



FOUR Js
The Power of Simplicity

Four Js Genero Desktop Client User Guide



Contents

Genero Desktop Client User Guide.....	4
What's new in Genero Desktop Client, v 3.00.....	5
General.....	6
What is the Genero Desktop Client.....	6
Starting and Configuring the GDC.....	6
Start the Genero Desktop Client.....	6
Configure the Genero Desktop Client.....	7
Apply an additional configuration file.....	17
GDC configuration file directories.....	17
Applications.....	19
The Shortcut System.....	19
Creating Shortcuts using the Shortcut Wizard.....	19
Shortcut Management.....	27
Shortcuts and environment variables.....	27
Customizing your Login Box.....	28
The Connections Panel.....	30
The Terminals Panel.....	31
The Debug Panel and Logging System.....	33
Using the Logging System.....	33
Viewing the Debug Console.....	34
Features.....	36
The Command Line.....	36
Command line options.....	36
Command line examples.....	40
Printing a screen shot.....	41
Local actions.....	42
List of local actions.....	42
Localization encoding list.....	44
Accessibility.....	45
Upgrading.....	46
New features of Genero Desktop Client.....	46
What's new in Genero Desktop Client, v 3.00.....	46
What's new in Genero Desktop Client, v 2.50.....	47
Genero Desktop Client 2.40 New Features.....	48
Genero Desktop Client 2.32 New Features.....	53
Genero Desktop Client 2.30 New Features.....	54
Genero Desktop Client 2.22 New Features.....	66
Genero Desktop Client 2.21 New Features.....	69
Genero Desktop Client 2.20 New Features.....	76

Upgrade Guides for the Genero Desktop Client.....	86
Genero Desktop Client 3.0 upgrade guide.....	86
Genero Desktop Client 2.5x upgrade guide.....	87
Genero Desktop Client 2.4x upgrade guide.....	87
Genero Desktop Client 2.3x upgrade guide.....	88
Genero Desktop Client 2.2x upgrade guide.....	90
Security.....	96
Security levels.....	96
Security level 0.....	96
Security level 1.....	96
Security levels 2 and 3.....	96
Security Connection Message.....	97
GDC and SSH.....	98
GDC and SSH overview.....	98
GDC and SSH prerequisites.....	99
GDC and SSH simple setup.....	100
Port Forwarding and Firewalls.....	101
Port forwarding.....	102
Port forwarding and the client-side firewall.....	107
Port forwarding and the server-side firewall.....	110
Implementing a Secure Server with GDC.....	116
Prerequisites.....	117
Solutions overview.....	117
The shell script.....	118
Setup SSH login.....	119
Setup telnet.....	122
Password management.....	124
AUTOPORTFIND source code example.....	126
Login script.....	130
SSH Configuration Troubleshooting.....	131
Wireless systems.....	132
Need to change the port that GDC listens on.....	132
Sessions expiring.....	132
GDC and Windows [™] XP Service Pack 2.....	132
GDC and Windows [™] XP Service Pack 2 Firewall Configuration.....	133
GDC and Windows [™] Vista.....	134
User Account Control.....	134
Genero Desktop Client installation on Windows [™] Vista.....	135
Running the Genero Desktop Client on Windows [™] Vista.....	135
Genero Desktop Client features affected by the User Account Control.....	136
Front End Extensions.....	138
Terminology.....	139
General terms.....	139
Security terms.....	140
Legal notices.....	142

Genero Desktop Client User Guide

Manual organization at a glance.

General on page 6	Applications on page 19	Features on page 36	Upgrading on page 46
<ul style="list-style-type: none"> • What's new in Genero Desktop Client, v 3.00 on page 5 • What is the Genero Desktop Client on page 6 • Starting and Configuring the GDC on page 6 	<ul style="list-style-type: none"> • The Shortcut System on page 19 • Customizing your Login Box on page 28 • The Connections Panel on page 30 • The Terminals Panel on page 31 • The Debug Panel and Logging System on page 33 	<ul style="list-style-type: none"> • The Command Line on page 36 • Printing a screen shot on page 41 • Local actions on page 42 • Localization encoding list on page 44 • Accessibility on page 45 	<ul style="list-style-type: none"> • New features of Genero Desktop Client on page 46 • Upgrade Guides for the Genero Desktop Client on page 86
Security on page 96	Front End Extensions on page 138	Terminology on page 139	
<ul style="list-style-type: none"> • Security levels on page 96 • GDC and SSH on page 98 • GDC and SSH simple setup on page 100 • Port Forwarding and Firewalls on page 101 • Implementing a Secure Server with GDC on page 116 • SSH Configuration Troubleshooting on page 131 • GDC and Windows XP Service Pack 2 on page 132 • GDC and Windows Vista on page 134 	<ul style="list-style-type: none"> • Front End Extensions on page 138 	<ul style="list-style-type: none"> • General terms on page 139 • Security terms on page 140 	

What's new in Genero Desktop Client, v 3.00

This publication includes information about new features and changes in existing functionality.

The following changes and enhancements are relevant to this publication.

Table 1: What's new in Genero Desktop Client Version 3.00

Overview	Reference
After the installation of a new version of the GDC, a popup allows you to import the configuration (shortcuts and options) of a previous version.	No additional reference.
GDC 3.00 is compatible with Genero runtime system (DVM) 3.00 and Genero runtime system (DVM) 2.50 for all connections, except for an HTTP connection through the Genero Application Server (GAS). When using an HTTP connection through the GAS: <ul style="list-style-type: none"> GDC 3.00 should use uaproxy (ua) and requires a Genero runtime system (DVM) 3.00. GDC 2.50 should use gdcproxy (ja) and requires a Genero runtime system (DVM) 2.50. 	See the <i>Genero Installation Guide</i> for more information.
Genero Desktop Client 3.00 supports Internet Protocol version 6 (IPv6), in addition to Internet Protocol Version 4 (IPv4), when: <ul style="list-style-type: none"> Using a web server (connected to a GAS). Using port forwarding through an ssh tunnel. <p>However, as DVM does not support IPv6, you cannot launch an application on a distant host with a GDC listening, using direct connection.</p>	See Port forwarding on page 102 and the Creating Shortcuts using the Shortcut Wizard on page 19 section.
GDC configuration files, supporting files, and cached files are now written to the User directory, providing each user with their own configuration settings (amongst other things) by default.	See GDC configuration file directories on page 17.
The Connections panel now displays information about cookies.	See The Connections Panel on page 30.
Chinese translation is now available.	No additional reference.
HTTP protocol enhancements: <ul style="list-style-type: none"> Support of single sign-on (SSO) mechanism. Support of auto logout. 	See the <i>Genero Application Server User Guide</i> for more information on the HTTP protocol.
Web component enhancements: <ul style="list-style-type: none"> Support of URL-based web components. Support of <code>call</code> frontcall. 	No additional reference.
When copying data, you can now select part of the text of a non-editable field.	No additional reference.
Fgltty is now based on Putty 0.65	No additional reference.
GDC is now based on Qt 5.5	No additional reference.

General

These topics introduce you to the Genero Desktop Client and provide guidance for starting and configuring this front-end.

- [What is the Genero Desktop Client](#) on page 6
- [Starting and Configuring the GDC](#) on page 6

What is the Genero Desktop Client

The Genero Desktop Client is a graphical front-end for a Genero application.

The Genero Desktop Client is multi-platform and can run under Windows™ systems, Mac OS, and Linux®.

For a detailed list of supported operating systems, refer to the System Support matrix (available on the Four Js Web site in the product and documentation download area) or contact your support center. This matrix also informs you which operating systems will no longer be supported as of the next release.

Starting and Configuring the GDC

Before displaying an application using the Genero Desktop Client as the front-end, you may need to configure and/or start the client.

- [Start the Genero Desktop Client](#) on page 6
- [Configure the Genero Desktop Client](#) on page 7
- [Apply an additional configuration file](#) on page 17

Start the Genero Desktop Client

The method used to start GDC is based on your operating system.

To start the GDC:

- Under Windows™ systems, you can use the shortcut on the Start Menu.
- Under X11 systems, you can also use the shortcut on the Start Menu, or performing `envgdc` shell will add the Genero Desktop Client binary directory to your path; you will be able to start with the following command : `gdc`.
- Under OS X systems, the installer will add GDC to the **Applications** folder. You can also create a desktop shortcut to launch GDC from the command line. See [Create a Genero Desktop Client desktop shortcut on OS X](#).

By default, GDC will listen for [Runtime System](#) connections on port 6400. You can specify the port by starting GDC with the parameter `-p`.

If the port is not available, GDC will try the next port, continuing until it finds the first available one.

See [command line](#) for a list of all command line options.

Create a Genero Desktop Client desktop shortcut on OS X

Create a desktop shortcut for the Genero Desktop Client (GDC) on the Mac OS X operating system.

1. Go to **Applications/Utilities > AppleScript > Script Editor**.
2. Complete and enter the following command:

```
try
do shell script "~/Users<USERNAME>/Applications/Genero\\ Desktop\\
Client\\"
```

```
<VERSIONNUMBER>.app/Contents/MacOS/gdc <COMMANDLINEOPTIONS>"  
end try
```

Note: For more information about the modes that you can start the shortcut in, see [command line options](#)

Option: to verify that the script is correct, click Run.

3. Save the script on the desktop as an `Application bundle`.

Configure the Genero Desktop Client

Configure the Genero Desktop Client by accessing the configuration tabs.

Click the Options icon to display the configuration options panel. The Options panel is only available in administration mode.

The configuration options are organized across tabs. After you modify a configuration, select **Apply** to validate and save your changes or **Restore** to discard your changes.

- [Preferences tab](#) on page 8
- [Advanced tab](#) on page 10
- [Connection tab](#) on page 15
- [Security tab](#) on page 16
- [Report tab](#) on page 17

Preferences tab

Use the Preferences tab to configure an images path, an icon path, and font overriding.

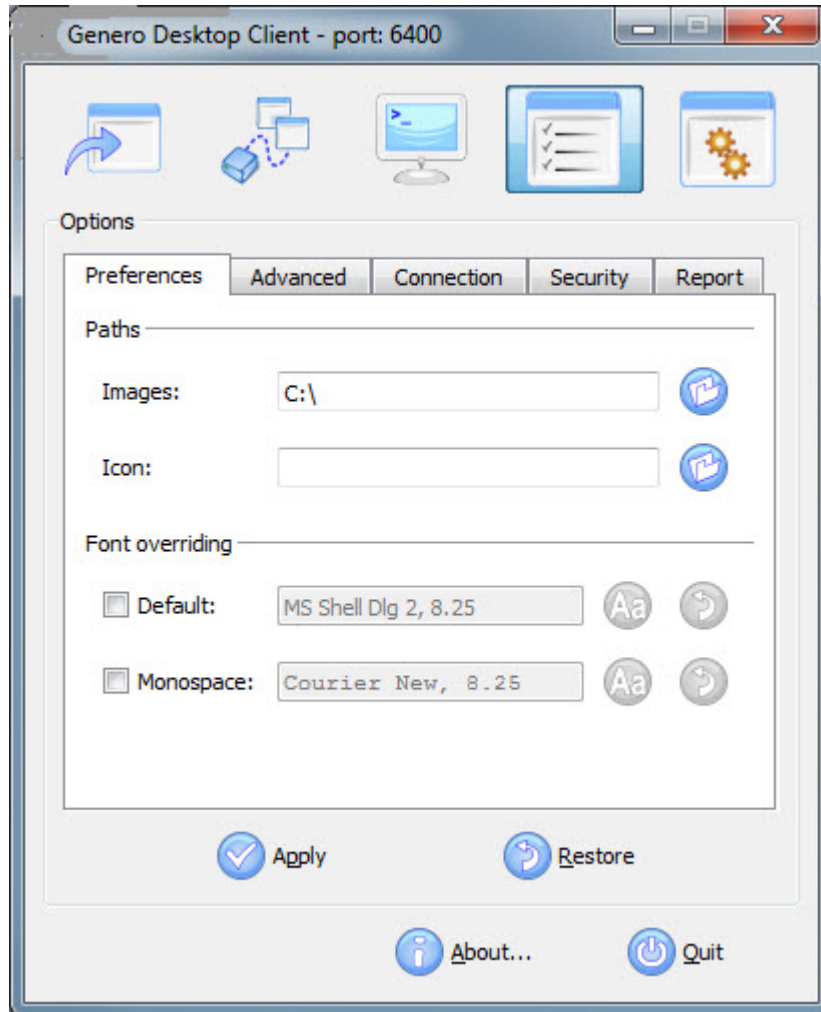


Figure 1: Genero Desktop Client Options; Preferences tab

The **Preferences** tab allows you to set path and font overriding information.

Important: Changes will not be applied until the "Apply" button is clicked.

Set path options

These paths can be set:

Images

Specifies the path for the GDC to search when an image is needed. GDC will first check if the name provided corresponds to an absolute file name; then it will look in the path you have specified. If it cannot find the image, it draws a "... picture.

Icon

Specifies the default icon for GDC. This is the default icon used for the taskbar, the systray icon (under Windows™ systems), the shortcuts, the Terminals and applications.

Note:

If you enter an invalid directory, the label turns red to warn you:

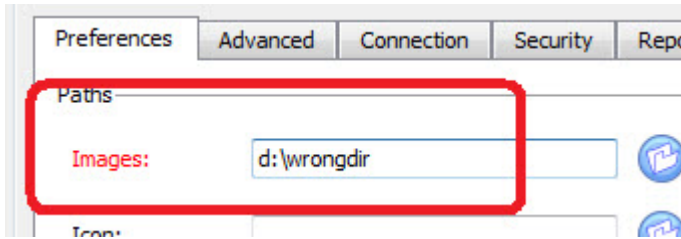


Figure 2: Images field with invalid directory entered

Most of the fields have auto completion:

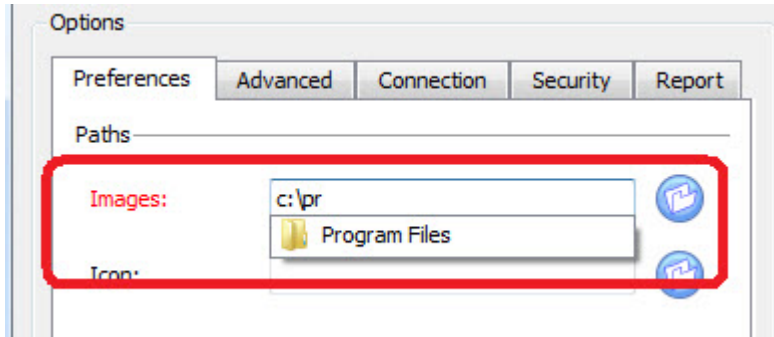


Figure 3: Images field with autocomplete feature

Set font overriding options

These font overriding options can be set:

Default

Specifies the default font for GDC. This font will be used everywhere in your applications.

Monospace

Specifies the default fixed font for GDC. This font will be used when the fixed font attribute is defined.

Advanced tab

Use the Advanced tab to configure the image cache, the stored settings, and the buttons style.

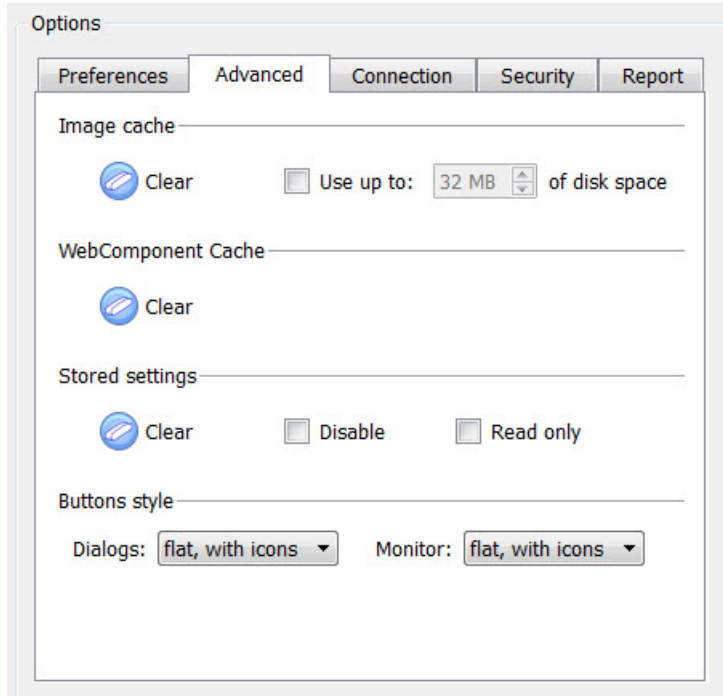


Figure 4: Genero Desktop Client Options; Advanced Tab

The following options can be configured in the *Advanced* panel:

- [Disk Image Cache](#)
- [WebComponent Cache](#)
- [Stored settings](#)
- [Buttons style](#)

Image Cache

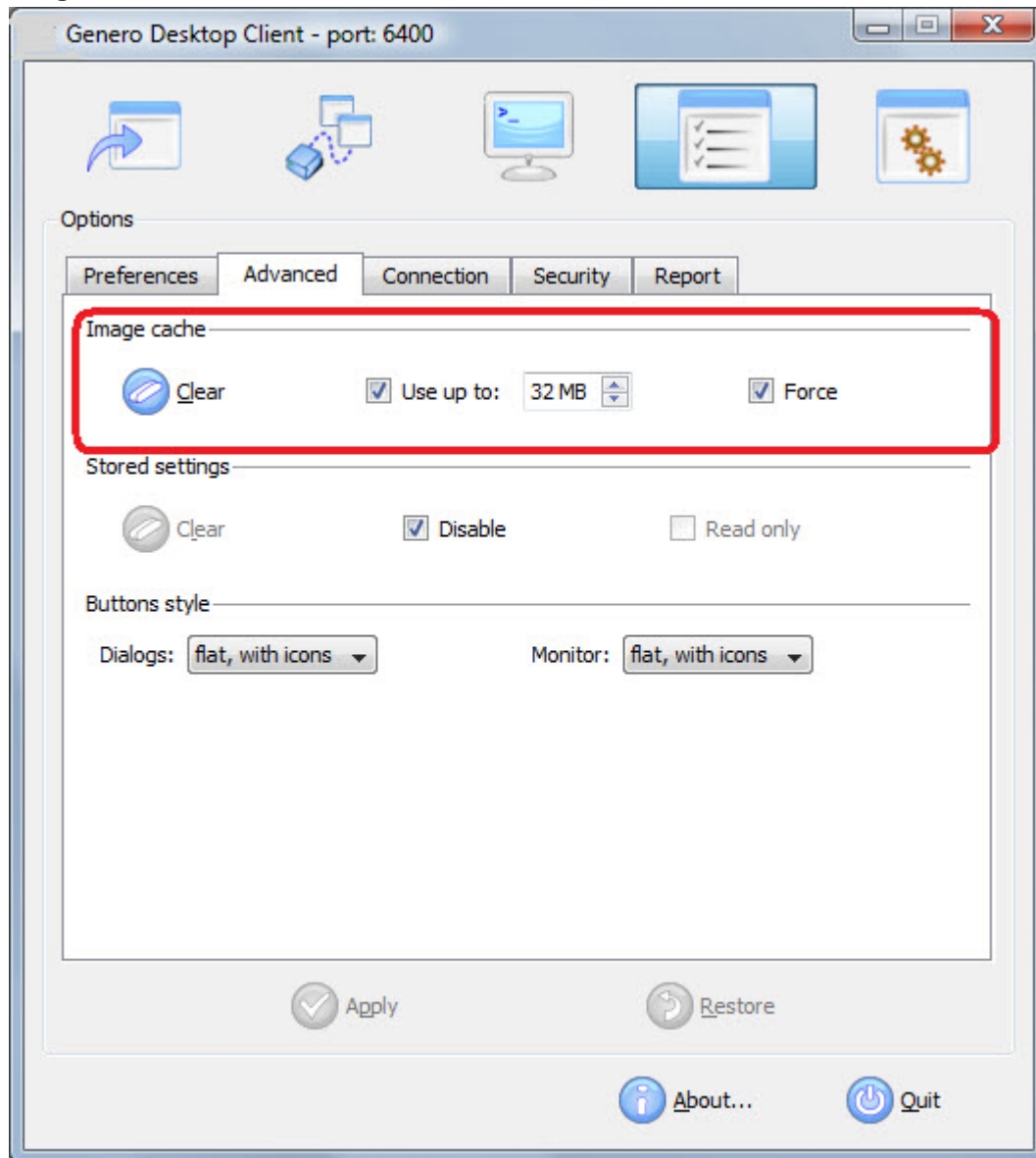


Figure 5: Image cache section of the Advanced tab

GDC will locally store images which have been retrieved remotely. This can happen when the images have been found using http (either because the url specifies http, or a PICTURE alias is used with Genero Application Server), or on the Runtime System side (using FGLIMAGEPATH). The size of the image cache can be configured. Images are stored in the `AppCacheDir` directory (see [GDC configuration file directories](#) on page 17 for more information). When the cache is full, the images which haven't been recently used are removed from the cache. The Clear button will clear the cache.

Tip: As with previous versions, a memory cache is still used by GDC. Images that are frequently used are cached to be loaded as fast as possible. The Clear Cache button will clear both caches (memory and local disk).

The `imageCache` style has been introduced to manage the cache; the disk cache and memory cache. It can be applied to any item using images and defines whether GDC must cache the image (based on the url).

Values can be `yes` (cache is used) or `no` (cache is not used). The default behavior depends (as in previous versions) on the item type:

- IMAGE fields are not cached.
- All other items (button, toolbar items) are cached.

Note: When configuration settings for the image cache in GDC are modified in one monitor, the settings are applied to all monitors for a user. For example, if you delete the image cache in one monitor, it is also deleted in all other monitors.

Note that the image cache is common to all applications for a user. This can result in an error if the cache is enabled and there are image files with the same file name. Take the following example of an error that occurs when two applications try to display two different images with the same file name:

1. The first application will write the first image to the cache.
2. The second application, while trying to load the second image with the same name, will search the cache by file name and load the first image.
3. The second application incorrectly displays the first image.

WebComponent Cache

The Clear button will clear the cache.

Stored settings

Stored Settings can be temporarily disabled by checking "Disable". If "Read Only" is checked, GDC will read the stored settings when forms are loaded, but they won't be updated when forms are closed. If you want to clear settings, click on "Clear". this button is disabled if there are no stored settings.

Attention: We **strongly recommend** that you clear stored settings when migrating to a new main release of GDC (for instance, when moving from 2.1x to 2.2x). Otherwise, you might encounter some side effects due to corrections or new functionality.

Buttons style

The look of the monitor and dialogs (shortcuts wizard, login, about box, debug console) buttons can be customized to match the look-and-feel of a regular Genero application.

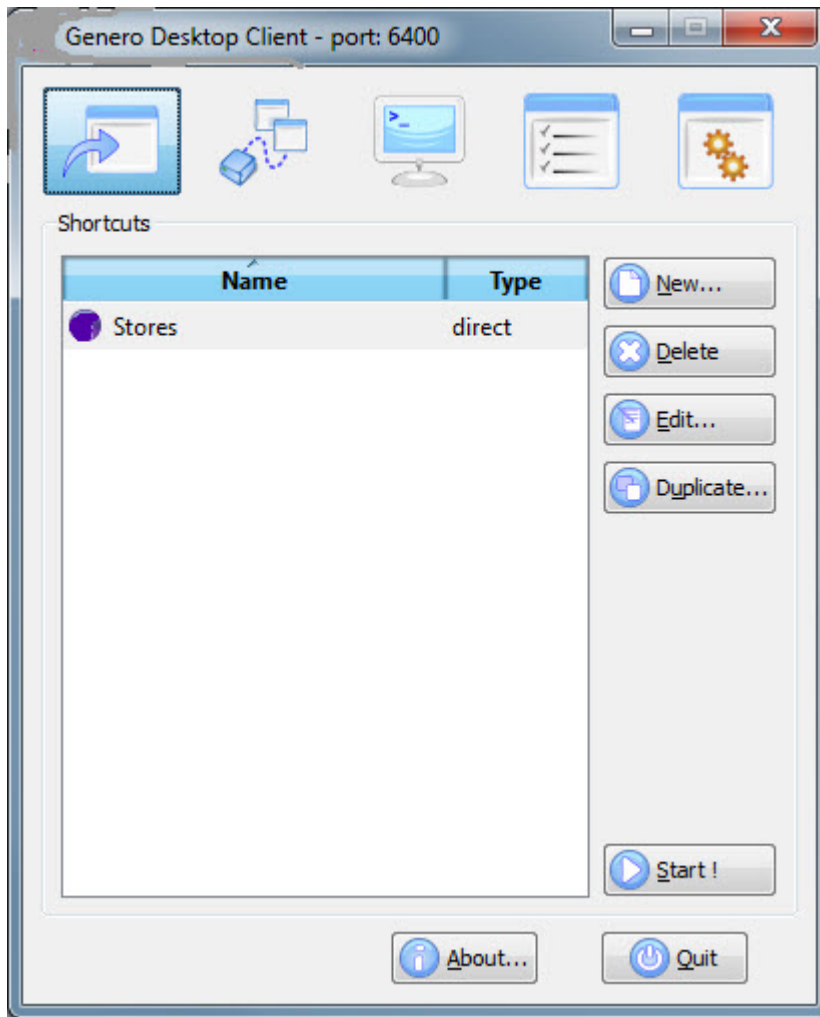


Figure 6: Raised buttons with icons

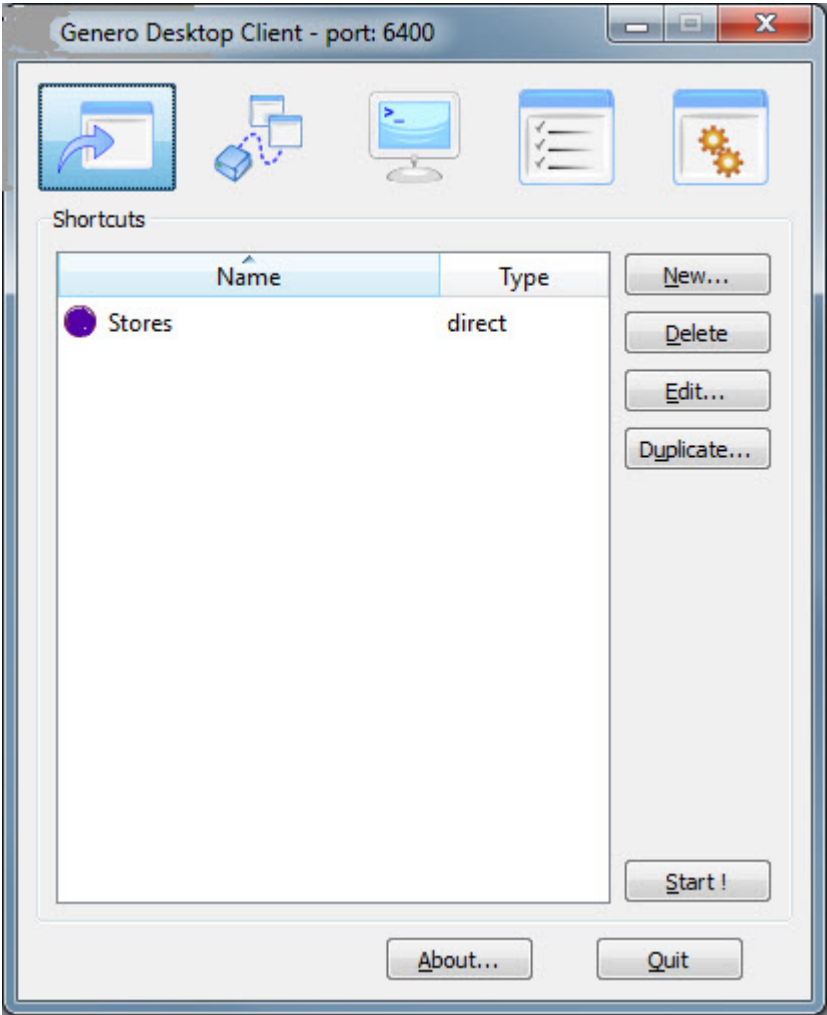


Figure 7: Raised buttons without icons

Connection tab

Use the Connection tab to configure HTTP proxy, HTTP retries, and ping events.

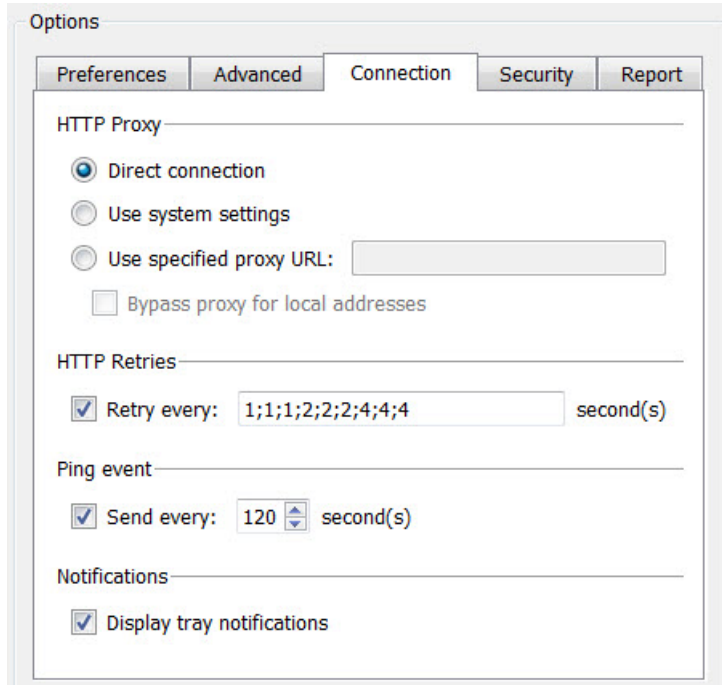


Figure 8: Genero Desktop Client Options; Connection Tab

These options can be configured in the *Connection* panel:

- [Http Proxy](#)
- [Http Retries](#)
- [Ping event](#)
- [Notifications](#)

Http Proxy

In the HTTP Proxy section, you can set up the default proxy used for:

- Http shortcut (can be overridden in each shortcut)
- Http image lookup in Direct and Local shortcut

Http Retries

In the HTTP Retries section, you configure how and when the GDC will resend the http request on socket or http error. If checked, the GDC will read the value from left to right, waiting the number of seconds entered between each separator before resending the failed request.

For example, the default value "1;1;1;2;2;2;4;4;4" means "on Socket/Http error, wait 1s before retrying, then, if the request still fail, wait 1s more, then 1s more, then 2s between each retry, then 4s between each retry".

Please note that this feature increases the time required for the detection of invalid hosts or dead servers, since the initial request will be retried at least 9 times with a total of 21 seconds to wait. You can temporarily disable it when creating a new shortcut, enabling you to quickly check the reachability of the server.

Ping event

The purpose of a ping event is to check whether the connection with the runtime system or the application server is still alive. To perform this check, GDC sends a "ping" signal over the network. By default, the signal is sent every two minutes. The interval can be changed in the Ping event section (for instance, to 300 seconds).

Notifications

Enable or disable the display of tray notifications.

Security tab

Use the Security tab to configure the security level and to clear stored passwords.

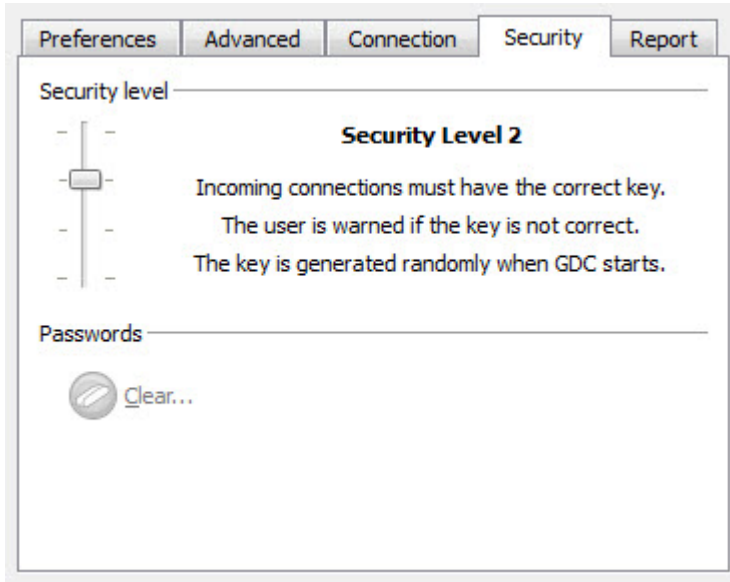


Figure 9: Genero Desktop Client Options; Security Tab

These options can be configured on the **Security** panel:

- [Security Level](#)
- [Clear Password](#)

Security Level

The security level can be changed here. See [Security](#) for more details.

Clear Password

Clears the passwords that are stored by GDC.

Report tab

Use the Report tab to configure default printer and font settings for REPORT TO PRINTER behavior.

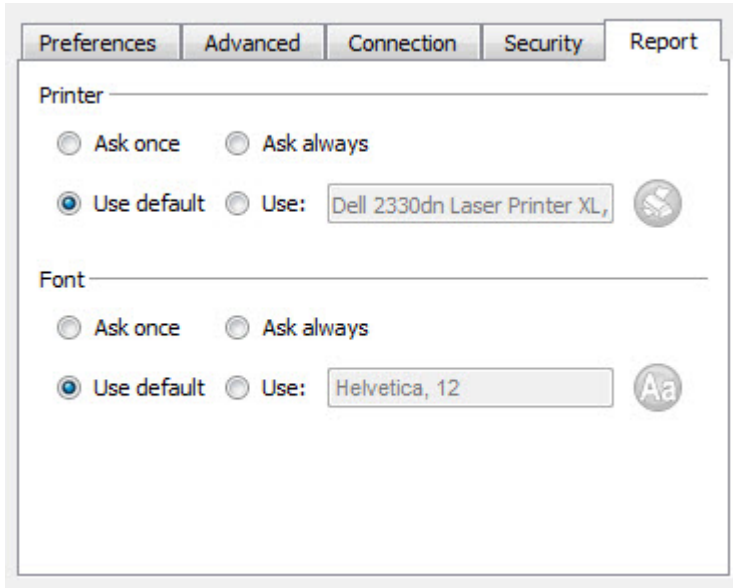


Figure 10: Genero Desktop Client Options; Report Tab

Options for the printer and font, used with REPORT TO PRINTER behavior:

- **Ask once:** The Genero Desktop Client will ask for the parameter once, and then keep the choice in memory until the Genero Desktop Client closes.
- **Ask on every report:** The Genero Desktop Client will ask every time a report is printed.
- **Use default:** Use the system default printer or the Genero Desktop Client's default font.
- **Use:** Use a specified printer or font.

Apply an additional configuration file

You can use the `--config` option to specify an additional configuration file. Configuration settings specified in this file take precedence over the configuration settings defined in the default configuration file.

The Genero Desktop Client stores most configuration options in a user-specific `config.xml` file (see [GDC configuration file directories](#) on page 17 for more information). Use an additional configuration file to override entries in the `config.xml` file. Specify the additional configuration file with the `--config` option.

Configuration options are read from:

1. The configuration file specified by the `--config` option.
2. The default `config.xml` for options not specified in the specific configuration file.

If an additional configuration file is specified, configuration changes will be stored in the additional configuration file. The default configuration file (`config.xml`) will not be altered.

GDC configuration file directories

There are two default directories where the GDC configuration files are stored: `AppDataDir` and `AppCacheDir`.

[Table 2: Configuration file directories](#) on page 18 shows the locations of the default directories for the GDC configuration files.

Table 2: Configuration file directories

Directory name	Directory location	Description
AppDataDir	<p>Windows®</p> <p>C: \Users<USERNAME>\AppData \Roaming \Four Js\gdc\ <VERSIONNUMBER>\</p> <p>Linux®</p> <p>~/.local/ share/Four Js/ gdc/<VERSIONNUMBER>/</p> <p>Mac®</p> <p>~/Library/ Application Support/Four Js/ gdc/<VERSIONNUMBER>/ Or /Library/ Application Support/Four Js/ gdc/<VERSIONNUMBER>/</p>	<p>Contains:</p> <p>AppData\sts.xml</p> <ul style="list-style-type: none"> • config.xml • webcomponent default sub-directory • dictionaries sub-directory
AppCacheDir	<p>Windows®</p> <p>C: \Users<USERNAME>\AppData \Local\Four Js \gdc\cache\</p> <p>Linux®</p> <p>~/.cache/Four Js/ gdc/</p> <p>Mac®</p> <p>~/Library/Caches/ Four Js/gdc/ or / Library/Caches/ Four Js/gdc/</p>	<p>Contains:</p> <p>AppData application cache</p> <ul style="list-style-type: none"> • images • ftcache sub-directory

Applications

These topics introduce you to the applications side of Genero Desktop Client.

- [The Shortcut System](#) on page 19
- [Customizing your Login Box](#) on page 28
- [The Connections Panel](#) on page 30
- [The Terminals Panel](#) on page 31
- [The Debug Panel and Logging System](#) on page 33

The Shortcut System

The Genero Desktop Client (GDC) can store the information needed to start an application. The information is stored as a *shortcut*.

Shortcuts are stored the same way internally on each platform. You add a shortcut for each application.

Note: You must be in the administration mode to create or modify shortcuts. By default, the Genero Desktop Client starts in *user mode*, where shortcuts and options cannot be modified.

To create or modify shortcuts as an administrator, start the Genero Desktop Client in *admin mode* by using the `--admin` or `-a` [command line](#) option.

Creating Shortcuts using the Shortcut Wizard

To assist you when setting up a new shortcut, a Shortcut Wizard is provided.

When creating a shortcut with the Shortcut Wizard, in step 1, you must choose the type of shortcut connection type that you want to use from the following 3 options:

Table 3: Shortcut connection types

Shortcut connection type	Description
Direct, connection is established through terminal emulation	With a Direct connection, the Runtime System is directly connected to the GDC using TCP/IP network. The Runtime System is on a distant host. The GDC will start it via telnet or SSH.
HTTP, through a web server	With an HTTP, through a web server connection, the GDC connects to the Runtime System via the Genero Application Server using the HTTP protocol. The Runtime System is on a distant host. The GDC will connect to it via Genero Application Server.
Local execution	With a Local execution shortcut, the Runtime System is on the same computer as the GDC. The Runtime System is on the same host. The GDC will start it as a local application.

Consider these options before you open the Shortcut Wizard. When you have decided on a Shortcut Connection, choose the corresponding task below and follow the instructions to assist you in completing the steps in the Shortcut wizard.

- [Creating a Direct Connection Shortcut](#) on page 20
- [Creating a HTTP Connection Shortcut through a web server](#) on page 24
- [Creating a Local Execution Shortcut](#) on page 25

Creating a Direct Connection Shortcut

This procedure guides you through the process of creating a Direct Connection Shortcut using the Shortcut Wizard.

To open the Shortcut Wizard, in the **Shortcuts** window, click the **New...** button.

Shortcut Wizard page 1: Shortcut identification and Connection type

1. Complete the fields of the **Shortcut identification** section:

- In the **Name** field, provide a name for the shortcut.
- Optional: In the **Icon** field, provide a file name that will be used to display an icon associated with this shortcut.
- Optional: If you want to store the shortcut locally for the current user, select the **Store shortcut in settings** checkbox.

By default, shortcuts are saved in the `&AppDataDir/config.xml` file (see [GDC configuration file directories](#) on page 17 for more information). Shortcuts written to this file are shared amongst all users who use this installation of the GDC. Selecting the **Store shortcut in settings** option is useful when the GDC is on a shared network drive and you do not want the current user to modify the common shortcuts.

Note: When the `config.xml` file is read-only, the **Store shortcut in settings** checkbox is selected by default and you do not have the option to deselect it.

Note: Any modification to a non-local shortcut displays a warning and creates a local copy of the shortcut.

2. In the **Connection type** section, select the **Direct, connection is established through terminal emulation** and click **Next**.

Shortcut Wizard page 2: Host information

- In the **Name** field, enter the hostname where the [Runtime System](#) is hosted. This can be omitted if you use the `-Host` or `-H` [command line](#) option.
- In the **Command** field, enter the command line that will be executed to start the application on the [Runtime System](#) side and click **Next**.

Within the command line, you can use the following tags:

Table 4: Tags for use at the command line

Tag	Replaced by
@FGL	<code>FGLSERVER=<IP Address>:<serv num> export FGLSERVER; FGLGUI=1; export FGLGUI</code>

Table 5: You can use one of the @FGL variants depending on your system

Tag	Replaced by
@FGLNT	<code>set FGLSERVER=<IP Address>:<serv num>&&set FGLGUI=1</code>
@FGLCSH	<code>setenv FGLSERVER "<IP Address>:<serv num>" ;setenv FGLGUI 1</code>

Tag	Replaced by
@FGLKSH	FGLSERVER=" <i><IP Address></i> : <i><serv num></i> ";export FGLSERVER;FGLGUI=1;export FGLGUI
@SRVNUM	<i><GDC listening port - 6400 (The second part of FGLSERVER)></i>
@PORT	<i><GDC listening port></i>
@USR	<i><Client current user name></i>
@LUSR	<i><Client current user name, lower case version></i>
@USER	<i><User name on the remote system></i>
@IP	<i><IP address of the client computer></i>
@COMPUTER	<i><Machine host name></i>
@E_SRV	export FGLSERVER
@4GLSRVVER	<i><GDC version></i>

These tags will automatically be replaced when the command is sent to the [Runtime System](#) host.

Additional environment variables may be set when using @FGL tags ; these variables are needed to manage [connection checks](#).

Shortcut Wizard page 3: Terminal protocol and Terminal options

5. In the **Terminal protocol** section, select from the list of options and click **Next**.

With a direct connection type, the basic mechanism (without any port forwarding configuration) is using 2 connections:

- From GDC to server: the GDC establishes the connection to a server using either the telnet protocol, the SSH protocol or the SSH2 protocol via the fgltty terminal.

Tip: The SSH2 protocol is recommended for security purposes.

- From server to GDC: the server, where the Genero environment is installed, executes a command line which starts the application on the GDC via a TCP/IP network. The IP address of the GDC is retrieved using the FGLSERVER environment variable.

Note: The telnet, SSH and SSH2 protocols are only used for establishing the first connection from GDC to server.

Using SSH or SSH2, port forwarding can be established to secure your connection. When you use this option, a SSH tunnel is created. This means that, in opposition with the basic mechanism without port forwarding, there are no longer two connections, but a single connection: when the server establishes the connection to the client, it can use the existing SSH connection to tunnel the graphical connection.

Note: While GDC 3.00 supports IPv6, as DVM does not support IPv6, you cannot launch an application on a distant host with a GDC listening using a direct connection.

6. In the **Terminal options** section, choose any of the available options and click **Next**.

The **Backspace key sends Control-H** option modifies the sequence sent by the backspace key in FGLTTY. By default, **Control-?(127)** is used but you may change it to **Control-H**. This will allow you, for instance, to use the backspace key in dbaccess.

If **Show terminal window** is checked, the window of FGLTTY, our Emulation Terminal Utility, will be visible. (Please refer to the [Terminals](#) section). This could help you check whether your command line is valid.

The **Start command in a new shell** option allows you to start a regular shell session before executing the remote host command.

Note: This option is mandatory when using **Automatic** port forwarding, which can be selected in step 4.

Shortcut Wizard page 4: Port forwarding mode

7. Select the port forwarding method you want from the list and click **Next**.

The following port forwarding options are available:

- If you select **Disabled**, port forwarding is disabled.
- If you select **Automatic**, the option to edit the Port range is provided.
- If you select **Fixed port**, you must enter the port that you want to be used.
- If you select **Command line port request**, you must enter the command to be executed on the remote host.

Note: The command must display a string in the form of *port=xxx*.

- If you select **HTTP port request**, you must enter the URL that you want to open for port forwarding port resolution.

Note: The URL body must include a terminal string in the format of *port=xxx*.

Shortcut Wizard page 5: Login form and Authentication method

8. In the **Login form** section, enter a filepath in the **Form file** field.

To use your own login form, specify the login form file to use. The form file must be a `.ui` file, which is a Qt designer's file format. See [Customizing your own login form](#). Check **Always on Top** to force the login form to always display on top.

9. Choose your **Authentication method** and click **Next**.

The Authentication method will either be Standard or Kerberos.

- The **Standard** authentication method:

In the **User** field, provide the *username* you are using to connect to the host. This can be omitted if you use the `-User` or `-U` [command line](#) option.

If **Password required** is checked, GDC will ask you for a password. If your configuration allows you to connect without a password, uncheck this option. If a password is still requested, review your configuration.

Important: GDC will not modify your configuration to allow you to connect without a password. It is up to you or your administrator to manage this.

The next two options concern the keeping of the password:

- If **Keep password** is checked, GDC keeps in memory the password you enter the first time you start a shortcut, and reuses the password when you restart. The password is stored for the session; it is kept in memory and is lost if you stop the GDC. The password is kept in memory while GDC is launched and automatically completed in the password field, but it is forgotten once GDC is stopped.
- If **Allow persistent save** is checked, GDC keeps the password between sessions. The GDC can be stopped and re-started, and the password is maintained. This option is only enabled if the **Keep password** option is checked.

Important: GDC never stores your password in a file or elsewhere unless **Allow persistent save** is checked. GDC stores your password on disk in an encrypted form which is very difficult to read but not impossible. Someone with strong knowledge in cryptology can eventually break the password protection.

The next two options involve the display of the login form:

- If **Don't ask again** is checked, GDC only displays the login box to ask for the password the first time a shortcut is launched. After that, the password will be silently sent without bothering the user with another login box, for the duration of the GDC session.
- If **Even after restart** is checked, the GDC uses the saved password (see Allow persistent save) to silently send the password in subsequent sessions; the password field is no longer displayed, even after the GDC is restarted. This option is only enabled if both **Allow persistent save** and **Don't ask again** are checked.

If any of those options save a password, it will be stored until manually cleared.

The **SSH key file** field: If you use an SSH connection, you can specify an ssh key file that contains the login information. The file format must use the PuTTY format and can be generated using PuTTY tools.

- The **Kerberos** authentication method:

On Windows™ platforms (all versions after Windows™ 2000) you can also use Kerberos authentication if your user and computer are registered on an ActiveDirectory that provides a Kerberos interface. Using this authentication method, you are free to **Allow Ticket Forwarding**; this allows the SSH server to forward the Kerberos ticket that identifies the user to other processes. You may also select a **Server realm**; this identifies the Kerberos domain. This field can be mandatory, depending on the ActiveDirectory / Kerberos server configuration. Ask your System administrator for further details.

Shortcut Wizard page 6: Terminal strings

10. Specify the **connection strings** settings and click **Next**.

On this page, the wizard allows you to specify **connection strings**. This table is used to tell GDC what to do when the [Runtime System](#) host displays a given string on the terminal. GDC can perform the following actions:

- Ask the user for a value, and send it back
- Display a message to the user
- Ask for a password
- Send the shortcut password
- Send the shortcut command
- Execute a local command and send the result
- Return a defined string
- Ignore the [Runtime System](#) string
- Send the login
- Get a free port number for [Port Forwarding](#)
- Show or hide the terminal
- Close the terminal

You can specify whether each string should be recognized only once or every time (check only once). Also, when **ignore remaining terminal strings** is checked, all the following strings are ignored.

The default terminal strings should be suitable in most of the cases, but you may have to adapt them to your system. For instance, the default string to send the command is `last login:` which may be different on your server.

Examples:

Table 6: Connection string examples

Recognized string	Description	Action performed by GDC
<code>password:</code>	This is the string used by the telnet daemon to ask for the password.	Sends the password

Recognized string	Description	Action performed by GDC
last login:	This is the string used by the telnet daemon to tell the user he has logged in successfully.	Sends the command
login:	This is the string displayed by the telnet daemon when the login has failed	Displays a message "Authentication has failed"

Please contact your System administrator if the default values are not appropriate.

Shortcut Wizard page 7: Fgltty Configuration

11. Configure your Fgltty settings and click **Finish** to complete the setup and exit the Shortcut Wizard.

Starting with Genero 2.30, these options are inherited from PuTTY. If you need more details on these options, please consult the [PuTTY documentation](#).

Creating a HTTP Connection Shortcut through a web server

This procedure guides you through the process of creating a HTTP Connection Shortcut through a web server using the Shortcut Wizard.

With an *HTTP connection shortcut*, the GDC connects to the Runtime System via the Genero Application Server, using the HTTP protocol.

Note: GDC 3.00 supports Internet Protocol version 6 (IPv6) when connected to a Genero Application Server.

Shortcut Wizard page 1: Shortcut identification and Connection type

1. Complete the fields of the **Shortcut identification** section:

- In the **Name** field, provide a name for the shortcut.
- Optional: In the **Icon** field, provide a file name that will be used to display an icon associated with this shortcut.
- Optional: If you want to store the shortcut locally for the current user, select the **Store shortcut in settings** checkbox.

By default, shortcuts are saved in the `&AppDataDir/config.xml` file (see [GDC configuration file directories](#) on page 17 for more information). Shortcuts written to this file are shared amongst all users who use this installation of the GDC. Selecting the **Store shortcut in settings** option is useful when the GDC is on a shared network drive and you do not want the current user to modify the common shortcuts.

Note: When the `config.xml` file is read-only, the **Store shortcut in settings** checkbox is selected by default and you do not have the option to deselect it.

Note: Any modification to a non-local shortcut displays a warning and creates a local copy of the shortcut.

2. In the **Connection type** section, select the **HTTP, through a web server** and click **Next**.

Shortcut Wizard page 2: Web server and HTTP Proxy mode

3. In the **URL** field, enter the URL for the application that you want to access.

- When accessing applications using a web server, a typical URL would be: `http://myserver/gas/ua/r/gdc-demo`.
- When accessing applications without using a web server, a typical URL would be: `http://myserver:6394/ua/r/gdc-demo`.

Note: If the URL of the web server that you enter begins with `https` (secure), a fourth page of the wizard is automatically added that provides the option to edit the client certificate mode. See *Optional: Shortcut Wizard page 4: Client certificate mode*.

4. Select a HTTP proxy mode from the list and click **Next**.

If your connection uses a proxy, you can configure it also. If you're attempting to connect to a local address, you can bypass the proxy.

Shortcut Wizard page 3: Login form and Authentication

5. In the **Login form** section, enter a file path in the **Form file** field. Optionally, you can select the **Always On Top** checkbox.
6. Enter a user in the **User** field and choose any of the password display options. Optionally, you can enter a realm in the **REALM** field.

Optional: Shortcut Wizard page 4: Client certificate mode

7. If the URL that you entered in step 3 began with *https*, you can specify a client certificate mode to authenticate the client to the *https* server from the following options:

- **Disabled**
- **Use certificate/key files**

Note: Currently, except for Microsoft™ Windows™ systems where you can use a system certificate, only two types of certificate are supported:

- PEM certificate: which requires a certificate and a private key.
- PKCS12 certificate: which includes both certificate and private key.

If your certificate is password protected, you will be prompted for a password when the certificate is installed. Please note that the password may be requested again, depending on the password options you selected in Step 1.

Important: When credentials are required for connecting to an application, the Genero Desktop Client attempts to use single sign-on, in order to avoid requiring the user to enter a password. If single sign-on fails, the Genero Desktop Client switches to the NTLM authentication protocol. The Genero Desktop Client only supports NTLM v1.

If you are using the NTLM authentication protocol with a Microsoft™ IIS Web server, you must verify that NTLM v1 is also supported. Starting with Microsoft™ IIS Web server 7.0 (Windows™ 2008 server), NTLM v2 is required and the Genero Desktop Client is not compatible.

See the *Genero Application Server User Guide* for more information on configuring applications.

- **Use system certificates**

Note: On Microsoft™ Windows™, there are five methods of selecting a system certificate:

- SUBJECT: use the first certificate in which the subject field contains the given string.
- ISSUER: use the first certificate in which the issuer field contains the given string.
- HASH: use a hexadecimal hash that identifies a certificate. (eg: A5 C8 3F 34 21 C5 FF 8B 0A 0B 24 57 DD B2 C8 9F 1C 7A 45 76)
- ANY: select the first one
- ASK: ask the user to choose in a list

8. Click **Finish** to complete the setup and exit the Shortcut Wizard.

Creating a Local Execution Shortcut

This procedure guides you through the process of creating a Local Execution Shortcut using the Shortcut Wizard

Shortcut Wizard page 1: Shortcut identification and Connection type

1. Complete the fields of the **Shortcut identification** section:
 - a) In the **Name** field, provide a name for the shortcut.
 - b) Optional: In the **Icon** field, provide a file name that will be used to display an icon associated with this shortcut.

- c) Optional: If you want to store the shortcut locally for the current user, select the **Store shortcut in settings** checkbox.

By default, shortcuts are saved in the `&AppDataDir/config.xml` file (see [GDC configuration file directories](#) on page 17 for more information). Shortcuts written to this file are shared amongst all users who use this installation of the GDC. Selecting the **Store shortcut in settings** option is useful when the GDC is on a shared network drive and you do not want the current user to modify the common shortcuts.

Note: When the `config.xml` file is read-only, the **Store shortcut in settings** checkbox is selected by default and you do not have the option to deselect it.

Note: Any modification to a non-local shortcut displays a warning and creates a local copy of the shortcut.

2. In the **Connection type** section, select the **Local execution** and click **Next**.

Shortcut Wizard page 2: Local execution information

3. Enter the Local execution information that you want to specify in the **Executable file**, **Working directory**, and **Parameters** fields

To start your program on the [Runtime System](#), GDC will simply start an executable (giving it some parameters). This executable will typically be one of the following types:

- `fglrun` started in the application directory
- a batch file that will start all applications

You can have, for example:

Table 7: Local execution shortcut examples

Executable file	Working Directory	Parameters	Remarks
<code>fglrun</code>	<code>/home/fgl/demo/</code>	<code>stores.42m</code>	<code>fglrun</code> should be in the PATH
<code>c:\<mydir>\fgl\bin\fglrun.exe</code>	<code>c:\genero\demo</code>	<code>stores.42m</code>	
<code>c:\demos\stores.bat</code>			<code>stores.bat</code> is a batch file that sets the environment and starts the program.

Note: For complicated commands, use a script.

Important: If you have the configuration shown below, you must be sure that all the environment variables are set **before** starting the application. The variables can be set in the **Environment Variables** system dialog.

Table 8: Local connection shortcut example revisited

Executable file	Working Directory	Parameters	Remarks
<code>c:\<mydir>\fgl\bin\fglrun.exe</code>	<code>c:\genero\demo</code>	<code>stores.42m</code>	

4. Click **Finish** to complete the setup and exit the Shortcut Wizard.

Shortcut Management

When in administration mode, you can edit, duplicate, import, export and remove shortcuts.

Starting shortcuts

Shortcuts can be started from the **Shortcuts** window in the user interface or from the command line.

In the user interface, there are two options for starting a shortcut:

- Double-click the **Shortcut** icon in the list.
- Highlight the shortcut in the list by clicking on it once and then click the **Start !** button.

Shortcuts can also be started via the [command line](#):

- `gdc -S <shortcutname>` will start the GDC (if needed) and the specified shortcut.
- `gdc myshortcut.gdc` will start the GDC (if needed) and the shortcut defined in `myshortcut.gdc` file. If several shortcuts have been exported, the first one will be started.

Duplicate shortcuts

To create a copy of a shortcut, select the shortcut and click on the **Duplicate...** button.

If only one shortcut is selected, a copy is created and the Shortcut Wizard opens, allowing you to modify the new copy.

If several shortcuts are selected, copies are created with unique new names, however the Edit Shortcut wizard is not displayed by default.

Remove shortcuts

To delete a shortcut, select the shortcut and click on the **Delete** button.

You are asked to confirm the delete. If you answer in the affirmative, the shortcut is removed.

Import and export shortcuts

Shortcuts can be exported as a `.gdc` file, an XML file containing the configuration details for one or more shortcuts. A `.gdc` file can be used to transfer shortcuts between GDC installations.

To export a collection of shortcuts:

1. Select one or more shortcuts.
2. Right-click in the shortcuts list and select **Export...** from the contextual menu.
3. Provide a file name in the **Choose a new shortcut file** dialog.

A shortcut file with a `.gdc` extension is created. This file can be used to import the collection of shortcuts.

To import a collection of shortcuts:

1. Right-click in the shortcuts list and select **Import...** from the contextual menu.
2. Select a file using the **Choose a shortcut file** dialog and click on the **Open** button.

The imported shortcuts appear in the shortcut list.

Shortcuts and environment variables

You can replace strings with the values of environment variables.

In some fields, GDC will replace any `$xxx` (X11 / Mac OsX) or `%xxx%` (Windows™) by the corresponding environment variables. The fields concerned are:

Table 9: Environment variable fields

Connection Type	Fields
direct	host, username, commandline
http	url
local	command line, working directory, parameters

If you want GDC to simply send the text instead of replacing the environment variable, use the "\" character to escape the variable (e.g. \\$HOSTNAME or \%HOSTNAME%).

Customizing your Login Box

The procedure for creating a customized login box.

Topics

- [Prerequisites](#)
- [How does it work?](#)
- [Tips](#)
- [Samples](#)

Prerequisites

In order to build your own Login Box, you will firstly need to download Qt Creator. This software is available from the Qt website: <http://qt-project.org/downloads#qt-creator>. Download the version for your Operating System.

How does it work?

Default login box:

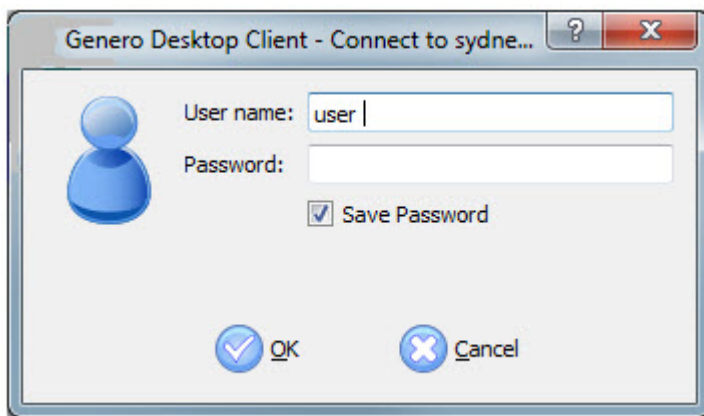


Figure 11: Default login box

To create the the login box you need to run the Qt Designer Form (included in Qt Creator). Select **New File or Project**, then in sub-category **File and Classes** select **Qt**, and, at the end, choose **Qt Designer Form**. Select **Widget** as template.

Now that you're in the Qt Designer Form, you have some rules to respect. You need the following items:

- a QWidget for the form

- a QPushButton named **m_OKPushButton** for the OK button
- a QPushButton named **m_CancelPushButton** for the cancel button
- a QLabel named **m_UserNameLabel** for the label dedicated to the user name
- a QLabel named **m_PasswordLabel** for the label dedicated to the password
- a QLineEdit named **m_UserNameLineEdit** for the edit field where the user enters his name
- a QLineEdit named **m_PasswordLineEdit** for the edit field where the user enters his password
- a QCheckBox named **m_SaveCheckBox** for the checkbox which allows the password to be saved

Optional :

- a QLabel named **m_TextLabel** if you're using a customized message when asking for the password again

Then you can assign it to a shortcut and it will be used instead of the default login box.

Tips

- As it is done in a Genero layout, use a Vertical Layout and Horizontal Layout as much as possible to correctly align and organize your widgets with each other.
- We strongly recommend you embed all elements in a Grid layout (QGridLayout). GDC will always resize the Login Box to its minimum size. When previewing (Alt+Shift+R or Tools -> Form Editor -> Preview) your form in the Qt Designer Form, you should not be able to resize it to very tiny size. Using a grid layout around the various items should help to avoid this. The other solution is to specify a minimum size for the QWidget Form. For this, change the parameters (Width, Height) of the attribute **minimumSize** of the QWidget Form.
- Use Horizontal and Vertical Spacers to better control the free space.
- You are free to add some widgets which are usually not used in a login box: TextEdit, RadioButton, and so on.

Sample

This login box is produced by a customized .ui file. You can request a copy of this file from your local support.



Figure 12: Customized login box example

The Connections Panel

The Connections panel lists applications and cookies being handled by the Genero Desktop Client (GDC).

Overview

The Connections Panel is comprised of two sections: **Connections** and **Cookies**.

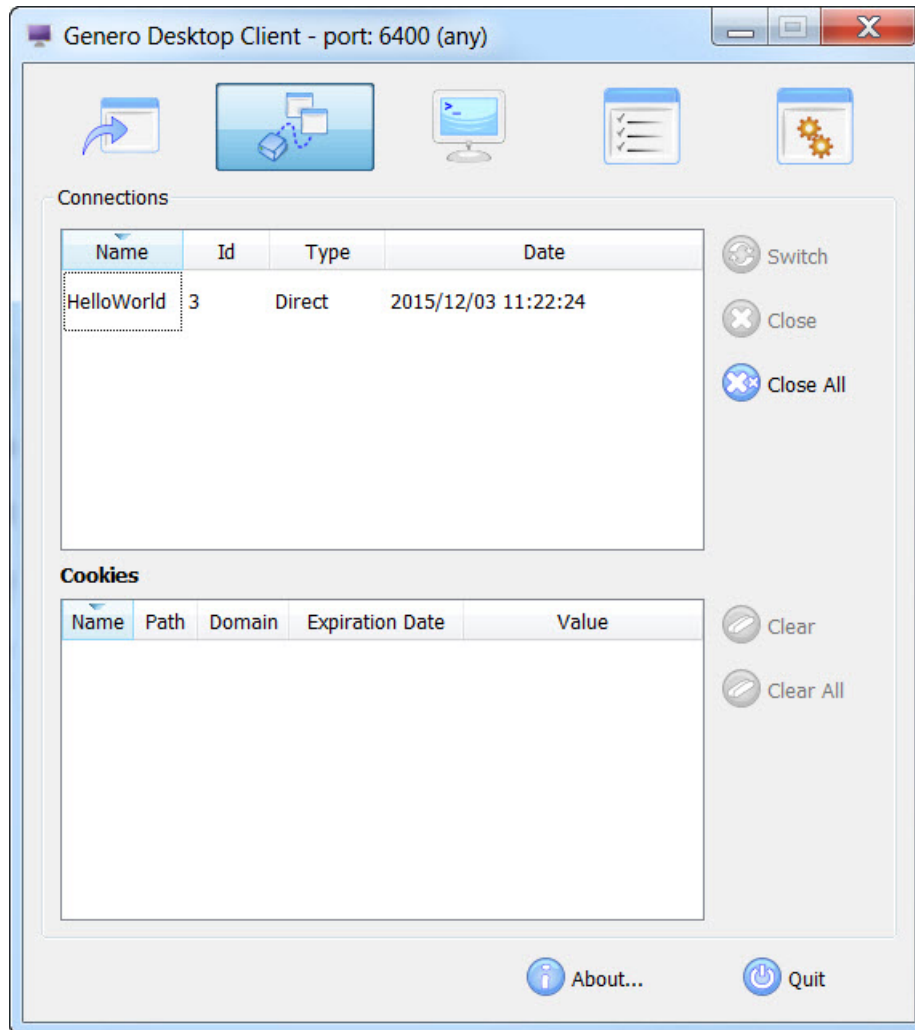


Figure 13: Connections panel

Connections

For each application, it displays:

Name	The name of the application. This refers to the text attribute of the <code>UserInterface Node</code> .
Id	An internal identifier.
Type	The connection type: Direct, HTTP or local execution.
Date	When the application was started.

The **Switch to** button raises the selected application to the top, and the focus sets on the current window.

Tip: This feature allows you to find your application easily when many programs are launched.

The **Close** button stops the selected application(s). When clicked, the information is sent to the [Runtime System](#) and the application is stopped by the GDC; the **Close All** button closes all running applications.

Important: GDC will first send a close request to the runtime system, which may be interpreted differently depending on your Genero application settings; see `OPTIONS ON CLOSE APPLICATION` in the *Genero Business Development Language User Guide*, and will close the network connection after a given delay if the Runtime System does not react.

Cookies

For each cookie, it displays:

Name	The name of the cookie.
Path	The path to the cookie.
Domain	The domain of the cookie.
Expiration Date	Date when the cookie expires.
Value	The value of the cookie.

The **Clear** button clears the selected cookie. The **Clear All** button clears all cookies.

The Terminals Panel

The terminal utility's main purpose is to launch programs with the parameters configured in shortcuts.

Topics

- [Overview](#)
- [Show/Hide](#)
- [Close / Close All](#)

Overview

Shortcuts use a terminal emulation utility (called `fglty`) to connect to the system hosting the [Runtime System](#). Each line of the list in the Terminals panel refers to an active instance of the utility.

Terminals are automatically started by the [Shortcut System](#).

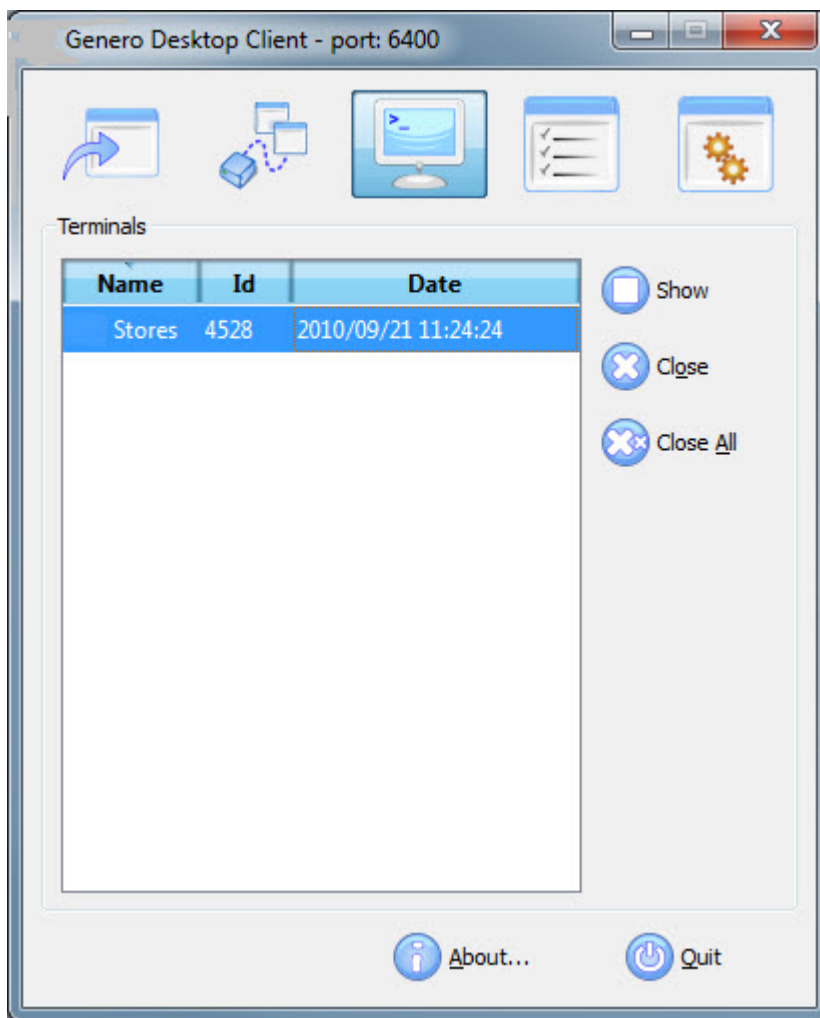


Figure 14: Genero Desktop Client; Terminals Panel

The terminal utility provided is called fgltty.

Important: Coupled with GDC, its main purpose is to launch programs with the parameters which are set in shortcuts. You may use it as a strict terminal emulation utility, but we can not guarantee it will function well, and it won't be maintained for this purpose.

Show / Hide

This button allows you to show or hide the selected Terminal. When you create a shortcut using the [Shortcut Wizard](#), you can specify whether the Terminal Utility is shown. With this button you can show a hidden terminal, or hide a visible one.

This is typically used to check why your application has not started. Showing the Terminal Utility will display what has happened.

Close and Close All

This button allows you to close selected Terminal Utilities. Close All will close all running Terminals.

Important: This may interrupt running applications as the Runtime System process may be terminated also.

The Debug Panel and Logging System

The debug facility for the Genero Desktop client includes logging and the debug console.

The Debug Panel shows the GDC debug facilities: the logging system and the debug console.

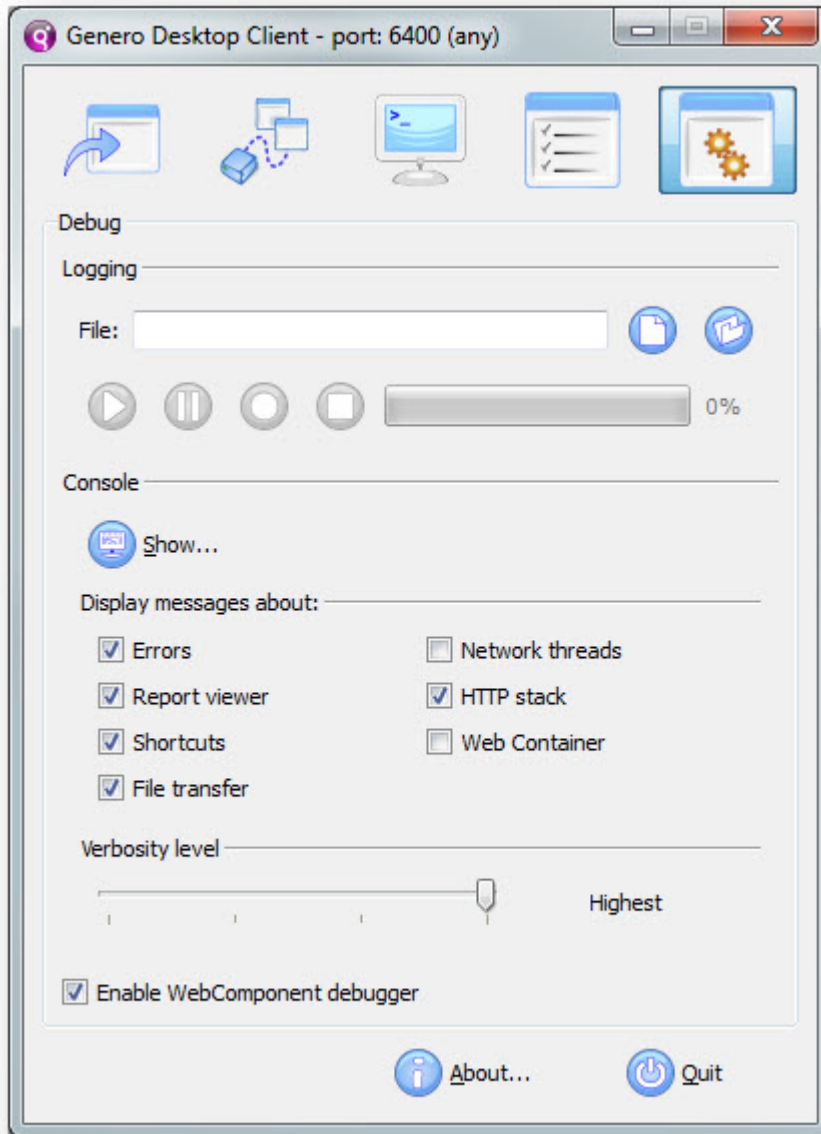


Figure 15: Debug panel

The Debug Panel is only available when GDC is started in [debug mode](#).

- [Using the Logging System](#) on page 33
- [Viewing the Debug Console](#) on page 34

Using the Logging System

The logging system can assist with debugging and creating a demo.

When GDC is started in [debug mode](#), the logging system is available. This system will help you to:

- debug your applications
- create a demo

What is logged? The complete communication between the front end and [Runtime System](#), so the Runtime System is not needed when you replay your demo.

Important: As only the communication is recorded, the "local-only" actions such as moving columns are not saved and replayed. Only the sent value of a field is saved; all user interactions (copy / paste, cursor, and so on) are not saved.

Recording Demo

To record a demo, first select a log file to store the scenario. Then, click on the record button to start recording.

Important: Only applications you launched AFTER starting recording are stored.

Replay Demo

To replay a demo, first select the log file where the scenario is stored. Then, click on the play button to start playing the demo. You can pause the replay by clicking on the pause button. The progress bar will indicate the progress of the demo.

Important: No user interaction is possible when replaying a demo, so you may have to stop recording the demo before the end of the application. Then, if you want to kill this application, you must use the [Connections panel](#).

Viewing the Debug Console

The debug console displays color-coded information about your session.

If you click on the **Console** button, a Window called Debug Console will appear.

Important: The debug console is only available in [debug mode](#).

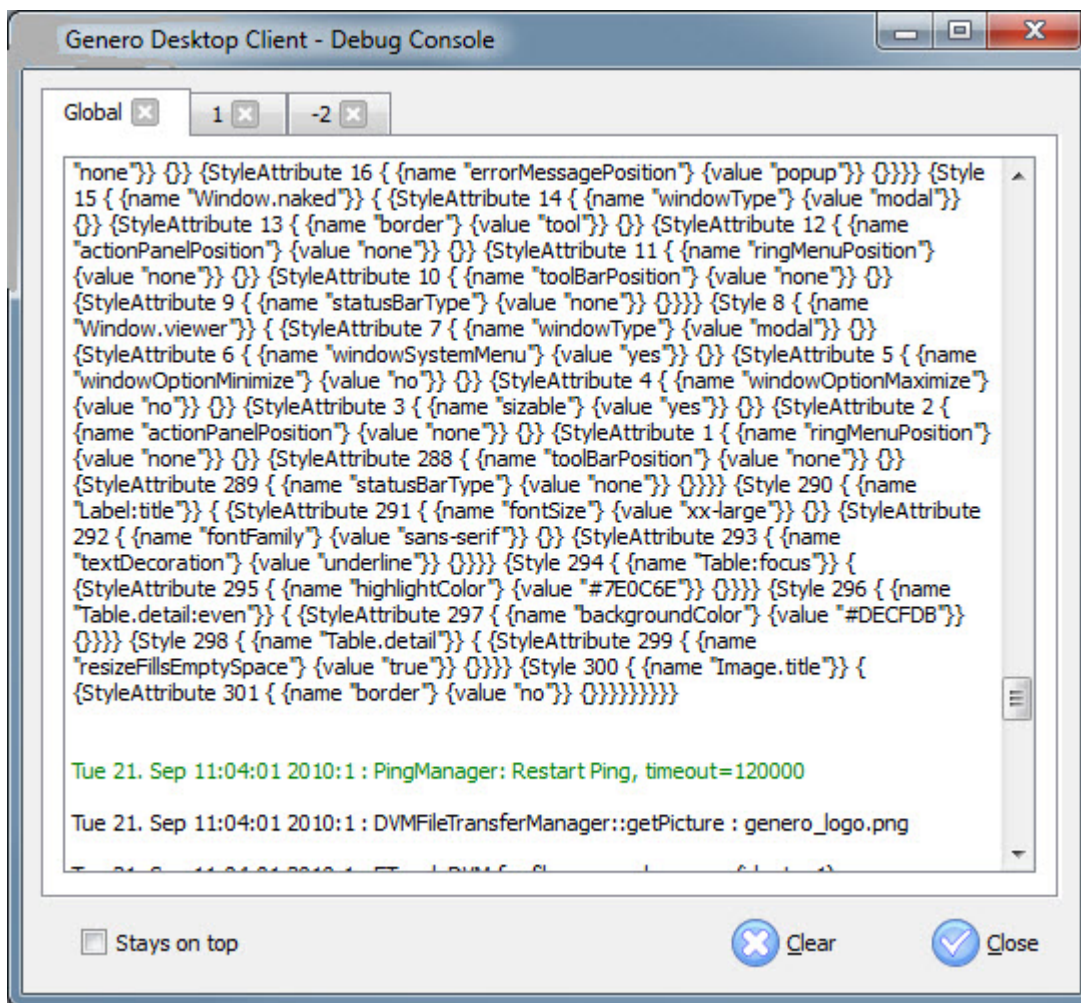


Figure 16: Debug console

In this window some debug information will be displayed. The information is categorized by color:

- in blue - what is sent by GDC to the [Runtime System](#)
- in black - what is received by GDC from the [Runtime System](#)
- in green - some comments or other information
- in red - error messages

This debug console could help you to see the communication between the GDC and the [Runtime System](#). The first folder contains all communication threads which are also reported separately in the other folder tabs (one for each application).

If you want the debug console to stay in foreground and have it always visible, check the Stays On top check box.

Features

The features of the Genero Desktop Client.

- [The Command Line](#) on page 36
- [Printing a screen shot](#) on page 41
- [Local actions](#) on page 42
- [Localization encoding list](#) on page 44
- [Accessibility](#) on page 45

The Command Line

Using the command line with the Genero Desktop Client.

- [Command line options](#) on page 36
- [Command line examples](#) on page 40

Command line options

The command line options of the Genero Desktop Client, organized by category.

Table 10: Genero Desktop Client command line options: Information

This table presents Genero Desktop Client command line options. Some options share the same attribute and description - these options are interchangeable.

Option	Parameter	Description
-h		Displays About Box and exits.
-V		
--Version		
-c		Defines an additional configuration file. See Apply an additional configuration file .
--config		

Table 11: Genero Desktop Client command line options: Network, System

This table presents Genero Desktop Client command line options. Some options share the same attribute and description - these options are interchangeable.

Option	Parameter	Description
-n		Starts a new instance of Genero Desktop Client.
--new		
-p	new_port	Genero Desktop Client will listen on the first available port starting with <i>new_port</i> . Important: If an instance is already running, -p has no effect if -n is not specified.
--port		

Option	Parameter	Description
-q		If the expected port (either 6400, or port specified by <code>--port</code>) is not available, Genero Desktop Client will stop (exit with -1).
-D		Starts Genero Desktop Client in debug Mode (debug Tree and debug Console are active)
-A	security level	Sets Genero Desktop Client's security level regarding the Runtime System's connection.
--Authentication		
--listen	ANY LOCALHOST NONE AUTO	<p>Specify the network listening mode of the Genero Desktop Client.</p> <p>ANY: Listen to any network for a new connection (old behavior)</p> <p>LOCALHOST: Listen to localhost only, DVM must be on the same host as Genero Desktop Client or must be on a host connected with port forwarding</p> <p>NONE: No listening at all. Only HTTP connection will work, this is the most secure operating mode.</p> <p>AUTO: Like LOCALHOST, but Genero Desktop Client will switch back to ANY when a regular direct shortcut (without port forwarding) is used, to allow a connection from outside. The Genero Desktop Client will switch back to LOCALHOST when no more connections and no more terminals are active, after a two minutes timeout.</p> <p>Important: AUTO is the default, which means connections from the outside that are launched without using the shortcut system will not work anymore.</p>

Table 12: Genero Desktop Client command line options: Start Application

This table presents Genero Desktop Client command line options. Some options share the same attribute and description - these options are interchangeable.

Option	Parameter	Description
	.gdc file	Starts Genero Desktop Client with the shortcut specified in the .gdc file. If the file contains several shortcuts, it starts with the first alphabetically. See The Shortcut Sysytem .
-S	shortcut_name	If Genero Desktop Client has not been launched, Genero Desktop Client will start minimized; then, the shortcut named shortcut_name will be started.
--Start		
-s		If Genero Desktop Client has not been launched, Genero Desktop Client will start minimized, using the information given by -U, -H, -T, -P and -C to connect to a DVM.
--startDirect		
-U	user name used	The specified user name will be used when a Direct Connection starts. This option can be used if you share Genero Desktop Client; then each user can create a link to the bin and differentiate the shortcut that will be launched.
--User		
-H	host name	The specified host name will be used when a Direct Connection starts with a defined shortcut (with -S), or starts directly (with -s).
--Host		
-P	password	The specified password will be used when a Direct Connection starts with a defined shortcut (with -S), or starts directly (with -s).
--Password		
-K		The password specified with -P option will be kept in memory and no longer requested.
--KeepPassword		
-C	command_line	The specified command_line will be used when a Direct Connection starts with a defined shortcut (with -S) or starts directly (with -s).
--Cmd		
-T	connexion_type	Defines which protocol should be used when an application starts with -s. Values can be: TELNET, SSH, SSH2. Default is SSH2.
--Type		

Option	Parameter	Description
		Note: Default is now SSH2.
-w		Defines whether the terminal window is visible (when --startDirect option is used). The terminal is hidden by default.
--ShowTerminal		
-f		If a password is provided with --Password, Genero Desktop Client won't display a login box when starting a shortcut. If you explicitly want the login box to be shown, with password and user pre-entered, use the -f option.
--ShowFirstLogin		
-e		Allows the user to save the password in a persistent way (It will not be asked again, even if Genero Desktop Client is stopped and restarted).
--AllowPersistentSave		
-k	Putty Key file (.ppk)	Uses the given Putty Key File as authentication method when Direct Connection .
--PuttyKey		
-u	Genero application URL	Starts the HTTP Genero application given by the URL.
--url		
-g	remote .gdc file	Starts directly the remote .gdc file.
--gdcfile		

Table 13: Genero Desktop Client command line options: Start Genero Desktop Client

This table presents Genero Desktop Client command line options. Some options share the same attribute and description - these options are interchangeable.

Option	Parameter	Description
-a		Starts Genero Desktop Client in admin mode.
--admin		
-c		Defines an additional configuration file. See Apply an additional configuration file .
--config		
-M		Starts Genero Desktop Client minimized.
--Minimized		
-i		Genero Desktop Client starts with ignore Stored Settings on.
--ignoreSettings		
-X		Closes Genero Desktop Client if there is no longer an application or terminal running.
--close		

Table 14: Genero Desktop Client command line options: Logging Mechanism

This table presents Genero Desktop Client command line options. Some options share the same attribute and description - these options are interchangeable.

Option	Parameter	Description
-l	Log file	Starts Genero Desktop Client and replays the given Log File
--logplay		
-L	Log file directory	Starts Genero Desktop Client and replays all the Log Files inside a givendirectory
--logdir		
-r	Log file	Starts Genero Desktop Client, records a log , and saves the given Log File .
--logrec		
-t	delay	By default, replays Log Files at their recording speed. You can change the delay (in milliseconds) between the steps. Caution: A delay that is too small will overcharge Genero Desktop Client. Please consider 100 milliseconds as the smallest acceptable value.
--logtimeout		

Mac OS X users

The command line can be used in either of the following ways:

- Start the terminal application (Applications, Utilities) then enter: `./Applications/gdc.app/Contents/MacOS/gdc command_line`. OSX expects the path to be absolute and not relative.
- Using the following Apple Script: `do shell script "./Applications/gdc.app/Contents/MacOS/gdc command_line"`

Warnings

- The `-S` and `-s` options must be used separately; `-S` is used to start an existing shortcut, and `-s` to start an application using the command line.
- When using `-s`, you must specify at least the host and the command line. The username and password will be prompted if needed.
- Even if you're using the `-q` option, Genero Desktop Client will first check whether another instance is already running. If you really want your Genero Desktop Client instance to stop if the port is not available, use `-n` and `-q` together. Using `-q` alone will stop Genero Desktop Client if the port is not free and not being used by another Genero Desktop Client.

Command line examples

Examples of the Genero Desktop Client command line.

- `gdc -p 6350`
Starts GDC on port 6350.
- `gdc -S demo`

Starts GDC, and the shortcut named **demo**.

- `gdc -S demo -U smith`

Starts GDC, and the shortcut named **demo** using **smith** as the user name.

- `gdc -s -T SSH2 -U smith -H server -P whatisthematrix -C "cd demo ; fgln demo" -X`

Starts GDC, then connects to **server** as the user **smith** with the password **whatisthematrix**. Once connected, performs the specified command line `cd demo ; fgln demo` and closes the GDC when all the applications or terminals are over.

Printing a screen shot

The Genero Desktop Client provides a feature to send the current window to any installed printer.

You can print a screenshot directly from the Genero Desktop Client. No additional tool is required.

To call this feature, you can:

- press CTRL + ALT + P
- press ALT + Print Screen (under Linux® systems only, under Windows™ this combination will be used by the system to put the current screenshot into the clipboard)
- Select the "Hardcopy" option in the System Menu (Windows™ only)

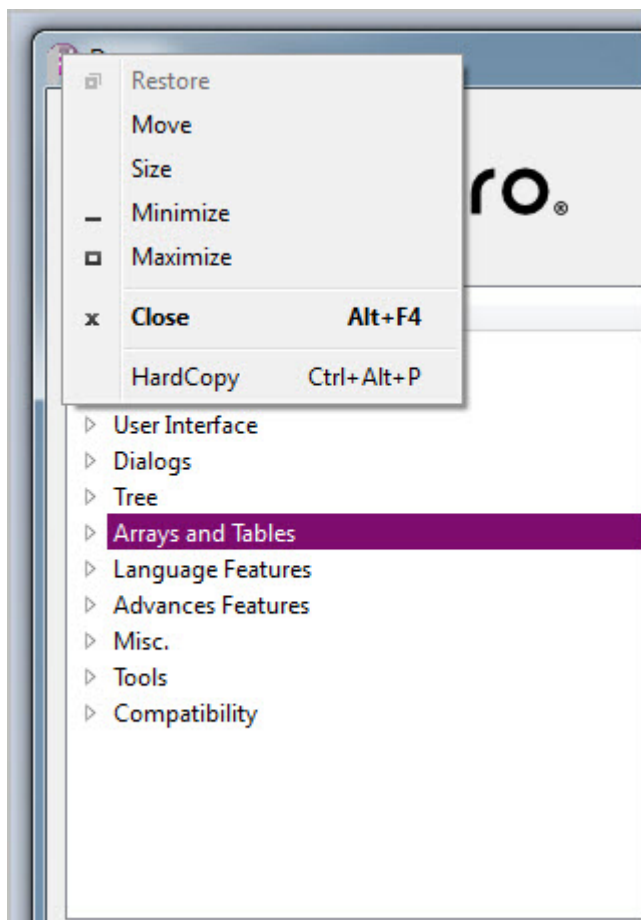


Figure 17: Print contextual menu

The classic "Print dialog" opens, allowing you to select the desired printer, configure it, and then print the current window.

Local actions

Recognize and customize local actions.

Some features of the GDC are defined as "local", including "completely local" features like `copy`, `cut` or `paste`. Some others depend on the DVM but concern local behavior, like the navigation in a table.

To allow you to customize these features with an accelerator, images, comments, and so on, the features are integrated as *local actions*. They follow the same rules as [Runtime System](#) actions, but they are created by the [Front End](#) instead of the [Runtime System](#).

You can create `actionViews` for local actions, as you can for any other action.

Example:

```
BUTTON btn1 = nextfield;
```

When this button is pressed, the focus will go to the next field.

Example:

```
<ActionDefault name="nextfield" accelerator="return">
```

The "return" key will be the accelerator to go to the next field.

List of local actions

The list of local actions is organized by category.

Topics:

- [Editing](#)
- [Navigation in fields](#)
- [Navigation in tables](#)
- [Selection in tables](#)
- [Navigation in folder page](#)
- [Interrupt](#)

Table 15: Local Actions: Editing

Local Action	Description	Shortcut Hotkeys
<code>editcopy</code>	copies the selected text into the clipboard	CTRL+C
<code>editcut</code>	cuts the selected text into the clipboard	CTRL+X
<code>editpaste</code>	pastes the content of the clipboard into the current field	CTRL+V

Table 16: Local Actions: Navigation in fields

Local Action	Description	Shortcut Hotkeys
nextfield	goes to the next field	TAB
prevfield	goes to the previous field	SHIFT+TAB

Table 17: Local Actions: Navigation in tables

Local Action	Description	Shortcut Hotkeys
firstrow	goes to the first row	HOME
prevpage	goes back one page (with TABLE, it follows Internet Explorer behavior)	PRIOR
prevrow	goes to the previous row	KEY_UP
nextrow	goes to the next row	KEY_DOWN
nextpage	goes forward one page (with TABLE, it follows Internet Explorer behavior)	NEXT
lastrow	goes to the last row.	END

Table 18: Local Actions: Selection in tables

Local Action	Description	Shortcut Hotkeys
selectall	Select all rows (if MultiSelection is enabled)	CTRL+A

Table 19: Local Actions: Navigation in folder pages

Local Action	Description	Shortcut Hotkeys
prevtab	raises the previous page	CTRL+SHIFT+TAB
nexttab	raises the next page	CTRL+TAB

Table 20: Local Actions: Interrupt

Local Action	Description	Shortcut Hotkeys
interrupt	sends interrupt to the Runtime System	

Other local actions are those related to the [Rich Text Editing](#).

Localization encoding list

A list of localization encodings supported by the Genero Desktop Client.

Table 21: Localization encoding list

Encoding List
Apple Roman
Big5
Big5-HKSCS
EUC-JP
EUC-KR
GB18030-0
IBM® 850
IBM® 866
IBM® 874
ISO 2022-JP
ISO 8859-1 to 10
ISO 8859-13 to 16
Iscii-Bng, Dev, Gjr, Knd, Mlm, Ori, Pnj, Tlg, and Tml
JIS X 0201
JIS X 0208
KOI8-R
KOI8-U
MuleLao-1
ROMAN8
Shift-JIS
TIS-620
TSCII
UTF-8
UTF-16

Encoding List
UTF-16BE
UTF-16LE
UTF-32
UTF-32BE
UTF-32LE
Windows-1250 to 1258
WINSAMI2

Accessibility

There are accessibility limitations with the Genero Desktop Client, when compared to a standard Microsoft™ Windows™ application.

By default, labels and other widgets that cannot receive the focus cannot be read or spoken by the narrator. The end user must force the reading of the entire window. For the default Windows Narrator, this is accomplished by the hot key combination CTRL+SHIFT+SPACEBAR.

With topmenus and toolbars, the narrator does not read item by item, even when the entire reading of the window is selected. The Genero Desktop Client also does not know when a topmenu or toolbar item is hovered or highlighted. As a result, it is recommended to not use topmenus or toolbars where accessibility is important; you should limit the use of the accessibility to action panels/menus and buttons.

Upgrading

These topics talk about what steps you need to take to upgrade to the next release of Genero Desktop Client, and allows you to identify which features were added for a specific version.

- [What's new in Genero Desktop Client, v 3.00](#) on page 5
- [What's new in Genero Desktop Client, v 2.50](#) on page 47
- [Genero Desktop Client 2.40 New Features](#) on page 48
- [Genero Desktop Client 2.32 New Features](#) on page 53
- [Genero Desktop Client 2.30 New Features](#) on page 54
- [Genero Desktop Client 2.22 New Features](#) on page 66
- [Genero Desktop Client 2.21 New Features](#) on page 69
- [Genero Desktop Client 2.20 New Features](#) on page 76
- [Genero Desktop Client 3.0 upgrade guide](#) on page 86
- [Genero Desktop Client 2.5x upgrade guide](#) on page 87
- [Genero Desktop Client 2.4x upgrade guide](#) on page 87
- [Genero Desktop Client 2.3x upgrade guide](#) on page 88
- [Genero Desktop Client 2.2x upgrade guide](#) on page 90

New features of Genero Desktop Client

These topics provide an look back at the new features introduced with each release of the Genero Desktop Client.

- [What's new in Genero Desktop Client, v 2.50](#) on page 47
- [Genero Desktop Client 2.40 New Features](#) on page 48
- [Genero Desktop Client 2.32 New Features](#) on page 53
- [Genero Desktop Client 2.30 New Features](#) on page 54
- [Genero Desktop Client 2.22 New Features](#) on page 66
- [Genero Desktop Client 2.21 New Features](#) on page 69
- [Genero Desktop Client 2.20 New Features](#) on page 76

What's new in Genero Desktop Client, v 3.00

This publication includes information about new features and changes in existing functionality.

The following changes and enhancements are relevant to this publication.

Table 22: What's new in Genero Desktop Client Version 3.00

Overview	Reference
After the installation of a new version of the GDC, a popup allows you to import the configuration (shortcuts and options) of a previous version.	No additional reference.
GDC 3.00 is compatible with Genero runtime system (DVM) 3.00 and Genero runtime system (DVM) 2.50 for all connections, except for an HTTP connection through the Genero Application Server (GAS). When using an HTTP connection through the GAS:	See the <i>Genero Installation Guide</i> for more information.
<ul style="list-style-type: none"> • GDC 3.00 should use uaproxy (ua) and requires a Genero runtime system (DVM) 3.00. 	

Overview	Reference
<ul style="list-style-type: none"> GDC 2.50 should use gdcproxy (ja) and requires a Genero runtime system (DVM) 2.50. 	
<p>Genero Desktop Client 3.00 supports Internet Protocol version 6 (IPv6), in addition to Internet Protocol Version 4 (IPv4), when:</p> <ul style="list-style-type: none"> Using a web server (connected to a GAS). Using port forwarding through an ssh tunnel. <p>However, as DVM does not support IPv6, you cannot launch an application on a distant host with a GDC listening, using direct connection.</p>	See Port forwarding on page 102 and the Creating Shortcuts using the Shortcut Wizard on page 19 section.
<p>GDC configuration files, supporting files, and cached files are now written to the User directory, providing each user with their own configuration settings (amongst other things) by default.</p>	See GDC configuration file directories on page 17.
<p>The Connections panel now displays information about cookies.</p>	See The Connections Panel on page 30.
<p>Chinese translation is now available.</p>	No additional reference.
<p>HTTP protocol enhancements:</p> <ul style="list-style-type: none"> Support of single sign-on (SSO) mechanism. Support of auto logout. 	See the <i>Genero Application Server User Guide</i> for more information on the HTTP protocol.
<p>Web component enhancements:</p> <ul style="list-style-type: none"> Support of URL-based web components. Support of <code>call</code> frontcall. 	No additional reference.
<p>When copying data, you can now select part of the text of a non-editable field.</p>	No additional reference.
<p>Fgltty is now based on Putty 0.65</p>	No additional reference.
<p>GDC is now based on Qt 5.5</p>	No additional reference.

What's new in Genero Desktop Client, v 2.50

This publication includes information about new features and changes in existing functionality.

The following changes and enhancements are relevant to this publication.

Table 23: What's new in Genero Desktop Client Version 2.50

Overview	Reference
<p>UI enhancement: scrollbar added to the display of an array (matrix with dimension).</p>	No additional reference.
<p>You now have the ability to specify a range of ports using automatic port forwarding. Command line port request and HTTP port request methods are deprecated and likely to be removed in a future version.</p>	See the Port Forwarding and Firewalls on page 101 section.
<p>Improved display of a one row MATRIX. The blue background has been removed and is replaced by a darker rectangle.</p>	No additional reference.

Overview	Reference
With the spell checker, you now have the ability to specify an URL for the dictionary path.	See the TextEdit style attributes topic in the <i>Genero Business Development Language User Guide</i> .
WEBCOMPONENT now supports the SIZEPOLICY attribute.	See the WEBCOMPONENT item type topic in the <i>Genero Business Development Language User Guide</i> .
Starting with the Genero Desktop Client 2.50, The Genero Desktop Client ActiveX (GDCAX) is deprecated. It is recommended that you use the Genero Web Client for HTML5 instead.	No additional reference.

Genero Desktop Client 2.40 New Features

This section describes the new features for Genero Desktop Client 2.40

Table 24: 2.40 New features: General features

Overview	Reference
Support for Summary Line introduced.	
Support for built-in search and fast-peek features introduced.	

Table 25: 2.40 New features: Widgets

Overview	Reference
A new Combobox style attribute, <code>completionTimeout</code> , has been added. This style attribute also applies to RadioGroups.	See the <i>ComboBox style attributes</i> section in the <i>Genero Business Development Language User Guide</i> .
A new ComboBox style attribute, <code>comboboxCompleter</code> , has been added.	See the <i>ComboBox style attributes</i> section in the <i>Genero Business Development Language User Guide</i> .
Style decorations that are applied on a given line (using <code>:odd/:even</code> pseudo selectors for instance) are now applied on the whole line, including the right hand side area where there may be no column, and which was not decorated in previous versions.	No additional reference.
<p>Web Component: Debugging information</p> <p>This topic introduces the debugging information added to assist with debugging the WebComponent.</p> <p>Introduced in 2.30, WebComponent is a powerful mechanism that allows you to integrate any "web based" component in your 4GL application. Because of the fully integrated aspect, it was sometimes difficult to setup web components within GDC, and to understand what could go wrong (for example: JavaScript™ error). GDC 2.40 is now</p>	

Overview	Reference
<p>showing new debugging information in the debug console:</p> <ul style="list-style-type: none"> • JavaScript Messages • WebComponent internal debugging info: url loaded, http errors • gICAPI object debugging info: creation, bridge (GDC // JavaScript) setup • API debugging: calls to onData, onProperty, onFocus, setData, setFocus, Action <p>You will have to enable webcomponent debugging in the console to see the messages.</p>	
WebComponent now accepts gzip encoding.	See the <i>WEBCOMPONENT item type</i> section in the <i>Genero Business Development Language User Guide</i> .
WebComponent now accepts cookies, which could be useful for authentication purposes. Cookies are kept in memory during the lifetime of the GDC.	See the <i>WEBCOMPONENT item type</i> section in the <i>Genero Business Development Language User Guide</i> .

Table 26: 2.40 New features: Traditional mode

Overview	Reference
Global (from CALL <code>ui.interface.LoadToolBar()</code>) Toolbar is now displayed in Traditional mode. Form toolbars are still not displayed, as it makes no sense in the traditional mode context.	See the <i>ui.Interface.loadToolBar</i> section in the <i>Genero Business Development Language User Guide</i> .
Traditional windows can be configured to have a status bar. COMMENT, ERROR and MESSAGES will be displayed there instead of their own LINE.	
MDI Container can now be a container for traditional applications (and "modern" at the same time).	See the <i>Window containers (WCI)</i> section in the <i>Genero Business Development Language User Guide</i> .
Traditional applications can now have a "Pop-Tree" StartMenu to start sub applications.	See the <i>Window style attributes</i> section in the <i>Genero Business Development Language User Guide</i> .

Table 27: 2.40 New features: Shortcut mechanism

Overview	Reference
Introduced in 2.30, Automatic port forwarding implied to use "start a new shell" configuration option. This is no longer the case. The feature can now be used with or without starting a new shell.	See Port Forwarding and Firewalls on page 101.

Table 28: 2.40 New features: Monitor

Overview	Reference
Debug console output can be configured to be more or less verbose. Items displayed are now categorized and you can decide which category is displayed in the console.	See The Debug Panel and Logging System on page 33.
The new <code>--listen</code> command line option and Active X <code>setListeningMode()</code> API function have been added to configure the tcp server.	See Genero Desktop Client 2.40 migration guide for more details.

Table 29: 2.40 New features: Miscellaneous

Overview	Reference
Genero Desktop Client displays reports faster due to improvements in 2.40. GDC is also responsible for the communication between Genero Report Engine and Genero Report writer. Performance has been highly improved and very large reports are now displayed much faster.	No additional reference.

Genero Features: Genero Desktop Client 2.40 New Features

This section introduces Genero new features in Genero Desktop Client 2.40.

Summary Line Support

Support for Summary Line introduced with Genero BDL 2.40.

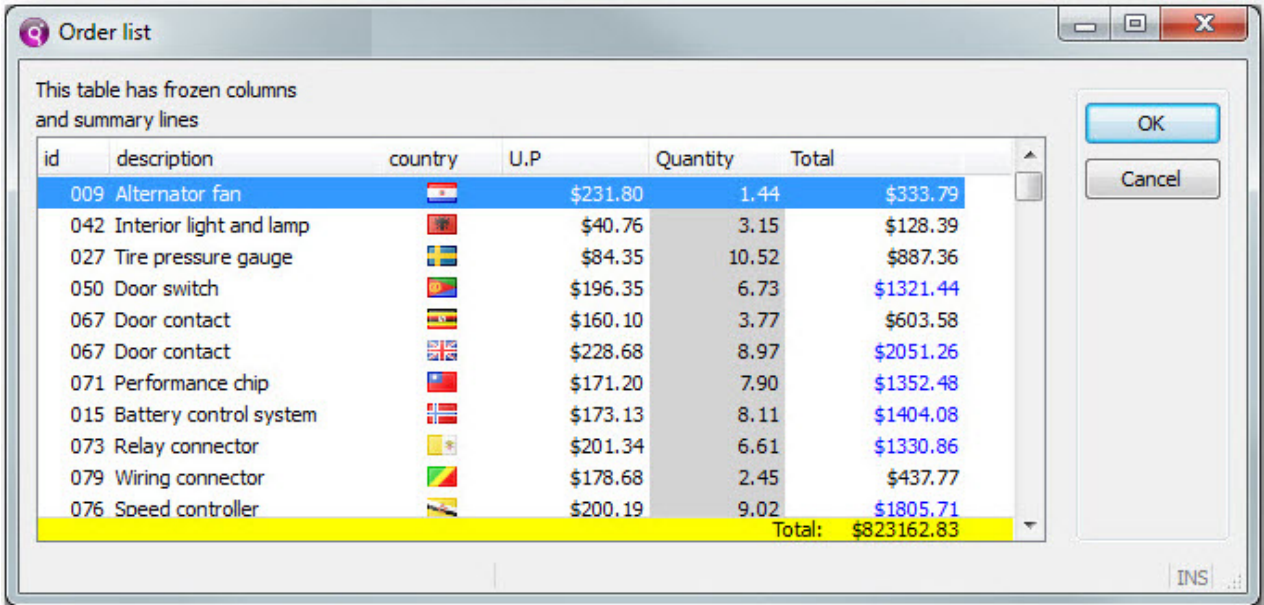


Figure 18: Summary line

Built-in search and fast seek

Support for built-in search and fast-peek features introduced with Genero BDL 2.40.

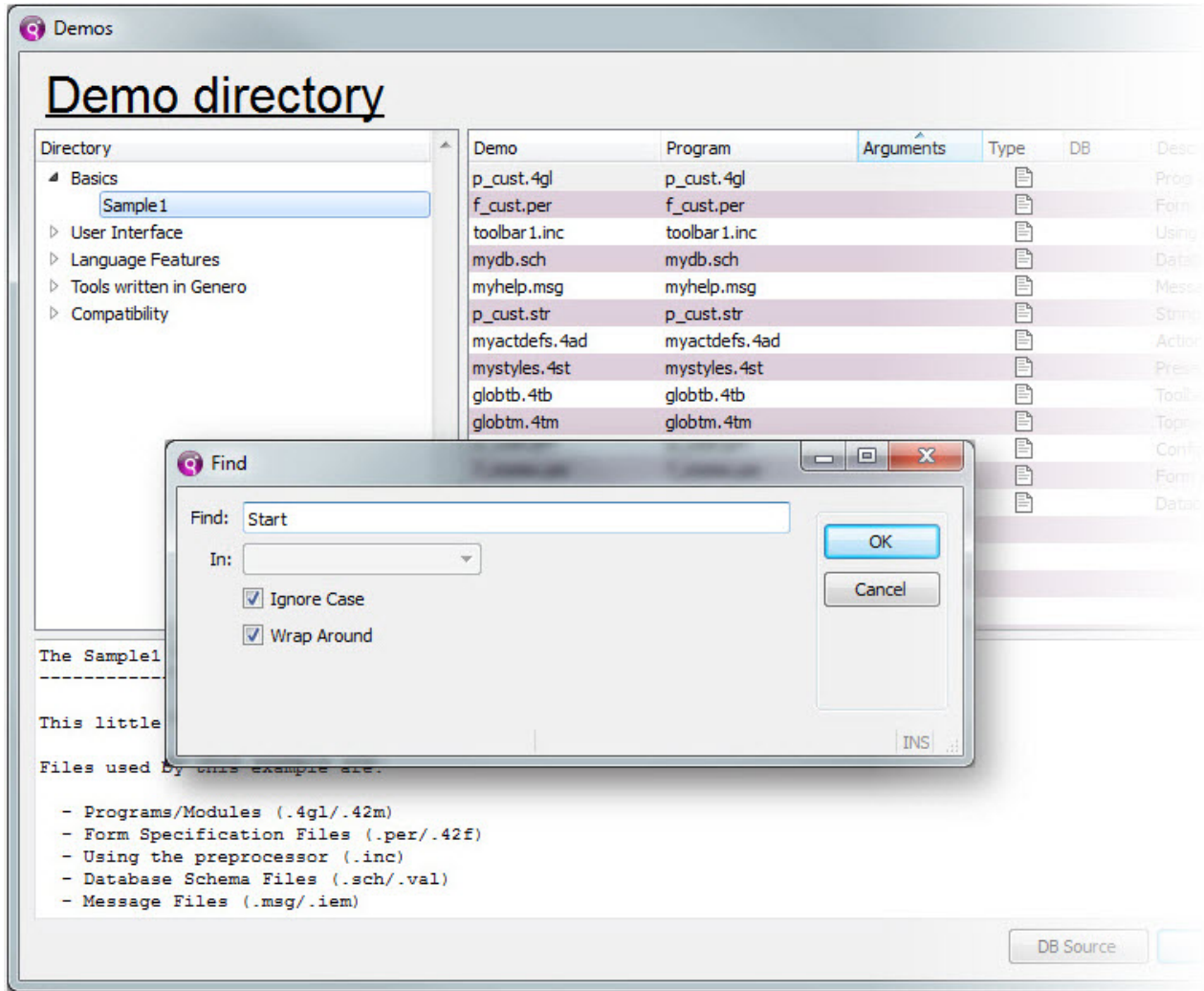


Figure 19: Built-in Search

Widgets: Genero Desktop Client 2.40 New Features

This section describes the widget-related new features in Genero Desktop Client 2.40

Web Component: Debugging information

This topic introduces the debugging information added to assist with debugging the WebComponent.

Introduced in 2.30, WebComponent is a powerful mechanism that allows you to integrate any "web based" component in your 4GL application. Because of the fully integrated aspect, it was sometimes difficult to setup web components within GDC, and to understand what could go wrong (for example: JavaScript™ error). GDC 2.40 is now showing new debugging information in the debug console:

- JavaScript™ Messages
- WebComponent internal debugging info: url loaded, http errors
- gICAPI object debugging info: creation, bridge (GDC // JavaScript™) setup
- API debugging: calls to onData, onProperty, onFocus, setData, setFocus, Action

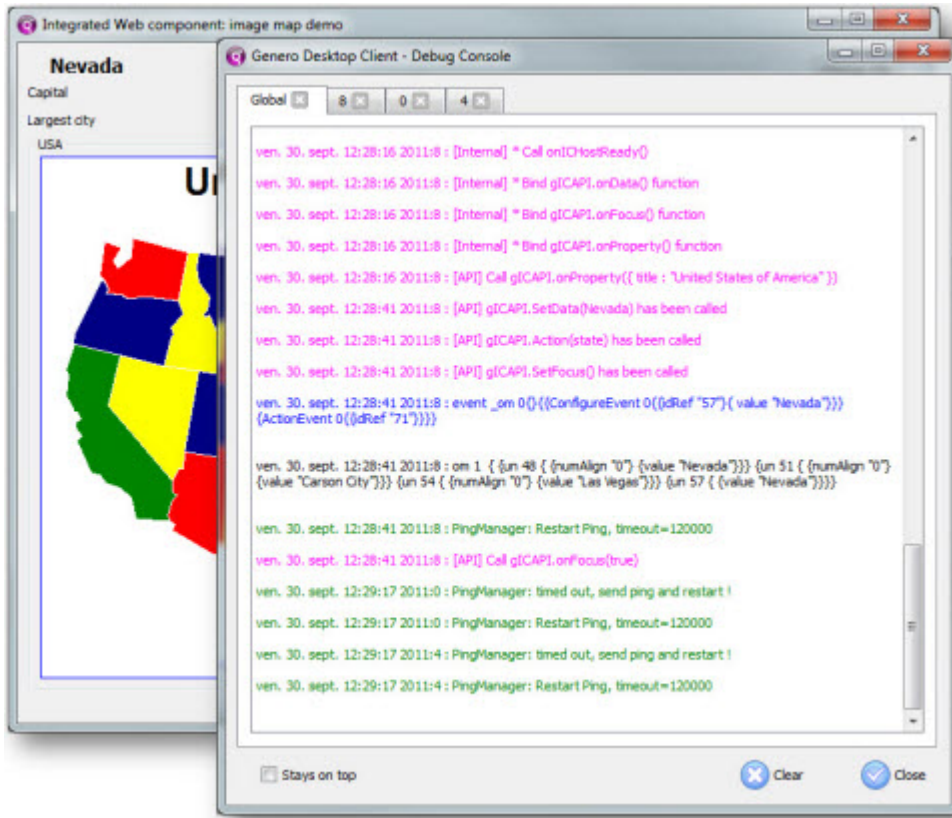


Figure 20: Debug console

You will have to enable webcomponent debugging in the console to see the messages.

Traditional Mode: Genero Desktop Client 2.40 New Features

This section describes the traditional-mode related new features for Genero Desktop Client 2.40

Status bar support

Traditional windows can be configured to have a status bar.

COMMENT, ERROR and MESSAGES will be displayed there instead of their own LINE.

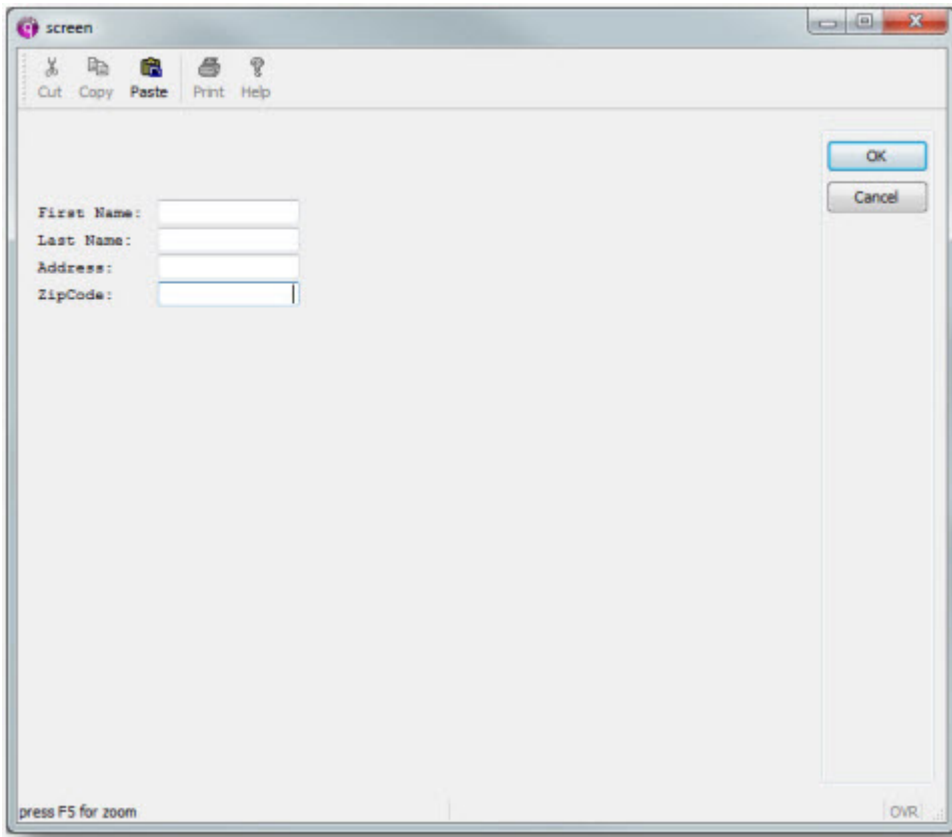


Figure 21: Status bar

Miscellaneous: Genero Desktop Client 2.40 New Features

This section describes miscellaneous new features for Genero Desktop Client 2.40.

Report Viewer: improved performance

Genero Desktop Client displays reports faster due to improvements in 2.40.

GDC is also responsible for the communication between Genero Report Engine and Genero Report writer. Performance has been highly improved and very large reports are now displayed much faster.

Genero Desktop Client 2.32 New Features

This section describes the new features for Genero Desktop Client 2.32

Table 30: 2.32 New features: General features

Overview	Reference
User can manage the image size on all screen elements	
Shows GDC version in windows for file association	

Table 31: 2.32 New features: Widgets

Overview	Reference
Display text of toolbar items next to the icon	
Datedit: Better handling of misformatted date	

Table 32: 2.32 New features: Miscellaneous

Overview	Reference
RingMenu/actionPanel: showing the beginning of the text when larger than buttons.	

Genero Desktop Client 2.30 New Features

This section describes the new features for Genero Desktop Client 2.30

Table 33: 2.30 New features: General features

Overview	Reference
Drag & Drop support introduced in BDL 2.30.	See the <i>Drag & drop</i> topic in the <i>Genero Business Development Language User Guide</i> .
<p>WebComponent support in BDL 2.30 allows you to add any html-based component to your BDL application.</p> <p>Support for WebComponent introduced with Genero BDL 2.30. GDC can embed a Component based on HTML / JavaScript™ / Flash and, via a small interface API, create a bridge between the component and 4GL. This allows you to add any html-based component to your 4GL application, such as a simple Image map which triggers a 4GL action when clicking on an area.</p> <p>The GDC internal browser uses WebKit technology. If you want to use a plug-in for your WebComponent, you need to make sure that the corresponding plug-in is installed on the workstation where GDC runs.</p> <p>Note:</p> <ul style="list-style-type: none"> • The plug-in must be compatible with WebKit ; usually Netscape plug-ins (the technology used by Mozilla Firefox) are supported by WebKit. • If you want to use Flash inside GDC, you need to install either the Stand-Alone Flash player or the FireFox Plug-in. Having only an Internet Explorer (IE) plug-in is not sufficient, as IE plug-ins are based on a different underlying technology. • Plug-ins are similar to external libraries that are loaded at runtime. This implies that the plug-in must be binary compatible with GDC: if you run a 64-bit GDC, you need a 64-bit plug-in. This may be an issue if you run a Flash-based plug-in on Windows®, as today Adobe® only provides a preview version 	See the <i>WEBCOMPONENT</i> topics in the <i>Genero Business Development Language User Guide</i> .

Overview	Reference
of their Flash player on 64-bit Windows® (code name "Square").	

Table 34: 2.30 New features: Widgets

Overview	Reference
Users of Genero Desktop Client 2.30 can now freeze table columns to ensure they remain visible when scrolling.	
Genero Desktop Client 2.30 supports "left", "right", "center", and "auto" alignment for column headers in a table.	
The <code>INCLUDE</code> attribute can now be used with the DateEdit calendar to prevent selection of invalid dates.	See the <i>DATEEDIT item type</i> topics in the <i>Genero Business Development Language User Guide</i> .
End users can now add their own words to the Textedit / Spell Checker dictionary.	See the <i>TextEdit style attributes</i> topics in the <i>Genero Business Development Language User Guide</i> .
<p>Window size will be adjusted to fit an opened form with the "resetFormSize" Form style.</p> <p>GDC tries to maintain the current window size. The window grows only if new content can't fit in the current size, and the GDC never shrinks a window. This is generally the best behavior, as you don't expect window size to jitter each time you change the content. Nevertheless, in the 4GL context of <code>OPEN FORM ... DISPLAY FORM</code>, it may make sense to shrink the window to the size demanded by the newly opened form, particularly if the new form is really small and you don't want to have a large window with lots of empty space. You can use the new "resetFormSize" form style attribute to indicate that the window must adapt its size to the newly opened form.</p> <pre><Style name="Form.f1"> <StyleAttribute name="resetFormSize" value="1" /> </Style></pre>	
Support for message styling has been added in 2.30. When using the <code>MESSAGE</code> or <code>ERROR</code> statement, you can customize how the information is displayed.	See the <i>Message style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
GDC 2.30 introduces the new style attribute for Window <code>ignoreMinimizeSetting</code> . The <code>ignoreMinimizeSetting</code> style attribute can be	See the <i>Window style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .

Overview	Reference
used to prevent a minimized window from being reopened in a minimized state.	
The <code>FORMAT</code> attribute can now be used with <code>SpinEdit</code> widgets to support leading 0 format.	See the <i>SPINEDIT item type</i> topics in the <i>Genero Business Development Language User Guide</i> .
The Delete and Backspace keys now select NULL in a ComboBox.	No additional reference.

Table 35: 2.30 New features: Monitor

Overview	Reference
fglty now uses qPutty for easier deployment and better support of OSX and Linux™.	See the qPutty documentation .
fglty now supports automatic port forwarding for SSH connections. Fglty is now able to detect a free port which can be used by the Port Forwarding mechanism.	See Port forwarding
<p>The GDC About box now has a Copy To Clipboard button, which will fill the clipboard with information that is useful when you contact your support center:</p> <p>The Clipboard will contain:</p> <ul style="list-style-type: none"> • GDC version information. • Command line used to start GDC. • Operating system information. • Copies of config.xml and hosts.xml. <p>When contacting your support center, you can copy/paste this information in your email; this should ease and speed support.</p>	

Table 36: 2.30 New features: Miscellaneous

Overview	Reference
<p>Settings for REPORT TO PRINTER (in case of DBPRINT=FGLSERVER)</p> <p>Printer and font settings can be overridden with two new 'standard' frontcalls.</p> <p>When using <code>DBPRINT=FGLSERVER</code> and <code>REPORT TO PRINTER</code>, text reports are printed via GDC. Printer and fonts can be configured using the Option panel.</p>	the Option panel ?
<p>Animated GIFs support</p> <p>Animated GIFs are supported for the IMAGE widget.</p>	

Overview	Reference
<p>If you load an image which is an animated gif, it will be displayed as animated.</p> <p>Animated GIFs are only available for the IMAGE form item. They are not supported where the image appears due to the IMAGE attribute (in buttons, toolbars, topmenus and so on). For performance reasons, animated GIFs are not supported in TABLE containers</p>	
<p>WinMAIL: Specific Port for smtp</p> <p>The server port can now be configured with a frontcall method when using WinMain and smtp.</p> <p>When using WinMail and smtp, you can now specify the smtp server port in the <code>SetSmtp</code> function. To define the port, use the <code>host:port</code> notation.</p> <pre>CALL ui.interface.frontCall("WinMail", "SetSmtp", [id, "smtp.mycompany.com:1234"], [result])</pre> <p>The default port remains 25.</p>	<p>For more information, see the <i>Windows Mail extension</i> documentation in the <i>Genero Business Development Language User Guide</i>.</p>
<p>WinDDE: handling of ASCII/Wide char data</p> <p>WinDDE can dialog with applications that require data in ASCII and applications that require wide char data such as UTF-16.</p> <p>WinDDE can now dialog both with applications that require data in ASCII and with applications that require wide char data such as UTF-16. Since GDC 2.22.x, only wide char data were supported. Prior to 2.22.x, WinDDE was only able to handle ASCII. By default, WinDDE will automatically guess what is the best encoding when connecting to the DDE server. However, in some instances the returned information can be misleading. In these cases, you will have to set an optional 'encoding' parameter manually in the following DDE functions: <code>DDEConnect</code>, <code>DDEExecute</code>, <code>DDEPeek</code> and <code>DDEPoke</code>. Possible values are: "UNICODE" and "ASCII". For instance:</p> <pre>CALL ui.Interface.frontCall("WINDDE", "DDEPoke", [prog, "Sheet1", "R1C1", "value", "UNICODE"], [res]);</pre>	<p>For more detail, see the <i>WinDDE</i> documentation in the <i>Genero Business Development Language User Guide</i>.</p>

Genero Features: Genero Desktop Client 2.30 New Features

This section describes new features for Genero Desktop Client 2.30

- [WebComponent support](#) on page 58

WebComponent support

WebComponent support in BDL 2.30 allows you to add any html-based component to your BDL application.

Support for WebComponent introduced with Genero BDL 2.30. GDC can embed a Component based on HTML / JavaScript™ / Flash and, via a small interface API, create a bridge between the component and 4GL. This allows you to add any html-based component to your 4GL application, such as a simple Image map which triggers a 4GL action when clicking on an area.

The GDC internal browser uses WebKit technology. If you want to use a plug-in for your WebComponent, you need to make sure that the corresponding plug-in is installed on the workstation where GDC runs.

Note:

- The plug-in must be compatible with WebKit ; usually Netscape plug-ins (the technology used by Mozilla Firefox) are supported by WebKit.
- If you want to use Flash inside GDC, you need to install either the Stand-Alone Flash player or the FireFox Plug-in. Having only an Internet Explorer (IE) plug-in is not sufficient, as IE plug-ins are based on a different underlying technology.
- Plug-ins are similar to external libraries that are loaded at runtime. This implies that the plug-in must be binary compatible with GDC: if you run a 64-bit GDC, you need a 64-bit plug-in. This may be an issue if you run a Flash-based plug-in on Windows™, as today Adobe™ only provides a preview version of their Flash player on 64-bit Windows™ (code name "Square").

Widgets: Genero Desktop Client 2.30 New Features

This section describes widget-related new features for Genero Desktop Client 2.30

- [Frozen columns for tables](#) on page 58
- [Text alignment in headers of tables](#) on page 59
- [Adapt window to new form size](#) on page 59
- [Window: Message nodes support styles](#) on page 60

Frozen columns for tables

Users of Genero Desktop Client 2.30 can freeze columns to ensure they remain visible when scrolling.

You can define whether table columns should be frozen in tables. Frozen columns will always be visible. When the table is smaller than the content, a scrollbar appears.

The scrollbar will only scroll those columns that are not frozen. You can freeze columns on the left or on the right side.

```
<Style name="Table.detail">
  <StyleAttribute name="tableType" value="frozenTable" />
</Style>
```

Right click on the header of the column that should be frozen:

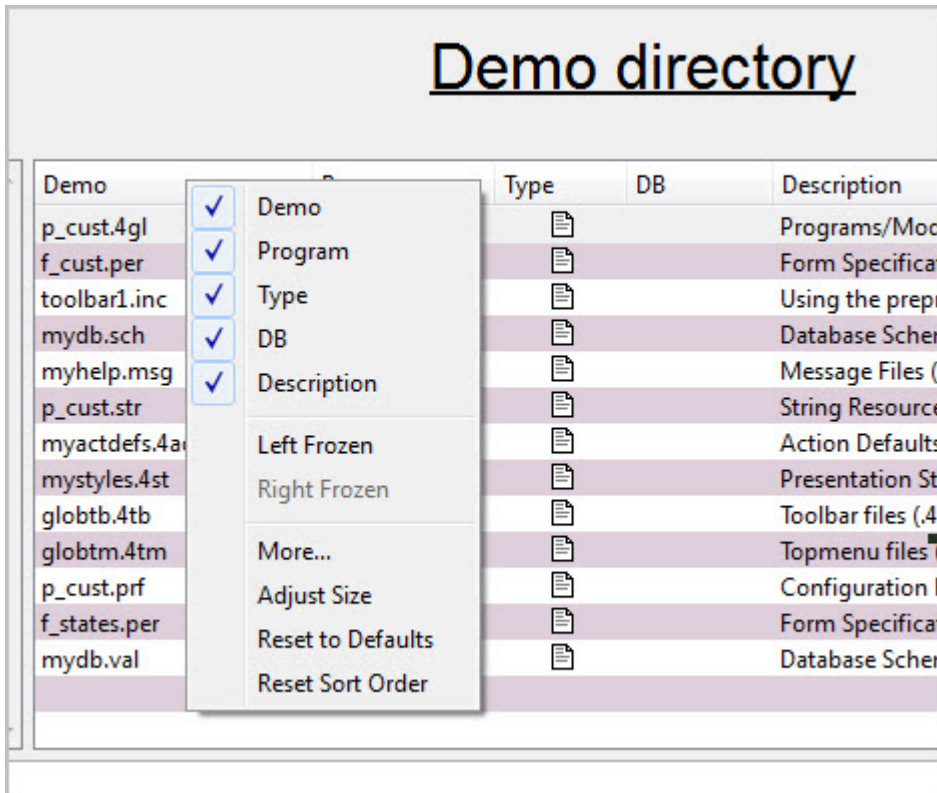


Figure 22: Select to freeze columns on the left or right side of the table

Now this column will always be visible; the scrolling will be done for the other columns.

This style also allows you to define columns that will be frozen on initial display:

```
<Style name="Table.detail">
  <StyleAttribute name="tableType" value="frozenTable" />
  <StyleAttribute name="leftFrozenColumns" value="1" />
  <StyleAttribute name="rightFrozenColumns" value="2" />
</Style>
```

Text alignment in headers of tables

Genero Desktop Client 2.30 supports "left", "right", "center", and "auto" alignment for column headers in a Table.

You can now align the headers of columns in a Table. Possible values are "left", "right", "center" and "auto". "auto" follows the justification of the content of the field which is specified by the JUSTIFY attribute in the form definition. If JUSTIFY is not specified, it follows the default data justification (for instance: left for a string field value, right for a numeric field, and so on.)

```
<Style name="Table.t1">
  <StyleAttribute name="headerAlignment" value="center" />
</Style>
```

Adapt window to new form size

Window size will be adjusted to fit an opened form with the "resetFormSize" Form style.

GDC tries to maintain the current window size. The window grows only if new content can't fit in the current size, and the GDC never shrinks a window. This is generally the best behavior, as you don't expect window size to jitter each time you change the content. Nevertheless, in the 4GL context of OPEN FORM ... DISPLAY FORM, it may make sense to shrink the window to the size demanded by the newly opened form, particularly if the new form is really small and you don't want to have a large window with lots of empty

space. You can use the new "resetFormSize" form style attribute to indicate that the window must adapt its size to the newly opened form.

```
<Style name="Form.fl">
  <StyleAttribute name="resetFormSize" value="1" />
</Style>
```

Window: Message nodes support styles

Styles can now be applied to MESSAGE and ERROR statements.

Support for message styling has been added in 2.30. When using the MESSAGE or ERROR statement, you can customize how the information is displayed. The AUI Tree defines ERROR and MESSAGE as the same node *Message*, so GDC 2.30 introduced new pseudo-selectors `Message:error` and `Message:message`. Now it is possible to:

- Apply a style to all Messages and/or Errors
- Apply a style to a specific Message or Error using `ERROR "error" ATTRIBUTES(STYLE="myStyle")`

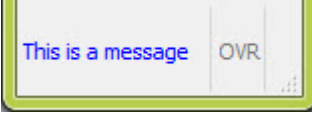
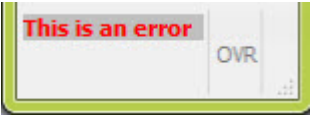
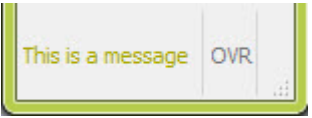
Supported Style Attributes are:

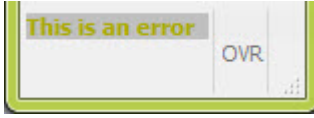
- Font style attributes (`textColor`, `fontWeight`, and so on)
- `position`, which is used to override the Window style property for the current message (`statusbar`, `popup`, `statustip`, both)
- `textFormat`, which is used to define whether a value should be interpreted as `plain text` or as `html`.

Example: 4st file:

```
<Style name="Message:error">
  <StyleAttribute name="textColor" value="red" />
  <StyleAttribute name="fontWeight" value="bold" />
</Style>
<Style name="Message:message">
  <StyleAttribute name="textColor" value="blue" />
</Style>
<Style name="Message.yellow">
  <StyleAttribute name="textColor" value="yellow" />
</Style>
```

Table 37: MESSAGE and ERROR examples

Statement	Result
MESSAGE "This is a message"	
ERROR "This is an error"	
MESSAGE "This is a message" ATTRIBUTES(STYLE="yellow")	

Statement	Result
<pre>ERROR "This is an error" ATTRIBUTES(STYLE="yellow")</pre>	

Note: Similar to simple fields, tty attributes have a higher priority than styles. By default, ERROR has the tty attribute REVERSE, which explains why ERROR messages have a REVERSE background even when using styles.

Monitor: Genero Desktop Client 2.30 New Features

This section describes the new monitor features for Genero Desktop Client 2.30.

- [Fgltty based on qPutty](#) on page 61
- [Automatic Port Forwarding](#) on page 62
- [GDC information for better support](#) on page 64

Fgltty based on qPutty

fglty now uses qPutty for easier deployment and better support of OSX and Linux®.

GDC uses fglty, a modified version of PuTTY ([PuTTY](#)). Before 2.30, fglty was only a modified version of PuTTY that removed anything which was not used by GDC. The main drawback was that, as PuTTY is mainly developed on Windows™, the support on OSX and Linux™ could be improved, particularly in terms of requirements (for instance, GTK2 was required for Linux™, Rosetta for OSX).

Since 2.30, fglty is based on qPutty, a Qt port of PuTTY ([qputty](#)). In addition to an easier deployment mechanism (fglty and GDC use the same Qt libs), GDC has now access to all PuTTY options, which can be configured for each shortcut:

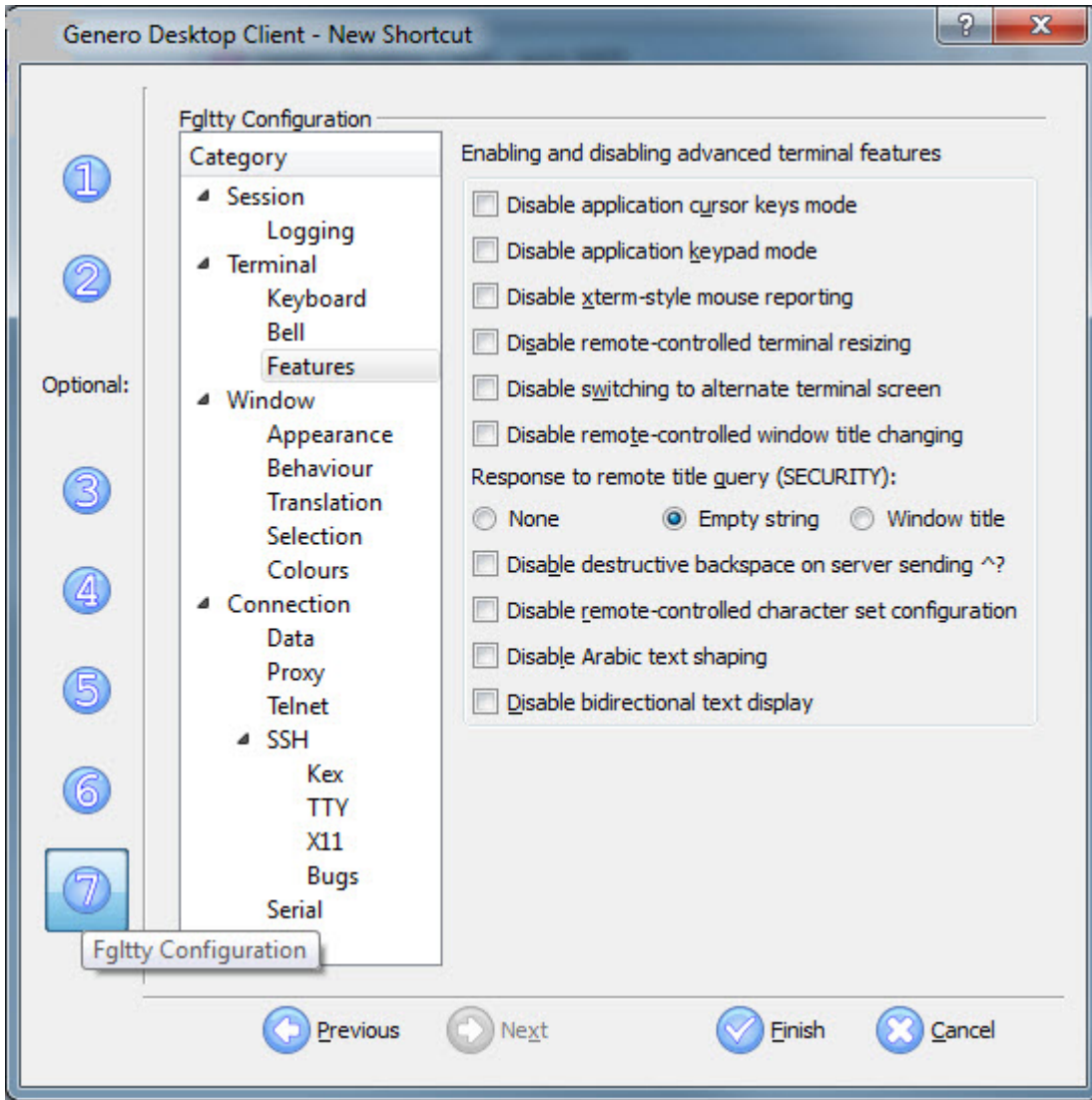


Figure 23: fgltty configuration window

Most of the PuTTY options have been included inside the GDC shortcut configuration. (Options which were already included are still found in the same place). For detailed information regarding PuTTY options, please have a look at [PuTTY's documentation](#).

Please note that GDC will simply create a configuration file which will be passed to fgltty, and that fgltty relies completely on PuTTY.

Automatic Port Forwarding

fglty now supports automatic port forwarding for SSH connections.

Fgltty is now able to detect a free port which can be used by the [Port Forwarding](#) mechanism.

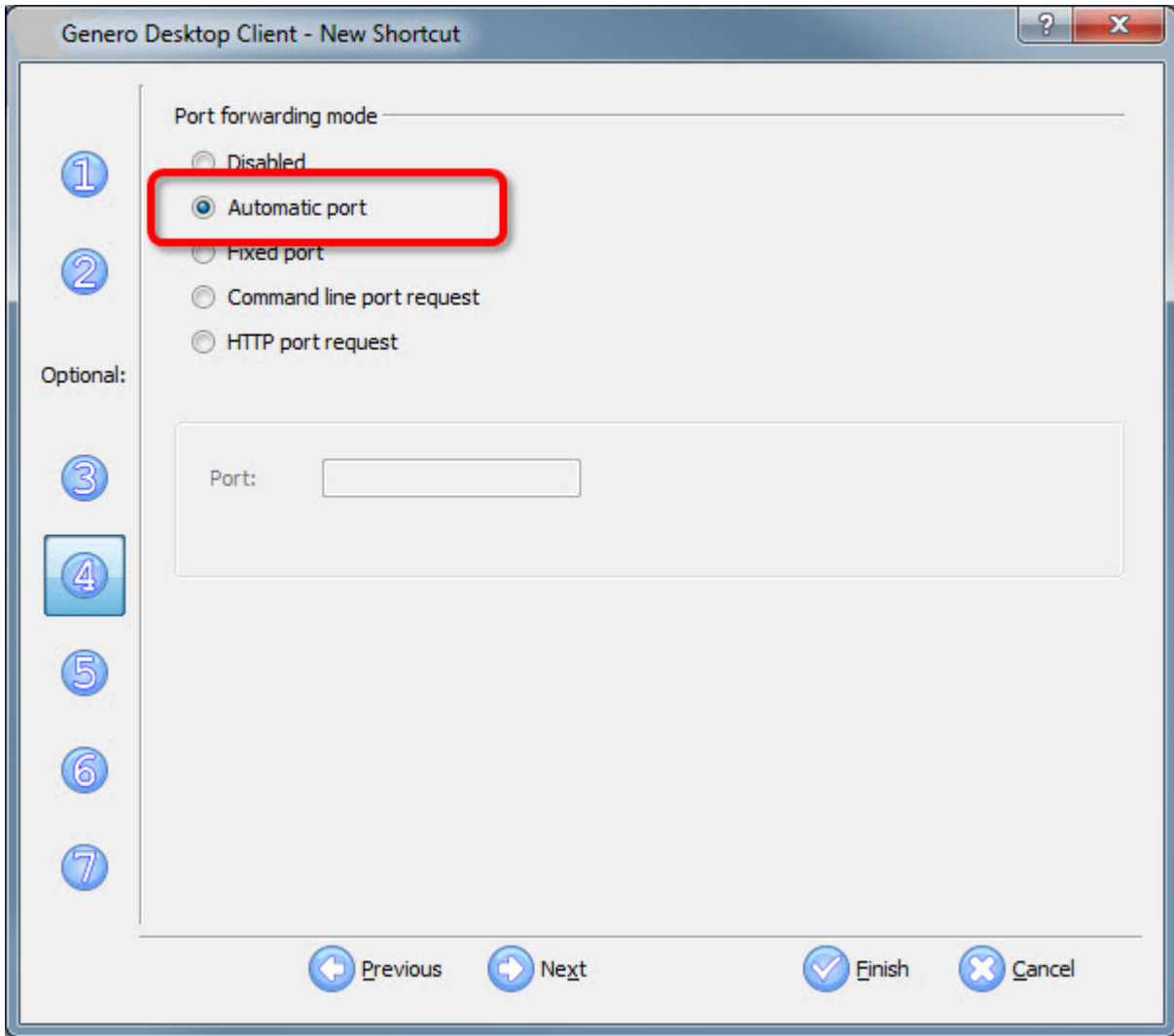


Figure 24: Automatic port selected under Port forwarding mode

With classic port forwarding, fglty connects once, retrieves the port to use, returns it to GDC, and GDC restarts fglty with the tunnel configuration. With automatic port forwarding, fglty establishes the tunnel during the initial connection step, finding a free port automatically. A new default Terminal String has been added: `<FGLAUTOPORT>=`, which corresponds to the message which is automatically written by fglty when selecting the auto port option.

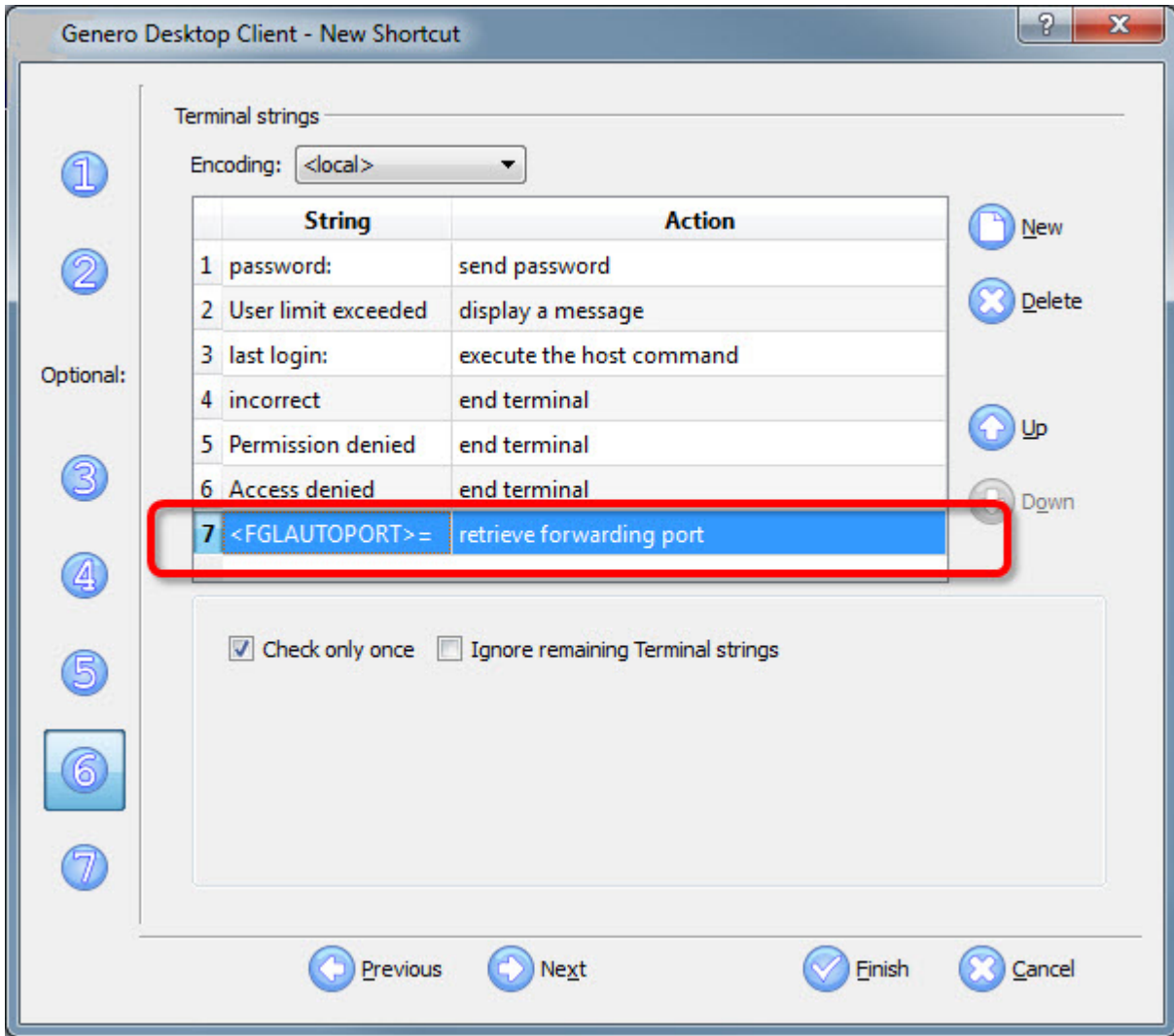


Figure 25: FGLAUTOPORT example

Other mechanisms are still supported if you've implemented your own port assignment system.

Note:

Because there is only one connection - and therefore only one start of fglty - remote command is not passed to fglty (@tags would be wrong as GDC has no way to know the port fglty will use before fglty has been started). This is why using automatic port forwarding always starts a new shell (the option will be mandatory), which means you may have to check the execute the host command connection string to match your server.

GDC information for better support

The GDC About box now has a Copy to Clipboard button that copies useful support information to the clipboard.

The GDC About box now has a **Copy To Clipboard** button, which will fill the clipboard with information that is useful when you contact your support center:

The Clipboard will contain:

- GDC version information.
- Command line used to start GDC.
- Operating system information.
- Copies of config.xml and hosts.xml.

When contacting your support center, you can copy/paste this information in your email; this should ease and speed support.

Miscellaneous: Genero Desktop Client 2.30 New Features

This section describes miscellaneous new features for Genero Desktop Client 2.30

- [Settings for REPORT TO PRINTER \(in case of DBPRINT=FGLSERVER\)](#) on page 65
- [Animated GIFs support](#) on page 65
- [WinMAIL: Specific Port for smtp](#) on page 65
- [WinDDE: handling of ASCII/Wide char data](#) on page 66

Settings for REPORT TO PRINTER (in case of DBPRINT=FGLSERVER)

Printer and font settings can be overridden with two new 'standard' frontcalls.

When using DBPRINT=FGLSERVER and REPORT TO PRINTER, text reports are printed via GDC. Printer and fonts can be configured using the [Option panel](#).

GDC 2.30 allows you to override these options for the current application. Two new "standard" frontcalls are available:

```
CALL ui.interface.frontCall("standard","setreportfont",
["Helvetica, Bold, Italic, 13"],[status])
CALL ui.interface.frontCall("standard","setreportprinter",
["moliere, Portrait, A4, 96 dpi, 1 copy, Ascendent, Color,
Auto"],[status])
```

Note:

- The values for Font and Printer are the same as the ones used in the Option panel. Simple values work (ex: "Helvetica, 18" / "Moliere"), however if you want to customize the font or the printer completely, the easiest way is to configure it with the GDC option panel and copy/paste the result. You can alternatively use "<ASK_ONCE>", "<ASK_ALWAYS>", "<USER_DEFINED>" or "<USE_DEFAULT>" as "printer" of "font" string to enforce the corresponding action.
- DBPRINT=FGLSERVER is limited and no further improvements are planned. We recommend using Genero Report Writer for your reporting needs.

Animated GIFs support

Animated GIFs are supported for the IMAGE widget.

If you load an image which is an animated gif, it will be displayed as animated.

Animated GIFs are only available for the IMAGE form item. They are not supported where the image appears due to the IMAGE attribute (in buttons, toolbars, topmenus and so on). For performance reasons, animated GIFs are not supported in TABLE containers.



WinMAIL: Specific Port for smtp

The server port can now be configured with a frontcall method when using WinMain and smtp.

When using WinMail and smtp, you can now specify the smtp server port in the SetSmtplib function. To define the port, use the host:port notation.

```
CALL ui.interface.frontCall("WinMail","SetSmtplib", [id,
"smtp.mycompany.com:1234"],[result])
```

The default port remains 25.

For more information, see the WinMail documentation in the *Genero Business Development Language User Guide*.

WinDDE: handling of ASCII/Wide char data

WinDDE can dialog with applications that require data in ASCII and applications that require wide char data such as UTF-16.

WinDDE can now dialog both with applications that require data in ASCII and with applications that require wide char data such as UTF-16. Since GDC 2.22.x, only wide char data were supported. Prior to 2.22.x, WinDDE was only able to handle ASCII. By default, WinDDE will automatically guess what is the best encoding when connecting to the DDE server. However, in some instances the returned information can be misleading. In these cases, you will have to set an optional 'encoding' parameter manually in the following DDE functions: DDEConnect, DDEExecute, DDEPeek and DDEPoke. Possible values are: "UNICODE" and "ASCII". For instance:

```
CALL ui.Interface.frontCall("WINDDE", "DDEPoke",
    [prog, "Sheet1", "R1C1", "value", "UNICODE"], [res] );
```

For more detail, see the WinDDE documentation in the *Genero Business Development Language User Guide*.

Genero Desktop Client 2.22 New Features

This section describes the new features for Genero Desktop Client 2.22.

These topics organize the new features of the 2.22 release of Genero Desktop Client by categories.

Table 38: 2.22 New features: Experimental features

Overview	Reference
<p>Genero Desktop Client now supports rich text editing with integrated toolbox or rich text local actions.</p> <p>In previous versions, TextEdits are able to display rich text. In input, it was possible to edit rich text, but this was not straightforward. Moreover, cursor and cursor2 attributes correspond to plain text, not to the real value.</p>	

Widgets: Genero Desktop Client 2.22 New Features

This section describes widget-related new features for Genero Desktop Client 2.22.

- [Rich Text Editing](#) on page 66

Rich Text Editing

Genero Desktop Client supports rich text editing with integrated toolbox or rich text local actions.

In previous versions, TextEdits are able to display rich text. In input, it was possible to edit rich text, but this was not straightforward. Moreover, cursor and cursor2 attributes correspond to plain text, not to the real value.

GDC 2.22 introduces rich text editing feature:

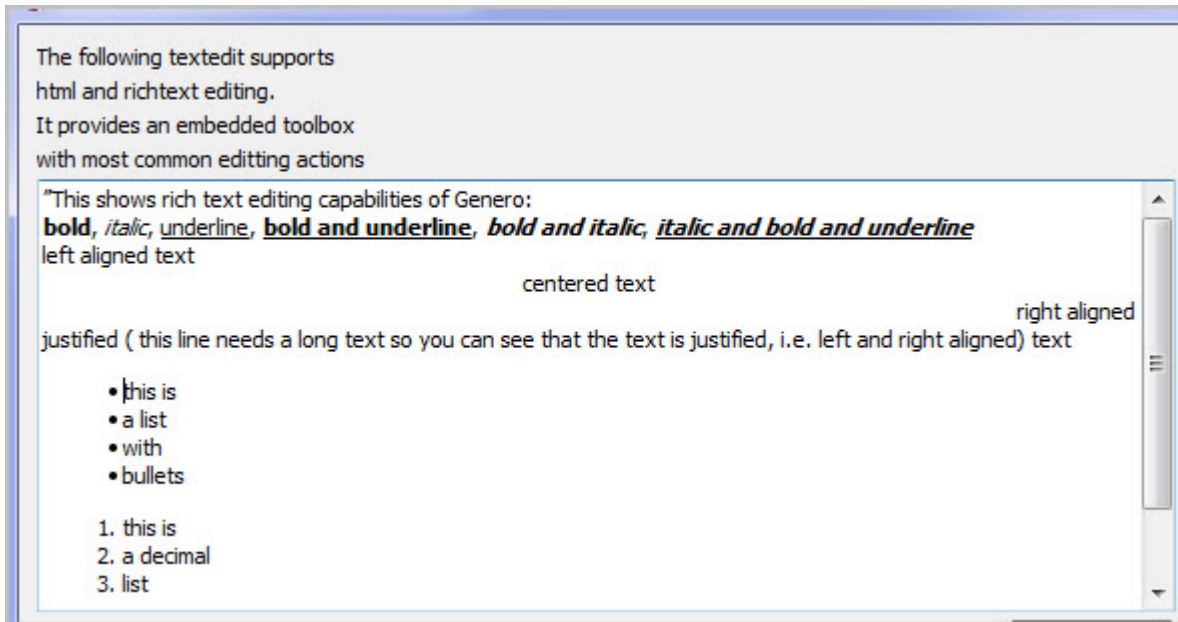


Figure 26: Rich text editing interface

GDC 2.22 provides:

- text format: bold, italic, underline
- paragraph alignment: left, center, right, justify
- lists: bullet, decimal
- paragraph indentation
- font size

To enable richtext editing, you need to set `textFormat` styleAttribute to "html" .

```
<StyleAttribute name="textFormat" value="html" />
```

To modify your document, you can use:

- [Integrated richtext toolbox](#)
- [rich text local actions](#)

Integrated richtext toolbox

By default, when the mouse reaches the top border of the TextEdit, a toolbox will appear. The toolbox will disappear when the mouse leaves the top border area.

This default behavior allows to keep the same height for your textedit as before - this is specially useful if you only use textedit to display rich text: the toolbox is only visible in input. If you want always display the toolbox, you can set the following styleAttribute:

```
<StyleAttribute name="textFormat" value="html" />
<StyleAttribute name="showEditToolBox" value="yes" />
```

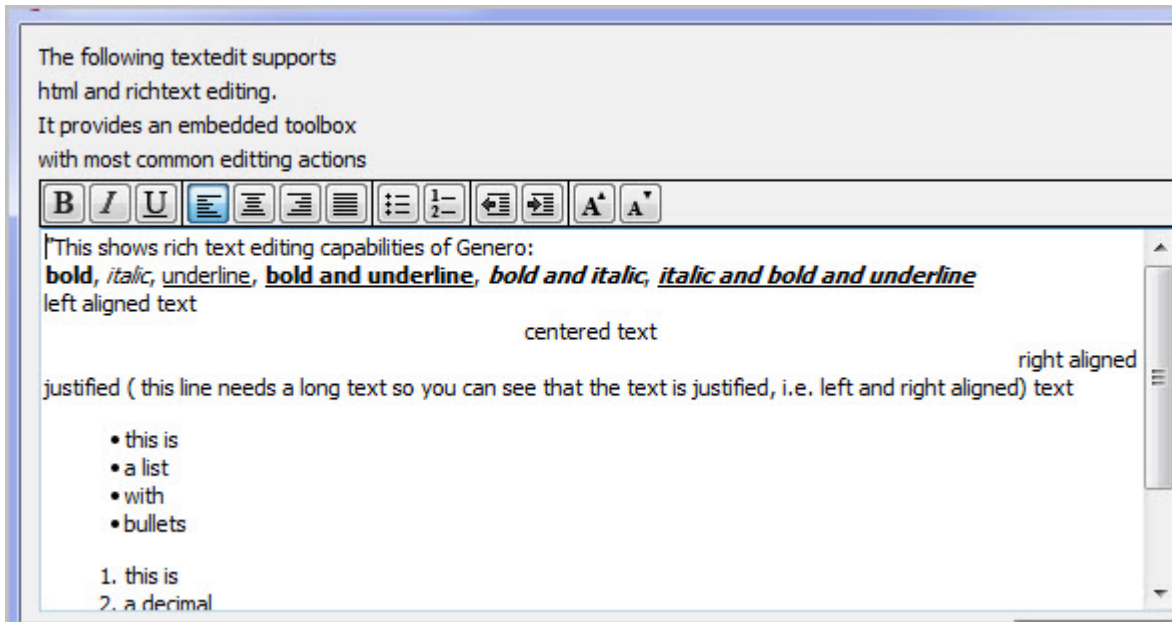


Figure 27: Rich text editing interface with toolbox always displayed.

Note:





- The textedit will be wide enough to display the toolbox entirely, even if you define a small width in the .per. Please take this in account when designing your form.
- The textedit will be high enough to display the number of lines defined in the .per (with the textedit font). This means that a textedit with a height of 1 will display toolbox and one line, which is much higher than without the toolbox.

Rich text local actions

Besides integrated toolbox, new local actions have been created for each rich text capability. As any local action, you can configure accelerator keys, and bind them to action views like ToolBar buttons.

Table 39: Local action names, accelerators, and icons

Name	Default Accelerator	Icon Name	Icon
richtextbold	control-b	textbold	B
richtextitalic	control-i	textitalic	<i>I</i>
richunderline	control-u	textunder	<u>U</u>
richtextalignleft	control-l	textleft	≡
richtextaligncenter	control-e	textcenter	≡
richtextalignright	control-r	textright	≡
richtextalignjustify	control-j	textjustify	≡
richtextlistbullet	None	textlistbullet	≡
richtextlistdecimal	None	textlistnumbered	1-2

Name	Default Accelerator	Icon Name	Icon
richtextdecreaseindent	None	textindentdecrease	
richtextincreaseindent	None	textindentincrease	
richtextdecreasefontsize	None	textfontsizeup	
richtextincreasefontsize	None	textfontsizeup	

Then you can hide the toolbox using the following styleAttribute:

```
<StyleAttribute name="textFormat" value="html" />
<StyleAttribute name="showEditToolBox" value="no" />
```

Important:

- We are not generating html code by ourselves. We are using a component dealing with rich text which provides a "toHTML" export. As we have nearly no way to influence the export, and are completely dependent on the component, future versions of GDC may behave differently if the component provider decides to change the export. Should this occur, we will add some entries in the Migration Guide.
- `cursor` and `cursor2` attributes are now following better html code, but they are still not 100% corresponding. For instance, if you load an html file with a hidden part, cursors will be wrongly set. We recommend using `cursor` and `cursor2` with care when `textFormat` is set to `html`.

Genero Desktop Client 2.21 New Features

This section describes the new features for Genero Desktop Client 2.21.

Table 40: 2.21 New features: General features

Overview	Reference
You can now run your GDC 2.11 application with GDC 2.21 and your application behavior will be the same as GDC 2.11. This can be useful if you need to mix FGL 2.11 and 2.2x installations: you only need to install GDC 2.21.	See the Genero Desktop Client 2.21 migration guide on page 91.
<p>The internal HTTP stack has been rewritten to support pre-2.20 features and HTTP retries. This enables pre-2.20 lost features like Kerberos Support, NTLM single Sign-on or Client certificate.</p> <p>GDC is also able to retry when network errors occur. This was already the case in previous (late MR) versions, but now it may retry in more situations (ex: in case of HTTP error 500). This can be useful if your network is not too reliable and sometimes messages may be discarded before reaching GAS or returning to GDC. You can now configure how GDC will retry:</p>	

Table 41: 2.21 New features: Windows

Overview	Reference
<p>The Genero Desktop Client automatically displays scrollbars around the form when the window is larger than the desktop size.</p> <p>This old feature has been completely reviewed to now work also when the window is not maximized. GDC will then automatically, when the window is larger, fit the size of the window to the desktop size (height or width) and display scrollbars around the form.</p> <p>Note: This feature should be used carefully. A Desktop application is not a web application, and having scrollbars (especially horizontal ones) around the form is not common. As GDC will try to always show the current field, this may lead to lots of scrolling when you move from one field to another if the fields are not all visible.</p> <p>We strongly recommend that you adapt your forms to the smallest desktop size you target; automatic scrollbars should only appear for "accident" cases.</p> <p>If you prefer avoid automatic scrollbars and retrieve the behavior of previous versions (only getting scrollbars when the current window is maximized) you can use the following style attribute:</p> <pre data-bbox="240 1129 797 1247"><Style name="Window.myWindow"> <StyleAttribute name="formScroll" value="no"> </Style></pre> <p>To achieve automatic scrollbars in a more stable way, the action frame (menu/dialog) has been reviewed. The new look is very slightly different, but the main behavior is the same.</p> <ul data-bbox="227 1423 833 1585" style="list-style-type: none"> • Navigations button are different if you are on the first or last row. • A "plus" button has been added to display in one click all remaining items. • A little animation shows scroll direction. 	
<p>As well as "maximized", windows can be started minimized using <code>minimized</code> as the value for the "windowState" style attribute.</p>	<p>See the <i>Windows style attributes</i> section in the <i>Genero Business Development Language User Guide</i></p>

Table 42: 2.21 New features: Widgets

Overview	References
DateEdit now has a presentation style named "showCurrentMonthOnly". This style configures whether the calendar shows only the current months, or displays (in light grey) days of the previous and next month.	See the <i>DateEdit style attributes</i> section in the <i>Genero Business Development Language User Guide</i>
You can now set a range on the SpinEdit widget with the new attributes valueMin and valueMax.	See the <i>SpinEdit item type</i> section in the <i>Genero Business Development Language User Guide</i>
Image field now supports style attribute "alignment" to define where the picture should be located when the container (widget) is bigger.	See the <i>Image style attributes</i> section in the <i>Genero Business Development Language User Guide</i>
Window supports new style attribute "toolBarDocking" to define if the toolbars are movable and floatable.	See the <i>Windows style attributes</i> section in the <i>Genero Business Development Language User Guide</i>

Table 43: 2.21 New features: Presentation styles

Overview	Reference
You can now set a point font size with a non-integer value, for example 8.3pt.	See the <i>Font sizes</i> section in the <i>Genero Business Development Language User Guide</i>

Table 44: 2.21 New features: Monitor

Overview	Reference
The Connection tab now displays both the application name and text.	See the Connection tab on page 15 topic.
Users now receive an error message if the number of user licenses is exceeded. User Limit exceeded is now a default terminal string, so the end user has feedback if the application can't start because of a license issue.	
Debugging information has been added to the debug console when a shortcut starts. Analyzing this log may help you to understand what GDC did, and why a connection may have failed.	
Genero Desktop Client 2.21 is now supported on Windows™ 7 platform.	
GDC 2.21 has been adapted to work with WinSSHd 5.0.9 (some changes in the server have been done by Bitwise too, so you'll need to upgrade the server part to at least version 5.0.9.)	

Table 45: 2.21 New features: FrontEnd functionCall

Overview	Reference
"Hardcopy" is now available as a frontCall.	See the <i>Hardcopy</i> section in the <i>Genero Business Development Language User Guide</i>
"launchurl" frontCall signature has the same signature in the Genero Web Client (GWC) and in the Genero Desktop Client (GDC)	See the <i>LaunchURL</i> section in the <i>Genero Business Development Language User Guide</i>

Table 46: 2.21 New features: Miscellaneous

Overview	Reference
Windows® Only: The HardCopy menu item is now available in the system menu for MDI child windows.	
Windows® now uses the MSI installer system.	
File association and start menu entries help to improve desktop integration: <ul style="list-style-type: none"> .gdc files are associated with the Genero Desktop Client to be run directly in your favorite explorer (Windows®, Linux®). Linux installer creates entries in your desktop start menu. 	

Table 47: 2.21 New features: Experimental features

Overview	Reference
It is now possible to see a Flash application in the pages you display with the Integrated browser. <p>Note: Note: This feature uses Netscape Plugin technology, which is also used by Mozilla Firefox or Google Chrome. So, you need to have Firefox, Chrome plugin, or stand-alone Adobe™ flash player installed. Having Microsoft™ IE plugin only is not enough.</p>	
Compositing allows you to make some fancy effects with window transparency. <p>Windows® Vista and Windows® 7 introduced Compositing. This allows you to make some fancy effects with window transparency. If you want a semi-transparent window in GDC, you may use the blurBackground style attribute:</p> <pre><Style name="Window.semitransparent"></pre>	

Overview	Reference
<pre data-bbox="240 205 816 289"><StyleAttribute name="blurBackground" value="yes" /> </Style></pre> <p data-bbox="224 321 834 485">See the classic demo application with a semi-transparent background, running on Windows® 7. This will not work on older version of Windows® (XP and before), but may work on Linux® depending on the windows manager capabilities.</p>	
<p data-bbox="224 510 818 636">Windows® Vista introduced a Command Link Button: a push button with a nicer design, showing the comment directly. When the mouse goes over the button, a nice gradient effect occurs.</p> <p data-bbox="224 653 797 747">If you want a Command Link Button in GDC, you may use "commandLink" as the value for the "buttonType" style attribute:</p> <pre data-bbox="240 779 797 894"><Style name="Button.commandLink"> <StyleAttribute name="buttonType" value="commandLink" /> </Style></pre> <p data-bbox="224 926 818 1020">You can now add a Windows® Vista (or Windows® 7) Command Link Button style in your application, and use the mouseover effect.</p> <p data-bbox="224 1037 824 1163">This will work on Windows® Vista / Windows® 7 only. On other systems it displays a simple button with the text and the comment, but without the nice mouseover effect.</p>	
<p data-bbox="224 1199 732 1262">Two new frontcalls have been added in the standard frontcall library:</p> <ul data-bbox="224 1278 818 1409" style="list-style-type: none"> • <code>storeSize</code> asks GDC to store the current size of the current window. • <code>restoreSize</code> asks GDC to restore the stored size. 	

General category: Genero Desktop Client 2.21 New Features

This section describes new features in the general category for Genero Desktop Client 2.21.

- [Internal HTTP stack has been rewritten](#) on page 73

Internal HTTP stack has been rewritten

The internal HTTP stack has been rewritten to support pre-2.20 features and HTTP retries.

This enables pre-2.20 lost features like Kerberos Support, NTLM single Sign-on or Client certificate.

GDC is also able to retry when network errors occur. This was already the case in previous (late MR) versions, but now it may retry in more situations (ex: in case of HTTP error 500). This can be useful if your network is not too reliable and sometimes messages may be discarded before reaching GAS or returning to GDC. You can now configure how GDC will retry:

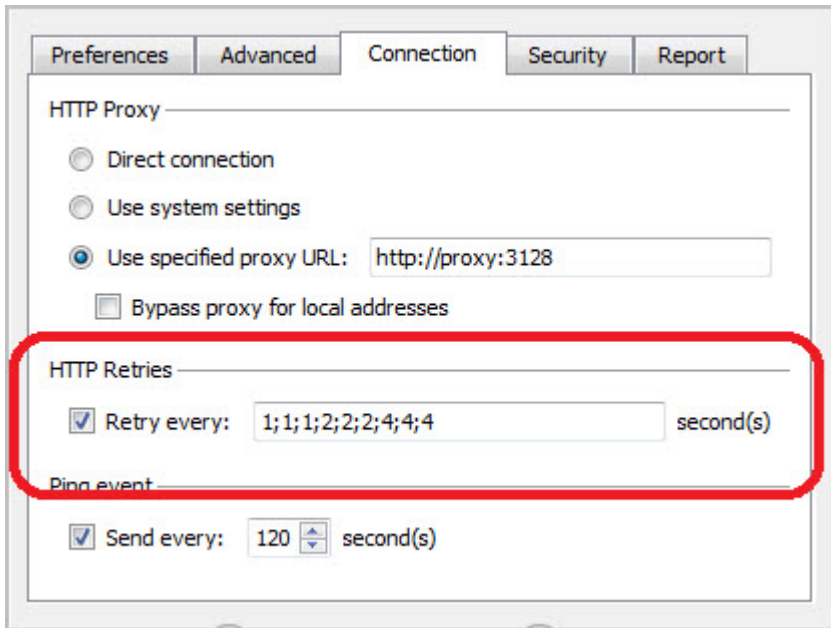


Figure 28: HTTP Retries section of Connection tab

The entry "1;1;1;2;2;2;4;4;4" means that GDC will retry 9 times. GDC will wait 1 second between the first three errors, then 2 seconds between the next three, and 4 seconds between the last three. GDC will then wait a maximum of 21 seconds before showing an error message.

Note:

This is the time GDC waits AFTER the system / network returns an error, not the complete waiting time. For instance, if the system needs time to answer (ex: connection timed out), GDC will wait:

1. for the system
2. for the configured wait time

GDC now shows some information in the systray icon when there is an HTTP connection issue.

windows: Genero Desktop Client 2.21 New Features

This section describes new windows features for Genero Desktop Client 2.21.

- [Automatic scrollbars](#) on page 74

Automatic scrollbars

The Genero Desktop Client automatically displays scrollbars around the form when the window is larger than the desktop size.

Maximized windows show scrollbars around the form if the window is larger than the desktop available size.

This old feature has been completely reviewed to now work also when the window is not maximized. GDC will then automatically, when the window is larger, fit the size of the window to the desktop size (height or width) and display scrollbars around the form.

Note: This feature should be used carefully. A Desktop application is not a web application, and having scrollbars (especially horizontal ones) around the form is not common. As GDC will try to always show the current field, this may lead to lots of scrolling when you move from one field to another if the fields are not all visible.

We strongly recommend that you adapt your forms to the smallest desktop size you target; automatic scrollbars should only appear for "accident" cases.

If you prefer avoid automatic scrollbars and retrieve the behavior of previous versions (only getting scrollbars when the current window is maximized) you can use the following style attribute:

```
<Style name="Window.myWindow">
  <StyleAttribute name="formScroll" value="no">
</Style>
```

To achieve automatic scrollbars in a more stable way, the action frame (menu/dialog) has been reviewed. The new look is very slightly different, but the main behavior is the same.

- Navigations button are different if you are on the first or last row.
- A "plus" button has been added to display in one click all remaining items.
- A little animation shows scroll direction.

Experimental features: Genero Desktop Client 2.21 New Features

This section introduces experimental new features for Genero Desktop Client 2.21.

Caution:

This version contains a few experimental features. Experimental features are available in the product, but:

- they are likely to be changed in future versions, or even simply removed from the product.
- they are not supported. We won't be able to fix all reported issues; most of the time, this is due to current technical limitations.
- they may not work 100%, not on all platforms, and likely not with all Front-Ends.

These experimental features are provided 'as is', so you can play with them and return your feedback, but we may not be able to fix an issue, or we may even remove the functionality if a serious side effect / performance issue is seen. They are usually based on unfinished work, or on third party tools on which we can't rely 100%. But, we believe it's nice to have them in the product and to show you today what we're likely to be able to do tomorrow. You can use them freely, but at your own risk.

- [Flash support](#) on page 75
- [Compositing](#) on page 75
- [Windows Vista introduced a Command Link Button](#) on page 76
- [Save and restore current window size](#) on page 76

Flash support

You can now watch your favorite YouTube video in your 4GL application.

It is now possible to see a Flash application in the pages you display with the [Integrated browser](#).

Note: This feature uses "Netscape Plugin" technology, which is also used by Mozilla Firefox or Google Chrome. So you need to have Firefox, Chrome plugin or stand-alone Adobe™ flash player installed. Having Microsoft™ IE plugin only is not enough.

Compositing

Compositing allows you to make some fancy effects with window transparency.

Windows™ Vista and Windows™ 7 introduced [Compositing](#).

This allows you to make some fancy effects with window transparency. If you want a semi-transparent window in GDC, you may use the "blurBackground" style attribute:

```
<Style name="Window.semitransparent">
  <StyleAttribute name="blurBackground" value="yes" />
</Style>
```

See the classic demo application with a semi-transparent background, running on Windows™ 7. This will not work on "old" windows (XP and before), but may work on Linux™ depending on the windows manager capabilities.

Windows™ Vista introduced a Command Link Button

Windows™ Vista introduced a Command Link Button: a push button with a nicer design, showing the comment directly. When the mouse goes over the button, a nice gradient effect occurs.

If you want a Command Link Button in GDC, you may use "commandLink" as the value for the "buttonType" style attribute:

```
<Style name="Button.commandLink">
  <StyleAttribute name="buttonType" value="commandLink"/>
</Style>
```

You can now add a Vista (or Windows™ 7) Command Link Button style in your application, and use the mouseover effect.

This will work on Vista / Windows™ 7 only. On other systems it displays a simple button with the text and the comment, but without the nice mouseover effect.

Save and restore current window size

Two new frontcalls have been added in the standard frontcall library: storeSize and restoreSize.

- **storeSize** asks GDC to store the current size of the current window.
- **restoreSize** asks GDC to restore the stored size.

This allows you to create the classic GUI with Show/Hide details.

When **show** is clicked, the window grows to show more information. When **hide** is clicked, the window returns to its original size.

```
ON ACTION details
  IF state = 1 THEN
    CALL f.setElementHidden("g2",1)
    CALL f.setElementText("details",&Show details")
    CALL ui.interface.frontCall("standard","restoreSize",[200],[ret])
    LET state = 0
  ELSE
    CALL ui.interface.frontCall("standard","storeSize",[],[ret])
    CALL f.setElementHidden("g2",0)
    CALL f.setElementText("details",&Hide details")
    LET state = 1
  END IF
```

The restoreSize frontcall takes an optional parameter to define the delay (in milliseconds) used to revert the window size. The window will then smoothly shrink or grow to reach the saved size instead of having its new size immediately.

Note:

- Calling restoreSize without calling storeSize, or on a different window, has no effect.
- The stored size is a desired size; the layout has always higher priority. For instance, if the saved size is 800x600 and the content of the window is 1024x768, GDC will not be able to shrink to the expected size.

Genero Desktop Client 2.20 New Features

This section describes the new features for Genero Desktop Client 2.20.

Table 48: 2.20 New features: General

Overview	Reference
Qt4 is now the internal library used for the Genero Desktop Client.	See Genero Desktop Client 2.20 migration guide

Overview	Reference
<p>SVG image format is now supported in two ways:</p> <ul style="list-style-type: none"> • If an SVG image is displayed to an IMAGE field (or static IMAGE), an SVG renderer is used. If the widget is resized (according to STRETCH and AUTOSCALE attributes), the image is resized without resize artifacts. • On other items, as they can't be resized, the SVG image is used as a pixmap, as well as all other image formats. 	<p>See the <i>Providing the image resource topic</i> in the <i>Genero Business Development Language User Guide</i>.</p>
<p>Images used in GDC are now copied and stored locally, to accelerate image lookup. The cache can be configured (enabled, cache size, and so on) in the 'Advanced' tab.</p>	<p>See Advanced tab</p>

Table 49: 2.20 New features: Monitor/Shortcut mechanism

Overview	Reference
<p>A customized login box can be created for shortcuts.</p>	<p>See Creating Shortcuts using the Shortcuts Wizard</p>
<p>A command line option, "-i" or "--ignoreSettings", has been added to force ignore settings.</p>	<p>See Command line options</p>
<p>A read only stored settings option has been added on the Advanced Tab. Settings are applied when loading a form, but they are not modified when closing the form.</p>	<p>See Advanced tab</p>
<p>A command line option, "-r [filename.log]" or "--logrec [filename.log]", has been added for recording a log.</p>	<p>See Command line options</p>
<p>SSH2 is default protocol type when -T is not set.</p>	<p>See Command line options and The Shortcut System</p>
<p>Shortcuts can be exported as and imported from a Genero Desktop Client (gdc) file. GDC can also be started with a .gdc file and the shortcut starts directly.</p>	<p>See The Shortcut System</p>
<p>The security level now ensures the application started by the Genero Desktop Client is the one you started.</p>	<p>See Security levels.</p>
<p>Several Connections and Terminals can be selected and closed (instead of closing them one by one).</p>	<p>See the The Connections Panel?</p>
<p>Default Proxy and Kerberos Realm can be defined in the Options / Connection panel. This information</p>	<p>See the The Connections Panel.</p>

Overview	Reference
is used when GDC connects to GAS as well as when GDC is looking for an image.	

Table 50: 2.20 New features: FrontEnd functionCall

Overview	Reference
The functionCall Standard <code>getWindowId</code> has been added.	See the <i>getWindowId</i> topic in the <i>Genero Business Development Language User Guide</i> .
Four new parameters have been added for the functionCall Standard <code>feInfo</code> : <ul style="list-style-type: none"> • <code>osversion</code> • <code>numscreens</code> • <code>screenresolution</code> • <code>fepath</code> 	See the <i>feInfo</i> topic in the <i>Genero Business Development Language User Guide</i> .

Table 51: 2.20 New features: Widgets

Overview	Reference
You can add a TreeView widget in your Genero application, based on a simple DISPLAY ARRAY.	See the <i>Tree views</i> topic in the <i>Genero Business Development Language User Guide</i> .
HTML hyperlinks are now managed in Labels, when styleAttribute <code>textFormat</code> is set to "html" .	See the <i>TextEdit style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
TextEdit can have an integrated search facility with style <code>integratedSearch</code> .	See the <i>TextEdit style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
A spelling checker is included for TextEdits.	See the <i>TextEdit style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
Folder nodes support a <code>position</code> style attribute.	See the <i>Message style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
Window style attributes have been introduced to define the decoration for the ActionFrame: <code>actionPanelDecoration</code> and <code>ringMenuDecoration</code> .	See the <i>Window style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
Menu "popup" supports a "position" style attribute.	See the <i>Menu style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
You can now use the mouse wheel to change the focus of the menu.	No reference.
The Button node now supports a "buttonType" style attribute.	See the <i>Button style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .

Overview	Reference
MDI Containers can now be displayed as "Tabbed MDI" (like FireFox / IE7) with two new style attributes: <code>tabbedContainer</code> and <code>tabbedContainerCloseMethod</code> .	See the <i>Window style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .

Table 52: 2.20 New features: Tables

Overview	Reference
<p>MultiSelection in DISPLAY ARRAY enables you to select different lines using the usual key combination (SHIFT, CTRL, and so on)</p> <pre> DISPLAY ARRAY ar TO sr.* BEFORE DISPLAY CALL DIALOG.setSelectionMode("sr", 1) END DISPLAY </pre>	
Columns can be sorted in INPUT ARRAY.	See the <i>Editable record list (INPUT ARRAY)</i> section in the <i>Genero Business Development Language User Guide</i> .
The TABLE widget can be used to render a Picture Flow, a widget to display images with an animated transition effect.	See the <i>Table style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .
When you resize a table and make it wider than the size of the content, an empty space can appear on the right side. You can now set the styleAttribute <code>resizeFillsEmptySpace</code> to "yes" and the last column will be automatically resized to fill the empty space.	See the <i>Table style attributes</i> topic in the <i>Genero Business Development Language User Guide</i> .

Table 53: 2.20 New features: Action mechanisms

Overview	Reference
To adapt to Multiple Dialogs and <code>ON ACTION ... IN FIELD</code> , the LocalAction mechanism has been slightly revised.	See the <i>ON ACTION block</i> topic in the <i>Genero Business Development Language User Guide</i> .
The Genero Desktop Client creates local actions prefixed by the screen record, for <code>instancesr1.nextrow</code> . This allows you to create ActionViews bound to a specific screen record, such as a dedicated navigation panel for a specific table.	

Table 54: 2.20 New features: Experimental features

Overview	Reference
<p>Look and Feel</p> <p>The <code>lookAndFeel</code> style attribute allows you to customize your application.</p>	<p>TOPIC TO BE REVIEW: https://intranet.4js.com/distrib/manuals/FOURJS/FJSONLY/trunk/fjs-gdc-latest-manual-html-draft/index.html#c_gdc_NewFeatures220_experimental_features_look</p>
<p>Form Layout</p> <p>The Form Layout is a style applied to a <code>GRID</code> that lets the Genero Desktop Client display the content of the section, ignoring the per alignment and so on. Simple fields are bound to their labels using the <code>TITLE</code> attribute.</p>	<p>TOPIC TO BE REVIEW: https://intranet.4js.com/distrib/manuals/FOURJS/FJSONLY/trunk/fjs-gdc-latest-manual-html-draft/index.html#c_gdc_NewFeatures220_experimental_features_form</p>
<p>Integrated Browser</p> <p><code>TextEdits</code> can display more or less complex html data, but can't act as a real browser. With styleAttribute <code>imageContainerType Image</code> style attribute set to "browser", your image container becomes a browser.</p> <p>Instead of setting an image name, you can set a URL.</p>	<p>TOPIC TO BE REVIEW: https://intranet.4js.com/distrib/manuals/FOURJS/FJSONLY/trunk/fjs-gdc-latest-manual-html-draft/index.html#c_gdc_NewFeatures220_experimental_features_integ</p>

Widgets: Genero Desktop Client 2.20 New Features

This section describes widget-related new features for Genero Desktop Client 2.20.

- [Treeview widget support](#)
- [TextEdit: Spell Checker](#) on page 80
- [Action Frame \(Action Panel and Ring Menu\): StyleAttribute for decoration](#)
- [Button: ButtonType style attribute to customize Buttons](#)
- [Tables: Genero Desktop Client 2.20 New Features](#) on page 81

TextEdit: Spell Checker

A spelling checker is included for `TextEdits`.

The spelling checker uses LGPL Hunspell (see <http://hunspell.sourceforge.net/>) which is the default spell checker of OpenOffice.org or Mozilla tools.

You need to provide the Genero Desktop Client (GDC) the two dictionary files needed for each language. These files can be downloaded from <http://extensions.openoffice.org/>. Extract the files in the `oxt` archive.

The files can be stored on the Runtime System side and can be uploaded to GDC using a simple `FGL_PUTFILE`. Then specify in the style the two files for the "spellCheck" `StyleAttribute`, using the following format:

```
"<affix file>|<dictionary file>"
```

By default, files are read from where you're executing GDC. You can, however, specify an absolute path:

```
<Style name="TextEdit.spellUs">
<StyleAttribute name="spellCheck"
  value="file:///c:/dics/en_US.aff|file:///c:/dics/en_US.dic" />
</Style>
```

GDC will underline unknown words, and a right click on the unknown word will add the list of suggestions for this word to the contextual menu.

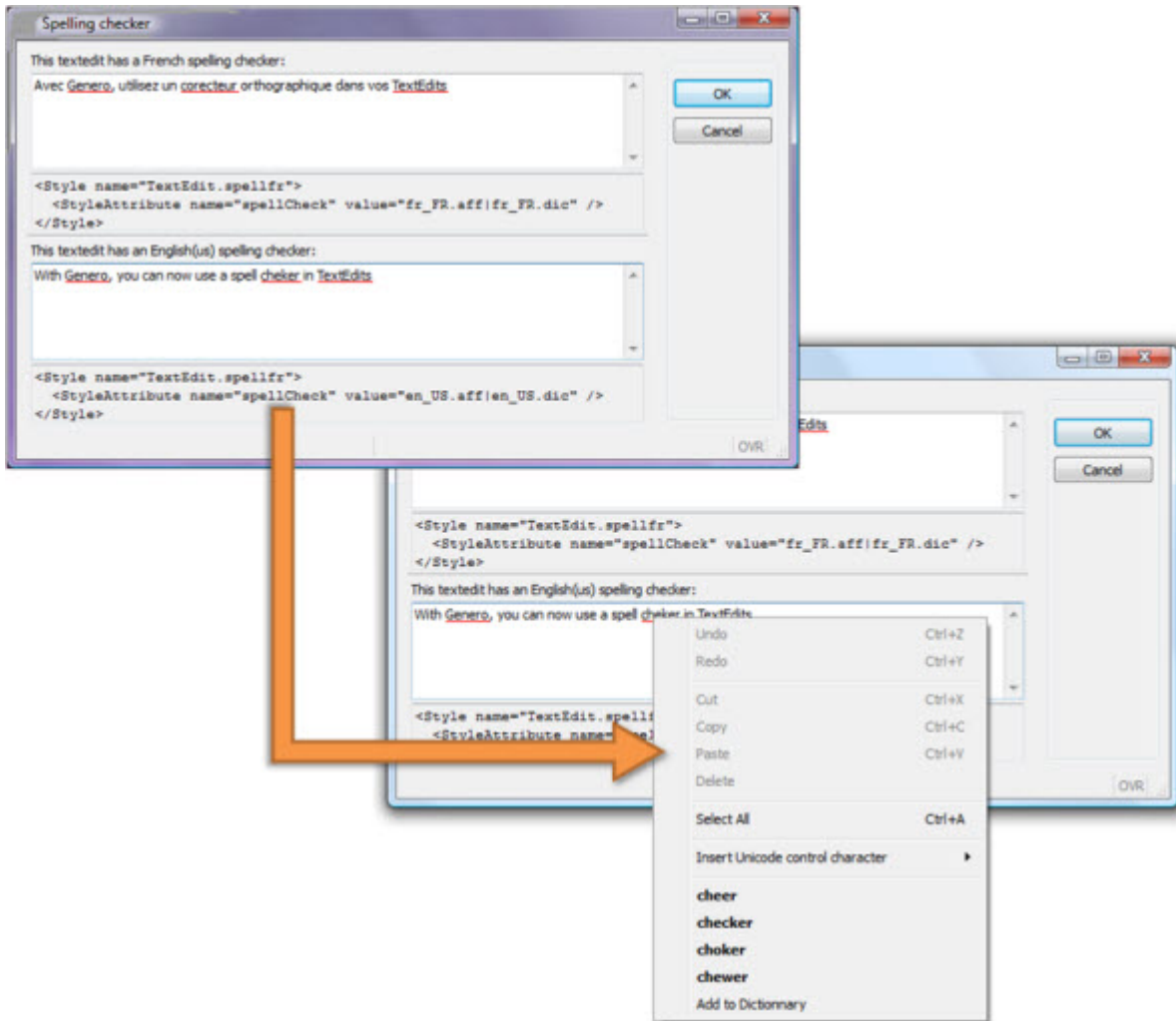


Figure 29: Spell checker window flow

```
<Style name="TextEdit.spellUS">
<StyleAttribute name="spellCheck" value="en_US.aff|en_US.dic" />
</Style>
```

Note:

- Loading the dictionary takes time (Approx. 5 seconds for the French dictionary for instance). This is why the spell checking may not be immediate when loading the form.
- If there is an error loading dictionary files, the checking will simply not be done. No error will be reported on the 4GL side.

Tables: Genero Desktop Client 2.20 New Features

- [Tables: MultiSelection in DISPLAY ARRAY](#)
- [Tables: Sort in INPUT ARRAY](#)
- [Tables: resizeFillsEmptySpace styleAttribute](#) on page 81

Tables: resizeFillsEmptySpace styleAttribute

When you resize a table and make it wider than the size of the content, an empty space can appear on the right side.

Set the styleAttribute `resizeFillsEmptySpace` to "yes" and the last column will be automatically resized to fill the empty space.

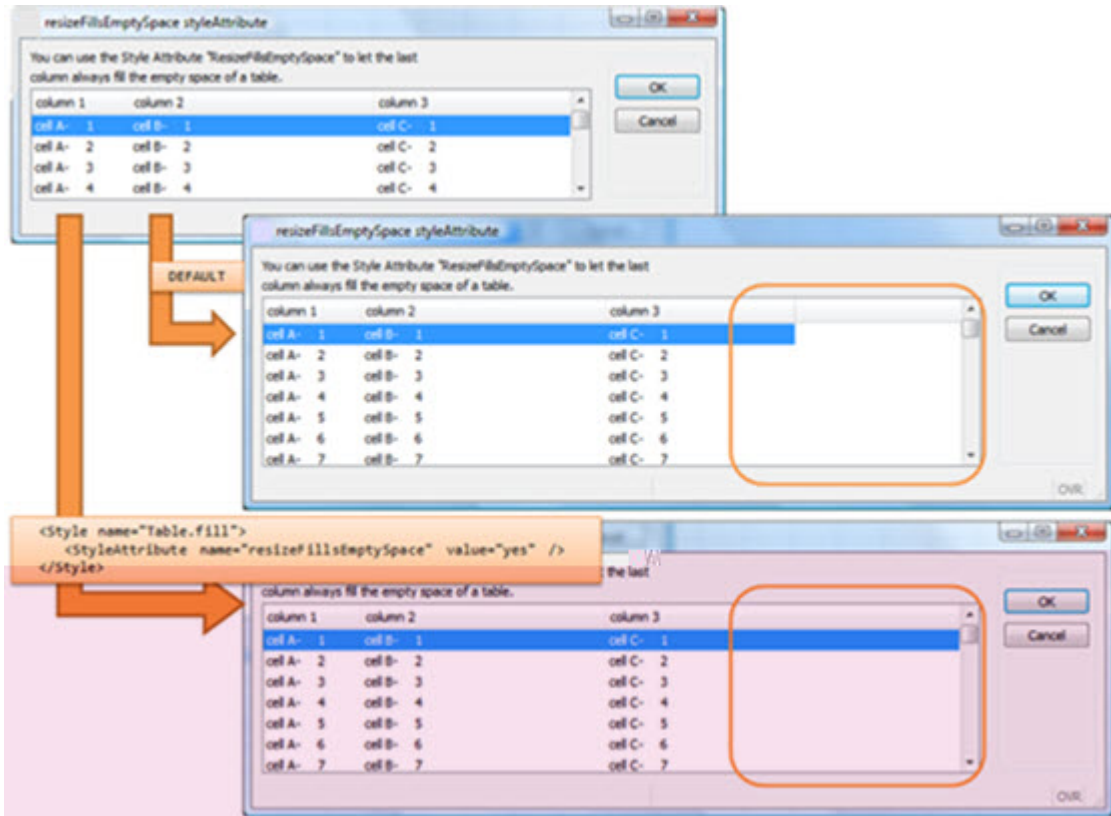


Figure 30: resizeFillsEmptySpace: Before and After

Action mechanisms: Genero Desktop Client 2.20 New Features

This section introduces action mechanisms new features for Genero Desktop Client 2.20.

- [Qualified Local Actions](#) on page 82

Qualified Local Actions

The Genero Desktop Client creates *qualified local actions*.

The Genero Desktop Client creates local actions prefixed by the screen record, for instances `sr1.nextrow`. This allows you to create ActionViews bound to a specific screen record, such as a dedicated navigation panel for a specific table.

Experimental features: Genero Desktop Client 2.20 New Features

This section introduces experimental new features for Genero Desktop Client 2.20.

Caution: This version contains a few experimental features. Experimental features are available in the product, but:

- they are likely to be changed in future versions, or even simply removed from the product.
- they are not supported. We won't be able to fix all reported issues; most of the time, this is due to current technical limitations.
- they may not work 100%, not on all platforms, and likely not with all Front-Ends.

These experimental features are provided 'as is', so you can play with them and return your feedback, but we may not be able to fix an issue, or we may even remove the functionality if a serious side effect / performance issue is seen. They are usually based on unfinished work, or on third party tools on which we can't rely 100%.

But, we believe it's nice to have them in the product and to show you today what we're likely to be able to do tomorrow. You can use them freely, but at your own risk.

- [Look and Feel](#) on page 83

- [Form Layout](#) on page 83
- [Integrated Browser](#) on page 84

Look and Feel

The `lookAndFeel` style attribute allows you to customize your application.

All widgets used by GDC can follow a given style, which defines how a widget must be drawn on the screen. By default, it uses the system platform style. The internal style used by GDC will be modified.

If you are a skilled C++ developer, you can build your own style, following [Qt's guidelines](#). This gives you an example of what can be done.

A few built-in styles are provided; the list varies based on the platform where the GDC runs. You can view the list of built-in styles provided for a platform by viewing the **Information** tab in the Genero Desktop Client console's **About** box.

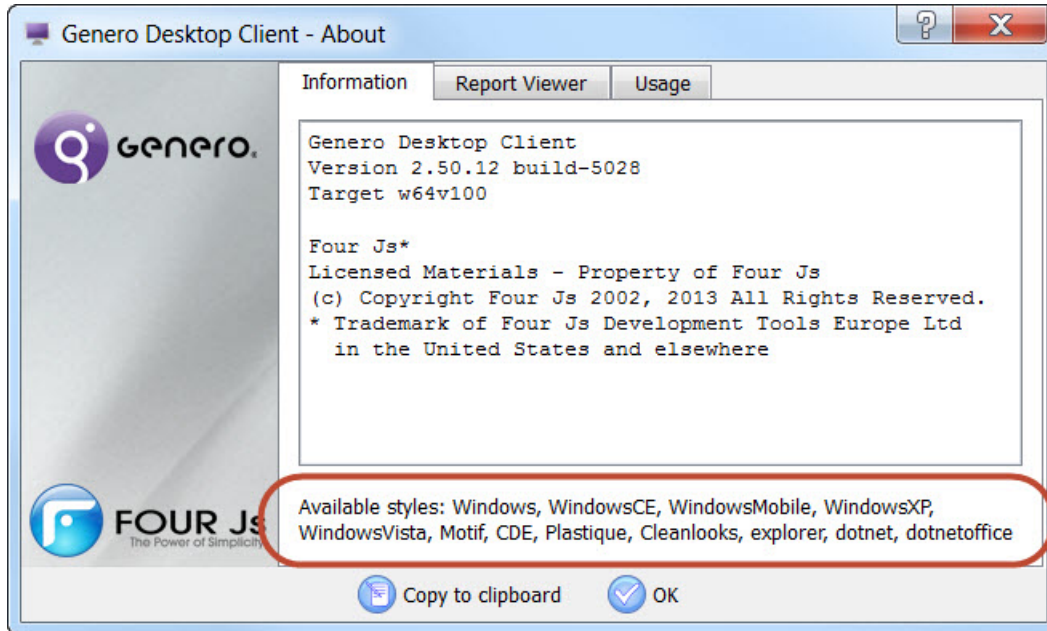


Figure 31: Viewing the list of look-and-feel presentation styles

Example

This example of a 4st file defines a window style using a "dotnetoffice" look-and-feel.

```
<Style name="Window.Office">
  <StyleAttribute name="lookAndFeel" value="dotnetoffice" />
</Style>
```

Form Layout

The Form Layout is a style applied to a GRID that lets the Genero Desktop Client display the content of the section, ignoring the `per` alignment and so on. Simple fields are bound to their labels using the `TITLE` attribute.

Example

The GRID section only contains the fields:

```
GRID (style="flayout")
{
  [firstname  ]
  [lastname  ]
}
```

```

    [address      ]
    [              ]
    [postcode    ]
    [city         ]
  }
END

```

The ATTRIBUTES section defines the TITLES:

```

ATTRIBUTES
EDIT  firstname  = FORMONLY.firstname, TITLE("&First Name:");
EDIT  lastname   = FORMONLY.lastname,  TITLE("&Last Name:");
TEXTEDIT address = FORMONLY.address,  TITLE("&Address:", stretch=y;
EDIT  postcode  = FORMONLY.postcode,  TITLE("&ZipCode:", PICTURE="#####");
COMBOBOX city   = FORMONLY.city,     TITLE("&City:",
      ITEMS=("Paris", "Strasbourg", "Erfurt"));

```

The 4st style file specifies the form layout:

```

<Style name="Grid.flayout">
  <StyleAttribute name="layoutType" value="form" />
</Style>

```

A form will be created, associating fields and their title. The alignment of the titles depends on the platform; if the title contains "&", pressing ALT + the following letter will set the focus on the attached field.

Integrated Browser

TextEdits can display more or less complex html data, but can't act as a real browser. With styleAttribute imageContainerType Image style attribute set to "browser", your image container becomes a browser.

Instead of setting an image name, you can set a URL.

Example

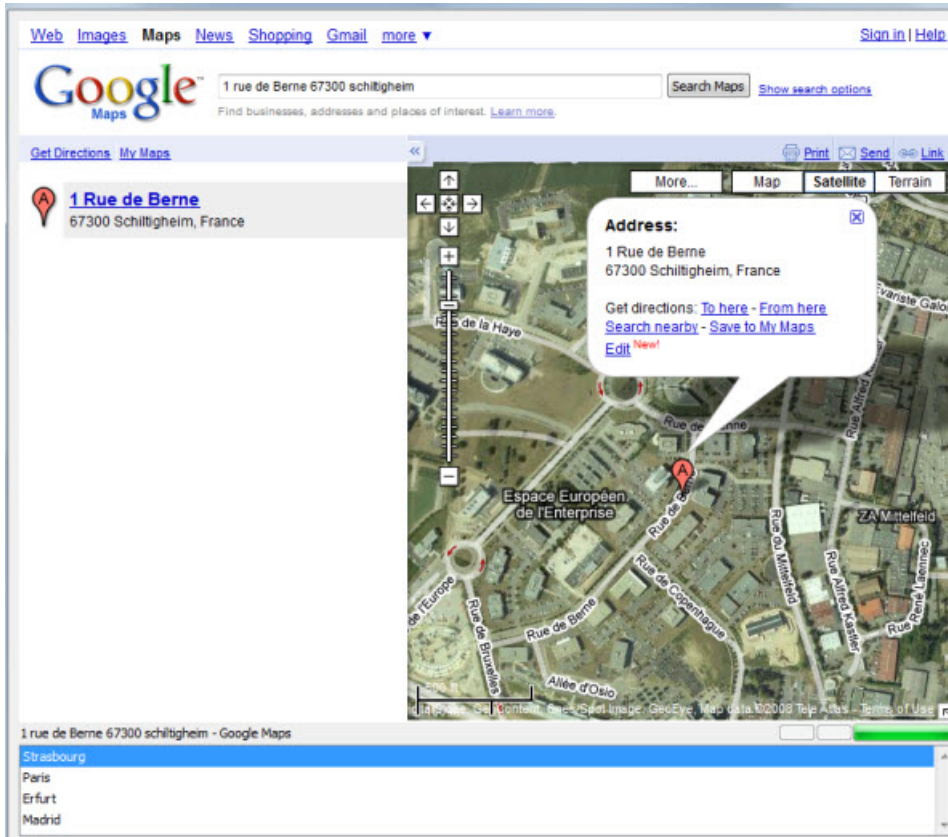


Figure 32: Google maps screenshot

Here is the per file:

```
LAYOUT (TEXT="Google Maps")
  GRID
  {
  [map
  [
  ]
  [
  ]
  [
  ]
  [
  ]
  [
  ]
  [
  ]
  [
  ]
  ]
  <T
  t
  [location
  [location
  [location
  ]
  ]
  ]
  END
  END
  ATTRIBUTES
    IMAGE map = FORMONLY.map, STRETCH=BOTH, style="map";
    TABLE t : WANTFIXEDPAGE SIZE, style="list";
    EDIT location = FORMONLY.location;
  END
```

```
INSTRUCTIONS
SCREEN RECORD sr(location);
```

Here is the corresponding 4st file:

```
<Style name="Image.map">
  <StyleAttribute name="imageContainerType" value="browser" />
</Style>
```

Important: This feature uses the current [WebKit](#) Open Source project; we use the version provided with Qt and we've no control over it. The aim is strictly to be able to display some HTML/Rich Text, not the most complicated pages of the web. Indeed, this feature comes with limitations, such as no Java™ support. We expect the version to be better supported by Qt in the future.

Upgrade Guides for the Genero Desktop Client

Each upgrade guide is an incremental upgrade guide that covers only topics related to a specific version of Genero. It is important that you read all of the upgrade guides that sit between your existing version and the desired version.

- [Genero Desktop Client 2.5x upgrade guide](#) on page 87
- [Genero Desktop Client 2.4x upgrade guide](#) on page 87
- [Genero Desktop Client 2.3x upgrade guide](#) on page 88
- [Genero Desktop Client 2.2x upgrade guide](#) on page 90

Genero Desktop Client 3.0 upgrade guide

This section describes differences you may encounter when upgrading to Genero Desktop Client 3.0.

Genero Desktop Client ActiveX (GDCAX) is desupported.

For new development, use GWC for JavaScript.

New location for configuration files

Starting with version 3.0, the default location of configuration files has changed. See [GDC configuration file directories](#) on page 17. Ensure that any administrative tasks relating to the configuration files take account of the new location.

Having users share a GDC configuration file

Starting with version 3.0, the GDC writes its configuration file (`config.xml`) in the User directory; see [GDC configuration file directories](#) on page 17 for the OS-specific path.

If you have been providing your end users with a shared drive installation of the GDC where multiple users (by default) share the same configuration file, you must act. To have users continue to share a common configuration file, use the `--config` option to specify the shared file. See [Apply an additional configuration file](#) on page 17.

Default images no longer in `/pics` directory

Starting with version 3.0, the `/pics` directory no longer exists in the GDC installation path. If you were using the default images from the `/pics` directory and you are still using Genero 2.5x, the images will not display. Genero 2.5x does not centralize default images on the runtime side. For further information, see the 3.0 upgrade topics in the *Genero Business Development Language User Guide*.

Genero Desktop Client 2.5x upgrade guide

This section describes differences you may encounter when upgrading to GDC 2.5x.

Starting with the Genero Desktop Client 2.50, The Genero Desktop Client ActiveX (GDCAX) is deprecated. It is recommended that you use the Genero Web Client for HTML5 instead.

Genero Desktop Client 2.50 migration guide

This topic lists differences you may encounter when upgrading to GDC 2.50.

Genero Desktop Client ActiveX (GDCAX) is deprecated

It is recommended that you migrate any GDCAX applications to use the Genero Web Client for HTML5.

Genero Desktop Client 2.4x upgrade guide

This section describes differences you may encounter when upgrading to GDC 2.4x.

Genero Desktop Client 2.40 migration guide

This topic lists differences you may encounter when upgrading to GDC 2.40.

WinDDE prevents simultaneously connecting twice with the same identifier (program + document)

WinDDE is now preventing from connecting twice simultaneously with a same identifier. This identifier is a concatenation of the *program* and the *document*. For better understanding to what *program* and *document* refer to in WinDDE terms, see the WinDDE documentation in the *Genero Business Development Language User Guide*. For instance, it may correspond to "EXCEL"+"Sheet1" on Microsoft™ Excel.

This change has been made because the various DDE objects are stored based on the identifier. When connecting a second time with the same identifier, the previously created DDEObjects were simply lost, as there were no longer references to them. You can no longer call DDEConnect twice with the same identifier without calling DDEFinish(All) in between.

GDC now listens to localhost only (default behavior)

Before 2.40, GDC opens a network server listening to any connection on a given port (6400 by default). This means that anyone can connect to GDC (depending on your firewall settings). The [Security Level](#) mechanism protects your installation, but the tcp port is still open.

GDC 2.40 default behavior is now to listen to the local interface only. Any connection from an other computer will be (by default) denied.

Connection from localhost (for developers having the DVM on the same machine, and when using SSH and port forwarding) will continue working as before. In order to facilitate GDC use for installations not using SSH and port forwarding, GDC will automatically listen to all interfaces when a direct shortcut (except for ssh+portforwarding) is started, therefore existing configurations are still working as before. This "any interface" server will be shut down a couple of minutes after the last connection (application or terminal) is over.

The recommended way to set up a direct connection is to use SSH and automatic port forwarding. In this configuration, your connection is secured by SSH, the communication is encrypted, and as GDC will be listening to the local interface only. No Firewall popup will occur when starting GDC the first time.

Override the default behavior with the command line argument `--listen`.

- `--listen ANY` makes GDC 2.40 work as before, listening to any connection.
- `--listen NONE` prevents GDC 2.40 listening at all. This is suitable when you are running HTTP connections only.
- `--listen LOCAL` makes GDC 2.40 listen to localhost only. This is suitable when you are running SSH + port forwarding and local development only.

The `-D` flag (to activate debug mode) implies the `--listen ANY` mode, to ease development. See [GDC command line](#) for more details.

Modal window behavior change

If you create a window using the `windowType=modal` StyleAttribute, GDC now considers this window as a real modal window:

- the window is modal to a base window (the previous current window)
- the modal window has no entry in the taskbar, and will not appear in the ALT+TAB (or windows KEY +TAB for Windows™ 7) sequence. Only the base window will appear.

Usually, the "modal chain" must be respected:

- it should be forbidden to close the base window without closing the modal window
- it should be forbidden to make current a non modal window if a modal window is displayed

To ease the migration from earlier versions, GDC will handle these cases by removing the modality of the window.

Title of an horizontal menu is now visible

When setting a title for an horizontal menu (by setting the style attribute `ringMenuPosition` to `top` or to `bottom`), this title is now visible on the form. It was not the case in previous GDC versions. Nevertheless, if the menu doesn't contain any button (because for instance all action views reported are on the ToolBar), the title will be automatically hidden.

Genero Desktop Client 2.3x upgrade guide

This section describes differences you may encounter when upgrading to GDC 2.3x.

Genero Desktop Client 2.30 migration guide

This section describes differences you may encounter when upgrading to GDC 2.30.

RichText: html generation

GDC is now based on the Qt 4.6 line and we've noticed small changes in HTML produced by `textedits`:

- `ol` and `ul` (decimal and bullet list) tags now have margin information in style

Before:

```
style="-qt-list-indent: 1;"
```

Now:

```
style="margin-top: 0px; margin-bottom: 0px; margin-left: 0px; margin-right: 0px; -qt-list-indent: 1;"
```

Experimental frontcall (re)store size changes

GDC 2.21 introduced experimental [front calls to store and restore window size](#). To follow existing front call names, the two front calls `storeSize` and `restoreSize` have been renamed in lower case: `storesize` and `restoresize`.

End of RLOGIN protocol support

RLOGIN is an old and unsecured remote connection protocol. Until now, it was supported for legacy reasons, mainly to have an easy direct `fglty` connection on really old UNIX™ servers which didn't have any access to a decent SSH server. SSH2 is now the default and recommended protocol for an `fglty` direct connection.

The RLOGIN protocol raises serious issues, the most serious being the need of "root" privileges on UNIX™ to be able to open a tcp port below 1024. This creates a real security hole, because malicious code could take advantage of fgltty being launched as root to damage or take control of the system. In addition, more and more UNIX™ desktop environments (Gnome, KDE, MacOS X, ...) simply forbid such programs to run in a graphical environment without being explicitly "accepted" by the end user (most forbid "sticky bit" completely and display a login box asking for the root password, which nullifies the passwordless login capability of RLOGIN). Moreover, as with TELNET, RLOGIN doesn't encrypt the communication, so passwords or other sensitive data are transmitted in "clear" channel through the network, another major flaw of the protocol.

For these reasons, it has been decided to remove rlogin support from GDC / fgltty.

End of RCP support

Such as RLOGIN, RCP is now de-supported mainly for security reasons. When RCP was enabled, GDC was allowing any rcp (remote copy) command even if it was not started by a 4GL program. Thus, you were likely to get some unexpected contents. For a similar result, you should rather use FGL_PUTFILE().

Linux®: minimum libc is 2.4

To support fancy GUI features introduced by recent Window Managers like KDE 4, Qt (and therefore GDC) needs to be compiled on a Linux® libc 2.4 machine instead of 2.3. As a result:

- If you were running GDC on a Linux® libc 2.3; GDC will not start anymore. You'll have to upgrade your system (glibc 2.4 has been released in 2006; we think that for desktop applications like GDC it's better to support recent WM features instead of old systems).
- The GDC theme may change; if you're running a libc 2.4 Linux™ and a theme which needs libc 2.4 features, GDC 2.2x was not able to load it. Now that GDC uses libc2.4 it will be supported, and the default look and feel will be adapted to your theme.

DateEdit: default date change

GDC 2.30 now supports the [INCLUDE attribute for DateEdit](#). If the field is NULL, the default date of the calendar will be the first date defined by the INCLUDE attribute, if the current date is not included. This prevents opening the calendar with a date you can't select 10 years later than the last accepted date.

Table: default different font size taken in account

By default, some systems are using a different font for table items than for simple form elements. For instance, on Windows™ 7, while the default font is Ms Shell Dlg, 8.25, Table font is Segoe UI, 9. GDC 2.2x is already using the different fonts, but has two issues:

- if you simply change the font size (e.g. to 12), the font family also changes (it becomes Ms Shell Dlg, 12 while it should be Segoe UI, 12)
- the row height was computed taking the wrong font size in account.

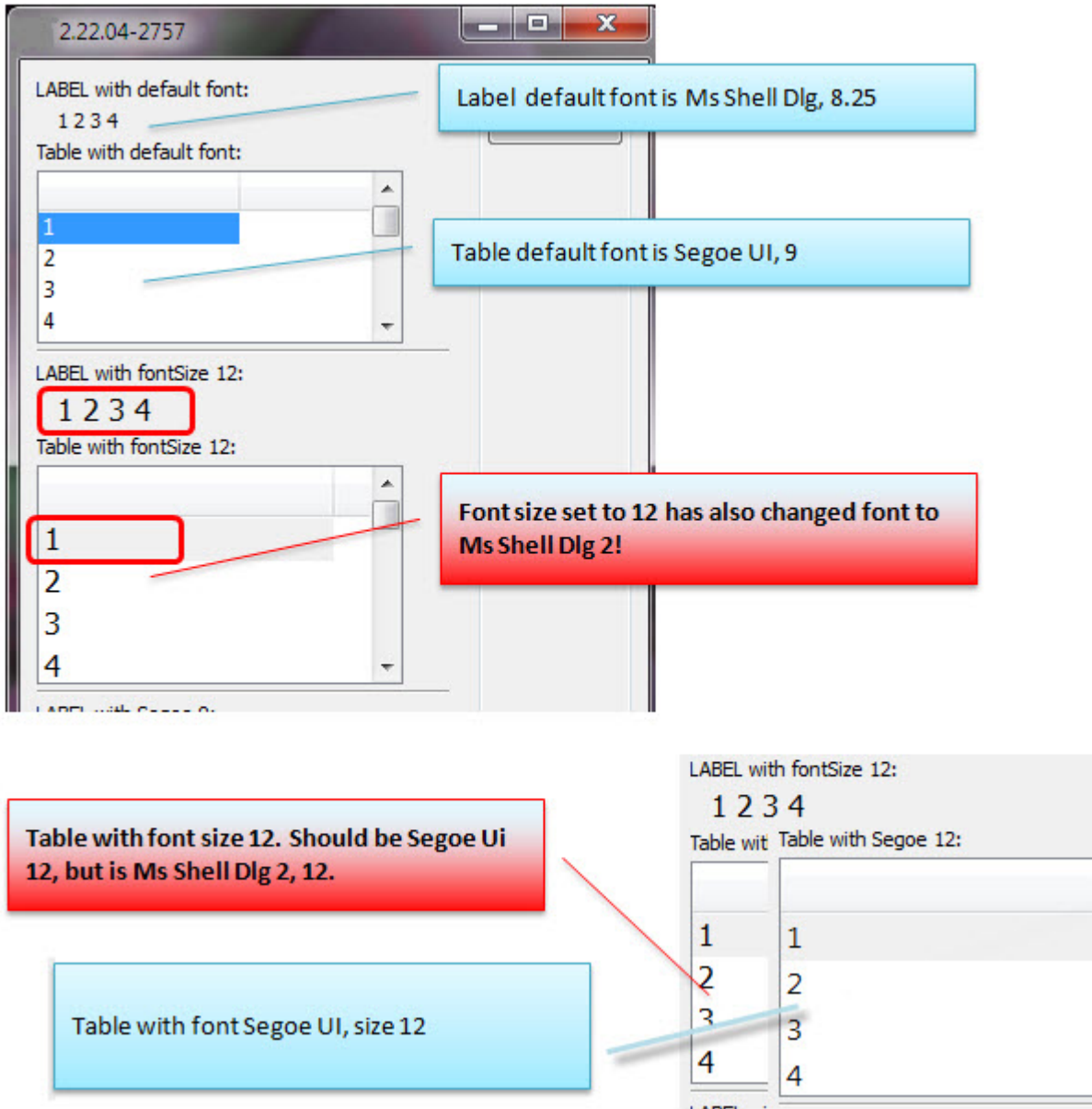


Figure 33: Font size examples

These two issues have been fixed in 2.30, but the side effect is that now the table row is taking the right font size, it's height may change depending on the system (from 17 to 18 pixels on Windows™ 7, for instance).

Genero Desktop Client 2.2x upgrade guide

This section describes differences you may encounter when upgrading to GDC 2.2x.

These pages will list differences you may encounter when upgrading to GDC 2.2x.

Genero Desktop Client 2.21 migration guide

This section describes differences you may encounter when upgrading to GDC 2.21.

Windows™: Installer uses MSI technology.

The installer and uninstaller have been rewritten using MSI technology. On Windows™ 7 or Vista, you will need elevated privileges to run the installer. The exe file we provide asks for elevated privileges, but this is not the case for:

- the .msi inside the .exe. If you manipulate the msi file directly (for silent install, for instance) you need to run it in an elevated command line prompt.
- the uninstaller. To uninstall GDC on Windows™ 7/Vista properly, use the **Setup** shortcut in the Start Menu and run it as Administrator.

ActiveX: embedded mode de-supported

Note: Earlier versions of GDC Active X proposed an *embedded* mode (the main window could be directly embedded in the html page instead of the monitor) . Unfortunately, we experience lots of focus conflict in this mode: in Genero, the focus is managed by the runtime system and not by the front-end. In embedded mode, system events can't be filtered as precisely as in *classic* mode, which leads to unacceptable focus issues. Therefore it has been decided to remove this too buggy feature. classic Active X mode is still available and supported.

Windows™: Default font size is 8.25

While implementing the feature request *support non integer font size*, we noticed that any font copy was using an integer font size value. So this means that the default font size was not 8.25 as the monitor shows, but only 8. This issue is now fixed ; while this is probably not noticeable in most of the cases, this may have an impact if you designed your form exactly for a given resolution.

Actions without names are now visible

Bug #14890 - Actions without names are not displayed - has been fixed in 2.21, to match TUI:

```
MAIN
  DEFINE cmd1 STRING
  DEFINE cmd2 STRING
  LET cmd1 = "cmd1"
  MENU "test"
    COMMAND cmd1
    COMMAND cmd2
    COMMAND "exit" EXIT MENU
  END MENU
END MAIN
```

GDC now behaves like TUI:

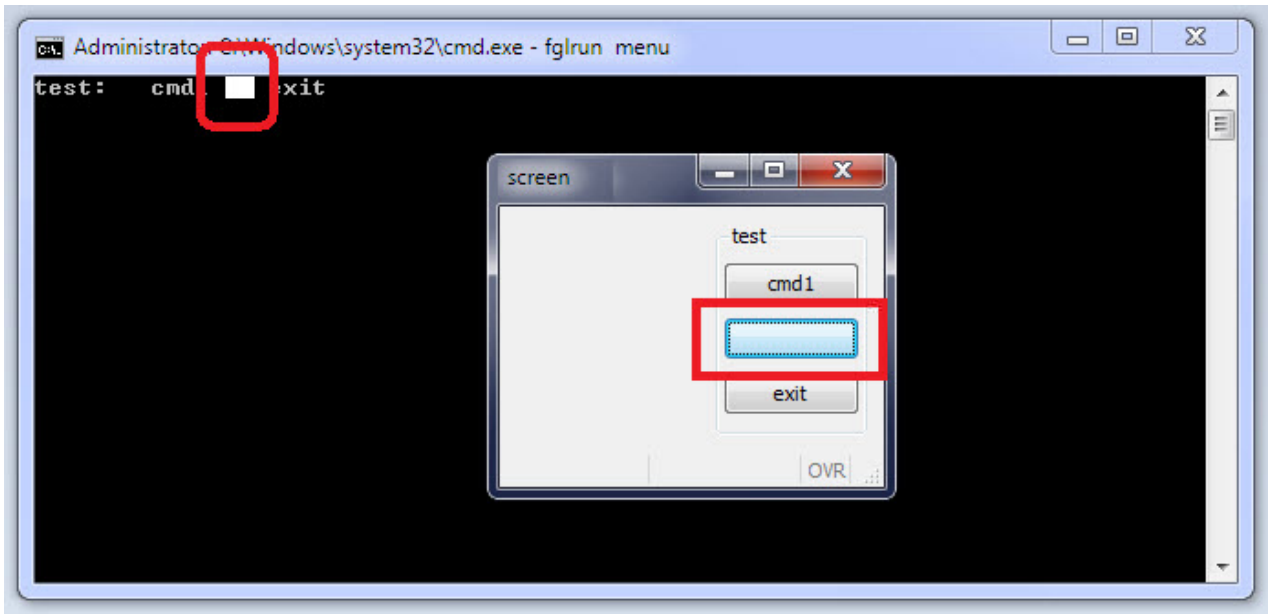


Figure 34: GDC behaving like TUI

This may have an impact on your programs if you are using actions without names, which was a classic pattern in early Genero Days:

```

MAIN
  DEFINE commands ARRAY[4] OF STRING
  DEFINE idx INT
  LET commands[1] = "cmd1"
  LET commands[2] = "cmd2"
  MENU "test"
  BEFORE MENU
    FOR idx = 1 TO 2
      SHOW OPTION commands[idx]
    END FOR
    FOR idx = 3 TO 4
      HIDE OPTION commands[idx]
    END FOR
  COMMAND commands[1]
  COMMAND commands[2]
  COMMAND commands[3]
  COMMAND commands[4]
  COMMAND "exit" EXIT MENU
  END MENU
END MAIN
  
```

This will result in an extra button with no label on your screen:

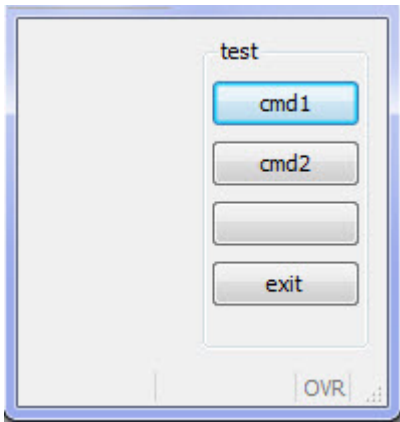


Figure 35: Extra button with no label

The reason is that

```
FOR idx = 3 TO 4
  HIDE OPTION commands[idx]
END FOR
```

is actually:

```
HIDE OPTION commands[3]
HIDE OPTION commands[4]
```

which is:

```
HIDE OPTION ""
HIDE OPTION ""
```

that is, hide twice the first action named "". The runtime system has no way to know you would like to hide a different option as they all have the same name. Therefore only the first action without a name is hidden, and all the other are visible.

To solve this issue and get the expected result, you have to give a different name to each of your actions:

```
MAIN
  DEFINE commands ARRAY[4] OF STRING
  DEFINE idx INT
  LET commands[1] = "cmd1"
  LET commands[2] = "cmd2"
  LET commands[3] = "cmd3" --give a name even if not used
  LET commands[4] = "cmd4" --give a name even if not used
```

Genero Desktop Client 2.20 migration guide

This section describes differences you may encounter when upgrading to GDC 2.20.

GDC is now compiled with Microsoft™ Windows™ Visual C++ 2008

As with any VC++ application, GDC needs VC Runtime files - basically DLLs Microsoft™ provides, the equivalent of the glibc on Linux®. The difference with GDC 2.1x is that Microsoft™ changed the way the runtime must be deployed. With VC 2003, it was possible to provide the DLLs with the application, but this was the cause of the [DLL Hell](#). Now The runtime system must be installed on the system directly, using a **VC Redist**. Our installer will always embed the corresponding VCRedist and install it if needed. But, if you were used to simply copying the GDC directory, you have to be sure that the correct redist is installed - if not, GDC will not be able to start.

Qt4: GUI changes

GDC is based on Qt, a multi platform C++ library. While 2.1x versions (and earlier) are based on Qt3, GDC 2.20 is the first Qt4 based version. Qt4 was a complete rewriting, and in lots of area Qt4 applications are different from Qt3. We've spent a lot of time to minimize the differences, but GDC 2.20 will not be 100% identical to GDC 2.11:

- Some changes are considered to be going in the right direction. For instance, [the default font](#) on Windows™ has changed, but this is to match Windows™ requirements.
- In some cases, it was technically not possible (or had too high a cost) to work around a change

The behavior of your applications should be unaffected. What may change is the look and feel, and the rendering.

Default font has changed

In Qt3, there was a bug which made Qt3 based applications not use the Microsoft™ recommended font ; this bug has been fixed in Qt4, and now GDC uses the font recommended by Microsoft™. More details:

- [Bug report for Qt3](#)
- [MSDN default font information](#): MS Shell Dlg 2 is the default font for Windows™ 2000 and after.

You can still set the default font in the GDC configuration panel, if you don't want GDC to use the correct system font.

Better adaptation to system style

The first version of Qt3 was released in 2001, before Windows™ XP. The theme mechanism was not designed to make use of all the new features introduced by Vista, OS X 10.4 or KDE4 (and even some items like Folder pages are very poorly supported on XP). Qt4 introduced a much better styling support, relying much more on the system theme - for instance on OSX (and some linux themes) the spacing between items is not fixed and varies depending on the item types. We've adapted these changes to Genero, but for some of them we let Qt apply the system defaults. Windows™: Items where the styling mechanism change has an impact:

- Top Menu (and StartMenu as menu) - GDC 2.11 had a Windows™ 2000 like style for menus. GDC 2.2x is using XP/Vista/Seven style, which is more modern but takes more space.
- Edit based fields now have a 3D effect, a rounded border, and the current fields shows a blue border.
- Comboboxes now have a grey gradient that becomes blue when the mouse moves over it. The side effect is that, in a Display Array, comboboxes on the current rows are not highlighted. (The system theme does not allow changing the gradient color).

Image attributes are handled differently

GDC 2.20 slightly modifies the handling of images attributes, as there were some inconsistencies in 2.11. Here are the major rules :

- It is important to differentiate the image and the image container (widget): when drawing your form, you're defining an image container. The image(s) that will be put in this container can be smaller or larger.
- SIZEPOLICY and WIDTH-HEIGHT attributes define the size of the container, not the size of the image.
- SIZEPOLICY is the priority attribute. It gives a directive for the size of the image container:
 - FIXED: exact size defined in the Form Specification File.
 - INITIAL: size is computed according to the content of the first display. Once done, the size is frozen. If the content is empty (no image), the container shrinks to its minimum size.
 - DYNAMIC: the width of the container grows and shrinks according to its content.
- WIDTH and HEIGHT attributes define the size of the container, but they are dependant on the SIZEPOLICY. It means the image container may grow or shrink even if WIDTH and HEIGHT are specified. If you really want to have a container with a fixed size, you have to use WIDTH and HEIGHT in combination with SIZEPOLICY=FIXED.

- AUTOSCALE allows the image to be scaled to the space of the image container. It's only useful if size of the image differs from the size of the container. It means there is no interest to use it with SIZEPOLICY=DYNAMIC, as the container always fits to the image size.
- STRETCH allows to make the container grow or shrink (and the image as well) when the parent container (for instance the window) is resized. This attribute doesn't conflict with the others.

Examples

```
-- image size: 80*80 pixels
WIDTH=150 PIXELS, HEIGHT=150 PIXELS, SIZEPOLICY=INITIAL, AUTOSCALE;
```

When it is first displayed, the container shrinks to 80*80 pixels in order to fit the image size. Its size is then frozen. AUTOSCALE is useful here only if another image of a different size is displayed afterwards in the same container.

```
-- image size: 80*80 pixels
WIDTH=150 PIXELS, HEIGHT=150 PIXELS, SIZEPOLICY=FIXED, AUTOSCALE;
```

The container has a fixed size of 150*150. The image is smaller, and AUTOSCALE allows scaling of the image to the whole space. When not using AUTOSCALE, you may also use the image style attribute *alignment* to define where the picture should be located when the container is bigger.

Report to printer differences

Introducing Qt4 and [Scribe](#), the text layouting system of Qt has been entirely rewritten. This has an impact on how GDC prints Report to printer: margins, spacings, font size have changed. You may have then to change your report fond to get similar result as before. Genero Report Writer is now the recommended way of producing reports.

W3C colors

While Qt3 is using X11 color definition, Qt4 is using W3C definition. Some colors have the same name in both definition, but not the same RGB value. This explains why using the term green for a color changed since 2.20.

Table 55: RGB values: X11 and W3C comparison

name	X11 RGB value	W3C RGB value
green	#00ff00	#008000
grey	#bebebe	#808080
maroon	#b03060	#800000
purple	#00ff00	#a020f0

Hitting German ß in an UPSHIFT field results in SS

If you hit the German ß in an UPSHIFT field, it will be immediately replaced by SS. It is something expected: SS is the official capitalization of ß. ß is nearly unique among the letters of the Latin alphabet in that it has no traditional upper case form. This is because it never occurs initially in German text, and traditional German printing never used all-caps.

Security

These topics cover security and the Genero Desktop Client.

- [Security levels](#) on page 96
- [GDC and SSH](#) on page 98
- [GDC and SSH simple setup](#) on page 100
- [Port Forwarding and Firewalls](#) on page 101
- [Implementing a Secure Server with GDC](#) on page 116
- [SSH Configuration Troubleshooting](#) on page 131
- [GDC and Windows XP Service Pack 2](#) on page 132
- [GDC and Windows Vista](#) on page 134

Security levels

The security level determines what verification occurs when a connection arrives on a listening port.

In previous versions, the Genero Desktop Client accepted all connections that arrived on the listening port, without any verification. In Genero 2.0, the security level was raised to level 2.

You may change the security level with the [command line](#) "-A" option, or through the [Security tab](#).

- [Security level 0](#) on page 96
- [Security level 1](#) on page 96
- [Security levels 2 and 3](#) on page 96
- [Security Connection Message](#) on page 97

Security level 0

Security level 0 is the least secure.

Command Line : `gdc -A 0`

Any connection started by the [runtime system](#) is authorized without any security message.

Security level 1

Security level 1 displays a warning.

Command Line : `gdc -A 1`

When the [runtime system](#) starts a connection, the GDC user is warned by a Security Connection Message.

Security levels 2 and 3

Security levels 2 and 3 use a key mechanism.

Command Line : `gdc -A 2` or `gdc -A 3`

Important: This only works when using a [direct connection shortcut](#) to start an application.

1. When started, the GDC generates a random key.
2. When a shortcut is started, @FGL (and @FGLxxx) sets this key into the `_FGLFEID` environment variable.
3. When the `fglrun` process is started, it reads the `_FGLFEID` environment variable and sends the key back to the GDC.
4. The GDC compares the internal key and the key sent back by the [runtime system](#). Only a connection with the identical key is accepted.

If a wrong key connection is encountered:

Security level 2: The Security Connection Message displays a message and asks whether to allow the wrong key connection to connect.

Security level 3: The connection is simply rejected.

Important: If the [runtime system](#) you are using does not handle this feature, you won't be able to run an application at this security level.

As of 2.20, the security mechanism also applied to GAS connections. Keys are transmitted by the Application Server to the Runtime System.

Security Connection Message

Depending on the security level set for the GDC, users may see a Security Connection Message with options to authorize or reject the connection.

Figure 36: Security Connection Message

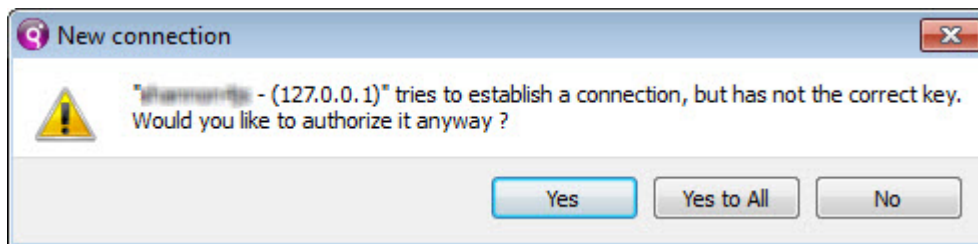


Table 56: Security Message Options

Action	Description
Yes	The GDC accepts this connection and only this connection. The connection information is stored in memory for the duration of the connection. Any additional connection causes the message to be displayed again.
Yes to All	The GDC accepts this connection and any other connection from the same host. This setting is saved to <code>\$AppDataDir/hosts.xml</code> (see GDC configuration file directories on page 17 for more information) when the GDC closes and is reapplied when the GDC is restarted. . You can modify the <code>hosts.xml</code> file if needed, or remove it to clear the authorized list.
No	The GDC rejects this connection. As a result, the application will not run on the front end .

Connection Message and Security Levels

The Security Connection Message is displayed depending on the set GDC security level.

Table 57: Connection message by security level

Security Level	Connection Message Displayed
0	Message is never displayed.
1	GDC checks if the IP of the host exists in the <code>\$AppDataDir/hosts.xml</code> . If not, the message is displayed.

Security Level	Connection Message Displayed
2	If the FEID (randomly generated key) is a match, the connection is accepted without displaying the message. Otherwise, the GDC checks if the IP of the host exists in the <code>\$AppDataDir/hosts.xml</code> file. If not, the message is displayed.
3	If the FEID (randomly generated key) is a match, the connection is accepted without displaying the message. Otherwise, the connection is refused outright and the message is not displayed.

GDC and SSH

This section provides an overview and the prerequisites of using GDC with SSH.

- [GDC and SSH overview](#) on page 98
- [GDC and SSH prerequisites](#) on page 99

GDC and SSH overview

GDC with SSH provides security and port forwarding.

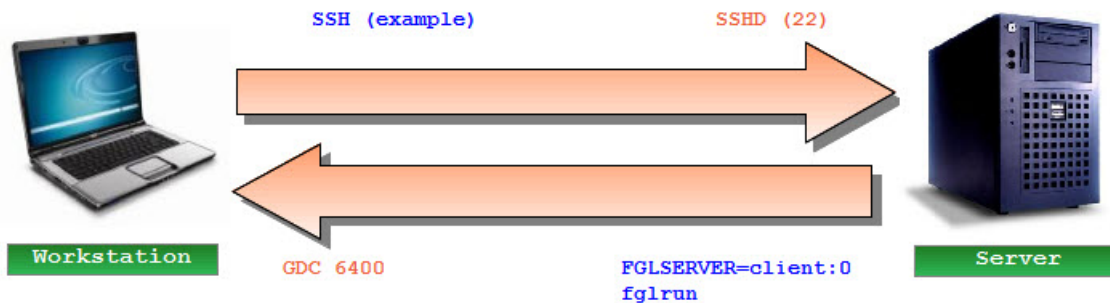


Figure 37: SSH communication flow between workstation and server

SSH stands for "Secure SHell". It was designed to replace the 'r' commands like rlogin and rsh, because they offer no real security. SSH encrypts all data end-to-end, offers data compression, and prevents snooping and connection hijacking. One additional feature it offers is *port forwarding*.

Port forwarding allows an application on one computer to connect to a local port and have its data tunneled through an SSH session to the other computer. This does not require you to open any ports on your [firewall router](#), other than the port you already have open for SSH. This is a distinct advantage. If you have [firewalls](#), this is advantageous as Genero needs to establish a connection from the client to the server to start the user application, and another connection originating from the user application to the client to display the graphical user interface. When Genero establishes a connection from the server to the client, it can use the existing connection to tunnel the graphical connection.

Important: Any environment that uses [firewalls](#) or connections over the Internet should use SSH, SSH2 or HTTPS for those connections. Furthermore, any production environment on an intranet or internet should use a secured layer. You should never send unencrypted data such as account numbers, social security numbers, and passwords through the Internet. Some companies might even consider using secure shell connections for internal use when handling sensitive data such as accounting and payroll information. At any point along the way, someone could be monitoring the data, for network diagnostics or possibly with malicious intent. Whatever the reason, encryption is simple and offers peace of mind.

SSH is comprised of two main components, the server component "sshd" and the client component "ssh". Genero provides its own client component (built-in).

GDC and SSH prerequisites

This topic covers prerequisites and SSH connection options.

Things you should know about your system

In order to determine how to proceed, you will need the following information about your environment:

- Is there a server-side [firewall](#) between the server and the client?
- Is there a client-side [firewall](#) between the server and the client?
- Is encryption needed for all your data?
- Are you using a [VPN \(Virtual Private Network\)](#) or [NAT \(Network Address Translation\)](#)?
- Will you need protection from inactive sessions timing out?
- Do you have more than one server to access from outside the firewall?
- Do you have more than one client accessing servers outside the firewall?

We recommend reading about SSH and how to configure it. We will not cover this topic in this document.

How do I make sure data is encrypted?

To ensure that your data is encrypted, select SSH or SSH2. Both offer data compression and port forwarding; the difference is SSH2 has different implementation code and a more advanced encryption algorithm than SSH.

If you are using the shortcut buttons in the Genero Desktop Client, two connections are established between the client and the server. The first connection is established from the client to the server, in order to log in and start the application. The second connection is made from the server's application to the client, in order to display the graphical data.

Use the [Table 58: Data encryption selection matrix](#) on page 99 to determine which settings you will need.

Table 58: Data encryption selection matrix

Type of connection	Command encrypted	GUI encrypted
telnet	NO	NO
ssh	X	NO
ssh port forwarding	X	X
ssh2	X	NO
ssh2 port forwarding	X	X

What connection method should I use?

Knowledge of your configuration will be necessary to make Genero work properly, as discussed at the start of this topic. Use [Table 59: Connection method support matrix](#) on page 100 to determine which connection methods will support what you are trying to do. Currently the SSH or SSH2 with Port Forwarding provides the most flexible connectivity.

[Table 59: Connection method support matrix](#) on page 100 uses the following legend:

- 1 - Requires configuring the server side [firewall](#) router to open or forward the port used by sshd.
- 2 - Requires configuring the client side [firewall](#) router to open or forward the port(s) used by the GDC.
- 3 - May require changes to [firewall](#) connection timers if [firewalls](#) are involved.
- X - Indicates full functionality with no changes.
- NO - Not supported

Table 59: Connection method support matrix

	telnet	SSH	SSH + Port Forwarding	SSH2	SSH2 + Port Forwarding
Firewall or NAT on Server Side	1	1	1	1	1
Firewall or NAT on Client Side	2	2	X	2	X
Firewall or NAT on Both Sides	1,2	1,2	1	1,2	1
Private Network	X	X	X	X	X
VPN (Same as Private Network)	X	X	X	X	X
Encryption of all Data	NO	NO	X	NO	X
Password/login Encrypted	NO	X	X	X	X
Keep Alive	NO	NO	X, 3	NO	X, 3

GDC and SSH simple setup

The simple setup assumes that you are on a corporate LAN with no firewalls, and allows for all connection methods.

All methods of connections are possible here (telnet, ssh, ssh2, with/without port forwarding) without any special set up. Using SSH or SSH2 will work fine and will offer encryption. The GUI connection will be made on the default port 6400. FGLSERVER will be set to '<client IP> :0' and it will expect to be able to access that IP and port directly.

Simple connection no Port Forwarding

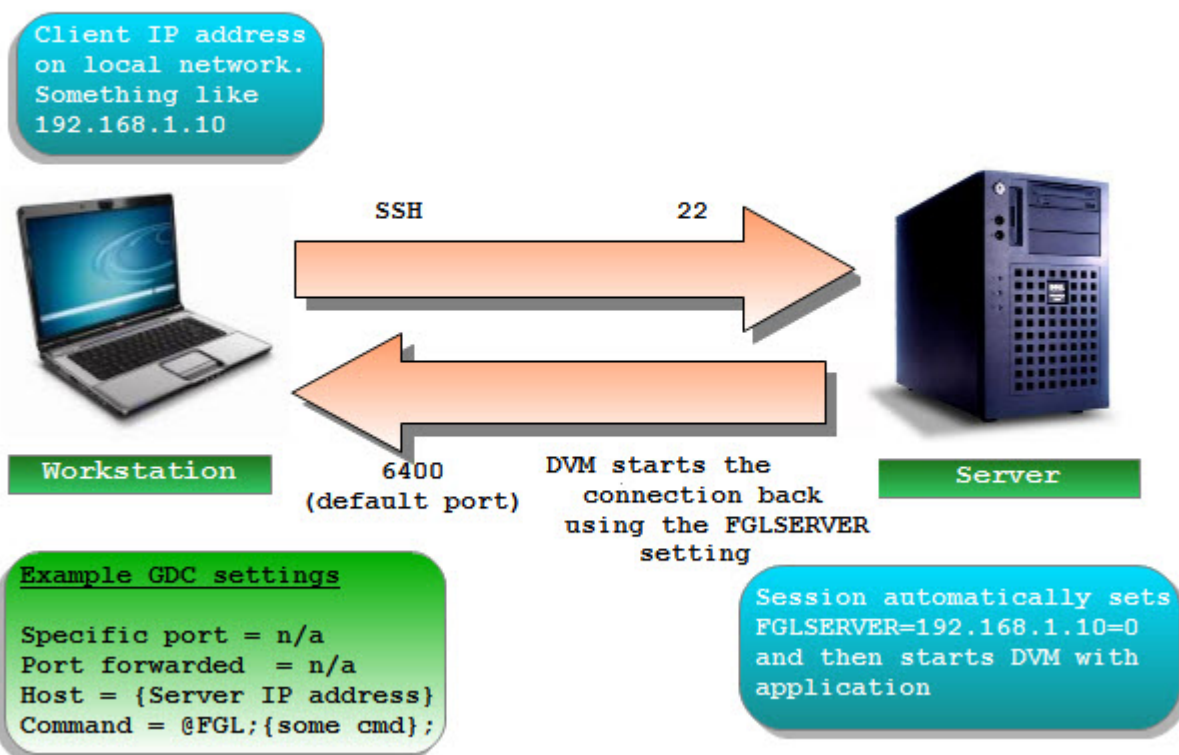


Figure 38: Simple connection no Port Forwarding

If you don't want any encryption or compression, select telnet as your method of connection.

What if you want to connect to a port other than 6400 for the GUI? Specify the option "-p <port>" on the [command line](#) for GDC, and GDC will listen on that port for the GUI connections. The FGLSERVER will have its information adjusted accordingly. For example, execute "gdc -p 7400". When you look at the value of FGLSERVER, it will contain "<client IP> :1000". It would contain "<client IP> :0" if the default of port 6400 was used (the number displayed after the colon is the port number that you specified minus 6400, the default number.)

If you do port forwarding while using "-p 7400" on the GDC command line, the offset number after the colon will still be your Port Forward value minus 6400. This is because fgldrun doesn't care what port you are listening on the client side, only what port needs to be connected on the server side. The tunnel takes care of connecting to the correct port on the client side. Using @FGL keeps everything automatic. If you have a need for multiple GDC's running at the same time, see [Port Forwarding and Firewalls](#).

Port Forwarding and Firewalls

This section covers configuration of Port Forwarding with client or server-side firewalls.

- [Port forwarding](#) on page 102
- [Port forwarding and the client-side firewall](#) on page 107
- [Port forwarding and the server-side firewall](#) on page 110

Port forwarding

Port Forwarding is used in situations where you want all data encrypted, no session timeouts, or simple firewall setup.

Note: Genero Desktop Client 3.00 supports Internet Protocol version 6 (IPv6), in addition to Internet Protocol Version 4 (IPv4), when using port forwarding through an ssh tunnel.

Simple connection with Port Forwarding

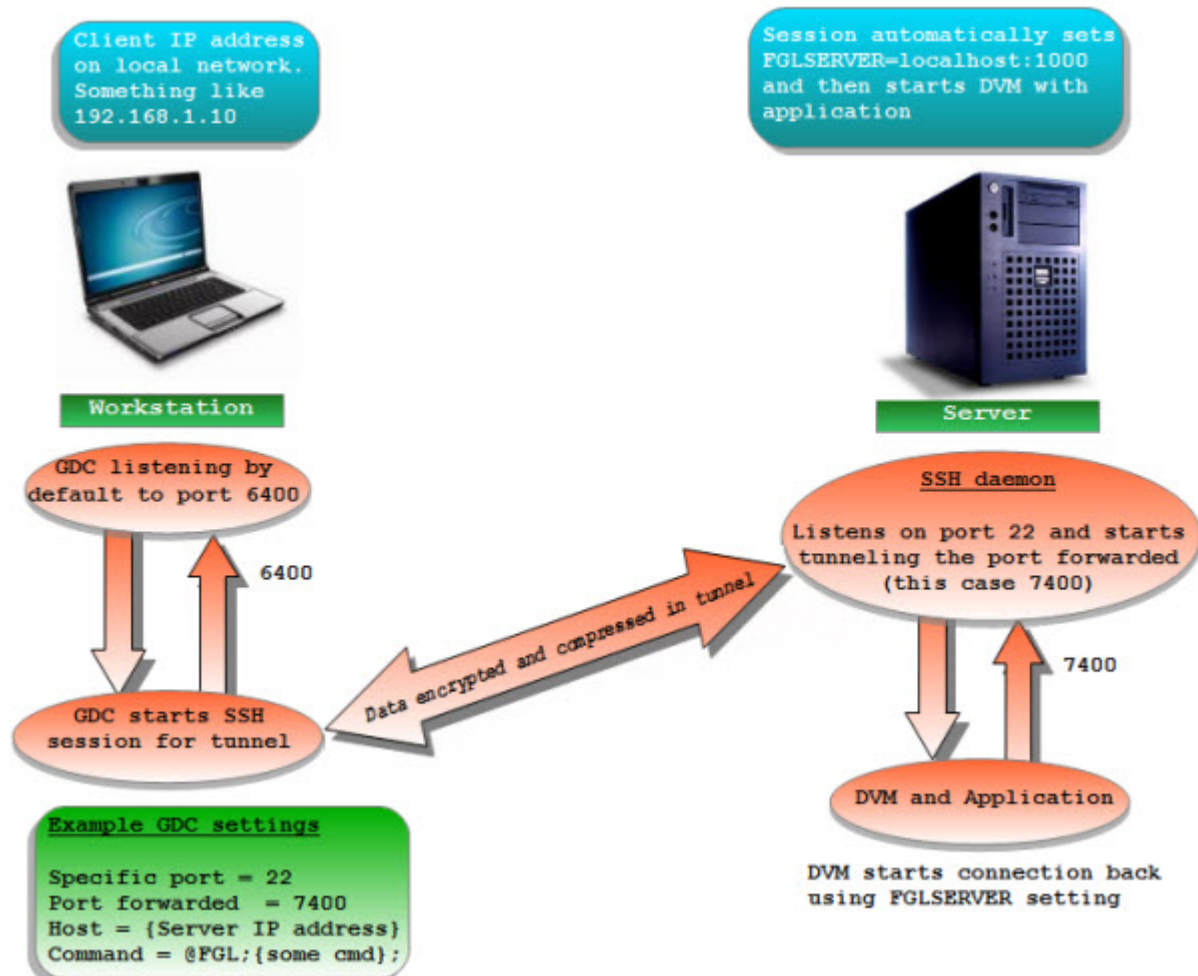


Figure 39: Simple connection with Port Forwarding

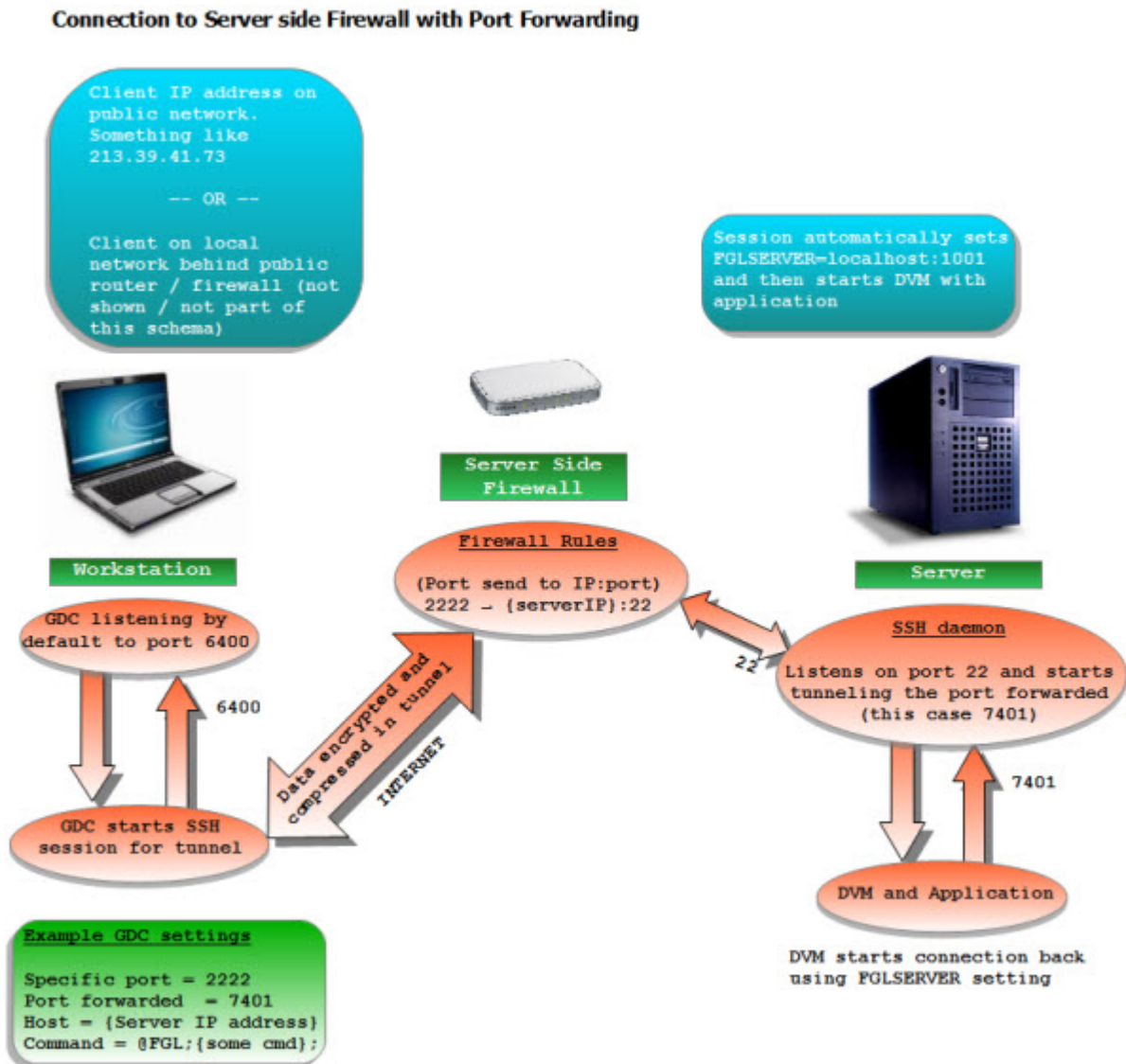


Figure 40: Connection to Server side Firewall with Port Forwarding

Figure 39: Simple connection with Port Forwarding on page 102 shows a simple configuration that does not involve a firewall. Sshd, the portion running on the server, will accept a connection from the GDC (client) and start your application. It will also set up a listener for a port that the application will connect to for the GUI. This port is then tunneled through the existing connection to the client, where the client will display the application. Note that both sides still use ports to accomplish this.

You must have ssh installed and set up on the server. If you are expecting to access your Genero application from somewhere on the Internet, you will most likely have a firewall router and must open a port on your router to allow connections to the sshd. See Figure 40: Connection to Server side Firewall with Port Forwarding on page 103 for an illustration of this.

Sshd is by default listening on port 22. You can set a port on the firewall to forward to sshd. Whatever port number you use must be set in the GDC using the "Specific Port" field:

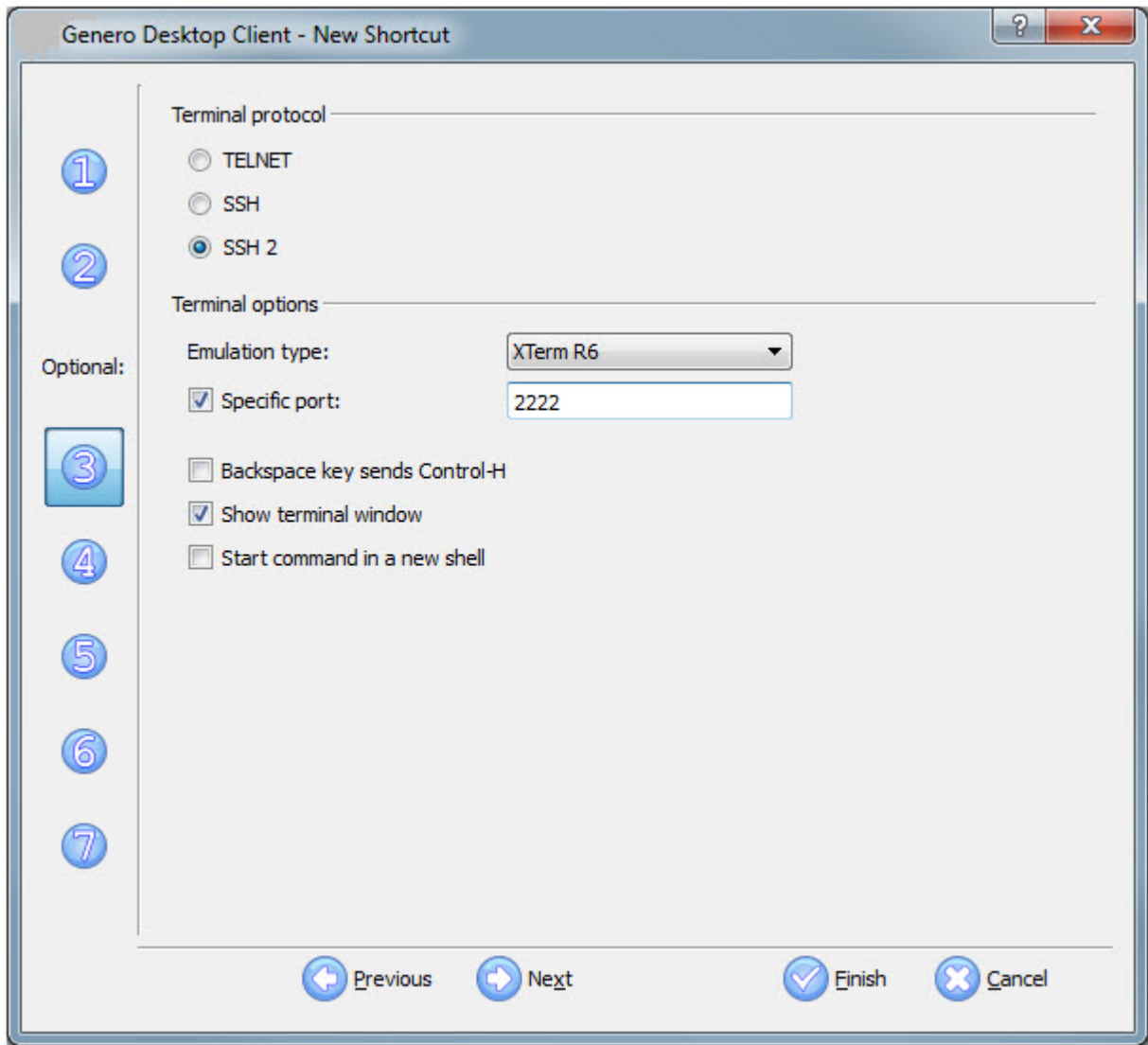


Figure 41: Specify specific port number 2222

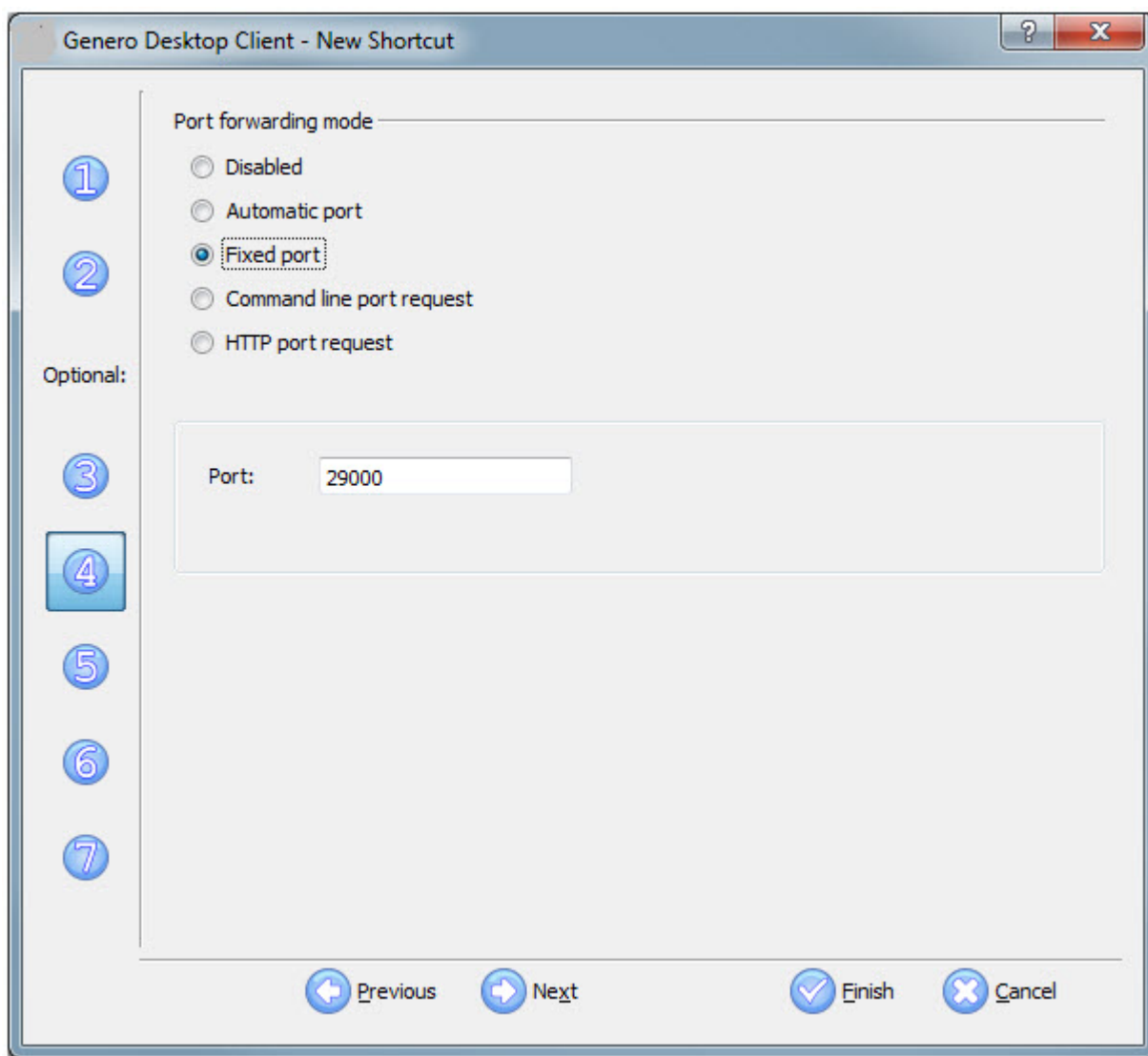


Figure 42: Specify fixed port number 29000

In [Figure 40: Connection to Server side Firewall with Port Forwarding](#) on page 103 we have set our firewall router to forward port 2222 to our server sshd. There is no reason you couldn't just use port 22 and pass it straight through to your server. If you have more than one server you need to access from outside your firewall, you must use different port numbers and map each server with a different port number. Most routers will allow the destination port to be different from the origination port. For example, a rule could be entered into your firewall router to forward port 2222 to a server on port 22; set another rule to direct 2223 to a different server on port 22, and so on. More details on this are in the [Firewall Server Side section](#).

In [Figure 42: Specify fixed port number 29000](#) on page 105 we have also set Port Forwarding to 29000. This will cause the sshd running on the server to listen to port 29000 for connections from the application. The FGLSERVER environment variable will be set to 'localhost:22600'. It is localhost because it will be tunneled and sshd is running on the same machine. The 22600 is an offset for the port. To clarify, Genero GDC listens on 6400 by default and any number after the colon in FGLSERVER is added to this number. So $22600 + 6400$ works out to be the port we specified on the client side configuration, 29000.

To use *Automatic Port Forwarding*, you can specify a command line that will execute on the server and return a free port number. As this application is really depending on the system where the Runtime System is installed, we can't provide a version for each system. This program must be used in combination with the GDC connection strings system.

Another way to achieve automatic port forwarding is to have a service running on an HTTP server. This can be a CGI. The program must return lines containing information for the coming SSH connection. One line is always like the following: <attribute name>=<attribute value> For the moment, the attributes managed are "host" and "port", which can indicate the host IP to connect to and the port the sshd will listen to on the server side. By default, the host IP is the same as the HTTP server machine.

Click "Next" for the configuration.

The IP address is that of the server machine unless the firewall on the server side is doing NAT (Network Address Translation). If it is doing NAT, the IP address should be set to the address of the firewall router. Put @FGL on the line labeled "Command Line", so Genero can set the FGLSERVER variable for you when it logs into the server. FGLSERVER will have the port number corresponding to the "Port Forwarding" value you put in the previous screen. Several commands can be placed on the command line and executed in succession. In UNIX™ you use a semi-colon (;) and in Windows™ you use two ampersands (&&) to separate the commands.

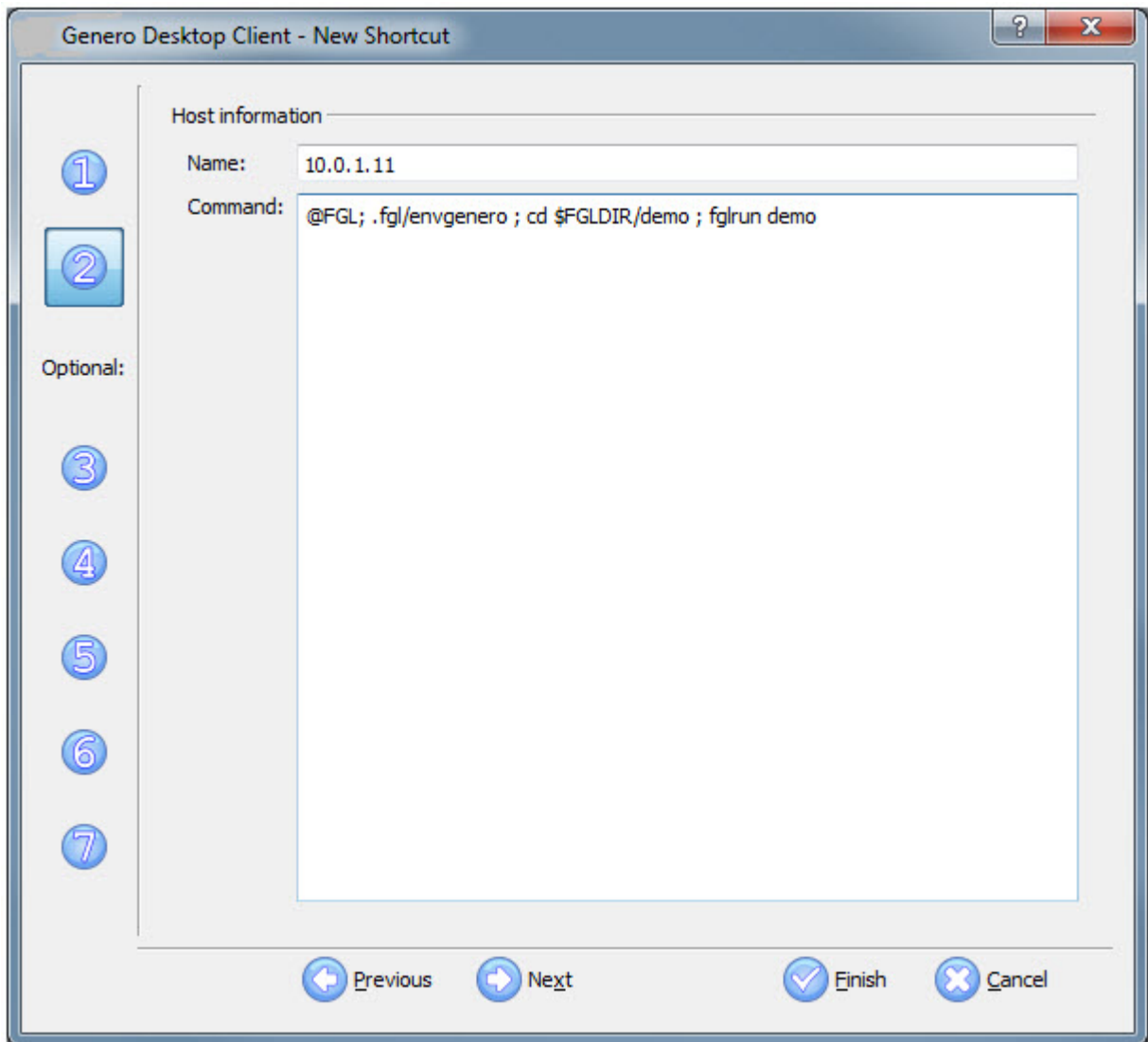


Figure 43: @FGL command example

Port forwarding and the client-side firewall

This section details how to configure port forwarding with a client-side firewall.

Connection from Client side Firewall with Port Forwarding

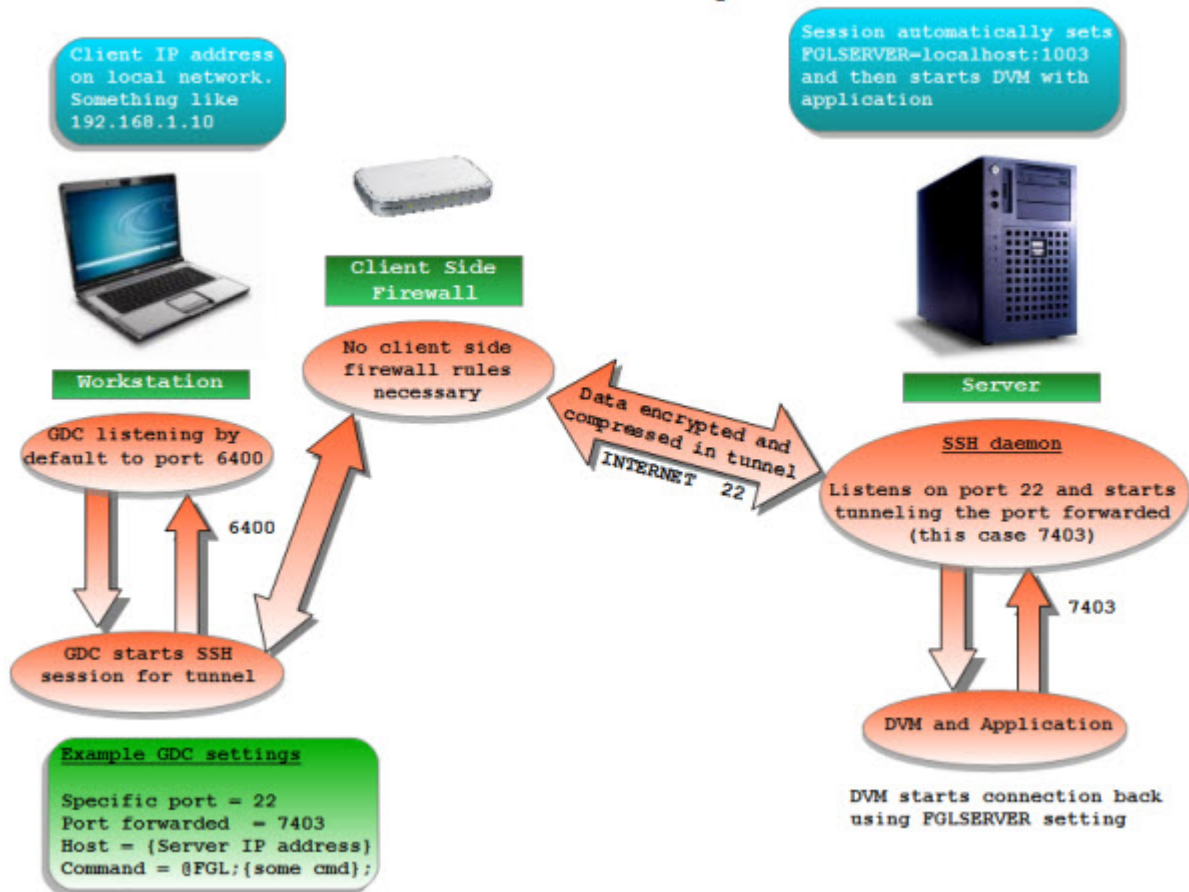


Figure 44: Connection from client side firewall with port forwarding

If you have a client side firewall, you cannot connect directly to your clients from outside the firewall. There are two solutions to this problem:

- First, you can set up port forwarding while using SSH or SSH2 (See [Figure 44: Connection from client side firewall with port forwarding](#) on page 107). This is by far the easiest and most secure method to connect without the help of a VPN.
- The second method requires adding rules to the router to allow connections (See [Figure 45: Connection from Client side Firewall](#) on page 108). The set up of the router will be covered here; [port forwarding](#) is covered in a separate section.

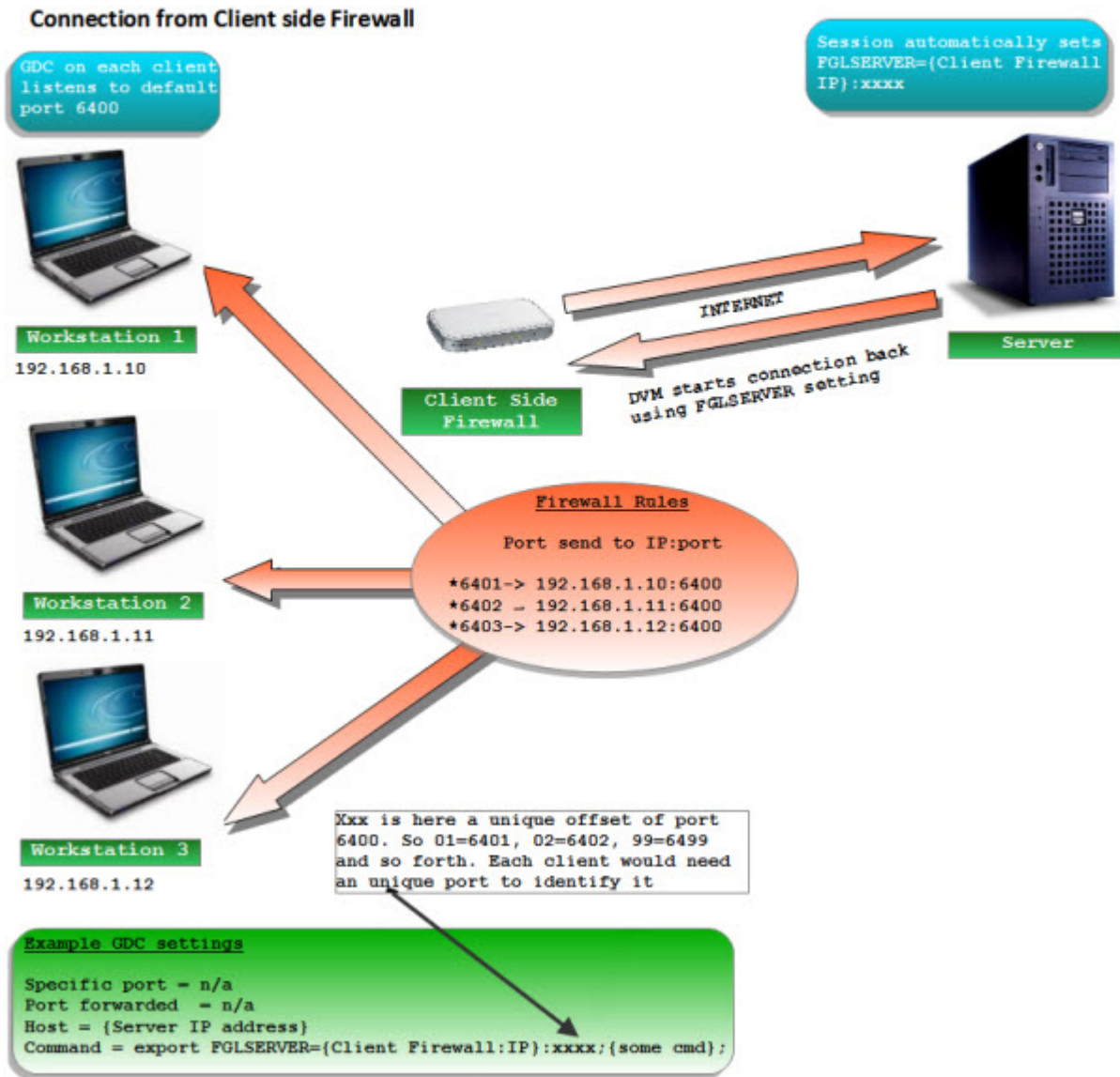


Figure 45: Connection from Client side Firewall

The router will need rules added to take a connection coming in on a specific port and direct it to one of your clients. The way Genero is normally configured, all clients would use port 6400. If you only have one client, you can add a rule to the router to forward 6400 to the client on port 6400. If you have more than one client, you will need to allocate other ports on the router to forward to the other clients.

Note: In the examples shown, the internal addresses are not public IP addresses. If you have public IP addresses on each client, you can open port 6400 for each of the clients.

Example rule:

```
Incoming 6400 -> 192.168.1.10:6400
```

If you have more than one client, you can map them as follows:

```
Incoming 6401 -> 192.168.1.10:6400  
Incoming 6402 -> 192.168.1.11:6400  
Incoming 6403 -> 192.168.1.12:6400
```

Another option if your firewall won't allow you to change the destination port number:

```
Incoming 6401 -> 192.168.1.10:6401  
Incoming 6402 -> 192.168.1.11:6402  
Incoming 6403 -> 192.168.1.12:6403
```

This last example requires that you start the GDC with the `-p` option, causing it to listen on a different port from the default port.

```
>gdc -p 6401  
>gdc -p 6402
```

If you are setting up multiple clients in this manner, you may want to avoid starting the first client on 6400; any misconfigured new clients will pop up on that user's console unexpectedly.

On the command line of the GDC shortcut setup, assign `FGLSERVER` to be the IP of the firewall router with the corresponding port of the router. This must be hard-coded, since there is no way for the client computer or Genero to know how the connection is established.

For example, if the client firewall router's IP address to the Internet is 213.39.41.73, and port 10000 is mapped to the client 192.168.0.53 port 6400, then the entry in the router would be:

```
Incoming 213.39.41.73:10000 -> 192.168.0.53:6400
```

The command line in the GDC would look like:

```
FGLSERVER=213.39.41.73:36000; fgldrun demo
```

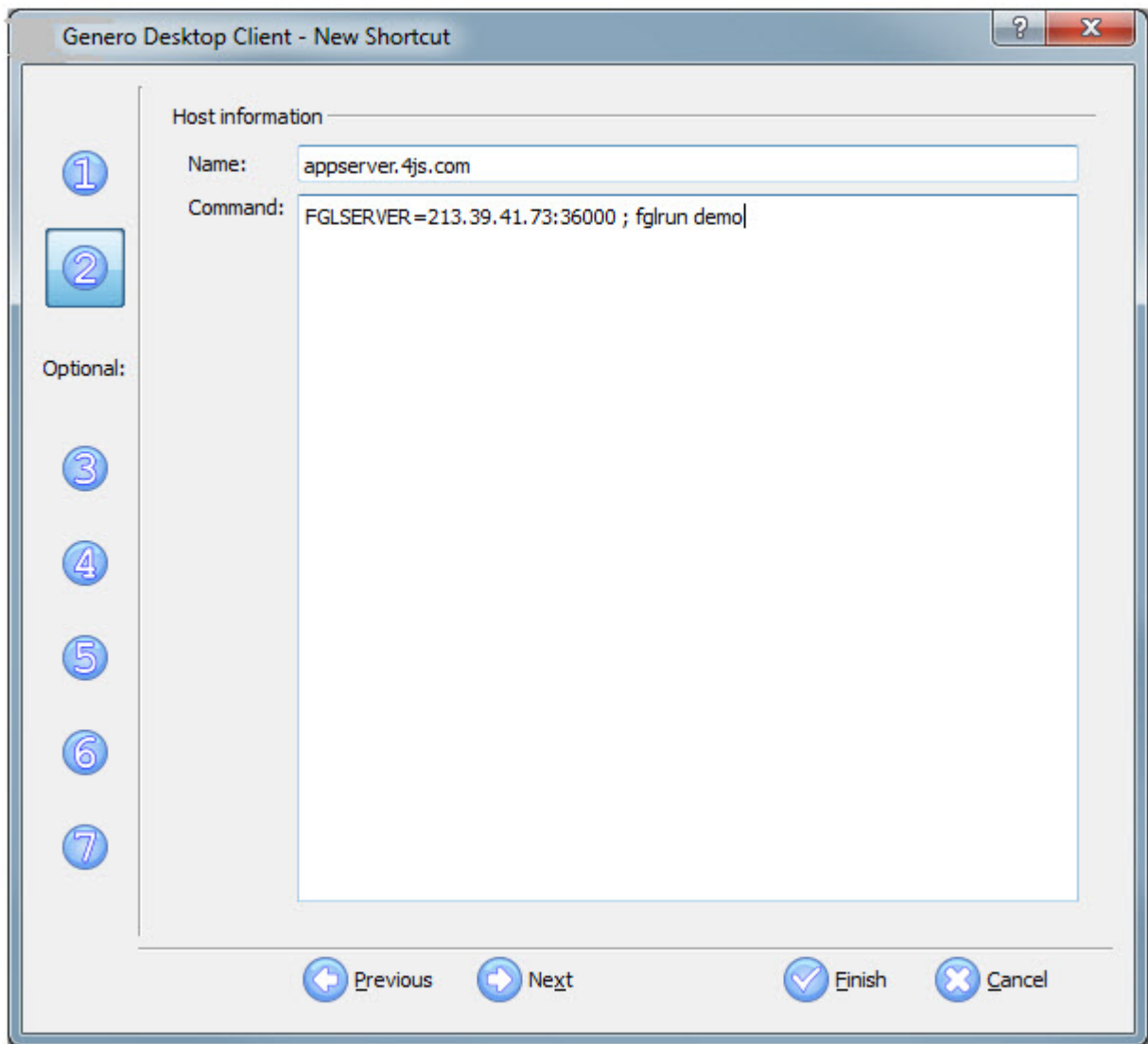


Figure 46: Entering the proper command for a GDC shortcut

The FGLSERVER variable is normally set using @FGL, but that would set FGLSERVER to the IP of the local client machine and the port specified when the GDC was started with -p. If the IP addresses used behind the firewall are public, this would be OK. If the addresses are not public, however, we must use the IP address of the router, and let the router translate and forward it. If the router is translating the port, then we must use the port that the router is expecting.

In our example the port that the router is looking for is 10000. The FGLSERVER port value must be set to 10000 minus 6400, resulting in 3600. This is because FGLSERVER=<ip> :0 tells Genero to connect on port 6400. The number after the colon is added to 6400.

Port forwarding and the server-side firewall

This section details how to configure port forwarding with a server-side firewall.

Having a server side firewall is the typical configuration on many systems. There is only one method for doing this, whether you use telnet or ssh: map a port to be forwarded to the server in the firewall router. It is not advised that you use telnet from the Internet for security reasons; that is usually why you have a firewall.

Decide which method of connectivity will be allowed, and determine what port you will use to forward to this service. If there is only one server involved, you can use port 22 for ssh or 23 for telnet and forward them straight through to the server. But if there are several servers involved and they do not have public IP addresses, you will need to pick different ports on the firewall router and let the router forward those ports to the different internal servers.

See [Figure 47: Connection to server side firewall](#) on page 111 for an example of how to do this for a telnet connection. Notice that the returning GUI path doesn't require any special handling unless there is a client side firewall. For details on this see the [Client Firewall section](#).

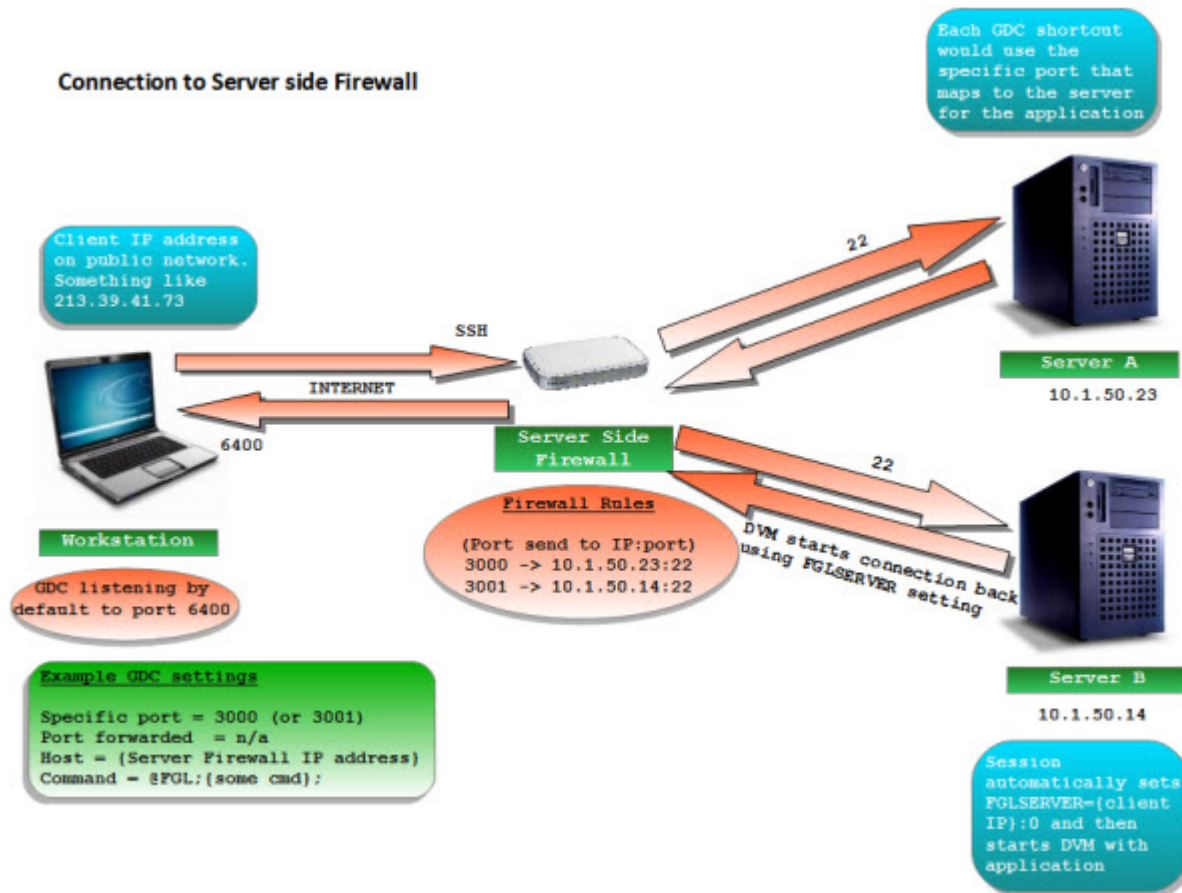


Figure 47: Connection to server side firewall

See [Figure 48: Connection to Server side firewall with port forwarding](#) on page 112 for an example of how to do this using ssh with port forwarding.

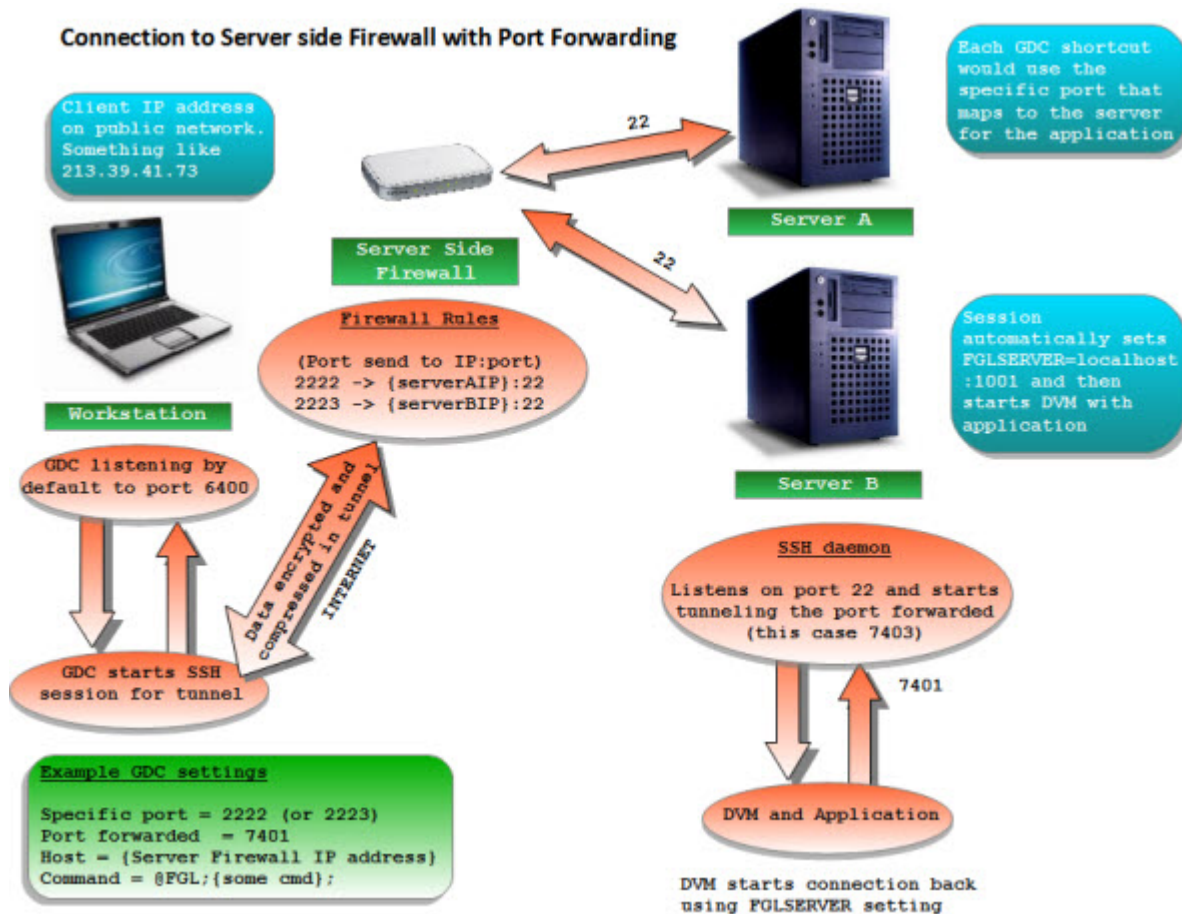


Figure 48: Connection to Server side firewall with port forwarding

The client GDC would connect to the server firewall router on port 3000 to access server 1, and port 3001 for server 2. We chose these ports arbitrarily; almost any port could be used. Numbers below 1024 are reserved for well-known services, so choose numbers above 1024.

Using port forwarding will work without modification because the GUI interface is tunneled through the initial connection, and the port it tells the server application to use is a local port to the server. Of course, the same methods as above must be used if there is more than one server.

Using telnet or non-port forwarded ssh will work also, because connections for the GUI originating from behind the server firewall will be allowed out without special mapping. If there is a client side firewall as well, see [client side firewall configuration](#).

Example:

We have two servers that will be accessed via clients somewhere on the Internet. They will use ssh2 with port forwarding to simplify client set up and keep things secure. The firewall on the server side has an IP address of 192.168.50.2 (only valid for this example). We have mapped the two servers:

```
213.39.41.73:3000 -> 10.1.50.23:22 213.39.41.73:3001 -> 10.1.50.14:22
```

The GDC client will need to be configured as well:

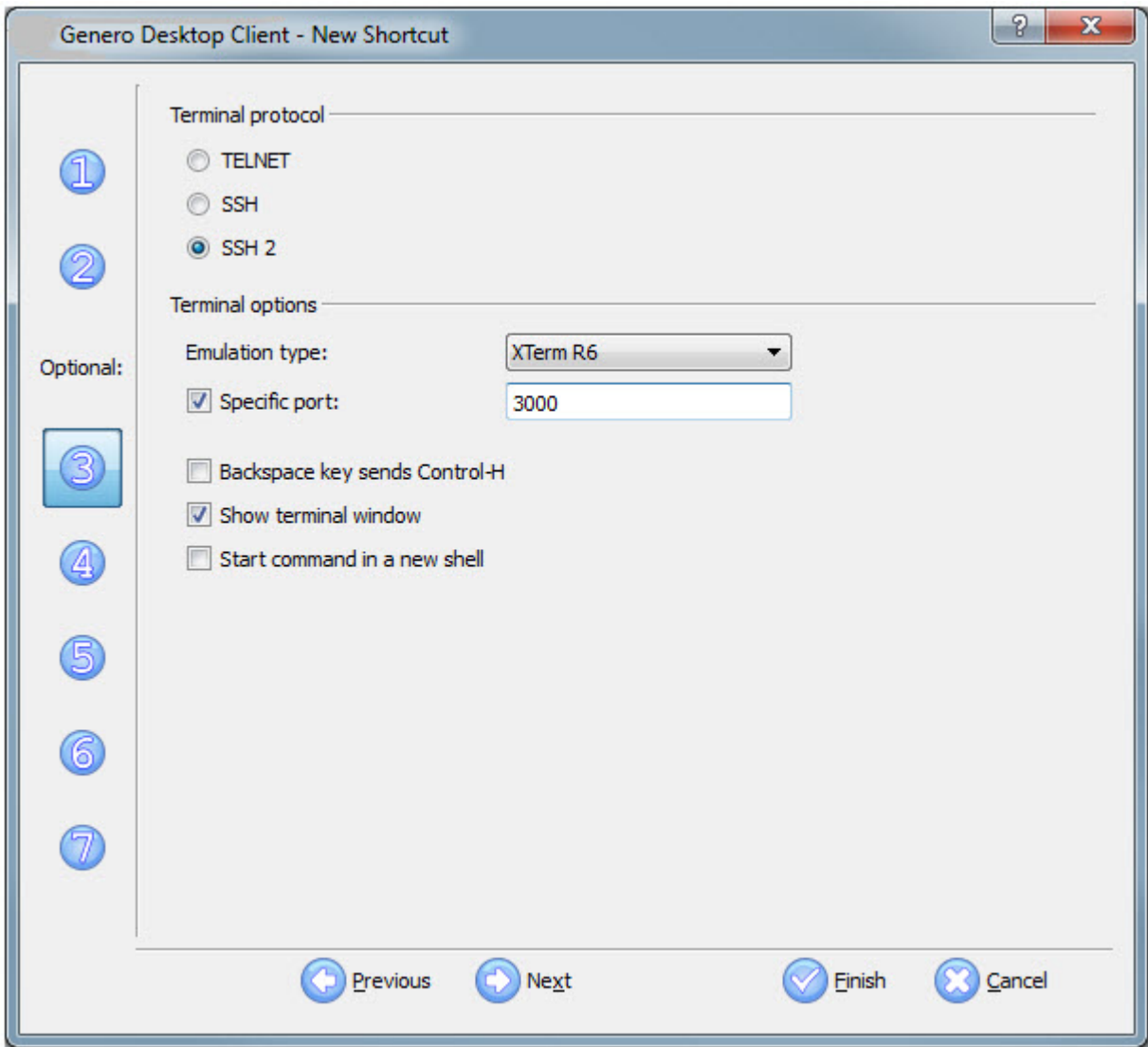


Figure 49: Showing configuration for access to Server 1

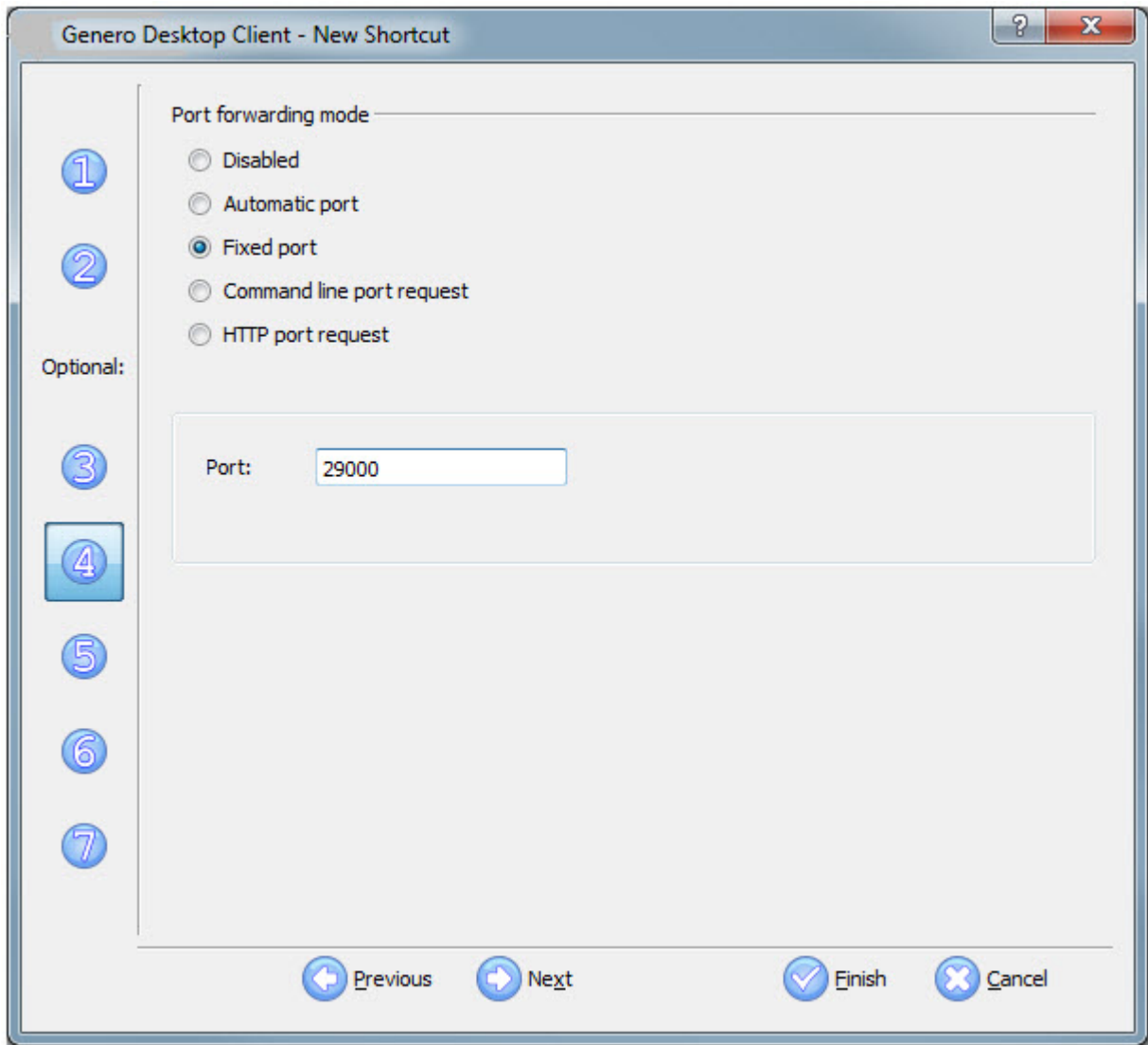


Figure 50: Showing configuration for access to Server 1

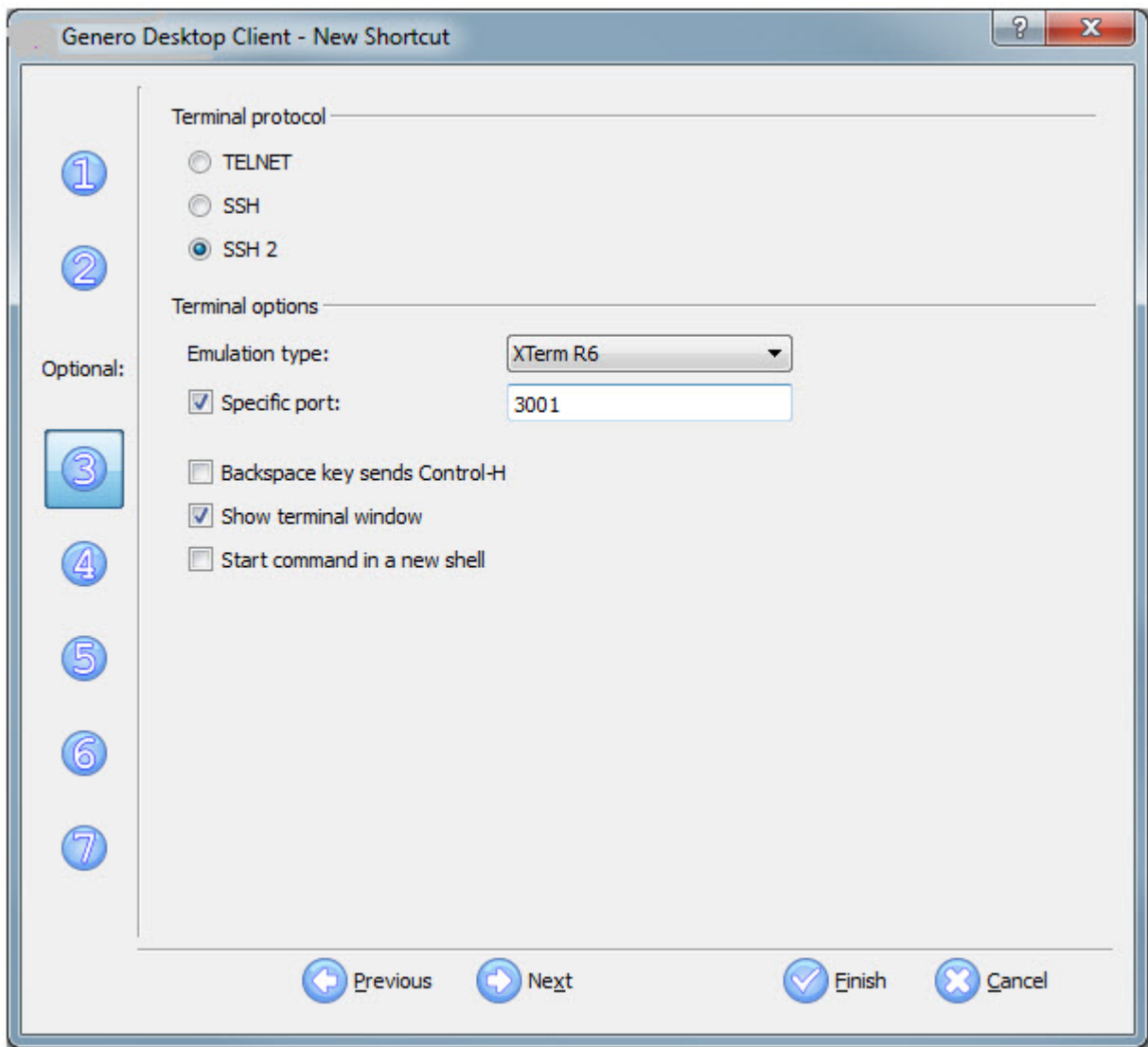


Figure 51: Showing configuration for access to Server 2

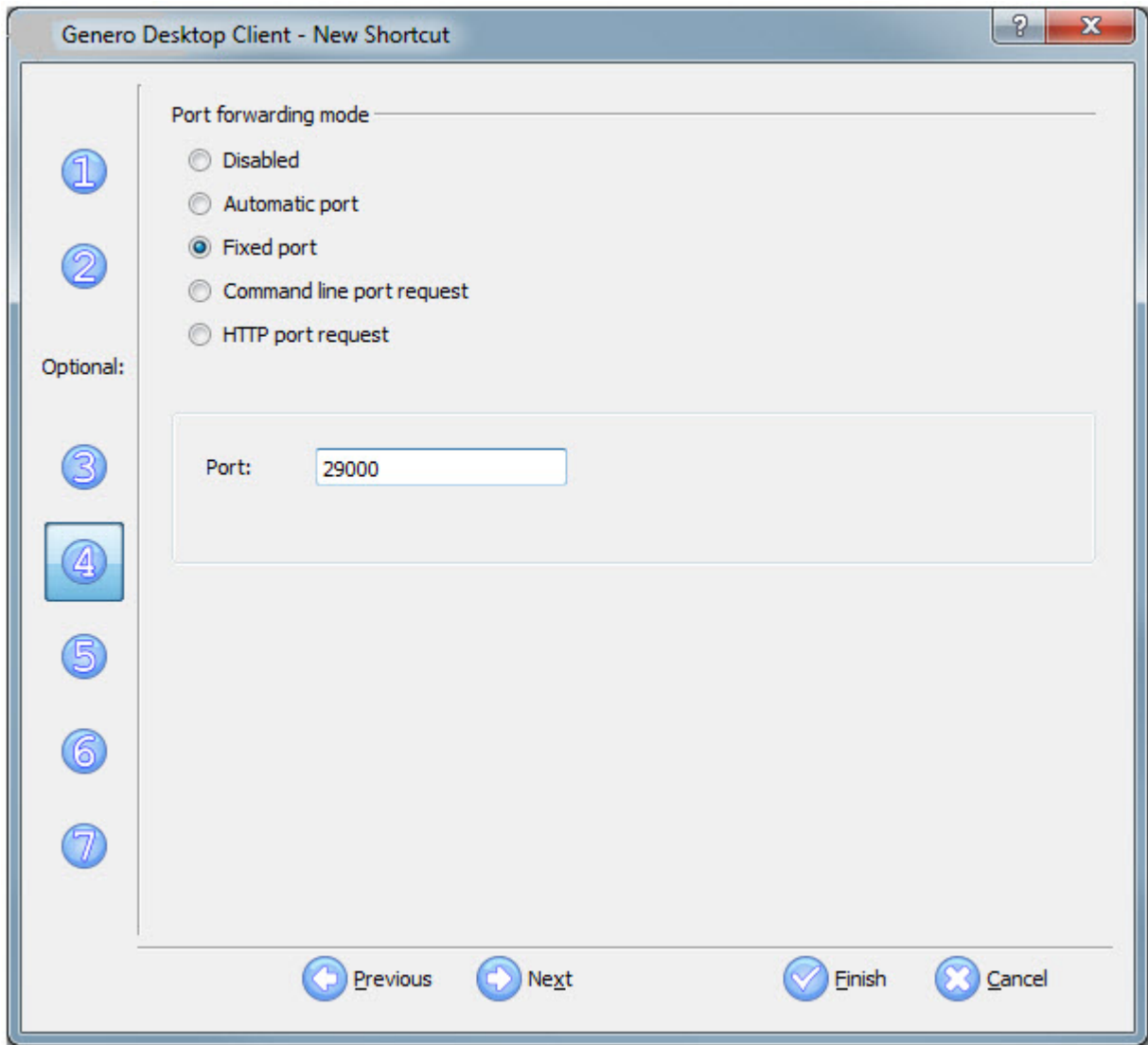


Figure 52: Showing configuration for access to Server 2

Figure 49: Showing configuration for access to Server 1 on page 113, Figure 50: Showing configuration for access to Server 1 on page 114, Figure 51: Showing configuration for access to Server 2 on page 115 and Figure 52: Showing configuration for access to Server 2 on page 116 show how to access each server by specifying the appropriate port for each, one with 3000, the other 3001. This will allow the firewall router on the server side to direct each to the appropriate server. The IP address used would be the IP of the router.

Keep in mind that if you have two users accessing the same server, you must manually select a different port forward number to keep them unique. See [Possible Configuration Problems](#).

Implementing a Secure Server with GDC

Implement a secure server by denying users access to the command line or shell.

In an enterprise deployment, it is typical for the Genero Desktop Client to be configured to launch in the default user mode with all application shortcuts pre-defined.

When the "-a" or "--admin" option is specified, however, the Genero Desktop Client launches in admin mode, and the user is able to modify existing shortcuts or create new shortcuts of their own. Therefore,

when in admin mode, a Genero Desktop Client user with sufficient knowledge can modify the string passed to the server (UNIX™ or Linux®) and effectively execute any command. While this is expected behavior -- if they can log in to the server, they can enter commands -- this ability can present a problem in some environments.

The following paragraphs explain how to implement a secure server preventing Genero Desktop Client users from executing arbitrary commands, by preventing client access to the (UNIX™ or Linux®) command line or shell while still allowing Genero applications to be started. This is accomplished by not giving them access to the shell, yet allowing the Genero Desktop Client to pass values to the system to indicate which application to start.

Important: This is intended to be the framework for a larger implementation and should be reviewed by your system administrator for any security concerns.

- [Prerequisites](#) on page 117
- [Solutions overview](#) on page 117
- [The shell script](#) on page 118
- [Setup SSH login](#) on page 119
- [Setup telnet](#) on page 122
- [Password management](#) on page 124
- [AUTOPORTFIND source code example](#) on page 126
- [Login script](#) on page 130

Prerequisites

This topic discusses prerequisites of configuring a secure server.

To implement a secure server, the following prerequisites must be met:

- Genero Desktop Client, version 1.32.1f or greater
- UNIX™ or Linux® platform
- SSH configured on the server
- Familiarity with Bourne or Korn shell programming
- Access to root for implementation

Solutions overview

This topic discusses replacing the login shell to implement a secure server.

When users log in, the system determines which shell to give them, based on a value in the `/etc/passwd` file. We will replace this shell with a shell script that will parse the values passed to it and set the environment accordingly. The application that is started will be from a list of valid applications; no other options will be accepted (thus controlling what a user can do).

Passing Values to the Script

The Genero Desktop Client must pass specific information to the script:

- The *application name* must be passed if more than one application exists. You can add additional logic to the script to control which users have access to specific applications.
- The *port* accepting connections for the Genero Desktop Client is important so that the application can connect back to the Genero Desktop Client to display information.
- The *two security values* prevent anyone from spoofing the connection. The DVM must make a socket connection to the Genero Desktop Client for the application screens and user interaction. `@FEID` and `@FEID2` contain a value that must match on both the client and server. The Genero Desktop Client compares the `@FEID` value it has internally and the one it received from the DVM attempting to connect. If they do not match, it assumes an application it did not start is trying to connect and rejects the connection. Likewise, `@FEID2` contains a value that the DVM must receive from the Genero Desktop Client in order to validate that the Genero Desktop Client is the one that started it. These

security values are enabled by specifying '-A 3' as a command-line argument when starting the Genero Desktop Client.

Auto Port Forwarding

With version 1.30, the automatic assignment of the port to use for port forwarding was added to the feature set of the Genero Desktop Client. Port Forwarding is the term used for tunneling with ssh. It allows applications to connect back to the client via a port that is open on the server, tunneled through the ssh secure client connection, then connects to the Genero Desktop Client on the client. The port is specified by the client, but it is usually not known whether this port is in use on the server prior to initiating the connection. In an enterprise this could be a problem, because every forwarded port must be unique between users.

The solution is to ask the server system for a port number to use. Because there is no way to reserve the port, we must get the number and open it quickly. Once we have the port opened for our session, we will have it until we log off and the connection is closed. We use a small C program that uses network system calls to allow the server to assign a port number. This port number is produced by the operating system by incrementing some internal OS counter and issuing numbers from a pool. If the port it would assign is in use, it will automatically increment the value until it finds an unused port. The next number it assigns to us, or to any other network request, will be managed the same way. This process insures to a large degree that the number we get will not be reassigned or used for some time, certainly long enough for our purposes.

Note: Version 2.30 introduces [Automatic Port Forwarding](#). Fgltty is now able to get a free port and pass it to GDC so the ssh tunnel can be set up automatically. In most of the cases this should work and fit your needs, but if you want to assign a specific port number or have a full control over the ports that are used, you can still follow this process:

Process Summary

- Log in.
- Get a port number from the system.
- Close the connection.
- Establish another connection and provide that port number for the tunnel.
- Log in (again).
- Start the application.

In normal situations the terminal activity of this process is hidden. The users simply see their application appear.

The shell script

This topic covers the steps required to replace a login shell with a customized script.

The shell script accepts the information on the command line and parses it, assigning values as needed to start the application. The application name is matched in a case statement, preventing direct execution of what the user sends.

The script [provided later in this section](#) is intended to be an example, and we expect you to tailor it according to your needs. Save it in a location where it can be executed but not changed by your users. Edit the /etc/passwd file to make a user call the script instead of a shell. Here is an example of the user "user1" running the script named "gdcstart".

```
user1:x:569:569: : /home/user1: /home/user1/gdcstart
```

The script [LOGIN_SCRIPT](#) is designed to recognize the difference between being started from sshd or from telnetd. You could modify it to handle either condition differently. For example, you may want it to start an application in text mode when accessed via telnet, or in GUI mode when accessed via ssh.

Setup SSH login

Configure a GDC shortcut to launch the application and implement port forwarding.

An advantage of using ssh and port forwarding is that the GUI information is encrypted during transmission. However, the unused port must be assigned on the server for the tunnel -- a difficult task if you are the system administrator. To solve this, we ask the server to tell us what port to use. This section shows how to implement this solution while maintaining system security.

As stated previously, we use a shell script to start the requested application instead of giving the user a shell; the [login script](#) is used for that purpose. In order for the script to work properly, the information in the Command Line field of the Genero Desktop Client shortcut must be altered accordingly to launch the application. The automatic assignment of the port forward number must also be set up.

This is the Genero Desktop Client shortcut entry for using ssh.

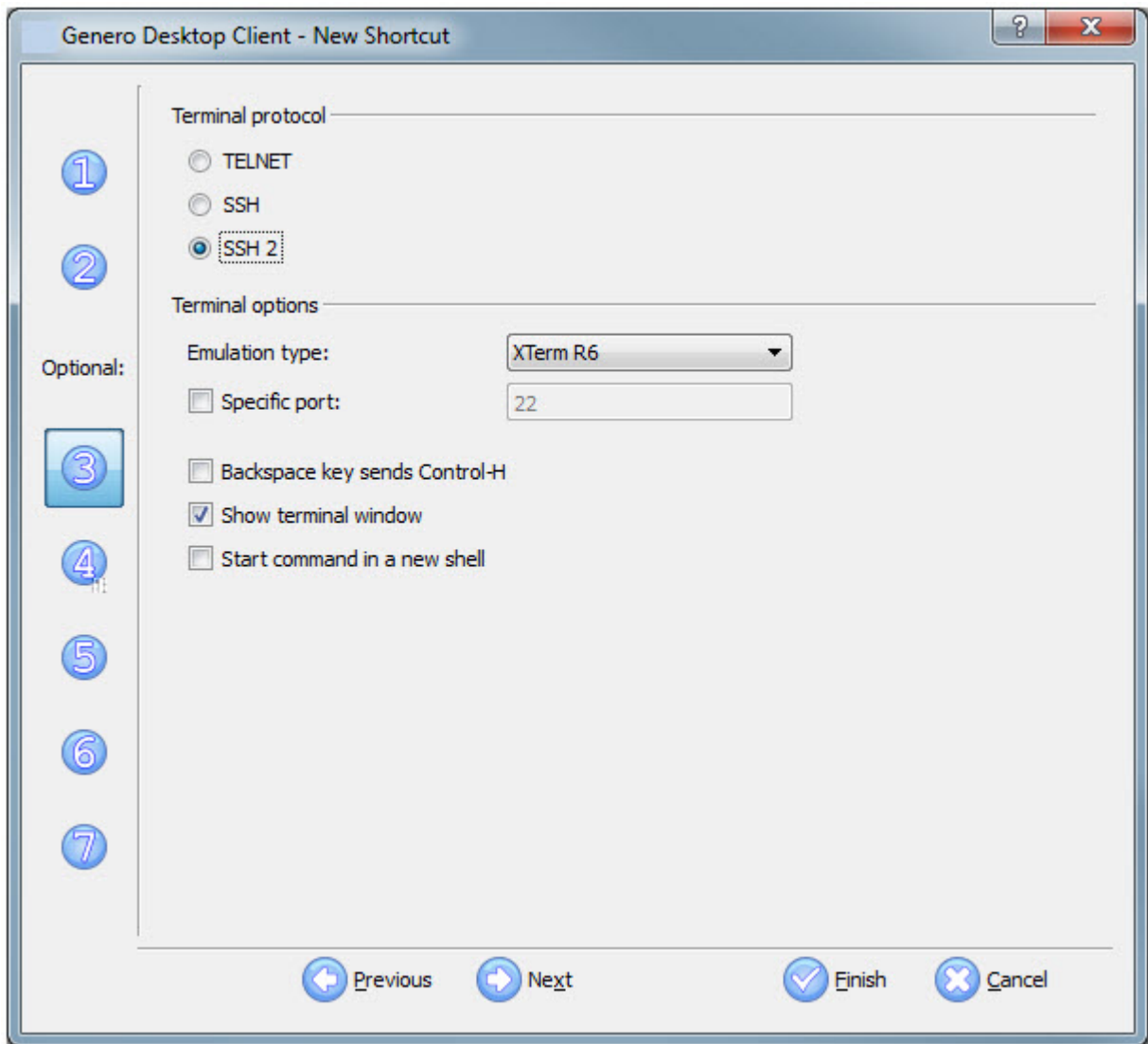


Figure 53: The Genero Desktop Client shortcut entry for using ssh.

In the Command field, we have specified `AUTOPORT`. This corresponds to an option near the end in the [login script](#).

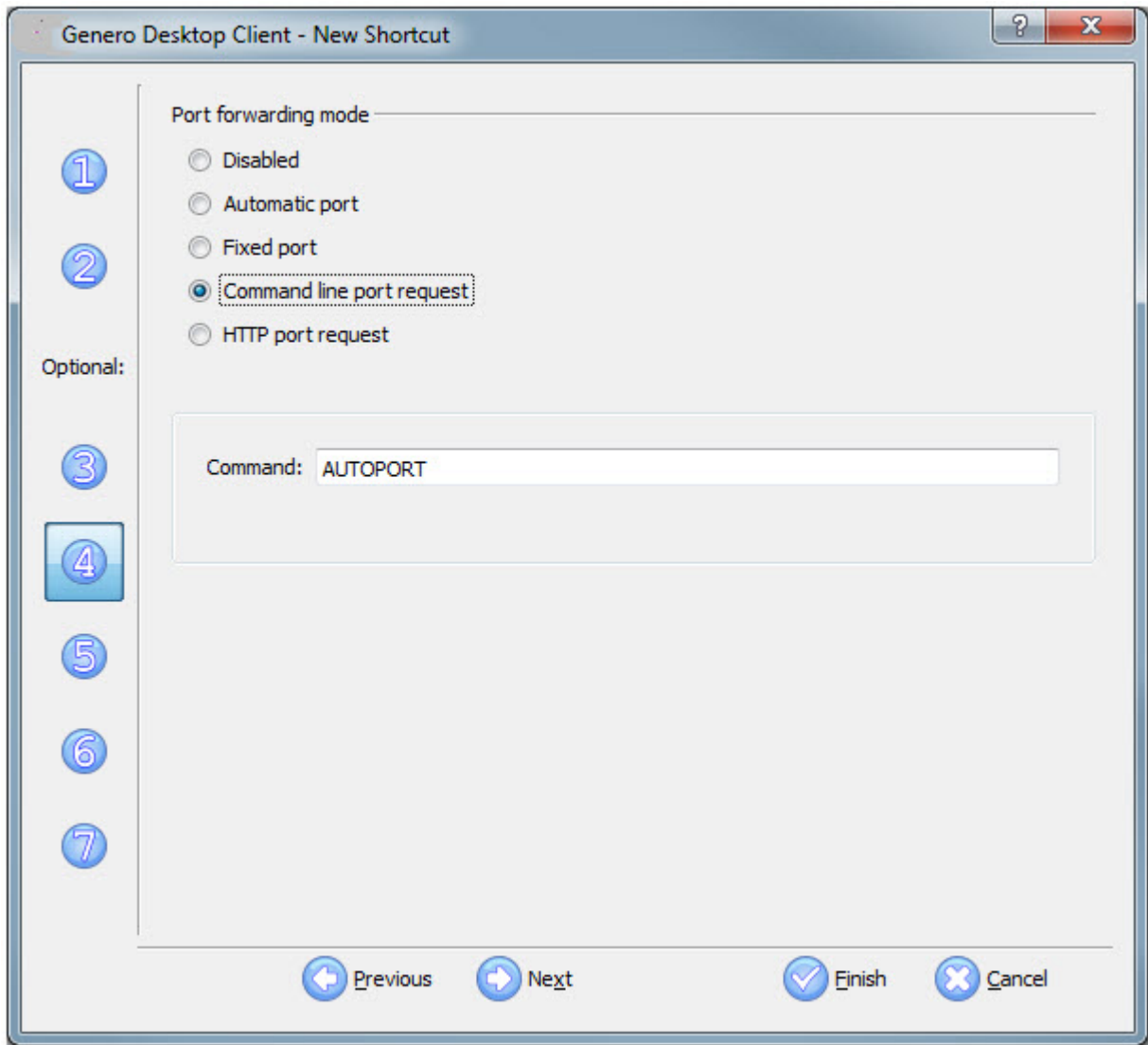


Figure 54: Setting AUTOPORT

When the login script receives "AUTOPORT", it executes a program called [autoportfind](#). The `-e` option will make it output a string like "FJSPORTFORWARD=*nnnn*" where *nnnn* is the port number provided by the operating system. The string matching rule we use looks for FJSPORTFORWARD= and retains the number following the =. This session is then closed and a new session is started using that number as the port to forward. It should not matter where in the sequence this rule is added.

You will also need to make an addition in **Terminal Strings**.

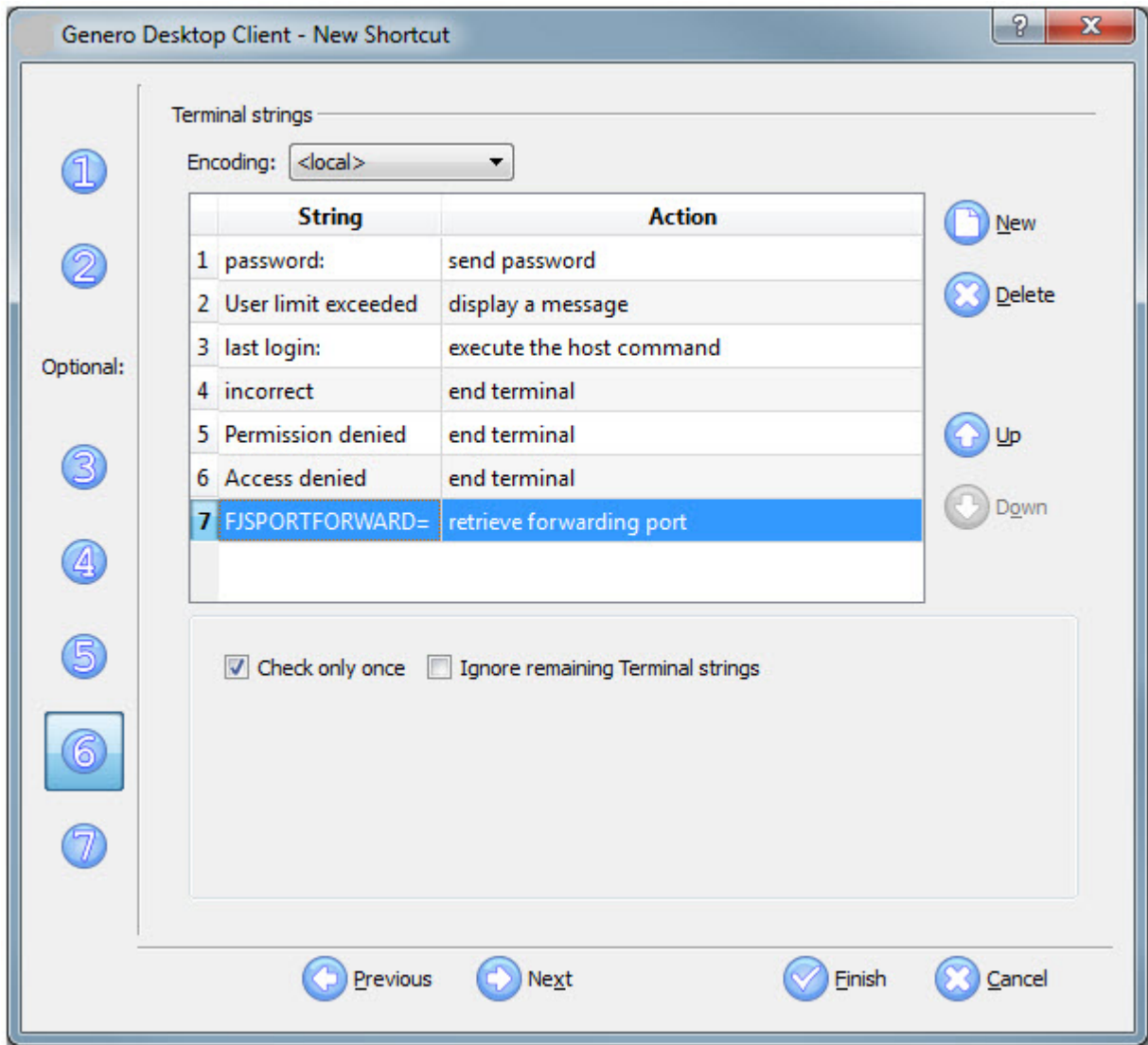


Figure 55: Configuring FJSPORTFORWARD in Terminal strings

Normally, the Command Line is passed to the shell that is started when a user logs in. Since we are using our shell script, the Command Line is where we specify the application to run, and pass the port number and the security fields. In our example we want to run the demo application. The command `DEMO` can be changed to your own application name, and an entry in the [login script](#) can then be added to start your application.

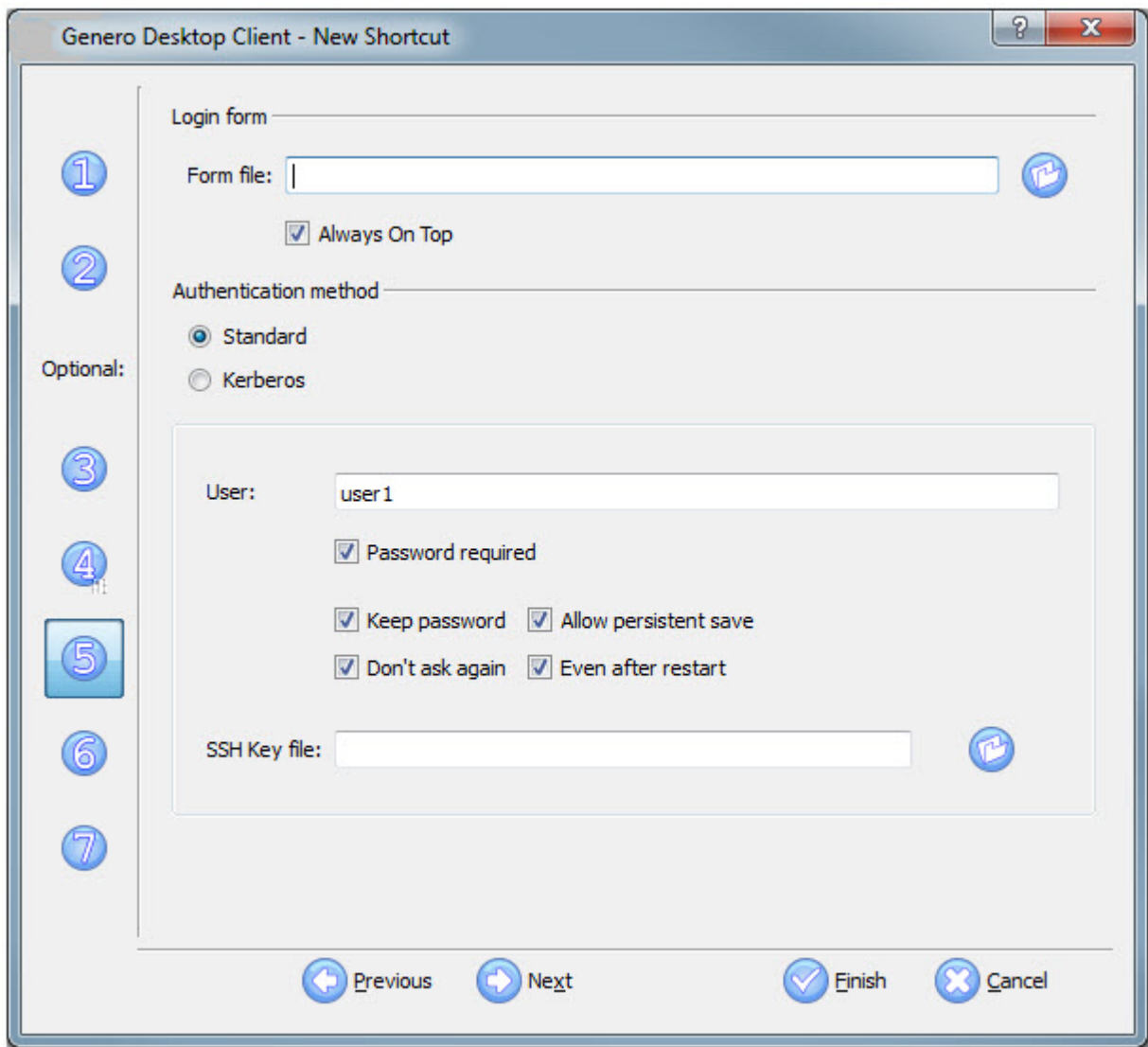


Figure 56: Run as user1

When the shortcut is run, it will log in using AUTOPOINT first. This will match a case statement in the script, and return a string "FJSPORTFORWARD=nnnn" where *nnnn* is a port number. Genero Desktop Client will then close the connection, and log in again using that port for the port to forward (tunnel) and pass it on the command line of the server @SRVNUM. This is what the login script uses to set the environment for the execution of the command DEMO. When using Port Forwarding, the server (127.0.0.1) is always the target for FGLSERVER (and therefore only the port number is needed).

Setup telnet

This topic describes the steps to configure GDC connections using telnet.

Telnet doesn't offer port forwarding, so the setup is a bit simpler. But it also doesn't give the flexibility needed when going through firewalls, and offers no encryption or privacy like ssh.

You simply need to pass the required arguments via the command line, and the [login script](#) sets the environment and launches the application.

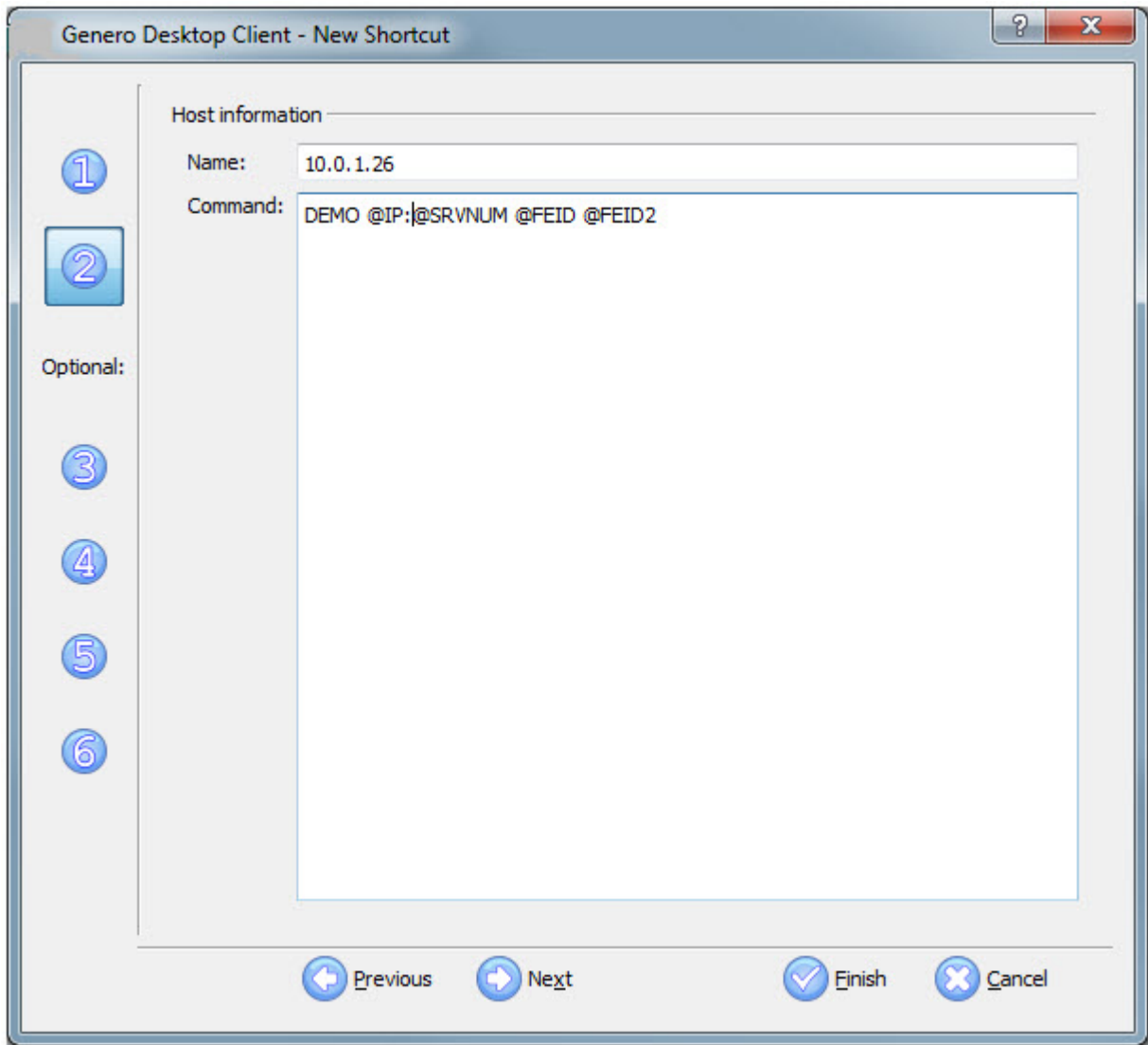


Figure 57: Specify the command line arguments for telnet

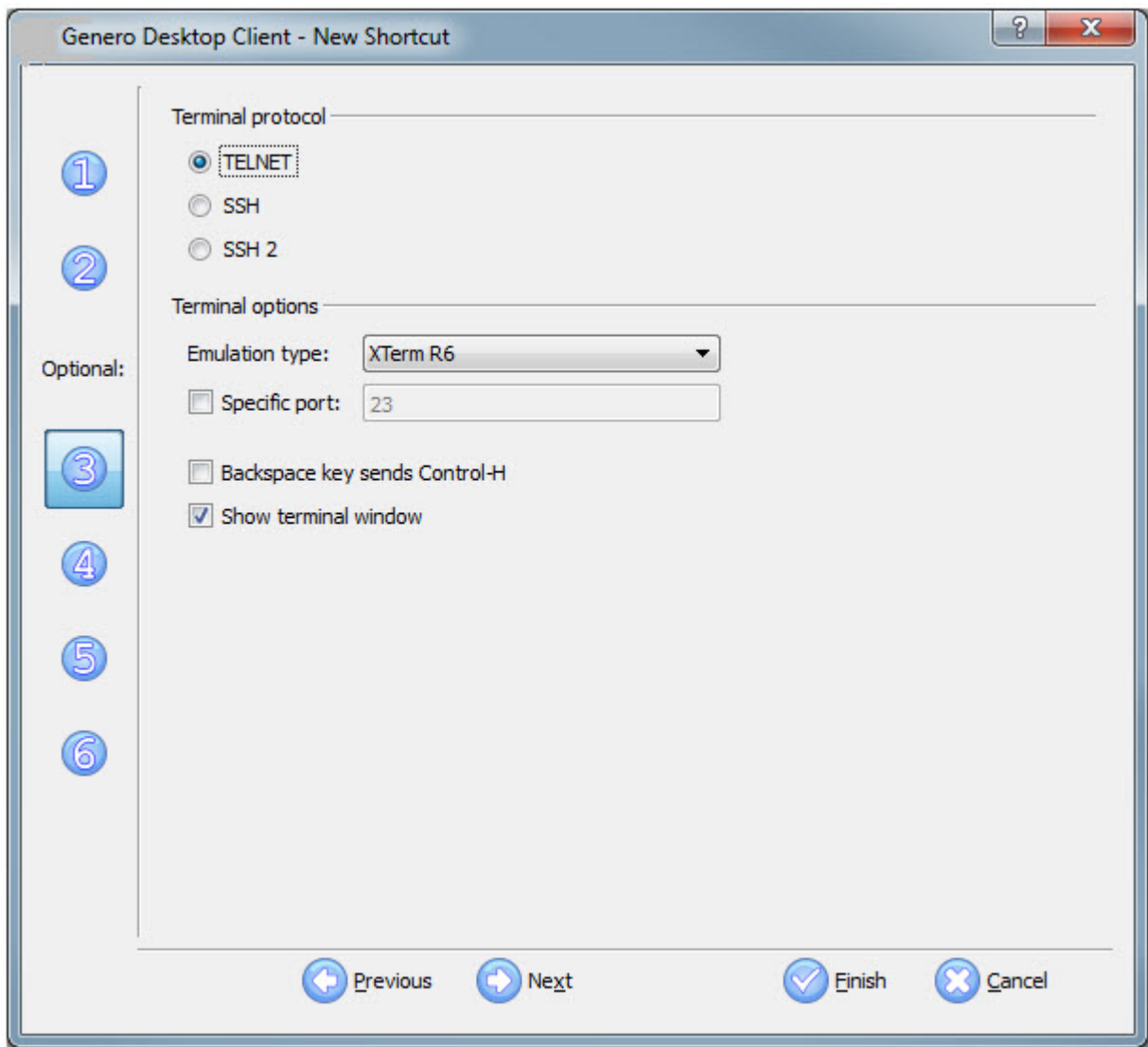


Figure 58: Select TELNET

With ssh and tunneling, the IP address is not needed because the tunnel is listening on the same server that will run the application. But with Telnet, we must pass the client machine's IP and port using @IP and @SRVNUM. The security values are passed as well, so the environment is complete. For the Genero Desktop Client to make use of the security values, you must start it with the option "-A 3" on the command line of the Genero Desktop Client. Put your application name in place of DEMO, and make an entry in the [login script](#) accordingly.

Password management

Secure server password management.

- [Handling expired passwords](#) on page 124
- [Changing passwords](#) on page 125

Handling expired passwords

This section explains how to configure GDC behavior for expired passwords.

To handle expired passwords, edit the shortcut and add a filter under **Manage Connection Strings**. For the string **Your password has expired**, the action of **show terminal** should be set.

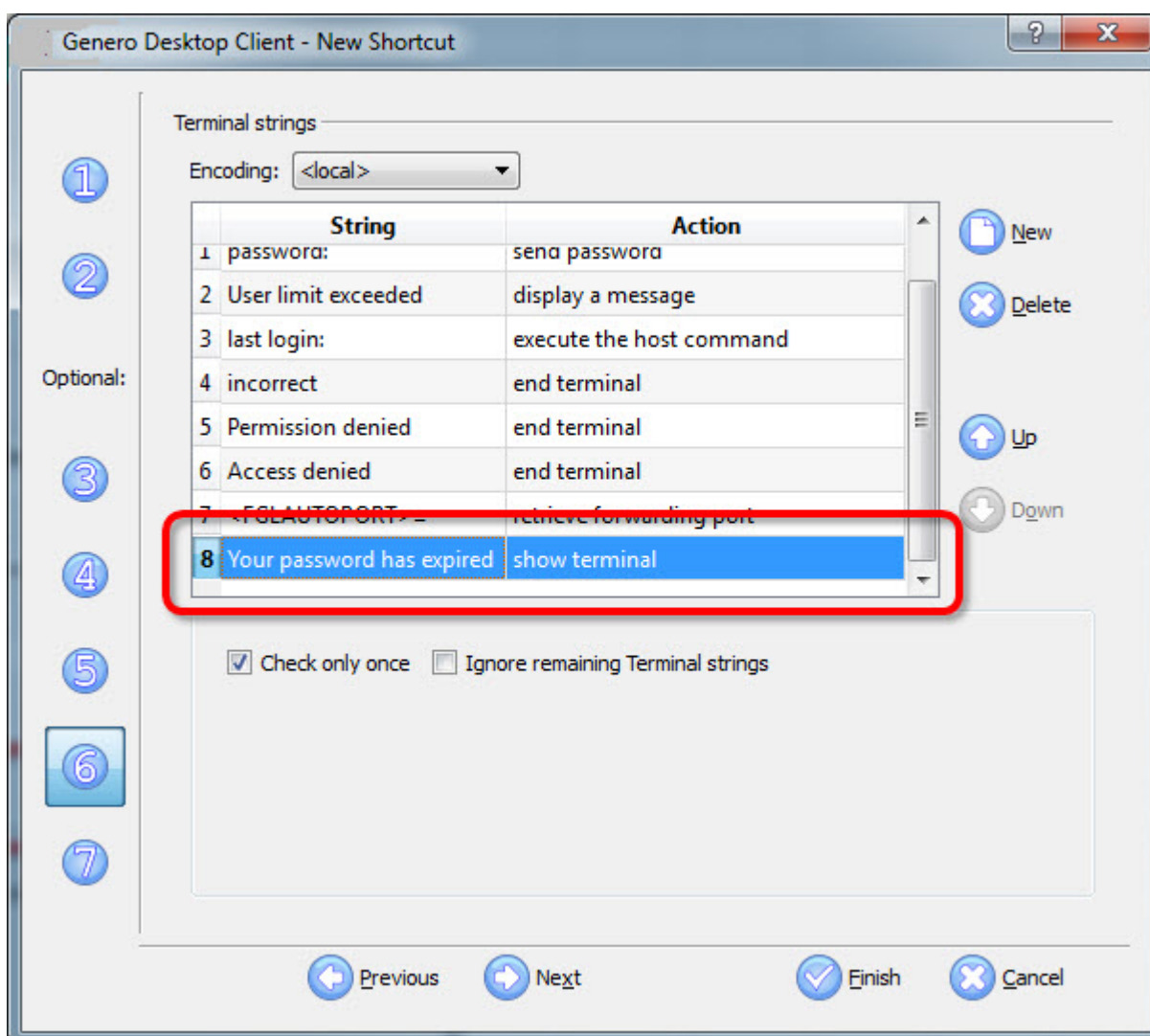


Figure 59: Setting Your password has expired

This rule looks for **Your password has expired** and open a text dialog window. Internally, the terminal window prompts for a new password from the server, as the existing password has expired. **Show the terminal** causes the Genero Desktop Client to display the server window, allowing the user to see the message and type in the correct passwords to complete the process. The window then closes and the user can click the shortcut once more and use the new password to start the application.

Important: The string entered in the Received String field must match the string displayed by the system. It is case-sensitive, where "Password has expired" does not match "password has expired". The string for an expired password may be different than the example shown above, based on your system. You should verify the string for an expired password that is returned by your system prior to implementing this solution.

Changing passwords

Create a shortcut to support password changes.

Users may want to change their passwords prior to expiration. To allow for this functionality, provide a shortcut in the Genero Desktop Client that issues the password command. The sample [login script](#) uses a case statement that checks for PASSWD. The specifics of the shortcut are as follows:

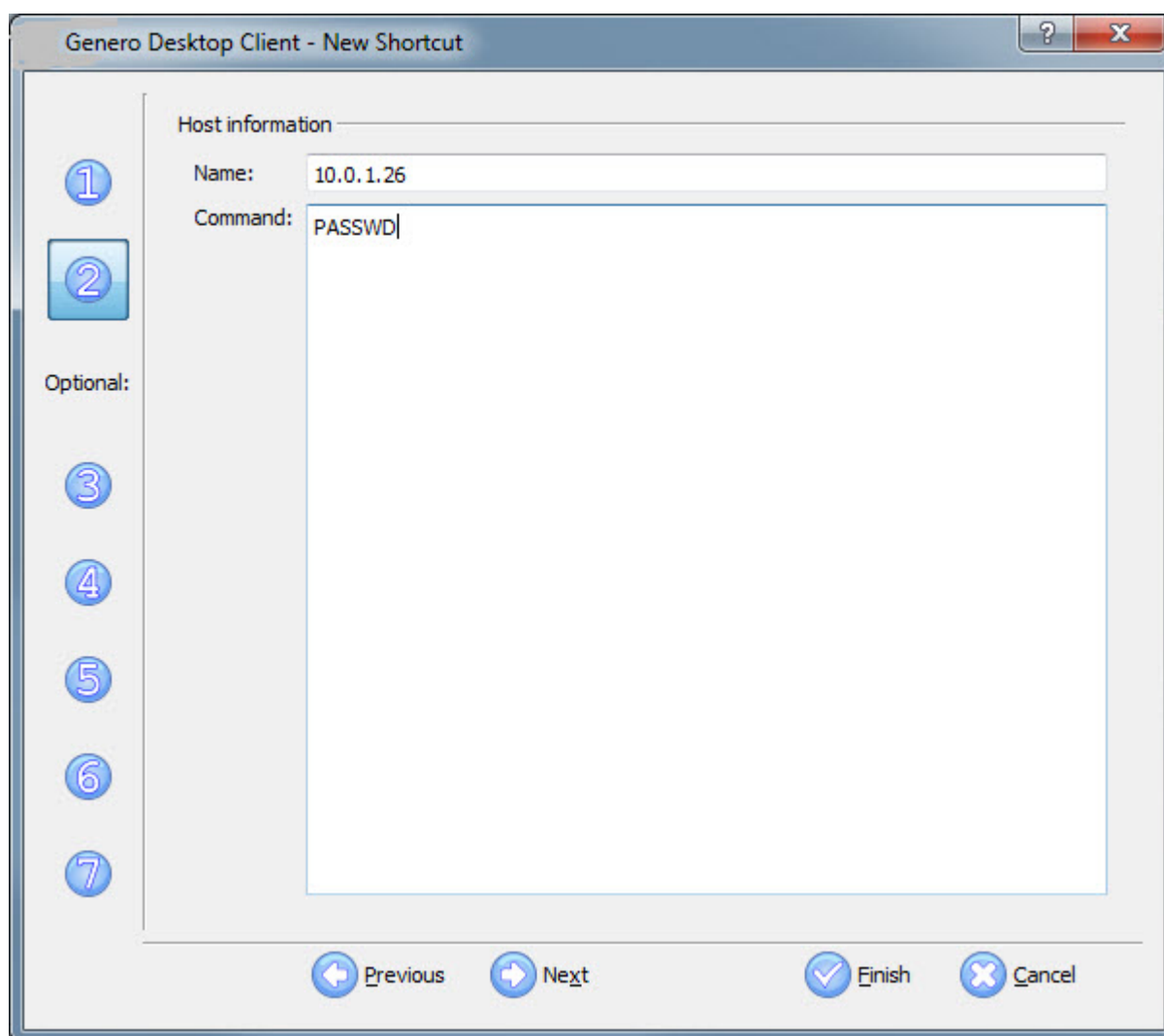


Figure 60: PASSWD command

AUTOPORTFIND source code example

This section provides an example of the source code to produce the port number for tunnelling with ssh.

This script should compile with little or no modification and does not need to be run as root.

```

Autoportfind.c/*
Written by John A. Hobach, Dallas Texas, May 5th, 2004
The purpose of the application is to return a port number that
will not be used for awhile. This port number can then be used
by the Genero client for port forwarding.
The operating system assigns ports in a round
robin fashion so the port assigned is unlikely to be used again
very soon. This will give the GDC time to start ssh and use
that port. The OS will automatically skip ports in use.
Revised 08/25/2004 Ver 2.1 to use bind() to get a port number
assigned. It is assigned a port automatically from the
operating system and we immediatly get it and return it.
Revised 10/25/2005 Ver 2.2 to support returning a port number
within a given range. This is accomplished by requesting ports
from the OS until it is within the range specified.
*/
#include <stdio.h>

```

```

#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#define USE_SOCKETS
#include "util.h"
char *progname;
static char *ver="autoportfind - Version 2.2, 2005-10-20";
static char *help=
"autoportfind [OPTION]\n"
"\n"
"Generate a port number for use with port forwarding.\n"
"\n"
"    -e, --env\n"
"        Send FJSPORTFORWARD=<port> to stdout.\n"
"\n"
"    -r    Cycle through port assignments to determine which ports\n"
"          the OS assigns to ports when originating connections.\n"
"    -u n  Upper limit. Request port numbers until one is returned\n"
"          below 'n'.\n"
"    -l n  Lower limit. Request port numbers until one is returned\n"
"          above 'n'.\n"
"    -h    Display this help message.\n"
"    -v    Display the version number.\n"
;
main(int argc, char **argv) {
    int sockfd, connected_socket, retval;
    int size, x, outofrange;
    int range_flag=0, env_flag=0;
    unsigned int    port, startport, highest,
                    lowest, cycle, direction,
                    llimit=0, ulimit=~0;
    int reuse_addr=1;
    char **arg;
    struct sockaddr_in serv_addr;
    progname=argv[0];
    arg=argv;
    while (--argc) {
        ++arg;
        if (!strcmp(*arg, "-r") || !strcmp(*arg, "--range")) {
            range_flag=1;
        } else if (!strcmp(*arg, "-e") || !strcmp(*arg, "--env")) {
            env_flag=1;
        } else if (!strcmp(*arg, "-u")) {
            ++arg;
            if (argc == 1 || *arg[0] == '-') {
                fprintf(stderr, "%s: Value missing for -u\n", progname);
                exit(1);
            }
            --argc;
            ulimit=atol(*arg);
        } else if (!strcmp(*arg, "-l")) {
            ++arg;
            if (argc == 1 || *arg[0] == '-') {
                fprintf(stderr, "%s: Value missing for -l\n", progname);
                exit(1);
            }
            --argc;
            llimit=atol(*arg);
        } else if (!strcmp(*arg, "-v")) {
            printf("%s\n", ver);
        }
    }
}

```

```

        exit(0);
    } else if (!strcmp(*arg, "-h") || !strcmp(*arg, "--help")) {
        printf("%s", help);
        exit(0);
    } else {
        fprintf(stderr, "%s: Unknown argument '%s'\n",
                progname, *arg);
        exit(1);
    }
}
lowest=~0;
highest=0;
startport=0;
cycle=0;
direction=1;
do {
    outofrange=0;
    memset((char*) &serv_addr, 0, sizeof(serv_addr));
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_port=0; /* allow system to assign */
    serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
sockfd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (sockfd < 0) {
        perror("socket");
        close(sockfd);
        exit(1);
    }
    if (bind(sockfd, (struct sockaddr *) &serv_addr,
            sizeof(serv_addr)) < 0) {
        perror("bind");
        close(sockfd);
        exit(1);
    }
    size=sizeof(serv_addr);
    if (getsockname(sockfd, (struct sockaddr *) &serv_addr,
                    &size) == -1) {
        perror("getsockname");
        exit(errno);
    }
    if (range_flag) {
        port=ntohs(serv_addr.sin_port);
        if (!startport) startport=port;
        if (port > highest) highest=port;
        if (port < lowest) lowest=port;
        if (direction==0 && port <= startport) {
            cycle++;
            direction=1;
        } else if (direction==1 && port >= startport) {
            cycle++;
            direction=0;
        }
    } else {
        port=ntohs(serv_addr.sin_port);
        if (port > llimit && port < ulimit) {
            if (env_flag) printf("FJSPORTFORWARD=");
            printf("%d\n", ntohs(serv_addr.sin_port));
        } else
            outofrange=1;
    }
    close(sockfd);
} while ((range_flag && cycle < 3) || outofrange);
if (range_flag)
    printf("Lowest port: %lu\nHighest port: %lu\n", lowest, highest);
exit(0);

```



```

}

---
Util.h
#ifndef UTIL_H
#define UTIL_H
#ifndef MAX
# define MAX(a,b) a>b?a:b
#endif
#ifdef USE_SOCKETS
# ifdef _WIN32
#   include <winsock.h>
# else
#   include <sys/types.h>
#   include <sys/socket.h>
#   include <netinet/in.h> /* struct sockaddr_in, ... */
#   include <netinet/tcp.h> /* TCP_NODELAY, ... */
#   include <arpa/inet.h> /* inet_addr, inet_ntoa, inet_aton */
#   include <netdb.h> /* gethostbyname */
# endif
#endif

#ifdef _WIN32
# define SOCKLEN_T int
#endif

#ifdef __osf__
# define SOCKLEN_T int
#endif

#ifdef _AIX
# ifdef USE_SOCKETS
#   include <sys/ioctl.h>
#   include <sys/time.h>
#   include <sys/select.h>
# endif
# define SOCKLEN_T socklen_t
#endif

#if defined (M_I386)
/* SCO */
# ifdef USE_SOCKETS
#   include <sys/ioctl.h>
#   include <sys/time.h>
#   include <sys/select.h>
# endif
# define SOCKLEN_T int
#endif

#ifdef linux
# define SOCKLEN_T socklen_t
#endif

#ifdef sun
# if defined USE_SOCKETS
#   undef USE_SYS_SOCKETIO
#   define USE_SYS_SOCKETIO
# endif
# define SOCKLEN_T int
#endif

#ifdef __hpux
# define SOCKLEN_T int
#endif

```

```

#endif
#define SOCKLEN_T size_t
#endif

#endif
#define MSG_DONTWAIT 0
#endif

#endif

```

Login script

This section provides an example of the login script that is executed when users log in.

It is intended to be an example, and we expect you to tailor it according to your needs. The login script is invoked via the `/etc/passwd` file.

```

#!/bin/sh

# Invoked directly by login mechanism such as telnetd, or sshd.
# This file is specified in the /etc/passwd file as being the shell. This
# gives us the control we need for users that should never be allowed a
# shell prompt.
#
# For backward compatibility we check to see if we are coming from a
# non-sshd source. If so then we invoke the shell as usual and have
# it source all the login scripts
#
# Arguments passed are <COMMAND> <PORT> <FEID> <FEID2>
#
# <COMMAND> string must match the case statements.
#

# set your env vars here
export FGLDIR=/fjs/f4gl/genero-training
export FGLRUN=fglrun
export FGLGUI=1

# The command line arguments passed from the GDC will be here. If there
# aren't any then we abort.

if [[ "$SSH_TTY" == "" && "$SSH_CONNECTION" == "" ]]
then
# coming in from telnet

echo -n "$ " # fake shell prompt for GDC
read APPLICATION FGLSERVER _FGLFEID _FGLFEID2

if [[ "$APPLICATION" == "" ]]
then
echo "exiting due to bad arguments"
sleep 5 # give time to view error because window will close
exit 0
fi

export FGLSERVER
export _FGLFEID
export _FGLFEID2

else
# coming in from ssh and sshd

```

```

if [[ "$1" == "" || "$1" != "-c" ]]
then
echo "exiting due to bad arguments"
sleep 5 # give time to view error because window will close
exit 0
fi

shift
args=(`echo $1`)
export APPLICATION="${args[0]}"
export FGLSERVER="127.0.0.1:${args[1]}"
export _FGLFEID="${args[2]}"
export _FGLFEID2="${args[3]}"

fi

#echo "APPLICATION=$APPLICATION"
#echo "FGLSERVER=$FGLSERVER"

# Add case statements according to 1st value passed from the GDC command
line.
# Never execute the value passed directly as this would be a security hole
# allowing the client to dictate what gets run.
#
case "$APPLICATION" in
YOURAPP) cd $FGLDIR/demo
/bin/bash --login -c "$FGLRUN demo"
;;

DEMO) cd $FGLDIR/demo
$FGLDIR/bin/$FGLRUN demo
;;

# SHELL) /bin/bash # don't leave this in for production
# ;;

AUTOPORT) /home/portfind/autoportfind -e
exit 0
;;

PASSWD) /usr/bin/passwd
exit 0
;;

*) echo "Unknown application '$APPLICATION'"
sleep 5 # allow time to read message
;;
esac

```

SSH Configuration Troubleshooting

Possible configuration issues when implementing SSH.

- [Wireless systems](#) on page 132
- [Need to change the port that GDC listens on](#) on page 132
- [Sessions expiring](#) on page 132

Wireless systems

Lost signals with wireless connections can cause connection loss.

The latest technology to use is 802.11(a,b or g). This is great at avoiding the wire mess, but there is a new risk. Under Windows™, if you are using a plugged in or built-in wireless card, the interface goes offline if the signal is lost for even a second. When this happens, it is treated similar to unplugging your network cable. The Windows™ drivers report to the network stack that the interface is now offline, and everything associated with that interface is removed. If an application has an open channel, it is signaled that it has closed. As a result, you lose all your connections and must wait for your signal to return in order to log in again.

A possible workaround is to use an external wireless device that doesn't take the connection down when the signal is lost. This works because it doesn't look like the cable was unplugged when it loses signal, so Windows™ doesn't know there is a problem. When the signal returns, everything works just at before.

Need to change the port that GDC listens on

GDC port can be changed when required.

Why would you want to change the port that GDC listens on?

You may need to run several versions of the GDC on the same machine. Since each one must have its own listening port, Genero allows you to specify the port. If you run more than one and don't specify the port, Genero opens the next available port. For example, the first instance would open 6400, the next instance would open 6401.

```
>gdc -<- The port assigned would be 6400
>gdc -n -<- The port assigned would be 6401
>gdc -n -p 7400 -<- The port assigned would be 7400
>gdc -n -p 7400 -<- The port assigned would be 7401
>gdc -q -p 7400 -<- GDC won't start since the port 7400 is already
assigned
```

Another reason to change ports might be that you can't use the ssh functionality. What if you haven't installed the SSH package yet, but you have more than one client behind the same firewall router? You can add rules to the router to send 6400 to the first client, 6410 to the second client, and so on. Each client would be started with the corresponding -p <port>, and the router would make sure each client gets the connections intended for it.

Sessions expiring

Routers may expire sessions.

If you have sessions expire or applications that disappear, check for routers that expire sessions. Most likely, there is a firewall router in the path. If you are using a firewall router, check for session expiration timers for the ports used to get through the firewall. The expiration duration (KeepAlive) should be set greater than the interval set in your operating system. This is set to 2 hours as a default on most computers. The operating systems can be tuned to have shorter values, but it is usually easier to adjust the router; use a value of 2 hours and 10 minutes.

GDC and Windows™ XP Service Pack 2

Issues to be aware of when running the Genero Desktop Client on Windows™ XP Service Pack 2.

- [GDC and Windows XP Service Pack 2 Firewall Configuration](#) on page 133

GDC and Windows™ XP Service Pack 2 Firewall Configuration

Windows™ Firewall Control Panel settings.

Microsoft™ has added several security systems in Windows™ XP Service Pack 2 (SP2). The firewall included in Windows™ XP has been improved and is now enabled by default. From the network point of view, GDC is a server: it listens on a defined port (6400 by default) for Runtime System connections.

When GDC starts, the firewall detects that it listens on port 6400 and warns the user: Press **Unblock** to allow the GDC to run correctly.

Important: Pressing **Keep Blocking** or **Ask Me Later** will keep GDC from working. Connections from the Runtime System will be blocked by the firewall.

If **Keep Blocking** has been pressed by mistake, this parameter can be changed in the Firewall settings (Control Panel).

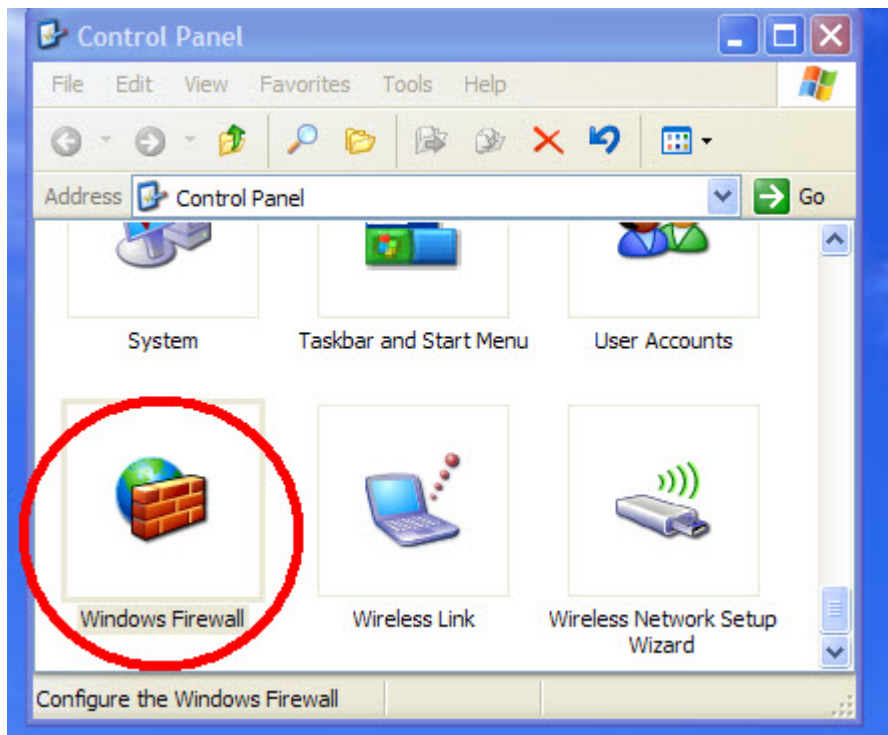


Figure 61: Control Panel; Windows™ Firewall

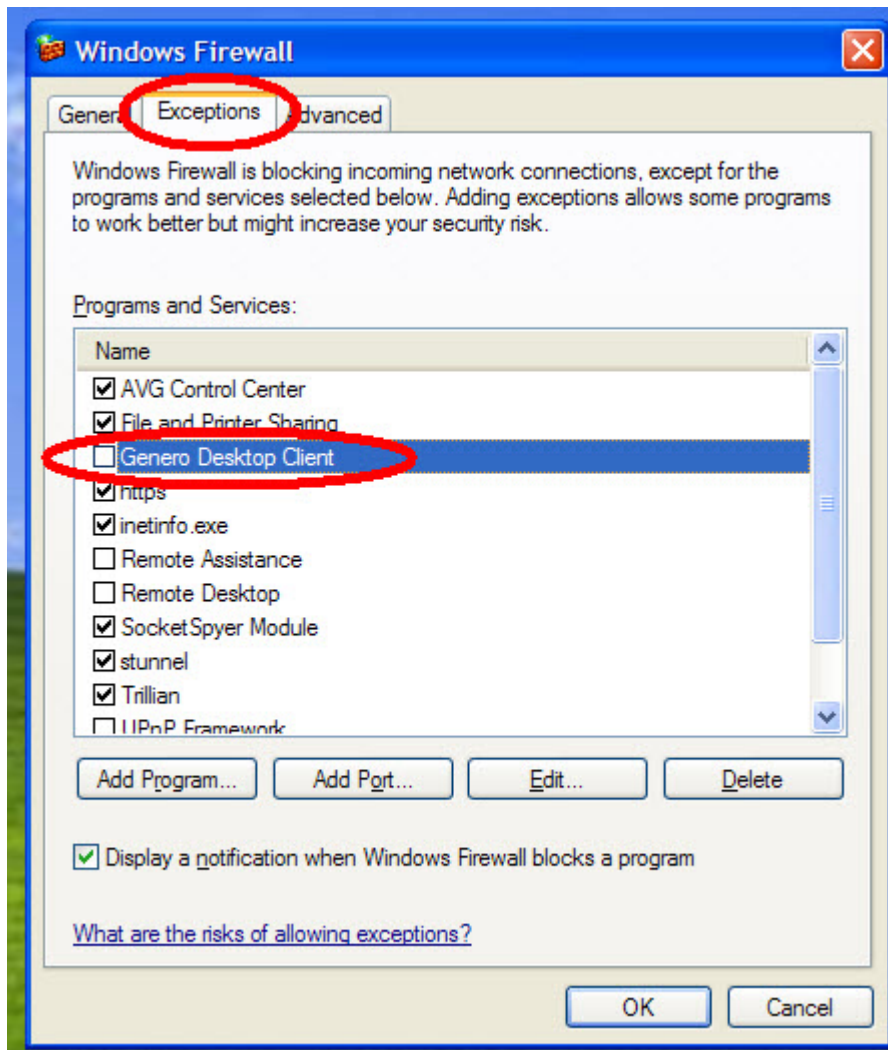


Figure 62: Windows™ Firewall; Exceptions Tab

In the **Exceptions** tab, Genero Desktop Client must be present and checked.

GDC and Windows™ Vista

Issues to be aware of when running the Genero Desktop Client on Windows™ Vista.

- [User Account Control](#) on page 134
- [Genero Desktop Client installation on Windows Vista](#) on page 135
- [Running the Genero Desktop Client on Windows Vista](#) on page 135
- [Genero Desktop Client features affected by the User Account Control](#) on page 136

User Account Control

User Account Control affects Genero Desktop Client.

One of the new features of Window Vista is the User Account Control (UAC). UAC prevents any software from silently hurting your system by prompting the user before any administrative actions such as:

- Installing a new program
- Modification of the registry

It requires a user with Standard User rights (users not in the *Administrator* group) to provide an Administrator login and password when running a program that performs system-level tasks. Administrator Users will only have to confirm their actions. More details can be found on the [Microsoft™ Web site](#).

The User Account Control feature affects the Genero Desktop Client [installation](#), as well as how [GDC is run](#).

Genero Desktop Client installation on Windows™ Vista

User Account Control prompts during installation.

When the installation program starts, you'll be prompted to validate the installation. If you are not logged in as an administrator, you will be asked for an administrator password.

The installation then continues as in Windows™ XP.

Running the Genero Desktop Client on Windows™ Vista

Starting the GDC after installation.

Once GDC is installed, the Windows™ Firewall will prompt the user to unblock the program, as described in [GDC and Windows™ XP Service Pack 2](#).

Although most of the features of Genero Desktop Client will work out of the box on Windows™ Vista, some features will only work if you start GDC as administrator.

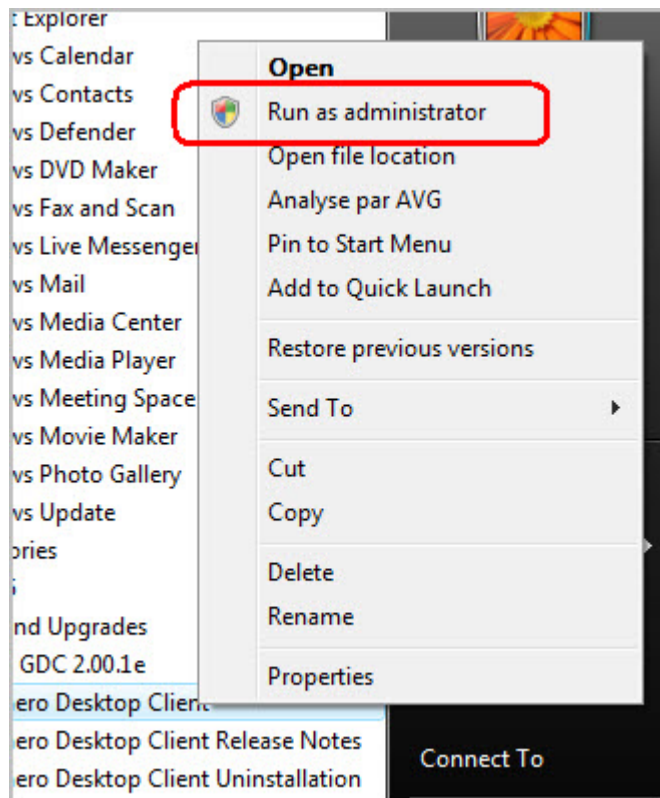


Figure 63: Run as administrator

Even an Administrator User has to run the program "as administrator". However, Administrator users can create a shortcut and specify in the **Compatibility** tab that this program is always run as an administrator.

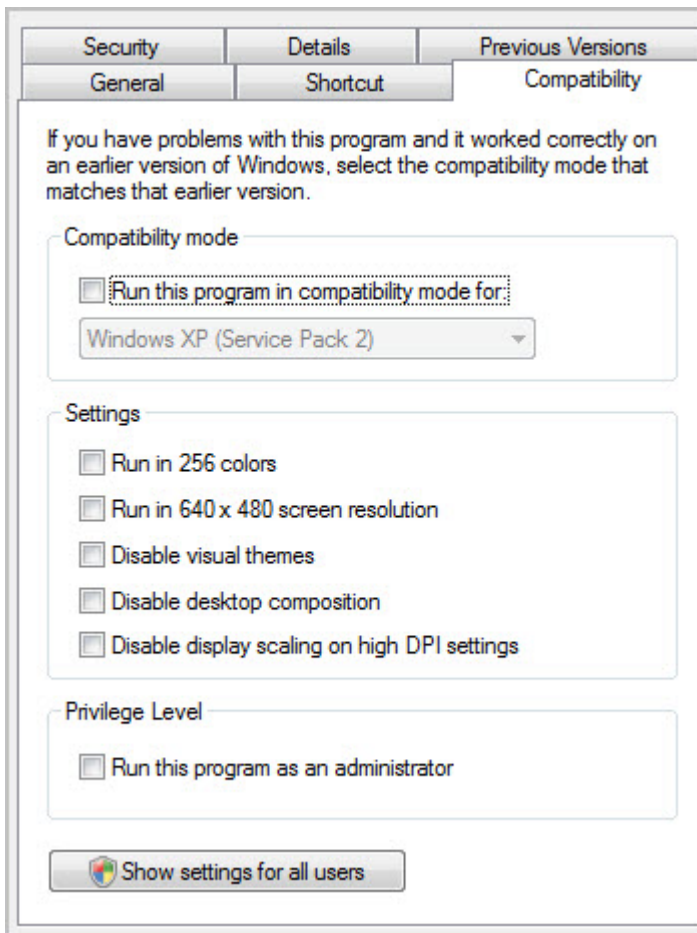


Figure 64: Compatibility tab

Note: Using the `-a` GDC command line option to run GDC in "admin mode", a mode where you can manage GDC shortcuts, is not the same as asking Windows™ Vista to start GDC "as administrator", a special mode where the program can perform system-level tasks.

Genero Desktop Client features affected by the User Account Control

Genero Desktop Client features affected by the UAC.

- [Configuration](#) on page 136
- [File Transfer](#) on page 136

Configuration

The GDC only allows changes via the registry with Windows™ Vista.

The folder `%Program Files%` is now protected with Windows™ Vista. So when starting GDC in normal mode, GDC will react as if the config file is read-only, and will only allow changes via the registry (local shortcuts). This may change in future versions of GDC.

File Transfer

Changes to file transfer with Windows™ Vista.

The ProgramFiles folder

As the folder `%Program Files%` is protected, Vista has introduced the concept of [Virtual Store](#): instead of using the `%Program Files%` folder, programs will access a special directory. This means that if you use `FGL_PUTFILE` without a destination, the file will use the `%VirtualStore%` folder to transfer the file.

The following example would use the virtual folder %VirtualStore%. to transfer the file myFile.txt:

```
CALL FGL_PUTFILE("myFile.txt", "myFile.txt")
```

Uploading files

Vista will prevent the upload of some files, such as executables or DLLs.

Example:

```
MAIN
  DISPLAY "upload text file..."
  CALL FGL_PUTFILE("myFile.txt", "myFile.txt")
  DISPLAY "... done"
  DISPLAY "upload DLL..."
  CALL FGL_PUTFILE("myFile.dll", "myFile.dll")
  DISPLAY "... done"
END MAIN

$fglrun ft
upload text file...
... done
upload DLL...
Program stopped at 'ft.4gl', line number 7.
FORMS statement error number -8066.
Could not write destination file for file transfer.
```

Running GDC as administrator will allow GDC to upload such files to the system.

Front End Extensions

The Genero Desktop Client allows you to call external functions from your Genero program. You can create your own front-end extensions.

These functions are dynamically loaded by the front end when needed. To create your own extensions and use them from within your Genero program, or to learn more about the APIs provided for Windows™ DDE support, Windows™ COM support and the Windows™ Mail extension, see the *Front calls* section of the *Genero Business Development Language User Guide*.

Terminology

In this section, many terms used throughout this document are briefly defined.

- [General terms](#) on page 139
- [Security terms](#) on page 140

General terms

This documentation uses several terms that must be clarified for a good understanding.

Product	The <i>Product</i> defines all software components that compose the information system managing a given domain. Usually, the domains covered by programs written in BDL are business oriented.
End User	The <i>End User</i> is the person that uses the Product; that person works on hardware called the Workstation .
Programs	The <i>Programs</i> are the software components that are developed and distributed by the supplier of the Product . <i>Programs</i> typically implement business rules and processing, usually called Business Logic. <i>Programs</i> are executed by the Runtime System on the Application Server machine. These components are typically p-code modules, forms and additional files.
Developer	The <i>Developer</i> is the person in charge of the conception and implementation of the Product components.
Application Data	<i>Application Data</i> defines the data manipulated by the Product . It is typically managed by one or more Database Systems . The <i>Application Data</i> has a volatile state when loaded in the Runtime System , and it has a static state when stored in the Database System.
Database	The <i>Database</i> is a logical entity regrouping the Application Data . It is managed by the Database System .
Database System	The <i>Database System</i> is the software that manages data storage and searching; it is usually installed on the Database Server machine and is supported by a tier software vendor. It is the software managing the Data in the Three-Tier C/S model.
Development Database	The <i>Development Database</i> is the Database used in the application development environment.
Production Database	The <i>Production Database</i> is the Database used on production sites.
Front End	The <i>Front End</i> is the software that manages the display of the User Interface on the Workstation

	<p>machine. This component is historically called "The Client", in a thin Client/Server context. It is the software managing the Presentation in the Three-Tier C/S model.</p>
Runtime System	<p>The <i>Runtime System</i> is the software that manages the execution of the Programs, where the Business Logic is processed. It is typically implemented by the <i>Dynamic Virtual Machine</i> (DVM) and historically called "The Runner". It is the software managing the Processing in the Three-Tier C/S model.</p>
User Interface	<p>The <i>User Interface</i> defines the parts of the Programs that interact with the end user, including interactive elements like windows, screens, input fields, buttons and menus. It is displayed on the Workstation. This can typically be implemented by different kinds of Front Ends, based on ASCII terminals, graphical platforms (Microsoft™ Windows™, X11) or even through web protocols like HTML over HTTP.</p>
Workstation	<p>The <i>Workstation</i> identifies the hardware used by the End User to interact with the Product. It can be an ASCII Terminal, a PC, a diskless station or even a cellular phone, as long as a Front End is available on that hardware.</p>

Security terms

The security section of the documentation uses several terms that must be clarified for a good understanding.

Firewall Router	<p>This is a device that isolates the corporate network from the Internet. It typically allows connections to the Internet, but also prevents connections from entering. They can usually be configured to allow/prevent several conditions. They can be configured to allow a port connection from the Internet to go through to a machine. This can be done either by allowing the connection straight through or translating it to a different port.</p>
NAT	<p><i>Network Address Translation</i> is a method of allowing computers to access the Internet without having them be assigned real Internet addresses. The connections must originate from the internal machines to reach Internet addresses. The <i>NAT</i> router will then put these on the Internet using the router's IP address. When data is returned it forwards the data to the requesting internal machine. Part of this process includes mapping what internal IP/Port combinations correspond to external port usage. Doing so allows the router to know where data needs to be sent when it returns. Special port mappings can be made to specific internal IP addresses to support connections</p>

originating from the Internet. Other configurable values might be session timers that will be explored in the section.

Private Network

This is the network used in the corporation that is private and trusted. Most companies tightly control what is plugged in so they can ensure the data is safe.

VPN

Virtual Private Network is a method of tunnelling through an existing connection back to the corporate LAN. It provides end-to-end encrypted connections. These types of connections are usually equivalent to being plugged into the office LAN.

Encryption of all Data

Genero requires a TCP connection for the GUI data transmission. If the GDC short cuts are being used there is also a connection needed to start the application that may require a log in. Both connections in this case are encrypted.

Password/login Encrypted

Genero logs in and executes an application when the short cuts are used. This connection is encrypted. The connection carrying the GUI data is not encrypted.

Keep Alive

Typical TCP connections don't cause any network traffic when idle unless the KeepAlive flag is set. This flag will prevent the session from timing out and thus prevent the session from closing. This also assumes that the firewalls don't expire the session during the keep alive interval.

Port Forwarding

The method referred to is implemented in the Secure Shell (ssh). The ssh can be told to listen to a port and tunnel it through an existing ssh session and present it to a port on the other machine. This method is used to listen to a port on the server side and direct the data to the GDC on the client side.

Note: This document covers system configuration using the following environment:

- Genero Desktop Client Release 1.20.1a (under Windows®, Linux® and Mac Os 10)
- Genero DVM Release 1.20.1a (Under Linux® and Windows®)
- Different Openssh Server 3.x.yy under Linux®

Legal notices

Genero Desktop Client legal notices.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.openssl.org/>).

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

This product includes software developed by CollabNet (<http://www.Collab.Net/>).

This product includes software developed by the University of California, Berkeley and its contributors.

This product includes software developed or owned by Caldera International, Inc