Developer Suite User's Manual

Accenture StormTest Development Center

Document ID: ST-11002

Revision Date: December 2016

Product Version: 3.4

Web: http://www.accenturestormtest.com

Contents

1 Preface

1.1 StormTest

Accenture StormTest Development Center is the leading automated test solution for digital TV services. It is designed to reduce the cost of getting high quality digital TV services to market faster.

StormTest Development Center greatly reduces the need for time-consuming, expensive and error-prone manual testing and replaces it with a more accurate and cost-effective alternative. It scales easily to large numbers and types of devices and integrates with existing infrastructure to give much greater efficiency in testing. It can be used to verify and validate services on a virtually every piece of consumer premises equipment (CPE), from set-top boxes to games consoles and from iPads to Smart TVs. It has been specifically designed to meet the needs of developers and testers of these CPE devices and the applications which run on them.

StormTest Development Center consists of:

- A choice of hardware units that can test 1, 4, or 16 devices. Each device under test can be controlled individually and independently and the audio/video from each device can be captured and analysed to determine the outcome of the test. The StormTest hardware supports capture of audio and video over HDMI interfaces and supports all HD resolutions up to 1080p. In addition there is a hardware upgrade option for the 16 device tester that will allow native capture of UHD content.
- Server software that controls all the hardware and devices in the rack as well as managing a central repository of test scripts and a central database of test results.
- A Client API that allows test scripts to interact with the server software
- A number of graphical tools that allow the user to directly control devices connected to StormTest Development Center, to create and schedule tests to run and to view the results of those test runs.

Test scripts can be run from any location – the tester needs only a network connection to the StormTest server. Video and audio output from the devices under test can be streamed over this network to any location, allowing remote monitoring and control of testing, either within a company LAN or across a WAN. Alternatively, scheduled tests can run directly on the server, negating the need for maintaining a continuous network connection to the StormTest server.

1.2 About This Document

This document describes the user of StormTest Developer Suite.

1.3 Related Documentation

The StormTest user documentation set comprises of the following documents:

1) StormTest Developer Suite User's Manual
2) StormTest Programmer's Guide

3) StormTest Client API
4) StormTest Hardware Installation Guides (HV01, HV04, HV16)
5) StormTest Software Installation Guide
6) StormTest Server Monitor User's Manual
7) StormTest Administration Console User's Guide
8) StormTest Administration Tools User's Guide

The latest version of these documents can always be found on our support website, in the "Docs" section: https://larisa.engage.s3group.com/docman/?group_id=6.

1.4 Revision History

Date	Version	Description
March 2012	2.7	Upgrade for OCR Filter Calculator, Daemon Information and new logos.
July 2012	2.8	Updated for public schedules and email notifications.
January 2013	2.9	Updated for HS64 server
April 2013	3.0	Update for HD, Navigator and change STB to DUT
September 2013	3.1	Addition of Resource Editor and revised GUI layout and reservation procedure.
August 2014	3.2	Addition of Screen Editor, Model Specific Resources, and change of Trainer Behavior.
October 2014	3.2.1	Update for Color Trainer/Resource Editor for showing flatness as floating point to 2 decimal places, corrections to AV Preview trainer, added description of Test Creator Preferences Ribbon. Update Image Trainer about icon mode.
November 2014	3.2.2	Addition of Rolling Record Buffer Preferences, Recently Used Config Server and Live Log View of running tests.
January 2015	3.2.3	Removal of DUT Allocations view
May 2015	3.2.5	Update for Find Screen in Navigator, Save Serial text to file, filter schedules, Remote Control Magnifier
November 2015	3.3.0	Update for log viewer pages
February 2016	3.3.2	Update for re-run of Unscheduled Tests
August 2016	3.3.6	Update for revised color scheme
October 2017	3.3.7	Change to preferences for low bandwidth mode

Table 1 - Revision History

2 StormTest Developer Suite

2.1 StormTest Developer Suite

The StormTest Developer Suite is the user interface for StormTest Development Center. For full use of the StormTest Developer Suite it is essential that all server components of StormTest Development Center are also updated to latest version.

This version introduced some changes in behavior. Users of previous versions of StormTest Developer Suite may wish to read about the changes.

2.1.1 Developer Suite Structure

The Developer Suite is based around the concept of applets. Each applet has distinct functionality and one or more views. On first use of an applet, a help page is displayed. All versions of the Developer Suite come with the preferences and help applets. This version comes with the Test Manager, Remote Control and Resource Editor applets. It adds the ability to define screen navigators and screen definition objects, implemented as document editors. The Test Creator introduced in version 2.0 is also a document editor and has been moved to the ribbon so that all documents are accessible from one location.

Navigation between applets is accomplished using the left hand panel. Once an applet is selected, navigation between views is also in the left hand panel while the views appear in the right hand panel.

The preferences and help applets are shared between all applets so that the preference or help for an applets may be accessed either via the applet itself or via the common preferences and help applets.

2.2 Changes in 3.1

This summarizes some important changes in version 3.1 compared to all earlier versions of StormTest Developer Suite. If you are upgrading from an earlier version of StormTest Developer suite, this information may help you.

2.2.1 Reserving DUTs

On earlier versions, the DUT was reserved 'per screen'. This meant that if you opened Test Creator, you had to reserve a DUT to run or debug tests and closing Test Creator meant the DUT was unreserved.

Version 3.1 changes the reservation to be 'per Developer Suite'. You now reserve a DUT from the main ribbon. Once you have done this, you can use it anywhere - Navigator, Test Creator, Remote Control, Resource Editor. You can release it from the main ribbon whichever applet is active. The ribbon controls now allow you to select a DUT from your reserved DUTs to be the one to use. If you only have one DUT reserved, then it is assumed that you want to use it.

This has a forced consequence for the trainers - it is no longer possible to reserve a DUT in the trainer and so the trainer cannot be invoked until a DUT is reserved.

2.2.2 Resource Editor

In version 3.0, the resources were only used in the Navigator and all editing and creation was done via the Navigator screen. In version 3.1, the resources can now be used everywhere. There is a dedicated resource editor applet which can fully edit the resource. Other applets are designed around using resources or creating them from existing literal values. The ability to manually set coordinates on a region is reserved to the Resource Editor.

2.3 Changes in 3.2

This summarizes some important changes in version 3.2 compared to all earlier versions of StormTest Developer Suite. If you are upgrading from version 3.1 of StormTest Developer suite, this information may help you. If you are upgrading from a version prior to 3.1, you may wish to review the changes in version 3.1.

2.3.1 Model Specific Resources

Each resource can have multiple values depending on the DUT model. All resources have a generic variant. Any resources created in earlier versions of StormTest Development Center are (and always were) generic. In version 3.2, it is possible to define a resource variant targeted at a specific DUT model. When StormTest (either a Python script or StormTest Developer suite) needs to find a value of a resource, it first looks to see if a model specific resource for the DUT exists. If it does not exist, then the generic version will be used. This allows you to write scripts using resources for a range of DUTs and so long as the operation of the different DUTs is the same, the style of the UI can be accounted for by DUT specific resources. This allows you to separate the style and the functionality of the DUT.

2.3.2 Trainers

The trainers have been redesigned. They are no longer modal dialogs which block the user interface. Instead they are a sub tab of the Test Creator, Navigator and Screen Editor. This means that you can have more than one trainer window open at the same time. It also means a simpler user interface: instead of an extra video window, remote control and resource panel, you use the underlying Test Creator, Navigator or Screen Editor controls. As a consequence some of the features of the old trainer dialog have been removed as they are no longer necessary. It used to be possible to edit a resource inside a trainer - this is no longer necessary because you can simply switch to the resource editor to edit a resource. In version 3.1, to switch to the resource editor from a trainer required closing the trainer and then re open the trainer to get back to it.

The user controls of the trainer have been moved to the ribbon with a special trainer tab that is active only when a trainer is active. The allows more space for the trainer window.

2.3.2.1 Screen Definition Objects

A new concept of a screen definition object has been introduced - the menu item calls it a 'screen' to save space. These objects improve the performance of tests as well as reducing the time to create and maintain tests.

2.4 Changes in Storm Test Developer Suite 3.3

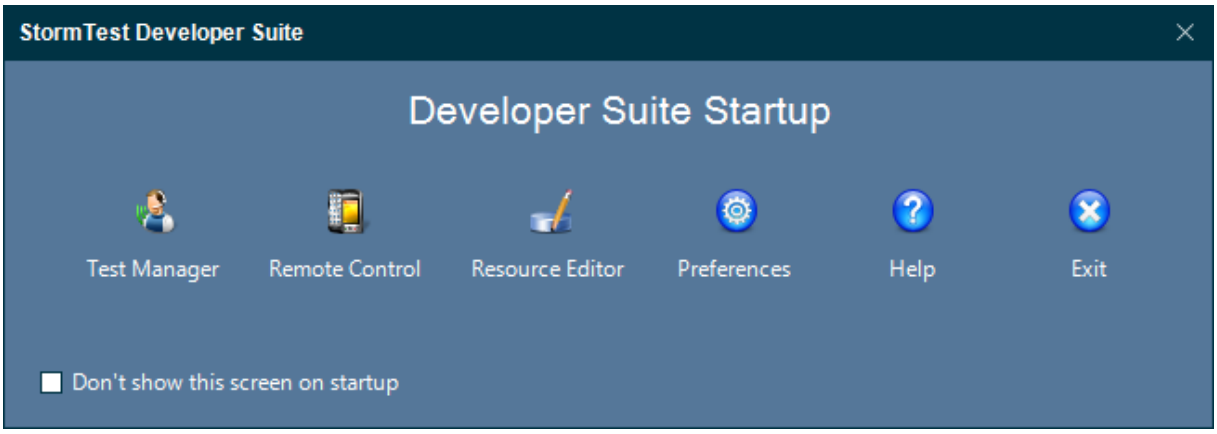
This summarizes some important changes in version 3.2 compared to all earlier versions of StormTest Developer Suite. If you are upgrading from version 3.2 of StormTest Developer suite, this information may help you. If you are upgrading from a version prior to 3.2, you may wish to review the changes in version 3.2.

2.4.1 Support for 4K servers

Developer suite supports the new 4K (Ultra High Definition) server. The architecture of the 4K server is similar to HD. Throughout Developer Suite, wherever HD is mentioned then this applies also to 4K unless there is a specific note that 4K is in some way different to HD.

2.5 Mini Toolbar

After the splash screen has disappeared, a mini toolbar appears:



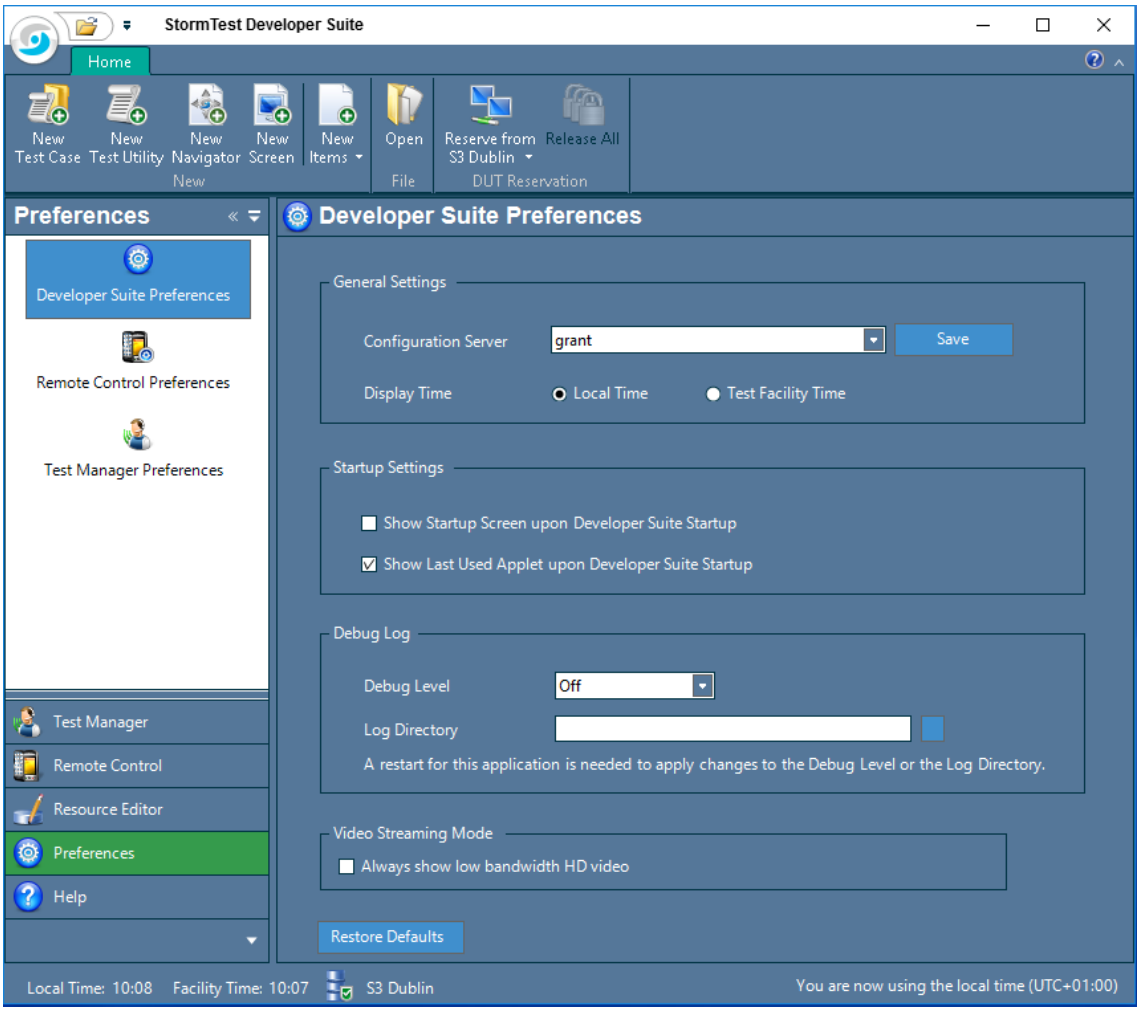
This will appear on the first use of Developer Suite. From here you can select one of the applets or exit Developer Suite. This mini toolbar can be disabled by checking the checkbox at the bottom of the screen or from the preferences applet.

The options available are:

- Test Manager Applet
- Remote Control Applet
- Resource Editor
- Preferences Applet
- Help Applet
- Exit Developer Suite

2.6 Developer Suite Panels

The application is divided into two large panels, left and right. The divider does not resize the panels, although the left hand panel may be made smaller by other means. There are other smaller features of the Developer Suite workspace.



2.6.1 Navigation Panel

The navigation panel is where you select which view of an applet you wish to work with. This can be customized to suit your work flow.

2.6.2 Applet Workspace

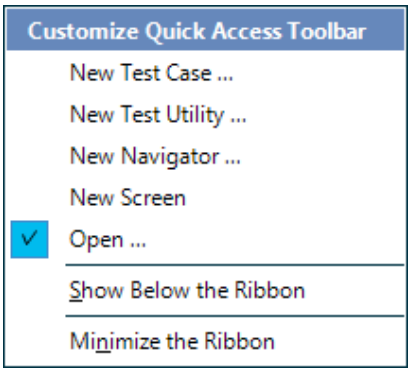
Each applet uses the right hand area for its work area. It is applet specific and documented in its own help file.

2.6.3 Background Activity Indicator

At the top of the application screen, is a small StormTest logo which spins when Developer Suite is working in the background.

2.6.4 Quick Access Toobar

The quick access toolbar was introduced in Version 3.0 of Developer Suite. It allows you to customise frequent actions. Click the down arrow to bring up a menu and select the items to appear on the toolbar:



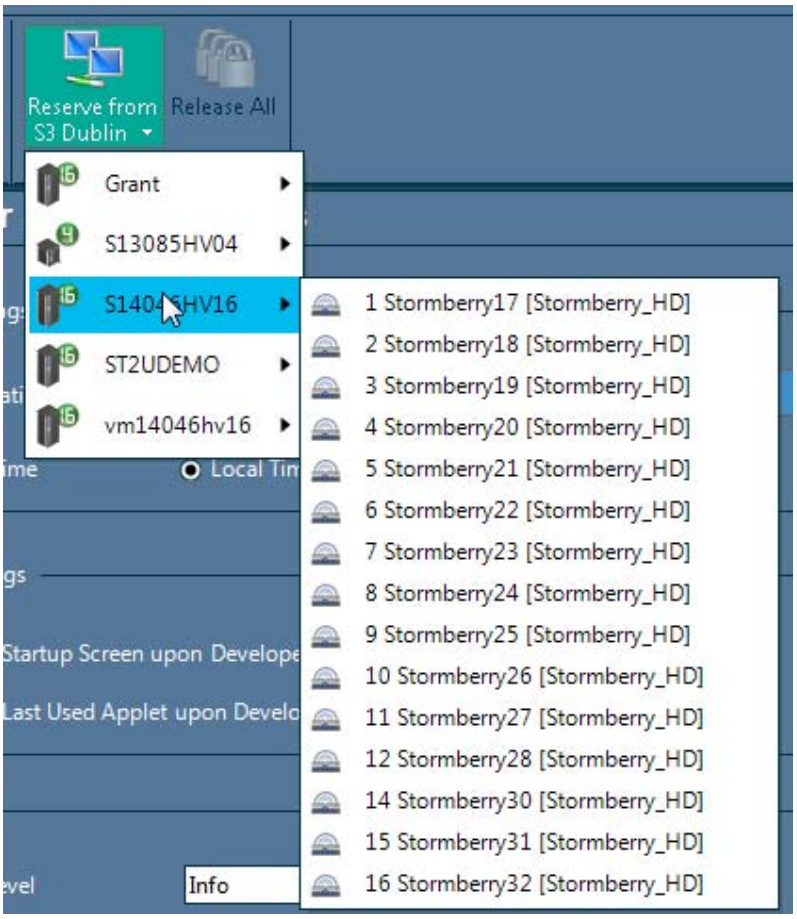
2.6.5 Ribbon

The ribbon allows access to common tasks across multiple applets.

The group of buttons allows an easy method to create a new test case, new test utility, new navigator or a new Screen Definition. The new Items button also allows a new log viewer and access to the Job Wizard to create a new job.

You can open a Test Creator, Navigator or Screen Definition file using the Open button.

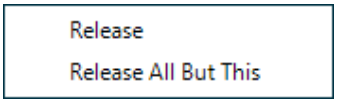
You can reserve a DUT for use by any applet by clicking the 'Reserve from' button. A drop down menu shows you all the servers and DUTs in the facility:



It will show you the DUTs reserved. Check a box to reserve a DUT. You can also free a DUT from this control.

Once reserved, the ribbon is updated with the DUT (or DUTs) reserved to StormTest Developer Suite:

You can release all DUTs (only those you actually reserved, not everyone's DUTs) by using the Release All button. Clicking an individual DUT brings up a menu to allow release of a DUT or all DUTs except this DUT:



This feature is available at any time and works across all applets. For convenience, the reservation buttons are repeated across the Test Creator, Navigator and Screen Definition document editors.

2.6.6 Local Time Display

This is the time on your machine.

2.6.7 Facility Time Display

The facility time is displayed next to the local time at the bottom of the application screen - this is the time as indicated by the Configuration Server. If there is a small difference between it and local time when you don't expect it, this could be because either your PC or the machine on which the Configuration Server is running has the system clock inaccurately set.

2.6.8 Time Warning

If the clock on your client machine is not synchronized to the clock in the main facility then this warning will appear. It is a warning not an error. The lack of synchronization can cause confusion: you might create a schedule to start at 11:00 but when 11:00 comes on your machine, the schedule has not started and might not start until 11:01. This warning will indicate the potential for such problems. You should make sure that the clocks of all clients and servers using StormTest are synchronized to the same source (for example by using the Windows time service or a time server managed by your IT department).

If you change the clock on the server or your client machine, you need to restart StormTest Developer Suite to remove the warning.

2.6.9 Current Facility

The name of the current facility (which is configured in the Admin Console and need not be the same as the name of the Config Server) is shown along with an icon indicating the current status of the connection. The possible icons are:

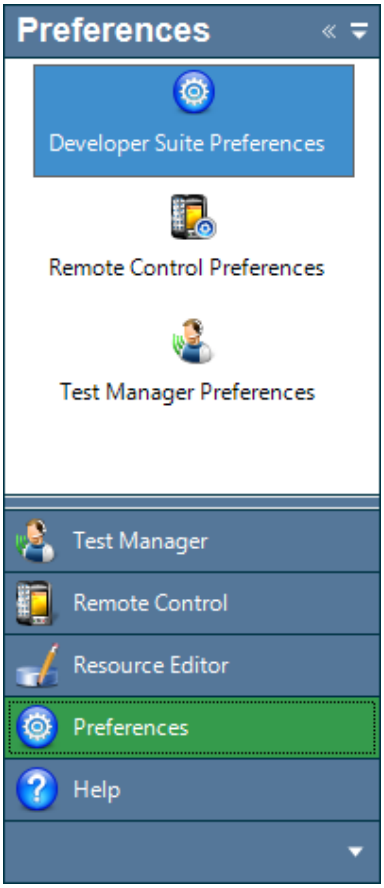
 Developer Suite is connected to the Config Server.

 Developer Suite has lost connection to the Config Server. It will try to reconnect.

Except when maximized, Developer Suite can be resized using the mouse - click an edge or corner and drag it. The caption bar is the standard Windows caption bar with minimize, maximize and close buttons.

2.7 Developer Suite Navigation

The left hand panel of Developer Suite is the core navigation panel. It has 2 modes of operation: normal and minimal. The normal mode is shown below:



At the top is the currently active applet title (in this case, preferences) along with 2 buttons:

will switch to minimal mode, at which stage it changes to to switch back to normal mode.

brings up a menu to change applets. Applets can also be selected by clicking on them in the stacked area beneath the separator bar.

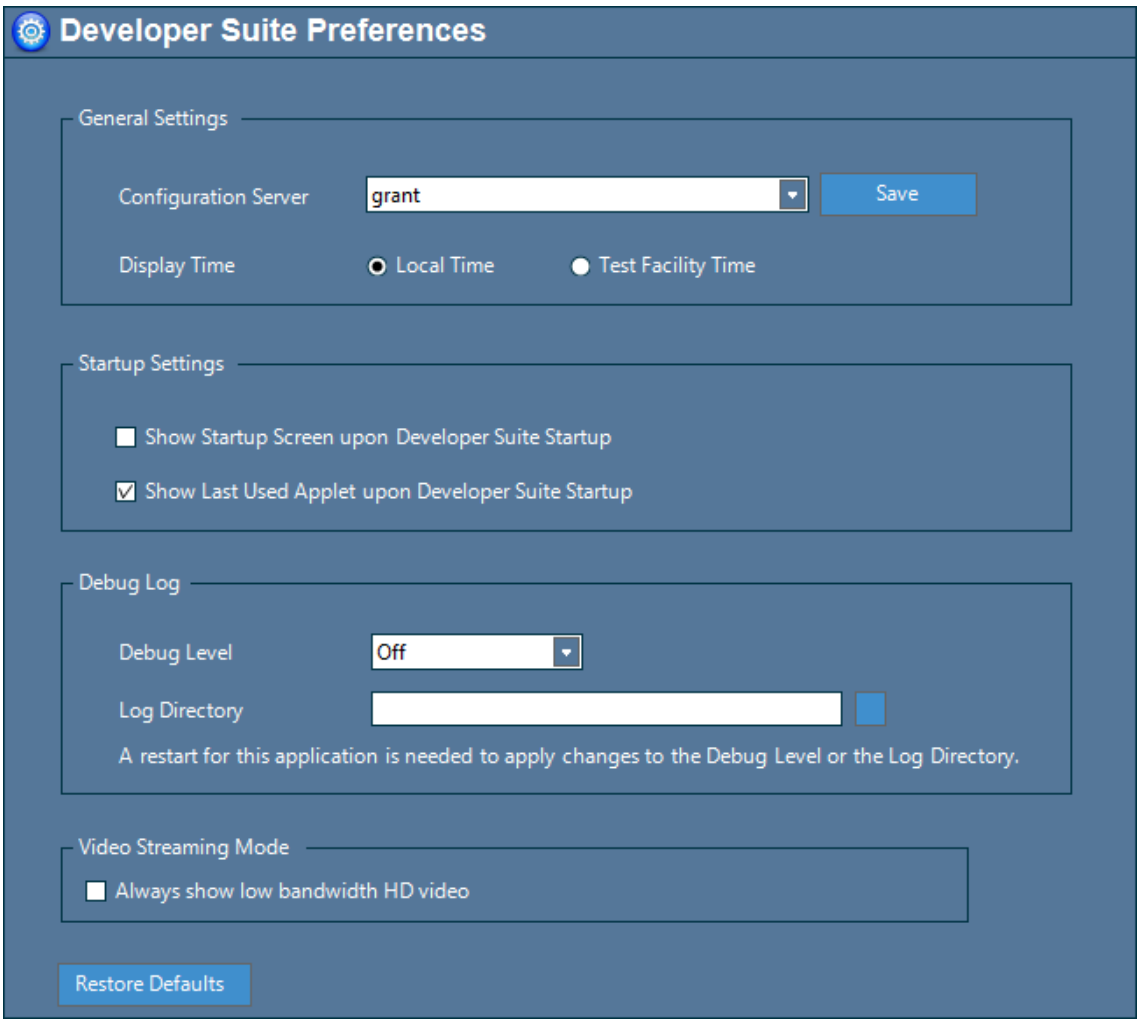
The size of the stacked area can be altered by dragging the separator bar, and the available applets can be changed by using the at the bottom right corner.

Each applet has one or more views, in the above case, the preferences applet is showing 1 view. Switching between views is by clicking them.

It is **not** possible to select views in minimal mode. Minimal mode allows the maximum possible area of the screen to be used for a view - at the expense of not being able to select views.

2.8 Preferences Applet

The preferences applet is where you can control how the application behaves.



2.8.1 General Settings

2.8.1.1 Configuration Server

The Configuration Server determines which logical facility the Developer Suite will use. It can be a host name, IP address or a full URL of the format http://servername:server port. If you use http and the default server port of 8001, the user interface shortens this to just the name. The server name is set during the installation process. If you have multiple facilities, Developer Suite remembers the list you have used. You can select from any prior used configuration server by using the drop down arrow. If you change the configuration server, Developer Suite connects to the new server and refreshes internal data. It is essential that a valid server is running for correct operation of the application.

Click the 'Save' button to save the change. A dialog will appear confirming the connection.

NOTE: After changing the configuration server you must restart Developer Suite - you will be prompted to restart after a successful change of the configuration server.

If you need to remove a configuration server from the list to tidy up the drop down list, use the right mouse button to bring up the context menu. You can remove all previous configuration servers or a specific one that you no longer use. You cannot remove the current configuration server.

2.8.1.2 Display Time

Some applets deal with time - you can choose whether the time show in your current local time or the time of the facility to which you are connected. The default at installation is local time. You may wish to work in the facility time if that is different from local time as DUT's will work in the facility time as does the StormTest scheduler. This is intended to allow you to work across time zones. It is not intended to correct any errors between the time of your client machine and the facility - it is expected that all machines using StormTest will have an accurate clock indicating the correct time.

2.8.2 Startup Settings

These control how Developer Suite starts.

2.8.2.1 Show Startup Screen

By default, Developer Suite starts with the mini toolbar screen. From this you can select an applet.

2.8.2.2 Show Last Used applet

By default, Developer Suite will go to the most recently used applet. If you disable this then Developer Suite will always go to the main help. **This has no effect unless the show startup screen is disabled.**

2.8.3 Debug Log

These settings control the generation of a debug log - mainly of value to support personnel. You must restart Developer Suite for the changes to be used.

2.8.3.1 Debug Level

This controls the amount of information in the debug log. There five levels: Off, Error, Warning, Info and Verbose. Default is Off.

2.8.3.2 Log Directory

Choose a directory for log files. Either type in a value or use the button to browse for a folder. Files are stored as HTML in a file called logFile.html within a directory called StormTestApp_date. If this field is left blank, then the log files are created in a folder S3Group\StormTest\Logs under the "My Documents" folder (or "Documents" under Windows 7).

2.8.3.3 Video Streaming Mode

This controls whether HD video is streamed in normal high resolution mode or a special low bandwidth mode. In low bandwidth mode, medium resolution JPEG snap shots are taken of the video and displayed at approximately 1 per second. In this mode, you can view all 16 slots of an HV16HD server on any machine. If your network connection is slow, this mode will show images at a lower rate than 1 per second but will still operate unlike the normal mode which will stall with low bandwidth network connections.

2.8.4 Restore Defaults

If you click this button, all settings are restored to the default settings. The Developer Suite will exit and restart after you have confirmed the action. The restore can not be undone and will restore all settings to the state immediately after installation.

2.8.5 Saving the settings

The settings are saved automatically without your intervention.

2.9 Help Applet

The help applet is common to all applets and is the place to find all the online documentation. Each applet may define one or more help books. These appear within a separate tab. You can have as many tabs open as you like - even opening the same book in multiple tabs.


on the right hand side brings up a menu to select a new book to open.


returns you to the start page of the book


and 
allow you to navigate backward and forward through the book. These may be disabled if it is not possible to move backward or forward.


selects between the active tabs - as does simply clicking on the tab. However, if you have many open tabs the 
button may be more convenient.

closes the currently active tab.

About brings up the about box with version information about Developer Suite

NOTE: When you enter the help system from within an applet, only the books supplied by that applet are available from the menu. However, when you use the main help applet, all books from all applets are available from the menu. This is by design.

2.10 Common Tasks

- Set the time zone
  Use the preferences applet and select either local time or facility time under General Settings.
- Access online help
  Use the Help Applet.

- Make an applet active
  Click on the applet in the lower stack of the left hand panel or use the button and select if from the menu.

  If the applet is not visible, try using the in the bottom right corner and selecting the 'Add or Remove buttons' item to add the applet back to the application.

- Change views in an applet
  Click on the view (on the white background). If the view is not visible, use the scroll bars to bring it back into view. You may also use the tab button and select with a space when the desired view has a dotted rectangle around it.

- Exit the Developer Suite
  Click on the close button at the top right hand corner of the application or use the Alt-F4 keystroke combination.

- Restore the default settings
  Go to the preferences applet and click 'Restore Settings'. The defaults will be restored.

- Force the mini toolbar to appear on startup
  Use the preferences applet and select 'Show startup screen on application startup'

- Make the last used applet start on startup
  Use the preferences applet, uncheck 'Show startup screen upon application startup' and check 'Show Last Used applet upon application startup'

- See as much of the applet view as possible
  Click the button - note that you cannot then change views. See also the application navigation topic.

- See as many view items as possible
  Drag the separator bar above the applets as low as it will go. You can also use the button, and from the menu select 'Show fewer buttons' as many times as you like to increase the space for the view icons.

- Restore the view navigation
  Click on the button. See also the application navigation topic.

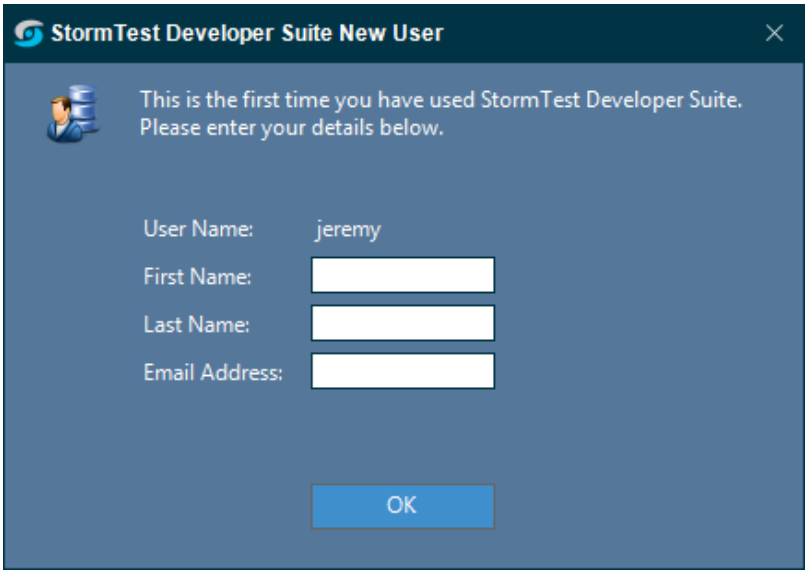
- Restore the applet stack to the default
  Drag the separator bar above the applet stack as far up as it will go. Also click on , select 'Add or Remove buttons' and select all the buttons.

- Start viewing a new help book in a new tab
  Click the next to the text 'New Tab:' and select a book to view.

- Enable debug log
  Use the preferences applet and select a folder for the log files and the log level - it must be something other than 'None' for a log file to be produced.

2.11 New User Dialog

After Developer Suite starts it checks that your Windows user name exists in the StormTest Development Center database. If it does not then you are prompted for some extra information:

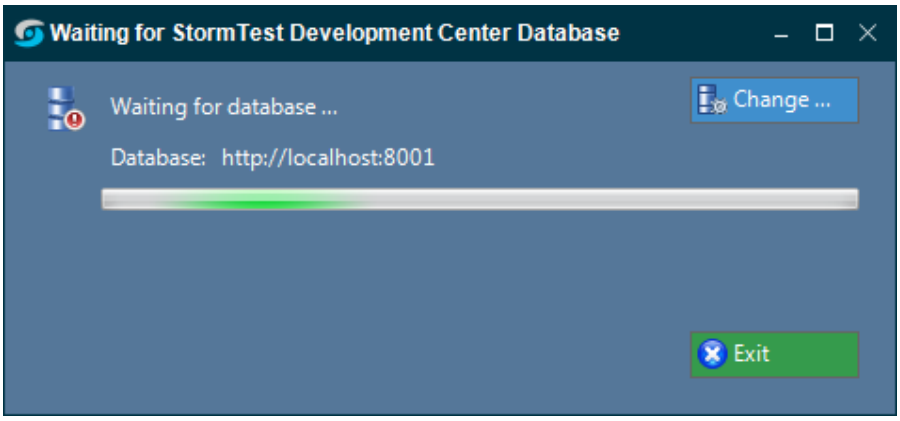


The user name cannot be changed - it was set when you logged on to Windows.

The other fields are optional. However, they will be used in later versions for display and possibly the email address will be used for notifications. Please enter your first and last names along with a valid email address. No checking is done by Developer Suite as to whether the fields are valid.

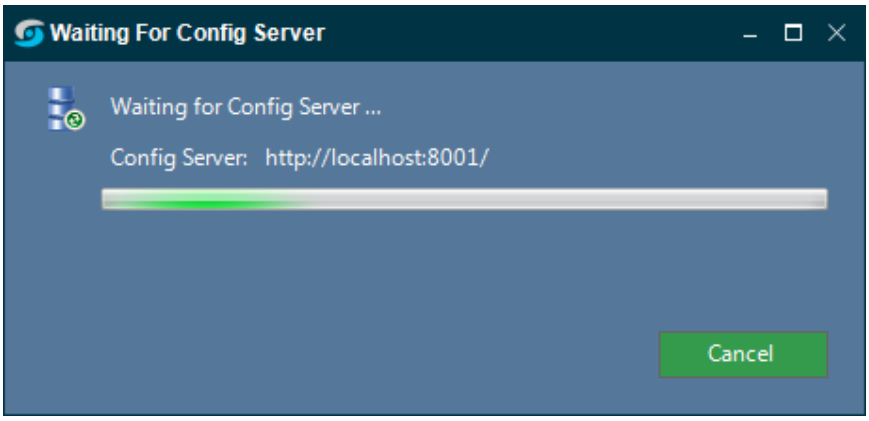
2.12 Waiting for StormTest Database

After Developer Suite initialization, Developer Suite checks for the Configuration Server. If it does not respond then Developer Suite waits for the Configuration Server to respond. If Developer Suite has been configured with an incorrect value for the Configuration Server then you can change it by clicking the Change button to bring up a dialog to change the Configuration Server.



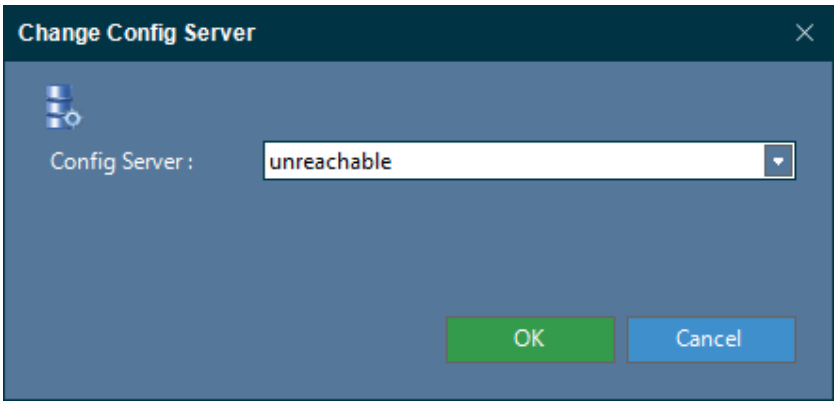
2.13 Waiting for Configuration Server

After changing the configuration server, Developer Suite attempts to contact the newly set Configuration Server. If you have made a mistake, you can cancel the dialog and set a new value for the Configuration Server.



2.14 Change Configuration Server

If the initial Configuration Server cannot be contacted then you can change the server with this dialog:



If the initial Configuration Server is contacted then you can switch to a new one from the preferences applet.

2.15 About Box

The about box shows information about the version of Developer Suite and is essential information when contacting support.

The version is the product version and the Build string identifies exactly the code version used to build Developer Suite.

It is accessed from the Help Applet by clicking on About button.

Your version and build number will vary from the illustration.

3 Test Manager

3.1 Core Concepts

The key concepts to understand in order to effectively use the Test Manager are:

1. StormTest Development Center Facility
2. File Repository
3. StormTest Development Center Scheduler
4. Client Daemon
5. User Identification

3.1.1 StormTest Development Center Facility

A facility is a logical concept intended to represent a single location with one or more StormTest Development Center servers. Each server has up to 16 DUTs. The facility exists as a data definition in a PostgreSQL database with each server and every DUT defined in the database. If the definitions are not correct, then the Test Manager and scheduler will not produce the expected results. You do not communicate directly with the database, instead you use a component called the Configuration Server (often shortened to Config Server). All the configuration details of the servers and DUTs are in the database. Actual configuration is done using a web based tool - the admin console which is documented elsewhere.

3.1.2 File Repository

Your test files (both the Python scripts and any associated image files) must be stored in the StormTest Development Center file repository in order for them to be executed by the StormTest Development Center scheduler. You can, of course, still execute the python scripts locally on your PC as was the case with earlier versions of StormTest Development Center. The Test Manager provides the Test Scripts view to help you manage the files in the file repository. The file repository is accessible to all the Client Daemons to run tests. The physical location of the repository is part of the StormTest Development Center Facility definition and is set during initial installation.

3.1.3 StormTest Development Center Scheduler

The scheduler is a software module which schedules test scripts. There is one and only one scheduler per facility. It relies upon the facility being correctly configured with the correct DUTs listed as being in the correct slots. It is important also that all servers in the facility are at version 2.0 or later. It is important to note that the scheduler will not schedule tests to empty slots - if the database indicates that there is no DUT in a specific slot, then no tests will run on it. Thus it is very important that users use the admin console to keep the information about DUT location up to date.

The scheduler permits a user to define schedules in time where tests can be run. One or more test scripts can be allocated to a schedule. Resources such as slots, DUTs or types of DUT are allocated to schedules to allow tests to run.

A single test script can be assigned to more than one schedule. A single DUT can also be assigned to more than one schedule. The core concept is that of a schedule, to which DUTs and tests can be attached.

3.1.4 Client Daemon

The scheduler mentioned above does not actually run the tests - it schedules them. The physical execution is performed by another module, the Client Daemon. This can run on any machine (subject to licensing restrictions). It is a good idea for a Client Daemon to be on a machine with a powerful graphics card because the test scripts often do video analysis and the StormTest Development Center libraries can take advantage of hardware acceleration in modern powerful graphics cards - put another way, if the Client Daemon is on a machine without a powerful graphics card, more CPU cycles are used doing video analysis.

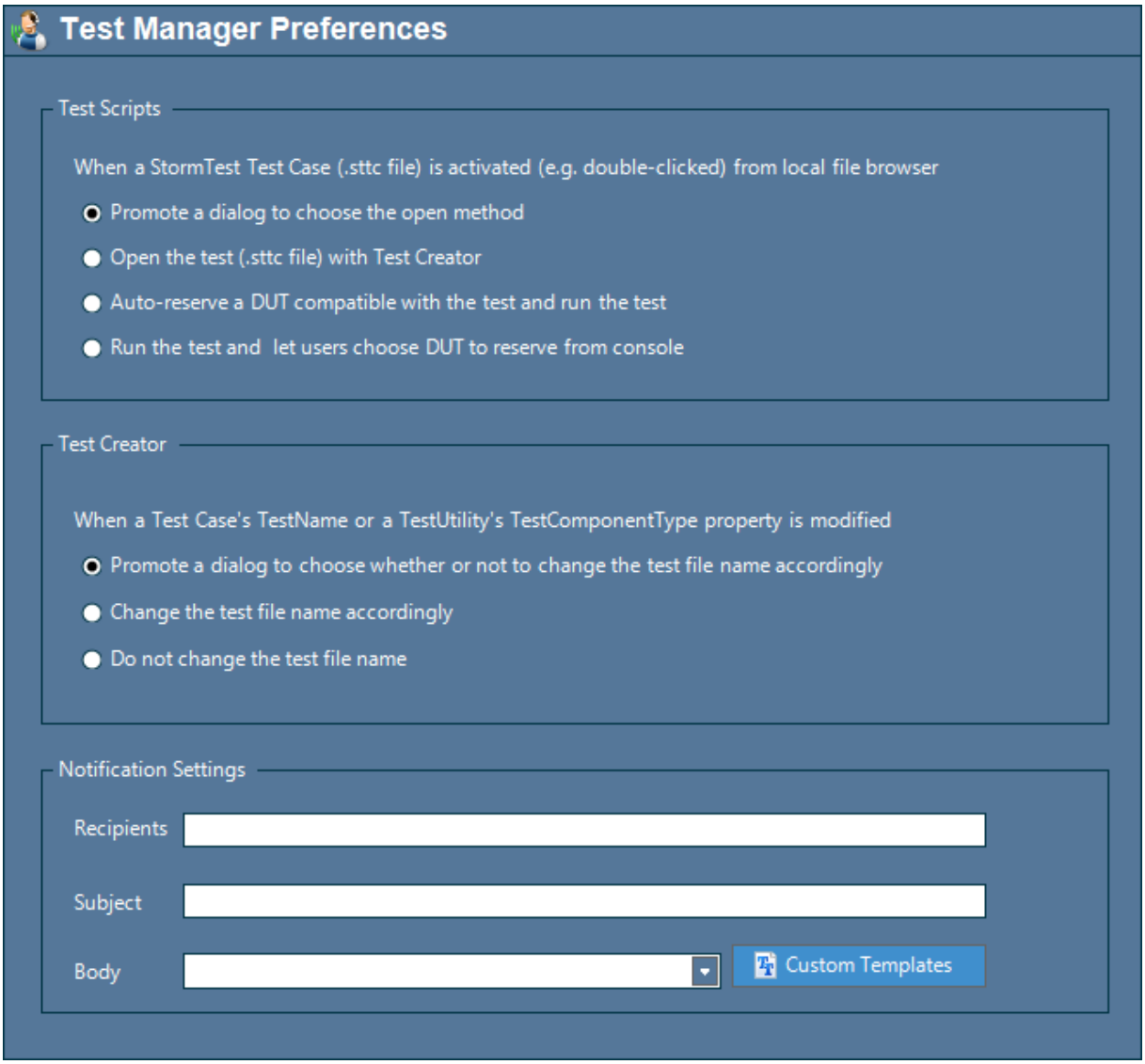
The scheduler can manage multiple Client Daemons and will balance the test load across the available Client Daemons. Your test log files are stored on the same machine that ran the test - and accessible via FTP directly from the Test Manager Test Results view.

3.1.5 User Identification

The StormTest Development Center client has always used your Windows logon name to identify itself to the server. The Test Manager is no different and filters all views apart from the Test Results view based on your Windows logon name. The current version imposes no access control based on user name - there are no Administrator users - all users are equal.

3.2 Test Manager Preferences

The test manager preferences control how the test manager handles tests created by test creator:



3.2.1 Test Scripts Options

When a Test case (.sttc file) is double clicked or opened, you can choose what action should happen. The mutually exclusive options are

1. Show a dialog so that you can choose each time the desired action
2. Open the test case for editing
3. Run the test case and automatically select a DUT to use
4. Run the test case and choose the DUT to use via the command line

3.2.2 Test Creator Options

When you modify the name of a test case or test utility, you can choose the action. The mutually exclusive options are

1. Show a dialog so that you can choose each time what to do
2. Change the file name to match the test case name without prompting

3. Leave the file name unchanged without prompting

3.2.3 Notification Settings

When a schedule ends, StormTest can send a notification via email. You can set up default values for that email to make the setup easier and quicker. These are:

3.2.3.1 Recipients

The list of email addresses to use by default for the email notification. Separate each address with a semicolon (;)

3.2.3.2 Subject

The subject line of the email. You may use keywords to substitute values for various parts of the schedule - see the email templates topic.

3.2.3.3 Body

The body of the email. It has to be selected from a list of templates.

3.2.3.4 Custom Templates

Click this button to bring up the custom templates dialog to manage the email templates.

3.3 Test Manager

The Developer Suite Test Manager helps you manage your tests. It has 4 core views, launch buttons for the Test Creator, Log Viewer and a wizard to help you submit jobs (tests) to be run on a scheduled basis. It gives access to the scheduler functionality of StormTest Development Center and allows you to see results of tests, both your own and other users'.

If this is your first use of the Test Manager, you will need to:

- Upload your tests using the Test Scripts view
- Create a schedule using the Test Schedules view or the Job Wizard

Prior to creating a schedule you may wish to check the usage of DUTs in the system using the DUT Allocations view. Once your test has run, you can view the results using the Test Results view.

See also:

- Core Concepts
- Writing a script to be scheduled
- Scheduler capabilities

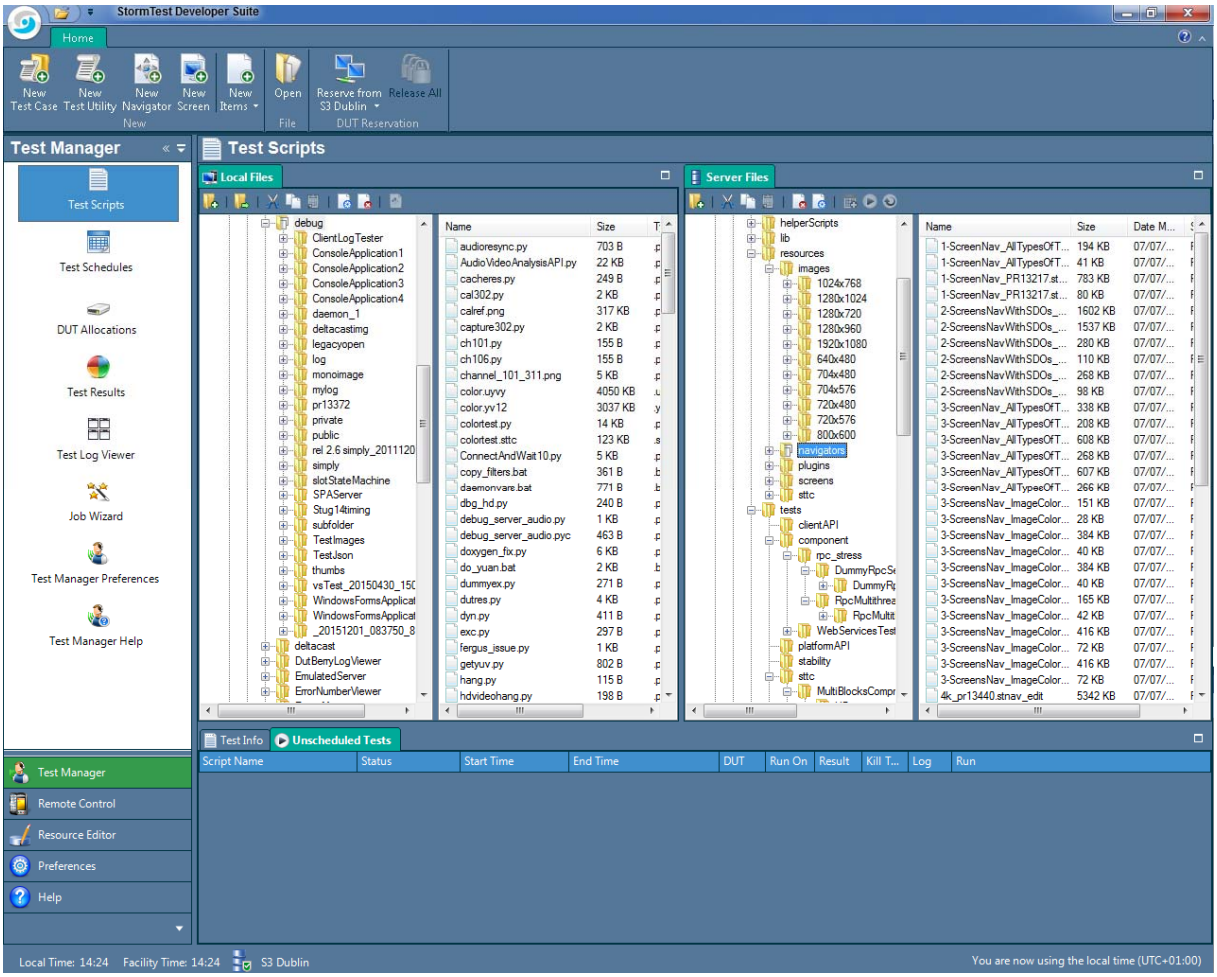
- Running a test without a schedule
- Test Creator
- Navigator
- Screen Definition Object Editor
- Log Viewer

3.4 Scripts

3.4.1 Test Scripts View

The test scripts view manages your scripts. It is divided into 2 major horizontal panels. The upper panel is further subdivided vertically with the left hand panel showing files on your local PC (titled Local Files) and the right hand panel showing the files in the StormTest Development Center file repository (titled Server Files).

The lower panel has two tabs: one to show information about the selected file in the StormTest Development Center file repository and the other tab to show you the unscheduled test status (also known as remote jobs)



3.4.1.1 Usage

Files may be transferred from your local PC to the server using:

- Drag and Drop from Local Files panel to the Server Files panel
- The toolbars above the Local Files and Server Files panel to copy and paste
- Right mouse click and use the menu to copy / paste.

Files may be transferred from the server to your PC in an analogous manner.

When a file in the Server Files panel is selected, the information in the Test Information panel will be updated. It is not automatically made visible if you are looking at the unscheduled tests panel.

Each panel has a maximize button ( ) which expands the panel to fill the work area. It can be restored to the 3 panel view by clicking the restore button ( ).

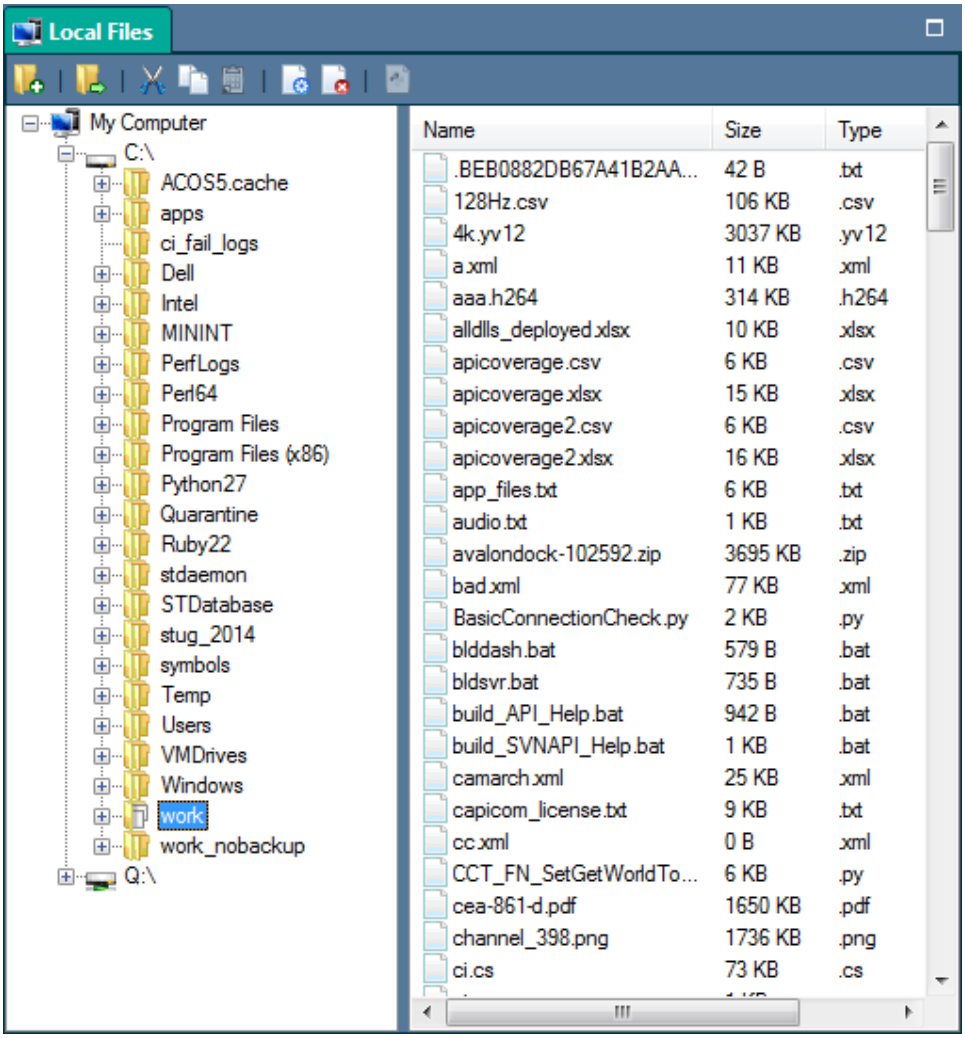
See also:

- Local Files
- Server Files
- Test Information

- Unscheduled Tests

3.4.2 Local Files

The local files panel shows you local files on your PC (or accessible from your PC via a network) organized in a manner similar to Windows Explorer. Files which are marked hidden or system (and not normally shown on your PC) are not shown. There is no option to show such files as they normally are not StormTest script files (or supporting files).



This local view is not a substitute for Windows Explorer - it supplies a subset of the functionality that is likely to be useful in managing files within Developer Suite. The columns above the files can be clicked to sort on any column.

NOTE: The file modification time is **always** local time - even if you are working in facility time for the scheduling.

You can drag files to the StormTest Development Center file repository shown as the Server Files panel. This is a two stage process - first the files are copied with a progress bar indicating the progress and then they are committed to the server (the server notes details about them and informs the client daemons of the change of file), again with a progress bar.

3.4.2.1 Toolbar

The toolbar above the files and the right mouse context menu allow you to:

 Create a new folder on the local hard drive (or network, subject to user privileges).

 Open or execute a file. Developer Suite will use the same method to open the file as Windows does if you double click in Explorer. You can also double click to open. If the file, however, is a Python file (.py or .pyc), you will be presented with a dialog to enter arguments for the script. Since Developer Suite is not involved in allocating DUTs to the test, the python script must handle that itself. If the file is a .sttc file you will be presented with a dialog to decide whether you wish to open the file with Test Creator or run the file as a test script.

 Cut file. It will be deleted but is available for pasting.

 Copy file. It is then available for pasting

 Paste file. If file already exists then it is overwritten.

 Rename file

 Delete file to recycle bin

 When a .sttc Test Creator file is selected, you can convert it to a Python file by clicking this button. A dialog appears allowing you to control how the Python file is created.

3.4.2.2 Context Menu

In addition to the above items, the context menu offers the following options:

Edit with Test Creator. This is only available for .sttc and .sttu files. They will be opened in Test Creator for editing.

Edit with Python Win, **Edit with IDLE**. This is only available for .py files. They will be opened in the python editor of choice. If you add extra Python editors and register them with Windows Explorer, they will also show up here.

New Test Case. This will create a new test case using the graphical Test Creator.

New Test Utility. This will create a new test utility using the graphical Test Creator. A test utility is a collection of actions that can be used by many different test cases. It is analogous to a subroutine in conventional programming languages.

3.4.2.3 Log Files

If you have run a test locally, you can browse to the log file folder and drag the logfile.html to the log viewer icon () and the log will be opened for you. You can also drag such a file from Windows Explorer.

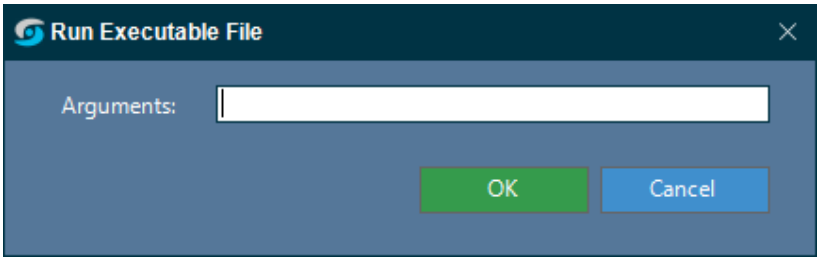
NOTE: the copy / paste does not copy / paste files to / from Windows Explorer.

 Clicking the maximize button will cause the local files panel to fill the complete applet area.

 When maximized, clicking the restore button will restore the normal view.

3.4.3 Run Now on Local Machine

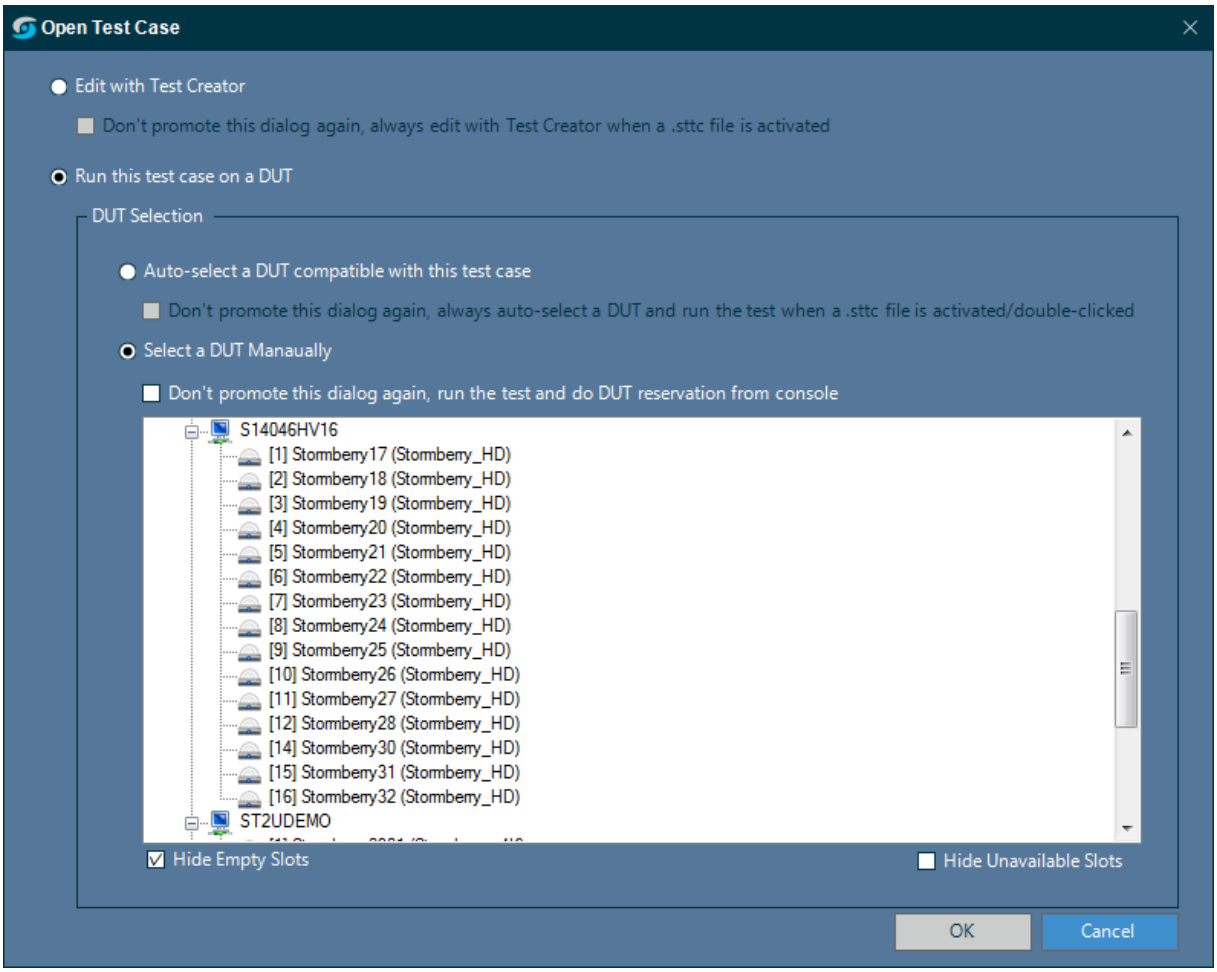
From the local files panel of the test scripts view you can select a .py, .pyc and .sttc files to run/execute. If you select a .py or .pyc Python file then you will see the dialog:



Enter any arguments that the script needs and click OK.

NOTE: In this case, Developer Suite is not involved in allocating DUTs to the script so your script must handle that. This facility is provided primarily for compatibility with previous versions of StormTest Development Center GUI application.

If you select a Test Creator .sttc file, you will see the dialog:



3.4.3.1 Usage

You select the DUT you wish to use for the test. You can choose automatic selection or a specific DUT. If you choose automatic selection, then Test Manager will choose a free DUT of the model type which matches that of the test. If no such models are free then the test will not run.

If you select a DUT manually then that will be used even if the model does not match the model of the test. Whether the test executes correctly cannot be determined in this case. For example, it may try to send IR commands that do not exist on the physical DUT - you will receive an error at run time and the test will terminate.

A DUT which is in use is shown in gray. If you hover over it, a tool tip appears showing you who is using the slot. A slot which is reserved to you but not actually in use will show as normal - you can use it as you reserved it. The tooltip will tell you that you have reserved it.

3.4.3.1.1 Hide Empty Slots

This is checked by default and the display of DUT's is filtered to remove any slot which does not have a DUT.

3.4.3.1.2 Hide Unavailable Slots

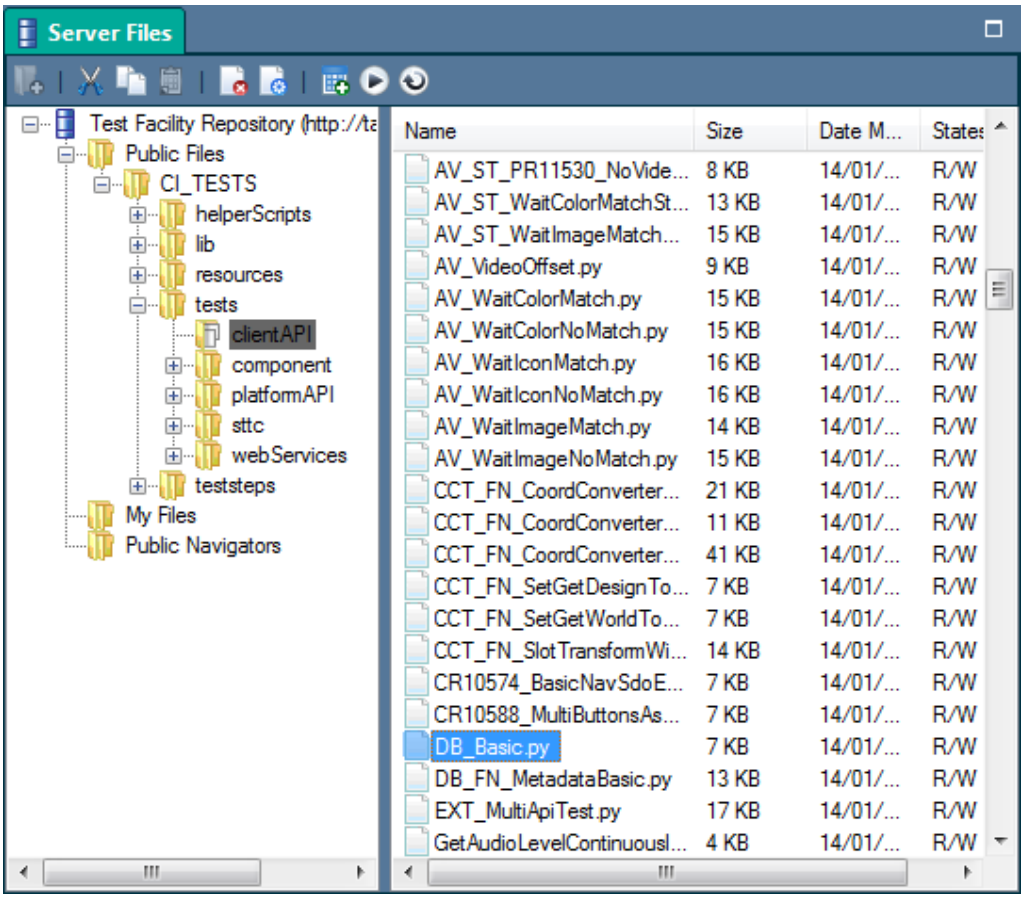
This is not checked by default. If you check it, then the display hides any DUT which is in use.

3.4.3.2 Test Execution

During test execution a black window will appear allowing you to see the progress of the test. Depending how you set up the test, you may also see live video. The window will disappear when the test has completed.

3.4.4 Server Files

The server files panel shows you the files in the StormTest Development Center file repository, organized in an analogous manner. There are three top level folders in the file repository and no user can create more top level folders nor can any user delete these folders.



They are:

- Public Files - all users see the same copy of public files. This folder is intended for use by all users. Anybody can create or delete files and folders in the Public Files folder.
- My Files - each user is allocated a private area for their files. Only you may view, create or delete files in your private area. It is possible for other uses to see the names of files that you create in this folder if you add them to a schedule. Any use may see the results of tests and this includes the full path to the file used to run the test.
- Public Navigators - all users see these files. The screen navigator is a component added in Version 3 of StormTest and is a file which encapsulates the navigation logic of a DUT. Any test can use a navigator to speed up the process of writing tests. Public Navigators can be assigned to a DUT model in the Admin Console so they can be used automatically in a test. You can upload navigator files to your private files or to the public files area but then your test script is responsible for identifying the file and choosing the correct navigator to use. This may be most convenient during the early days of developing a navigator. NOTE: You can only add files with the StormTest navigator file extensions (.stnav_edit and .stnav_runtime) to the Public Navigators folder.

You can drag files from the StormTest Development Center file repository to the local files panel. The columns above the files can be clicked to sort on any column.

NOTE: The file modification time is **always** local time - even if you are working in facility time for the scheduling. Hovering over a file will show a tooltip with information about the last test run on that file:

The toolbar above the files and the right mouse context menu allow you to:

 Create a new folder in the file repository. You will be prompted for a folder name.

 Copy file. It is then available for pasting.

 Paste file. If file already exists then it is overwritten.

 Delete a file or folder from the server repository. You will be asked for confirmation before doing this. Note: Although deleted, this operation does not free up space on the server as the internal repository is an optimized compressed database of files.

 Add the file to a schedule. You will be prompted for the schedule to use along with the arguments to pass to the test. This option is only enabled for .py, .pyc and .sttc files.

Run now as an unscheduled test. You will be prompted for the arguments and DUTs to use. Then the test will run under the control of one of the available client daemons. This option is only enabled for .py, .pyc and .sttc files. The status and result of the test is available in the unscheduled test panel.

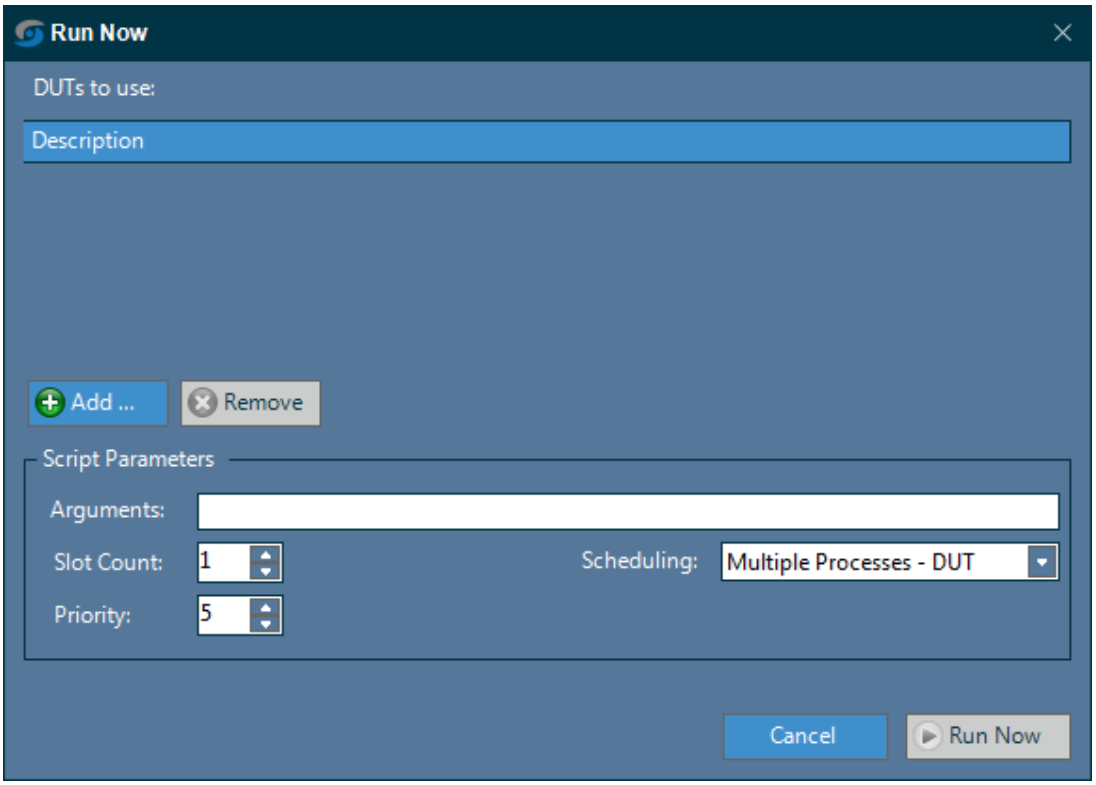
Rerun the selected script. This will pick up the most recent unscheduled test on that script that you ran and re run it with the exact same parameters on the same slots as it ran before. If you chose a single script but allocated a pool of 10 slots and the script used just one slot then the rerun will be on just the slot that it used last time - it won't get allocated the original pool of 10 slots. If you have not run the script before then the button is disabled.

Clicking the maximize button will cause the local files panel to fill the complete applet area.

When maximized, clicking the restore button will restore the normal view.

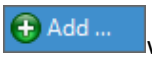
3.4.5 Run Now on Server

From the server files panel of the scripts view, you can select a .py, .pyc or .sttc file to execute. This will run under the control of the scheduler and a client daemon. You will be presented with a dialog to control how the test will run:

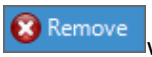


3.4.5.1 DUTs to Use

You need to specify one or more DUTs to use for your test. This is controlled by the buttons:



will bring up a dialog to allow you to add DUTs to use. Note that you can only add DUTs that are currently in a physical StormTest Development Center server within the facility. Normally, you can add DUTs to a schedule that are configured to be part of the facility but are not configured to be in a server. The expectation is that they will be moved to a physical server when the schedule runs. However, for a 'run now',this makes no sense so that option is not available. Note also, that the only method of adding a DUT is by DUT name - the dialog does show you the model and physical location. Again, the reason is that you will be running the test 'now' so the current location of DUTs is important - not where they might be later.



will remove the selected DUTs from the run now - useful if you make a mistake adding the wrong DUT.

3.4.5.2 Script Parameters

Here you specify how the test should run:

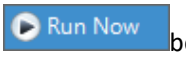
Arguments are passed directly to the script - enter whatever is needed by your script. The .sttc Test Creator files do not need any arguments.

Slot count is the number of slots that you wish to run the script on. You can enter a number less than the number of DUTs allocated but not more. If you select a lower number you are saying to StormTest Development Center that it should find the number needed from a larger pool of possible DUTs to use.

Priority is the priority of the script and influences the order of execution of other 'run now' scripts that you may have waiting. 1 is highest priority, 10 is lowest. It has limited effect unless you submit a lot of 'run now' scripts on the same set of DUTs. In this case you will find that you may pre-empt a waiting script with a new one by submitting it at a higher priority. Once submitted, you can not change the priority.

Scheduling Mode determines how the scheduler will schedule jobs of equal priority. This has limited value for the 'run now' type of job - other than to select single or multiple process mode.

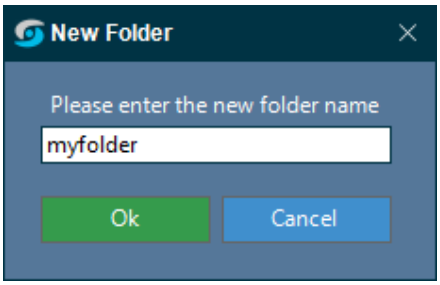
3.4.5.3 Starting the Job



becomes enabled only when you have selected a DUT. Click on it to start the job. Developer Suite checks the parameters and informs the scheduler of the job. It will then appear in the unscheduled tests panel.

3.4.6 New Server Folder

When you create a new folder in the server repository (from the Server Files panel), you will be presented with a small dialog:

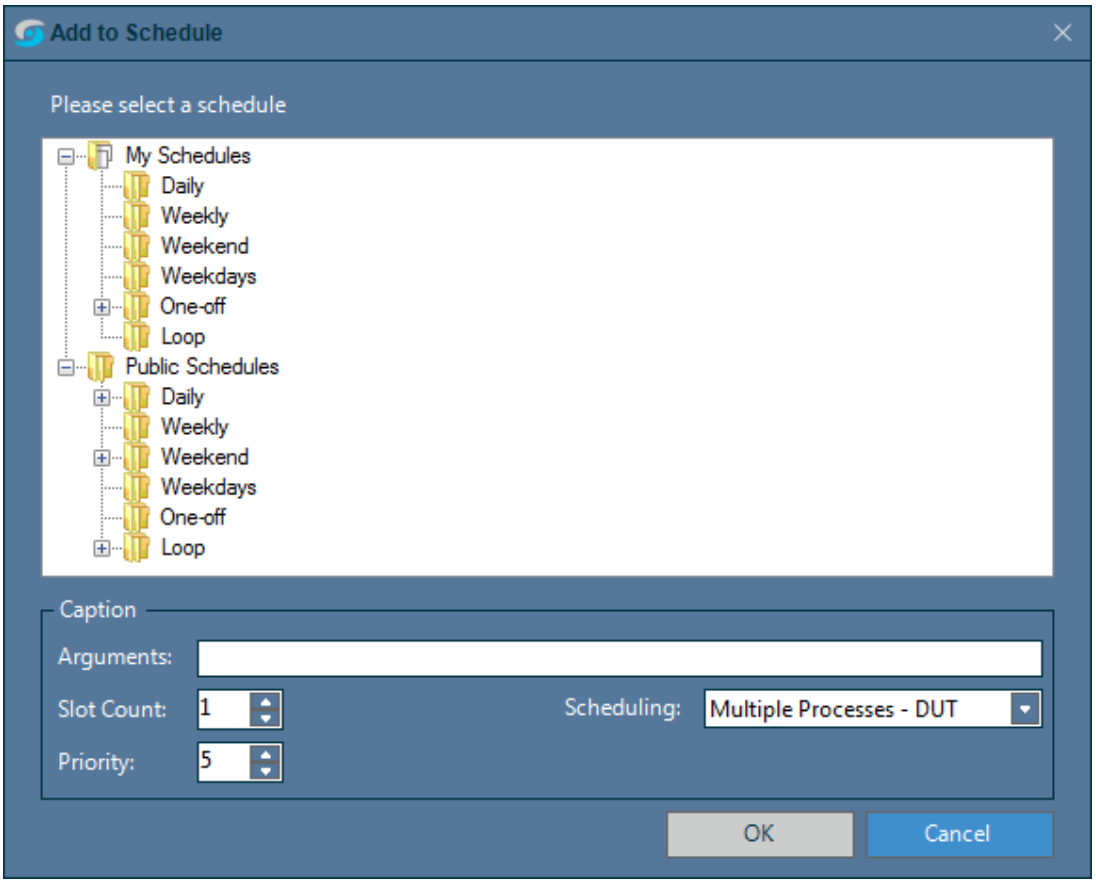


You need to supply a unique name for the new folder and click OK. If you wish to cancel, just click cancel and no folder will be created. If you supply a name that already exists, then no new folder is created but you don't get a message box about this.

NOTE: Folder names are not case sensitive - "abc" is the same as "ABC" as far as Developer Suite is concerned.

3.4.7 Add to Schedule

The add to schedule dialog allows you to specify a schedule for the script, the arguments and scheduling mode for the resulting job.



You select a schedule from your available schedules or public schedules. You cannot create a new schedule at this stage.

Arguments are passed directly to the script - enter whatever is needed by your script.

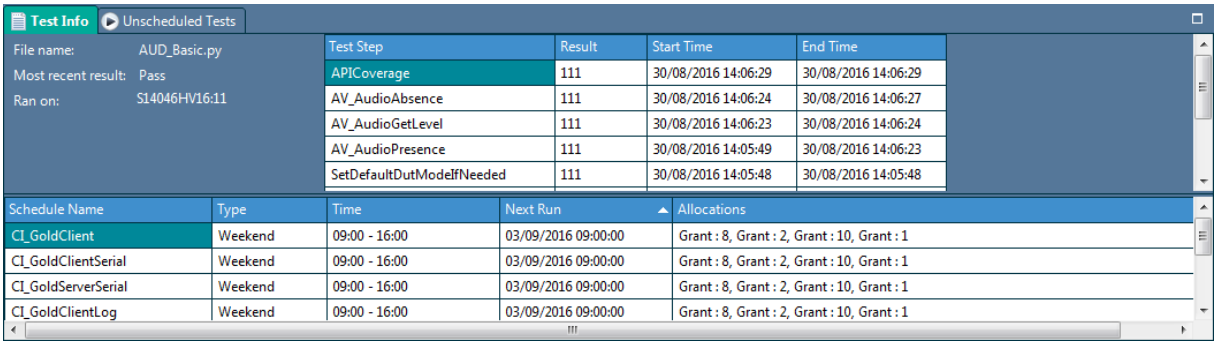
Slot count is the number of slots that you wish to run the script on. You can enter a number greater than the number of DUTs allocated to the schedule, however, the script will run on no more DUTs than are allocated to the schedule.

Priority is the priority of the job and influences the order of execution of jobs on the schedule. 1 is highest priority, 10 is lowest.

Scheduling Mode determines how the scheduler will schedule jobs of equal priority.

3.4.8 Test Information

The test information panel is updated whenever a file is selected in the Server Files panel. It is not updated on a folder selection and if multiple files are selected the values are for the first file selected of the group. The test information shares the screen space with the unscheduled tests panel.



The upper portion shows the selected file, the most recent result and where the test script ran (in the form of server name: slot number). To the right is a resizable panel showing the test steps within the test script. It will be empty, as shown above if test steps are not used. Test steps are defined in the script using the API calls BeginTestStep() and EndTestStep(). The resizing slider is to the left of the grid - it is low profile so not permanently visible. Depending on your screen size, you may wish to make the test step panel larger or smaller.

The lower portion is the list of schedules which use the selected script. Only your schedules are shown - if the script is a public script then it may be used by other users. It also shows when the script will next run as well as the DUTs allocated on the schedule - you cannot edit the schedule from this screen; it is here for informational purposes to save you having to refer to the schedules view in order to see what is happening.

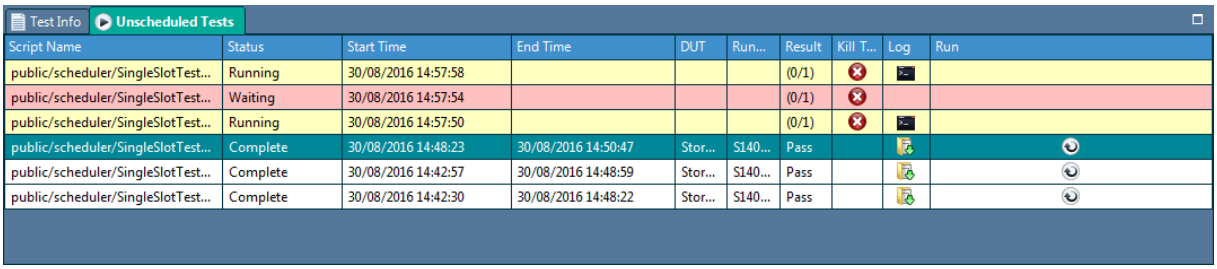
You can sort the list of schedules by clicking on the column headers. Each column can also be resized using the mouse by dragging the divider between columns.

Clicking the maximize button will cause the test information panel to fill the complete applet area.

When maximized, clicking the restore button will restore the normal view.

3.4.9 Unscheduled Tests Panel

The unscheduled tests panel is part of the scripts view. It shows the status and results of all your tests (and only your tests) that have run via the 'run now' ( ) feature of the Server Files panel.



Each column is resizable by clicking on the divider between it and an adjacent column and dragging the divider. The columns can be sorted by clicking on the header.

A non idle test is shown with a colored background with the status in the second column. A test can be killed by clicking the red button in the 'Kill Test' column.

While running, you can view the log in the live log viewer. Files created with Test Creator (STTC files) do not output anything on the console so will show nothing in the live log viewer. Python files will show the output that would be shown on the console.

When complete the 'Log' column contains a link to the log files. Clicking the icon opens the built in Log Viewer and starts to play back your test. If you right click, you will see a menu to view the log files. If you click that, Developer Suite opens your default FTP client software to browse the folder of log files. This would normally be your preferred Web Browser program. If you get a message about a missing FTP client configuration and a Title of "Win32 Exception", please consult you local IT expert to configure an FTP client. If the test was run locally on your PC, then Windows Explorer will be opened instead of an FTP client.

When all scripts of an unscheduled test are complete, the rerun button appears. You can click this to rerun the unscheduled test again on the slots that it has run on. If the unscheduled test has multiple scripts then all will be rerun. If you used multiple slots, then the same set of slots will be used. However, the scripts may not be allocated to the same slots on the re run.

Clicking the maximize button will cause the unscheduled tests panel to fill the complete applet area.

When maximized, clicking the restore button will restore the normal view.

3.4.9.1 Unscheduled Test Status Values

An unscheduled test can have the following statuses:

- Active - it has been submitted and is waiting for the allocated DUTs to become free
- Running - it is running the test on the allocated DUTs
- Complete - it has completed and there is a result (whether it be pass or fail). If multiple scripts have been added to one unscheduled test, the color will show yellow until all scripts on the test have completed and then go white.

3.4.9.2 Run Now on Multiple DUTs

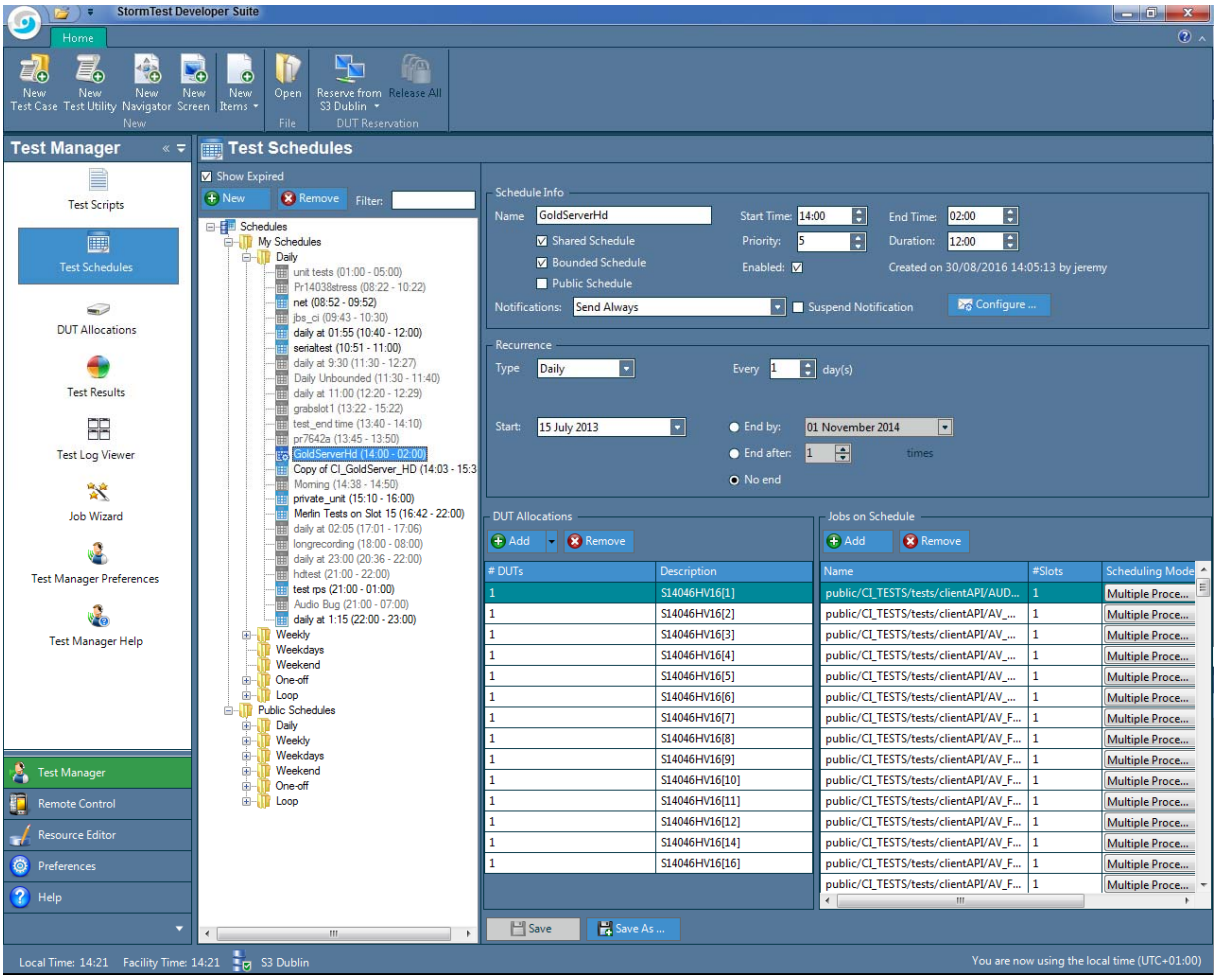
It is possible to set up a 'run now' job on multiple DUTs using the multi-process mode. In this case, killing the unscheduled test will terminate all processes of the test that have not run to completion - it is not possible to kill just one instance in this case.

3.5 Schedules

3.5.1 Test Schedules View

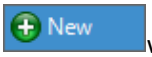
The test schedules view manages your schedules. It is divided into 2 major panels vertically. The left hand panel shows you all your schedules. The right hand panel allows you to view, create and edit an individual schedule. It is subdivided into the panel:

- The schedule details panel occupies the upper portion and controls when the schedule will run.
- The DUT allocations panel occupies the lower left portion and controls which DUTs will be used by the schedule
- The jobs panel occupies the lower right portion and determines which jobs will run on the schedule.

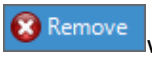


3.5.1.1 Usage

Your schedules are organized by type of schedule, represented by folders in the left hand panel. You can expand the folders to see your schedules by clicking on either the folder itself or the little '+' sign. If you click on an existing schedule, the right hand panel opens up so that you can edit it.



will create a new schedule. If you have a folder selected when you click 'new' then the schedule created will be of that type, else it defaults to 'daily'. You may freely change the type during the edit phase.



will remove a selected schedule.

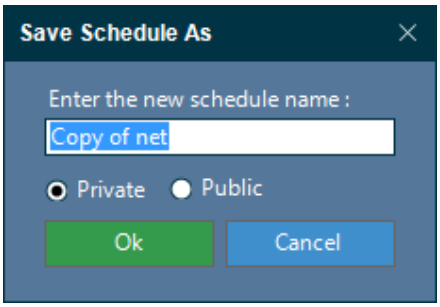
You can filter the view of schedules using the filter box next to the Remove button. The filtering is not case sensitive and shows all schedules (yours and public) where the specified text is in the schedule name.

will save an edited schedule. It is enabled only after making a change to the schedule. If you forget to save the schedule and try to navigate away from the view, Developer Suite will prompt you to save the schedule so that you don't lose your work.

will save a schedule as a copy, creating a new schedule and leaving the original unchanged. The saved schedule must be valid so if you select an expired schedule, you must change the date/time and then use Save As to make a copy.

3.5.2 Save Schedule Copy

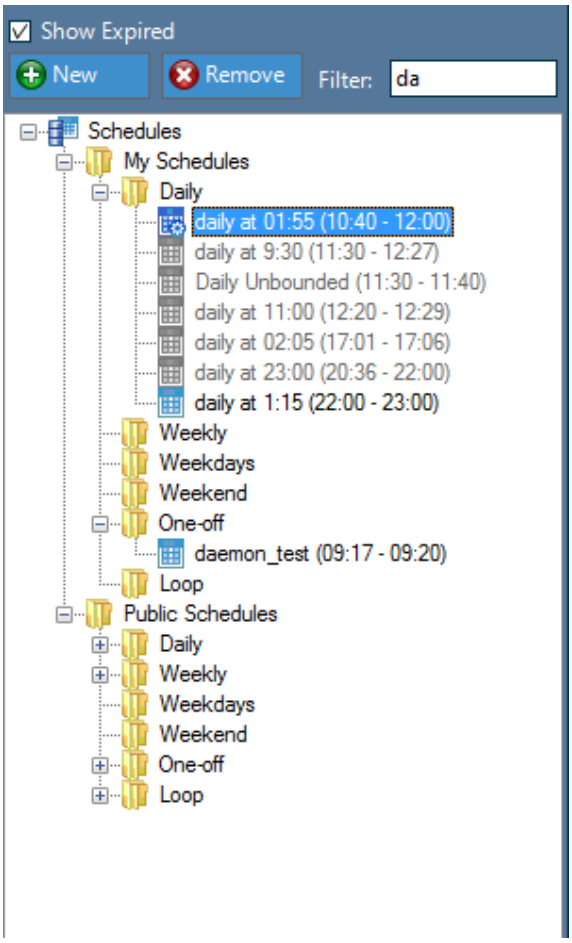
When you click the Save As button on the schedules page, you are presented with a dialog to save a copy of the schedule:



You enter a new name for the schedule (by default it will be name "Copy of " plus the original name). By default the schedule will be saved as a private schedule but if you select the Public radio button then it will be saved as a public schedule.

3.5.3 Your Schedules

This panel shows your schedules and public schedules. The schedules are organized into folders based upon whether the schedule is private to you or public and then the type of schedule.



By default, Developer Suite shows all your schedules and all public schedules including those that have expired. The check box at the top of the panel can be used to filter the view to show only the non expired schedules.

A schedule is defined as expired if it will not run again. This happens when, for instance, you set up a schedule to end on a certain date and that date is in the past or run a pre-defined number of times and it has run that number of times.

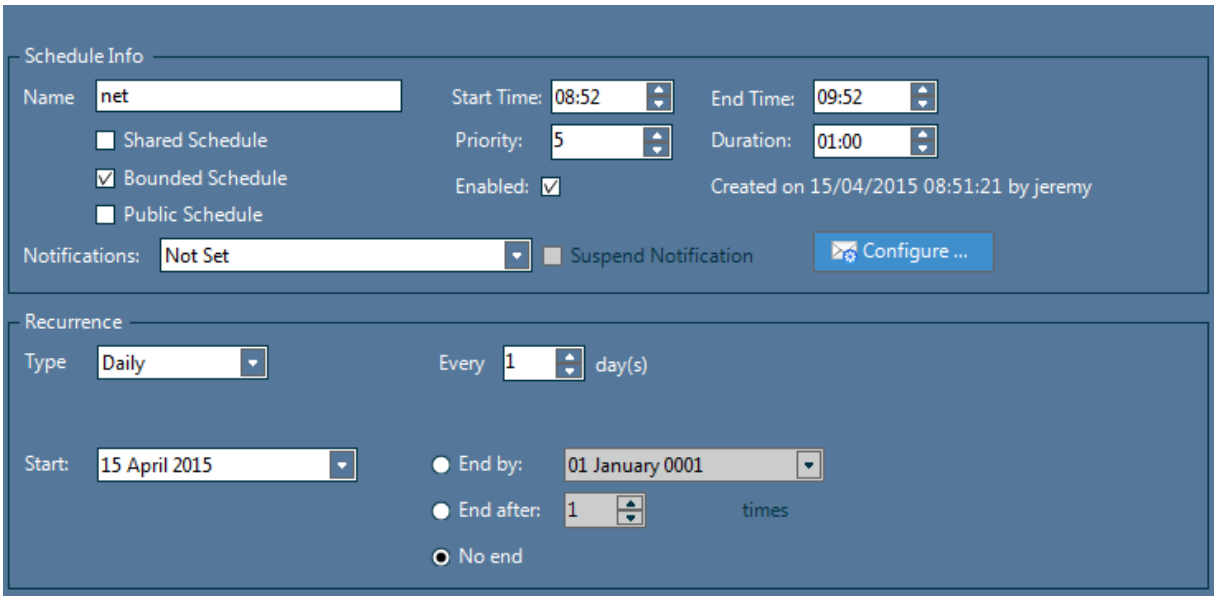
If you edit an expired schedule, you can convert its status to non expired and it will run again.

The image above shows the schedule 'daily at 01:55' as being edited. It is a daily schedule (because it is in the Daily folder not because of its name). The value in brackets '10:40 - 12:00' above is the actual time it will run - as you can see that is not as the name would suggest in the above example.

You can filter the view of schedules using the filter box next to the Remove button. The filtering is not case sensitive and shows all schedule (yours and public) where the specified text is in the schedule name.

3.5.4 Schedule Details

The schedule details determine when the schedule runs.



The upper panel is common for all types of schedule and contains the fields:

Name - the name of the schedule. It is only used for display purposes and has no other significance

Start Time - the time that the schedule will start on each day that it starts. It is hours:minutes in the format that is configured on your Windows machine. The schedule will always start on a whole number of minutes.

End Time - the time that the schedule will end on each day. If this is before the start time then the schedule runs over the day transition at midnight. Units are the same as for start time.

Priority - the priority of the schedule. 1 is the highest priority, 10 is the lowest.

Duration - the length of time the schedule will run. This can be used with the start time instead of the end time. The end time will be updated as you change the duration.

Created on - the date and time that the schedule was created. This is not changed on modification of a schedule. It also shows the user who created the schedule

Shared schedule - marks the schedule as a shared schedule.

Bounded Schedule - marks the schedule as bounded. This is the default.

Public Schedule - marks the schedule as public. This allows other users to edit your schedule or to move tests to it.

Enabled - marks the schedule as enabled. If you disable a schedule, the scheduler will not run the schedule but it has not been deleted. This is useful for temporarily prevent schedules running. A disabled schedule is shown greyed out in the schedule list.

Notifications - Use this area to set up an email notification when the schedule ends. You can set up default options for the email in the Test Manager preferences screen. Then you can quickly set up a notification for any schedule by selecting under what circumstances you want an email. If you don't set the preferences then a dialog appears for detailed configuration. Your options are:

- Not Set - never send an email
- Send on Pass - send an email if all tests on the schedule return a Pass Status
- Send on Fail - send an email if any test on the schedule returns a fail, or any test fails to run
- Send always - always send an email

Suspend Notification - if you check this then the configured notification will no longer be sent. This is useful to temporarily suspend the notification without deleting it. You can recreate the notification by unchecking the check box.

Configure - Bring up a detailed dialog to configure the email notifications.

The lower panel controls the recurrence of the schedule. The fields that are visible depend on the type of schedule:

Type - the type of the schedule. Details are described in the scheduler features.

Start - the date on which the schedule first runs. Clicking on the arrow to the right will bring up a calendar. You can also click the individual fields of day month or year and type the desired value. Use a number for the month not the name.

Every - this is the scheduling interval. The units 'day(s)' in the above view will change as appropriate. You can set up a daily schedule to run every 2 days, for example. It is disabled for the weekday and weekend types. It is not visible for the loop or one off types

Count - the number of times to run a loop schedule.

Mon to **Sun** - the days of the week to run a weekly schedule. This is the day the schedule will start (not end).

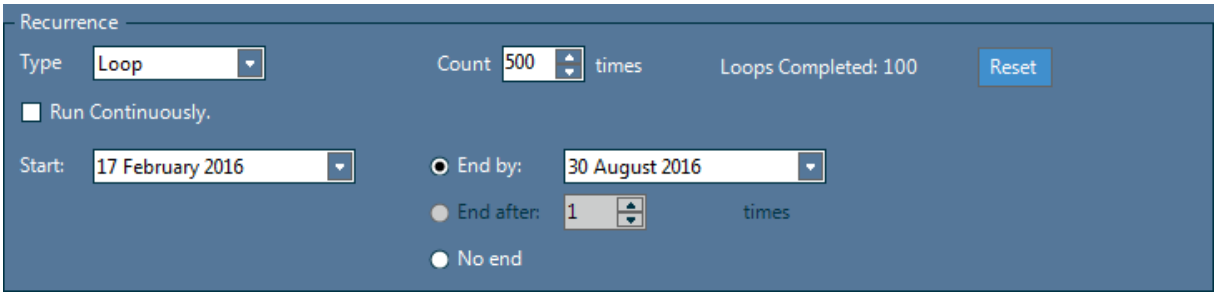
End By - select end by and enter an end date to force the schedule to run up until a specific date. It will run for the last time on this date.

End After - select the number of times to run. This is disabled for loop and one off schedules as it makes no sense.

NOTE: This is the number of times the schedule runs not the length. So a weekend schedule with a End After of 3, for instance will run on 3 distinct days (Saturday, Sunday and following Saturday) nor for 3 weekends.

No End - the schedule will run forever. This is not available for a one off schedule. In reality the schedule will end in January 2038 due to the internal clock used.

A looping schedule has a special setup:



Type - the type of the schedule. Details are described in the scheduler features.

Count - the number of times to run the loop schedule.

Reset - Click this to immediately clear the count of current loops. If the schedule is running, it will continue to run as will any test in progress but the count will have been reset to zero at the instant you clicked the button.

Loops Completed - this shows you how many loops have been completed. In the example above 8 loops have been completed meaning every job on the schedule has run at least 8 times. Some may have run 9 times.

Run Continuously - Check this to force the loop schedule to run continuously. In this case the start time and end time are not used on a daily basis but only to determine the first and last hours of the overall run. If not checked then the schedule will run only between the start and end times on each day.

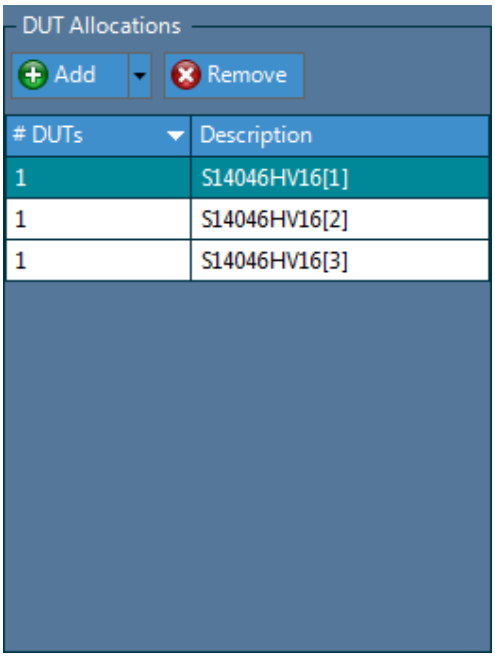
Start - the date on which the schedule first runs. Clicking on the arrow to the right will bring up a calendar. You can also click the individual fields of day month or year and type the desired value. Use a number for the month not the name. The first run will start at on this day at the time specified in the upper panel.

End By - select end by and enter an end date to force the schedule to run up until a specific date. It will end at on this date at the time specified by the end time in the upper panel.

No End - the schedule will run forever. In reality the schedule will end in January 2038 due to the internal clock used.

3.5.5 DUT Schedule Allocations

Every schedule will need one or more DUTs allocated to it in order to run a test script.



The panel shows which DUTs are allocated to the schedule. The format depends on how the DUT is allocated. There are 3 ways to allocate a DUT to a schedule. Clicking the arrow to the right of the Add button brings up a menu:

3.5.5.1 Add by Name

Adding a DUT by name allows you to add an explicit DUT to the schedule. Just that DUT will be used. A dialog will appear allowing you to browse the available DUTs. DUTs added like this will appear as 'DUT name (model name)' in the list.

3.5.5.2 Add by Slot

Adding a DUT by slot allows you to add a physical slot to the schedule. Whichever DUT is in the slot at the time that the schedule will run is used. This does not always lead to predictable test results, however, it is the way earlier StormTest products worked so is provided for consistency. A dialog allow you to select the slot(s). DUTs added like this will appear as 'Server Name[slot number]' in the list - this is in fact as shown above.

3.5.5.3 Add by Model

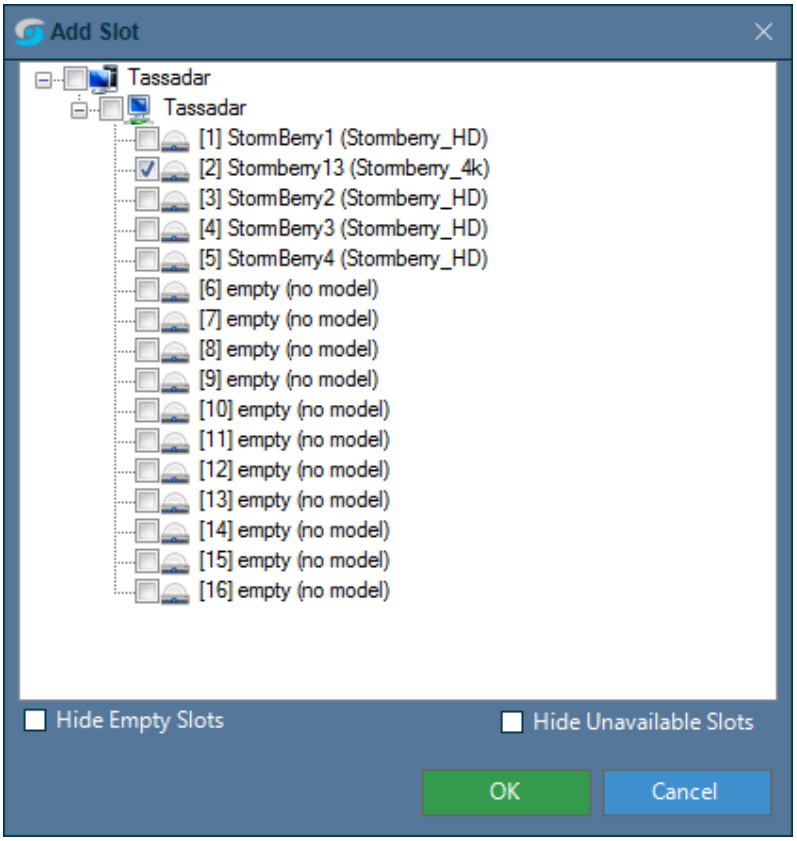
Adding a DUT by model allows you to specify a model type and a maximum number to use for your schedule. The scheduler will find the models at run time that are free and use them. A dialog will appear showing you the models available. DUTs added like this will appear as 'Model Name (Model Type)' in the list. Also the column headed #DUTs will show how many of the model will be used.

3.5.5.4 Removing DUTs

Simply click on the remove button and the DUT will be removed.

3.5.6 Add Slot to Schedule

This dialog allows you to add DUTs by slot to a schedule.



Simply expand the tree view and check the slots you want to add to the schedule. The expanded tree shows you the current DUT name and, in brackets, the model that is currently in the slot. You can select or deselect all the slots in a server by checking or unchecking the box next to the server name. For obvious reasons, you cannot select all the boxes in the facility - despite the check box beside it.

3.5.6.1 Hide Empty Slots

Check this to hide all slots without an DUT present. You can add slots to a schedule but no test will run until you use the Admin Console to add a DUT to the slot.

3.5.6.2 Hide Unavailable Slots

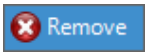
Check this to hide slots that are currently in use. This is most useful when you are scheduling a test to run in the next few minutes and want to make sure the box is not in use.

3.5.7 Add DUT to Schedule

This dialog allows you to add DUTs by name to a schedule or to a run now immediate execution test.

You can sort the DUTs by clicking the column header and adjust the width of each column by clicking on the divider between columns and clicking.

Check the DUTs you wish to add to the schedule / run now test. When the dialog appears, it will have check boxes against those DUTs already on the schedule - you can remove them by unchecking

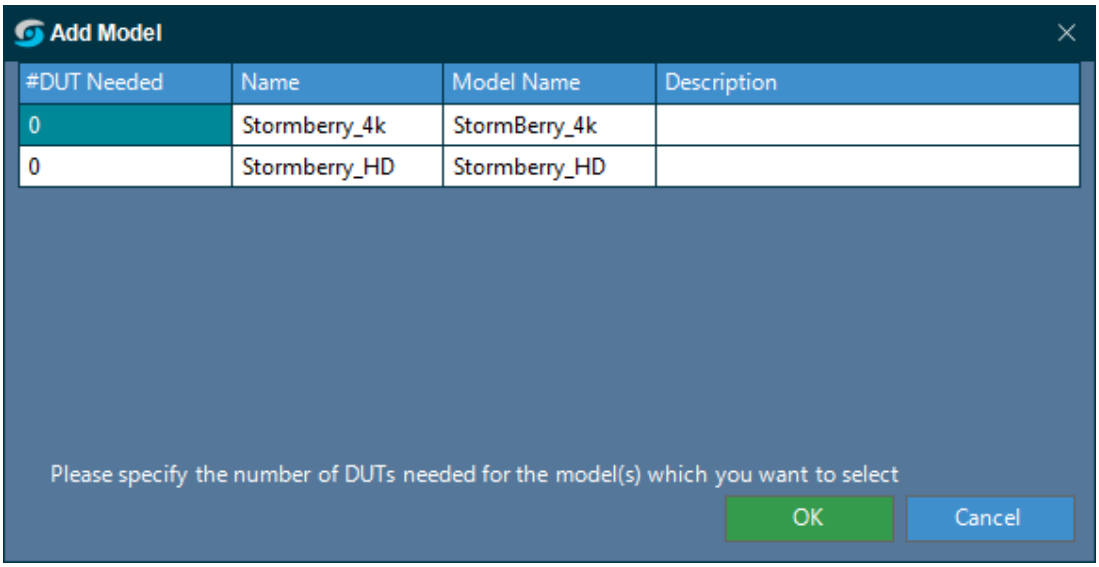
the box(es). This is not the only way of removing DUTs from a schedule / run now test. They can be highlighted from the schedule / run now screen and the 

button can be clicked.

When you are adding DUTs to a schedule, you may see the server & slot column empty - this means that the DUT is defined in the facility but is not in a physical rack at present. Although you can add it to a schedule, it will not be used unless you put it in a rack and use the admin console to inform StormTest Development Center about the position of the DUT.

3.5.8 Add DUT Model to Schedule

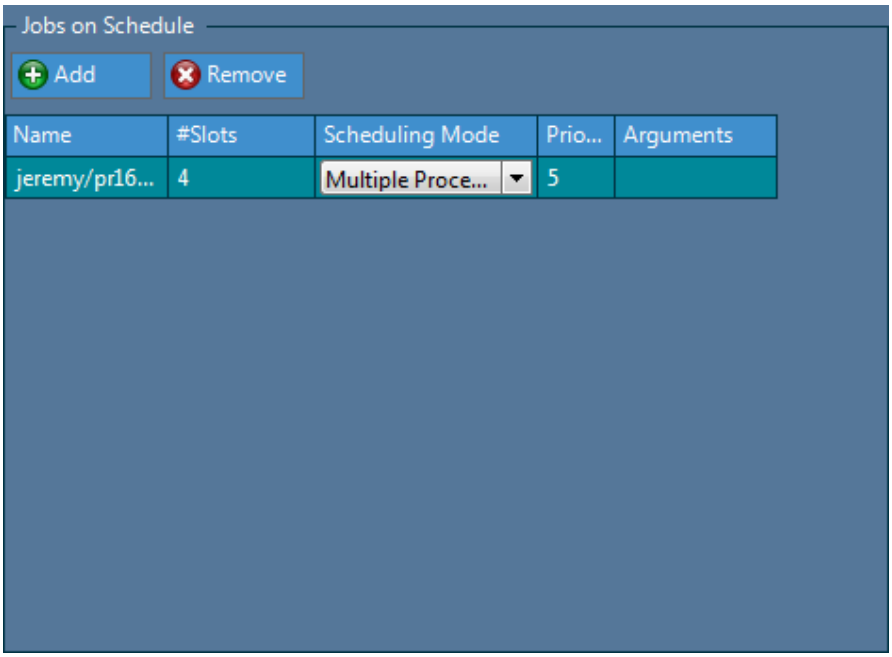
This dialog allows you to add DUTs by model to a schedule. You can specify a maximum number of models of a type to use. This is useful when you have a lot of DUTs of the same model to be shared among several testers - if there were no limits then one tester could easily monopolise all the DUTs of a single model.



Simply type in the number of DUTs needed against the model you desire. If you make a mistake, you can make sure none of a model are added by setting the number to 0. The other columns are for information purposes.

3.5.9 Scheduled Jobs

The scheduled jobs panel shows which jobs are on the schedule and how they will be scheduled.



Clicking on the add button will bring up a dialog allowing you to browse the server repository and add scripts to become jobs on the schedule. You may only schedule scripts which are in the server repository. The scripts view allows you to move script files and resources from your local machine to the server repository. It is important to note that all image files and supporting library files must also be placed in the server repository.

Clicking the remove button will remove a job from the schedule.

The panel shows you information about the jobs. You can sort each column by clicking the header. You can resize each column by clicking on the divider between each column and dragging the column. The columns show:

Name - the path to the script file in the server repository. Public scripts begin with public/ while private scripts begin with your user name. It is possible to mix public and private scripts within a schedule.

#Slots - the number of DUTs that will be used by the job.

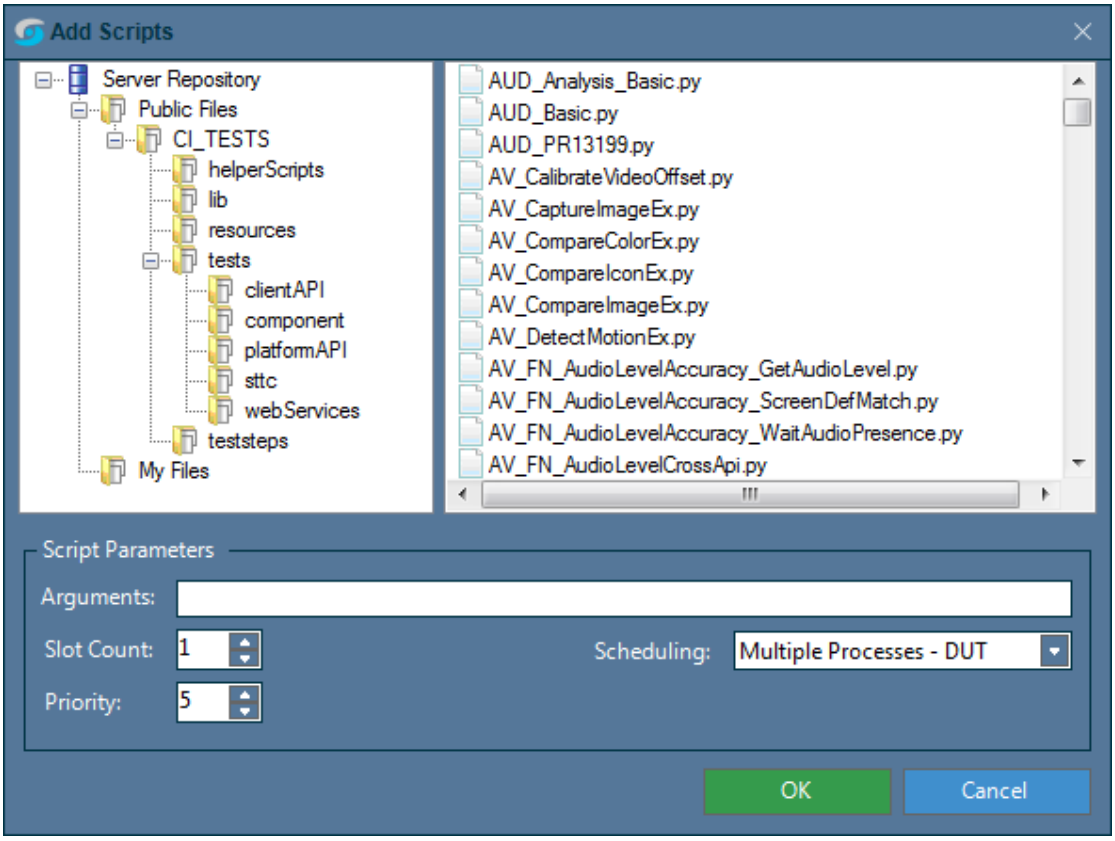
Scheduling Mode - how the job will be scheduled.

Priority - the priority of the job. 1 is highest, 10 is lowest

Arguments - optional arguments to be supplied to the running script. They must not exceed 200 characters in length in total - this is a Windows limitation.

3.5.10 Add script to Schedule

This dialog allows you to add scripts to a schedule.



You may select multiple scripts within the same directory using the control and shift keys in combination with the mouse.

3.5.10.1 Script Parameters

Here you specify how the job should run. The parameters will apply to all selected scripts (and resulting jobs).

Arguments are passed directly to the script - enter whatever is needed by your script.

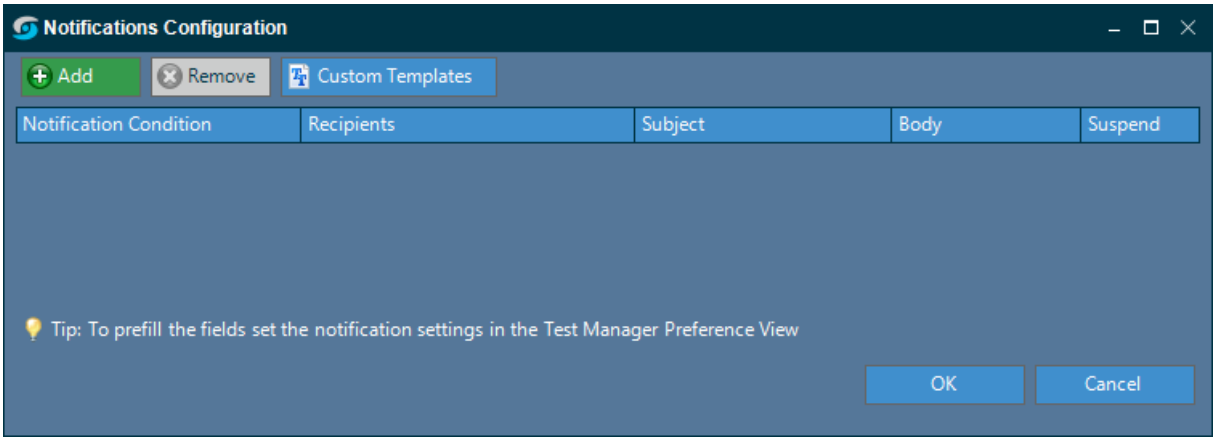
Slot count is the number of slots that you wish to run the job on. The scheduler will attempt to use this number but if they are not all available when the job runs, then the number that are allocated will be used.

Priority is the priority of the job and influences the order of execution of other jobs on the schedule. 1 is highest priority, 10 is lowest.

Scheduling Mode determines how the scheduler will schedule jobs of equal priority.

3.5.11 Configure Email Notification

StormTest can send an email when a schedule completes. For repeating schedules such as daily schedules, an email will be sent each day. To use this feature, the StormTest administrator must have set up the email server in the Admin Console. Otherwise, the feature will be disabled in StormTest Developer Suite.



3.5.11.1 Notifications

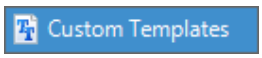
You may add any number of notifications per schedule. Each notification is considered to be independent and if you duplicate the recipients then they will get multiple emails. For each email, you set up:

Notification Condition - this is the condition to send the email. The options are:

- Pass - send an email if all tests on the schedule return a Pass Status
- Fail - send an email if any test on the schedule returns a fail, or any test fails to run
- Always - always send an email

Recipients - List of email recipients. Separate each recipient with a semicolon (;).

Subject - the subject line of the email. It may be blank.

Body - the body of the email. You can leave this blank or select from a list of pre-defined templates. You may add templates of your own by clicking the 

button. The template may also include a customized subject line - this will override any subject line you specify here.

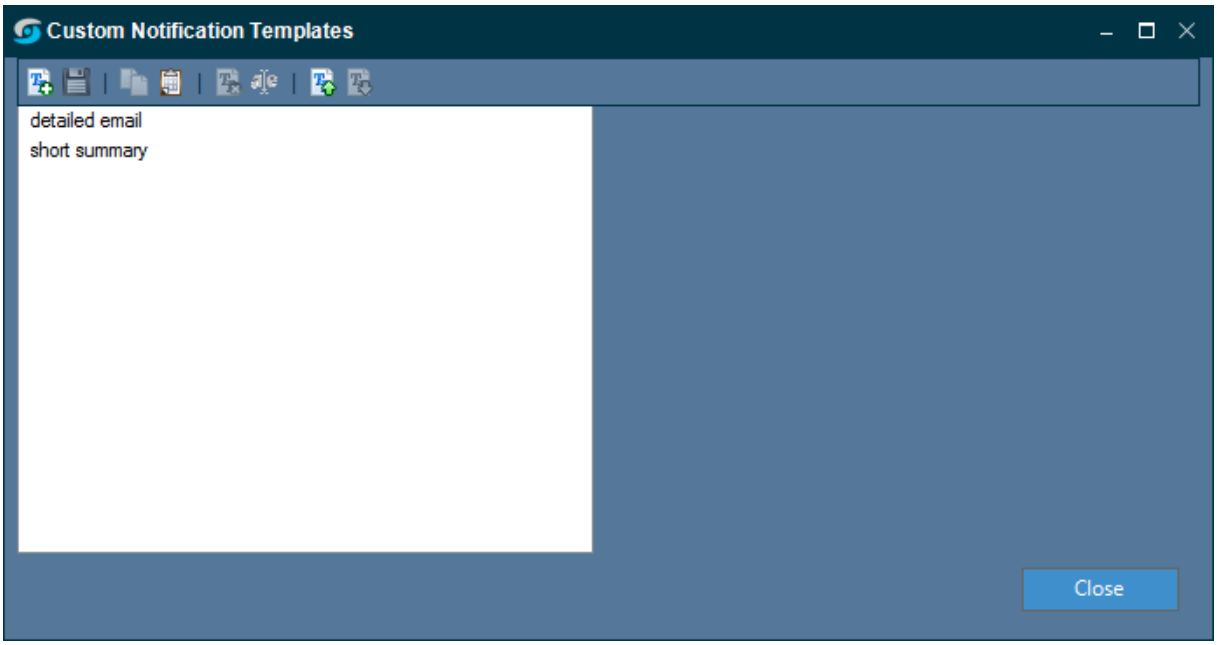
Suspend - suspend this notification. While a notification is suspended, it will not trigger an email.

3.5.11.1.1 Default notifications

If you set up values in the test manager preferences view, these will be entered into each new notification created as the default values. If you leave the subject and body blank, StormTest will use default values for both and generate a reasonably verbose email.

3.5.12 Custom Email Templates

You can customize the email sent with a template. The dialog to manage these templates is:



The options are:

Create a new template. The right hand panel allows you to type in the contents of the template.

Save a modified template.

Copy the template to the clipboard or the highlighted portion of text if editing a template.

Paste text from clipboard into a template or a complete template that has been copied.

Delete the selected template.

Rename the selected template.

 Upload a file from your machine as a template.

 Download the selected template from the server and save as a text file on your machine.

3.5.12.1 Templates:

The template is a text file which uses special keywords which will be replaced when the email is sent with information about the schedule. An email has two parts: a subject line and a body. Both can be specified in the template but the subject line is optional - if omitted the subject line set when configuring a notification will be used. If a custom subject line is used, it must be the first line of the template in the form:

```
$SUBJECT= subject of email.
```

There must be no space before the = symbol. The subject must be all on one line. If the keyword is misspelled, then it will appear to be part of the body and appear in the email.

3.5.12.1.1 Keywords

In both the subject and the body, the following keywords may appear. They will be replaced when the email is generated. The keywords **must** appear exactly as shown below. Although any key word may appear in the subject line, the use of the $RESULTS_DEFAULT or $RESULTS_VERBOSE in the subject line is likely to cause severe problems, possibly preventing the email being sent or causing errors in the email server itself.

$SCHEDULE_NAME - name of the schedule.

$SCHEDULE_ID - internal unique id of the schedule. This is not usually useful other than to identify if more than one schedule has the same name.

$SCHEDULE_START_TIME - time the schedule started. This will be formatted in the style of the locale of the facility.

$SCHEDULE_START_DATE - date the schedule started. This will be formatted in the style of the local of the facility.

$SCHEDULE_END_TIME - time the schedule ended. This will be formatted in the style of the locale of the facility.

$SCHEDULE_END_DATE - date the schedule ended. This will be formatted in the style of the locale of the facility.

$SCHEDULE_STATUS - the pass status of the schedule. This will be PASS, FAIL, PARTIAL PASS or NOT RUN.

$SCHEDULE_PERCENTAGE - % jobs with a pass result.

$USER_NAME - user name on whose behalf the schedule ran

$RESULTS_DEFAULT - the set of results formatted in a high level overview format.

$RESULTS_VERB0SE - the set of results formatted in a verbose format which includes test steps.

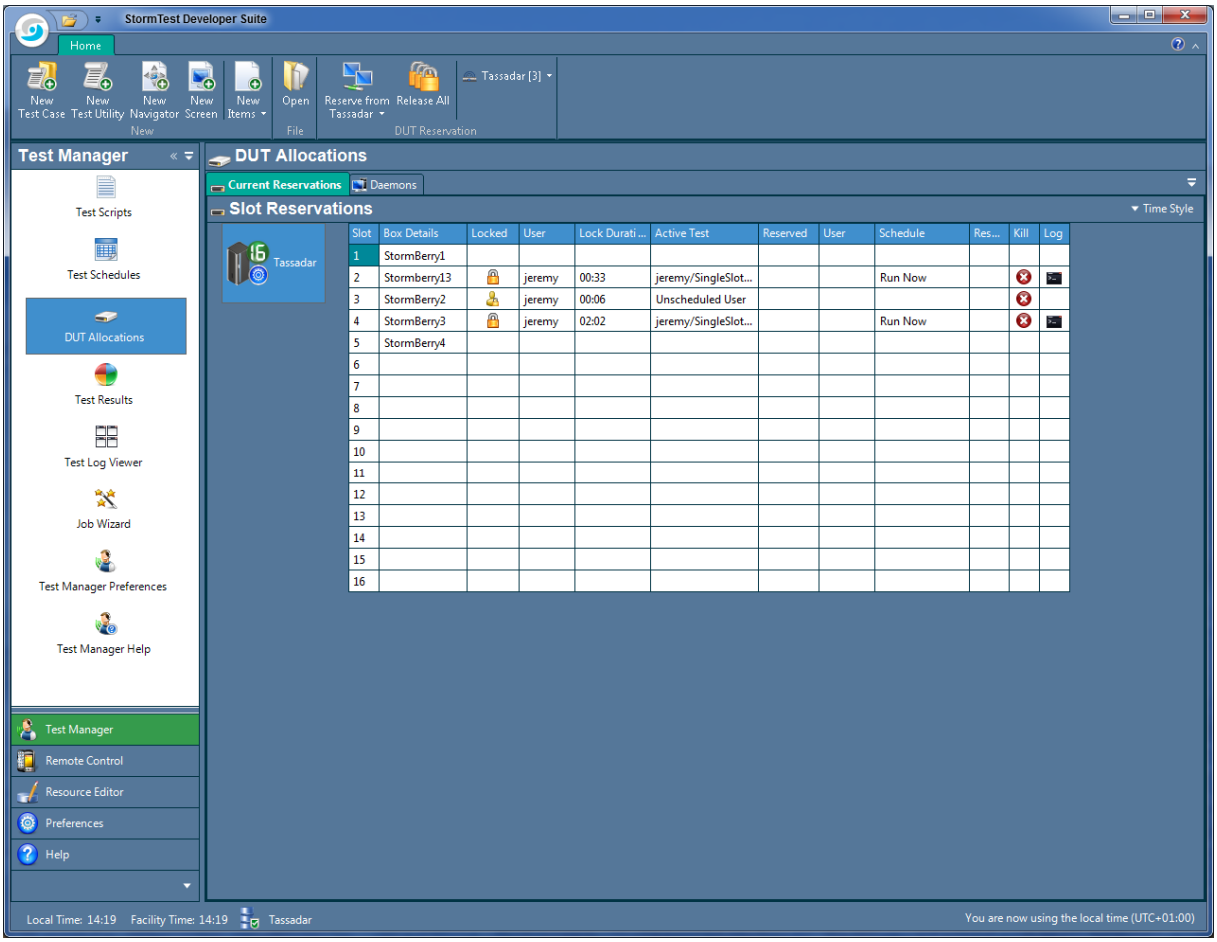
3.5.12.2 HTML Email

The normal format for email is plain text. However, it is possible to create and upload an HTML document to be used to send email. If you do that, StormTest developer suite will not be able to edit the template - it does not provide an HTML editor. Note also that most HTML editors will not allow the $SUBJECT to preceded the HTML doctype or header so that you cannot easily customise the subject line if using HTML email.

3.6 Status

3.6.1 DUT Allocations View

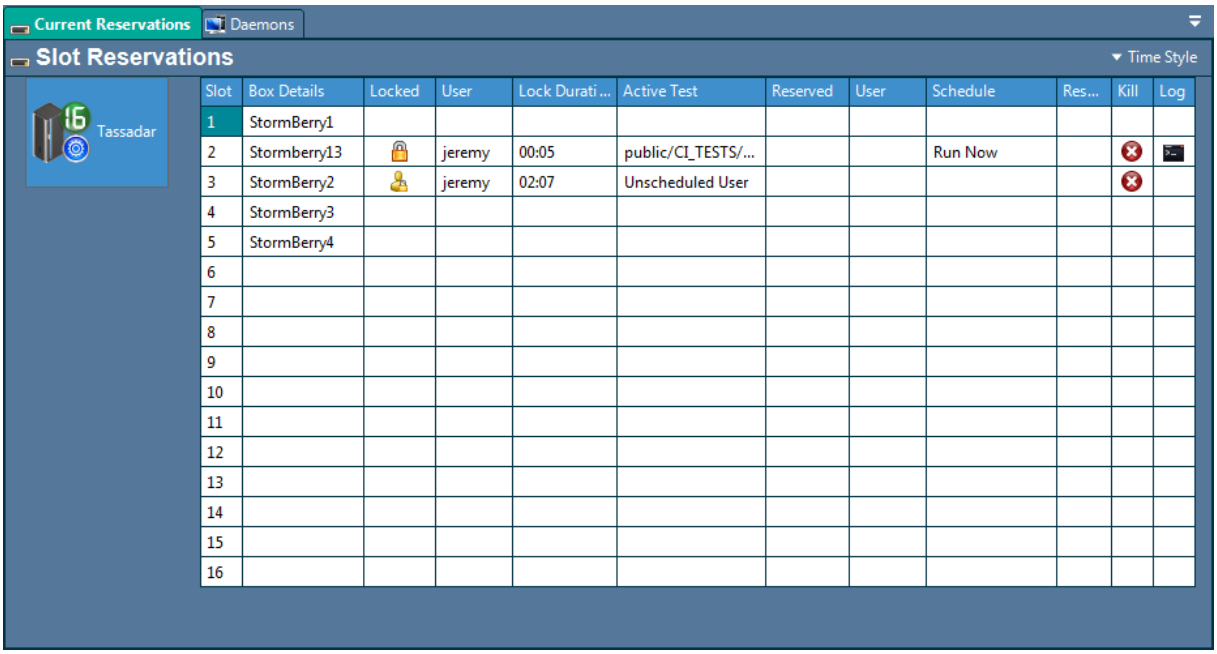
The DUT Allocations view is where you can see who has reserved what DUT and the current status of the use of DUTs. It has 2 tabs:



- Current Reservations - this shows you the status now and is updated live.
- Daemons - this shows you the status of the client daemons and is updated live.

3.6.2 Current Reservations

The current reservations shows you the current usage of DUTs and is updated live:



3.6.2.1 Usage

Select a server from the left column by clicking it. The grid shows all 16 slots and their current status. Columns can be resized by clicking on the divider and dragging and they can be re-ordered. There will be fewer slots on an HV04 or HV01 system. An HS64 system will show 16 slots as the HS64 can reserve banks of 4 only - each bank of 4 is considered 1 slot divided into 4 sub slots.

The columns are:

Slot - the numerical slot number

Box Details - the name of the box in the slot

Locked - an icon appears if the slot is currently in use by a user. The icon appears if the slot is used by the StormTest scheduler and the icon if the slot is used directly from the command shell or a GUI.

User - the user name who is using the slot. This will be the user's Windows logon id.

Lock Duration/Lock Time - the length of time the slot has been locked. A menu allows you to decide whether to display duration of the lock or the time the lock started. The default is to show the duration.

Active Test - the name of the test script (and arguments) if the slot is locked by the StormTest scheduler.

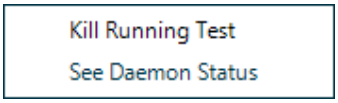
Reserved - a icon appears if the slot is currently reserved - that means a schedule is running which has this slot allocated by slot or by DUT and the schedule is NOT shared. A shared schedule allows other users to use the slots on the schedule so there is no firm reservation on it.

User - the user who owns the schedule that has reserved the slot

Schedule - the name of the schedule which has reserved the slot. Note that if many schedules overlap on the same slot and same time then only one schedule name is listed. If the schedule is a looping schedule, then the current loop number and the total loops will also be displayed.

Reserve End / Remaining Reserve - the time at which the reservation ends. Note that a user may have overlapping schedules with the same DUT allocated to it - the time shown is the earliest schedule's end time. When that time is reached, you will see the reserved until time updated to the next earliest time. It is an indication of the earliest possible time that the slot can become unreserved. A menu allows you to change the display to show the time remaining until the reservation ends.

Kill - a  icon appears if a slot is locked. Clicking it will bring up a menu with two options:



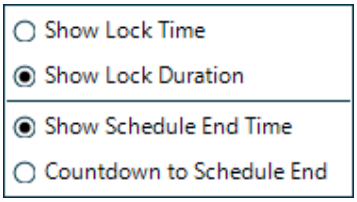
- kill the running test (if scheduled) and release the slot. If an interactive user has the slot locked, then the slot is released and the user will not be able to continue using that slot.
- See the daemon status. This is useful if the slot is locked but nothing is running because the daemon is busy.

NOTE: *If a schedule has slots allocated by model (instead of slot or DUT) then a locked slot will be shown as locked by the scheduler with a schedule name but no reservation icon or user name. The reservation time will be the schedule end time but of course, it could be released before then.*

Log - a  icon appears if the slot is running a test under the daemon. Clicking it will bring up the live log viewer to let you see what the test would output on the console if it were running locally instead of under the daemon. Tests created by Test Creator GUI do not output anything to the console - Python based tests provide a log to the console.

3.6.2.2 Timer Style Menu

At the top of the window is an option to show a drop down menu to change the style of time displays:

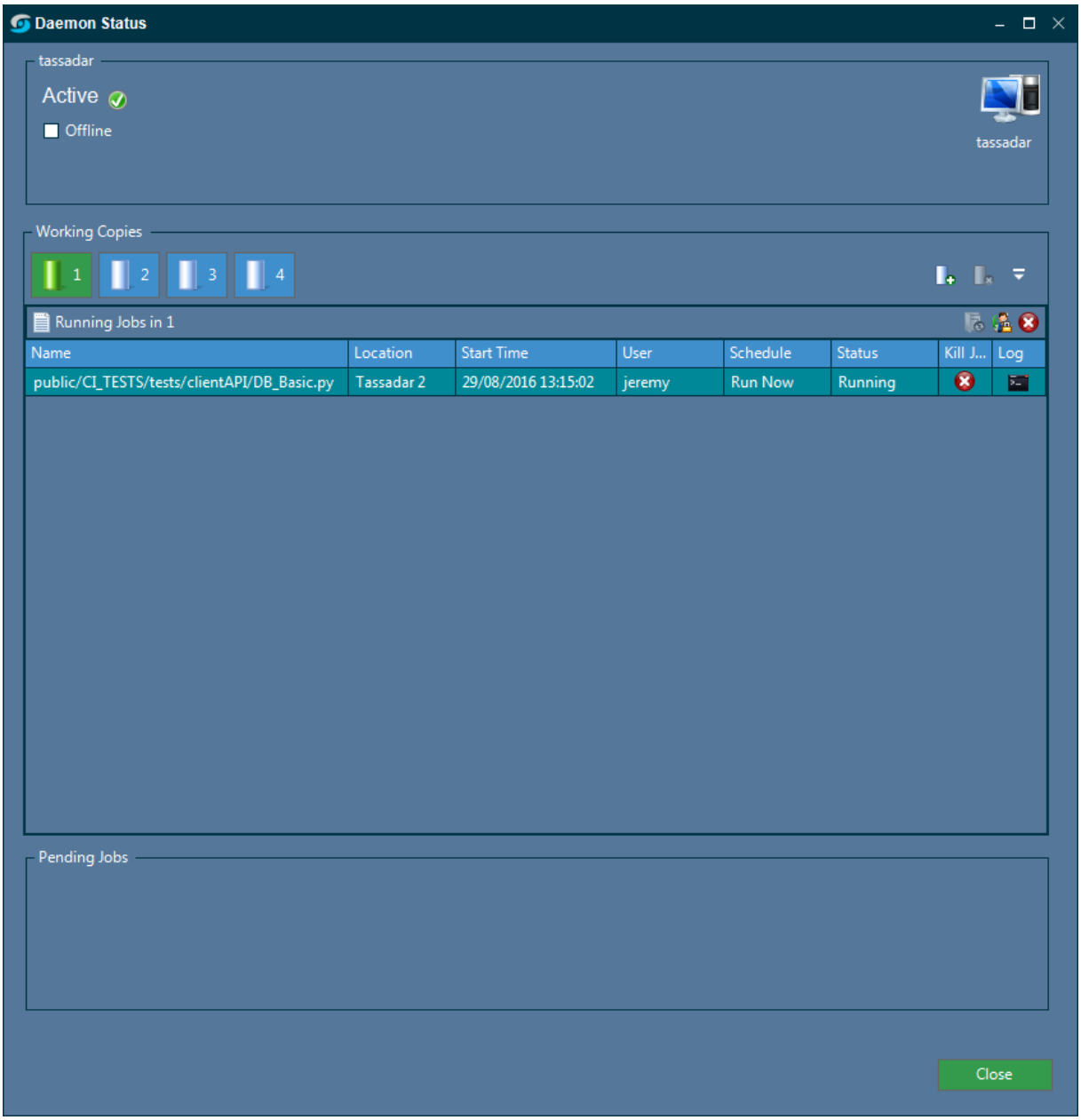


The upper pair of radio buttons determine whether the lock time column shows the absolute lock time or the duration of the lock. The lower pair of radio buttons determine whether the end reservation time is shown as a time or a countdown until the end.

Duration and countdown is shown as minutes:seconds unless it exceeds an hour and then the hours are shown. Likewise if the duration or countdown exceeds 1 day, the number of days are shown. The absolute times are shown with a date if the date is different from the current date.

3.6.3 Daemon Details

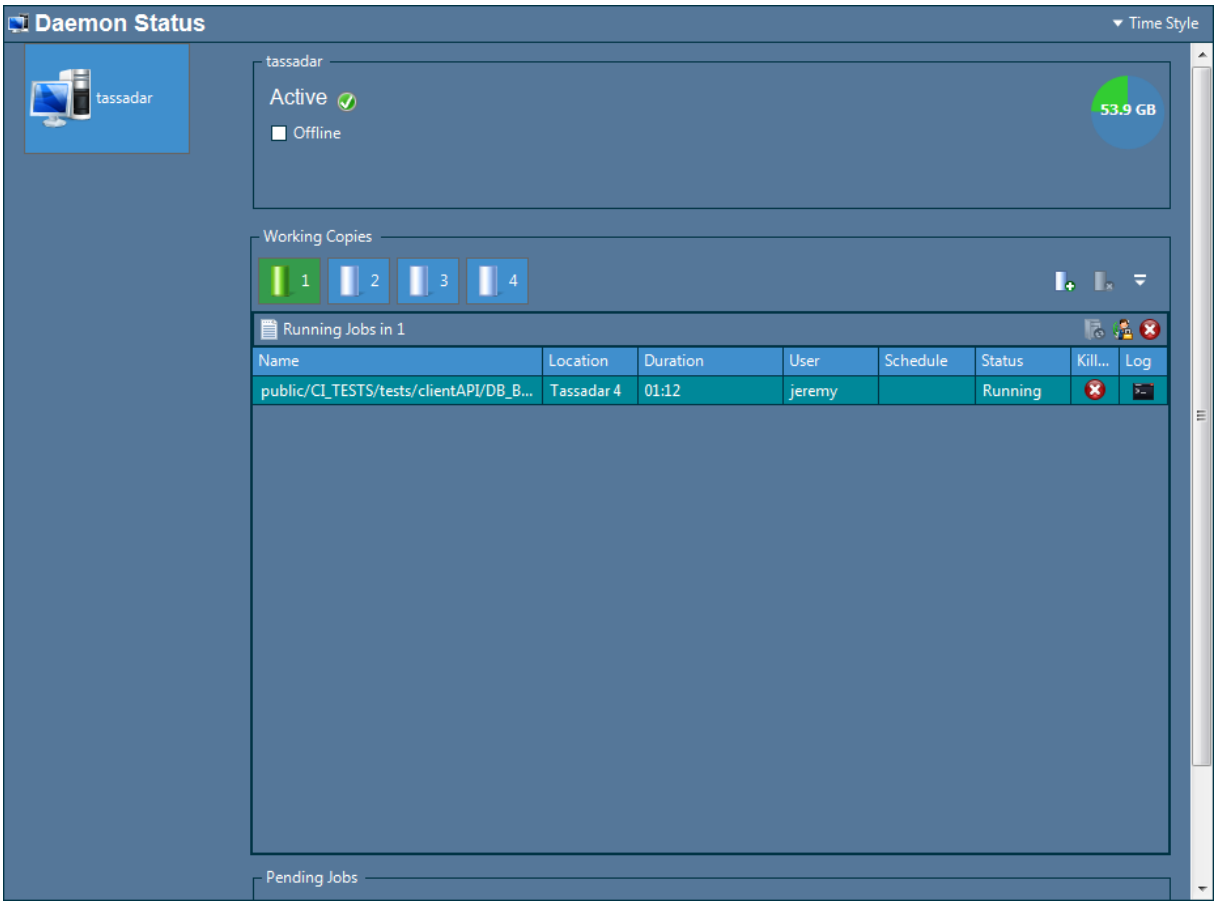
From the current reservations screen, using the Kill job menu can bring you to the daemon status of the daemon that is running a scheduled job.



This has the same controls as the main daemon status page in a dialog. The difference is that it applies to the daemon that is running the selected job.

3.6.4 Daemon Status

The Daemon Status screen shows you the status of the client daemons in the system. This helps explain what scheduled jobs are running, and if some expected job is not running, you can see why. The basic screen is:



3.6.4.1 Client Daemon Operation

The client daemon runs your jobs. To do this it has a copy of the files that you uploaded using the test scripts view. This is called a 'working copy'. This is updated automatically by the StormTest client daemon after every change that any user makes in the test scripts view. While updating, it is not possible to run any tests. So, if a test is running the client daemon has to wait until all tests stop, then update the working copy and then start new tests.

It is not possible for the daemon to know if the update is significant or not. Nor can it tell if a particular file might be needed by a script. In versions prior to 2.7, there was just one working copy so if a long running job was running and a user updated a file, it could be a very long time until further jobs could run.

Release 2.7 changed this to use 4 working copies by default. Now when a script is uploaded, the working copies without jobs running can be updated. As jobs are run and scripts updated, the daemon can allocate jobs to other working copies allowing much more efficient use of the client daemon.

3.6.4.2 Usage

Select a client daemon from the left column by clicking it. The main grid is then updated with the detailed status of the selected daemon. The icons on the left show the overall status of the daemon with more detail in the main panel.

3.6.4.2.1 Daemon Status

Each daemon can be Out of Date, Inactive, Offline, Warning, Waiting, Idle or Active.

3.6.4.2.2 Offline Mode

Once you have selected a daemon, you can set it off line or on line. Each daemon starts on line. But if, for some reason, you don't want a specific daemon to take any more jobs, you can set it off line. Existing jobs will continue but it will not accept new jobs. Simply click the 'Offline' check box.

3.6.4.2.3 Disk Space

The image on the right shows the disk usage on the disk with the test log files. Used space is in blue. Free space is green if the amount of free space is above the limit where the daemon stops accepting jobs (10 GB by default). It turns red on an error. The amount of free space is shown in the image.

If the amount of free space drops to a critical level, a button appears allowing you to clear some test log files. If the disk is full because of files other than test logs, then that button will not delete any files other than test logs.

The file clean up runs in the background and running jobs will continue without being affected. Normally the daemon will delete log files older than 21 days. When you request a disk clean up, the daemon will delete files older than 20 days and then check if there is enough free space. "Enough" means twice the minimum set or 50GB whichever is lower. So by default, it looks for 20GB. If the first delete was not sufficient, it deletes those older than 19 days. This process continues until enough free space exists or all log files have been deleted.

If the retention time is changed then the process starts at then retention time minus 1 day from the knowledge that files older than the retention time will have been deleted under normal clean up process.

3.6.4.2.4 Working Copy Status and control

The status of the working copy is shown by icons:

The working copy is up to date but not being used to run jobs.

The working copy is updating. It cannot be used to run jobs.

The working copy is running 1 or more jobs.

The working copy has an error. The daemon will delete the full working copy and attempt to rebuild it. This may take some time.

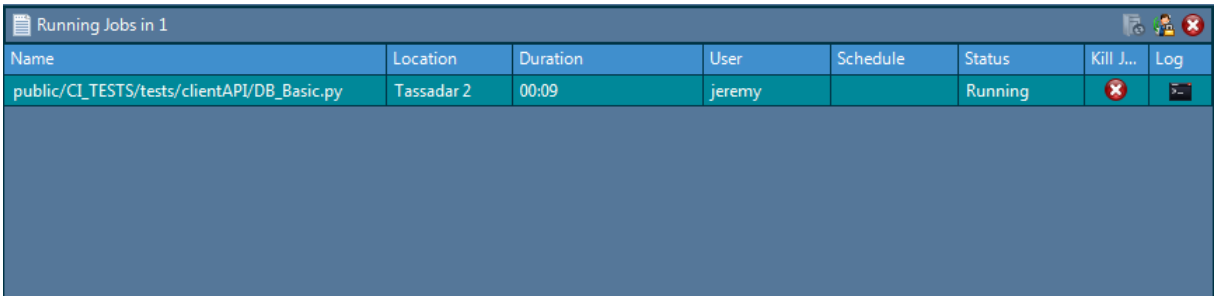
If you find that the daemon is often waiting with all working copies busy running jobs and updates are pending, you can increase the number of working copies. This may be needed in an

environment where the work flow consists of a sequence: run job, update, run job, update in quick succession and the jobs are relatively long running and all updates are to public folders.

The  button will create another working copy. Each working copy consumes disk space so you should only do this if needed.

If you find you don't need all the working copies and need to free up disk space, you can select the highest numbered working copy and click the  button. If the working copy is busy, you can't delete it. But then, if it is busy, maybe you need all the working copies so should not delete it.

Beneath the icons is a grid showing you which job(s) are running on the selected working copy:



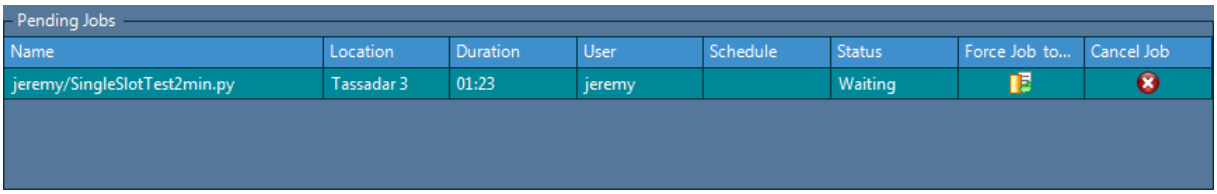
This shows the path to the script, the server/slot in use, how long the test has been running, the user who owns the scheduler, the schedule name and the status of the script. Thekill button allows you to kill the test if you wish the  icon appears if the slot is running a test under the daemon. Clicking it will bring up the live log viewer to let you see what the test would output on the console if it were running locally instead of under the daemon. Tests created by Test Creator GUI do not output anything to the console - Python based tests provide a log to the console.

You can filter the list to show just your running jobs by clicking the  icon. If the working copy is in use and there are updates pending you can use the  icon. A dialog appears showing you the list of pending updates.

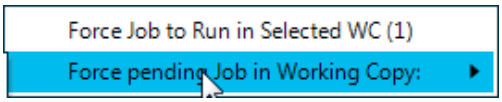
3.6.4.2.5 Pending Jobs

When all the working copies are updating or busy and have updates are pending, jobs go into the pending jobs queue. When a working copy is free to take jobs, they are allocated to a working copy and start executing. On occasions, you may find that all the working copies are in use but you know that it is safe to run a job on an out of date working copy.

Consider the case that you already have a job running on working copy 1 and somebody else updates a public folder which you have no interest in. And then you start a job which is now pending. You could usefully run that on working copy 1 - you know the updates are not relevant to you (the StormTest Client daemon does not know that).



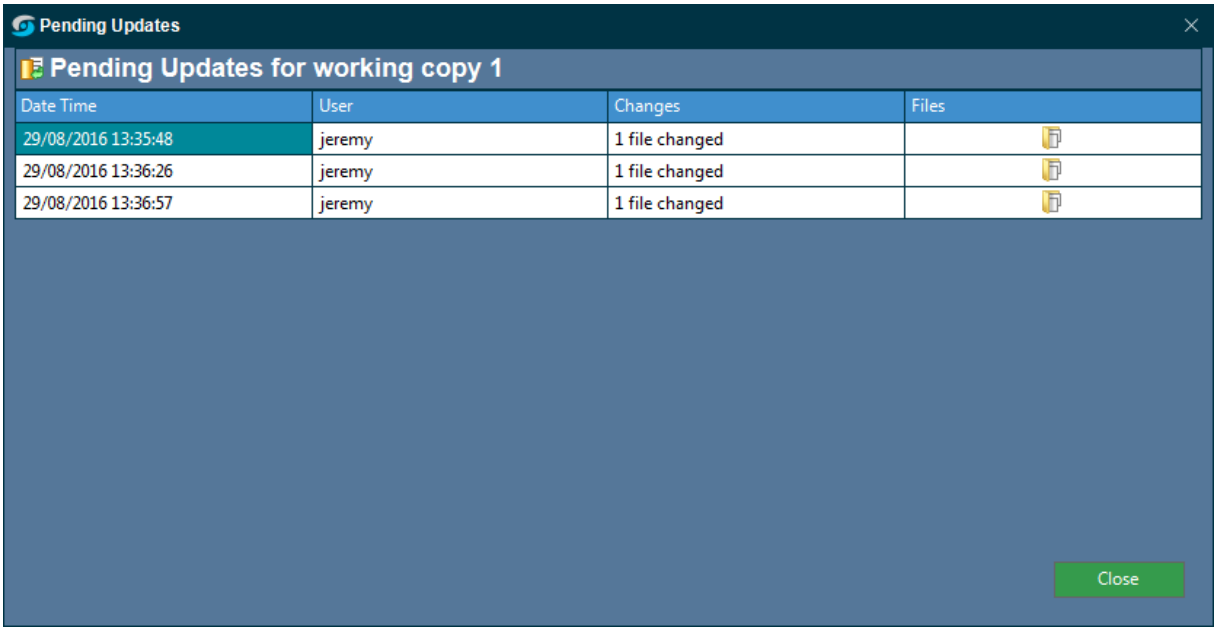
In this case you can click the icon to bring up a menu:



Here you can select the currently visible working copy or force the job to any working copy of your choice. You can only select working copies that are not updating. The job will then run on the selected working copy.

3.6.5 Pending Updates

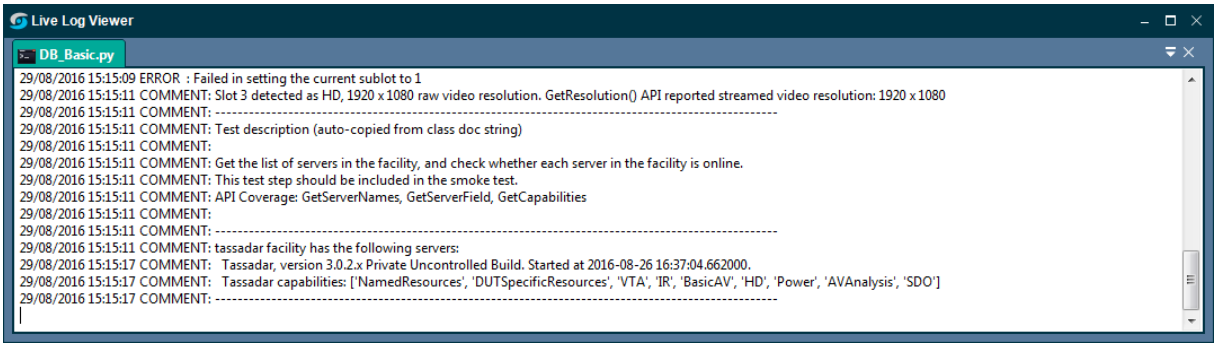
This dialog shows the updates to files that are pending on a specific working copy.



It has one line per update that is pending. It shows the time of the update, the user who initiated the update and a summary of the update (number of files added, changed, deleted, renamed) and an icon to show details. Clicking the icon in the files column shows the list of actual files in a message box. It can take some seconds to retrieve the detailed list - it is a time consuming operation. Usually the user, time and summary is enough to tell you how important the update was.

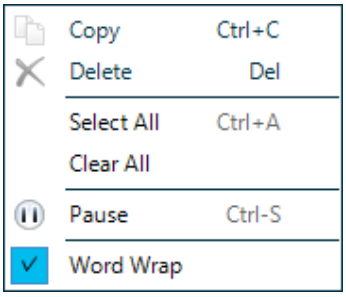
3.6.6 Live Log View

The live log view can be invoked from the unscheduled tests, daemon status view or the DUT allocations view by clicking on the console icon. It shows the output of the running test. This is available for Python scripts running under the daemon (not locally on your hard drive). You will see the last 500 log lines from the test. Lines produced by the running test are added to the bottom of the view and older lines are removed.



3.6.6.1 Usage

The right mouse menu provides control, along with keyboard shortcuts.



Copy: Copy selected text to the clipboard. If you are getting a lot of log lines, you should pause the log and then select the part of the log to copy.

Delete: Delete selected text from the live log view. This **does not** delete the text from the test's log file, only the view on the screen.

Select All: Select all the text in the log view

Clear All: Clear all the text in the log view. This **does not** change the test's log file, only the viewer. This is useful to view newly added lines.

Pause: Pauses the addition of lines to the view. This is useful if the test is producing a lot of logging and you wish to scroll backwards to view something. Use Ctrl+Q to resume (or the menu). Resume will cause lines received but not displayed to be added immediately. This **does not** interfere with the running test.

Word Wrap: Allows you to decide how you wish to view log. When off, all time stamps align on the left and to see the right hand side of long lines requires scrolling.

3.6.6.1.1 Multiple Logs

You can view multiple logs. By default, other logs appear as tabs within the live log viewer. However, you can drag off a log to float independently as well as docking them in any layout that helps you view multiple logs.

3.6.6.1.2 Tab icons

While running, the tab shows the status of the test:

 Running normally.

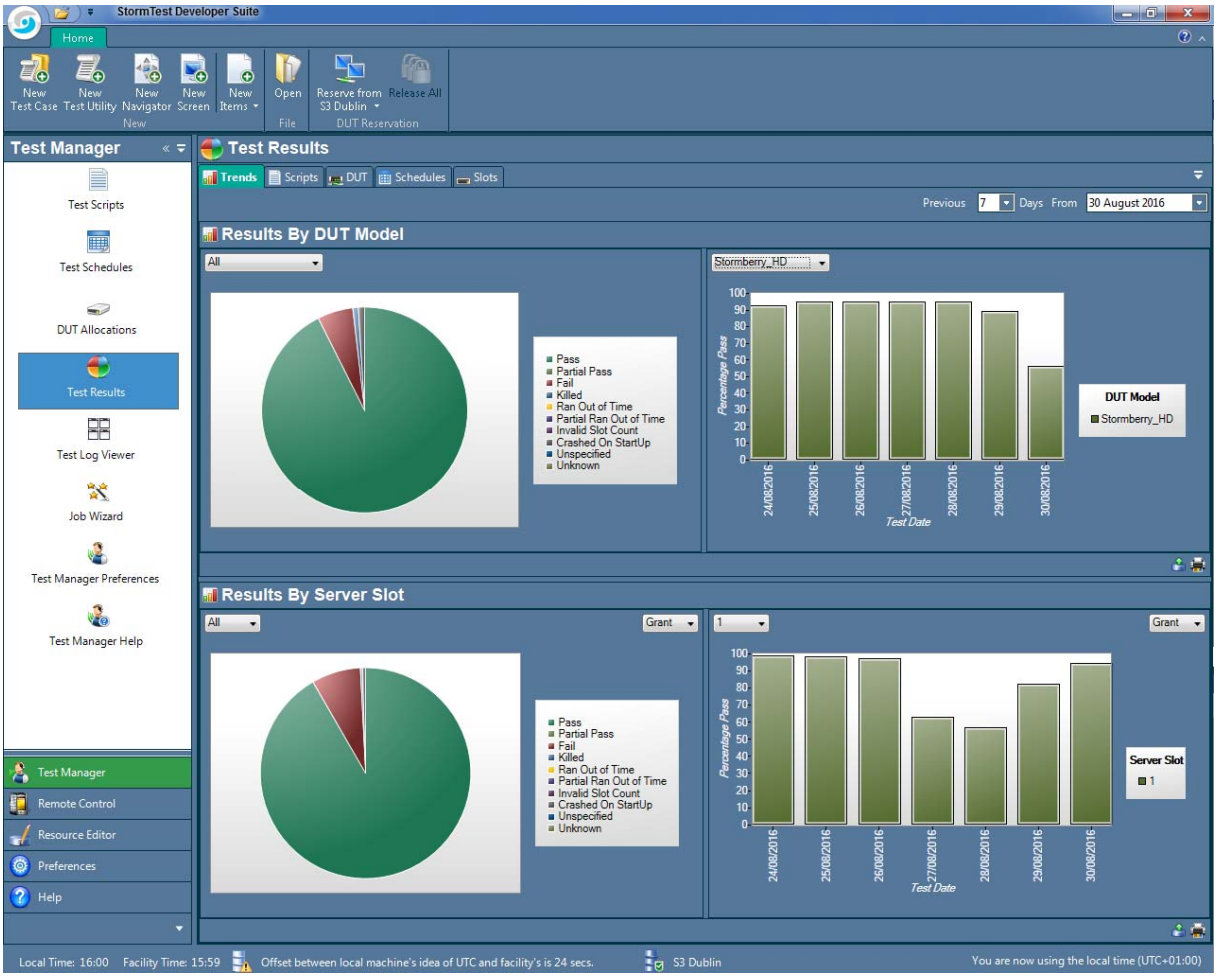
 The log has been paused and no new lines are being added. Use right menu or Ctrl+Q to resume display of the log.

 The test has finished. No more lines will be added to the log.

3.7 Results

3.7.1 Test Results View

The test results view is the view where you can see the results of your tests. You can also view the results of tests by other users. All test results that are in the StormTest Development Center database are available for viewing - the only results not stored in the database are those run directly from the command line. Even then, the script needs to avoid calling ReturnTestResult() or has to call ReturnTestResult() explicitly requesting that the result not be stored.



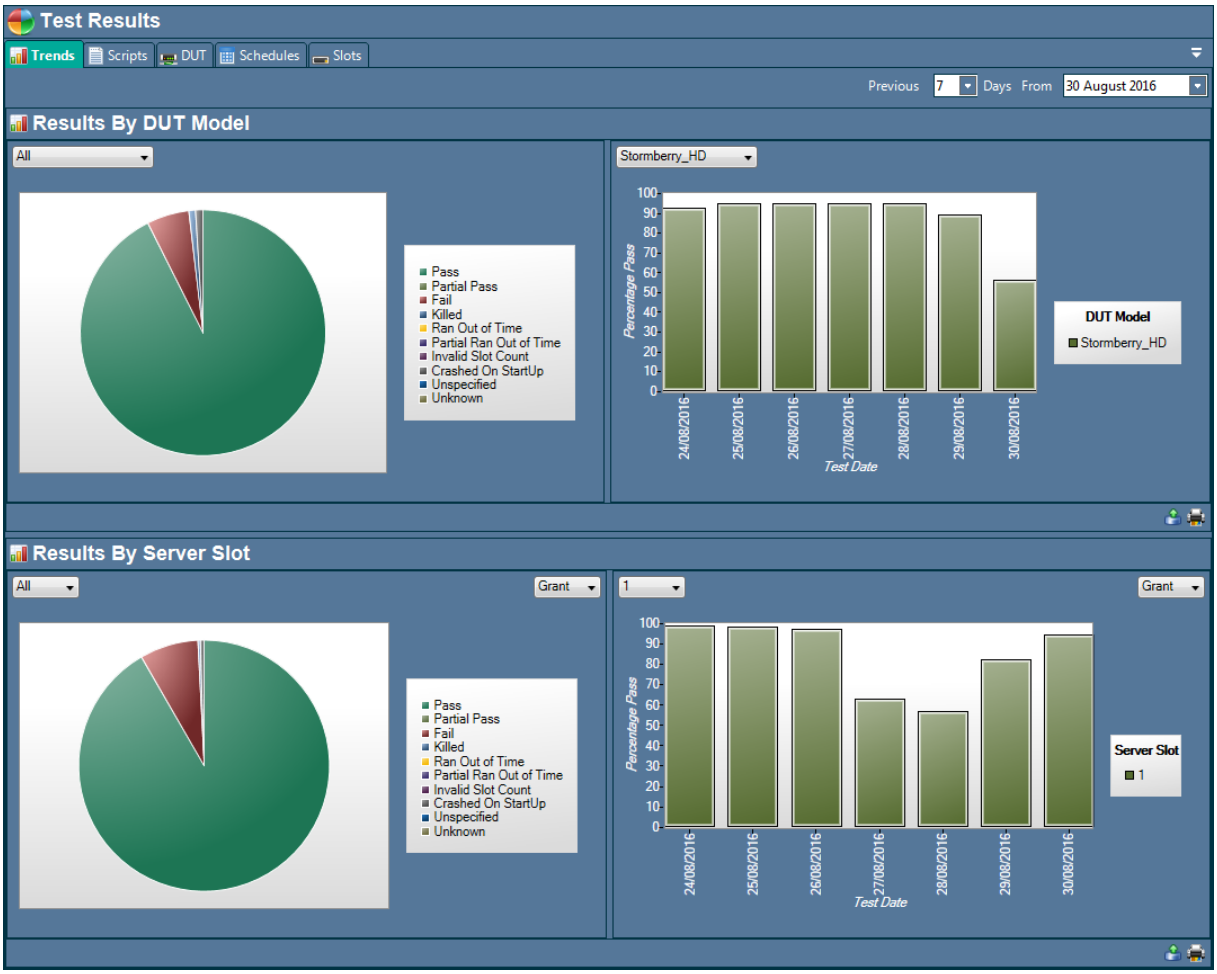
There are 5 tabs to allow the results to be viewed using different criteria:

- Long Term trends
- Arranged by script
- Arranged by DUT
- Arranged by Schedule
- Arranged by Slot

Each view allows you to select the time range to view. This is done by selecting a date and a time span. The results are displayed backward from the selected date. If you are on a slow network and select a large time period (for example, 1 year) then it may take some time to gather all the results. During this time, the StormTest logo will spin - you may use other views in Developer Suite but not the results during that time.

3.7.2 Result Trends

The result trends shows you a high level view of the results of all users and all types of tests. There are two panels each divided into a summary pie chart and a more detailed bar chart.

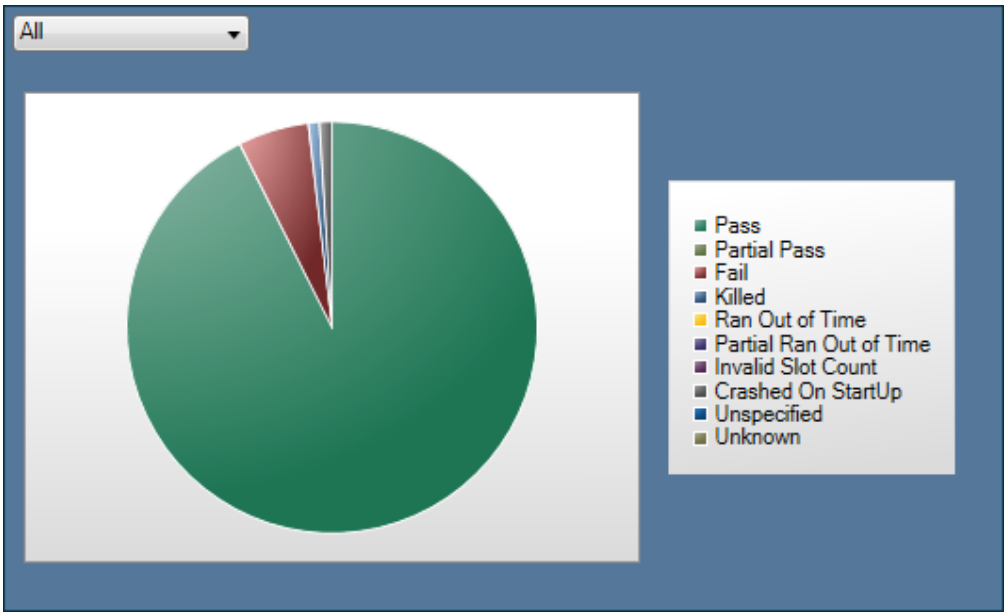


The upper panel shows the trend by DUT model. The lower panel shows the trend by physical slot.

3.7.2.1 Usage

Select an end date and time span using the control in upper right corner. The graphs will update.

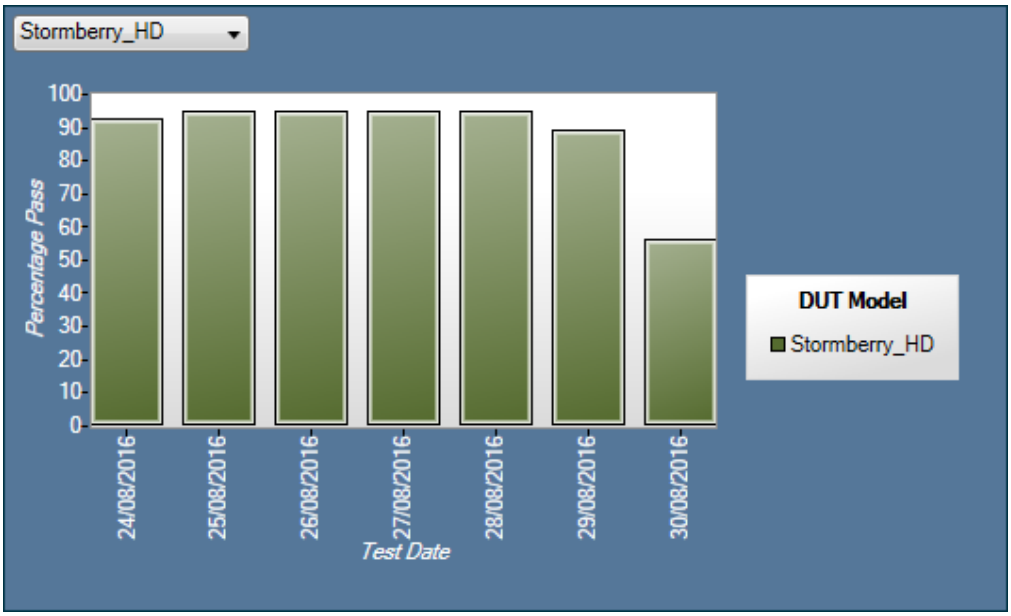
3.7.2.1.1 Pie Chart



The pie chart show the proportion of results of each type (above by box type). Hovering over a colored part of the pie chart will bring up a tooltip with more information:

The combo box in the pie chart allows you to filter the view. In the upper panel you can select a box model whereas the lower panel has two boxes: one to select the physical server and then another to select the slot of interest.

3.7.2.1.2 Bar Chart



The bar chart shows the results per day. The % of tests that pass is shown. The physical size is always the same but the scale may change - in the above example, the bottom point is 40% pass so on all days at least 40% passed. As with the pie chart, a tooltip gives more information. In an analogous manner to the pie chart, combo boxes are present to select box, server and slot as appropriate.

3.7.2.1.3 Print and Export

 Next to each chart there is an export button. Clicking this brings up a dialog to allow you to configure some of the export parameters. The file that is exported is XML and has a schema suitable for importing into a tool such as Microsoft Excel for you to further process the data. The file contains a lot of information so may take some time to download.

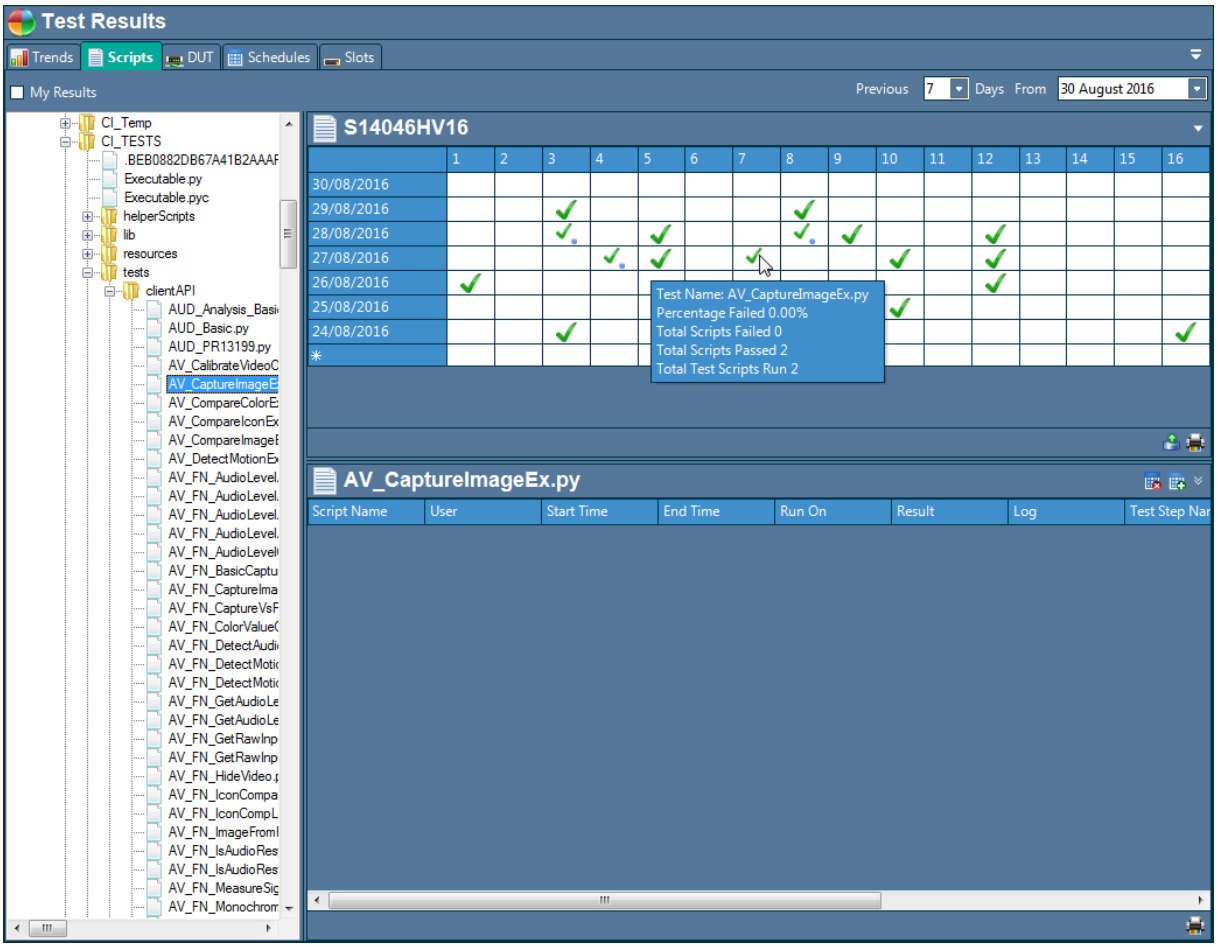
 Next to each chart there is a print button. Clicking this brings up a menu allowing you to select either print preview or print. The pie and bar charts are printed on the same page. Developer Suite uses the normal Windows print mechanism so that you can select any printer, including for instance, a PDF writer.

See Also:

- Test Result Codes

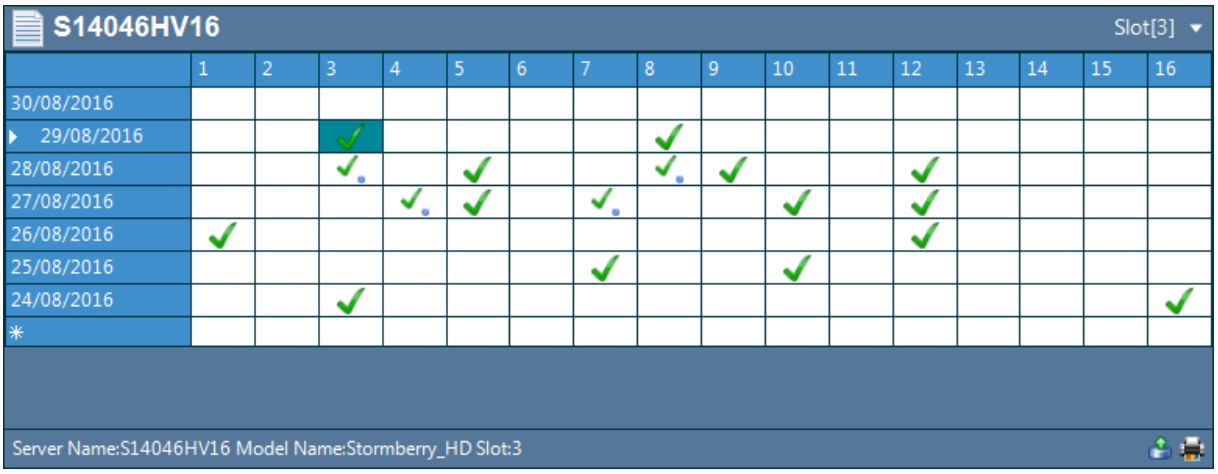
3.7.3 Results by Script

The results by script page arranges the results based on the name of the test script.



3.7.3.1 Usage

Select an end date and time span using the control in upper right corner. Check the 'My Results' box to limit results to tests ran by you, when unchecked tests run by all users will be shown. In the left panel are all the scripts that you may view. These are the public scripts plus your private scripts. Selecting a script retrieves results for that script. These are shown in the upper right panel:



The horizontal axis is slot number with the ability to select the server using the arrow on right hand side. The vertical axis is date. You can click on the date column header above the dates to change the order of dates. In each intersection point is an icon representing the results for that day on the corresponding slot. A tooltip gives more information.

Clicking on an icon will populate the lower details view. The details view is the same among all the results pages (except for the trend view where there is no details panel)

 Click on the export button to bring up a dialog to allow you to configure some of the export parameters. The file that is exported is XML and has a schema suitable for importing into a tool such as Microsoft Excel for you to further process the data.

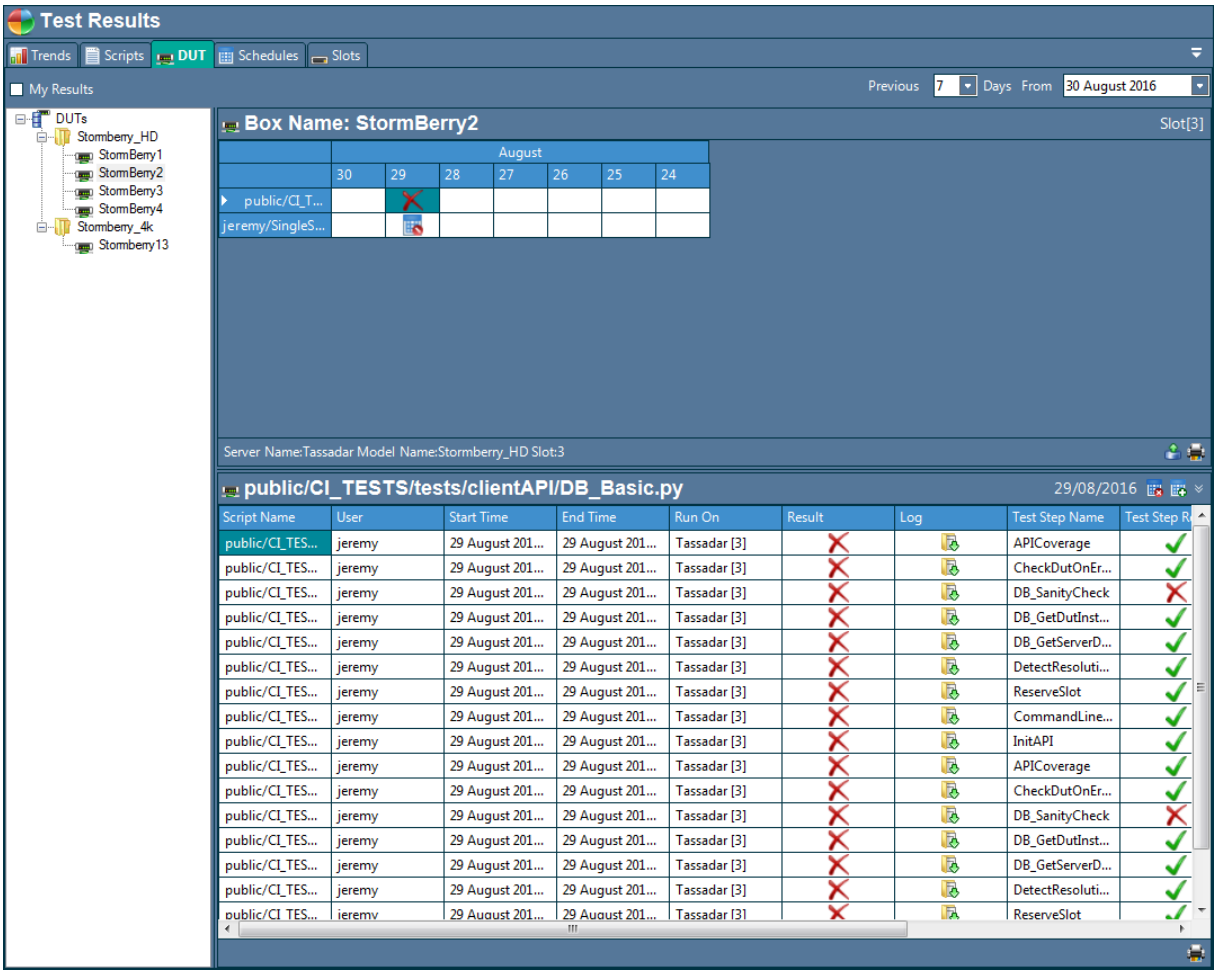
 Click on the print button to bring up a menu allowing you to select either print preview or print. Developer Suite uses the normal Windows print mechanism so that you can select any printer, including for instance, a PDF writer.

See Also:

- Test Result Icons
- Details View

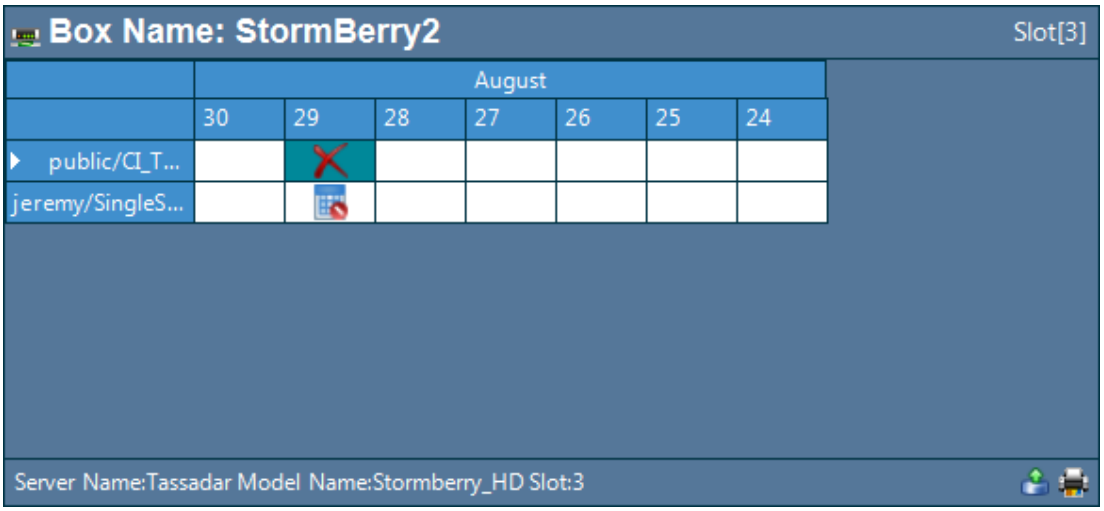
3.7.4 Results by DUT

The results by DUT page arranges the results based on the individual DUT.



3.7.4.1 Usage

Select an end date and time span using the control in upper right corner. Check the 'My Results' box to limit results to tests ran by you, when unchecked tests run by all users will be shown. In the left panel are all the DUTs in the facility arranged in folders by model. Selecting a DUT retrieves results for that DUT. These are shown in the upper right panel:



The horizontal axis is the date. The vertical axis is the script path. You can click on the script column header to change the order of scripts and also each date header to group the results by the type of result. The list of scripts will include the scripts run interactively from the command line - in that case the start of the script name will be the machine name followed by the local path. In each intersection point is an icon representing the results for that day on the corresponding slot. A tooltip gives more information.

Clicking on an icon will populate the lower details view. The details view is the same among all the results pages (except for the trend view where there is no details panel)

 Click on the export button to bring up a dialog to allow you to configure some of the export parameters. The file that is exported is XML and has a schema suitable for importing into a tool such as Microsoft Excel for you to further process the data.

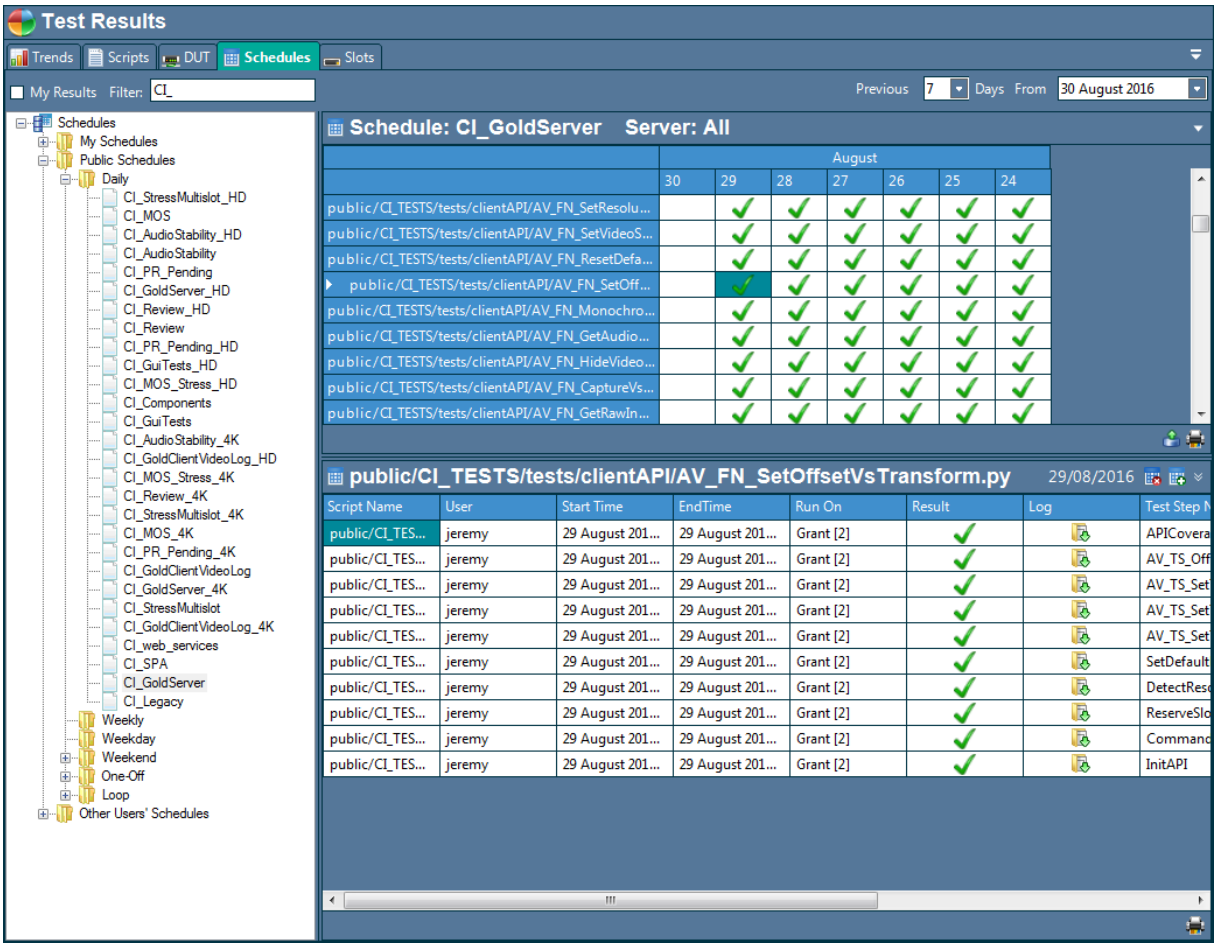
 Click on the print button to bring up a menu allowing you to select either print preview or print. Developer Suite uses the normal Windows print mechanism so that you can select any printer, including for instance, a PDF writer.

See Also:

- Test Result Icons
- Details View

3.7.5 Results by Schedule

The results by Schedule page arranges the results based on the schedule:

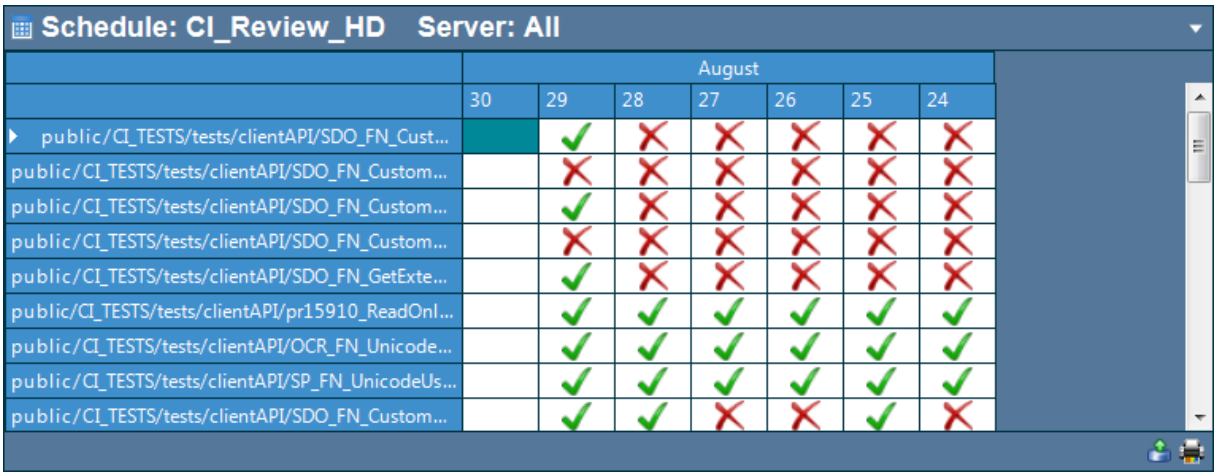


3.7.5.1 Usage

Select an end date and time span using the control in upper right corner. Check the 'My Results' box to limit results to tests ran by you, when unchecked tests run by all users will be shown. In the left panel are all the schedules in the facility arranged in folders by user name, type and time.

You can filter the view of schedules using the filter box next above the view of schedules. The filtering is not case sensitive and shows all schedules (yours, public and other users) where the specified text is in the schedule name.

Selecting a schedule retrieves results for that schedule. These are shown in the upper right panel:



The horizontal axis is the date. The vertical axis is the script path. You can click on the script column header to change the order of scripts and also each date header to group the results by the type of result. In each intersection point is an icon representing the results for that day on the corresponding slot. A tooltip gives more information. By default, results across all servers are shown. However, by clicking the arrow on the right hand side, you can filter results by a specific server.

This view is the only view where you can see what did not run. As an example, if you had a schedule which never got scheduled (due to other users, for example), there will be no results by DUT, Slot or script. However, the schedule view here will show failures of scripts which should have run on the schedule but did not - this is the purpose of the 'all' servers option - in the graphic above, all the failures were due to such a problem.

Clicking on an icon will populate the lower details view. The details view is the same among all the results pages (except for the trend view where there is no details panel)

 Click on the export button to bring up a dialog to allow you to configure some of the export parameters. The file that is exported is XML and has a schema suitable for importing into a tool such as Microsoft Excel for you to further process the data.

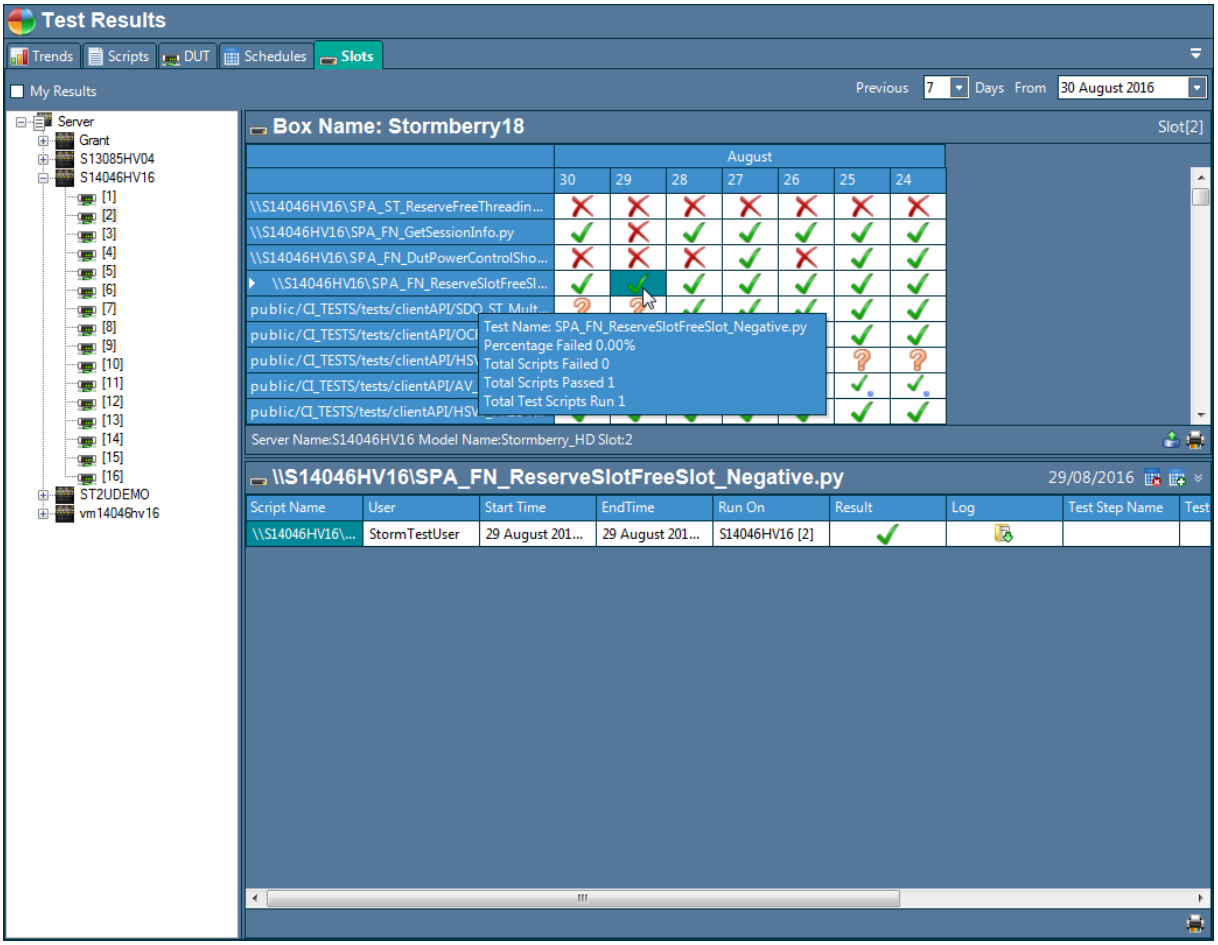
 Click on the print button to bring up a menu allowing you to select either print preview or print. Developer Suite uses the normal Windows print mechanism so that you can select any printer, including for instance, a PDF writer.

See Also:

- Test Result Icons
- Details View

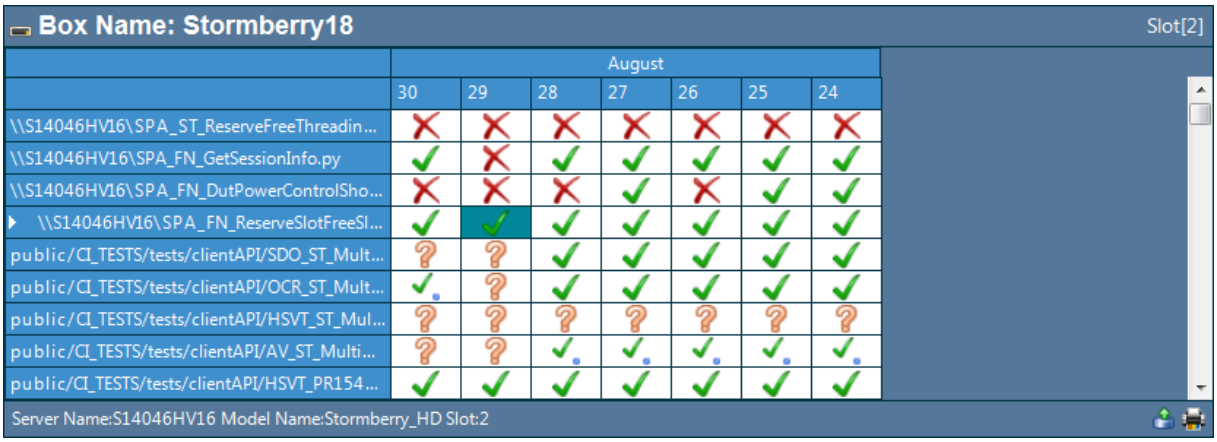
3.7.6 Results by Slot

The results by slot page arranges the results based on the physical slot:



3.7.6.1 Usage

Select an end date and time span using the control in upper right corner. Check the 'My Results' box to limit results to tests ran by you, when unchecked tests run by all users will be shown. In the left panel are all the slots the facility arranged in folders by server. Selecting a slot retrieves results for that slot. These are shown in the upper right panel:



The horizontal axis is the date. The vertical axis is the script path. You can click on the script column header to change the order of scripts and also each date header to group the results by the type of result. The list of scripts will include the scripts run interactively from the command line - in that case the start of the script name will be the machine name followed by the local path. In each intersection point is an icon representing the results for that day on the corresponding slot. A tooltip gives more information.

Clicking on an icon will populate the lower details view. The details view is the same among all the results pages (except for the trend view where there is no details panel)

 Click on the export button to bring up a dialog to allow you to configure some of the export parameters. The file that is exported is XML and has a schema suitable for importing into a tool such as Microsoft Excel for you to further process the data.

 Click on the print button to bring up a menu allowing you to select either print preview or print. Developer Suite uses the normal Windows print mechanism so that you can select any printer, including for instance, a PDF writer.

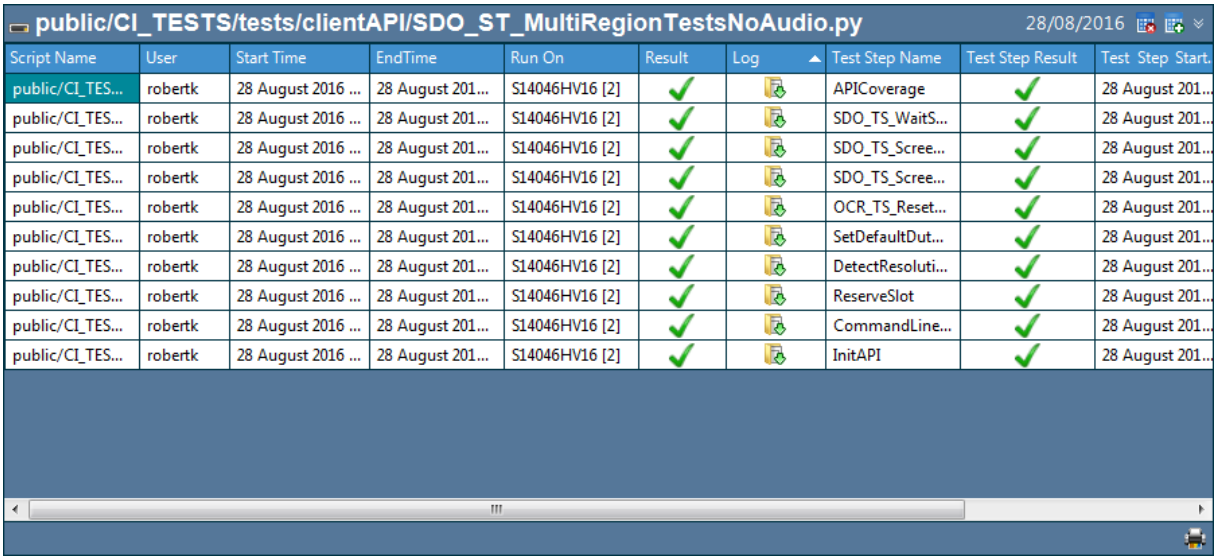
See Also:

- Test Result Icons
- Details View

3.7.7 Result Details

In each of the results pages (except for the trends page) is a result details panel. It is below the summary for the page and to the right of the selection tree. It is populated when you select a result

icon in the summary. Since the summary view groups results by day and there could be more than one result per day (for the selected combination of script, slot or DUT), this panel gives the detailed results.



Script Name	User	Start Time	EndTime	Run On	Result	Log	Test Step Name	Test Step Result	Test Step Start
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		APICoverage	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		SDO_TS_WaitS...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		SDO_TS_Scree...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		SDO_TS_Scree...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		OCR_TS_Reset...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		SetDefaultDut...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		DetectResoluti...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		ReserveSlot	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		CommandLine...	✓	28 August 201...
public/CI_TES...	robertk	28 August 2016 ...	28 August 201...	S14046HV16 [2]	✓		InitAPI	✓	28 August 201...

Most columns can be sorted by clicking on the column header. The columns can be resized by clicking the divider and dragging it. There may be more than one row for a given test if the test used the concept of test steps (by calling the BeginTestStep() and EndTestStep() API calls). The columns available are:

Script Name - the full path to the test script that generated the result

User - the name of the user who ran the test

Start Time - the time the script started

End Time - the time the script ended

Run On - where the script ran in the form of 'server name [slot number]'

Result - an icon showing the result. A tooltip will give a textual explanation of the result. It is limited to the pass and fail icons (fail being all non pass statuses)

Log - a link to the log file. Developer Suite will open the built in Log Viewer and start to play back your test. If you right click, you will see a menu to view the log files. If you click that, Developer Suite will open your default FTP viewer and show you the raw files - you can then download them. If the test was run locally on your PC, then Windows Explorer will be opened instead of an FTP client.

Test Step Name - name of test step (blank if the test does not use test steps)

Test Step Result - numeric result of test step. Since the codes are totally user defined, there is no way to indicate pass or fail

Test Step Start Time - the time the test step started

Test Step End Time - the time the test step ended

Test Step Comment - the comment that the script used when ending the test step.

 The remove from schedule button allows you to remove the selected script from whatever schedule it was on. If it was not on a schedule then there is no action. You can only remove tests from your schedules or from schedules marked as public.

 The move to schedule button allows you to remove the selected script from whatever schedule it was on and place it on another schedule. If it was not on a schedule then there is no action. You can only move from your schedules to public schedules and only add to your own schedules or public schedules. A dialog will appear allowing you to choose the schedule.

3.7.8 Test Result Codes

Within the StormTest Development Center database, each test script returns a single numeric test result. The final result can come from several sources:

- The script itself via the ReturnTestResult() API call
- From the Client Daemon when the test exits (if there is an error or ReturnTestResult() is not used, the Windows Exit code is used)
- From the Scheduler if the test could not be scheduled for some reason

These numeric codes are interpreted into human readable text when displayed in the Results Trend view or in tooltips in other Results Views. When exported as XML, the numeric codes are used. The possible codes, and meaning are:

Result Code	Text Description	Detailed Interpretation
0	Unspecified	The script did not call ReturnTestResult() but otherwise exhibited no problems
1	Crashed on StartUp	The script crashed upon start under the client daemon - perhaps the schedule contained a reference to a script that had been deleted. Or the script just has a bug.
10	Ran Out of Time	The script never ran on the specified schedule - the schedule ended before the scheduler could find a DUT for it. See note below on looping schedules.
11	Invalid Slot Count	The script was a 'run now' remote job and specified single process mode with more slots than were

		allocated. It did not run at all.
12	Partial Ran Out of Time	The script was a multi-process script scheduled and it ran on some of the requested DUTs but not all because the schedule ended its allocated time. If the schedule was allocated, for example, 4 boxes but the job required 6, this result will NOT be returned if it ran on 4 boxes. It would only be returned if it ran on 1, 2 or 3. If it ran on 0 then code 10 (Ran Out of Time is returned). See note below on looping schedules.
100	Fail	The test returned 'fail' from ReturnTestResult()
109	Unknown	The test was run using the prototype .NET Client API instead of the normal Python API and failed to call ReturnTestResult().
110	Partial Pass	The test returned 'partial pass' from ReturnTestResult()
111	Pass	The test returned 'Pass' from ReturnTestResult()
120	Killed	The test was killed, either by the user if a 'run now' job or by the scheduler if it was on a bounded schedule that ended.

NOTE: A looping schedule will often generate Ran Out of Time or Partial Ran Out of Time codes if it did not complete all iterations in the allocated time. Whether the Partial Ran Out of Time or simply Ran Out of Time is returned depends upon whether a complete number of iterations of the loop is completed. If only one test is on the script then only Ran Out of TIme is reported but if more than one script is on the schedule then a Partial Ran Out of Time is reported if, for example, 3 out of the 4 scripts completed the interrupted last loop iteration.

3.7.9 Test Results Icons

In the Results views and the detailed results views, icons are used to indicate the status of the test script result. Test steps are interpreted on the assumption that users have chosen to use the same return values as for test scripts. Hovering over the icon will produce a more detailed result code. The icons used and meanings are:


There is one test result and the status is PASS

There is more than one test result and the status of all of them is PASS

There is one test result and the status is FAIL

There is more than one test result and the status of all of them is FAIL

There is more than one test result and the results are a mixture of values so no single status can be determined

The script failed due to a scheduler error (ran out of time)

The script crashed and failed to return a valid result.

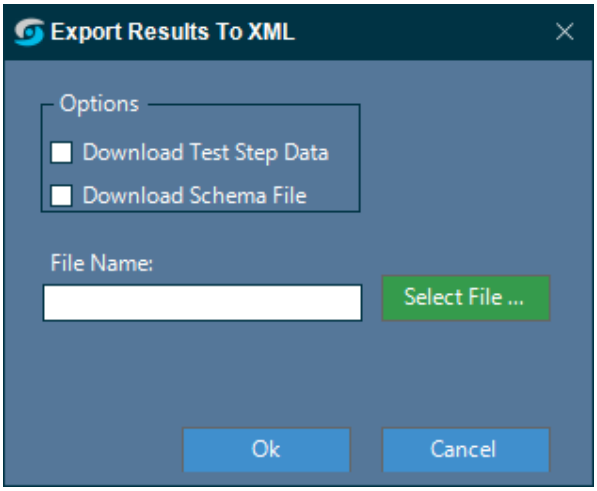
For a test step only, this means the test step used a return code unknown to Developer Suite.

There is a currently active test here (either scheduled or via the 'run now' method - interactive command line tests are not detected in the results views until they have completed)

FAIL means not PASS so all result codes other than PASS (code 111) are considered FAIL

3.7.10 Export to XML

After clicking the button to export the data, a dialog appears:



3.7.10.1 Usage

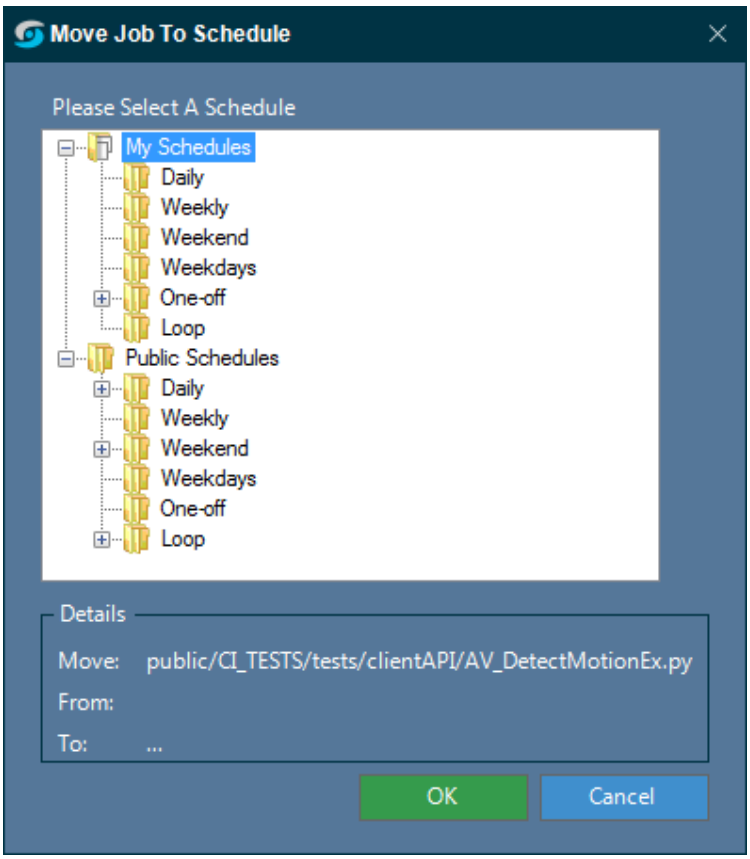
You need to select a file by clicking on the 'Select File ...' button or by typing the filename directly.

You must also select what you wish to download. To fully import the data into other tools you will need to download the schema file - this only needs to be done once as, irrespective of which data you export, the schema is the same. The schema file must be in the same directory as the full data file in order for it to be usable.

By default, Developer Suite does not download the test step results - these can be quite numerous. Checking the 'Download Test Step Data' will ensure that they are downloaded as well. Each record in the XML file will be one test step.

3.7.11 Move To Schedule

When you click the  from the detailed results view, you are presented with a dialog:



Your schedules and the public schedules are shown. Expand the folders as needed and select a schedule. The script will be copied to that schedule.

3.8 Wizard

3.8.1 Job Wizard

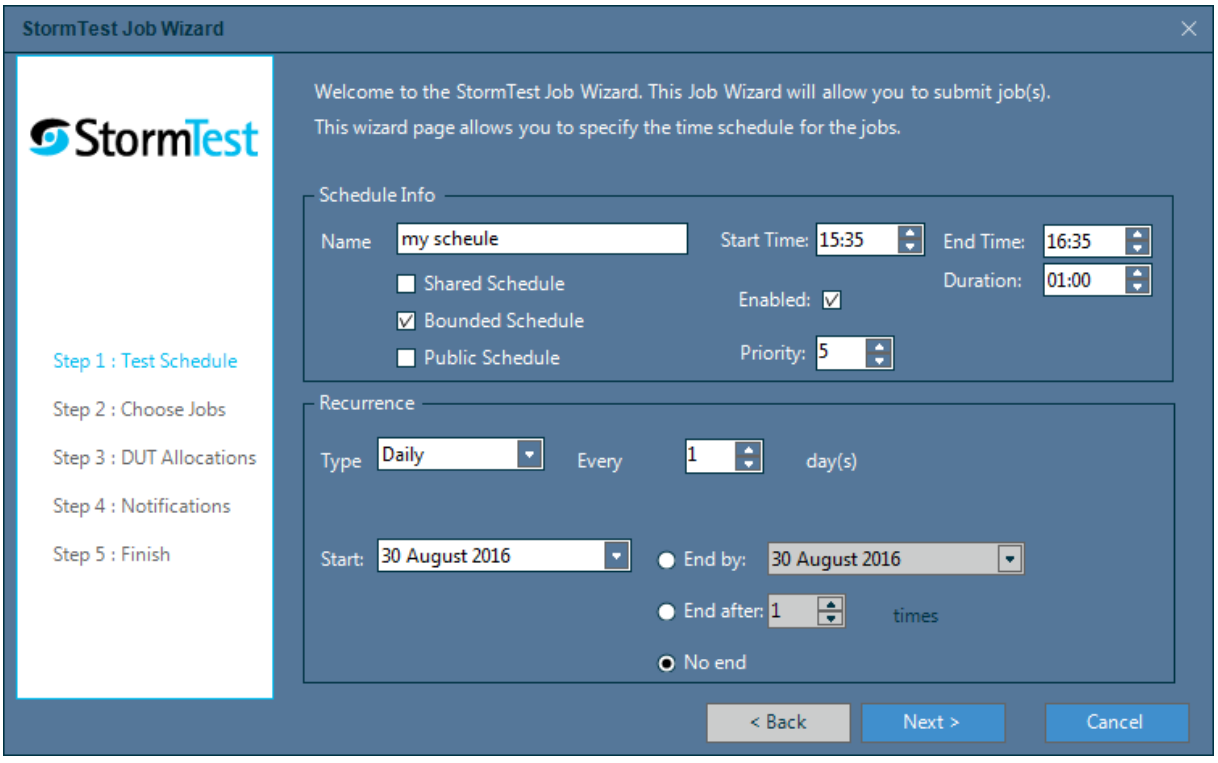
The job wizard takes you through the steps to create a schedule, add DUTs and scripts to it and submit it. It can be invoked from the test manager view or the ribbon at the top of the application. There are no features in the wizard that are not available from the individual views and after a while, you may find it quicker to use the individual schedules view. The steps of the wizard are:

1. Create a schedule
2. Add scripts to the schedule
3. Add DUTs to the schedule
4. Add email notifications to the schedule
5. Summary and finish.

The job wizard is invoked from the left hand navigator by clicking the  icon or from the New Items button of the main ribbon.

3.8.2 Job Wizard - Create Schedule

This is the 1st step of the job wizard:



You fill in the fields as needed. The recurrence panel will change slightly based on the type of schedule being created. The fields are the same as in the Schedule Details panel of the schedules view.

Name - the name of the schedule. It is only used for display purposes and has no other significance

Start Time - the time that the schedule will start on each day that it starts. It is hours:minutes in the format that is configured on your Windows machine. The schedule will always start on a whole number of minutes.

End Time - the time that the schedule will end on each day. If this is before the start time then the schedule runs over the day transition at midnight. Units are the same as for start time.

Duration - the duration of the schedule. You can use this instead of calculating the end time.

Priority - the priority of the schedule. 1 is the highest priority, 10 is the lowest.

Shared schedule - marks the schedule as a shared schedule.

Bounded Schedule - marks the schedule as bounded. This is the default.

Public Schedule - marks the schedule as public. This allows other users edit your schedule or to move tests to it.

Enabled - marks the schedule as enabled. If you disable a schedule, the scheduler will not run the schedule but it has not been deleted. This is useful for temporarily prevent schedules running. A disabled schedule is shown greyed out in the schedule list.

The lower panel controls the recurrence of the schedule. The fields that are visible depend on the type of schedule:

Type - the type of the schedule. Details are described in the scheduler features.

Start - the date on which the schedule first runs. Clicking on the arrow to the right will bring up a calendar. You can also click the individual fields of day month or year and type the desired value. Use a number for the month not the name.

Every - the is the scheduling interval. The units 'day(s)' in the above view will change as appropriate. You can set up a daily schedule to run every 2 days, for example. It is disabled for the weekday and weekend types. It is not visible for the loop or one off types

Count - the number of times to run a loop schedule.

Mon to **Sun** - the days of the week to run a weekly schedule. This is the day the schedule will start (not end).

End By - select end by and enter an end date to force the schedule to run up until a specific date. It will run for the last time on this date.

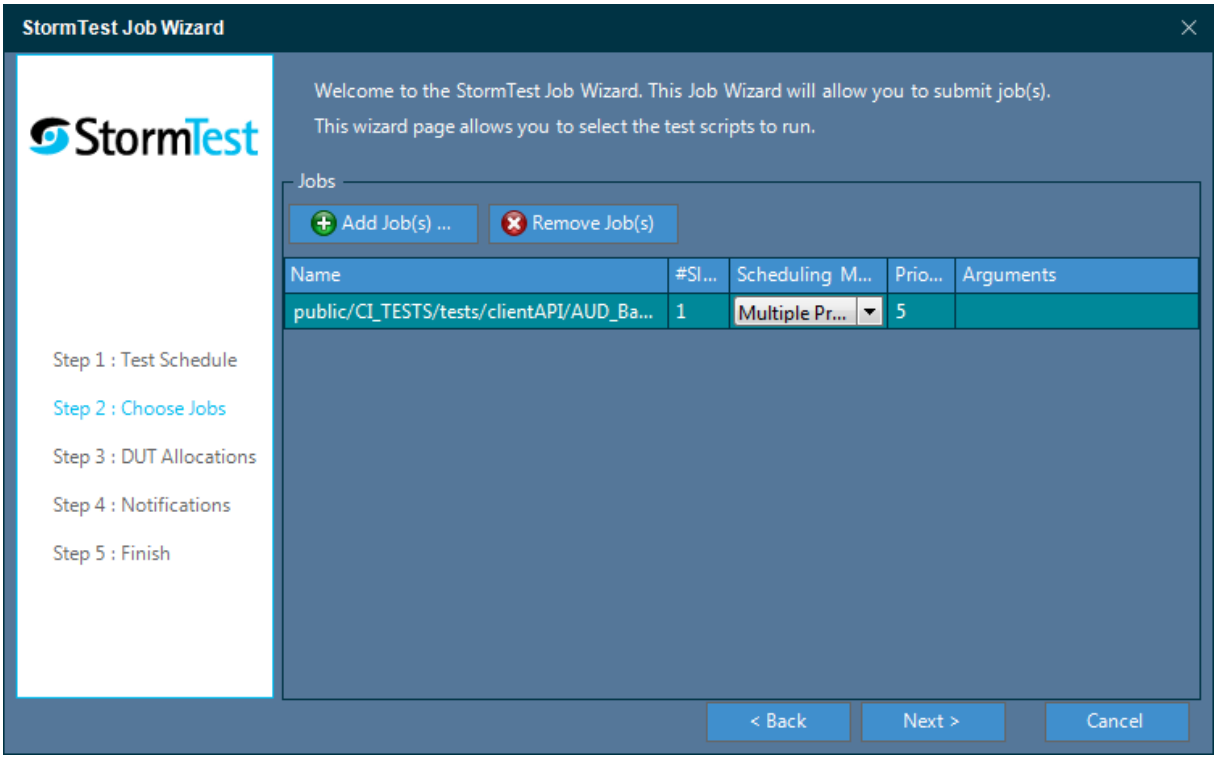
End After - select the number of times to run. This is disabled for loop and one off schedules as it makes no sense. Note that this is the number of times the schedule runs not the length. So a weekend schedule with a End After of 3, for instance will run on 3 distinct days (Saturday, Sunday and following Saturday) nor for 3 weekends.

No End - the schedule will run forever. This is not available for a one off schedule. In reality the schedule will end in January 2038 due to the internal clock used.

Run Continuously - Check this to force a loop schedule to run continuously. In this case the start time and end time are not used on a daily basis but only to determine the first and last hours of the overall run. If not checked then the schedule will run only between the start and end times on each day.

3.8.3 Job Wizard - Add Scripts

This is the 2nd step of the job wizard:



You must add at least one job before continuing. The fields are the same as the Scheduled Jobs panel of the Schedules View. You add and remove jobs by using the buttons marked Add Job(s) and Remove Job(s). Adding a job brings up the dialog to add scripts to a schedule.

The fields are:

Name - the path to the script file in the server repository. Public scripts begin with public/ while private scripts begin with your user name. It is possible to mix public and private scripts within a schedule.

#Slots - the number of DUTs that will be used by the job.

Scheduling Mode - how the job will be scheduled.

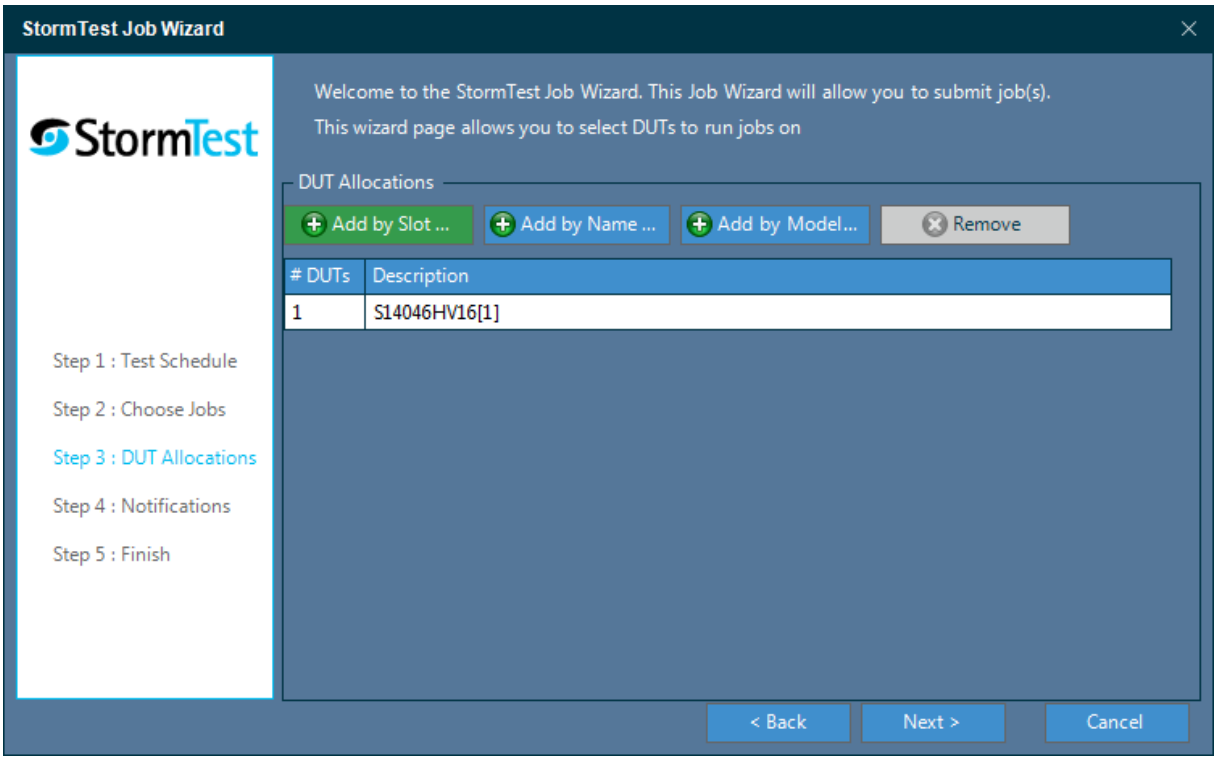
Priority - the priority of the job. 1 is highest, 10 is lowest

Arguments - optional arguments to be supplied to the running script. They must not exceed 200 characters in length in total - this is a Windows limitation.

You can modify everything except the script name after adding it to the schedule.

3.8.4 Job Wizard - Add DUTs

This is the 3rd step of the job wizard:



You must add at least one DUT before continuing. The fields are the same as the Scheduled Allocations panel of the Schedules View.

3.8.4.1 Add by Slot

Adding a DUT by slot allows you to add a physical slot to the schedule. Whichever DUT is in the slot at the time that the schedule will run is used. This does not always lead to predictable test results, however, it is the way earlier StormTest products worked so is provided for consistency. A dialog allow you to select the slot(s). DUTs added like this will appear as 'Server Name[slot number]' in the list - this is in fact as shown above.

3.8.4.2 Add by Name

Adding a DUT by name allows you to add an explicit DUT to the schedule. Just that DUT will be used. A dialog will appear allowing you to browse the available DUTs. DUTs added like this will appear as 'DUT name (model name)' in the list.

3.8.4.3 Add by Model

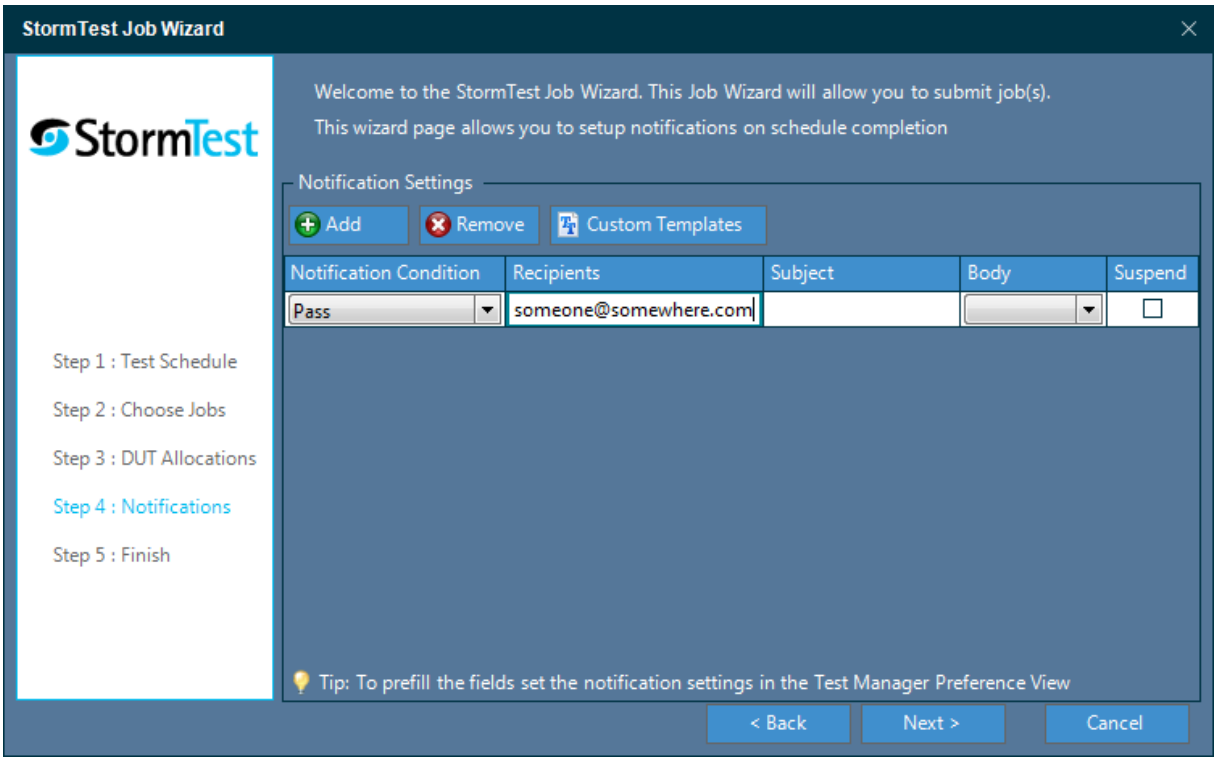
Adding a DUT by model allows you to specify a model type and a maximum number to use for your schedule. The scheduler will find the models at run time that are free and will use them. A dialog will appear showing you the models available. DUTs added like this will appear as 'Model Name (Model Type)' in the list. Also the column headed #DUTs will show how many of the models will be used.

3.8.4.4 Remove

Clicking the remove button removes the highlighted DUT from the schedule being created.

3.8.5 Job Wizard - Notifications

This is the 4th step of the wizard:



StormTest can send an email when a schedule completes. For repeating schedules such as daily schedules, an email will be sent each day. To use this feature, the StormTest administrator must have set up the email server in the Admin Console. Otherwise, the feature will be disabled in StormTest Developer Suite.

You may add any number of notifications per schedule. Each notification is considered to be independent and if you duplicate the recipients then they will get multiple emails. For each email, you set up:

Notification Condition - this is the condition to send the email. The options are:

- Pass - send an email if all tests on the schedule return a Pass Status
- Fail - send an email if any test on the schedule returns a fail, or any test fails to run
- Always - always send an email

Recipients - List of email recipients. Separate each recipient with a semicolon (;).

Subject - the subject line of the email. It may be blank.

Body - the body of the email. You can leave this blank or select from a list of pre-defined templates. You may add templates of your own by clicking the 

button. The template may also include a customized subject line - this will override any subject line you specify here.

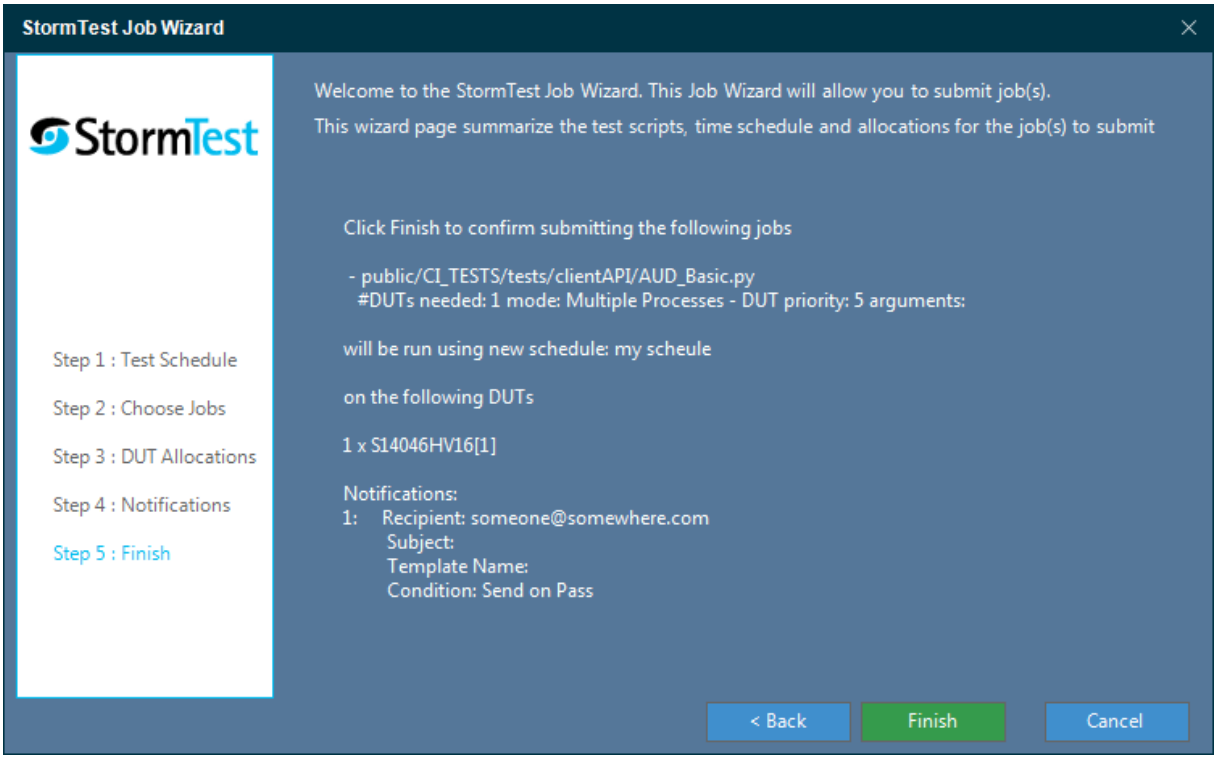
Suspend - suspend this notification. While a notification is suspended, it will not trigger an email.

3.8.5.1.1 Default notifications

If you set up values in the test manager preferences view, these will be entered into each new notification created as the default values. If you leave the subject and body blank, StormTest will use default values for both and generate a reasonably verbose email.

3.8.6 Job Wizard - Summary

This is the 5th and last step of the Job Wizard:



You see a summary of the schedule you wish to create. At this stage you may use the back button to make change, the Finish button to complete or the cancel button to fully cancel the creation.

3.9 Scheduler

3.9.1 Scheduler Features

3.9.1.1 Types of Schedule

There are a variety of types of schedule which can be defined. Each type has a pre-defined priority in that if a single user defines 2 schedules which overlap then a schedule of a higher priority will pre-empt a lower priority schedule if the schedules use the same DUT resources. The schedules are described below starting with the highest priority.

3.9.1.1.1 One-off Schedule

A one schedule defines a single continuous time span with a start date/time and an end date / time. A one-off schedule may last more than 24 hours and since it only runs once, it is the highest priority.

3.9.1.1.2 Weekly Schedule

A weekly schedule runs on a weekly basis, for between one and 7 days. However, if all 7 days are specified, a daily schedule should be used.

3.9.1.1.3 Weekday Schedule

A weekday schedule runs on the weekdays of every week. These are Monday to Friday inclusive irrespective of locale - no attempt is made to account for those countries of the world who consider the working week to be Sunday through Thursday.

3.9.1.1.4 Weekend Schedule

A weekend schedule runs on the weekends of every week. This means Saturday and Sunday irrespective of local - no attempt is made to account for those countries of the world who consider Friday and Saturday to be the non working days of a weekend. Note that a schedule running 23:00 - 01:00 at the weekend starts on Saturday and Sunday at 23:00; it is the start not the end time that is used.

3.9.1.1.5 Daily Schedule

A daily schedule runs on a daily basis, the same time each day.

3.9.1.1.6 Loop Schedule

This is the lowest priority schedule. It runs from a start date and time until an end date and time. Optionally, it can be constrained to run between specified times on each day. The schedule attempts to run the selection of tests a fixed number of times. If each test manages to complete once before the end of the day's allocated time, the first test in the sequence is run again and the time used to run as many tests as possible.

3.9.1.1.7 Remote Job Execution

The 1.2 release of StormTest introduced the concept of remote job execution. This is still available and is invoked by 'run now' from the scripts view and is treated as if it were a schedule of lowest priority and the DUT allocation is considered non exclusive. It cannot pre-empt a reserved slot nor a reserved DUT unless the slot or DUT is on a shared schedule. You can view such remote jobs by using the unscheduled tests panel of the scripts view - this includes remote jobs submitted via a Python script.

3.9.1.2 Schedule Timing

3.9.1.2.1 Start and End Dates

All schedules have a start date – this can be set in the future to set up a schedule to start, for example, in a week's time.

Each schedule may take an end date – the schedule will be stopped on that date. Alternatively, a repeat count for the schedule may be given, from which the scheduler will calculate an effective end date and use that calculated value. When an end date is specified, the schedule starts its last run on that date - it is "run up to and including" that date.

3.9.1.2.2 Schedule Duration

With the exception of a one-off schedule, all schedules are less than 24 hours per day. The end time may be set to be before the start time – in this case, the actual end time is the following day from the start. So, for example, a weekly schedule, running every Monday and starting at 21:00 and ending at 03:00 (end < start) will run each Monday, starting at 21:00 and ending at 03:00 on Tuesday.

3.9.1.2.3 Bounded Schedules

A schedule can be marked as bounded – this means that when the end time of the schedule is reached, any running tests are killed and no more are scheduled. If a schedule is not bounded, then any running tests will continue and other tests on the schedule will continue to be scheduled but subject to the priority rules.

Bounded mode is a good mode to use when unsure about the stability of tests – a badly behaved test will be stopped and not affect other users. On the other hand, once tests are stable, it can be limiting to have tests stopped due to the difficulty of estimating the time needed for a schedule.

3.9.1.2.4 Schedule Priorities

Each schedule can be given a priority between 1 and 10. 1 is the highest priority. This allows a priority to be given between schedules of the same type only – so if you have several weekly schedules which may overlap, the priority can be used to determine which one gets the DUTs allocated.

The scheduler works on a job by job basis with jobs receiving a priority based on the schedule priority so a high priority schedule overlapping a low priority schedule but starting later than the lower priority schedule on the same slot will not cause termination of jobs on the low priority schedule nor will the high priority schedule prevent the earlier low priority schedule starting. And if jobs on the low priority schedule are long running, it may prevent the high priority schedule running any jobs at all.

3.9.1.2.5 Shared Schedules

An advanced concept is that of shared schedules. This is a way to loosen the priorities associated with DUT allocation. Normally, a schedule is not shared. Sharing a schedule allows the exclusivity of slot or DUT allocation to be ignored and the slot or DUT to be available to other users but only when all the tests of the originating schedule are finished.

It allows, for example a user to set up a long daily schedule, with exclusive use on certain known slots but allow other users to use those slots or DUTs once all the tests have completed. Although, using DUT models may have a similar effect, it would not allow the user to confine the tests to certain servers or portions thereof. Using a DUT model would permit the scheduler to use the model on any server but using a shared schedule with DUTs in just one server would confine the test to that server while not making the use of the DUTs exclusive to the user creating the schedule.

3.9.1.3 Tests on Schedules

A test script is allocated to a schedule to be run - this combination of schedule and test script is referred to as a job. In general a schedule will have more than one job allocated to it. Each job has a priority and a scheduling mode which determine how the scheduler will run the jobs.

3.9.1.3.1 Job Priority

Each job can have a priority between 1 and 10. 1 is the highest priority. Jobs within a schedule will be executed in priority order unless resource usage precludes it. If multiple jobs have the same priority then the order is determined by the order the jobs were assigned to the schedule (first assigned being executed first).

Each job needs one or more DUTs and this can adversely affect the scheduling priority. Consider the case of a schedule which has 4 DUTs allocated to the schedule. Consider, also that the highest priority jobs require just 1 DUT as do low priority jobs. But there is a medium priority job which requires 4 DUTs in single process mode. When the schedule starts, the high priority jobs will be allocated to the DUTs as needed. As they finish, the medium priority job will not be able to be run as the 4 DUTs are not all free. The scheduler will find that it could schedule a lower priority job and it will do that. Thus, the medium priority job becomes, in effect, the lowest possible priority (when, finally, all 4 DUTs become free).

For this reason, it is not a good idea to mix multi-DUT jobs with single DUT jobs on the same schedule. In general, the advice is against using multiple DUTs in a single job.

3.9.1.4 DUT Usage and Scheduling Mode

When a job is defined, there is the option to run the job on more than one DUT. The scheduling mode determines how this is done.

3.9.1.4.1 Multi-Process vs. Single Process

A test which requires more than one DUT can be scheduled in one of two modes. Which mode to use depends on how the test is written (or the test is written based on the mode selected).

In single process mode, the test is expected to reserve all the DUTs in the script via multiple ReserveSlot() function calls. The scheduler will allocate from its pool of DUTs sufficient DUTs for the script to be able to reserve all the DUTs simultaneously. This is not the easiest way to write a test unless the test is simple – the question of error handling when one DUT does not respond as expected is complex. In single process mode, all the DUTs have to be in the same rack.

In multi process mode, the test is expected to reserve just one DUT in the script via a single ReserveSlot() function call. The scheduler will allocate just one DUT to the script. It will then invoke the script multiple times, each time with a different DUT until the count of DUTs has been

satisfied. This is an easier method to write tests which need to test multiple DUTs. In this mode, the DUTs do not have to be in the same rack as each test is totally independent.

Within one schedule it will not run the test more than once on the same DUT – so if, for example, the schedule has 4 DUTs allocated but the test has been set up to run on 10 DUTs, then it will run just 4 times, once on each DUT allocated to the schedule. This may seem strange but does have advantages if a lot of tests are on one schedule and suddenly, the schedule can not use as many DUTs as originally envisaged – the job definitions do not need to be modified. Just the allocation to the schedule.

3.9.1.4.2 DUT Coverage vs. Test Coverage

If more than one job on a schedule has the same priority and if the jobs are set up to use multi-process mode and more than one DUT then another factor determines the order of execution of tests.

In DUT coverage mode, the scheduler attempts to run each job in sequence on each allocated DUT without gaps in time. This gives maximum coverage of the DUTs – at the risk that the schedule may end before some jobs even get to run. If one DUT takes longer to run a test then it will become out of step with the other DUTs.

In Test coverage mode, the scheduler instead concentrates on running the jobs in sequence using whichever DUT is available. It will run each job once before attempting to run the first job for a second time. This gives maximum coverage of the test scripts – at the risk that not all DUTs will necessarily run any single job. The scheduler will not run the same job more than once on any single DUT within the schedule.

The order of jobs is not 100% guaranteed to be in sequence as an DUT may be available to run a job but has already run the next job in the normal sequence – in this case, another job will be selected (one that would normally be later in the sequence).

3.9.1.5 DUT Allocations

Each schedule needs at least one DUT allocation in order for a job to run and may have many DUTs allocated to it. You can choose to allocate DUTs using one of three methods.

3.9.1.5.1 By slot

In this method, a server and slot number are allocated to the schedule. This provides exclusive use of the specified server / slot to the user owning the schedule (other overlapping schedules of the user may use the server / slot if it is not in use by the original owning schedule)

3.9.1.5.2 By DUT

In this method, a specific DUT instance is allocated to the schedule. This provides priority use of the specified DUT to the user owning the schedule (other overlapping schedules of the user may use the DUT if it is not in use by the original owning schedule). If the DUT happens to be in a slot reserved by a different user, then the slot reservation will take priority.

This reservation method is useful for tests which, for some reason, must run on a specific individual DUT.

3.9.1.5.3 By DUT Model

In this method, a DUT model is allocated, along with a quantity. This provides no more than a request and no guarantee of exclusivity. The scheduler attempts to find a free DUT of the right model to use. The quantity is to limit the maximum number of DUTs a schedule can use.

A DUT will only be allocated using this method if it is not already allocated by slot or explicit DUT allocation.

3.9.1.6 Priorities

When a DUT is free, the scheduler attempts to run a job on the DUT to maximise the use of the system. After determining the set of jobs which are eligible for the DUT, the scheduler uses a priority based approach.

A job acquires a final priority based on the job priority, the schedule type and the schedule priority.

Schedule type is the next most dominant priority. A job on a one-off schedule will get preference over a job of any other type of schedule.

Schedule priority is then examined, followed lastly by job priority. Within job priority, the scheduling mode is used to order the jobs.

3.9.2 Writing a Schedulable Script

Writing a script to be scheduled requires very little changes from writing a script to be invoked from the command line. In most cases, no changes are needed at all and the same script may be used. This topic describes the important API issues.

3.9.2.1 ConnectToServer()

This function defines the server to be used. When running as a scheduled script, the function must still be called but the value of the server name is ignored because the scheduler will decide which server to use. Thus, you should not assume in the script that because you chose a specific server in ConnectToServer() that your script is connected to that server.

3.9.2.2 ReserveSlot()

This function defines the slot to reserve. When running as a scheduled script, the function must still be called but the slot number is ignored because the scheduler will decide which slot to use. Thus you should not assume in the script that because you chose a specific slot in ReserveSlot() that your script really is using that slot.

It should be noted that you may code your script to use a specific slot and pass that to other API functions that need a slot number. You may also rely upon the return values of API calls which return a slot number to return the same value as you specified in ReserveSlot(). This is particularly useful if your script uses 2 slots and you call ReserveSlot() twice with different slots – although you

don't know which physical slots you are using, your script can be coded as if you do know – the API is fully internally consistent.

3.9.2.3 Image Files

Often in a script, external image files are needed, for instance to compare with captures from the DUT. When using these, relative path names must be used because the script will not be run from any specific directory. Absolute path names cannot be used.

Also, because the script runner has its own storage, care should be taken that path names do not exceed the Windows limit of 260 characters – it is recommended that the internal path name for a script not exceed 200 so that some room is allowed for the directories used by the script runner.

3.9.2.4 SetDirectories()

The SetDirectories() call will not cause an error but will be ignored – the log files will be placed where the script runner decides.

3.9.2.5 What won't work

What won't work is setting up a test to take a slot as a command line parameter and using that number to take special action to the DUT based on that number, for instance, knowing that if it is slot 5, then the box is type XXX while if it is slot 6 then box must be YYY and so a different IR command sequence is needed.

To perform such actions, the command line should take the box type as a parameter and the scheduling should be set up so that the script runs on a specific slot or DUT.

As mentioned above the use of absolute file paths does not work at all. This means that you cannot use SetDirectories() to redirect a log file, then open the log file in the test script and do your own parsing of the log that you just generated.

4 Test Creator

4.1 Tutorial

4.1.1 Tutorial

This tutorial covers all the major steps and concepts of creating graphical test cases and test utilities. It also explains the features of the editor. It is divided into a collection of smaller tutorials, each of which builds on earlier tutorials. The available tutorials are:

Hello World - the first test case

Send IR Command - Doing something useful and writing the results to the StormTest Development Center database

Decisions and Navigation - comparing an image and acting on the result. Navigating to screens.

Adding a loop - repeating the same task

Functions and Utilities - use of private functions and common utilities

Reference Links and Resources - use of links between blocks and resources to create complex tests

See Also:

Test Block Reference

Compare Image Trainer

Compare Color Trainer

OCR Trainer

Detect Motion Trainer

Press Multiple Buttons Trainer

Detect Audio Trainer

Audio Video Analysis Preview

Computation Trainer

4.1.2 Tutorial 1 - Hello World

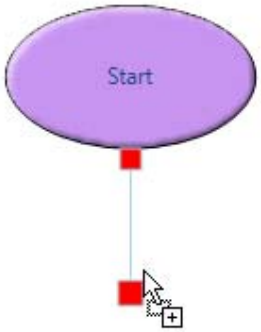
In the time honored way, we start with the very simplest program - in our terminology this is a test case.

4.1.2.1 Start the Test Creator

 From the main ribbon, click the `New Test Case` button. This will bring up the new test case dialog. Choose a name for the test case and a location to store the files. We are going to call this `Hello World` and leave the default location as `my documents`. You could of course change this.

Next we choose a DUT model for the test case. Every test case is designed for a specific model. You can run the test case on any DUT and future versions of StormTest Development Center may allow automatic allocation of test cases to DUTs. For now choose a DUT model from the available list and where there is a real DUT that you might be allowed to use. Click OK.

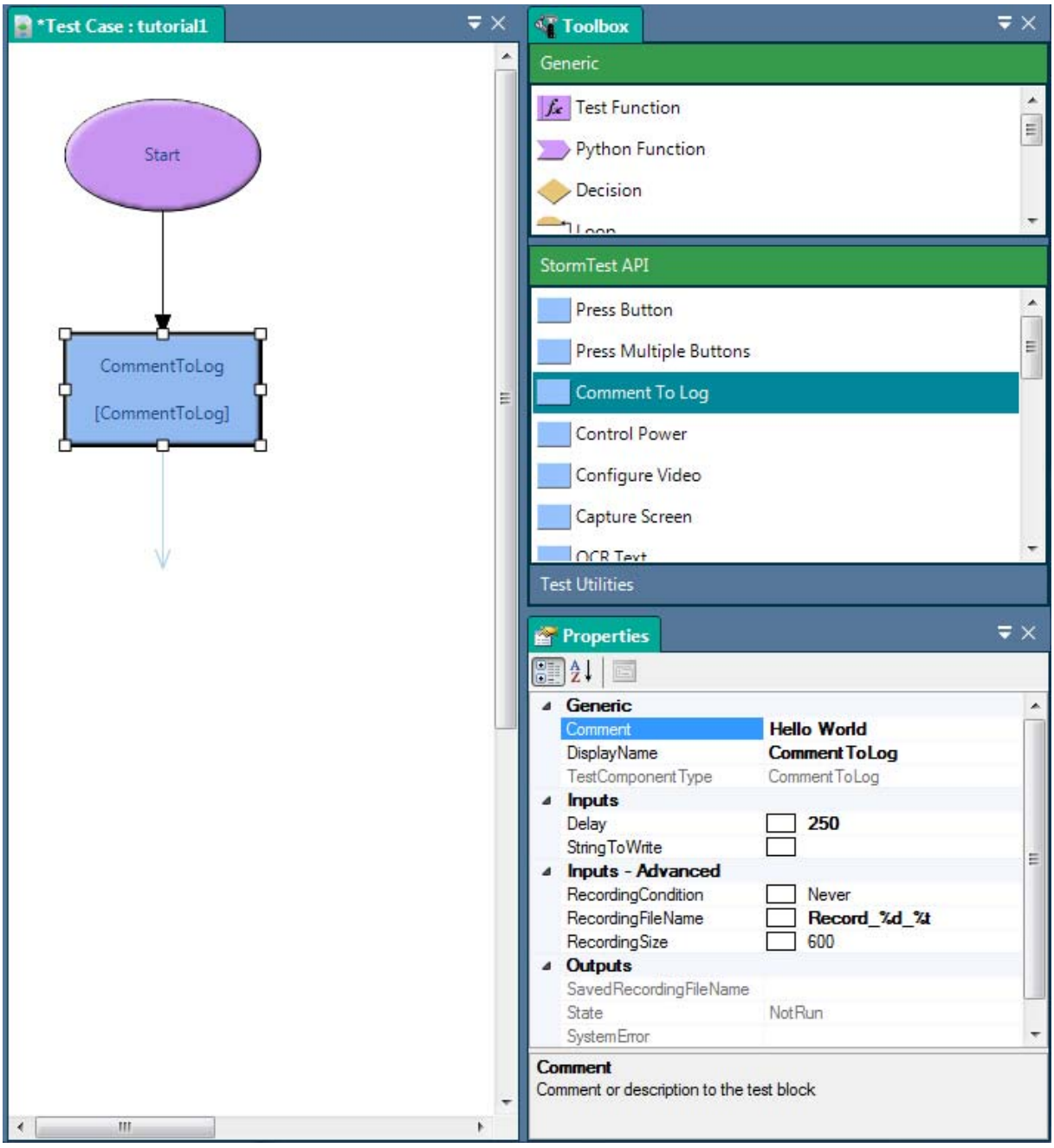
4.1.2.2 Adding to the drawing



You will now have a screen with a start block already added and an arrow below it. The arrow is pale blue because it is not connected. Every drawing has exactly one start block and you cannot delete it nor add more. Now, from the StormTest API control on the left, select the Comment To Log block and drag it onto the arrow. The arrow will change to have red squares top and bottom. Drop the block and it connects up below the start block. You will see a warning triangle in top right corner - this indicates some information is missing in the block.

Click the newly dropped block and you will see its properties on the right hand panel. Under heading Inputs, select String To Write and type in the box to the right, the words "Hello World". The warning triangle now disappears because the missing information (the comment to write) has now been filled in.

Your drawing should now look as shown below:



4.1.2.3 Run the test

 Now just click the run button on the ribbon. If all is well, the test runs. The Test Creator finds a free DUT for you (you see a dialog as it does this) and runs your test. It is very quick - you may see a brief flash on the screen. This is the interactive mode and useful for debugging. This is not the only way to run a test case. We can run it as a stand alone task which generates a log file so that we can see what happened. To do that:

Save the test case (either via the quick access toolbar, the main menu or simply Ctrl-S from the keyboard). Go to the scripts view of Test Manager, navigate in the local files to the file Hello World.sttc in your documents folder (or wherever you saved the test case).

Select the file, right mouse click and select Open from the menu. A dialog appears. Make sure the Run this test case on a DUT is selected and then, within the DUT selection area, choose Auto-select a DUT compatible with this test case. Click OK.

You will see a black text window appear, then a video window and then after a few seconds both will disappear. Your test has been run and a log file generated.

There will be a folder created called Hello World_dddd_tttt where dddd and tttt are the date and time of the test case execution. Open the folder and you will see the files created. The log of the test is called logFile.html. You can view it as described below:

4.1.2.4 Viewing the result of a Test Case

There are two ways to view the output.

1. Simply double click the logFile.html file, or use the right menu and open. The file opens in your default web browser.
2. Drag the file to the Test Log Viewer.

In this tutorial, we are going to use the Test Log Viewer, so drag the file to the icon. The help for the log viewer explains the log viewer in detail. If you scroll down the log file you will see the line:

2011-01-21 15:10:23.417 COMMENTHello World

4.1.2.5 Summary of Tutorial 1

We made a simple test case which wrote a line of text to the log file. We executed that test in two different ways and viewed the log file of the test. We will do something a bit more useful in tutorial 2.

4.1.3 Tutorial 2 - Send an IR Command

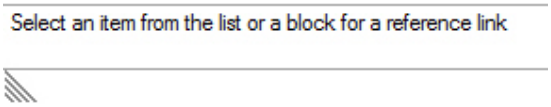
4.1.3.1 Open the file from Tutorial 1

If not still open, open it by using the Test Manager local files view or by starting Test Creator, selecting `open test` and browsing to the Hello World.sttc file. Let's start by renaming the test to something that we can use going forward. Deselect all test blocks by clicking outside of any. We can now change the properties of the test case itself. The reference page gives full details of all properties of all test blocks including the test case.

Select TestName and modify the text, say to tutorial. You are prompted with a dialog as you can either rename both the file and test case or just the test case. Click Yes and we rename both the file and test case.

4.1.3.2 Add some more blocks

Drop in a `Press Button` block from the StormTest API tools and then a `Pass` block from the Generic tools. Now select the Press Button block to show its properties. Under the inputs header in the properties, you see the property `ButtonName`, change that to something sensible. You see a custom window:

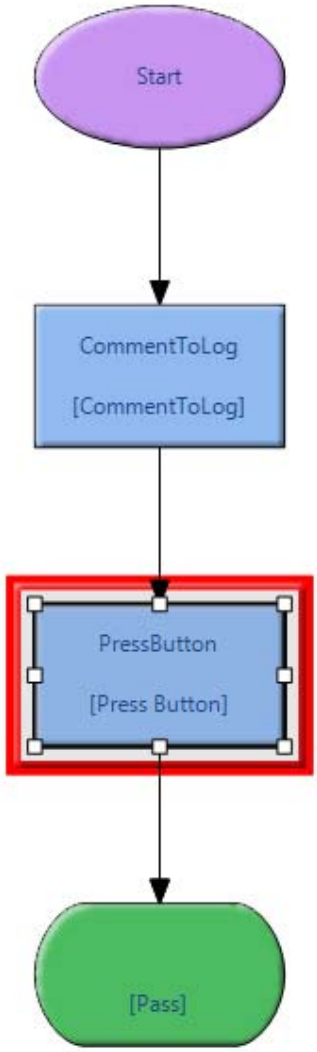


For now, just use the upper drop down to select a value from a list. Tutorial 6 deals with reference links.

You are presented with a list of valid buttons based on the DUT model you have selected. If there are none available, then you need to configure your system. The next property is the `Delay in Milliseconds` which is the time to wait after pressing the IR button before continuing. For this test set it to a long number such as 2000 (2 seconds).

4.1.3.3 Run it interactively

As before, click run. After the DUT has been reserved, each block will be shown executing. Since the `Press Button` is quite long you will now see it with a red box around it:



If you are doing this straight after tutorial 1, then the DUT is still reserved so you won't need to reserve it. The ribbon will let you know if the DUT is still reserved and which box is being used.

4.1.3.4 The Pass block

The last block is new. It indicates whether the test passed or failed. It also ensures there all arrows are terminated. This is important as the layout and auto layout tools on the ribbon can only work when all arrows are terminated. If you leave an arrow unterminated then when the test is run, the termination is considered to be a fail state. The test will end and a failure code is written to the database. This can be useful if you have a lot of error checking in the test and you want to just end on any error.

4.1.3.5 Experimenting with more blocks

You can drag other blocks onto any existing arrow - room will be made and the new block dropped into place. You can disconnect any existing arrow by clicking it and removing the arrow head. You cannot detach the non arrow head end. It can sometimes be difficult to see if an arrow is connected or not: black arrows are connected but light blue are not. Also the arrow head changed shape.

4.1.3.6 Naming blocks

If you add many blocks to the diagram you will find that they are somewhat unimaginatively named on the drawing. Each block has 2 lines of text. The lower line in square brackets indicates the type of block and cannot be changed. The upper line is called the `Display Name` and can be changed in the properties of each block - you can make it more meaningful. The display name must conform to certain rules: it must start with a letter and only contain letters, numbers or the underscore character (_)

4.1.3.7 Viewing video while running a test

Up to now you just saw a red block around the executing test block. If you click the video tab, you see the live video from the DUT. You may find it useful to reposition the video tab. Likewise, you also have access to an on screen remote control.

4.1.3.8 Viewing the result in the database

When you run interactively, Test Creator is in 'debug' mode or interactive mode. No data is written to the database. This is very useful when developing your tests. So let us do a test run where results are put into the database. Save the test case and run it as described in tutorial 1 from the scripts view. Now it has run, you can find the results. However, there is a problem: where did the test run? Since it was run on your machine, it does not show up in the scripts or schedules views within Test Manager. Nor does it show up in the unscheduled tests. It will show up in the slots and DUTs result views.

So, if you open the log file, near the top you will see some information such as:

```
16:40:19.746 INFO      Successfully reserved STB. Actual Server daniels, Slot 4 and unique id 5701
```

If you find your server and slot, you will see the result and it will be pass.

4.1.3.9 Summary of Tutorial 2

We have edited a test case, decided on a pass/fail status, given the block sensible names and have viewed a running test in real time. Tutorial 3 will bring in decisions and some debugging aids.

4.1.4 Tutorial 3 - Decisions and Navigation

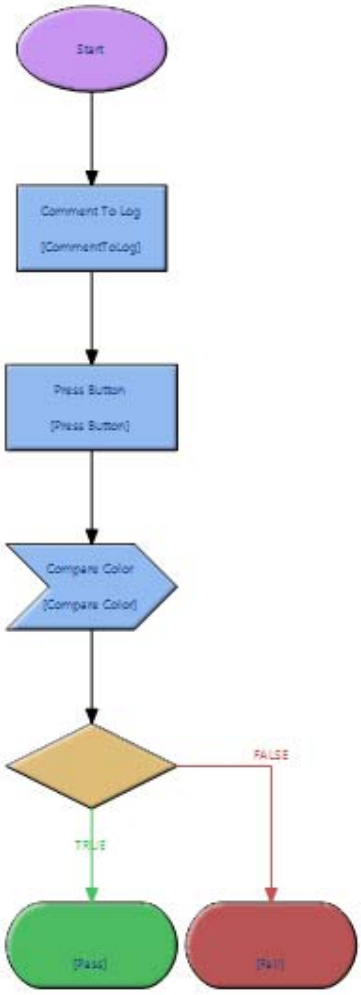
Up to now, all the test cases have been simple action type test cases. This tutorial introduces the decision block to allow execution of test blocks based on a condition of a previous block.

4.1.4.1 Add a Measurement Block

A decision block works on the output of the previous block. And, to be useful, the previous block needs to do some measurement. These blocks have a unique shape. So starting from the test case of tutorial 2, drag a `Compare Color` block after the `Press Button` (and before the `Pass` block). For now, we can leave its properties at default values (the reference guide explains them in full). The compare color block will compare the whole screen to the color 'empty' which defaults to black, and set an output property, `State`, based on the success of that comparison. It is likely that it will fail as not many DUTs output a pure black screen.

4.1.4.2 Add the decision block

Drag a decision block (equivalent to if-then-else in a standard programming language) after the `compare color` block and before the `pass` block. You will have an unterminated arrow out of it (marked FALSE), add a `Fail` terminator to it to get a complete drawing. Leave the Condition property of the decision block empty - its use is explained in tutorial 6. You should now have a drawing:



4.1.4.3 Run test and output properties

Run the test. You should see the red box, indicating the active block, move down the blocks and settle on `pass` or `fail` - most likely `fail`. Then it disappears as the test has completed. Click on the compare color block and look at the properties. Every block has some output properties and measurement blocks have a greater number than action blocks. You should see something similar to:

Outputs	
Detected Color	50, 71, 110
Detected Flatness	40
Detected Peak Error	94
Saved Recording File Name	
State	Fail
SystemError	

The State property is common to all blocks and is the one used by a decision block - in this case it is fail and so the FALSE branch out of the decision will be taken. You also see the actual color detected dark blue with RGB values of 50, 71, 110) along with other parameters that were measured. The reference guide explains in detail. Since dark blue is not black, the final state is fail. The Saved Recording File Name is part of the rolling record feature.

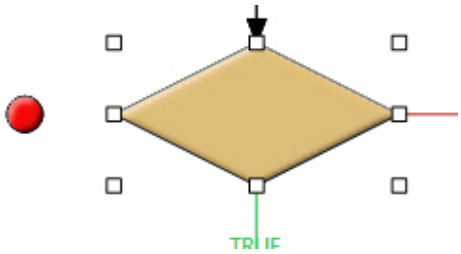
Since each block has a State property, it is important that the decision is placed immediately after the measurement block of interest - any later action block will set the State to Pass unless there is some fundamental reason why the action could not happen (eg a hardware failure).

4.1.4.4 Single stepping and breakpoints

To see what is happening, we can single step through the test. Now that the test has finished, click on the Step In button () on the ribbon. The test will go to the first block and stop, with it highlighted in a red box. You can see the properties of the box and also, by clicking on the correct tab, the video. You can re-arrange these panels to see the video, remote control and properties all at once if you like. You can use the remote control at any time to change the state of the DUT.

You will see the state of a block is NotRun indicating it has not yet run. Single step again to reach the Press Button block. If you now examine the properties of the comment block, you will see the state has changed to Pass.

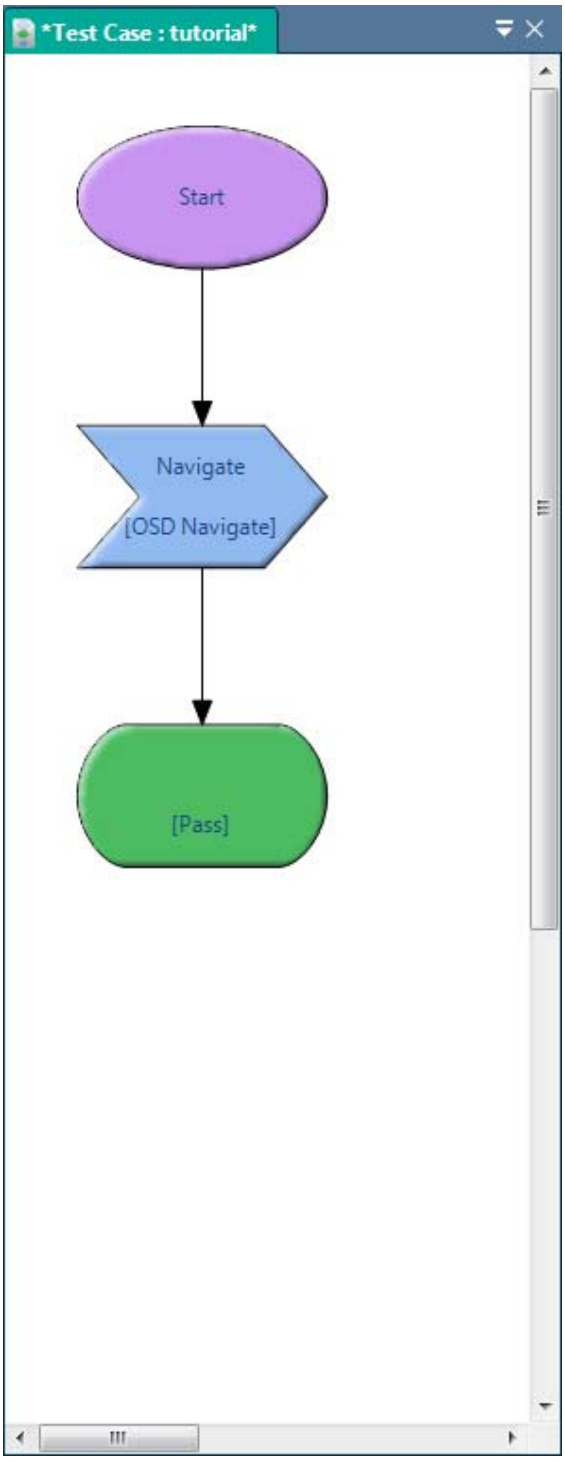
Instead of single stepping through each block, you can set a breakpoint. To see this work: Stop the test using the stop button on the ribbon. Select a block and then click the breakpoint button on the ribbon. The block will be shown with a red circle:



Now just click the run button and the test will run until the decision and then stop on it. You can now single step or continue at normal speed by clicking the continue button on the ribbon.

4.1.4.5 Navigation

Many tests require navigation to a screen and checking that the screen is the correct screen. Up until version 3 of StormTest, you would use the blocks above to build up a library of functions to do this. Version 3 introduced the navigator component. This allows you to define a whole system of screens graphically along with validation criteria. Once defined, the file is uploaded to the public navigators and configured to a DUT in the Admin Console. Now a test uses the Navigate Block instead of the PressButton and CompareColor blocks. As an example, to navigate to the 'TVGuide' screen would be:



and set the properties to:

Inputs	
Delay in Milliseconds	0
Destination	**TV Guide**
Navigator	
Source	

The block will now navigate to the screen and validate all intermediate screens needed. You will see the Navigator property highlighted. This was introduced in version 3.2 to allow you to select the Navigator to use for this block. By default, the default navigator of the DUT is used. However, each Navigate block can override this to specify an alternative navigator. Each test utility and test function can also have a navigator specified which will apply to all blocks in that test utility or test function. There is a hierarchy: the navigator specified in a Navigate block overrides the navigator used in the owning test utility. The navigator specified in the test utility overrides the default navigator of the test case.

4.1.4.6 Summary of Tutorial 3

We have added a simple decision to a test. We have used a very simple measurement block and seen how it produces values and a pass / fail status. We used the powerful new navigator of StormTest Version 3 to navigate to a screen. In addition, we have seen how to single step through the test and set breakpoints. In tutorial 4 we will add the other major flow control block - a loop.

4.1.5 Tutorial 4 - Adding a Loop

The last major flow control block to use is the loop (the loop is a Do-while loop in that it always executes at least once). We will also edit the previous test case that was used in Tutorial 3. Delete the Compare Color, the decision and terminators. You can select each one and hit delete key or use the mouse to highlight all the blocks and then delete. If you make a mistake, you can undo the delete (Ctrl-Z or use the quick access toolbar)

4.1.5.1 Add a Loop and some blocks

Drop the `loop` after the press button. Now drop the `compare color` into the middle of the loop. Now add a `decision` after the loop with a `pass` and `fail` terminator. If you click the start of the loop, you can set the loop count and the condition when the loop ends. For this tutorial, set the `LoopCount` to 10 and the `LoopEndCondition` as `LastBlockFail`.

4.1.5.2 Run the test and observe

You may find it helpful to fit the diagram to your page using the `fit` button on the ribbon as the drawing may not fit on the page. Most likely the loop is executed once and the final status is fail. This is because the compare color failed. Change the `LoopEndCondition` to `LastBlockPass` and run again. Now the loop will run 10 times and the test will still fail.

4.1.5.3 Loop Options

The basic loop has 5 means of ending:

- after the predetermined number
- when the last block passes
- when the last block fails
- when a condition is satisfied
- when a condition is not satisfied

In all cases, the LoopCount sets the maximum number of loops to be executed. The case of satisfying a condition is explained in tutorial 6. Unlike other blocks, the end of loop does not alter the state and so a decision placed after the loop uses the output of the block prior to the loop end.

Loops may contain other loops, decisions, terminators and any other block. However, you may not jump into a loop from outside a loop - you will find that you cannot make such a connection even though the block can be highlighted.

4.1.5.4 Summary of Tutorial 4

In this tutorial, we added a loop to run a set of blocks multiple times with an option to escape from the loop based on a condition. In tutorial 5 we will look at functions and utilities.

4.1.6 Tutorial 5 - Functions and utilities

This tutorial covers the 2 advanced and powerful topics of functions and utilities. They are covered together because the design method of both is the same but the usage is different. In this tutorial we will create a trivially simple function and an equally trivial utility.

4.1.6.1 Create a function

A function is private to a test case or a test utility so there is no menu item call new function. So open the test case from tutorial 4. Now drag the test function object onto the drawing. An empty function has been created. Select it and alter the `TestComponentType` property to MyFunction. This is the 'type' of the function - in some programming languages, it would be the name of the function.

4.1.6.1.1 Define the contents of the function

Double click the function to open it. Or right click and select `open`. Now you see the normal start block with an empty arrow. Drop a press button block and then a pass block to create an incredibly simple function. In the press button properties select a button to press.

This function simply presses a button and always returns pass. You could define a function to do far more. It need not return a useful value if you do not ever intend to add a decision block afterwards.

Now you can close the drawing - there is no need to save it because it is part of the test case.

4.1.6.1.2 Use the function

Drag the function to be in the loop before the compare color block. Now, highlight it, copy it (Ctrl-C, ribbon or right menu) and Paste it (Ctrl-V, ribbon or right menu). Another block called `MyFunction` appears. Drag that to after the Press Button block before the loop.

4.1.6.1.3 Use by Reference

You now have 2 blocks call `MyFunction`. They are the same. If you edit either block and change it then the other will change as well. This is because there is just one copy of your function and each block refers to that one copy - it is called a **reference**. This is an important concept.

You can use the function any number of times simply by doing a copy and paste on the function. It is as if you had created a new custom type of block - this is why it has no name and you changed the `TestComponentType` property.

4.1.6.2 Creating a second function

Now suppose you want a second function in your test case, similar to the first but not quite the same. Obviously, copying and pasting the `MyFunction` block won't do that - you get multiple uses of `MyFunction`.

So use the test function from the generic toolbox and drag it to your drawing. Change the `TestComponentType` to another name, for example `function2`. Now you can open `Function2` and define it. So to duplicate the contents of `MyFunction`, edit `MyFunction`, Ctrl-A to select all, Ctrl-C to copy all. Then edit `Function2`, Ctrl-A to highlight everything and Ctrl-V to paste. You can now modify the contents of `function2` based on the contents of `MyFunction`.

4.1.6.3 Create a utility

A test utility is not private to a test case (unlike the function described above). It is stored in the same folder as the test case. So to create a utility, use the menu, select a name and also a DUT model. For now, select the same model as you used for the test case. This version of Test Creator does not fully use the DUT model - later versions will use it to automatically select the correct test utility to use when running a test case. After creating the utility a new window opens for you to edit it. For this example, create a very simple utility. The structure here is identical to the function described above.

When complete close the utility window, agreeing to save the utility.

4.1.6.3.1 Use the utility

Open the Test Utilities tool box below the StormTest API - you should see your newly created utility there. Drag it to your drawing. You can copy and paste like you did with a function. You can edit the utility the same way. It is also used by reference in the same way.

4.1.6.4 Utility or Function?

Since the utility and function seem to be almost the same what is the difference and when should you use each one? If you now create a new test case in the same directory, you will find that you can use `util` in that test but not the functions. This is the core difference - the utility can be used in many test cases but the function is private to the owning test case. So, if the blocks inside the function could be useful in other test cases, it should be a utility but if not then it should be a function.

4.1.6.5 Single Stepping

If you single step through your test, you will find that the functions and utilities are opened and the active block goes through each sub block. This can be annoying if you know the function or utility works perfectly. In this case, you can use the Step Over button on the ribbon. This will execute the utility or function at normal speed without you having to keep pressing single step.

4.1.6.6 Summary of Tutorial 5

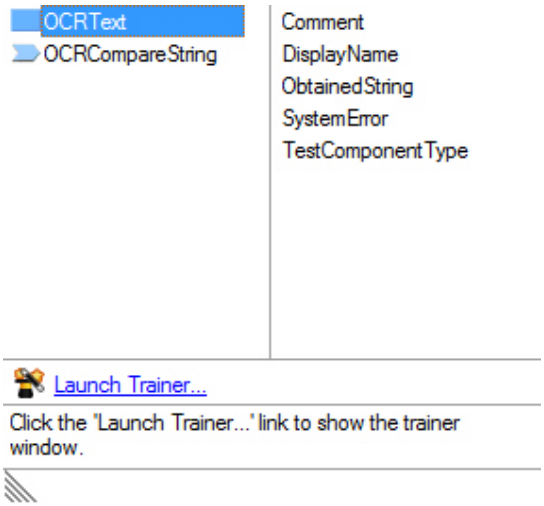
In this tutorial we used the advanced concepts of functions and utilities. The next tutorial adds the concept of linking inputs of blocks to other blocks and the computation block.

4.1.7 Tutorial 6 - Reference Links, Resources And Computation Blocks

In all previous tutorials, the test blocks had fixed properties typed in at design time. Test Creator supports the option of using values which reference other blocks, these can be resolved at run time. It also supports a computation block which allows calculation of values at run time. These features can also be used to create complex Decision and Looping blocks, enhancing the features used in tutorial 3 and tutorial 4.

4.1.7.1 Simple Reference Links

For this example, start a new test case. We won't worry about the core logic of this test so drop an `OCRText` and an `OCRCompareString` block after the start block. Now highlight the `OCRCompareString` block and then the `ExpectedString` property. This is the string that is used as a reference. Instead of typing in a value or using the trainer, click the arrow at the far right of the property line to show a new dialog. You see the list of blocks in the drawing that can be linked to. Click on `OCRText` and all the properties that can be used are shown:



Now click the `ObtainedString` property and the dialog closes and you see the resulting link in the property grid:

Now, do the same for the `OCRRegion` property of the `OCRCompareString` and link it to the `OCRRegion` property of `OCRText` block.

4.1.7.1.1 Link Summary and GUI hints

When selecting properties with links or blocks during linking, the source block in the diagram will have a dotted border to help you understand which block is being used. Only the compatible properties are shown as potential sources. By this, you need to understand that each property has a 'type'. Most are strings but others can be integers or floating point numbers. In the case of the `OCRRegion`, it is a rectangle - you can't link a rectangle such as the `OCRRegion` to a string like the `ObtainedString`. Also, you cannot like a property to itself.

Note how a link is comprised of 2 parts: the block name and the property name and that these are separated by a period (.)

NOTE: If you open a test made with versions prior to 2.5, you may not see a list of blocks to link to. This is because only blocks whose name conforms to the 2.5 and later naming convention are shown. Block names must be unique within a flowchart, start with a letter or underscore and contain only letters, numbers or the underscore. So `my_block_name_5` is OK, but `my block` is not. To use a test from a previous version of test creator with reference links, rename the blocks you wish to use.

4.1.7.1.2 Running the test

If you run the test, what actually happens? The `OCRText` block runs as a normal block and attempts to OCR the entire screen (we did not set the region). But `OCRCompareString` is more interesting. When it runs, it looks at the `OCRText` block to find the region to use - in this case the whole screen. It then runs the OCR and compares what it finds to whatever the `OCRText` block had found. Most likely they are the same and so the `OCRCompareString` block passes. If you now

change the `OCRRegion` of `OCRText` then both blocks will still use the same region for OCR - you don't need to update the `OCRCompareString` block.

4.1.7.1.3 Using the trainer with Links

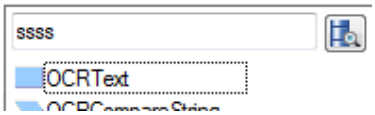
You will have noticed the little link at the bottom of each Reference link box to launch the trainer. If you launch the trainer for `OCRCompareString` you see information icons telling you where a link has been used. The expected text is empty because the `OCRText` block has not been run. But you can still use the trainer by entering the needed values manually. When you close the trainer you will be prompted to keep the original link or overwrite the link with a new explicit value.

4.1.7.2 Resources

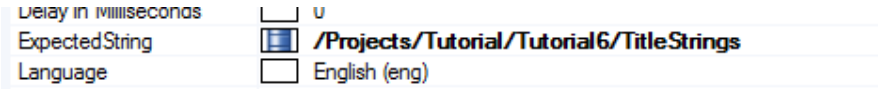
Often a group of tests will need to use the same value of text, rectangle, image or color. Although you can just enter the value in each test, using a resource allows central management of the value. If it changes, then it can be changed in the Resource Editor and all tests using that resource will use the new value.

4.1.7.2.1 Select a Resource

On `OCRCompareString` block, select the `ExpectedString` property and click arrow at the far right of the property line as above to bring up the dialog. Click the small database looking button:



Clicking the button produces a dialog showing you the existing resources in the facility. It is filtered by the type required for the property. In this case only named strings are shown. Simply select a resource to use. It will be shown using a resource icon in the final property grid:



You can also type in the path directly.

4.1.7.2.2 Running a test with Resources

When you run a test with resources, the value of the resource will be looked up in the database as the block is executed. The system uses a cache so each resource is only looked up once per test execution. The exception is when running in Test Creator debugger, where changes by other users are reflected immediately. If the resource is missing (someone deleted it, perhaps) then an exception is thrown at run time.

4.1.7.2.3 Using the trainer with Resources

The trainers work with resources but to protect against unintended changes, the fields using resource are read only.

4.1.7.3 Computation Block

The computation block allows you to make a computation of any complexity. However, let us start with a simple calculation. To illustrate the computation, it will compute the last 4 characters of the OCR found by `OCRText` and convert it to upper case.

Drag a computation block after the `OCRText` block. This makes sure it is executed after the `OCRText` block. The computation is filled in using the `Expression` property - the result will be in the `Output` property. You can type in the expression or use the trainer to help you develop the expression. The required expression is the scary looking:

```
ToUpper(Substring(OCRText.ObtainedString,OCRText.AllSymbols-4,4))
```

You can type this in directly. If you make a mistake, a message box will warn you when you attempt to navigate away from the property. Let us examine this in detail.

`ToUpper()` - this is a function to convert a string to upper case. It converts everything inside the brackets to upper case. In this example, the string to be converted is another function.

`Substring()` - this is a function which takes 3 parameters. These are the string to work on, the start position in the string and the number of characters to extract. It takes the `ObtainedString` from the `OCRText` as input. The starting position is another expression, in this case a number: `OCRText.AllSymbols`. After an OCR is performed, this output is set to the number of symbols found. So we start 4 in from the end and the last parameter is 4. So this will extract the last 4 characters.

4.1.7.3.1 Using the output of a computation block

Once the computation block has been run, the output is available. It can be used as input to any other block - for instance it could be used as the input to an `OCRCompareString` block.

4.1.7.3.2 The type of the computation block

The output in this example is a string but it can be an integer, floating point number or a boolean depending on the expression. Test Creator automatically finds the type and ensures that links to the computation block are made of the correct type. You can use literal expressions if you like in the computation block but the real power comes in using outputs and inputs from other blocks.

4.1.7.4 Enhanced Decision Block

If you ran the above test and nothing was found by the OCR then you would have got an exception complaining about an error evaluating the expression. This is because the number of symbols found would have been 0 and you can't find the last 4 characters. So we need some error handling here. That requires a `Decision` block so add one after the `OCRText` and before the computation. Now the standard decision block only tests that the last block passed. And finding no text is a pass condition - Test Creator has no way of knowing.

4.1.7.4.1 Add a condition to the decision

The decision block has a property called `Condition`. You can type in any expression that can evaluate to true or false (a boolean type) or link to any block that has a boolean property. Here we

are going to check that the OCRText produced at least 4 characters. To do that is quite simple so set the Condition to:

```
OCRText.AllSymbols >= 4
```

If the number of symbols is 4 or more, then this is true, else false. Now if you run the test and the OCRText block fails to find any text, the False branch out of the decision is taken - you could end the test or take some other action.

You can use the trainer as well as typing in the expression. Anything you can do in a computation block, you can do in the Decision Condition but it must evaluate to true or false - not a number or a string.

If you clear the Condition then the Decision will simply use the pass status of the previously executed block.

4.1.7.5 Enhanced Loop

In the same manner as the Decision block above, the Loop component can also take a condition. It is has the same characteristics as the Decision block. You need to set the LoopEndCondition property of the Loop to be ConditionSatisfied or ConditonNotSatisfied. If you set one of these but fail to set a Condition, then the loop will consider the Condition to be False always.

4.1.7.6 Summary of Tutorial 6

In this tutorial we used links between blocks to alter the input parameters of blocks at run time. We used resources to manage values of properties that are common among many tests. We also used the computation block to provide a means to manipulate data. We also used this technique to provide more control over the Decision and looping control blocks.

4.2 Using Test Creator

4.2.1 Test Creator - Main Window

The main window of Test Creator is divided into 3 main panels with a ribbon above them to provide access to commands. At the very top of the window, next to the StormTest icon, which acts as the main menu, is a quick access toolbar providing one click access to frequently used commands.

All the separate panels can be resized to suit your screen layout. They can also be made to float. The description below refers to the default layout after installing Developer Suite for the first time.

4.2.1.1 Work Area

The main central panel is the work area. This is where you create and edit Test Cases, Test Utilities and Test Functions, referred to as drawings. It uses drag and drop along with the usual keyboard short cuts. You can copy / paste items both within and between individual test cases, utilities and functions.

See the tutorial for detailed information on editing and creating tests.

4.2.1.2 Toolbox Area

The right panel is the toolbox and properties panel. The toolbox contains all the items that you can use in creating tests. It is divided into 2 separate panels. The upper panel, titled `Generic`, contains flow control blocks. You simply drag them on to the drawing.

The lower panel contains the remainder of the blocks you can use. These are further subdivided into 2 categories: `StormTest API` and `Test Utilities`. You switch between them by clicking the header.

The StormTest API contains the core blocks of the classic StormTest API in a high level manner. It is a subset of the features available in the full Python API.

The Test Utilities are the utilities that you have written. These can be dragged to the drawing to use them.

By default, the properties panel is below the toolbox.

4.2.1.3 Control Area

The left hand panel is the control area. It includes the Remote Control, Resources, Serial, Video and Overview panels. The layout of these can be altered to suit your working environment and screen size.

4.2.1.4 Ribbon

Above all the panels is the control ribbon. It has three tabs.

4.2.1.5 Ribbon Home Tab

The home tab is:

The controls are enabled and disabled as needed. The commands are grouped and a brief overview of each is:

4.2.1.5.1 Clipboard

This group deals with the Windows clipboard.

 Paste. You can only paste test blocks that you have selected to the main drawing. Ctrl-V also works

 Copy. Blocks copied to the clipboard can be pasted as text or an image to other Windows packages. As text, you see the raw XML.

 Cut. Copies and then deletes a block.

 Delete. Deletes the selected block.

4.2.1.5.2 DUT in Use

This group deals with the DUT in use for this test case.

 This shows you the DUT (by server[slot]) of the DUT used by the test case for running and debugging. If you have reserved more than one DUT, then you can change the DUT in use by clicking this and selecting an alternative DUT.

 Auto reserve. If you have no DUT reserved then clicking this will reserve a DUT of the same model type as the test case.

4.2.1.5.3 DUT Reservation

This group is the same as the main ribbon with the same functionality.

4.2.1.5.4 Diagram

This group is used to edit and create drawing

 Layout Diagram. If a drawing has all connections terminated, then clicking adjust the layout the drawing in a consistent manner.

 Auto layout. When a drawing has all connections terminated, then choosing auto layout will cause Test Creator to layout the drawing on each change. This can be time consuming and distracting for a large drawing.

4.2.1.5.5 Debug

Once a drawing has been opened, you can debug it using the group of commands:

 Run a test. It starts immediately and runs at full speed. If necessary, a DUT is reserved.

 Stop a test. It does not release the DUT that has been reserved.

 Break into a test and stop it. After this, you can single step through the test observing what is happening.

Set a break point on the currently selected block. Execution will stop at the block. There is no limitation on the number of breakpoints that may be set.

Step into a test block. Single steps to the next block. If the next block is a test utility or test function, the debugger goes to the first block of the utility or function.

Step over next block. If the next block is a test utility or test function, the debugger executes the entire utility or function as a single block without stepping through all the lower level blocks.

Launch Trainer. For many blocks, a graphical trainer exists and clicking this will launch the trainer.

4.2.1.6 Ribbon View Tab

The view tab controls how you can view the drawing. It is:

4.2.1.6.1 Zoom

This group controls the zooming. This can also be controlled from the keyboard.

Zoom in - makes the drawing larger, showing more detail. Press the control key and the + key on the numeric pad will do the same.

Zoom out - makes the drawing smaller, showing less detail but more of the drawing. Press the control key and the - key on the numeric pad will do the same.

100% - makes the drawing unscaled (the initial default size).

The current zoom level is shown as a % of full size and can be changed directly

Fit the drawing. 3 options are available: Fit to Page, Fit to Width, Fit to Height.

4.2.1.6.2 Show

This group of check boxes shows/hides the panels.

4.2.1.7 Preferences Tab

The preferences tab is:

4.2.1.7.1 Global Press Button Delay

This controls the delay after every button press. Whenever you create a multiple button sequence by entering the buttons directly in the multiple buttons trainer within either Navigator or Test Creator a delay is used. You can set the default to use here. You can also set the default from within Navigator editor.

The delay can be made persistent so that it will apply next time Developer Suite starts by ensuring 'Remember as default' is checked. If you check 'Update Test' then all button delays within the current test case or utility, including blocks in test functions that are used by the test case or utility. Any referenced Test Utilities will **not** be modified as they are stand alone shared files.

4.2.1.7.2 Test Creator Block Delay

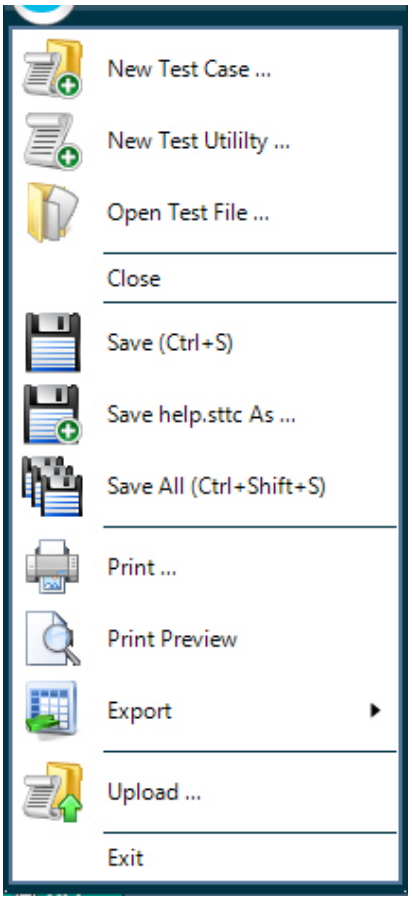
This controls the time at the end of each test block with a test case, test utility or test function. The delay is a property of each block and the default delay can be set here. The currently open test case can be updated by clicking the Update Test button. As with the Global Press Button delay, all blocks in test functions are updated as well as blocks in the test case or utility and referenced test utilities are **not** modified.

4.2.1.7.3 Test Creator Recording Size

This controls the default length of the rolling record buffer. Each block can record the most recent video and audio to a file. The length of the file, in seconds, can be controlled within the block but there is a default value set when you create the block. You can control this default value here. The currently open test case can be updated by clicking the Update Test button. As with the Global Press Button delay, all blocks in test functions are updated as well as blocks in the test case or utility and referenced test utilities.

4.2.1.8 Main Menu

Clicking the stormtest icon at the top left brings up the main menu.



The options are:

`New Test Case`: Creates a new test case, opening a new window.

`New Test Utility:` Creates a new test utility, opening a new window.

`Open Test File`: Opens an existing test case or test utility

`Close`: closes the currently open drawing so that you can re-use the window

`Save`: saves the currently open drawing

`Save As`: saves the currently open drawing in a new location and / or name.

`Save All`: saves all open drawings in the window, this includes all functions that are open.

`Print`: prints the drawing

`Print Preview`: a preview of what will be printed

`Export`: exports the drawing. Two formats are supported

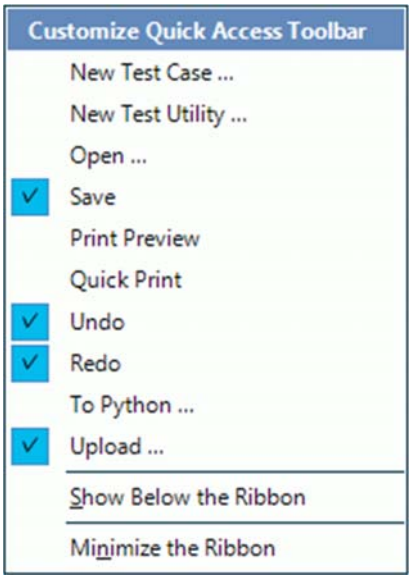
1. ZIP File. All needed files and images are zipped up into one file so they can be transferred to another machine
2. Python files. The open test case or utility is converted into a Python script

`Upload`: Upload the test case to the server repository so that it can be scheduled. This also uploads all images needed by the test case.

`Exit`: close the window

4.2.1.9 Quick Access Toolbar

Above the ribbon is the quick access toolbar. Clicking the arrow on the right hand side brings up a menu allowing you to choose which items are on the quick access toolbar. The available options are:



4.2.1.10 Customizing the layout

Each panel is resizable by clicking and dragging the divider bar between it and an adjacent panel. All panels can be floated by clicking the title bar and then dragging it to another location.

4.2.1.10.1 Docking

While dragging a panel, you will see on each panel a docking indicator:

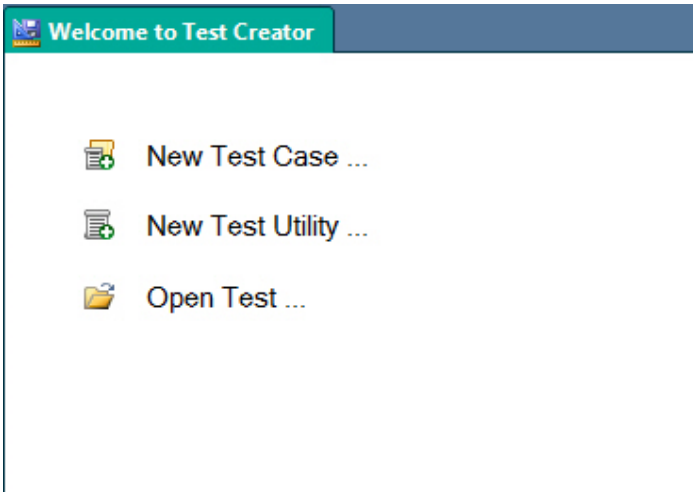
Dropping the panel on one of the 5 squares will dock in the associated location. If you drop on one of the squares with an arrow, the receiving panel will split and put the dropped panel on the appropriate side. If you drop on the center square then the dropped panel will occupy the whole area of the receiving panel and a new tab will be created to host the dropped panel.

As you drag, you will also see 4 isolated squares around the overall window:

 Dropping here will create a new top level panel in the area indicated.

4.2.2 Test Creator Welcome Page

The welcome page allows you just 3 options by clicking on the relevant button:

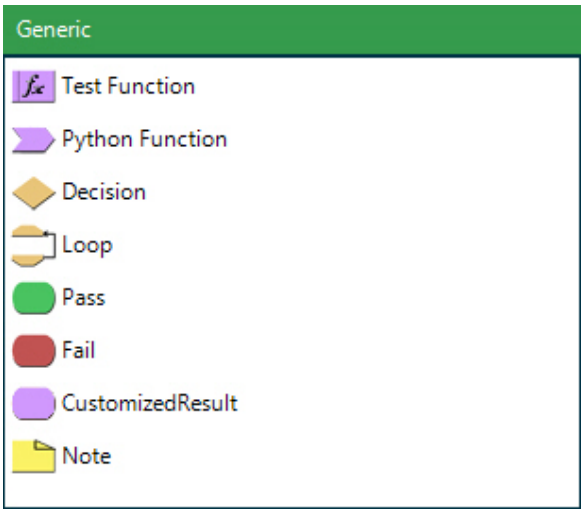


You will get to this page only if you start to create a new test case and then choose to cancel.

You can create a new test case, create a new test utility or open an existing test case or utility. You may find it quicker to use the Test Manager scripts view right menu button to directly create or open the test cases and utilities.

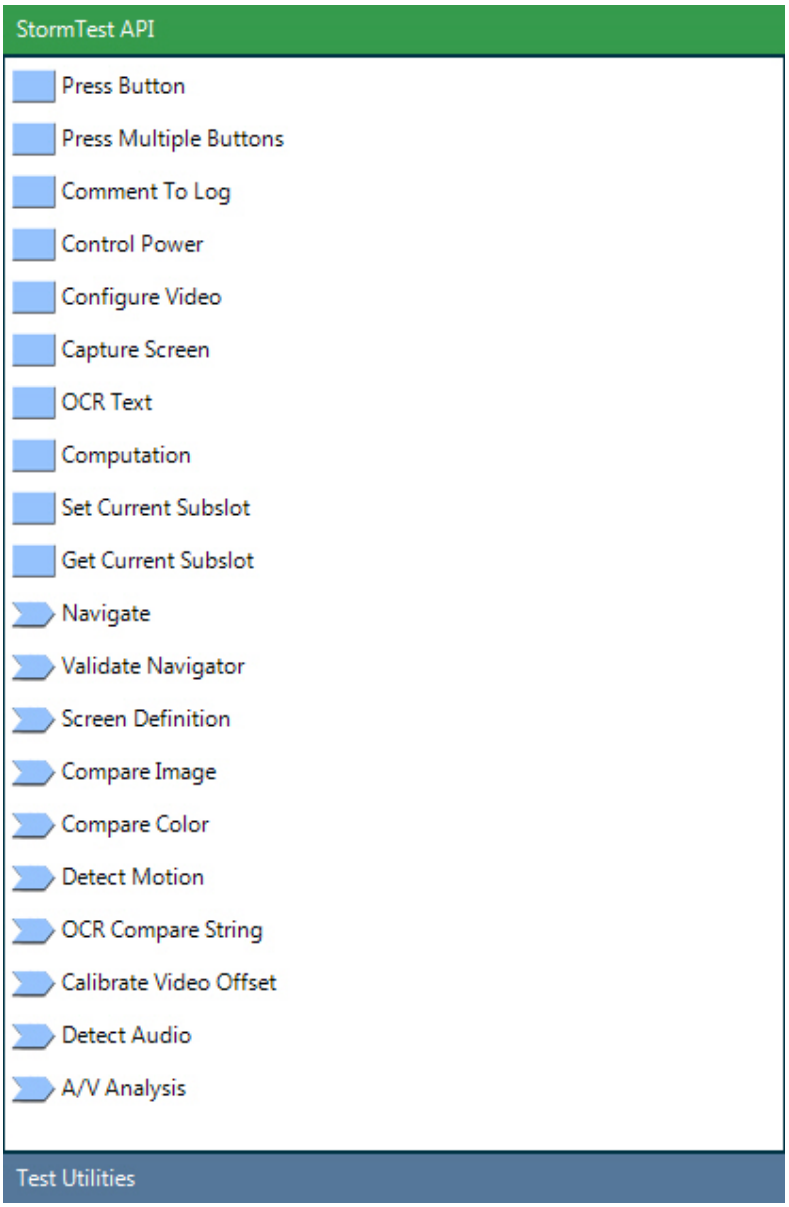
4.2.3 Common Tools

The common tools are essentially the flow control tools. Simply drag them to the drawing to use. These are always available.



4.2.4 API Tools

The API tools are the StormTest API test blocks which are used to control DUTs. They can be used in any test case, function or utility.

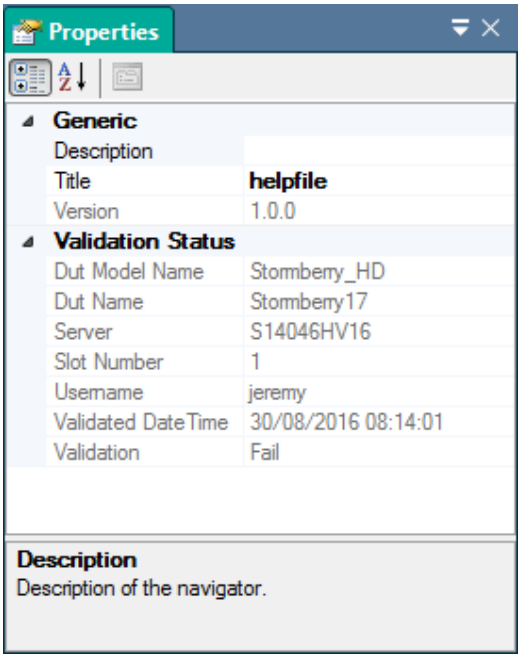


The upper items in rectangle are action blocks which do something and the lower items with a stylized arrow are measurement blocks which will measure something and generate a pass or fail status.

4.2.5 Properties Panel

The Properties panel is a standard properties grid showing the properties of the selected object. In Navigator and Screen Definition Editor, there are 2 properties grids. The upper is for the navigator, screen, link or screen definition depending on selection and the lower is for the selected test. This

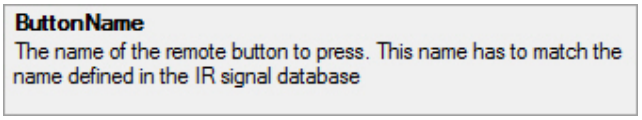
lower grid is tabbed with one tab per test. If there are no tests defined then this lower grid wil be hidden. In Test Creator there is just one properties grid for the test case or selected block.



The upper panel of each grid controls how the properties are displayed:

You can sort the properties by category (default) or alphabetically (by clicking the AZ button).

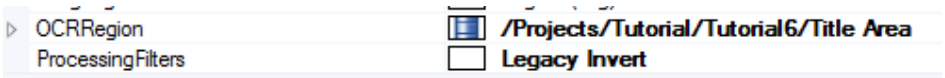
The lower panel gives a brief overview of the selected property:



In this case, the ButtonName property is described.

The middle panel is the properties of the block. Any property in gray cannot be edited, for instance the outputs are always gray as they cannot be altered by you.

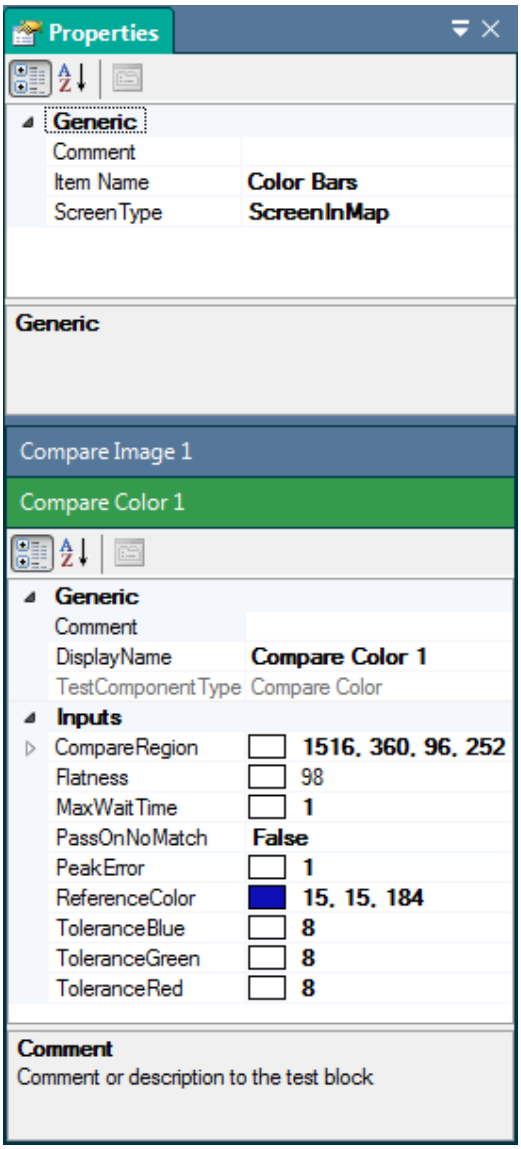
Some properties can use named resources. These are stored centrally in the database and the value is looked up at run time.



The OCRRegion above is a resource called `Title Area` in the folder `Projects/Tutorial/Tutorial6`. Reference links allow for dynamic data within a single test case but named resources allow separate test cases to use the same definitions for common items.

4.2.5.1 Screen Properties

When you select the properties of a screen, you see two sets of properties. The upper panel refers to the screen and the lower is tabbed with one tab per screen region. In the screen definition editor, the upper grid is always visible and reflects the properties of the screen definition object.



Click on the title bars to change the active tab (for example, clicking on 'Compare Image 1' above will show the properties of the Compare Image region on the screen).

4.2.5.2 Test Creator Ref Links

Within Test Creator, most properties have the option of being linked to another block within the test case, utility or function:

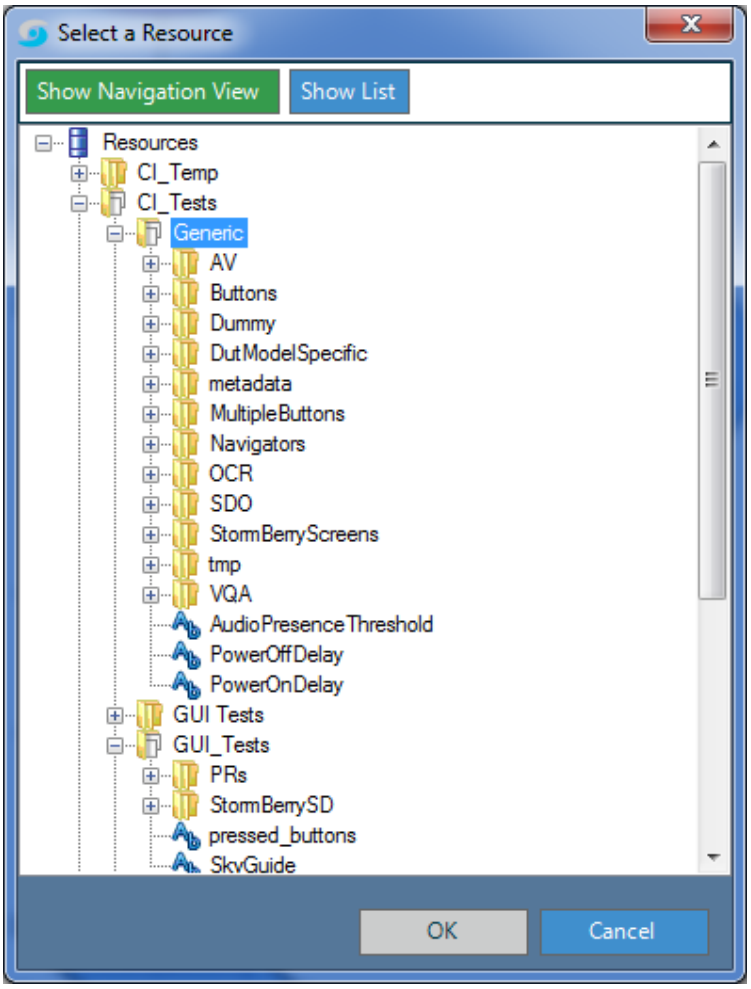
Analysis RunTime	AVAnalysis.RunTime
Analysis Type	Both

The Analysis Run Time above (an input) is linked to the RunTime property of the block called AVAnalysis. This is indicated by the blue link symbol. The white empty rectangle by the Analysis Type indicates that it is not linked and the property is unique to the block shown. The tutorial on reference links shows how to setup reference links.

This option is not availble from Navigator or the screen definition editor.

4.2.6 Select A Resource

From the Test Creator properties, you can select a resource for a property. The dialog appears:



You browse to the desired resource. Only those relevant resources are shown. The above example only shows regions as the underlying property was a rectangle and only regions can be used where a rectangle is needed.

You can view all resources in a list by changing to the list view - in this case you can sort by name or type instead of having to browse through a tree structure.

4.2.7 Remote Control

The remote control allows you to send commands to the DUT. If it is greyed out, it means that no DUT is reserved. You select the command to send by clicking the button.

 this icon will appear for the duration of sending an IR command and as feedback that you clicked a button.

In addition the keyboard can be used to send commands. The following keys are supported:

0 - 9

left arrow

right arrow

up arrow

down arrow

Page Up (sends channel+)

Page Down (sends channel-)

+ (sends volume+)

- (sends volume-)

The image used and whether or not you can actually send keys depends on the correct IR being trained and added to the StormTest Development Center database using the Admin Tools application.

If you click on a key with the right mouse button a menu appears allowing you to copy the key name to the clipboard. The name copied is the name of the button that would be needed in a Python script.

4.2.7.1 Macro Mode

At the top of the Remote control is the Macro Toolbar:

The controls are:

4.2.7.1.1 Record

Click this to start recording buttons from the IR remote control. Both the key and the time delay after the key is recorded. To stop recording, click the record button again. You will be prompted to save the macro to a file. If you choose not to then you may use it while the remote control window is available. After the window is closed, the macro is lost.

4.2.7.1.2 Play

Playback the macro. It plays back with the timing and keys previously recorded. While playing, the button changes to a Stop button so that you can interrupt playback.

4.2.7.1.3 Open

Click this to load a macro that you have previously saved to disk. You can then play back that macro.

4.2.7.1.4 Save

If you chose not to save a macro after recording, you can change your mind and save the macro later by clicking the save button.

4.2.7.1.5 Repeat

Normally, the macro plays just once. However, you can click the repeat button and it will play back continuously until you click the repeat button again.

4.2.7.1.6 Recent Macros

A drop down list of recently used macros so that you can switch between several macros easily.

4.2.7.1.7 Auto Hide

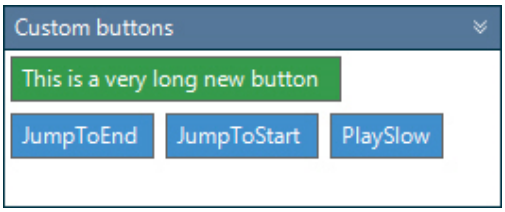
The right most pin symbol auto hides the macro toolbar - useful for small screens. Move the mouse near the top of the remote control window to show the toolbar again.

4.2.7.1.8 Importing

You can import a macro into a Press Multiple Buttons block, either via the right mouse menu or the Press Multiple buttons trainer.

4.2.7.2 Custom Buttons

Some IR remote controls have extra custom buttons that are not distinct button on the remote control. For example, some IR remotes behave in a special way if you hold down a key for a long time. Such button appear in the bottom panel and can be pressed using the mouse:



4.2.8 Video Window

The video window shows you live video whenever a DUT is reserved in the Test Creator. Below the video window is a small toobar with tools on it:

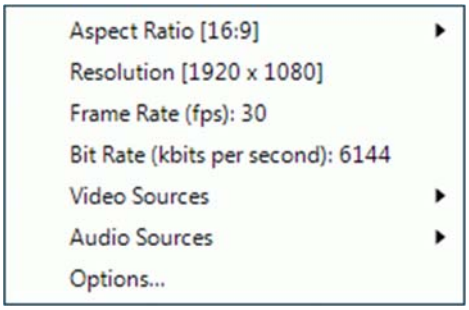
These are:

Audio mute. The mute state is shown. This is a toggle of the audio. If you hear no audio, check that your speaker on the PC is enabled as that can be muted and this control does not alter your PC audio status.

Close the video stream. This has no effect on the reservation of the DUT.

Open the video stream. Reverses the close

Change the video settings. Most of these are via a menu:



But the full range is possible by selecting options.

Grab a still image of the video - a normal file save dialog is presented

Start saving the video to a local file - a normal file save dialog is presented

Power status / control. The indicator is green as shown when the power is applied to DUT. Clicking it will turn the DUT off.

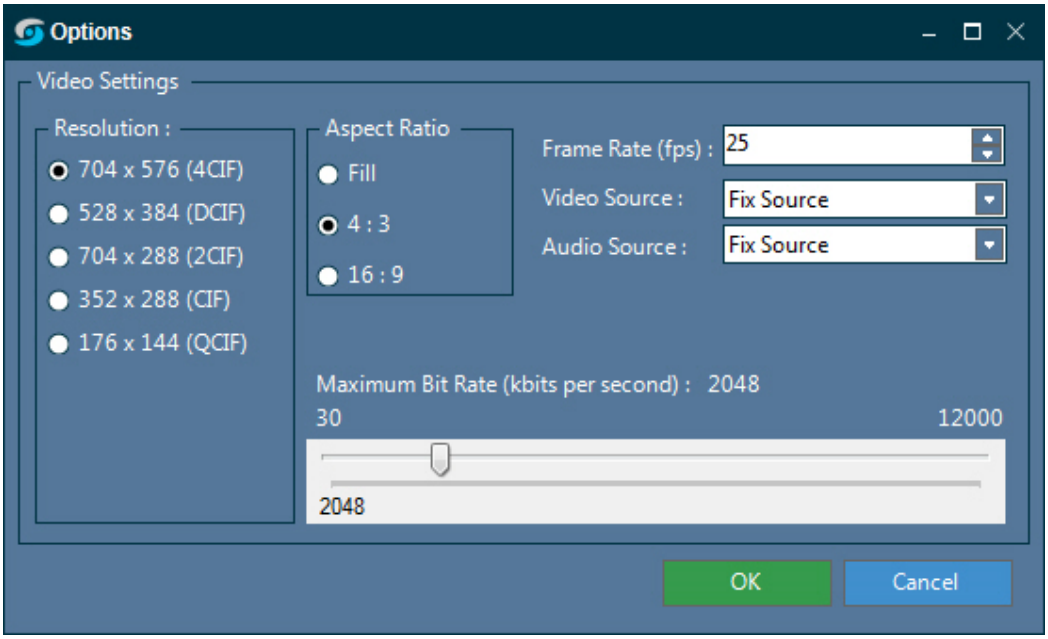
Power status / control. The indicator is red as shown when the power is OFF. Clicking it will turn on the DUT.

Sub slot control. If the DUT reserved is in an HS64 server then an extra icon is visible. From this a menu appears to allow you to view and select the sub slot in use. This icon is hidden for all non HS64 servers.

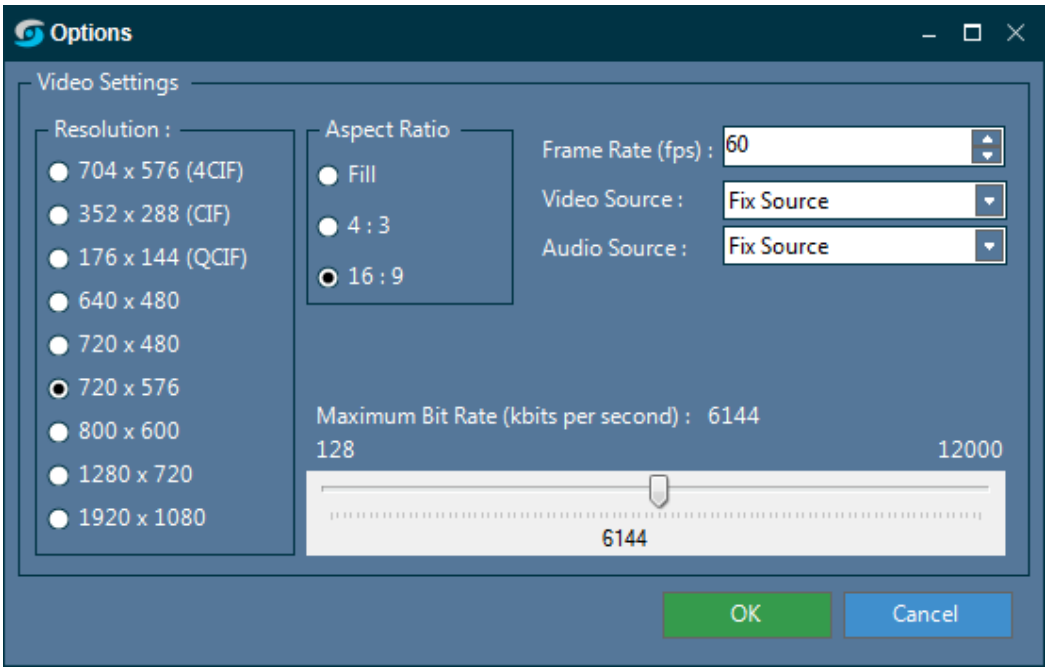
At the far right is the name of the server and the slot number of the video. If the video is an HD source then there is a little icon indicating HD: HD

4.2.9 Video Options

The video options allow you to set the video streaming parameters:



And if the source of the video is HD or 4K:



4.2.9.1 Resolution

Choose one of the fixed resolutions. The named values of the form xCIF are for PAL, those names xSIF are for NTSC. All other resolutions are for HD systems. The HD servers will use the negotiated raw HD resolution (usually 1920 x 1080) but for a 4K server, you may choose a resolution.

4.2.9.2 Aspect Ratio

This controls how the video is shown. Fill will use the available screen space and this will usually distort the video. HD streams will by default be shown as 16:9 (the normal aspect ratio for HD).

4.2.9.3 Frame Rate

This is the frame rate in frames per second. For PAL systems, the maximum is 25, for NTSC the maximum is 30. For HD systems, the maximum is 60.

4.2.9.4 Video Source

The video source. It will be 'Fix Source' unless the server has an audio/video switch installed. In that case, you can select the source of the video for streaming.

4.2.9.5 Audio Source

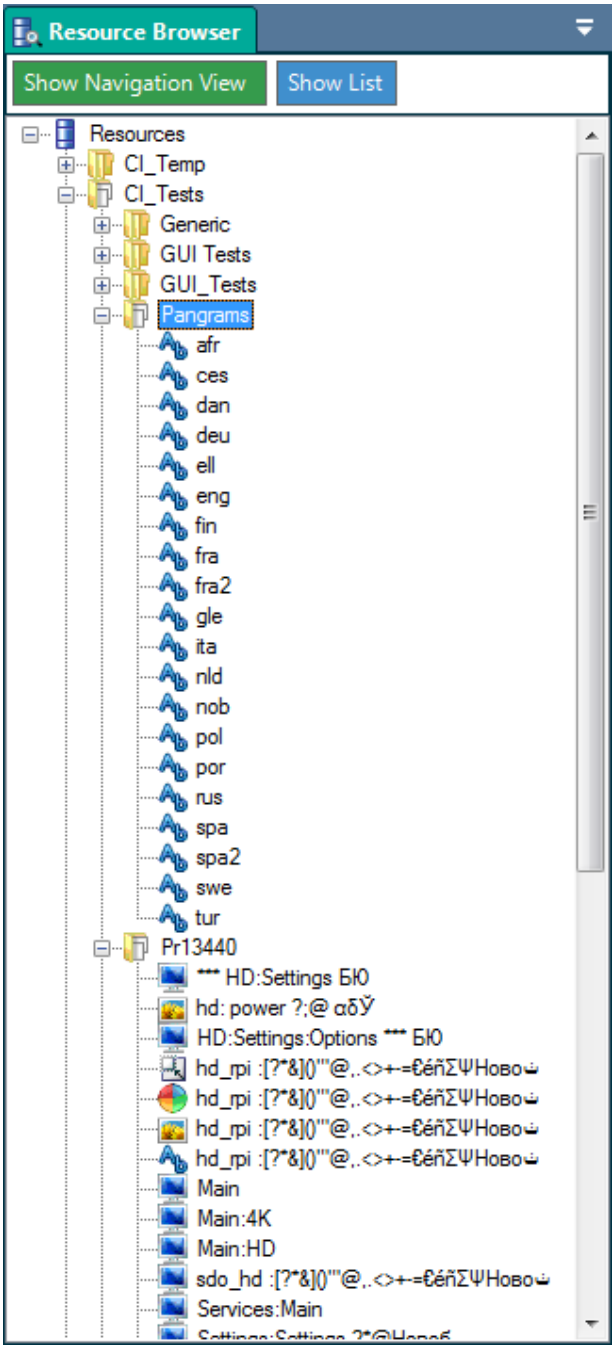
The audio source. It will be 'Fix Source' unless the server has an audio/video switch installed. In that case, you can select the source of the video for streaming.

4.2.9.6 Maximum Bit Rate

This is the maximum number of bits per second that will be sent. If the picture is static or slow moving, then far fewer bits per second will actually be sent. The slider controls the values in kbits/sec. **NOTE**: The accuracy of the maximum bit rate is approximately 10% so a setting of 1000 kbits per second could result in an actual bit rate 10% higher or lower as the performance of the video encoder is subject to a variety of optimizations. The maximum bit rate for SD systems is 4000 kbits/sec and for HD servers 12000 kbits/sec

4.2.10 Resources Panel

The resources panel allows you to browse and select resources:



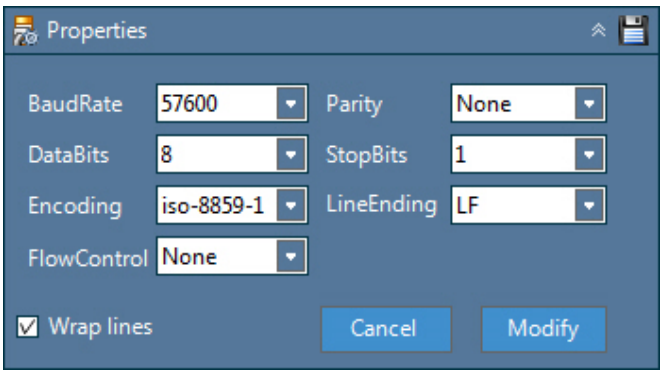
This is the same control as in the Resource Editor applet. You can see the resources, and perform some limited resource management (move, copy, delete, create new empty resources).

4.2.11 Serial Window

This window gives access to serial data from the DUT. Not all DUTs supply serial data so this may be empty. It only has meaning when the DUT is reserved. The serial data appears in the upper panel.

4.2.11.1 Properties

The properties panel may be shown or hidden using the arrows on the top right hand side of the properties title bar. The open view is:



You can change all the parameters. You must select the baud rate, data bits, parity, stop bits and flow control to match the DUT serial port characteristics.

The Line Ending alters how the serial data is displayed on screen - if it does not seem to have line wrap where you expect, change the line endings. The three normal settings are supported:

LF (just a line feed character typical of *nix operating systems)

CRLF (carriage return and line feed typical of Windows operating systems)

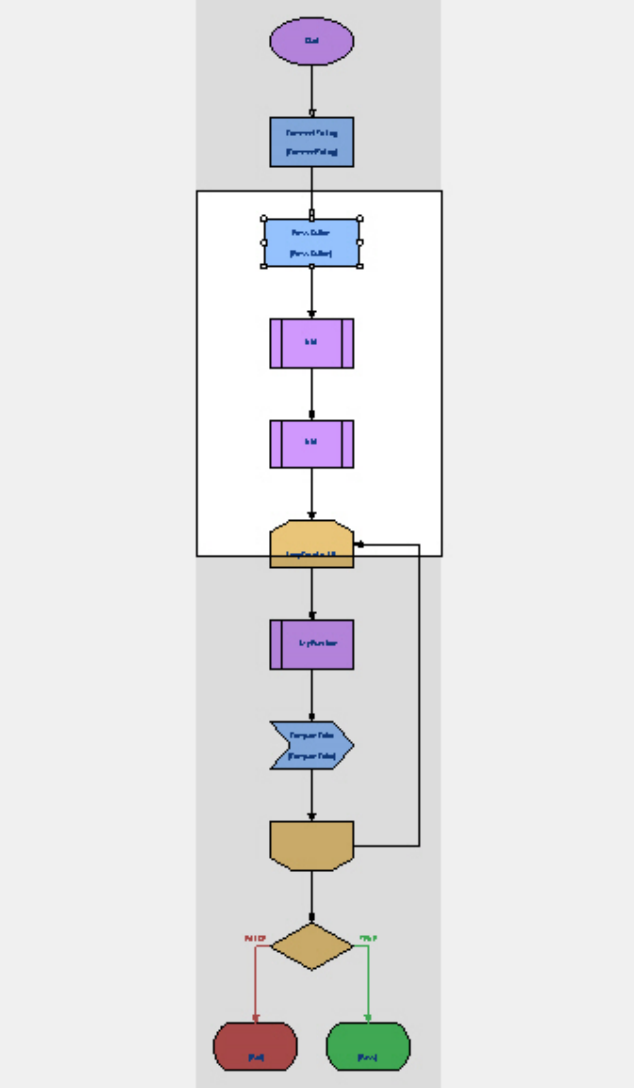
CR (just a carriage return - rarely used but available for completeness)

The Encoding determines how any characters above 127 are interpreted - for international use, you may need to select an alternative encoding. The default is ISO-8859-1, also known as Latin-1 and suitable for Western European Languages.

You can save the contents of the serial data by using the disk icon on the title part of the panel. You are presented with the usual Windows dialog box to select the folder to use.

4.2.12 Test Overview

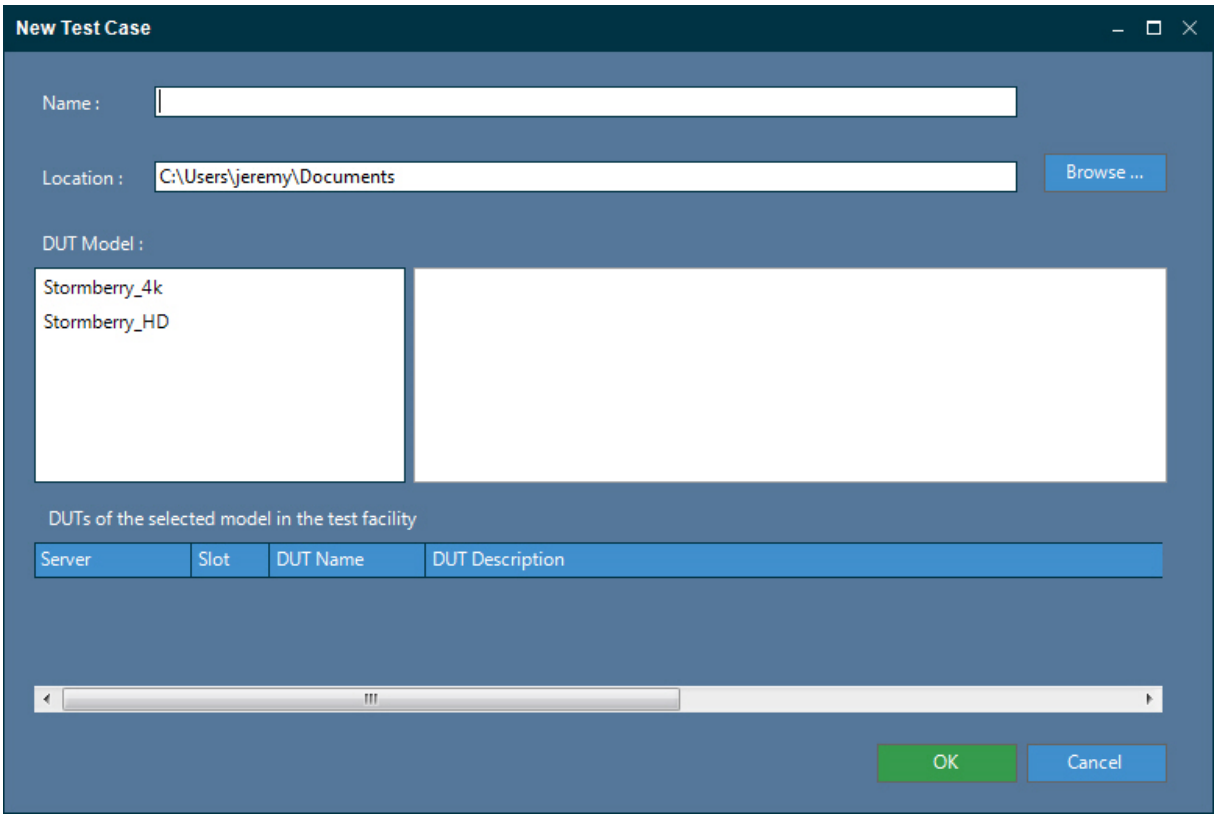
The test overview shows you an overview of the entire test, with the visible portion visible.



You can drag the highlighted area and the main work area will scroll to it. In a large drawing this is an easy way to scroll to the area that you want to view.

4.2.13 New Test Case

When you create a test case, you are presented with a dialog:



Name: This is the name of the test case. It will also be the name of the file used to store the test case.

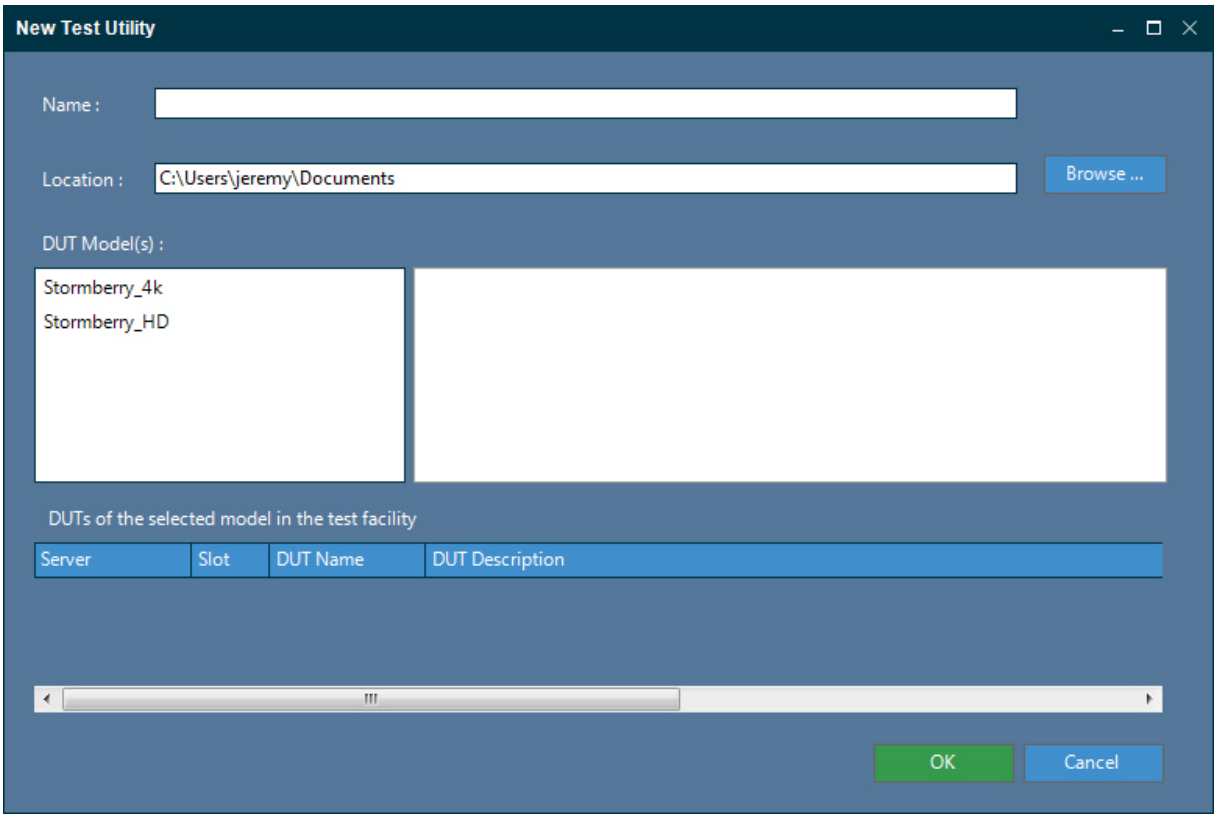
Location: This is the folder where the test case will be stored. You can type in a name directly or use the Browse button to select a folder. Note that if you type a name you must ensure it exists as Test Creator does not create folders for you.

DUT Model: This is the model for which the test case will be designed. Each test case has a model for which it is designed. You can run the test case on any model (but it may not work as you intend). The value is used by Test Creator to automatically find a free DUT during test creation. You can change this when editing the test case.

As you select a model, the right hand panel gives you a summary of the model you have selected and the lowest panel shows you each instance of the selected model in your facility so that you know how many exist and in which server(s) and slot(s).

4.2.14 New Test Utility

When you create a test utility, you are presented with a dialog:



Name: This is the name of the test utility. It will also be the name of the file used to store the test utility.

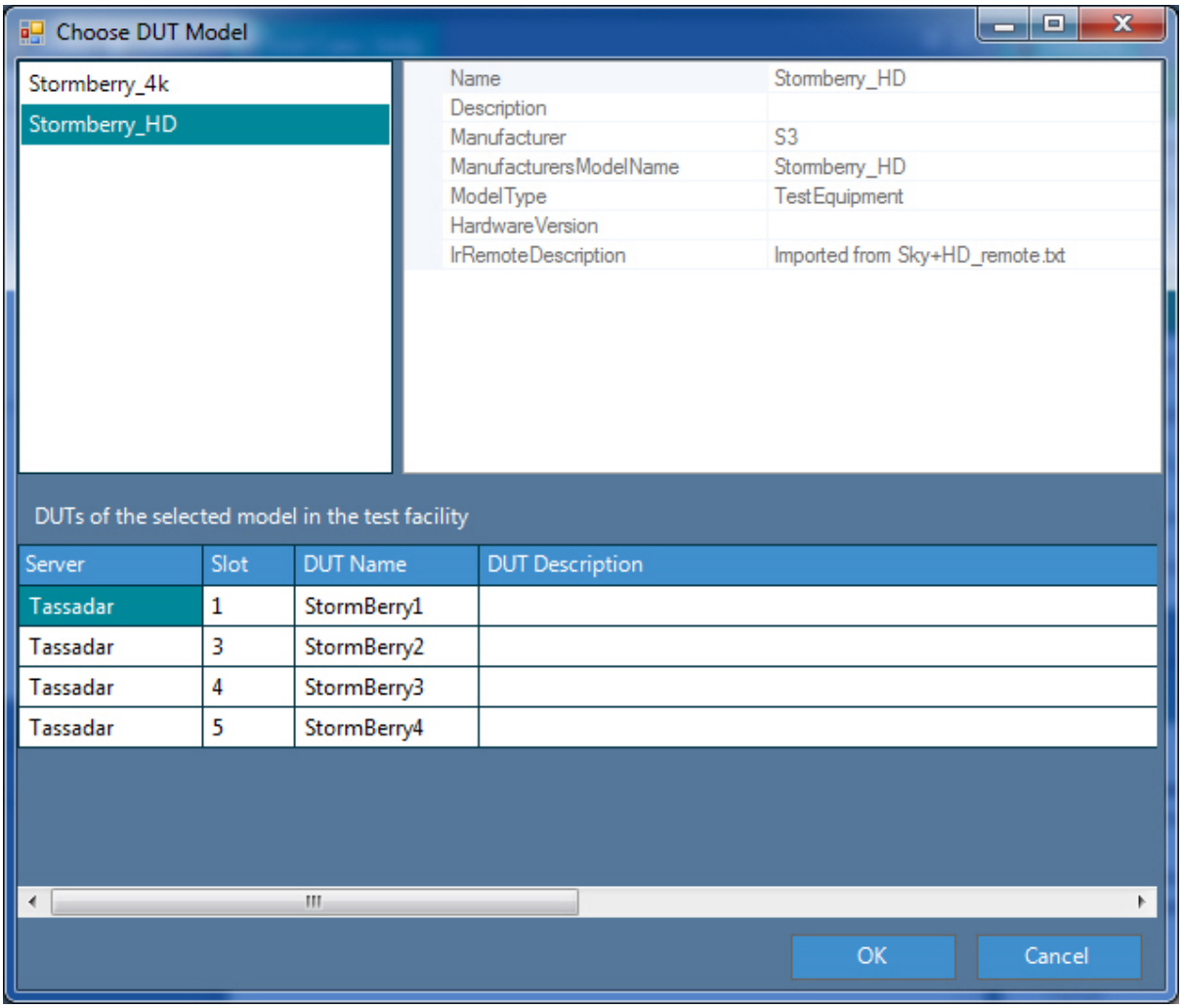
Location: This is the folder where the test utility will be stored. You can type in a name directly or use the Browse button to select a folder. Note that if you type a name you must ensure it exists as Test Creator does not create folders for you. Only test cases in the same folder can use the utility.

DUT Model: This is the model for which the test utility will be designed. Each test utility has a model for which it is designed. You can use the test utility in any test case on any model (but it may not work as you intend).

As you select a model, the right hand panel gives you a summary of the model you have selected and the lowest panel shows you each instance of the selected model in your facility so that you know how many exist and in which server(s) and slot(s).

4.2.15 Choose DUT Model

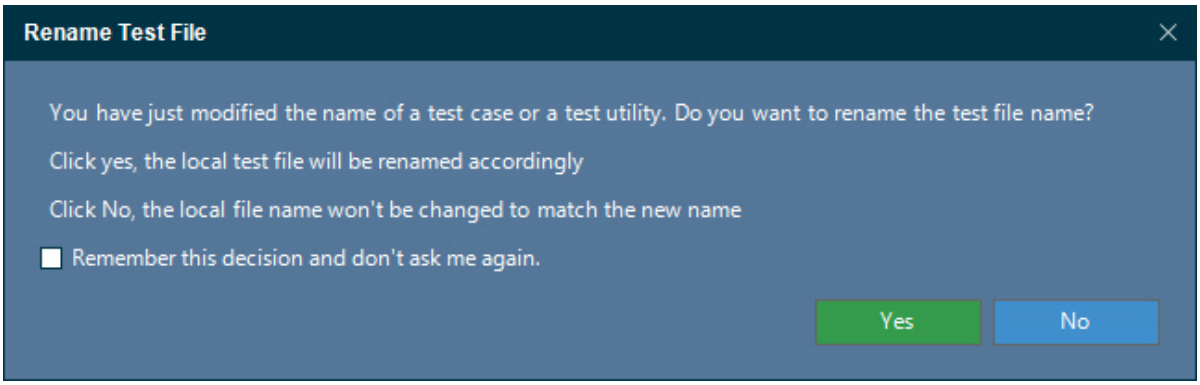
Once you have chosen a DUT for a test case, you can change it from the properties page. You see the dialog:



You need to select a model to use for the test case or utility from the list on the left. When you do, the details of the DUT model appear on the right and in the lower panel is the list of all instances of the model in the facility.

4.2.16 Rename File

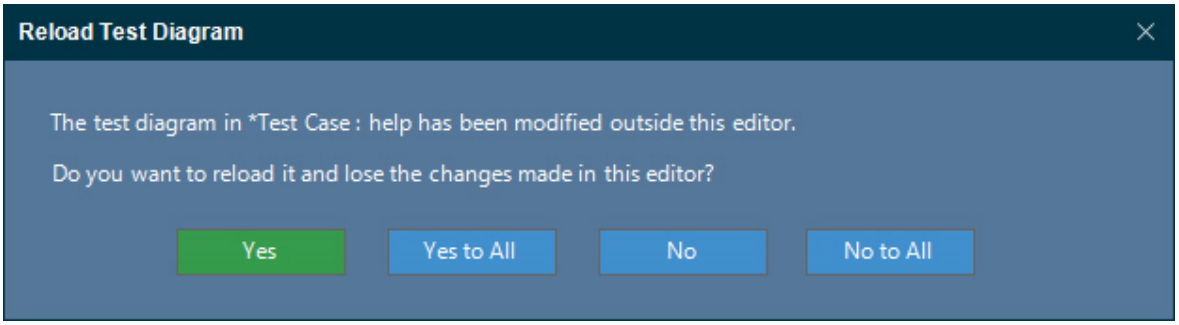
When you attempt to rename a test case or test utility you will be prompted with a simple dialog:



It is a good idea to keep the file name to be the same as the test case or utility name but you don't have to. If you check the checkbox then you won't be prompted each time but the action will be whatever you selected on this occasion. You can reverse this and have the prompt using the Test Manager preferences screen.

4.2.17 Reload Test Diagram

Whenever a test case or test utility is open in two different Test Creator windows, there is a possibility of a change in one window affecting the other window. When this happens you will see the dialog:

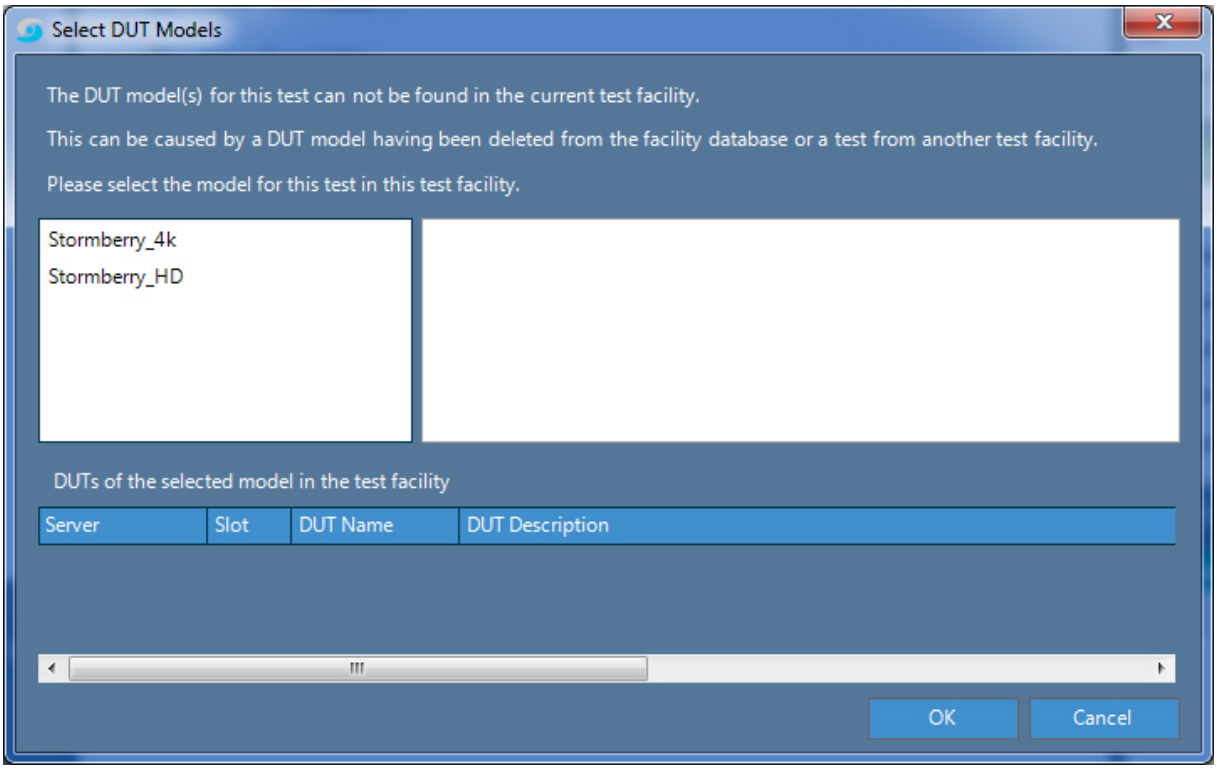


The answer should nearly always be yes. However, if you really are modifying the same item in two different windows then you need to review your changes carefully and decide exactly what you intend to do. Test Creator cannot help you there - it just warns you before you overwrite something.

If you do not reload and then make a change and save it then you will lose the previous changes in the other window.

4.2.18 Resolve DUT Model

This dialog appears when you open a test case or test utility and Test Creator cannot identify which DUT to use for the test. Each test case and test utility stores information about which DUT model it was designed for. When it is opened, Test Creator checks the database for that model and if it cannot be found this dialog appears:



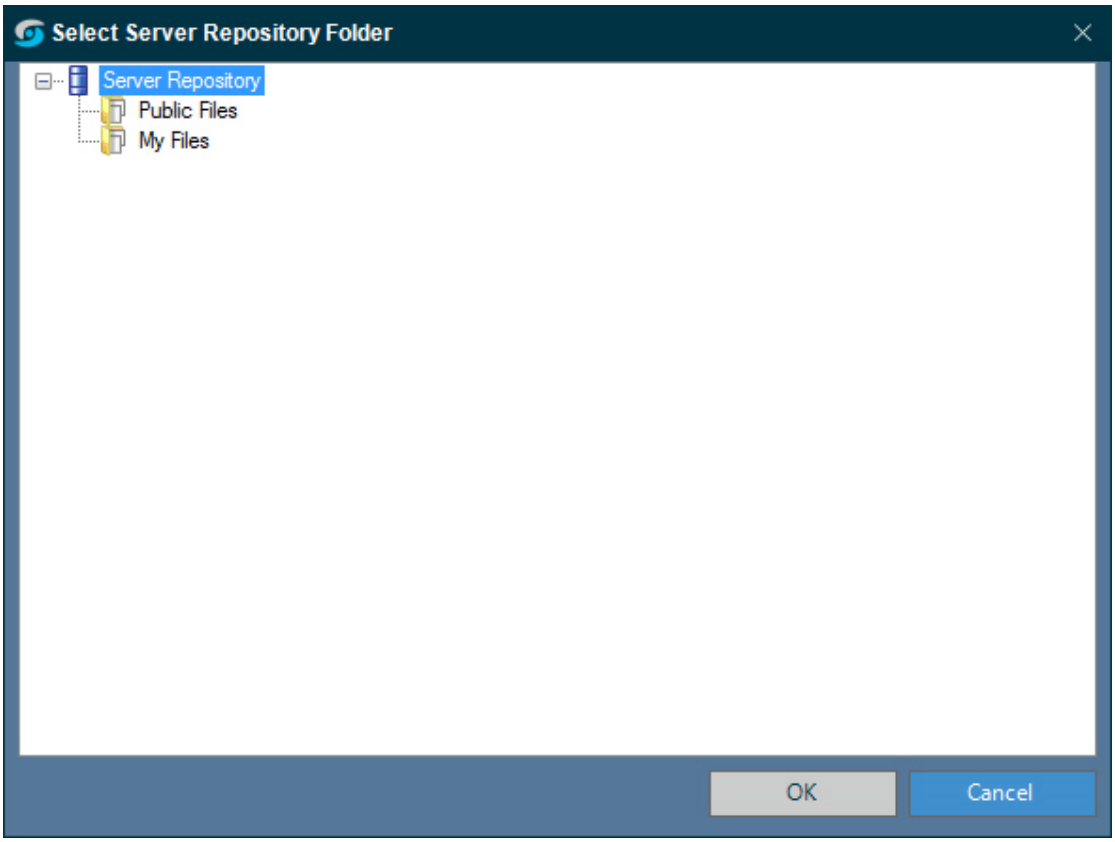
This can happen for 2 reasons:

1. Someone has removed the model from your database.
2. You are opening a test that was written for another facility and obviously the models will not be the same (at least, the internal identifiers won't be).

You need to select a model to use for the test case or utility from the list on the left. When you do, the details of the DUT model appear on the right and in the lower panel is the list of all instances of the model in the facility.

4.2.19 Select Server Folder

Select a folder from the list - this is where your test cases, utilities and associated images will be uploaded:



If you click cancel, you cancel the upload.

4.3 Reference

4.3.1 Terminology

The following terms, listed in alphabetical order, are used by the test creator:

.sttc File: An XML file that holds a Test Case.

.sttu File: An XML file that holds a Test Utility.

Reference Link: A link where the input value of one block is determined by the value of a property in another block, typically an output but it may be an input block.

Named Resource: A region, image, string, color or screen definition stored in the central StormTest database that is accessible to all tests and all users. It is a means to share common data among a group of tests.

Test Block: This is the fundamental unit of operation. A Test Function, Test Utility and a Test Case are each made up of 1 or more Test Blocks. Test blocks can be action type blocks (e.g. 'Press Button'), measurement type blocks (e.g. 'Compare Image') or flow control blocks (e.g. 'Loop').

Test Case: This is a complete test generated by the Test Creator. It has a file extension .sttc and includes all the logic necessary to start a test and write the results to the StormTest Development Center database.

Test Function: This is similar in concept to a Test Utility but is private to a Test Utility or Test Case. It is not stored in a separate file and cannot be shared between Test Utilities or Test Cases.

Test Step: This term is used in the Test Manager and by Python scripts to delimit a part of a python test script (which can be considered to be the same as a Test Case). Version 2.0 of Test Creator does not support this concept but a later version will.

Test Utility: This is a self contained object, stored as a file with a .sttu extension. It cannot be executed but is referenced by one or more test cases.

Trainer: A GUI area where you can set the parameters of a block in a graphical way. Most trainers have some automatic tuning function where the ideal parameters can be set for you.

4.3.2 Test Block Reference

This reference lists all the properties of all the blocks.

4.3.2.1 Common Properties

Common properties are not listed against each block but are listed here.

4.3.2.1.1 Generic Category

`Comment`: This is common to all blocks except Start. It is free format text for you to add any comments you wish.

`DisplayName`: This is common to all blocks except Start, Function, Decision, Loop, Utility and Test Case. It is the name shown within the block.

`TestComponentType`: This is common to all blocks except Start and Test Case. It is read only in most cases, however, you can change it for the Function and Utility where it becomes the displayed name.

4.3.2.1.2 Inputs - Advanced Category

These 3 properties are common to all blocks except Start, Function, Decision, Loop, Pass, Fail, CustomizedResult, Note, Utility and Test Case.

`RecordingCondition:` This determines the condition for the rolling record buffer to be saved after the block executes.

`RecordingFileName:` This is the file name that will be saved if the rolling record buffer is to be saved. It is a relative path and you can use %d and %t as place holders for the date and time.

`RecordingSize:` This is the length of the recording to save in seconds.

4.3.2.2 Start

No properties.

4.3.2.3 Test Case

4.3.2.3.1 Description

The top level test case, representing a full test.

4.3.2.3.1.1 Configuration Category

`DisableClientImaging`: When true, video is not streamed to the client nor is video preview available. The video measurement blocks for standard definition may work more slowly as they have to request images from the server as still images.

`ShowVideoPreview`: When true, the video window is shown during test execution.

`DUTModel`: The DUT model for which the test case is designed.

`UseVideoOffsets:` When True, the test case will apply the video offset stored in the database to all images captured before performing Image compare, color compare, detect motion or OCR operations. This allows you to write a test for one box and run it on a similar model from a different manufacturer without having to re-capture all the images. If this is false, no video offsets are applied. This is only used for tests running against non HD servers. Offsets are not needed in HD systems.

NOTE: if the test contains a Calibrate Video Offset block, then the calibrated offsets will always be applied after execution of the Calibrate Video Offset block.

4.3.2.3.1.2 Generic Category

`Owner`: Read only. This is the user who created the test case. The Windows login name is used.

`RecordVideo`: When true, a video log is produced of the test run.

`TestName`: The name of the test case.

4.3.2.3.1.3 Inputs Category

`Navigator:` The Navigator to use for this test case. You can only select from among the Navigators configured for the model for which this test case is being designed. If not configured then the default preferred Navigator for the DUT Model will be used.

4.3.2.3.1.4 Outputs Category

`AutoFail`: When true, any unterminated branch is considered to be a fail. When false, an unterminated branch is an error and stops the test.

`State`: The state of the test case. NotRun, Pass or Fail.

4.3.2.4 Test Function

4.3.2.4.1 Description

A private function. It belongs to a test case or a test utility.

4.3.2.4.1.1 Configuration Category

`DUTModels`: The list of DUT models for which the function is designed. A function may have a design for >1 model.

4.3.2.4.1.2 Generic Category

`Owner`: Read only. This is the user who created the function. The Windows login name is used.

`CurrentDUTModel`: The mode that the function is currently being used upon.

4.3.2.4.1.3 Inputs Category

`LogAsTestStep:` If true then this function will be considered a test step. The result of the function will be written to the results database as the result of the test step. The log file will also be delineated with the blocks of the function with a test step block.

`Navigator:` The Navigator to use for this function. You can only select from among the Navigators configured for the model for which this function is being designed. If not configured then the Navigator of the parent of the function will be used.

`TestStepName:` The name of the test step in the database results and the log file. If empty then the TestComponentType value will be used.

4.3.2.4.1.4 Outputs Category

`AutoFail`: When true, any unterminated branch is considered to be a fail. When false, an unterminated branch is an error and stops the function and test.

`State`: The state of the function. NotRun, Pass or Fail.

4.3.2.5 Test Utility

4.3.2.5.1 Description

A test utility, which can be used by more than one test case.

4.3.2.5.1.1 Configuration Category

`DUTModels`: The list of DUT models for which the utility is designed. A utility may have a design for >1 model.

4.3.2.5.1.2 Generic Category

`Owner`: Read only. This is the user who created the utility. The Windows login name is used.

`CurrentDUTModel`: The mode that the utility is currently being used upon.

4.3.2.5.1.3 Inputs Category

`LogAsTestStep:` If true then this utility will be considered a test step. The result of the utility will be written to the results database as the result of the test step. The log file will also be delineated with the blocks of the utility with a test step block.

`Navigator:` The Navigator to use for this utility . You can only select from among the Navigators configured for the model for which this function is being designed. If not configured then the Navigator of the parent of the utility will be used.

`TestStepName:` The name of the test step in the database results and the log file. If empty then the TestComponentType value will be used.

4.3.2.5.1.4 Outputs Category

`AutoFail`: When true, any unterminated branch is considered to be a fail. When false, an unterminated branch is an error and stops the utility and test.

`State`: The state of the utility. NotRun, Pass or Fail.

4.3.2.6 Python Function

4.3.2.6.1 Description

Call an external Python function. This does NOT work during execution of a test in Test Creator. It only works when you export a test case to python code.

4.3.2.6.1.1 Debug Category

`Simulated Return`: Set to true or false so that you can debug the rest of the test using Test Creator. The return state of the function will be treated as this value.

4.3.2.6.1.2 Inputs Category

`Delay`: The time to wait in ms after finishing the function before continuing with the next step.

`FunctionToCall`: name of the python function to call.

`ModuleToImport`: name of python module to import. Can be a simple name or a file name (which will be converted to an import statement)

`Parameters`: list of parameters to pass to the function. No checking to the syntactic correctness is performed.

4.3.2.6.1.3 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the function. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.7 Decision

4.3.2.7.1 Description

Branch on a condition. This can be the pass/fail status of the previous block or a custom condition.

4.3.2.7.1.1 Inputs Category

`Condition:` A boolean expression (or a link to a boolean property) to control whether the True or False branch out of the block should be used. If left blank then the pass or fail condition of the previous block is used.

4.3.2.8 Loop

4.3.2.8.1 Description

Execute a sequence of steps in a loop. It is split graphically into two parts: a begin loop and an end loop. The end loop only has the common properties. The start of the loop has extra properties.

4.3.2.8.1.1 Looping Category

`Condition:` A boolean expression (or a link to a boolean property) to control when the loop finishes. If left empty, it is considered to be false. This is important if the `LoopEndCondition` is set to `ConditionSatisfied` or `ConditionNotSatisfied.`

`LoopCount`: The maximum number of times to execute the loop.

`LoopEndCondition`: The condition to end the loop. Options are:

1. `LoopCountReached` - when the `LoopCount` number of loops are complete
2. `LastBlockPass` - when the loop count has been reached OR the last block passed, whichever occurs first
3. `LastBlockFail` - when the loop count has been reached OR the last block failed, whichever occurs first
4. `ConditionSatisfied` - when the loop count has been reached OR the condition specified in Condition is true, whichever occurs first
5. `ConditionNotSatisfied` - when the loop count has been reached OR the condition specified in Condition is false, whichever occurs first

4.3.2.9 Pass

4.3.2.9.1 Description

Terminate the test and write a pass status code. There can be any number of pass blocks in a test. It only has the common properties.

4.3.2.10 Fail

4.3.2.10.1 Description

Terminate the test and write a fail status code. There can be any number of fail blocks in a test. It only has the common properties.

4.3.2.11 CustomizedResult

4.3.2.11.1 Description

Terminate the test and write a custom result as status code. There can be any number of CustomizedResult blocks in a test.

4.3.2.11.1.1 Misc Category

`TestResult:` The test result to return. It can be a literal value or a link to the return value of a function, utility or python function block.

4.3.2.12 Note

4.3.2.12.1 Description

Just a delimiter in the test. It has no effect on test execution. It only has the common properties.

4.3.2.13 Press Button

4.3.2.13.1 Description

Send an IR button to the DUT under test. A single button is sent.

4.3.2.13.1.1 Inputs Category

`ButtonName`: Name of the button to send. The list shown is selected from the database based on the test case, function or utility containing the block.

`Delay:` The time to wait in ms after finishing the function before continuing with the next step.

4.3.2.13.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.14 Press Multiple Buttons

4.3.2.14.1 Description

Send multiple buttons to the DUT under test.

4.3.2.14.1.1 Inputs Category

`ButtonNames`: List of names of the button to send. This property brings up the Press Multiple Buttons Trainer.

`Delay`: The time to wait in ms after finishing the last button press and before continuing with the next step. Delays between individual buttons are set using the Press Multiple Buttons Trainer.

4.3.2.14.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.15 Comment To Log

4.3.2.15.1 Description

Write a string to the log file. If no log file is being produced then nothing is written.

4.3.2.15.1.1 Inputs Category

`StringToWrite`: The string to write to the log file.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.15.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.16 Control Power

4.3.2.16.1 Description

Alter the power state of the DUT under test.

4.3.2.16.1.1 Inputs Category

`PowerAction`: The action to take: On, Off or PowerCycle

`OffTime`: Time in seconds to wait in the off state if the action is power cycle.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.16.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.17 Configure Video

4.3.2.17.1 Description

Set the video streaming parameters.

4.3.2.17.1.1 Inputs Category

`FrameRate`: Set the frame rate in frames/second. Maximum of 25 for PAL systems, 30 for NTSC systems and 60 for HD systems. Since the Test Creator designer does not know where the final test

will be run, you can enter any number from 1 to 60. However, you may get a runtime error if you set an inappropriate value.

`MaxBitRate`: The maximum bit rate to stream in kbit/sec. Accuracy is 10% so that up to 10% more bits may be sent if there is a large degree of movement in the video.

`PictureSize`: The size of the picture. Select from a drop down list. This has no effect if the server used to run the test is an HD server using Hardware Compression. If the server has been set up to use GPU (also referred to as software) compression then the `PictureSize` will be set.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.17.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.18 Capture Screen

4.3.2.18.1 Description

Capture the video screen into a file during the test.

4.3.2.18.1.1 Inputs Category

`CropRegion`: The area to save. Value of 0,0,0,0 means capture whole screen.

`JpegQuality`: The quality of the JPEG capture in the range 1 - 100 with 100 being best. Only relevant if saving the image as JPEG.

`SaveFileName`: The file name to use. It can be an explicit name or it may contain the wild cards %d for current date and %t for the time (accurate to 1 second). The extension specifies the format. PNG, BMP,TIF and JPEG are supported. If an unknown extension is used, then PNG files will be saved.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.18.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`CapturedFileName:` The actual file name saved.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.19 OCR Text

4.3.2.19.1 Description

Capture OCR text. The text is stored in the log file but nowhere else.

4.3.2.19.1.1 Inputs Category

`OCRRegion`: The area to OCR. Value of 0,0,0,0 means capture whole screen.

`Language`: The language to use for recognizing words. A language with an asterisk requires an extra OCR license.

`LegacyInvert:` Use the old StormTest invert processing. On releases prior to 3.2, the processing filters had an option called 'Legacy Invert' but it did not actually do anything. On version 3.2, that Legacy Invert is no longer a processing filter but a flag and it now works correctly.

`ProcessingFilters`: The list of filters to use to pre-process the image. The possible filters are:

- 3x3 Gaussian Blur
- 5x5 Gaussian Blur
- Contrast alteration in 5 levels
- 3x3 Sharpen
- 5x5 Sharpen
- Invert Colors
- Color component. Any single color can be retained, along with a grayscale or monochrome conversion

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.19.1.2 Inputs - Advanced Category

`AutoCorrection`: The type of auto correction to be applied to the captured string. Options are:

- None - the text is left exactly as captured from the OCR engine.
- All Characters - the text is assumed to be all characters so digit 1 is converted to I, digit 0 to letter O, digit 5 to S.
- All Numbers - the text is assumed to be all numbers so the reverse of All Characters is performed converting letters which resemble digits into digits.

4.3.2.19.1.3 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.19.1.4 Outputs - OCR Results Category

`ObtainedString`: The string returned from the OCR engine.

4.3.2.19.1.5 Outputs - OCR Statistics Category

`AllSymbols`: Total number of symbols recognized.

`AllWords`: Total number of words recognized.

`SuspiciousSymbols`: Number of suspicious symbols detected.

`UnrecognisedSymbols`: Number of unrecognized symbols detected.

4.3.2.20 Computation

4.3.2.20.1 Description

Perform a general purpose computation which can later be used anywhere. The output is re-calculated only when the block is encountered in the flow diagram.

4.3.2.20.1.1 Inputs Category

`Expression`: The expression to evaluate. A range of operations and functions are defined for use. The final expression has a type of string, integer, floating point or boolean depending on the values and operations used. The tutorial guides you through the use of this block. The expression can be as complex as you like and may reference properties in other blocks to perform the calculation. Brackets (and) are used to define the order of evaluation. The available operations are listed below. A number means integer or floating point.

+ Add numbers or concatenate strings. If one value is a string and the other a number, then the number is considered to be a string.

- Subtract numbers

* multiply numbers

/ divide numbers

% compute the remainder after division of 2 numbers. So for example 47 % 5 is equal to 2 because 47 divided by 5 is 9 remainder 2.

& Perform a bitwise and on 2 integers. Each bit in the integer is set to one if the corresponding position in the arguments are both 1.

| Perform a bitwise or on 2 integers. Each bit in the integer is set to one if either of the bits in the corresponding position in the arguments is 1.

^ Perform a bitwise exclusive or on 2 integers. Each bit in the integer is set to one if just one of the bits in the corresponding position in the arguments is 1.

== Compare two numbers, booleans or strings for equality. The result is boolean True if the values are equal.

!= Compare two numbers, booleans or strings for equality. The result is boolean True if the values are not equal.

>= Compares two numbers or strings. The result is boolean True if the 1st number or string is greater than or equal to the second. Strings compare using ASCII ordering of letters and so may give unexpected results in non English languages.

> Compares two numbers or strings. The result is boolean True if the 1st number or string is greater than the second. Strings compare using ASCII ordering of letters and so may give unexpected results in non English languages.

<= Compares two numbers or strings. The result is boolean True if the 1st number or string is less than or equal to the second. Strings compare using ASCII ordering of letters and so may give unexpected results in non English languages.

< Compares two numbers or strings. The result is boolean True if the 1st number or string is less than the second. Strings compare using ASCII ordering of letters and so may give unexpected results in non English languages.

&& Combines two boolean values using a logical and operation. The resulting boolean is true if both the values are true.

`||` Combines two boolean values using a logical or operation. The resulting boolean is true if either or both of the values are true.

`!` Inverts a boolean value from True to False and vice versa.

`Trim()` operates on a string and removes space characters from both the front and end of the string

`ToLower(s)` operates on a string `s` and converts all letters into lower case. It does not alter characters that are not part of the A-Z of English.

`ToUpper(s)` operates on a string `s` and converts all letters into upper case. It does not alter characters that are not part of the a-z of English.

`Substring(s, loc, len)` operates on a string `s` and extracts a substring starting at position `loc` of `len` characters. The first character is at position `0` not `1`. So `Substring("hello",1,3)` will return the string `"ell"`.

`int(value)` will convert the floating point or string called `value` into an integer. A floating point number is converted by ignoring everything after the decimal point.

`double(value)` will convert the string called value into a floating point `value`.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.20.1.2 Outputs Category

`Output:` The value of the calculated expression.

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail. This block cannot fail so it is NotRun or Pass.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.21 Set Current Subslot

4.3.2.21.1 Description

Sets the current sub slot if the server is HS64. It is an error to use this block if the server where the test runs is NOT an HS64. All later audio and video related blocks will act on this sub slot on an HS64 server.

4.3.2.21.1.1 Inputs Category

`SubslotNumber`: The sub slot to set as current sub slot. Valid sub slots are 1 to 4.

4.3.2.21.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

4.3.2.22 Get Current Subslot

4.3.2.22.1 Description

Gets the current sub slot if the server is HS64. It is an error to use this block if the server where the test runs is NOT an HS64.

4.3.2.22.1.1 Inputs Category

None.

4.3.2.22.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SubslotNumber`: The current sub slot after execution of this block.

4.3.2.23 Navigate

4.3.2.23.1 Description

Navigates to a screen using the default navigator for the DUT selected for the test. This requires a navigator to be configured for the DUT.

4.3.2.23.1.1 Inputs Category

`Destination:` The name of the destination screen to navigate to.

Navigator: The Navigator to use for this block. You can only select from among the Navigators configured for the model for which this function is being designed. If not configured then the Navigator of the parent of the block (function, utility or test case) will be used.

Source: The name of the screen that should be assumed to be visible at the start of the navigation. If this is not set, then the navigator will go to the pre-defined start screen and navigate from there. This is usually necessary on first use of the navigator but after that, tests will be more efficient to specify the correct start screen.

Delay: The time to wait in ms before continuing with the next step.

4.3.2.23.1.2 Outputs Category

SavedRecordingFileName: If the rolling record buffer has been configured then this is the name of the video file that was saved.

State: The state of the block. NotRun, Pass or Fail.

SystemError: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.24 Validate Navigator

4.3.2.24.1 Description

Validates the entire navigator. This can be a very slow operation with a large navigator. It uses the default navigator for the DUT selected for the test. This requires a navigator to be configured for the DUT.

4.3.2.24.1.1 Inputs Category

Navigator: The Navigator to use for this block. You can only select from among the Navigators configured for the model for which this function is being designed. If not configured then the Navigator of the parent of the block (function, utility or test case) will be used.

Start: The name of the screen that is assumed to be on screen and from where validation should start.. If this is not set, then the navigator will go to the pre-defined start screen and validate from there. The normal operation is to leave this empty. However, if the navigator has no start screen defined (an unusual situation) then you must define a screen to start from - it is the responsibility of the prior test blocks to navigate to this screen and validate it is correct.

Delay: The time to wait in ms before continuing with the next step.

4.3.2.24.1.2 Outputs Category

SavedRecordingFileName: If the rolling record buffer has been configured then this is the name of the video file that was saved.

State: The state of the block. NotRun, Pass or Fail.

SystemError: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.25 Screen Definition

4.3.2.25.1 Description

Executes a screen definition object and return a true/false status. This block encapsulates a raw screen definition object and controls how it is executed.

4.3.2.25.1.1 Generic Category

`DesignSize:` Read Only. The design size of the screen definition object. Will be 0,0 if there is no design size in the encapsulated screen definition object.

4.3.2.25.1.2 Inputs Category

ScreenDefinition: The object. This can be set from a file. You cannot design a screen definition object from this block, you must use one previously designed with the Screen Definition Editor.

`Delay in Milliseconds`: The time to wait before continuing with the next step.

4.3.2.25.1.3 Inputs - Advanced Category

`JpegQuality`: The quality of the JPEG file in the range 1 - 100 with 100 being best. Only relevant if saving the image as JPEG.

`WaitAction`: How the block should wait. Options are:

- `NoWait` - the block does one execution of the screen definition object and continues
- `WaitForPass` - the block does multiple executions of the screen definition object until it gets a pass state (or a timeout)
- `WaitForFail` - the block does multiple executions of the screen definition object until it gets a fail state (or a timeout)

`MaxWaitTime`: The time to wait for the screen definition object to meet its condition (see WaitAction above)

`WaitTimeGap`: The time between successive comparisons when waiting. If set too low, it will impose a heavier processing load on the client machine running the test.

`SaveScreenCapture`: Determine whether the block should save a screen grab. Options are:

- `Never` - never saves a screen grab
- `Always` - always saves a screen grab
- `IfResultPass` - only if the execution of the screen definition succeeded
- `IfResultFail` - only if the execution of the screen definition failed.

`SaveFileName`: The file name to use to save the screen from the screen definition object. It can be an explicit name or it may contain the wild cards %d for current date and %t for the time (accurate to 1 second). The extension specifies the format. PNG, BMP,TIF and JPEG are supported. If an unknown extension is used, then PNG files will be saved.

4.3.2.25.1.4 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`CapturedFileName:` The name of the file if `SaveScreenCapture` and the final status combine to need a file to be saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.26 Compare Image

4.3.2.26.1 Description

Compare image and return a true/false status. It captures an image and then does the comparison.

4.3.2.26.1.1 Inputs Category

`CompareRegion`: The area to compare. Value of 0,0,0,0 means capture whole screen.

`RefFileName`: Name of the reference file. It is local to the owning test case/utility.

`SimilarityPercentage`: The degree of sameness of the 2 images.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.26.1.2 Inputs - Advanced Category

`IconMode`: When true, the reference image is treated as an icon and it is compared with an equivalent sized portion of the video - the compare region is ignored.

`RegionOffset`: Offset to apply to the reference icon before comparing. So an offset of 100,100 would compare the icon in the file with the video at 100,100 and size of the file.

`WaitAction`: How the block should wait. Options are:

- `NoWait` - the block does one comparison and continues
- `WaitForPass` - the block does multiple comparisons until it gets a pass state (or a timeout)
- `WaitForFail` - the block does multiple comparisons until it gets a fail state (or a timeout)

`MaxWaitTime`: The time to wait for the image compare to meet its condition (see WaitAction above)

`WaitTimeGap`: The time between successive comparisons when waiting. If set too low, it will impose a heavier processing load on the client machine running the test.

`SaveScreenCapture`: Determine whether the block should save a screen grab. Options are:

- `Never` - never saves a screen grab

- `Always` - always saves a screen grab
- `IfResultPass` - only if the comparison succeeded
- `IfResultFail` - only if the comparison failed.

4.3.2.26.1.3 Outputs Category

`ImageSimilarity`: The actual similarity between the images.

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.27 Compare Color

4.3.2.27.1 Description

Compare video for a region of a specific color

4.3.2.27.1.1 Inputs Category

`CompareRegion`: The area to compare. Value of 0,0,0,0 means capture whole screen.

`Flatness`: How much color variation exists. This is calculated as the standard deviation of each component and then expressed as a percentage where a value of 100% is a standard deviation of <1, 99% is a standard deviation of <2 etc.

`ReferenceColor`: The color to compare against. The alpha component is ignored so it should be set to 100%.

`ToleranceRed`,Blue and Green: The limit either side of the standard color that is allowed for the color to be considered a match. The value compared is the average (arithmetic mean) of each component over the region.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.27.1.2 Inputs - Advanced Category

`PeakError`: The % of pixels that are allowed to exceed the tolerance and still be considered a match. It is quite possible to get an image where the average color is the desired color, the flatness is as desired but there are a small number of pixels dramatically away from the average. This is the peak error - depending on the image and DUT, you may wish to set this to a value other than 0 (the default)

`WaitAction`: How the block should wait. Options are:

- `NoWait` - the block does one comparison and continues
- `WaitForPass` - the block does multiple comparisons until it gets a pass state (or a timeout)

- `WaitForFail` - the block does multiple comparisons until it gets a fail state (or a timeout)

`MaxWaitTime`: The time to wait for the image compare to meet its condition (see WaitAction above)

`WaitTimeGap`: The time between successive comparisons when waiting. If set too low, it will impose a heavier processing load on the client machine running the test.

`SaveScreenCapture`: Determine whether the block should save a screen grab. Options are:

- `Never` - never saves a screen grab
- `Always` - always saves a screen grab
- `IfResultPass` - only if the comparison succeeded
- `IfResultFail` - only if the comparison failed.

4.3.2.27.1.3 Outputs Category

`DetectedColor`: The actual color detected.

`DetectedFlatness`: The actual flatness detected.

`DetectedPeakError`: The actual peak error detected.

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.28 Detect Motion

4.3.2.28.1 Description

Compare video for a region of a motion.

4.3.2.28.1.1 Inputs Category

`DetectRegion`: The area to compare. Value of 0,0,0,0 means whole screen.

`Percentage`: The degree of motion allowed and still consider a static picture.

`Timeout`: Time to wait for motion.

`SaveScreenCapture`: Determine whether the block should save a screen grab. Options are:

- `Never` - never saves a screen grab
- `Always` - always saves a screen grab
- `IfResultPass` - only if the comparison succeeded
- `IfResultFail` - only if the comparison failed.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.28.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.29 Detect Audio

4.3.2.29.1 Description

Compare audio for audio presence or silence.

4.3.2.29.1.1 Inputs Category

`Threshold`: The value in dBu above which audio is considered to be really audio. Below this limit is "silence". The audio is measured over 128 ms across the bandwidth 20 Hz to 8 kHz. Unless the input is a pure continuous tone, the actual levels detected could be quite low due to the 128 ms measurement period.

`MaximumWaitTime`: Time to wait for audio or silence. This is ignored if the `Wait Action` is `NoWait`.

`WaitAction`: How the block should wait. Options are:

- `NoWait` - the block does one comparison and continues
- `WaitForAudio` - the block waits until the audio level exceeds the threshold (or a timeout)
- `WaitForSilence` - the block waits until the audio level is below the threshold (or a timeout)

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.29.1.2 Outputs Category

`DetectedAudioLevel:` The audio level detected in dBu. If the block failed on a time out, this level is the closest it got to passing; for a wait for audio, this is the highest level detected. For a wait for silence, this is the lowest level detected.

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.30 A/V Analysis

4.3.2.30.1 Description

Perform audio analysis or video analysis or both on an input. This is a complex block. Both the audio and video analyses rely upon a predefined video stream being decoded by the DUT. Customers can be helped in generating such a stream - please contact StormTest support at stormtest-

support@s3group.com. The block runs for a user defined time and provides a pass or fail indication. Some of the settings can be adjusted based on DUT model using the Admin Tools application of StormTest - these calibration values are stored in the central database and used by the A/V analysis block. Typically each distinct model of DUT will have different audio levels or color balance and so will need calibrating.

4.3.2.30.1.1 Inputs Category

`AnalysisRunTime`: Time to run the analysis in seconds.

`AnalysisType:` The type of analysis to run. Options are:

- `Both` - Run audio and video analysis in parallel.
- `Audio`- Run just audio analysis.
- `Video` - Run just video analysis.

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.30.1.2 Inputs - Video Analysis Category

`FrameRate`: The frame rate to use for video analysis. The video analysis is a high load on the server and it is not possible for the server to run 16 video analyses at 25 frames per second at the same time. In fact the total frames per second of all simultaneous video analyses must not exceed 100. A value of 5 frames per second gives good results and assures tests that they can always run video analysis.

4.3.2.30.1.3 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.30.1.4 Outputs - Audio Analysis Category

`ImpulseResult:` The result of the audio impulse test. This looks for impulses in the time domain which exceed a specific threshold set during calibration. The value will be `True` if all samples are below the threshold.

`MeanToneResult:` The result of the audio mean tone amplitude test. This looks to ensure the expected tone exceeds the expected tone on average across multiple samples. Used in conjunction with the Tone Result, it determines whether the tone is totally missing (Mean Tone Result `False`) or has occasional drop outs (Mean Tone Result `Pass` but Tone Result `Fail`).

`OutOfBandResults:` The result of the audio out of band amplitude test. This looks to ensure there is no signal except at the expected tone frequency. The value will be `True` if the total signal apart from the expected tone is below the calibrated threshold.

`SpuriousToneResult`: The result of the audio spurious tone test. This detects whether there is any tone above a specified signal at a pre-defined set of frequencies. It is a finer grained version of

the out of band test (the out of band test measures total power but the spurious tone looks for peaks across the range). The value will be `True` if all tones are below the calibrated threshold.

`SubBandResult:` The result of the audio sub band test. This detects audio across a fixed number of equally spaced bands, ensuring that the power in each band is below the specified threshold. This is used to ensure the noise level remains below a threshold across the entire spectrum. The value will be `True` if all bands have a power below the calibrated threshold.

`ToneResult:` The result of the audio tone test. This looks to ensure the expected tone is above the expected amplitude on all samples. The value will be `True` if all audio analyses during the run detected a tone at the right frequency and amplitude.

4.3.2.30.1.5 Outputs - Video Analysis Category

`QualityResult:` The result of the video quality analysis. This looks to ensure the colors are correct during the analysis, there is no macro blocking or picture break up - the perceived visual quality. The value will be `True` if there are no detectable defects in the quality of the video.

`TimingResult:` The result of the video timing analysis. This looks to ensure there is no frozen video or other timing related issues with the video. The value will be `True` if there are no detectable defects in the timing of the video.

4.3.2.31 OCR Compare String

4.3.2.31.1 Description

Capture OCR text and compare against a string.

4.3.2.31.1.1 Inputs Category

`ExpectedString`: The string to compare against.

`OCRRegion`: The area to OCR. Value of 0,0,0,0 means capture whole screen.

`Language`: The language to use for recognizing words. A language with an asterisk requires an extra OCR license.

`LegacyInvert:` Use the old StormTest invert processing. On releases prior to 3.2, the processing filters had an option called 'Legacy Invert' but it did not actually do anything. On version 3.2, that Legacy Invert is no longer a processing filter but a flag and it now works correctly.

`ProcessingFilters`: The list of filters to use to pre-process the image. The possible filters are:

- 3x3 Gaussian Blur
- 5x5 Gaussian Blur
- Contrast alteration in 5 levels
- 3x3 Sharpen
- 5x5 Sharpen
- Invert Colors
- Color component. Any single color can be retained, along with a grayscale or monochrome conversion

`Delay`: The time to wait in ms before continuing with the next step.

4.3.2.31.1.2 Inputs - Advanced Category

`AutoCorrection`: The type of auto correction to be applied to the captured string. Options are:

- None - the text is left exactly as captured from the OCR engine.
- All Characters - the text is assumed to be all characters so digit 1 is converted to I, digit 0 to letter O, digit 5 to S.
- All Numbers - the text is assumed to be all numbers so the reverse of All Characters is performed converting letters which resemble digits into digits.

`ResultDistance:` Maximum number of unmatched characters permitted for which the block considers the string a match. The algorithm used is the Levenshtein distance algorithm. Note that the auto correction is applied before the Levenshtein distance algorithm is calculated.

`MaximumWaitTime`: Time to wait for OCR to match. This is ignored if the `Wait Action` is `NoWait`.

`WaitAction`: How the block should wait. Options are:

- `NoWait` - the block does one OCR comparison and continues
- `WaitForPass` - the block waits until the OCR string matches the expected string (or a timeout)
- `WaitForFail` - the block waits until the OCR string does NOT match the expected string (or a timeout)

4.3.2.31.1.3 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

4.3.2.31.1.4 Outputs - OCR Results Category

`ObtainedString`: The string returned from the OCR engine.

4.3.2.31.1.5 Outputs - OCR Statistics Category

`AllSymbols`: Total number of symbols recognized.

`AllWords`: Total number of words recognized.

`SuspiciousSymbols`: Number of suspicious symbols detected.

`UnrecognisedSymbols`: Number of unrecognized symbols detected.

4.3.2.32 Calibrate Video Offset

4.3.2.32.1 Description

Calibrate the DUT video offset. Some DUTs have a slightly different offset compared to each other. This can make the image comparisons difficult across models that are the same if the same reference image is used. You could change the image, capturing it each time on the correct box. Or you can set the video offset. This is stored in the database but can be calculated using this block. You need a reference image and previous blocks must drive the DUT to the screen which should match the reference image.

4.3.2.32.1.1 Inputs Category

`Delay`: The time to wait in ms before continuing with the next step.

`MaximumSearchX,Y`: The limits of the video offset that are possible. Large values here will slow down the block.

`RefFileName`: Name of the reference file. It is local to the owning test case/utility.

`Threshold`: Degree of match to be considered acceptable.

`SaveToDatabase`: True if you want to update the database, false if just use the result in this test case.

4.3.2.32.1.2 Outputs Category

`SavedRecordingFileName:` If the rolling record buffer has been configured then this is the name of the video file that was saved.

`State`: The state of the block. NotRun, Pass or Fail.

`SystemError`: True if a system type error occurred, eg a Python exception or a hardware failure.

`CalibrationError`: The worst error detected.

`DetectedOffset`: The offset that was detected as OK, not valid if block fails.

5 Navigator

5.1 Concepts

5.1.1 Navigator Screens

The navigation map is a collection of screens linked together by links. The Navigator has 3 types of screen, each treated slightly differently. They are set from the properties grid - the property called screen type.

5.1.1.1 Screen In Map

The majority of screens in the navigation map are of type `ScreenInMap` and this is the default. It can be thought of as a "normal" screen. It is possible and often useful to make a navigation map entirely of such screens.

5.1.1.2 Start Screen

You can define one and only one start screen. A start screen is a screen which can be reached from any other screen in the navigation map and also from any screen outside of the navigation map. This means that whatever state the DUT is in, you can use the properties of the Start Screen to reach the start screen.

It has some unique properties which no other screen has.

5.1.1.2.1 Power Cycle

The power cycle allows you to enable a power off followed by a power on to reach the start screen. Some DUTs may not have any other means to arrive at a known state from any other state. For performance reasons, you should try to avoid using this feature as it is slow. To enable power cycle, the `power cycle` property should be set to true. The time that the Navigator will wait between power off and powering on is set by the `Power Off Time` property. After power on, Navigator will wait for the `Power Delay` before continuing.

5.1.1.2.2 Key Presses

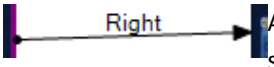
You can select a series of key presses to bring the DUT to the start screen. If the power cycle is specified then the key presses will be used after the `Power Delay` time has expired. Multiple keys can be used and configured by the Multiple Keys Trainer. Normally links have key presses, not screens. The start screen is the one exception.

5.1.1.3 Ignored Screen

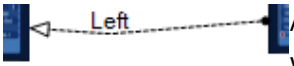
An ignored screen is for user convenience and is ignored in the navigation map. You can not navigate to it. It is also ignored in validation. This can be useful during development of a navigation map where maybe some screens are only partly know or the links are incomplete.

5.1.2 Links

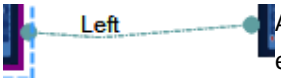
Links connect navigator screens together. The link graphic tells you information about the link.



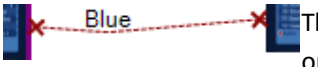
A link which has been validated and passed. The Navigator can use the specified key press to go between the screns that are linked. It is black with filled arrow head.



A link with open arrow and dotted line has not been validated. It might work or it might not.



A link is being checked during validation. It is blue with filled dots on each end.



The link has failed validation. It is red with cross marks each end. This only occurs after a validation has completed and is cleared when the map is re-validated. It is to help you identify the failure point. ('Blue' here refers to the key not the color of the link)

An unconnected link is shown in light blue with solid arrow head.

Links can be selected and connected and disconnected as you need. If you double click a link, the Press Multiple Button trainer appears so that you can select the key presses.

5.1.2.1 Retries and Dynamic Menus

Each link has a retry count which defaults to 0. This value can be changed. It is used in two separate situations.

5.1.2.1.1 Avoiding Missed Keys

If the DUT is slightly unreliable and has a habit of missing keys then the overall reliability of the navigation can be improved by setting a retry count greater than 0. The navigator will send the key and test the screen. If the screen fails then the navigator will send the key again and test again. This process will continue up to the retry count. The maximum number of keys sent is 1 + retry count (assuming a single key on a link).

5.1.2.1.2 Dynamic Menus

Some DUTs have dynamic menus where it is not possible to define in advance how to reach a specific screen. An example would be where to reach screen 'Foo', the user would use the right arrow until the title matched 'Foo'. At some times of day or when other actions had taken place on the DUT this may require 3 key presses or 4 key presses.

Setting a retry count here will solve the problem. The navigator will send the right arrow until the screen matches the test criteria for the specified screen.

5.1.2.2 Short Cut Link

You can enter a shortcut link from the ribbon. It initially appears as a simple icon. You must then drag a link from this icon to another screen. You can then define the button press. Once this happens, it means that the specified button will always go the linked screen from any other screen in the navigation map. This is similar to the start screen concept except that the shortcut link is only guaranteed to work from within the navigation map concerned.

5.1.3 Algorithms Used

This section describes the algorithms used by StormTest Navigator. It is not essential to understand these to use the navigator but it may help to explain the behavior at times.

5.1.3.1 Navigation

When asked to navigate between two screens, the navigator uses Dijkstra's shortest path algorithm, a well known algorithm in computer science to find the shortest path between two points.

5.1.3.1.1 Link Distance

To calculate this, the navigator needs to know what is meant by 'shortest'. Navigator calculates a time for each link and considers the shortest path to be that with the lowest time. In calculating the time to travel from screen A to screen B, it calculates the delays of all the key presses and then adds in an estimate of time for each test of the destination screen.

5.1.3.1.2 Effect of Start Screen

The start screen may have a power cycle defined. This is slow. The navigation is smart and will avoid using the start screen if at all possible. If the only way from screen D to F is via the start screen and it has a power cycle defined on the start screen then that will be used.

5.1.3.2 Validation

Validation is very different from navigation. In validation, the navigator attempts to traverse every possible link. It uses a solution to the Chinese Postman Problem to calculate the optimum path through the map. It is important to understand that it validates links not screens.

The terminology is potentially confusing - when editing a screen you can validate the screen: this checks that the colors, images etc defined will work for the screen and successfully detect when the screen is displayed.

The navigator validation checks each link works. This means that the same screen will be visited many times. Consider a very simple map with 3 screens: A, B and C. There are links from A-B, A-C, B-A, C-A and B-C. If it starts at A it must go to B but also sometime go back to A to check A-C. So A is visited at least twice. Depending on the path, it may need to be visited more often.

5.1.3.2.1 Start Screen and Short Cuts

These two items present some confusion. Since you can reach the start screen from anywhere, there is, in effect, a link from all screens to the start screen. This is not drawn as that would be confusing. But during validation, those links are tested - the navigator makes sure you really can get to the start screen from anywhere. If the start screen has a power cycle then that means the validation will be slow as the DUT is power cycled from each screen. A good reason to avoid using the power cycle feature.

Likewise a short cut has, in effect, links from every screen to the destination screen but these are not shown as that is too confusing. Validation will involve navigating to each screen, pressing the short cut key and checking that the destination is reached. A map with several short cuts and a lot of screens may take a long time to validate.

5.1.3.3 Strongly Connected Map

To validate the map must be what is known as 'strongly connected'. There must be a path from every screen to every other screen in the map. Consider a very simple map showing the problem. It has 3 screens A, B and C. Links have been defined from A to B and B to C. This cannot be validated - there is no path from C to B or from B to A.

If you make a map like this an error message is produced complaining about a lack of strongly connected map. There are various solutions. Marking A as a start screen would solve the problem: now the path to A from C is to use the start screen sequence. And from C to B is the start sequence and then from A to B.

Another possible problem is an orphaned screen: a screen entered without any links to it. A solution to this is to mark it "ignored" This technique can also be used in the prior case (but as both B and C would need to be ignored the map would have just one screen from a validation point of view).

5.1.4 Resources

Navigator, Test Creator and Screen definitions can use resources for the tests on a screen, in a test case or as part of a screen definition. These allow sharing of parameters between different parts of a user's test system and thus centralised storage. It allows you to make changes across a large number of different elements in one place. Resources can be fully edited and managed from the Resource Editor applet. Resources can be used from a Python script as well.

5.1.4.1 Types of Resource

StormTest stores the following types of resource:

- Region - this is simply a rectangle and defined by 4 numbers.
- Image - this is an image along with the intended size of the image.
- Color - this is the RGB color, tolerance to use when matching, flatness and peak error. All the values that are needed as input to a color comparison.
- String - this is just a string of characters.
- Screen - this is a screen definition object and combines multiple test areas into a single logical unit.

All of these resources are stored in the StormTest database. Each resource has a name and the resources can be organized in a hierarchical manner similar to folders on a disk. As images can be quite big, care should be taken not to store more than are needed.

5.1.4.2 Using a Resource

In many places resources can be used. For instance, instead of specifying the region coordinates directly a region resource can be used. The actual values of coordinates are determined at run time by reading the database.

Images deserve a special mention here. The Navigator assumes the image stored by name matches the entire region of interest in the test. It has no concept of using a full screen as a source image and specifying the sub region of interest. (Both Test Creator and Python scripts can work this way but not the Navigator).

5.1.4.3 Model Specific Resources

Version 3.2 of StormTest introduced the concept of model specific resources. This increases the power of resources by allowing even more decoupling between the functional part of a test or navigator and the visual attributes of a screen.

When you create a resource, you initially create a 'generic' resource - all resources created prior to release 3.2 are generic. Once this has been created you can add a variant - this is configured for one of more DUT models (the models are defined in the StormTest Admin Console). You can assign any number of models to a variant.

When a resource is used, the StormTest system examines the DUT model in the slot where the resource is to be used - this is true from Python, in Test Creator, in Navigator and when using screen definition objects. If a variant of the resource for that model exists, then its values are used. If not then the generic version is used.

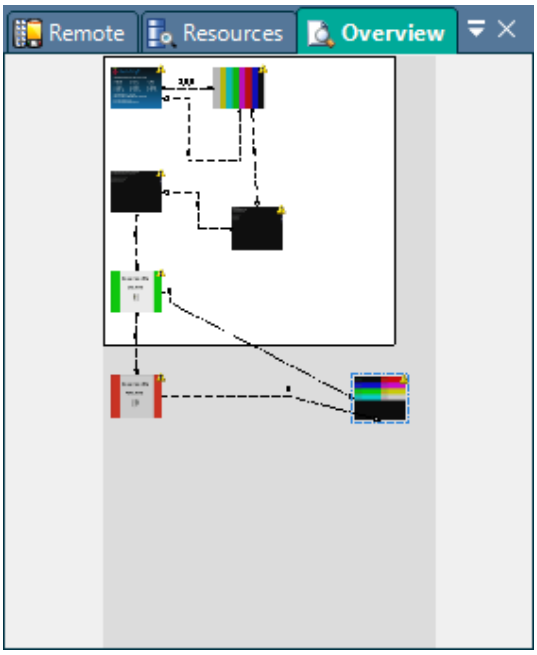
As an example, you could define a named region called '/myregions/title' which has coordinates (0,0,100,20). This is the generic version. If you now have a DUT, with a model name of 'NewBox' which has the title in a different location, you can create a variant of '/myregions/title' - it has no separate name. You can then assign, for example a value of (100,0,100,20) to this variant's region.

Now when you write a test or design a navigator you can use the region '/myregions/title'. If you run the test on a slot with 'NewBox' in it then the region (100,0,100,20) is used - for all other tests on other slots, the generic (0,0,100,20) will be used.

5.2 Editor

5.2.1 Overview Panel

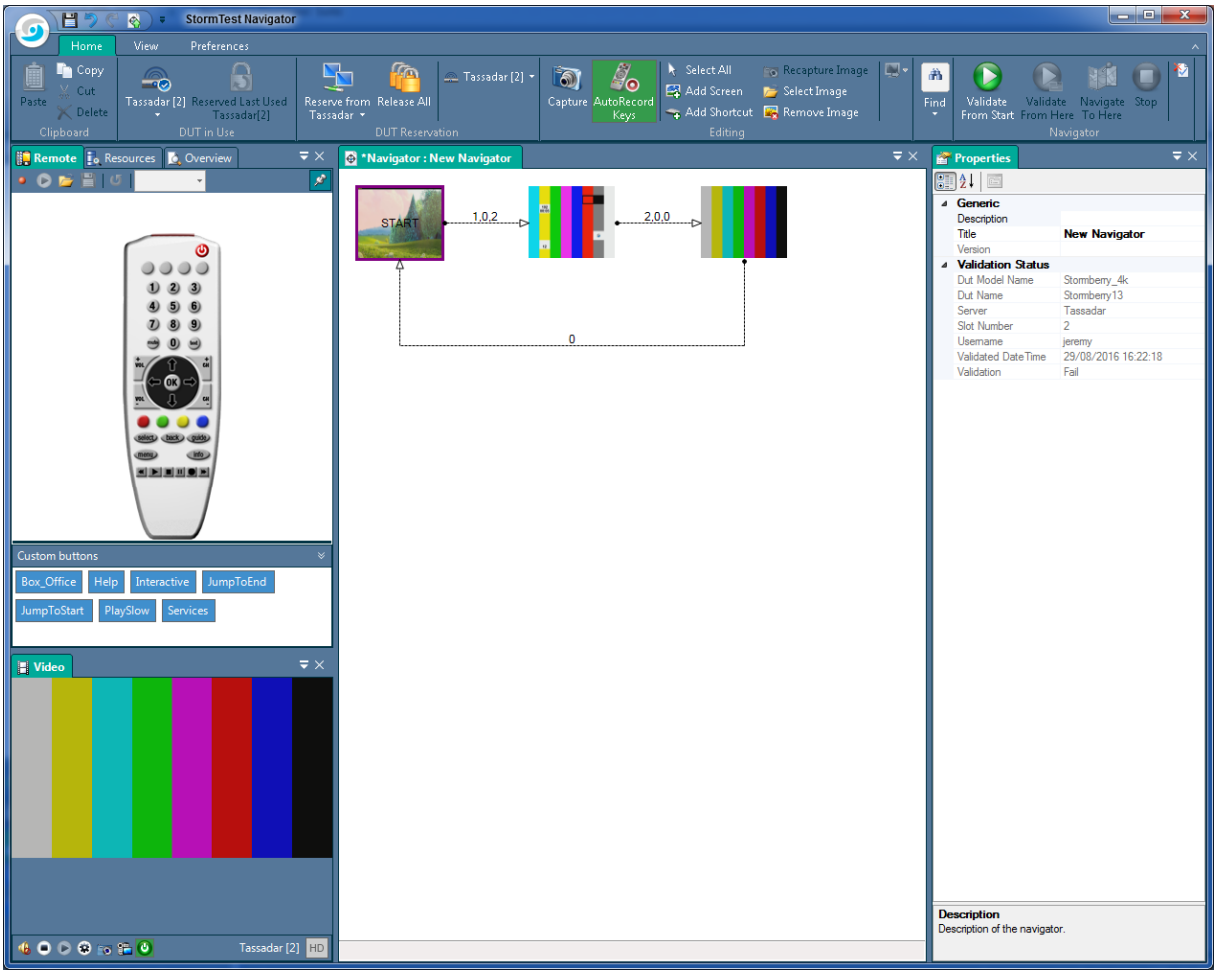
The overview panel shows the entire navigator scaled to the window with a highlighted area showing the amount visible in the main editor. You can drag the highlight around for easier navigation.



Use the mouse to move the white area and the main editor will follow.

5.2.2 Navigator Main Screen

The navigator main screen is where you do most of the work. The default layout is:



5.2.2.1 Ribbon

The ribbon at the top of the screen contains the commands needed for creating and editing navigators.

5.2.2.2 Properties Panel

The properties panel, shown on the right, shows you the properties of the selected item. The view above shows the properties for the navigator.

5.2.2.3 Navigator View

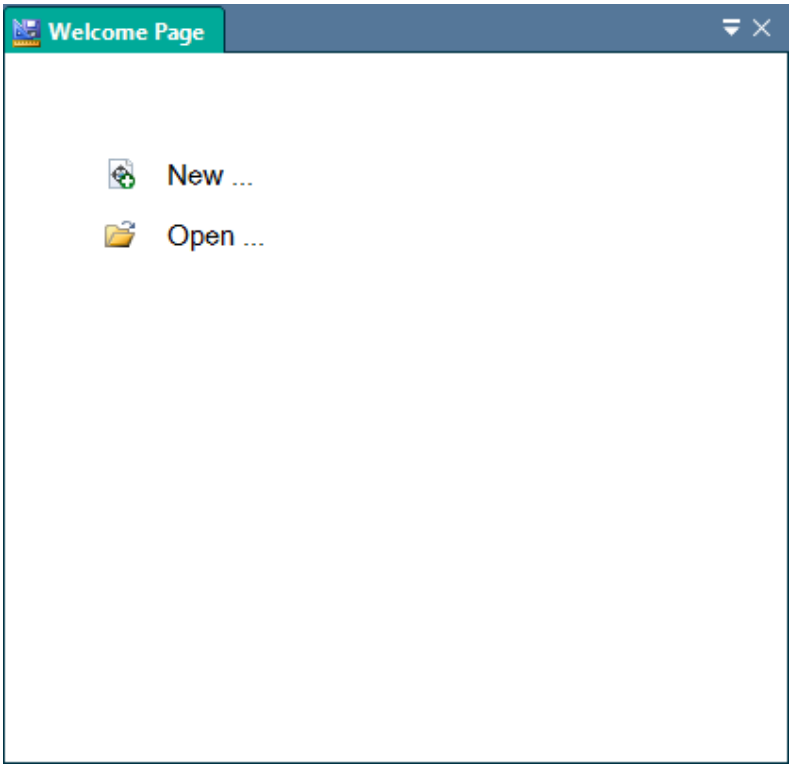
The central panel above shows the overall navigator - blank initially. As you add screens, this shows the overall map of the DUT screens. Here you can add screens and links as well as edit them.

5.2.2.4 Control Panels

The panels on the left show the control panels: the remote control, the video, resource and an overview of the navigator.

5.2.3 Welcome Page

If you close a navigator using the 'X' on the central panel, you will see the welcome page:



For here you can open a navigator or create a new one. The already open window will be used. This allows you to work in a single window with a single DUT reserved and open multiple navigators. When you use the quick access toolbar to open a Navigator a new window is opened.

5.2.4 Navigator Ribbon

The main ribbon at the top of the navigator (while viewing the navigation map) has 3 tabs.

5.2.4.1 Home Tab

The home tab is:

The controls are enabled and disabled as needed. The commands are grouped and a brief overview of each is:

5.2.4.1.1 Clipboard

This group deals with the Windows clipboard

 Paste. You can only paste items that you have selected to the main drawing. Ctrl-V also works

 Copy. Items copied to the clipboard can be pasted as text or an image to other Windows packages. As text, you see the raw XML.

 Cut. Copies and then deletes an item or group of items.

 Delete. Deletes the selected screens or links.

5.2.4.1.2 DUT in Use

This group deals with the DUT in use by the Navigator.

 This shows you the DUT (by server[slot]) of the DUT used by the Navigator for developing it. If you have reserved more than one DUT, then you can change the DUT in use by clicking this and selecting an alternative DUT.

 Reserve Last Used. If you have not reserved a DUT via the main ribbon, this allows you to reserve the DUT that was last used for validation (even if that failed). It will be disabled if the Navigator has never run the validation routine.

5.2.4.1.3 DUT Reservation

This group is the same as the main ribbon with the same functionality.

5.2.4.1.4 Editing

This group is used during the creation and editing of the Navigator.

 captures a screen shot from the video and makes a new screen in the navigator map. If auto capture is enabled then it also puts a link from the previous screen to the newly captured screen and puts in the button presses that you have used.

 A check button that selects auto capture mode. In auto capture mode, the IR keys that you press are recorded and automatically added to links on each capture. Once the main screens of the navigation map have been captured, you may wish to turn this off. If left on, then the keys needed to backtrack in the map will be recorded.

 Select all items in the map (Ctrl+A also works)

 Add a screen. The screen will be empty without an image. You can add an image from a file or later by capturing from a DUT.

 Add a shortcut link.

 Recapture an image. This will take a screen shot of the current video in the DUT and put it into the selected screen, replacing the previous image.

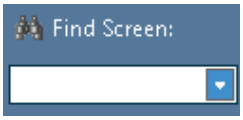
 Select an image from a file for a screen. This will replace the image in the selected screen. Particularly useful if you have the original images for the DUT in high quality graphics. Note that the size should match the StormTest Capture size (704 x 576/480 for SD and the expected HD resolution for HD systems).

 Delete the image from the screen. The screen will be blank afterwards.

 Convert one or more screens to a screen definition objects. A menu appears offering a choice of converting the screens to Named Screen resources or to embedded screens. An embedded screen will be a group of all existing tests in the Navigator screen as a single screen definition object. This gives the advantages of a screen definition object but without the central management of resources.

5.2.4.1.5 Find

This group is used to find a navigator screen. It has a simple text box to enter the screen name:



Simply type the name of the screen - it will auto complete from the list of screens in the current navigator. The screen will be selected and scrolled into view when you use the 'Enter' key on the keyboard. Using Ctrl-F will bring the control into focus even if another part of the ribbon is active.

5.2.4.1.6 Navigator

This group deals with the navigation map as a single entity.

 Validate navigation from start. This requires a start screen to have been defined. Validation means every link in the map will be traversed at least once and the destination screen checked that it meets the validation criteria. The validation algorithm is known as the Chinese Postman problem. Please read the detailed description.

Validation criteria are defined by opening the screen (double click, Enter key or right click and select from menu) and then using the mouse to define a region and select whether that should be an image comparison, color comparison, motion detection or an OCR operation. Multiple regions can be defined per screen. Collectively these are the 'validation criteria' that are used by navigator to determine whether the screen on the DUT is the expected screen.

 Validate from the current screen. The validation is similar to above except that it makes the assumption that the DUT is showing the selected screen and validates. If you have not specified a start screen then you must select a screen and use Validate from here.

 Navigate to the selected screen. The navigator uses the start screen and then directs the DUT to the selected screen. This is intended as an aid during development in that you don't have to

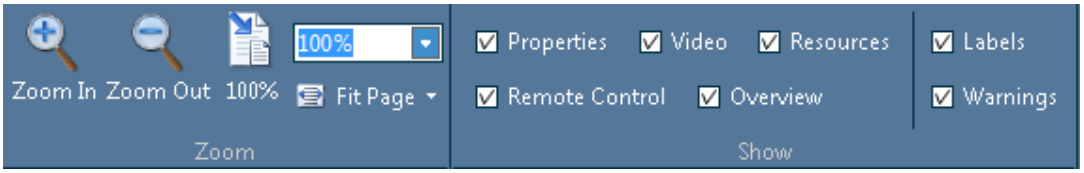
remember how to get to every screen.

 Stop validation - used during a lengthy validation process to stop.

 Clear the validation status. After a validation is run (or partly run), the Navigator stores which links have been validated and which have not. Editing the map will clear any affected link. However, if you wish to re-run the entire validation, use this button.

5.2.4.2 View Tab

The second tab is the view tab:



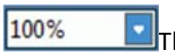
5.2.4.2.1 Zoom

This group deals with the zooming of the drawing.

 Zoom in - makes the drawing larger, showing more detail.

 Zoom out - makes the drawing smaller, showing less detail but more of the drawing

 100% - makes the drawing 1:1 scaled so that you see the actual captured resolution of each screen (these are thumbnails, not full size)



The current zoom level is shown as a % of full size and can be changed directly

 Fit the drawing. 3 options are available: Fit to Page, Fit to Width, Fit to Height.

5.2.4.3 Show

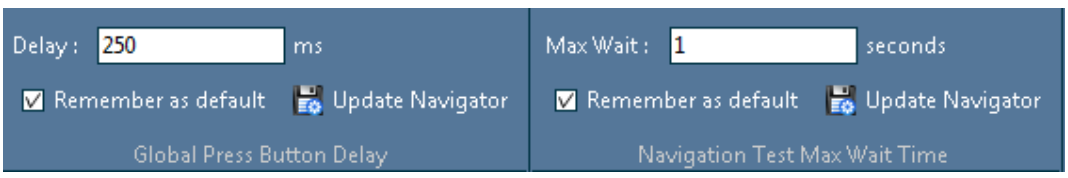
This group of check boxes shows/hides the panels. The right hand column controls how the main Navigator is drawn.

Show Labels - when checked, the button names are shown on each link

Show Warnings - when checked, the screens without validation criteria are shown with a warning triangle in top right corner.

5.2.4.4 Preferences Tab

The preferences tab is:



5.2.4.4.1 Global Press Button Delay

This controls the delay after every button press. Whenever you create a multiple button sequence by entering the buttons directly in the multiple buttons trainer within either Navigator or Test Creator a delay is used. You can set the default to use here. You can also set the default from within Test Creator.

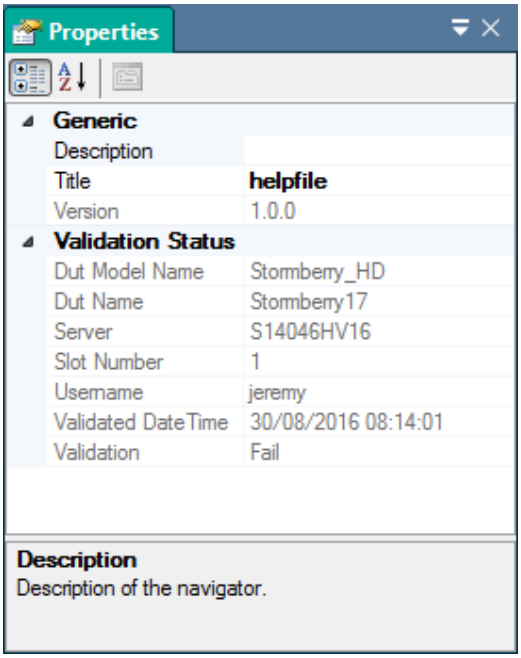
The delay can be made persistent so that it will apply next time Developer Suite starts by ensuring 'Remember as default' is checked. If you check 'Update Navigator' then all button delays on the current navigator will be updated.

5.2.4.4.2 Navigation Test Wait Time

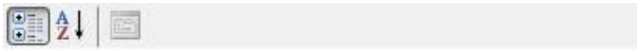
This controls the time at the end of each test on a screen within the Navigator. A timeout is needed on each test. The default timeout can be set here. The currently open Navigator can be updated by clicking the Update Navigator button.

5.2.5 Properties Panel

The Properties panel is a standard properties grid showing the properties of the selected object. In Navigator and Screen Definition Editor, there are 2 properties grids. The upper is for the navigator, screen, link or screen definition depending on selection and the lower is for the selected test. This lower grid is tabbed with one tab per test. If there are no tests defined then this lower grid wil be hidden. In Test Creator there is just one properties grid for the test case or selected block.

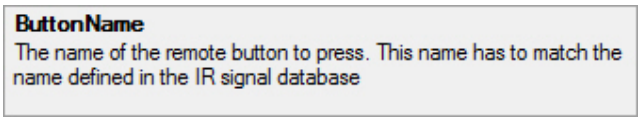


The upper panel of each grid controls how the properties are displayed:



You can sort the properties by category (default) or alphabetically (by clicking the AZ button).

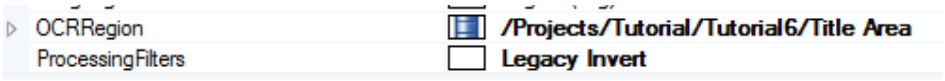
The lower panel gives a brief overview of the selected property:



In this case, the ButtonName property is described.

The middle panel is the properties of the block. Any property in gray cannot be edited, for instance the outputs are always gray as they cannot be altered by you.

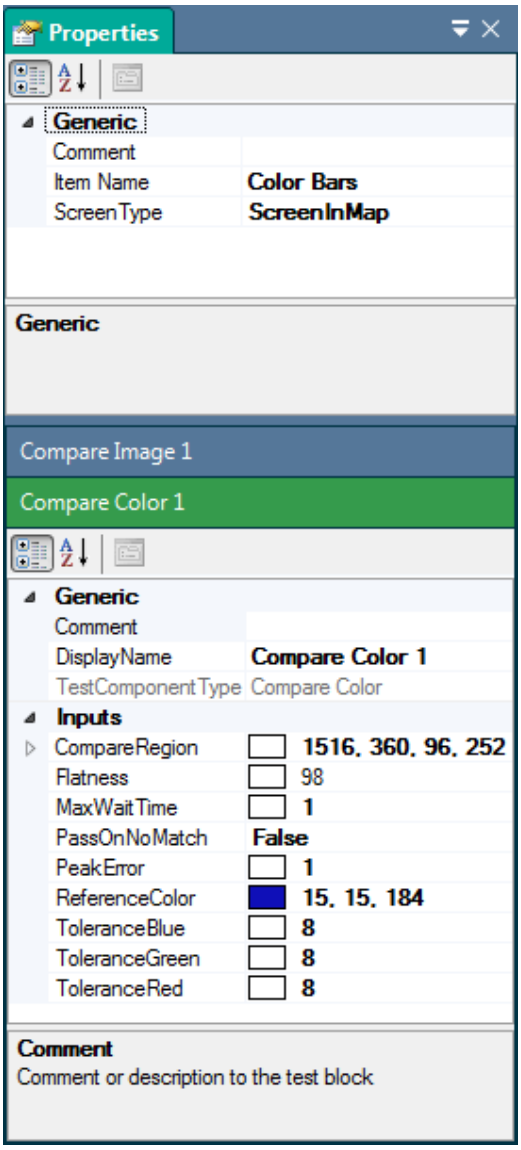
Some properties can use named resources. These are stored centrally in the database and the value is looked up at run time.



The OCRRegion above is a resource called `Title Area` in the folder `Projects/Tutorial/Tutorial6`. Reference links allow for dynamic data within a single test case but named resources allow separate test cases to use the same definitions for common items.

5.2.5.1 Screen Properties

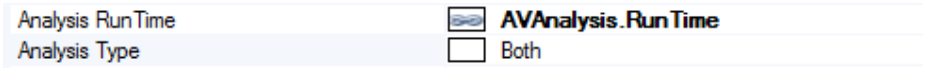
When you select the properties of a screen, you see two sets of properties. The upper panel refers to the screen and the lower is tabbed with one tab per screen region. In the screen definition editor, the upper grid is always visible and reflects the properties of the screen definition object.



Click on the title bars to change the active tab (for example, clicking on 'Compare Image 1' above will show the properties of the Compare Image region on the screen).

5.2.5.2 Test Creator Ref Links

Within Test Creator, most properties have the option of being linked to another block within the test case, utility or function:



The Analysis Run Time above (an input) is linked to the RunTime property of the block called AVAnalysis. This is indicated by the blue link symbol. The white empty rectangle by the Analysis Type indicates that it is not linked and the property is unique to the block shown. The tutorial on reference links shows how to setup reference links.

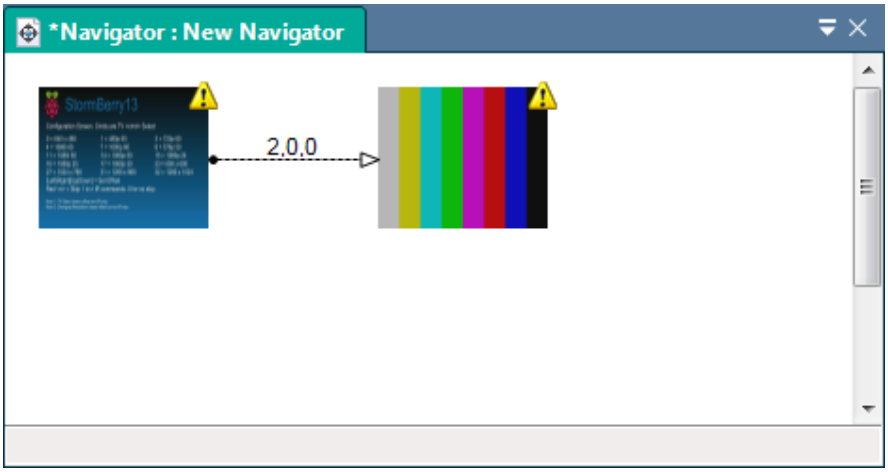
This option is not availble from Navigator or the screen definition editor.

5.2.6 Adding Screens

To add a screen to a map, simply drive the DUT to the desired screen and click the capture button on the ribbon. A new screen is added to the navigator map. If the auto record mode has been set on the ribbon, then a link will be added between the previous screen and the screen just captured with the keys needed to traverse to it.

5.2.6.1 Example

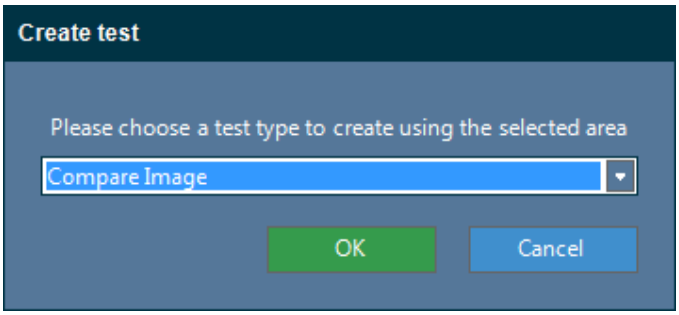
Start with an empty navigator and the default auto record mode. Press a button on the remote control and then click the capture button. A single screen appears. Now press another button on remote control and the capture button again. A very simple navigator is produced:



The yellow warning icon indicates that there are no validation criteria for the screen defined. This is not good because StormTest Navigator will not be able to check that the right screen is visible. It will take on trust the reliability of the DUT to respond to IR commands.

5.2.6.2 Adding Validation Criteria

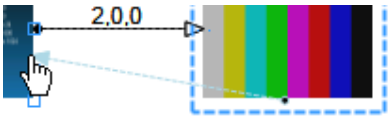
To add criteria for validation to a screen, double click the screen or right mouse click and select open. The use the mouse to select a region (click on top left corner of the desired area and then drag to the lower right corner and release the mouse). A popup appears and you choose the type of test to perform on the region:



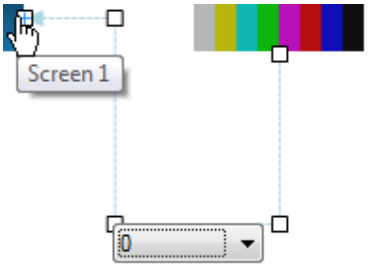
5.2.6.3 Adding Links

In auto record mode, links are added automatically. This helps with initial adding of screens but poses a problem later. In the example above, we will want to add a means to go from the second screen back to the first, in this case by pressing the 'Power' button. If we just use the remote control and capture again, we end up with 2 screens the same. Not good. So to add a link, move the mouse over the box until the hand appears:

Click and drag the arrow to the destination box:



Then release the mouse and the link is formed. At this stage a drop down box appears for you to select the desired button press:



5.2.7 Remote Control

The remote control allows you to send commands to the DUT. If it is greyed out, it means that no DUT is reserved. You select the command to send by clicking the button.

 this icon will appear for the duration of sending an IR command and as feedback that you clicked a button.

In addition the keyboard can be used to send commands. The following keys are supported:

0 - 9

left arrow

right arrow

up arrow

down arrow

Page Up (sends channel+)

Page Down (sends channel-)

+ (sends volume+)

- (sends volume-)

The image used and whether or not you can actually send keys depends on the correct IR being trained and added to the StormTest Development Center database using the Admin Tools application.

If you click on a key with the right mouse button a menu appears allowing you to copy the key name to the clipboard. The name copied is the name of the button that would be needed in a Python script.

5.2.7.1 Macro Mode

At the top of the Remote control is the Macro Toolbar:

The controls are:

5.2.7.1.1 Record

Click this to start recording buttons from the IR remote control. Both the key and the time delay after the key is recorded. To stop recording, click the record button again. You will be prompted to save the macro to a file. If you choose not to then you may use it while the remote control window is available. After the window is closed, the macro is lost.

5.2.7.1.2 Play

Playback the macro. It plays back with the timing and keys previously recorded. While playing, the button changes to a Stop button so that you can interrupt playback.

5.2.7.1.3 Open

Click this to load a macro that you have previously saved to disk. You can then play back that macro.

5.2.7.1.4 Save

If you chose not to save a macro after recording, you can change your mind and save the macro later by clicking the save button.

5.2.7.1.5 Repeat

Normally, the macro plays just once. However, you can click the repeat button and it will play back continuously until you click the repeat button again.

5.2.7.1.6 Recent Macros

A drop down list of recently used macros so that you can switch between several macros easily.

5.2.7.1.7 Auto Hide

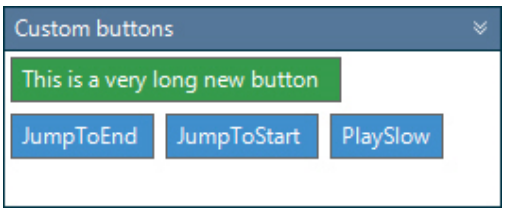
The right most pin symbol auto hides the macro toolbar - useful for small screens. Move the mouse near the top of the remote control window to show the toolbar again.

5.2.7.1.8 Importing

You can import a macro into a Press Multiple Buttons block, either via the right mouse menu or the Press Multiple buttons trainer.

5.2.7.2 Custom Buttons

Some IR remote controls have extra custom buttons that are not distinct button on the remote control. For example, some IR remotes behave in a special way if you hold down a key for a long time. Such button appear in the bottom panel and can be pressed using the mouse:



5.2.8 Video Panel

The video window shows you live video whenever a DUT is reserved in the Navigator. Below the video window is a small toobar with tools on it:

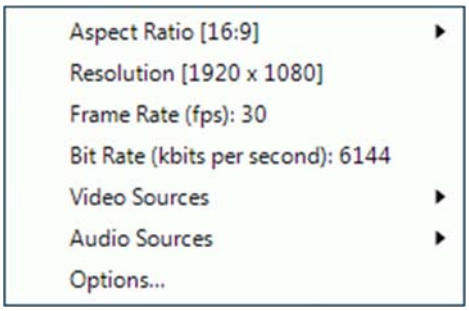
These are:

Audio mute. The mute state is shown. This is a toggle of the audio. If you hear no audio, check that your speaker on the PC is enabled as that can be muted and this control does not alter your PC audio status.

Close the video stream. This has no effect on the reservation of the DUT.

Open the video stream. Reverses the close

Change the video settings. Most of these are via a menu:



But the full range is possible by selecting options.

Grab a still image of the video - a normal file save dialog is presented

Start saving the video to a local file - a normal file save dialog is presented

Power status / control. The indicator is green as shown when the power is applied to DUT. Clicking it will turn the DUT off.

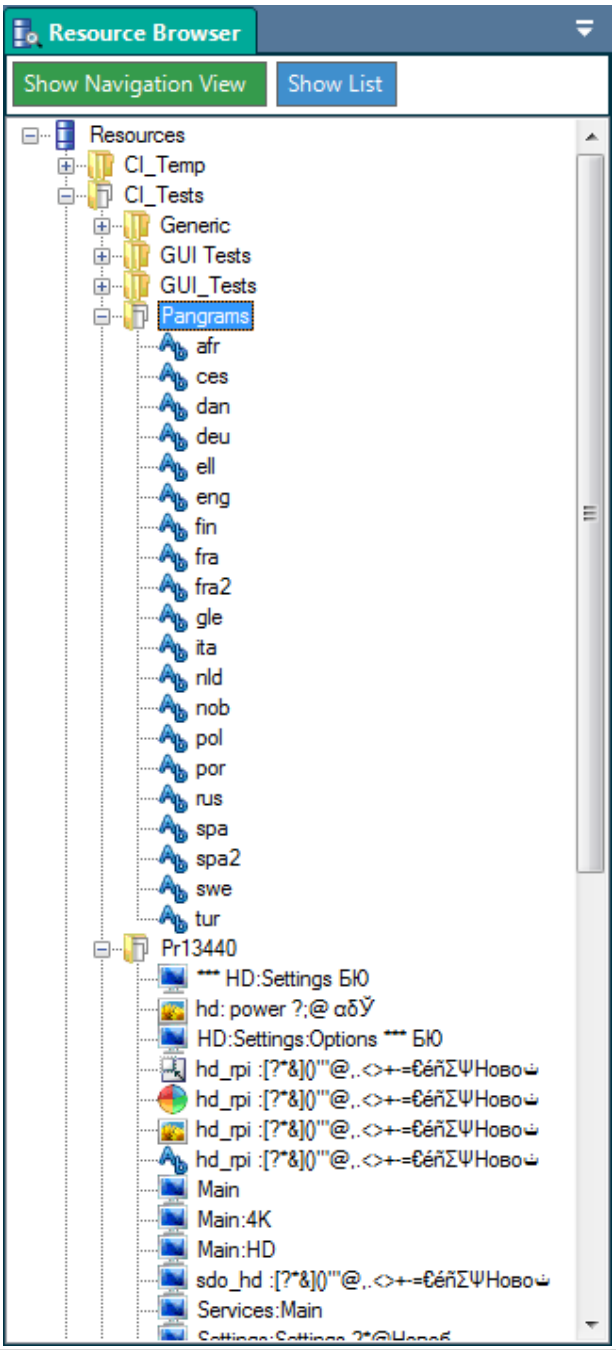
Power status / control. The indicator is red as shown when the power is OFF. Clicking it will turn on the DUT.

Sub slot control. If the DUT reserved is in an HS64 server then an extra icon is visible. From this a menu appears to allow you to view and select the sub slot in use. This icon is hidden for all non HS64 servers.

At the far right is the name of the server and the slot number of the video. If the video is an HD source then there is a little icon indicating HD: HD

5.2.9 Resources Panel

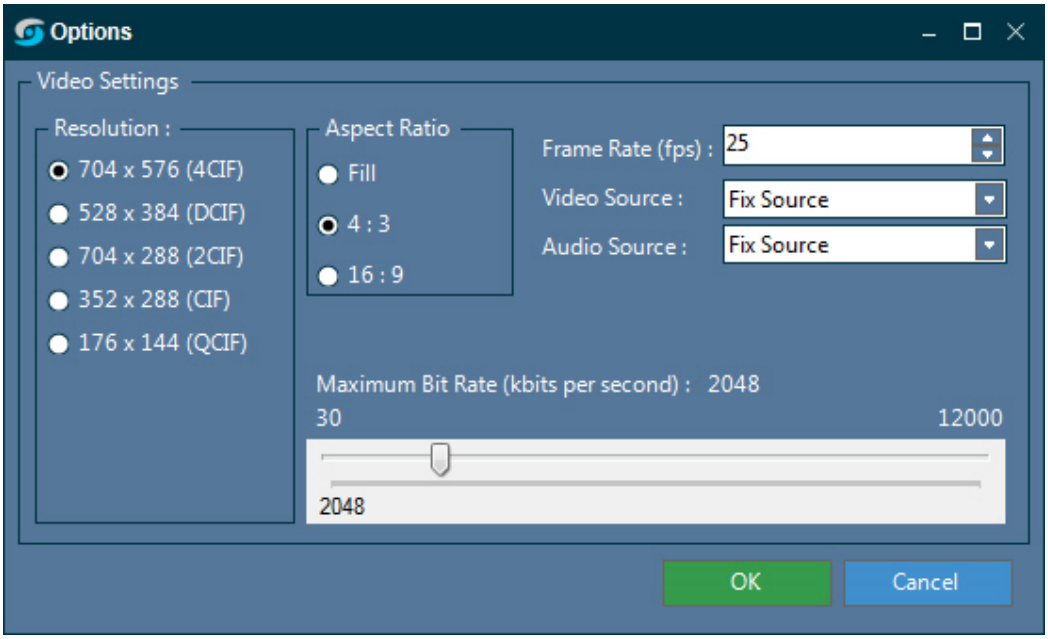
The resources panel allows you to browse and select resources:



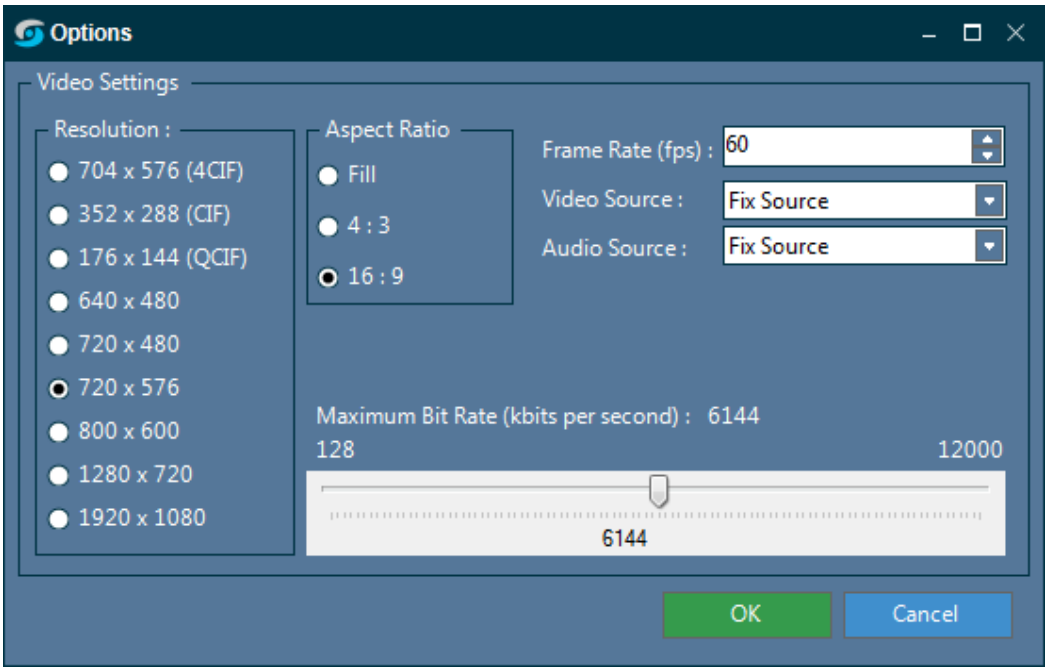
This is the same control as in the Resource Editor applet. You can see the resources, and perform some limited resource management (move, copy, delete, create new empty resources).

5.2.10 Video Options

The video options allow you to set the video streaming parameters:



And if the source of the video is HD or 4K:



5.2.10.1 Resolution

Choose one of the fixed resolutions. The named values of the form xCIF are for PAL, those names xSIF are for NTSC. All other resolutions are for HD systems. The HD servers will use the negotiated raw HD resolution (usually 1920 x 1080) but for a 4K server, you may choose a resolution.

5.2.10.2 Aspect Ratio

This controls how the video is shown. Fill will use the available screen space and this will usually distort the video. HD streams will by default be shown as 16:9 (the normal aspect ratio for HD).

5.2.10.3 Frame Rate

This is the frame rate in frames per second. For PAL systems, the maximum is 25, for NTSC the maximum is 30. For HD systems, the maximum is 60.

5.2.10.4 Video Source

The video source. It will be 'Fix Source' unless the server has an audio/video switch installed. In that case, you can select the source of the video for streaming.

5.2.10.5 Audio Source

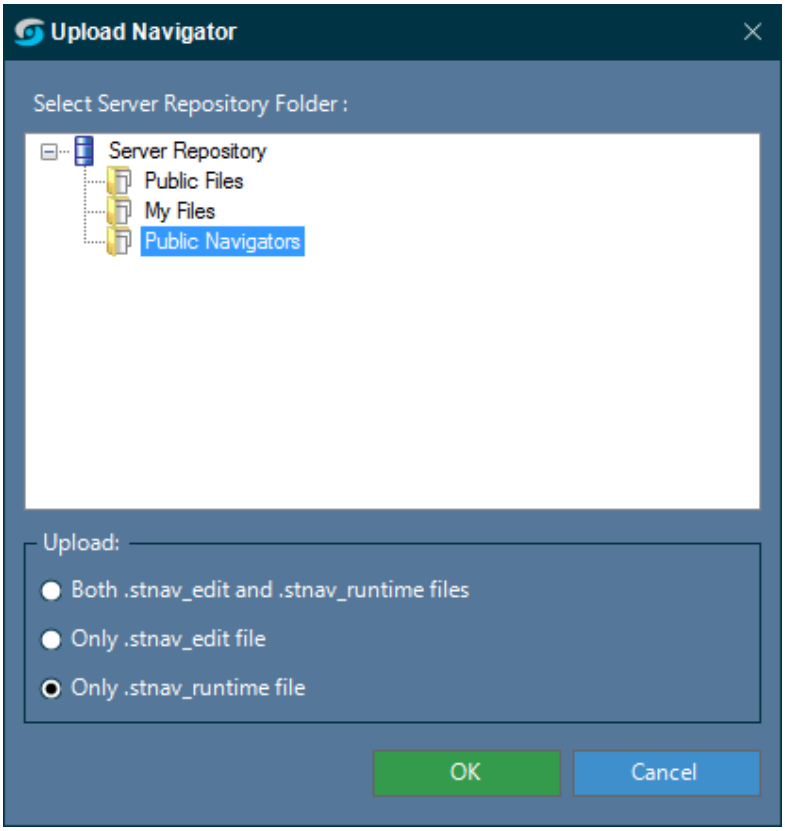
The audio source. It will be 'Fix Source' unless the server has an audio/video switch installed. In that case, you can select the source of the video for streaming.

5.2.10.6 Maximum Bit Rate

This is the maximum number of bits per second that will be sent. If the picture is static or slow moving, then far fewer bits per second will actually be sent. The slider controls the values in kbits/sec. **NOTE**: The accuracy of the maximum bit rate is approximately 10% so a setting of 1000 kbits per second could result in an actual bit rate 10% higher or lower as the performance of the video encoder is subject to a variety of optimizations. The maximum bit rate for SD systems is 4000 kbits/sec and for HD servers 12000 kbits/sec

5.2.11 Upload Navigator

When you upload a navigator, the dialog is shown:



You can select the location of the navigator - the folder must exist. This dialog won't allow creation of new folders. Use Developer Suite Test Manager to do that.

You can upload either the runtime file, the edit file or both files.

To use the file in the Admin Console you must upload the runtime file to the public navigators area (these options are selected by default).

5.2.11.1 Considerations

If you just upload the runtime file then any user with access to the folder can use the navigator but not edit it.

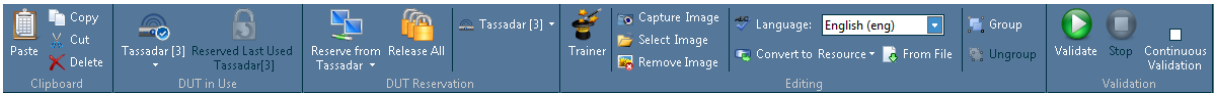
If you just upload the edit file, then any user with access to the folder can take a copy and edit it (and produce the runtime file) but no one can use the file in the Admin Console (unless they do the work of uploading the run time file).

5.2.12 Edit Screen Ribbon

When editing a Navigator screen, the ribbon shown is similar to the main ribbon. It has 3 tabs.

5.2.12.1 Home Tab

The Home Tab is:



The controls are enabled and disabled as needed. The commands are grouped and a brief overview of each is:

5.2.12.1.1 Clipboard

This group deals with the Windows clipboard

 Paste. You can only paste regions that you have selected. Ctrl-V also works

 Copy. Regions copied to the clipboard can be pasted as text to other Windows packages. As text, you see the raw XML.

 Cut. Copies and then deletes a region.

 Delete. Deletes the selected region.

5.2.12.1.2 DUT in Use

This group deals with the DUT in use by the Navigator.

 This shows you the DUT (by server[slot]) of the DUT used by the test case for running and debugging. If you have reserved more than one DUT, then you can change the DUT in use by clicking this and selecting an alternative DUT.

 Reserve Last Used. If you have not reserved a DUT via the main ribbon, this allows you to reserve the DUT that was last used for validation (even if that failed). It will be disabled if the Navigator has never run the validation routine.

5.2.12.1.3 DUT Reservation

This group is the same as the main ribbon with the same functionality.

5.2.12.1.4 Editing

This group is used during the creation and editing of the screen.

 Launch the appropriate trainer for the selected region

 Recapture an image. This will take a screen shot of the current video in the DUT and put it into the screen, replacing the previous image.

 Select an image from a file for the screen. This will replace the image in the screen. Particularly useful if you have the original images for the DUT in high quality graphics. Note that the size should match the StormTest Capture size (704 x 576/480 for SD and the expected HD resolution for HD systems).

 Delete the image from the screen. The screen will be blank afterwards.

 Select the default language for the navigation map. This value will automatically be set as the language for all new OCR Compare String tests created. It does not alter existing OCR tests.

 Create a new resource from the selected test region. You can convert the test to a region, image, string or color. This can be later used in the Navigator. It can also be used to update an existing resource with new data.

 Create a new Screen Definition Test within the navigator screen by importing a Screen Definition from a file. This adds the Screen Definition to the list of existing tests.

 Group the selected test or tests into a single Screen Definition Test.

 Ungroup the selected Screen Definition into its component sub tests.

5.2.12.1.5 Validation

This group deals with the validation of the screen.

 Validate the screen. It runs all tests for the screen and updates the summary at the bottom of the screen. Note that this button does exactly what the runtime Navigator will do. So if there are no tests defined, the button is still enabled and will do nothing but claim success. So would the runtime navigator.

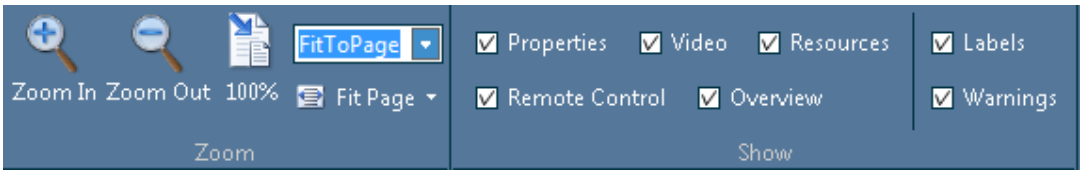
Validation criteria are defined by using the mouse to define a region and select whether that should be an image comparison, color comparison, motion detection or an OCR operation. Multiple regions can be defined per screen. Collectively these are the 'validation criteria' that are used by navigator to determine whether the screen on the DUT is the expected screen.

 Stop validation - only useful if you define a very large number of tests for a screen.

Continuous Validation - a simple check box. When checked, validation on the screen is run continuously so that you can see how multiple validations would work. It waits for 1 second after completing the full set of tests. It is also useful to see if moving away from the screen causes a failure (as it should if the tests are carefully designed). If you have an OCR test defined, the validation only runs for 10 minutes - this is to protect you from running out of OCR license should you be called away from your computer.

5.2.12.2 View Tab

The view tab is:



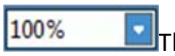
5.2.12.2.1 Zoom

This group deals with the zooming of the screen.

 Zoom in - makes the screen larger, showing more detail.

 Zoom out - makes the screen smaller, showing less detail but more of the screen.

 100% - makes the screen 1:1 scaled so that you see the actual captured resolution of each screen.



The current zoom level is shown as a % of full size and can be changed directly

 Fit the drawing. 3 options are available: Fit to Page, Fit to Width, Fit to Height.

5.2.12.3 Show

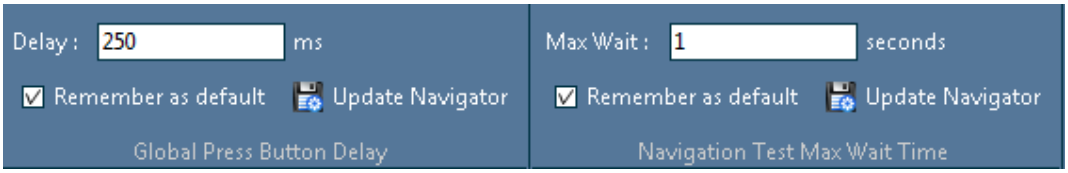
This group of check boxes shows/hides the panels. The right hand column controls how the main Navigator is drawn.

Show Labels - when checked, the button names are shown on each link

Show Warnings - when checked, the screens without validation criteria are shown with a warning triangle in top right corner.

5.2.12.4 Preferences Tab

The preferences tab is:



5.2.12.4.1 Global Press Button Delay

This controls the delay after every button press. Whenever you create a multiple button sequence by entering the buttons directly in the multiple buttons trainer within Navigator a delay is used. You can set the default to use here.

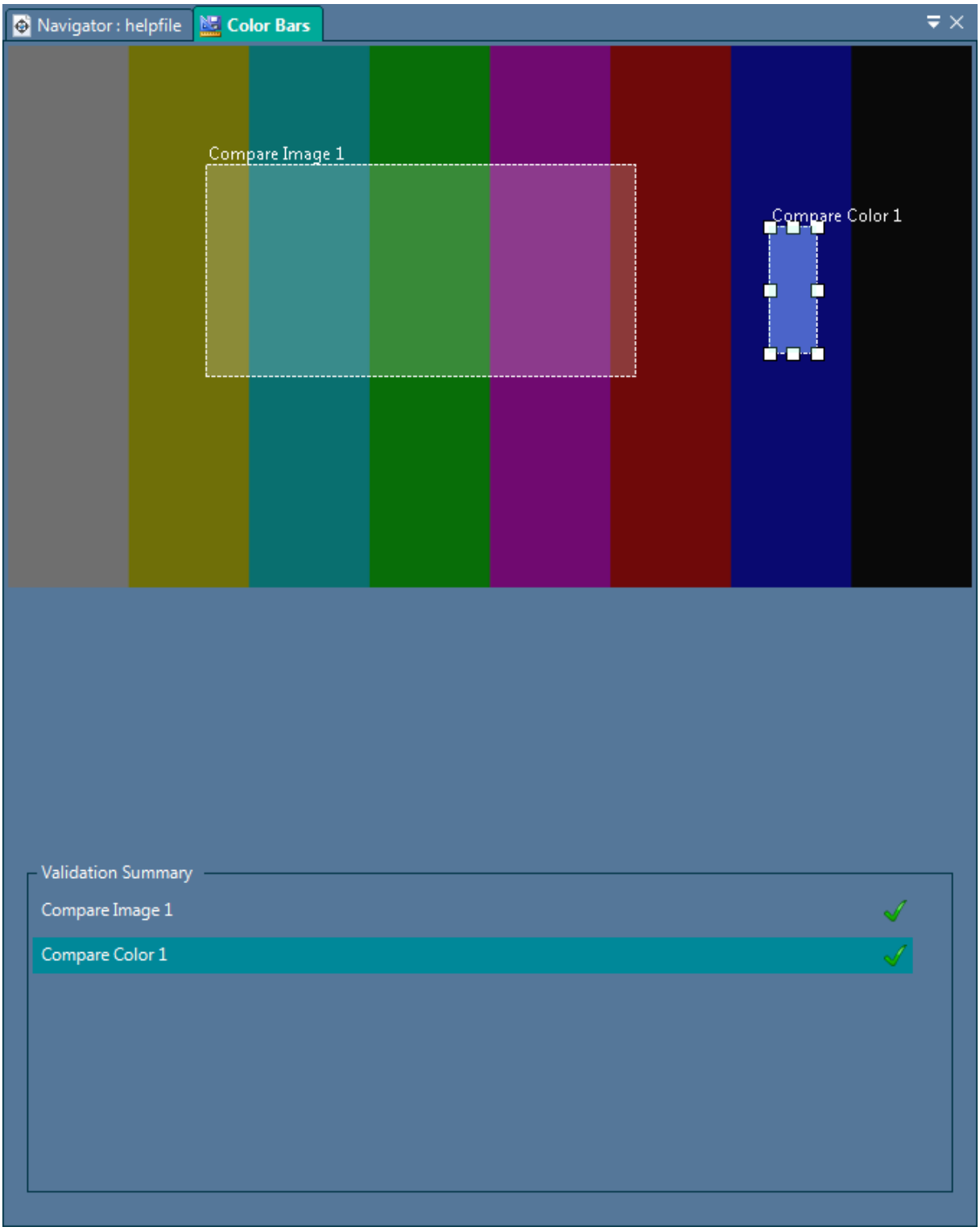
The delay can be made persistent so that it will apply next time Developer Suite starts by ensuring 'Remember as default' is checked. If you check 'Update Navigator' then all button delays on the current navigator will be updated.

5.2.12.4.2 Navigation Test Wait Time

This controls the time at the end of each test on a screen within the Navigator. A timeout is needed on each test. The default timeout can be set here. The currently open Navigator can be updated by clicking the Update Navigator button.

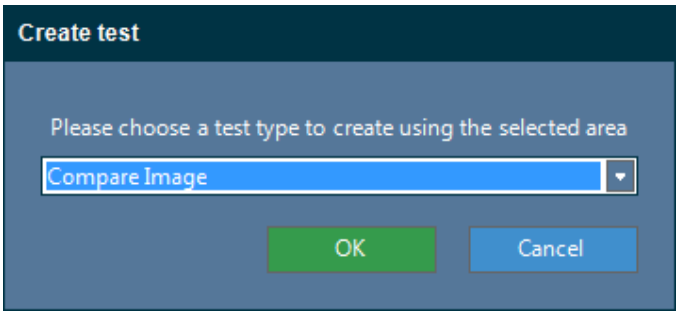
5.2.13 Screen editor

The screen editor is where you set and edit validation criteria for a screen. You can only do this if a DUT has been reserved. The main panel changes to show the screen with the regions selected:



5.2.13.1 Usage

To create a new validation region, use the mouse and highlight an area. On completion of the selection a small popup window appears asking you to choose the type of region to create:



There are four possible types of region:

- Compare Image - the region is tested that the correct image appears
- Compare Color - the region is tested for color.
- OCR Compare String - the region is tested using OCR for the correct string
- Detect Motion - the region is tested for motion

After selecting the region, the current values of the captured screen are used to set the validation criteria for the region. Suitable defaults for tolerances are selected.

You can move the region around to get the exact location - the values of image, color and OCR are updated to match. You can also resize the region and values are updated.

5.2.13.1.1 Screen Definition Objects

You can use a screen definition object as a test 'region'. It will appear as an icon below the screen ( ). You can create a screen definition object in 3 ways. You can group a selection of existing tests, you can import s screen definition object from a file or you can drag a screen definition resource from the resources panel onto the navigator screen.

5.2.13.2 Deleting a Region

Just select the region and use the delete button on the ribbon, the Delete key or the right menu.

5.2.13.3 Validating the Regions

You can use the ribbon Validate button to check that current video will validate with the regions selected. Below the screen image will appear the validation status along with a summary of the problem if validation failed.

5.2.13.4 Trainers

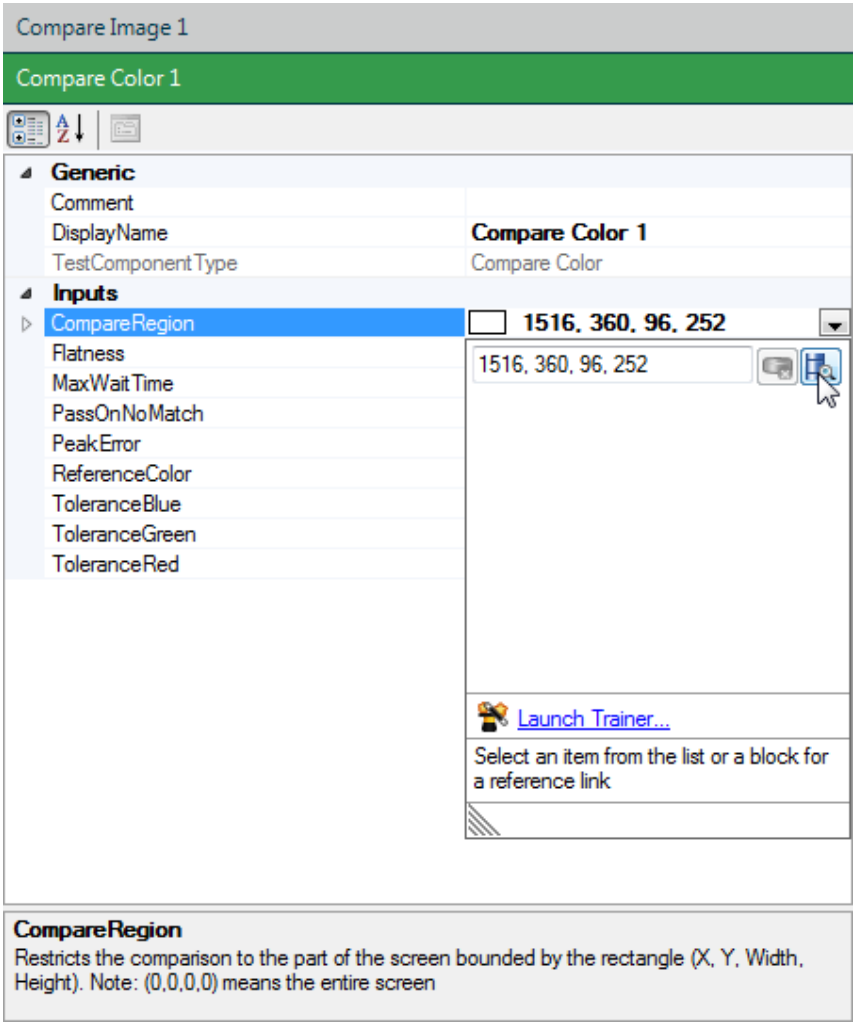
If the automatically selected values for the parameters don't seem to be perfect, then you can use the Compare Image trainer, Compare Color trainer, OCR trainer, detect motion trainer or screen definition trainer to adjust values. The trainers appear in the main form replacing the image panel. You cannot edit other screens or the main map while the trainer is visible. A breadcrumb tab appears:

Select the main screen to close the trainer, confirming all its changes. While the trainer is open, you can still use the properties panel to make changes and if those changes affect the trainer, the results will be seen immediately. Each trainer has its own ribbon control.

5.2.13.5 Using Resources

Instead of explicitly setting values for the region, image, color, string or screen definition object you can use named resources. These can be created in advance using the Resource Editor or can be created during editing.

From the properties items, look for the resource icon and click it. For example, whereever coordinates are needed, you can use a named region:



Clicking on the little database symbol will bring up a dialog to select the region. This dialog only shows the resources that are relevant at the time (you cannot select an image into a region).

Images are used in Compare Image tests by selecting the Resource Icon property.

Colors are used in Compare Color tests by selecting the color, tolerance, flatness or peak error properties.

Strings are used in OCR Compare String tests by selecting the Expect String property.

5.2.13.5.1 Creating Test from Resource

You can use the resource panel to directly create a test. If you drag a named region on to the screen then you can create a test there in the same was as if you had selected a region with the mouse. You are prompted for the type of test to create.

You can also drag a screen definition resource on to the image and it will be added to the tests for the screen.

6 Screen Editor

6.1 Screen Definition Objects

Version 3.2 of StormTest introduces the concept of a screen definition object. This is the name used by the Python API. Within the Developer Suite, it is referred to as a 'screen' on menu items and toolbars and in this document as a 'screen' where the context is clear and as a 'screen definition' otherwise.

A screen definition is a collection of 'regions' within a screen. Each region represents some test to be performed on a screen. The available tests are image, color, ocr, motion and audio. These correspond to existing blocks within test creator and Navigator (with the exception of audio which is not used in Navigator). In many ways a screen definition is the data portion of a Navigator screen. However, it has been enhanced and is now used as a standalone concept. This allows it to be saved in a file or stored in the resource database.

Once a screen definition has been created it can be used in the following situations:

- In a Test Creator Screen Definition Block
- In a Navigator Screen
- Directly from a Python script

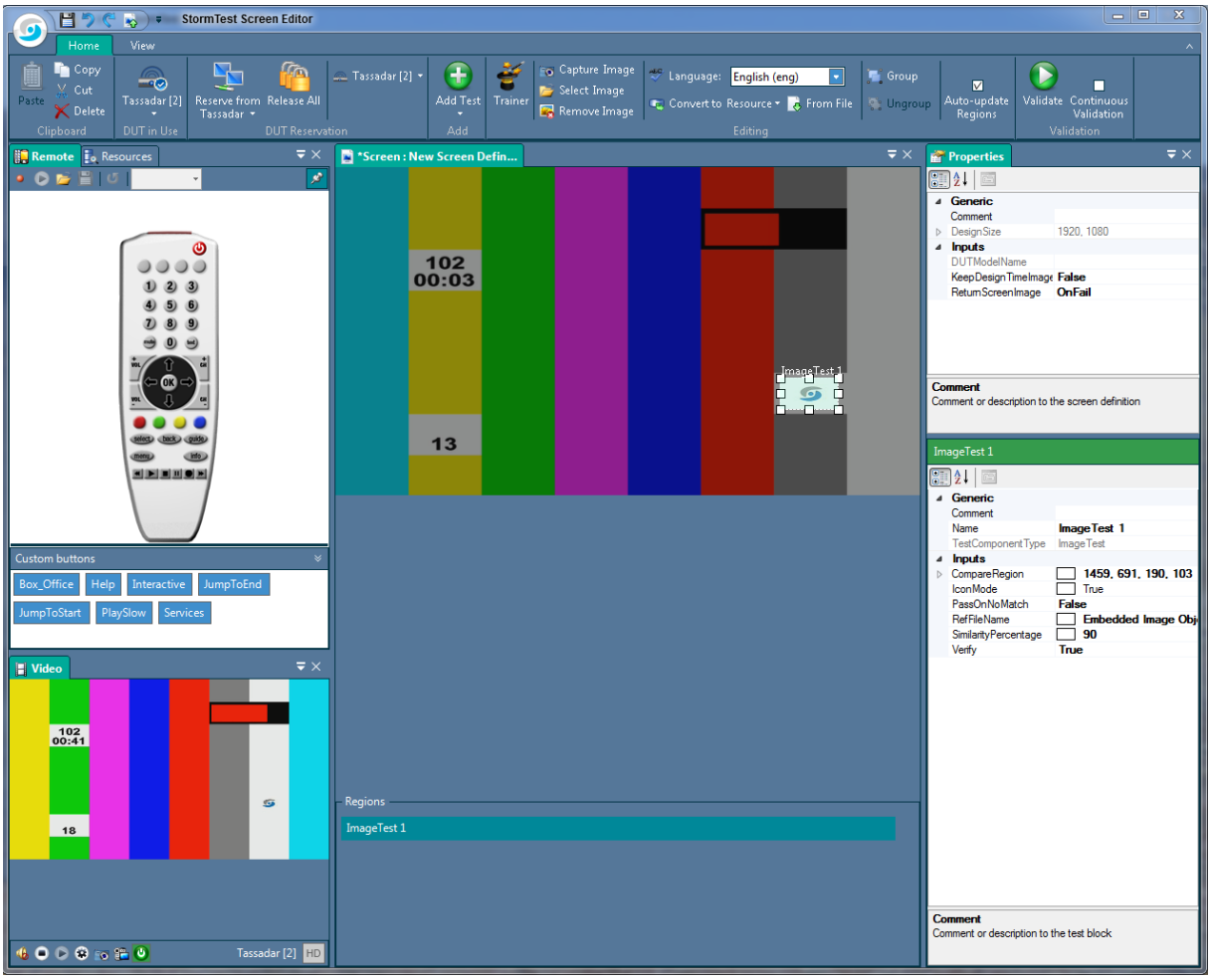
In all cases, the screen definition can be used from a file or from a resource.

The screen definition can also be used within another screen definition to create a hierarchy of screen definitions. There is no limit imposed by StormTest on the nesting level of such objects. However, to edit such a complex hierarchy becomes difficult if resources are not used. Without resources, you have to open a new window for each layer in the hierarchy and this can get confusing. If you think you need to used nested screen definitions, you are advised to consider using resources for the nested screens - these can be managed and manipulated independently and the final screen definition is then a container for such independently managed items.

The full set of properties is documented in the screen definition reference.

6.2 Screen Definition Editor Main Screen

The screen editor main screen is where you do most of the work of creating and editor a screen definition object. The default layout is:



6.2.1 Ribbon

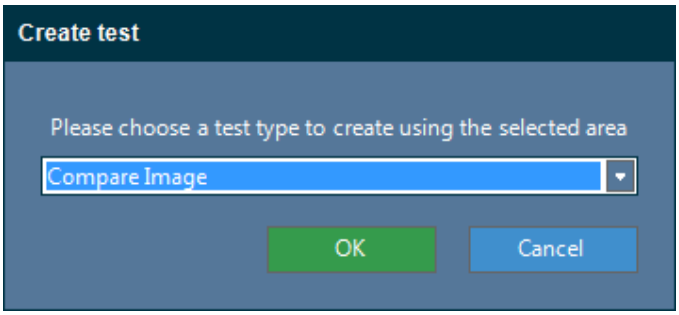
The ribbon at the top of the screen contains the commands needed for creating and editing screen definitions.

6.2.2 Properties Panel

The properties panel, shown on the right, shows you the properties of the screen definition in the upper portion and the selected test within the screed definition in the lower portion. This is a tabbed grid and will be empty if no tests are defined.

6.2.3 Screen view

The central panel above shows the screen definition - blank initially. You can capture a background image to work with and add tests to this. You add tests using the mouse to select a region (click on top left corner of the desired area and then drag to the lower right corner and release the mouse). A popup appears and you choose the type of region to add:



You can also drag a named region onto the screen and the same dialog appears. To add an audio region use the ribbon.

6.2.4 Control Panels

The panels on the left show the control panels: the remote control, the video and resources.

6.3 Screen Editor Ribbon

The main ribbon at the top of the screen definition editor has 2 tabs.

6.3.1 Home Tab

The home tab is:

The controls are enabled and disabled as needed. The command are grouped and a brief overview of each is:

6.3.1.1 Clipboard

This group deals with the Windows clipboard

 Paste an item from the clipboard. Ctrl-V also works

 Copy. Items copied to the clipboard can be pasted as text or an image to other Windows packages. As text, you see the raw textual representation of items.

 Cut. Copies and then deletes an item or group of items.

 Delete. Deletes the selected screen regions.

6.3.1.2 DUT in Use

This group deals with the DUT in use by the Screen Editor.

 This shows you the DUT (by server[slot]) of the DUT used by the screen editor for developing the screen definition. If you have reserved more than one DUT, then you can change the DUT in use by clicking this and selecting an alternative DUT.

6.3.1.3 DUT Reservation

This group is the same as the main ribbon with the same functionality.

6.3.1.4 Add

This group has just one button to add tests directly to the screen definition. For those tests that have required region (color, image, motion, ocr) then a default area is used and you can move it to the required position.

6.3.1.5 Editing

This group is used during the creation and editing of the screen definition

 Launches the trainer for the selected region (the compare color trainer, compare image trainer, ocr trainer, audio trainer or motion trainer depending on the selected region).

 captures a screen shot from the video and uses it as the background for the screen definition. This image is used only for the design phase. The `KeepDesignTimeImage` property of the screen definition controls whether this image is stored when the screen definition is saved.

 Select an image from the file system to use as the background image. The size of the image should match the size of the video of the device being used. This affects the design size of the screen definition. If you load an image with an incorrect size for the DUT reserved you will be prompted to make a decision about what to do - you can scale the image, cancel the load or change the design size. The recommendation is to discard the image and find a better one. Next best option is to scale the image. Changing the design size may cause real confusion - you should probably select a new DUT as well.

 Remove the background image from the screen. The screen will be blank afterwards.

 Select the default language for the screen definition. This value will automatically be set as the language for all new OCR regions created. It does not alter existing OCR regions.

 Create a new resource from the selected test region. You can convert the test to a region, image, string, color or a screen definition. This can be later used in the Navigator, Test Creator or a screen definition. It can also be used to update an existing resource with new data.

 Create a new Screen Definition Test within the current screen definition by importing a Screen Definition from a file. This adds the Screen Definition to the list of existing tests.

 Group the selected test or tests into a single Screen Definition Test.

 Ungroup the selected Screen Definition into its component sub tests.

Auto Update regions: This check box causes the regions' contents to be updated automatically when it is moved. If you move or resize a color region, for example, then the background color, flatness and peak error properties are recalculated to match the resized of moved color region. For an OCR region, the text will be re-read using OCR.

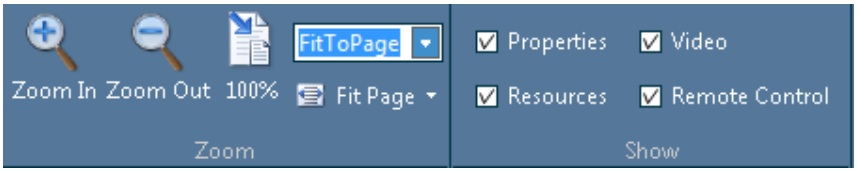
6.3.1.6 Validation


Validate the screen definition. The results of the individual tests are show below the screen.

You can check the `Continuous Validation` box and Developer Suite will run continuous validations on the screen definition.

6.3.2 View Tab

The second tab is the view tab:



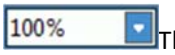
6.3.2.1 Zoom

This group deals with the zooming of the screen.


Zoom in - makes the drawing larger, showing more detail.


Zoom out - makes the drawing smaller, showing less detail but more of the drawing.


100% - makes the drawing 1:1 scaled so that you see the actual captured resolution of each screen.



The current zoom level is shown as a % of full size and can be changed directly


Fit the screen. This tab is common to all editors and although 3 options are available: Fit to Page, Fit to Width, Fit to Height, only the Fit to Page is active for the screen editor.

6.3.3 Show

This group of check boxes shows/hides the panels.

6.4 Screen Definition Reference

This lists all regions and the properties that are supported from the Screen Editor. Please see the Python API documentation for the equivalent Python properties.

6.4.1 Screen Definition Properties

Comment: A free form field for you to put in any data that you wish.

Design Size: The size for which the screen definition is designed. Automatically filled in by the screen editor. It can be set explicitly from a Python script. All coordinates are assumed to be relative to the design size. If the screen definition is used on a DUT where the captured size is not the same as the design size then all coordinates will be scaled to make the screen definition work. This is most useful in HD systems where the scaling from 1920x1080 to 1280x720 is almost certainly perfectly linear.

DUTModelName: This is read only and filled in only when the screen definition is read from a resource. It thus supports the model specific resources of StormTest version 3.2

KeepDesignTimeImage: This flag instructs the screen editor to keep the design time background image when it saves the screen. This property does not exist in the Python API and is for the GUI only. If you spend a long time working on one screen, it may be useful to save the background image used along with the screen. This makes files larger. It defaults to False meaning the screen is saved without the design time image.

ReturnScreenImage: This controls when the screen definition should return an image on execution. The default is OnFail meaning only when the screen fails.

6.4.2 Region Common Properties

The regions are considered 'tests' to be performed on the screen. They have some common properties which are described here instead of repeating the description on each section below.

Comment: A free form field for you to put in any data that you wish.

Name: The name you wish to give to the region. It will be used in log files so it should be kept short.

TestComponentType: The type of the region - read only and reflects the underlying type of test region.

PassOnNoMatch: Reverses the pass condition. Normally the pass condition is when the image from the DUT matches the criteria in the region but on occasions you want a negative test - set the PassOnNoMatch flag in that situation.

Verify: When this is true, the region's values are verified against the criteria producing a pass or fail which contributes to the overall pass or fail of the screen definition. When this is false, the region is a 'read region' - the values are read but there is no pass or fail. Such regions cannot be usefully used in Navigator or Test Creator. However, they have uses from Python: you can set up a screen to verify some regions and read others. For instance on a settings screen, you could verify a color, an icon and some text and then set up many read regions of type OCR to read the settings.

6.4.3 Color Region

CompareRegion: The rectangle to use for the color. Either a literal set of coordinates or a Named Region resource.

Flatness, PeakError, ReferenceColor, ToleranceBlue, ToleranceGreen, ToleranceRed: same as for the Test Creator Compare Color block.

6.4.4 Image Region

CompareRegion: The rectangle to use for the image compare. Either a literal set of coordinates or a Named Region resource.

ReferenceImage: The reference image to use. Can be set from a file or a resource. When set from a file, the value becomes "Embedded Image Object". Unlike Test Creator, the screen definition does not use external image files but packages the image into the screen definition.

SimilarityPercentage: The degree of similarity in the range from 1 to 100 that is required. The images are the 'same' if the calculated similarity is greater than or equal to this value.

IconMode: When true, the image is considered to be an icon. That means it is expected to be smaller than the captured image and the left and right values of the Compare Region indicate the location of the icon in the captured image (the width and height of the Compare Region are ignored in icon mode). Because icon mode results in a more compact screen definition and thus saves network bandwidth, this is the preferred mode of operation.

6.4.5 OCR Region

OCRRegion: The rectangle to use for the OCR. Either a literal set of coordinates or a Named Region resource.

ExpectedString: The string that is to be expected. It can be a literal string or a Named String resource. If the screen definition has been loaded from a file made in Python then this can be a single string in JSON format such as ["a","b"] - edit this with care if you wish to use the screen definition in Python.

AutoCorrection, Language, ProcessingFilters, UseLegacyInvert: same as for the Test Creator OCR Compare String block.

6.4.6 Motion Region

DetectRegion: The rectangle to use for the motion detection. Either a literal set of coordinates or a Named Region resource.

Percentage, Timeout: same as for the Test Creator Detect Motion block.

6.4.7 Audio Region

For consistent naming, this is called a region but there is no rectangle as audio is not part of video. It does allow you to add a test for audio presence along with video tests in a screen definition.

Channel: Which channel to use. This is only applicable to HD systems where you can select left or right audio for detection.

Threshold: The value in dBu above which audio is considered to be really audio. Below this limit is "silence". The audio is measured over 128 ms across the bandwidth 20 Hz to 8 kHz. Unless the input

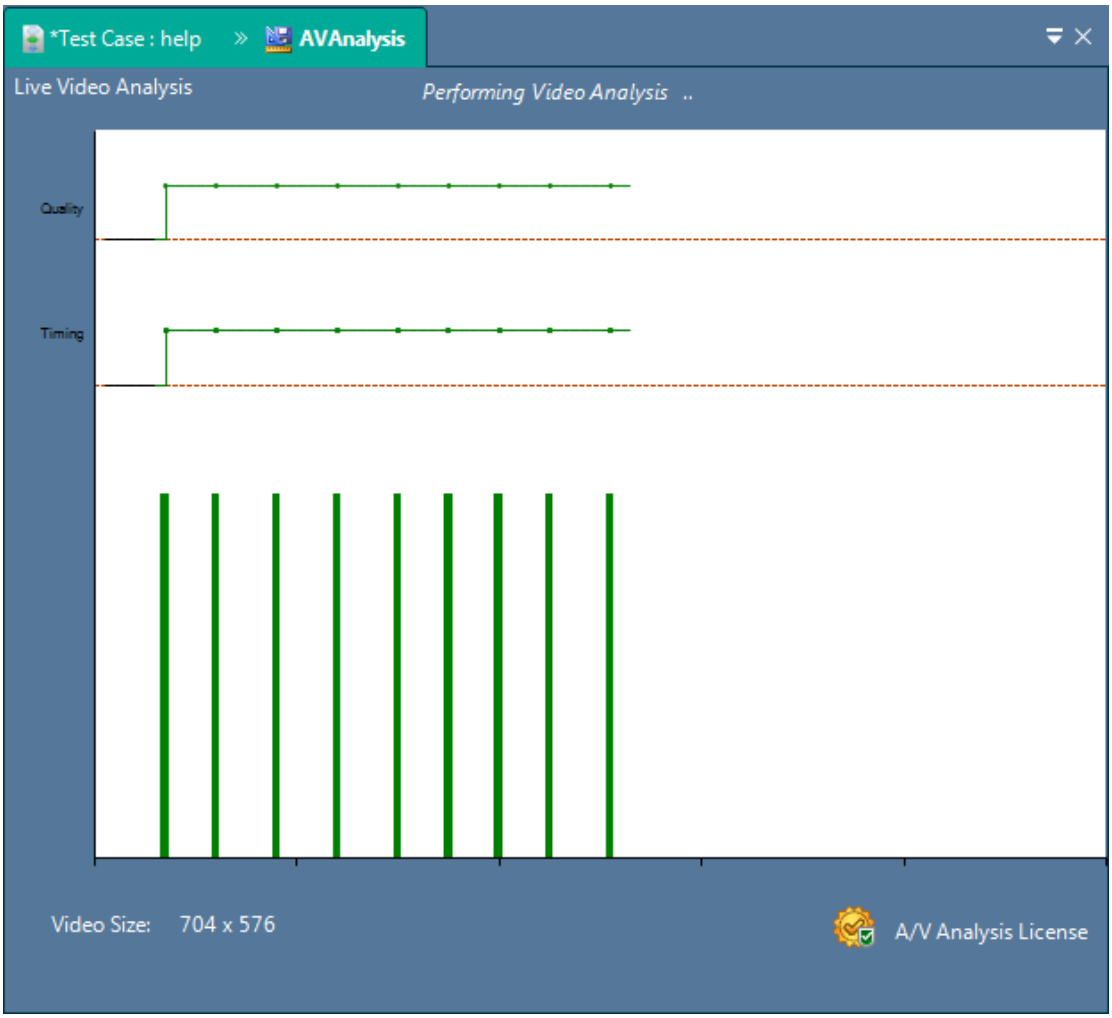
is a pure continuous tone, the actual levels detected could be quite low due to the 128 ms measurement period.

Timeout: Time to wait for audio to be detected (or not, if PassOnNoMatch is true).

7 Trainers

7.1 Audio Video Analysis Preview

When you select the A/V `Analysis` block, you can invoke the preview by using the `Trainer` button on the ribbon, the right mouse context menu, the keyboard shortcut ALT+T followed by T or selecting any input property and selecting `Launch Trainer` after clicking the drop down to show the reference links.



7.1.1 Purpose

The A/V analysis preview is designed to show you whether the selected DUT will pass the video analysis. Calibration of the settings is performed using the separate Admin Tools application. Audio analysis is no longer supported in Developer Suite.

7.1.2 Usage

The panel shows the status of the video analysis. It has a line plot of the pass/fail status of the quality and timing sub parameter of the video analysis. The bar graph is a simple pass/fail color and shows the overall video analysis pass or fail status.

The video analysis is run for the duration specified in the A/V `Analysis block` with a maximum time of 5 seconds (so if the block requests 10 seconds, the preview will use 5 seconds). The text '`Performing Video Analysis ...`' is displayed while the video analysis is running. The analyses rely upon a predefined stream and use the calibration from the database.

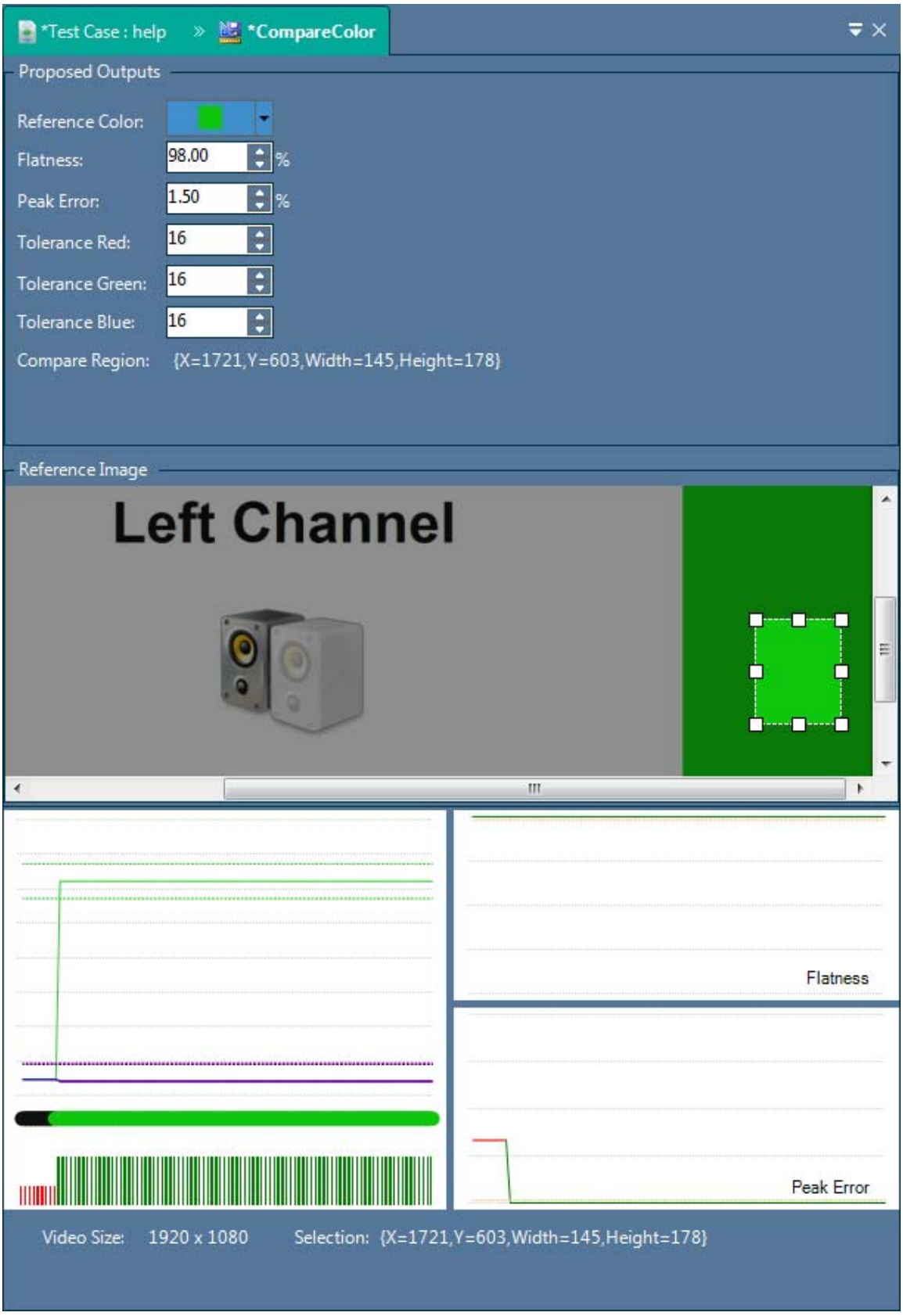
Please refer to the reference page for a description of the meaning of each of the parameters of the video analysis.

7.1.3 License Status

The lower edge of the trainer shows whether the server where the DUT is placed has an AV analysis license. If it does not, you will not be able to run the preview and a more obvious message will appear. The AV analysis is a separately licensed feature of a StormTest server.

7.2 Compare Color Trainer

When you have a Compare Color Test Region, you can invoke the trainer by using the `Trainer` button on the ribbon, the keyboard shortcut ALT+T followed by T or selecting any input property and selecting `Launch Trainer` after clicking the drop down box. The trainer appears as a breadcrumb tab in the same tab as was previously open. The drawing below shows the Test Creator version of the trainer opened from a Compare Color block of the test case `help`.



7.2.1 Purpose

The Compare Color trainer is designed to help you select the parameters for the compare color test region.

7.2.2 Usage

The upper panel is the proposed output settings from the trainer. These can be set manually or calculated automatically using the Auto Tune option. Close the trainer and confirm changes by clicking on the parent test case, navigator screen or screen definition object (the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case, navigator screen or screen definition object.

The middle panel is a reference image to allow modification of the rectangle used. In Test Creator it is empty by default - you need to select or capture an image using the ribbon. For Navigator and Screen Editors, the reference image is the original screen that is being modified and cannot be altered in the trainer.

The flatness and peak error can be set to a resolution of 2 decimal places. The underlying code works to full double precision accuracy, however. If you are looking at block converted from a resource made by Python, then the resolution of the flatness and peak error may be more than 2 decimal digits. However, the view on this screen is rounded to 2 decimal places. Changes you make will be saved to 2 decimal places.

The lower panel is the live comparison plot.

7.2.2.1 Ribbon commands

The Test Creator version of the ribbon is shown and described here. The Navigator and Screen Editor versions of the color trainer do not have the `Image` group but are otherwise the same.

7.2.2.1.1 Clipboard Group

This is the standard copy/cut/paste/delete grouping.

7.2.2.1.2 Image Group

This is only available if the color trainer has been invoked from Test Creator. Because a Test Creator block has no embedded reference image and could be used on any screen from the DUT during a test, there can be no automatic reference image. The reference image is optional - it can be useful to adjust the region. You can use the live video to select a region, however, only the reference image supports zooming so if fine control of the region is needed, you should set the reference image.

Select: Select an image from the file system to use as the reference image. It must be of the same size as the DUT screen or else the coordinates will not make sense.

Capture: Capture an image from the DUT and use that as the reference image.

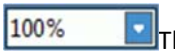
7.2.2.1.3 Zoom Group

Zooms the reference image. This will be disabled if there is no reference image.

 Zoom in - makes the screen larger, showing more detail.

 Zoom out - makes the screen smaller, showing less detail but more of the screen.

 100% - makes the screen 1:1 scaled so that you see the actual captured resolution of each screen.



The current zoom level is shown as a % of full size and can be changed directly

 Fit Page - scales the reference image so that it fills the area available on screen.

7.2.2.1.4 Results Group

This controls the results.

Clear Chart: Clears the live comparison plot, restarting the plot from the left of the diagram.

Auto Tune: Turns auto tune on or off. The auto tune algorithm attempts to find the 'best' set of color, tolerance, flatness and peak error values such that the selected region will pass the color test. It works by setting targets for everything except the color. The color is just the average color over the most recent 5 samples. These are quite tight (flatness 98%, peak error 0%, tolerance based on color component value but wider for low values which are in the noise floor). If these pass, then that is the final value. Otherwise, the tuning widens the values slowly until a pass is achieved. However, there are limits: it will never set the flatness to 0, peak error 100 and tolerance 255 even if that is the only way to get a pass. The system assumes you have set a region that looks like a flat color to the human eye.

Apply Offset: If the connected DUT has a video offset defined in the database, you can choose to use that offset in the trainer or not. You should use it if you are simply testing a block which was defined on another DUT. You should NOT use the offset if you are designing a test to run on the DUT in use. A warning appears near the live comparisons if the video offset is applied. The button is disabled if the database has no video offset for the DUT.

7.2.2.1.5 Resources Group

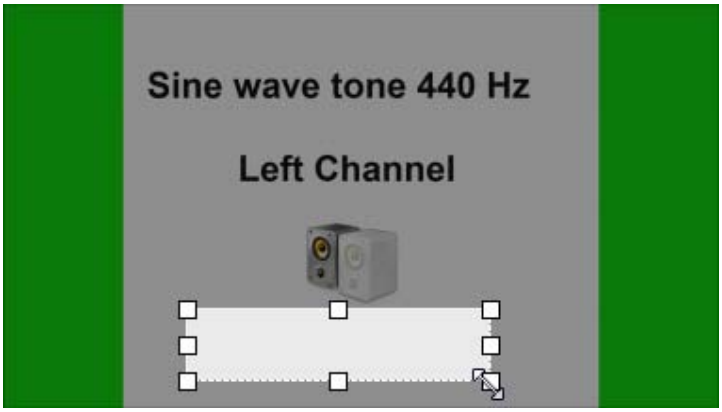
This allows you limited control over the resources used for this color test.

Convert To: Brings up a sub menu to select a resource type. This converts the selected rectangle or color to a named resource.

Clear All: Clears the resources from the color test. The region and reference color are replaced with the contents of the named resource and can then be edited.

7.2.3 Selecting a Region

You can select a region for comparison using the mouse on the video window or on the reference image. You can adjust a selected region using the mouse. Clear it by clicking outside the region. The keyboard arrow keys can move the selection by 1 pixel at a time:



Beneath the plots, you will see a textual representation of the video size being streamed and the selected region.

You can select the reference color directly, along with the flatness, peak error and tolerances. The Test Creator reference guide explains these terms.

7.2.4 Live Comparison

The left hand plot has 3 lines in red, blue and green indicating the currently detected red, blue and green levels against time. There are dotted lines in red, blue and green showing the band where the actual values must lie in order for a perfect color match to occur.

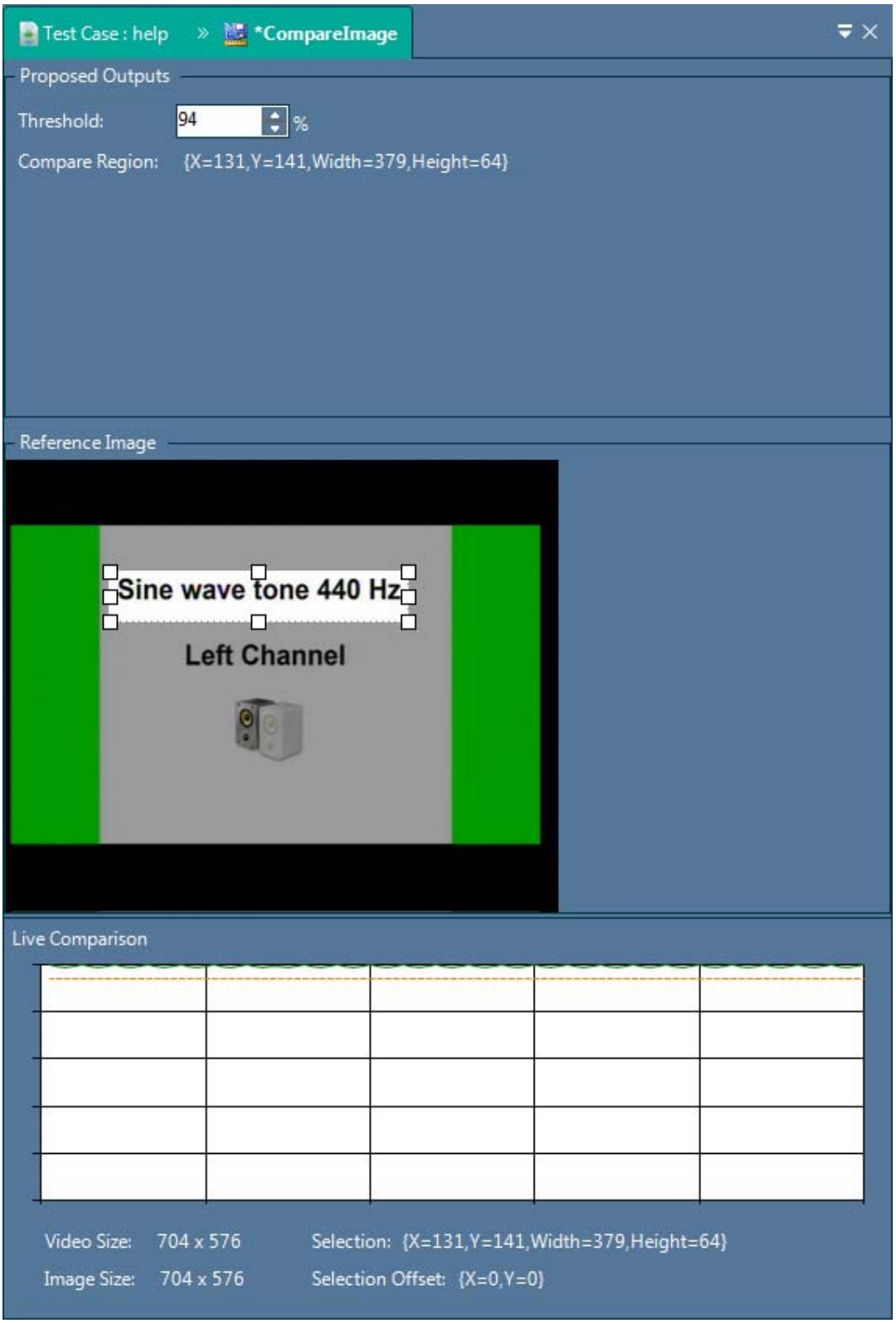
Beneath this is a solid large horizontal line which is the actual color detected (the visible combination of red, blue and green lines). In the screen shot above, you can see it is a fixed blue color. Beneath that again is a bar graph showing overall pass or fail - color coded but also the pass values are taller than the fail.

The right hand pair of plots show the Flatness and Peak Error values. Orange lines show the limits that have been set. The line plot is color coded for pass or fail. The compare color block is the most difficult to explain but quite easy to understand:

The comparison passes when the average color in the selected region is close to the set value and it is quite flat (not much variation) and there are very few pixels outside of the tolerance limits. The meaning of 'close', 'not much variation' and 'very few pixels' correspond to the tolerances, flatness and peak error values.

7.3 Compare Image Trainer

When you have a Compare Image test, you can invoke the trainer by using the `Trainer` button on the ribbon, the keyboard shortcut `ALT+T` followed by `T` or selecting any input property and selecting `Launch Trainer` after clicking the drop down box. The trainer appears as a breadcrumb tab in the same tab as was previously open. The drawing below shows the Test Creator version of the trainer opened from a Compare Image block of the test case `help`.



7.3.1 Purpose

The Compare Image trainer is designed to help you select the parameters for the compare image test.

7.3.2 Usage

The upper panel is the proposed output threshold from the trainer. This can be set manually or calculated automatically using the Auto Tune option. Close the trainer and confirm changes by clicking on the parent test case, navigator screen or screen definition object (the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case, navigator screen or screen definition object.

The middle panel is a reference image to allow modification of the rectangle used. In Test Creator it is empty by default - you need to select or capture an image using the ribbon. For Navigator and Screen Editors, the reference image is the original screen that is being modified and cannot be altered in the trainer.

The image comparison can be in one of two modes: icon mode or full image mode. In icon mode, the reference image is considered to be an 'icon' - a small image which should match part of the captured screen. In non icon mode, the reference image is assumed to be full size image and then the rectangle selects the (same) portion of each image to be compared. This graphic is shown when in icon mode. The Navigator always works in Icon mode so this graphic will always be present.

The lower panel is the live comparison plot.

7.3.2.1 Ribbon Commands

The Test Creator version of the ribbon is shown and described here. The Navigator and Screen Editor versions of the image trainer do not have the `Image` group but are otherwise the same.

7.3.2.1.1 Clipboard Group

This is the standard copy/cut/paste/delete grouping.

7.3.2.1.2 Image Group

This is only available if the image trainer has been invoked from Test Creator. Because a Test Creator block has a distinct embedded reference image and could be used on any screen from the DUT during a test, you can select the reference image from the trainer.

Select: Select an image from the file system to use as the reference image.

Save: Save the current image to the local file system. All captured files must be saved some time in order to use them later in a test block.

Capture: Capture an image from the DUT and use that as the reference image.

Crop: Crop the image to the selected rectangle. After this, icon mode for the trainer is automatically selected.

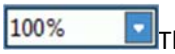
7.3.2.1.3 Zoom Group

Zooms the reference image. This will be disabled if there is no reference image.

 Zoom in - makes the screen larger, showing more detail.

 Zoom out - makes the screen smaller, showing less detail but more of the screen.

 100% - makes the screen 1:1 scaled so that you see the actual captured resolution of each screen.



The current zoom level is shown as a % of full size and can be changed directly

 Fit Page - scales the reference image so that it fills the area available on screen.

7.3.2.1.4 Results Group

This controls the results.

Clear Chart: Clears the live comparison plot, restarting the plot from the left of the diagram.

Auto Tune: Turns auto tune on or off. The auto tune algorithm attempts to find a suitable threshold such that the image block always passes. It looks over the last 50 samples and finds a threshold that always passes and then applies a small adjustment equal to the standard deviation of the actual match over those samples. This should allow a suitable margin for error. Under no circumstances will the algorithm set a threshold less than 50%.

Apply Offset: If the connected DUT has a video offset defined in the database, you can choose to use that offset in the trainer or not. You should use it if you are simply testing a block which was defined on another DUT. You should NOT use the offset if you are designing a test to run on the DUT in use. A warning appears near the live comparisons if the video offset is applied. The button is disabled if the database has no video offset for the DUT.

7.3.2.1.5 Resources Group

This allows you limited control over the resources used for this image test.

Convert To: Brings up a sub menu to select a resource type. This converts the selected rectangle or image to a named resource.

Clear All: Clears the resources from the image test. The region and reference image are replaced with the contents of the named resource and can then be edited.

7.3.3 Selecting a Region

You can adjust the position of the selected region using the mouse. The keyboard arrow keys can move the selection by 1 pixel at a time.

The equivalent area is shown on live video as highlighted.

7.3.4 Live Comparison

The plot shows the comparison value against time. It is color coded as well. The dotted orange line shows the threshold value. Plots above this line 'pass' and below 'fail'. You may at times want the image compare to fail.

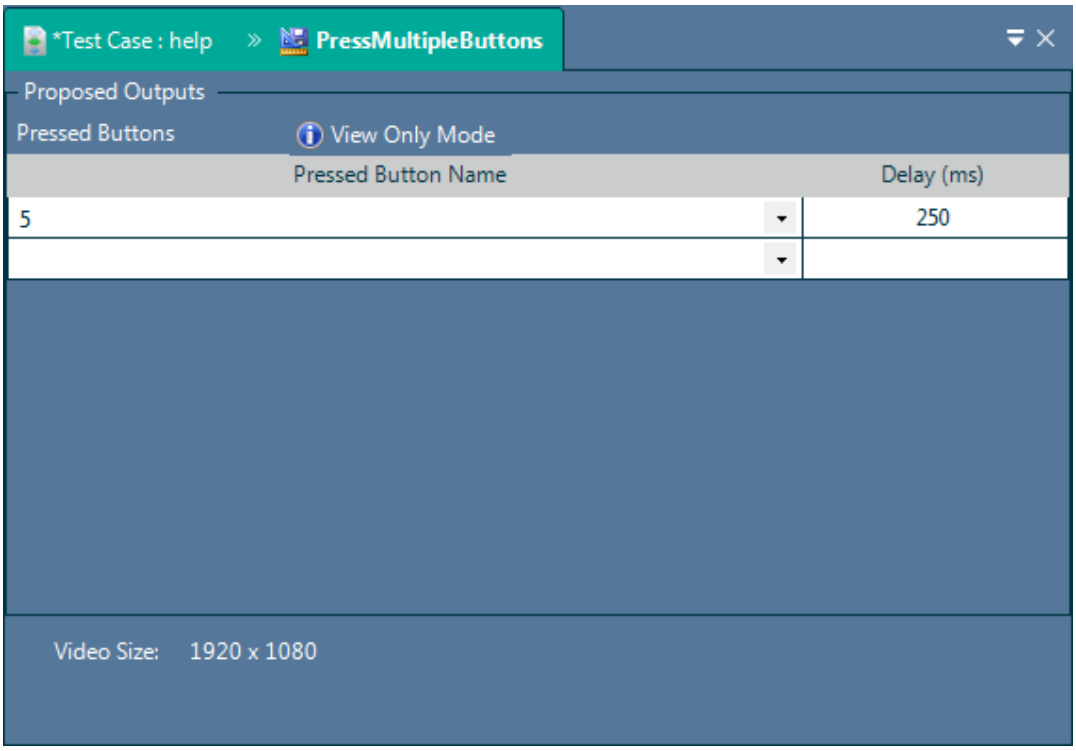
7.3.5 Auto Tuning

When auto tuning, the trainer calculates the image difference over the last number of samples (maximum 100 and all on graph), and sets the threshold such that it always passes - with a little margin to spare.

Clearing the graph resets the tuning which is useful if you have just changed the reference image or video via the remote control.

7.4 Press Multiple Button Trainer

When you select a link in Navigator, you can invoke the trainer by double clicking the link or from the RemoteControlKeyPress item on the properties grid. When you select the `Press Multiple Buttons` block in Test Creator, you can invoke the trainer by using the `Trainer` button on the ribbon, the right mouse context menu, the keyboard shortcut ALT+T followed by T or selecting any input property and selecting `Launch Trainer` after clicking the drop down to show the reference links. The trainer appears as a breadcrumb tab in the same tab as was previously open. The image below shows the Test Creator version of the trainer opened from a test case `help`.



7.4.1 Purpose

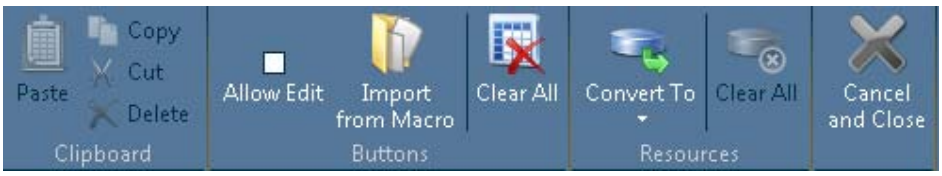
The Press Multiple Button trainer is designed to help you enter multiple buttons with individual delays for the link, the start screen or a Press Multiple Buttons block.

7.4.2 Usage

The upper panel is the proposed output settings from the trainer. Close the trainer and confirm changes by clicking on the parent test case or navigator screen (the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case or navigator screen.

To prevent accidental editing, the list of buttons is disabled on invocation of the trainer. This allows you to use the remote control to navigate to a screen on the DUT without the trainer recording the keystrokes. Use the ribbon controls to enable editing.

7.4.3 Ribbon Commands



7.4.3.1.1 Clipboard Group

This is the standard copy/cut/paste/delete grouping.

7.4.3.1.2 Buttons Group

Allow Edit: Check/uncheck this to allow the button list to be edited and the remote control to add buttons to the list. When in edit mode you can add new buttons by selecting the button to add from the lowest dropdown list. You can also use the remote control to add buttons. You can modify an existing button or delay time. To delete a button highlight the row in the list and use the `delete` key on the keyboard.

Import from Macro: Use this button to select a macro file to import - the buttons will be appended on to the existing set of buttons.

Clear All: Clear all buttons.

7.4.3.1.3 Resources Group

This allows you limited control over the resources used for the button list.

Convert To: Brings up a sub menu to select a resource type. This converts the buttons to a string named resource. The format is the same as the .mac file.

Clear All: Clears the resources from the button list replacing them with the contents as literal values. It also enables edit mode so you can edit them.

7.5 Detect Motion Trainer

When you have a Detect Motion test you can invoke the trainer by using the `Trainer` button on the ribbon, the keyboard shortcut `ALT+T` followed by `T` or selecting any input property and selecting `Launch Trainer` after clicking the drop down box. The drawing below shows the Test Creator version of the trainer opened from a Detect Motion block of the test case `help`.

7.5.1 Purpose

The detect motion trainer is designed to help you select the parameters for the detect motion test.

7.5.2 Usage

The upper panel is the proposed output threshold from the trainer. Close the trainer and confirm changes by clicking on the parent test case, navigator screen or screen definition object (the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case, navigator screen or screen definition object.

The middle panel is a reference image to allow modification of the rectangle used. In Test Creator it is empty by default - you need to select or capture an image using the ribbon. For Navigator and Screen Editors, the reference image is the original screen that is being modified and cannot be altered in the trainer.

The lower panel is the live comparison plot.

7.5.2.1 Ribbon Commands

The Test Creator version of the ribbon is shown and described here. The Navigator and Screen Editor versions of the detect motion trainer do not have the `Image` group but are otherwise the same.

7.5.2.1.1 Clipboard Group

This is the standard copy/cut/paste/delete grouping.

7.5.2.1.2 Image Group

This is only available if the detect motion has been invoked from Test Creator. Because a Test Creator block has no embedded reference image and could be used on any screen from the DUT during a test, there can be no automatic reference image. The reference image is optional - it can be useful to adjust the region. You can use the live video to select a region, however, only the reference image supports zooming so if fine control of the region is needed, you should set the reference image.

Select: Select an image from the file system to use as the reference image.

Capture: Capture an image from the DUT and use that as the reference image.

7.5.2.1.3 Zoom Group

Zooms the reference image. This will be disabled if there is no reference image.

Zoom in - makes the screen larger, showing more detail.

Zoom out - makes the screen smaller, showing less detail but more of the screen.

100% - makes the screen 1:1 scaled so that you see the actual captured resolution of each screen.

100% The current zoom level is shown as a % of full size and can be changed directly

 Fit Page - scales the reference image so that it fills the area available on screen.

7.5.2.1.4 Results Group

This controls the results.

Clear Chart: Clears the live comparison plot, restarting the plot from the left of the diagram.

7.5.2.1.5 Resources Group

This allows you limited control over the resources used for this detect motion test.

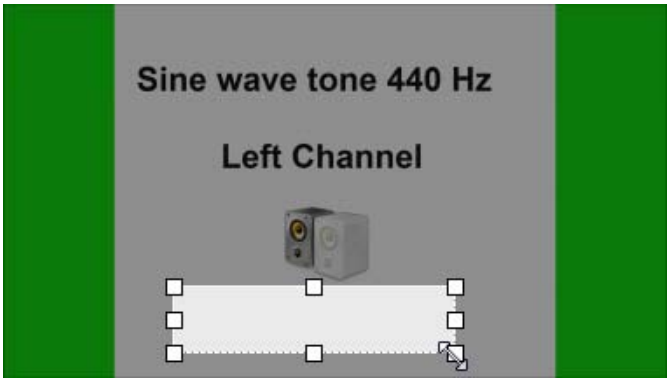
Convert To: Brings up a sub menu to select a resource type. This converts the selected rectangle to a named resource.

Clear All: Clears the resources from the image test. The region is replaced with the contents of the named resource and can then be edited.

7.5.3 Selecting a Region

You can adjust the position of the selected region using the mouse. The keyboard arrow keys can move the selection by 1 pixel at a time.

The equivalent area is shown on live video as highlighted.



Beneath the plot, you will see a textual representation of the video size being streamed and the selected region.

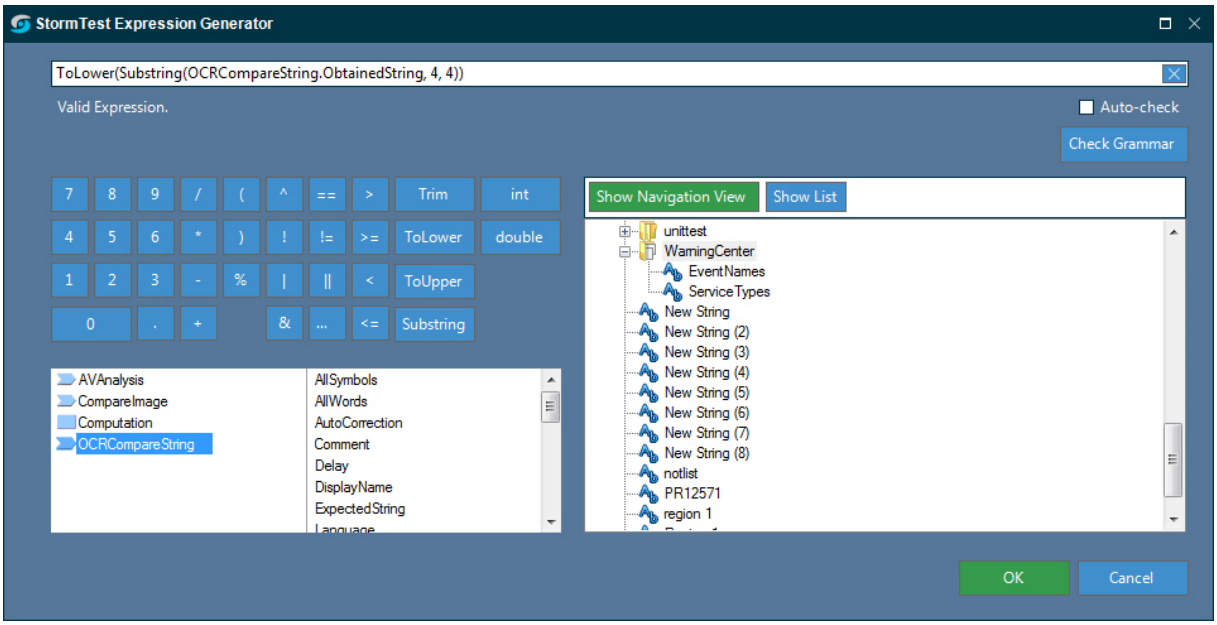
7.5.4 Live Comparison

The graph shows the degree of motion as a line chart in real time. The orange dotted line is the threshold that has been set.

7.6 Computation Trainer

When you select the `Computation`, `Decision` or `Loop` blocks, you can invoke the trainer by using the `Trainer` button on the ribbon, the right mouse context menu, the keyboard shortcut

ALT+T followed by T or selecting any input property and selecting Launch Trainer after clicking the drop down to show the reference links.



7.6.1 Purpose

The Computation trainer is designed to help you create expressions for the Computation, Decision and Loop blocks. It is a modal form - while active, you cannot work on another window in Developer Suite.

7.6.2 Usage

The trainer window is a simple dialog modelled on the Windows Calculator. You can enter an expression directly or by pressing the buttons. You can see all the functions available on the screen. Pressing them will insert the function into the expression and put in names for the parameters - you need to edit those to supply real values. You can insert links to blocks and properties either by typing in exactly the block and property or by clicking in the list in the bottom right of the screen.

7.6.2.1 Auto Check

If you check this check box then as you enter the expression you will see a message telling you whether the expression is valid or not.

7.6.2.2 Check Grammar

When not using auto check, you can check whether the expression is valid by just clicking this button.

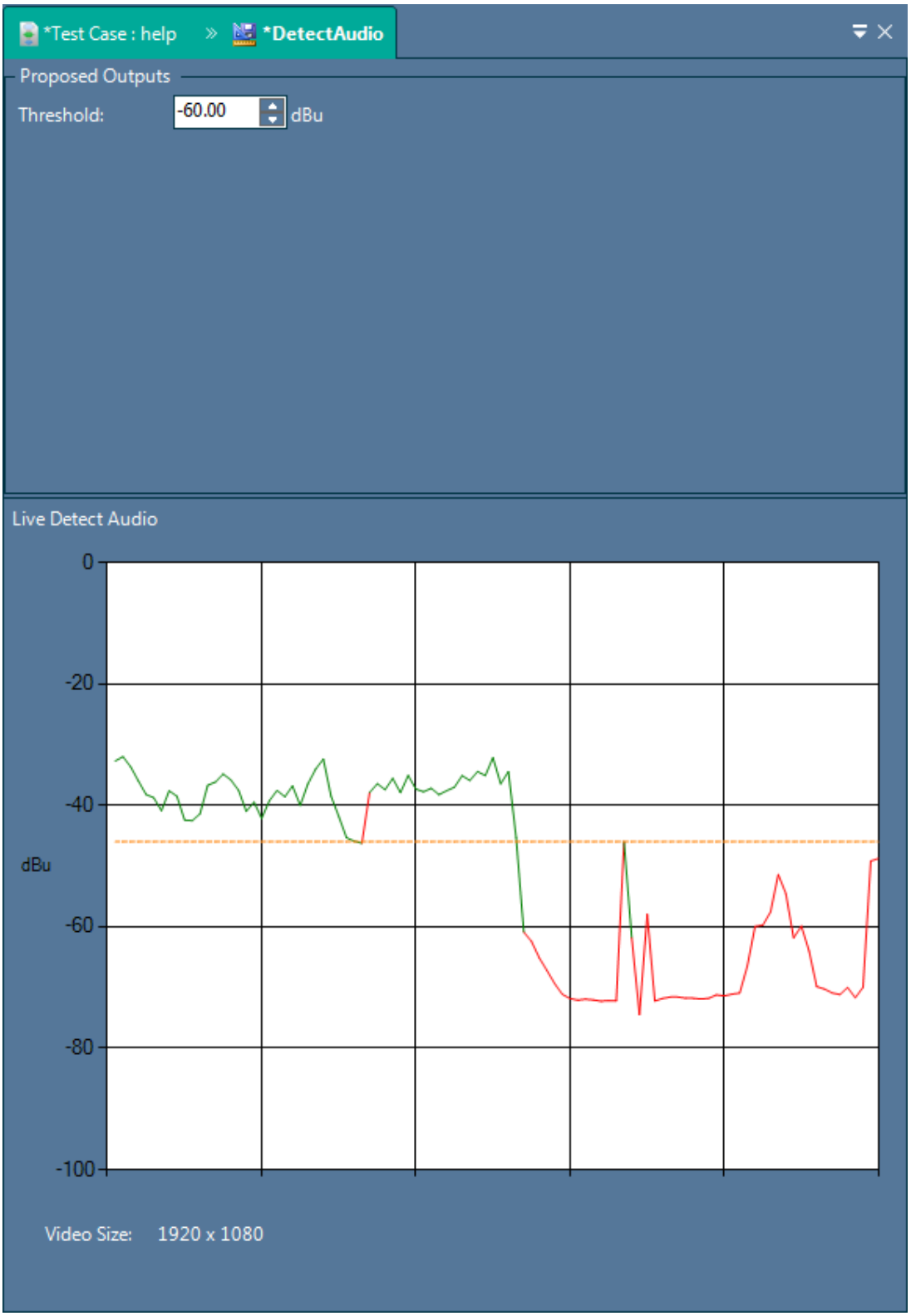
7.6.2.3 Resources

Instead of a string element, you can use a string resource. You can also type in a resource such as:

```
[res:/Projects/Tutorial/Tutorial6/TitleStrings]
```

7.7 Detect Audio Trainer

When you select the `detect audio` block, you can invoke the trainer by using the `Trainer` button on the ribbon, the right mouse context menu, the keyboard shortcut ALT+T followed by T or selecting any input property and selecting `Launch Trainer` after clicking the drop down to show the reference links. The drawing below shows the Test Creator version of the trainer opened from a Detect Audio block of the test case `help`.



7.7.1 Purpose

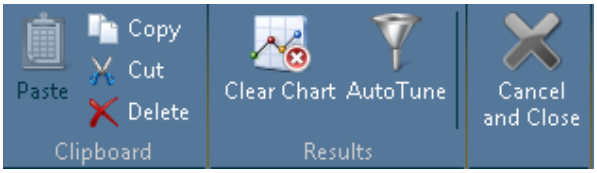
The detect audio trainer is designed to help you select the threshold parameter for the detect audio block.

7.7.2 Usage

The upper panel is the proposed threshold from the trainer. This can be set manually or calculated automatically using the Auto Tune option. Close the trainer and confirm changes by clicking on the parent test case, navigator screen or screen definition object (the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case, navigator screen or screen definition object.

The lower panel is the live comparison plot.

7.7.2.1 Ribbon Commands



7.7.2.1.1 Clipboard Group

This is the standard copy/cut/paste/delete grouping.

7.7.2.1.2 Results Group

This controls the results.

Clear Chart: Clears the live comparison plot, restarting the plot from the left of the diagram.

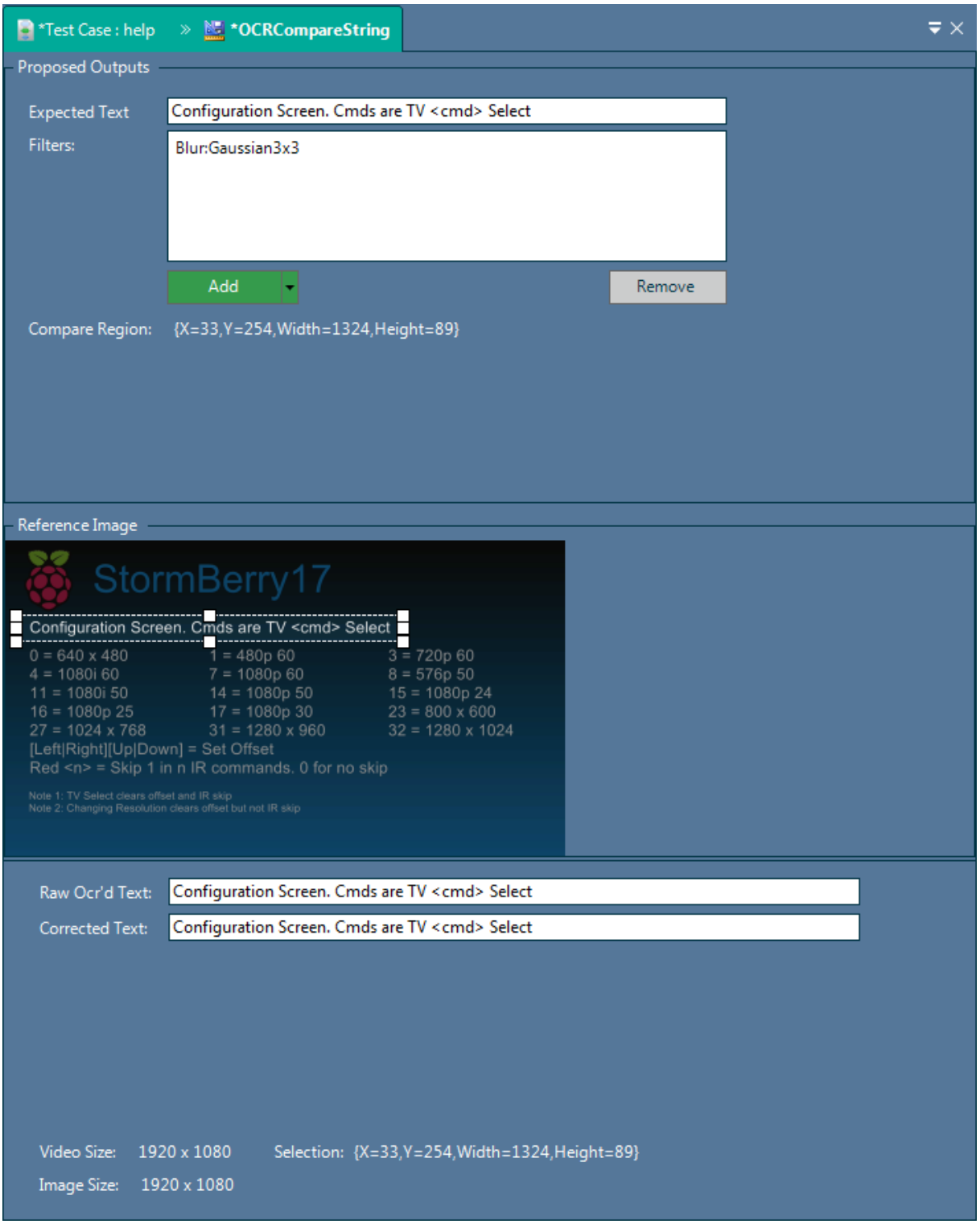
Auto Tune: Turns auto tune on or off. In this mode, the trainer analyzes the audio and sets up the threshold in such a way that the detect audio always passes.

7.7.3 Live Detect Audio

The graph shows the audio level as a line chart in real time. The orange dotted line is the threshold that has been set. The line is color coded as green above the threshold and red below.

7.8 OCR Trainer

When you have an OCR Compare String test, you can invoke the trainer by using the `Trainer` button on the ribbon, the keyboard shortcut ALT+T followed by T or selecting any input property and selecting `Launch Trainer` after clicking the drop down box. The trainer appears as a breadcrumb tab in the same tab as was previously open. The drawing below shows the Test Creator version of the trainer opened from a OCR Compare String block of the test case `help`.



7.8.1 Purpose

The OCR trainer is designed to help you select the parameters for the OCR compare string test.

7.8.2 Usage

The upper panel is the proposed output settings from the trainer. Close the trainer and confirm changes by clicking on the parent test case, navigator screen or screen definition object (the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case, navigator screen or screen definition object.

The middle panel is a reference image to allow modification of the rectangle used. In Test Creator it is empty by default - you need to select or capture an image using the ribbon. For Navigator and Screen Editors, the reference image is the original screen that is being modified and cannot be altered in the trainer.

The lower panel is the output of performing an OCR on the live image.

7.8.2.1 Ribbon Commands

The Test Creator version of the ribbon is shown and described here. The Navigator and Screen Editor versions of the color trainer do not have the `Image` group but are otherwise the same.

7.8.2.1.1 Clipboard Group

This is the standard copy/cut/paste/delete grouping.

7.8.2.1.2 Image Group

This is only available if the OCR trainer has been invoked from Test Creator. Because a Test Creator block has no embedded reference image and could be used on any screen from the DUT during a test, there can be no automatic reference image. The reference image is optional - it can be useful to adjust the region. You can use the live video to select a region, however, only the reference image supports zooming so if fine control of the region is needed, you should set the reference image.

Select: Select an image from the file system to use as the reference image. It must be of the same size as the DUT screen or else the coordinates will not make sense.

Capture: Capture an image from the DUT and use that as the reference image.

7.8.2.1.3 OCR Group

Perform OCR: This performs an immediate OCR on the live screen, updating the results in the lower panel.

OCR Image: This performs an OCR on the reference image, updating the results in the lower panel.

Filter Trainer: This brings up the auto filter selection dialog.

Apply Filters: Shows the effect of the filter processing on the captured image. It should be noted that the legacy invert filter cannot be shown here because it is implemented inside the OCR engine and the result is not shown to outside code.

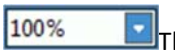
7.8.2.1.4 Zoom Group

Zooms the reference image. This will be disabled if there is no reference image.

 Zoom in - makes the screen larger, showing more detail.

 Zoom out - makes the screen smaller, showing less detail but more of the screen.

 100% - makes the screen 1:1 scaled so that you see the actual captured resolution of each screen.



The current zoom level is shown as a % of full size and can be changed directly

 Fit Page - scales the reference image so that it fills the area available on screen.

7.8.2.1.5 Results Group

This controls the results.

Apply Offset: If the connected DUT has a video offset defined in the database, you can choose to use that offset in the trainer or not. You should use it if you are simply testing a block which was defined on another DUT. You should NOT use the offset if you are designing a test to run on the DUT in use. A warning appears near the live comparisons if the video offset is applied. The button is disabled if the database has no video offset for the DUT.

7.8.2.1.6 Resources Group

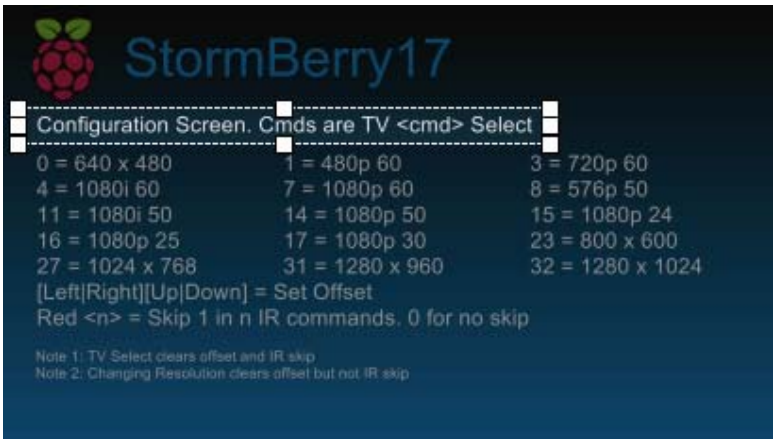
This allows you limited control over the resources used for this color test.

Convert To: Brings up a sub menu to select a resource type. This converts the selected rectangle or string to a named resource.

Clear All: Clears the resources from the color test. The region and expected text are replaced with the contents of the named resource and can then be edited.

7.8.3 Selecting a Region

You can select a region for OCR comparison using the mouse on the still image window. You can adjust a selected region using the mouse. Clear it by clicking outside the region. The keyboard arrow keys can move the selection by 1 pixel at a time:



7.8.4 Expected Text

You can select the expected text. If the OCR does not match this, then it fails.

7.8.5 Filters

Various image filters can be applied to improve the image comparison. Add as desired. Each filter will take time to work, especially on the whole image.

7.8.6 Raw OCR'd Text

After doing the OCR, the actual text is shown, along with statistics about it. Also, whether the block passed or failed.

7.8.7 Corrected Text

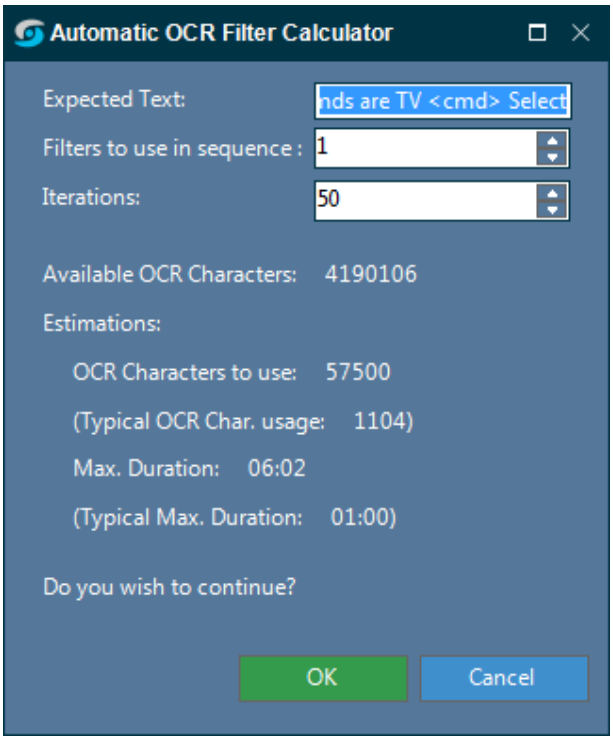
If a correction has been specified in the properties, then the result of correcting the text is displayed here. If no correction has been specified then the raw and corrected text are identical.

7.8.8 Auto Filter Selection

By requesting automatic filter selection, the OCR trainer will select filters in turn, perform an OCR and determine the best combination. This can take some time and can use quite a lot of the available OCR licensed characters depending on the screen.

7.8.8.1 Filter Settings.

Before performing the auto selection, a dialog allows you to select the settings to use.



Expected Text: This is the text that you expect in the selected rectangle. ***Caution***: If you make a mistake here it is likely that no screen will ever match and you will spend a lot of time and characters for no good purpose. The text is case sensitive.

Filters to Use in sequence: The total length of the sequence of filters to apply. The default is 1 which will often be enough. Longer numbers take exponentially more time. There is a limit of 5. That could, in theory take up to 20 hours.

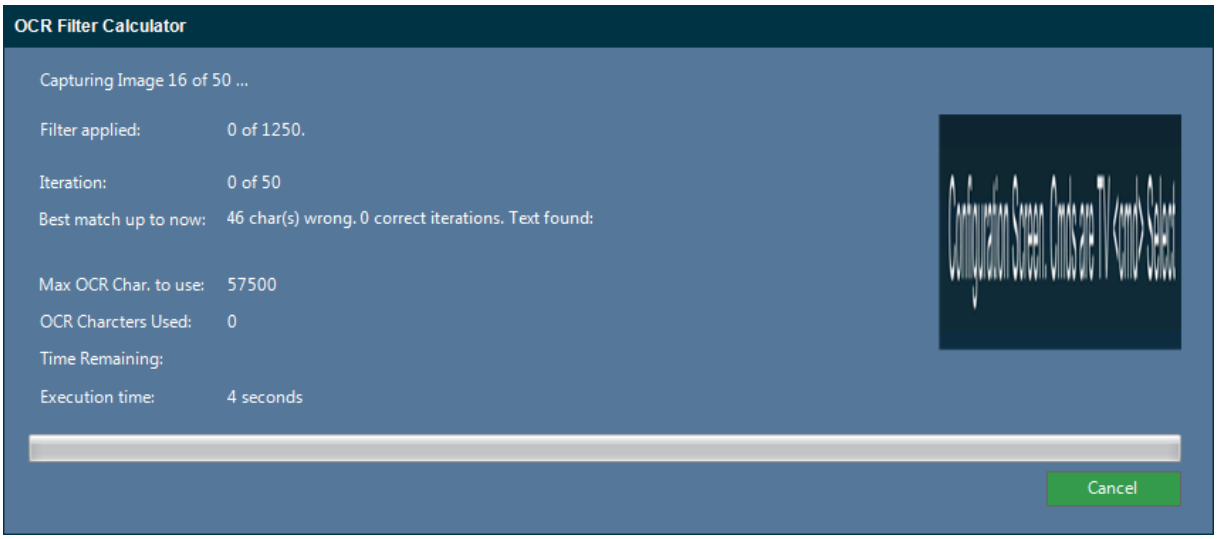
Iterations: The number of images to use. The trainer grabs this number of images (which it assumes are all the same picture) and tries the filters on each one. This is because each capture can vary slightly due to the nature of analog capture. There is a maximum of 100.

Estimations: This is the estimate of usage. The max possible OCR characters used, a typical value that often occurs, the maximum duration and a typical duration. The typical values are those experienced in the lab with reasonably good video sources and no guarantee is made. The values are updated during progress.

Algorithm: The trainer first tries zero filters, then one filter, then all combination of 2 filters and so on. The order of filters tried is based on experience of most likely to work. If all images pass with any combination, then the training stops as that is considered a good combination.

7.8.8.2 Filter Progress

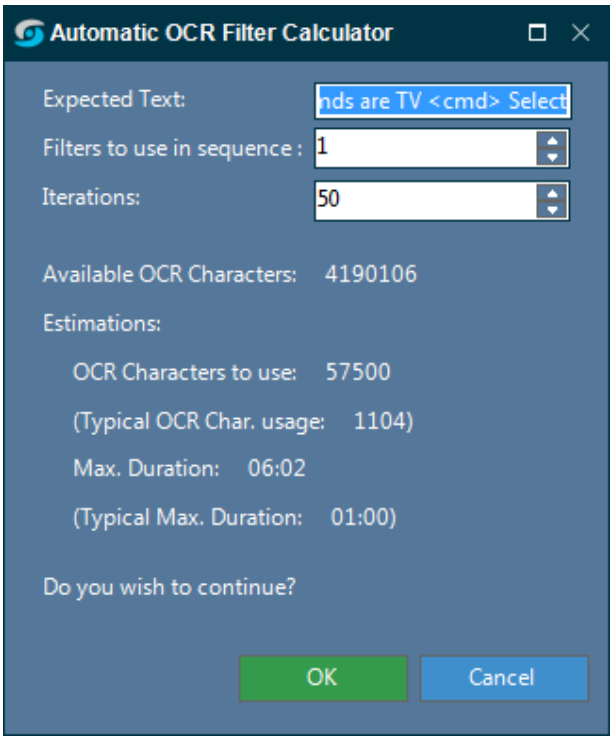
During the filtering a progress bar is shown:



The image resulting in each filtering operation is shown during progress along with an update of timing.

7.9 Automatic Filter Selection Dialog

Before running the OCR trainer in automatic mode, you are given the opportunity to adjust the settings used:



Expected Text: This is the text that you expect in the selected rectangle. ***Caution***: If you make a mistake here it is likely that no screen will ever match and you will spend a lot of time and characters for no good purpose. The text is case sensitive.

Filters to Use in sequence: The total length of the sequence of filters to apply. The default is 1 which will often be enough. Longer numbers take exponentially more time. There is a limit of 5. That could, in theory take up to 20 hours.

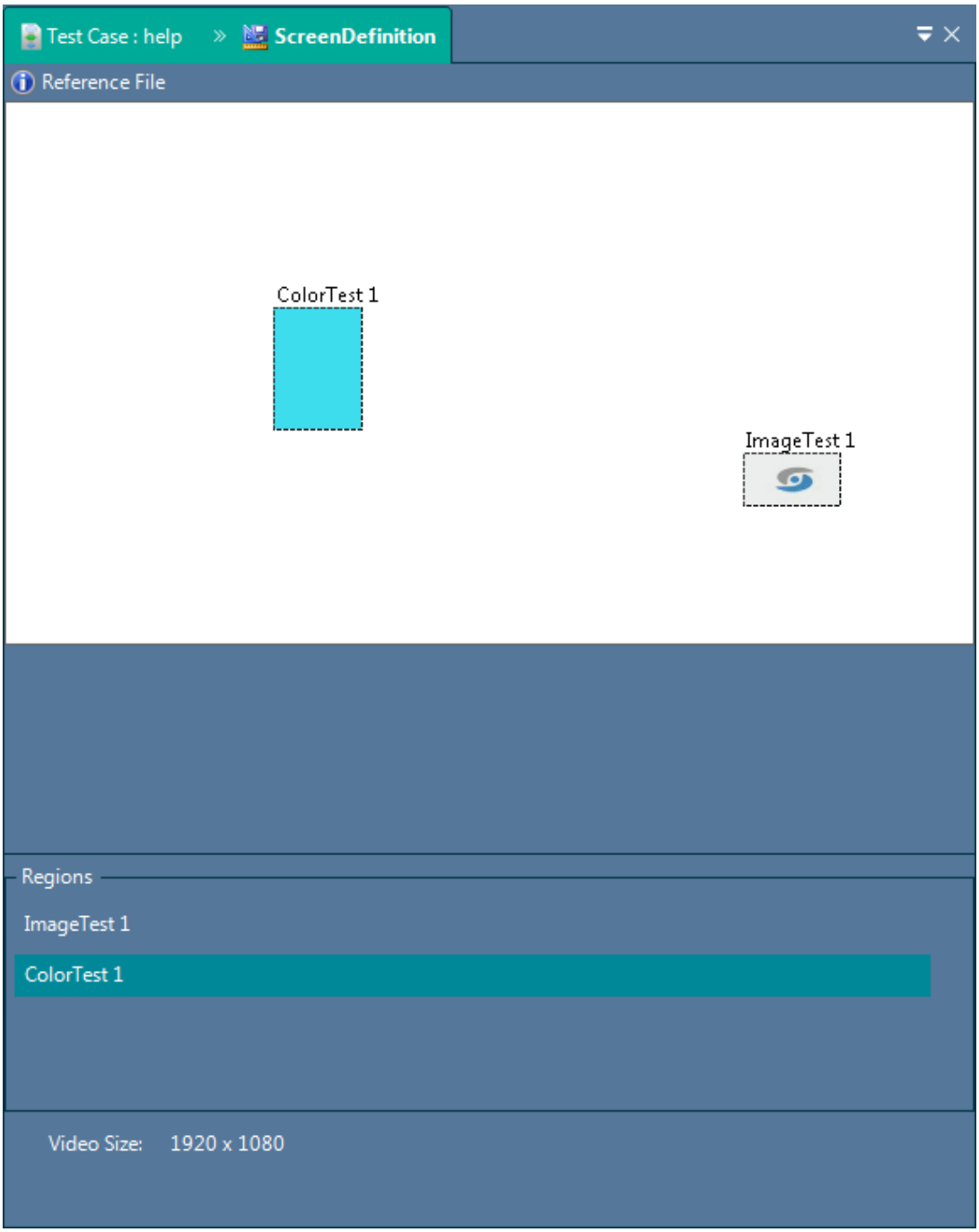
Iterations: The number of images to use. The trainer grabs this number of images (which it assumes are all the same picture) and tries the filters on each one. This is because each capture can vary slightly due to the nature of analog capture. There is a maximum of 100.

Estimations: This is the estimate of usage. The max possible OCR characters used, a typical value that often occurs, the maximum duration and a typical duration. The typical values are those experienced in the lab with reasonably good video sources and no guarantee is made. The values are updated during progress.

Algorithm: The trainer first tries zero filters, then one filter, then all combination of 2 filters and so on. The order of filters tried is based on experience of most likely to work. If all images pass with any combination, then the training stops as that is considered a good combination.

7.10 Screen Definition Object Trainer

When you have a Screen Definition test, you can invoke the trainer by using the `Trainer` button on the ribbon, the keyboard shortcut `ALT+T` followed by `T` or selecting any input property and selecting `Launch Trainer` after clicking the drop down box. The trainer appears as a breadcrumb tab in the same tab as was previously open. The image below shows the Test Creator version of the trainer opened from a Screen Definition block calls `Screen_guide` of the test case `tc`.



7.10.1 Purpose

The Screen Definition trainer is designed to help you see inside a screen definition. It shows you the component tests. If you wish to edit the screen you need to invoke the screen definition object editor.

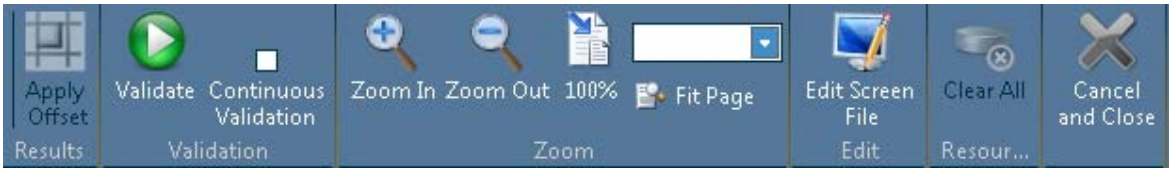
7.10.2 Usage

The upper panel is the screen definition. In the example above, it shows that it was imported from a file. Close the trainer and confirm changes by clicking on the parent test case or navigator screen

(the left hand item of the highlighted tab). Use the ribbon `Cancel and Close` button to cancel any proposed changes of the trainer. The small ☒ button closes the test case or navigator screen.

The lower panel is the list of regions in the screen definition object. They also show the status of the validation if you validate the screen in this trainer.

7.10.2.1 Ribbon Commands



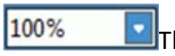
7.10.2.1.1 Zoom Group

Zooms the screen definition display.

Zoom in - makes the screen larger, showing more detail.

Zoom out - makes the screen smaller, showing less detail but more of the screen.

100% - makes the screen 1:1 scaled so that you see the actual captured resolution of each screen.



The current zoom level is shown as a % of full size and can be changed directly

Fit Page - scales the screen so that it fills the area available on screen.

7.10.2.1.2 Validation Group

Validate: This will execute the screen definition once, with the results being visible in the lower panel.

Continuous Validation: If this is checked, then the trainer will run continuous validations on the screen definition. This can show up potential instabilities in any of the test - you might see occasional failures if the screen definition regions' settings are close to borderline pass conditions.

7.10.2.1.3 Edit Group

Edit Screen File: When the screen definition comes from a file, this button is enabled. It will launch the screen definition editor to allow you to edit the file. In Navigator, you can use a screen definition stored inside the Navigator file - this is considered a 'file' for the purposes of the trainer and you can edit it using the Edit Screen File button.

7.10.2.1.4 Resources Group

Clear All: If the screen definition is a resource, then this button is enabled to allow you to convert the resource to a literal value. You will need to save it to a file when using test creator as test creator only uses files or resources.

7.11 Resources in Trainers

Most of the trainers can be used with resources. For full editing of a resource, the Resource Editor should be used. The trainer is for selecting values to 'tweak' a test in Test Creator, Navigator or the Screen Definition Object editor.

7.11.1 Usage

The behavior is common and described here. The ribbon has 2 buttons:

 Convert the active element of the trainer to a resource. Trainers can have up to 2 active areas so a menu appears to allow you to choose which you want to convert. You will be prompted with a dialog for the folder and name of the resource to save. After conversion, the equivalent property is considered to be a resource and when you exit the trainer the underlying block is updated with the name of the resource. This button will also make the resources panel visible in the trainer.

 Clear all the resources. Any resources used in the trainer are removed and converted to their current values. This allows you to go back to using literal design time values in the trainer.

From the resources panel visible, you can click on any resource and it will be immediately copied to the appropriate property and used in the trainer. Only the relevant property types will be visible in the panel - it is filtered for you.

7.11.2 Trainers using Resources

All trainers that use resources can have the selected rectangle as a named region. The following trainers can use resources. The other fields that can use a resource are listed.

7.11.2.1.1 OCR Trainer

Expected Text (Named String)

7.11.2.1.2 Compare Image Trainer

Reference Image (Named Image)

7.11.2.1.3 Compare Color Trainer

Color, Tolerance, Flatness and Peak Error (Named Color)

7.11.2.1.4 Detect Motion Trainer

No additional properties can use resources

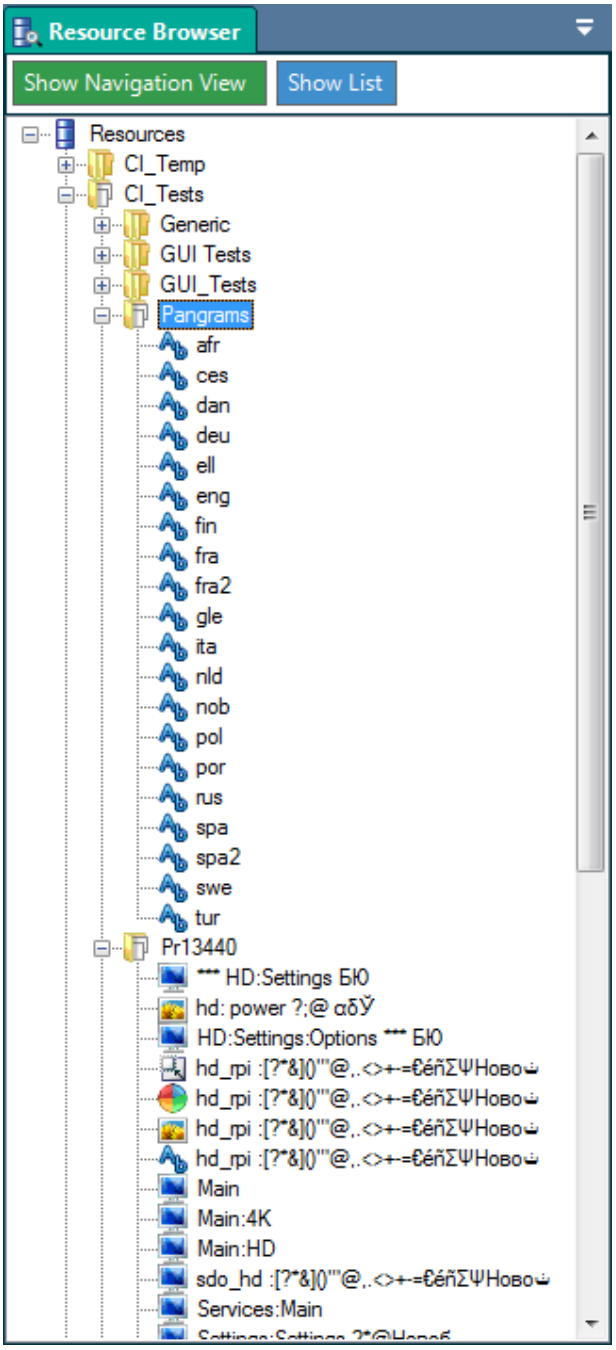
7.11.2.1.5 Screen Definition Trainer

Only embedded screen definition objects can be modified to use a screen definition resource.

8 Resources Editor

8.1 Main Resource Browser

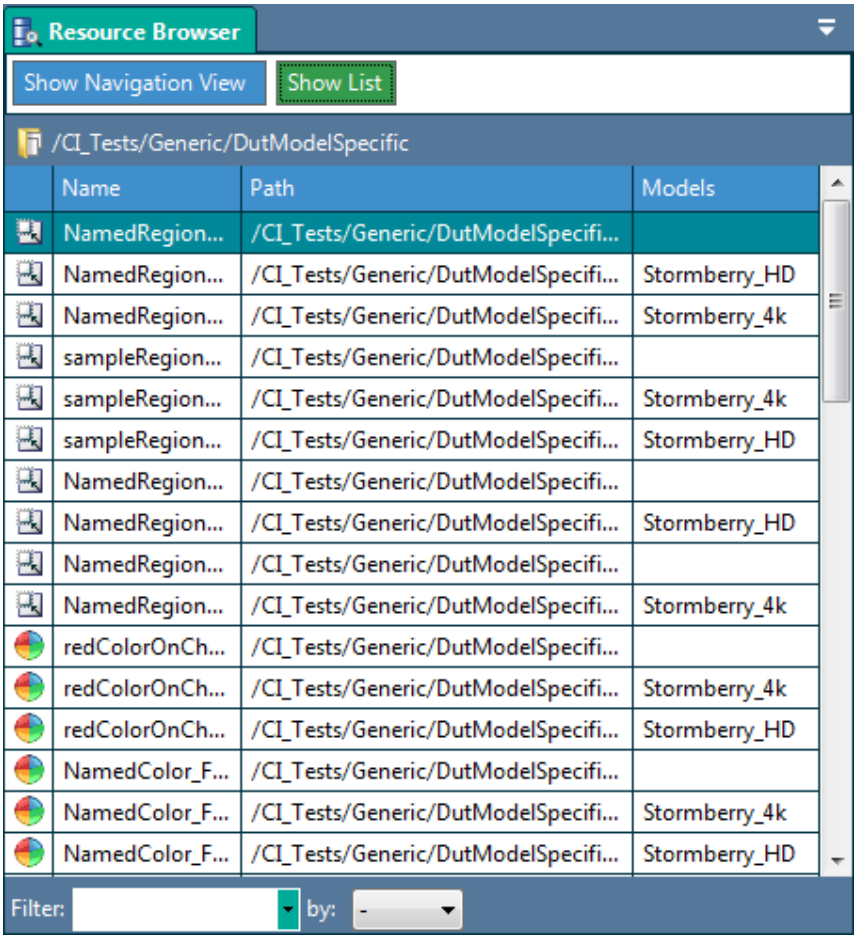
The main resource browser has two views. A tree view and a list view.



All the resources are shown. As you select resources, the resource details panel reflects the properties of the resource.

Model specific resources are shown as a DUT icon (the Background Color above) with the model name(s). You can edit these in the same manner as any resource. You add a variant from the right mouse menu. When editing a model specific resource you can add and detach models to the resource from the editor.

The buttons at the top of the panel allow for viewing as a tree or as a flat list with all the resources listed alphabetically by name, path or type. As you switch to list view, only those resources in the selected folder or sub folders are shown. This is to save network bandwidth when you may only be interested in a subset of the resources.



The lower filter panel allows you to specify search criteria for name, path, type or models. The search is case insensitive and starts at the beginning of the string. Wild cards are not supported so "Ti*" looks for all names beginning with Ti followed by an asterisk (ie ti*7 but not ti77).

A right menu allows the creation of new folders and resources while in tree view but not in list view.

8.2 Resource Details

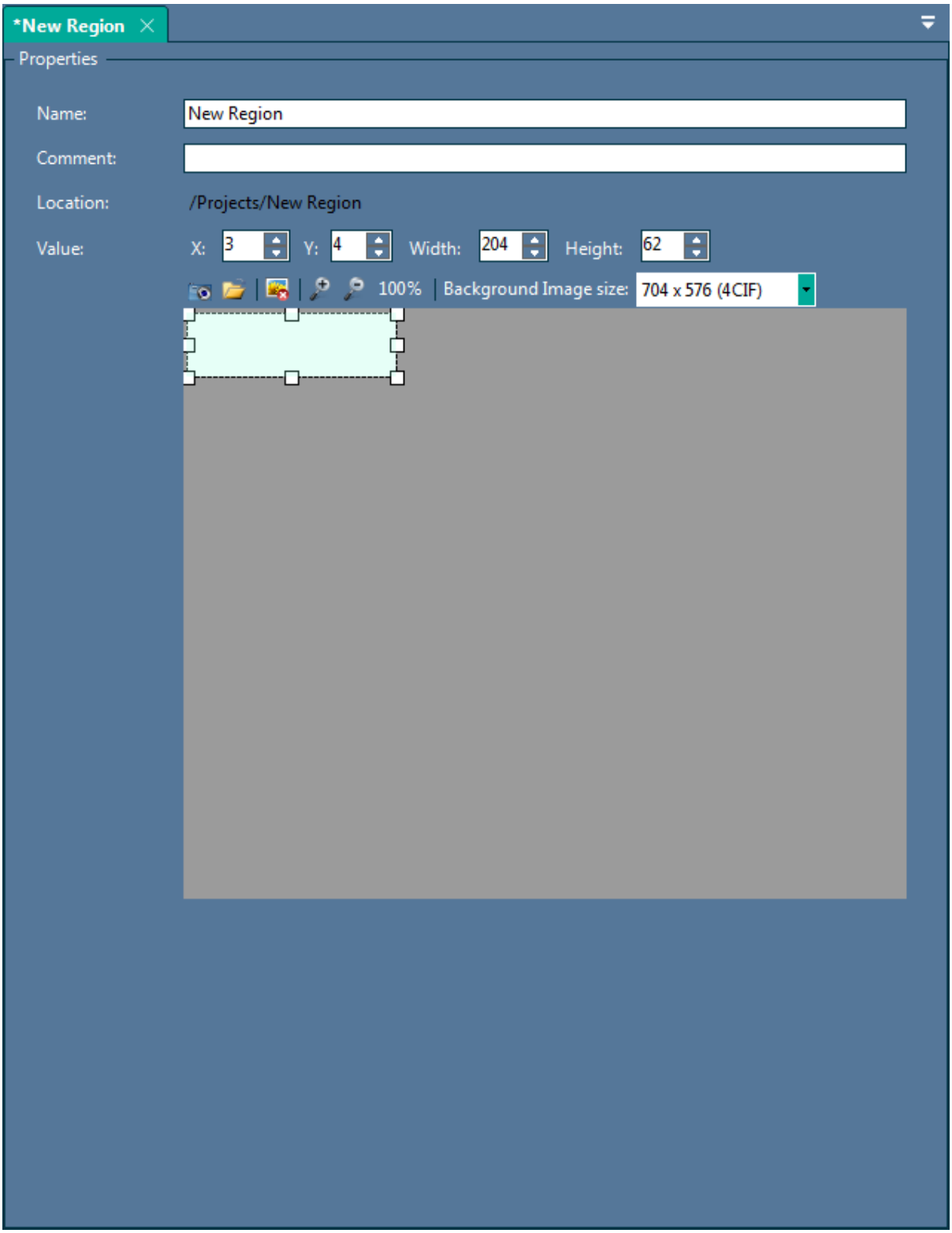
The resource details shows the details of the resource being edited. The layout depends on the resource. There are five views, one for each type of resource.

These are the Named Region view, the Named Image view, the Named Color view the Named String view and the Named Screen view.

All the editors allow changing the name of the resource and adding a comment about the resource.

8.3 Named Region Editor

The editor for named regions shows the region graphically as well as the numeric details.



8.3.1 Usage

You can enter the coordinates explicitly or you can use the mouse to draw the region. You can select a background image, for example a screen grab, to allow you to select the exact region. If you don't

select an image, the dark gray area represents the screen size - you can select the size of interest from the drop down list of supported StormTest image sizes.

The buttons below the values allow you to:

 capture an image from the DUT

 Load an image from a file

 Clear a loaded image

 Zoom in, making the view larger

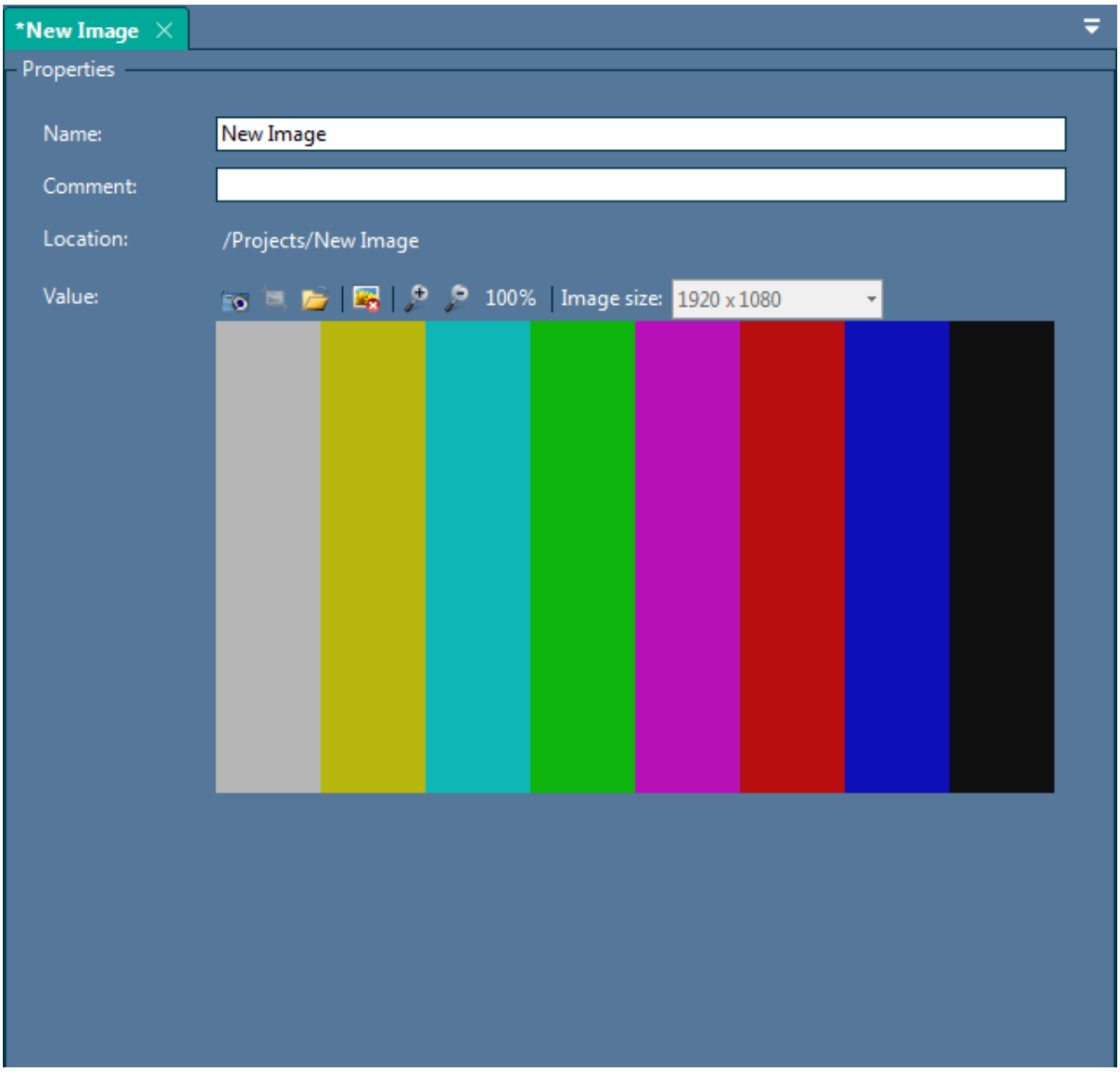
 Zoom out, making the view smaller

Reset to 100% making 1 pixel on screen equal to 1 pixel in the view of the background.

Select the background size - you can choose the background size so that you can see graphically the size and location of the region on different size DUT backgrounds.

8.4 Named Image Editor

The editor for named images shows the image graphically.



8.4.1 Usage

You can select an image from the DUT or from a file. You can select a region and crop the image. It is highly recommended that you store the smallest possible image as a named image, especially for HD systems where a full screen image can be 1MB or more in size.

The buttons below the values allow you to:

capture an image from the DUT

Crop the image to the selected region

Load an image from a file

Clear a loaded image

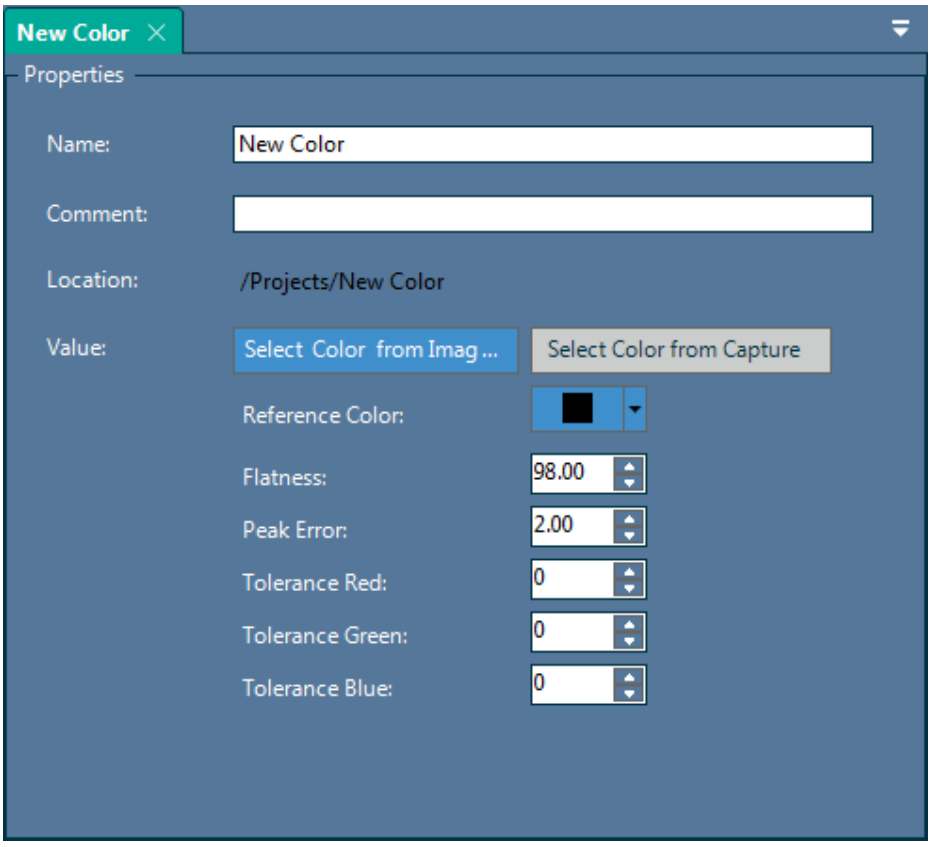
Zoom in, making the image larger

Zoom out, making the image smaller

100% Reset to 100% making 1 pixel on screen equal to 1 pixel in the image.

8.5 Named Color Editor

The editor for named colors shows the color graphically as well as the numeric details.



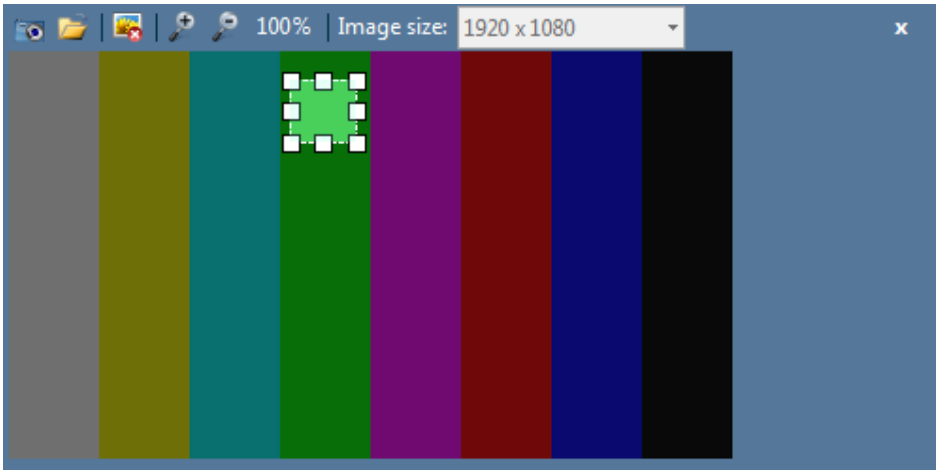
8.5.1 Usage

You can enter the color values manually. You can select a background image from the file system by clicking `Select Color From Image...` This will bring up a dialog to select an image. Just select the region of interest and the color values will be calculated for you.

If you have a DUT reserved then the `Select Color from Capture...` button is enabled. This will capture an image from the DUT and allow to select a region. The color values will be calculated automatically for you.

The flatness and peak error can be set to a resolution of 2 decimal places. The underlying code works to full double precision accuracy, however. If you are looking at a resource created by a programmer in Python, then the resolution of the flatness and peak error may be more than 2 decimal digits. However, the view on this screen is rounded to 2 decimal places. Any changes you make here will be saved to 2 decimal places.

8.5.1.1 Color Selector Tool



Once an image has been selected, either from the live video or the file system, you can select a region. The toolbar above the image allows you to:

Capture an image from the live video, replacing the existing image. It will be disabled if no DUT is reserved

Select an image from the file system, replacing the existing image.

Remove the image from the image panel

Zoom in, making the image larger

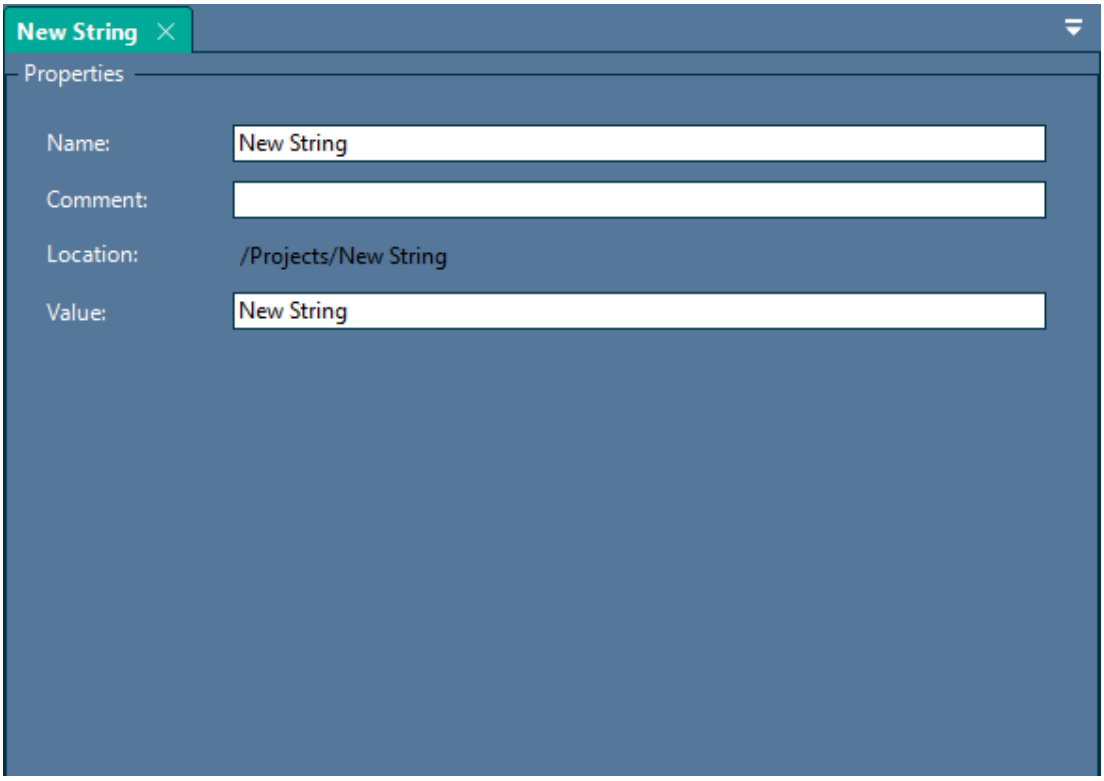
Zoom out, making the image smaller

100% Reset to 100% making 1 pixel on screen equal to 1 pixel in the image.

Closes the color selector tool completely.

8.6 Named String Editor

The named string editor is very simple.

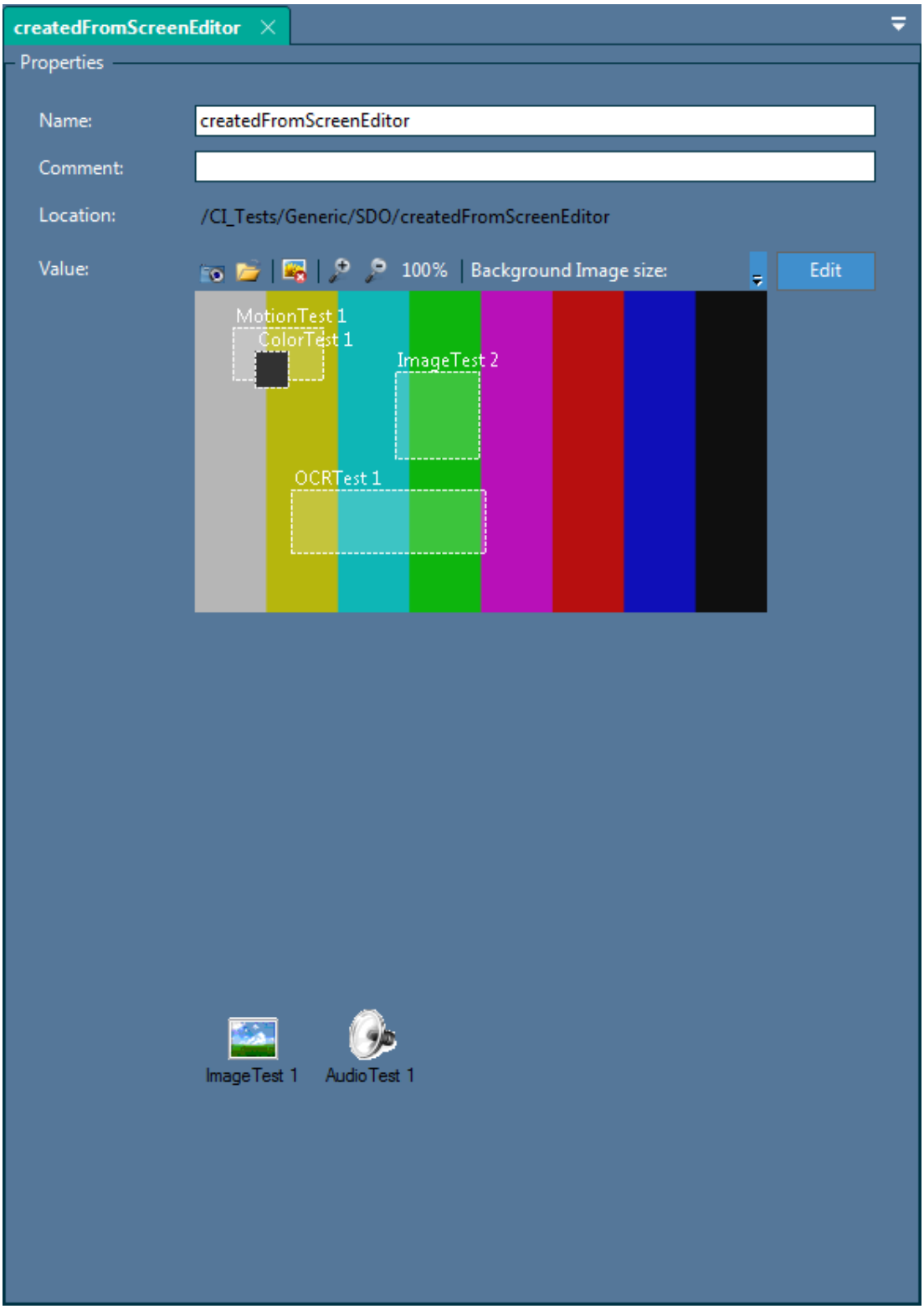


8.6.1 Usage

You can enter the value of the named string.

8.7 Named Screen Editor

The named screen editor is actually more of a preview than an editor:



8.7.1 Usage

You can view the screen definition object against different potential background images. This is particularly useful if the screen definition object has not been saved with a background image. The buttons allow you to:

 capture an image from the DUT and show it behind the screen definition object.

 Load an image from a file and show it behind the screen definition object.

 Clear a loaded image to see the screen definition object on a plain background.

 Zoom in, making the view larger

 Zoom out, making the view smaller

 Reset to 100% making 1 pixel on screen equal to 1 pixel in the view of the background.

Note: If the screen definition object was saved with its design time image, you can remove that image with the  button from the view. However, this does **not** alter the underlying resource and the design time image remains as originally saved. Likewise, loading a new image here does **not** alter the resource.

Click the Edit button to bring up the screen definition object editor to edit the screen definition. It appears as a separate window.

9 Remote Control

9.1 Remote Control Panel

When one or more DUTs are reserved, the remote control panel is visible. If more than one DUT is reserved with different remote controls you select which remote to use by clicking on the video window. However, when you send an IR command, it goes to all reserved DUTs - this may or may not work if you have a mix of DUTs reserved.

 this icon will appear for the duration of sending an IR command and as feedback that you clicked a button.

In addition the keyboard can be used to send commands. The following keys are supported:

0 - 9

left arrow

right arrow

up arrow

down arrow

Page Up (sends channel+)

Page Down (sends channel-)

+ (sends volume+)

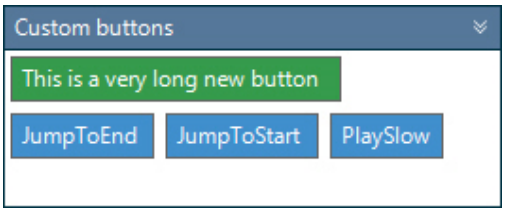
- (sends volume-)

The image used and whether or not you can actually send keys depends on the correct IR being trained and added to the StormTest Development Center database using the Admin Tools application.

If you click on a key with the right mouse button a menu appears allowing you to copy the key name to the clipboard. The name copied is the name of the button that would be needed in a Python script.

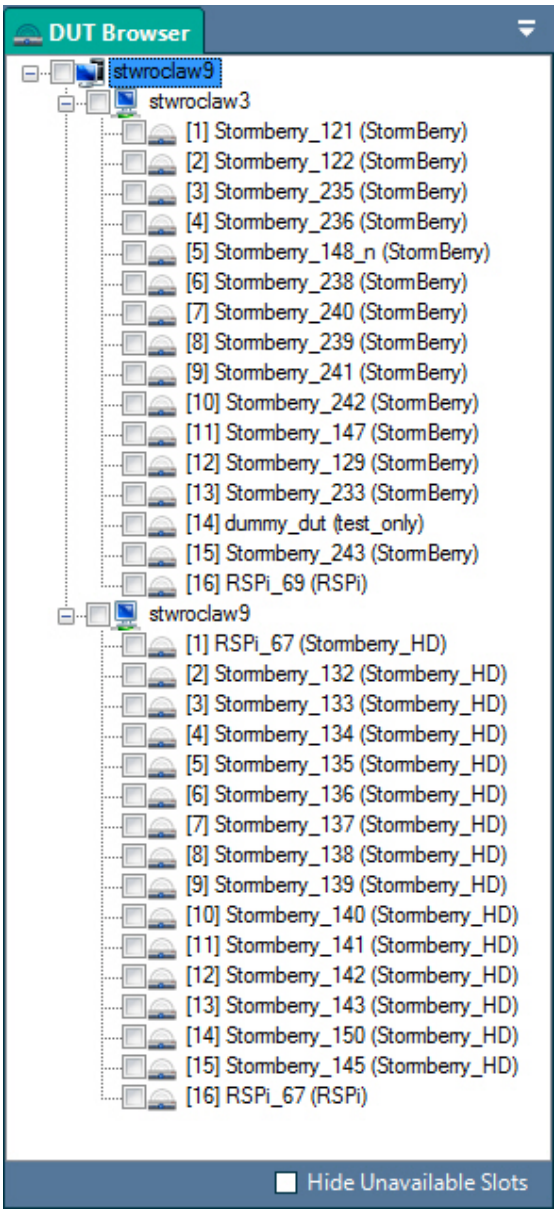
9.1.1 Custom Buttons

Some IR remote controls have extra custom buttons that are not distinct button on the remote control. For example, some IR remotes behave in a special way if you hold down a key for a long time. Such button appear in the bottom panel and can be pressed using the mouse:



9.2 DUT Browser

The DUT browser shows you all the DUTs in the facility, organized by server:



9.2.1 Usage

If you check a DUT then the Remote Control will monitor video and serial data from the DUT. The video window is added to the video mosaic view. You can monitor any DUT. A DUT which is in use is shown in gray. If you hover over it, a tool tip appears showing you who is using the slot. A slot which is reserved to you but not actually in use will show as normal - you can use it as you reserved it. The tooltip will tell you that you have reserved it.

You can select all DUTs in a server by clicking the server - but as there are 16 DUTs per server, this can be a heavy load on your PC.

9.2.1.1 Hide Unavailable Slots

This is not checked by default. If you check it, then the display hides any DUT which is in use.

9.3 Video Mosaic

The video mosaic shows the video of every DUT that you are monitoring. As you add more DUTs, the video windows are resized and organized to fill the available space. By default the windows are laid out on a grid basis, each one being the same size.

9.3.1 Video Layouts

selects the layout of the video windows. 3 options are available:

9.3.1.1 Grid Layout

Every video window is the same size and positioned on a grid. This is the default

9.3.1.2 Frame Mosaic

This can be used with up to 13 video windows. One window is in the center and up to 12 others are arranged around it. You select a window to be the center (larger window) by double clicking the header to make it full size and then double clicking it again to put it to the center.

9.3.1.3 Side Mosaic

This can be used with up to 11 video windows. One window is on the left hand side, larger than the others with 10 windows around it. You select a window to be the larger window by double clicking the header to make it full size and then double clicking it again to put it to the left.

9.3.2 Video Control

Each video window has a control bar below it and above it.

9.3.2.1 Upper controls

The header acts as the selection mechanism for a video - it changes style. The Remote control changes to reflect the active video window. Double clicking the header will make the video as large as is possible, hiding all others. This is only relevant if more than one video is open.

 will open the serial data panel below the video window. Here you will see the serial data from the DUT. If the DUT has been reserved, you can fully control the serial data. If it has not been reserved you can set the line ending and character set so that the data can be correctly interpreted.

 Reserve the DUT. When it is reserved, the header changes style and clicking the same icon will release the DUT. This can be used as an alternative to the main ribbon.

 Maximize the video to be the only visible window.

 will close the video and stop monitoring the video. If the DUT has been reserved by the Remote Control applet then it will be unreserved.

9.3.2.2 Lower Controls

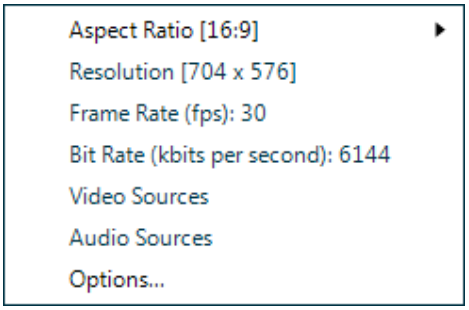
Some of these are only available when the DUT is reserved. The controls are:

 Audio mute. The mute state is shown. This is a toggle of the audio. If you hear no audio, check that your speaker on the PC is enabled as that can be muted and this control does not alter your PC audio status.

 Close the video stream. This has no effect on the reservation of the DUT.

 Open the video stream. Reverses the close

 Change the video settings. Most of these are via a menu:



But the full range is possible by selecting options.

 Grab a still image of the video - a normal file save dialog is presented

 Start saving the video to a local file - a normal file save dialog is presented

 Power status / control. The indicator is green as shown when the power is applied to DUT. Clicking it will turn the DUT off.

 Power status / control. The indicator is red as shown when the power is OFF. Clicking it will turn on the DUT.

 Change and view the sub slot in use on an HS64 Server. It brings up a menu to show the current sub slot and allow you to change the sub slot. This icon is not visible on non HS64 servers.

At the far right is the name of the server and the slot number of the video. If the video is an HD source then 2 more status icons appear:

is shown for all HD sources.

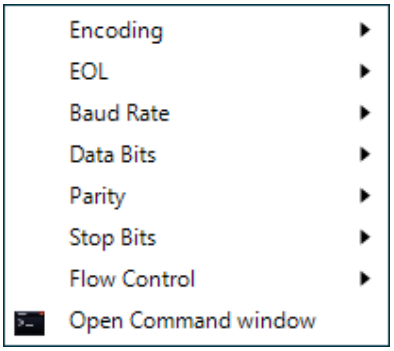
is shown if low bandwidth mode has been selected in the Preferences Applet.

9.3.2.3 Serial data panel

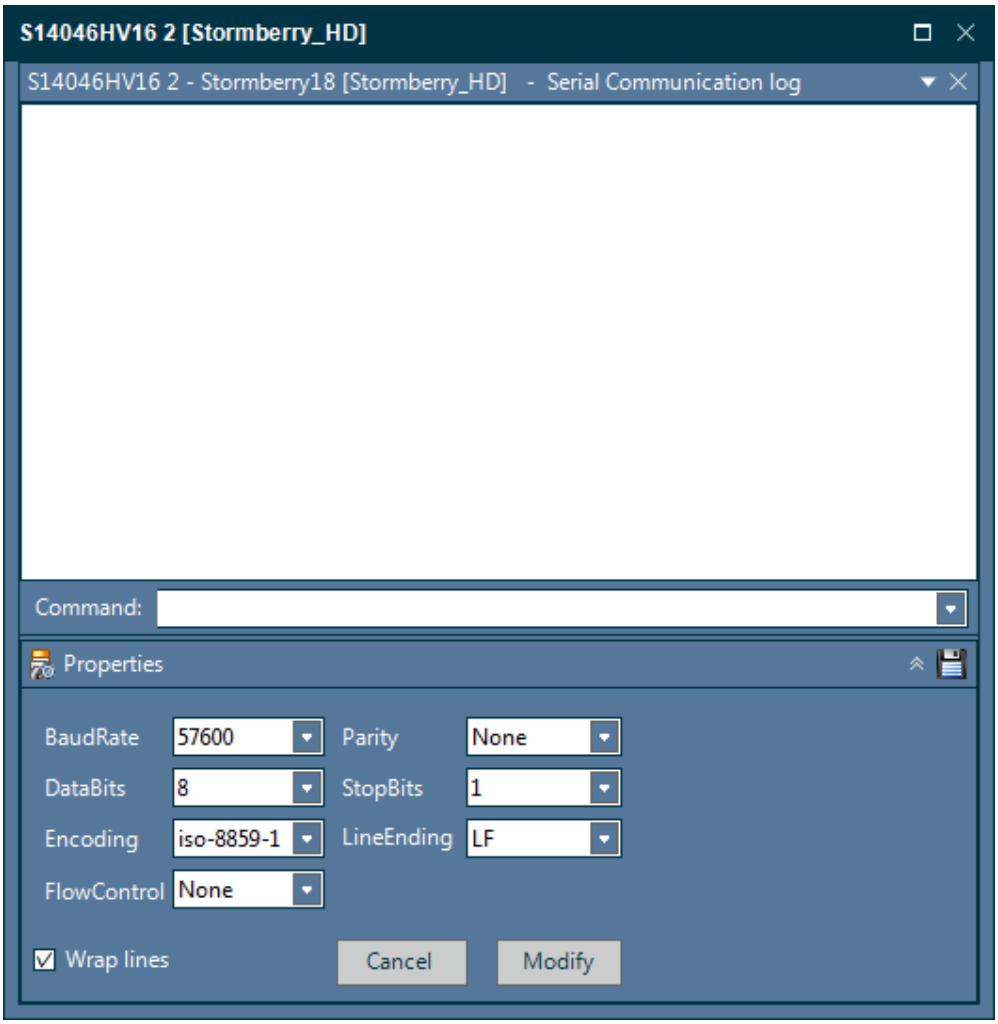
After clicking the  a panel opens:

 The save button allows you to save the contents of the serial window to a file. It brings up the normal Windows dialog allowing you to choose where to save the file.

Here serial data from the DUT is displayed. The  brings up a context menu to control the serial parameters of the DUT:

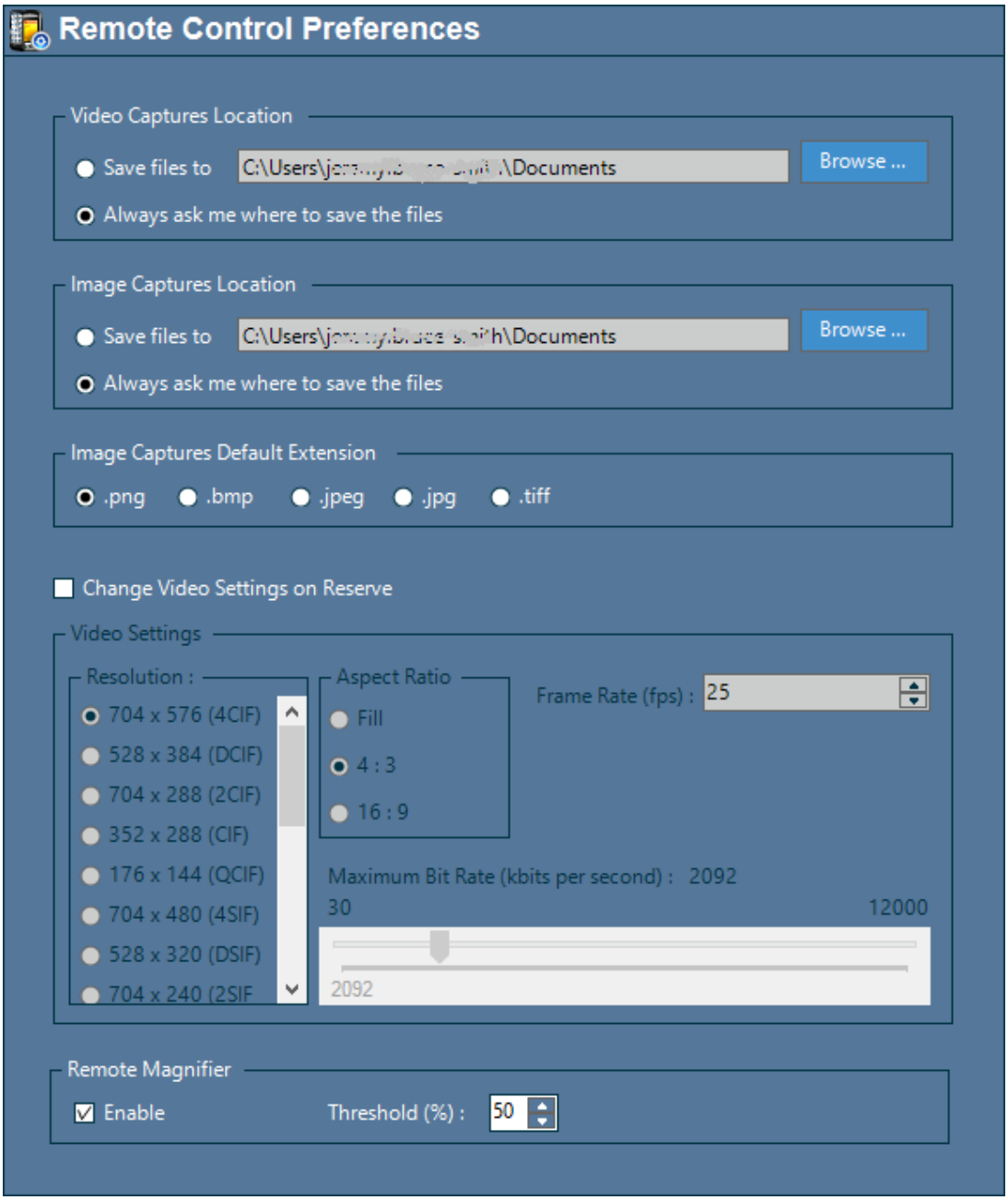


If the DUT is not reserved only the encoding and EOL (End of Line) settings are enabled. These determine how the serial data is presented in the panel and don't change the DUT. The other items all change the serial data from the DUT and so can only be changed by someone who has reserved the DUT. The Open Command Window will bring up a popup window where you can send commands to the DUT. That window also duplicates the serial control settings so that you don't need to keep going back to the main window.



9.4 Remote Control Preferences

The Remote Control preferences control the default behavior of the applet and are divided into 6 groups:



9.4.1 Video Capture Location

This controls how video files are captured. The mutually exclusive options are:

1. Always save files to a specific location
2. Prompt before saving each file

If you request a prompt each time then starting a video file can take a few seconds as you decide where to put the file. Setting a default location means that if you see something interesting in the video you can start the video with a single click.

Use the browse button to select a folder for saving files.

NOTE: Developer Suite will not create a folder so that if you specify a non existent folder or delete the folder, you will not get any video files saved.

9.4.2 Captured Image Location

This controls where still images are saved. The mutually exclusive options are:

1. Always save image files to a specific location
2. Prompt before saving each image

Use the browse button to select a folder for saving files.

NOTE: Developer Suite will not create a folder so that if you specify a non existent folder or delete the folder, you will not get any image files saved.

9.4.3 Image Captures Default Extension

Use this to select the default extension for image captures. The format follows the extension. .JPEG and .JPG will save the same file, the only difference is whether you prefer a 3 character extension (.JPG) or 4 character.

9.4.4 Video Settings on Reserve

Checking the 'Change Video Settings on Reserve' box will cause the Remote Control to set the video streaming parameters whenever you reserve a DUT. This can be useful if you are on a slow network. You cannot be sure how any DUT has been left by a previous script so it could be at a very high bit rate. Note that the settings do not always make sense for all DUTs. You could set the resolution to 4CIF but that will not work on an HD box. If the settings can not be applied you will see a message box. The options are:

9.4.4.1 Resolution

Choose one of the fixed resolutions. The named values of the form xCIF are for PAL, those names xSIF are for NTSC. All other resolutions are only applicable for the HD servers. By default, the HD server will use the negotiated raw HD resolution (usually 1920 x 1080.)

9.4.4.2 Aspect Ratio

This controls how the video is shown. Fill will use the available screen space and this will usually distort the video. HD streams will by default be shown as 16:9 (the normal aspect ratio for HD).

9.4.4.3 Frame Rate

This is the frame rate in frames per second. For PAL systems, the maximum is 25, for NTSC the maximum is 30. For HD systems, the maximum is 60.

9.4.4.4 Maximum Bit Rate

This is the maximum number of bits per second that will be sent. If the picture is static or slow moving, then far fewer bits per second will actually be sent. The slider controls the values in kbits/sec. **NOTE**: The accuracy of the maximum bit rate is approximately 10% so a setting of 1000 kbits per second could result in an actual bit rate 10% higher or lower as the performance of the

video encoder is subject to a variety of optimizations. The maximum bit rate for SD systems is 4000 kbits/sec and for HD servers 12000 kbits/sec

9.4.5 Remote Magnifier

This configures the magnifier that appears over the remote control picture. When the remote control is resized to be below the specified threshold a magnified version of the image is shown as you hover over the control allowing easy use of small buttons. You can enable or disable this feature. When enabled, it applies to the remote control everywhere in StormTest Developer Suite user interface.

9.5 Remote Control

The Developer Suite Remote Control applet combines viewing and controlling DUTs into one applet. You may view the video and serial output of any number of DUTs in any server in the facility.

If the DUT is not reserved, you may reserve it and then use an on screen graphical remote control to send IR commands to the DUT. You can also send serial commands to the DUT and power cycle the reserved DUT.

The Remote Control applet has a single view, along with a preferences view and a help view.

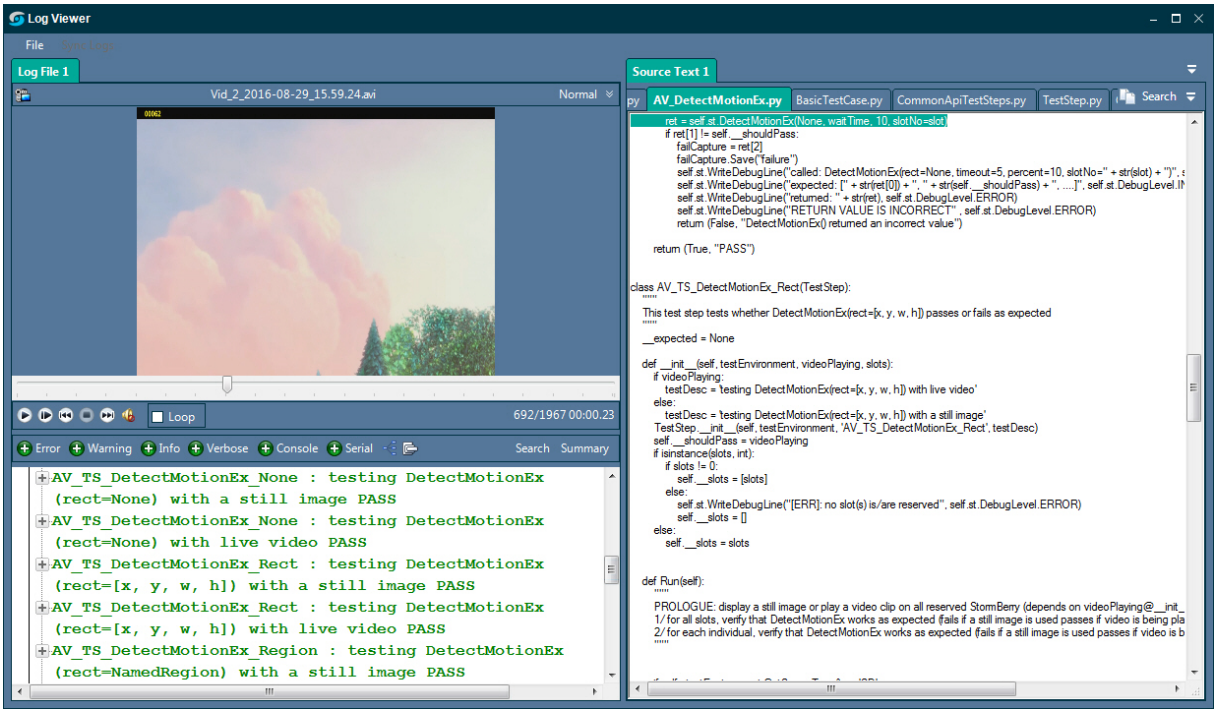
See also:

- Video Mosaic
- Remote Control Panel
- DUT Browser
- Preferences

10 Video Log Player

10.1 Log Viewer Main View

The Video Log Player is divided into 3 main panels. However, if you open two log files or a log file which reserved many DUTs then the number of panels and windows starts to increase beyond 3. If you open a log file produced by a Python script then the standard layout is:



10.1.1 Video View

The top left panel is the video that was recorded. If no video was recorded during a script then this will be empty. It has controls underneath it to play the video.

The Loop check box will play the video or log file in a loop continuously.

The slider lets you jump to a place in the video recording quickly. The other controls let you play/pause, advance by a single frame, jump to start, stop and jump to the end of the video. You can also mute and unmute the audio.

10.1.2 HTML Log View

The lower left panel, beneath the video, is the log file in an html viewer. This is the log file produced by the test.

10.1.3 Source Code View

To the right is the source code that was used to produce the test. It will be Python for python based tests and the Test Creator diagram for test creator graphical tests.

10.1.4 Synchronization

The 3 views are synchronized to each other. Clicking in the text view, moves the video to that point and also highlights the source line or block which produced it. Likewise on each view.

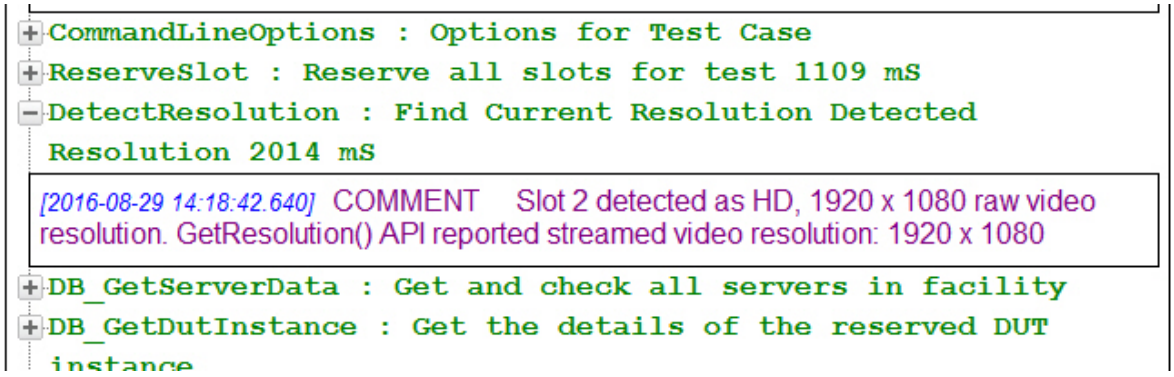
It should be noted that the accuracy is approximately 1 second of real time so occasionally the video will seem to be up to 1 second away from where you would expect. The log file to source code is perfectly synchronized, however.

10.1.5 Opening Files

The video log player is launched from Test Manager by dragging a log file or from the results pages. It can playback files stored on your PC and also files stored on the remote server that were scheduled by the scheduler. When playing back remote files, the video is streamed over the network so you need a good network connection. The initial opening can take several seconds while Video Log Player buffers the video.

10.2 HTML Log Browser

The HTML log browser shows you the log file, color coded for ease of use:



The buttons at the top allow you to filter the information. They use JavaScript and CSS style sheets so can be a little slow on a large log file.

The buttons after `Serial` allow you to open () or close up () all the test steps (or log regions) in the file. They do nothing if you have not used test steps or log regions in the Python code.

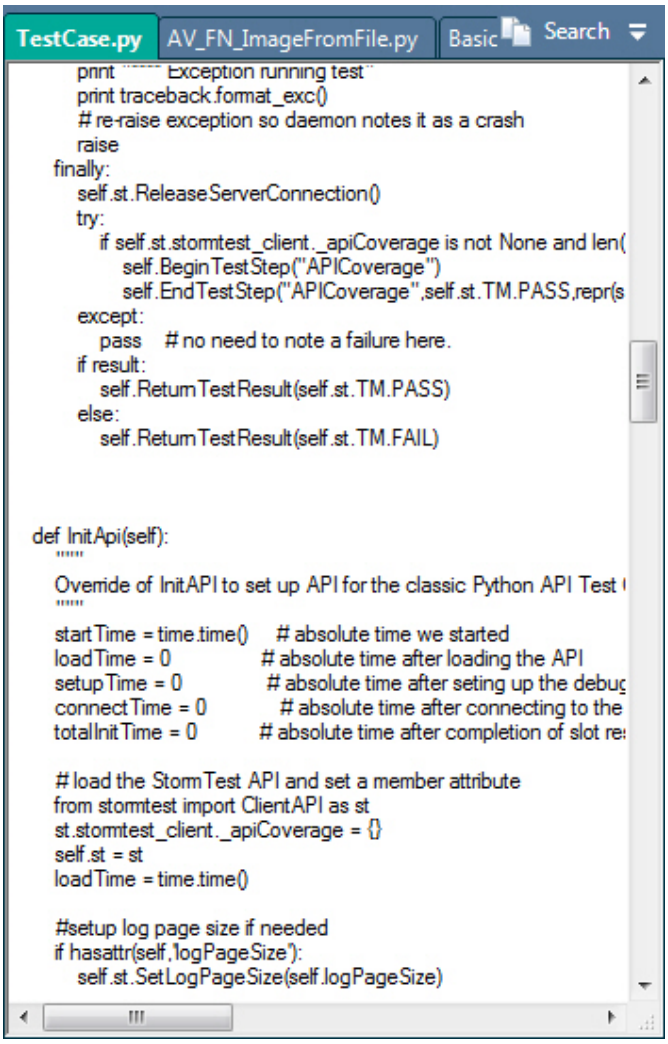
The summary button appears if you have used test steps in your code. It brings you to a page which shows you all the steps (and only the steps) in your file. You can switch between the main log view and the summary view. In this release, if your log file is large, it will be split into pages and the toolbar will change to:

The right context menu is the same as used for Internet Explorer - the view is really Internet Explorer without the usual control bars and menus.

Clicking on a line in the file will scroll the video and source code to the same location.

10.3 Python Source Viewer

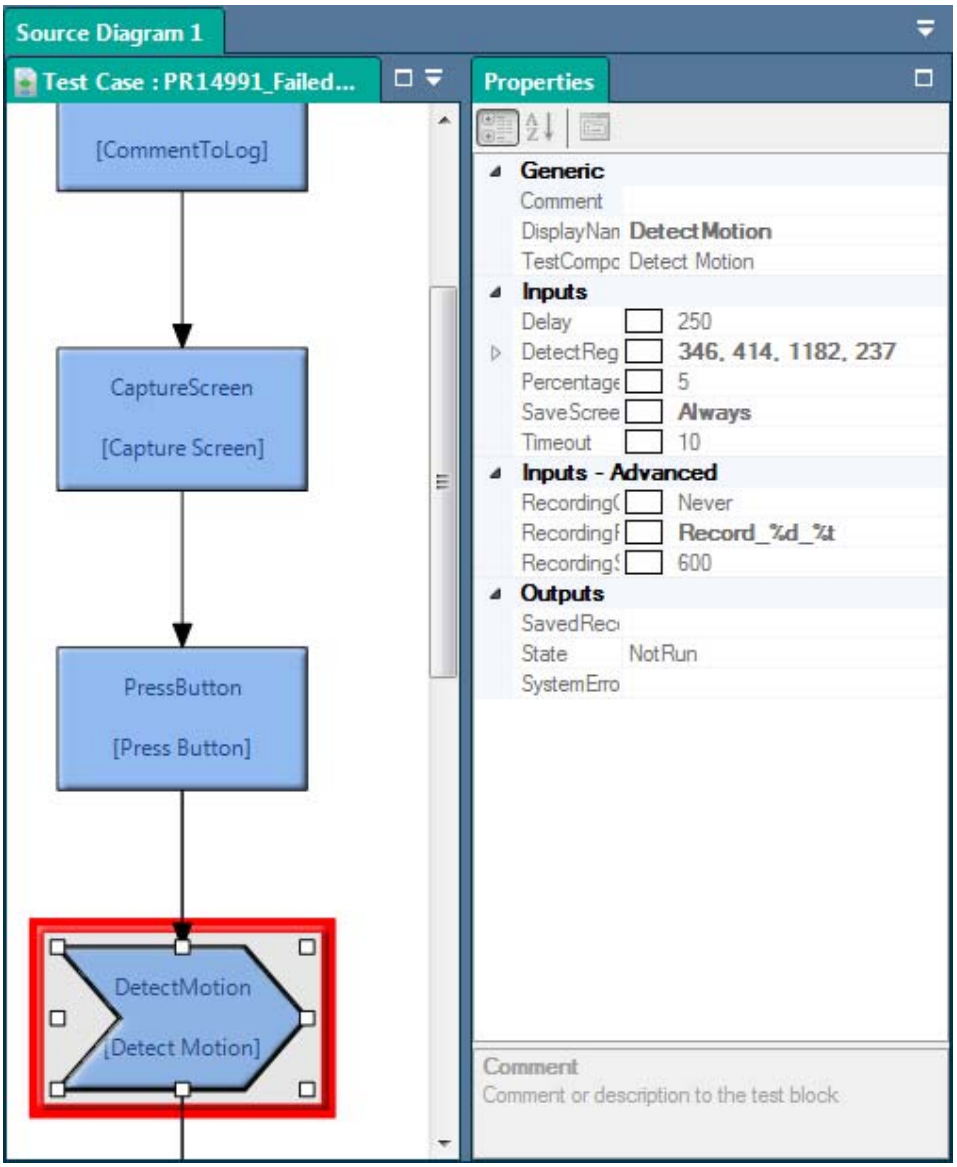
The python source viewer is shown if the log file was generated from a Python script:



If the python script imported other user defined modules that called StormTest API functions then they will also be shown as separate tabs. The viewer does not show files from the StormTest API itself nor files which do not use the StormTest API. You can jump to the StormTest API function calls only - and then the log file and video will be synchronised.

10.4 Test Diagram View

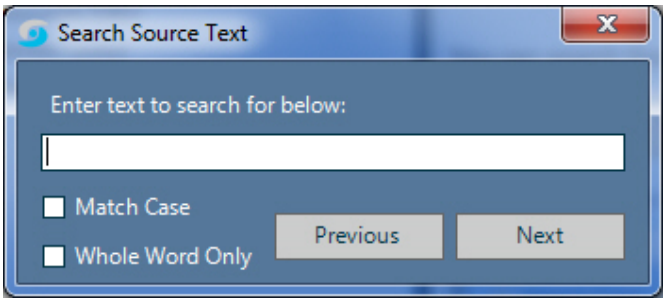
The Test Diagram viewer is shown if the log file was generated from a Test Creator test:



The left hand panel is the original test and the right shows the properties of the highlighted object. These are read only as they represent the completed test. You can click on blocks in the diagram and the log file and video will be synchronised to them.

10.5 Search Source Text

You can search the Python source code using the simple dialog:



If you wish to cancel the search, just close the window.