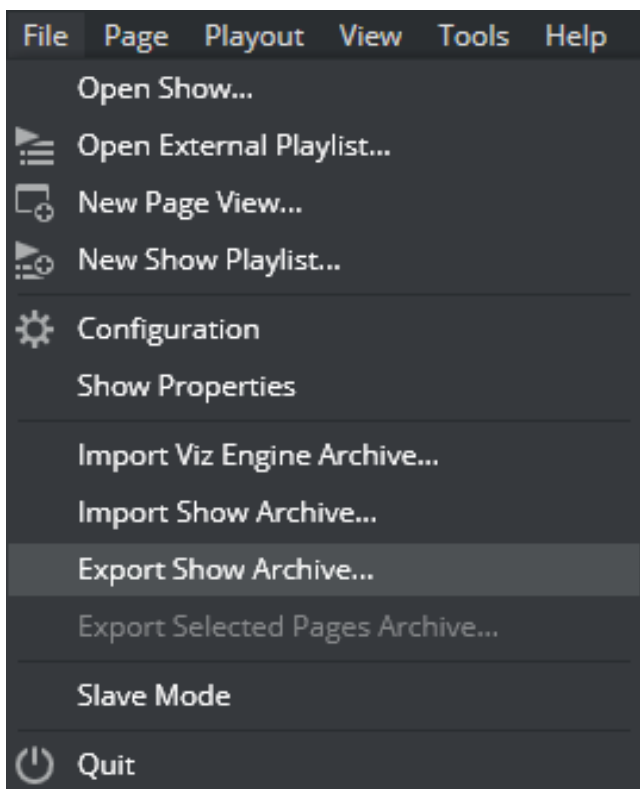
- **Viz archive (scenes, images, materials, objects):** Viz Engine archives should normally be included in an export. An exception is when the export is done to the same data root, or a data root with the same structure and content as the one exported from.
- **Elements (templates, groups and pages):** Includes all the show's templates, groups and pages.
- **Page Views:** Maintains the show's page view organization.
- **Show Macros and Key Bindings:** Includes the show's macro and key shortcut specifications (see the [Show Properties](#) sections).
- **Script Units:** Includes the show's script units that are stored on the Media Sequencer (show scripts). File scripts must be added manually to *Associated files* (see the [Show Properties](#) section).
- **VTW Templates:** Includes the show's Viz Template Wizard templates.
- **Databases:** Includes the show database setups.

- **Variables:** Includes the show stored variables (for example a counter) and their intermediate information (see the [Show Variables](#) section).
- **Show Playlists:** Includes the show playlists that are created as part of the Viz Trio show, hence, this is not related to Viz Pilot and newsroom playlists.
- **Associated Files:** Includes the show associated files (see the [Show Properties](#) section).
 - **Select all (button):** Selects all associated files.
 - **Deselect all (button):** Deselects all associated files.

 **Note:** The export file extension is **.trioshow*.

Export Selected Pages Archive

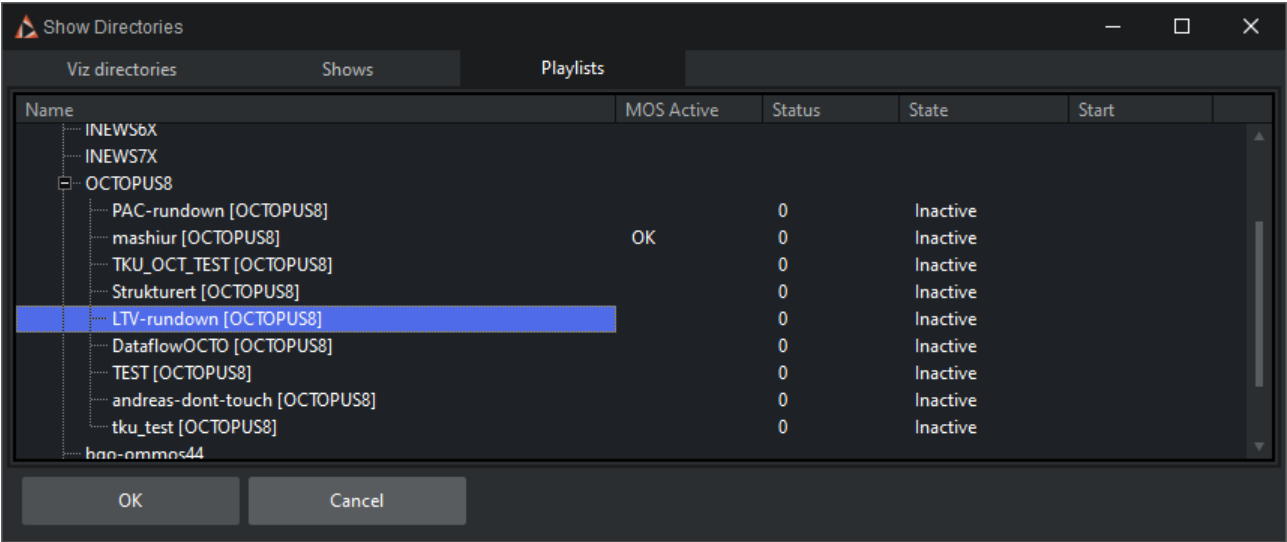
To access this feature click on **File > Export Selected Pages Archive...**. This process can also be triggered by using the macro command `gui:show_export_selected_page_dialog`.



This menu option is available when there are pages selected in the trio page list. Clicking on this menu triggers the **Export Show** dialog to export the selected pages. It is almost the same as the [Export Show](#) feature, but it exports only selected pages and all its associated items.

Playlists

The Playlists tab allows the operator to open a Viz Pilot and/or newsroom playlist. With a Viz Pilot or newsroom system integration, Viz Trio can be used to playout Viz Pilot and newsroom elements.



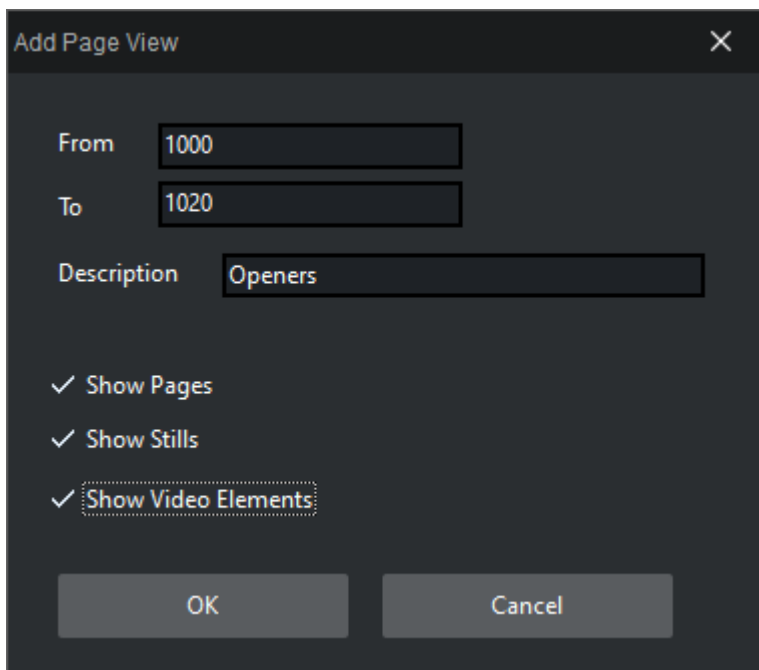
- **Name:** Shows the names of the playlists provided by the newsroom control system.
- **Start:** Specifies the start time of the rundown.
- **MOS Active:** Indicates whether the rundown is activated from the newsroom system.
- **Status:** Indicates a playlist's current status as given in the newsroom system.
- **State:** Indicates the currently active profile.

 **Note:** Newsroom playlists are by default read only. However, there is a [Import and Export Settings](#) setting that can be disabled to allow editing of MOS playlists.

5.20.2 Add Page List View



Clicking the **Create page view** button adds a filtered Page List view. There are no limits to the number of Page List views.



When a new view is created, a callup code range must be specified (for example 1000 to 2000). All pages within that range are displayed in the new view. The original views, such as the template and/or page list, remain unchanged. The additional views are filtered views of the main view.

5.20.3 Creating Playlists

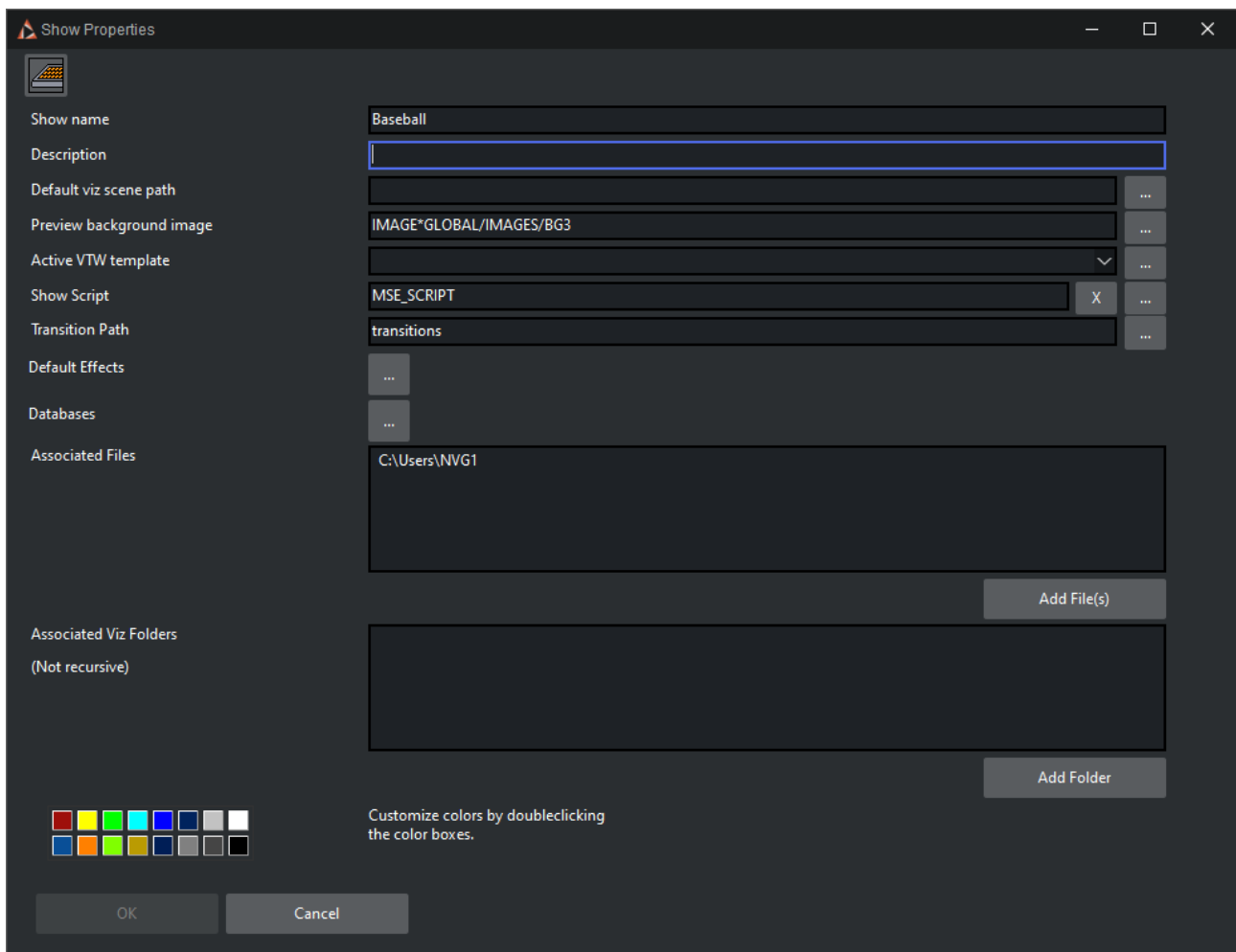


Clicking the **Create Playlist** button creates a show playlist that can be filled with Viz Trio pages, Viz Pilot and newsroom data elements.

5.20.4 Show Properties



Clicking the **Show Properties** button or going to **File > Show Properties** on the main menu opens this dialog box:



- **Show name:** Name of the show.
- **Description:** Description of the show.
- **Default Viz scene path:** Default Viz Engine path for imported scenes.
- **Preview background image:** Sets the image path for the preview background image. For more information about preview modes, see [Preview modes RGB, Key and RGB with background image](#).
- **Active VTW template:** Sets a Viz Template Wizard template to be used for the current show. Clicking the browse button opens the VTW Templates: Current Show window. From this window templates can be imported, exported or removed. When a template is set for a show it is displayed as part of the dockable windows.

Note: For information on how to export Viz Pilot templates, see the Template Wizard section in the [Viz Pilot User Guide](#).

- **Show Script:** Sets a show script that acts as a global script for all templates and pages.
- **Transition Path:** Configures the show's transition effects' scene path. Effects are set per page.
- **Default Effects (ellipse button):** Opens the **Default Effects** window where you can select *Default Video/Still Effect* and *_Default Scene Effect _* from a dropdown. If no effects are selected, default effect settings are used, which are sufficient for most usages.

- **Databases (ellipse button):** Opens the Database Config editor that allows you to [create a new database connection](#) for linking tab fields to values in a database or file (for example [Microsoft Excel and Unicode Support](#)).
 - In early versions of Viz Trio there was only support for database linking of text properties. It was possible to link a value of a single text property to one cell within a spreadsheet. This is called scalar linking as it is only possible to hold one value at a time.
 - In more recent versions of Viz Trio table properties were added, and introduced the concept of table linking, where the contents of the table property were linked to the contents within a Microsoft Excel spreadsheet. Thus it is possible to link several rows or columns at once.
 - It's important to understand the principle difference between the two as they manifest in different versions of the database link editor (see the [Database Linking](#) section).
- **Associated Files:** Displays a list of associated show files, for example video and audio clips to the show. The files are saved with the show when it is exported.
 - **Add File:** Opens a Windows file browser to browse for and add a file to the current Show Properties.
 - **Delete File(s):** Removes the selected file from the current Show Properties.
 - **Delete All:** Removes all associated files from the current Show Properties.
- **Associated Viz Folders (not recursive):** Displays a list of associated show folders. The files are saved with the show when it is exported. When imported it is imported as part of the `_Viz` archive `_option`.
 - **Add Folder:** Opens a Viz Engine browser to browse for and add scene, image, geom, material or font folders to the current Show Properties.
 - **Delete Folder:** Removes the selected folder from the current Show Properties.
 - **Delete All:** Removes all associated folders from the current Show Properties.
- **Customize colors by double-clicking the color boxes:** Double-click on the color boxes to customize show specific character colors (see also how to [edit the text color](#)).

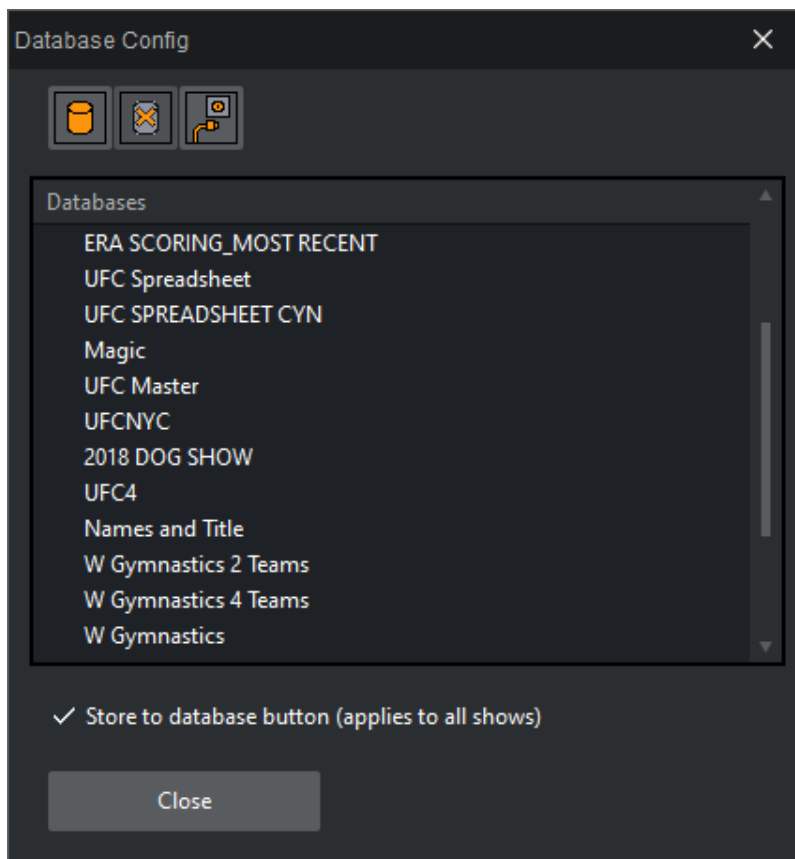
Setting the Transition Effects Path

1. Click the **Show Properties** button to open the [Show Properties](#) pane.
2. Click the **ellipse (...)** button next to the **Transition Path**.
3. Search Viz for the **transitions** effects folder, and click **OK**.
4. Click **OK** to close the Show Properties window.

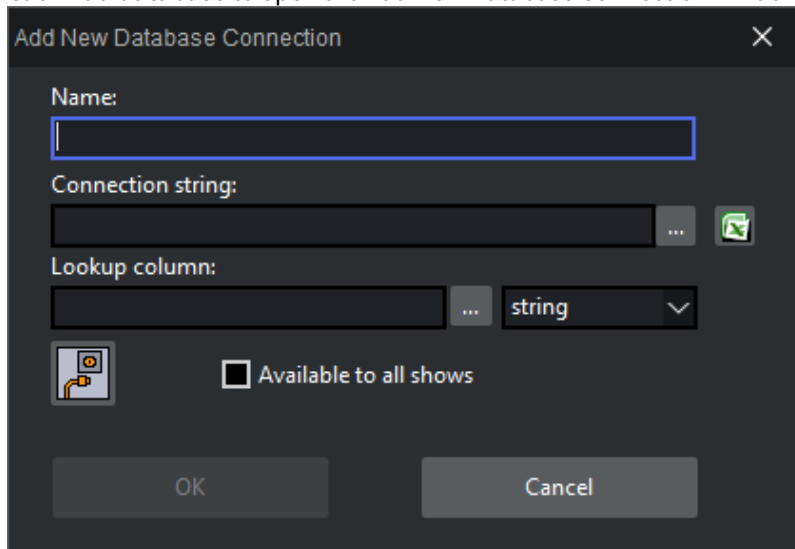
 **Note:** The effects folder should ideally be placed according to the Viz Pilot preferences which is at the root of the Viz database.

Creating a New Database Connection

1. Click the Database settings ellipse (...) button in the **Show Properties** section to open the Database Config.



2. Click **Add database** to open the Add New Database Connection window.



3. Enter a **Name** for the connection and select one of the following options:

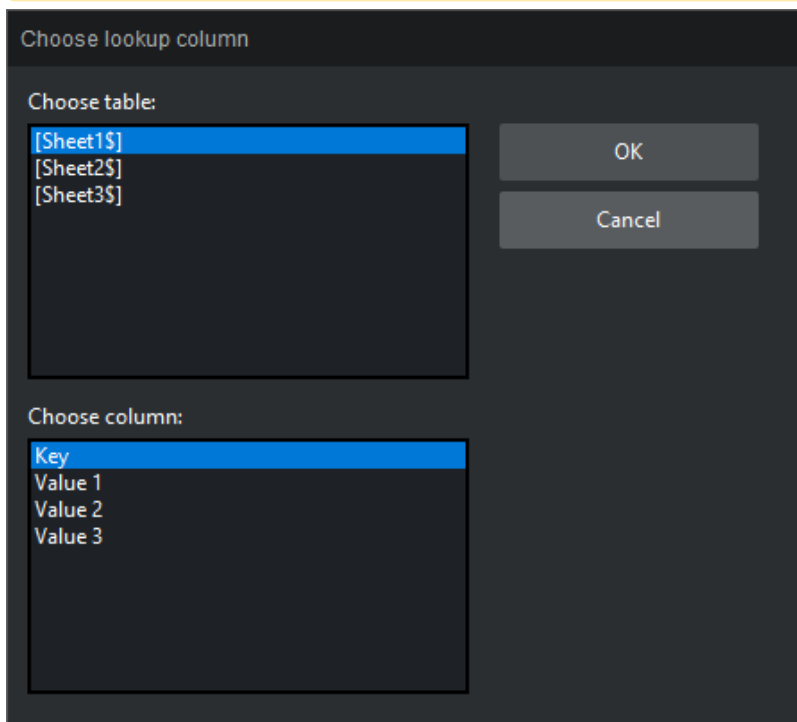
4. *Option 1:* OLE DB

- a. Click the **Connection string's** ellipse (...) button to configure the Data Link Properties
- b. Select an **OLE DB** provider, and click **Next**.
- c. Add the connection parameters.
- d. Click the **Test Connection** button.

5. *Option 2:* Microsoft Excel

- Click the **Microsoft Excel** button to use a spreadsheet. Configuring an Excel spreadsheet automatically, sets the correct provider.
6. Click the **Test connection** button.
 7. Click the **Lookup column's** ellipse (...) button to **choose lookup column**.

Note: When clicking the Lookup column's ellipse (...) button, if the connection parameters are correct, the connection button turns green.



8. *Optional:* Select if the lookup column type should be string or integer.

Note: Specifying the lookup column is only necessary in scalar linking cases.

Note: A Microsoft Excel spreadsheet column is used to match a key value in order to select the scalar value for the text property. For table linking it can be left blank.

9. Specify whether the database should be available to all shows.
 - **Available to all shows:** Shows/pages that are exported and imported into other shows, also import the Excel file and database link settings.
10. Click **OK** to save the database connection.

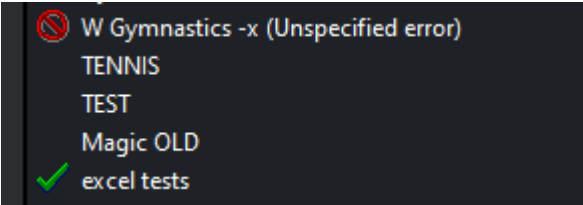
Editing a Connection

Double-click the database connection to open the Edit Database Connection <name> window.

Testing a Connection



Click a database connection in the list and click the Test Connection button.
The result should either be a green hook or an error:



See [Troubleshooting Excel Database Connections](#) for troubleshooting of Excel connections.

Removing a Connection



Click the **Delete Database** button in the configuration dialog.

5.20.5 Cleanup Channels



This function clears all loaded graphics from memory for the page list and all the playlists on the program and preview renderers for the output profile currently in use. It should be used before initializing a new show or in order to re-initialize the same show into the renderer’s memory.

Note: Cleanup commands affect all Viz Trio clients that are connected to the same Media Sequencer, and using the same output profile.

5.20.6 Initializing Channels



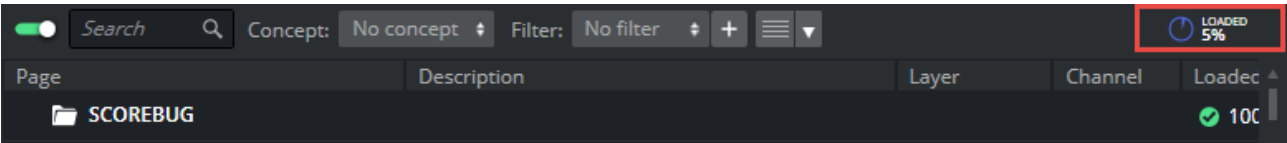
The **Initialize** button loads the current show’s graphics on the preview and program renderers.

Note: In case of transition logic scenes, the state of the background scenes are reset.

Initialize does not refresh everything (it performs a load, not a reload on the Viz Engine). If changes have been made to a scene that was already loaded, a Cleanup renderer command must be issued, and thereafter an Initialize command.

Each playlist and pagelist elements have its own initialize indicator. The Loaded column indicates for each element, the status with percentage and a color-code, Yellow: loading and Green: loaded.

A progress indicator icon indicates the load for all elements in the pagelist or playlist with a tooltip indicating in percent total load.

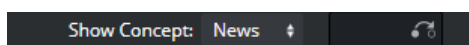


Several shows can be initialized if necessary. After having initialized the first, another show can be switched to and initialized. The graphics for the first one remain there, ready for use.

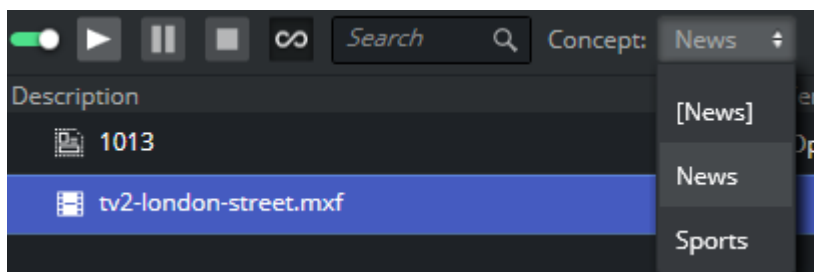
When initializing shows that require a lot of memory, please take the memory use on the program and preview renderers into consideration when loading the graphics. Too many graphics on the renderer(s), use all the physical memory, causing the performance to drop below real-time, which in turn may cause the renderer(s) to become unstable.

Note: Initialize commands affect all Viz Trio clients that are connected to the same Media Sequencer, and using the same output profile.

5.20.7 Show Concept

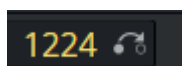


A concept can be switched directly from the [Toolbar](#). It is also possible to switch concepts using context variables in a macro or a script.



Switching concepts for a show's playlist is done in the same way as for a show. However, they can also be switched independently. Concept in brackets, for instance [News] refers to the concept set in the show. This can be overridden explicitly by selecting from the drop down.

5.20.8 Callup Page



Part of the show control is a field and a button used for selecting and reading specific pages. When the pages are read, they are also opened in the [Page Editor](#).

Reading a Page

Enter the page's callup code (number) using the keyboard's numeric pad, and press the **plus (+)** key or the Read Page button to read the page.

See Also

- [Playlist Modes](#)
- [Show, Context and Tab-field Commands](#)
- [Macro Commands and Events](#)
- [Keyboard Shortcuts and Macros](#)
- [Scripting](#)

5.21 Playlist Modes

There are three playlist modes available in Viz Trio:

- Viz Trio show playlists are created using the [Create Playlist](#) button and exists only as part of the show.
- Viz Pilot [Playlists](#) are requested by the Viz Trio operator and modified when integrated with a Viz Pilot system.

Note: A Viz Pilot playlist element cannot be edited, only the structure of the playlist. The Viz Pilot playlist elements can be added to a Viz Trio show playlist, but not a page list.

- Newsroom Playlists are requested by the Viz Trio operator, but not modified by default (see [General](#) user interface settings). Edits to any element in the newsroom playlist is by default discarded as soon as it is updated by the newsroom system. In most cases, Viz Trio is therefore only used to read and play out elements when monitoring newsroom playlists.

For the playout of newsroom playlists to work, the Media Sequencer must establish a connection to Viz Gateway (see [MOS](#) integration).

All shows and playlists are stored in the Media Sequencer for playout, and all Show Playlists can be monitored, stopped and started from the [Active Tasks](#) window.

This section covers the following topics:

- [Activate or Deactivate a Playlist](#)
- [Take Pages On-Air in a Carousel from a Show Playlist](#)
- [Show or Hide Columns in the Playlist](#)

5.21.1 Activate or Deactivate a Playlist

Description	Template Id	Channel	Concept	Loaded	Duration
TITLE OPEN	9402	[A]	[A]	100%	00:00:04.76
TALENT	9025	[A]	[A]	100%	00:00:01.86
SINGER	9025	[A]	[A]	100%	00:00:01.86

1. [Create Playlist](#) or open [Playlists](#).
2. Select Active or Inactive in the menu bar for the currently selected playlist to activate or deactivate the playlist. Activating a playlist makes the Media Sequencers actively monitor the content of the playlist, and start the transfer of video clips to Viz Engine. The duration column shows the duration of the elements. The duration column shows the actual duration of a video clip, or the default duration of the animation for graphic elements. The duration is used when the playlist is played out as a carousel. Then the duration is how long the element stays in the Viz Engine before the next element is taken.

5.21.2 Take Pages On-Air in a Carousel from a Show Playlist

1. Add pages to the playlist.
 - *Optional:* Enable looping.

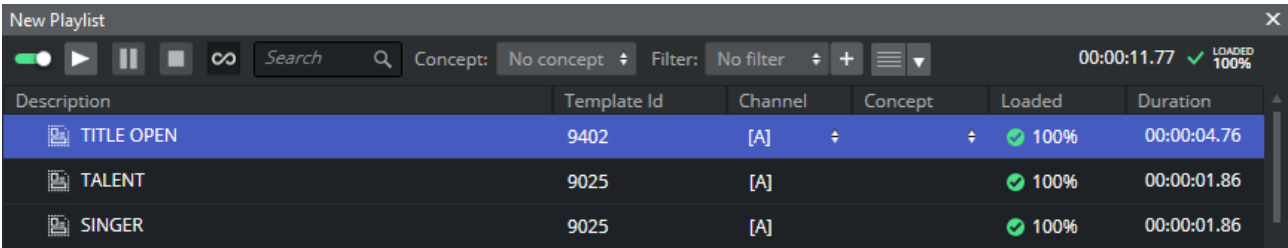
- *Optional:* Set the concept for the playlist or for each element.
 - *Optional:* Set the variant for each element.
 - *Optional:* Set the duration for each element or keep the default duration.
2. Click the **Play** button to start the carousel.
 3. Click the **Pause** or **Stop** button to pause or stop the playlist.

✖ IMPORTANT! Scenes with stop-points and out-animations are being cut as they are **not** automatically continued when used in a carousel playlist.

5.21.3 Show or Hide Columns in the Playlist

- Right-click the playlist header and select columns to display or hide from the menu.
 - The **Auto width** option enables automatic resize of columns within the MOS Playlist area.
 - Disable the **Auto width** option to manually resize the columns.

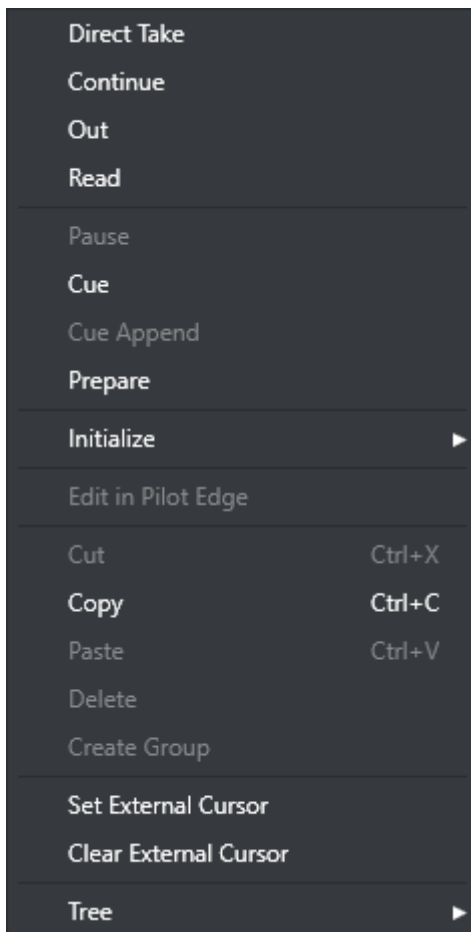
Playlist



Description	Template Id	Channel	Concept	Loaded	Duration
 TITLE OPEN	9402	[A]	÷ ÷	✓ 100%	00:00:04.76
 TALENT	9025	[A]		✓ 100%	00:00:01.86
 SINGER	9025	[A]		✓ 100%	00:00:01.86

The control bar is used to set the concept, search, set filters, manage filters, run, pause, stop and loop the playlist. Available concepts are the same as for the show. However, a concept set for the show does not override the playlist's concept, nor vice versa.

Playlist Context Menu



The Playlist context menu provides options to manage pages and data elements in the playlist.

Note: On composite groups containing video, only the following operations can be performed: **Direct Take**, **Read**, and **Out**. On standalone videos, all commands can be performed (*Cue*, *Pause*, *Prepare*, and so on).

- **Direct Take:** Performs a direct take on the selected data element.
- **Continue:** Continues the playout of an element.
- **Out:** Takes the element out (hard cut).
- **Read:** Reads the element, opening it in the page editor and local preview.
- **Pause:** Pauses the element.
- **Cue:** Prepares the clip for playout so the *first frame is ready* in the player.
- **Cue Append:** Prepares the clip for playout. The clip *starts automatically* when the current clip ends.
- **Prepare:** Prepares the clip in the pending player without affecting the current clip, so it is ready to be played out. The clip does *not start automatically* after the current clip. Requires Viz Engine 3.3 or above.
- **Initialize:** Initializes the selected playlist or element (loads graphics) on the external output channels. Sub menu items for *Initialize*:
 - **Playlist:** initializes the playlist.

- **Element:** initializes the currently selected element.
- **Edit in Pilot Edge:** Open the selected playlist element in Viz Pilot Edge.
- **Cut:** Cuts an element to the pasteboard (shortcut: **Ctrl+x**).
- **Copy:** Copies an element to the pasteboard (shortcut: **Ctrl+c**).
- **Paste:** Pastes in the current pasteboard element (shortcut: **Ctrl+v**).
- **Delete:** Deletes the current selected element.
- **Create Group:** Creates a group in the playlist. Groups can be nested.
- **Set External Cursor:** Sets the global cursor (normally related to GPI or Automation system setups) to the selected data element. This requires that the Show External Cursor (GPI) option has been enabled in the User Interface configuration dialog, see the [Page List/Playlist](#) and [Cursor](#) sections for further details.
- **Clear External Cursor:** Clears the current global cursor.
- **Tree:** The Tree option contains sub-menus for selecting, expanding, collapsing, and moving data elements in the playlist.
 - **Select All:** Selects all data elements in the playlist.
 - **Select None:** Removes all selections.
 - **Expand All:** Expands all nodes in the tree, revealing the grouped data elements.
 - **Collapse All:** Collapses all nodes in the tree, hiding the grouped data elements.
 - **Hide Empty Groups:** Hides empty groups. This setting only affects the current playlist. In order to make this setting global for all playlists, enable the **Store as Default** option in the playlist column context menu.
 - **Wrap Text:** Wraps all text properties of the element, adjusting the row height accordingly.
 - **Font...:** Opens the Font Chooser to set a different font for the playlist.
 - **Show Row Lines:** Switch. Select to show Row Lines.
 - **Show Alternating Row Colors:** Switch to show alternating row colors.

Playlist Columns

Right-Click on the playlist columns-bar to select values to display:

Available
Begin Time
✓ Channel
✓ Concept
✓ Description
✓ Duration
Effect
End Time
Layer
✓ Loaded
Loop
Mark In
Mark Out
MOS ObjAir
Playout Mode
Story #
Template Description
✓ Template Id
Timecode
Variant
Store as Default
Use Default
✓ Auto Fit Columns
Enable Sorting

- **Available:** Displays the status of external resources needed by the Viz Engine (e.g. transferred video, and if it is available on the video playout engines). Errors are shown as tooltips. This column was previously named *Progress*.
- **Begin Time:** Shows the activation start time for a group (format hh:mm:ss).
- **Channel:** Shows which output channel an element should be sent to. Various elements can be sent to different output channels. The output channels can be set directly in the column. By default the main

[PROGRAM] output channel is selected, but this can be overruled by only setting an alternative channel for this element or in a template (that all data elements made from it have). By creating a group and placing elements within it, all elements in the group are organized by having the same channel. The Channel column is presented by default. See the [Channel referencing underlying element](#) section for more information about using channels with playlist elements.

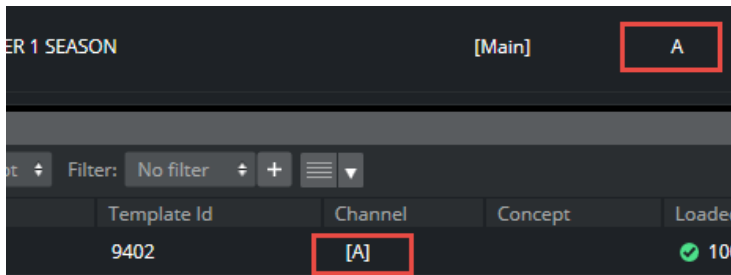
- **Concept:** Shows which concept(s) the data element is associated with.
- **Description:** Shows the description of the element. By default, this shows the path of the scene (or the element name of a stand-alone media file). May be edited inline in the playlist.
- **Duration:** Shows the length of the element. The duration is often set by Viz Pilot or a newsroom system. However, the individual elements duration can be set in Viz Trio by editing the Duration column directly. The total duration of the elements in the playlist are shown in the heading for the playlist. The duration of an element is the time before the next element in the list is taken.
- **Effect:** Opens the Choose Effect dialog, which makes it possible to select a transition effect between two pages.
- **End Time:** Shows the activation end time for a group (format hh:mm:ss).
- **Layer:** Allows loading of graphics in separate layers on Viz Engine (front, middle, back). For example, a lower third can be shown in front of a virtual studio set or any other background, or a bug can be shown in the front layer while a lower third is shown in the middle layer. This column is presented by default.
- **Loaded:** Shows the loaded status (in memory) of the scene and images used for a data element of that scene. Errors are shown as tooltips.
- **Loop:** Displays a loop information column. Only a playlist or videos can be looped, not groups or individual elements in a playlist.
- **Mark In:** Sets mark in times for video clips.
- **Mark Out:** Sets mark out times for video clips.
- **Story #:** Shows the story number for stories in MOS playlists. This is only supported from the ENPS newsroom system.
- **Template Description:** Shows the template description (e.g. name).
- **Template Id:** Shows the template ID.
- **Timecode:** The timecode is an offset time on format *hh:mm:ss:ff*. It indicates that an element should be played out relative to the parent group or video. This is used for instance in composite groups with a video and overlay graphics that is played out on a timeline.
- **Variant:** Select a concept's variant from the drop-list (see the Concept column).
- **Store as Default:** Stores the layout as the default.
- **Use Default:** Reverts to the default layout.
- **Auto Fit Columns:** Automatically fits all displayed columns to the given width of the playlist.
- **Enable Sorting:** Enable sorting when clicking the column headers.

Channel Referencing Underlying Element

Square brackets in the channel column denote two different scenarios. [PROGRAM] refers to the program channel that is defined in the profile. However, an element with a channel in square brackets in a playlist, for example [A], refers to its underlying element in the pagelist.

Square brackets in a playlist therefore denote a link back to the element in the pagelist. If a channel contains square brackets in a playlist, and the channel is changed in the pagelist, the changed channel in the pagelist is automatically reflected in the playlist. When a channel designated to a playlist element does not have square brackets, it plays out on that channel regardless of the channel designated to the underlying element.

In this scenario, changing the channel in the pagelist (the top panel), it automatically reflects in the channel column in the playlist:



Playing Out an Element from a Playlist and Pagelist

It is possible to designate the channel that an element plays out on either from the pagelist or playlist. Furthermore, it is possible to play out both the underlying element in a pagelist and a playlist element on different channels. You can manage these operations in the channel column.

To play out a playlist element on the same channel as a pagelist element, ensure that the same channel is selected for the pagelist and the playlist. For example, channel B in the pagelist is channel [B] in the playlist. Once a channel appears with square brackets in the playlist it always syncs with the channel in the pagelist. This means that a channel changed in a pagelist is always automatically reflected in the corresponding item in the playlist.

Alternatively, select different channels for the element in the playlist and its underlying element in the pagelist to play them out on different channels. If a channel is changed in the playlist and not in the pagelist, the channel selection isn't updated in the pagelist.

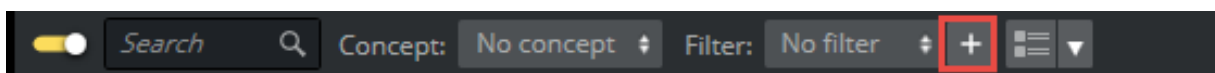
Note: The [PROGRAM] channel is designated with the same square brackets that are used to show a reference to an underlying element. Do not confuse the [PROGRAM] channel with the square brackets for an element in the playlist that refers to its underlying element in the pagelist.

Playlist Filters

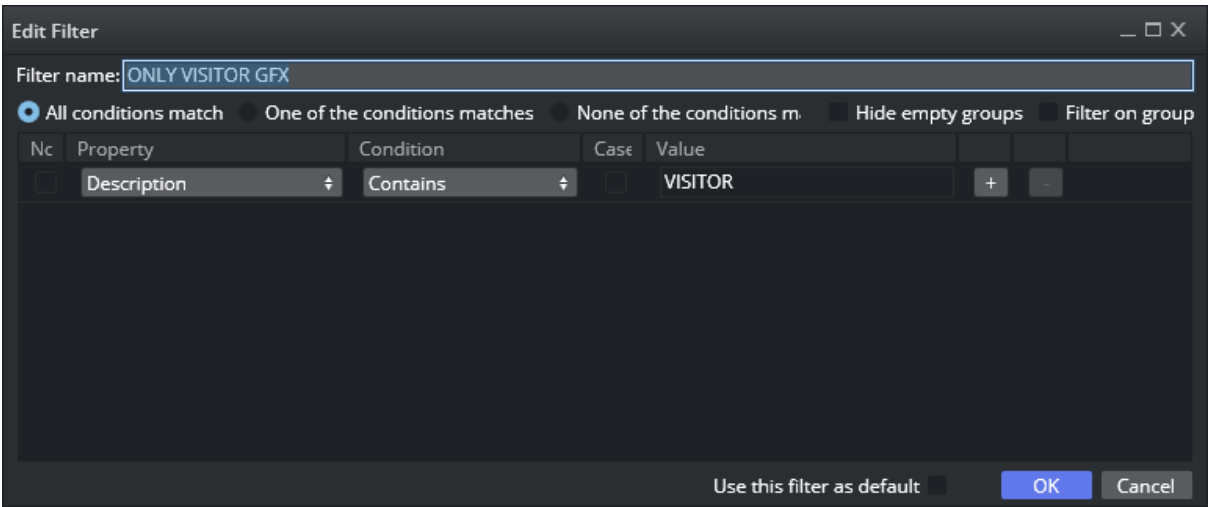
You can add a filter to the playlist to narrow down the list of items displayed.

Note: Filtering is for display purposes only and has no effect on carousel playout behavior.

Create a Playlist Filter



1. Click the **Plus button** on the toolbar.
2. In the Edit filter window that opens, type a descriptive name for the filter.



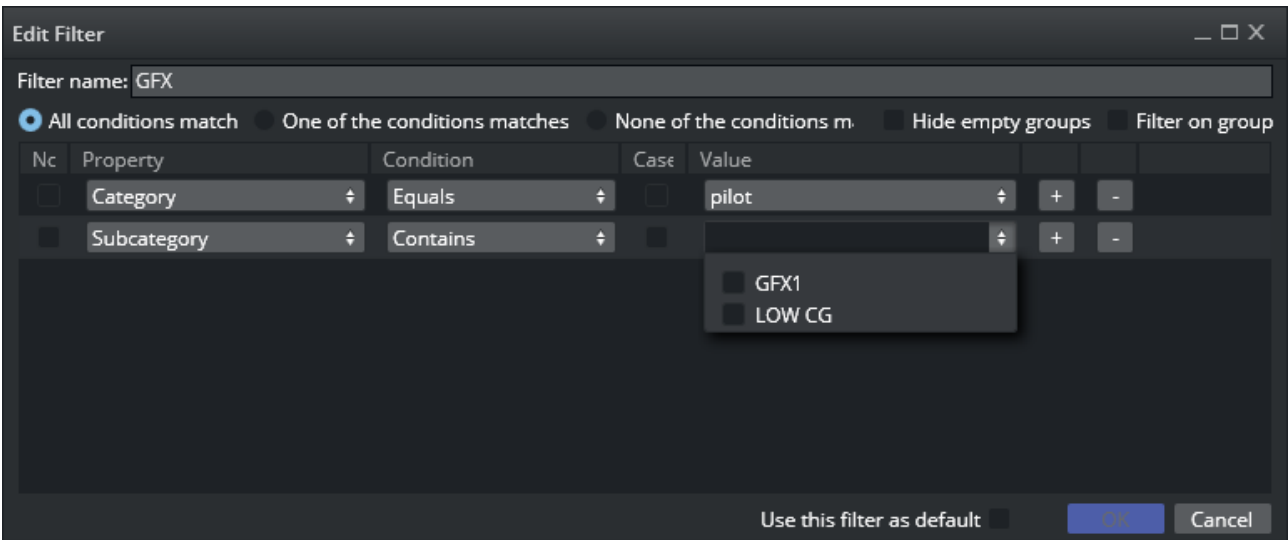
- 3. Fill in the filter values, for example properties and conditions.
- 4. Click the **OK** button.
The filter then becomes available in the Playlist filter list.

Note: Starting with Viz Trio 3, composite videos behave like regular videos when filtering on a playlist.

Add a Playlist Filter for Viz Pilot Integration

Viz Trio can access playlists based on Viz Pilot templates as well as playlists from Newsroom systems via Viz Gateway.

The designer of the Viz Pilot templates can categorize templates by logically dividing them into *Categories* and *Channels*. You can use the Viz Trio filters option to filter playlist views based on these categories created by the Viz Pilot template designer as illustrated in the screenshot below. In this example the playlist can be filtered as **GFX** or **LOW CG**.



To use Viz Pilot template categories as filters:

- 1. Create a new filter or edit an existing filter for a playlist.

2. In the add/edit filter panel set the column **Property** to Subcategory and the **Value** column is populated with the Categories defined by the Viz Pilot template designer. You can now select values for filtering and save the filter. The playlist filtered according to the values selected.

Add a Default Playlist Filter

1. Create a playlist filter, see [To create a playlist filter](#).
2. Select the **Use this filter as default** check box in the Edit filter window.
3. Click the **OK** button.

When enabling the default filter option, all playlists of the specified Viz Trio client always use the selected filter. The default filter affects only one Viz Trio client. The default filter is saved and continues to be activated when restarting the Viz Trio client.

These filters are not deleted even if you delete the corresponding page view. However, they can be deleted manually in the filter drop down. If you create two page views with the same filter properties, they share the same filter.

5.22 Field Linking and Feed Browsing in Viz Trio

Viz Trio supports browsing tab-field values from external sources. Instead of editing tab-field values manually, values from an external feed can be selected.

 **Note:** Viz Trio only supports the Atom 1.0 feed standard.

This section covers the following topics:

- [Workflow](#)
- [Overview](#)
- [Technical](#)
- [Field Linking](#)
- [Feed Browsing](#)

5.22.1 Workflow

1. Import a regular scene as a template in Viz Trio.
2. Read the template.
3. On top of each tab-field of the template, a link button opens the collection picker, where you can specify a feed URI. You must also specify which part of the feed item should be used as a value: title, content, link with a certain link relation, entry or locator.

 **Note:** The link button is not available for pages. Linking to an Atom feed is a Viz Trio template property that is shared by all pages using that template.

4. Press **Save template**.
5. When establishing a valid feed connection, the tab-field value can be selected from the feed items.

5.22.2 Overview

Feed URIs can be defined in the control object of the scene design in Viz Artist. the feed URI for a tab-field in a template can also be defined inside Viz Trio. Each page then uses the feed browser to edit the tab-field value. The specified feed URI in Viz Trio overwrites any existing feed URI from the control object. Reimporting the scene does not replace the specified feed URI with the field definition specified in control object.

A tab-field contains several properties, each of which relate to a control plug-in in the scene. Properties are grouped under a tab-field using a naming convention.

 **Example:** The ControlClassicText plugins specifying the field identifiers 1.name, 1.score and 1.image forms a tab-field 1 with the properties' name, score and image. Selecting a feed item for tab-field 1 instantly applies the parts of the selected item to the properties name, score and image. The individual property values are editable as usual. The feed item selection assists fill-in only.

Feed linking can be used across all tab-field types.

 **Note:** Make sure to define matching object types such as a thumbnail link for control image tab-fields.

Field linking in Viz Trio is a two-step procedure:

1. **Field Linking:** Linking the fields in a template to various parts of data that comes from entries in a feed. This is a setup step that is usually done once for a template.
2. **Feed Browsing:** After field linking has been set up and saved for a template, a new property editor for selecting an entry from a feed is available. This feed browser lets you select an item in a feed that contains the values that should be applied to the field values.

 **Note:** Viz Trio only supports the Atom 1.0 feed standard.

The feed browser supports two types of feeds:

- Flat feeds.
- Hierarchical feeds (folder structure).

Tabfield Grouping

To be able to set multiple tabfield values from the same feed entry, the tabfields must be grouped together. This is done by giving them the same prefix followed by the period character. For example, the following tabfield names generate two groups of tabfields (`candidate1` and `candidate2`):

```
candidate1.name candidate1.image candidate2.name candidate2.image
```

5.22.3 Technical

The following XML Namespace Prefixes are used when referring to XML elements:

XML Namespace Prefixes

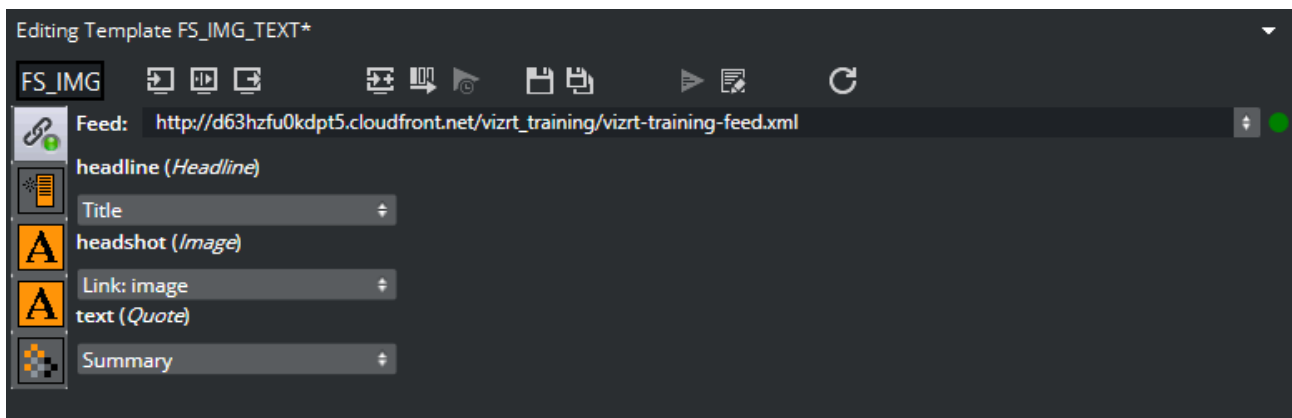
Prefix	URL
atom	http://www.w3.org/2005/Atom
viz	http://www.vizrt.com/types
vaext	http://www.vizrt.com/atom\ -ext
media	http://search.yahoo.com/mrss/
thr	http://purl.org/syndication/thread/1.0
opensearch	http://a9.com/-/spec/opensearch/1.1/

Example: The notation `<atom:entry>` is to be interpreted as referring to the same element as `<entry xmlns="http://www.w3.org/2005/Atom">`.

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in RFC 2119.

5.22.4 Field Linking

Field linking is the process of linking the fields in a template to various parts of data that comes from entries in a feed.



Fields That Can be Linked

- `<atom:author>`

The personal details of the author of the selected `<atom:entry>`, which might be an individual person or an organization. The data is provided in child elements as follows:

- `<atom:name>` : The name of the author. The value of the `<atom:name>` element inside the first `<atom:author>` element that applies for the entry is recorded as the field value. This child element is required if you are specifying the `<atom:author>` element.
- `<atom:uri>` : A URL associated with the author, such as a blog site or a company web site. The value of the `<atom:uri>` element inside the first `<atom:author>` element that applies for the entry is recorded as the field value. This child element is optional.
- `<atom:email>` : The email address of the author. The value of the `<atom:email>` element inside the first `<atom:author>` element that applies for the entry is recorded as the field value. This child element is optional.

- `<atom:content>`

- Both inline and URL content is supported.
- There is no type match, so it is up to the Viz Trio operator to figure out what content can be used.
- For URL content the resource is downloaded first and then applied as the value. If you just want the URL you should use `<atom:link>`.

- `<atom:contributor>`

Name one contributor to the feed entry. The name is provided in a child element:

- `<atom:name>` : The name of the contributor. The value of the `<atom:name>` element inside the first `<atom:contributor>` element that applies for the entry is recorded as the field value. This child element is required if you are specifying the `<atom:contributor>` element. Example:

```
<contributor>
  <name>Twitter</name>
</contributor>
```

- `<atom:entry>`

The whole entry XML.

- `<atom:link>`

This element defines a reference from an entry to a Web resource; in other words this is the value of the `href` attribute.

- `<vaext:locator>`

The text of the Vizrt Atom Extension `<vaext:locator>` element which has a `vaext:type` attribute equal to the mediatype attribute of `<viz:fielddef>` element in the model.

- `<atom:published>`

This element is a Date construct indicating the initial creation or first availability of the entry. The value of the `<atom:published>` element in the entry is recorded as the field value.

- `<atom:summary>`

This element is a Text construct that conveys a short summary, abstract, or excerpt of an entry. The content of the `<atom:summary>` element in the entry is recorded as the field value.

- `<atom:thumbnail>`

- The value of the url attribute of the first thumbnail element in the entry is recorded as the field value.

- `<atom:title>`

This element is a Text construct that conveys a human-readable title for an entry or feed; the title of the selected `<atom:entry>` .

- `<atom:updated>`

This element is a Date construct indicating the most recent instant in time when the selected `<atom:entry>` was modified. The value of the `<atom:updated>` element in the entry is recorded as the field value.

Fields That Cannot be Linked

- `<atom:category>`
- `<atom:id>`
- `<atom:rights>`
- Any other elements.

Elements in Atom Feed

Elements that should be present in the `<atom:feed>` for the full user experience:

- **OpenSearch link:**

```
<atom:link rel="search" type="application/opensearchdescription+xml"
href="http://example.com/opensearchdescription.xml"/>
```

- Optional, required for support for server side searching.
 - The opensearchdescription xml file must contain a template node that returns search results as an atom feed:

```
<opensearch:Url type="application/atom+xml" template="http://..." />
```

- See <http://www.opensearch.org/Specifications/OpenSearch/1.1> for more details.

- **Up link:**

```
<atom:link rel="up" type="application/atom+xml;type=feed" href="http://..." />
```

- Optional, required for nested collections:
 - The up link should link to the parent folder feed for hierarchical feeds.
 - The up link must have a type string equal to `"application/atom+xml;type=feed"`.

Elements in Atom Entry

Elements that should be present in the `<atom:entry>` for the full user experience:

- **Self link:**

```
<atom:link rel="self" type="application/atom+xml;type=entry"
href="http://..." />
```

- The self link must link to the URL that returns the `<atom:entry>` xml.
 - The self link must have a type value equal to `"application/atom+xml;type=entry"`.
 - Needed for refresh/update of values from the item to work.
 - Needed in combination with the up link for remembering and showing the selected feed entry in a hierarchy of feeds (folder structure). Not needed for remembering selection in flat feeds since then the `atom:id` is used.

- **Up link:**

```
<atom:link rel="up" type="application/atom+xml;type=feed" href="http://..." />
```

- The up link must have a href value that is the URL of the feed the entry is in.

- The up link must have a type value equal to `"application/atom+xml;type=feed"`.
- Needed in combination with the self link for remembering and showing the selected feed entry in a folder structure.

- **Down link:**

```
<atom:link rel="down" type="application/atom+xml;type=feed" href="http://..." />
```

- Required if this `<atom:entry>` is to be considered a subfolder instead of a normal `<atom:entry>`.
 - The down link must have a type value equal to `"application/atom+xml;type=feed"`.
 - The down link must have a `href` value that is the url of the feed (folder) the entry represents.
 - Link may contain `thr:count` attribute (RFC4685) indicating how many children there are. If the value of `thr:count` is 0 (zero), then the folder isn't loaded because it is empty. This is an optimization.

- **Thumbnail link:**

```
<media:thumbnail url="http://..." />
```

- Required for thumbnail icons to display on the entries in the feed browser:
 - The URL value must reference a JPEG or PNG image resource.
 - If many thumbnails are defined, the first one is selected as the default. This is according to Media RSS Specification Version 1.5.0.

- **Locator**

```
<vaext:locator type="application/vnd.vizrt.viz.geom">GEOM*Vizrt/Tutorials/VizTrio/...</vaext:locator>
```

- Required to be able to link a tabfield to a locator. A locator is a path that is not representable with the URI `href` in an `<atom:link>`. The selected value is the text node in the locator, so it can in theory be used for any kind of text.
 - For Viz Trio it is meant to be used to select paths to resources that are not URIs, like the resources on the Viz Graphics Hub that is currently in use.
 - The `type` must match the tabfield type.
 - The value from the first locator that matches is used.
 - If no locators matches, an error message is logged in Viz Trio.
 - The Viz Engine types are:
- TEXT: `text/plain`
 - RICHTEXT: `application/vnd.vizrt.viz.richtext+xml`
 - MATERIAL: `application/vnd.vizrt.viz.material`
 - DUPLET: `application/vnd.vizrt.viz.duplet`
 - TRIPLET: `application/vnd.vizrt.viz.triplet`
 - FONT: `application/vnd.vizrt.viz.font`

- CLOCK: `application/vnd.vizrt.viz.clockcommands`
- GEOM: `application/vnd.vizrt.viz.geom`
- IMAGE: `application/vnd.vizrt.viz.image`
- AUDIO: `application/vnd.vizrt.viz.audio`
- VIDEO: `application/vnd.vizrt.viz.video`
- MAP: `application/vnd.vizrt.curious.map+xml`
- TRIOScroll: `application/vnd.vizrt.trio.scrollelements+xml`

- **Other links:**

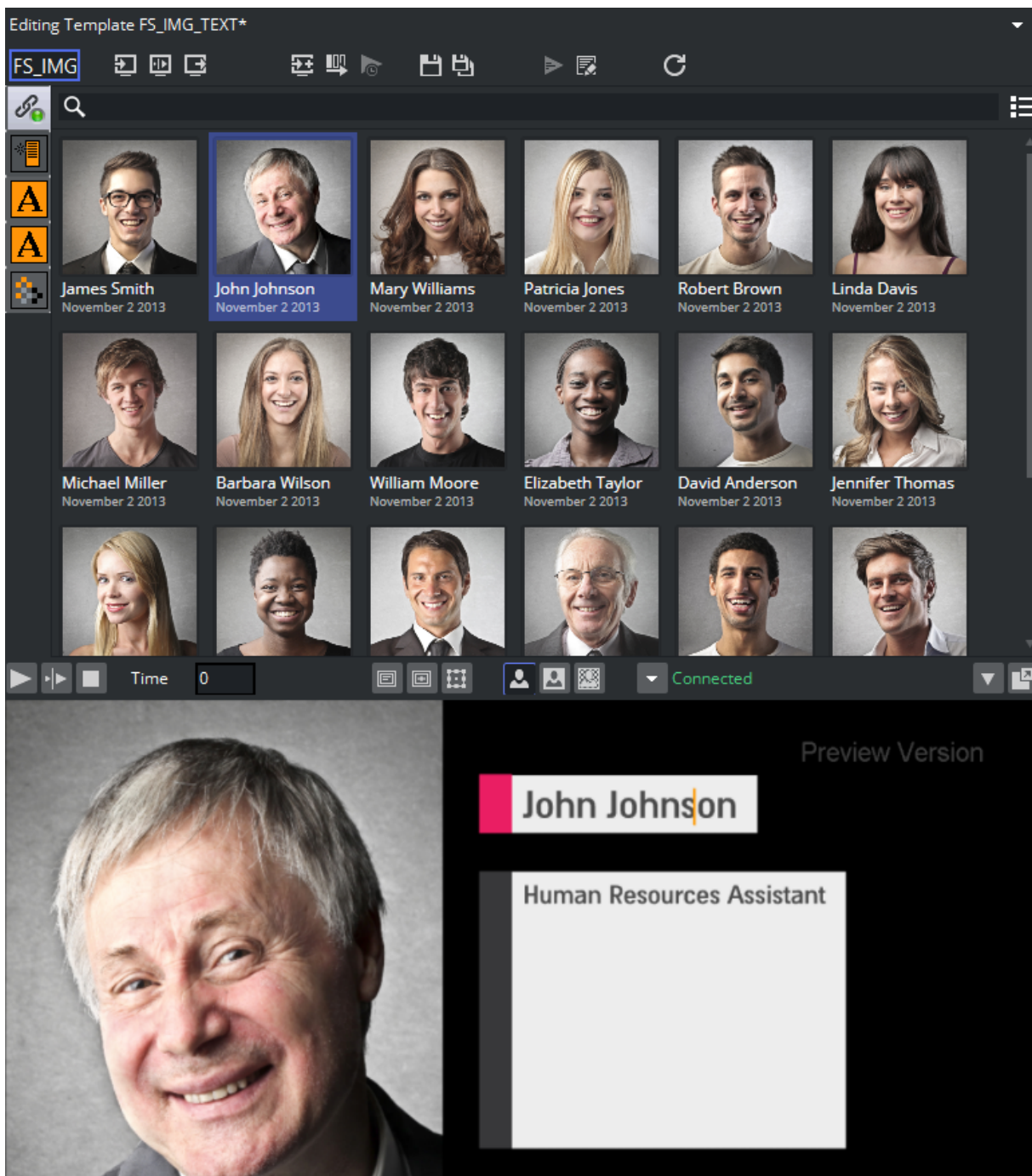
```
<link rel="..." href="http://..." type="..."/>
```

- Required in order to select the URL to external resources. This can be images, videos, text, and so on.
 - Limitation 1: The atom 1.0 specs allow multiple links with the same relation, but when linking a tabfield to a link in Viz Trio, only the first link is selected.
 - Limitation 2: Viz Trio can only use the `href` URL as the value. It cannot download the resource at the URL and use its contents as the value. The Viz Engine control plugin for the property has to be able to understand the URL and download the resource.

5.22.5 Feed Browsing

Given a valid Field Linking URL the elements can now be browsed and selected in the **Feed Browser** window. The example below uses the link:

http://d63hzfu0kdpt5.cloudfront.net/vizrt_training/vizrt-training-feed.xml



- **Authentication:**

The feed browser supports basic HTTP authentication. The server must return a HTTP header based on the HTTP standard, like this:

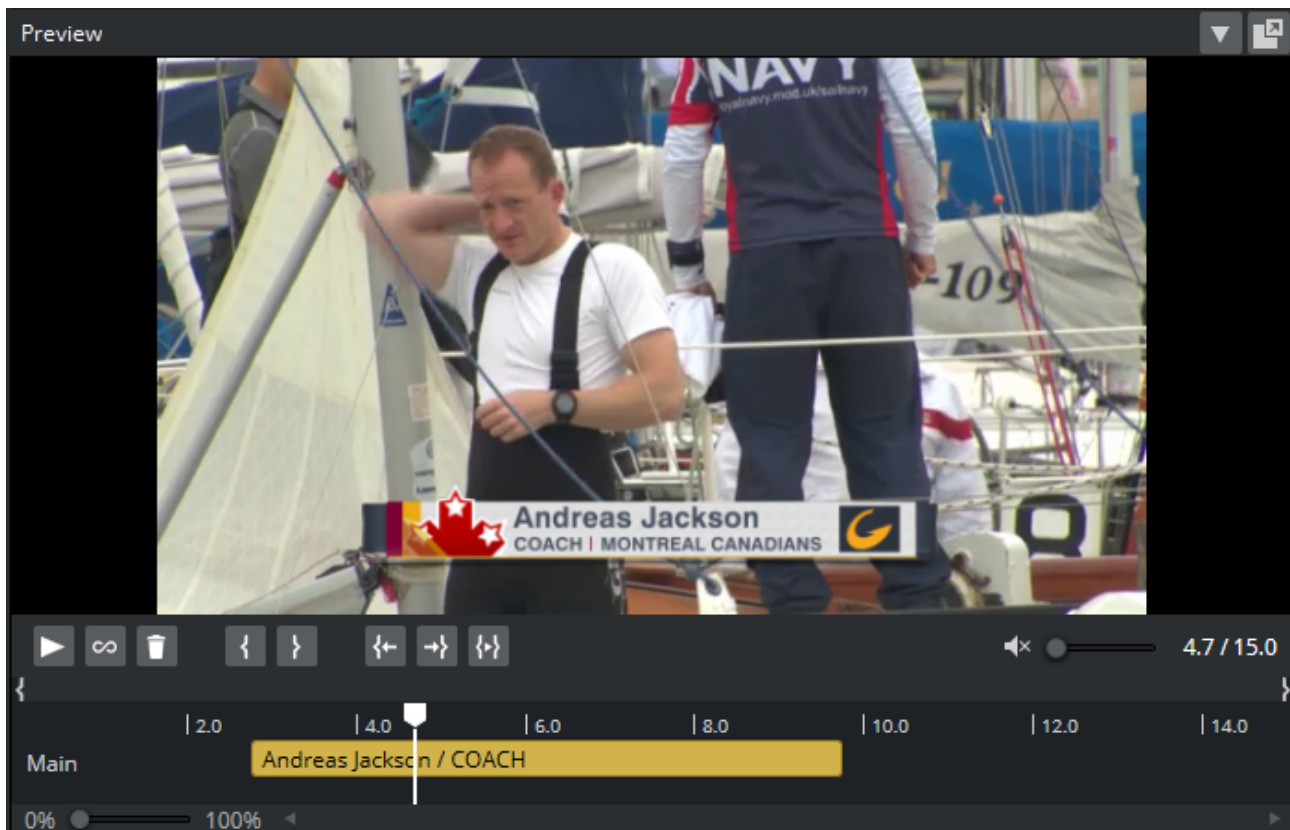
```
WWW-Authenticate: Basic realm="..."
```

- **Searching and Filtering:**

If the feed supports OpenSearch, the search box is enabled and all searches are done on the server. If not, the search box is a local text filter box.

5.23 Timeline Editor

Add graphics to a video timeline with the Timeline Editor.



Note: Creating video timelines is only supported for clips in a playlist, not the page list.

- **Search** for a video using the [Search Media](#) editor and open it in the Timeline editor.
- Drag a clip into a **playlist**.
- **Open** the clip.
- Drag **graphics** pages or playlist elements onto the timeline of the clip. The video preview displays snapshots of the graphics as overlays (fetched from Preview Server), and can be manually adjusted on the timeline.
- Remember to **Save** the video timeline.
- A story item containing clip and graphics references from the Timeline editor appears as a video composite group in the playlist. Issuing a [Take](#) command on the group triggers the video clip and timed graphics.

This section covers the following topics:

- [Timeline Editor Functions](#)
 - [Editor Control Bar Functions](#)
 - [Timeline Editor Functions](#)
 - [Multiple Tracks in the Timeline Editor](#)
 - [Timeline Editor Keyboard Shortcuts](#)
 - [Inline TimeCode Monitor](#)

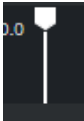
- [Working with the Timeline Editor](#)
 - [Adding Graphics to a Video Clip Timeline](#)
 - [Editing a Composite Element](#)
 - [Setting the Mark-in and Mark-out point](#)
- [Troubleshooting and Known Limitations](#)
 - [Known Limitations](#)

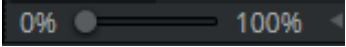
5.23.1 Timeline Editor Functions

Editor Control Bar Functions

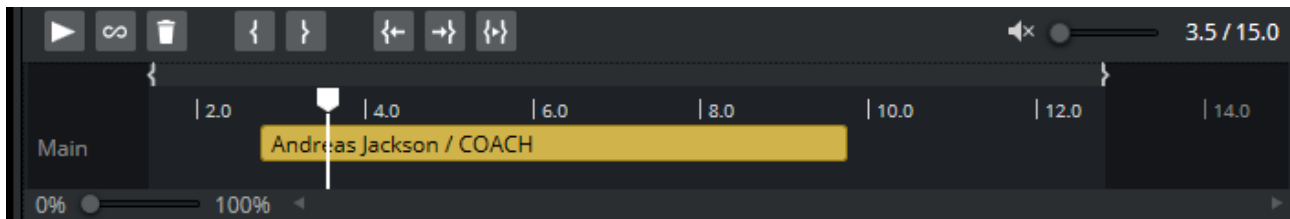
The Timeline editor is divided in two, a control bar and a preview window.

Timeline Editor Functions

Function	Description
	Play or Pause the clip.
	Adjust the local volume of the clip. This does not affect the volume when taken on the program engine.
	Delete the currently selected graphics from the timeline.
	If on, a clip loops between the clip in and out times. Looping affects the clip only in the playout mode.
	Scrub marker (cursor). Drag to scrub the clip.
	Set the mark-in our mark-out point to the current position of the cursor.
	Jump the cursor to the mark-in point.
	Play only the area between the mark-in and mark-out points.

Function	Description
	Drag the Zoom control to increase or decrease the time scale shown on the timeline.

Multiple Tracks in the Timeline Editor



- The Timeline editor supports multiple tracks for transition logic scenes. This allows graphics to overlap and play out correctly, but requires that all data elements are based on the same transition logic set.
- The tracks reflect the layers in the transition logic scene. A track is displayed in the Timeline editor if an element using that layer has been added.
- Graphics from the same layer should not be overlapped, and the Timeline editor indicates a possible conflict by coloring the graphics placeholders red.

Timeline Editor Keyboard Shortcuts

Key	Description
Mouse wheel	Zoom In/Out
CTRL + '+'	Zoom In
CTRL + '-'	Zoom Out
CTRL + 0	Reset Zoom
DELETE	Remove currently selected graphics from the timeline.
SPACE	Play/Pause
SHIFT + SPACE	Play only the area between the mark-in and mark-out points.
CTRL + ARROW	Move graphics in small steps back and forth on the timeline.
CTRL + SHIFT + ARROW	Move graphics in big steps back and forth on the timeline.
CTRL + ALT + ARROW	Increase or decrease duration by a small amount.

Key	Description
CTRL + ALT + SHIFT + ARROW	Increase or decrease duration by a large amount.
CTRL + L	Move the selected graphics next to the one to the left.
CTRL + R	Move the selected graphics next to the one to the right.
CTRL + ALT + L	Stretch the selected graphics next to the one to the left.
CTRL + ALT + R	Stretch the selected graphics next to the one to the right.
CTRL + H	Sets the video position to the start of the clip.
CTRL + E	Sets the video position to the end of the clip.
T	Sets a video to loop. (Note that this shortcut only works if the cursor is first placed in the timeline editor.)



Note: Media keyboard shortcuts for play, pause and mute should also work.

Inline TimeCode Monitor



In addition to the basic [TC Monitor](#) that opens in a separate window, an inline TimeCode Monitor becomes available above the local Preview panel in Viz Trio when playing a video. It displays the channel associated with the video and the video's name in this format: <channel-name>: <clip-name>.

It also displays a progress bar showing how much of the video has played. The progress bar is green when the video is playing, yellow when it is paused and red when 10 seconds remain in the video.

5.23.2 Working with the Timeline Editor

Adding Graphics to a Video Clip Timeline

1. Drag a video from the [Search Media](#) editor to a playlist.

Note: The video must be dragged to a playlist in order to add graphics and create a composite element. Graphics cannot be added to videos that are dragged to a page list.

2. Double-click the video item.
 - The item opens in the Timeline editor.

- If the video does not appear in the Timeline editor as expected, see [Troubleshooting and Known Limitations](#).
3. To add graphics, drag an item from the page list to the timeline. A graphics placeholder appears on the timeline.
 4. Use the mark-in/mark-out points to adjust the duration of the animation.
 5. *Optional:* Repeat steps 3 and 4 to add more graphics.

 **Note:** Although graphics may be overlaid on the timeline, graphics that use the same layer (front, middle, back or transition logic layer) do not play out correctly.

6. Click **Save** to create a new composite element, consisting of both the video and graphics.
 - The composite element is presented in the playlist with the video element as parent, and the graphics items as children.

Editing a Composite Element

1. First, create a composite element, consisting of both video and one or more graphics items, see [add graphics to a video clip timeline](#).
2. Double-click the composite element in the playlist.
 - The item opens in the Timeline editor.
3. Drag graphics to the timeline.
4. Click **Save as** in the upper right corner to update the composite element.

Setting the Mark-in and Mark-out point

1. In the Timeline editor, scrub the cursor to the desired frame.
2. Click the **Set Mark-in** or **Set Mark-out** button.
3. Click **Jump to Mark-in** or **Jump to Mark-out** to view the points.

5.23.3 Troubleshooting and Known Limitations

Known Limitations

- Video timeline elements are only supported in playlists, not the page list.
- Although graphics may be overlaid on the timeline, graphics that use the same layer (front, middle, back or transition logic layer) do not play out correctly. The Timeline editor always indicates a possible conflict by coloring the graphics placeholders red. To use overlapping graphics based on transition logic scenes, see [Multiple Tracks in the Timeline Editor](#).
- Current limitations on Media Sequencer cause the following behavior:
 - a. If you issue a **Take** on a timed group (1), and while that group is being played out issue a **Take** on graphics in another layer (2), Media Sequencer takes out the last taken element (2) when issuing the timed **Take Out** command (for the timed graphics).
 - b. Hence, it is currently not recommended to take other data elements on-air while a timed group is being played out.
- The Timeline editor may behave unexpectedly in some virtual machine or remote desktop environments.

6 Macro Commands and Events

This section contains reference information on the following topics:

- [Events](#)
- [Macro Commands](#)

6.1 Events

There are two types of events in scripts: *template scripts* with **template** events and *show scripts* with **show** events. Although these events look similar, they differ.

When used in scripts, events are only called if the specific event described by the event-handler occurs.

 **IMPORTANT!** Template scripts are loaded only when a page/template is *read*. Some template events are triggered only if the template or a page using the template is read.

This section covers the following topics:

- [Template Event Callbacks](#)
- [Show Event Callbacks](#)
- [Using the OnDelete Script Event](#)

6.1.1 Template Event Callbacks

- **OnContinue():** Called when the **Continue** button is pressed or a continue macro command is executed. Only triggered if the template or a page using the template is read.
- **OnInitialize:** Called when the **Initialize** button is pressed or an initialize macro command is executed.
- **OnBeforeSave():** This procedure is called before Viz Trio saves the current page. The script can cancel saving by returning false (*OnBeforeSave = false*).
- **OnSave:** Called when the **Save** button is pressed or a save macro command is executed.
- **OnBeforeTake():** This function is called before Viz Trio runs a take on the current page. The script can cancel the take by returning false (*OnBeforeTake = false*). Only triggered if the template or a page using the template is read.
- **OnTake():** Called when the **Take** button is pressed or a take macro command is executed. Only triggered if the template or a page using the template is read.
- **OnTakeOut():** Called when the **Take Out** button is pressed or a take out macro command is executed. Only triggered if the template or a page using the template is read.
- **OnUpdate():** Called when an update action is executed in Viz Trio (See keyboard shortcut list on control page).
- **OnUserClick():** Called when the user clicks the **Run macro** button or when the Viz Trio macro command [execute_script](#) is executed.
- **OnValueChanged(PropertyName,NewPropertyValue):** Used with template scripts. The event is called when a property in the page is changed, sending in the name of the property and the new value.
- **OnSocketDataReceived:** Called when socket data is received on the defined port. A socket object must be configured on the configuration page.
- **OnPropertyFocused(PropertyName):** Called when a tab-field editor gets focus in the Viz Trio user interface. It sends the property name into the function that uses the event.
- **OnShowVariableChanged(Name, Value):** This procedure is called when a global variable is changed or added. All Viz Trio clients (including the client where this command was issued) that have the same show open, receive the event when the command *show:set_variable* is used.

- **OnGlobalVariableChanged(Name, Value):** This procedure is called when a show variable is changed or added. All Viz Trio clients that are connected to the same Media Sequencer with the same show open, receive the event when the command *trio:set_global_variable* is used.

6.1.2 Show Event Callbacks

- **OnShowOpen():** Called when show is opened.
- **OnShowClose():** Called when show is closed.
- **OnRead():** Called when the **Read** button is pressed or a read macro command is executed.
- **OnRead(PageName):** Called when a specific page is read.
- **OnContinue(PageName):** Called when the **Continue** button is pressed or a continue macro command is executed.
- **OnInitializeShow():** Same as OnInitialize for template scripts.
- **OnCleanupRenderers():** Called when the *trio:cleanup_renderers* or *trio:cleanup_all_channels* commands are executed.
- **OnBeforeSave(PageName):** Called before a specific page is saved.
- **OnSave(PageName):** Called when a specific page is saved.
- **OnBeforeTake(PageName):** Called before a specific page is taken.
- **OnTake():** Called when the Take button is pressed or a take macro command is executed.
- **OnTake(PageName):** Called when a specific page is taken.
- **OnTakeOut:** Called when the Take Out button is pressed or a take out macro command is executed.
- **OnTakeOut(PageName):** Called when a specific page is taken out.
- **OnUpdate(PageName):** Called when a specific page is updated.
- **OnValueChanged(PageName, PropertyName, NewPropertyValue):** Used with show scripts to specify the page (PageName) that is called. The event is called when a property in a show's page is changed. It sends in the name of the page, the name of the property and the new value.
- **OnSocketDataReceived(data):** Called when data received on socket.
- **OnPropertyFocused(PageName, PropertyName):** Called when property focused.
- **OnCopy(OldName, NewName):** Called when a page is copied. The event is only available to show scripts.
- **OnCopy(OldName, NewName):** Called when page copied from old to new.
- **OnMove(OldName, NewName):** Called when a page is moved or renamed. The event is only available to show scripts.
- **OnDelete(PageName):** Called when a page is deleted. The script can abort deletion by returning false (*OnDelete = false*).



Note: This event is a function, hence, it can return a value. If you set it to *false*, it does not delete the page in question. If you set it to anything else or do not return a value, the page in question is deleted. The event is only available to show scripts (see [using the OnDelete script event](#)).

- **OnShowVariableChanged(Name, Value):** Called when show variable changed.
- **OnGlobalVariableChanged(Name, Value):** Called when global variable changed.



IMPORTANT! Events with the same name but different argument values are *not* the same. For example *OnRead()* and *OnRead(Pagename)* are not the same event.

6.1.3 Using the OnDelete Script Event

For example, the following script cancels the delete operation of any page named `2000`.

```
function OnDelete(PageName) if PageName = "2000" then      OnDelete = false  
else      OnDelete = true end if end function
```

The following are examples of commands that trigger this event:

- [Gui:copy_selected_pages_to_number](#)
- [Gui:copy_selected_pages_with_offset](#)
- [Gui:move_selected_pages_to_number](#)
- [Gui:move_selected_pages_with_offset](#)
- [Show:copy_pages_to_number](#)
- [Show:copy_pages_with_offset](#)
- [Show:delete_all_pages](#)
- [Show:delete_templates](#)
- [Show:move_pages_to_number](#)
- [Show:move_pages_with_offset](#)
- [Show:rename_page](#)
- [Page:delete_page](#)
- [Page:delete_pagerange](#)

See Also

- [Scripting](#)
- [Overview of Commands](#)

6.2 Macro Commands

6.2.1 Viz Trio Macro Language

The Viz Trio macro language lets you script many of the operations that are normally done in the interface. The general syntax is:

```
Command [argument]
```

i Info: Some commands take several arguments, such as command scaling that must have the x-, y- and z-axes specified. See the command list for more information.

Macro commands can be used in three ways:

- As part of a shortcut key using the [Keyboard Shortcuts and Macros](#) window.
- As part of a script using the [Script Editor](#).
- As part of an external application where commands are executed over a socket connection.

Overriding User Prompts

The macros you create will often use commands that usually trigger a dialog for user input. For example *page:delete 1000* will normally ask for user-confirmation of a deletion.

To override user prompts

You can override pop-up menu behavior with the following modes:

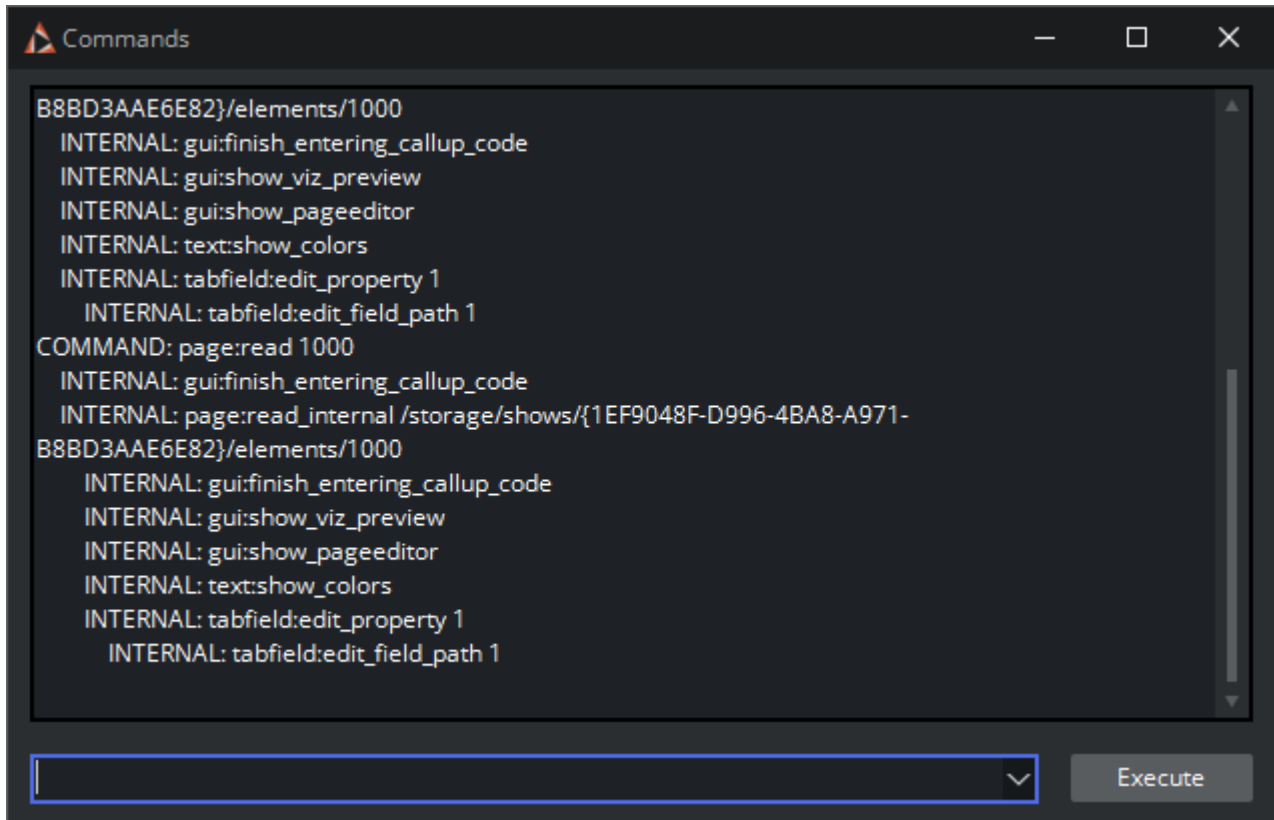
- **gui:set_silent_mode:** Controls whether to show dialogues to the user.
- **gui:set_interactive_mode:** Controls whether to show dialogues to the user.

This section covers the following topics:

- [Working with Macro Commands](#)
 - [Working with Macro Commands over a Socket Connection](#)
 - `trio:wait`
 - [Overview of Commands](#)
 - [Shortcut Keys](#)

6.2.2 Working with Macro Commands

Viz Trio Client Commands Window



The Viz Trio user interface basically uses the same macro commands as the scripting support uses. To help find the commands that are executed when different user interface operations are performed, use the commands window and look at which commands are sent.

The actual commands can be seen after the colon:

```
page:read 1000
```

Note: The Viz Trio Client Commands window can be very helpful when learning the system and testing out macro commands. Enter commands in the text field at the bottom and click **Execute** to run them.

This section covers the following topics:

- [Viz Trio Client Commands Window](#)
- [To Enable the Viz Trio Client Commands Button](#)
- [To Open the Viz Trio Client Commands Window](#)
 - [Apropos and Help Commands](#)
 - [Example I: Apropos](#)

- [Example II: Help](#)
- [Show, Context and Tab-field Commands](#)
 - [Show Variables](#)
 - [Concept, Context and Variant Variables](#)
 - [Commands](#)
 - [Tab-field Variables](#)

To Enable the Viz Trio Client Commands Button

Click Viz Trio's **Config** button, and in the **User Interface > User Restrictions** section, clear the **Call Viz Trio Commands** check box.

To Open the Viz Trio Client Commands Window

Click the Viz **Trio Commands** button in the lower-right corner of the application window.

Apropos and Help Commands

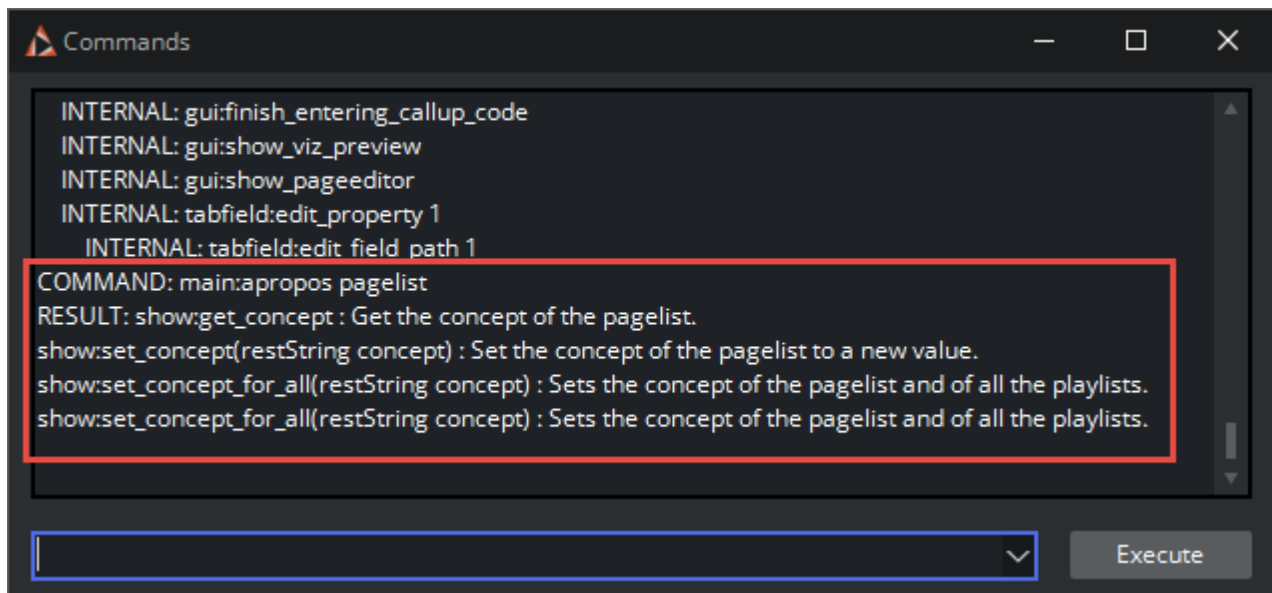
The Viz Trio Client Commands window contains useful features for searching for information about a command.

Use keywords **apropos** and **help** to perform look-ups in the available set of commands:

- **apropos**: Searches command names and descriptions for a string.
- **help**: Shows help for the specified command, or lists commands.

Example I: Apropos

The **apropos** command does a free text search for the given text in the macro command documentation.



```
COMMAND: main:apropos context_variable RESULT: show:get_context_variable(string
variable) : Get the show-context-variable. show:set_context_variable(string variable,
restString value) : Set the show-variant to a new value.
```

Example II: Help

The **help** command provides help on a specific macro command.

```
COMMAND: main:help read_template RESULT: Parameters: (string templateid) Description:
Read a template if concepts and variants are enabled. If not, read a normal template.
```

Show, Context and Tab-field Commands

Show variables are global as all variables are stored on the Media Sequencer. This means that all Viz Trio applications that use the same Media Sequencer can set and get variables, and any script within a show can access the variable.

Show Variables

Use the Viz Trio Client Commands window to set show variables. The variables can be used by scripts as a means of storing intermediate values such as counters.

Commands:

- show:set_variable: Set the value of a show variable
- show:get_variable: Get the value of a show variable

Example commands:

```
COMMAND: show:set_variable MyShowVar Hello World! COMMAND: show:get_variable
MyShowVar RESULT: Hello World!
```

Script example:

```
Sub OnUserClick()    dim myLocalShowVar    myLocalShowVar =
TrioCmd("show:get_variable MyShowVar") MsgBox("Variable value is: " & myLocalShowVar)
myLocalShowVar = InputBox("Enter a new Value:", "New Value")
TrioCmd("show:set_variable MyShowVar " & myLocalShowVar) End Sub
```

Concept, Context and Variant Variables

Use the Viz Trio Client Commands window to set context variables. Context variables are different than [Show Variables](#) as context variables are used to select the context a page is taken on-air with. The following contexts are available to Viz Trio users:

- **Concept:** A concept can be News or Sports or any other concept. Graphics in the same concept are created as individual scenes belonging to a specific concept.

- **Variant:** A concept can have different variants of the same scene. This can be a scene with the same tab-fields, but displayed as a top or lower third.
- **Context:** Additional user-defined contexts can be used to further differentiate a concept and its variants. For example the context Platform could have SD, HD, Mobile phones, web and so on defined within the concepts News and Sport.

Commands

If a concept, context or variant for a show is set using one of the show commands, for example *show:set_variant top*, this overrides the selection done in the page list's user interface, marking the variant with square brackets. Issuing the same command, without the variant name (*show:set_variant*) resets the variant to the initial selection done using the page list user interface.

 **Note:** A get command cannot retrieve a variable's value if a set command has not been issued first.

Available commands:

- **show:enable_context():** Enables concept and variant for the current show. This also separates templates and pages into their own lists.
- **show:set_concept(restString concept):** Sets the show's concept to a new value.
- **show:get_concept():** Gets the show's concept if set by a set command.
- **show:set_context_variable (string variable, restString value):** Sets the show's context to a new value.
- **show:get_context_variable (string variable):** Gets the show's context variable if set by a set command.
- **show:set_variant (restString variant):** Sets the show's variant to a new value.
- **show:get_variant():** Get the show's variant if set by a set command.
- **playlist:set_concept(string value):** Sets the concept for a show playlist.

 **Note:** The *set_variant* and *get_variant* commands are aliases for the context commands when setting and getting variants, and are not applicable in other contexts.

Example commands:

```
COMMAND: show:set_context_variable concept Sport COMMAND: show:set_context_variable
variant Lower COMMAND: show:set_context_variable platform HD COMMAND:
show:get_context_variable concept RESULT: Sport COMMAND: show:get_context_variable
variant RESULT: Lower COMMAND: show:get_context_variable <contextname> RESULT: HD
```

A parameter must first be set using the commands before it can be shown using the commands. Output normally differs according to the scene design, all the output is therefore not shown.

 **IMPORTANT!** Command parameters are case sensitive.

Tab-field Variables

Use the Viz Trio's commands window to set custom properties for tab-fields. The property can be used for scripting to set intermediate properties for tab-fields.

Commands:

- `set_custom_property` (string tabfieldName, string value)
- `get_custom_property` (string tabfieldName)
- `show_custom_property_editor_for_current_tabfield`

Example commands:

```
COMMAND: tabfield:set_custom_property 01 string COMMAND: tabfield:get_custom_property  
01 RESULT: string
```

Working with Macro Commands over a Socket Connection

Viz Trio can be controlled using custom made control applications over a TCP/IP socket connection on port 6200 (the port number can be overridden, see [Command line Parameters](#)). This feature lets you use macro commands that can trigger actions in Viz Trio.

This section covers the following topics:

- [Sending Macro Commands](#)
 - [Configure Viz Trio for Socket Connections](#)
 - [Connect to Viz Trio Using Telnet](#)
 - [Send Commands to Viz Trio Using Telnet](#)
 - [Enable Telnet for Windows 7](#)
- [Unicode Support](#)
 - [Toggle the Socket Encoding](#)
- [Escape Sequences](#)
 - [Escape Sequences](#)
- [Error Messages](#)

Sending Macro Commands

This section shows how to establish a connection with Viz Trio over a socket connection in order to send Macro Commands to control Viz Trio. Telnet is used as an example below.

Configure Viz Trio for Socket Connections

1. Open the [Configuration Window](#) and select [Socket Object Settings](#).
2. Set the following in the menu:
 - Set Socket type to Server socket.
 - Set Text encoding to UTF-8
 - Check **Autoconnect** and set Socket Host to localhost and Socket Port to 6200.
3. Click **OK**.

Connect to Viz Trio Using Telnet

1. Open Telnet by clicking **Start**, typing Telnet in the Search box, and then clicking **OK**.
2. Type telnet localhost 6200 and press Enter.

 **Note:** Windows 7 users must enable Telnet as it is disabled by default.

Send Commands to Viz Trio Using Telnet

1. Create a page in Viz Trio with the page number 1000.
2. Open a telnet connection to Viz Trio, type `page:read 1000` and press Enter.

Enable Telnet for Windows 7

1. Click **Start**, click Control Panel, select Programs, and then click **Turn Windows features on or off**. Enter the administrator password or confirmation if prompted.

2. In the Windows Features dialog box, select the **Telnet Client** check box.
3. Click **OK**. Installation may take several minutes.

Unicode Support

Commands that can set and get text values (such as *set_value* and *getproperty*) support the full set of Unicode characters. The internal format is UTF-16 (WideString). However, when communicating with Viz Trio over a socket connection on port 6200, an 8-bit string must be used, and the default (and preferred) encoding is therefore UTF-8. It's also possible to use latin1 (ISO-8859-1, western european) encoding.

Toggle the Socket Encoding

- Send one of the following commands to the socket: **socket_encoding latin1** or **socket_encoding utf8**.

 **Note:** The `socket_encoding` command is not a macro command, and can only be used with socket communication.

Escape Sequences

A socket command is terminated by a new line. To support setting and getting multi-line parameters and results, some characters must be escaped, i.e. represented by sequences of other characters. Backslash is used as the escape character for socket commands.

See the table below for supported escape sequences.

Escape Sequences

Escape Sequence	Meaning
<code>\n</code>	New line
<code>\r</code>	Carriage return
<code>\xHH</code>	Two-digit hexadecimal character code.
<code>\</code>	Backslash (\)

Error Messages

When controlling Viz Trio over a socket connection, you can enable error messages using the `commandserver` command:

```
commandserver:enable_error_messages true commandserver:enable_error_messages false
```

 **Note:** Typing `commandserver` is optional.

Enabling error messages sends error messages to the connected client and displays them on the interface (on a per socket connection). The client receives all error messages that are normally only displayed on the Viz Trio user interface.

 **Note:** All errors are sent. For example, if an operator attempts to read a non-existing page, this results in error messages being sent to all clients connected over a socket connection.

Errors have the following format:

```
ERROR: <Error message>
```

If error handling is disabled, the client still sees errors caused by commands not being found and commands that return an error.

This command is only implemented for the command server on port 6200, and is not related to [Socket Object Settings](#) in the [Configuration Window](#).

trio:wait

The `trio:wait` macro command is useful to hold execution flow for some time. The waiting time is specified as a mandatory argument and is set in milliseconds:

```
trio:wait 5000
```

Unlike the `trio:sleep` command, `trio:wait` does not block other commands execution.

It is recommended to use `trio:wait` in all cases, where a delay in script or macro execution is required.

`trio:sleep` is now considered deprecated while kept for backward compatibility, and specific cases.

- [Controlling Post-Execution Waiting Times for Macro Commands](#)
- [Trio Folder](#)
- [Adjusting Post-Execution Waiting Times](#)
- [Overall Guidelines](#)

Controlling Post-Execution Waiting Times for Macro Commands

Various setups may have their own latency in terms of how much time it takes to finish execution of a certain command.

It is a common practice to use the `trio:sleep` command to wait until the right time, while executing a macro or a script. A typical application in the macro would look like this:

```
===== MyMacro1 =====
...
page:read 1002
trio:sleep 2000
...
=====
```

This code can be interpreted as "read the page 1002 and wait for 2 seconds", assuming this would be enough time for the page to be read.

Each time a user adjusted the waiting time period, the corresponding lines would have to be modified for all macros and scripts, in all shows (for example, changing the 2000 value to 3000).

Since Viz Trio 4.4, there is the possibility to control waiting times from one place, without modifying existing macros and scripts.

Trio Folder

In the Viz Trio folder (*C:\Program Files\Vizrt\Viz Trio*), you can find the `trio.config` file, an XML configuration file that can be used for centralized control over command waiting times.

Edit the file's content, save it, and then restart Viz Trio to apply the new configuration.

Using the example above, this is the following configuration:

```
<?xml version="1.0"?>
<configuration>
  <commands>
    <command name="page:read" delay="2000"/>
  </commands>
</configuration>
```

The configuration specifies that any call to the `page:read` command from any macro or script, is followed by a wait time of 2000 milliseconds. Therefore, if we use this configuration with the macro example mentioned previously, the overall waiting time is 4000 milliseconds (2000 milliseconds done automatically by Viz Trio, and an additional 2000 milliseconds requested by `trio:sleep`).

It is recommended to remove the direct waiting command, so the updated MyMacro1 resembles the following:

```
===== MyMacro1 =====
...
page:read 1002
...
=====
```

Adjusting Post-Execution Waiting Times

Any Viz Trio command's post-execution waiting time can be adjusted in the configuration file.

Info:

- Specify the full command name (with the namespace format `namespace:command`) in the configuration file.
- Viz Trio uses a "non-intrusive" method for post-execution waiting (running the `trio:wait` command each time it needs to wait). Please refer to the `trio:wait` description for more information on its behavior.
- By default, when a command is not mentioned in the `trio.config` file, the post-execution waiting time is zero. There is no benefit using negative delay values.
- If the `trio.config` file content is damaged and violates XML rules, it is ignored by Viz Trio.

Overall Guidelines

1. Identify which commands require waiting time adjustments.
2. Modify the content of the `trio.config` file in a text editor, to include a separate command tag for all necessary commands:

```
<command name="fullcommandname1" delay="2000"/>
<command name="fullcommandname2" delay="500"/>
<command name="fullcommandname3" delay="1000"/>
```

3. Save the `trio.config` file.
4. Modify macros and/or scripts to remove `trio:sleep` calls after these commands. Ensure your changes to macros and/or scripts are saved.
5. Restart Viz Trio.

 **Note:** If you have several scripts and don't want to modify them, you can still set the waiting time for commands in `trio.config`. However, you may need to be more conservative with the waiting time values set in the configuration, as they are added to times specified in existing `trio:sleep` calls.

Overview of Commands

The commands are grouped as follows:

- [Channelcontrol](#)
- [dblink](#)
- [Gui](#)
- [Macro](#)
- [Main](#)
- [Page](#)
- [Playlist](#)
- [Proxy](#)
- [Rundown](#)
- [Script](#)
- [Scroll2](#)
- [Settings](#)
- [Show](#)
- [Sock](#)
- [Tabfield](#)
- [Table](#)
- [Text](#)
- [Viz Trio](#)
- [Util](#)
- [Viz](#)
- [vtwtemplate](#)

Channelcontrol

Command Name and Arguments	Description
<code>add_channel(restString channelName)</code>	Adds a new channel to the active profile. The named channel must not exist.
<code>get_active_profile</code>	Returns the name of the active profile.
<code>get_preview_channel</code>	Gets the name of the currently configured Viz preview channel. Returns an empty string if no Viz preview channel is defined.
<code>get_program_channel</code>	Gets the name of the currently configured Viz program channel. Returns an empty string if no Viz program channel is defined.

Command Name and Arguments	Description
get_video_preview_channel	Gets the name of the currently configured video preview channel. Returns an empty string if no video preview channel is defined.
get_video_program_channel(restString channelName) @TODO check args	Gets the name of the currently configured video program channel. Returns an empty string if no video program channel is defined.
list_profiles	Returns a list of existing profiles, separated by line breaks.
list_video_channels	Returns all video channels of the current profile as comma separated list.
list_viz_channels	Returns all Viz channels of the current profile as comma separated list.
remove_channel(restString channelName)	Removes the channel with the given name from the active profile. The channel name must be valid.
set_active_profile(restString profileName)	Change the active channel profile.
set_asset_storage (string host-name, restString storage-name)	Sets the publishing point to be used for all handlers in the active profile. The storage name is the user visible title for the asset storage.
set_channel_hosts(string channelname, restString hostlist)	Changes the list of Viz hosts for a channel. The referenced channel must exist. The list of hosts can be space/comma/semi-colon separated.
set_preview_channel(restString channelName)	Changes the Viz preview channel. The channel name must be valid. If empty the Viz preview channel is undefined.
set_program_channel(restString channelName)	Changes the Viz program channel. The channel name must be valid. If empty the Viz program channel is undefined.
set_video_channel_hosts(string channelname, restString hostlist)	Changes the list of video hosts for a channel. The referenced channel must exist. The list of hosts can be space/comma/semi-colon separated.

Command Name and Arguments	Description
set_video_preview_channel(restString channelName)	Changes the video preview channel. The channel name must be valid. If empty the video preview channel is undefined.
set_video_program_channel(restString channelName)	Changes the video program channel. The channel name must be valid. If empty the video program channel is undefined.
wait_vizcmd_to_channel(string channel, restString cmd)	Sends a Viz Engine command to a named channel and waits for the returned answer.

dblink

Command Name and Arguments	Description
link_database (string property key, string connection name, string table name)	Links a property of a page to a table of a database connection. Example 1: <pre>TrioCmd("dblink:link_database 1 ExcelTest" "'['Scalar Tests\$']'")</pre> Example 2: <pre>{{TrioCmd("dblink:link_database 1 ExcelTest" "'['Table Tests\$']'")}}</pre> 1 refers to the Control List, Chart or Text's Field Identifier that in Viz Trio can be identified as the template's tab field 1. {{ExcelTest refers to the name of the configured database connection made in the Database Config window. (See Creating a New Database Connection). Scalar Tests and Table Tests refers to the table that is chosen in order to select the lookup column.
map_database_column (string table key, string property column key, string database field name)	Maps a column of a table property to a database field. Example: <pre>TrioCmd("dblink:map_database_column 1 3 Name")</pre> . 1 refers to the Control List's or Control Chart's Field identifier property that in Viz Trio can be identified as the template's tab field 1. 3 refers to the Control Text Field identifiers in the scene that make out column number 3 that in Viz Trio can be identified as the table column in the page editor. Name refers to the column the data should be read from.

Command Name and Arguments	Description
select_value (string property key, string value column, string key column, string key value)	Selects a single value (scalar linking) of a database and links it to the property. The value is selected from the <value column> where the <key column> matches the <key value>.Example: <pre>TrioCmd("dblink:select_value 1 Color Key 2")</pre> 1 refers to the Control Text's Field identifier property that in Viz Trio can be identified as the template's tab field 1. Color refers to the column the data should be read from. Key refers to the lookup column defined in the Database Config window. (See Creating a New Database Connection). 2 refers to the row the data should be read from.
store_to_database (string property key)	Stores a linked property of a page to its value in the database.Example: <pre>TrioCmd("dblink:store_to_database 1")</pre> 1 refers to the Control Text's Field identifier property that in Viz Trio can be identified as the template's tab field 1.
update_from_database (string property key)	Updates a linked property of a page with its value in the database.Example: <pre>TrioCmd("dblink:update_from_database 1")</pre> 1 refers to the Control Text's Field identifier property that in Viz Trio can be identified as the template's tab field 1.
use_custom_sql (string property key, restString custom SQL)	Uses a custom SQL clause to select the database value. An empty SQL clause disables the usage of custom SQL.Example: <code>TrioCmd("dblink:use_custom_sql 1 SELECT Text FROM ['Table Tests\$']")</code> 1 refers to the Control List's or Control Chart's Field identifier property that in Viz Trio can be identified as the template's tab field 1. Text refers to the column the data should be read from. Table Tests refers to the database table (i.e. a single spreadsheet in Microsoft Excel).

Gui

Command Name and Arguments	Description
<code>add_view (title min max elementTypeList)</code>	Creates a new view. If called without any parameters, the PageView dialog is shown. <code>Min</code> and <code>max</code> may optionally be specified to filter elements with numeric names. Elements with non-numeric names are always included. <code>elementTypeList</code> is an optional space-separated list of types of elements to include in the view. Possible values are: pages stills videos empty_groups. If none are specified, they are all included. Example 1: <code>gui:add_view "The Title" 2000 3500 pages stills</code> . This shows pages and stills in the range between 2000 and 3500. Videos are not shown and empty groups are hidden. Example 2: <code>gui:add_view title2 videos</code> . This shows all videos, nothing else.
<code>artist_mode</code>	Enter design mode (Viz Artist).
<code>browse_image</code>	Browse local file system for images.
<code>change_path</code>	Shows the directory selector.
<code>clear_page_field</code>	Clears the content of the page edit's text field.
<code>change_path</code>	Opens the browser for either shows or external playlists.
<code>copy_selected_pages_to_number(res tString destnumber)</code>	Copy the selected pages to number or show a dialog if no number is supplied.
<code>copy_selected_pages_with_offset(res tString destnumber)</code>	Copy the selected pages with an offset or show a dialog if no number is supplied.
<code>design_mode</code>	Enter Viz Trio design mode.
<code>edit_begin_time</code>	Shows the in place editor for the selected page.
<code>edit_end_time</code>	Shows the in place editor for the selected page.
<code>edit_group_name</code>	Shows the in place editor for the selected page.
<code>edit_page_program_channel</code>	Shows the in place editor for the selected page.

Command Name and Arguments	Description
edit_scroll	Show the scroll editor.
edit_template(restString templateName)	Edit the specified template in a dialog. This only works for templates in concept-enabled shows.
error_message (restString message)	Display an error message in the status bar. The message is also logged.
find	Shows the find dialog.
focus_active_view	Set the keyboard focus to the active page view.
focus_next_view	Activate the next page view.
focus_prev_focused_view	Focus the previous focused view.
focus_prev_view	Activate the previous page view.
get_callup_code	Gets the value of the callup code window.
get_selected_pages	Retrieves the currently selected pages as a space separated list.
goto_config	Opens Viz Trio's Configuration Window .
hide_timecode_monitor	Hides the TimeCode monitor. See also show_timecode_monitor .
hit_test(x:integer, y:integer)	Test if a tab field exists on the X,Y coordinate.
load_image_from_viz	Show the viz image browser for the current tab field.
log_message(restString message)	Add a message to the logfile.
move_selected_pages_to_number(re stString destnumber)	Move the selected pages to number or show a dialog if no number is supplied.
move_selected_pages_with_offset(re stString destnumber)	Move the selected pages with an offset or show a dialog if no number is supplied.
open_playlists	Shows the directory selector for the external playlists. Typically MOS playlists.

Command Name and Arguments	Description
open_shows	Opens the show browser.
open_effect_dialog	This command opens the effect dialog. It does not open if the current page isn't set.
play_video	Play video in video preview.
playout_mode	Enter playout mode.
post_render	Opens the post render editor.
prepare_snapshot	Open the prepare snapshot editor.
print_selected_pages	Print the selected pages in the active view.
read_page_following_last_taken	Read the page succeeding the page that was last taken.
reimport_selected_pages	Reimport the selected pages in the active view.
search_video (restString search_text (optional video id))	Search for video.
select_last_taken_page	Select the page that was last taken.
select_nextpage	Select the next page in the active view.
select_prevpage	Select the previous page in the active view.
set_autopreview(boolean autopreview)	Sets the auto-preview state (single-click read) for page views.
set_bounding_box(visible:boolean)	Toggles the bounding boxes for the elements in the preview (same as the BB button).
set_column_visible(string columnname, boolean active)	Sets the visibility of the column. Applies to all pageviews.
set_focus_to_editor	Sets focus to the currently active tab field editor.
set_font (restString fontname)	Set the default GUI font.

Command Name and Arguments	Description
set_interactive_mode(boolean active)	Sets interactive mode which controls whether to show dialogs to the user or not.
set_run_script_button_enabled(boolean isEnabled)	Sets the run script button to enabled/disabled.
set_safearea(visible:boolean)	Toggle safe area.
set_silent_mode(boolean active)	Sets silent mode which controls whether to show dialogs to the user or not.
set_statusbar_color(int R, int G, int B)	Sets the color of the statusbar based on the RGB values given.
set_titlearea(visible:boolean)	Toggles the title area in the preview window.
set_mam_service_document_uri [uri [username password]]	The parameters are optional. If no parameters are provided the command should disable the Viz One configuration. Otherwise it enables the configuration if necessary and sets the new URI and possibly also the credentials if those parameters are provided.
set_media_search_credentials username password	This command changes the credentials used for when doing media searches on Viz One.
show_viz_preview	Show Viz preview.
show_custom_prop_column(boolean value)	Shows or hides custom property column in tab field list.
show_custom_property_editor_for_current_tabfield	Opens the inplace editor for the custom property of the current tab-field if the custom column in the tab-field tree is visible.
show_export_dialog	Show the export show dialog.
show_export_selected_page_dialog(string showPath)	Display the selected pages export show dialog. <i>showPath</i> parameter is optional. It is almost same as the Export Show feature but it exports only selected pages & all it's associated items.
show_external_cursor(boolean value)	Show or hide the external cursor in the page view.

Command Name and Arguments	Description
show_feed_streamer_moderation	Opens the feed streamer moderation tool if Viz SocialTV is installed.
show_import_dialog	Show the import show dialog.
import_viz_archive	Initiate the viz archive import process. This triggers the file open dialog to select the viz engine archive(.via) file from disk.
show_pageeditor	Shows the page editor.
show_pages_in_rundown(boolean doShow)	Shows or hides pages in the rundown for the current show.
show_pageview	Show the page view.
show_playlist (string refPath, boolean requestMos, restString description)	Shows the playlist with the specified path or name. Can choose to request a MOS playlist. Description is optional and defaults to the playlist name.
show_scripteditor	Shows the script editor.
show_scroll_template_creator	Shows the scroll template creator view.
show_settings	Open the show properties view.
show_search_media	Shows the search media view.
show_search_video	Shows the search video view.
show_shortcut_list	Show the view of local shortcuts and macros.
show_showscript_editor	Shows the show script editor.
show_status_icons(boolean value)	Shows or hides status cells in the callup grid.
show_templates_in_rundown(boolean doShow)	Shows or hides templates in the rundown for the current show.
show_timecode_monitor	Shows the TimeCode Monitor. See also hide_timecode_monitor .

Command Name and Arguments	Description
show_triocommands	Toggles the Viz Trio Commands window.
status_message(restString message)	Display a text message in the status bar. The message is also logged.
take_callup	Take the current callup number directly, without a read.
take_snapshot(string path, string handlerName)	Takes snapshot from the specified handler in RGB format.
take_snapshot_rgba(string path, string handlerName)	Takes snapshot from the specified handler in RGBA format.
toggle_design_mode	Toggle between Viz Trio design and playout mode.
toggle_error_message_window	Toggle the visibility of the error message window.
toggle_hide_nameless_groups(visible:boolean)	Toggle hide/show groups with no name.

Macro

Command Name and Arguments	Description
export_macros_to_xml(string filename)	Exports the macro commands to an XML file.
export_macros_shortcuts_to_xml (string filename)	Exports shortcuts information of global macro commands to an XML file.
import_macros_shortcuts_from_xml (string filename)	Imports shortcuts information of global macro commands from an XML file.

Main

Command Name and Arguments	Description
apropos(search:string)	Search command names and descriptions for a string.
call_macro(restString name)	Call (invoke) a named macro.

Command Name and Arguments	Description
<code>define_local_macro(string name, restString commands)</code>	Define a named, local macro.
<code>define_local_script(string name, restString commands)</code>	Define a named, local script.
<code>define_macro(string name, restString commands)</code>	Define a named, global macro.
<code>define_script(string name, restString commands)</code>	Define a named, global script.
<code>delete_global_macro(name:string)</code>	Delete a global macro.
<code>delete_local_macro(name:string)</code>	Delete a local macro.
<code>get_global_macros</code>	List the names of global macros (separated by new lines).
<code>get_local_macros</code>	List the names of local macros (separated by new lines).
<code>get_shows</code>	Get the names of all the shows in the client.
<code>get_version</code>	Get a version string.
<code>help(command:string)</code>	Shows help for the specified command, or lists commands.
<code>macro(macro)</code>	Runs the specified macro command.

Page

Command Name and Arguments	Description
<code>arm(pageName:string, channelName:string)</code>	Arms the specified page on the specified channel. If channelName is empty, the default channel is used.
<code>arm_current(channelName:string)</code>	Arms the currently read page on its specified program channel. If channelName is empty, the default channel is used.

Command Name and Arguments	Description
arm_selected	Arm the selected page.
cancel_refresh_linked_data(page name (optional))	Cancel refreshing of linked data, started with the 'page:refresh_linked_data_async' command.
change_channel("string FromChannel, string ToChannel, boolean All)	Switch from "FromChannel" to "ToChannel". "All" changes all elements using "FromChannel".
change_description(string pageNr, restString desc)	Sets the page description.
change_template(string templateName)	Changes the template of the selected pages. Empty parameter("") take the number in the callup field and use that as the template ID.
change_template_for_pages(string templateName, restString pages)	Changes the template of the given pages.
change_thumbnail_uri (string pageName, restString newThumbnailURI)	Changes the thumbnail URI for the referenced page. The URI and the page name must be valid.
change_variable(string variable, restString value)	Change the variant for the currently edited page.
continue(pageName, channelName)	Only applicable to videos. Continues a video on a named channel. If channel is not mentioned it plays the video on its designated channel. If the clip name is not mentioned the current clip is continued. Such as, page:continue 1000 A continues the clip "1000" on channel "A" while just page:continue 1000 would continue clip "1000" on its designated channel.
cue(pageName, channelName)	Cues a page on a named channel. If channel is not mentioned it cues the page on its designated channel. If the page name is omitted the current page is cued. Such as, page:cue 1000 A cue the page "1000" on channel "A" while just page:cue 1000 would cue page "1000" on its designated channel.

Command Name and Arguments	Description
<code>cue_append(pageName, channelName)</code>	Only applicable to videos. The command interrupts the current playing clip and immediately starts playing the selected video on the named channel. Such as, <code>page:cue_append 1000 A</code> cue the video "1000" on channel "A" while just <code>page:cue_append 1000</code> would cue video "1000" on its designated channel.
<code>cut(pageNumber, channelName)</code>	Cuts a page on the named channel. If channel is omitted it cuts the page on its designated channel. Such as, <code>page:cut 1000 A</code> cuts the page "1000" on channel "A" while just <code>page:cut 1000</code> would cut page "1000" on its designated channel.
<code>delete_page</code>	Delete selected page(s) in active view.
<code>delete_pagerange(int start, int end)</code>	Deletes the page inside from the begin range to (including) the end range parameter.
<code>direct_continue</code>	Continue selected page in active view.
<code>direct_cut</code>	Cut selected page in active view.
<code>direct_take</code>	Take selected page in active view.
<code>direct_takeout</code>	Take out selected page in active view.
<code>direct_cue</code>	Cue the selected video on the video program channel
<code>direct_cue_append</code>	Cue append the selected video on the video program channel
<code>direct_pause</code>	Pause the currently playing video on the video program channel
<code>export_property(string key, string filename)</code>	Export a property value to file (UTF-8 encoded).
<code>fire(channelName:string, action:string, clearAfterFire:boolean)</code>	Fire a channel with the given action (take, continue, out, and so on). If <code>clearAfterFire</code> is true, the armed element on the channel is un-armed after fire.

Command Name and Arguments	Description
fire_all(action:string, clearAfterFire:boolean)	Fire all armed channels with the given action (take, continue, out, and so on). If clearAfterFire is true, the armed elements on the channels are un-armed after fire.
get_current_propertykey	Get the current property-key.
get_current_tabfield_name	Get the name of the current tab field.
get_layers(string page)	Get the layer name(s) the combination or transition logic page is based on, hence, this command is not valid for pages based on standalone scenes (returns an empty string).
get_program_pagechannel(restString pageid)	Gets the program channel for the page.
get_property(string name)	Get the value of the specified tabfield property.
get_tabfield_names (optional: pageName)	Retrieves the tab-field names of the given page or the current page if no page is specified. The returned list is space separated.
get_tabfield_count	Gets the number of tabfield on the current page.
get_template_description(string page)	Gets the template description for the page specified.
get_viz_layer(restString page)	Gets the viz layer for a page. If no page name is supplied, current page is used. Only valid for scene-based pages.
getpagedescription	Get the description of the current page.
getpagename	Get the name of the current page.
get_property_keys (pageName optional)	Retrieves the property keys of the given page or the current page if no page is specified. The returned list is space separated.
get_variable(string variable)	Retrieves the value of a variable of the currently read page.
getpagetemplate	Get the template of the current page.

Command Name and Arguments	Description
<code>import_property(string key, string filename)</code>	Import a property value from file (UTF-8 encoded).
<code>load_page_data(filename:string (optional))</code>	Load page data from XML.
<code>pause(pageName, channelName)</code>	Only applicable to videos. Pauses a named video on a named channel. If channel is not mentioned it pauses the element on its designated channel. If the video name is not mentioned then current video is paused. Such as, <code>page:pause 1000 A</code> pause the video "1000" on channel "A" while just <code>page:pause 1000</code> would pause video "1000" on its designated channel.
<code>prepare(pageName, channelName)</code>	Prepares a named element on a named channel. If channel is not mentioned it prepares the element on its designated channel. If the element name is not mentioned the current element is prepared. Such as, <code>page:prepare 1000 A</code> prepare the page "1000" on channel "A" while just <code>page:prepare 1000</code> would prepare page "1000" on its designated channel.
<code>prepare_current_on_channel <channel-name></code>	Prepares the currently read element to a channel other than the specified channel for that element.
<code>print</code>	Print the current page. Use settings <code>printer/left_margin</code> , <code>printer/right_margin</code> , <code>printer/top_margin</code> , <code>printer/bottom_margin</code> to control the size, where the value is the percent of the page width/height.
<code>read(pageNr:integer)</code>	Reads the specified page number.
<code>read_parentpage</code>	Read the parent of a sub page.
<code>read_rootpage</code>	Read the root page in a page hierarchy.
<code>read_subpage(string string)</code>	Read a sub page of another page, for instance a scroll element. Argument format: "tabfield key/pagename", eg. "1/1003".
<code>read_template(string templateid)</code>	Reads a template if concepts and variants are enabled. If not, read a normal template.

Command Name and Arguments	Description
<code>read_template_with_context(string template, string contextPairs)</code>	Read a master template with context variables that can override the show properties. The context is given a whitespace-separated list of keys and values.
<code>readnext</code>	Read next element in active view.
<code>readprevious</code>	Read previous element in active view.
<code>redo</code>	Redo edit in current page history.
<code>refresh_linked_data(page name (optional))</code>	Refresh linked data if the page has any tabfields that are linked to data from an external source.
<code>refresh_linked_data_async(page name (optional))</code>	Start refreshing linked data if the page has any tabfields that are linked to data from an external source. The refresh can be cancelled with the 'page:cancel_refresh_linked_data' command.
<code>run_command_with_variables_on_channel(string command, string channel, restString variables)</code>	Run a command (for example a take) on a page with variables to a specific channel.
<code>save</code>	Saves current page.
<code>save_page_data(filename:string (optional), version:string (optional, defaults to 2.6))</code>	Save page data as XML in specified version (2.5 or 2.6).
<code>save_scene</code>	Save current page as a new Viz scene.
<code>save_thumbnail</code>	Save a thumbnail for the currently read Viz Trio page, based on the content of the renderer. The thumbnail is stored in the Media Sequencer and shown in the page list.
<code>saveas(integer)</code>	Goes in saveas mode or saves page to page given as argument. Triggers the save as action for a template or page, saving it with the page number as parameter. If no parameter is given, the callup code is set to the next available page number. However, the page isn't saved.

Command Name and Arguments	Description
set_effect(restString effect)	Sets the transition effect to be used between (scene-based) pages. Empty string specifies hard cuts (default).
set_effect_on_selected(string effect)	Sets the transition effect to be used between (scene-based) pages on the currently selected one. Empty string specifies hard cuts (default).
set_expert_mode_enabled(boolean enabled)	Enable or disable "expert mode" for the current page. This enables all tab field properties to be accessed instead of a filtered subset. Restarting Viz Trio resets the expert mode.
set_pagestatus(status: string)	Set the status of the selected pages. Valid arguments: FINISHED/UNFINISHED/NONE.
set_pagestatus_one(pagenr: integer, status: string)	Set the status of page. Valid status arguments: FINISHED/UNFINISHED/NONE.
set_pagetime(string pageid, string type, string value)	Sets the begin/end time for the page specified.
set_program_pagechannel(string pageid, restString channelname)	Sets the program channel for the page.
set_property(string name, restString value)	Set the value of a named tab field property.
setpagetemplate(string templateName)	Change the template of the current page.
store	Save the page with the name given in the read page edit field (the left one).
tab	Tab to the next tab field.
tab_backwards	Tab to the previous tab field.
tabto(tabfieldnr:integer)	Go to a specific tab-field number.
tabtotabfield(tabfieldname:string)	Go to a specific tab-field number.

Command Name and Arguments	Description
take (pageName, channelName)	Plays the named element on its designated channel. If channel is not mentioned it plays the element on its designated channel. If the page name is omitted then the current page is played. Such as, page:take 1000 A takes the page "1000" on channel "A" while just page:take 1000 takes the page "1000" on its designated channel.
take_exchange	Take current page and read the last page that was taken.
take_with_variables(restString variables)	Take a page with variables.
take_with_variables_on_channel	Take a page with variables to a specific channel.
takenext	Take current page and read next page in active view.
takeout(pageName, channelName)	Takes out the named element from the named channel. If channel is not mentioned it takes out the element from its designated channel. If the element name is not mentioned then the current element takes out from the designated channel. Such as, page:takeout 1000 A takes out the page "1000" on channel "A" while just page:take 1000 would take out page "1000" from its designated channel.
unarm(channelName:string)	Unarms the specified channel.
unarm_current	Unarms the currently read page on the channel it is armed.
unarm_all	Un-arms all armed pages.
undo	Undo edit in current page history.
update	Update the program renderer (do a take without animating it in).
update_preview	Update the external preview renderer with the contents of the local preview.

Playlist

Command Name and Arguments	Description
<code>add_page(string pagename, restString duration)</code>	Add a page to the end of the playlist. The duration argument is optional. If not given, the page duration is used.
<code>create_playlist(string name)</code>	Adds a new empty global playlist and returns the path to the new playlist. The name of the playlist is optional. If not specified, a dialog pops up to ask for it. The new playlist shows up in the "playlists" in the "Change Show" dialog.
<code>close (restString: id)</code>	Closes a specified playlist. The id parameter is either the path or the unique identifier of the playlist to close. If omitted the current playlist is closed.
<code>copy_to_show (restString start_num)</code>	Copy the elements in the playlist to the current show
<code>clear (restString id)</code>	Clears the specified playlist. The id can be the playlist path or its identifier. If omitted the current playlist gets cleared.
<code>direct_take_out_selected</code>	Take out the currently selected playlist element.
<code>direct_take_selected</code>	Take the selected playlist item onair without previewing it.
<code>get_concept</code>	Retrieves the concept of a playlist.
<code>get_current</code>	Retrieves the path of the playlist which is the current one. May return an empty string.
<code>get_playlists</code>	Retrieves a list of all open playlists in the current show as space separated list of playlist references which can be used to refer to playlists in other commands like <code>playlist:set_current</code> .
<code>go_to_next_story</code>	Select the next story (group) in the active playlist.
<code>go_to_previous_story</code>	Select the previous story (group) in the active playlist.
<code>goto_manual_graphics_mode(restString enable)</code>	Enable/disable manual playout of graphic elements. Use "true" or "" for manual and "false" for automatic mode.

Command Name and Arguments	Description
initialize_playlist	Initialize the playlist (load graphics) on the external output channels.
jump_to_ncs_cursor	Jump to the element that the NCS-cursor (newsroom control system) is pointing to.
pause_carousel	Pause the carousel.
read_next_item	Read the next playlist item.
read_prev_item	Read the previous playlist item.
read_selected	Read the currently selected playlist element.
reload_thumbnails (restString: template_names)	Reload the thumbnails for the given templates on all open playlists. Reloads all thumbnails if no templates are given.
run_selected_with_custom_action (string Action)	This macro command is generic and can append any Media Sequencer command as a parameter for playing out selected graphics and/or video clip elements in any playlist. Example: <pre>playlist:run_selected_with_custom_action "prepare" {{playlist:run_selected_with_custom_action "take"}}Note that if you have a version of the Media Sequencer that is newer than the version released with your version of Viz Trio, you can still use new commands supported by the Media Sequencer by using this macro command.</pre>
select_first	Select the first element in the playlist.
select_next	Select the next element in the active playlist.
select_next_by_description(restString phrase)	Selects the next element in the active playlist which has the given phrase in its description.
set_channel_on_selected (restString channelName)	Set the channel for all selected items in the current playlist
set_current (string: id)	Defines the specified playlist as the current one for operations like add_item. The id can be the playlist path or its identifier.

Command Name and Arguments	Description
<code>select_previous</code>	Select the previous element in the active playlist.
<code>set_attribute_on_selected(string attribute, restString value)</code>	Sets an attribute on all the selected items in a playlist.
<code>set_concept(restString value)</code>	Set the concept for a playlist.
<code>set_filter (restString: filterName)</code>	Set the filter of the currently active playlist. The filter must already exist. Calling with no filter name resets the filter of the playlist.
<code>set_looping(restString looping TRUE or FALSE)</code>	Set to true if the carousel shall be looping.
<code>start_carousel (playlist name or id (optional, defaults to active playlist))</code>	Run the elements in the playlist as a carousel.
<code>stop_carousel (playlist name or id (optional, defaults to active playlist))</code>	Stop a playlist running as a carousel.

Proxy

Command Name and Arguments	Description
<code>enable_viz_proxy(boolean enable)</code>	Enables/disables Viz Trio acting as a Viz Proxy on the port configured.
<code>set_proxy_port(int port)</code>	Set the port that Viz Trio should listen on.
<code>set_viz_host(string host)</code>	Set the host where the proxy can locate Viz.
<code>set_viz_port(int port)</code>	Set the port the proxy should use to access Viz.

Rundown

Command Name and Arguments	Description
<code>go_to_next_story</code>	Sets focus on the next story/group in the active rundown.
<code>go_to_previous_story</code>	Sets focus on the previous story/group in the active rundown.

Command Name and Arguments	Description
reload_template_images(restString templates)	Reloads the thumbnails for all listed scene paths. If the scene paths are not given all thumbnails are reloaded.

Script

Command Name and Arguments	Description
eval(restString scriptExpression)	Evaluate the VBScript expression given as a parameter, and return the value converted to a string.
execute_script	Tries to run the OnUserClick event on the current template script. Doesn't report any errors e.g. if the event is not handled in the template script.
get_showscript_name	Gets the name of the current show script. Specify full name, does not add .vbs to the script name
import_script(string scriptName, restString scriptFilename)	Imports the script into the current show.
run_macro_script(restString scriptCode)	Run the VBScript script code given as a parameter. The snippet can contain multiple lines of code. The macro returns an empty string or an error message.
run_script(string scriptFunction, restString argumentList)	Run the specified script function. The argument list is a list of strings separated by commas or white space. Arguments that can contain commas or white space must be enclosed by double quotes.
run_showscript(string scriptFunction, restString argumentList)	Run the script associated with the current show. The argument list is a list of strings separated by commas or white space. Arguments that can contain commas or white space must be enclosed by double quotes.

Scroll2

Command Name and Arguments	Description
add_easepoint(string pagename)	Add an easepoint to the current scroll.
clear_elements	Clear scroll elements (remove all).

Command Name and Arguments	Description
<code>copy_element(int sourceIndex, int targetIndex)</code>	Add a copy of scroll element at sourceIndex to targetIndex.
<code>create_scroll_template(string sceneName, string templateDescription, float width, float height, boolean showBounds)</code>	Create a new scroll template with the given parameters.
<code>edit_easepoint(string subpage, string easepoint)</code>	Edit an easepoint in the current scroll.
<code>insert_objects(int index, restString objects)</code>	Insert SCENE or GEOM objects from viz into the current scroll.
<code>insert_pages(int index, restString pages)</code>	Insert pages from current show into current scroll at index.
<code>insert_range(int index, string range)</code>	Insert an inclusive numerical range of pages (e.g 1000-2000) from the current show, or a single page.
<code>move_element(int sourceIndex, int targetIndex)</code>	Move scroll element from sourceIndex to targetIndex.
<code>next_easepoint</code>	Go to next ease point editor.
<code>next_element_data</code>	Go to next element data (extra spacing) editor.
<code>preview_continue</code>	Continue the current scroll in preview.
<code>preview_start</code>	Start the current scroll in preview.
<code>preview_stop</code>	Stop the current scroll in preview.
<code>previous_easepoint</code>	Go to previous ease point editor.
<code>previous_element_data</code>	Go to previous element data (extra spacing) editor.
<code>program_continue</code>	Continue the current scroll on the program channel (must be taken on-air first).
<code>program_start</code>	Start the current scroll on the program channel (must be taken on-air first).

Command Name and Arguments	Description
program_stop	Stop the current scroll on the program channel (must be taken on-air first).
reload_all_page_data	Reload all scroll page data by copying from original pages.
reload_page_data(int index)	Reload scroll page data at index by copying from original page.
reload_template_graphics	Reload all scroll elements (reload geometry objects created during import of templates).
reload_templates	Reload all scroll elements by merging with the current show templates.
remove_easepoint(string subpage, string easepoint)	Remove a named easepoint in the current scroll.
remove_element(int index)	Remove scroll element at index.
rename_easepoint(string subpage, string oldName, string newName)	Rename an easepoint in the current scroll.
set_focus(int index)	Focus on scroll element.

Settings

Command Name and Arguments	Description
apply_gateway_settings	Applies all MOS gateway settings which have been set up previously.
get_setting(string path)	Gets a Viz setting.
get_global_setting (string: path)	Get a global setting.
get_vcp_database_connection_string	Get VCP database connection string
get_vcp_database_schema	Get VCP database schema

Command Name and Arguments	Description
export_to_file (boolean: include_settings, boolean: include_profiles, restString filename)	Export settings to a file. If filename is empty, a file dialog is shown.
enable_gateway (boolean: flag)	Enables or disables the MOS gateway according to the given flag. After running this command settings:apply_gateway_settings have to be run in order to put the setting operative.
import_from_file (restString: filename)	Export settings to a file using old format for import on Viz Trio 2.6 and earlier. If filename is empty, a file dialog is shown.
import_from_file (restString filename)	Import settings from a file. If filename is empty, a file dialog is shown.
start_gateway	Starts the MOS gateway handler of the Viz Media Sequencer Engine.
restart_gateway	Restarts the MOS gateway handler of the Viz Media Sequencer Engine.
set_gateway_host (string hostname [<port_nummer>])	Changes the MOS gateway host name and port. <host-name> must be a valid host name or IP address where the gateway is running. <port-number> must be a valid port number. It is optional and defaults to 10640. After running this command settings:apply_gateway_settings have to be run in order to put the setting operative.
set_gateway_ncs_id (string: id)	Changes the gateway NCS ID. After running this command settings:apply_gateway_settings have to be run in order to put the setting operative.
set_gateway_mos_id (string: id)	Changes the gateway MOS ID. After running this command settings:apply_gateway_settings have to be run in order to put the setting operative.
set_gateway_messages_per_second (int: seconds)	Changes the number of messages per second to process by the MOS gateway. A zero value disables this setting. After running this command settings:apply_gateway_settings have to be run in order to put the setting operative.

Command Name and Arguments	Description
<code>set_vcp_database (string: connection_string, string: Schema)</code>	Set VCP database connection setting
<code>set_global_setting</code>	Sets a global Viz setting (e.g. <code>settings:set_global_setting encoding ISO-8859-1</code>).
<code>set_setting(string path, restString value)</code>	Set a show setting.
<code>set_send_item_stat (boolean: flag)</code>	Enables or disables the item stat sending of the MOS gateway according to the given flag. After running this command <code>settings:apply_gateway_settings</code> have to be run in order to put the setting operative.
<code>stop_gateway</code>	Stops the MOS gateway handler of the Viz Media Sequencer Engine.

Show

Command Name and Arguments	Description
<code>append_pages_from_xml(boolean showDialogs, string filename)</code>	Loads pages from XML and replaces any pages that are overwritten.
<code>copy_pages (restString: pageList)</code>	Copies the given page(s) to the clipboard
<code>copy_pages_to_number(int destination, restString CommaSeperatedList)</code>	Copy the pages to a destination number.
<code>copy_pages_with_offset(int offset, int CommaSeperatedList)</code>	Copy the pages to a destination number.
<code>create_combo_template (string: name, string: pagelist)</code>	Create a combo master template.
<code>create_group(string groupname)</code>	Create a page group in the current show, before the focused node.
<code>create_playlist(restString name)</code>	Adds a new playlist to the show and returns the path to the new playlist. If name is omitted, a dialog pops up asking for a name.

Command Name and Arguments	Description
<code>create_show(restString showPath)</code>	Create a new show. The show path may specify a show within a show folder.
<code>cut_pages (page list)</code>	Cuts the given page(s) to the clipboard.
<code>delete_all_pages</code>	Deletes all the pages in the page view.
<code>delete_templates(restString templateList)</code>	Delete the given templates from the current show. Does nothing if the show does not have concepts and variants enabled.
<code>enable_context</code>	Enable concept/variant for the current show.
<code>export_setting(string name, restString value)</code>	Configure an export setting, use this before <code>export_show</code> .
<code>export_show(filename:string)</code>	Export a show (Viz Trio archive).
<code>get_associated_files</code>	Get the list of associated files for the show
<code>get_background_scenes</code>	Get a space-separated list of background scenes in the show.
<code>get_concept</code>	Get the show-concept.
<code>get_context_variable(string variable)</code>	Get the show-context-variable.
<code>get_custom_colors</code>	Get the custom colors for the current show.
<code>get_default_viz_path</code>	Get the default viz scene path for the current show.
<code>get_group_pages</code>	Returns a list of pages in the specified group.
<code>get_groups</code>	Returns a list of groups in the current show.
<code>get_library_path</code>	Get the default viz scene path for the current show.

Command Name and Arguments	Description
<code>get_page_xml(restString page_path)</code>	Get the xml for the specified page. The path is given as showpath/<page-id> (e.g. <code>get_page_xml myshow/1000</code>). For "untitled show" use empty showpath (e.g. <code>get_page_xml /1002</code>). Note that show names are case sensitive.
<code>get_pages</code>	Returns a list of pages in the current show.
<code>get_preview_background_image</code>	Get the Preview background image for the current show.
<code>get_scenes_for_pages(restString pagelist)</code>	Get the list of scenes for the given pages.
<code>get_setting(string path)</code>	Get a show setting.
<code>get_settings</code>	Get all settings.
<code>get_show_description</code>	Get the description of the current show.
<code>get_show_name</code>	Get the name of the current show.
<code>get_templates</code>	Returns a list of templates in the current show.
<code>get_variable(string propname)</code>	Get the value of a show variable.
<code>get_variant</code>	Get the show-variant.
<code>goto_viz_dir(path:string)</code>	Change to a show with the same name as a viz directory, creating it if necessary.
<code>import(scenes:string)</code>	Shows the Import Scenes pane or imports specified scene(s).
<code>import_and_initialize(scenes:string)</code>	Import scenes to templates, then reload on external renderers.
<code>import_no_gui(restString sceneNames)</code>	Imports the scenes given as parameter without showing/closing the import GUI.

Command Name and Arguments	Description
<code>import_pages_from_xml(string filename, int offset, boolean interactive, boolean pageNames)</code>	Import the named pages from an XML file. An offset different from zero causes pages with numerical names to be offset by this number, unless they are templates. Pages are overwritten automatically if interactive is false.
<code>import_recursively(string path)</code>	Import all the scenes recursively from this folder.
<code>import_setting(string name, restString value)</code>	Configure an import setting, use this before <code>import_show</code> .
<code>import_show(filename:string)</code>	Import a show (Viz Trio archive).
<code>load_armed</code>	Load all armed elements on their channels.
<code>load_locally</code>	Loads all scenes referenced by pages in the current show on the local viz preview.
<code>load_show</code>	Load show (scenes, textures, fonts) on external output channels. Replaces the old initialize command.
<code>load_show_on_renderers(string ProgramChannelName, restString PreviewChannelName)</code>	Load current show on the specified channels. First argument is treated as a program channel, second as preview.
<code>move_pages_to_group(string group, restString pages)</code>	Move pages into the specified group. If the group name is empty, move to top level.
<code>move_pages_to_number(int destination, restString CommaSeperatedList)</code>	Move the pages to destination number
<code>move_pages_with_offset(int destination, int CommaSeperatedList)</code>	Move the pages with an offset.
<code>paste_pages</code>	Pastes the given page(s) from the clipboard to the current show.
<code>page_exists (restString page)</code>	Checks if a page with the specified name exists in the current show. Returns "true" or "false".
<code>reimport_templates (string: templateList)</code>	Reimport the given templates

Command Name and Arguments	Description
reload_current_show	Reloads the current show
reload_on_renderers (string: template_l	Reload (re-initialize) graphics for the specified master templates or scene info on program and preview channels.
load_on_renderers(restString pages)	Load (initialize) graphics for the specified pages on program and preview channels.
rename_page(string old_name, restString new_name)	Rename a page name in the opened show
save_pages(string FileName, restString CommaSeperatedList)	Saves the pages given as parameter to XML.
save_pages_to_xml(string filename)	Saves pages to XML.
save_selected_pages(restString fileName)	Saves the selected pages to XML.
set_associated_files(restString filelist)	Set the associated files for the show. ("file1" "file2" "file3").
set_concept(restString concept)	Set the show-concept to a new value.
set_concept_for_all (restString concept)	Set the given concept to the show and to all playlists, or no concept if the parameter is omitted.
set_context_variable(string variable, restString value)	Set the show-variant to a new value.
set_custom_colors(restString colors)	Set the custom colors for the current show.
set_default_scene_effect (restString: effect)	Sets the default transition effect to be used between (scene-based) pages. Empty string specifies hard cuts (default).
set_default_video_effect (restString: effect)	Sets the default transition effect to be used between videos and stillstore images. Empty string specifies hard cuts (default).
set_default_viz_path(path:string)	Set the default viz scene path for the current show.

Command Name and Arguments	Description
set_dir(restString path)	Changes the show directory to the path specified.
set_external_cursor(restString pagename)	Set the external cursor to page with page name or to current page if page name is empty.
set_filter	Set the filter of the page list or the currently active page view. The filter must already exist. Calling with no filter name resets the filter of the page view.
set_library_path(restString path)	Set the design-library path.
set_preview_background_image(path:string)	Set the preview background image for the current show.
set_setting(string path, restString value)	Set a show setting.
set_show_description(description:string)	Set the description of the current show.
set_show_name(name:string)	Set the name of the current show.
set_variable(string propname, restString value)	Set the value of a show variable. The value is stored on the Media Sequencer.
set_variant(restString variant)	Set the show-variant to a new value.
show_export_dialog_for_path(restString showpath)	Exports the show.

Socket

Command Name and Arguments	Description
connect_socket	Explicitly connect (open) the socket, if not already connected.
disconnect_socket	Disconnect (close) the socket, if connected.
send_socket_data(restString text)	Send UTF-8 encoded text over the socket. If it is a server socket, send to all clients.

Command Name and Arguments	Description
set_com_baud(int comBaudRate)	Set baud rate to use for the serial port.
set_com_number(int comPortNumber)	Set COM port number to use for the serial port.
set_socket_autoconnect(boolean autoconnect)	If auto-connect is true, a client socket is connected the first time data is sent, or a server socket is opened automatically.
set_socket_data_separator(string separator)	Set the separator for the OnSocketDataReceived callback.
set_socket_encoding(string encoding)	Set text encoding for socket to either UTF-8 or ISO-8859-1.
set_socket_host(string hostname)	Set the socket hostname. Does nothing if the socket type is server.
set_socket_port(int port)	Set the socket port number.
set_socket_type(string type)	Set the socket type to client or server.
socket_is_connected	Returns true if the socket is connected, false otherwise.

Tabfield

Command Name and Arguments	Description
active(active:string)	Toggle active flag of current tab field or set its value (0 or 1).
browse_file	Change to the browse file editor of the current tabfield or set the full path to a file.
clock	Change to clock editor of current tab field or set its value (string).
downcase_tabfield	Convert current tab field to lower case, if it contains text.
edit_property(string propertyName)	Changes editor to the specified property i.e 1.image.

Command Name and Arguments	Description
geom(vizPath:string)	Change to geometry editor of current tab field or set its value (GEOM*path).
get_custom_property(string tabfieldName)	Gets the custom property of the tab field. "" gets it for the current tab field.
get_tabfield_property(tabfieldnr:integer, property:string)	Gets a property value for a given tab field.
image(vizPath:string)	Change to image editor of current tab field or set its value (IMAGE*path or filename).
image_position	Change to image position editor of current tab field or set its value (two floats).
image_scaling	Change to image scaling editor of current tab field or set its value (two floats).
import_value_from_file(restString filename)	Load a tab field value from a text file (UTF-8 encoded).
kerning(value:float)	Change to kerning editor of current tab field or set its value (float).
material(vizPath:string)	Change to material editor of current tab field or set its value (MATERIAL*path).
next_property	Select the next property of the current tab field, or the first if the last one is currently selected.
position(value:float)	Change to position editor of current tab field or set its value (floats).
previous_property	Select the previous property of the current tab field, or the last if the first one is currently selected.
rotation(value:float)	Change to rotation editor of current tab field or set its value (floats).
scaling(value:float)	Change to scaling editor of current tab field or set its value (float).
set_custom_property(string tabfieldName, string value)	Sets the custom property of the tab field. "" as tab-field name uses the current tab field.

Command Name and Arguments	Description
set_tabfield_property(prop:string, value:string)	Set a property value.
set_value(value:string)	Sets the value of the current tab-field property.
text(value:string)	Change to text editor of current tab field or set its value (unicode string).
toggle_current_custom_property(string value1, string value2)	Toggle the custom property of the current tab field between the two given values.
upcase_tabfield	Convert current tab field to upper case, if it contains text.
value	Change to default editor.

Table

Command Name and Arguments	Description
clear_sorting(string table key)	Clear the sorted state of a table property.
delete_rows(string table key, string row index list)	Delete a number of rows from a table property.
get_row_num (string: table_key)	Get the number of rows of a table property.
insert_rows_with_xml (string: table_key, int: row_index, restString: XML_data_to_insert)	Insert several rows with their data into a table property.
get_cell_value(string table key, string column key, int row num)	Get a cell value of a table property.
list_columns (string: table_key)	Returns the column IDs of a table property as space separated list.
insert_rows(string table key, int row index, int number of rows to insert)	Insert a number of rows into a table property.
move_current_row_down	Move the selected row in the current table editor one step downwards.

Command Name and Arguments	Description
<code>move_current_row_up</code>	Move the selected row in the current table editor one step upwards.
<code>move_row(string table key, int row index to move, int target row index)</code>	Move a row downwards behind the target row or upwards before the target row.
<code>set_cell_value(string table key, string column key, int row num, restString new value)</code>	Set a cell value of a table property.
<code>sort_column(<table key> <column key> [ascending or descending] (optional, default: ascending))</code>	Sort a table ascending or descending after a column.

Text

Command Name and Arguments	Description
<code>incr_alpha(string key, string selectionStart, string selectionEnd, float alpha)</code>	Increment or decrement the transparency of selected text.
<code>incr_kerning(string key, string selectionStart, string selectionEnd, float kerning)</code>	Increment or decrement the kerning of selected text.
<code>incr_position(string key, string selectionStart, string selectionEnd, float posX, float posY, float posZ)</code>	Increment or decrement the position of selected text.
<code>incr_rotation(string key, string selectionStart, string selectionEnd, float rotX, float rotY, float rotZ)</code>	Increment or decrement the rotation of selected text.
<code>incr_scale(string key, string selectionStart, string selectionEnd, float scaleX, float scaleY, float scaleZ)</code>	Increment or decrement the scaling of selected text.
<code>revert(string key)</code>	Revert current text to saved value.
<code>set_alpha_mode</code>	Set text manipulation mode to Alpha.
<code>set_color(string key, string selectionStart, string selectionEnd, string color)</code>	Change the color of selected text.
<code>set_color_mode</code>	Set text manipulation mode to Color.

Command Name and Arguments	Description
set_justification(string key, string justification)	Set the text justification (LEFT, CENTER, RIGHT).
set_kerning_mode	Set text manipulation mode to Kerning.
set_position_mode	Set text manipulation mode to Kerning.
set_rotate_xy_mode	Set text manipulation mode to Rotate x and y.
set_rotate_z_mode	Set text manipulation mode to Rotate z.
set_scale_mode	Set text manipulation mode to Scale.
show_replace_list	Shows the pre-configured character palette for the character to the left of the cursor.

Viz Trio

Command Name and Arguments	Description
activate_current_playlist	Activate current playlist
assign_script(string page, boolean fileScript, restString scriptunit)	Assign or change the script for the template of the specified page. Note: This affects all pages using this template. Example: trio:assign_script 1001 True page1001
cleanup	Performs a clean-up of the current show with all playlists on external renderers by clearing Viz layers and freeing memory.
check_channels_status	Check connection status of program and preview channels in profile. Both for Viz Engine and video channels. Returns "true" if all are good, "false" otherwise.

Command Name and Arguments	Description
<code>cleanup_all_channels</code>	Cleans up all channels in the current profile by clearing Viz layers and freeing memory. Does not clear the channel for transition logic pages.
<code>cleanup_channel(restString channelname)</code>	Cleans up the specified channel by clearing Viz layers and freeing memory. Note: The <code>trio:clear_channel</code> command duplicated the <code>trio:cleanup_channel</code> command and is therefore as of version 2.10 deprecated.
<code>cleanup_renderers</code>	Cleans up the renderers of the current playlist (frees graphics and clears output) on the utilized external output channels.
<code>clear_preview</code>	Cleans up Viz preview channel by clearing Viz layers and freeing memory.
<code>clear_program</code>	Cleans up Viz program channel by clearing Viz layers and freeing memory.
<code>continue_last_taken_page</code>	Continue the page that was last taken.
<code>create_snapshot_page</code>	Takes a snapshot and creates a page with that snapshot.
<code>deactivate_current_playlist</code>	Deactivate current playlist
<code>exchange</code>	Exchange the content of the program and external preview.
<code>exit</code>	Close Viz Trio.
<code>freeze_program_clock(string string)</code>	Freeze a clock on the program machine.
<code>get_global_variable(string propname)</code>	Get the value of a global variable.
<code>get_next_savenumber</code>	Get the number that is proposed on the next save as or with a <code>create_snapshot_page</code> .

Command Name and Arguments	Description
help	Opens Viz Trio user guide. The help file is also opened by clicking the Help button in the user interface (next to Config), or by pressing the F1 key unless it has been assigned to another action (e.g. macro). See also General settings.
initialize	Initializes the current show (both page list and all playlists) on external renderers.
initialize_show_or_playlist	Initialize current show or playlist on external renderers.
invoke_with_current_page(string commandName)	Invoke the given command with the name of the current page as a parameter (useful for macro shortcuts).
invoke_with_current_scene(string commandName)	Invoke the given command with the name of the current scene as a parameter (useful for macro shortcuts).
offair	Set client in off-air mode.
onair	Set client in on-air mode.
open_printsettings_dialog(restString pages)	Open the printer-settings dialog.
print_and_save_snapshots(restString pages)	Print the pages. Use settings printer/left_margin, printer/right_margin, printer/top_margin, printer/bottom_margin to control the size, where the value is the percent of the page width/height.
print_snapshots(restString pages)	Print the pages. Use settings printer/left_margin, printer/right_margin, printer/top_margin, printer/bottom_margin to control the size, where the value is the percent of the page width/height. With no argument, the content of the preview is printed.
process_ii_string(restString value)	Sends a string to the intelligent interface handler on the Media Sequencer and lets it process it.
rename_show_folder (string: old_name, restString: new_name)	Rename a folder.

Command Name and Arguments	Description
rename_show (string: old_name, string: new_name)	Rename a show.
restart_local_viz	Restarts the local Viz preview channel.
restart_local_viz_program	Restarts the local Viz program channel. Only available when local program channel has been started.
save_snapshots(restString pages)	Print the pages. Use settings printer/left_margin, printer/right_margin, printer/top_margin, printer/bottom_margin to control the size, where the value is the percent of the page width/height.
send_local_vizcmd(command:string)	Send a command or a list of commands separated by newlines to the local viz preview. Returns a list of return values from viz. Newlines within a single command must be escaped (\n).
send_preview_vizcmd(command:string)	Send a Viz command to the preview renderer.
send_preview_vizcmd_noerror(restString command)	Send a Viz command to the preview renderer if possible, otherwise do nothing (no error message).
send_vizcmd(command:string)	Send a Viz command to the program renderer.
send_vizcmd_to_channel(string channelName, restString command)	Send a Viz command to a specified channel.
set_global_variable(string propname, restString value)	Set the value of a global property. The value is stored on the Media Sequencer.
set_mcu_folder(string string)	Sets the folder of the MCU/AVS plug-in.
set_vizconnection_timeout(int timeout)	Set the Viz connection timeout.
set_next_savenumber (int: name)	Set the number that is proposed on the next save as or with a create_snapshot_page
show_trio_settings_dialog	Show dialog containing a tree view of Viz Trio settings.
show_settings_dialog (restString: category)	Show dialog containing a tree view of Viz Trio settings

Command Name and Arguments	Description
sleep(int millisec)	Sleeps a page a certain amount of milliseconds. For example: <code>TrioCmd("page:take 1000")</code> <code>TrioCmd("trio:sleep 10000")</code> <code>TrioCmd("page:take 1001")</code> <code>TrioCmd("trio:sleep 10000")</code> <code>TrioCmd("page:takeout")</code>
swap_channels	Swap the program and preview channels.
synchronize_clocks	Synchronize the clocks to the clocks on the program renderer.
takeout_last_taken_page	Continue the page that was last taken.
unfreeze_program_clock(string string)	Freeze a clock on the program machine.
wait(int millisec)	Pauses execution flow for a specified time in milliseconds without blocking other commands, unlike “sleep” which blocks all command execution during the waiting period (see trio:wait).

Util

Command Name and Arguments	Description
get_node_text(string string)	Gets the node text of a Media Sequencer node.
list_archive_files(filename:string)	Returns a list of files in a Viz Trio archive (also works for other tar files).
list_archive_pages(filename:string)	Returns a list of pages in a Viz Trio archive (creates a list of all_pages.xml within the Viz Trio archive).
list_archive_parameters(restString filename)	Returns a list of parameters of a Viz Trio archive.
list_xml_pages(restString filename)	Returns a list of pages in an XML file.
run_mse_entry(string path)	Runs the node at the specified path. No context is given.

Command Name and Arguments	Description
set_node_text(string path, restString value)	Sets the node text of a Media Sequencer node.

Viz

Command Name and Arguments	Description
cleanup_locally	Cleanup (unload) all viz resources locally. If a scene is active in the local renderer, it is not cleaned up.
continue	Continue the animation in viz preview (if stopped).
import_image_from_file(string filename, restString pool folder)	Import the given image file into the given viz image pool folder.
play	Start (play) the animation in viz preview.
load_data_locally (boolean: paths)	Load viz resources locally. The space-separated list of paths must include the pool prefix and full path, e.g. SCENE*folder/name
reload_data_locally(boolean paths)	Reload viz resources locally. The space-separated list of paths must include the pool prefix and full path, e.g. SCENE*folder/name.
set_bounding_box(boolean visible)	Toggle bounding box visibility.
set_key_alpha	Viz Artist setting.
set_key_preview	Viz Artist setting.
set_rgb	Viz Artist setting.
set_safearea(boolean visible)	Toggle safe area visibility.
set_stage_pos(restString pos)	Set the current stage position.
set_titlearea(boolean visible)	Toggle title area visibility.
stop	Stop the animation in viz preview.

Command Name and Arguments	Description
set_format(string format)	Switch preview to the given adaptive scene design format.
set_next_format	Switches preview to the next available adaptive scene format.
set_previous_format	Switches preview to the previous available adaptive scene format.

vtwtemplate

Command Name and Arguments	Description
get_template	Gets the VTW template for the current show.
import_template(string ScriptName, restString FileName)	Imports a template from file to the current show. Overwrites any template with the same name.
reimport_active_template	Reimports the active VTW template for the current show.
run_vtw_function(string functionName, restString argumentList)	Deprecated, use <i>vtwtemplate: invoke_function</i> instead. Runs a script function in the active VTW template. The argument list is a list of strings separated by commas or white space. Arguments that can contain commas or white space must be enclosed by double quotes.
invoke_function(string functionName, restString argumentList)	Execute a script function in the active VTW template. The argument list is a list of strings separated by commas or white space. Arguments that can contain commas or white space must be enclosed by double quotes.
run_vtw_script(string FunctionName, restString Data)	Runs a script function in the active VTW template. Only a single parameter is supported by the Data parameter, so use <i>vtwtemplate:invoke_function</i> instead if possible.
set_template(restString templateName)	Sets the VTW template for the current show.

Shortcut Keys

With Viz Trio you can assign macro commands and scripts to shortcut keys on either a show-by-show or global basis.

This section covers the following topics:

- [Working with Shortcut Keys](#)
- [Adding Functions](#)

Working with Shortcut Keys

- For shortcut keys assigned on a *global* level, use the [Configuration Window](#).
- Show-specific shortcut keys are assigned using [Show Properties](#).

Managing Shortcut Keys

To reassign a shortcut key

- Double-click the command in the list to perform changes.

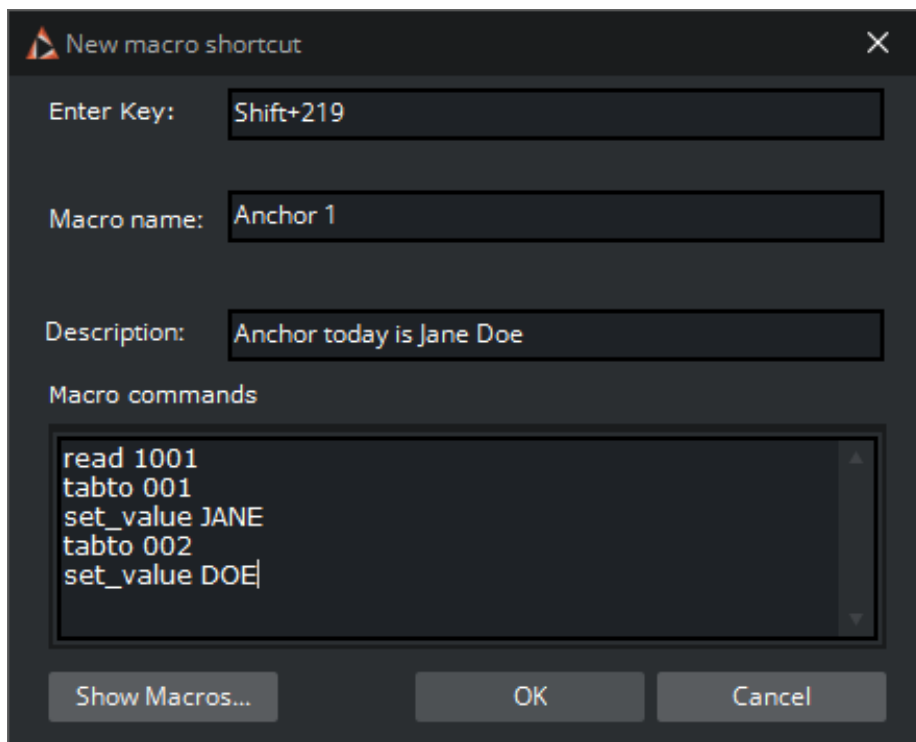
To remove a shortcut key

- Select a macro or script that has been manually added and then either
 - a. Right-click and select **Remove** from the context menu
 - b. On the keyboard press **Delete**.

Macros and Scripts

To assign a macro or script to a shortcut key

1. Click the **Add Macro** or **Add Script** button. The Macro Commands editor opens:

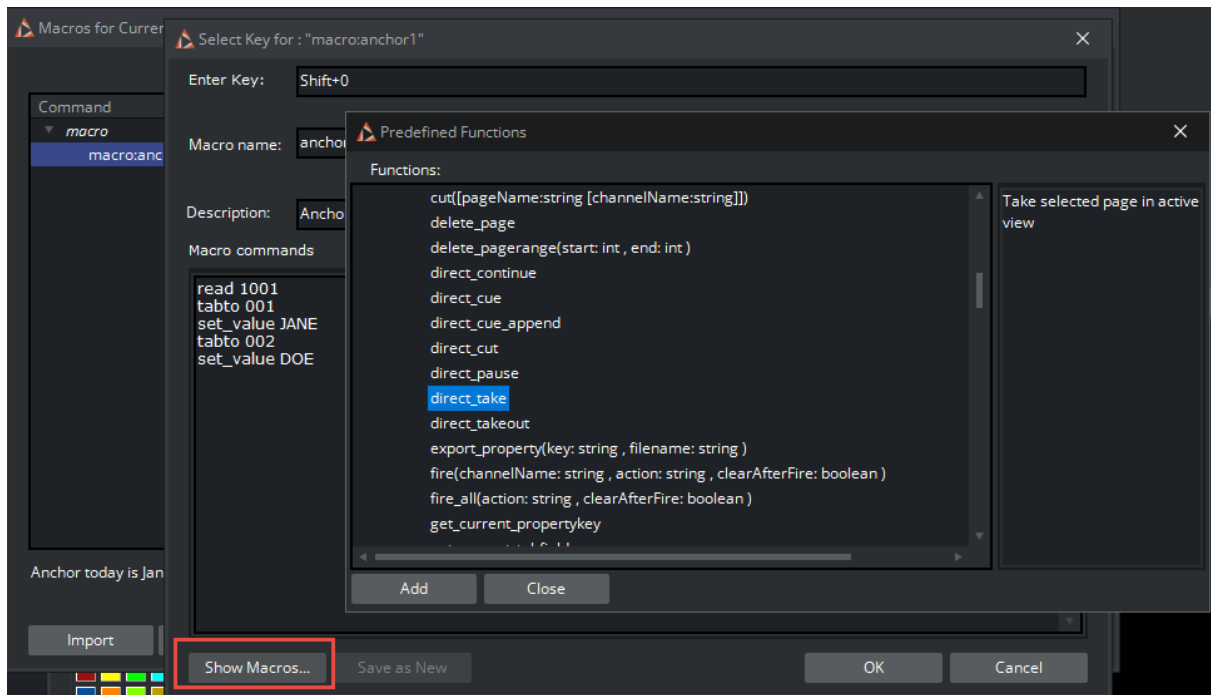


2. Select the keyboard keys for the shortcut. This includes combinations of **CTRL**, **SHIFT**, **ALT**, **ALT GR** with other characters.
 - If the selected shortcut key is already in use you may override it, unassigning it from the previous macro or script.
3. Enter a **Macro** or **Script Name** and Description of the macro or script.
 - If the macro name is already assigned, use **Save as New**
4. Add macro or script commands in the **Macro commands** (or Script) input area.
5. To confirm, click **OK**.

Adding Functions

To add a predefined function to a macro or script shortcut

1. Open the macro/script for editing.
2. Under **Macro commands**, click **Show Macros** to open a list of predefined functions.



3. Select required function and click **Add**.
4. Save changes.

Info: Predefined macro commands can also be applied to *scripts*.

7 Scripting

Scripts in Viz Trio can be stored on Media Sequencer, on a per show basis, and as files on a drive (preferably shared).

A show script is only accessible to machines connected to the same Media Sequencer using the same show. Scripts can be assigned to templates and shows.

 **Note:** It is only possible to assign one script per show or template. However, you can include other scripts as part of the main script in order to extend its functionality.

All edits to a script are carried out using the Viz Trio [Script Editor](#). Changes made to a show script only affect the selected show and those clients that control the same show. Changes made to a script file on a shared script repository affect all shows that use the same script.

 **Note:** File scripts are read into the show each time a show is opened.

Only templates can have scripts assigned. All instances of a template inherit the template script.

This section contains reference information on the following topics:

- [Viz Trio Scripting](#)
- [Viz Template Wizard Scripting](#)

 **Tip:** Always remember to escape backslashes correctly.

7.1 Viz Trio Scripting

A VBScript can be attached to any show or template, enabling custom functionality for a show or template. Typical applications for scripting include importing data from database sources and giving users guidance. All template instances with a script inherit the script.

Viz Trio supports a set of [Macro Commands and Events](#) that may be used as part of the script.

This section covers the following topics:

- [Script Directory](#)
- [Script Editor](#)
- [Script Backup](#)
- [Script Errors](#)

7.1.1 Script Directory

All show scripts are stored on Media Sequencer. However, a local directory is still needed for script that are stored as files:

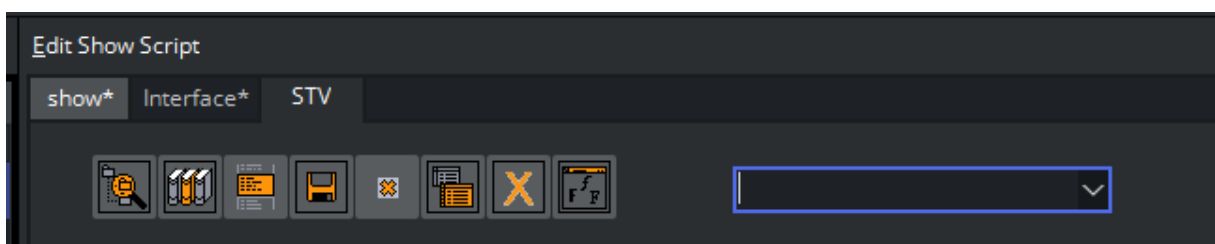
- Open the Configuration window, and set the *Script Path* under the [User Interface](#) section.

The default script directory is `%ProgramData%\Vizrt\Trio\scripts` normally `C:\ProgramData\Vizrt\Trio\scripts`.

If a show is imported without its underlying scripted templates in the script folder, Viz Trio creates an empty script file. Since the script files are empty, you must reassign the scripts and restart Viz Trio to load the updated script memory.

Note: Versions of Viz Trio prior to 3.0 store scripts in `C:\Program Files (x86)\Vizrt\Viz Trio\scripts`. Script files for a scripted show or template are copied to the Viz Trio script folder.

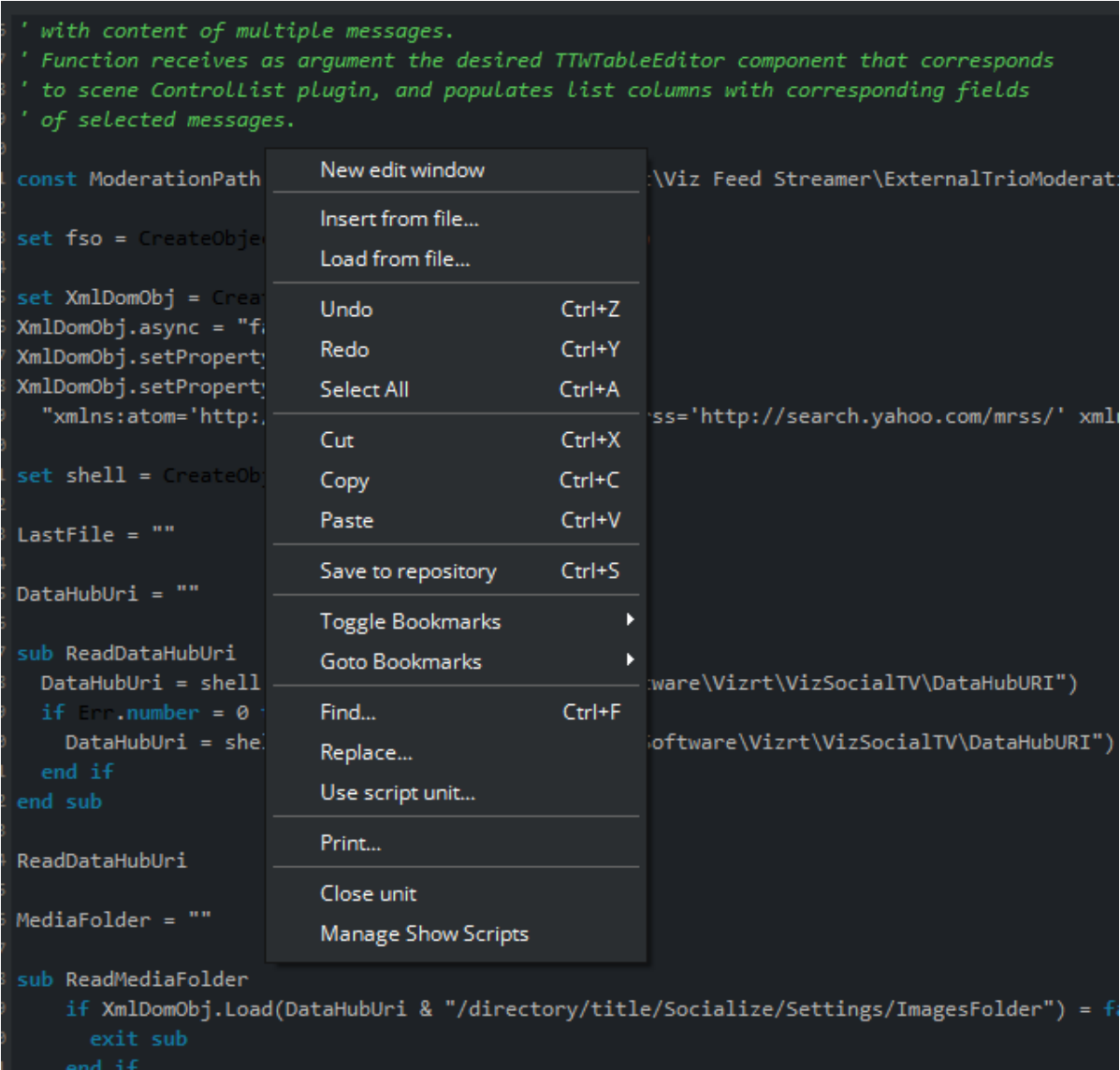
7.1.2 Script Editor



- **Predefined Functions:** Opens the [Predefined Functions](#) window containing [Overview of Commands](#) and [Events](#) that can be used in scripts.
- **Show Library Scripts:** Opens the [Library Scripts](#) window displaying scripts which can be attached to a template as a 'USEUNIT reference.
- **Syntax check:** Checks syntax correctness.
- **Save script:** Saves a script to the server. You can also save the script by right-clicking inside the script window and selecting **Save to repository** from the [Context Menu](#), where you can also save units to file.
- **Manage show scripts:** Opens the [Script Manager](#) for the show that's open.
- **Close all units:** Closes all script units.

- **Change font:** Changes font settings for the script editor.
- **Combo box:** Jumps between script functions and procedures within the same script.

Context Menu



- **New edit window:** Opens a new edit window.
- **Insert from file:** Inserts script from file.
- **Load from file:** Loads script from file in a new tab window.
- **Undo:** Undoes latest change.
- **Redo:** Redoes latest undo.
- **Select All:** Selects the whole script.
- **Cut:** Cuts selected text.
- **Copy:** Copies selected text.
- **Paste:** Pastes clipboard content.

- **Save to repository:** Saves the script.
- **Toggle Bookmarks:** Toggles between inserting and removing a bookmark at the selected row.
- **Go to Bookmarks:** Moves the cursor to the selected bookmark.
- **Find:** Opens a search window. Search for text strings and expressions.
- **Replace:** Opens a search and replace window. Search for a string to be replaced by another.
- **Use script unit:** Opens a window with a list of accessible units. Select one to use with another script or script unit.
- **Print:** Opens a print window.
- **Close unit:** Closes the unit.
- **Manage show scripts:** Opens a dialog where you can import, export, and delete a script from a template.

Shortcut Commands

- **CTRL + TAB:** Switches between the open scripting tabs.
- **CTRL + SPACE:** Displays the Code Insight Window containing a list of parameters, variables, functions, etc. available to the currently open script or in attached sub scripts.
- **CTRL + SHIFT + SPACE:** Displays code completion hints.
- **CTRL + Left mouse button:** Opens the highlighted function; works across scripts and scripts units, but not within the same script. For this to work, the '*USEUNIT <scriptname>*' notation must be added to the script in order to link the scripts together.

Predefined Functions



Almost all script functions in Viz Trio are macro [Overview of Commands](#) that can be wrapped in the `TrioCmd("macro_command")` format. The predefined functions window also contains a set of [Events](#) that can be added to scripts.



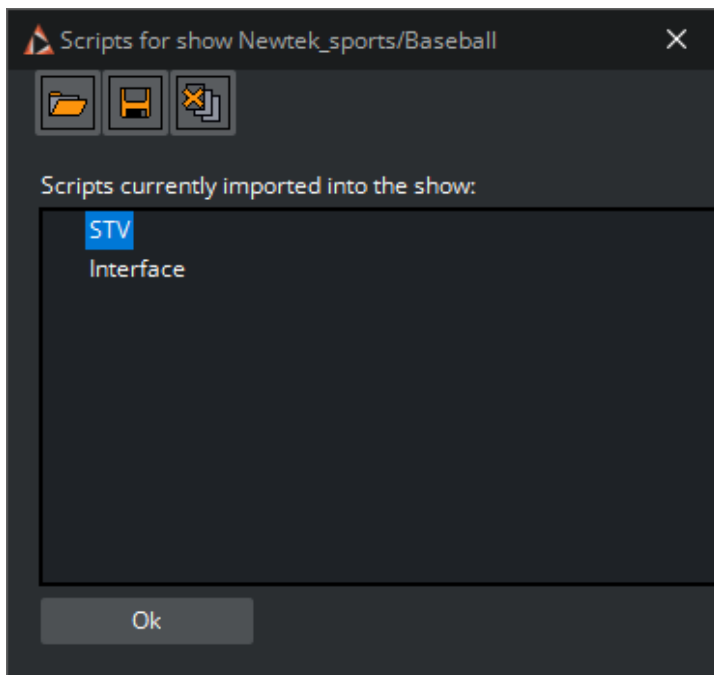
Note: `TrioCmd()` supports UTF16. If you need UTF8 support, you can use `TrioCmdUTF8()`.

Library Scripts



Script units can be added to an assigned template script, letting you write generic and specialized script units that are useful for more than one template. When using script files, a local or shared script path must be configured under the [Graphic Hub](#) settings.

Script Manager



The **Manage Show Scripts** button opens a window where you can import, export, and delete VBScript files from the current show.

Assigning a Script to a Show

1. Open the [Show Properties](#).
2. Click the **browse** button next to the Show Script field.
3. From the Choose Show Script window select either a file script (local or shared) or a Media Sequencer script.
4. Click **OK**.

Assigning a Script to a Template

1. Right-click a template and select **Script > Assign script > New**.
2. *Optional:* Assign an existing script from the **File** or **Show** options.
3. Enter a **name** for the script, and store the script on the Media Sequencer or to a local or shared script path (see [Graphic Hub](#)).
4. Click **OK**.

Adding a Predefined Function

1. Open a template with an assigned script.
2. Click **Edit script** (see [Controls](#)).
3. Click **Predefined functions** in the [Script Directory](#).
4. Select a function, and click **Add**.

Adding a Library Script Unit

1. Open a template with an assigned script.
2. Click **Edit script** (see [Controls](#)).
3. Click **Library scripts** in the [Script Directory](#).
4. Select a script unit to use, and click **OK**.

Editing a Show Script

- Right click the Template list pane and select **Script** and **Edit Show Script** from the menu.

Executing or Editing a Template Script



- Open a template with an assigned script, and click **Execute Script** or **Edit Script** (see [Controls](#)).

7.1.3 Script Backup

It's recommended to save scripts created for a specific show together with the show on Media Sequencer. A backup of Media Sequencer files is therefore necessary.

7.1.4 Script Errors

If the script contains an error, the script window automatically opens and highlights the script error location. The script error is also reported in the [Error Messages Window](#).

7.2 Viz Template Wizard Scripting

A Viz Template Wizard template can be used to further extend a show's functionality.

This section covers the following topics:

- [Dynamically Adding Components](#)
- [Set and Get Component Values](#)
- [Set and Get Show Values](#)

7.2.1 Dynamically Adding Components

Viz Trio supports the *CreateVTWComponent* function in Viz Template Wizard for dynamically adding components at run-time.

The following is an example where a text box and a button are used to generate a label at run-time:

```
curY = AddButton.Top + AddButton.Height + 5 Sub AddButtonClick(Sender)    lbl =
CreateVTWComponent("TTWUniLabel", Sender.Parent)    lbl.Parent = Sender.Parent
lbl.Left = Sender.Left    lbl.Top = curY    curY = curY + lbl.Height + 2
SetUnicodeValue lbl, GetUnicodeValue(SampleEdit)End Sub
```

7.2.2 Set and Get Component Values

Viz Trio supports the *SetUnicodeValue* and *GetUnicodeValue* functions in Viz Template Wizard.

The following is an example of a text box being used to get and set a value in another text box without using TrioCmd:

```
Sub TWUniEdit1Change(Sender)    Text = GetUnicodeValue(TWUniEdit1)    SetUnicodeValue
TWUniEdit2, Text End Sub
```

7.2.3 Set and Get Show Values

Using Viz Template Wizard to create standard templates for a show is useful as it enables you to execute default commands for an entire show. It's therefore possible to set and get show values; however, there are some subtle differences between the two.

For example, when issuing a command such as *TrioCmd("page:read 1000")* within a Viz Template Wizard template, the page numbered 1000 is read and previewed.

However, in order to get (return) values, a command must be properly triggered by another event since all Viz Trio commands are queued. The return value is therefore QUEUED. When a top-level command is executed (from the interface or a macro) it's added to an internal queue and executed after other queued commands are finished.

In order to get return values, the code using `TrioCmd()` must be issued by another top-level command. In a VTW template this is achieved by adding Viz Trio commands to events:


```
Function OnMyButtonClick(Sender)    TrioCmd("page:read 1000")
TrioCmd("vtwtemplate:run_vtw_script GetDescription")    TrioCmd("page:read
1100")    ... End Function Function GetDescription()    returnvalue =
TrioCmd("page:getdescription")    msgbox returnvalue End Function
```

In the example above, the second command *vtwtemplate:run_vtw_script* is triggered within *GetDescription* and returns the description value.

8 Appendix

This section covers:

- [Enabling Windows Crash Dumps](#)
- [Logging](#)
- [Rapid Import Requirements](#)
- [Preview Server](#)
- [Cherry Keyboard](#)
- [Glossary](#)
- [Viz Vector Integration](#)
- [Troubleshooting Excel Database Connections](#)

8.1 Enabling Windows Crash Dumps

Log files can be a valuable in understanding and analyzing unexpected program behavior. It's also recommended to allow Microsoft Windows to generate User-mode crash dumps. This can simplify debugging, particularly if hardware or general Windows issues are affecting program behavior.

Enabling User-mode Dumps requires Windows 7 or higher. To enable, configure the following registry setting:

```
HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps
```

For more information, see the Microsoft page [Collecting User-Mode Dumps](#).

8.2 Logging

Viz Trio log files, together with log files on Media Sequencer and Viz Engine, are crucial in helping Vizrt support troubleshoot any issues related to our products.

The Viz Trio logfile can be found on: *C:\Users\USER\AppData\Roaming\Vizrt\Viz Trio*.

This section covers the following topics:

- [Viz Trio Log Files](#)
- [Viz Trio Error Messages](#)
- [Viz Engine and Media Sequencer Log Files](#)

8.2.1 Viz Trio Log Files

 **Info:** The Viz Trio logfiles can be found on: `C:\Users\USER\AppData\Roaming\Vizrt\Viz Trio`.

Viz Trio Log Levels

Viz Trio Log Levels can be specified to define the level of detail. For example, log level 0 provides information about critical errors only, while log level 9 provides information about a wide range of events.

Specify log levels in the [General](#) section of Configuration:

- **Loglevel 0:** Only error messages are logged.
- **Loglevel 1:** Warning messages are added.
- **Loglevel 2:** Timer messages are added. These show the amount of time different operations take in the program.
- **Loglevel 5:** Status messages are added. These show ordinary program events such as "page loaded", "page taken", and so on.
- **Loglevel 9:** All commands sent to the Viz Engine rendering process are added.

 **Tip:** Messages on log level 0 and 1 (error and warning messages) can also be viewed in the [Error Messages Window](#).

 **Caution:** In many cases it may be beneficial to set a low Media Sequencer Logging level, preferably to 0, since a high level of logging might affect performance when running in a production environment.

Changing Log Level

1. Click the menu option **File > Configuration** to open the Viz Trio Configuration.
2. Select [General](#) from the [User Interface](#) section.
3. In the **Log level** box, enter a log level according to the available Viz Trio Log Levels.

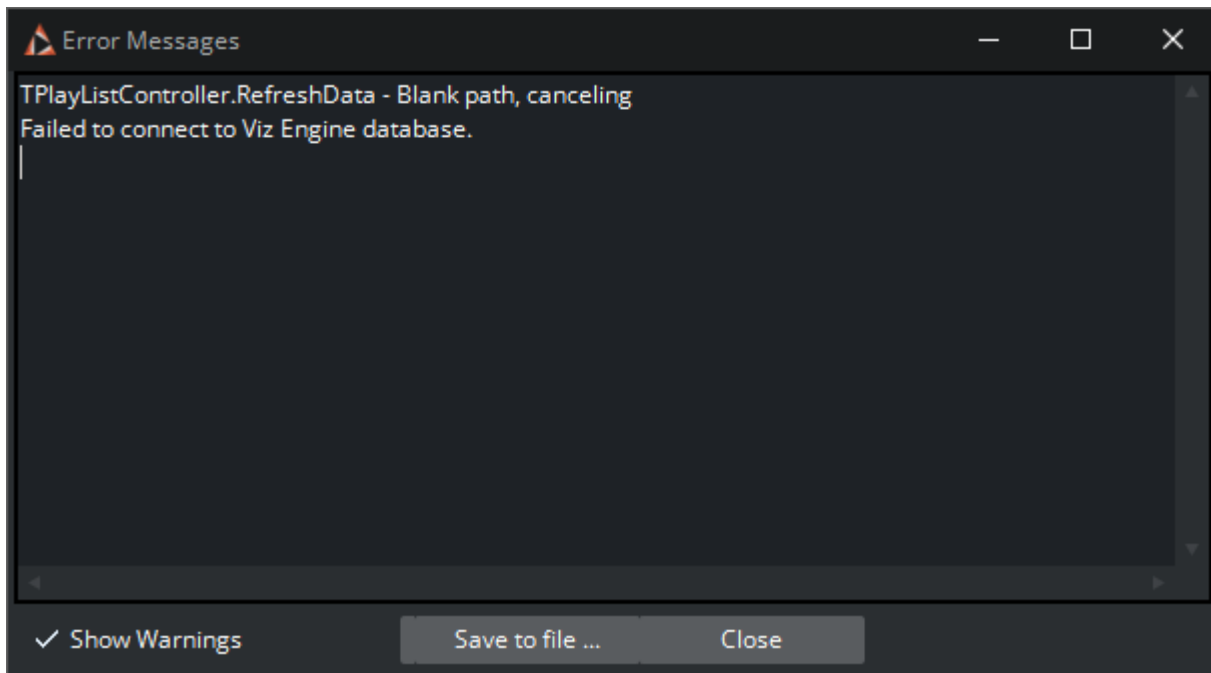
Changing the Log File Path

1. Click the **Config** button in Viz Trio's main window to open the Viz Trio Configuration.
2. Select [Paths](#) from the [User Interface](#) section.
3. In the **Logfile path** box, enter the path to where the log files should be located from now on.

8.2.2 Viz Trio Error Messages

Messages on log level 0 and 1 (error and warning messages) can be viewed in the Error Messages Window as well as in log files.

Error Messages Window



Viewing Error Messages

Click the **Errors** button in the lower right corner of the Viz Trio main window. The Error Messages Window then opens.

Viewing Warnings

1. Click the **Errors** button in the lower right corner of the Viz Trio main window.
2. In the Error Messages Window that opens, select the **Show Warnings** check box.

Saving Error Messages Window Content

1. Click the **Errors** button in the lower right corner of the Viz Trio main window.
2. In the Error Messages Window that opens, click **Save to file**.
3. In the **Save As** dialog that opens, define the path and enter a descriptive name for the .txt file.

8.2.3 Viz Engine and Media Sequencer Log Files

In addition to Viz Trio's log files created on the client side, it's recommended to read and, if required, send log files to Vizrt support that were created on Media Sequencer and Viz Engine.

Viz Engine Logs

Viz Engine log files are located in the program folder: `C:\ProgramData\Vizrt\viz4`.

For example, `VizRender_1159345015.log`

Media Sequencer Config Files

The *default.xml* files are stored in Win Vista and 7/8: `C:\ProgramData\Vizrt\Media Sequencer Engine`.

Media Sequencer Logs

The log level for Intelligent Interface specific messages on Media Sequencer can be set using Intelligent Interface (IIF).

All logging of output can be assigned a log level. Messages are only written to the log if the current log level is greater than the log level of the message. Log levels are defined as integers in the range 0-100. At a given log level, all the information specified for lower log levels is also logged.

The log level can be changed while Media Sequencer is running, and the logging output immediately reflects the new log level.

Info: For more information on how to set log levels for Media Sequencer, see the [Media Sequencer](#) documentation.

Caution: Remember to set a low log level, preferably to zero, since a high level of logging might affect performance when running in a production environment.

All Predefined Log Levels

Name	Level	What's logged
Never	0	Nothing.
Bug	5	Bugs detected in the software.
Failure	10	A permanent error, for example part of the software was disabled and no further automatic retries are attempted.
Lost link	15	A network connection or other link or precious resource was unexpectedly lost. The software might attempt to reestablish the link automatically.

Name	Level	What's logged
Error	20	Something is incorrect. An internal or external operation could not be completed successfully.
Warning	30	A cause for concern has been detected.
Notice	35	An infrequent but expected event occurred.
Connection	40	A network connection or connection to another precious resource was intentionally opened or closed.
Operation	50	High level operations that are executed as requested or planned.
Input	60	Data that is received into the system from external connections.
Output	70	Data sent from the system through external connections.
State	80	A change in the significant state of the system.
Analysis	90	Derived information and meta information generated during execution, such as running times of the executed actions.
Trace	95	The various stages the internal operations of the system perform during execution.
All	100	All possible logging information, including operation scheduling and loop iteration, and running state statistics.

8.3 Rapid Import Requirements

8.3.1 General

Rapid Import requires specific component versions:

Minimum and Recommended Versions	Optional
Graphic Hub 3.9.2+ Graphic Hub REST 2.9.2+ Media Sequencer 5.6.0+ (recommended 5.8.0*)	Viz Engine 5.2.1 *

Scene Requirements

To be able to import the scene via Rapid Import, it must be created or re-saved in Viz Artist version 5.1 or newer.

Optional Requirements*:

1. Versions of Media Sequencer prior to 5.8.0, do not set a template description during import.
2. Versions of Media Sequencer prior to 5.8.0, do not detect expert mode fields during import, which means they are displayed even when the Expert Mode setting is not checked.
3. If Viz Engine 5.2.1 is used, there are no special scene requirements.

8.3.2 Logging

Import information can be found in the Viz Trio log file `%AppData%/Roaming/Vizrt/Viz Trio/Logs`. The Rapid Import logs prefix is *Rapid Import*:

General information is also printed in the template import information panel.

The log file keeps a comprehensive chronological record of scene import operations, tracking everything from initial validation through completion with detailed progress markers. It captures critical diagnostic information about potential failure points including validation errors, missing data, and compatibility issues, enabling users to quickly troubleshoot import problems.

8.4 Preview Server

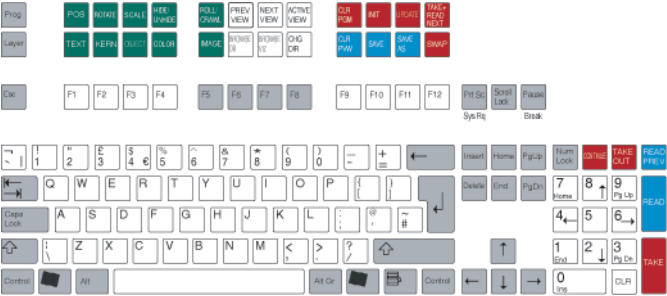
Preview Server is a Windows service that manages a pool of Viz Engine processes. Client software, such as Viz Multichannel, Viz Pilot, and Viz Trio, can send requests to Preview Server, requesting available Viz Engine processes. The responding Viz Engine process may provide frames for thumbnails and snapshots in an ongoing preview process.

As its name indicates, Preview Server provides Viz Engine connections to clients during the creative preview process. When the output is finalized and actually being played out, Viz Engine processes provided by Preview Server are replaced by the Viz Engine specified directly by each client.

The preview snapshots provide the user with an indication of the look of the graphics when they are played out in high resolution on a Viz Engine. Preview Server provides load balancing, and you can use it to create a redundant renderer pool.

For more details, see the [Preview Server Administrator Guide](#).

8.5 Cherry Keyboard



Note: Early versions of Viz Trio were shipped with an older version of the Cherry Keyboard. If you are still using this keyboard, you should still be able to use it with current versions of Viz Trio.

The keyboard contains two rows with extra function keys which have been assigned to different Viz Trio actions.

Note: The keyboard has its own configuration software. A Viz Trio configuration file must be loaded to create the correct keyboard map. A keyboard mapping file must be imported to assign the correct actions to the keys. This is pre-installed on all Viz Trio clients, so there is normally no need to change these settings.

This section covers the following topics:

- [Editing Keys \(green\)](#)
- [Navigation Keys \(white\)](#)
- [Program Channel Keys \(red\)](#)
- [Preview Channel Keys \(blue\)](#)
- [Program and Preview Channel Keys \(red and blue\)](#)

8.5.1 Editing Keys (green)



The green keys perform editing operations.

Note: The current tab-field must have the property of the key exposed for editing. If not, the key has no effect and an error message is written in the log file when the key is pressed.

- **POS:** Displays the position editor.
- **ROTATE:** Displays the rotation editor.
- **TEXT:** Displays the text editor.

- **KERN:** Displays the character kerning editor.
- **SCALE:** Displays the scaling editor.
- **OBJECT:** Displays the object pool where 2D and 3D objects can be browsed for.
- **HIDE/UNHIDE:** Hides/shows the tab-field.
- **COLOR:** Shows the pool of colors.
- **ROLL/CRAVL:** Opens the scroller editor.
- **IMAGE:** Opens the image pool.

8.5.2 Navigation Keys (white)



The white keys shift between different views and editors in the program.

- **PREV VIEW:** If extra page views have been defined, this key shifts the view to the one above the currently active view, see [Add Page List View](#).
- **NEXT VIEW:** If extra page views have been defined, this key shifts the view to the one below the currently active view.
- **ACTIVE VIEW:** When the program has the active focus on some part outside the current page view, pressing this key brings the page view back to an active state; navigate between the pages with the arrow keys.
- **BROWSE DB:** When on an image tab-field, pressing this key opens the [Search Media](#) frame to allow media searches.
- **BROWSE VIZ:** When on an image tab-field, pressing this key opens Viz Engine's image database.
- **CHG DIR:** Displays the change directory or show window.

8.5.3 Program Channel Keys (red)



The red function keys affect actions on the program channel.

- **CLR PGM:** Clears all loaded content on the program channel.
- **INIT:** Initializes the current show on both the program and preview channel.
- **UPDATE:** When a change has been made to a page that is already On-air, pressing **UPDATE** incorporates the changes without running any animations. This is typically used for fixing typing errors. If a page is changed and **Take** is used instead of **Update**, all animation directors in the scene are executed, which normally creates an unwanted effect.
- **TAKE+ READ NEXT:** Takes the page currently read, and reads the next one in the list.
- **SWAP:** The swap key takes to air what is currently read and visible in preview, and it takes off what is currently on-air and reads that page again.

8.5.4 Preview Channel Keys (blue)



The blue keys affect actions on the preview channel.

- **CLR PVW:** Clears the preview channel.
- **SAVE:** Saves the page currently shown on the preview channel.
- **SAVE AS:** Saves the page currently shown in preview to the page number entered.

8.5.5 Program and Preview Channel Keys (red and blue)



The blue keys affect actions on the preview channel; the red keys affect actions on the program channel.

- **CONTINUE:** When a scene based page halts at a stop point, pressing **CONTINUE** resumes the animation.
- **TAKE OUT:** If transition logic is used, **TAKE OUT** takes out any page loaded in the layer that is currently read. If transition logic is not used, **TAKE OUT** performs a clear, which is a hard cut. To obtain a smooth out animation, the scene must be designed with a stop point and an out animation, and **CONTINUE** must be used to take out the page.
- **READ PREV:** Reads the previous page in the page list.
- **READ:** Reads the page currently highlighted by the cursor.
- **TAKE:** Takes the page that is currently read.

See Also

- Keyboard Shortcuts and Macros

8.6 Glossary

The following terms are used for Viz Trio:

- **Control Plug-ins:** A graphics scene can contain all sorts of objects that can be controlled from a template such as text, back plates, images, colors and more. The graphics designer uses control plug-ins to expose objects as tab fields.
- **Control Object:** Every scene with control plug-ins must have one instance of the control object plug-in at the root level of a scene tree. Control Object reads the information from all other control plug-ins. When a scene is imported to Viz Template Wizard, it reads the information about other lower level control plug-ins through Control Object.
- **Forked Execution:** With the new profile configuration, Viz Trio now supports forked execution by having more than one graphics render engine per channel. Simply put, you can trigger the same scene on multiple render engines where one can act as your backup.
- **MAM-system:** A Media Asset Management (MAM) system takes care of ingestion, annotation, cataloging, storage, retrieval and distribution of digital media assets. Viz Trio works together with Vizrt's MAM-system Viz One.
- **Newsroom Component:** In a Newsroom Computer System (NCS), the Newsroom Component (NrC) is used to add data elements to a story. The user is typically a journalist working on a story. The NrC is an embedded application in the NCS that connects to a database of templates. The templates can be filled with text, images, video and maps.
- **Page:** A page is based on a template and saved as a data instance of the template. A page contains a set of data and references to where data (for example, images and video clips) can be found. When a page is loaded, all static elements are loaded from the template and the variable elements (items/tab fields) are set to what they were when the page was saved. Therefore, a page is not saved as a complete scene, only the values of the tab fields are saved together with a reference to the template-scene.

 **Note:** Only **pages** can be taken On Air.

- **Scene:** A scene is built in Viz Artist. It can be a single scene, or one part (layer) of a combination of scenes (transition logic).
- **Template:** A template is based on a Viz Artist scene, and is created by Viz Trio on-the-fly while importing it to Viz Trio. The template is used to create pages that are added to a show for playout. A template can be based on one or several (transition logic) scenes (known as a combination template).
- **Transition Logic Scene:** A set of scenes built in Viz Artist. A transition logic scene contains one scene that controls the state of or toggles a set of scenes (layers). The layered scenes are used by the controlling scene to toggle in and out the layered scenes, using pre-configured or customized transitions effects, without the need to take scenes already on-air, off-air. For example: A lower third may be on-air at the same time as a bug, and the lower third may be taken off-air without taking the bug off-air or reanimating it.
- **Viz Artist:** The design tool where the graphics scenes and all animations are created.
- **Viz Engine:** The render output engine used for playout of graphics and video.

8.7 Viz Vector Integration

8.7.1 Macro Calls to Integrated Systems

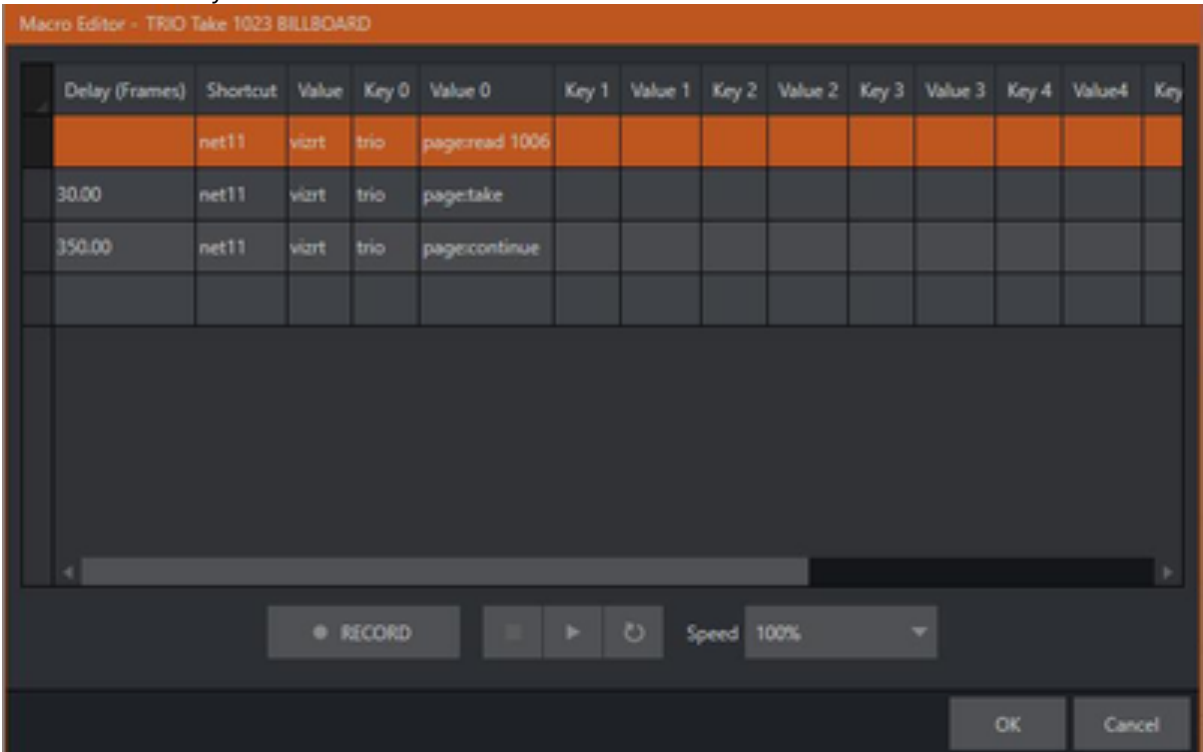
You can use macros to send commands to external systems.

Sample Setup: Viz Vector Plus to Viz Trio

Viz Vector Plus can connect to a Viz Trio system, to control CG graphics. Viz Vector Plus communicates with Viz Trio through commands over NDI.

✓ **Tip:** For some general background on how to integrate with external system through NDI, refer to the NewTek publication [Automation & Integration Guide](#).

1. Most setup is performed on the Viz Trio side.
For details on how to handle macros, please check [Working with Macro Commands](#).
2. The *net* shortcut entry in Viz Vector Plus macros can be used to send commands over a specific NDI input. The workflow is:
 - a. A Viz Engine (either Program or Viz Trio Preview) is set up to output NDI.
 - b. In Viz Vector Plus, the NDI output from the Viz Engine is assigned as one of the inputs.
 - c. The Macro system in Viz Vector Plus can be used to send commands via the NDI streams of the inputs.
3. As an example, if the NDI output from the Viz Engine is on input number #11 in Viz Vector Plus, the shortcut entry *net11* can be used to address this NDI stream.
 - a. **Value:** *vizrt*
 - b. **Key:** *trio*
4. The value of the key is the command to send to Viz Trio.



5. On the Viz Engine side, the Viz Trio commands are forwarded to a Viz Trio client on localhost (or to the one specified in configuration).

Viz Engine configuration

- If the Viz Engine receiving the macro commands is the Viz Trio local *Preview* engine, no configuration is needed.
- If another Viz Engine (for example, on *Program*) is receiving the macro commands from Viz Vector Plus and these commands are sent to a Viz Trio client on another host, the configuration on this Viz Engine must specify the host address of the Viz Trio client:

```
180 smm_master_eng_port = NONE
181 smm_thread_count = 1
182 ## smm content will be send from spec
183 smm_enable_master_poll = 0
184 trio_address = 10.0.101.98
185 trio_tcp_port = 6200
186 MulticastIp = 224.1.1.1
187 EnableUdpAndMulticast = 0
```

The relevant config settings in the *cfg config file for the Viz Engine are:

- trio_address = 127.0.0.1
- trio_tcp_port = 6200



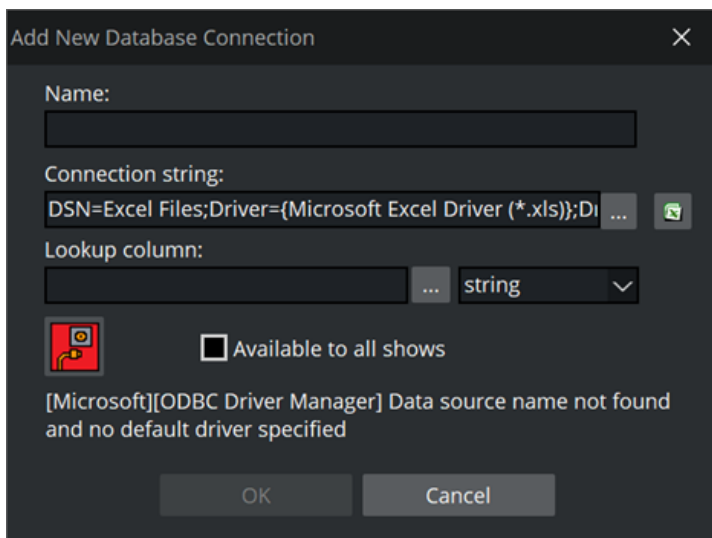
Note: Viz Trio requires an additional license.

8.8 Troubleshooting Excel Database Connections

For Database connectivity with Excel files (.xls, .xlsm, .xlsb and .xlsx), you may need to install **Microsoft Access Database Engine 2016 Redistributable** (accessdatabaseengine_X64.exe) if Microsoft Office is not installed on the machine.

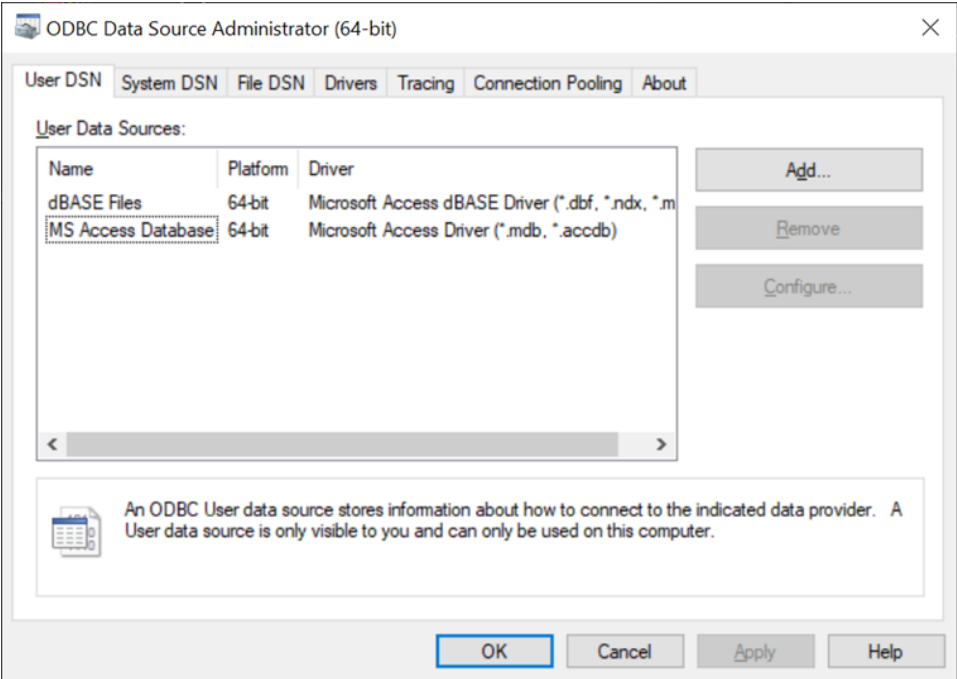
It can be found on the Microsoft site <https://www.microsoft.com/en-us/download/details.aspx?id=54920>.

In case of insufficient system set up, the following error may occur **[Microsoft][ODBC Driver Manager] Data source name not found and no default driver specified**.



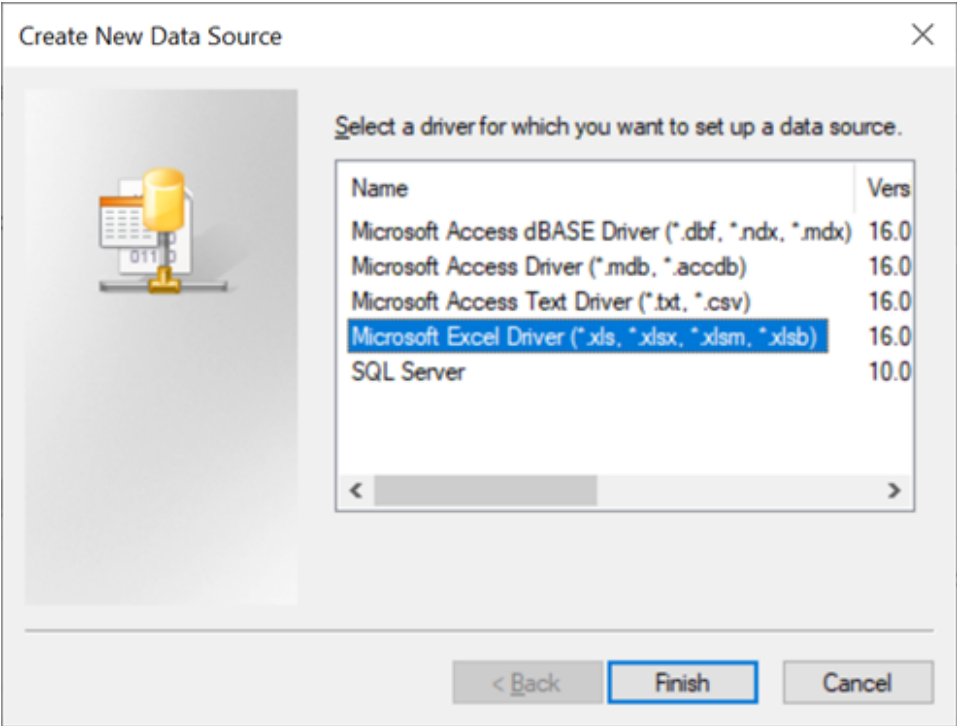
This happens when there is no ODBC Data Source set in the system (for instance the Microsoft Office is not installed) or if the data source exists but its name is specified wrong in a connection string.

To check the ODBC data sources, you should run the 64-bit version of ODBC Data Sources Administrator application (it is crucial to use the 64-bit version and not the 32-bit, because Viz Trio 4.x is 64-bit application).



On the image above you can see there is no existing data source that uses Microsoft Excel Driver. To set up a new one click **Add** button.

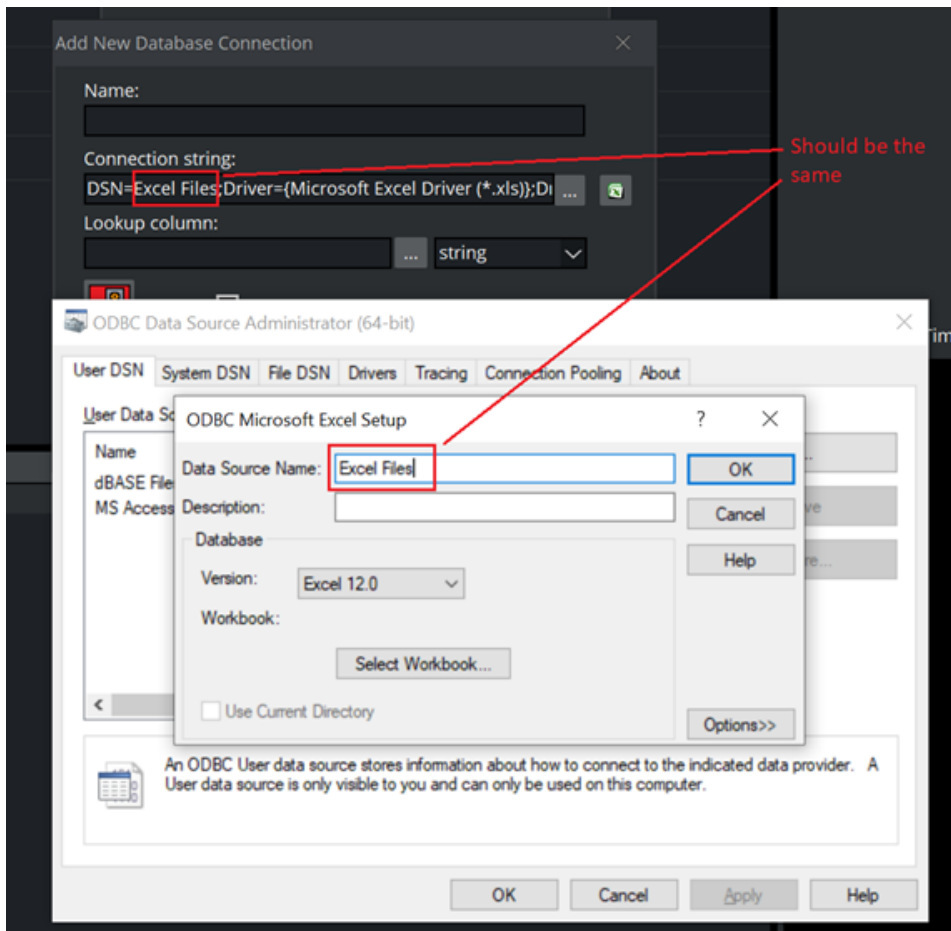
In the wizard, select the **Microsoft Excel Driver** option and click **Finish**.



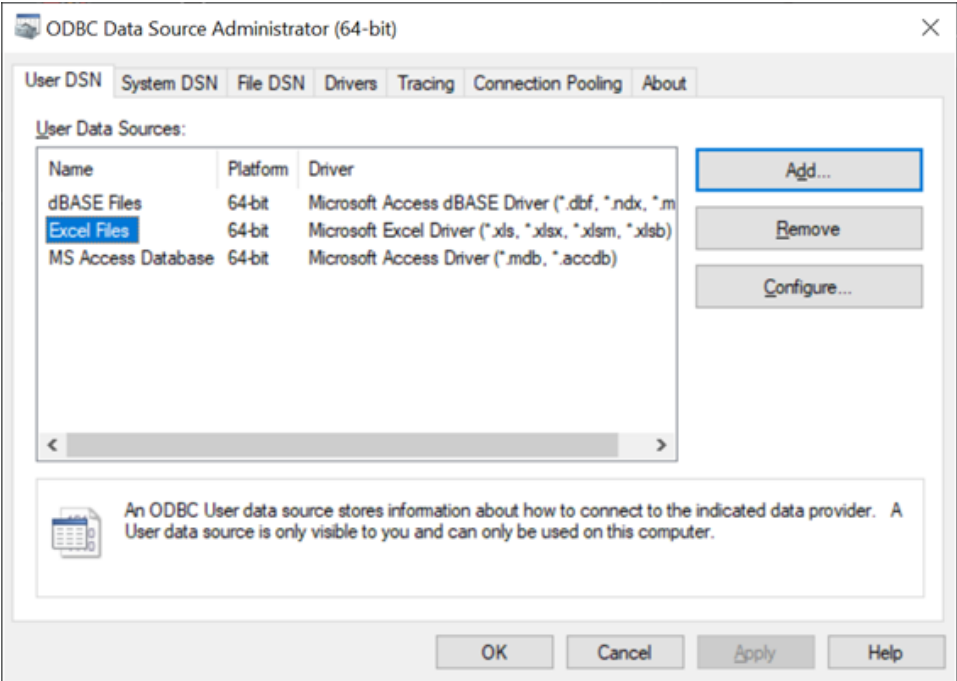
In the following dialog box, you should set the new data source name. The data source name (DSN) is the text that is mentioned in the connection string in Viz Trio, which is crucial to be the same as in Viz Trio.

By default, Viz Trio uses the **Excel Files** text as a data source name (this name is also the default name that Microsoft Office may automatically create during installation).

If you want the name to be different, you must adjust this name in Viz Trio in the connection strings, for all your database connections that use Excel files:

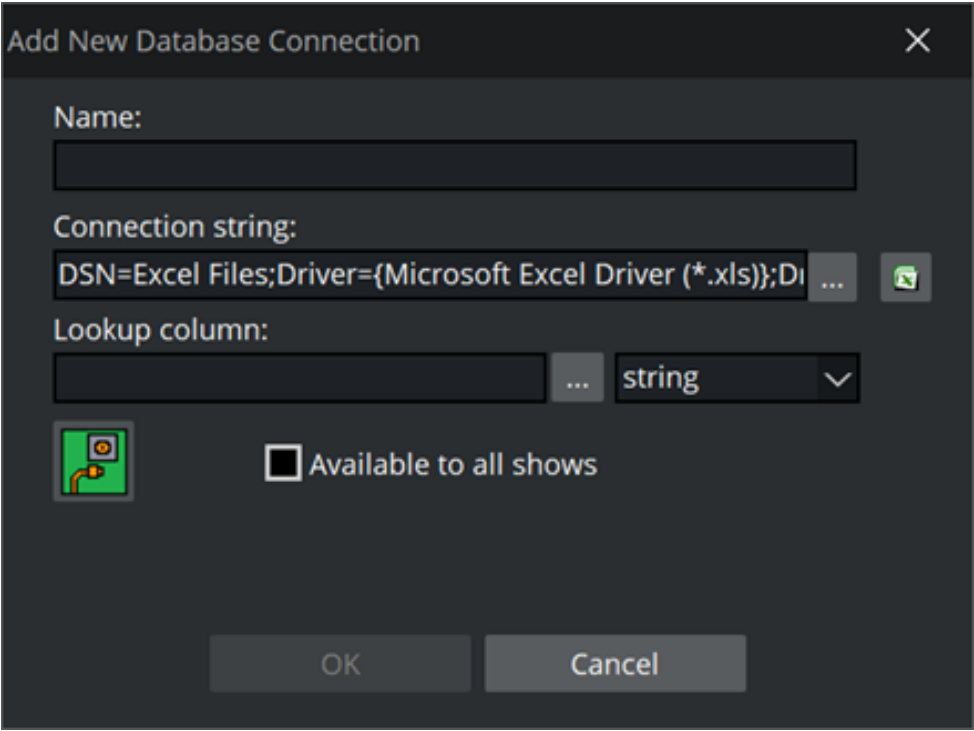


After setting a name, click **OK** to apply. Make sure the data source name appears in the list:



Note: You may set up the data source using the **User DSN** or **System DSN** tabs. The data sources set in the **System DSN** tab are shared to all users on this computer, and the data sources set in the **User DSN** tab can only be used by current user.

After you established the data source, go back to Viz Trio and test the connection again:



If you use the data source with the name that is not the default, you must edit the DSN parameter value in the connection string text box.