



# Yamcs Studio User Guide

October, 1st 2015

[www.yamcs.org](http://www.yamcs.org)



# **Yamcs Studio User Guide**

YAMCSSTCORE-SA-MA-001

This version was published October, 1st 2015  
A later version of this document may be available at [www.yamcs.org](http://www.yamcs.org)

© Copyright 2015 – Space Applications Services, NV

# Table of Contents

<b>1. Introduction</b>	<b>2</b>
Overview	3
Installation	4
First Steps	5
Understanding the User Interface	7
Connecting to Yamcs	10
<b>2. Running Displays</b>	<b>13</b>
OPI Runtime Perspective	14
<b>3. Editing Displays</b>	<b>16</b>
OPI Editor Perspective	17
Resource Management	20
Rules & Scripts	21
<b>4. Views</b>	<b>22</b>
Archive	23
Event Log	25
Alarms	26
Command Stack	27
Command History	31
Command Queues	33
<b>5. Troubleshooting</b>	<b>34</b>
Capturing Log Output	35

# Chapter 1. Introduction

## 1.1. Overview

### Brief History of Yamcs

Yamcs started as a server software first and foremost. While it initially started as a swiss-army knife to fill gaps in existing traditional mission control systems, it gradually grew to cover the whole spectrum of TM processing and TC commanding. Missions can have very specific software requirements, and often include a varied stack of software. Over the years Yamcs was extended in various ways to play nice with different kinds of TM and TC software.

Along the way, standalone client GUIs were developed as the need arose. This includes Archive Browser, Event Viewer, Packet Viewer and Yamcs Monitor. These tools are being used in many missions.

For many years, however, the main thing that was felt missing from the Yamcs software stack, was a display solution. And this is where Yamcs Studio comes into play.

### Yamcs Studio

Yamcs Studio is a desktop frontend to Yamcs. Its main attraction is its support for operator displays, but it also includes other facets that cover TC commanding and insight into various runtime aspects of Yamcs. Most of our legacy client GUIs have also been ported into Yamcs Studio (with the exception of the Packet Viewer), for an integrated solution.

Yamcs can be made to integrate with display software other than Yamcs Studio (and in fact, this is often the case in long-running missions where Yamcs was added in the mix after the project's initial conception), but there are advantages to working with Yamcs Studio:

- Increased semantical coherence
- Single point of contact
- Opportunities for customisation that covers both server and client
- Integrated views operational views

### Technology

Yamcs Studio is an Eclipse RCP application, and builds upon Open Source software libraries like CS-Studio, Netty, Protobuf and of course our own Yamcs API.

The main programming language is Java 8.

### License

Yamcs Studio follows a similar licensing scheme as Yamcs Server. The core of Yamcs Studio is open-source and licensed under the Eclipse Public License. Mission-specific extensions can be developed on a case-by-case basis and under custom licenses.

We believe that having an open-source core, is not only fun and exciting, but that this increases the quality of our products and benefits all of our customers equally.

## 1.2. Installation

### Install Java 8

You will need Oracle Java 8 installed. We currently recommend the latest [Oracle JDK 8](#).

### Download Yamcs Studio

[Download](#) the latest Yamcs Studio release for your platform. Extract to your preferred location and just launch it. When it asks you to choose a workspace, choose a new directory inside for example your home directory. Workspaces contain displays, scripts and user preferences. By default your workspace will be populated with a few sample projects. These projects contain displays that show simulated parameters as produced by a default-configured Yamcs Server.

### Troubleshooting

Most problems related to starting Yamcs Studio, have to do with Java not being correctly detected, or by trying to launch Yamcs Studio with an old version of Java. Both of these issues are usually resolved by installing Oracle JDK 8.

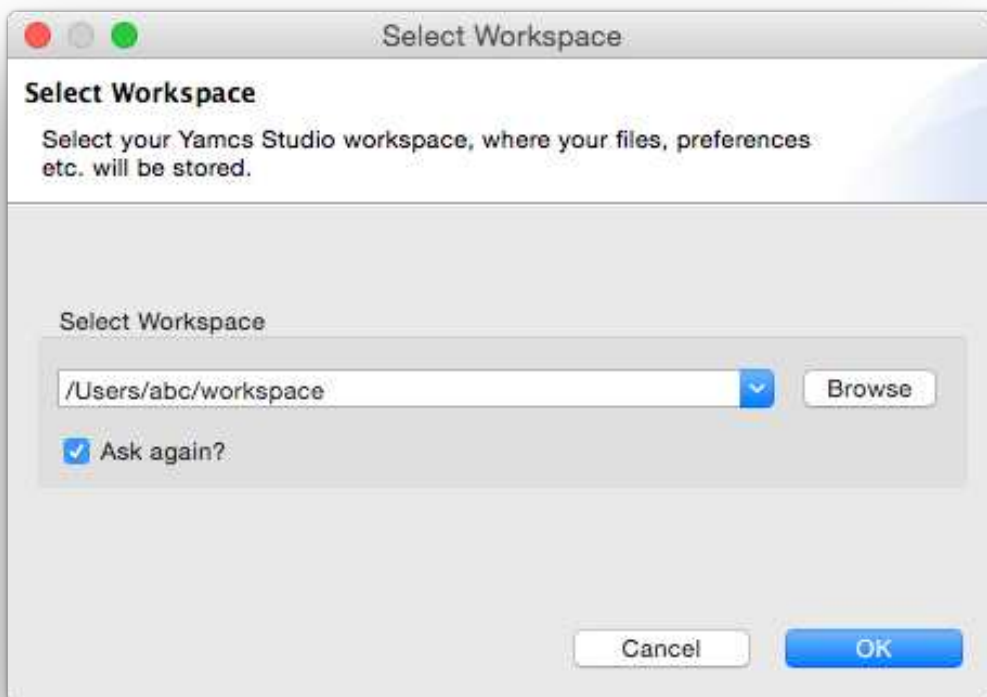
In case that didn't help, Try defining the `-vm` property in the root `yamcs-studio.ini` file. Refer to [these instructions](#).

## 1.3. First Steps

### Launching Yamcs Studio

When you launch Yamcs Studio for the first time it will ask you to choose a workspace. A workspace is where you store your resources (e.g. a display file).

With Yamcs Studio, you are always working on one workspace at a time. Usually workspaces are fairly static, and you can often do with just one of them. If you untick the Ask again? option you will no longer see this message at startup.



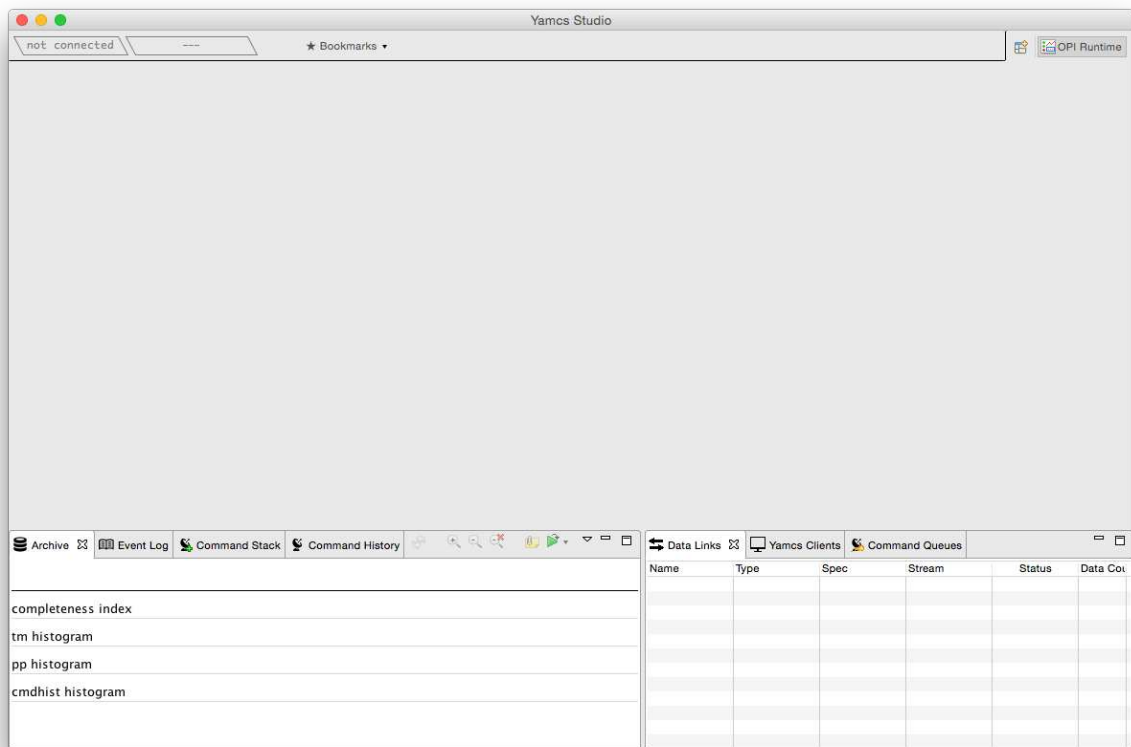
Choose your preferred location, and click OK.



If you unticked the Ask again? option, but you want to switch workspace at a later moment, open **File > Switch Workspace...** from the window menu to choose a different directory.

### Empty Workspace

Yamcs Studio is now launched and you should see an empty workspace with the default window arrangement:



The empty area in the middle is where displays will open. In the middle of the screen we have a




Yamcs Studio has two different modes (called *perspectives*). OPI Editor and OPI Runtime. When Yamcs Studio is launched for the first time the user will be welcomed with the default OPI Runtime perspective, which is used during realtime operations, or for testing out displays with live telemetry.

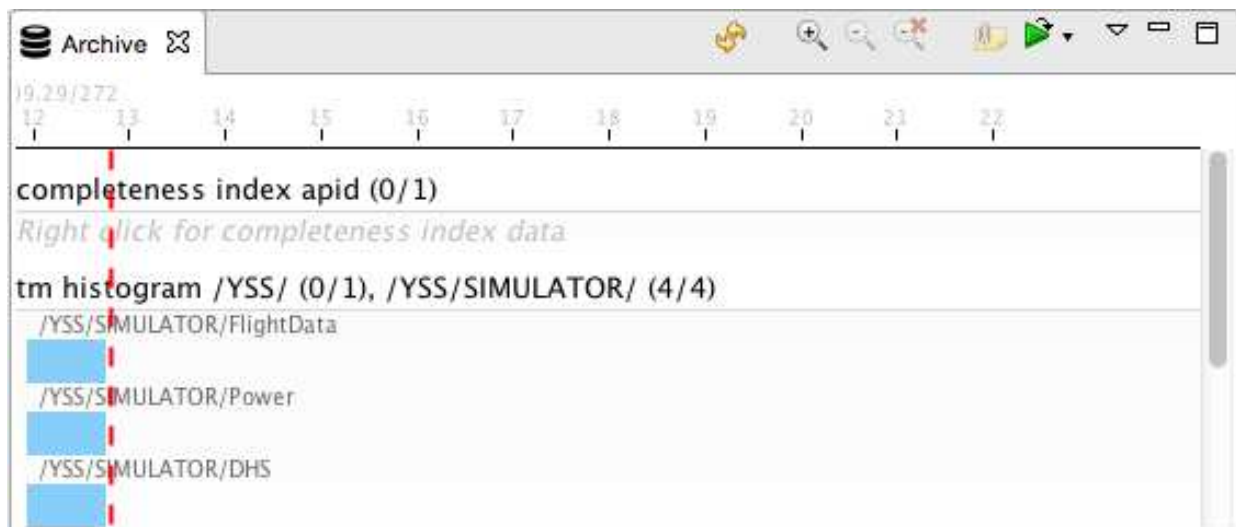


## 1.4. Understanding the User Interface

Yamcs Studio is composed out of multiple views that are arranged together in a perspective. The user has great flexibility in modifying the default arrangement to his liking.

### Views

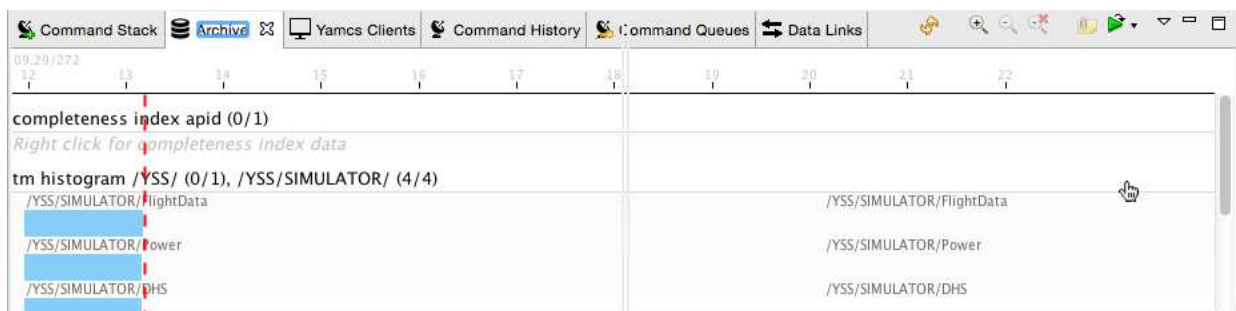
Views all share the same user interface organization. On the left you see a tab with the view icon, followed by a title, and then a close icon. On the outer right there are actions to  Minimize or  Maximize the view. Some views (such as the one in the screenshot) also have a third pull-down icon  with view-specific actions in it. Most views, though, add dedicated colored icons next to the standard icons. The pull-down menu is used to hide less-often used actions.



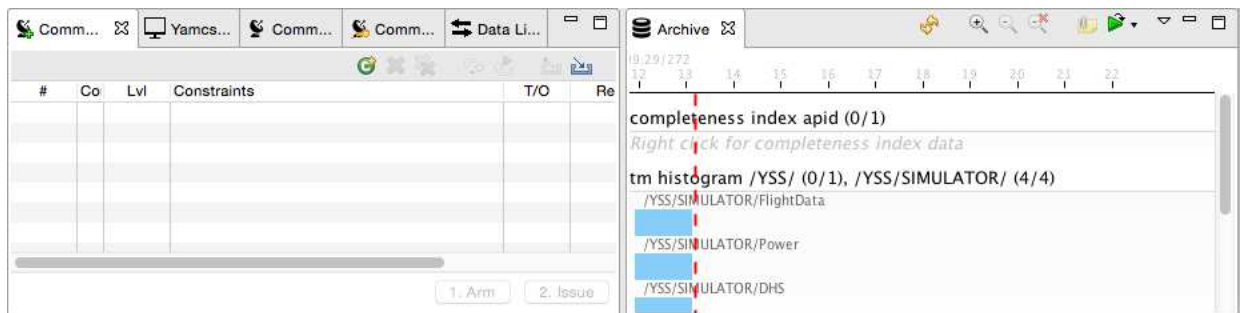
To reopen a view which you closed earlier, or to open another view choose Window > Show View.

Views can be resized, moved and stacked. This allows you to customize your workspace to your own personal preference.

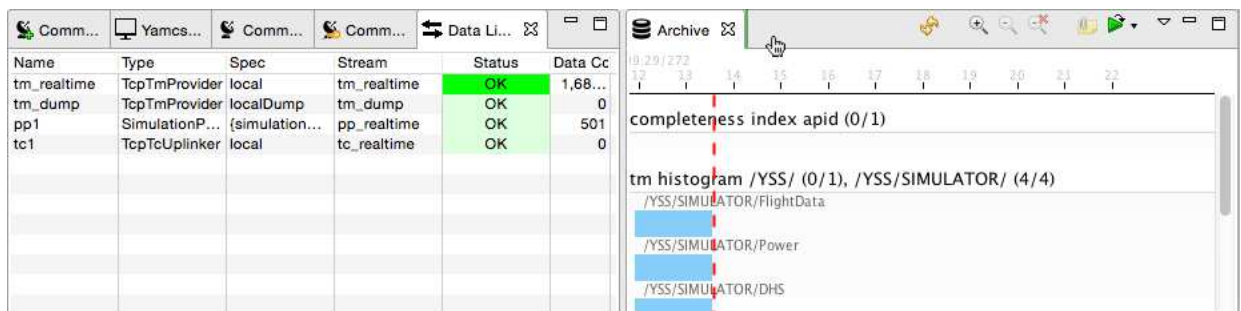
For example, let's say we want to put the Archive view in its own dedicated location. Click on the tab title, and while holding the mouse down, drag towards the right. If you move far enough, you will notice an outline suggesting the view's new position (this may look slightly different on your platform).



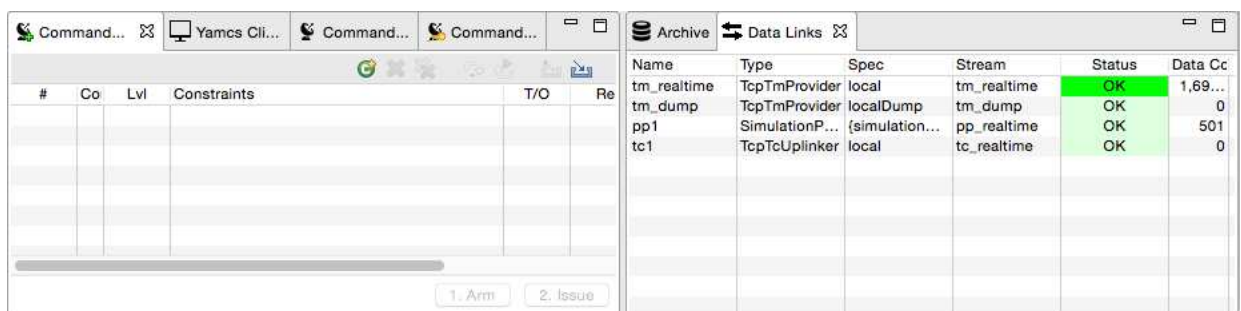
Release the mouse to confirm this view's new position.



Let's say we want to move the Data Links view to the right as well. Again, click its title and drag your mouse next to the Archive tab. You will see a green bar suggesting this tab's new placement.



Release the mouse to confirm this view's new position.



Feel free to experiment some more with the drag feature. As you try dragging to different locations, you will notice that Yamcs Studio has several hot spots where you can attach your views. For example, you can detach windows by dragging them outside of your application window. This provides additional screen space if your workstation supports multiple monitors.


When you close Yamcs Studio and reopen it, it will restore your preferred view and window arrangement.



Yamcs Studio stores the information about your view arrangement in a `.metadata` folder inside your workspace. This is how it knows how to restore this information through restarts. If you share your workspace with other users through a version control system, you should consider *not* committing this `.metadata` folder. This way everybody can have his own preferred arrangement without colliding with each other.

## Perspectives

Perspectives contain an organization of views. As you were performing the above actions with views, you were working within a certain *perspective*.

In the top right bar, you can see the Perspective Switcher. This is where you choose your current perspective. By default Yamcs Studio puts you in the OPI Runtime perspective, but by clicking the plus icon  you can switch to the OPI Editor perspective, which has a different arrangement of views.

Again you can modify the views in this perspective to your heart's content, but as a general precaution we would advise that you distinguish between 'Running Displays' (OPI Runtime), and 'Editing Displays' (OPI Editor). In future versions of Yamcs Studio we may make this distinction more apparent, or even go as far as to offer two different products.

Notice, as you go back to the OPI Runtime perspective, that your earlier view arrangement is nicely restored.

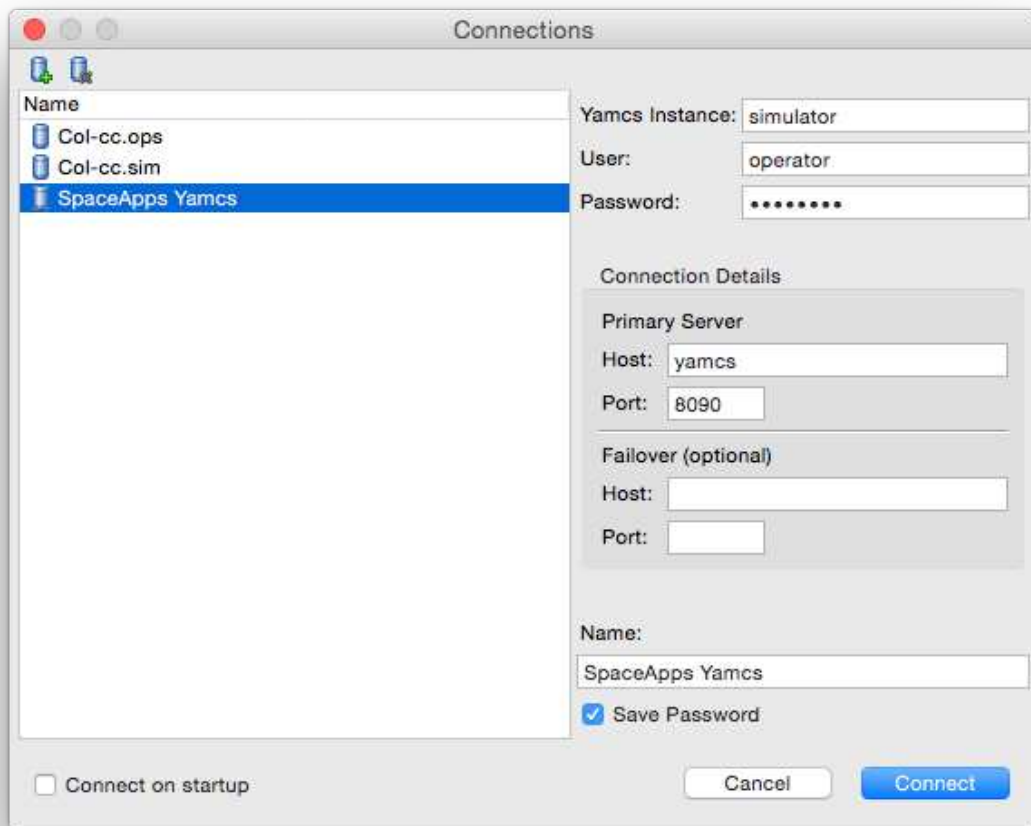
If at any time you want to reset your perspective to the defaults, select Window > Reset Perspective....



## 1.5. Connecting to Yamcs

Yamcs Studio is a client application that's meant to be connected with Yamcs Server.

Yamcs Server, or 'Yamcs' handles the processing, archiving and dispatching of telemetry data. Yamcs Studio is one of the possible Yamcs clients for receiving telemetry data.

To configure a Yamcs connection, select File > Connect.... This will open the Connections window where you can manage your connections. For many missions, one connection will do just fine, but depending on how Yamcs is deployed at your site, you may have multiple Yamcs instances on the same server, or even multiple Yamcs servers.



Click  Add Server to add a server connection, or  Remove Server to remove the selected server connection.

### Connection Properties

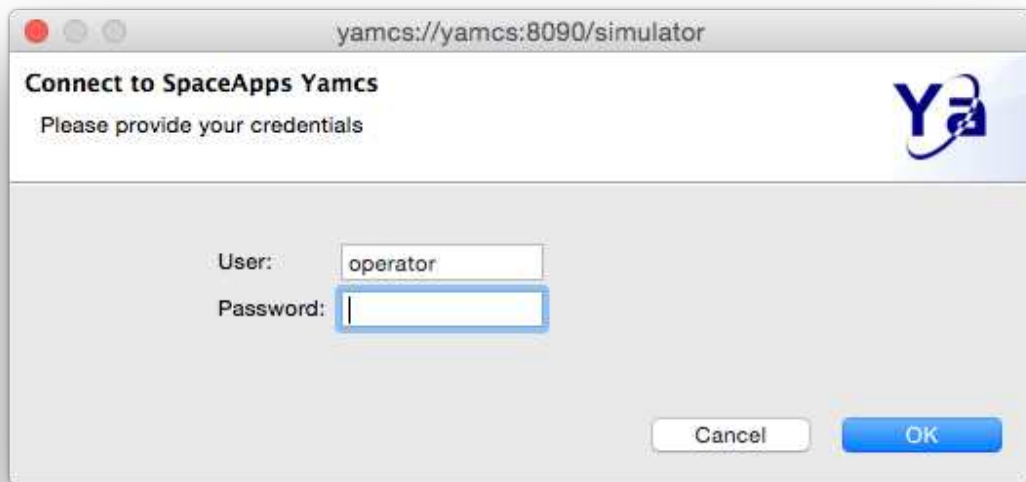
The right panel contains editable details for the selected server connection. We document the available properties below, but if you're unsure what to fill in, ask details to the person that is responsible for installing Yamcs at your site.

Yamcs Instance	Required	Yamcs can run multiple instances in parallel. You can think of instances like different environments, where every instance is completely separated from the other instance. While Yamcs Server may be running multiple instances in parallel, Yamcs Studio will always connects the user to one specific instance, which you have to configure here.
User / Password	Optional	If your Yamcs instance is secured, fill in your user and password here.
Primary Server	Required	Specify your actual host and port connection details here. The port is usually 8090.
Failover Server	Optional	<p>If you specify a second host/port configuration, then Yamcs Studio will automatically failover to this second server in case connection with the primary server could not be established, or was lost.</p> <p>On the server-end, this setup requires two distinct Yamcs servers that are being kept in sync.</p>
Name	Required	You can give your configuration a name of your choosing. This name will be used to represent this connection in the left panel of the Connections window.
Save Password	Optional	If you prefer not to enter your password at every occasion, tick this box to save your password to disk. Please be aware that your password will be saved in a manner that is difficult, but not impossible, for an intruder to break.

## Connecting

All changes you make are automatically saved when you click Connect. If you want to discard your changes click Cancel.

Select the Connect on startup option, if you would like Yamcs Studio to automatically reconnect to the last used Yamcs instance during start-up. If this connection requires privileges and you chose not to save your password to disk, you will see a specialised login window everytime you start Yamcs Studio:



Connection preferences are stored in a hidden folder under your home directory, and will continue functioning whenever you upgrade your copy of Yamcs Studio.

You can verify that your copy of Yamcs Studio is properly connected by looking at the top left processor indicator of the OPI Runtime perspective:

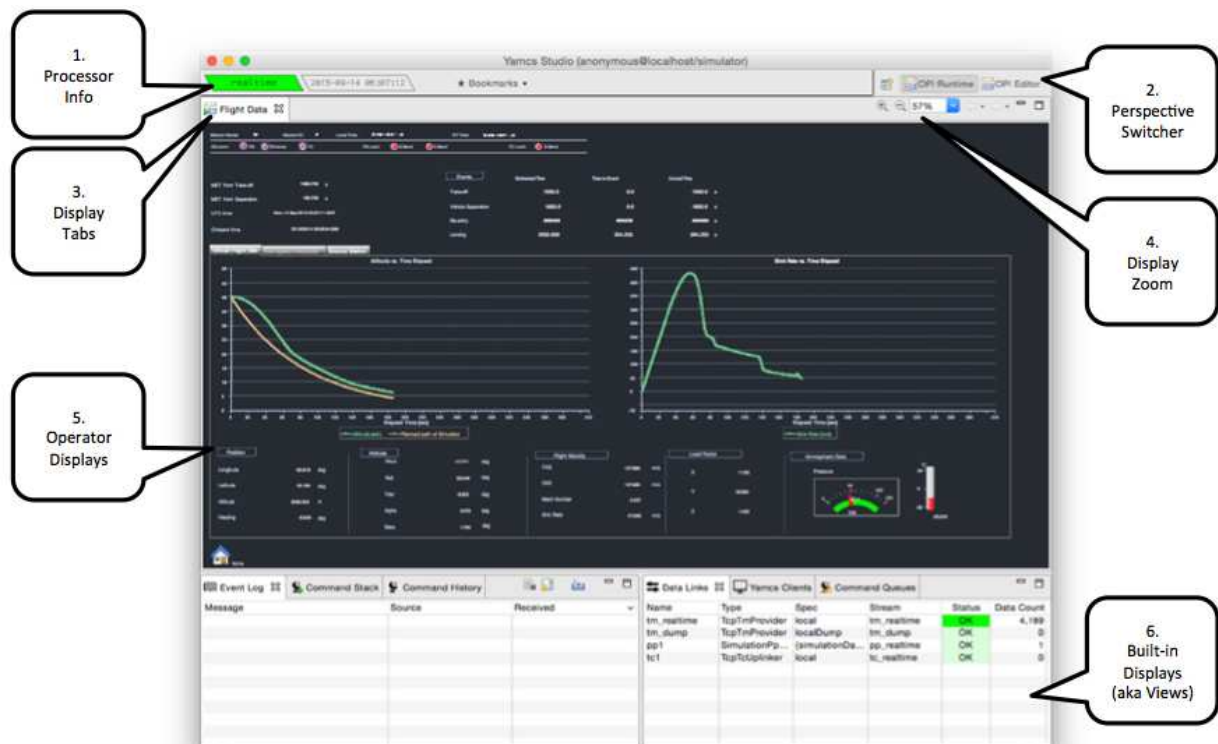


If it says realtime, then you've successfully connected.

## Chapter 2. Running Displays

## 2.1. OPI Runtime Perspective

The OPI Runtime perspective is useful for realtime operations, or for testing out displays as they are being built. The default layout looks like this:



### 1. Processor Info

This zone holds two status indicators. The first indicator light shows the processor that Yamcs Studio is currently listening to. Yamcs supports many concurrent processors (realtime, replay channels). By default Yamcs Studio will always connect to `realtime`.

Next to that we see a second indicator which currently shows the processor time as provided by Yamcs. The simulator outputs generation times equal to the local system clock. If however we were to start a replay of archived data, we would notice this time adjusting to the location of our replay channel.

### 2. Perspective Switcher

When you launch Yamcs Studio it will open in `OPI Runtime` mode (OPI means Operator Interface). With the perspective switcher you can switch Yamcs Studio to the `OPI Editor` mode. Doing so will store and close your current arrangement of windows and views, and will open a different arrangement that is optimised for editing displays.

Note that it is possible to make builds of Yamcs Studio that include *only* the runtime perspective. This can significantly improve UX during operations.

### 3. Display Tabs

Displays open in different tabs. These are not normal tabs, though. By clicking and dragging these tabs we can easily create split screens, or different tab stacks. We can also drag a tab



out of its parent window into a new window. In fact, Yamcs Studio is optimised for multi-monitor systems. Window layouts are restored even through restarts of Yamcs Studio.

#### 4. Display Zoom

The above display was configured in such a way that it automatically stretches (while preserving aspect ratio) to fit the available screen space. This behaviour can be turned on or off by the display author. Regardless of its setting, as a display user we can always zoom in or out of the display using these controls.

#### 5. Operator Displays

This area contains displays that were authored in the OPI Editor perspective. Displays contain any number of widgets. Most widgets can be connected to TM, which will also make them alarm-sensitive. In practice this means that they will be highlighted with different decorations depending on the alarm level. There are also things like button widgets which can for example open other displays, or launch a telecommand, or open dialog boxes, etc. All widgets are highly customisable using scripts and/or rules (rules are a user-friendly layer on top of scripts). We are in the process of documenting and expanding the library of functions that can be called from scripts.

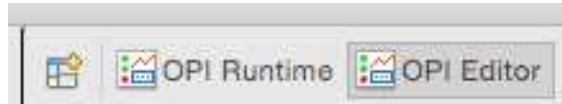
#### 6. Built-In Displays

Yamcs Studio comes with an array of built-in displays that offer more dynamic views on different aspects of Yamcs. These built-in displays (or *Views*, as we call them inside Yamcs Studio) include such things as commanding, event logging, alarm overviews (upcoming) and archive insight.

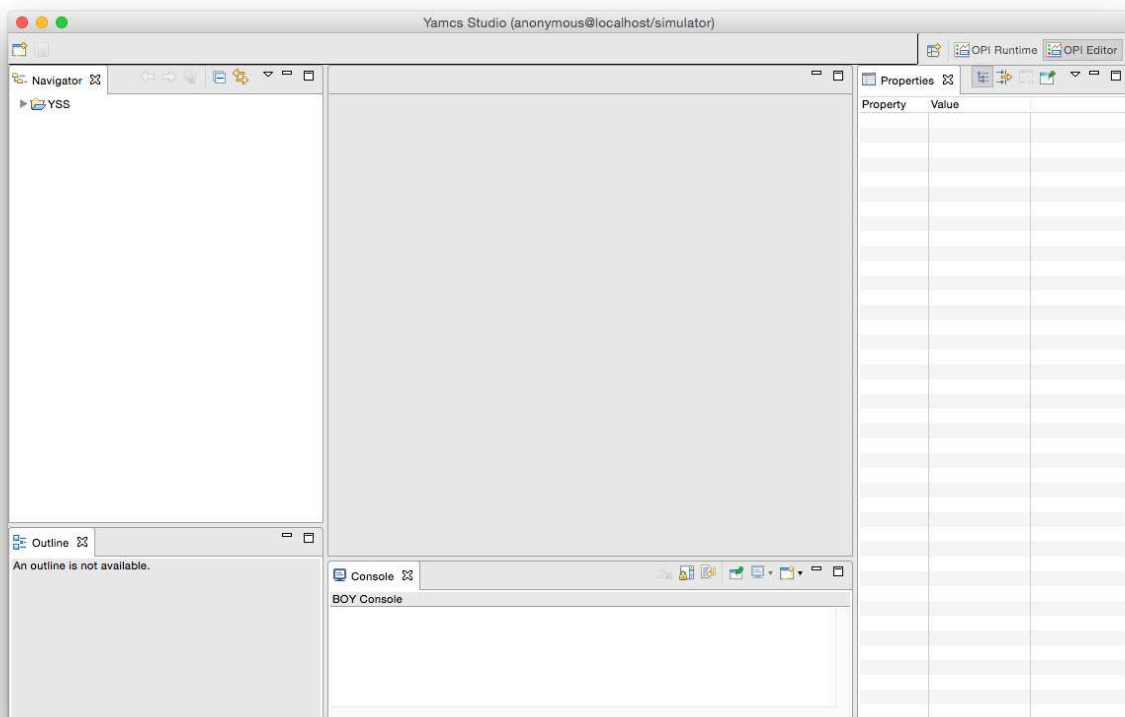
## Chapter 3. Editing Displays

## 3.1. OPI Editor Perspective

The OPI Editor perspective is used to create or edit displays. In the top right, change your copy of Yamcs Studio to OPI Editor mode (in case you don't see it, choose it from the dialog that opens up when clicking the plus-icon).



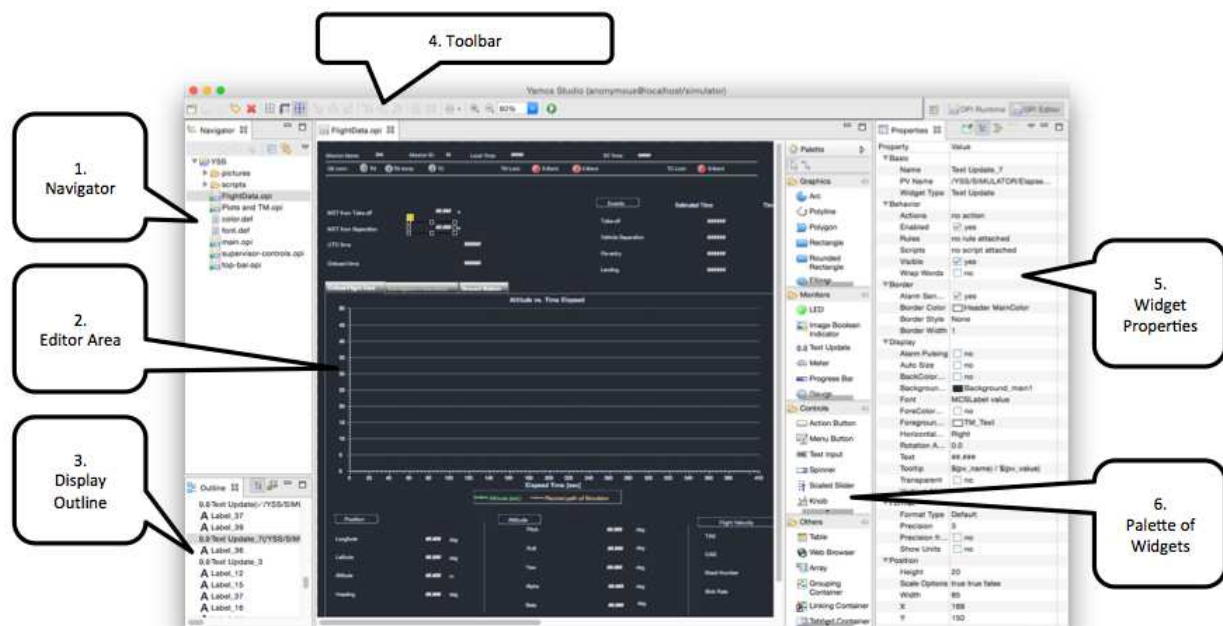
Your window arrangement changes to something like this.



In the left navigator, expand the YSS project and open for example our `FlightData.opi` by right-clicking and choosing Open With > OPI Editor.

Note: we are aware that right-clicking it is slightly annoying. The left-click action by default opens the OPI file with OPI Runtime. Once you've successfully opened an OPI with the OPI Editor the left-click action will from that point always open it with the OPI Editor, as it remembers its last handler. We definitely want to improve the user experience here. But for now, please bear with us as we do the needed development work.

The window layout can be decomposed like this:



### 1. Navigator

The navigator contains all projects within the current workspace. In general a project is at the same level as a mission, but this is not strictly necessary. When we launch Yamcs Studio with a new workspace, it will always automatically create the YSS project. Once you have added your own project, you can remove YSS and it won't be autocreated anymore.

A project contains Operator Displays ( `*.opi` ), images, color profiles ( `*.def` ), custom scripts ( `*.js` or `*.py` ), etc. Familiarise yourself with the right-click option as you go about opening displays. Displays can be opened in a few different modes within the OPI Editor.

- In editing mode
- In runtime mode in a Standalone window (*beta*)
- In runtime mode within the workbench itself (this will split your window to make room for it)
- In a new window using the green launch button in the toolbar

It is useful to have all these options when you're in the process of editing and testing displays with realtime telemetry, but do pay attention to treat the OPI Editor like an editor, not like a runtime viewer. During operations you should switch back to the OPI Runtime.

### 2. Editor Area

This area contains tabs for every OPI that was opened for editing. This offers familiar editing controls. Widgets can be selected, grouped, dragged and deleted to your personal taste.

### 3. Outline

The Outline view gives a hierarchical breakdown of all the widgets within the currently active editor tab. It is useful for finding back widgets. Widgets that were named will be easily identifiable.

#### 4. Toolbar


The toolbar offers context-sensitive controls. This includes general *Save* functionality, as well as handy features like grid toggling or space distribution among different widgets.

#### 5. Properties

This view shows the properties of widgets (or of the display itself). Notable properties include the PV Name which allows you to connect a widget with a specific Yamcs parameter (with autocompletion support). Other properties allow the display author to greatly tweak default widget behaviour. And in cases where the properties are not sufficient, we can always escape to more customization options using scripts and rules (there are properties for adding these as well).

#### 6. Palette

The palette contains the widgets that are available in your copy of Yamcs Studio. Select a widget from the Palette, and then click somewhere in the editor area to place it down.

When you're done doing changes, make sure to save them (File > Save All). You can now test out your changes by clicking the launch button  from the toolbar.

This will open a new runtime window (notice it uses the OPI Runtime perspective). If you leave this window open, and you save more changes, do a right-click in your display tab and choose Refresh OPI. You will do this a lot as you go about editing displays. You can also refresh by hitting F5, but make sure that your display actually has focus (for example by clicking somewhere in the editor before hitting F5).

## 3.2. Resource Management

Workspace

Projects

Searching

Projects

Within a workspace we can have one or more projects, which provides a way to group similar resources together. For many missions, having just one project is more than enough.

Opening a display

Customizing Yamcs Studio

### 3.3. Rules & Scripts

abc

# Chapter 4. Views



## 4.1. Archive

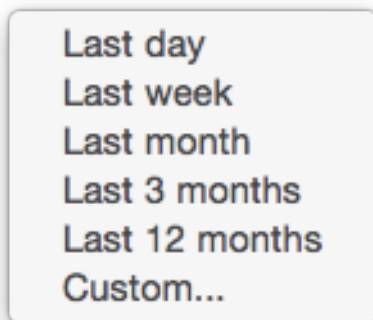
The Archive view represents a visual view of the data stored in the Yamcs archive. Through this view we can also initialize and control replays of archived telemetry.

### User Interface

We are aware that the current user interface takes some getting used to. The Archive view always works on a range of indexed data, which it fetches from the server. All further actions like zooming happen client-side on the loaded data range.

### Choosing a Data Range

As a first step you should select your data range. Click the pull-down icon ▼ to bring up this menu:



You can choose one of the predefined half-open time intervals, or you can select Custom... to specify your preferred range. Ranges can be half-open, which means they will always grow to include more bordering data as it becomes available.

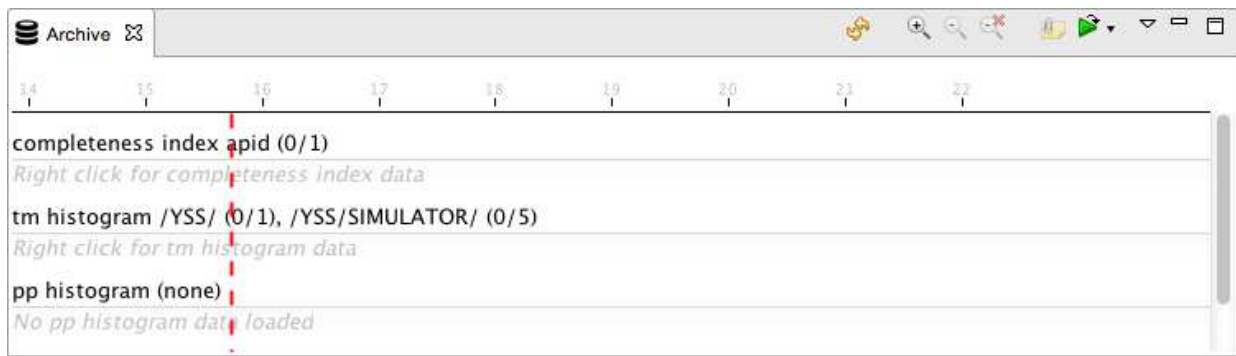
If you choose for example Last Day, Yamcs Studio will fetch an index of the archive for that time period, and refresh your view.

Your chosen data range is stored in your user preferences and will be restored the next time you open Yamcs Studio.

### Selecting Data

If this is the first time you have opened Yamcs Studio on your workstation, you will still not be able to see anything, other than some empty zones named:

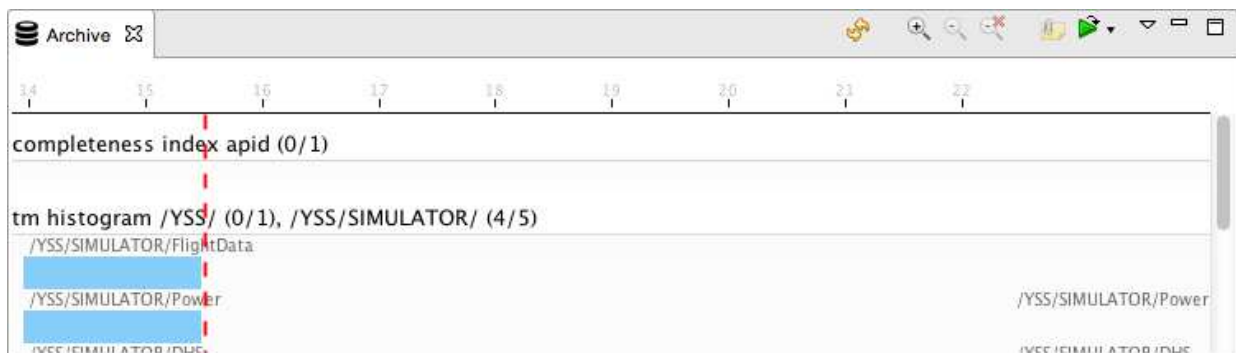
- Completeness Index APID
- TM Histogram
- PP Histogram
- CMDHIST Histogram



You need to choose which index data you actually want to display in your view. If there is data available for a zone, you can right-click it to bring up a pop-up menu where you select Add Packets > ... > All Packets. Your view will then update to show the selected packets.



We say *packets* since this is typically what we're interested in when browsing the Archive, but really it covers any kind of data that can be displayed through the Archive view.



Note that the view does not refresh itself, so hit 🔄 Refresh whenever you want to load the latest data for your selected time range.

## Navigating

The vertical red locator shows the current time as provided by Yamcs. When we hover the mouse over the view, a grayed-out locator indicates the current mouse position.

You can 🔍 Zoom In or 🔍 Zoom Out. If you are interested in a specific range, then select a time range by clicking and dragging your mouse before you click Zoom In.

Notice as you are zooming that a horizontal scroll bar appears. This allows you to scroll left and right within the initially load time range.

To clear your zoom stack, select 🔍✖

## Replaying data

## 4.2. Event Log

abc

## 4.3. Alarms

This view is under development, and will offer a centralised view of all active alarms, with the opportunity to acknowledge alarms or to mute any sounds. The Alarms view is not yet bundled in current copies of Yamcs Studio, but will be so in the short term.



Currently the only way to be noticed of alarms in Yamcs Studio is by following events (if your Yamcs Server is configured to report alarm state changes as events), or by manually iterating your displays to look for widgets that have red (= major alarm) or orange (= minor alarm) colored borders around them. Our upcoming Alarms view aims to improve this process.

---

## 4.4. Command Stack

The Command Stack allows operators to prepare stacks of commands for manual command issuing. The process is intentionally click-heavy to make sure that operators are aware of what they are doing.

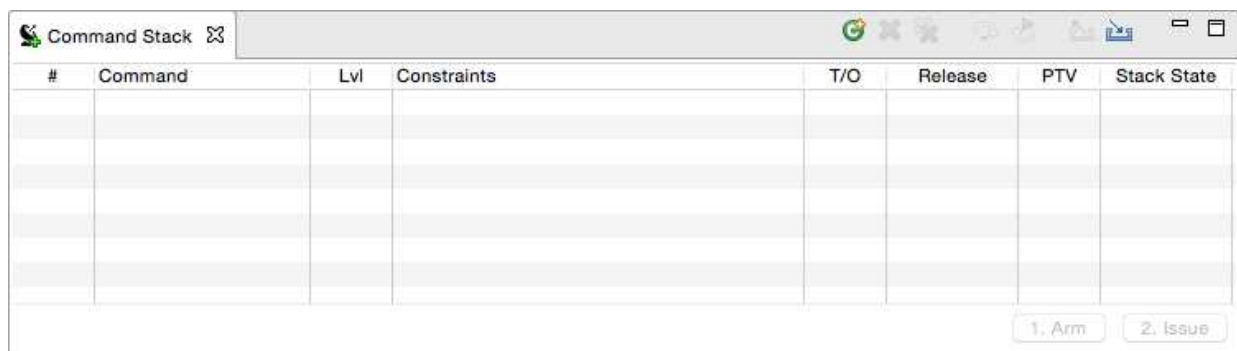
When you issue commands from Yamcs Studio, these are queued to Yamcs Server, which will perform any further steps.


We plan to bring many improvements to this view for better editing, but it is usable in its current state.

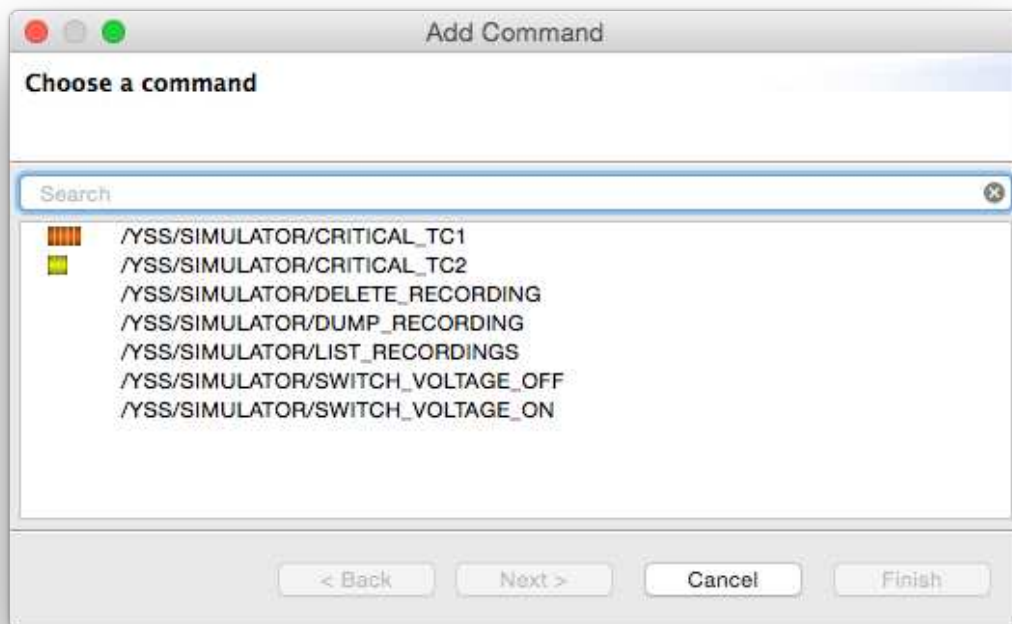
### Preparing a Stack

You can prepare a stack of commands only when you are connected to Yamcs. Yamcs Studio uses this connection to retrieve the list of available commands or to perform server-side validations.

When you start Yamcs Studio, the Command Stack view (available from the OPI Runtime perspective) is by default shown below the operator displays. If you can't find it back, select Window > Show View > Command Stack



Add a command by clicking the  Add Command button.



This opens a wizard dialog with the list of available commands. You can filter the list with the search box on top.

Commands are identified by their fully qualified XTCE name. This name matches the hierarchical structure of the commands as defined in the mission database of the connected Yamcs instance. In future versions we may include a tree representation in addition to the current flat representation.

Commands can have varying levels of criticality (called *significance* in XTCE parlance). The icon in the leftmost column indicates the defined criticality for the command.

Icon	Criticality
	Watch
	Warning
	Distress
	Critical
	Severe

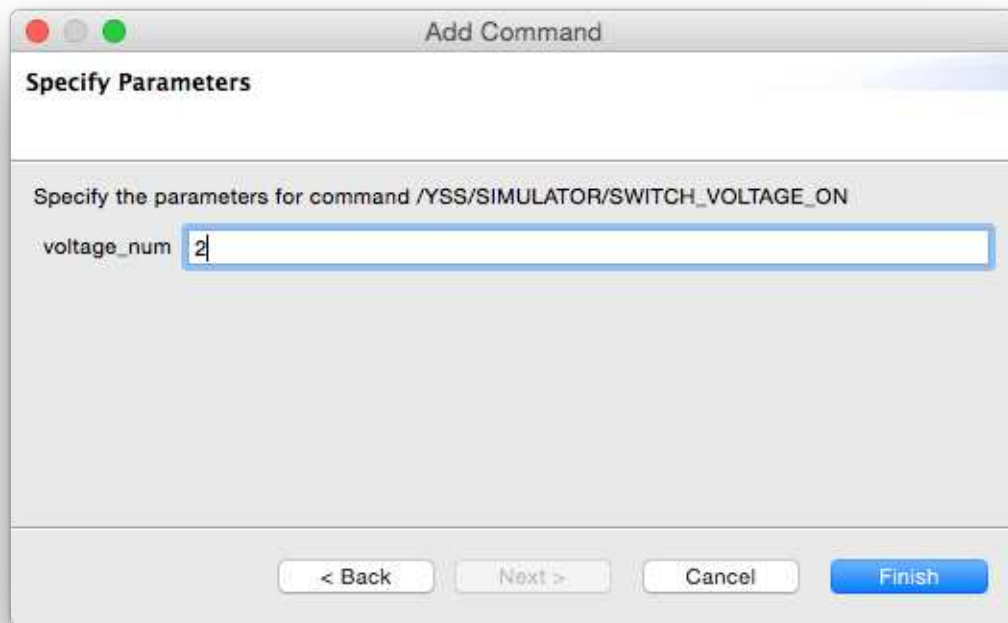
If an explanatory message regarding the criticality was defined in the mission database, this will show in the top title area of the dialog when the command is selected. Currently, only numbers or text can be entered.

Once you've chosen your command, hit Next to go the next page in the wizard, where you can specify any arguments that need to be provided for the command.



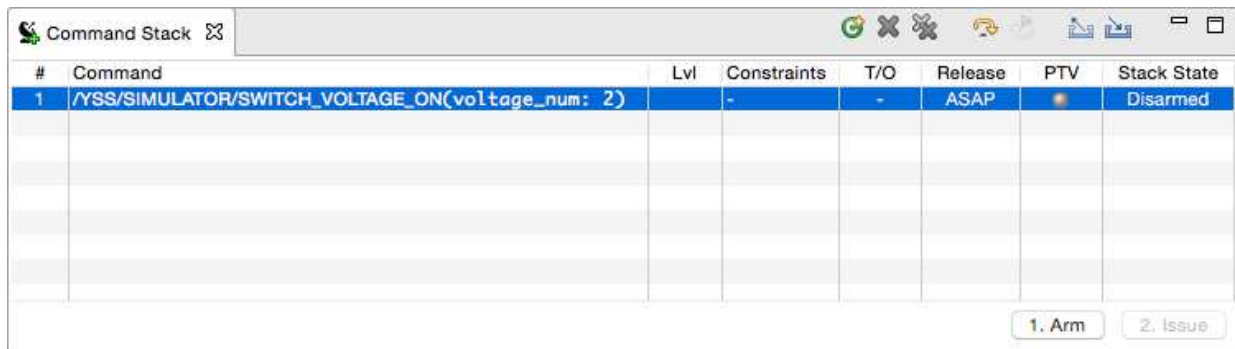
You can close the wizard from the first page as well by clicking Finish instead of Next. If the command requires any arguments, you will have a chance to add them afterwards as well by editing your stacked

command.



The 'Add Command' dialog box has a title bar with standard macOS window controls. Below the title bar is a section titled 'Specify Parameters'. Inside this section, there is a text prompt: 'Specify the parameters for command /YSS/SIMULATOR/SWITCH\_VOLTAGE\_ON'. Below the prompt is a text input field labeled 'voltage\_num' which contains the number '2'. At the bottom of the dialog are four buttons: '< Back', 'Next >', 'Cancel', and 'Finish'.

Click Finish to append your command to the end of your current stack.



The 'Command Stack' window displays a table with the following data:

#	Command	Lvl	Constraints	T/O	Release	PTV	Stack State
1	/YSS/SIMULATOR/SWITCH_VOLTAGE_ON(voltage_num: 2)		-	-	ASAP		Disarmed

At the bottom right of the window are two buttons: '1. Arm' and '2. Issue'.

You can review your provided arguments by double clicking the command. To remove the selected command from the stack select Delete. Clear the entire stack with .

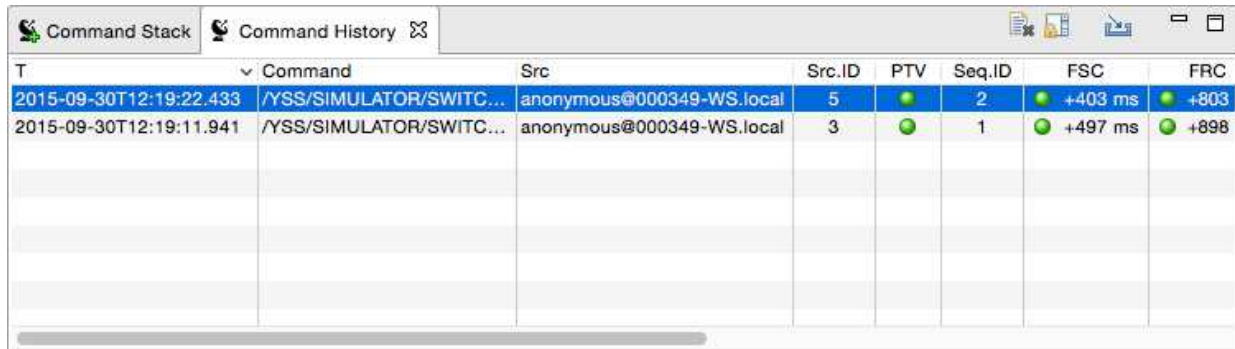
If a stacked command does not pass static validation checks (sometimes referred to as SPTVs – Static PreTransmission Verification) it will be marked with error indicators. This will prevent the user from attempting further execution of the stack until the error is resolved.







## 4.5. Command History


The Command History keeps track of all commands that were issued using Yamcs (not just by yourself, but by anyone).





T	Command	Src	Src.ID	PTV	Seq.ID	FSC	FRC
2015-09-30T12:19:22.433	/YSS/SIMULATOR/SWITC...	anonymous@000349-WS.local	5		2	+403 ms	+803
2015-09-30T12:19:11.941	/YSS/SIMULATOR/SWITC...	anonymous@000349-WS.local	3		1	+497 ms	+898

It will by default only show commands that were received on the realtime processor since you started your copy of Yamcs Studio. To load the command history for an earlier time range, select  Import.

Clear your view by clicking  Clear. You can always import the cleared commands again at a later moment.

When Yamcs Studio becomes aware of a new command that was issued by Yamcs, it will automatically select and reveal it. You can prevent this default behaviour by toggling the  Scroll Lock.

The displayed columns are as follows.

T	Time when the command was issued
Command	The command in textual format
Src	The origin of the command. Typically in <i>user@host</i> format
Src.ID	The ID that was given to the command by the issuing application. This number is assigned by the source application. In case of Yamcs Studio it is an incremental number that resets to 1 on every restart of Yamcs Studio.
PTV	Result of the Pretransmission Verification as performed by Yamcs. For example, some commands may only be applicable for 10 seconds and needs certain other parameters to be set to specific values. When the PTV bubble colors red, these type of context-dependent checks could not be validated, and therefore the command was not actually issued.
Seq.ID	The id that was determined by Yamcs before further dispatching the command. This is an incremental number that resets on every restart of Yamcs.
Further Columns	<p>Indicate acknowledgments of ground hops as the command is being dispatched. The exact number and name of the columns depends largely on how Yamcs is deployed at your site. Yamcs typically calculates the state of these bubbles based on incoming telemetry.</p> <p>The bubble becomes green  or red  depending on the verification</p>



result. The column value shows the time difference with the issuing time  $T$ .

## 4.6. Command Queues

This view allows controlling the Yamcs queues from the point of view of Yamcs Server. With sufficient privileges, queues can be blocked or disabled.

# Chapter 5. Troubleshooting

## 5.1. Capturing Log Output

In case you need to debug an issue with a deployed Yamcs Studio client, it can be useful to capture the logging output. Instructions are specific to the platform.

### Linux

Launch the `Yamcs Studio` executable from a terminal window while redirecting all output to a file named `log.txt`

```
./Yamcs\ Studio >log.txt 2>&1
```

### Mac OS X

With `Terminal` navigate into the Yamcs Studio application bundle and launch `./Yamcs Studio` directly from there while redirecting all output to a file named `log.txt`. For example:

```
cd Yamcs\ Studio.app/Contents/MacOS  
./Yamcs\ Studio >log.txt 2>&1
```

### Windows

With `Command Prompt` navigate into the location where you installed Yamcs Studio and launch `Yamcs Studio.exe` while redirecting all output to a file named `log.txt`. For example:

```
"Yamcs Studio.exe" >log.txt 2>&1
```