Natural Screen Tester

Designing and Developing a Test Case

Innovation Release

Version 1.2

April 2018

This document applies to Natural Screen Tester Version 1.2 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: NSR-DESDEVAPP-12-20180416

Table of Contents

1 Managing Test Projects

A test project corresponds to the Natural application you want to test. It consists of one or more test cases.

See Test Project Configuration Parameters in the *Natural Screen Tester Reference Guide* for further details regarding all the test project parameters.

Creating a Test Project

In this task, you will create a new Natural Screen Tester test project. Every Natural Screen Tester test project is connected to a host and has a repository containing all the Natural Screen Tester entities. The steps detailed in this task include the basic steps required to create a test project. Advanced configuration is explained under *Test Project Configuration Parameters* in the *Natural Screen Tester Reference Guide*.

To create a Natural Screen Tester test project

Before creating a new test project, ensure that Natural Screen Tester server is available and running, and start the Designer. Login to the relevant server or add an additional server as required. See also *Natural Screen Tester Designer (Software AG Designer)* in the *Getting Started* documentation.

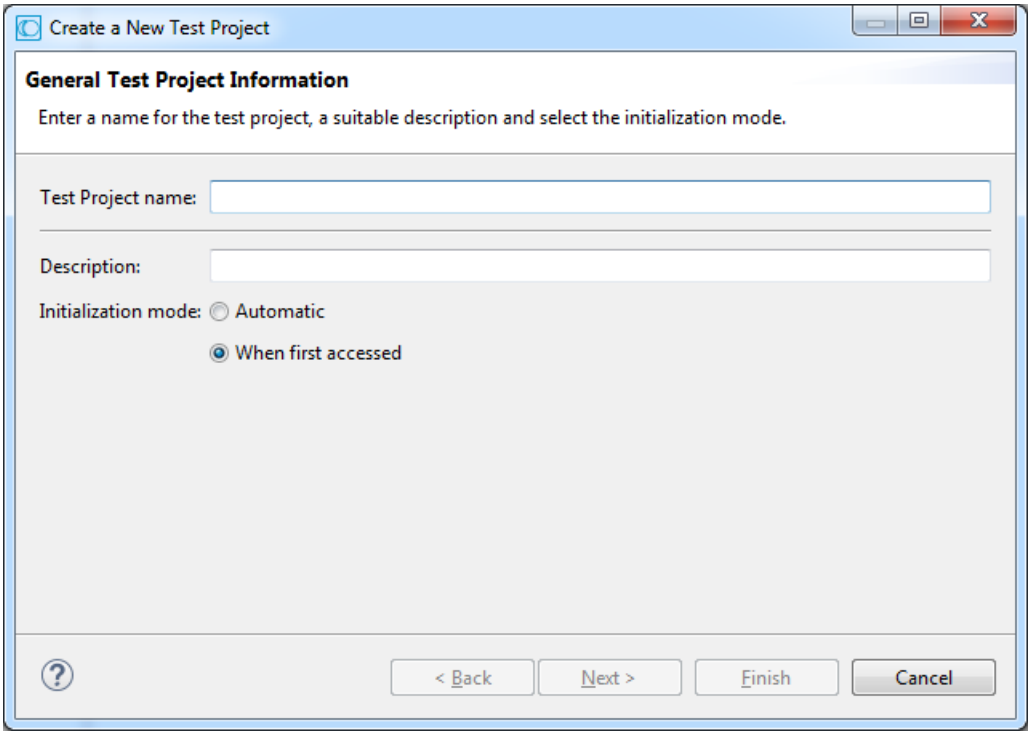
1 In the Natural Screen Tester Explorer, right-click on the Natural Screen Tester Server and choose **New test project...** .

Or:

Choose the menu item **File > New > Other...**. The *New Project wizard window* is displayed.

Expand the **Software AG** node and choose **test project** then click **Next** to display the New Natural Screen Tester test project wizard:

Create a New test project wizard is displayed.



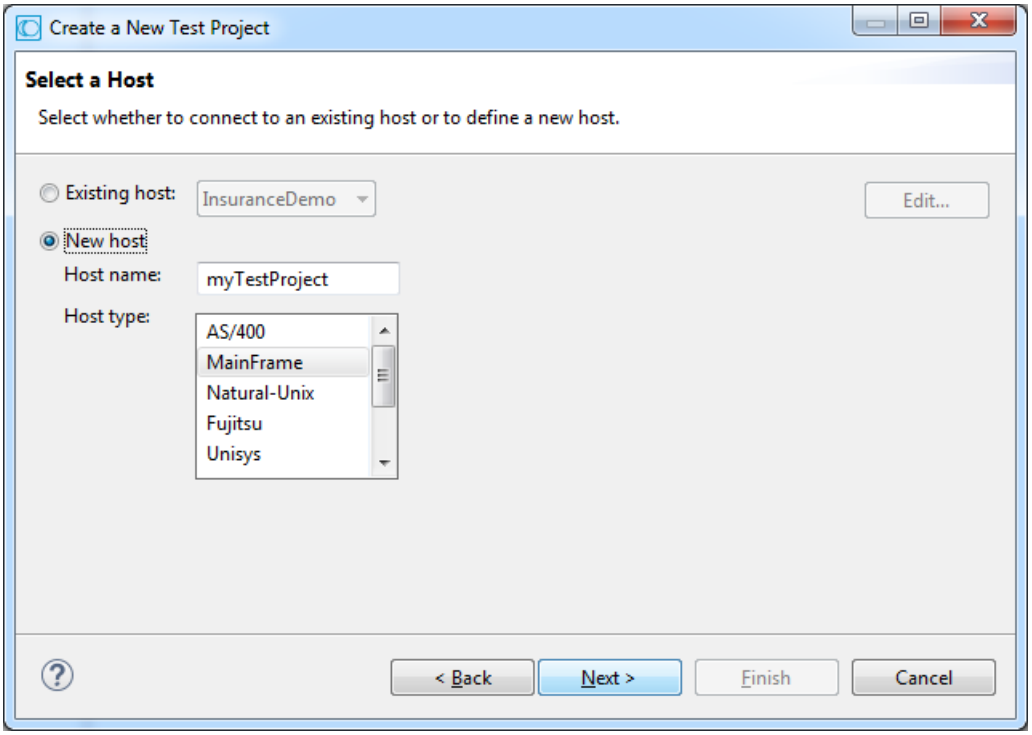
2 Enter a name for the test project and a suitable description.

3 Select the **Initialization mode**:

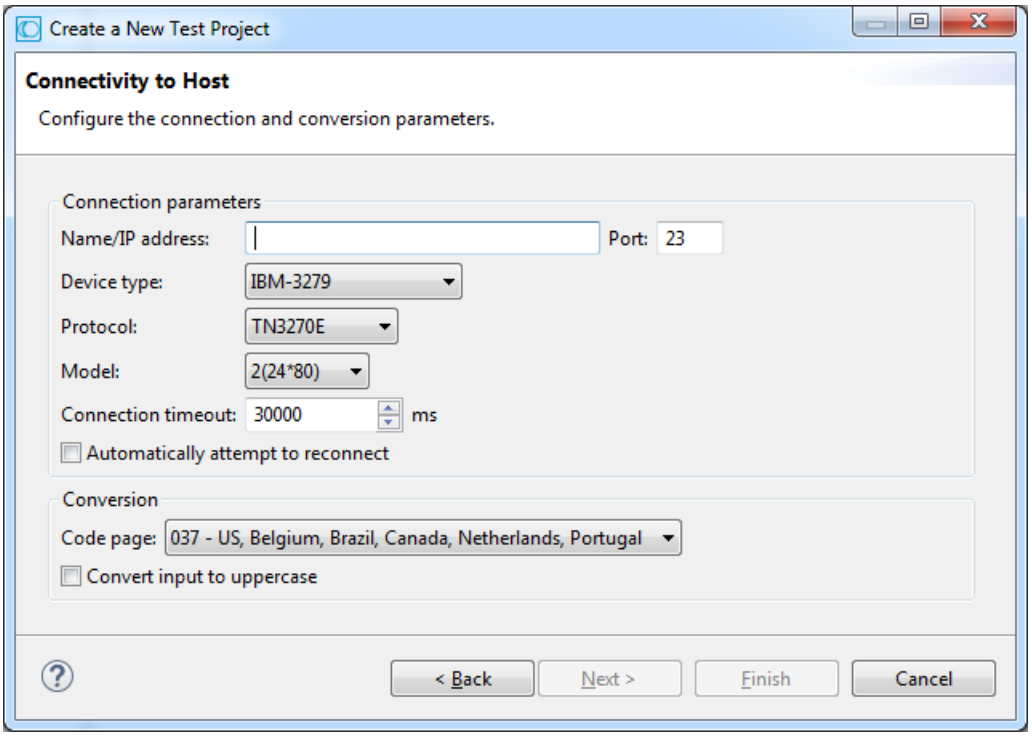
Automatic
Automatically loaded when the server is started.

When first accessed
Loaded when first accessed, in other words, when the code that initializes startup is first called (default).

4 Click **Next**. The **Select Host** screen is displayed. Choose whether to use an existing host or create a new one.



5 If you choose to define a new host, enter a name for the host and select the relevant type of host. Click **Next**. The host connection and conversion parameters are displayed.



It is mandatory to enter the host's name/IP address (IPv4 and IPv6 address formats are supported). Configure the other parameters as required. For details see *Host Configuration: Connectivity* in the *Natural Screen Tester Reference Guide*.

6 Click **Finish**. The test project is displayed in the Natural Screen Tester Explorer. It is possible to further configure the test project properties by right-clicking on the test project name and selecting **Properties**. See also Test Project Configuration Parameters in the *Natural Screen Tester Reference Guide*.

Editing a Test Project's Configuration (Advanced Configuration)

The *Test Project Properties* dialog box contains additional parameters that can be configured such as the language, process tracing, screen content definitions etc. For details of each node see Test Project Configuration Parameters. To edit a test project's configuration, right-click on the test project's node (in the Natural Screen Tester Explorer) and choose **Properties**.

Note: Notice that in the **Host name** field (in the Host tab), only hosts of the same type as the configured host are listed.

Deleting a Test Project

To delete a test project

- Right-click on the test project's node and choose **Delete**. In addition to deleting the test project it is also possible to determine whether when the host used in this test project is not used in any other test project, you would like to delete the host and also whether you would like to delete the entire test project from the file system.

Reloading a Test Project

Reload a test project when there is new data that the test project requires, such as when repository definitions change or when the database file is replaced. Reloading the test project will reload the repository, close all connection pools, stop running connection pools and if the database is synchronized at the end of the reload process will automatically restart the connection pools.

To reload a test project

- Right-click on the test project's node and choose **Reload Test Project**.

Importing a Test Project's Configuration or Entities

It is possible to import:

- a complete test project - including the configuration, the test project entities and a trace file - contained in a gxar file
- a test project's entities in a gxz file

It is not possible to import entities exported from a higher server version than the current version that you are using.

Important: As the import process may consume a lot of memory, it is recommended to restart Natural Screen Tester server after completing the process.

This section covers the following topics:

Importing a Complete Natural Screen Tester Test Project

When importing a complete Natural Screen Tester test project, you will require a gxar (Natural Screen Tester test project archive) file. This file includes the test project configuration, Natural Screen Tester entities (as a read only gxz file) and a trace file. The gxar file can be imported into Natural Screen Tester in two different methods:

"Hot Deploy"

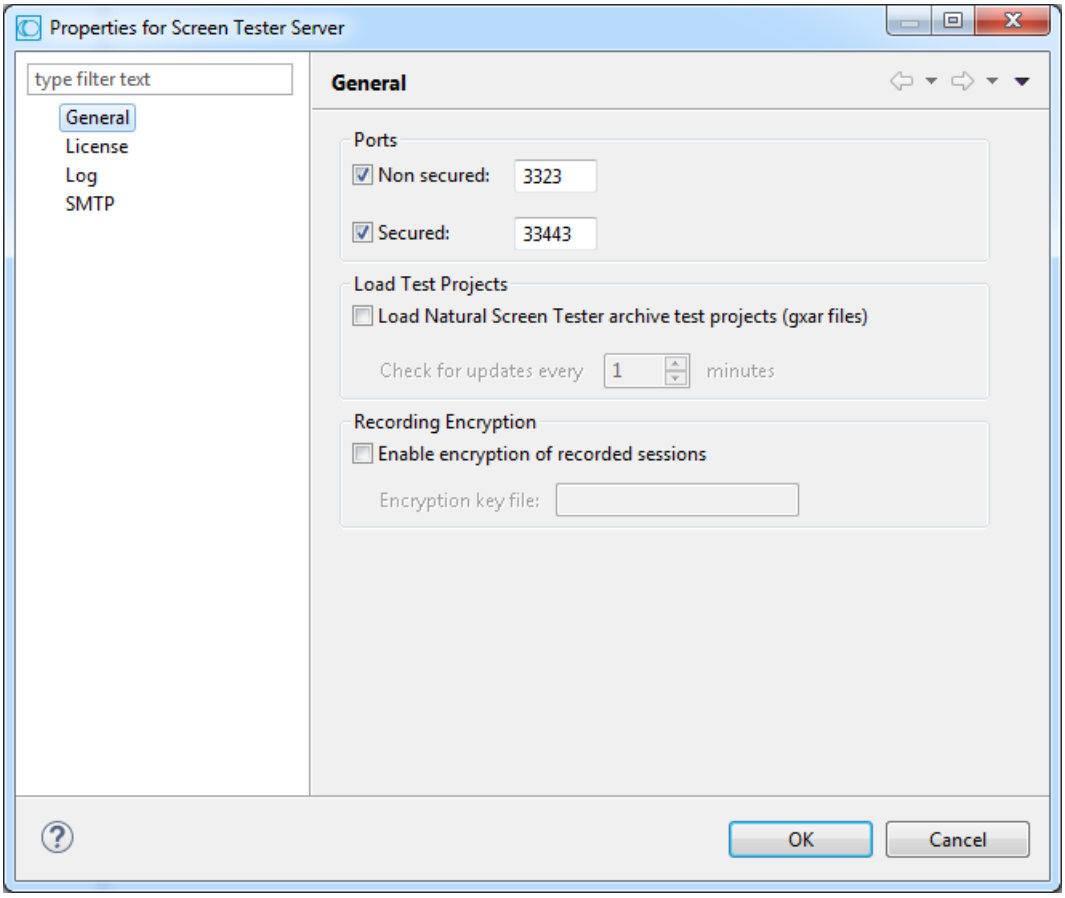
Using the "Hot Deploy" method, the gxar file is simply located and placed in the *host-applications* directory as the file to be imported.

To import a test project in the "Hot Deploy" method

1 Open your Windows Explorer.

2 Locate the relevant gxar file and place it in the *host-applications* directory of your Natural Screen Tester installation.

3 In Natural Screen Tester Designer, right-click on the server and choose **Properties.**



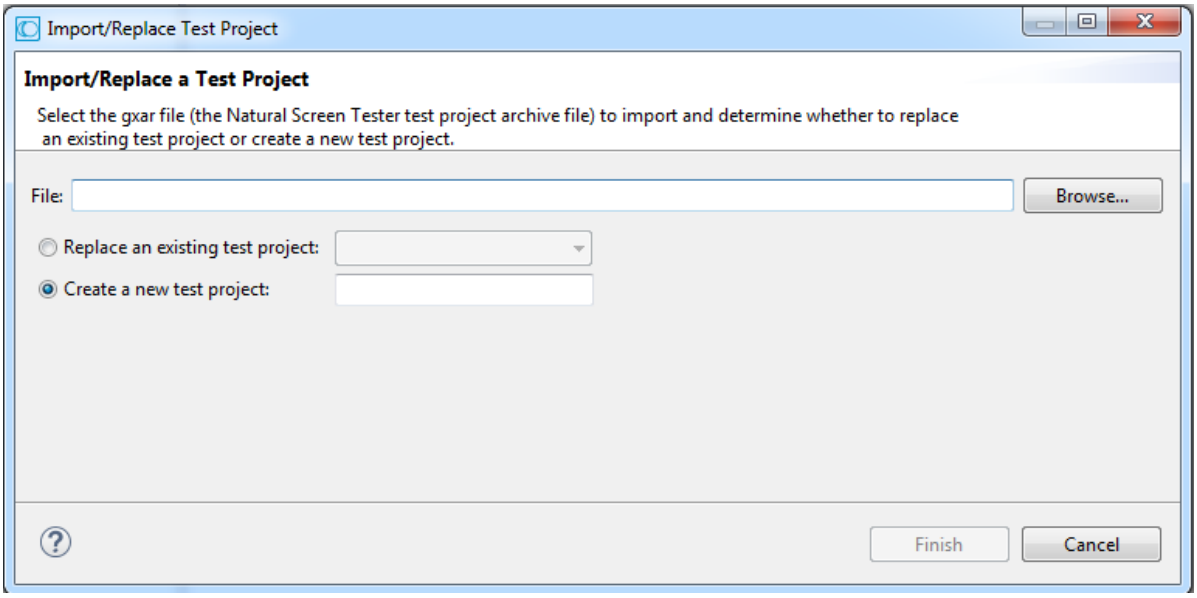
4 In the **General** tab, determine to load Natural Screen Tester archive test projects. Set the interval to look for updates. This determines how often the Natural Screen Tester server should check to see if the gxar file to use has changed. Once a different gxar file is detected, it will be used.

It is also possible to use the gxar file by reloading the server (right-click on the server and choose **Reload**).

Import Wizard

To import a test project

1 Right-click on the test project's node and choose **Import/Replace test project**. The **Import/Replace Test Project** wizard is displayed.



2 Enter a file name, or browse and select the file to import.

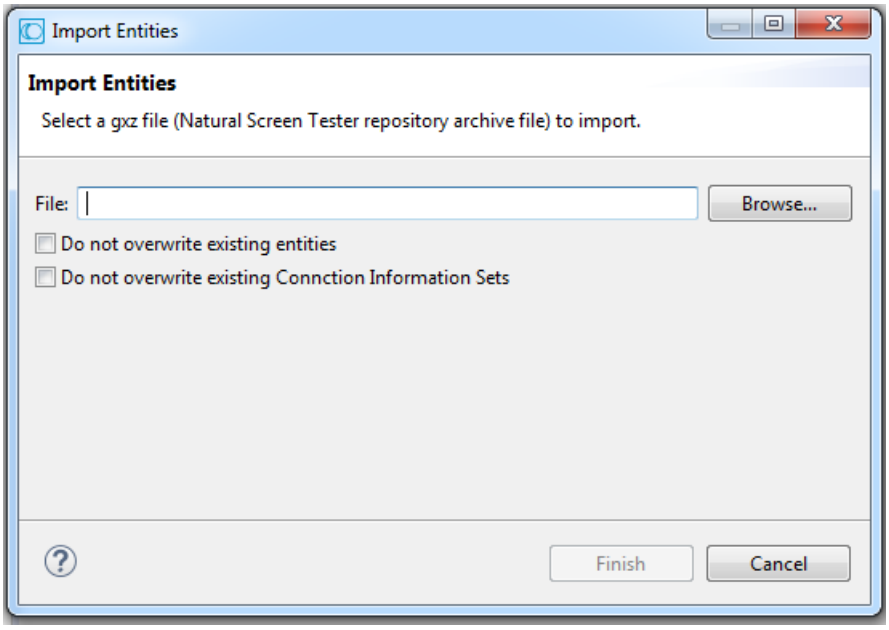
3 Select whether to replace an existing test project (selected by default), or to create a new test project. Enter a name accordingly.

4 Click **Finish**. Wait a few minutes while the test project is imported.

Importing a Test Project's Entities

To import entities

1 Right-click on the test project's Root node and choose **Import Entities**. The **Import Entities** wizard is displayed.



2 Enter a file name, or browse and select the file to import.

Check **Do not overwrite existing entities** to determine not to overwrite an existing entity with an imported entity of the same name.

Check **Do not overwrite existing Connection Information Sets** to determine not to overwrite an existing connection information set with a set of the same name.

Notes:

1. If the first option is checked, the second option is checked automatically and cannot be unchecked.
2. The Session Data entity will be merged with the existing Session Data entity. When there is a conflict between the imported to the existing Session Data entity, your selection in this checkbox will determine how the Session Data entity will be.
3. When importing an entity to a test project that has the same entity name, the existing entity will be overwritten by the new one. The timestamps of the new entity (for example creation and modification date) also overwrite the values of the original entity. These changes take effect immediately, not after reloading the test project.

3 Click **Finish**. The entities are imported into the test project.

Note: Clicking Cancel on the Progress Bar dialog box will cause the process to be stopped and the entities will not be imported.

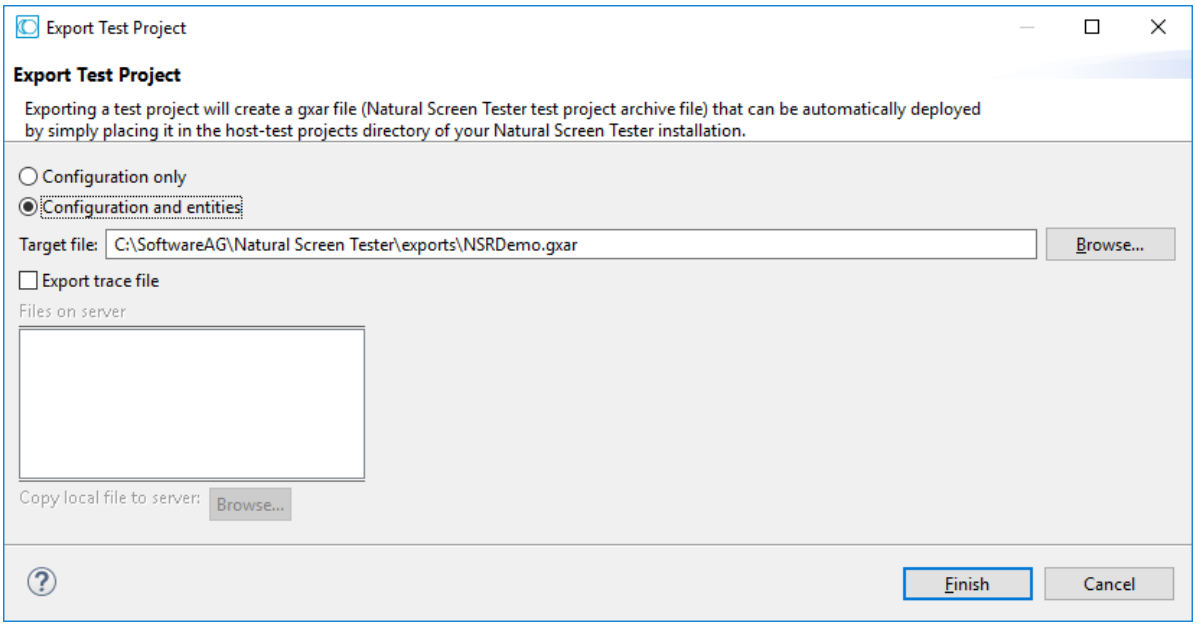
Exporting a Test Project's Configuration or Entities

The Export Wizard allows you to export a test project configuration and its repository residing on various database systems into a standard zip file. This wizard guides you through the steps of exporting data.

Note: When exporting entities, the Session Data entity will always be exported.

To export a test project configuration with/without its entities

1 Right-click on the test project's node and choose **Export test project**. The Export test project wizard is displayed.



2 Select whether to export just the test project configuration or the test project configuration together with the entities.

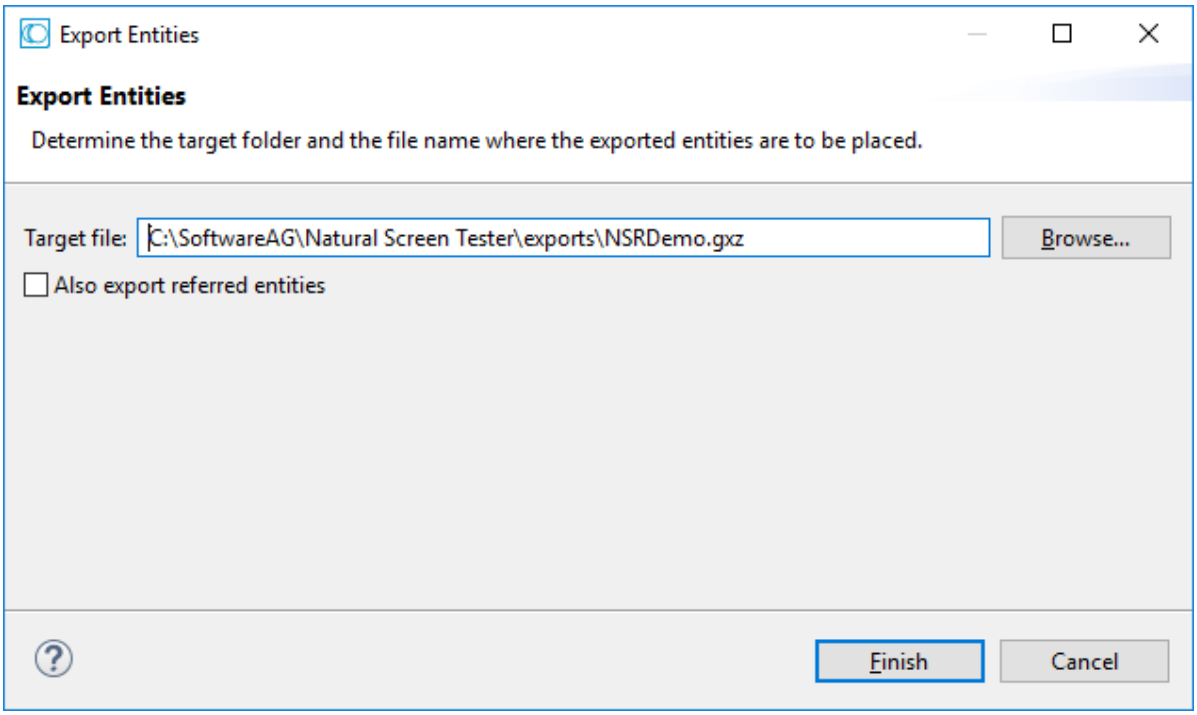
3 Enter a folder name, or browse and select the target folder.

4 Check the **Export trace file** check box to export a trace file. A list of replay (gct) files that are on the server are displayed. Select a file or click **Browse...** to select a local file to copy to the server.

5 Click **Finish**. The test project will be exported to the defined folder.

To export entities only

1 Right-click on the test project's Repository node and choose **Export Entities**. The Export Entities wizard is displayed.



2 Browse and select the target folder and file name where the exported entities are to be placed.

3 Click **Finish**. The export process will commence and the entities will be exported to the selected folder.

Note: The Session Data entity will always be exported, regardless of the entities selected to export.

Recording Trace Files

The Trace File feature enables recording a file, which traces the connection communication between the Natural Screen Tester server and the host, for each connection. It is possible to define whether a single trace file will be created, replacing the previously saved file or whether the data will be saved to a new file for every new user session. Identifying the separately saved files is possible by inserting identifying parameters in the file name (the session ID, creation time and/or connection ID). It is highly recommended to create a separate file for each session/connection or creation time.

Note that trace files can be created from within the session definition overriding the test project definition. This is recommended as it prevents conflict with other existing sessions.

The trace files can be also defined to be created in folders per Day Month and Year. This is recommended as it prevents conflict with other existing sessions.

The trace files can be compressed to save space on the disc. This is particularly suitable when tracing a large number of files on a regular basis.

When required, you can select to encrypt the trace files.

To record a trace connection file

1 In the Natural Screen Tester Explorer, right-click on the relevant test project and choose **Properties**.

2 Click on the **Host>Recording** node. See *Host Parameters: Recording*configParms_host_recording

3 Check **Record terminal sessions (trace files)**.

4 Check the **Compress (create files in zip format)** check box to compress the file. Compressed files will have the suffix .zip.

5 Check the **Encrypt (using server private key)** check box to encypt the file. In order to encrypt files, you must first define the encryption key (In the Server properties, General tab). Encrypted files will have the suffix .gctx. A file which is both compressed and encrypted will have the suffix .gctx.zip.

6 Choose **Suppress hidden fields** to conceal passwords and hidden fields.

Note: The "Suppress hidden fields" option is not supported when recording using Terminal Emulation Proxy.

7 Provide a name for the file.

8 Check the relevant check box to determine that a new file will be created for each session/creation time or connection ID. It is possible to add the following parameters to the file name instead of using the check boxes: %u will insert the session ID. %t will insert the creation time stamp of the connection. %c will insert the connection ID.

9 Browse and select the location of the folder where the files will be stored. Determine whether sub folders will be created for each year/month/day.

10 Click **OK** to save your changes.

See also Test Project Configuration Parameters.

Working with Offline Sessions

In order to develop, debug and reconstruct specific scenarios, it is possible to replay (GCT) files that have traced the emulation protocol for each user. See **Recording Trace Files**. In order to replay such files, the test project configuration definitions must indicate that instead of working online with the host, Natural Screen Tester must, when connecting, access and replay the specific file. Note that replay file can be defined from within the session definition overriding the test project definition. This is recommended as it does not conflict with activities of other users.

Note: Offline sessions support replaying encrypted and/or compressed files. Refer to **Recording Trace Files** for additional details

To define working with offline replay files

1 In the Natural Screen Tester Explorer, right-click on the relevant test project and choose **Properties**.

2 Expand the **Host** node and select the **Offline** node.

3 Check the **Work offline** check box.

4 Either enter or browse to select the file name including the full path.

5 Natural Screen Tester enables a session replayed by a GCT to simulate the host's communication delay or to predefine a time delay to wait before showing the information (this is because generally, the test project is faster when replayed). This is determined in the **Simulate host delay** field. Available values: No delay, Simulate host, 500- 10000 ms (by default No delay is selected).

6 Click **OK** to save your changes.

See also Test Project Configuration Parameters.

Changing the Initialization Mode of a Test Project

To change the initialization mode of a test project

1 In the Natural Screen Tester Explorer, right-click on the relevant test project and choose **Properties**.

2 Ensure that the **General** node is selected.

3 In the **Initialization mode** field, select the relevant initialization mode:

When first accessed
Loaded when the code that initializes startup is called.

Automatic
Loaded when the server is started.

4 Click **OK** to save your changes.

See also Test Project Configuration Parameters.

Setting Session Timeout for Idle Users

To set the timeout for idle users

1 In the Natural Screen Tester Explorer, right-click on the relevant test project and choose **Properties**.

2 Select the **Host** node.

3 Select whether the session timeout will be unlimited or disconnected after a specific number of seconds.

4 Click **OK** to save your changes.

See also Test Project Configuration Parameters.

Handling Flickering of Host Screens

It is necessary to use the Flickering of Host Sessions feature when one of the following happens:

- In the browser, a blank screen is displayed when navigating between two host screens.

  Note: Refer to *Blank Screen Timeout* in *General Host Parameters* for additional information on handling blank screens.

- In the browser you are required to submit the Enter key (or any other key) twice in order to navigate to the next host screen.

The initial need for Flicker arises when specific host screens are received 'split' between several buffers of data. Thus Natural Screen Tester Server needs to be informed to wait an additional amount of time for the complete screen to arrive. This additional amount of time is defined (in milliseconds) in the Flicker parameter in the Host node of the test project Properties dialog box. The flicker setting applies to the entire Natural Screen Tester test project, meaning that if the flicker is set to 500ms, after each host transaction the flicker time will be added to the communication time. In other words, the entire test project will be 'slowed down' by the flicker time. Therefore this value should only be set for the entire test project according to the following guidelines:

As a rule of thumb, there is no reason to use the flicker setting in the test project configuration for block mode hosts. Specific 'problematic' screens should be handled using wait conditions, both in navigation paths and in Web test projects. For less specific cases (when a wait condition for a specific screen cannot be used), it is possible to set the flicker setting only for certain actions (using either the path dialog or the Base Object API in Web test projects).

To define the flicker time

1 Click the relevant test project in the Natural Screen Tester Explorer.

2 Right-click a test project and choose Open. The test project dialog box is displayed.

3 In the Host tab enter a value in the Flicker field. Possible values are from 0 to 10000 ms.

4 Click Apply to save all changes, without closing the dialog box. Click OK to save changes and close the dialog box. You will be required to confirm the change.

See also Test Project Configuration Parameters.

Defining Natural Screen Tester RPC Parameters

The RPC connections pool is used for holding pre-connected RPC connections. When an RPC connection is required to execute a program, a connection from the pool is used, saving the connection time and executing the program immediately. This feature is available for AS/400 hosts only. This feature is available in SOA test projects only.

To set the RPC pool connections settings

1 Open the *test project Properties* dialog box and click on the **Host>RPC** node.

2 Choose **Use Connections Pool**.

3 Configure the RPC connection parameters. Refer to Test Configuration Host Parameters: RPC for further details regarding the fields in this dialog box.

4 When required, configure the username and password needed to connect to the AS/400 programs. See also Host Configuration Parameters: RPC.

Mapping Keyboard Keys

It is possible to map specific keyboard keys to the Keyboard Mapping tab in the test project Properties dialog box.

To define Keyboard Mappings

1 Right-click on the relevant test project and choose **Properties**.

2 Focus on the Keyboard Mapping tab.

3 To map one of the existing mappings, locate it in the list of keyboard mappings and ensure that it is marked as Enabled. If it is not enabled, click on the Enable Key hyperlink.

4 To add a new keyboard mapping, click on the Add Keyboard Mapping hyperlink.

5 Enter the key combination in the Keyboard field.

6 Either select the host key from the standard list of host keys or enter a host key in the Host key field. Ensure you place square brackets around the key.

Note: When using the keyboard keys within the web test project, the CTRL+N and CTRL+K keys are blocked by default as they cause multiple browser windows to use the same session (this can be manually set in the *config/gx_keyboardMappings.xml* file).

Key combinations defined here may conflict with Eclipse key bindings. When such a conflict occurs, the key combination defined here will be ignored during runtime. To see the list of Eclipse key bindings, open the **Windows>Preferences** menu, expand the **General** tab and choose **Keys**. A key binding which appears in this list, and whose `When` field is set to "In Windows" or "In Dialogs and Windows" will cause a conflict.

When such a conflict occurs, either redefine the keyboard mapping in Natural Screen Tester, or change the Eclipse key binding. The Eclipse key binding can be changed either by choosing a different option in the `When` field (the key binding will be disabled and it will not be possible to enable it), or by editing the *plugin.xml* file of the Natural Screen Tester plug-in (add the new key binding in the org.eclipse.ui.bindings extension point with Natural Screen Tester scheme ID and empty commandId. This way, whenever the Natural Screen Tester key binding scheme is activated, the problematic key bindings will be disabled, but in other schemes the key bindings will be enabled.)

Supporting RTL Languages

The **test project Configuration>Language** node enables you to define the language used in the test project as well as direction settings, relevant mainly for right-to-left languages.

To configure RTL language settings

1 In the Natural Screen Tester Explorer, right-click on the relevant test project and choose **Properties**.

2 Select the **Language** node.

3 Select the test project's language.

Note: The following settings are relevant for right-to-left languages. The option you select is the default setting for all the screens, but can be changed for a specific screen in the relevant screen, in the Screen Direction tab.

4 Set the **Screen direction**. Relevant for mainframe hosts only. The screen direction of right-to-left languages differs according to the original host settings, and when incorrectly set, can cause the screen to be illegible. In order to correct this, define the suitable screen direction.

5 Set the **Typing direction**. Right-to-left languages may display typed-in text in the Display Session View and HTML fields, aligned to the left of the field. In order to display the text aligned to the right, select the right-to-left option.

6 Set the **Tab direction**. When pressing the TAB button the cursor moves to the next consecutive field. The direction the cursor moves (moving to the next field to the left or moving to the next field to the right) must be correctly defined in order to preserve the screen logic.

Recording Steps while Navigating

The test project Map view enables viewing a navigation map between screens. Once the navigation option is selected, in the test project Properties, Navigation tab, the test project map will record the navigation steps between the screens. See also Navigation Parameters and Test Project Map.

Repository

The Natural Screen Tester repository is an internal database used to hold Natural Screen Tester entities (metadata) such as the Screens, Paths, Connection Pools, Programs and Procedures.

As a Natural Screen Tester user, you may like to divide test projects into folders within your repository, according to the specified interests and needs, such as development teams, sub-test projects etc. The same folder contains different types of entities and sub-folders varying according to each test project. This allows you greater flexibility organizing your entities.

The repository can be read only. It is recommended to use a read-only repository for production.

Selecting a Read-only Repository

To set up a read-only repository

1 Open the test project.

2 Right-click on the repository node.

3 Choose **Lock Repository (read only)**

Defining Host Windows

The Windows definitions are used to correctly identify screens and to open host windows as separate pop-up windows. In the test project Properties it is possible to define the Host Windows per the test project.

When adding a window you are required to select the type of modal windows the host sends. There are two types: Reversed Video or Frame. For test projects that have more than one level of windows (a window within a window), where each level is defined with a different Window Type, be sure to define the windows in the correct order.

Note: When the host is a Natural UNIX host, this tab is disabled, as the windows' definitions are included in the Natural-UNIX protocol and do not require being defined via Natural Screen Tester.

To define windows

1 Right-click on the relevant test project and choose Properties.

2 Add a Frame or Reversed Video

3 Configure the parameters as detailed in Test Project Configuration Parameters: Windows.

Multiple Developers Working on the same Test Project

Natural Screen Tester test projects are typically developed by more than one user. This can sometimes cause conflicts on the Natural Screen Tester server. Working methodologically and investing time and effort in planning the development and design of the project can help prevent such conflicts. We recommend you divide responsibilities between the developers (such as developers working on specific entity types, or workflows).

Typical conflicting scenarios and outcomes:

- More than one developer editing the Test Project Properties: Natural Screen Tester will save the changes of the first developer who saves the changes.
- More than one developer editing an entity:

  When more than one developer edits the same entity, and one of the developers saves the entity, the other developers receive a message indicating that this entity has been saved by another developer. You are required to determine whether you would like to work on the newly saved entity (and update your editor to reflect the newly saved entity) or to continue working on the outdated editor.

  If you choose to continue working on the outdated editor then when trying to save the entity, you will be informed of the name of the user who made the changes and you will be able to decide whether to either:

  - Overwrite the changes that the other developer has made.
  - Save the entity with a different name. Note that references that pointed to the original entity will not point to this entity and need to be added manually. References that this entity referred to will be maintained.
  - Discard the changes that you made.

2 Defining Hosts

The host containing the application you want to test.

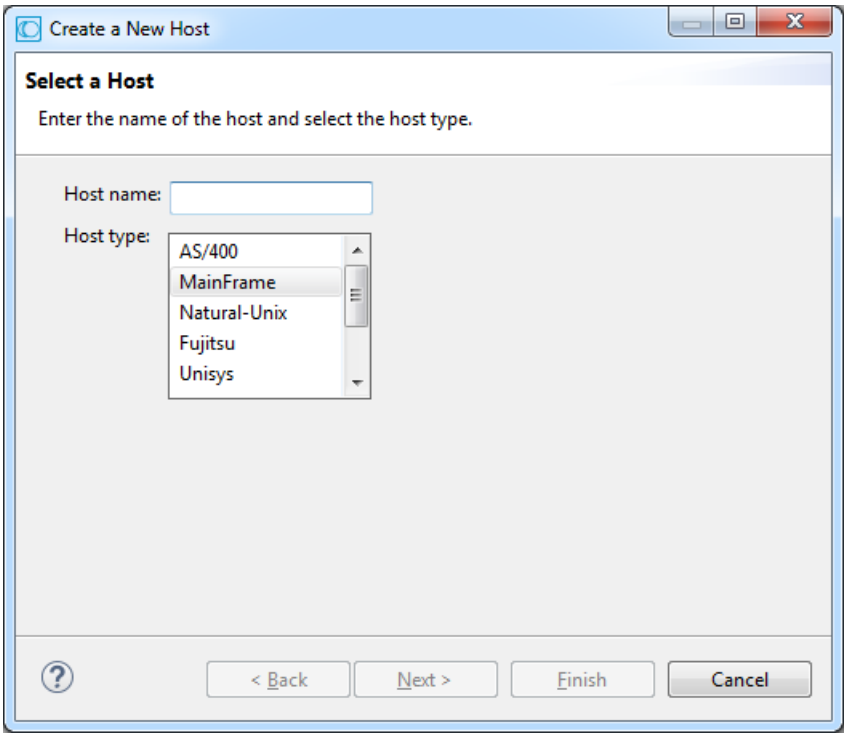
See also Host Configuration Parameters in the *Natural Screen Tester Reference Guide*.

Defining a New Host

A new host may be defined either in the process of creating a new test project (refer to **Creating a Test Project**) or simply as an additional host in the host node.

To create a new host

1 In the Natural Screen Tester Explorer, right-click on the Host node and choose **New Host**. The New Host wizard is displayed.



2 Enter a name for the host and select a host type. Click **Next**.

3 Set the connection and conversion parameters, which may differ according to the host type. Click **Finish**. Refer to Host Configuration Parameters in the *Natural Screen Tester Reference Guide* for further details.

To delete a host, in the Natural Screen Tester Explorer, right-click on the relevant host and choose **Delete Host**. Confirm this action.

To edit the current host configuration, in the Natural Screen Tester Explorer, right-click on the relevant host and choose **Properties**. The Host Properties' are displayed.

Defining an RPC Host (AS/400 Hosts only)

After defining an AS/400 host, open the host's properties (right-click on the host in the Natural Screen Tester Explorer and choose **Properties**), and in the RPC tab configure the parameters.

UNIX Hosts Based on Natural Applications

Standard UNIX protocols lack information required for extensive integration into modern environments. Natural-UNIX, a proprietary protocol of Software AG, provides you with this additional information, and it is therefore recommended to be used with all Natural applications that run on UNIX machines.

Refer to Natural-UNIX Installation for installation details.

To configure a Natural-UNIX host

1 Create and access the Host dialog box as detailed in Adding a Host Definition.

2 Configure the new host: enter a name, IP address (IPv4 and IPv6 address formats are supported), port and service (optional - limited to 24 characters). In the **Device type** field, choose Natural-Unix. Select Natural-APX (or Natural-NSW for Natural versions prior to Natural version 6.2.5 and 6.3.2). Configure the relevant parameters.

There are applications which require entering a user name and password. Click on the **Security** tab. Enter the default **User name** and **Password** (both fields are limited to 24 characters). You can override these settings, either when connecting to a session (in the *Connection Properties* dialog box, **Host user name** and **Host password** fields).

Note: The Test Case Properties > Windows tab will be disabled, as window's definitions are included in the Natural-UNIX protocol and do not require being defined via Natural Screen Tester.

SSH

What is SSH?

Secure Shell (SSH), sometimes known as Secure Socket Shell, is a command interface and protocol for securely getting access to a remote computer. It is widely used by network administrators to control Web and other kinds of servers remotely. SSH commands are encrypted and secure in several ways: both ends of the client/server connection are authenticated using a digital certificate, and passwords are protected by being encrypted. SSH uses RSA public key cryptography for both connection and authentication.

Natural Screen Tester supports the SSH Password authentication method. This means the user needs to enter his/her SSH user ID and SSH password in order to establish an SSH connection. This should be done when connecting through the display session configuration or connection pool.

Uppercase Input

There are fields or screens that their input, whether uppercase or lowercase, is automatically uppercased by the host after receiving it from the user. Therefore, these screens should be configured in Natural Screen Tester to have their input fields contents uppercased before being sent to the host, otherwise Natural Screen Tester Server will not handle them correctly.

There are two options for doing this:

1. You can add some code for uppercasing the relevant fields before sending them to Natural Screen Tester Server.
2. In the Host Configuration dialog box, check the check box **Uppercase input**. This will convert to uppercase all fields sent to the host. This configuration may be overridden for specific screens using the test project parameter described in the previous section.

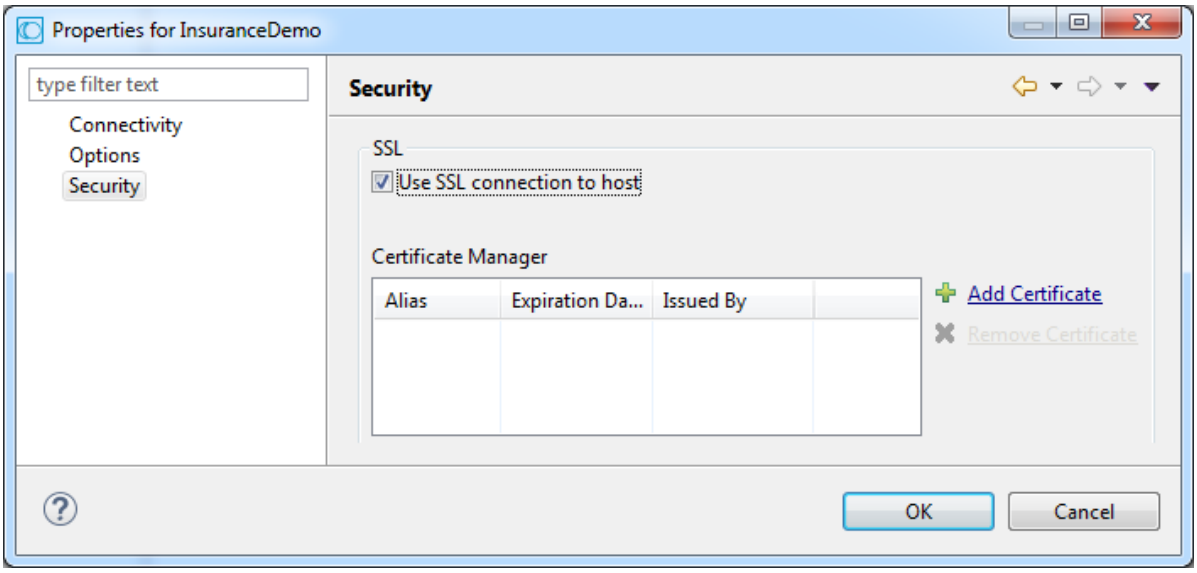
Configuring the SSL Connection

SSL connection is used to ensure a secure connection between the Natural Screen Tester server and the remote host. Connecting using SSL V3 enables encrypting all the traffic between the server and the host and requires the server to present proof of identity (server authentication).

Note: This can be used in any block mode host, however this has only been tested on a 3270 Mainframe host.

To define the SSL connection

1 In Natural Screen Tester, enable the SSL Connection option for the specific host (access the *Host Properties* dialog box of the relevant host, and in the *Security* tab check the **Use SSL connection to host** check box).



2 Add a valid X509 certificate to connect to the SSL enabled host.

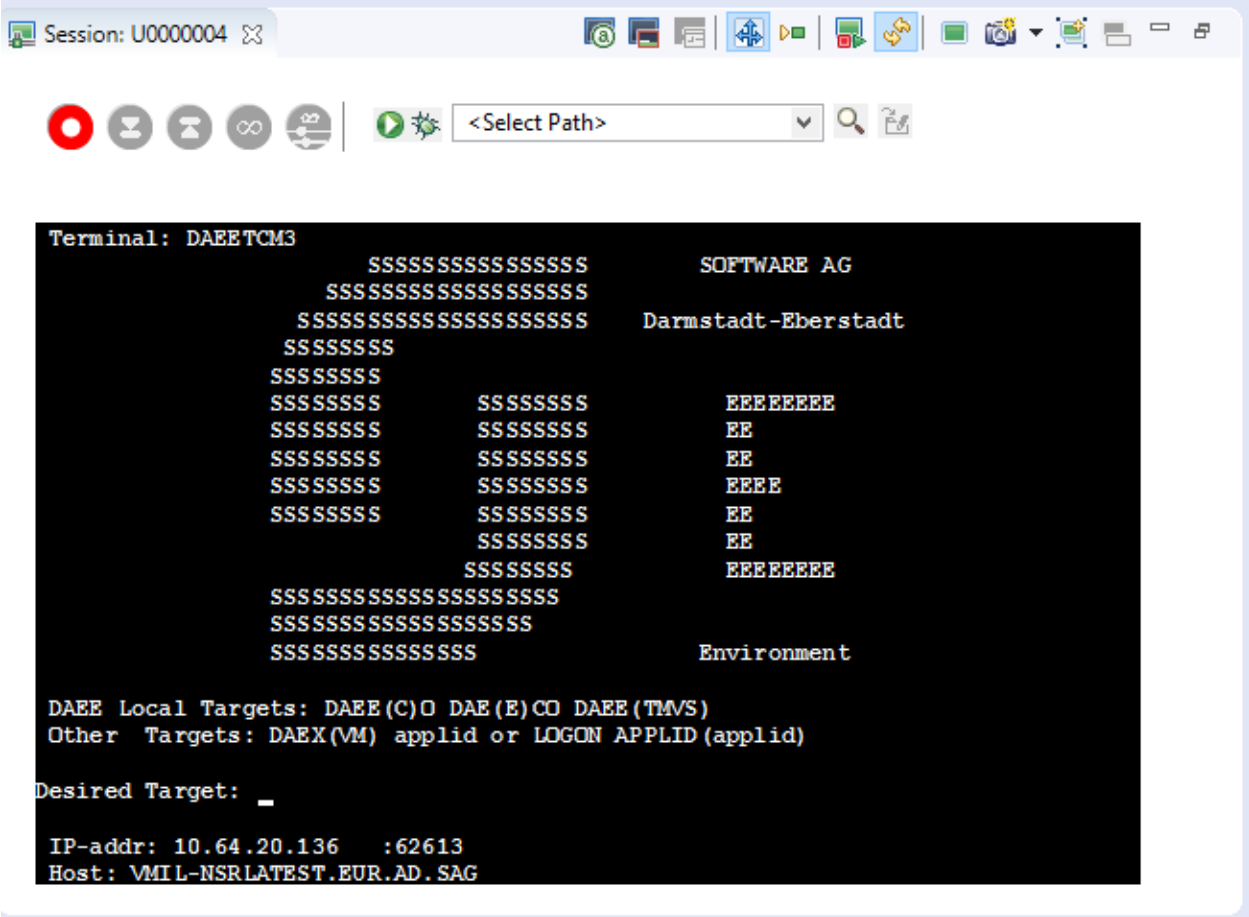
Refer to SSL Cipher Suites Supported by Natural Screen Tester.

Defining Host Parameters Required when working with RPC

In the Host Properties dialog box, set the RPC parameters. Refer to RPC Parameters for details.

3 The Session View

The Session View is especially useful when working with screens or test cases, either offline (replay file) or online.



The toolbars are described under Session Properties and Toolbars in the *Natural Screen Tester Reference Guide*.

Creating a Display Session

The definitions configured here override test project definitions and are relevant for a specific session.

To create a display session

1 Open the relevant test project, right-click on the **Sessions** node and select **New Display Session Definition**.

2 Enter a name and description for the session.

3 Select the connectivity type:

- **Use test project configuration**: Use the configuration set in the Test Project Properties.
- **Online**: Connect online to the host, and enter the name of the device.
- **Offline (using trace files)**: Connect to the session using a trace file. Browse and select the required trace file.
- **Connection Pool**: Select a connection pool from which to take the session.

4 Click **Next** to define the Session General Properties.

5 Enter the session ID and password.

6 Select whether or not to create a trace file and where and how it should be saved.

7 Click **Finish**.

Duplicating a Session

Duplicating a session will create a new session definition will be created using the same configuration as the original one. Session definitions can be duplicated by right-clicking on the original session and selecting **Duplicate Session Definition**.

Changing the Screen Definition Mode

Screen Creation Definitions are a set of basic definitions which are used to create a new Natural Screen Tester screen. These definitions include the default screen name, initial identifiers and determine whether input fields will be created. Creating a new screen using screen creation definitions reduces the complexity and the time required to identify new screens.

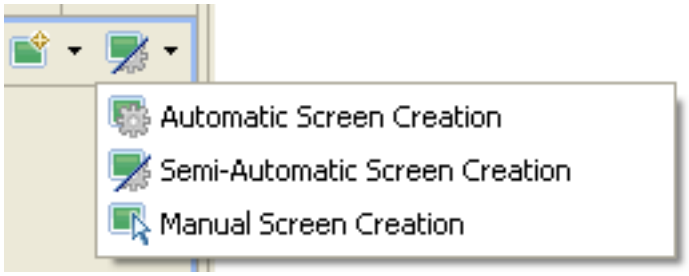
How does it work?

To use this feature, at least one set of Screen Creation Definitions must be created within a Screen Group (in the dedicated tab). When navigating within a session, and reaching an unidentified screen, Natural Screen Tester searches for Screen Groups which match the unidentified screens, and then creates a new screen based on the screen group's identifiers and on parameters defined in the creation definition configured in the Screen Group. This process can be initiated in three different modes:

- Automatically: A screen will automatically be created for each unknown screen that matches a screen group. This screen will include basic screen definitions, which are defined in the screen group.
- Semi-automatically: The New Screen Wizard will automatically be displayed for each unknown screen that suits a screen group. This screen will include basic screen definitions, which are defined in the screen group.

- Manually: The New Screen Wizard is displayed when clicking on the Identify New Screen icon.

The desired mode is determined by clicking on the **Change Screen Definition Mode** icon in the Session View.



New Screen/Screen Group

An unidentified screen can be identified as a screen/screen group by clicking on the New Screen icon in the Session Toolbar. The Create New Screen wizard is displayed. Refer to **Creating a New Screen** or **Creating a New Screen Group** for further details.

Updating a Screen's or Screen Group's Image

The screen image of a screen can sometimes change and no longer suit the relevant screen entity. In the Session View there is a **Update Screen Image** icon, which enables updating the screen image. Click on the arrow next to the icon to view all the relevant screens/screen groups and select the desired one.

Recording a Test Case

A test case can be created either via the Session view or by creating a new test case entity (refer to Creating a Test Project for further details). When creating the test case in the Session, use the test case toolbar to record the test case and then you can edit the test case in the test case dialog box.

To create a test case from the session

1. Navigate to the first screen from which you want the recording to commence.
2. Click on the icon to display the **Test Case Toolbar**.
3. Click on the **Start recording** icon.

4 Navigate to the various screens to record the relevant test case.

5 Click the **Mark possible input** icon to mark all the predefined potential input fields in purple. Click on one of the marked fields you want to include as input for your test case. The field will be marked orange.

6 Click the **Mark possible output** icon to mark all the predefined potential fields in purple. Click on a field to define it as an assertion for your test case. The field will be marked blue.

7 If you want to record a step that repeats again and again until a certain condition is reached, click the **Loop** icon, and follow the instructions in the wizard. This kind of definition can be applicable for collecting data of a host table.

8 To end a recording click the **Stop recording** icon. Enter a name for the test case.

9 Click **Finish**. The editor will open where you can manually edit and add nodes.

Running/Debugging a Test Case

A test case can be run/debugged in the session viewer, by selecting a test case from the list and then clicking on the run/debug icon.

When running a test case, you will be required to insert inputs as necessary.

When debugging a test case, the **Debug** perspective will open.

Testing the Test Project Map

The screen navigation defined in the Test Project Map can be tested to ensure that the navigation behavior is as expected. This is done in the Session View, using the Test Project Map toolbar. The toolbar enables selecting a screen to which you expect to be able to navigate to from the current screen and then attempting to navigate to this screen. If the Test Project Map is correctly defined, then Natural Screen Tester will successfully navigate to the selected screen. If the Test Project Map does not have the relevant steps defined to reach the screen you selected or if these steps are not "Approved" steps, a pop-up message will inform you of this.

Note: The Test Project Map toolbar is not available when working offline.

See Test Project Toolbar.

Navigating within a Session

Within a session, you can navigate online, interacting directly with the host, or to replay a previously recorded navigation session. When working directly with the host, you can create a trace file that records the navigation between the screens. This can later be used and replayed, enabling you to work offline (in replay mode).

Online Mode Navigation

Online session navigation using the session view is similar to regular host navigation:

Using Keyboard Keys

- Use the keyboard to enter any alphanumerical input.
- Use the Enter key to send Enter to the host.
- Use the function keys (F1-F12) or the PF buttons toolbar to send corresponding PF1-PF12 keys to the host.
- Use the SHIFT key combined with the function keys to send PF13-PF-24 keys to the host (Shift+PF1=PF13, Shift+PF2=PF14 and so on).
- Use the Customize Host Keys to define and send any other host key.

Creating Trace Files

When working online, you can use the Trace file feature to record a file, and trace the connection communication (connection pool or user) between the Natural Screen Tester server and the host, for each connection. Details for setting the default trace file parameters can be found in Recording Trace Files. You can override the test project settings and define settings per session either when creating a new session connection or in the session properties of an existing session.

Replay Mode Navigation

In order to debug or reconstruct specific scenarios, you can replay (GCT) files that have traced the steps of these scenarios. In order to replay such files, the test project configuration definitions must indicate that instead of working online with the host, Natural Screen Tester must, when connecting, access and replay the specific file. The default trace file used is defined in the Test Project Properties dialog box (refer to **Working with Offline Sessions** for further details). To override these settings, define the relevant settings per session either when creating a new session connection or in the session properties of an existing session.

When replaying a file, you may want to navigate to a specific screen. The replay navigator enables this, simply and efficiently, using the Replay Navigator slider. Note: In the replay mode, any function keystroke (such as [ENTER], PF keys etc.) displays the following screen, according to the

screen order in the recorded GCT file, therefore, key selection is meaningless. This feature is not available when connecting to a host connection pool (online or offline).

Using the Replay Navigator

1. When connecting to a session which has been defined as working offline, in replay mode, (refer to **Working with Offline Sessions** for further details) the Replay Navigator slider is displayed.

   This toolbar is displayed by default and can be hidden by clicking on the toolbar icon.

   To navigate to the different screens either drag the slider's indicator along until reaching the relevant screen number (the screen number is indicated by a ToolTip), or enter the number in the box to the right of the slider and click the **Go** arrow or select a screen from the list and click the Go arrow. The slider indicator as well as the number in the box on the right will change according to the present screen number.

   You can search for screens associated with a specific screen group, by selecting the screen group from the list of screens/screen groups. Clicking on **Go** will search for the first screen encountered, which is associated with the selected screen group. Clicking **Go** additional times will search for additional screens associated with this screen group.

2. Click on the Show user input icon (the last button) to display the input that the user entered while the GCT was recorded (when the button is not clicked the screen is displayed as it was before the changes were made). When the button is clicked, the contents of the input fields that were changed appear in red, a string representation of the host key sent appears in the toolbar and the cursor is positioned in the position it was in when the screen was sent to the host.

Note: When replaying a file that has a Connection Pool, the Replay Navigator slider is not available for the Natural UNIX protocol.

Using the Test Project Toolbar

Select the destination screen from the drop down list of screens, then click on the Navigate to screen icon. See Test Project Map Toolbar.

Note: The Test Project Map toolbar is not displayed when working offline.

Switching between Online and Replay Mode

Switch between these modes by changing the connection type in the Session Properties, or by editing the file _inc_applconf.asp.

Displaying Windows when Navigating within a Session

When there is a window within a screen, you can define that the screen area outside the window will be grayed out or displayed in regular colors. This is done by clicking on the Window icon on the session toolbar  .

Note that it is not possible to navigate outside the window.

4 Working with Entities

Creating a New Entity

New entities can be created via the **File > New > Entity** menu item. The name you enter can only include a-z, A-Z, 0-9, or "." and must not start with a digit (0-9) or ".".

To edit an entity

- Double click on the entity in Natural Screen Tester Explorer. A single click will display the entity's properties in the **Properties** area. The description of the entity can be edited in the **Properties** area.

To delete, copy or paste an entity

- Right-click on the entity in Natural Screen Tester Explorer and choose the relevant option.

To locate an entity

- Right-click on the repository in the Natural Screen Tester Explorer and choose **Open Entity...**.

  Or:

  From the **Navigate** menu, choose **Open Entity...**. Use "?" to substitute a single character or "*" to substitute a string.

Creating a New Folder

As a Natural Screen Tester user, you may like to divide test projects into folders according to the specified interests and needs, such as development teams, subprojects etc. The same folder contains different types of entities and subfolders varying according to each test project. This allows you greater flexibility organizing your entities. A new folder can be created via the **File>New>Entity>New Folder** menu item.

Copying Information from Host Screens

The Designer has built-in access to the host session, this session can be used to make the definition of entities easier. This is achieved using the Session's Copy from session feature. To copy from the session, mark the relevant string of data in the host's screen, and next to the relevant field click the Copy from session icon  . The string and/or position are copied.

Copying Entities

Entities can be copied between from folder to folder within an test projects and/or between test projects/servers. This can be performed by dragging and dropping the relevant entity or by using the Copy/Paste actions. When copying between test projects, the pasted entities will be placed in the root node of the repository.

Note: When coping an entity to a test project that has the same entity name, the existing entity will be overwritten by the new one. The timestamps of the new entity (for example creation and modification date) also overwrite the values of the original entity. These changes take effect immediately, not after reloading the test project.

Opening/Finding a Specific Entity

You can search and open a specific entity in the Editor area by right-clicking on the Repository node and selecting **Open Entity (Find)...**. The search can be filtered by entity and you can determine to search within a specific folder. You can sort the results by created/modified timestamp.

Renaming an Entity

You can rename an test project or entity (though not a field) by selecting it in the Natural Screen Tester Explorer and then either selecting **Rename...** from the **File** menu or by right-clicking on the test project/entity and selecting **Rename...**.

Renaming Fields

Field renaming is performed via the Field Mappings tab in the Screen Editor where the relevant field is mapped. Before renaming a field, you may want to see which other entities refer to this field. This is done by clicking on the **Find Referring Entities** icon next to the **Field name** field. To rename a field, click on the **Edit Entity** icon next to the **Field name** field, and access the field editor. In the field editor click, on the **Rename Field** button and rename the field.

Note: A field which does not have references can be edited directly in the **Field name** field in the screen editor.

Viewing an Entity's References

Natural Screen Tester entities may be interrelated with each other. The Natural Screen Tester Server maintains tables that keep track of entities that refer to other entities. For example, a field entity: the screens it is mapped to, the procedures it is an input or output parameter of, or the paths that employ it as an input / setting variable / test by field. This feature allows you to keep in check the database and avoid situations in which deleting one entity makes another entity non valid.

To view an entity's references

- In the Natural Screen Tester Explorer, right-click on the entity and select **Find Referring Entities**.

  The references are displayed in the reference view. Double-click on an entity to edit it.

5 Screens

Introduction

A single Natural Screen Tester screen represents a corresponding screen in the host. A Natural Screen Tester screen is usually defined using unique text identifiers from the host screen, but it is also possible to use other identifiers such as the position of the cursor when the screen is loaded or field attributes (Protected, Intensified or Hidden). When navigating in the host application, Natural Screen Tester tries to find the screen entity that matches the host screen it encounters. Once a match is found, Natural Screen Tester can use the data and definitions of the host screen in its various components. Sometimes the screen image needs to be updated, click on the Capture image icon  to update the image. The following guidelines will help you determine which screens to identify:

- screens that will be part of navigation paths (recommended for easier path definition)
- screens that will contain field mappings
- any important screen in the host application

To allow the Natural Screen Tester Server to recognize the corresponding host screen; each screen must be identified by giving it a unique name, and defining the screen's identifiers. In addition, when relevant, you may want to map fields to the screen, associate screen groups to the screen and define steps in the application map.

Natural Screen Tester screens can be created

- manually, via the Session View
- by importing application host maps such as Natural, BMS and MFS
- automatically or semi-automatically, using identification definitions

Creating Screens from the Session View Manually

Introduction

Screens can be created via the **New Entity** menu or from within the session view.

Note: The screen must be displayed in the session view when creating the new screen.

To create a screen via a session

1 In the Emulation session, navigate to the relevant screen.

2 Select text in the emulation screen.

3 Click on the **Identify Screen** button. The New Screen wizard is displayed, with a suggested name displayed in the Name field. Enter a suitable description and determine the folder where the screen is to be located. Click **Finish**. You have now created a new screen entity in the repository.

Define Identifiers

For the Natural Screen Tester Server to recognize a host screen, you need to identify the screen in Natural Screen Tester. The screen is identified using "identifiers". Different identifier types include identifying specific text on the screen, the screen cursor position or the field attributes. The Natural Screen Tester Server analyzes the current screen and tries to match it to all of the screen identification strings stored in the application repository (database). For a screen to be recognized, all its identifiers must be matched. You may define an unlimited number of identifiers. The following identifiers are available:

- Window Identifiers

Text Identifiers

Use text identifiers to identify a screen by selecting specific text that appears on the screen. Use the Capture from Screen feature to identify a string of text accurately from the host screen and insert this string into the text box.

To define a text identifier

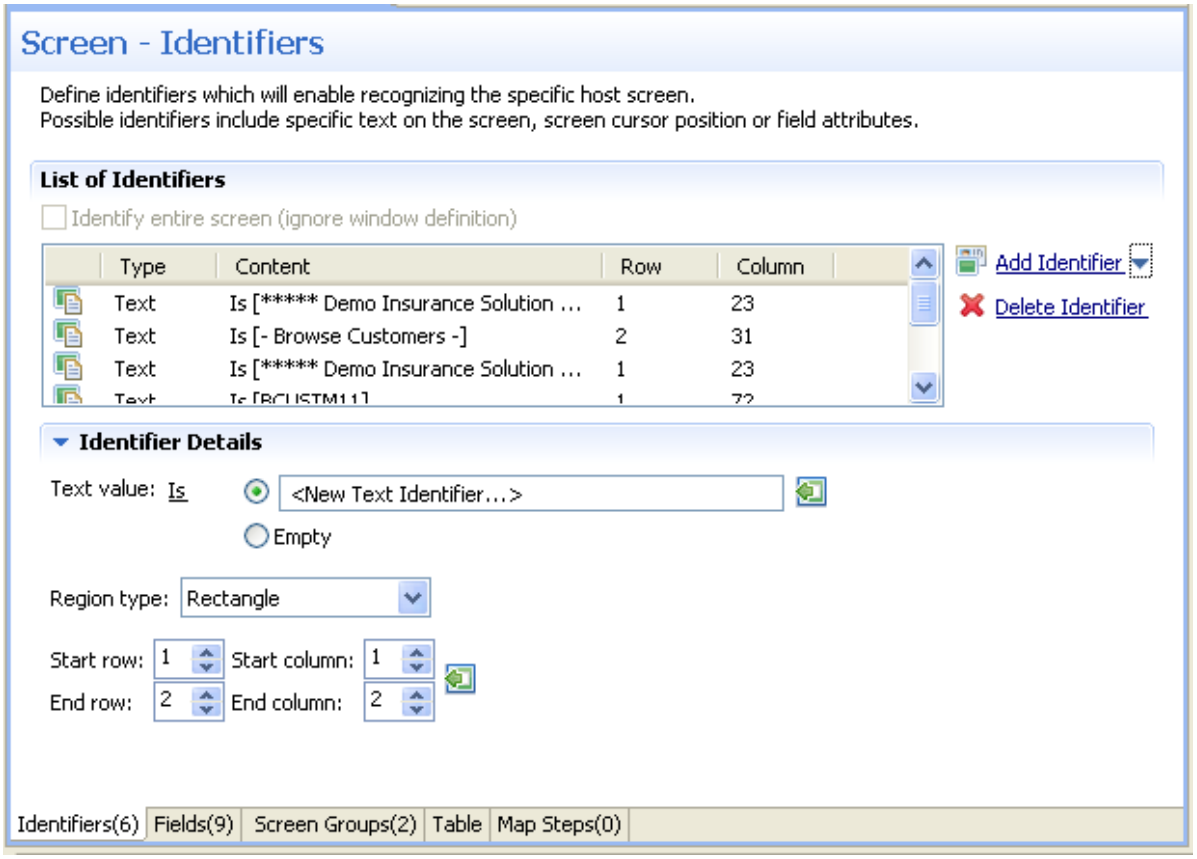
1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 Determine whether to ignore the application's window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used. This check box is only enabled when the screen has a window.

Note: When changing the state of the **Ignore entire screen (ignore window definition)** check box, you must manually edit the identifiers and mappings to support the new positions.

4 Click the arrow on **Add Identifier** and select **Text**. Another row will be added to the list of identifiers.



5 Select **Is** (default) in order to match the exact text on the host screen. Alternatively, select **Is NOT** if any text other than the specified is suitable as a screen identifier. Check **Empty** to identify an empty space on the host screen. This clears the **Text** text box.

6 Select **Rectangle** to indicate that the identifier should be searched for within the defined rectangle, **Position**, to indicate that the identifier must start at a specific position or **Anywhere on screen** to indicate that the identifier may be anywhere on the screen.

7 If you select Rectangle, define the rectangle range (start and end row and column). If you select Position, select the row and column.

8 It is also possible to add an identifier by selecting the area in the host screen, and then clicking Add Identifier. If you select an area that is a rectangle, identifiers will be added for each and every row of the selected rectangle.

Cursor Identifiers

Use the Cursor Position identifier to identify a screen by the cursor position when the screen is loaded.

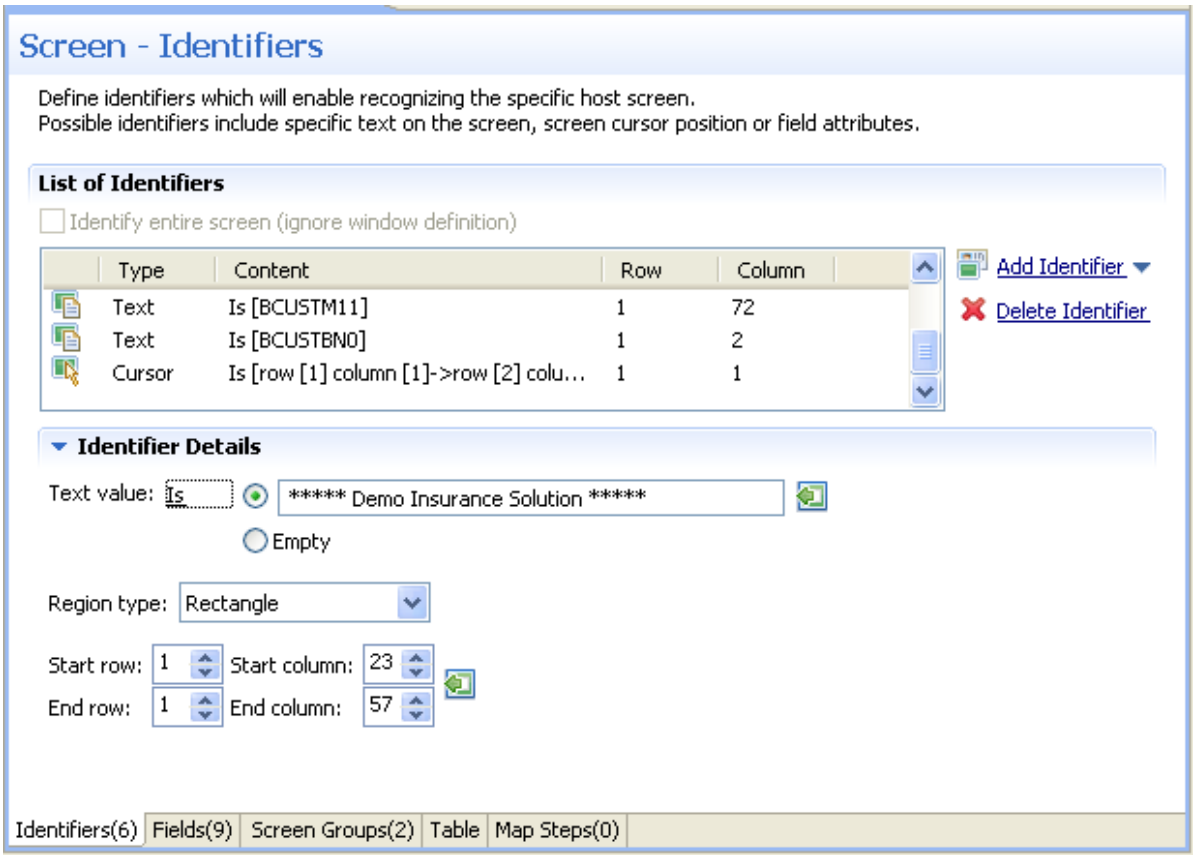
To define a cursor position identifier

1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 To ignore the application's window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

4 Click the arrow on **Add Identifier** and select **Cursor Position**. Another row will be added to the list of identifiers.



5 Select **Is** to indicate that the cursor is positioned within the region area details that follow. Alternatively, select **Is NOT** to indicate that the cursor is not positioned within the region area details that follow.

6 Select **Rectangle** to indicate that the cursor position should be searched for within the defined rectangle or **Position**, to indicate that the cursor position must start at a specific position

7 It is also possible to add an identifier by selecting the area in the host screen, and then clicking Add Identifier.

Attribute Identifiers

Attribute Identifiers: Use the Attributes identifier to identify a screen by the host's attributes. Attributes of the host screen may be Protected, Intensified or Hidden.

To define an attribute identifier

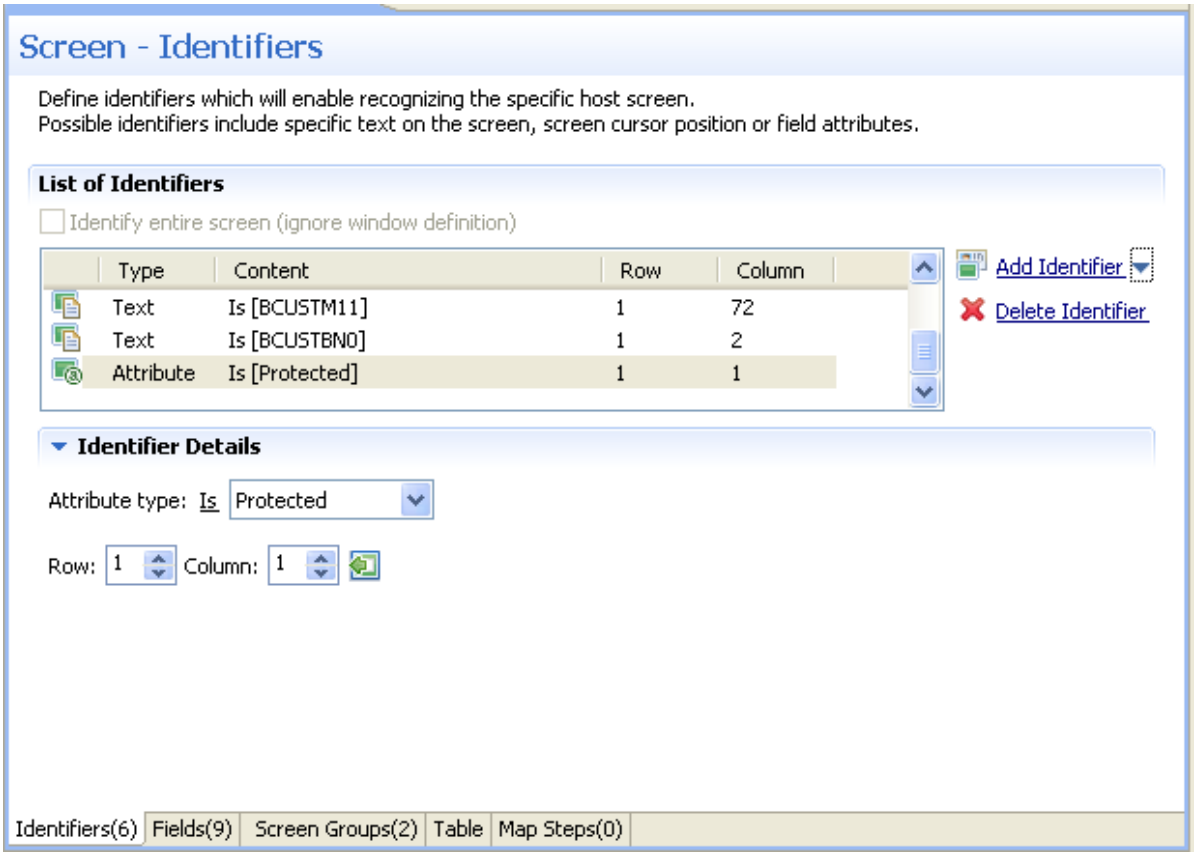
1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 To ignore the application's window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

Note: When changing the state of the **Ignore entire screen (ignore window definition)** check box, you must manually edit the identifiers and mappings to support the new positions.

4 Click the arrow on **Add Identifier** and select **Attribute**. Another row will be added to the list of identifiers.



5 Select whether the Attribute Type **is/is not Protected**, **Hidden**, **Intensified** or **Reversed video**.

6 Enter values or use the Capture from Screen feature to indicate a specific location of the attribute by its **Row** and **Column**.

7 It is also possible to add an identifier by selecting the area in the host screen, and then clicking Add Identifier.

Window Identifiers

Use the Window identifier to identify a screen by determining whether it is or is not a window. In Natural UNIX, it is possible to identify a screen by determining whether it is a window and whether it has specific text in the title.

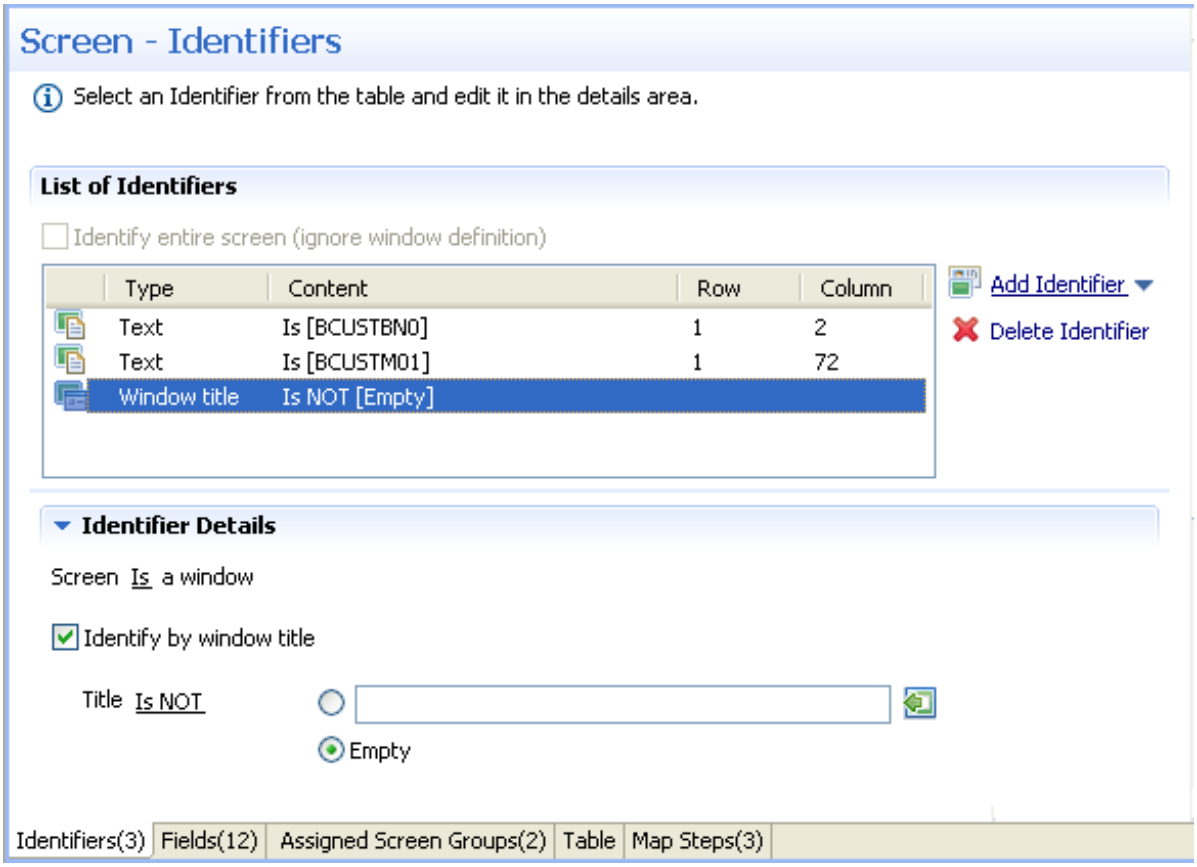
To define a window identifier

1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 To ignore the application's window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

4 Click the arrow on **Add Identifier** and select **Window**. Another row will be added to the list of identifiers.



5 Select **Is** to indicate that the screen is a window. Alternatively, select**Is NOT** to indicate that the screen is not a window.

6 For Natural UNIX hosts: Select the check box to indicate to identify by the window title. You can enter text for the title or select the text in the session view and click the Capture from screen icon. You can also identify a screen which is a window and does not have a title by selecting the **Empty** radio button.

Mapping Fields

Introduction

When a screen is identified, field entities can be mapped to it. A field makes relating to a string on the host screen much simpler. Fields offer an enhanced way of referring to host fields - by their name rather than by their position. In addition, mapped test project fields simplify future maintenance of the test project - when a field changes in the original host application, updates only need to be made to the application field definition. The following guidelines will help you determine which fields to identify:

- Fields that will be part of navigation paths.
- Fields that will be used as test case input and output attributes.
- Recommended: all input and output fields (not constants).
- Any important field in the host application.

There are limitations when defining a mapping. A mapping cannot:

- Start in an unprotected field and end in a protected field, or vice versa.
- Start with an attribute byte.
- Overlap another mapping on the same screen.
- Start in one line and end on another line.
- Start on one line and end off the screen's limits.

An identified field makes relating to a string on the host screen much simpler. Fields offer an enhanced way of referencing host fields - by their name rather than by their position. Using fields has many advantages:

- A field's ID is meaningful regarding the contents of the field, unlike a numeric position.
- You define a field only once, and later can reference that definition in all of your code - knowing you always mean the same definition.
- If changes in the host application occur in the future, the only place that requires to be changed is the field definition.

Field properties can be accessed by right-clicking on the Repository node and selecting **Open Entity (Find)...** and then searching for the specific field.

In the Field Editor, it is possible to set the Input Mask (Defines the values that can be entered in the field) and the field type (numeric or alphanumeric).

When a screen is identified, field entities can be mapped to it. A field makes relating to a string on the host screen much simpler. Fields offer an enhanced way of referring to host fields - by their name rather than by their position. In addition, mapped test project fields simplify future maintenance of the test project - when a field changes in the original host application, updates only need to be made to the application field definition.

Fields can be mapped according to their position in the screen or according to their leading label - dynamic field mapping (do not set the same field to be mapped by position and by its leading label).

Mapping according to leading labels (dynamic mapping) is particularly useful in the following cases:

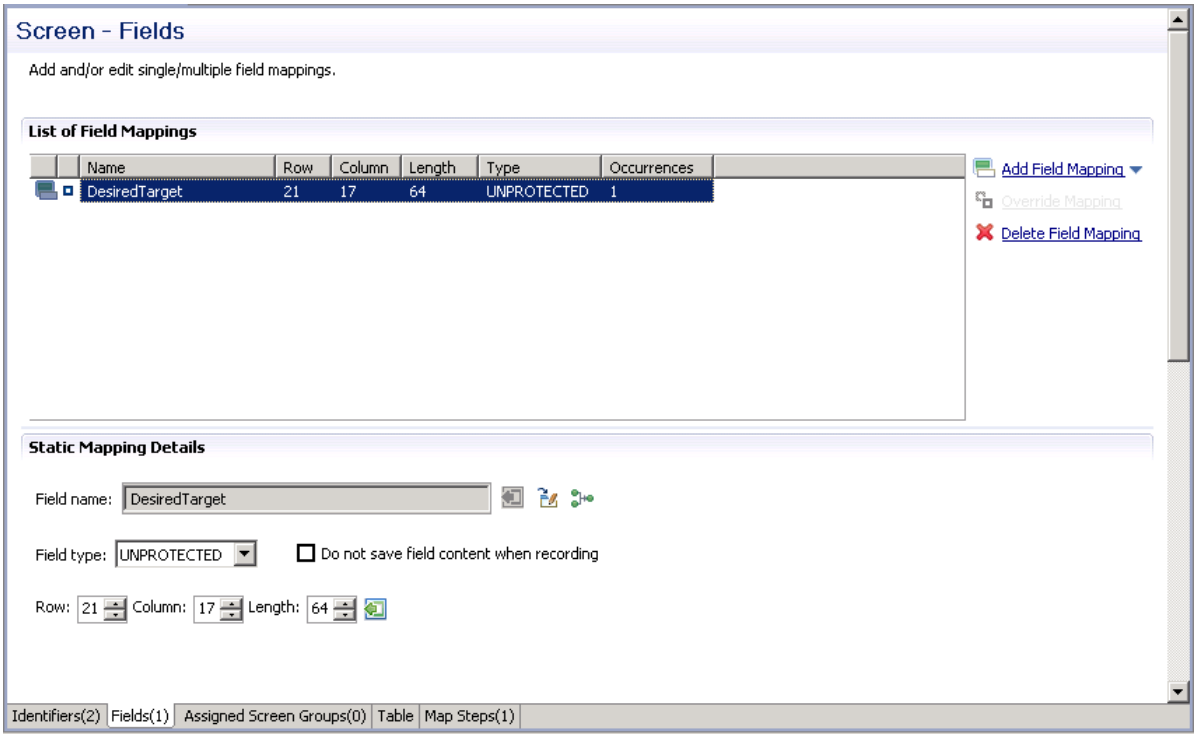
- The application has fields that are dynamically drawn on the screen.
- Host applications are sometimes changed and items can be moved. Using dynamic field mappings, the fields will continue to be mapped and identified and the application will not be affected.
- Using dynamic field mappings enables more flexibility when using Screen Groups, as Screens which include the same field, can be associated with the same Screen Group even when the field is located in a different position.

Mapping Fields According to Position on the Screen

To map fields according to their position in the screen

1 Click the **Fields** tab to view and/or modify the fields in this screen.

2 Click the arrow next to **Add Field Mapping**, and select **Single Mapping** or **Multiple Mapping** to add a new field mapping.



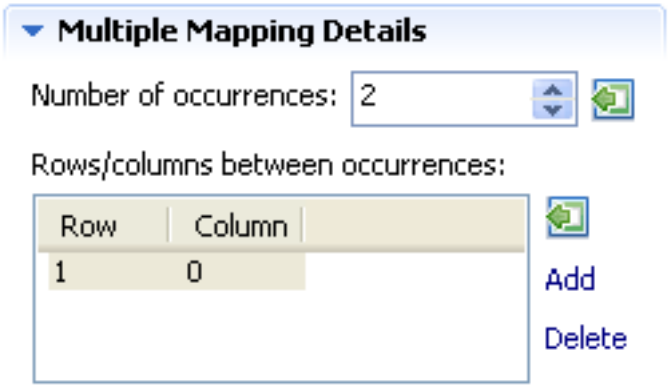
3 In the **Field name** field, either enter a field name or enter the first letters of a field name and then click CTRL and SPACE to display relevant fields.

4 Select the **Protection type**: **Protected** (cannot be edited in host), **Unprotected** (editable in host) or **Both** (may sometimes be editable and sometimes not).

5 The **Do not save field content when recording** check box enables not saving data which you do not want to be displayed when viewing the recorded trace file.

6 The **Row** and **Column** fields indicate the position of the field on the screen. The values of these fields can be changed using the **Capture from Screen** feature or by entering values in the fields.

7 When adding a multiple field mapping, determine the number of occurrences and the rows and/or columns between occurrences.



Note: You can automatically map all of the unprotected fields in the screen by clicking on the **Automatically identify unprotected fields** icon.

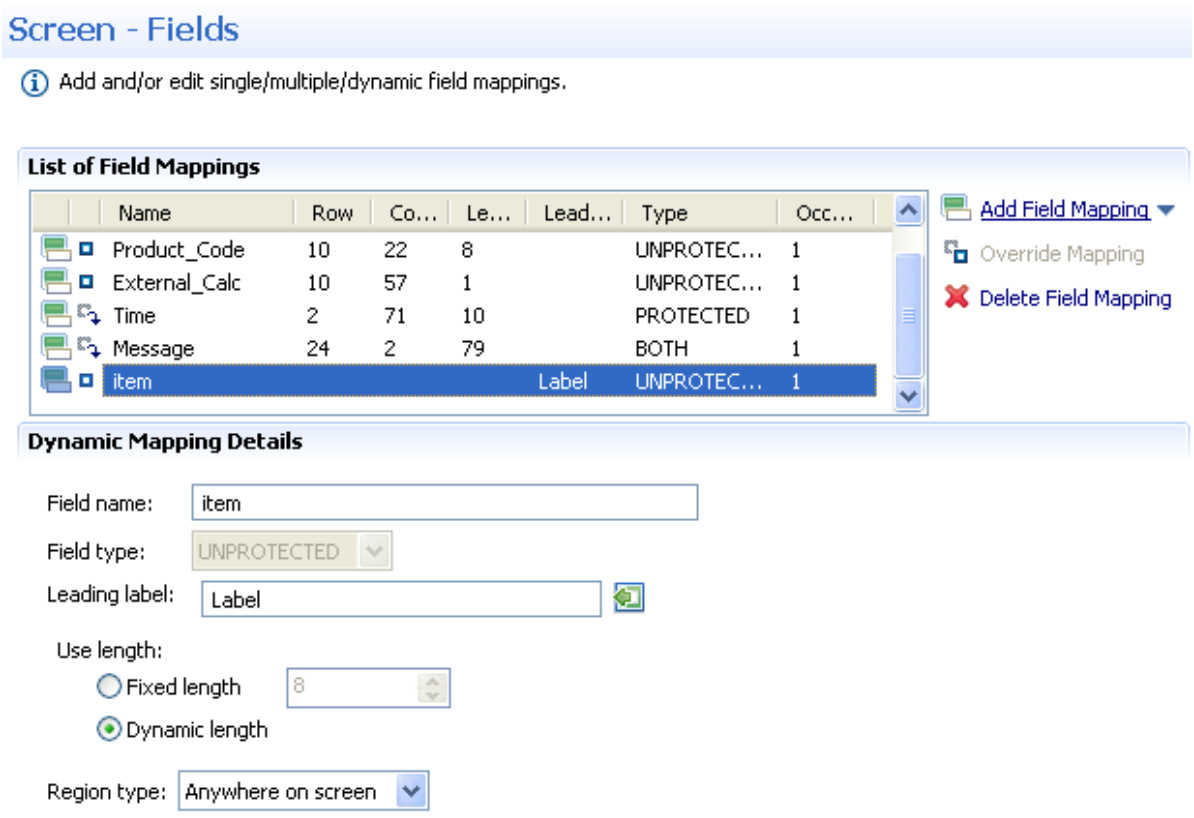
Mapping Fields According to Leading Label

To map fields according to their leading label (not available in right to left and character mode applications)

In some screens, the position of a field may vary. It is therefore necessary to map some fields in such a way, that even if they appear in a different position, they will still be recognized and mapped. This is possible by defining the label near the field (the label must be to the left of the field) and identifying the field according to this label. This mapping type is called "Single Dynamic" as the mapping position changes dynamically according to the leading label. Refer to *Dynamic Field Mapping Limitations*.

1 Click the **Fields** tab to view and/or modify the fields in this screen.

2 Click the arrow next to **Add Field Mapping**, and select **Single Dynamic** to add a new field mapping.



3 In the **Field name** field, either enter a field name or enter the first letters of a field name and then click CTRL and SPACE to display relevant fields.

4 Select the field protection type: Unprotected, Protected or Both. Prefer to set the field type either as **Protected** or **Unprotected** when you are sure of the type of the dynamic field. Use **Both** when you are not sure of the type of the dynamic field.

5 Determine the leading label: the leading label is the label which appears before the field. This label is inserted as the Field name and can be edited. The label you define should be as accurate as possible (do not use strings which normally appear within the field).

6 Determine the method to use to search for the label:

- Contains text: Searches for the given text anyway in the defined region (default).
- Exact phrase: Searches for the given text within the defined region. The text must be preceded and followed by at least two white space characters.
- Contains word: Searches for the given text as word(s) within the defined region.
- Regular expression: Searches for the given pattern within the defined region. Refer to *Regular Expression Syntax* in the *Natural Screen Tester Reference Guide*.

Note: If you select a different option, after defining a regular expression search definition, the value of the previous regular expression definition will be lost.

7 Define the region type in which the label will be searched for: anywhere in the screen, within a specified rectangle or within a specified row range. The **Row** and **Column** fields indicate the position of the label on the screen. The values of these fields can be changed using the **Capture from Screen** feature or by entering values in the fields.

8 Use length:

In Unprotected fields: Determine whether to recognize a field mapping according to the leading label and the length of the input field (Fixed length) or just according to the leading label (Dynamic length).

In Protected fields: The length defined here is the length of the mapping. The identified field is cropped to the specified length.

9 Define the region type in which the field will be searched for: anywhere in the screen, within a specified rectangle or within a specified row range. The **Row** and **Column** fields indicate the position of the field on the screen. The values of these fields can be changed using the **Capture from Screen** feature or by entering values in the fields.

Note: The region will be displayed in the Session View with a yellow frame, and the background of the screen within this region will be dotted.

Guidelines

- It is preferable to define your regions as rectangles, that do not overlap each other, and whose size are minimal.
- The feature is best suited to locate dynamic fields which change their location vertically so that labels are aligned vertically in one region, and fields aligned vertically on a region to the right of the labels.
- Mappings inherited from screen groups are indicated with an icon. You can override this mapping to suit the current screen by clicking on the **Override** hyperlink. Hereafter, this mapping will be marked with the icon. You can restore the inherited mapping by clicking on the **Restore Inherited Mapping** hyperlink. Note that when selecting an inherited mapping, you cannot change the mapping properties. When selecting a local mapping, all properties can be changed. If you select an overridden mapping, all properties can be changed, except for the name.
- To delete a field mapping, click on the **Delete Field Mapping** hyperlink.
- It is not possible to remove mappings which are used in other entities, such as in procedures or tables.

Sorting

You can sort the contents of a **List of Field Mappings** table by field name or by any other column. Click the header of the column by which you want to sort. The contents are sorted in ascending order. Click again to sort in descending order. This feature is useful when detecting duplicate field entries. The sort will be cancelled when you close the screen entity.

Creating Screens from the Session View using Screen Creation Definitions

Screen Creation Definitions are a set of basic definitions which are used to create a new Natural Screen Tester screen. These definitions include the default screen name, initial identifiers and determine whether input fields will be created. Creating a new screen using screen creation definitions reduces the complexity and the time required to identify new screens.

To use this feature, at least one set of Screen Creation Definitions must be created within a **Screen Group** (in the dedicated tab). When navigating within a session, and reaching an unidentified screen, Natural Screen Tester searches for Screen Groups which match the unidentified screen, and then creates a new screen based on the screen group's identifiers and on parameters defined in the creation definition configured in the Screen Group. This process can be initiated in three different modes:

- Automatically: A screen will automatically be created for each unknown screen that matches a screen group. This screen will include basic screen definitions, which are defined in the screen group.
- Semi-automatically: The New Screen Wizard will automatically be displayed for each unknown screen that suits a screen group. This screen will include basic screen definitions, which are defined in the screen group.
- Manually: The New Screen Wizard is displayed when clicking on the Identify New Screen icon.

The mode is set in the session view by clicking on the Change screen creation mode icon in the tool bar. Refer to *Changing the Screen Definition Mode*.

This section covers the following topics:

Introduction

Using Screen Creation Definitions Manually

The difference between creating a screen in the manual mode and creating a screen in the semi-automatic mode, is that in the manual mode, when you reach an unidentified screen, you are required to click on the Identify New Screen icon in the Session view. You can then select from a list of Screen Groups relevant to this scReen, the screen group which has screen creation definitions which you would like to use (you can select to create a new screen without using screen creation definitions by clicking on **No** in the *Use a Screen Creation Definition* dialog box). By default the list of screen groups which are relevant to this screen are displayed. It is also possible to show and select a screen group from all the application screen groups that have screen creation definitions. If you want Natural Screen Tester to always use the **Screen Group with the highest priority**, select the **Don't ask again** check box. It is possible to display this dialog box again by changing the **Always use the highest prioritized screen group** check box in **Windows > Preferences > Software AG > Natural Screen Tester**.

The new screen has a number of basic definitions such as the screen name and a number of identifiers. These are determined according to the definitions in the **Screen Creation Definition** tab in the relevant Screen Group entity (You can select to create a new screen without using screen creation definitions by clicking on **No** in the *Use a Screen Creation Definition* dialog box). The screen name is determined according to the screen name position definition in the Screen Creation Definition tab. The list of identifiers is determined according to the Screen Group identifiers, as well as the strings which appear in the list of identifier locations in the Screen Creation Definition tab. To automatically create input fields in the new screen, select **Automatically create input fields**. The labels near the input fields are used as the input field names.

To manually create screens using screen creation definitions

1 Within the relevant Screen Group, **create a Screen Creation Definition**.

2 Ensure that in the session view the selected mode is Manual Screen Creation: in the Session view click on the Screen Creation Definition mode icon and select Manual Screen Creation.

3 Navigate within a session.

4 Once a screen group is recognized within an UNKNOWN screen, click on the Identify New Screen icon. The *Use a Screen Creation Definition* dialog box is displayed. Here you can determine whether to base the new screen on a Screen Creation Definition or not. Click **No** to create a new screen without using Screen Creation Definitions.

5 This dialog box displays, by default, the screen groups which have screen creation definitions and are relevant to the UNKNOWN screen. You can display all the screen groups which have screen creation definitions and select a different screen group. Click **Yes.**

6 The New Screen wizard is displayed. The default name is taken from the text which appears in the screen name position definition in the Screen Creation Definition tab. Click **Finish.**

7 The new screen is displayed in the Editor and is listed in the repository.

Refer to ***Automatically Creating Screens*** and **Semi-Automatic Creation of Screens**.

Using Screen Creation Definitions Semi-automatically

Similar to the automatic mode, when navigating in a session and a screen group is recognized, and the current screen has not yet been identified, the process of creating a new screen entity is initiated. The new screen has a number of basic definitions such as the screen name and a number of identifiers. These are determined according to the definitions in the **Screen Creation Definition** tab in the relevant Screen Group entity (You can select to create a new screen without using screen creation definitions by clicking on **No** in the *Use a Screen Creation Definition* dialog box). The screen name is determined according to the screen name position definition in the Screen Creation Definition tab. The list of identifiers is determined according to the Screen Group identifiers, as well as the strings which appear in the list of identifier locations in the Screen Creation Definition tab. To automatically create input fields in the new screen, select **Automatically create input fields**. The labels near the input fields are used as the input field names. As there may be more than one Screen Group which has Screen Creation Definitions, it is necessary to determine which screen group is to be used. This is done in the *Use a Screen Creation Definition* dialog box which is displayed once the unknown screen is reached. By default the list of screen groups which are relevant to this screen are displayed. It is also possible to show and select a screen group from all the application screen groups that have screen creation definitions. If you want Natural Screen Tester to always use the **Screen Group with the highest priority**, select the **Don't ask again** check box. It is possible to display this dialog box again by changing the **Always use the highest prioritized screen group** check box in **Windows>Preferences>Software AG>Natural Screen Tester**.

To semi-automatically create screens using screen creation definitions

1 Within the relevant Screen Group, **create a Screen Creation Definition**.

2 Ensure that in the session view the selected mode is **Semi-Automatic Screen Creation**: in the Session view click on the **Screen Creation Definition** mode icon and select **Semi-Automatic Screen Creation**.

3 Navigate within a session.

4 Once a screen group is recognized within an unidentified screen, the *Use a Screen Creation Definition* dialog box is displayed. Here you can determine whether to base the new screen on a Screen Creation Definition or not. Click **No** to create a new screen without using Screen Creation Definitions.

5 This dialog box displays, by default, the screen groups which have screen creation definitions and are relevant to the UNKNOWN screen. You can display all the screen groups which have screen creation definitions and select a different screen group. Click **Yes**.

6 The New Screen wizard is displayed. The default name is taken from the text which appears in the screen name position definition in the Screen Creation Definition tab. Click **Finish.**

7 The new screen is displayed in the Editor and is listed in the repository.

Refer to *Automatically Creating Screens* and **Manually Creating Screens**.

Using Screen Creation Definitions Automatically

When navigating to an unidentified (UNKNOWN) screen in a session and a screen group is recognized, a new screen entity is created. The new screen has a number of basic definitions such as the screen name and a number of identifiers. These are determined according to the definitions in the **Screen Creation Definition** tab in the relevant Screen Group entity. The screen name is determined according to the screen name position definition in the Screen Creation Definition tab. The list of identifiers is determined according to the Screen Group identifiers, as well as the strings which appear in the list of identifier locations in the Screen Creation Definition tab. When more than one screen group is relevant to the screen, the screen creation definition is based on the definition in the **screen group with the highest priority**. To automatically create input fields in the new screen, select **Automatically create input fields**. The labels near the input fields are used as the input field names.

To automatically create screens using screen creation definitions

1 Within the relevant Screen Group, **create a Screen Creation Definition**.

2 Ensure that in the session view the selected mode is Automatic Screen Creation: in the Session view click on the **Screen Creation Definition mode** icon and select **Automatic Screen Creation**.

3 Navigate within a session. New screens are automatically added to the repository.

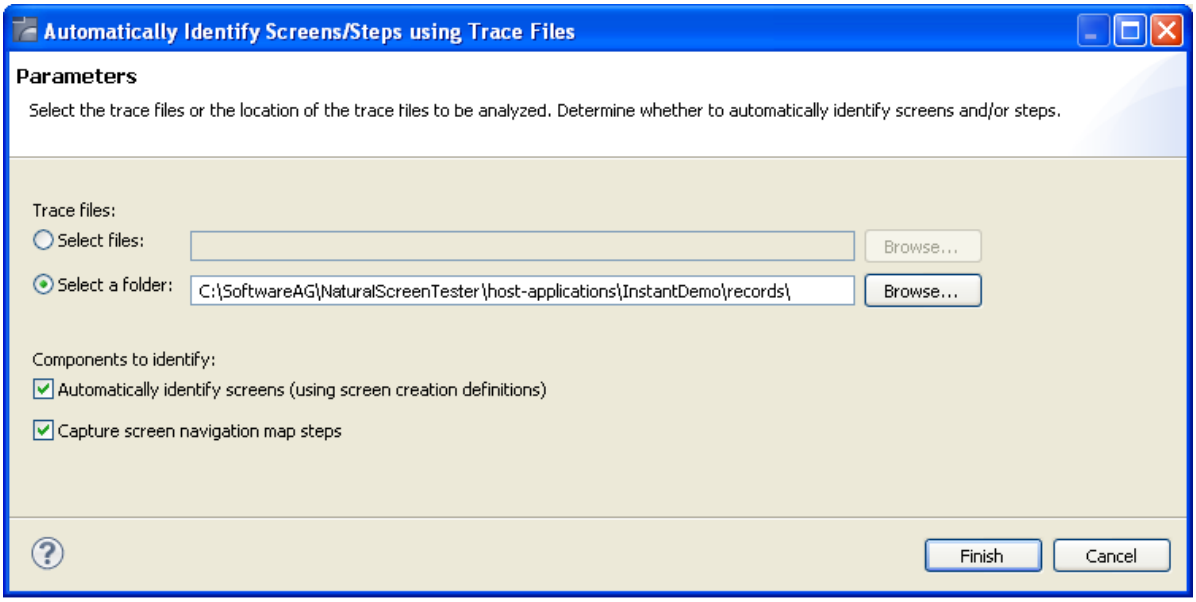
Refer to *Semi-Automatic Creation of Screens* and *Manually Creating Screens*.

Automatically Identify Screens, using the Trace Files (GCT) Wizard

Using trace (GCT) files, you can automatically identify screens. In order to do this, you must ensure that a Screen Group includes a **Screen Creation Definition**.

To automatically identify screens using trace files

1 Right-click on the relevant application, and select **Automatically Identify Screens/Steps using Trace Files...**. The wizard is displayed.



2 Select whether to analyze files or a folder. Locate the files/folder.

3 Select the **Automatically identify screens** check box.

You can automatically identify steps by selecting the **Capture screen navigation map** steps check box. Refer to **Automatically Identify Steps using Trace Files (GCT)** for further details.

4 Click **Finish**.

The Console displays the outcome of the process.

Note: When identifying a new screen, the screen name is taken from the area you defined in the Screen Creation Definitions. If a screen with this name already exits, the newly identified screen's name will be the name as of the existing screen followed by an incremental number. If the area you defined in the Screen Creation Definitions is empty, the screen will be named: "Screen" and then an incremental number.

Importing a Host Application's Maps

- Introduction
- Importing Maps using the Import Host Screen Wizard

- Importing Maps using a Batch File

Introduction

Standard maps, such as Natural, BMS and MFS, are used in host applications to describe the structure of the host screen including the static text and input and output fields. Natural Screen Tester enables importing these application maps, saving time and effort spent on manually identifying screens and simplifying the maintenance process when changes are made in the host application. When importing application maps, a screen is automatically created from each map, minimizing errors that may occur when creating the screens manually, one by one. The Natural Screen Tester screen created includes identifiers (based on the static text in the map) and fields (based on the input and output fields in the map). Natural Screen Tester supports a number of different types of maps:

- NATURAL: Natural map support (from SYSTRANS/SYSOBJH or NaturalONE files).
- BMS: CICS basic map support.
- MFS: IMS message format service.
- SDFX: Natural Screen Tester generic map format, used for other standard maps. To create SDFX files refer to SDFX File Format Definition in the *Natural Screen Tester Reference Guide*.
- SDF: Compatible with Software AG's JIS product.

The import map feature can be used to import an application's maps for a new application or to maintain and update previously imported maps. When updating previously imported maps, screen identifiers will be deleted and replaced, existing fields will be updated with their new positions and their references to other entities will be preserved. Fields that were previously imported, but no longer exist on the host will be deleted.

Note: Invalid entity names, such as names which include invalid characters such as "#" or begin with a digit, will be automatically corrected by omitting the invalid characters.

Note: The colors displayed in the screen image may sometimes be different than the colors that are displayed in the original host screen.

Maps can be imported either using the Import Host Screen Maps wizard or using a batch file (screen_import.bat in Windows and screen_import.sh in UNIX).

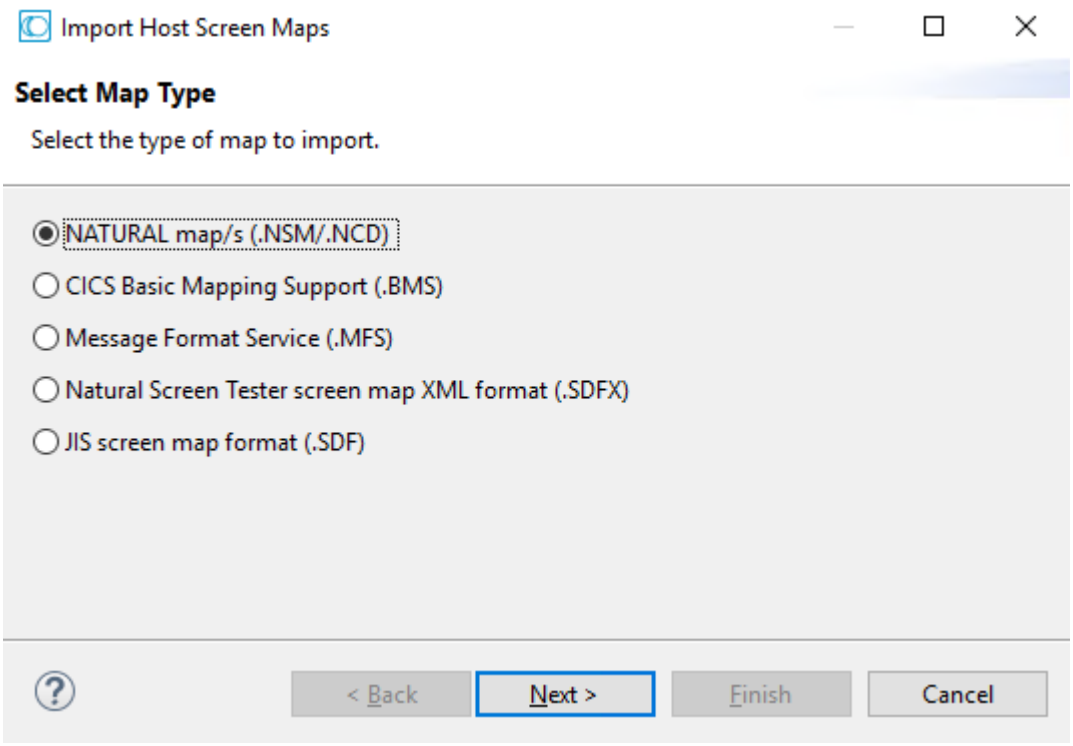
Importing Maps using the Import Host Screen Wizard

To import maps using the Import Host Screen Maps wizard

Ensure that the relevant Natural Screen Tester application and that the relevant map files are available.

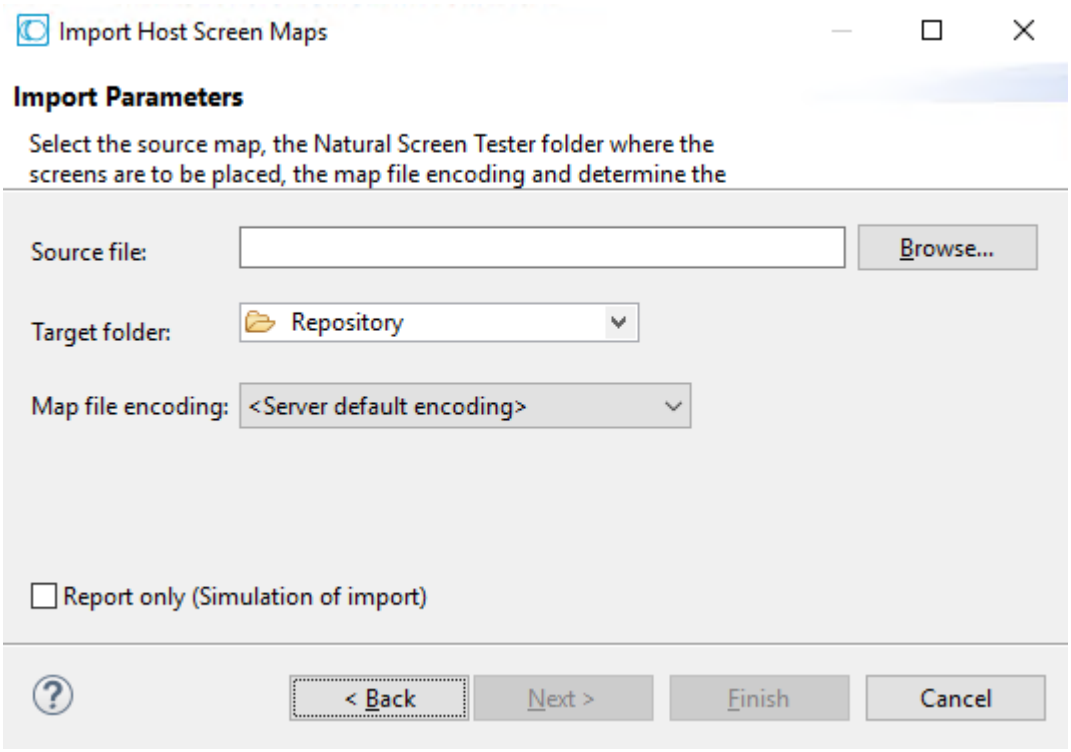
1 From the **Application** menu, select **Import Host Screen Maps...** or right-click on the repository node in the relevant application and then select **Import Host Screen Maps...**.

2 The *Import Host Screen Maps* wizard is displayed.



3 Select the type of map that is to be imported. Click **Next.**

4 The **Import Parameters** screen is displayed.



Select the source map file/s that you wish to import.

5 Select the Natural Screen Tester repository folder where the newly created screens are to be placed.

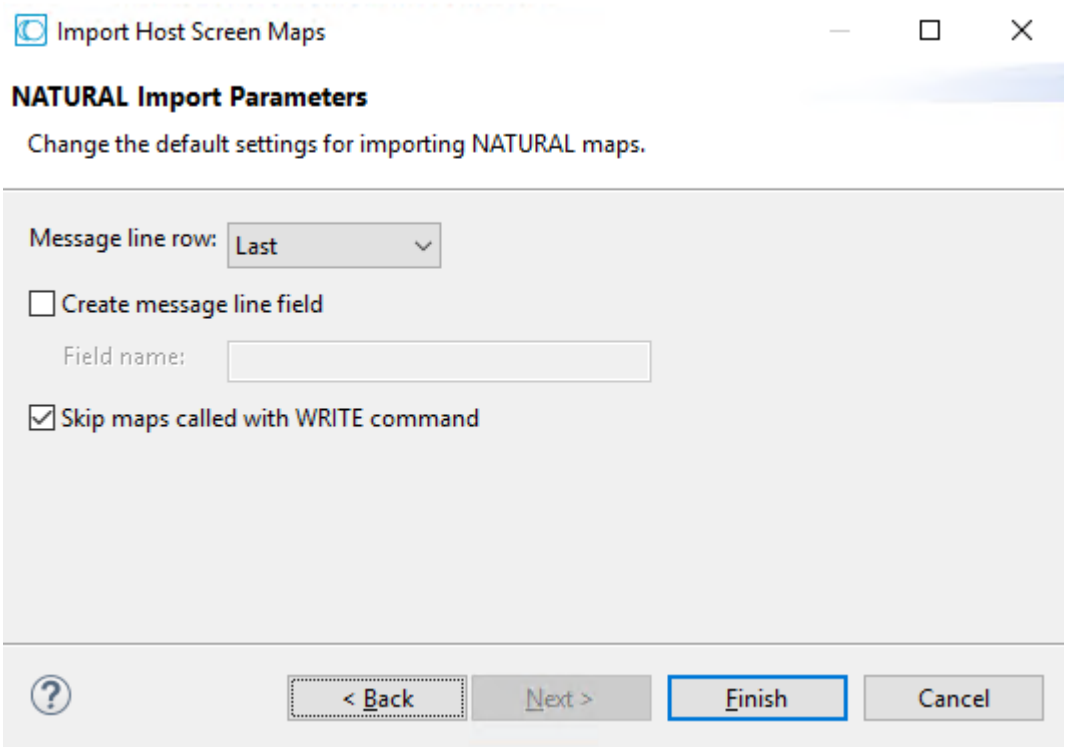
6 Select the map file encoding. By default, the server operating system default encoding is used.

Note: When importing maps from NaturalONE it is recommended to use UTF-8 encoding.

7 Determine whether you wish to simulate the import process, and create a report which details the import process or whether to actually import the maps and create the screens as well as the report.

8 If you selected to import a Natural map, you can click **Next** to configure the Natural Import Parameters, otherwise click **Finish** to complete the wizard.

9 Natural Import Parameters screen:



Message line row
Indicates where the error line is located - in the first line, one, two, three or four lines from the bottom or in the last line.

Create message line field
Selecting this will map a field in every screen, where a message can be displayed. When selecting this option, you are required to enter a name for the field.

Skip maps called with `WRITE` command
Select this in order not to generate maps that are called with the Natural `WRITE` command.

Click **Finish**.

The screens created appear in the directory you determined in the **Target folder** field. The names of the screens are identical to the map names.

The report is displayed in the Eclipse console and includes a list of the screens added as well as the fields and identifiers created/updated/deleted.

Importing Maps using a Batch File

To import screens via a batch file (using the command prompt window)

1 Open a command prompt window.

2 Change the current directory to the Natural Screen Tester home directory.

3 Type screen_import.bat/sh followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

Parameter	Description	Default
-a	Natural Screen Tester application name (Required parameter)	
-af	Natural Screen Tester target folder within the application repository.	Root folder
-mf	Message line field name (Natural maps only)	MessageLine
-k	Don't skip map with write command. (Natural maps only)	true (skip)
-e	Map file encoding: Includes all encodings which are supported by Java. **Note:** When importing maps from NaturalONE it is recommended to use UTF-8 encoding.	Server operating system default encoding
-f	File name, or directory name (when importing more than one file). Required parameter.	
-t	Map type. Possible values: "sdf", "sdfx", "natural", "bms", "mfs" (required parameter).	natural
-s	Server address	127.0.0.1
-m	Indicates where the error line is located: "first", "last", "lastm1" (last minus 1), "lastm2", "lastm3", "lastm4" (Natural maps only)	last
-p	Server port	3323
-u	Natural Screen Tester user name (Required parameter)	
-x	The file extension. All files from the given directory that have this extension will be loaded (when not specified, the default extension for the map type is used: ".sdf", ".sdfx", ".ncd" or ".nsm"(for Natural), ".bms", and ".mfs"). When a file name is provided (and not a directory), the specific file will be imported (this parameter is not required in this case).	
-w	Natural Screen Tester user password	Empty by default

The screens created appear in the directory you determined in the **Target folder** field. The names of the screens are identical to the map names.

The report is displayed in the command window and includes a list of the screens added as well as the fields and identifiers created/updated/deleted.

Configuring Screens

Define Steps in the Test Project Map

Refer to the ***Test Project Map Map*** for details. In this tab it is possible to manually define additional steps between screens that already exist in the test project map. The steps should be defined in the source screen and determine the destination screen of the step. It is possible to add a simple step between screens or to add an inner path step which executes a path. When you add a simple step, determine the target screen, the key to be sent, and the relevant inputs. When defining an inner path, select the path. Change the status of the step using the links (Approve, Set to Pending, Set to Not Approved). Once you save the new steps, you can check these navigation definitions in the Session View using the Test Project Map toolbar. Simply select the screen you want to navigate to and click on the **Go** icon to attempt to navigate to that screen.

Define Screen-based Tables

A screen based table represents a logical table based on data gathered from a single Natural Screen Tester screen. When creating a table you can configure:

- The header: A rectangle that contains all the column headers, and will later be used in the framework to hide this area.
- Filters: empty row, and duplicate row filters will filter the table according to the value defined in the Primary Key column.
- Table Columns: Includes the table definitions such as column details, occurrences, primary keys etc.

Once defined, you may view and design a table in a Natural Screen Tester Web application, or access it through a designated API in the Natural Screen Tester Base Object or from the Test Case Procedure Screen Mapper.

To create a table

1 Access the relevant screen and click on the Tables tab.

2 Click on the **Create a New Table** link.

- A default name (new_table), which you can edit will appear in the Table name field.
- The header definition is filled with values.

- All the multiple fields which are in the screen are added to the list of columns.

3 Ensure that you check and refine the table definitions as required:

- Header: To change the default header definitions, select the header area in the preview of the host screen and then click on the **Capture from screen** icon that is displayed next to the header row and column fields.
- Filter rows: There are two types of filters which can be defined: filtering of empty rows (rows in which the values of the primary key columns are empty) and filtering of duplicated rows (rows in which the values of the primary key columns are identical to those of a previous row). By default these checkboxes are not selected.
- Remove, add or edit table columns:
  - In the Name field enter a logical name to use when programming using the API or the framework, in the Caption field enter the name of the column as it will appear in the Web application, and in the Mapped to field select the host field to which the column will point.
  - Use the Move Up and Move Down options to define the order of the columns.
  - Define the table's primary key. A primary key is a key in the table that uniquely identifies each record. A table must always have one and only one primary key. The primary key can consist of one column or a combination of several columns, whose values give a unique key of each record. If no column is defined as the primary key of the table while creating the table, Natural Screen Tester automatically creates a numeric primary key for each record.

    To define a primary key, select the relevant column and click on the **Define Primary Key** link.

Associate Screen Groups

A screen that belongs to a group inherits all the field mappings that are mapped to the group.

To associate screen groups to a screen

1 Access the relevant screen and click on the Assigned Screen Groups tab. The list of Screen Groups currently associated with this screen are listed.

2 Click on Assign Screen Group to select additional screen groups to be associated with this screen.

Determine Screen Direction

This tab is only displayed in right-to-left applications and enables overriding application settings per a specific screen.

6 Procedures

A Natural Screen Tester procedure is a well-defined encapsulation of a complete process, and contains process input arguments, process output arguments and the process definition itself. A procedure group is a container of several procedures. The following procedure types are available:

- **Test Case Procedure**
  Encapsulates a process of navigation in host screens, collecting data or submitting data.
- **Flow Procedure**
  Encapsulates a complex process that can combine host sessions and other data sources: databases, host transactions (RPC), other web services.
- **Web Procedure**
  Encapsulates a process of navigating and selecting elements in the Web.
- **Program Procedure**
  Encapsulates a host transaction (in COBOL or RPG), invoked via RPC and not via the screens layer.
- **External Web Services**
  Encapsulates a web service that is external to Natural Screen Tester, invoked via SOAP.

See also Procedures in the *Natural Screen Tester Reference Guide*.

How to know which Type of Procedure to use?

The test case procedure is similar to the flow procedure. To know which one to use, ask yourself the following:

- Do you need to encapsulate a transaction from a single host session into a unit test (this is the most common case)? Use a test case procedure.
- Do you need to integrate data from different sources, such as database, several host sessions, external web services or RPC-based sessions? Use a flow procedure.
- Do you need to integrate data from the web? Use web procedures.

Test Case Procedures

Test case procedures on the one hand allow navigating between host screens and can be used for navigation, data gathering or transaction execution on a single host session. On the other hand, it is a procedure, and can therefore be integrated into a unit test, using a generated procedure client or called by another flow procedure. Procedures, provide a rich variety of logical nodes and expressions that make the host navigation very flexible without needing client-side code. This section covers the following topics:

- Test Case Procedure Methodologies
- Failure Log

See also Test Case Editor | Nodes | Expressions under *Procedures* in the *Natural Screen Tester Reference Guide*.

Creating a Test Case Procedure

A test case can be created either via the Session or the Repository. When creating the test case procedure in the session, you use the **Test Case** toolbar to record the test case and then you can edit the test case in the dialog box. When creating the test case procedure via the Repository, you need to manually define the test case procedure using the available nodes. It is recommended to create the test case procedure via the session and then fine-tune and handle errors.

To create a test case procedure via the Session view

1 Navigate to the first screen from which you want the recording to commence.

2 Click **Record** on the Test Case Procedure Toolbar.

3 Navigate to the various screens to record the relevant test case.

4 Click the **Mark possible input** icon to mark all the predefined potential input fields in purple. Click on one of the marked fields you want to include as input for your test case. The field will be marked orange.

5 Click the **Mark possible output** icon to mark all the predefined potential fields in purple. Click on a field to define it as an assertion for your test case. The field will be marked blue.

6 If you want to record a step that repeats again and again until a certain condition is reached, click the **Loop** icon, and follow the instructions in the wizard. This kind of definition can be applicable for collecting data of a host table.

7 To end a recording click the **Stop recording** icon. Enter a name for the test case.

8 Click **Finish**. The editor will open where you can manually edit and add nodes.

Refer to the **Procedures** section for more information about Procedures.

To create a test case procedure from Repository

1 Create a new test case procedure via **File > New>Other... > Software AG > Entities > Procedure**.

2 The New Test case procedure wizard is displayed. Enter a name for the Test case procedure.

3 Define inputs and outputs in the bottom panel of the screen.

4 Define nodes. It is recommended to always create a Try/Catch node in order to catch exceptions within the procedure.

Note: An image of the Procedure tree can be saved as a PNG file for later reference, by right-clicking on the Procedure root node, and selecting **Save as Image...**.

5 Save the procedure.

6 Right-click on the procedure in the Explorer and select **Run** to run and test the procedure, or select **Debug** to open the **Debug** window and debug the procedure.

Test Case Procedure Methodologies

This section provides tips and methodologies that may be helpful when incorporating test case procedures in your test project.

- Retrieving Field Attributes and Host Screen Properties
- Searching for a Specific Value in a Multiple Field

Retrieving Field Attributes and Host Screen Properties

When executing test case procedures, you may require retrieving field attributes or screen properties. One example for this may be collecting all positive values from a numeric list where all positive numbers are colored green and negative numbers are colored red. Another example is when submitting certain values to the host a modal window may appear which would require us to perform a slightly different navigation than usual.

The required field attributes and screen properties can be retrieved using the "Field Attributes" and "Screen Properties" expressions.

To retrieve field attributes and screen properties

1 Right-click on the Expressions node in the mapper area, or if you are creating a condition, click on the Expr link and select Screen Data>Field Attributes/Screen Properties.

2 When working in a Mapper area, map the attribute or property to the appropriate variable.

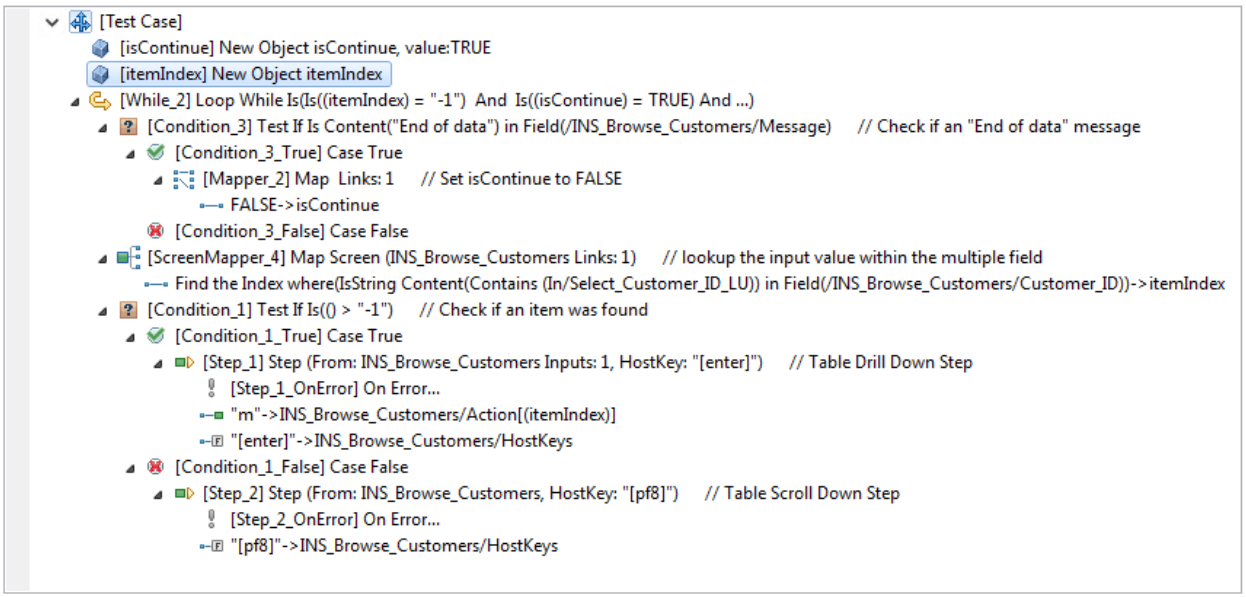
From within a condition, select the relevant attribute or property.

Searching for a Specific Value in a Multiple Field

When you need to search for a specific value within a host table that spans over several screens, what you actually need to do is to search for a specific value within a multiple field. A possible scenario may be, for example: A web page that contains a table with data collected from several screens. When clicking on a specific line, you would like to drill down and see additional details. The problem is having collected several screens on one web page, the selected line doesn't appear on the current host screen anymore - you need to look for it. In your Natural Screen Tester session, you need to scroll through the table again, and look for the particular data or a unique ID from

the selected line on the web page. In order to do this, your test case procedure requires two loops: One loop, to scroll through the table and another loop, to search each line for the selected value.

This can be implemented using the Find Field Index function. This function allows you to check a single condition within a multiple field, testing each field and returning the index of the first field that matches the condition.



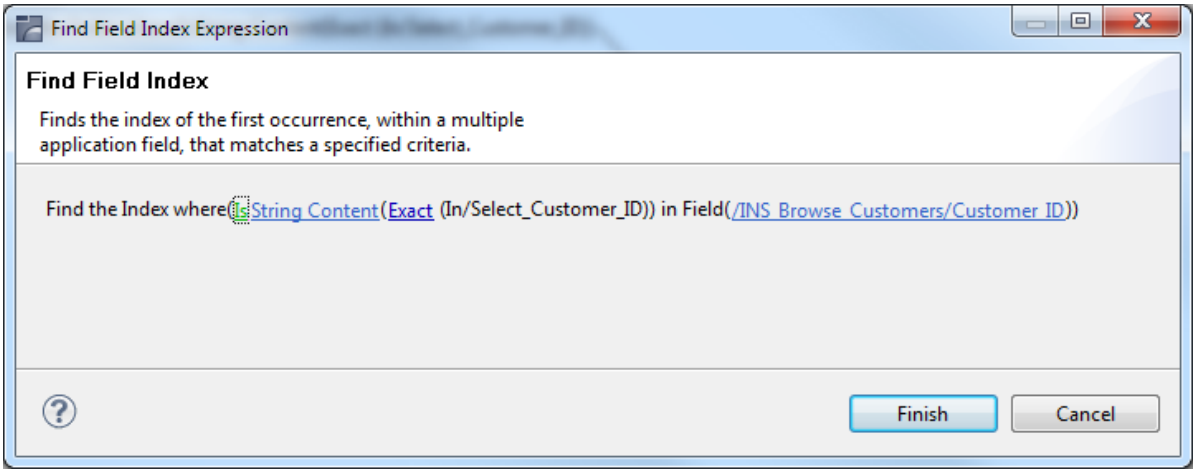
The loop will continue iterating through the table as long as both of these conditions are met:

- The `itemIndex` parameter is set to -1. This means no match to the search value has been found yet.
- The `isContinue` parameter is set to True. This parameter will be set to false when the end of the table is reached. This will prevent creating an infinite loop when no match is found.

The loops are defined as follows (see screen capture above)

1 The [Condition_3] condition checks whether you have reached the end of the table. When the end of the table is reached, IsContinue is set to False.

2 Configure the Find Field Index parameters:



- Attribute type: Select an attribute type, in this case String Content. Other possible attributes include: protected, hidden, intensified, reversed video, background color, foreground color, blinking or underlined.
- Select comparison type: You can check whether a string value is part of the field content or you can look for an exact match. Numeric values can be compared (less than, equals, greater than, etc) to the field's numeric value.
- Select the screen and multiple field whose content you wish to check.

Using Find Field Index, [ScreenMapper_4] searches for the value inside an entire table column (multiple field) and maps the result (the matched index) to the itemIndex parameter previously created. When no result is found, the mapping is not performed.

3 The [Condition_1] condition checks whether a match was found by checking if itemIndex is bigger than -1 (which means a result has been found). If a match was found, use itemIndex to implement a drill down action. If a match was not found, perform a scroll down action to continue the search.

Failure Log

Natural Screen Tester enables generating a log that includes debug data regarding procedures that fail in runtime. The log includes a snapshot in ASCII characters of the screen that caused the failure.

To log data regarding procedures that fail in runtime

1 In the *<nsr_home>/config/log/nsrlog_config.xml* file under the com.sabratec.util.flow.error_tracking category tag, change the <level value="off"/> line to be commented and uncomment the <!-level value="debug"/--> line.

2 Restart the Natural Screen Tester Server.

The log will be created in the <*nsr_home*>/*log* directory, and the file names will have the following format: debugging_error_in_%I_%t.log, %I being identifying information about the user and procedure name, and %t being the timestamp. The location and name of the file can be changed in the nsrlog_config.xml file.

Flow Procedures

Flow Procedures are able to execute other types of Natural Screen Tester procedures in addition to being able to connect to databases and perform operations such as Select, Execute, Commit and Rollback. This allows retrieving data from sources other than host screens.

Creating a Flow Procedure

When creating a new flow procedure, it is important to first define your goals and what tools you need in order to reach these goals. These tools may include test cases or Data Structures (that represents inputs or outputs), programs or a database.

To create a Flow Procedure

1 Create a new Flow Procedure via File>New>Other...>Software AG >Entities>Procedure.

2 Define inputs and outputs in the bottom panel of the screen.

3 Define nodes. It is recommended to always create a Try/Catch node in order to catch exceptions within the procedure.

Note: An image of the Procedure tree can be saved as a PNG file for later reference, by right-clicking on the Procedure root node, and selecting **Save as Image...**.

4 Save the procedure.

5 Right-click on the procedure in the Explorer and select **Run** to run and test the procedure, or select **Debug** to open the **Debug** window and debug the procedure.

Refer to the **Procedures** section for more information about Procedures.

Web Procedures

The Web Page Integration feature enables simulating web browser activity and exposing it as a standard web service or integrating it with other Natural Screen Tester procedures. Web browser activity is simulated within Natural Screen Tester by recording and entering/capturing relevant web content using Web browser based tools. The web content is used by the new type of Procedure - the Web Procedure. This Procedure is specifically designed to enable integrating the User Interface (UI) of Web Pages within Natural Screen Tester, taking advantage of the flexible and dynamic capabilities incorporated within Natural Screen Tester Procedure infrastructure. The Web procedure can be exposed as a service in the same way the existing Natural Screen Tester procedures such as Test Case and Flow Procedures are exposed.

This section covers the following topics:

See also Web Procedures in the *Designing and Developing a Test Case* documentation.

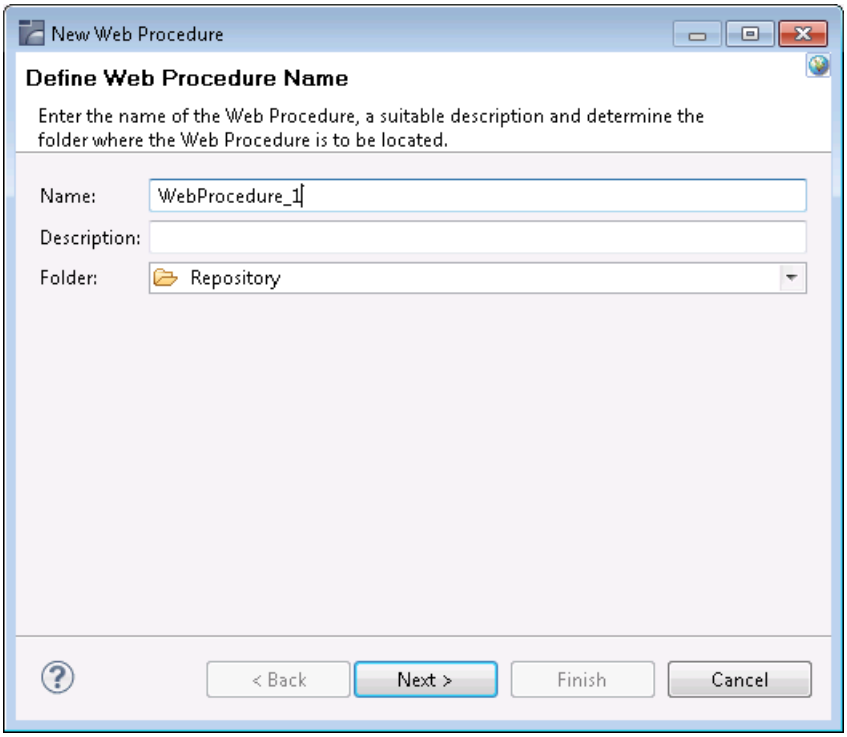
Creating a Web Procedure

As part of the process of creating the Web Procedure you will record and capture the elements to be used in the Web Procedure. Before creating a new Web Procedure, it is important to first decide and plan what the scenario is that you would like to record, and also determine which elements you would like to capture and later use in the Web Procedure.

To create a Web Procedure and capture the elements from within your browser session

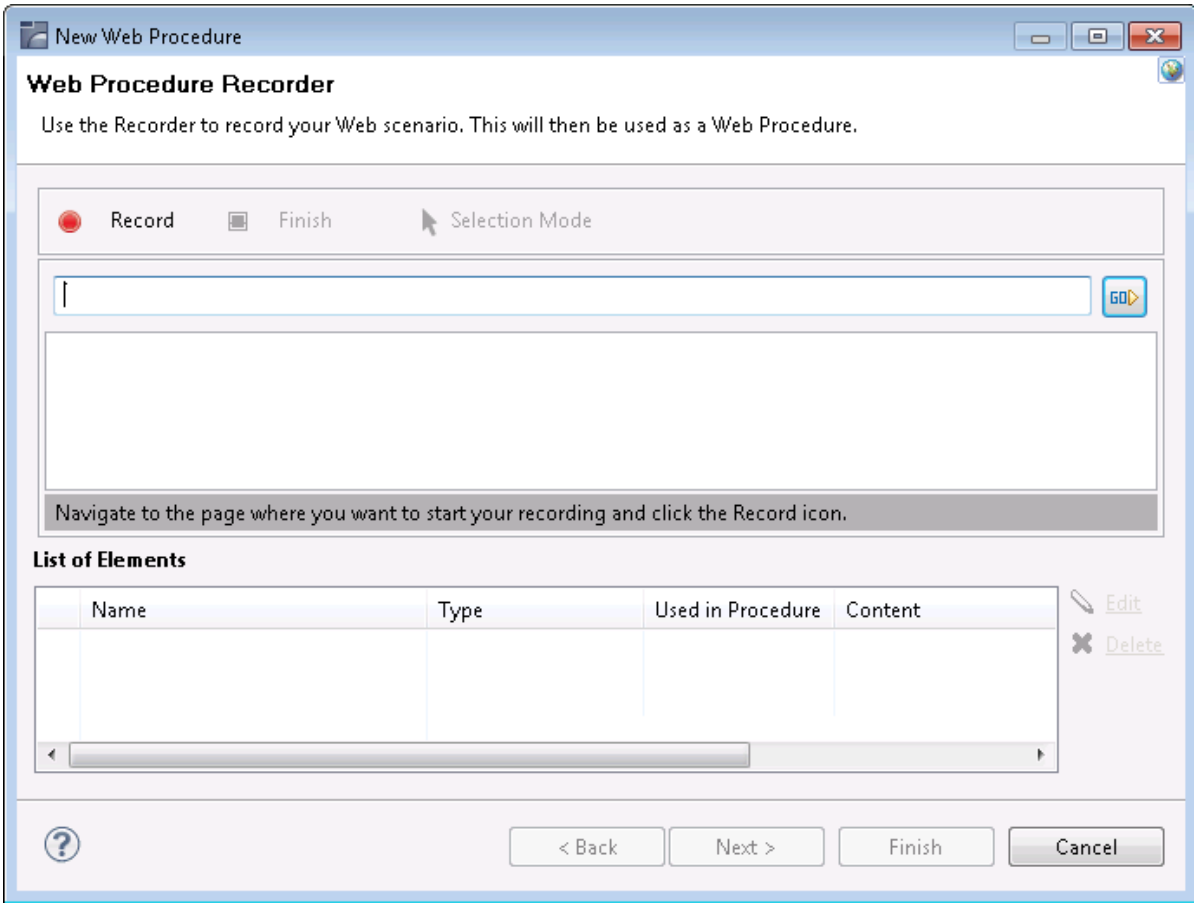
1 Select the relevant application or the Root node of the application.

2 In the **File** menu, select **New>Entity>Web Procedure…**. The *Web Procedure wizard* is displayed.



3 Enter a name for the Web Procedure, a suitable description and determine the folder where the procedure is to be located.

4 Click **Next**. The *Web Procedure Recorder* is displayed. The Recorder is comprised of a customized toolbar, an address bar, the browser area where the Web page content is displayed and a summary of the list of captured elements.



5 Enter the name of the URL and click the **GO** button or press **Enter** to navigate to the URL where you would like to begin recording.

6 **Recording:**

Click on the **Record** button. When recording there are two possible modes: **Navigation mode** and **Selection mode**. Use the Navigate mode to navigate through the web pages as in any Web browser, and record inputs and actions such as clicking on buttons or links, entering data. Every action you do is recorded and can later be edited in the Web Procedure editor.Use the Selection mode to "freeze" the page, so you can capture outputs and inputs, single elements or a list of similar elements. When in the Selection mode, you can no longer navigate to other pages until you return to Navigation mode.

7 **Selecting and Capturing Elements:**

1. Entering the Selection Mode:

   When you want to select and capture elements on a specific page, click on the **Selection Mode** button 

.

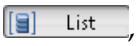
2. Understanding the highlight colors:

Light blue: When you move the mouse over different elements within the browser, the color of the element changes to light blue to indicate that this is the element which will be selected if you now click on this element.

Orange: Once you click on an element, it is highlighted in orange.

Dark blue: Captured elements are colored in dark blue.

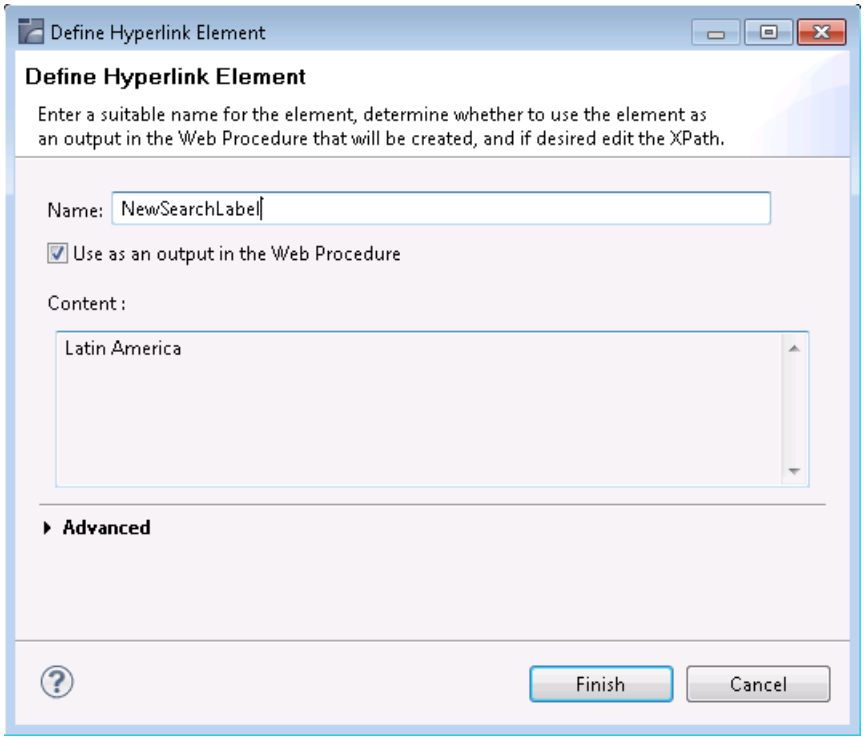
3. Selecting single elements and lists of elements:

   Within a page, you can capture either a single element or a list of elements. When entering the Selection mode, by default, the Recorder is set to capture single elements, by clicking once on an element. When you would like to capture a list of elements, click on the List button on the toolbar 

, then click on one element within your list and then on a second element. Before clicking on the second element, hovering over an element highlights in a light blue color the complete list of elements that will be captured once you click on the second element. Ensure that the Single button is pressed in when you wish to select single elements, and that the List button is pressed in when you wish to select a list of elements.

4. Capturing an element:

   Click on a required element. The *Define Element* dialog box is displayed and the element color within the browser changes to orange.



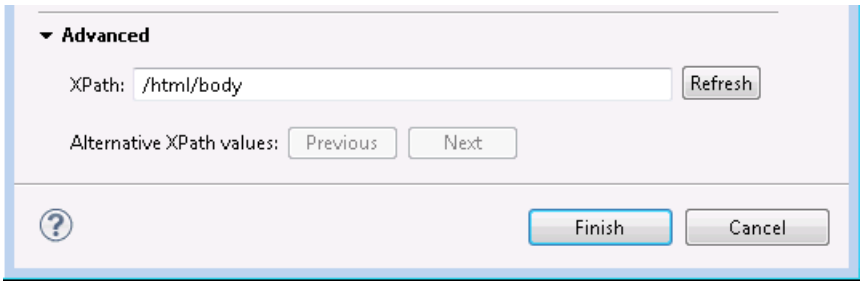
This dialog box may slightly vary according to the type of element captured such as text fields, text areas, passwords, radio buttons, check boxes and drop-down lists, buttons, and hyperlinks.

5. Defining the captured element:

   Provide a suitable name for the element. By default if the HTML element has a name, then this is used by default. If it does not have a name, then the ID is used, and if it does not have an ID then the default is <element type>_<number> (for images, the alt attribute is used). Determine whether this element is to be used as an output/input (depending on the element type) in the procedure (by default the element is selected to be used as the input/output of the procedure, the only exception being when defining a list of elements, when for some types, this check box is not selected by default). The Content area provides a textual display of the captured element (this area does not appear in the Button element).

6. Understanding the element's XPath:

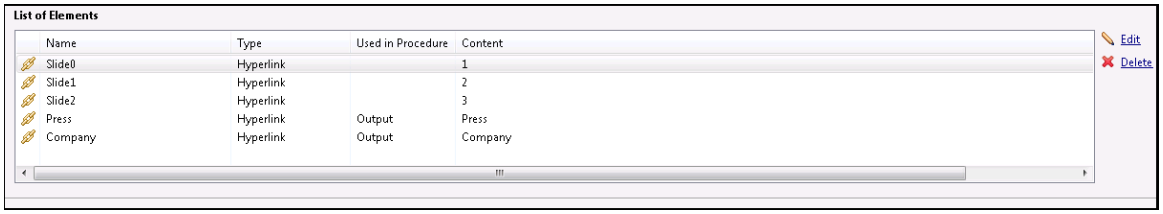
   We use the XPath, the XML Path Language, to locate the element within the Web Page. The wizard tries to suggest the most suitable XPath and in most cases you are not required to make changes. The Advanced section enables you to view the current XPath and if desired change it.



   This may be necessary, for example, to ensure a more robust XPath or if the element that is highlighted in the browser is not the desired element. You can scroll through a list of suggested alternative XPaths (using the Previous and Next buttons) or edit and change the XPath yourself. When making changes to the XPath, it is recommended to click on the Refresh button to ensure that the XPath is valid (for more information regarding XPaths, refer to What is an XPath?). This may also change the contents of the Content area.

7. Reviewing the list of captured elements:

   Once you have completed defining the captured element, the element will appear in the **List of Elements** table beneath the browser area. This list includes all the elements which have been selected on the page currently displayed. Captured elements are hihglighted in dark blue. When you select an element within the List of Elements table, the element will be highlighted in orange.



Repeat steps a-f to select and capture additional elements.

8 **To edit a captured element:**

Either click on the relevant element within the Element List table and then click on Edit, or click on the element within the browser. The *Edit Element* dialog box is displayed and you are able to make the required changes.

9 **To delete a captured element:**

Click on the relevant element within the Element List table and then click on Delete.

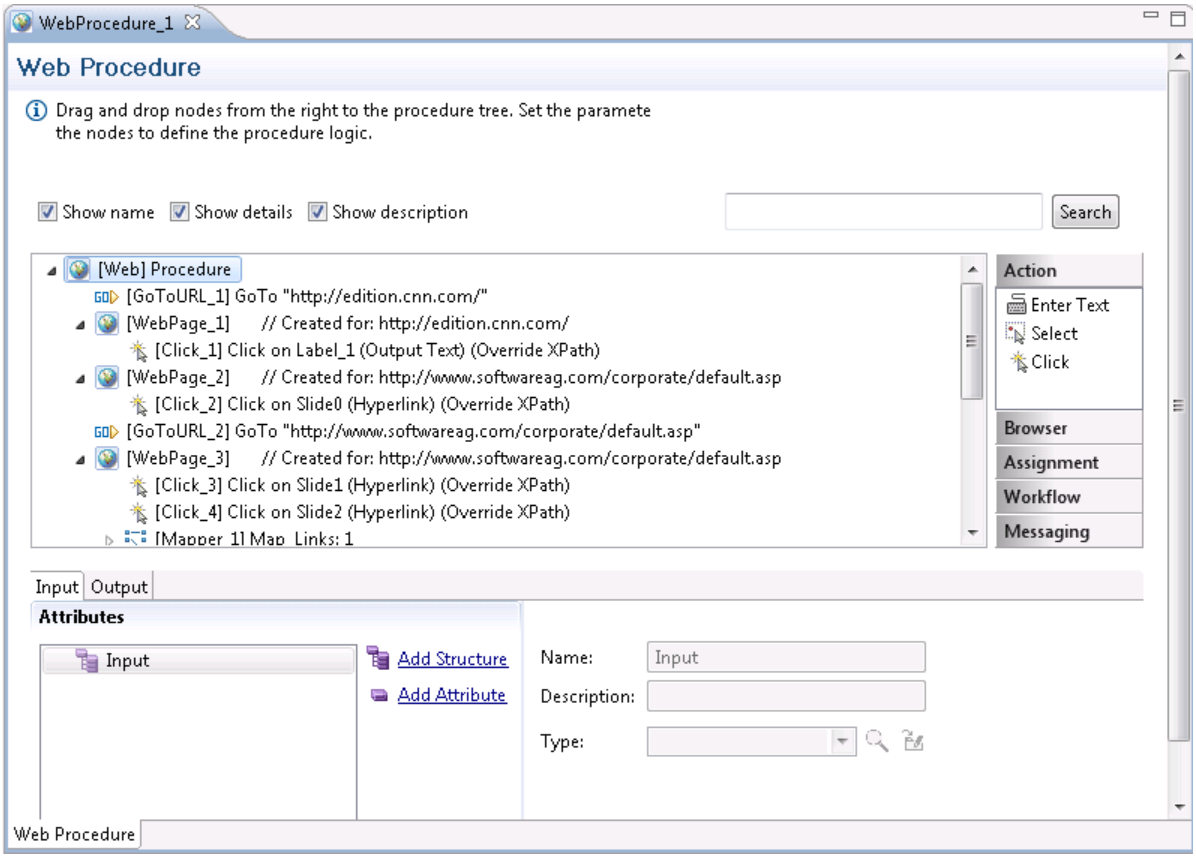
10 **To exit the Selection Mode and continue navigating within the browser**

Click on the Selection Mode button.

11 **To navigate to a different page:**

Either enter the new address in the address field or click on the desired link.

12 To complete the recording and save the entity, click on the **Finish** button. The entity is opened for editing in the Editor area.



Running the Web Procedure

The Web Procedure can be used to expose web services (Procedure Groups are required for this) or executed from within another procedure. To check that the procedure functions as expected, after you have completed **recording the Web Procedure** you can run the procedure from within the Designer.

To run the procedure, within the Editor, right-click on the relevant procedure and select **Run**. The execution flow is displayed in the Console and the main steps are written to the server log. Sometimes, the procedure requires entering values for input parameters. In such a case a pop-up appears.

For example:

```
<In>

<password></password>

<reconnect>true</reconnect>

<user>demo5</user>
```

```
</In>
```

When running the procedure for the first time, the values displayed within the tags are default values that were recorded when creating the Web Procedure. When running the procedure additional times, the last entered values are used.

Sometimes the procedure may fail to run. The Troubleshooting section in the in the *Natural Screen Tester Reference Guide* may provide some useful tips.

Editing the Web Procedure

- Adding a GoTo Node, to Navigate to a URL
- Adding Conditional Logic to the Web Procedure
- Adding a New Element to an Existing Web Page
- Handling Elements whose XPath has Changed
- Working with Dynamic XPaths
- Checking whether an Element Exists within a Web Page
- Retrieving an Element's Value

When completing the process of creating a new Web Procedure, it is opened in the Editor. As with other entities, it is possible to open the Web Procedure and edit it at a later stage. In the Editor you can refine the Web procedure, add logic, add, edit or delete elements and maintain the Web Page structure.

Adding a GoTo Node, to Navigate to a URL

After the procedure is created, the original site may have changed and another page may have been added making your recording out of date. It is also possible that you simply want to add navigation to an additional URL. This can be done by adding a GOTO URL node to the procedure, and defining there the relevant URL. Refer to ***Working with Procedure Nodes*** in this section and Web Procedure Nodes in the *Natural Screen Tester Reference Guide*.

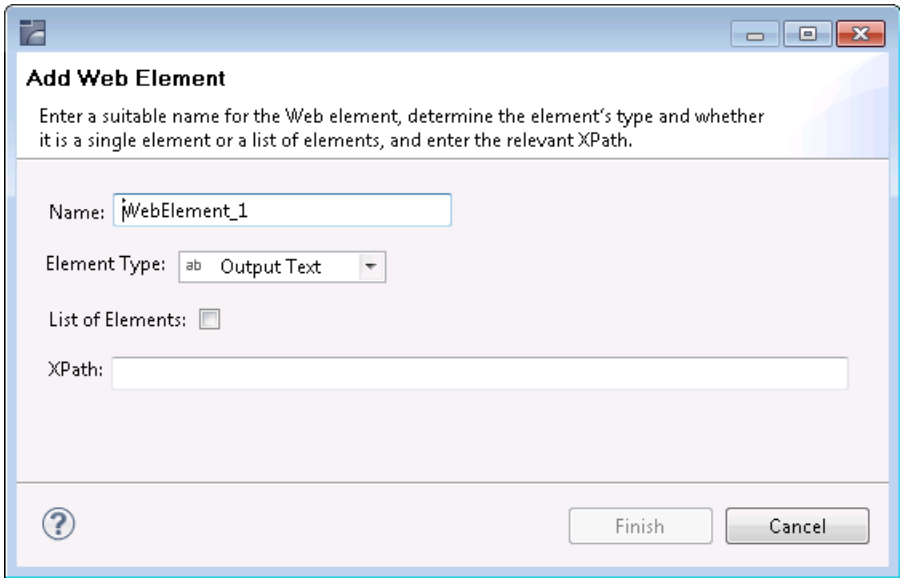
Adding Conditional Logic to the Web Procedure

When recording the procedure, it is not possible to record a conditional scenario such as if a specific field has a specific value then the procedure should then navigate to a specific web page. Using the procedure nodes, actions and expressions to determine the condition, and then the GOTO URL navigation node to navigate to a different URL, you can add such logic into your procedure. Refer to **Working with Procedure Nodes** in this section, and General Nodes and Web Procedure Nodes in the *Natural Screen Tester Reference Guide*.

Adding a New Element to an Existing Web Page

You may find that you need to add an additional element to the Web Procedure. This can be for a number of reasons such as you may realize that you did not capture all the elements that you require or that the Web Page you originally recorded has changed and now you would like to add new elements that did not previously exist.

To add an element, select a Web Page node in the Procedure tree, and in the node details area, where the structure of the Web page and its elements are displayed, right click on the root node of the Web page and select **Add Web Element**. The *Add Web Element* dialog box is displayed.



Enter a name for the element, select the relevant type and enter the XPath. Click **Finish**. The new element will be added to the Web Page schema and can then be used in child nodes.

Handling Elements whose XPath has Changed

A Web site which undergoes changes can cause the XPath of an element to change. As you would still like the Web procedure that you originally defined to be valid, you need to correct the XPath of the element to suit the new one:

1. **Add a new element** to the Web page.
2. In all the Mappers and actions, change the references to refer to the new element instead of to the old, outdated, element.
3. Delete the old element.

Working with Dynamic XPaths

A Web element that is dependent on a user's action or input in runtime, requires an XPath that can change dynamically to suit this. In such cases, after recording the Web procedure and capturing the relevant element, change the XPath (in the procedure editor) by clicking on the Override option where this element is used (in one of the following Web Procedure Nodes: Enter Text action node, Select action node or Click action node) and defining the XPath using logic that will provide the required flexibility in runtime.

When mapping such an element as the output of the procedure, you may require using the Web Element Content expression.

Checking whether an Element Exists within a Web Page

It is possible to check whether an element exists within a specific web page using the Test Web Element Exists expression within the Web Page context.

To check whether a Web element exists

1 Select Test Web Element Exists from the list of Boolean expressions.

2 Click Is/Is not to define the condition.

3 Click the XPath to open the *Free Text* dialog box and enter the XPath (you can copy-paste the XPath of an existing element by right-clicking an element and selecting Copy XPath to Clipboard).

Retrieving an Element's Value

As part of the Web procedure's logic, you may want to use or check the value of a certain element. This is done using the Web Element Content expression.

To retrieve the value of an element

1 Select Web Helper>Web Element Content expression.

2 Select the XPath from which to retrieve the content.

Defining Procedure Inputs and Outputs

Procedure inputs and outputs are defined when selecting the procedure root node. They may be used in any node of the procedure.

To define inputs and outputs

1 Within the Procedure root node of a procedure which you created, select the Input or Output tab as required.

2 Add attributes. See Procedure Input and Output Attribute Types in the Natural Screen Tester Reference Guide and **Data Structures**.

Working with Procedure Nodes

Nodes are used in Procedures and determine how the procedure will behave. Procedures consist of a number of nodes. These nodes are defined by the user to perform logical operations and are arranged in the order that these operations must be executed.

Note: It is recommended to maximize the Editor screen when defining a procedure.

The nodes in the procedure can display different information regarding the nodes by clicking on the different check boxes (**Show name, Show details, Show description**).

To search within nodes

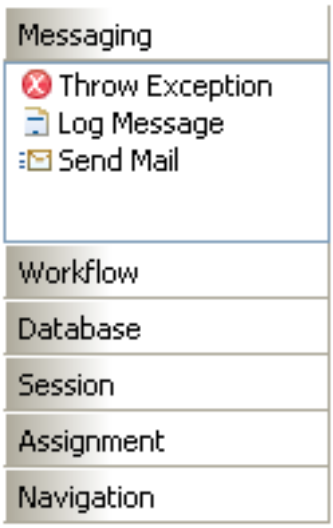
- The Search field enables searching for a key word within the procedure nodes. The search mechanism is not case sensitive. The text is searched for within the description, details or name of the nodes. The nodes which contain the text will be selected and expanded in the tree, while other nodes will be collapsed.

To save the procedure tree as an image

- Right-click on the Procedure root node and selecting **Save as Image...** to save an image of the Procedure tree as a PNG file.

To add a node

1 In the Procedure editor, select a node from the list of nodes.



2 Drag and drop the node to the desired location within the procedure.

3 Edit the node.

To delete a node

1 Select the node within the Procedure.

2 Right-click on the node and select **Delete**.

Moving a node within the Procedure

- A node can be moved within a procedure by simply dragging and dropping it in the desired location.

To add/edit an expression

- Expressions are used in procedure nodes. To edit an expression, right-click and select from the list.

Using the Mapper to Map Source Elements to Target Elements

Introduction

The Mapper tool enables mapping source elements to target output elements. The left-panel contains the entire source data elements that are available at the current flow scope depending on the node type (these element may also be expressions). The right-panel contains the potential target elements depending on the flow scope and the node type. Using the drag-and-drop operation, it is possible to define that when the Mapper is executed, the value/s in the source element will be copied to the target element. These definitions are indicated by lines, linking between the source element and the target element, and also are listed under the relevant node in the procedure nodes tree.

Note: When mapping a source element (e.g inputAttribute in the screen shot below) to a target element (e.g helpAttribute), and in the same mapper mapping the same element (helpAttribute) to another target element (outputAttribute), as the mapper does not enable defining the order in which the elements will be mapped, this can cause a conflict. Therefore it is recommended to perform such a mapping using two separate mappers.

The procedure nodes that enable using the Mapper tool:

- Create Mappings
- Group Arrays (the target will include objects relevant to the specific Group Array node)
- Execute Procedure (lists only the inputs relevant to the specific procedure)
- Flow Procedure nodes:
  - Create Emulation Session (lists only the inputs relevant to the specific procedure)
  - Create DbSession (lists only the inputs relevant to the specific procedure)
  - DbSelect (lists the dynamic inputs. Refer to Creating a dbSelect node)
  - DbExecute- relevant for Flow Procedures only.
  - RollBack (lists only the inputs relevant to the specific procedure)
  - Commit (lists only the inputs relevant to the specific procedure)
  - Create RPC Session- relevant to Flow Procedures only.
  - End Session (lists only the inputs relevant to the specific procedure)
- Test case nodes:

- Step (input to source screen\screen group, output from target screen\screen group)
- Test case (lists only the inputs relevant to the specific test case)
- Screen Mapper

- The expressions that enable using the Mapper tool:
  - Free Text (using tokens)
  - Execute Procedure
  - Host Keys (using tokens)

Mapping a Simple Type Element to a Simple Type Element

Drag the element in the source frame of the mapper to the relevant element in the target frame. When the Mapping is executed, the source element will be copied to the target element.

Mapping an Array Type Element to an Array Type Element

When mapping an array to an array, it is possible to copy all the source data in an array element to the target element, or it is possible to copy only some of the data to a certain place in the target array.

To copy an array element to an array element

1. Drag the element in the source frame of the mapper to the relevant element in the target frame. When the Mapper is executed, the source element will be copied to the target element. A line will be displayed between the two, indicating that the value from the source element will be placed in the target element.
2. By default, the data is appended to the existing data. This is indicated by a + sign which appears on the line connecting between the source element to the target element. If the + sign does not appear, double-click on the middle area of the mapper (between the area of the source data and the area of the target data) to display the *Link Properties* dialog box. Select the Append check box and click OK.

To copy an array element or part of an array element using an index

1. To define that all or some of the source array values will be placed in a particular place in the target array, drag a line from the element in the source frame of the mapper to the element in the target frame. A line will be displayed between the two, indicating that the value from the source element will be placed in the target element.
2. Right-click on the middle area of the line to display the Link Properties dialog box.
3. To overwrite existing data, ensure that the Append check box is not selected.
4. To copy the data which is from a certain index in the array, right-click and define the Source Index.

5 To determine that the data will be placed in the middle of the array starting from a certain index, right-click and define the Target Index.

6 When you do not need the entire source array but only a specific number of values, define the number of values in the Count field.

7 Click OK to close the dialog box.

Caution: Ensure, that in the Link Properties dialog box, either the Append check box is selected or the Source or/and Target Index have values, otherwise the procedure may not function as you intended and may also cause performance problems.

Mapping a Simple Type Element to an Array Type Element

The simple element can be added to the array in two ways: by appending the value to the existing array or by placing the value in a specific place in the array.

To append a simple type element to an array

1 Drag the element in the source frame of the mapper to the relevant element in the target frame. When the Mapper is executed, the source element will be copied to the target element. A line will be displayed between the two, indicating that the value from the source element will be placed in the target element.

2 Right-click on the middle area of the mapper (between the area of the source data and the area of the target data) to display the Link Properties dialog box. In the Target Index field right-click on <exp> to set the index value where the element will be placed.

3 Click OK to close the dialog box.

Caution: Ensure that if you remove the select from the Append check box, you define the Target Index, otherwise the value of the simple type element will be placed in the [0] index.

Mapping an Array Type Element to a Simple Type Element

To map a value from an array type element to a simple type element, it is necessary to define the index of the relevant value in the array. If you do not define this index, the [0] index will be used.

To place an array type element value in a simple type element

1 Drag the element in the source frame of the mapper to the relevant element in the target frame. When the Mapper is executed, the source element will be copied to the target element. A line will be displayed between the two, indicating that the value from the source element will be placed in the target element.

2 Right-click on the middle area of the mapper (between the area of the source data and the area of the target data) to display the Link Properties dialog box. In the Source Index field right-click on <exp> to set a value for the index.

3 Click OK to close the dialog box.

Mapping a Structure Type Element to a Structure Type Element

The process of mapping the attributes of a structured element to the attributes of a structured element is similar to mapping simple and array elements (refer to Mapping Source Elements to Target Elements). When mapping an element in a structured element to an element in a structured element, map the relevant internal elements and do not map the structured root element.

Caution: Mapping the structure root to the structure root will only map elements of the source structure to the elements of the target structure it they have the same name. Always verify that the correct links are made when mapping different types of structures.

Mapping a Number of Simple Elements to a Structure Type Element

It is possible to map a number of simple elements to a structure.

To map a number of simple elements to a data structure type element

- Select more than one simple element in the source frame of the mapper using CTRL or SHIFT. Drag the selected elements to the relevant data structure in the target frame. When the Mapper is executed, the source element will be copied to the target element within the data structure. A line will be displayed between the two, indicating that the value from the source element will be placed in the target element.

Program Procedures

To understand the Program feature in Natural Screen Tester, we first need to overview the concept of Remote Procedure Call (RPC). RPC is a description for APIs that permit distributed applications to interact with each other across the network. As part of its host integration capabilities, the Natural Screen Tester server implements RPC calls to execute host programs and routines, thus adding another type of interaction with hosts in addition to screens and screen data.

This Wizard guides you through a sequence of instructions to import a program object and corresponding field elements. Currently, the Wizard supports the import of program objects for COBOL and RPG programs for AS/400 hosts.

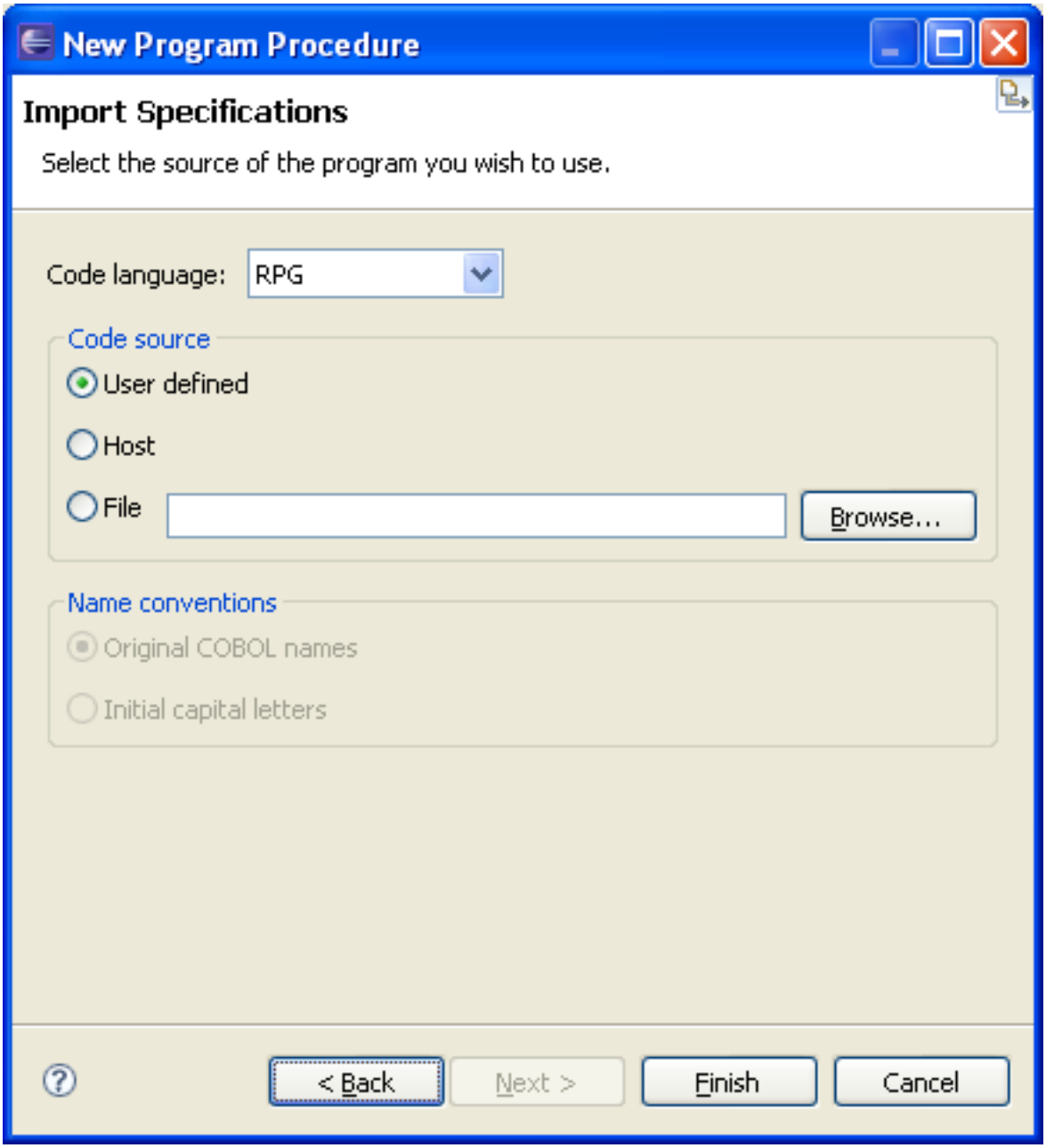
To create a new Program Procedure

1 Select the relevant application or the Root node of the application.

2 In the Explorer tool bar, click on the arrow to the right of the **Create new entity** icon and select **New Program Procedure**. The New Program Procedure wizard is displayed.

3 Enter a name and description (optional) and click **Next**.

4 The *Import Specifications* screen is displayed.



5 From the Code language drop-down list select COBOL or RPG.

6 Specify the code source:

- User defined: Enables you to define a new program.
- Host: Enables you to import a program from a host.

- File: Enables you to import a program from a file. Specify the code source file.

7 For more convenient use, Natural Screen Tester can rename the parameters automatically for you. Advanced users (AS/400 hosts only): When RPG is selected, Natural Screen Tester can retrieve the Program's source file directly from the AS/400 machine. Click the Code retriever settings button to display the Program - Advanced dialog. Specify the Host Library, File, Member and Line Length (default is 100). Enter the host's address or select a previously defined host from the drop-down list. Enter the Host's User name and Password, otherwise the default values as set in the host configuration are used. Click **OK**. Click **Next.**

Defining Program Procedure Inputs and Outputs

- Double-click on the relevant Program Procedure from within the Repository. The Editor area displays the Procedure Program. Add/edit parameters. Refer to the parameter details.

External Web Services

The External Web Service feature enables creating procedures based on a WSDL which describes a Web service within the organization. An External Web Service Procedure can be invoked, either by Flow, Test Case or Web Procedures.

Refer to *External Web Services* in the *Natural Screen Tester Reference Guide*.

Current limitations:

Natural Screen Tester supports only DOC/Literal (wrapped and bare) WSDLs.

Natural Screen Tester supports the following XSD types (refer to http://www.w3.org/2001/XMLS-chema) and any ComplexType which uses them:

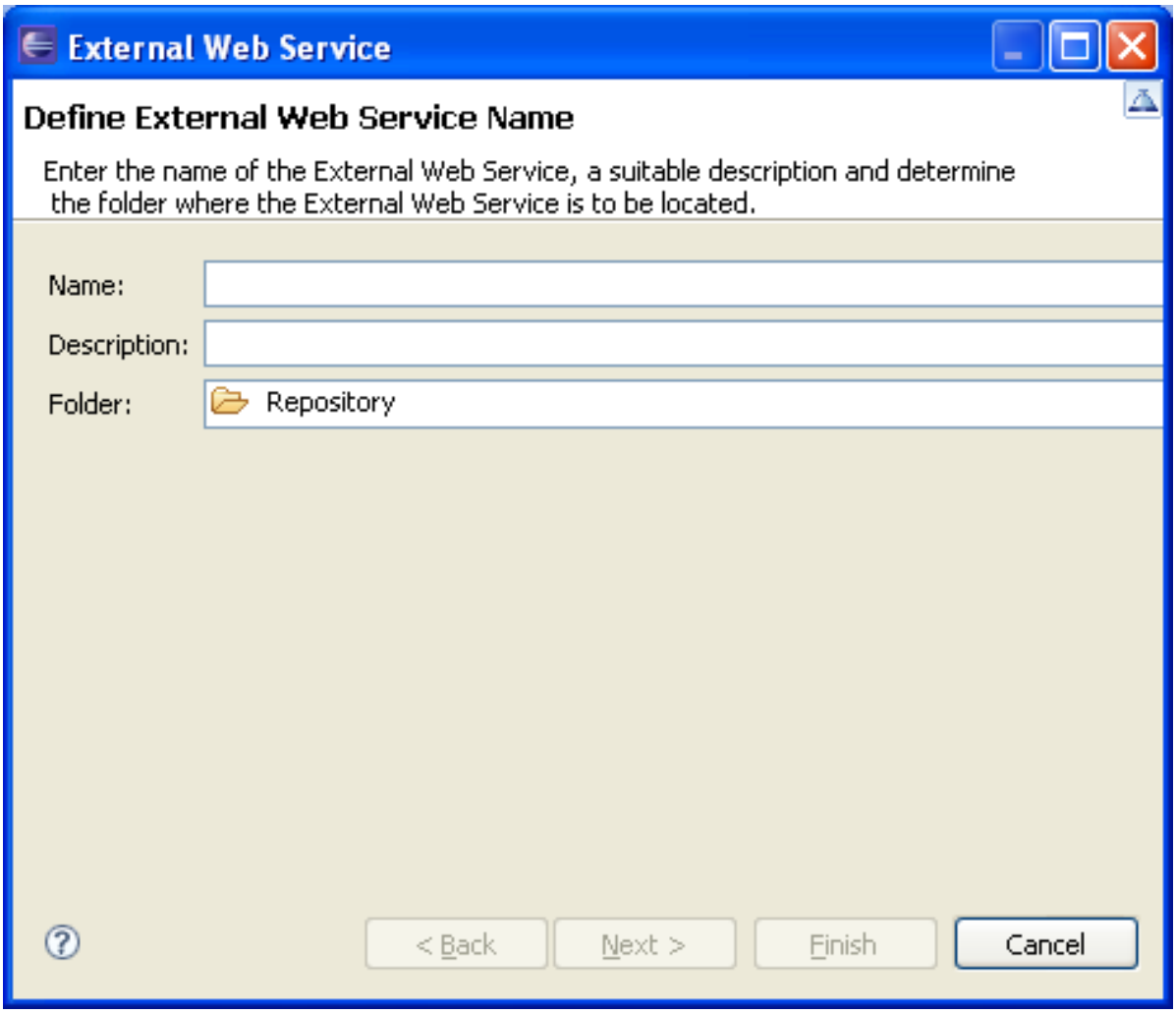
- String
- Boolean
- Double
- Float
- Int
- Integer
- Long
- Date (The format can only be 2001-07-04T12:08:56.235.
- dateTime

To create an external web service

1 Select the relevant application.

2 In the **File** menu choose **New>Entity>External Web Service**.

3 The External Web Service wizard is displayed.



4 Enter a name and click Next.

5 The Web Service Source screen is displayed.

6 Select to use a WSDL URL (currently the only option available).

7 When relevant, enter the WSDL URL or select a file name. You may be required to provide a user name and password.

8 Click **Finish**. The External Web Service is displayed in the Editor area. The General tab displays the basic information about the web service and the Structure tab displays the contents of the Web service.

Note: It is possible to change the target server. This can typically be used when changing the environment, such as from developing and testing environment to the production environment (which may be on a different server than the developing and testing server), as when changing the environment, the URL will be broken, causing the External Web Services not to work. This is done by clicking on the relevant button in the General tab and then selecting the new address (IPv4 and IPv6 address formats are supported) and port. When necessary, you can also select to update the WSDL URL accordingly. These changes can be applied just to the current Web Service, or to all External Web Services using the same target server (when selecting this option, the list of updated services appears in the server log).

Debugging Procedures

After creating a procedure, use the Debugger tool to observe the run-time behavior of your procedure and determine the location of logical errors. With the debugger, you can control the execution of your program by setting breakpoints, stepping through your code, running the code to the cursor position and examining the contents of variables. Refer to the Eclipse help for further details on the Debugger.

You can debug the procedure by running the whole procedure and then viewing the output in the Debugger Panel (click on the Play icon), or by debugging the procedure step-by-step (click on the Step icon). It is also possible to run the procedure to a certain line by clicking on the line and then selecting Run to Cursor from the right-click shortcut menu. When debugging a test case procedure, the Session Viewer is also displayed in the debug mode and displays the different steps in the procedure.

To debug a procedure

1 Right-click on the relevant procedure and select Debug or Run.

Note: The first time a test case procedure that uses a Connection Pool is debugged, the Session Selection dialog box is displayed, requiring you to select the relevant session to use when debugging this test case procedure.

2 Sometimes, the procedure requires entering values for input parameters. In such a case a pop-up appears. For example:

```
<In>
<password></password>
<reconnect>true</reconnect>
<user>demo5</user>
</In>
```

The values displayed within the tags are default values set by the user.

- When deleting a text value and/or not entering a value (e.g. `<password></password>`), an empty string is used.
- When deleting a boolean, numeric or date value and/or not entering a value, the default is used (default values are as follows: for a boolean parameter - false, numeric - 0 and date - current date).
- When removing a whole element (such as deleting `<reconnect>true</reconnect>`), the user defined default is used when one exists, otherwise, the default is used (default values are as follows: for string parameters - null, boolean - false, numeric - 0 and date - current date).

7 Screen Groups

A screen group binds several screens that share common visual or logical attributes. The relationship between screen groups to screens is many-to-many: a screen group can include many screens, and a screen may be included in many groups.

See also *Screens and Screen Groups* in the *Natural Screen Tester Reference Guide*.

Introduction

A screen that belongs to a group inherits all the field mappings that are mapped to the group. Unidentified screens are associated with screen groups based on the screen group identifiers defined. Identified screens relate to specific, predefined screen groups (specified in the Screen entity dialog box), ignoring the screen group identifiers. In addition, it is possible to define a screen group that includes all the unidentified screens.

Screen groups are typically used:

- To identify a large number of Host Screens without having to identify all the relevant Natural Screen Tester screens.
- To define once a Field Mapping that is repeated in multiple screens (error message, screen title, command line, menu selection, etc.).
- To implement **Screen Creation Definitions**

Note that when there are a number of screen groups, it is necessary to determine the order of priority between the screen groups. For more details refer to ***Prioritizing Screen Groups***

When using Screen Groups, you may want to associate to a specific Screen Group all the screens which contain a certain field, wherever this field may appear on the screen. For example, you may want all screens which contain a menu selection field to belong to the same screen group. This is possible using the mapping type called "Single Dynamic". This mapping type maps fields according to their leading label, no matter where this field appears on the screen.

Fields can be mapped according to their position in the screen or according to their leading label:

Creating a Screen Group

The Screen Group entity binds several Screens that share common visual or logical attributes. Each screen group must have unique name. A screen group may have identifiers enabling the screen group to be implicitly associated to specific unidentified screens or have no identifiers, enabling the screen group to be associated with all unidentified screens. This section covers the following topics:

- Creating a New Screen Group
- Defining Identifiers
- Mapping Fields

Creating a New Screen Group

To create a new Screen Group

1 Select the relevant application.

2 In the **File** menu select **New>Entity>Screen Group**. The *New Screen Group wizard* is displayed.

3 Enter a name for the Screen Group, a suitable description and determine the folder where the screen is to be located. Click **Finish**. You have now created a Screen Group entity in the repository.

Defining Identifiers

For the Natural Screen Tester Server to recognize a host screen, you need to identify the screen in Natural Screen Tester. The screen is identified using "identifiers". Different identifier types include identifying specific text on the screen, the screen cursor position or the field attributes. The Natural Screen Tester Server analyzes the current screen and tries to match it to all of the screen identification strings stored in the repository. For a screen to be recognized, all its identifiers must be matched. You may define an unlimited number of identifiers. The following identifiers are available:

- Text Identifiers
- Cursor Position Identifiers
- Attribute Identifiers

- Window Identifiers

Text Identifiers

Use text identifiers to identify a screen by selecting specific text that appears on the screen. Use the Capture from Screen feature to identify a string of text accurately from the host screen and insert this string into the text box.

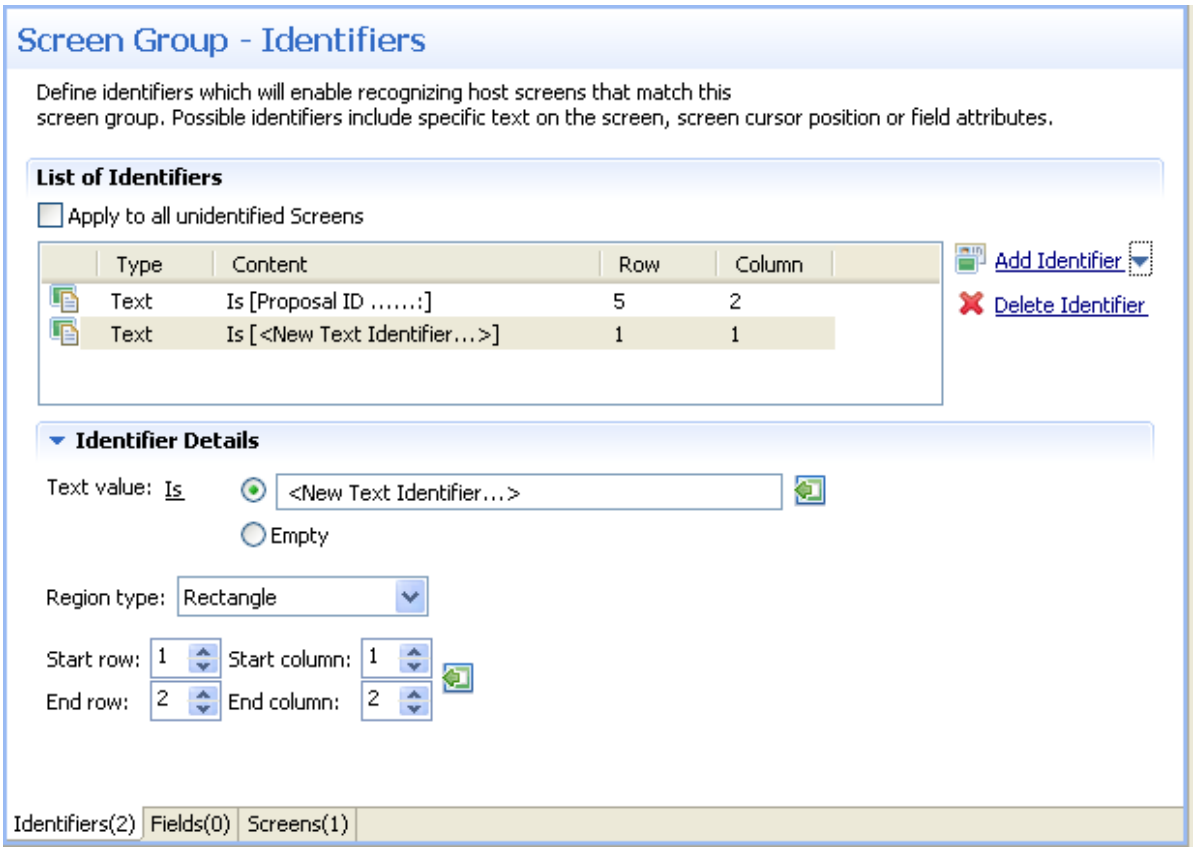
To define a text identifier

1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 Determine whether to ignore the window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

4 Click the arrow on **Add Identifier** and select **Text**. Another row will be added to the list of identifiers.



5 Select **Is** (default) in order to match the exact text on the host screen. Alternatively, select **Is NOT** if any text other than the specified is suitable as a screen identifier. Check **Empty** to identify an empty space on the host screen. This clears the **Text** text box.

6 Select **Rectangle** to indicate that the identifier should be searched for within the defined rectangle, **Position**, to indicate that the identifier must start at a specific position or **Anywhere on screen** to indicate that the identifier may be anywhere on the screen.

7 When selecting Rectangle, define the rectangle range (start and end row and column.). When selecting Position, select the row and column.

8 It is also possible to add an identifier by selecting the area in the host screen, and then clicking Add Identifier. When selecting an area which is a rectangle, identifiers will be added for each and every row of the selected rectangle.

Cursor Position Identifiers

Use the cursor position identifier to identify a screen by the cursor position when the screen is loaded.

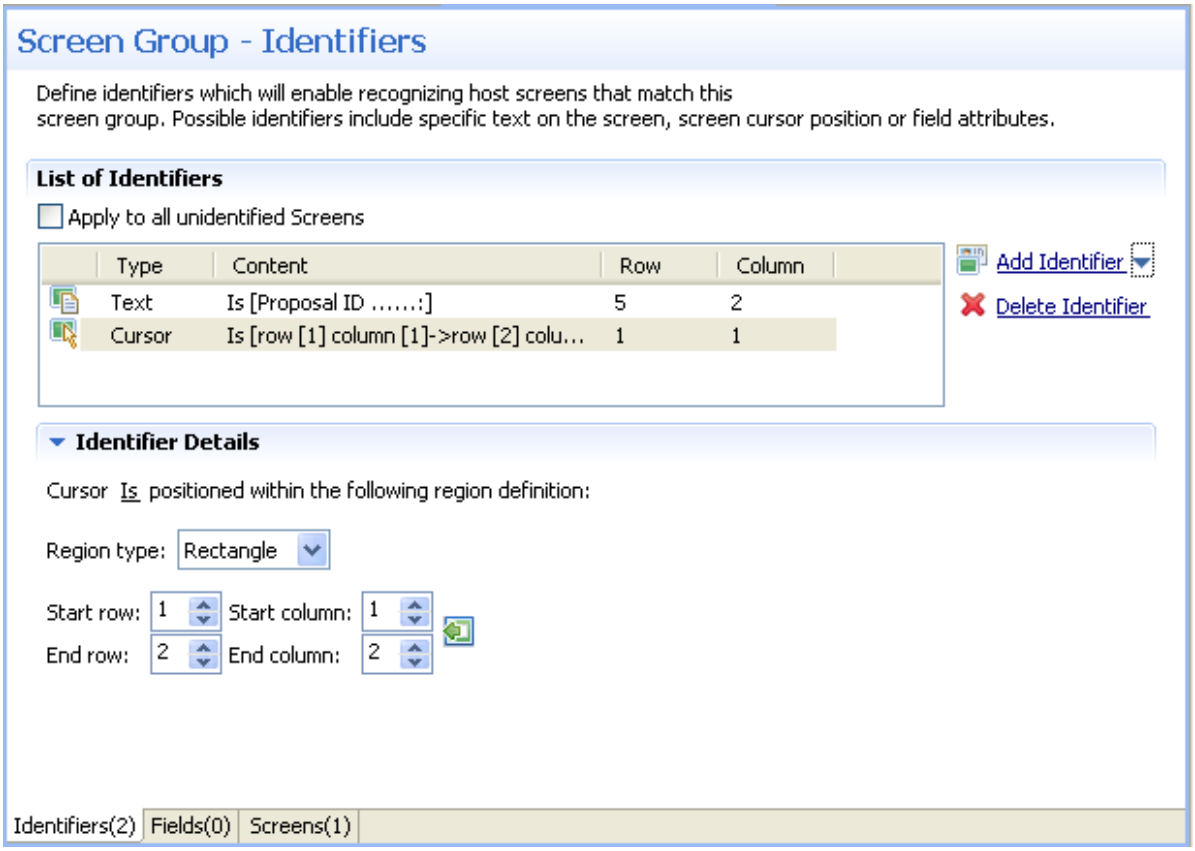
To define a cursor position identifier

1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 To ignore the window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

4 Click the arrow on **Add Identifier** and select **Cursor Position**. Another row will be added to the list of identifiers.



5 Select **Is** to indicate that the cursor is positioned within the region area details that follow. Alternatively, select **Is NOT** to indicate that the cursor is not positioned within the region area details that follow.

6 Select **Rectangle** to indicate that the cursor position should be searched for within the defined rectangle or **Position**, to indicate that the cursor position must start at a specific position

7 It is also possible to add an identifier by selecting the area in the host screen, and then clicking Add Identifier.

Attribute Identifiers

Use the Attribute identifier to identify a screen by the host's attributes. Attributes of the host screen may be Protected, Intensified or Hidden.

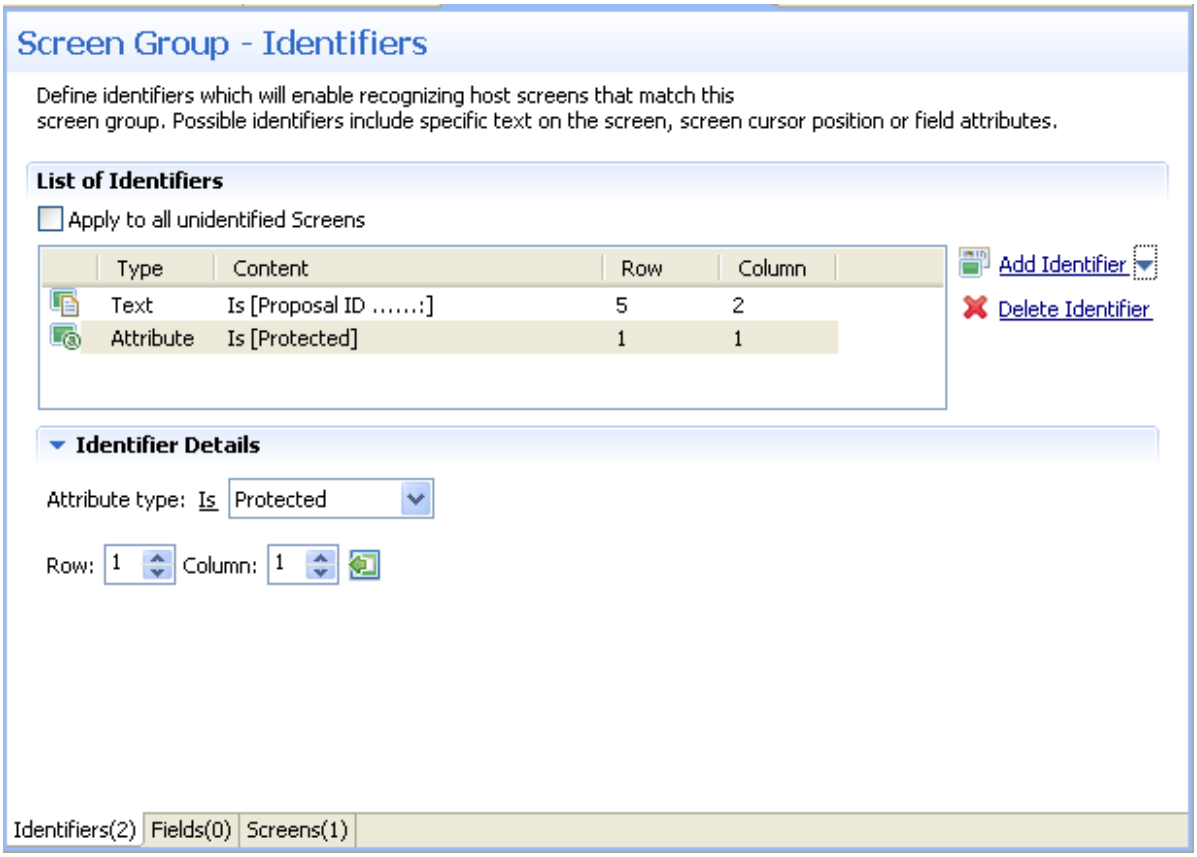
To define an attribute identifier

1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 To ignore the window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

4 Click the arrow on **Add Identifier** and select **Attribute**. Another row will be added to the list of identifiers.



5 Select whether the Attribute Type **is/is not Protected**, **Hidden**, **Intensified** or **Reversed video**.

6 Enter values or use the Capture from Screen feature to indicate a specific location of the attribute by its **Row** and **Column**.

7 It is also possible to add an identifier by selecting the area in the host screen, and then clicking Add Identifier.

Window Identifiers

Use the Window identifier to identify a screen by determining whether it is or is not a window. In Natural UNIX, it is possible to identify a screen by determining whether it is a window and whether it has specific text in the title.

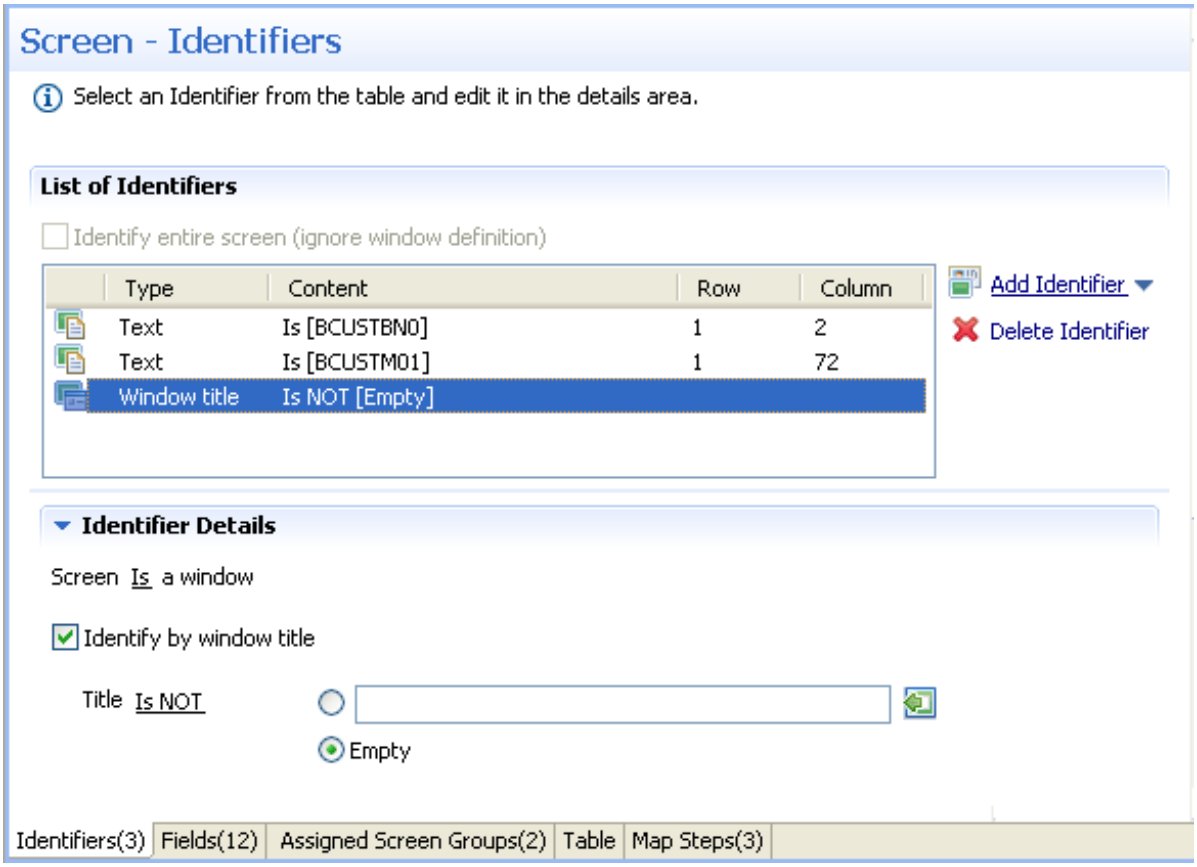
To define a window identifier

1 Access the relevant screen.

2 Select the **Identifiers** Tab.

3 To ignore the window definitions during the identification of a screen, select the relevant check box. Once the screen is identified the full screen is used.

4 Click the arrow on **Add Identifier** and select **Window**. Another row will be added to the list of identifiers.



5 Select **Is** to indicate that the screen is a window. Alternatively, select **Is NOT** to indicate that the screen is not a window.

6 For Natural UNIX hosts: Select the check box to indicate to identify by the window title. You can enter text for the title or select the text in the session view and click the Capture from screen icon. You can also identify a screen which is a window and does not have a title by selecting the Empty radio button.

Mapping Fields

Introduction

When a screen is identified, field entities can be mapped to it. A field makes relating to a string on the host screen much simpler. Fields offer an enhanced way of referring to host fields - by their name rather than by their position. In addition, mapped fields simplify future maintenance - when a field changes in the original host application, updates only need to be made to the field definition. The following guidelines will help you determine which fields to identify:

- Fields that will be referenced in code (when binding Web page controls to host fields by name).
- Fields that will be part of navigation paths.
- Fields that will be used as test case input and output attributes to expose host transactions as assertions.
- Recommended: all input and output fields (not constants).
- Any field in the host application you would want to expose as an assertion.

An application field makes relating to a string on the host screen much simpler. Fields offer an enhanced way of referencing host fields - by their name rather than by their position. Using fields has many advantages:

- A field's ID is meaningful regarding the contents of the field, unlike a numeric position.
- You define a field only once, and later can reference that definition in all of your code - knowing you always mean the same definition.
- If changes in the host application occur in the future, the only place that requires to be changed is the field definition.

Field properties can be accessed by right-clicking on the Repository node and selecting **Open Entity (Find)...** and then searching for the specific field.

In the Field Editor, it is possible to set the Input Mask (Defines the values that can be entered in the field) and the field type (numeric or alphanumeric).

Input Masks

An input mask consists of literal characters along with special characters that together determine the value that may be entered into input fields. When a user defines a mask on an input field, client side validation will be carried out on the input format. Common usage is in date/time and number or currency fields.

To implement this feature you must change the web application configuration in the configuration editor

1 Open a new browser and run your Web application.

2 Click on the Configuration link. The Configuration Editor will be displayed.

3 In the **Instant** node select the **Reflect emulation behavior** check box.

The following table shows some useful input mask definitions and examples of values you can enter. Refer to Valid input Characters for details on the codes used to create input mask definitions.

Input Mask Definition	Example Values
(000) 000-0000	(206) 555-0248
(999) 999-9999	(206) 555-0248 ( ) 555-0248
(000) AAA-AAAA	(206) 555-TELE
#999	-20 2000
>L????L?000L0	GREENGR339M3 MAY R 452B7
00000-9999	98115- 98115-3007
L??????????????	Maria pierre
ISBN 0-&&&&&&&&&&-0	ISBN 1-55615-507-7 ISBN 0-13-964262-5

Valid Input Mask Characters

Natural Screen Tester interprets characters in the Input Mask property definition as shown in the following table. To define a literal character, enter any character other than those shown in the table, including spaces and symbols. To define one of the following characters as a literal character, precede that character with a "\".

Character	Description
0	Digit (0 through 9, entry required; plus [+] and minus [-] signs not allowed).
9	Digit or space (entry not required; plus and minus signs not allowed).
#	Digit or space (entry not required; blank positions converted to spaces, plus and minus signs allowed).
L	Letter (alphabetic, entry required).
?	Letter (alphabetic, entry optional).
A	Letter or digit (entry required).
a	Letter or digit (entry required).
&	Any character or a space (entry required).
C	Any character or a space (entry optional).
\	Causes the character that follows to be displayed as a literal character. Frequently displayed any of the characters listed in this table as literal characters (for example, \A is displayed as just A).

When a screen is identified, field entities can be mapped to it. A field makes relating to a string on the host screen much simpler. Fields offer an enhanced way of referring to host fields - by their name rather than by their position. In addition, mapped fields simplify future maintenance - when a field changes in the original host application, updates only need to be made to the field definition. The following guidelines will help you determine which fields to identify:

- Fields that will be referenced in code (when binding Web page controls to host fields by name).
- Fields that will be part of test cases.
- Fields that will be used as test case input and output attributes as assertions.
- Fields that will be used within tables.
- Recommended: all input and output fields (not constants).
- Any important field in the host application.

There are limitations when defining a mapping. A mapping cannot:

- Start in an unprotected field and end in a protected field, or vice versa.
- Start with an attribute byte.
- Overlap another mapping on the same screen.
- Start in one line and end on another line.

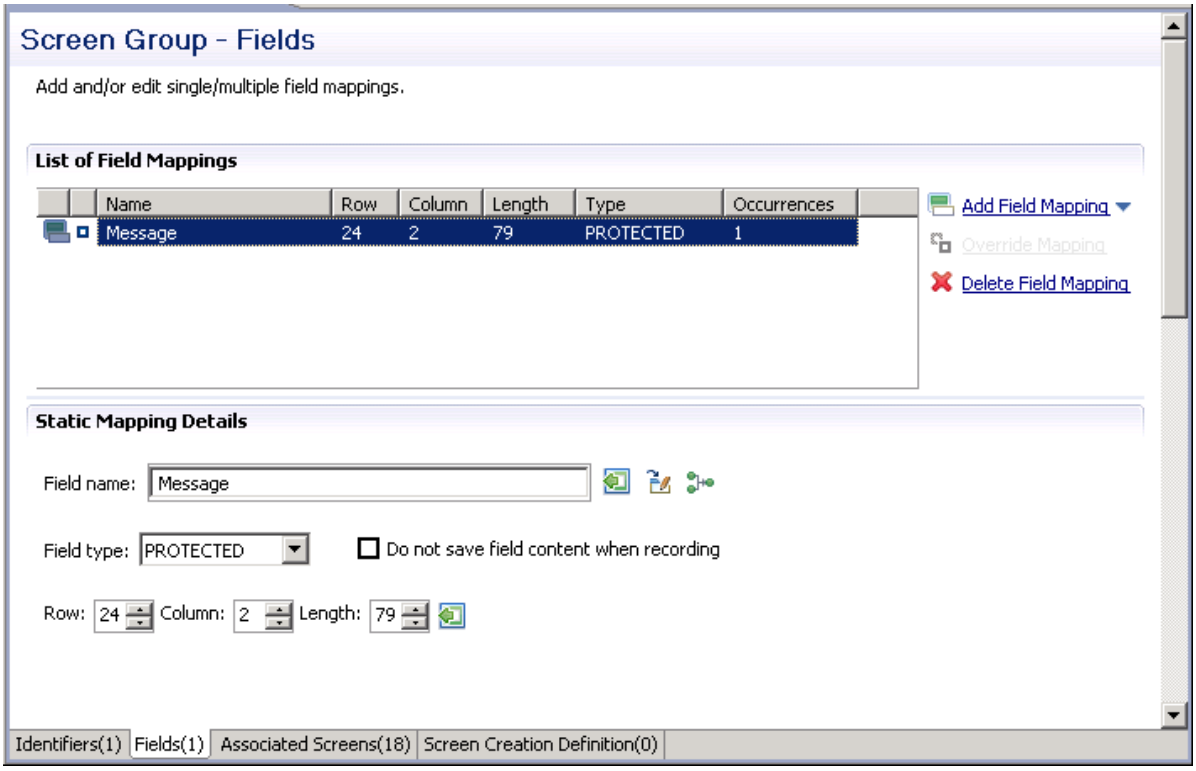
- Start on one line and end off the screen's limits.

Mapping Fields according to their Position in the Screen

To map fields according to their position in the screen

1 Click the **Fields** tab to view and/or modify the fields in this screen group.

2 Click the arrow next to **New Mapping**, and select **New Single Mapping** or **New Multiple Mapping** icon to add a new field mapping.



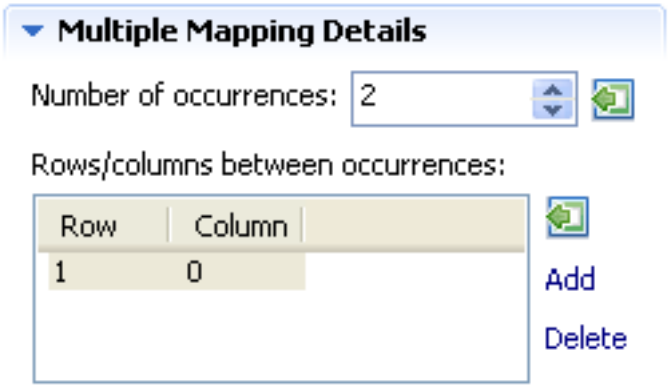
3 In the **Field name** field, either enter a field name or enter the first letters of a field name and then click CTRL and SPACE to display relevant fields.

4 Select the **Protection type**: **Protected** (cannot be edited in host), **Unprotected** (editable in host) or **Both** (may sometimes be editable and sometimes not).

5 The **Do not save field content when recording** check box enables not saving data which you do not want to be displayed when viewing the recorded trace file.

6 The **Row** and **Column** fields indicate the position of the field on the screen. The values of these fields can be changed using the **Capture from Screen** feature or by entering values in the fields.

7 When adding a multiple field mapping, determine the number of occurrences and the rows and/or columns between occurrences.



To delete a field mapping, click on the **Delete Field Mapping** hyperlink.

Mapping Fields according to their Leading Label

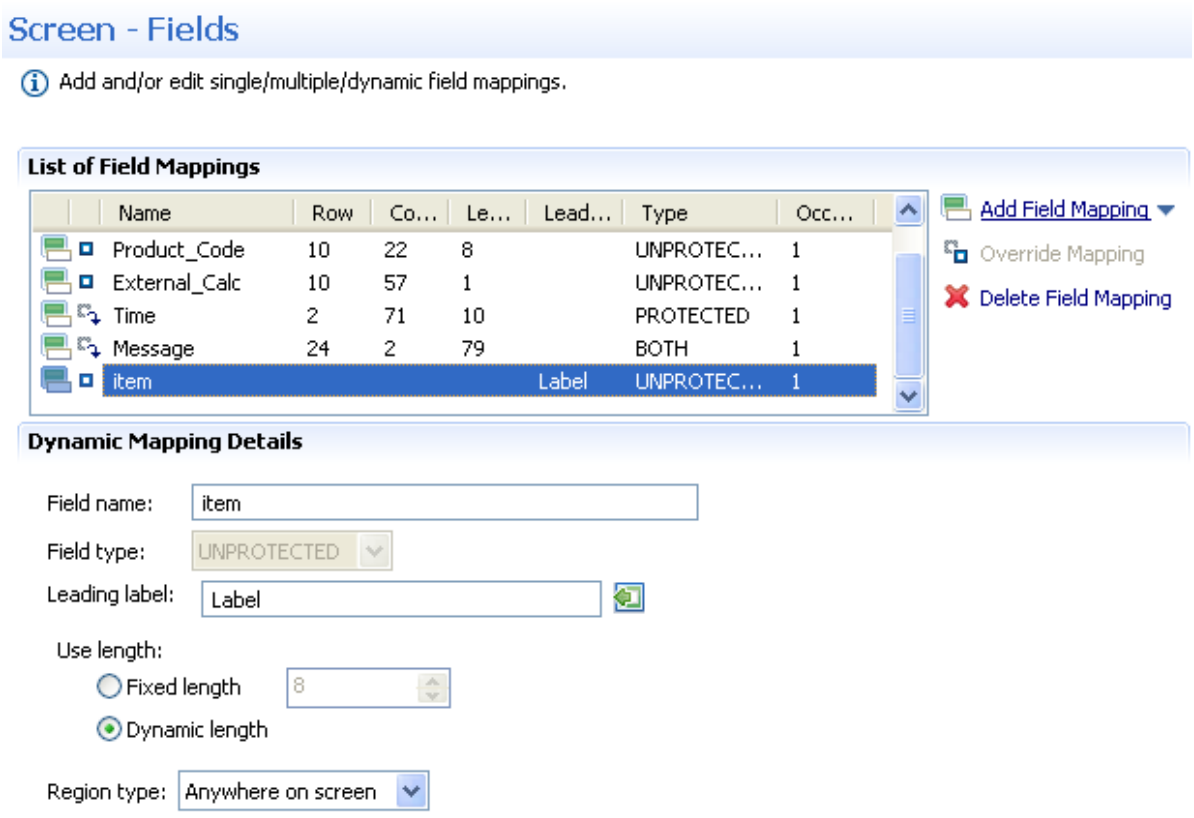
Note: This feature is not available in right to left and character mode applications.

To map fields according to their leading label

In some screens, the position of a field may vary. It is therefore necessary to map some fields in such a way, that even if they appear in a different position, they will still be recognized and mapped. This is possible by defining the label near the field (the label must be to the left of the field) and identifying the field according to this label. This mapping type is called "Single Dynamic" as the mapping position changes dynamically according to the leading label. Use this type of mapping only when the screens in the screen group have a very similar structure. Refer to *Dynamic Field Mapping Limitations*.

1 Click the **Fields** tab to view and/or modify the fields in this screen.

2 Click the arrow next to **New Mapping**, and select **Single Dynamic** to add a new field mapping.



3 In the **Field name** field, either enter a field name or enter the first letters of a field name and then click CTRL and SPACE to display relevant fields.

4 Select the field protection type: Unprotected, Protected or Both. Prefer to set the field type either as **Protected** or **Unprotected** when you are sure of the type of the dynamic field. Use **Both** when you are not sure of the type of the dynamic field.

5 Determine the leading label: the leading label is the label which appears before the field. This label is inserted as the Field name and can be edited. The label you define should be as accurate as possible (do not use strings which normally appear within the field).

6 Determine the method to use to search for the label:

- Contains text: Searches for the given text anyway in the defined region (default).
- Exact phrase: Searches for the given text within the defined region. The text must be preceded and followed by at least two white space characters.
- Contains word: Searches for the given text as word(s) within the defined region.
- Regular expression: Searches for the given pattern within the defined region. Refer to *Regular Expression Syntax*.

Note: When selecting a different option, after defining a regular expression search definition, the value of the previous regular expression definition will be lost.

7 Define the region type in which the label will be searched for: anywhere in the screen, within a specified rectangle or within a specified row range. The **Row** and **Column** fields indicate the position of the label on the screen. The values of these fields can be changed using the **Capture from Screen** feature or by entering values in the fields.

8 Use length:

In Unprotected fields: Determine whether to recognize a field mapping according to the leading label and the length of the input field (Fixed length) or just according to the leading label (Dynamic length).

In Protected fields: The length defined here is the length of the mapping. The identified field is cropped to the specified length.

9 Define the region type in which the field will be searched for: anywhere in the screen, within a specified rectangle or within a specified row range. The **Row** and **Column** fields indicate the position of the field on the screen. The values of these fields can be changed using the **Capture from Screen** feature or by entering values in the fields.

Note: The region will be displayed in the Session View with a yellow frame, and the background of the screen within this region will be dotted.

The following guidelines apply when mapping fields according to their leading labels:

- It is preferable to define your regions as rectangles, that do not overlap each other, and whose size are minimal.
- The feature is best suited to locate dynamic fields which change their location vertically so that labels are aligned vertically in one region, and fields aligned vertically on a region to the right of the labels.

Configuring Screen Groups

Associate Screens

A screen that belongs to a group inherits all the field mappings that are mapped to the group.

To associate screens to a screen group

1 Access the relevant screen and click on the Associated Screens tab. The list of Screen Groups currently associated with this screen are listed.

2 Click on Assign Screens to select additional screens to be associated with this screen group.

Create Screen Creation Definitions

Screen Creation Definitions are a set of basic definitions which are used to create a new Natural Screen Tester screen. These definitions include the default screen name, initial identifiers and determine whether input fields will be created. Creating a new screen using screen creation definitions reduces the complexity and the time required to identify new screens.

To use this feature, at least one set of Screen Creation Definitions must be created within a Screen Group (in the dedicated tab). When navigating within a session, and reaching an unidentified screen, Natural Screen Tester searches for Screen Groups which match the unidentified screens, and then creates a new screen based on the screen group's identifiers and on parameters defined in the creation definition configured in the Screen Group. This process can be initiated in three different modes: automatic, semi-automatic and manual (defined in the view session). Refer to Changing the Screen Definition Mode.

To create screen creation definitions

1 In the Screen Creation Definition tab, click on Create Definition to define a new screen creation definition.

2 Determine the position of the screen name in the host screen. Click on the Capture from screen icon after selecting the relevant position in the screen to automatically enter the position.

3 Determine the location of the identifiers to be used in the new screen. It is possible to add a location by selecting the area in the host screen, and then clicking Add Identifier Location.

4 Determine whether to automatically create input fields.

5 When automatically creating input fields, determine whether to suggest nearby labels as field names and whether to add the screen name as a prefix to the input fields' names.

Prioritizing Screen Groups

It is necessary to determine the order of priority of the screen groups in the following cases:

- When two screen groups match an unknown screen and contain the same field but with different mapping position of the field. The order of the screen groups will determine which screen will be used.
- In the framework, when generating a page for two screen groups and when two screen groups match an unknown screen, the screen group appearing former in the list will be applied.

To prioritize screen groups

1 In the repository, right-click on any Screen Group and select **Prioritize Screen Groups...**. The **Prioritize Screen Groups** dialog box is displayed.

2 Determine the order of precedence of the screen groups by clicking on the **Move Up** and **Move Down** links.

3 Click **OK** to save your changes.

Converting a Screen to a Screen Group/Converting a Screen Group to a Screen

Sometimes, after creating a screen, you may notice that it is relevant to a number of screens and therefore would like it to be a Screen Group. On the other hand, you may have created a screen group which you later realize is suitable to be a screen. Natural Screen Tester provides the option to convert a screen to a screen group and vise versa simply by right-clicking on the relevant screen/screen group and selecting Convert to Screen Group/Convert to Screen.

8 Test Project Map

The Test Project Map view displays thumbnails of the screens which the user navigated through when working with test project. Natural Screen Tester saves the navigation through these screens including the steps between each screen such as the action key pressed and the fields sent. The test project map can be used in test case procedures to navigate to a specific screen, using the NavigateTo method.

Creating Steps

Steps can be created automatically from within the Session View when working online or offline, and also via a wizard, using a number of trace files. Steps can also be added manually in the Steps tab of the Screen editor. To automatically add steps as you navigate, you must first enable recording the navigation steps. To perform the identification of steps based on a trace file, use the Automatically Identifying Test Project Map Steps wizard, and all the steps in the trace file will be identified. Each of these tasks are detailed below.

Note: Steps can only be added to identified screens.

Note: When working with an online session, the steps (i.e action keys pressed) between each screen, are the actual keys you press. When working with a trace file, the steps (i.e action keys pressed) between each screen, are the steps recorded within the trace file.

- Recording Navigation Steps from within the Session View
- Automatically Identifying Test Project Map Steps using Trace Files (GCT)
- Manually Identifying Test Project Map Steps

Recording Navigation Steps from within the Session View

To Record Navigation Steps

1 Recording navigation steps is enabled by default. To change this setting, right-click on an application and select **Properties**.

In the **Navigation** tab, select the check box which enables/disables recording the navigation steps.

2 Connect to a session and start navigating. Natural Screen Tester will automatically add the new steps.

Note: When running an offline session, you can just navigate by pressing the **Enter** key. The actual steps which were recorded are added.

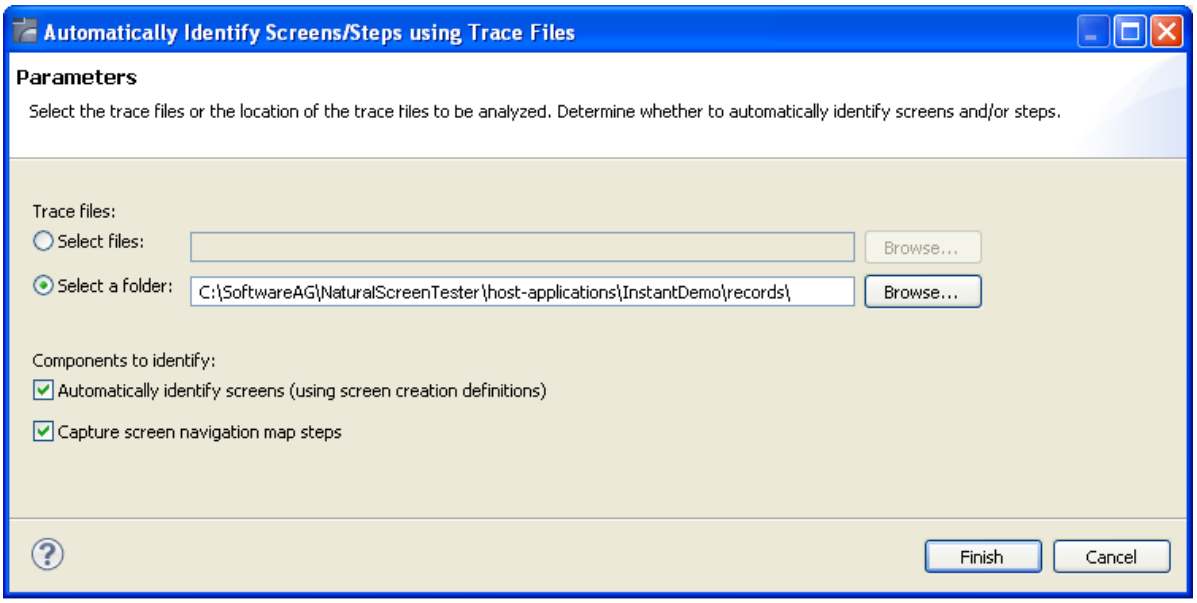
Automatically Identifying Test Project Map Steps using Trace Files (GCT)

Automatically identify steps by running the *Automatically Identify Screens/Steps using Trace Files* wizard.

Note: This wizard, is also used to automatically identify screens. If your application includes unknown screens which you would like to be displayed in the map, you must identify these screens either manually, or automatically (using this wizard), after first creating a Screen Creation Definition within a Screen Group.

To automatically identify steps using trace files

1 Right-click on the relevant application, and select **Automatically Identify Screens/Steps using Trace Files...**. The wizard is displayed.



2 Select whether to analyze files or a folder. Locate the files/folder.

3 Select the **Capture screen navigation map steps** check box.

You can automatically identify screens using this wizard by selecting **Automatically identify screens**. Refer to **Automatically Identify Screens using Trace Files (GCT)** for further details.

4 Click **Finish**.

The Console area in the Designer displays the outcome of the process.

Manually Identifying Test Project Map Steps

In the **Steps** tab of the **Screen** editor, it is possible to manually define additional steps between screens that already exist in the test project map. The steps should be defined in the source screen and determine the destination screen of the step. It is possible to add a simple step between screens or to add an inner procedure step which executes a procedure. When you add a simple step, determine the target screen, the key to be sent, and the relevant inputs. When defining an inner procedure, select the procedure. Change the status of the step using the links (Approve, Set to Pending, Set to Not Approved). Once you save the new steps, you can check these navigation definitions in the Session View using the Test Project Map toolbar. Simply select the screen you want to navigate to and click on the **Go** icon to attempt to navigate to that screen.

The Test Project Map View

Within the Test Project Map view, you can perform a number of actions:

View Step Information

Display a tooltip with the step information including the key pressed and the information sent. Just hover over the arrow and the tooltip is displayed.

Zoom-in

Change the zoom level of the view. Right click anywhere within the map and select the view zoom.

When moving the mouse over a specific screen image (and the current display is not 100%), the screen image will be enlarged.

Viewing Steps of a Specific Status

The Test Project Map view displays the steps in different colors to indicate the status of each step: green indicates approved, yellow - pending and red - not approved.

Clicking on the Approved, Pending and Not Approved icons in the toolbar, it is possible to display only the steps of the selected status.

Changing the Steps' Status

Right click on the desired step/arrow and select a status to make changes to the step's status. Using the Control key, select a number of steps/arrows and make status changes to all of these steps.

Link with Entity

The Test Project Map View can either display the whole test project or focus on the entity selected in the Explorer tree and only display the screens related to the selected entity. The radius of related screens can be determined by scrolling to the relevant number in the toolbar. Enable this feature by clicking on the Link with Entity icon, and then selecting the radius of screens to be displayed by clicking on the arrows, or entering a number 

.

Lock/Unlock View

The view displaying the map can be set to change dynamically, as events occur, (e.g., newly created screens are displayed in the map view) or it can be set to preserve the current view (Lock icon on the session toolbar).

Approving Steps

In the Session view, navigate through the screens. In the Application Map view, thumbnails are created for all the identified screens through which you have navigated. Arrows between the screens indicate the direction of the navigation. These arrows are, by default, in "Pending" status (colored in yellow). This is because, you do not always want users to enable users to navigate to all the screens (i.e. you do not want users to be able to navigate to the login page where usernames and passwords appear). In order to use the Test Project Map with a test case procedure, the status of the navigation route must be set to "Approved" (green).You can choose to display or hide all screens that have steps of a certain status (pending, approved or not approved) by clicking on the appropriate icons in the map view.

- To manually add additional steps between screens, edit the source screen in the Editor. In the Map Steps tab manually add steps. Refer to **Creating a Screen**.
- To approve a number of pending steps, right-click on one of the screens and select **Set Pending Steps to Approved**, or use the Control key to select a number of steps.
- You can select to set a screen as the first screen.

Testing the Test Project Map

The screen navigation defined in the Test Project Map can be tested to ensure that the navigation behavior is as expected. This is done in the Session View (online), using the Test Project Map toolbar. The toolbar enables selecting a screen to which you expect to be able to navigate to from the current screen and then attempting to navigate to this screen. If the Test Project Map is correctly defined, then Natural Screen Tester will successfully navigate to the selected screen. If the Test Project Map does not have the relevant steps defined to reach the screen you selected, a pop-up message will inform you of this.

Troubleshooting

An individual screen thumbnail within the Test Project Map can indicate problems or incompatibility between the defined identifiers and the screen image attached to the screen. Such problems are indicated by a red frame around the specific screen.

Possible problems:

- Incompatibility between the defined identifiers and the screen image attached to the screen. This can happen as a result of creating a screen and then manually changing an identifier.
- The screen suits the screen image, but there is a screen which is more suitable or identical to the screen image. Look at the name of the screen which is on the screen image and check that it suits the current screen. Note that the name which is on the screen image indicates how the server identified this screen image given all current screen definitions.

9 Data Structure

A Data Structure is an object that is characterized by attributes. Once defined, the Data Structure entity can be used as an input or output variable in Procedures or as an attribute of a Data Structure.

To create a new data structure

1 Select the relevant test project.

2 In the **File** menu, choose **New > Entity > Data Structure**. The New Data Structure wizard is displayed.

3 Enter a name for the Data Structure, a suitable description and determine the folder where the Data Structure is to be located. Click **Finish**. You have now created a new Data Structure in the repository and it is displayed in the Editor area.

4 Click **Add Structure** to add a structure. Enter a name, description, select a type and determine whether it is an array.

5 Click **Add Attribute** to add an attribute. Enter a name, description, select a type, determine whether it is an array and provide a default value.

6 Use the **Move Up** and **Move Down** links to change the order of the data structures.

10 Database Connection

A database connection is an entity that is used for direct database access in a flow procedure. Once the database connection is defined, it is possible to connect to it in any procedure and perform execute, select, commit and rollback statements.

To create a Database Connection

1 Create a new Database Connection via the **File > New > Entity** menu item.

2 Enter a name and description and click **Next**.

3 Select the type of database that will be used as the application's database connection.

Database	Driver	Requires Driver Files
Apache Derby	org.apache.derby.jdbc.EmbeddedDriver	
SQL Server	com.Microsoft.jdbc.sqlserver.SQLServerDriver	Yes
Oracle	oracle.jdbc.driver.OracleDriver	Yes
MySQL	org.git.mm.mysql.Driver	Yes
DB2	COM.ibm.db2.jdbc.app.DB2Driver	Yes

4 Click **Finish**. The Database Connection details are displayed.

11 Connection Pools

A connection pool enables you to immediately get a host connection that is ready in a specific screen, the “initial screen”. This saves the time of establishing the connection with the host and navigating to the relevant screen. See Navigation in the *Natural Screen Tester Reference Guide*.

See also: Connection Pools and Connection Information Sets in the *Natural Screen Tester Reference Guide* | **Connection Information Sets** in this manual.

Creating a New Connection Pool

To create a new Connection Pool

1 Select the relevant application.

2 In the **File** menu select **New>Entity>Connection Pool**. The *New Connection Pool wizard* is displayed.

3 Enter a name for the screen, a suitable description and determine the folder where the screen is to be located. Click **Finish**. You have now created a new Connection Pool in the repository.

4 Edit the settings in the Editor area as detailed in the Connection Pool reference.

Changing the Initialization Mode of a Connection Pool

To change the Initialization Mode of a Connection Pool

1 Double-click on the relevant Connection Pool.

2 Select the **General** tab.

3 In the **Initialization mode** field, select the relevant mode:

- Manual: An administrator manually initializes the connection pool (the connection is initiated by selecting the connection pool in the Management node, and then choosing Start Pool from the right-click shortcut menu).
- When first accessed: When Web applications or sessions request to connect to this connection pool.
- Automatic: Automatically initializes the connection pool when the application is loaded.

Starting and Stopping Connection Pools in the Designer

You can manage connection pools from the Natural Screen Tester Explorer of the Software AG Designer.

To start a connection pool

- Select the connection pool, and from the context menu choose **Start Connection Pool**.

To stop a running connection pool

- Select the connection pool, and from the context menu choose **Stop Connection Pool**.

Changing the Log Level

Fine-tuning connection pool parameters or identifying problems are reasons you may want to change the level of detail that the log displays. It is possible to set a different detail of logging for each connection pool: None, Errors, Warnings, Information and Details. The server logger should be set to no less that the **Normal** log level, in order to see connection pool logs. Once the project is in the production phase, **Error** level is the recommended level to use.

Defining an Initialization Path

An initialization path is a path that can be executed on any new connection to the host that navigates the connection to one of the Initial Screens. In order to select a path from different folders, click the folder selection button.

Open the Navigation tab of the relevant Connection Pool and select the test case or test case procedure from the Initialization drop-down list.

Defining a Termination Path

A termination path is a path that can be executed on any connection to the host (used or new) that should be performed before destroying a connection (for example, a host-side logout procedure). A termination path is triggered automatically by the Natural Screen Tester Server in the following situations:

- when a connection is canceled
- when a connection is disconnected, with "disconnect after usage" configured
- when the connection pool is stopped or exceeds the number of allowed connections based on the pool policy and needs to reduce the number of available connections
- when the connection is trying to return to the pool but can't reach the initial screen; the termination path will be used to cancel the connection gracefully

To define a termination path, open the **Navigation** tab of the relevant connection pool and select the test case or test case procedure from the **Termination** drop-down list.

Connection Pool Life Cycle

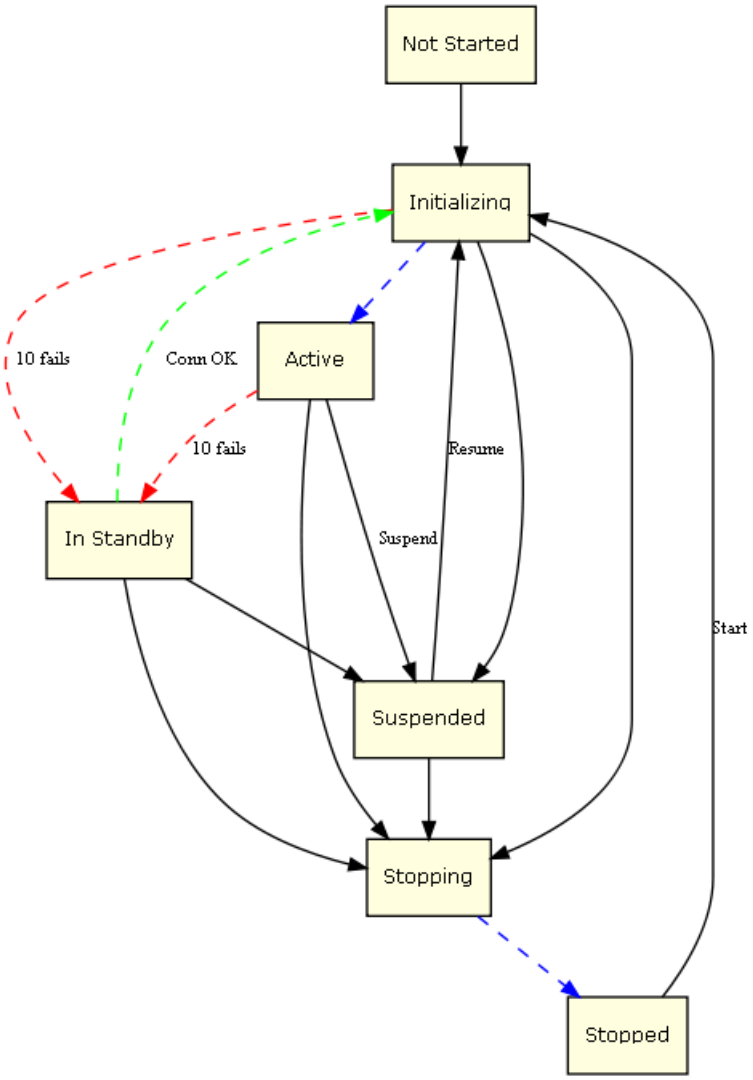
Connections in the pool automatically go through different states, according to the pool policy as defined in the Connection Pool Entity. The connection policy is built from:

- Pool size type: fixed, limited, flexible
- Keep-alive path and interval
- Run recycle anyway flag
- Disconnect after usage flag
- delays and timeouts

Other factors may also impact pool behavior.
See Pool Size Control Policy Considerations | License Key Concept | **Connection Information Sets**.

Connection Pool States

- The Connection Pool itself can have seven states:

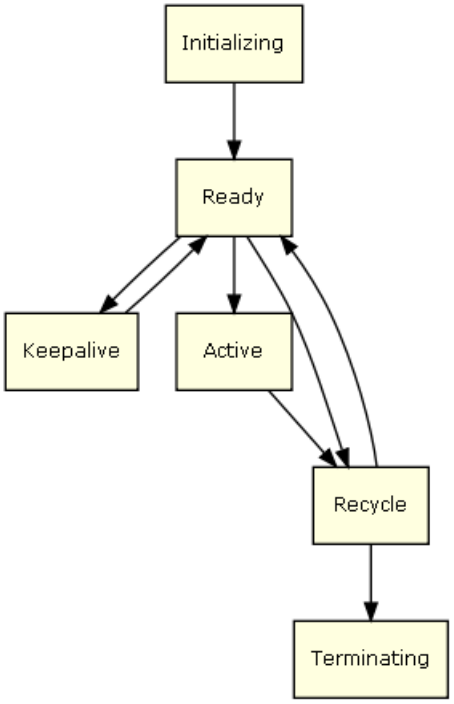


State	Description
Not Started	The connection pool was not initialized yet. If you request a connection, the connection pool will return an immediate error.
Initializing	The connection pool is trying to reach Active status, but does not have any "ready" connections yet. If you request a connection, the connection pool will return an immediate error.
Active	The connection pool is working and managed to create at least one connection to the host.

State	Description
In Standby	The connection pool had several consecutive errors trying to create new connections to the host. The connection pool will continue to try to connect if user requests arrive, but will not initiate new connections otherwise. If a new connection is successfully created, the status will automatically change to Active.
Suspended	The connection pool is blocked for new users and does not maintain its connections. If you request a connection, the connection pool will return an immediate error.
Stopping	The connection pool is trying to reach Stopped status, but still has connections in different stages of termination. When all connections are down, the status will automatically change to Stopped. If you request a connection, the connection pool will return an immediate error.
Stopped	The connection pool does not have connections or maintenance. If you request a connection, the connection pool will return an immediate error.

Note: the Stopped state will only happen after Stopping; the Initializing state will happen before Active. These are "in between" states, typically not lasting more than a couple of seconds.

- Host connections inside the connection pool can have six states:



State	Description
Initializing	Host connection will be in initializing state when created and until it reaches its initial screen, usually while running the initialization path. If you are using a connection info set: the parameters of a single record are retrieved.
Ready	Connection connected to the host but not yet serving any requests (session/user). Waiting on the initial screen for a request.
Active	Attached to a session/user, currently serving a request.
Keep-alive	If a keep-alive path is defined in the connection pool, the connection will be in this state while the keep-alive path is running (typically no longer that a few seconds).
Recycle	If a recycle path is defined in the connection pool, the connection will be in this state while the recycle path is running.
Terminating	Disconnecting the connection. If a termination path is defined in the connection pool, it will be executed. If using a connection info set: this will release the parameters of the record used.

Note: Connections can also appear to be in Broken state. This is not a "real" state, it just indicates that the connection cannot get to a Ready state. Broken connections do not count with regards to license capacity and do not use any system resources. Broken connections will be automatically cleared after a certain time. See **Connection Pool Troubleshooting** for more information.

Connection Pool Troubleshooting

Symptom	Explanation	Possible Reasons	How to Check
All my connections are broken	Host sessions become *broken* if they cannot get to Ready state. Ready means that: 1. The connection is alive. 2. If an "initial screen" is defined by the pool, the session must get there.	Host is unavailable.	Try to connect to the host without the pool.
		Initialization path needs to be modified.	Create a connection outside the pool, run the initialization path and verify it gets to the "initial screen".
Connections disconnect after use but I want to reuse them	The pool may be configured to allow reusing a host connection that was used by a previous user. This would require returning the connection to the initial screen, either by	The **Disconnect after usage** option is checked.	Observe the state of the Disconnect after usage checkbox in pool tab of the connection pool.
		The logic of the user activity or the recycle path causes a disconnection from the host.	Capture a trace file for the user activity and recycle path. Observe that the host session is not terminated during the invocation of user activity or the recycle path.

Symptom	Explanation	Possible Reasons	How to Check
	implementing a robust navigation logic inside the invoked procedure or in the recycle path.	When a connection is returned to the pool and it is not in the "initial screen", and there is no recycle path that successfully navigates to the initial screen.	Capture a trace file for the user activity and recycle path. Observe that at the end of the user activity and the recycle path invocation, the connection is in the initial screen.
Connections remain Active after use	Upon finishing the user activity on a pool connection, the state of the connection would change according to the pool configuration and the state of the user session. The state of the connection would remain active if the session used by the pool has not ended.	A flow procedure creates an emulation session on a pool connection, but the "end session" node is not called on that session.	Verify the existence of an "end session" node in the flow procedure that creates the emulation session. Verify that the "end session" node is reached by capturing a trace file or by logging the invocation of the flow procedure.
Connections remain Terminating after use	A pool may be terminating a connection based on the pool's configuration, connection count and the status of the session. When doing so, the connection status would become "Terminating" and the termination path would run on the connection. After the termination path is completed, the connection is removed from the pool.	An exception in the termination path is causing the host connection to either get stuck or to terminate before completing the path.	Capture a trace file that includes the invocation of the termination path and observe that the path has completed successfully. Search for exceptions in the server logs that occurred during the invocation of the termination path.
A used connection is disconnecting while being used		The host is disconnecting the pool connection based on the user activity or due to the host state.	Capture a trace file for the user activity, observe the last packet transmissions between the server and the host. Look in the server log for an error indicating a socket close around this timestamp. Also ask the host administrator to inspect the host's log for disconnections.
		Another host connection is using the same host credentials/device name and hijacking the session.	When capturing trace files, you would notice a trace file created at the time when a previous trace file is closed. Both traces will include send sections containing the same credentials or device name.

Note: When recording trace files to capture the symptoms mentioned above, we recommend using the following variables in the trace file name: connection ID (`%c`), session ID (`%u`), creation time of file (`%t`). See *Recording Trace Files* for more information.

12 Connection Information Sets

A connection information set supplies a pool of possible connection parameters required for the initial connection to the host (such as the device type or host address) and for the execution of the connection pool initialization path when it exists (such as the required user name and password). See also *Connection Information Sets* in the *Natural Screen Tester Reference Guide*.

The connections in a connection pool are created before an end user actually connects to Natural Screen Tester. Therefore, connection or login parameters that are usually supplied by the user (such as user ID, user password, device name, etc.) must be supplied by a different source. This source is the Connection Info Set (also known as Connection initialization parameters). A connection information set supplies a pool of possible connection parameters required for the initial connection to the host (such as the device type or host address) and for the execution of the connection pool initialization path when it exists (such as the required user name and password).

Each information set may list a number of records. When initializing a connection, the parameters of a single record are retrieved. The number of times that a single record can be used concurrently is defined in the Repeat column of the Connection Information Set dialog box. When set to zero, this record can be used an unlimited number of times. More than one connection pool can use the same connection information set for its connections, though the total number of connections using the connection information set will not exceed the repeat value.

To define a Connection Information Set

1 Select the relevant test project.

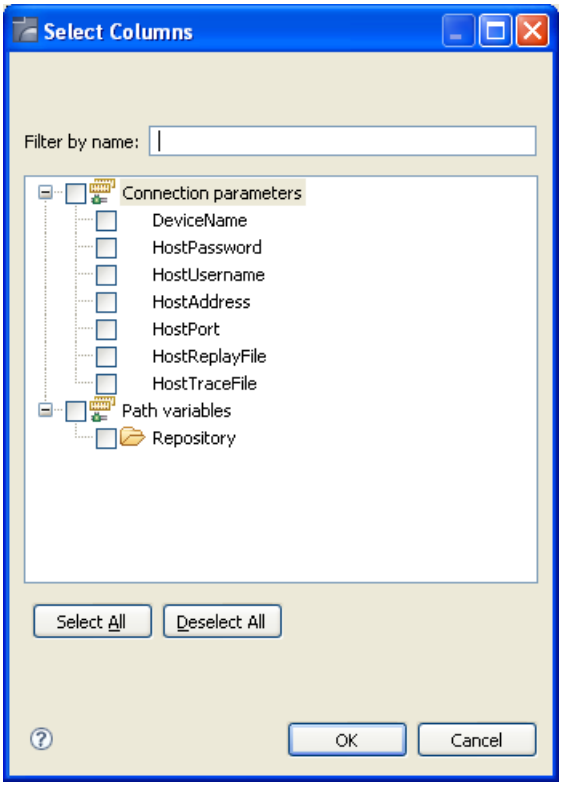
2 In the **File** menu select **New>Entity>Connection Information Set**. The *New Connection Information Set wizard* is displayed.

3 Enter a name for the Connection Information Set, a suitable description and determine the folder where the Connection Information Set is to be located. Click **Finish**. You have now created a new Connection Information Set in the repository and it is displayed in the Editor area.

4 Click **Add Record** to add additional rows to the table. Each row represents a connection information set record. Click on a row to edit the parameters of the row.

5 Determine the Repeat limit: In the Repeat column determine the number of times that a single record can be used concurrently. When set to zero, this record can be used an unlimited number of times.

6 Click **Define Columns** to display the Select Columns dialog box.



Select and/or remove your selection to determine the columns to be included in this Connection Information Set.

Select Connection parameters to define parameters required for initializing a session (e.g. device name, host name etc.). Select fields and variables to define parameters that are required by the initialization path used in connection pools.

Setting a password column

When a column is set as a password column the values in this column are scrambled in the database and appear as asterisks (*****) instead of the actual typed characters. In addition, the password values are not displayed in the log. Any of the columns can be set as a password column, except the first two columns (ID and Repeat) and Test Project Parameter columns. Once you set a column as a password column, you cannot change the column back again to be a non-password column.

In order to remove the password feature you must delete the column and add it again with its entire cell content.

1 Access the relevant Connection Information Set.

2 Click on Set Password.

3 In the pop-up window displayed, select the relevant column. This option may not be available for some columns: Test Project parameters, the first 2 columns (ID and Repeat) and columns that are already password columns.

13 Session Data

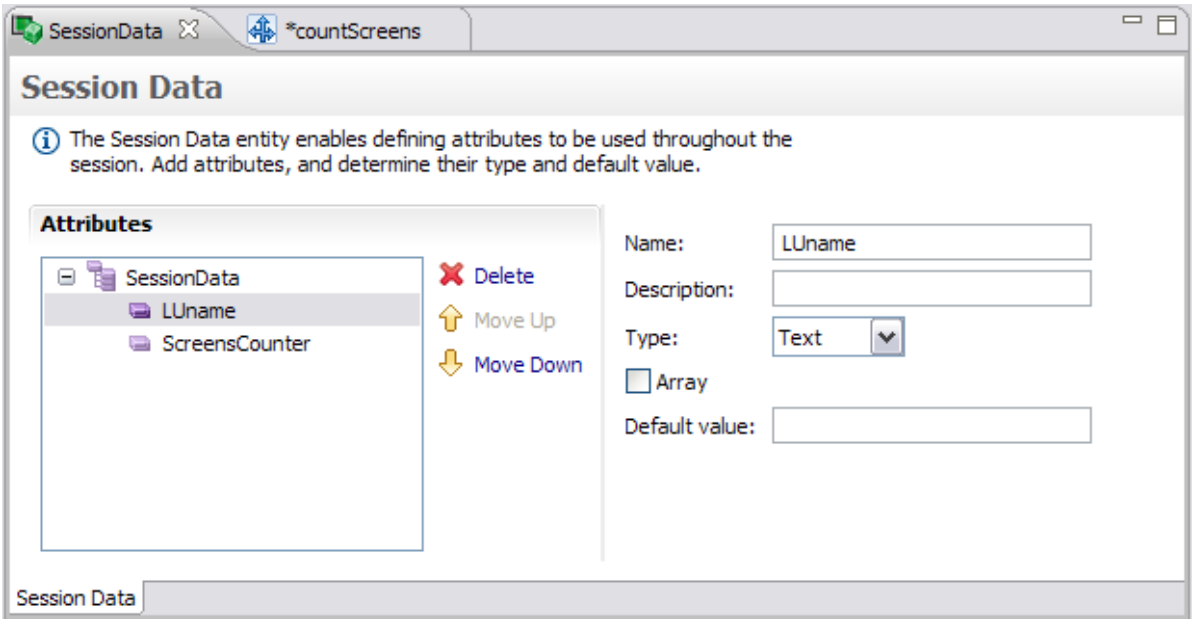
Session data is an entity used to save session context information. Each test project has a single session data entity, located in the repository folder. New test projects are created with an empty Session Data entity which cannot be deleted. The Session Data entity enables defining variables and their types in the session context during design time. At runtime, a Session Data object is created for each session, and it is possible to set and then use the variables' values in the current session. When using these variables in Connection Pools, whenever a connection is returned to the pool, the Session Data is reset with the default values.

Note: When executing a procedure using the **Execute Procedure** node, by default a new session is created for the executed procedure, therefore a new session data object is created for the executed procedure. To use the same session data object as the procedure, map the session ID to the executed procedure ID.

The Session Data entity can be used in Test Case Procedure mappers.

To define variables and types of variables

1 Within the Repository node of the test project, double-click on **SessionData**. The Session Data entity is displayed.



2 Click **Add Attribute**. An attribute is added to the Session Data.

3 Provide a name for the attribute and a relevant description.

4 Select the type of attribute: Text, Long, Boolean, Double, Integer, Float, Byte or Date.

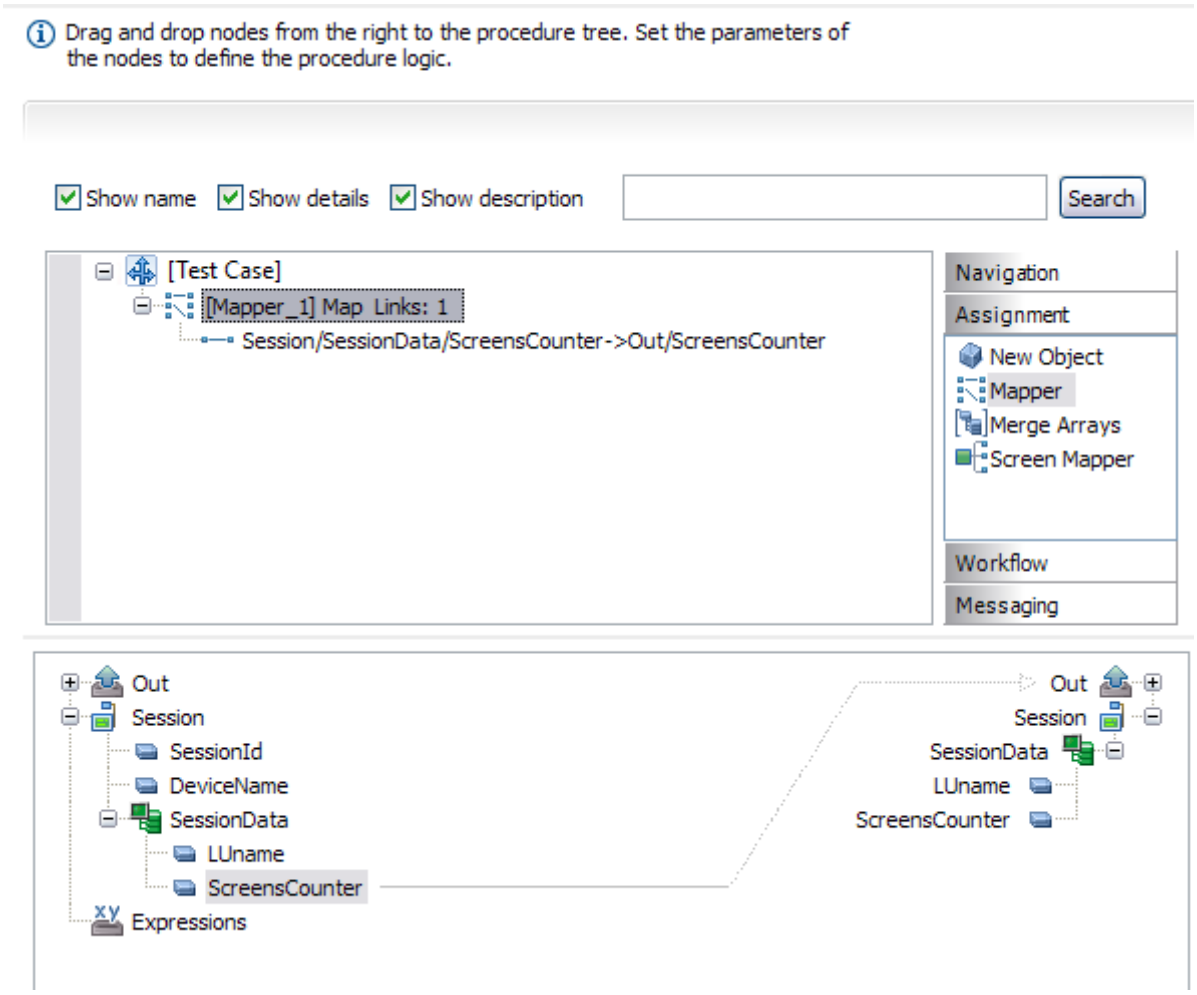
5 Determine if it is an array.

6 Enter a default value.

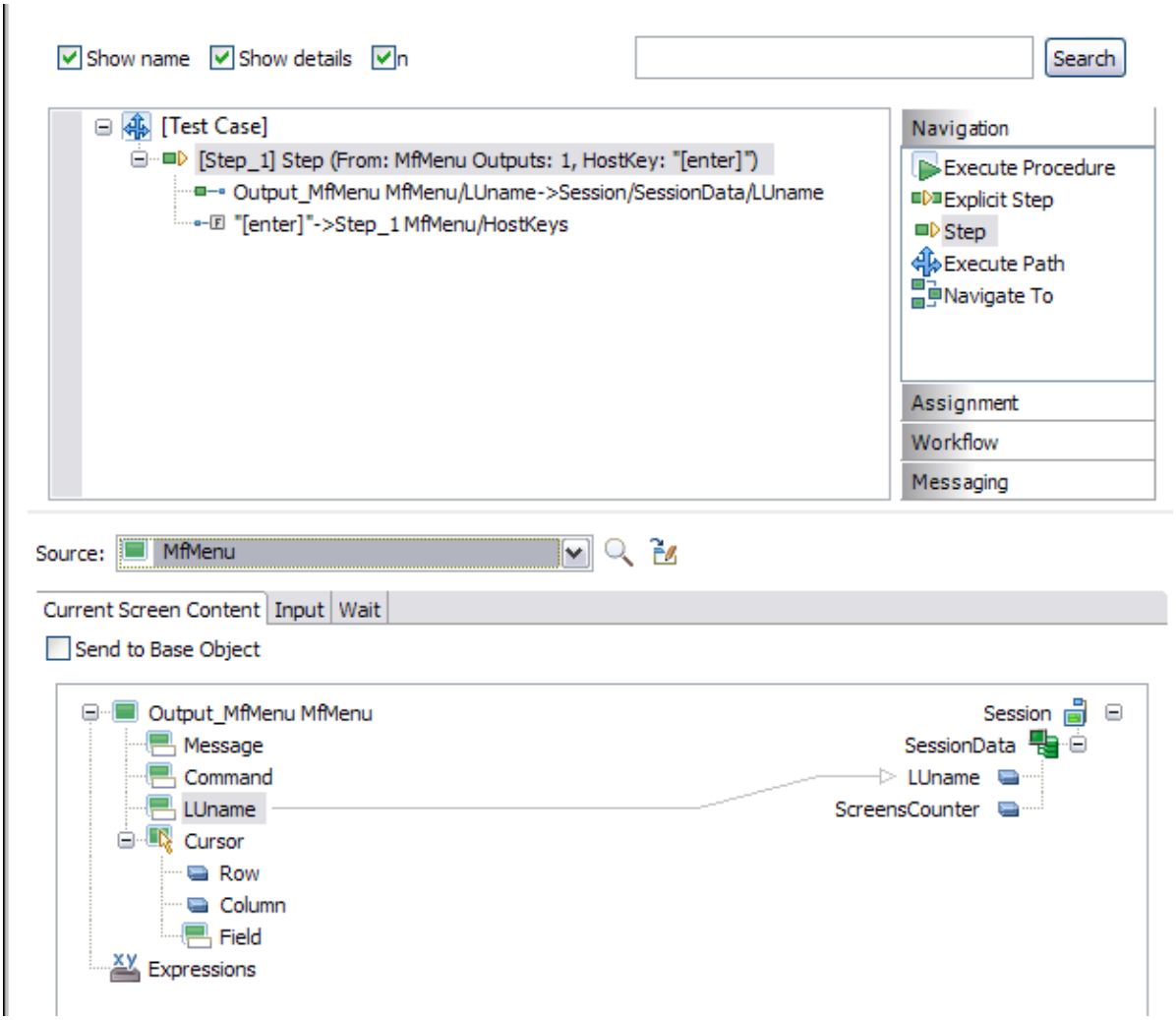
To set and use the variables

- Setting and using the values is implemented by mapping to or from the session data variables (see **Using the Mapper to Map Source Elements to Target Elements**).

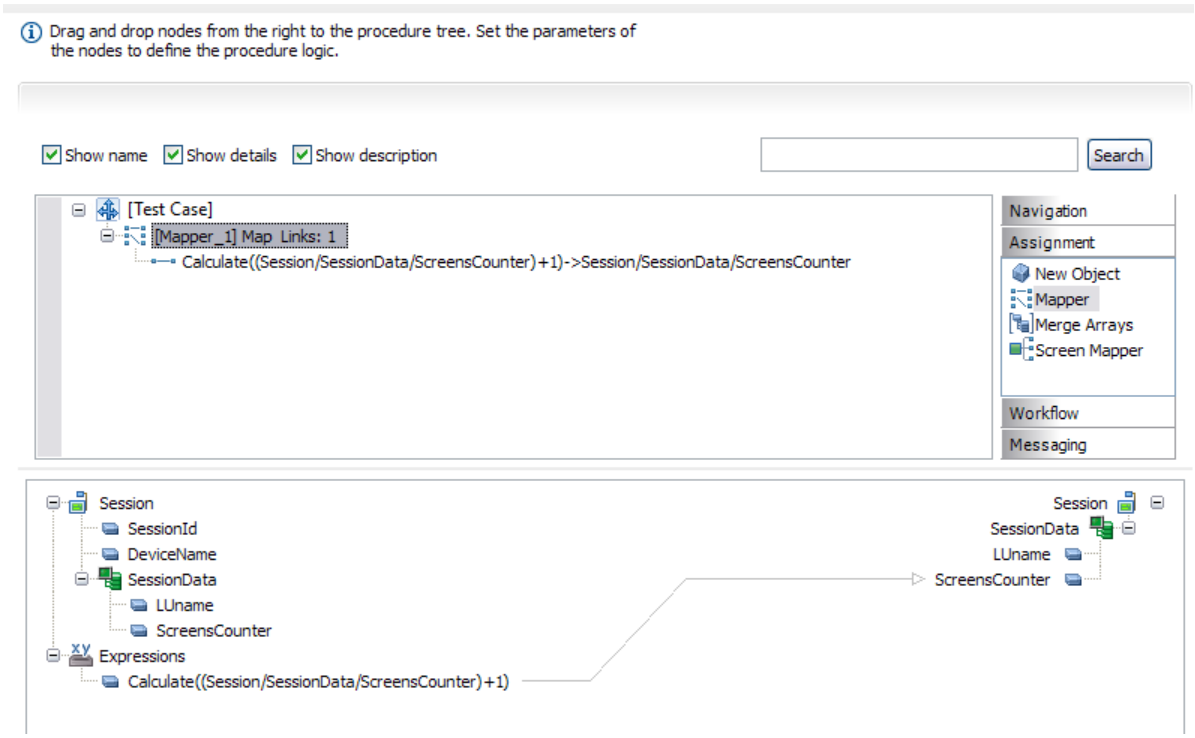
  The following illustrates mapping from the Session Data to the Procedure's outputs.



The following illustrates mapping from the screen's field content to the Session Data



The following illustrates mapping from an Expression which uses the Session Data, to the Session Data.


Drag and drop nodes from the right to the procedure tree. Set the parameters of the nodes to define the procedure logic.
Show name
Show details
Show description
Search
[Test Case]
[Mapper_1] Map Links: 1
Calculate((Session/SessionData/ScreensCounter)+1)->Session/SessionData/ScreensCounter
Navigation
Assignment
New Object
Mapper
Merge Arrays
Screen Mapper
Workflow
Messaging
Session
SessionId
DeviceName
SessionData
LUname
ScreensCounter
Expressions
Calculate((Session/SessionData/ScreensCounter)+1)
Session
SessionData
LUname
ScreensCounter

14 Working with JSON Files

Introduction

JSON is a text file that can convert to any JavaScript object and can be sent to a server. You can also convert any JSON received from the server into JavaScript objects. This way you work with the data as JavaScript objects, with no complicated parsing and translations. In Natural Screen Tester, you could generate the test case with initial test file *Data.json* file containing the default inputs and assertions given by the test case procedure. For example, the following input and output parameters:

```
{
"inputs":
   {
   "Code":"cu"
   "Password":"demoy2"
   "UserId":"demo02"
   "structure1":
      {
      "Birthday":
      "Customer_ID":
      "First_Name":
      "Last_Name":
      "Sex":
      "TotalPoliciesAmount":
      "Type":
      }
   }

"Assertions":
    {
    "Firstname":"Charles</Firstname>
    "House_Number":"12</House_Number>
    "Lastname":"Chappell</Lastname>
    "Mobile":
    "Municipal_Code":
    "Nationality_Long":"British"
    "Nationality_Short":"UK"
    "Occupation_Long":"Actor"
    "Occupation_Short":"81177"
    "PO_Box":
    "Person_Sex>M</Person_Sex>
    }
} ↩
```

Working with Arrays

You can also provide input using an array. The `runTest` method is provided for sending inputs from a given JSON file to the Base Object and verifing the assertions from the JSON file, using the response data.

Each input entry in the JSON file is converted to `GXElement` so that the test case can run. Each output attribute given by the Base Object response is converted to a JSON object and compared with the assertion loaded from the resources folder automatically.

JSON Configuration File

The JSON configuration file enables you to change the following values for excuting the tests:

```
{
   "serverPort": "2323",
   "serverAddress": "localhost",
   "nsrUser": "administrator",
   "nsrPassword": "",
   "hostApplication": "SAG_MainFrame"
}
```

Parameter `hostApplication` is mandatory; all others are optional. If a parameter is not defined, the test will try to run with the default value.

If parameter `hostApplication` is not defined, the test will run on actual application host.

Working with Multiple JSON Files

The following mechanisms are provided to enable you to work with multiple JSON files:

* (asterisk)	any files
`f1.json`, `f2.json`...	list of files

The following rules apply:

- The file extention must be ".json".
- The list must not be recursive.
- The file name must be valid for the operating system.

Example:

```
public static Collection <Object[]> getTestData() {
    String[] fileNames = {"file1.json", "file2.jsOn"};
     return TestCase.getTestData(fileNames);
}
```

The default filter for test data is "*".

Comparing the Results of a Unit Test

If a failure occurs in a unit test, you can use a compare function to easily locate the source of failure.

To compare the actual results of unit test with the expected results

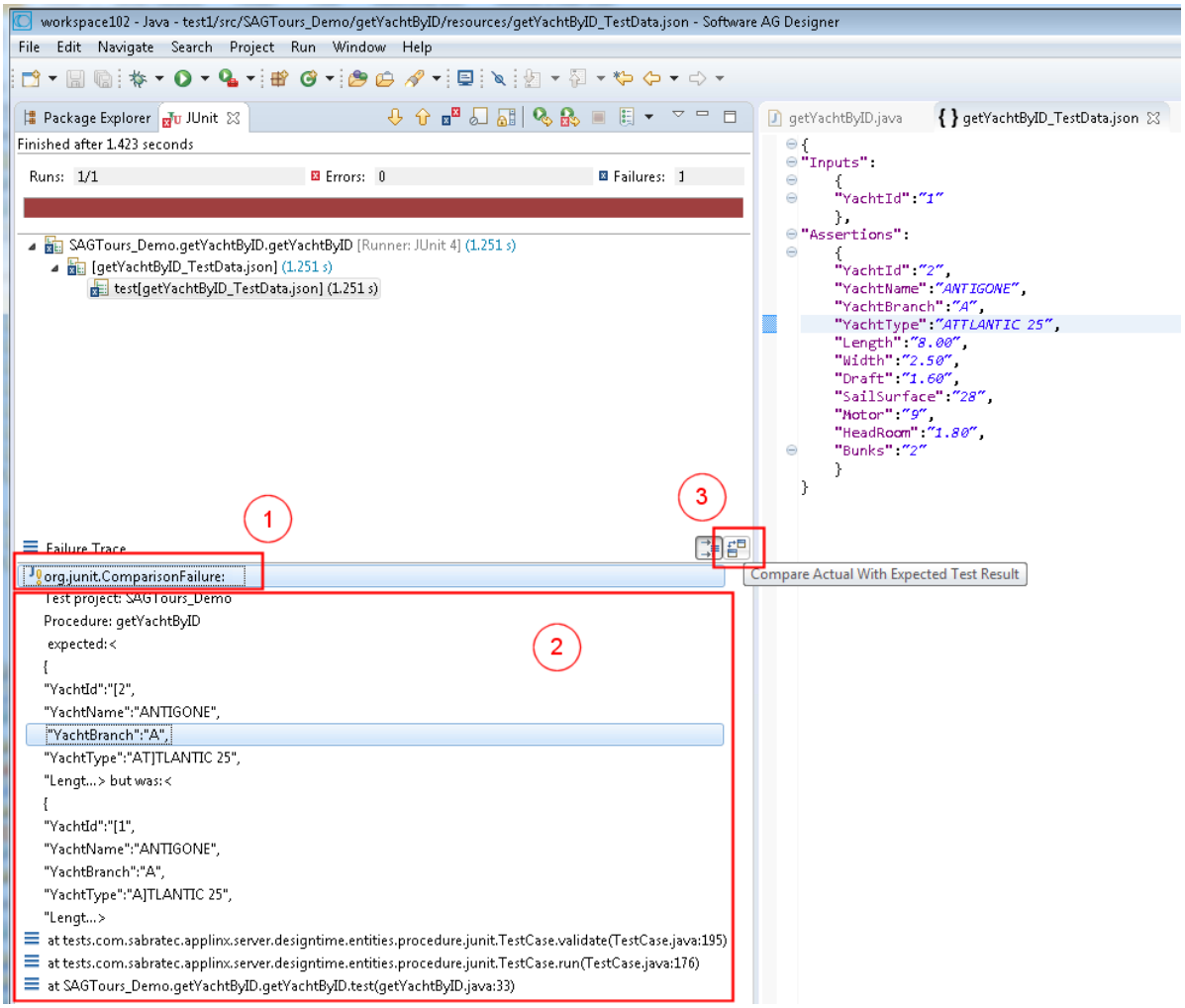
- Double click the line `org.junit.ComparisonFailure` in the JSON file. See (1) in screenshot below.

  Or:

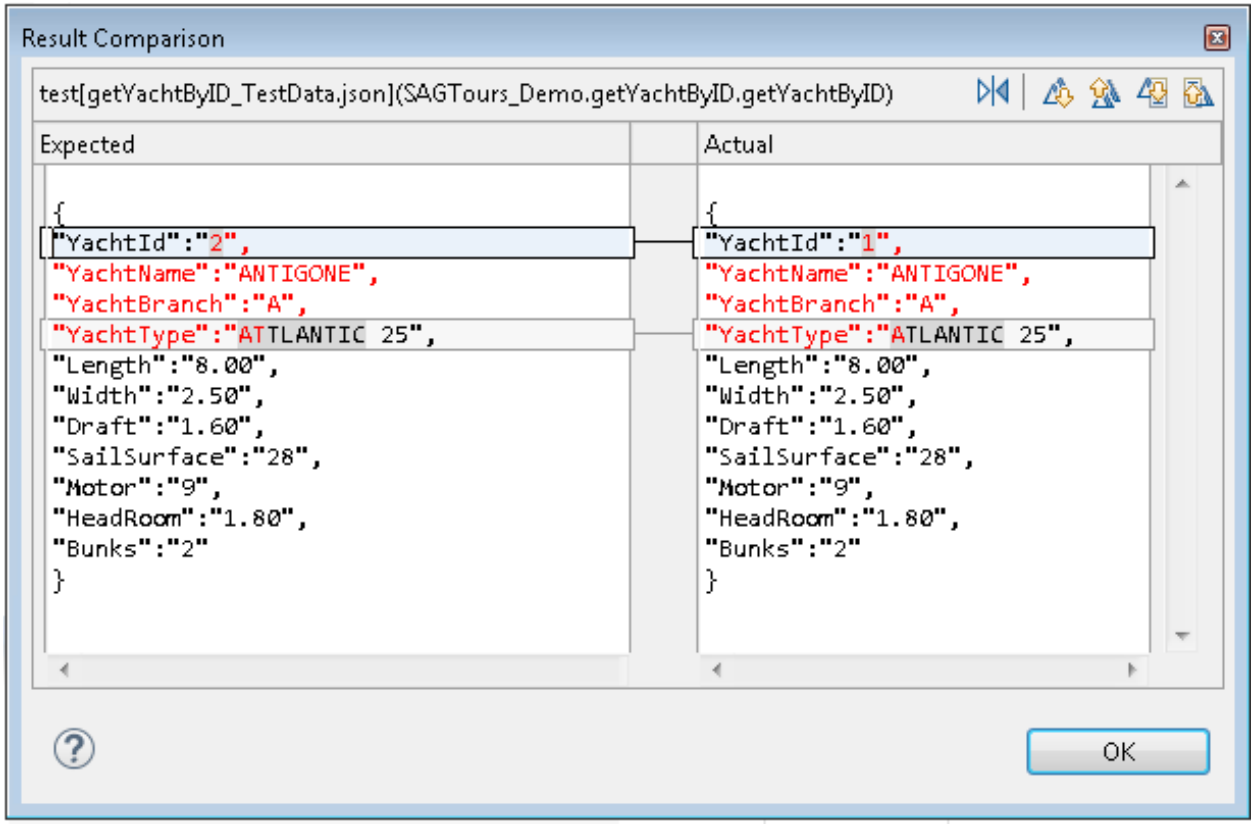
  Select the line in the marked rectangle area (2) in the screenshot below, and from the context menu choose **Compare Actual With Expected Test Result**.

  Or:

  Select the line and click the **Compare** icon  . See (3) in the screenshot below.



This produces the following output, where the discrepancies are highlighted.


Result Comparison
test[getYachtByID_TestData.json](SAGTours_Demo.getYachtByID.getYachtByID)
Expected
Actual
{
"YachtId":"2",
"YachtName":"ANTIGONE",
"YachtBranch":"A",
"YachtType":"ATTLANTIC 25",
"Length":"8.00",
"Width":"2.50",
"Draft":"1.60",
"SailSurface":"28",
"Motor":"9",
"HeadRoom":"1.80",
"Bunks":"2"
}
{
"YachtId":"1",
"YachtName":"ANTIGONE",
"YachtBranch":"A",
"YachtType":"ATLANTIC 25",
"Length":"8.00",
"Width":"2.50",
"Draft":"1.60",
"SailSurface":"28",
"Motor":"9",
"HeadRoom":"1.80",
"Bunks":"2"
}
OK

15 Batch Automation Utilities

You may be in a situation where a GUI interface is not available. This may be a common problem when a UNIX station is used. The batch files can be found in the utilities directory in Natural Screen Tester installation directory.

Importing Screens using a Batch File

Standard maps, such as Natural, BMS and MFS, are used in host test projects and include the screen data such as static data and dynamic fields. Natural Screen Tester enables importing these test project maps, saving time and effort spent on manually identifying screens and simplifying the update process when changes are made in the host. When importing test project maps, a screen is automatically created from each map, minimizing errors that may occur when creating the screens manually, one by one. The Natural Screen Tester screen created includes identifiers (based on the static data) and fields (based on the dynamic data). Natural Screen Tester supports a number of different types of maps:

- Natural: Natural map support (from Systrans file).
- BMS: CICS basic map support.
- MFS: IMS message format service.
- SDFX: Natural Screen Tester generic map format, used for other standard maps. To create SDFX files refer to *SDFX File Format Definition*.
- SDF: Compatible with Software AG's JIS product.

The import map feature can be used to import an test project's maps for a new test project or to maintain and update previously imported maps. When updating previously imported maps, screen identifiers will be deleted and replaced, existing fields will be updated with their new positions and their references to other entities will be preserved. Fields that were previously imported, but no longer exist on the host will be deleted.

Note: Invalid entity names, such as names which include invalid characters such as "#" or begin with a digit, will be automatically corrected by omitting the invalid characters.

Maps can be imported either using the Import Host Screen Maps wizard or using a batch file.

To import screens via a batch file (using the command prompt window)

1 Open a command prompt window.

2 Change the current directory to the <nsr_home>/Utilities directory.

3 Type screen_import.bat/sh followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

Parameter	Description	Default
-s	Server address	127.0.0.1
-p	Server port	3323
-u	Natural Screen Tester user name (Required parameter)	
-w	Natural Screen Tester user password	Empty by default
-a	Natural Screen Tester test project name (Required parameter)	
-f	File name, or directory name (when importing more than one file). Required parameter	
-x	The file extension. All files from the given directory that have this extension will be loaded (when not specified, the default extension for the map type is used)	
-af	Natural Screen Tester target folder within the repository.	Root folder
-t	Map type. Possible values: "sdf", "sdfx", "natural", "bms", "mfs" (required parameter).	natural
-m	Indicates where the error line is located: "first", "last", "lastm1" (last minus 1), "lastm2", "lastm3", "lastm4" (Natural maps only)	last
-mf	Message line field name (Natural maps only)	MessageLine
-k	Don't skip map with write command. (Natural maps only)	true (skip)

The screens created appear in the directory you determined in the **Target folder** field. The names of the screens are identical to the map names.

The report is displayed in the Eclipse console and includes a list of the screens added as well as the fields and tables created/updated/deleted.

Export Batch File

To activate the Export batch file by using the command program prompt

1 Open a command prompt window.

2 Change the current directory to the <nsr_home>/Utilities directory.

3 Type exportapp followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

exportapp Parameters

Parameter	Description	Default Value
-s	Server address	127.0.0.1
-p	Server port	3323
-u	User name	
-w	Password	
-a	Test project name	
-f	The target folder and/or file name. ■ When only the path is specified (<pathname>followed by "\"), the file name is the test project name. ■ When only the file name is specified, the file is created in the current location.	<current location>\<test project name>
Include one of the following parameters:		
-c	Export only the test project configuration (gxar file)	
-e	Export only entities (gxz file)	
-l	Export the test project configuration and the entities (gxar file)	

Examples for the -f parameter

- **Specify target folder and file name:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -f C:\<SAG_install_dir>\NaturalScreenTester\utilities\export\MyTestProject -c`

  The file MyTestProject.gxar is created in the C:\<*SAG_install_dir*>\NaturalScreenTester\utilities\export\ direrctory.

- **Specify target folder only:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -f C:\<SAG_install_dir>\NaturalScreenTester\utilities\export\ -c`

  The file InstantDemo.gxar is created in the C:\<*SAG_install_dir*>\NaturalScreenTester\utilities\export\ directory

- **Specify file name only:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -f MyTestProject -c`

  The file MyTestProject.gxar is created in the current local directory.

- **Default when the parameter is not specified:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -c`

  The file InstantDemo.gxar is created in the current local directory.

Examples for the -c, -e and -l parameters

- **Export test project configuration only:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -f C:\<SAG_install_dir>\NaturalScreenTester\utilities\export\ -c`

  The file InstantDemo.gxar is created.

- **Export entities only:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -f C:\<SAG_install_dir>\NaturalScreenTester\utilities\export\ -e`

  The file InstantDemo.gxz is created.

- **Export test project configuration and entities:** `exportapp.bat -s localhost -p 3323 -u administrator -a InstantDemo -f C:\<SAG_install_dir>\NaturalScreenTester\utilities\export\ -l`

  The file InstantDemo.gxar is created.

Import Batch File

To activate the Import batch file by using the command program prompt

1 Open a command prompt window.

2 Change the current directory to the <nsr_home>/Utilities directory.

3 Type importapp followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

```
importapp.bat/sh [-a [attribute]] [-s [attribute]] [-p [attribute]] [-o [attribute]]
[-u [attribute]] [-f [attribute]] [-w [attribute]]
```

Note: To importing a complete Natural Screen Tester test project, you will require a gxar (archive) file. This file includes the test project configuration, Natural Screen Tester entities (as a read only <gxz> file) and a trace file. When importing only the test project's entities, you require the gxz file only.

importapp Parameters

Parameter	Description	Default Value
-a	Natural Screen Tester test project name [required].	
-s	Natural Screen Tester Server address.	localhost
-p	Natural Screen Tester Server port.	3323
-o	One of the following operations can be performed: ■ x - Import entities from a gxz file, overriding conflicting entities. ■ c - Import test project from a gxar file, retaining the existing host configuration. The repository will be read-only. ■ h - Import test project and host configuration from a gxar file (overriding existing host configuration). The repository will be read-only. ■ r - Import test project and host configuration. When importing, retain the repository configuration (import the gxz within the gxar, to the currently configured repository). ■ hr - Perform both 'h' and 'r' operations. When not set, entities and/or configuration will be imported to an existing test project when one exists, or to a new test project when there is no existing one.	
-u	Natural Screen Tester user [required].	
-f	The path and name of the gxz/gxar file [required].	
-w	Natural Screen Tester user password.	

Note: The Session Data entity will be merged with the existing Session Data entity. When there is a conflict between the imported to the existing Session Data entity, your selection in this checkbox will determine how the Session Data entity will be.

Examples:

```
importapp.bat -u Administrator -a tp2 -f c:\entities.gxz -o x
```

This command imports entities into the "atp2" test project (only if it exists), overriding any conflicting entities within the test project.

```
importapp.bat -u Administrator -a tp1 -f c:\app.gxar -o c
```

This command imports the host and test project configuration, from the provided gxar file as a new test project. The repository will be read-only.

```
importapp.bat -u Administrator -a tp1 -f c:\entities.gxar -o h
```

This command retains the existing host configuration, and imports the "tp1" test project (entities and configuration) from the provided gxar file, overriding any conflicting entities within the test project. The repository will be read-only.

```
importapp.bat -u Administrator -a tp1 -f c:\entities.gxar -o r
```

This command imports the host configuration, and the "tp1" test project (just the configuration) from the provided gxar file, and imports the entities into the repository, using the existing repository configuration. The repository will not be read-only.

```
importapp.bat -u Administrator -a tp1 -f c:\entities.gxar -o hr
```

This command imports the host configuration, retains the test project configuration, and imports the "tp1" test project entities from the provided gxar file to the existing repository, overriding any conflicting entities within the test project. The repository will not be read-only.