G2 Developer's Interface

User's Guide

Version 2015

G2 Developer's Interface User's Guide, Version 2015

December 2015

The information in this publication is subject to change without notice and does not represent a commitment by Gensym Corporation.

Although this software has been extensively tested, Gensym cannot guarantee error-free performance in all applications. Accordingly, use of the software is at the customer's sole risk.

Gensym Corporation
52 Second Avenue
Burlington, MA 01803 USA
Telephone: (781) 265-7100
Fax: (781) 265-7101

Part Number: DOC019-1200

Contents

Preface

Describes this guide and the conventions that it uses.

About this Guide

This guide describes the Windows-like G2 menu bar that provides menu options and dialogs for the classic G2 main menu and workspace menu. You can reuse or customize these menus and dialogs for your module applications.

Audience

This guide is for KB developers who want to use the Windows®-like menus and menu tools that the GDI utilities provide. You should be familiar with the key features in G2, as described in the *G2 Reference Manual*.

A Note about the API

The GDI API, as described in this guide, is not expected to change significantly in future releases, but exceptions may occur. A detailed description of any changes will accompany the GDI release that includes them.

Therefore, it is essential that you use GDI exclusively through its API, as described in this guide. If you bypass the API, you cannot rely on your code to work in the future, since GDI may change, or in the present, because the code may not correctly manage the internal operations of GDI.

If GDI does not seem to provide the capabilities that you need, contact Gensym Customer Support at 1-781-265-7301 (Americas) or +31-71-5682622 (EMEA) for further information.

Organization

This guide contains six chapters:

	Title	Description
1	Overview of GDI	Describes the four G2 utilities in the G2 Developer's Interface.
2	Using GDI Menus and Dialogs	Describes the menus in the G2 menu bar, the dialogs that support these menus, and the Print dialogs.
3	Modifying GDI Menus and Dialogs	Describes the some techniques, ranging from simple to complex, for modifying the menus and dialogs to suit your needs.
4	Customizing the Dialogs	Describes the file, module, and G2 session management capabilities provided by G2UIFILE.
5	Customizing the Tree View Control Dialog	Describes the tree-view capabilities provided by the G2UITREE module.
6	Customizing the Print Dialogs	Describes how the print dialogs interact with the Printer Setup system table through attribute settings and how you can use the print API.

Conventions

This guide uses the following typographic conventions and conventions for defining system procedures.

Typographic

Convention Examples	Description
g2-window, g2-window-1, ws-top-level, sys-mod	User-defined and system-defined G2 class names, instance names, workspace names, and module names
history-keeping-spec, temperature	User-defined and system-defined G2 attribute names
true, 1.234, ok, “Burlington, MA”	G2 attribute values and values specified or viewed through dialogs
Main Menu > Start KB Workspace > New Object create subworkspace Start Procedure	G2 menu choices and button labels
conclude that the x of y ...	Text of G2 procedures, methods, functions, formulas, and expressions
new-argument	User-specified values in syntax descriptions
text-string	Return values of G2 procedures and methods in syntax descriptions
File Name, OK, Apply, Cancel, General, Edit Scroll Area	GUIDE and native dialog fields, button labels, tabs, and titles
File > Save Properties	GMS and native menu choices
workspace	Glossary terms

Convention Examples	Description
`c:\Program Files\Gensym\`	Windows pathnames
`/usr/gensym/g2/kbs`	UNIX pathnames
`spreadsh.kb`	File names
`g2 -kb top.kb`	Operating system commands
`public void main()` `gsi_start`	Java, C and all other external code

Note Syntax conventions are fully described in the *G2 Reference Manual*.

Procedure Signatures

A procedure signature is a complete syntactic summary of a procedure or method. A procedure signature shows values supplied by the user in *italics*, and the value (if any) returned by the procedure *underlined*. Each value is followed by its type:

g2-clone-and-transfer-objects
(*list*: class item-list, *to-workspace*: class kb-workspace,
delta-x: integer, *delta-y*: integer)
-> *transferred-items*: g2-list

Related Documentation

G2 Core Technology

- *G2 Bundle Release Notes*
- *Getting Started with G2 Tutorials*
- *G2 Reference Manual*
- *G2 Language Reference Card*
- *G2 Developer's Guide*
- *G2 System Procedures Reference Manual*

- *G2 System Procedures Reference Card*
- *G2 Class Reference Manual*
- *Telewindows User's Guide*
- *G2 Gateway Bridge Developer's Guide*

G2 Utilities

- *G2 ProTools User's Guide*
- *G2 Foundation Resources User's Guide*
- *G2 Menu System User's Guide*
- *G2 XL Spreadsheet User's Guide*
- *G2 Dynamic Displays User's Guide*
- *G2 Developer's Interface User's Guide*
- *G2 OnLine Documentation Developer's Guide*
- *G2 OnLine Documentation User's Guide*
- *G2 GUIDE User's Guide*
- *G2 GUIDE/UIL Procedures Reference Manual*

G2 Developers' Utilities

- *Business Process Management System Users' Guide*
- *Business Rules Management System User's Guide*
- *G2 Reporting Engine User's Guide*
- *G2 Web User's Guide*
- *G2 Event and Data Processing User's Guide*
- *G2 Run-Time Library User's Guide*
- *G2 Event Manager User's Guide*
- *G2 Dialog Utility User's Guide*
- *G2 Data Source Manager User's Guide*
- *G2 Data Point Manager User's Guide*
- *G2 Engineering Unit Conversion User's Guide*
- *G2 Error Handling Foundation User's Guide*
- *G2 Relation Browser User's Guide*

Bridges and External Systems

- *G2 ActiveXLink User's Guide*
- *G2 CORBALink User's Guide*
- *G2 Database Bridge User's Guide*
- *G2-ODBC Bridge Release Notes*
- *G2-Oracle Bridge Release Notes*
- *G2-Sybase Bridge Release Notes*
- *G2 JMail Bridge User's Guide*
- *G2 Java Socket Manager User's Guide*
- *G2 JMSLink User's Guide*
- *G2 OPCLink User's Guide*
- *G2 PI Bridge User's Guide*
- *G2-SNMP Bridge User's Guide*
- *G2 CORBALink User's Guide*
- *G2 WebLink User's Guide*

G2 JavaLink

- *G2 JavaLink User's Guide*
- *G2 DownloadInterfaces User's Guide*
- *G2 Bean Builder User's Guide*

G2 Diagnostic Assistant

- *GDA User's Guide*
- *GDA Reference Manual*
- *GDA API Reference*

Customer Support Services

You can obtain help with this or any Gensym product from Gensym Customer Support. Help is available online, by telephone, by fax, and by email.

To obtain customer support online:

➔ Access G2 HelpLink at *www.gensym-support.com*.

You will be asked to log in to an existing account or create a new account if necessary. G2 HelpLink allows you to:

- Register your question with Customer Support by creating an Issue.
- Query, link to, and review existing issues.
- Share issues with other users in your group.
- Query for Bugs, Suggestions, and Resolutions.

To obtain customer support by telephone, fax, or email:

➔ Use the following numbers and addresses:

	Americas	Europe, Middle-East, Africa (EMEA)
Phone	(781) 265-7301	+31-71-5682622
Fax	(781) 265-7255	+31-71-5682621
Email	*service@gensym.com*	*service-ema@gensym.com*

1

Overview of GDI

Describes the four G2 utilities in the G2 Developer's Interface.

Introduction

The G2 Developer's Interface (GDI) is a set of G2 modules that enable you to develop a consistent Microsoft Windows®-like Graphical User Interface (GUI) for G2.

GDI provides:

- A starting point for G2 menu KB development that conforms to Windows standards.
- Pre-constructed menus and supporting dialogs for G2, using GMS, GFR, UIL, and GOLD.
- A development environment that lets you completely customize and localize all GDI capabilities, using conventional G2 techniques.

The four GDI modules include a Windows-like menu bar for G2 menu options, dialog masters that you can reuse, clone, or customize, a GUI for printing, a customizable tree-view control, a class and workspace browser utility, and API calls for each utility.

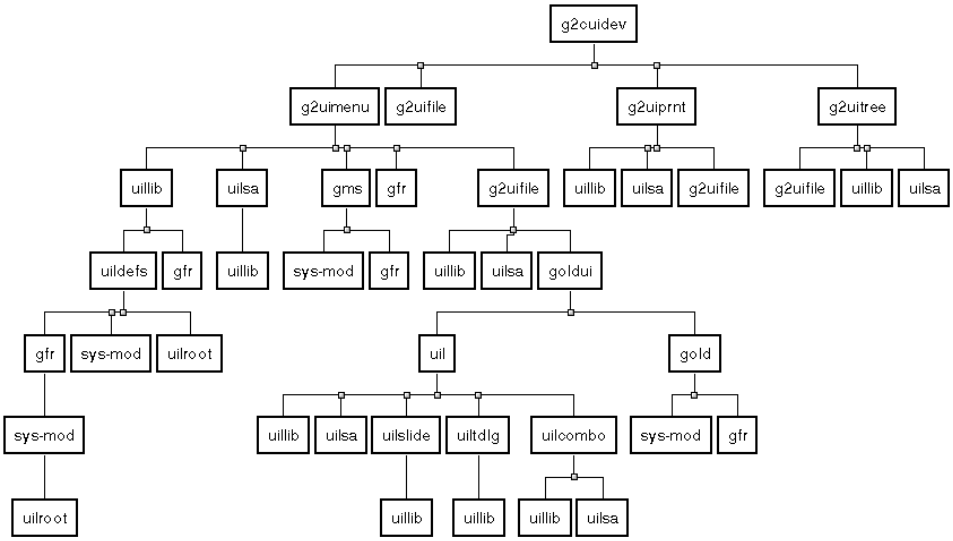
The GDI Modules

GDI is composed of a set of modules that deliver various tools and GUI components to developers of G2 applications. The top-level GDI module is g2cuidev. There are four GDI modules, each of which you can use as a separate utility.

The four modules of g2cuidev are:

- g2uimenu – A standard Windows-like menu system for G2.
- g2uifile – A series of dialogs for G2, KB, and module management.
- g2uiprnt – A general purpose printing tool for use by G2 applications.
- g2uitree – A tree-view tool that can be used for displaying the organization of files, classes, workspaces, or any hierarchical data.

Each GDI module is independent and can be loaded separately. The following figure shows the complete g2cuidev module hierarchy:



Using the GDI Modules in Your KB

You can use the GDI modules as they are, or you can customize them to meet most of your needs. You can use GDI in your KB by merging g2cuidev into your KB and making it a required module in your module hierarchy.

If you want to use only one or two of the GDI modules, you can merge the specified module into your KB and make it a required module in your module hierarchy.

There are three “levels” of GDI usage that are reflected in the parts of this guide:

- The first level lets you use of the menus and dialogs without making any changes to them. At this level, you can use GDI as it stands without any modifications by you. See Using GDI Menus and Dialogs.
- The second level lets you use the G2 menu bar with modest changes to the menus and dialogs. See Modifying GDI Menus and Dialogs.
- The third level lets you reuse and customize the dialogs to suit your needs. See Modifying GDI Menus and Dialogs. For information on master dialogs, IDs, attributes, setup actions, and API calls, see Customizing the Dialogs, Customizing the Tree View Control Dialog, and Customizing the Print Dialogs.

Using Other G2 Utilities in GDI

To create menus and dialogs, GDI uses other G2 utilities. These other utilities are well-documented and you should refer to this documentation when you are customizing a GDI module.

GDI makes use of the following G2 utilities:

- G2 Menu System (GMS) provides the ability to create Windows style menus.
- G2 Foundation Resources (GFR) provide the ability to localize applications and control the order of module initialization through the use of gfr-startup-blocks.
- User Interface Library (UIL) controls reside on the dialog and provide procedures, rules, and user-menu-choices that create and manipulate dialogs.
- G2 Online Documentation (GOLD) provides the ability to view documentation online through the use of an HTML browser.

Displaying GDI Top-Level Workspaces

The *g2cuidev.kb* provides access to all GDI top-level workspaces. The g2uimenu G2 utility also requires the gms, gfr, g2uifile, goldui, and uil G2 utilities.

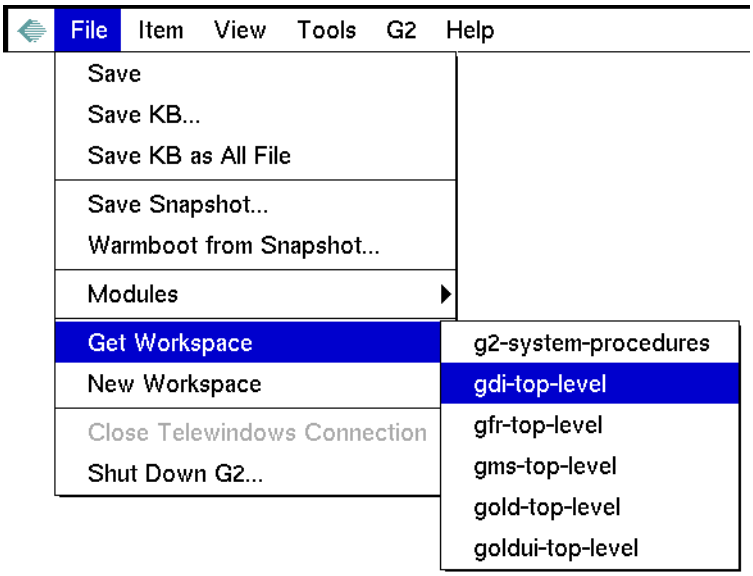
To navigate to the top-level workspace for GDI:

1 Choose Main Menu > Load KB and choose *g2cuidev.kb* from the *utils* subdirectory of the *kbs* directory under the *g2* directory.

2 Choose Main Menu > Start.

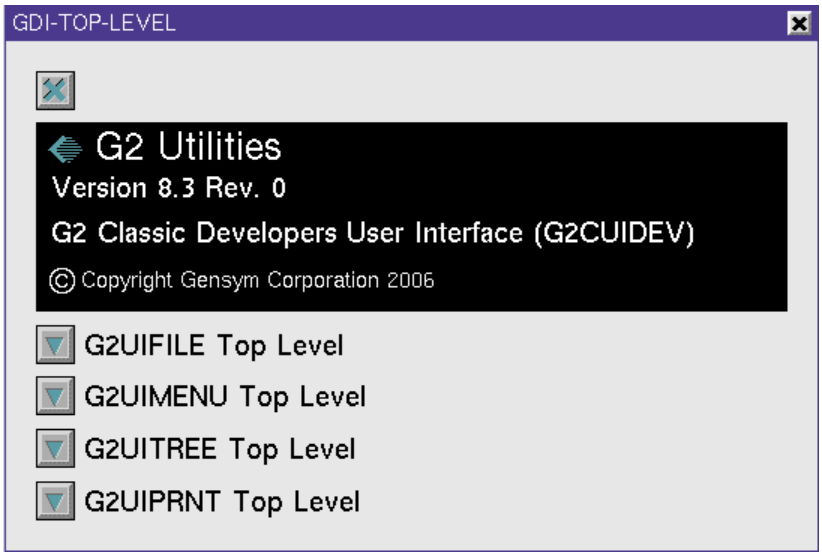
The GDI top-level menu bar appears.

Here is the menu bar with the Get Workspaces submenu visible:



3 Choose gdi-top-level from the File > Get Workspace menu.

The following figure shows the GDI top-level workspace:



The GDI top-level workspace contains links to all the GDI modules.

To navigate to the top-level workspace of one of the GDI modules:

➔ Click the navigation button to the left of the GDI module name.

For information on...	See...
Using the top-level menu bar	Using GDI Menus and Dialogs.
g2uifile	Modifying GDI Menus and Dialogs. Customizing the Dialogs.
g2uimenu	Modifying GDI Menus and Dialogs.
g2uitree	Customizing the Tree View Control Dialog.
g2uiprint	Customizing the Print Dialogs.

2

Using GDI Menus and Dialogs

Describes the menus in the G2 menu bar, the dialogs that support these menus, and the Print dialogs.

Introduction

This chapter describes the menus and dialogs available with GDI. They are ready to use and provide a Windows®-like graphical user interface. The G2 menu bar gives you the menu selections that you can find with classic G2. The dialogs have a Windows look and feel and conform to Windows standards that are familiar to PC users. The Print dialogs enable you to print G2 workspaces.

Using the G2 Menu Bar

The G2 menu bar provides menu selections for the G2 menus plus dialogs for reading and writing text files. You can also load GIF files.

To display the G2 menu bar:

1. Choose Main Menu > Load KB and choose *g2cuidev.kb* from the *utils* subdirectory of the *kbs* directory under the *g2* directory.
2. Choose Main Menu > Start.

The G2 menu bar appears at the top of the G2 window.

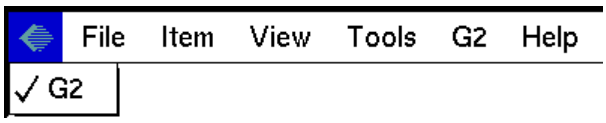
G2 developers can add or remove main menu headers and the menu items they contain. Using GFR, you can localize menus and dialogs. Using the GOLD user interface, you can access application-specific online help.

The G2 Logo Menu

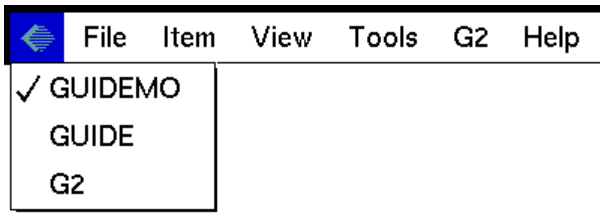
Using the G2 Logo menu, you can access other menu bars that are not currently active in the current G2 window. To enable users to switch between the different module-specific or application-specific menu bars, you can specify that every module have its own menu bar.

Note To enable your users to switch between different menu bars, Gensym recommends that you include a gms-switch-menu-bar-cascade-template in your menu. For more information, see Programming Hints.

The following figure shows the G2 menu as the only available menu. The checkmark indicates that the G2 menu bar is the active menu bar.



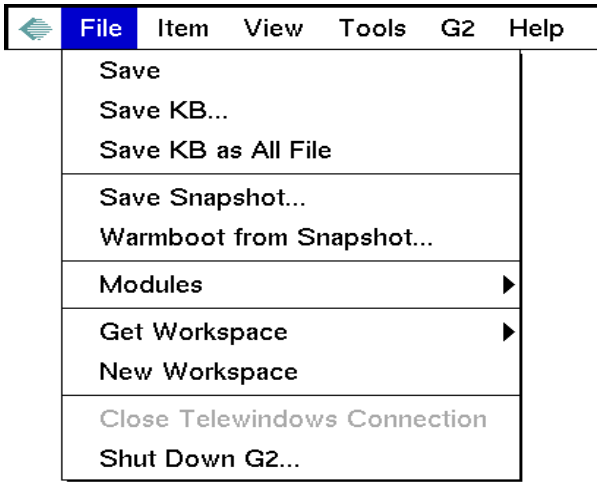
The following figure shows the G2 Logo menu when you have loaded guidemo, which automatically loads GUIDE. The guidemo module has its own menu bar, built from the G2 menus delivered with g2uimenu. GUIDE also has its own menu bar, also built from g2uimenu.



To switch among the three available menu bars, select GUIDEMO, GUIDE, or G2 from the G2 Logo menu.

The File Menu

The File menu contains menu options that act on KBs and modules. It also provides access to workspaces and G2 session termination actions. If you are logged in via a Telewindows connection, the Close Telewindows Connection menu choice is enabled. If you select either Close Telewindows Connection or Shut Down, G2 launches a confirmation dialog that gives you the option of saving first.



Saving KBs

The Save menu options relate to saving KBs.

Save

Saves only the top-level module in its current location. G2 automatically saves the KB without displaying a dialog. You can use this option to quickly save the top-level module.

Save KB

Launches the Save Module dialog (see Saving Modules). The Save Module Dialog enables you to specify the file name and the directory where you want to save the KB.

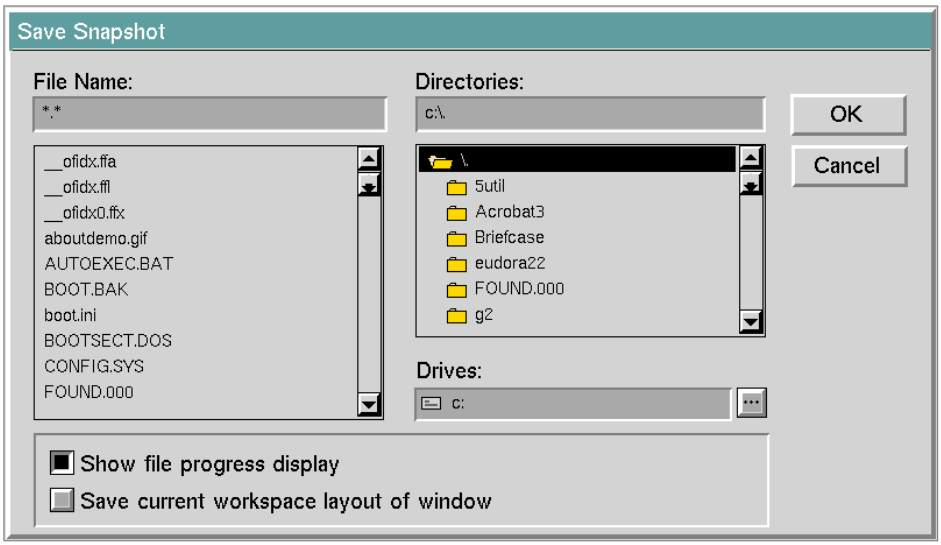
Save KB as All File

Saves all the modules into a file named *top_level_module_name*`-ALL.kb` into the current G2 default directory.

Managing Snapshots

You can save all of a KB's permanent and transient knowledge, including the real-time data associated with the G2 run-time environment, by saving the KB as a snapshot. You can reload the snapshot, using the warmboot procedure, which loads and automatically resumes running the current KB as if it had merely been paused.

The Save as Snapshot option saves all the KBs loaded with the run-state information. In the Snapshot dialog, shown in the following figure, you can specify the file name and the directory. You can include the word "snapshot" and the date in the file name.



Two options exist on the Snapshot dialog.

Show file progress display

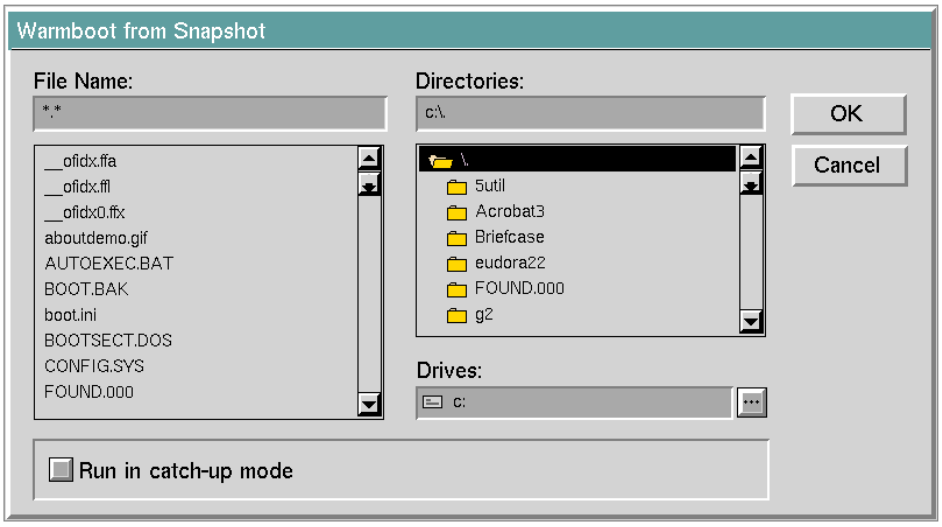
Displays the names of the files as they are saved.

Save current workspace layout of window

Saves the position of the workspaces in the current G2 window.

Warmbooting from a Snapshot

The Warmboot from Snapshot dialog enables you to locate a snapshot file and begin loading it:



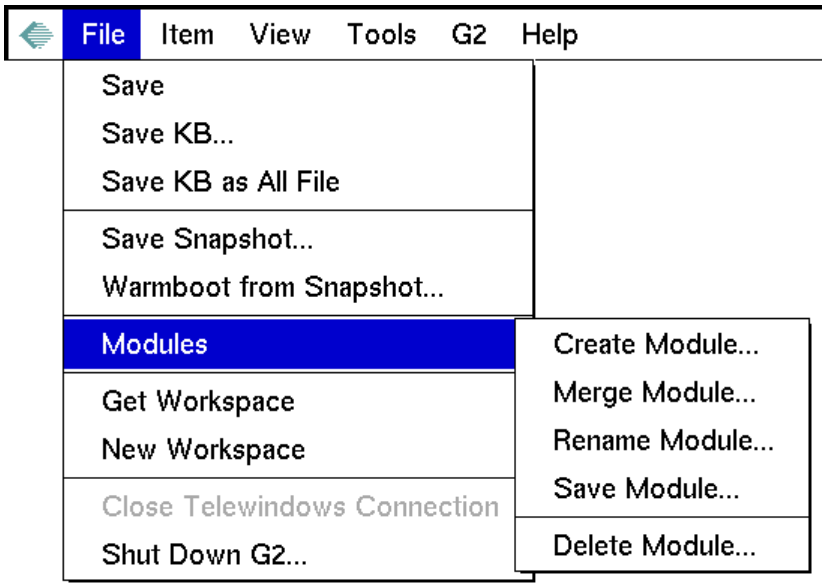
You can specify "catch-up" when warmbooting:

Run in catch-up mode

Warmboot a KB that is designed to run continuously so that its processing can "catch up" from the *current time* saved in the snapshot file to the *current real time*.

Managing Modules

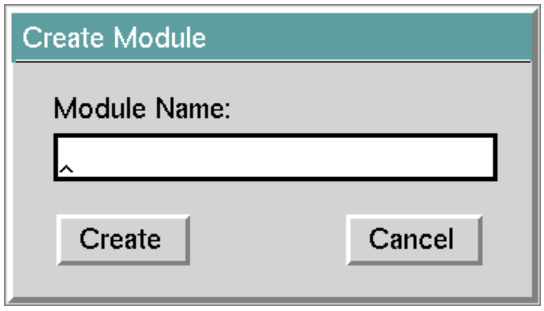
The Modules submenu, shown in the following figure, enables you to create, merge, rename, save, and delete modules. You can customize and localize these dialogs.



These dialogs are built upon the component file dialogs provided by g2uifile. These dialogs handle modules rather than files. For more information on customizing these dialogs, see Customizing the Dialogs.

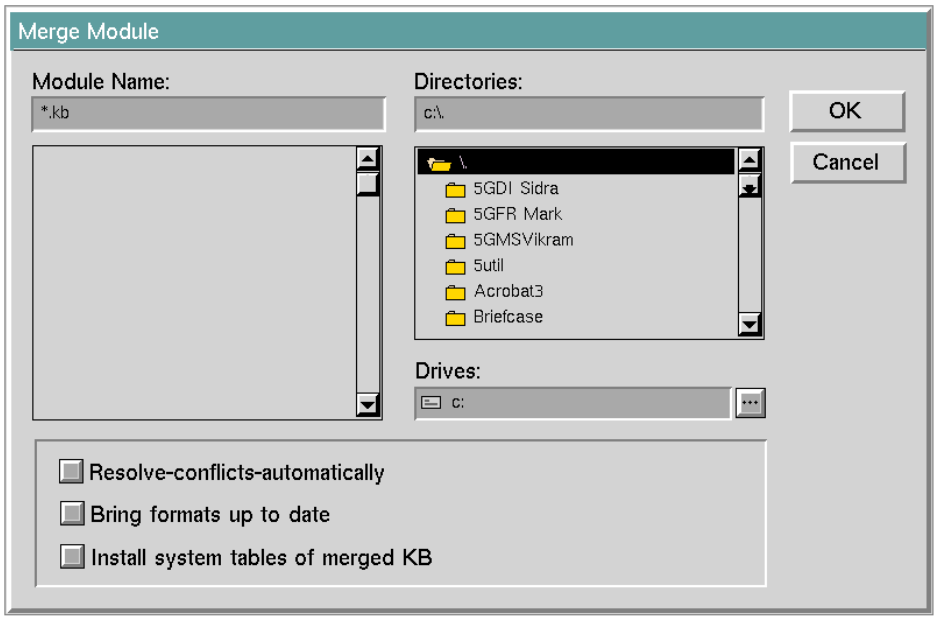
Creating Modules

The Create Module dialog prompts the user for a module name, validates the entry, then creates the module, using the system procedure, g2-create-module:



Merging Modules

The Merge Module dialog, shown in the following figure, enables you to select a module and merge it into the current module in the current G2 session.



There are three option buttons.

Resolve-conflicts-automatically

Checks for name conflicts among definitions contained in the merged KB and in any directly or indirectly required modules.

Bring formats up to date

Applies to all loaded items the formatting defaults specific to the current version of G2. This option can significantly change the appearance of your module.

Install system tables of merged KB

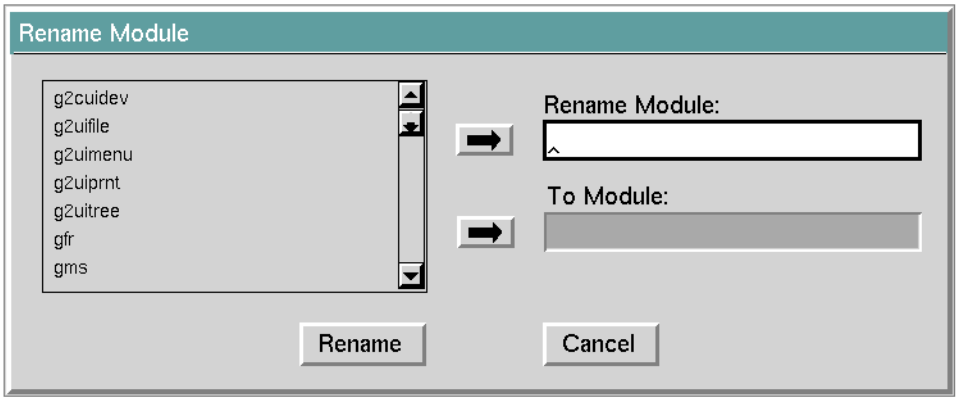
Installs the system tables of the merged KB, replacing the currently installed tables.

Renaming Modules

The Rename Module changes the module assignment of all top-level workspaces in the source module to reside in the destination module.

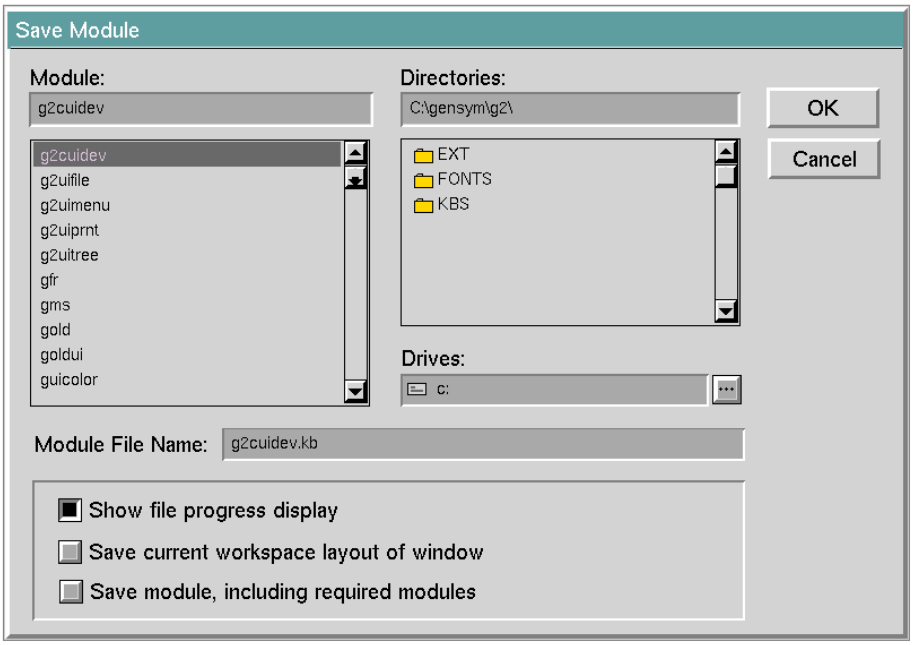
You can type the name of the destination module into the To Module edit-box. If the destination module does not exist, the rename operation confirms that you must create this module before continuing with the rename operation.

You can also select the module in the scroll-area to the left and then move the selected module over to the Rename Module edit-box by clicking on the arrow push-button directly to its left.



Saving Modules

The Save Module dialog enables you to specify a module name and directory for saving a module. Use the suffix *.kb* for each saved module name.



There are three options on the Save Module dialog.

Show file progress display

Displays the names of the files as they are saved.

Save current workspace layout of window

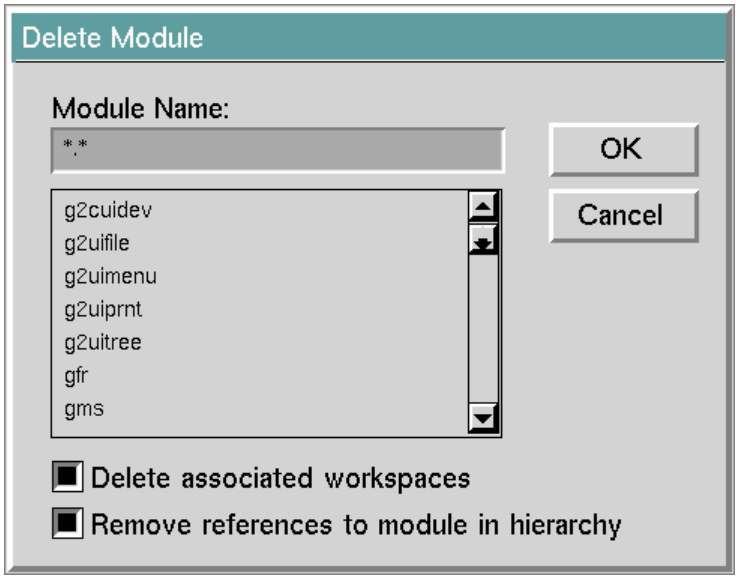
Saves the position of the workspaces in the current G2 window.

Save module, including required modules

Saves the entire module hierarchy into separate files.

Deleting Modules

The Delete Module dialog enables you to select a module for deletion from the current KB. You can also delete associated workspaces and remove references to the module from the current module hierarchy.



There are two options on the Delete Module dialog.

Delete associated workspaces

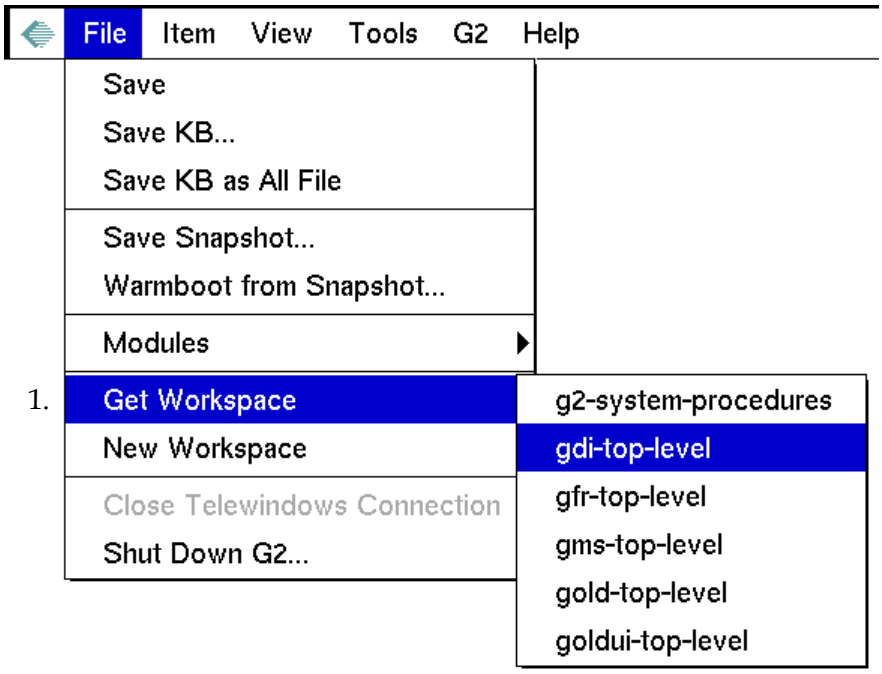
Deletes all KB workspaces assigned to the specified module.

Remove references to module in hierarchy

Deletes references to the specified module in the module-information tables.

Managing Workspaces

There are two menu options for displaying (1 – Get Workspace) and creating workspaces (2 – New Workspace), as shown in the following figure:

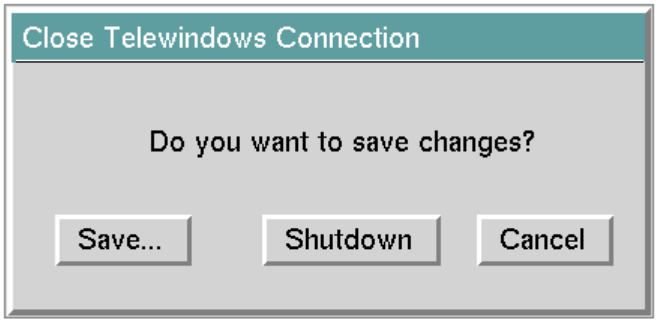


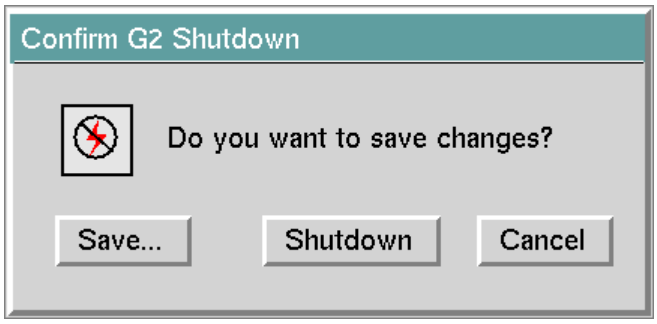
You can display or create a workspace:

- The Get Workspace option enables you to display a named workspace.
- The New Workspace option enables you to create a new workspace.

Ending a G2 Session

You can close a Telewindows connection or shutdown G2. The Shutdown dialog enables you to save your work before closing or shutting down, as shown in the following figures.



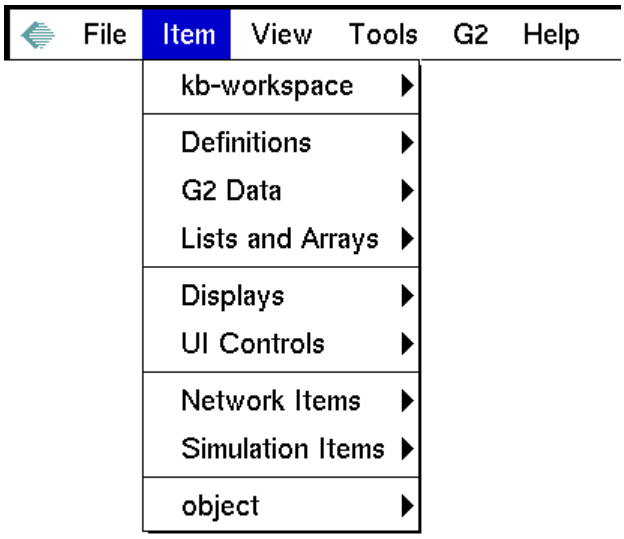


If you choose Save, G2 launches the Save KB dialog, described in Saving Modules.

If you choose Shutdown, the Telewindows connection simply closes or G2 shuts down without saving your work.

The Item Menu

The Item menu, shown in the following figure, contains all the new object options available on the workspace menu of G2:



The menu options are arranged in logical groups to:

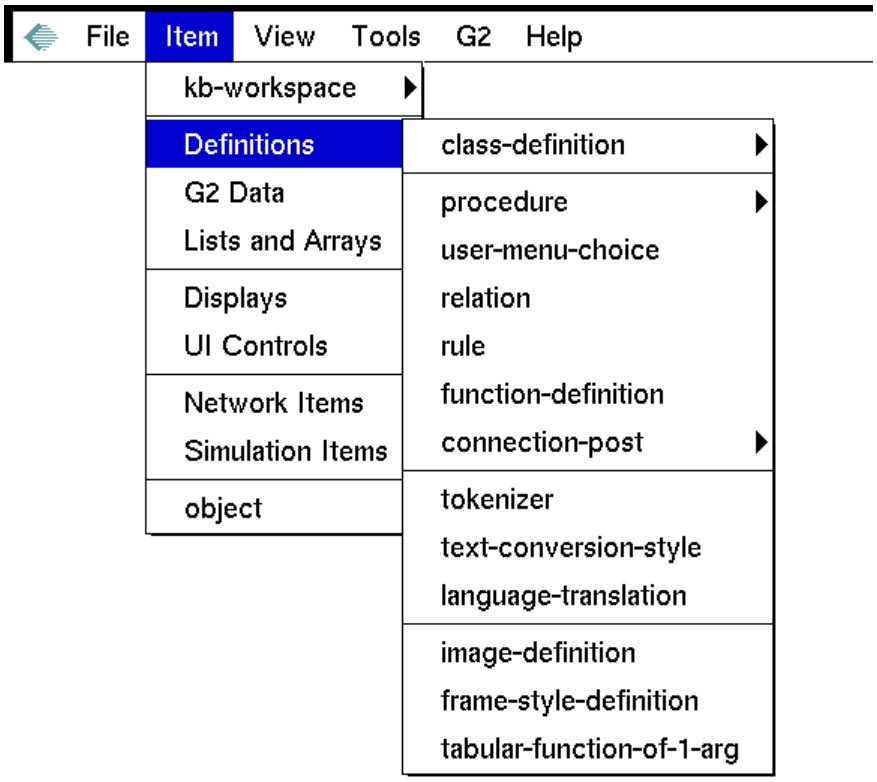
- Make the menu length more manageable.
- Provide a better road-map for creating new items.

kb-workspace

The kb-workspace menu choice enables you to create a KB workspace from the available workspace classes.

Definitions

The Definitions submenu contains an extensive list of definitional items you can create in G2. The following figure shows the Definitions submenu:



- Class-definition – Many classes reside in the class-definition submenu, for example, class, object, connection, and message. These classes are extensible. You can use these G2 system-defined classes to define new classes, each capable of having their own attributes and behavior.
- Procedure – The Procedure submenu displays the names of classes that are subclasses of Procedure. By default, these include: procedure, method, and method declaration. Procedures contain a predefined sequence of operations. A method is a specialized procedure that implements an operation for items of a particular class. To use a method, declare the name of the method in a method declaration.
- User-menu-choice – You can associate a user menu choice with a particular class so that it appears on the menu of the specified class and its subclasses.
- Relation – A relation definition creates a type of association between items of a first class (relation source) and items of a second class (relation object). To use relations in your KB, first you create a relation definition and then you use the

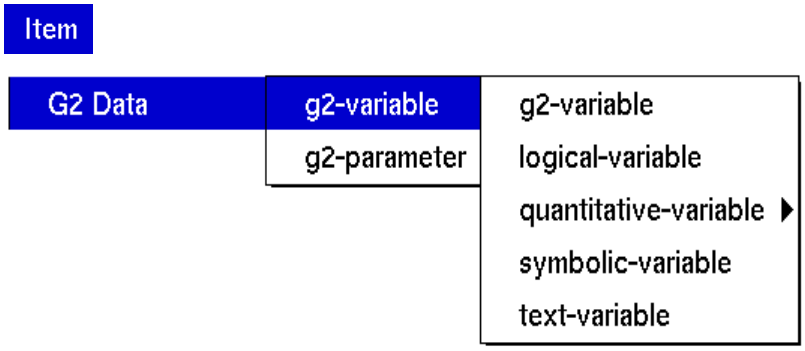
conclude action to create, remove, or replace relations based on their definition.

- Rule – A rule expresses a programmatic response to a set of conditions. A rule has two parts: antecedent ("if") and consequent ("then"). The antecedent tests for a condition. The consequent specifies the actions to take when the condition returns a value of true.
- Function-definition – You can create a function and refer to it by name or by arguments in procedures or expressions.
- Connection-post – You can connect objects across workspaces by creating a connection post. If you have defined any connection posts, they appear in the submenu for the connection-post choice.
- Tokenizer – You can create regular expressions that define a set of tokens, which are atomic units of a language. Tokens consist of a syntax description and a type name.
- Text-conversion-style – You can create items to represent default character sets for importing and exporting text styles such as the Gensym character set and Cyrillic text.
- Language-translation – You can use language translation to substitute one symbolic name for another. You can use an alternative symbol for any system- or user-defined menu choice.
- Image-definition – You can create an image definition to contain the name of an image in your KB and other attributes of a image.
- Frame-style-definition – You can create custom borders for workspaces by using a frame-style definition, which specifies the color of the border and its thickness in workspace units.
- Tabular-function-of-1-arg – Tabular functions begin with a single-argument, user-defined function. You can enter one or more arguments and values to derive, or cause G2 to compute, multiple comparative or associative values.

G2 Data

The G2 Data menu choice enables you to create G2 variables and parameters. It is a dynamic menu that can include other items, such as subclasses. You can use variables and parameters to represent changing data and as a means of maintaining a history of important values. Parameters are simple variables. Variables have additional functions.

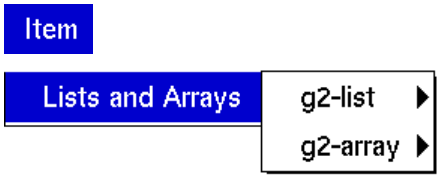
The types of variables and parameters are logical, quantitative, float, integer, symbolic, and text. They are represented as menu choices in the submenus for G2-Variable and G2-Parameter, as shown in the following figure for variables:



The Quantitative-Variable and Quantitative-Parameter submenus are also dynamic and contain the choices for quantitative, float, and integer variables or parameters.

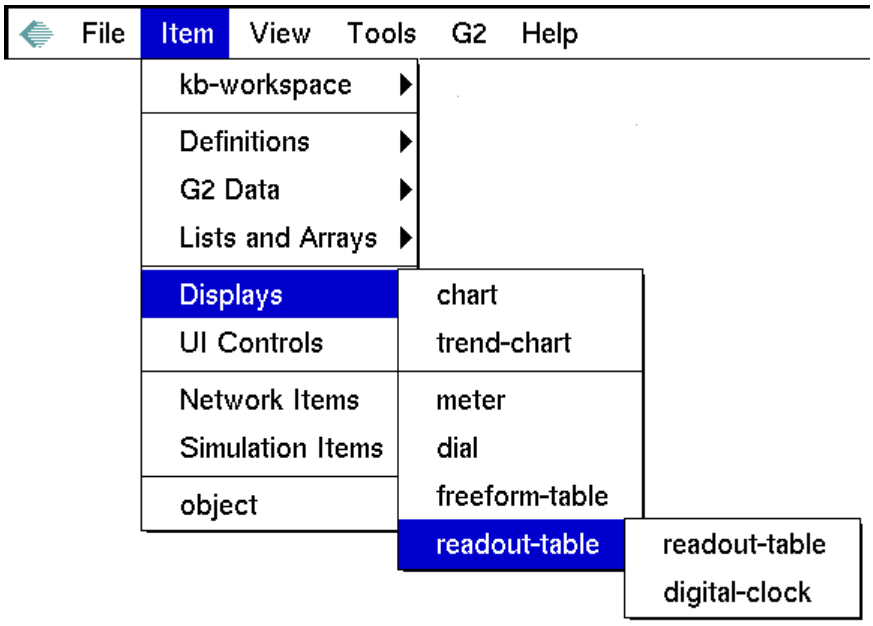
Lists and Arrays

You can use lists and arrays as placeholders for items and values in your KB. A list is a group of one or more elements and has no fixed length. An array contains a sequence of zero or more elements and has a fixed length, which you can change by editing the length attribute.



Displays

Displays enable you to monitor graphically variables and parameters, using the following displays and tables:

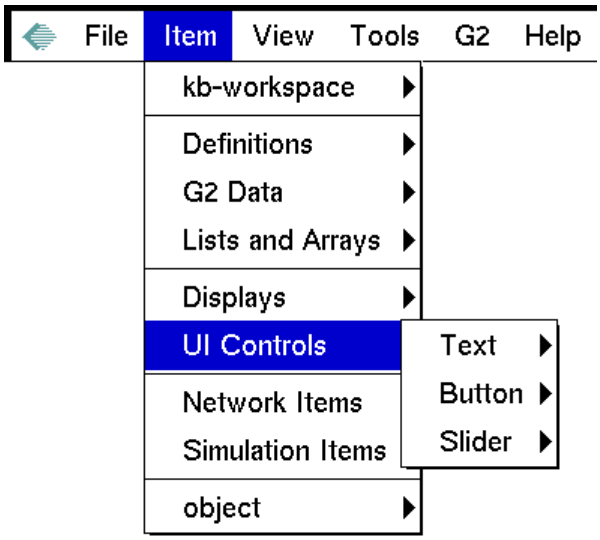


You can use these displays to show a:

- Chart – Plots numerical data in the form of one or more data series. A data series consists of one or more quantities, which must be elements of a quantity list or quantity array.
- Trend-chart – Plots time series or historical data over a designated period of time.
- Meter – Shows the value of any quantity variable, parameter, or expression as a pointer on the display and causes data-seeking, if the workspace upon which it resides is not hidden.
- Dial – Shows the value of any quantity variable, parameter, or expression as a pointer on the display and causes data-seeking, if the workspace upon which it resides is not hidden.
- Freeform-table – Is a tabular display similar to a spreadsheet.
- Readout-table – Is a rectangular, divided box that shows a label or an expression on the left-hand side and displays any quantity variable, parameter, or expression on the right.
- Digital clock – Is a special kind of readout table that displays the value of the expression the current time in the time stamp format.

UI Controls

The User Interface (UI) controls enable you to add text, buttons, and sliders to your application:



The Text submenu contains choices for:

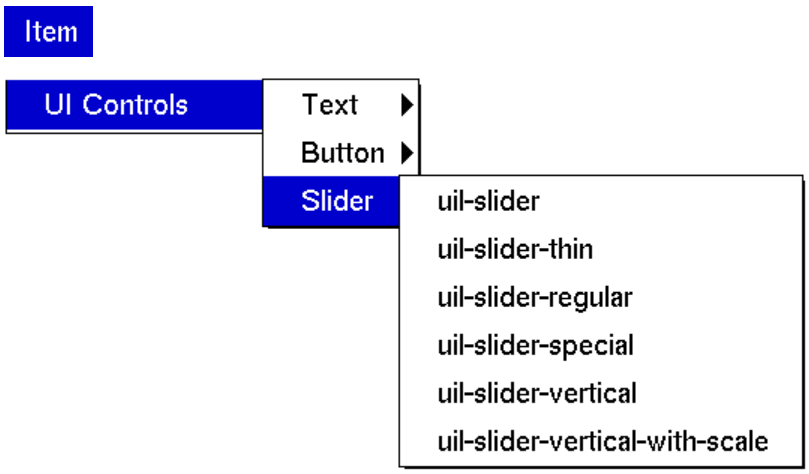
- Free-text – Labels various items in your KB and surrounds the text with a border.
- Borderless-free-text – Labels various items with no border.
- Text-inserter – Enables you to click on an object to insert text directly into the text editor. Four types of text inserters reside in the Text-inserter submenu:
  - Text-inserter – Inserts all of its text into the text editor.
  - Word-inserter – Inserts a single word at a time.
  - Character-inserter – Inserts the selected characters only.
  - Character-sequence-inserter – Inserts a sequence of characters that you select.
- Message – Provides information to the user. You can create a new instance of the message class.

The Button submenu contains choices for:

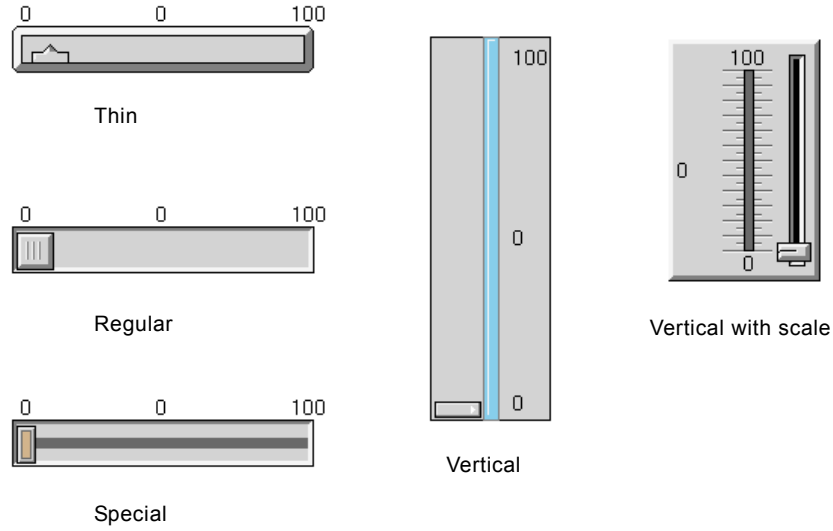
- Action-button – Causes G2 to perform one or more actions.
- Radio-button – Assigns one of a mutually exclusive set of symbols, numbers, logical values, or text values to a variable or parameter.

- Check-box – Assigns an on or off value to a symbolic, quantitative, logical, or text-value variable or parameter.
- Slider – Assigns numeric values to a variable or parameter as a pointer slides along a horizontal guide.
- Type-in box – Enables you to enter a symbolic, quantitative, logical, or text value for a variable or parameter.
- Uil-navigation-button – Enables you to create the following buttons:
  - Uil-goto-workspace-button – Navigates to a subworkspace of the workspace that contains this button or to another named workspace.
  - Uil-goto-superior-button – Navigates to the superior workspace of the workspace that contains this button.
  - Uil-goto-next-button – Navigates to a subworkspace of the workspace that contains this button or to another named workspace.
  - Uil-goto-previous-button – Navigates to a subworkspace of the workspace that contains this button or to another named workspace.
  - Uil-hide-button – Hides the workspace that contains this button.
  - Uil-help-button – Navigates to a workspace on which you can place help information.

The Slider submenu contains choices for types of sliders:

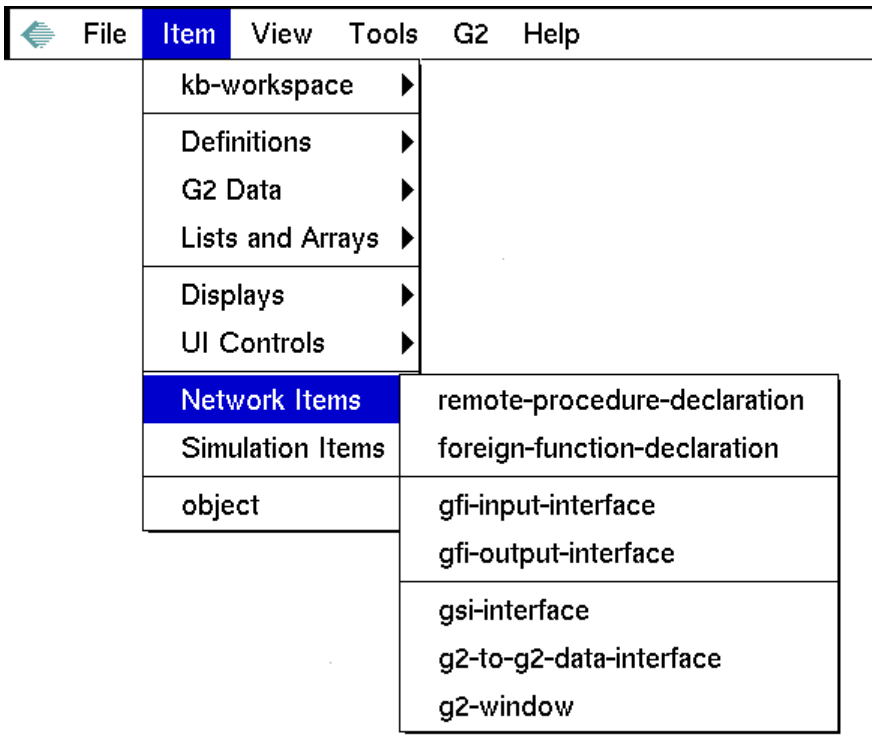


The choices for sliders appear in the following figure:



Network Items

These menu choices specify operations performed outside of G2 or that create connections to external devices:

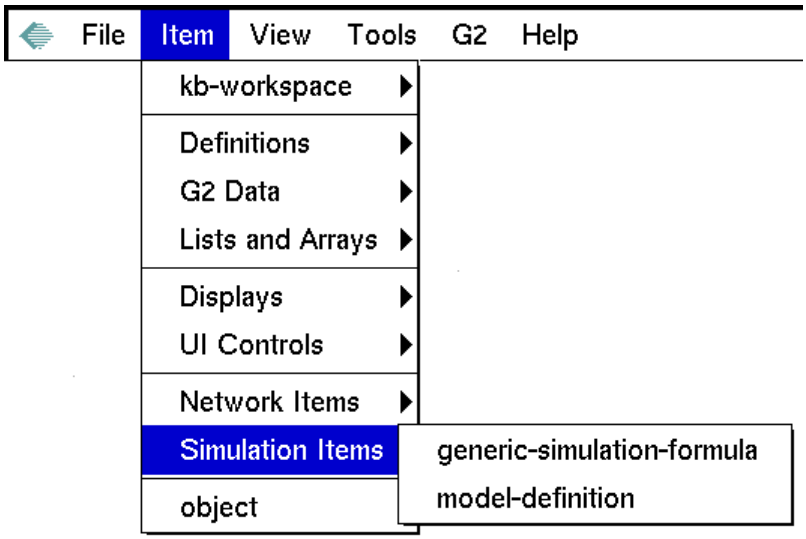


The Network Items submenu contains choices for:

- Remote-procedure-declaration – Creates a declaration for a Remote Procedure (RPC) call. You can use RPCs in any application that requires G2 to execute a procedure in another G2 across an ICP interface object.
- Foreign-function-declaration – Creates a declaration for a foreign function. A foreign function is a function written in C or C++ code that a KB can access as if it were a local function.
- Gfi-input-interface – Creates an object so that you can use the G2 File Interface (GFI) to read from a data file.
- Gfi-output-interface – Creates an object so that you can use the G2 File Interface (GFI) to write to a data file.
- Gsi-interface – Creates a G2 Gateway object. Using these objects, you can send values to, and receive values from, an external G2 Gateway process, using the TCP/IP communications protocol.
- G2-to-g2-data-interface – Creates a G2 data interface object. The G2-to-G2 interface is the general facility that permits one G2 to communicate with another.
- G2-window – Creates a G2 window object. G2 supports the practice of associating a g2-window item, created and maintained by your application, with the window that a Telewindows process opens after connecting to a G2 process.

Simulation

You can create a generic simulation formula or a model definition, using the Simulation Items menu option.



The Simulation Items submenu contains choices for:

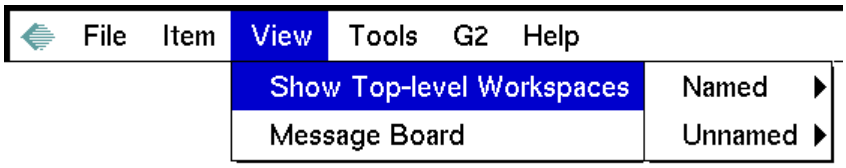
- Generic-simulation-formula – Creates an icon for a generic simulation formula, which you can specify for a class of variables by creating a generic simulation definition.
- Model-definition – Creates an icon for a model definition. A model is made up of its variables, which are either stand-alone variables or attributes.

Object

The Object submenu contains choices for various classes that record information on the operation of G2, on classes that relate to GUIDE, GFR, and GOLD, and on objects created by the user. You can use this submenu to create instances of non-G2 classes where Object is the direct superior class.

The View Menu

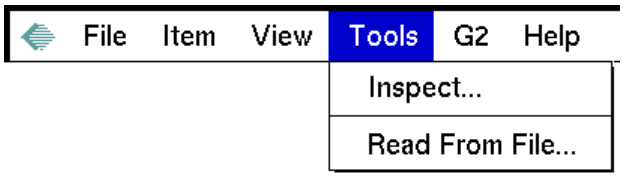
The View menu contains two menu choices: Show Top-level Workspaces (named and unnamed) and Message Board. If no Message Board is active in the G2 session, this menu choice is disabled.



Note If an application provides palettes, toolbars, or status bars, they should be available on the View menu. For example, when you load GUIDE/UIL, GUIDE adds a Palettes submenu to the View menu.

The Tools Menu

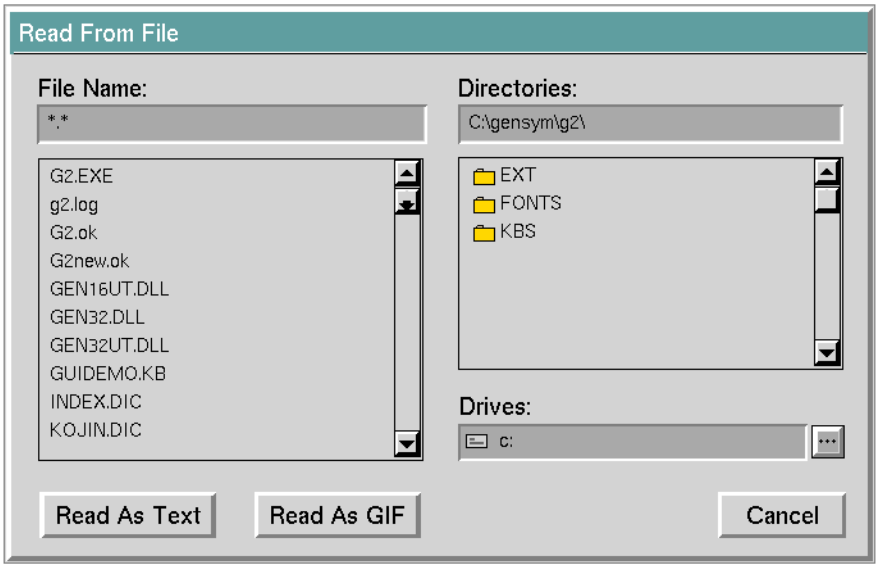
The Tools menu contains menu choices for Inspect and for reading from a file. To search a KB for items based on their type, class, attributes, and location, launch Inspect. To import text and GIF images into G2, choose Read From File to display the Read From File dialog.



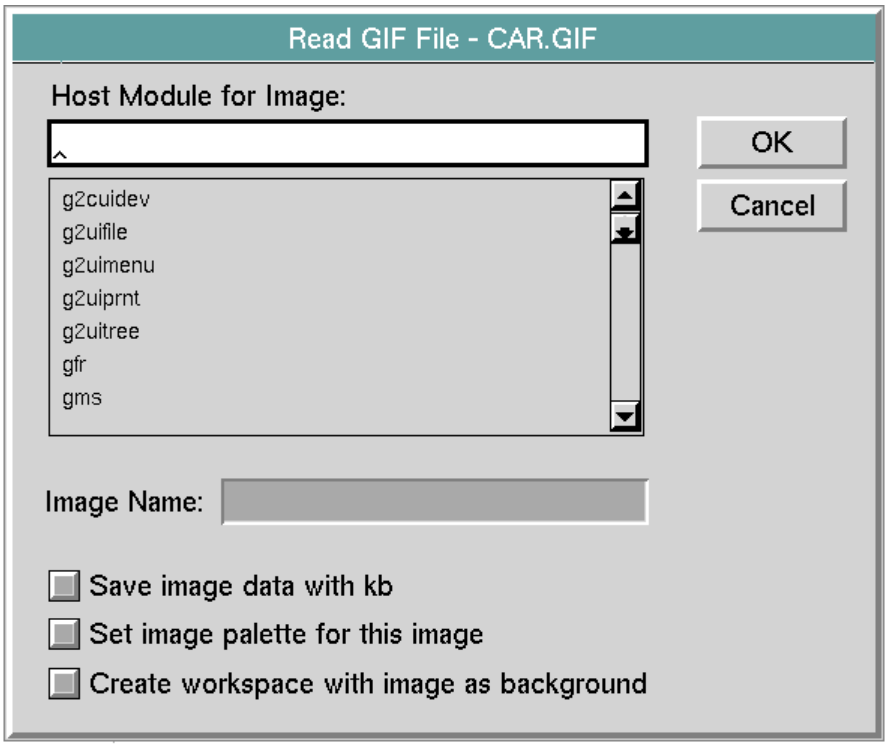
Reading from a File

To copy the text in a specified file into a free-text object, select a text file from the file scroll-area in the Read From File dialog and click the Read As Text button.

To copy a GIF file, select a GIF file from the file scroll-area and click the Read As GIF button:



If you select the Read As GIF button, G2 launches the Read GIF File dialog that provides options for an image:



The Read GIF File dialog performs these actions:

- Creates an image-definition with the specified image name.
- Configures the file-name-of-image attribute of the image definition to reference the specified GIF file.
- Creates a workspace residing in the specified host module and transfers the image-definition to it.

There are three buttons in the Read GIF File dialog

Save image data with kb

Sets the save-image-data-with-kb attribute of the image definition from false to true.

Set image palette for this image

Sets the image-palette attribute of the Drawing Parameters system-table from its default value of standard colors, to custom colors for *image-name*.

Create workspace with image as background

Sets the background-images attribute of the workspace containing the image definition to *image-name* at (0, 0).

You can read a GIF file, such as the one in following figure, an image of a car:

Writing to a File

You can access a Write to File dialog from textual-type items in G2. Clicking the mouse on a textual-type item and selecting the Write to File menu option launches a Save File dialog that enables you to write the text of the item to a named file.

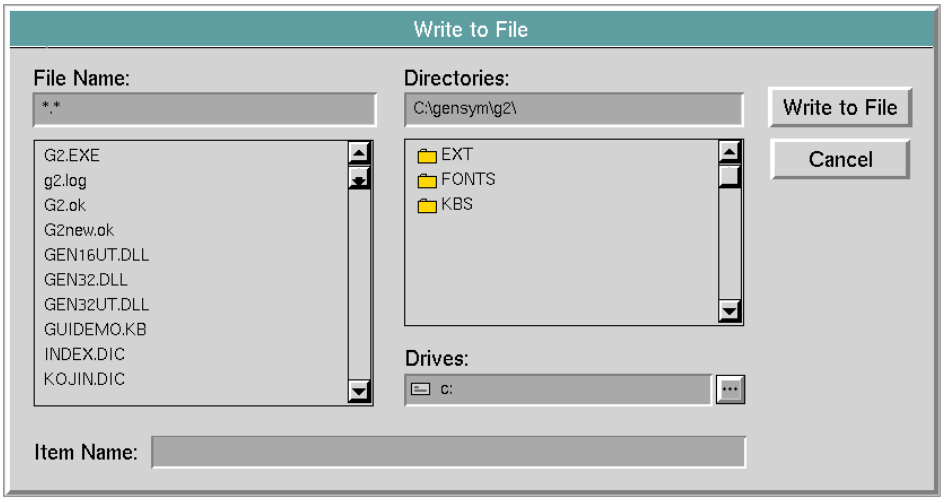
Note You can disable the user menu choice for the Write to File dialog by setting the logical parameter, g2uifile-user-menu-choices-active, to false, or by deselecting the Enable User Menu Choice for g2uifile toggle button on the g2uifile top-level workspace. See Customizing the Dialogs.

The textual-type items that support the Write to File dialog are:

- free-text
- borderless-free-text
- message

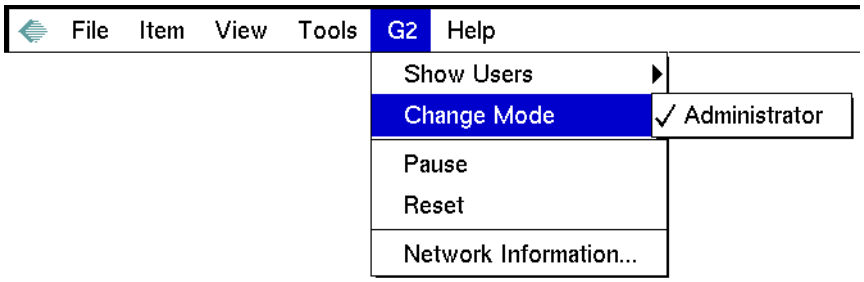
- statement
- text-inserter
- word-inserter
- character-inserter
- character-sequence-inserter
- language-translation
- generic-formula
- generic-simulation-formula
- rule
- function-definition
- procedure
- method
- units-of-measure-declaration
- remote-procedure-declaration
- foreign-function-declaration

The following figure shows the Write to File dialog. The name of the selected textual-type item appears in the edit-box labeled Item Name, unless the selected item has no name.



The G2 Menu

The G2 menu contains G2 session-specific items. The submenu Show Users displays a list of the users currently connected to the G2 session. It shows each user's node, its port number, and the user's mode.



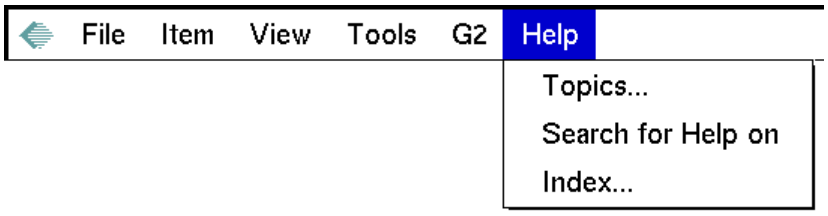
In the Change Mode submenu, developers can add application specific modes for G2. By default, Administrator is the only mode on this sub-menu.

Pause, Reset and Network Information call the corresponding system procedure. Selecting Pause pauses G2 and thus makes the menus inactive. Select Resume or Start from the G2 Classic main menu to start or restart a G2 session.

The Help Menu

The Help menu provides access to online help via the GOLD G2 utility. The submenus Topics and Index display the table of contents and index for the online version of the *G2 Developer's Interface User's Guide*. Search for Help on displays the Help dialog, from which you can navigate to the table of contents and index for any online book, find topics in an online book, and set up online help.

Note By pressing the F1 key over a menu choice, you can also launch online help for that menu choice.



For more information on online help and GOLD, see the *G2 OnLine Documentation User's Guide*.

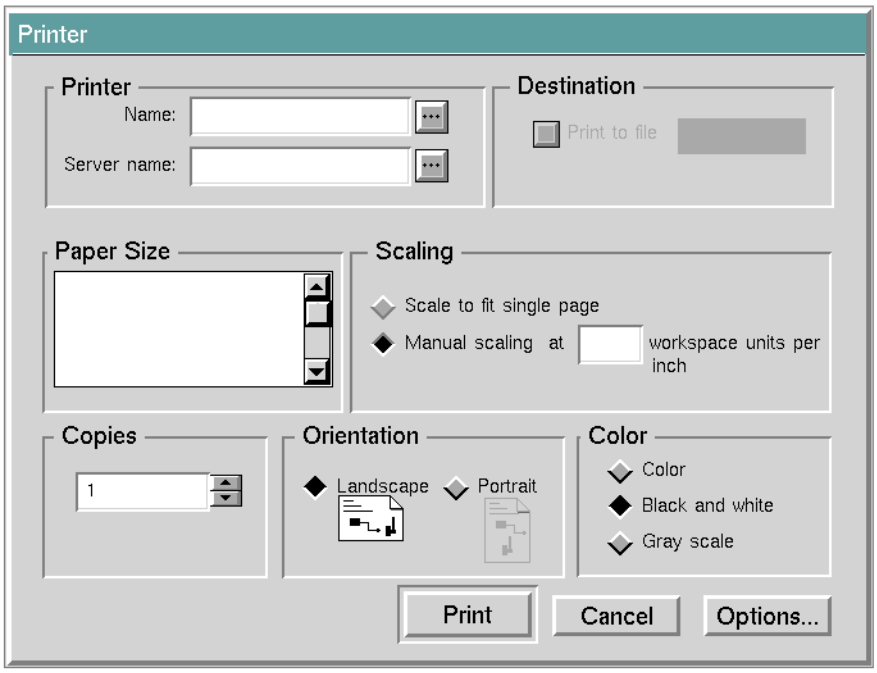
Printing G2 Workspaces

The Print dialogs provide a complete user interface for presenting and controlling printing options when printing G2 workspaces. The Print dialogs enable you to specify a print job, select an available printer, set margins, and otherwise define your print request.

To print a G2 workspace:

➔ Choose Print from the popup menu for a specified workspace.

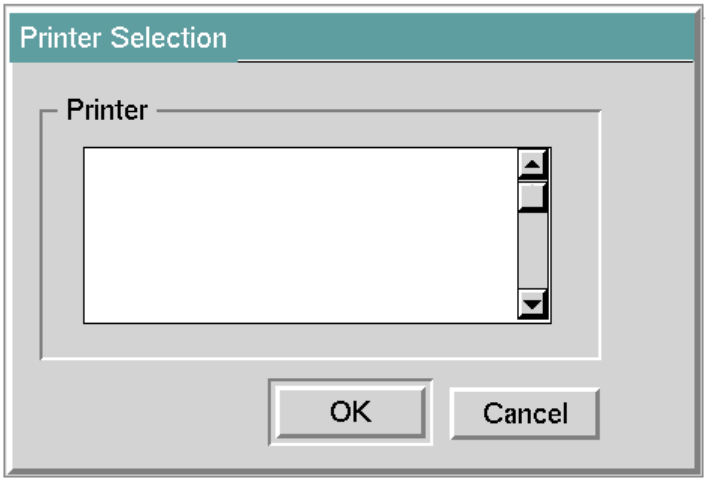
When you choose Print, G2 displays the main Print dialog:



The main Print dialog has controls that determine the values stored in the Printer Setup system table. The rest of this section describes these controls.

Specifying a Printer

The buttons to the right of both the printer name and the server name enable you to specify the printer and server from the available printers at your site. Clicking on one of these buttons displays the Printer Selection dialog:



Enter the name of the printer at your site under Printer.

If your operating system is Windows, you must also enter the name of a site print server. In UNIX configurations, the server entry field is disabled.

The scroll area lists the names of site printers and servers stored in the files *printers.txt* and *servers.txt*. The names of these files are determined by values of the text parameters g2ui-print-printer-file-filename and g2ui-print-server-file-filename. Selecting any entry from the list enters the selection into the edit box on the main Print dialog.

The files *printers.txt* and *servers.txt* are both optional. If the print utility locates them in the working directory, G2 places their contents into these scroll areas at startup. The format of these files is: one line for each name, whether printer or server. For example:

```
mailroom
engineering
phaser3
```

The location of these files is determined by the value of the text parameter g2ui-print-printer-file-pathname.

To point to a new location for the printer and server files:

➔ Conclude a value into g2ui-print-printer-file-pathname.

For example:

```
conclude that g2ui-print-printer-file-pathname = “c:\”
```

Overriding this parameter must be done at initialization. The best way to do this is through an initially rule or a GFR startup object. For information on GFR startup objects, see the *G2 Foundation Resources User's Guide*.

Specifying a Print Request

The options on the main Print dialog enable you to specify a print request.

To print to a file:

1 Click to Print to File button under Destination.

2 Specify a pathname for the printer file in the edit box to the right of the button.

If you select this button, the print job is directed to the indicated file. If you do not select this button, the print job is spooled to indicated printer and server.

To specify paper size:

➔ Click the paper size from the available sizes under Paper Size.

The list of paper sizes are those supported by G2. Refer to the *G2 Reference Manual* for details.

To specify scaling:

➔ Choose either scaling a workspace to fit the page or set a manual scaling factor under Scaling.

Scaling controls the scale (in workspace units) that the printer uses to print the workspace. You can set scaling to maximize the scale of the workspace to fit on a single page or set it manually when you require precise scaling.

When you select manual scaling, type a scaling factor in workspace units in the Workspace Units per Inch edit box. You control the choice of units (inch or centimeter) from the Print Options dialog.

To specify the number of copies:

➔ Type in the number of copies from 1 to 99 under Copies, or use the up and down arrows to the right of the edit box.

To specify page orientation:

➔ Choose either the landscape or portrait buttons under Orientation.

To specify color settings:

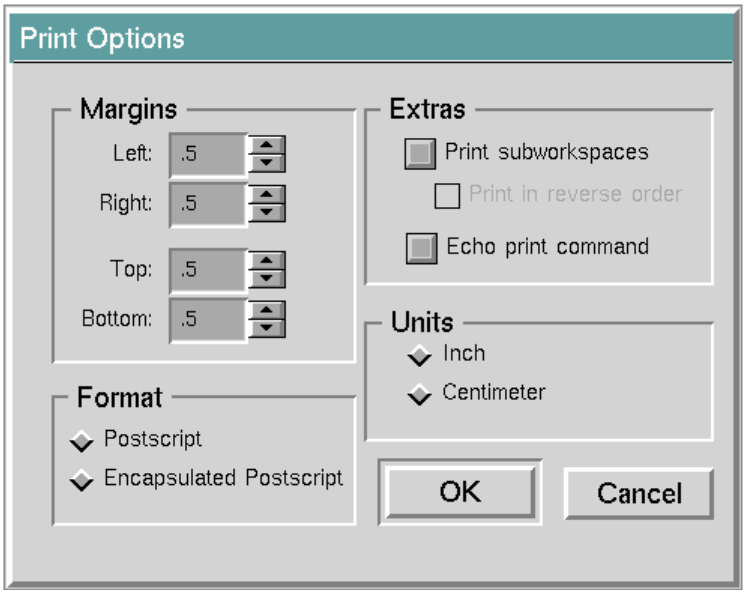
➔ Click on Color, Black and White, or Gray Scale.

The color control allows full color, black and white conversion, or gray scale conversion. G2 maps certain colors to black and others to white. Gray scale colors are approximated by G2.

To print the workspace according to the settings you specified, click the print button. To cancel the print operation, click Cancel. To specify additional print options, click the Options button.

Specifying Printing Options

Clicking the Options button launches the Print Options dialog, which enables you to further specify your print request:



To set page margins:

➔ Type in the left, right, top, and bottom margins, or use the up and down arrows to the right of the edit boxes.

The units are either inches or centimeters that you specify in the Units option.

To print subworkspaces:

➔ Click the Print Subworkspaces button under Extras.

Selecting Print subworkspaces prints all of the workspaces below the selected workspace in the workspace hierarchy.

The printing order is depth-first ordering unless you select the Reverse Print order button, in which case the printing is still depth-first order, but in reverse. This option enables you to work with printers that print front-to-back or back-to-front.

To display a dialog showing the exact print command:

➔ Click the Echo Print button under Extras.

Selecting the Echo Print button presents a dialog that, during the print job, shows the exact command used to send the print job to the printer. This option is intended as a debugging aid and should not be necessary in routine circumstances.

To specify output format:

➔ Click either the PostScript or Encapsulated PostScript buttons under Format.

To set the units for margins and workspace scaling:

➔ Click either the Inch or Centimeter button.

3

Modifying GDI Menus and Dialogs

Describes the some techniques, ranging from simple to complex, for modifying the menus and dialogs to suit your needs.

Introduction

This chapter describes some of the techniques for modifying the menus in the G2 menu bar and the dialogs called by these menus. The modifications range from changing the background color of a dialog to changing the task carried out by a dialog.

Some of the modifications require only small changes to implement and essentially do not alter the way the menus and dialogs work. Some modifications require many changes to implement and do alter the way the menus and dialogs work. This chapter presents the simplest modifications first and proceeds to other modifications of increasing complexity.

The G2 menu bar uses the G2 Foundation Resources (GFR) and G2 Menu System (GMS):

- Using GFR, you can specify the default language of the menus, as well as make other languages available. You can switch between languages by changing the default language.
- Using GMS, you can add or remove menu options on the menu bar and you can specify what the menu options do.

Note For more information on GFR and GMS, see the *G2 Foundation Resources User's Guide* and the *G2 Menu System User's Guide*.

Modifying Dialog Appearance

You can modify the dialog colors and text by cloning preferences and settings objects from the g2uifile palette to your application and by changing the attributes of your cloned copies. Palettes are a common user interface device in G2 applications. A palette is a workspace from which users can clone items to use in a KB they are developing.

As a way of creating instances, palettes have several advantages over the New Object command on the KB Workspace menu:

- Objects on palettes can be initialized in any way required by the application.

  When you use the New Object menu, the object you obtain might not be initialized properly. For example, an object created in this manner cannot have a subworkspace.

- Palettes immediately present the objects that the user is meant to see.

  Use of the New Object menu could require complex navigation through many menu levels to find the desired class. The names of the classes that must be traversed might be unfamiliar to the user. Additionally, navigation through the class hierarchy might require traversal of private classes, which should be hidden from the end user.

- Palettes allow users to identify visually the object they want to create, even if they cannot remember the class name.
- Users recognize classes by their icons more easily than by their class name.

Because of these advantages, palettes have become a standard way to present items to the user, and they are found in many KB modules.

Each of the GDI utilities have palettes. The File Dialog, Printing, and Tree View Control utilities offer icons for configuring user preferences and adjusting dialog settings.

This section discusses the palette for g2uifile. For a discussion of the palette for printing, see Customizing the Print Dialogs. For a discussion of the palette for the tree-view control, see Customizing the Tree View Control Dialog.

Displaying the G2UIFILE Palette

The g2uifile palette enables you to clone the preferences or settings icons and place them in your module. The palette is the top-level workspace for g2uifile.

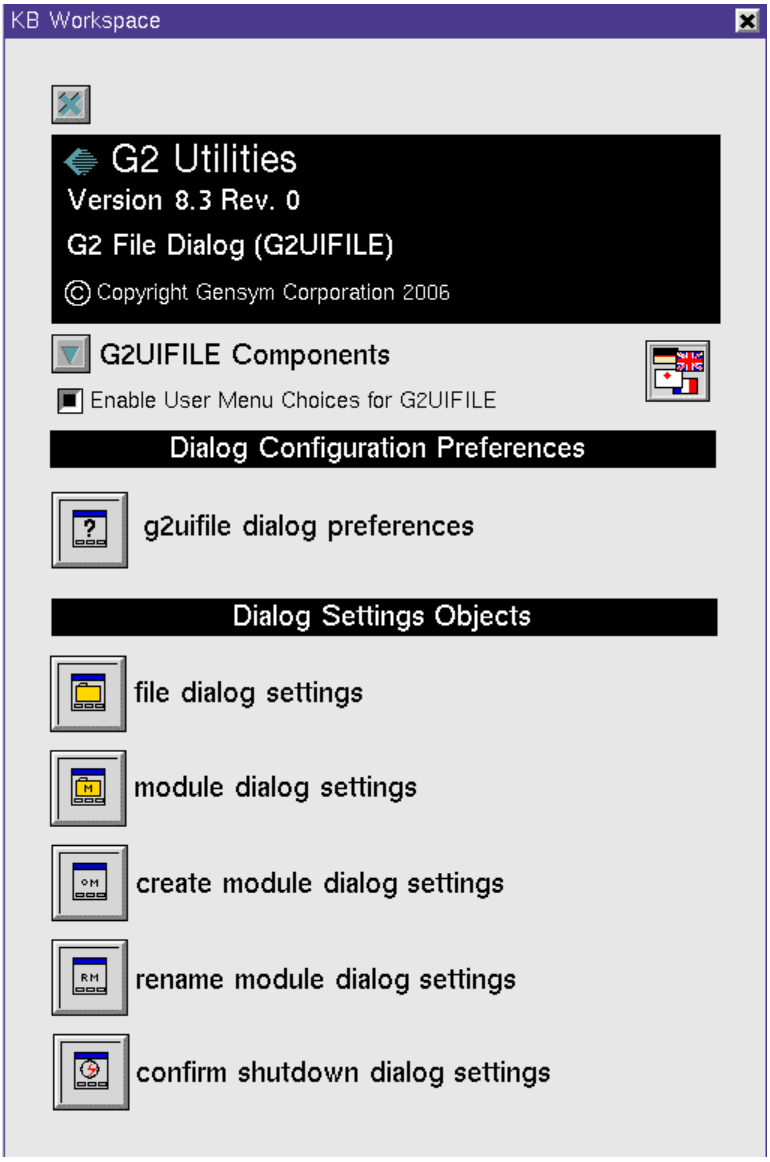
To display the G2UIFILE palette:

1. Display the GDI top-level workspace.

   See Displaying GDI Top-Level Workspaces.

2. Click the G2UIFILE Top Level button.

The following figure shows the top level workspace for the g2uifile module. The G2UIFILE Components button enables you to navigate to the dialog master dialogs, their APIs, and dialog examples. A GFR text resource group icon named g2uifile-text-resources specifies the language resources for the module. The icons for dialog preferences and settings can be cloned when you are customizing your own versions of these dialogs.



Note To enable the write to file user menu choice, make sure that the Enable User Menu Choice for G2UIFILE toggle button is selected.

Modifying Dialog Colors

The attributes of the g2uifile-dialogs-module-settings object (labeled g2uifile dialog preferences) specify the GUIDE/UIL classes for the dialog title, text, and background colors for all the dialogs in the g2uifile module. You can change the default configurations and develop your own configuration classes. The configuration classes at the highest level of your module hierarchy determine the attributes for dialog title, text, and background colors.

You can change these colors to suit your needs by cloning the uil-dialog-configuration class and copying it to your application. Make all your changes to your copy of the configuration class.

If you create a new configuration class, you must restart G2 so that your changes can be recognized.

To navigate to the default UIL dialog configuration objects:

➔ Use the go to command in Inspect followed by the name of a class, for example, uil-dialog-configuration.

The attributes of the g2uifile-dialogs-module-settings object are:

Attribute	Description
dialog-title-configuration	The color of the title bar and other text on the dialog.
Allowable values:	Any class name
Default value:	uil-dialog-title-configuration
dialog-configuration	The background and foreground colors of the dialog.
Allowable values:	Any class name
Default value:	uil-dialog-configuration

Modifying Dialog Title Colors

The attributes of the uil-dialog-title-configuration class specify the colors of the title of all dialogs.

Attribute	Description
uil-text-background color	The background color of the dialog title text.
Allowable values:	Any color
Default value:	cadet-blue
uil-text-border-color	The border color of the dialog title.
Allowable values:	Any color
Default value:	cadet-blue
uil-text-color	The color of the dialog title text.
Allowable values:	Any color
Default value:	white
uil-edit-box-diabled-text-color	The color of the text of an edit box that is disabled.
Allowable values:	Any color
Default value:	light-gray
uil-edit-box-diabled-background-color	The background color of an edit box that is disabled.
Allowable values:	Any color
Default value:	cadet-blue

Attribute	Description
uil-edit-box-diabled-border-color	The border color of an edit box that is disabled.
Allowable values:	Any color
Default value:	cadet-blue

Modifying Dialog Background and Foreground Colors

The attributes of the uil-dialog-configuration class specify the background and foreground colors of all dialogs.

Attribute	Description
uil-dialog-background color	The background color of the dialog.
Allowable values:	Any color
Default value:	light-gray
uil-dialog-foreground-color	The foreground color of the dialog.
Allowable values:	Any color
Default value:	black

Modifying Dialog Text

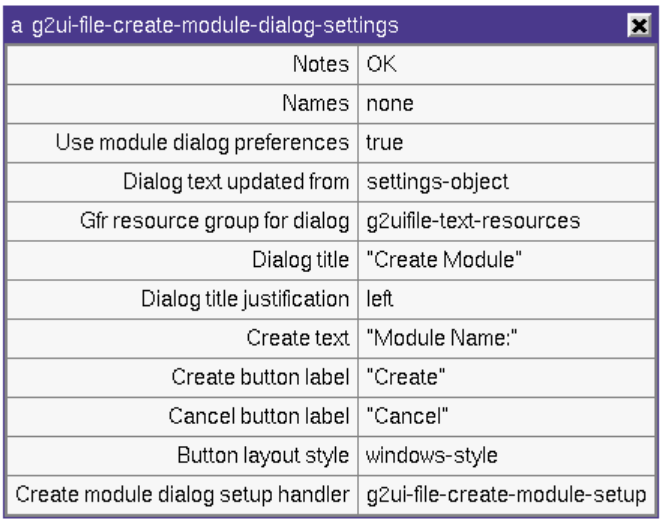
You can clone the settings objects for the file, module, create module, rename module, and shutdown dialogs and place them in your module. The module settings object works for merging, saving, and deleting. From the cloned settings objects, you can change the attributes for menu text and menu handlers.

You can use attributes to change GFR localized text resources, dialog title, text, button labels, and menu handlers.

If you create a new settings object, you must restart G2 so that your changes can be recognized.

The settings object at the highest level of your module hierarchy determines the attributes for all the dialogs in your module hierarchy.

For example, the following figure shows the attributes table for the g2ui-file-create-module-settings object, which applies to the Create Module dialog:



a g2ui-file-create-module-dialog-settings

Notes	OK
Names	none
Use module dialog preferences	true
Dialog text updated from	settings-object
Gfr resource group for dialog	g2uifile-text-resources
Dialog title	"Create Module"
Dialog title justification	left
Create text	"Module Name:"
Create button label	"Create"
Cancel button label	"Cancel"
Button layout style	windows-style
Create module dialog setup handler	g2ui-file-create-module-setup

For a description of the attributes of the various dialogs, see:

- Customizing the Dialogs.
- Customizing the Print Dialogs.
- Customizing the Tree View Control Dialog.

Modifying G2 Menu Bar Appearance

Most GMS behavior is defined in menu specifications. Some higher-level GMS properties do not apply to any particular menu but to all GMS menus. These properties are "global settings." The global settings include actions at menu startup and localized text. All GMS global settings are specified by attributes of the gms-global-settings object.

User preferences control the way GMS menus appear to the user. They are similar in many ways to global settings, and in some cases can override them. When several users log in to the same G2, GMS customizes the menus that each user sees as specified by the preferences applicable to that user.

Menu specifications and global settings establish GMS properties that are the same for all users. You can customize some GMS properties to be different for different users. GMS properties that you can customize on a per-user basis are called user preferences.

You can modify global settings and user preferences by creating your own objects and then customizing them.

To customize global settings and user preferences:

1. Display the gms-top-level workspace.
2. Create new global settings and user preferences objects by cloning them from the GMS palette and place them in your module.
3. Give these objects a higher priority than other settings and preferences objects.

Using Global Settings

The gms-global-settings object contains an attribute that enables you to specify whether or not to use GFR for menu text translation. Setting the gms-use-translations attribute to true invokes GMS to locate menu text in gfr-local-text-resources.

The gms-global-settings object also contains a list of inactive menu items. The gms-inactive-keys attribute comprises a list of menu keys for those items that should not be displayed when the menu is managed. Menu item keys can be dynamically added to and removed from this list.

Attribute	Description
gms-priority	Sets the priority that GMS uses to select which Global Settings object to put into effect.
Allowable values:	Any nonnegative integer
Default value:	1
gms-initiate-automatically	Specifies whether GMS creates and displays menus automatically when G2 starts.
Allowable values:	true, false
Default value:	true
gms-keep-compiled-resource	Specifies whether GMS keeps compiled resources.
Allowable values:	true, false
Default value:	false

Attribute	Description
gms-use-translations	Specifies whether GMS uses GFR to internationalize menus.
Allowable values:	true, false
Default value:	true
gms-default-language	When gms-use-translations is true, specifies the language to translate to when the user has specified no other.
Allowable values:	Any G2 language for which GFR translations are defined.
Default value:	none
gms-maximum-entries-count	The largest number of menu entries that the GMS compiler should be prepared to compile.
Allowable values:	Any positive integer
Default value:	350
gms-check-for-global-consistency	Whether GMS should syntax-check menu specifications before compiling them.
Allowable values:	true, false
Default value:	true
gms-inactive-keys	Specifies a list of menu keys for those items that should not be displayed when the menu is managed.
Allowable values:	Any value array
Default value:	a value-array

Note For more information on GMS global settings, see the chapter on managing global settings in the *G2 Menu System User's Guide*.

Using User Preferences

The gms-preferences object contains attributes that determine the appearance of the GMS menus. All GMS user preferences are specified by attributes of a gms-preferences object.

Attribute	Description
gms-priority	Sets the priority that GMS uses to select which User Preferences object to put into effect.
Allowable values:	Any nonnegative integer
Default value:	0
gms-applicability	Specifies the scope of the User Preferences object.
Allowable values:	by-registration, default
Default value:	by-registration
gms-language	When gms-use-translations is true in the current global settings, specifies the language to translate to.
Allowable values:	Any G2 language for which GFR translations are defined or the symbol window-or-system-default. Using window-or-system-default as the value for gms-language sets the language to the window-specific language designated for the window or the current-language specified in the Language Parameters system table.
Default value:	english
gms-font-size	Specifies the font size in which all menu entries are displayed.
Allowable values:	small, large, extra-large
Default value:	large

Attribute	Description
Gms-normal-text-color	Specifies the color of the text in a menu entry that is neither selected nor disabled.
Allowable values:	Any G2 color.
Default value:	black
gms-normal-background-color	Specifies the color of the background in a menu entry that is neither selected nor disabled.
Allowable values:	Any G2 color.
Default value:	white
gms-highlighted-text-color	Specifies the color of the text in a selected menu entry.
Allowable values:	Any G2 color.
Default value:	white
gms-highlighted-background-color	Specifies the color of the background in a selected menu entry.
Allowable values:	Any G2 color.
Default value:	medium-blue
gms-disabled-text-color	Specifies the color of the text in a disabled menu entry.
Allowable values:	Any G2 color.
Default value:	gray

Attribute	Description
gms-header-background-color	Specifies the color of the background in a popup menu header.
Allowable values:	Any G2 color.
Default value:	black
gms-header-text-color	Specifies the color of the text in a popup menu header.
Allowable values:	Any G2 color.
Default value:	light-goldenrod-yellow
gms-sticky-menus	Specifies that sticky menus are available.
Allowable values:	true, false
Default value:	true
gms-show-help-message	Sets GMS to display the Help label of any menu entry that has one whenever that entry is selected.
Allowable values:	true, false
Default value:	false
gms-help-message-fontsize	Specifies the font size in which Help labels are displayed in the help bar.
Allowable values:	small, large, extra-large
Default value:	large

Attribute	Description
gms-blinks-on-activation	Sets the number of times a chosen menu entry blinks before the effect of the choice begins.
Allowable values:	Any nonnegative integer.
Default value:	0
gms-initial-menu-bar	A symbol that is the key of the menu bar that GMS should display when the user first logs in.
Allowable values:	Any symbol, true, false, or none.
Default value:	none
gms-raise-menu-bar-interval	An integer specifying the number of seconds after which the menu bar is raised to the top of the workspace stack.
Allowable values:	Any nonnegative integer.
Default value:	0
gms-separator-height	The height occupied by a separator in a transient menu.
Allowable values:	default, or an integer not less than 6.
Default value:	default

Note For more information on GMS user preferences, see the chapter on customizing the GMS interface in the *G2 Menu System User's Guide*.

Localizing Text in the G2 Menu Bar

The g2uimenu utility uses the G2 Foundation Resources utility (GFR) to enable localized text. The GFR local text resource contains mappings between keys and text for the specified language in GMS menus. If you want user-visible text, such as menus, dialogs, error messages, and workspace labels, to be easily translatable into other languages, you must prepare your menu text by using GFR's local text resources and text resource groups.

Note For more information on localization, see the chapter on localizing KBs in the *G2 Foundation Resources User's Guide*.

Displaying the G2UIMENU Palette

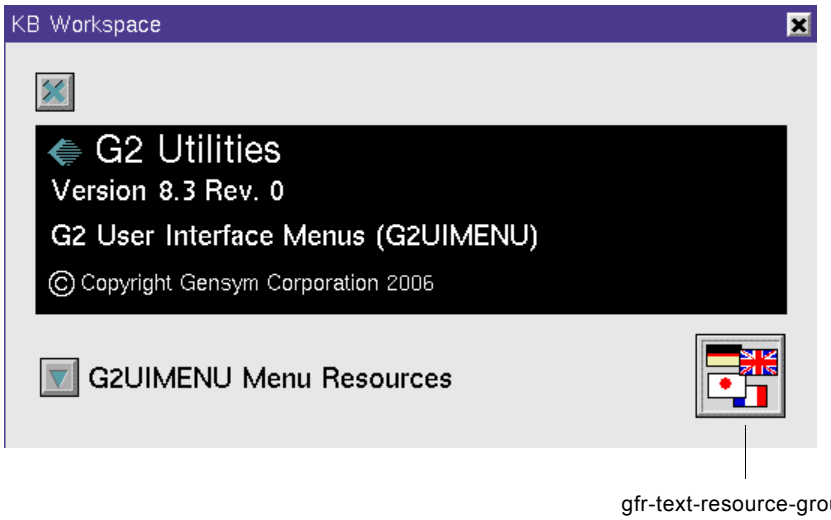
To display the G2UIMENU palette:

1 Display the GDI top-level workspace.

See Displaying GDI Top-Level Workspaces.

2 Click the G2UIMENU Top Level button.

The g2uimenu top-level workspace provides a gfr-text-resource-group, which contains local text resources for localizing the text of the top-level menu bar:



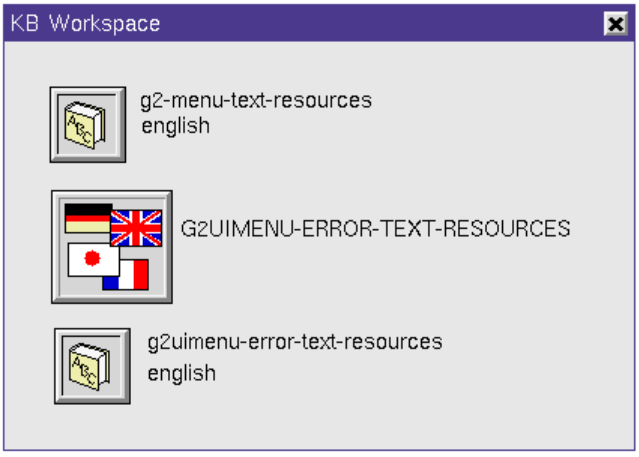
For a description of the G2UIMENU Menu Resources, see Modifying Menu Items in the G2 Menu Bar.

Displaying the Local Text Resources

The subworkspace of gfr-text-resource-group contains the local text resources for the G2 top-level menu bar. It also provides a text resource group and local text resources for various error text.

To display the local text resources:

➔ Click the gfr-text-resource-group on the G2UIMENU top-level workspace:



The G2 menu bar's gfr-local-text-resource object is named g2-menu-text-resources. There is also a gfr-text-resource-group and associated gfr-local-text-resource for the text of error messages.

GDI provides a gfr-local-text-resource for English. You can support other languages by creating gfr-local-text-resource objects for other languages and supplying text translations for each of the menu choice in the G2 menu.

Note If you want to modify menu text, use GFR to create your own localized text resources and place them in your module. By creating your own GFR text resources, you can accept future updates of GDI without affecting your customizations.

Using Local Text Resources

GFR stores language-specific text in gfr- local-text-resource objects, for example:

A local text resource contains symbol-text pairs that associate a symbolic key with a text in a certain language. Although the data structure used to store the symbol-text pairs is private, you can think of the contents of a local text resource object as a table with two columns and any number of rows:

Key	Text Value
ALERT-MSG-1	“A sample alert message”
ALERT-MSG-2	“Delete this [1]?”
...	...
...	...

To create a gfr-local-text-resource to support another language:

1. Display the gfr-top-level workspace.
2. Clone a gfr-local-text-resource object from the palette and configure its attributes, as the following sections describe.

Attributes of gfr-local-text-resource

A gfr-local-text-resource has the following attributes:

Attribute	Description
gfr-language	The language of the texts stored in the resource.
Allowable values:	Any symbol
Default value:	The symbol english
gfr-resource-group	The name of the gfr-text-resource-group that is associated with this object.
Allowable values:	The name of any gfr-text-resource-group
Default value:	The symbol unspecified
gfr-version	A text giving version information about the object.
Allowable values:	Any text
Default value:	The current version of GFR
gfr-file-location	An optional text string giving a file name where the resource can be saved to or loaded from.
Allowable values:	Any text which names a valid file location on your file system
Default value:	"" (the empty string)
gfr-preload-resource	A flag indicating whether a file containing the symbol-text pairs is to be loaded at G2 startup, if the resource is not permanently stored in G2.
Allowable values:	true or false
Default value:	false

Entering Symbol-Text Pairs into a Local Text Resource

The three ways to enter symbol-text pairs into a local text resource are:

- Create an external text file using any text editor, then load it into the text resource object.
- Load the G2 XL spreadsheet module and edit the text resource from inside G2.
- Programmatically add symbol-text pairs, using the GFR API procedure, gfr-add-to-local-text-resource.

The first two options are explained in the following sections. For a description of the programmatic option, see the *G2 Foundation Resources User's Guide*.

Using an External Text Editor to Edit a Local Text Resource

GFR supports editing of local text resources outside of G2. This design enables someone who does not know G2 to translate your KB into a new language, using any word processor or text editor. Then, you can load the translation into G2 with almost no effort.

To prepare a new local text resource

➔ Create a text file.

The first three lines of the text file must contain the following, with each item on a separate line:

- The resource group name, as a symbol.
- A line of version information which may help you identify the file, as a quoted text.
- The language, as a symbol.

Starting on the fourth line of the file, type the symbol-text pairs, separated by a comma, one pair per line. Enclose the texts following the symbol keys in double quotation marks.

Note Do not type carriage returns in the body of the texts.

For example, the text file could look like this:

```
MY-ALERT-TEXTS
"Version 7.0"
ENGLISH
ALERT-MSG-1, "A sample alert message"
ALERT-MSG-2, "Delete this [1]?"
```

For clarity, symbols are in capital letters, but this is not necessary.

If you have embedded quotation marks in any texts, use two sets of double quotation marks. For example, if the text to be loaded is abc "def" geh, represent this in the file as "abc""def""geh".

You can use any special or foreign characters that are part of the Gensym character set, as described in *G2 Reference Manual*. For example, the following text file is for English and includes a newline character, the copyright symbol, and an accented character for the word "café:"

```
MY-ALERT-TEXTS
"Version 7.0"
ENGLISH
ALERT-MSG-1, "The first line.@LThe second line."
ALERT-MSG-2, "Copyright ~| 2002"
ALERT-MSG-3, "Caf~e"
```

To load the file contents into a local text resource object:

1. Start G2 and clone a local text resource from gfr-top-level workspace.
2. Display the table of the local text resource object and edit these attributes:
   a. Change the gfr-file-location attribute to the path of the file you want to load.
   b. Change the gfr-resource-group attribute to the resource group named in the file (MY-ALERT-TEXTS, in the example).
3. Choose load text resource from the local text resource object's menu.

This menu choice loads the data from your file, replacing whatever was previously stored.

Note The menu choices for loading and saving the resource to a file only appear when the gfr-file-location attribute is specified.

Using the G2 XL Spreadsheet to Edit a Local Text Resource

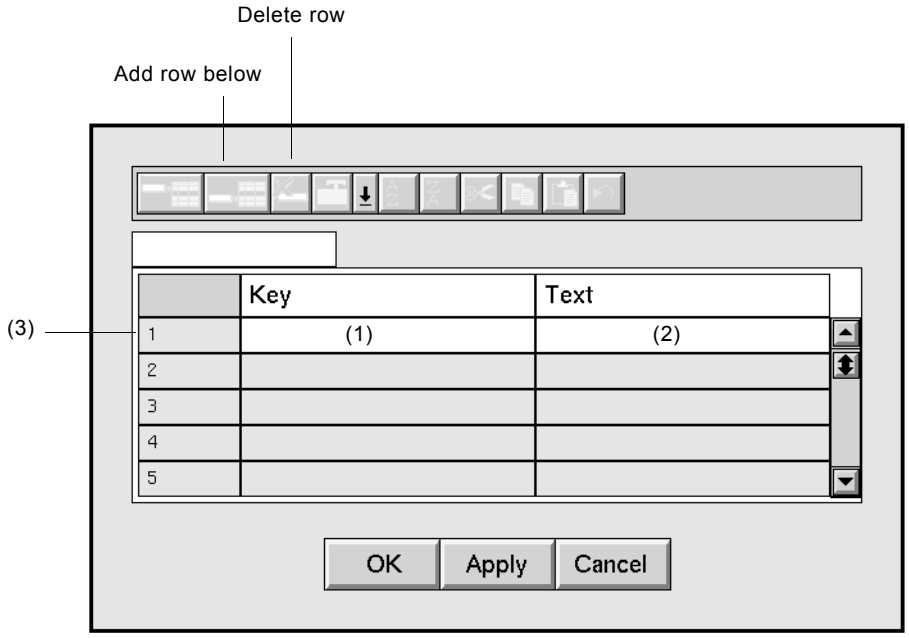
The G2 utility G2 XL Spreadsheet (GXL) provides a convenient way to edit local text resources from within G2.

To use a spreadsheet to edit a local text resources:

1. Merge the *gxl.kb* module located in the *utils* subdirectory of the *kbs* directory under the *g2* directory.
2. Display the gxl-top-level workspace and click the check box labelled array and list editing on this workspace.

   When you enable array and list editing, G2 adds the edit resource menu choice to the menu of local text resource objects.

3 Choose edit resource to see the following spreadsheet workspace:



To enter the first key-text pair:

1 Click in the cell labelled (1), type the desired symbolic key, and press Return when you are finished.

2 Click in the cell labelled (2) and type the text.

Do not use enclosing quotation marks when entering texts in the spreadsheet.

To enter special characters:

➔ Use keystroke commands, as described in the *G2 Reference Manual*.

For example, to enter a carriage return, type Ctrl + j.

To add a second key-text pair.

1 Click in the cell labelled (3) to select the first row of the spreadsheet.

2 Click the add row below button (second from the left on the toolbar) to add an empty row below the first row.

3 Repeat this process to add as many key-text pairs as you need.

When you exceed five rows, a vertical scroll bar appears on the right side of the spreadsheet to allow you to access rows not shown on the spreadsheet.

You can also delete a row by selecting the row and clicking the delete button (third button from the left on the toolbar).

When you are finished entering data, click the OK button. This will save the values you have entered into the local text resource. If you do not want to save your edits, click Cancel.

For information on sorting, cutting, pasting and other spreadsheet functions, see the *G2 XL Spreadsheet User's Guide*.

When you are finished editing resources, you can delete the GXL spreadsheet module.

To delete the spreadsheet module:

➔ Choose Main Menu > Miscellany > Delete Module > gxl.

Choose the All option to also delete all workspaces in the module.

Storing Local Text Resources

You have several options on how G2 stores key-text pairs in a local text resource:

- Store the values permanently in the KB.
- Store the values permanently in a file and load them each time G2 is started.
- Store the values permanently in a file and load them only on demand.

Storing Local Text Resources in a KB

If you store the values permanently in a KB, you have the convenience of not having to manage auxiliary files, because all information is stored in G2. However, there is a certain memory penalty for storing the information in G2 on a permanent basis. The penalty is roughly about 200 bytes per key-text pair, although this depends on the length of the texts.

To store the information in a local text resource as a permanent part of a KB:

➔ Choose make resource permanent from the menu of the resource or call gfr-make-local-text-resource-permanent.

For more details, see the *G2 Foundation Resources User's Guide*

Loading a Local Text Resources File at Startup

If you choose to load local text resources at G2 startup, the memory requirement is reduced to about 100 bytes per key-text pair, depending on the length of the texts. However, starting up G2 is slower because the file is loaded when G2 is started. Also, you must make sure that the path to the file named in gfr-file-location is always valid.

To load local text resources at G2 startup:

➔ Specify the file name in the gfr-file-location attribute and set the gfr-preload-resource attribute to true.

Loading a Local Text Resources on Demand

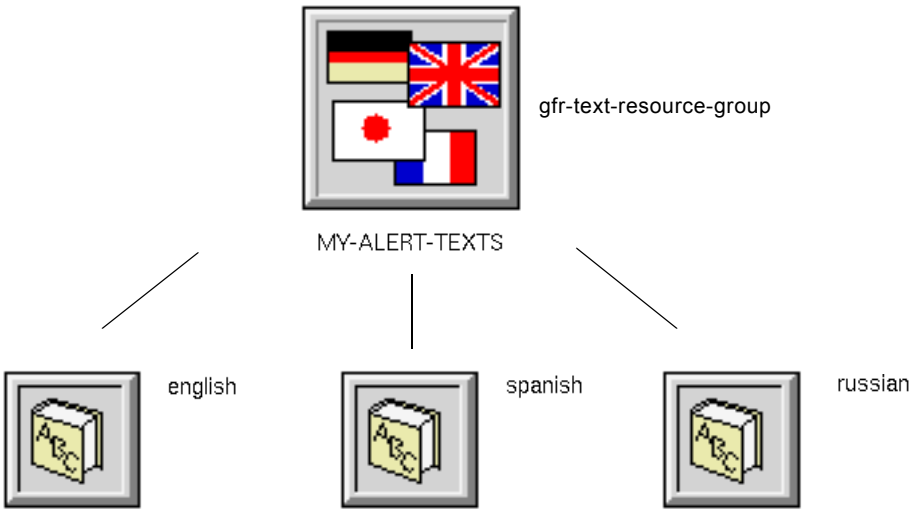
If you choose to load the resource only on demand, you save 100% of the memory for the local text resources that are not used during a G2 session. If you are supporting several languages, and only one is currently in use, this option saves you from loading unused languages. However, the first time any resource is demanded, CPU time is allocated to loading the file.

To load local text resources on demand:

➔ Specify the file location and set gfr-preload-resource to false.

Using Text Resource Groups

Each local text resource is associated with exactly one gfr-resource-group, which serves to link resources in different languages. All local text resources associated with a resource group contain the same keys but different languages, as shown in the following figure:



Within a KB, you can have as many resource groups as you like. Usually, each module supplies its own resource group or groups.

To create a text resource group:

1. Display the gfr-top-level workspace.
2. Clone a gfr-text-resource-group object from the palette and configure its attributes.

Gensym recommends that, within a module, you divide texts according to their purpose and have multiple resource groups, rather than putting all the texts in one resource group. This both shortens access time and makes it easier to

organize your texts. For example, you might make a resource group for error messages, another for dialog labels, and another for texts appearing on menus.

Caution You cannot have more than one local text resource with the same language in a given resource group.

The attributes of a gfr-text-resource-group are:

Attribute	Description
gfr-version	A text giving version information about the object.
Allowable values:	Any text
Default value:	The current version of GFR
gfr-default-language	Indicates which language to use when the requested language is not supported or unspecified, or if the key is not found in the requested language.
Allowable values:	Any symbol
Default value:	The symbol english
gfr-use-default-language	A flag indicating whether the default language is to be used.
Allowable values:	true or false
Default value:	true

Modifying Menu Items in the G2 Menu Bar

The g2uimenu module uses the G2 Menu System (GMS) utility module to modify menu items in the G2 menu bar. You construct the menu items by cloning menu objects from the GMS palette, then by linking the menu objects together on your workspace. You can use GMS to:

- Add or remove menu items in the G2 menu bar.
- Change entire submenus.
- Change the behavior of menu items.
- Enable or disable menu items.

To modify a g2uimenu menu resource, clone it, place the cloned copy in your own module, then edit your copy as needed.

Note For more information on creating GMS menu specifications, see Part II of the *G2 Menu System User's Guide*.

Displaying the G2UIMENU Palette

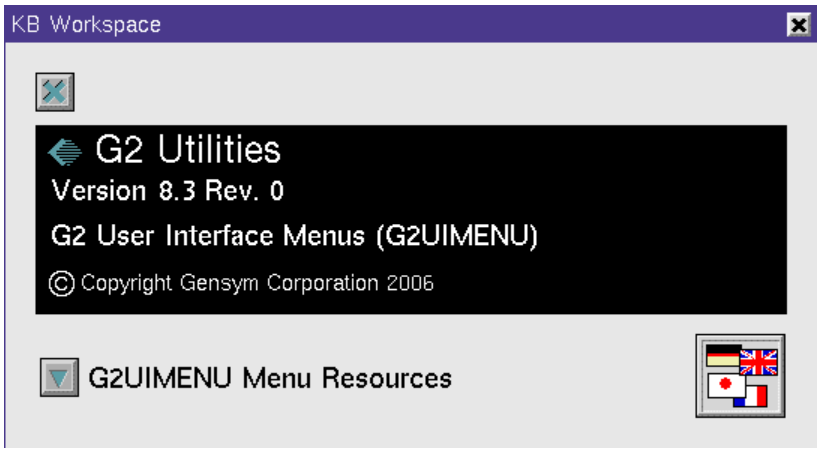
To display the G2UIMENU palette:

1 Display the GDI top-level workspace.

See Displaying GDI Top-Level Workspaces.

2 Click the G2UIMENU Top Level button.

The g2uimenu-top-level workspace contains a button labeled G2UIMENU Menu Resources, which allows you to modify the menu choices on the G2 menu bar:

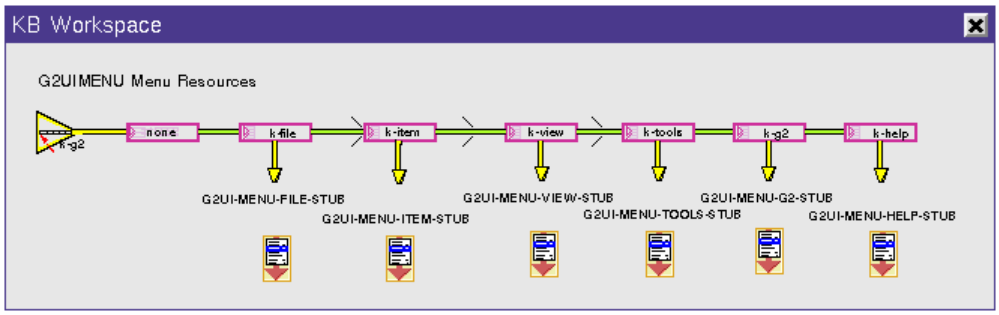


Displaying the Menu Resources

To display the menu resources:

➔ Click the G2UIMENU Menu Resources button on the G2UIMENU top-level workspace.

The following figure shows the menu resources for the G2 menu bar provided by g2uimenu:



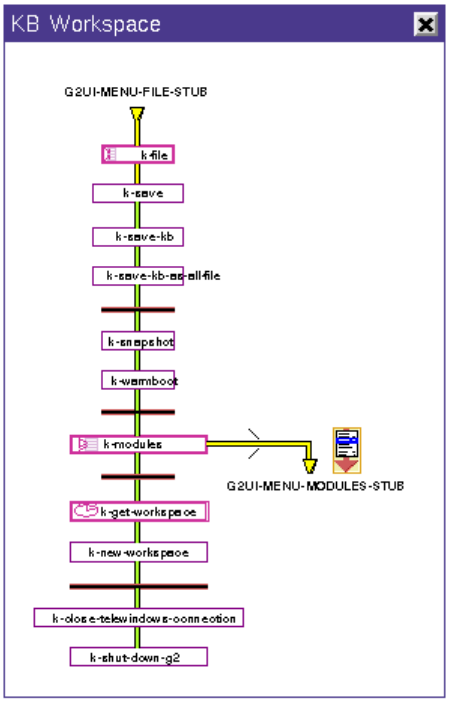
So that you can better maintain and organize pull-down menus, they are defined on separate subworkspaces called GMS subpanels. Each of the menus you see in the G2 menu bar appears: File, Item, View, Tools, G2, and Help. The complete specification for each menu is found in the subpanel.

GMS matches the names of the stub and subpanel. For example, the name of the file menu item g2ui-menu-file-stub matches the name on the subpanel g2ui-menu-file-stub.

To display the subpanel of a menu item:

➔ Choose go to subworkspace on the subpanel object.

The following figure shows the subpanel for the menu resource for the File menu, g2ui-menu-file-stub.

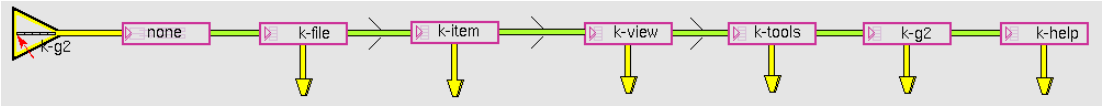


Adding and Modifying Menu Items

In the following steps, the menu bar for GUIDE provides an example of how you can start with an existing menu bar and then add or modify only a few parts of it to suit your needs.

To modify the menus in the G2 menu bar:

1 Clone the entire G2 menu bar, shown in the following figure:



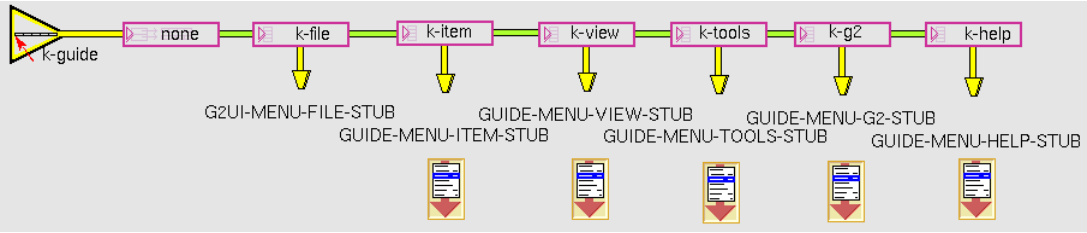
Tip You can use Operate On Area to clone the resource and then transfer it to a workspace in your own module.

2 Change the name of the initiating entry, the menu bar template, to one of your choice, such as an application or module-specific name.

   For example, change the name from k-g2 to k-guide.

3 For those menus that you do not plan on changing, make sure that the connection stubs have the same names as the stubs from which they were cloned.

   For example, the following figure shows GUIDE's menu resource. GUIDE makes no additions to the File menu in the G2 menu bar, so the connection stub for the File menu has the same name as the connection stub in the menu delivered with g2uimenu.



   GUIDE makes additions to all the other menus, so the names for the menu stubs are changed to be GUIDE-specific.

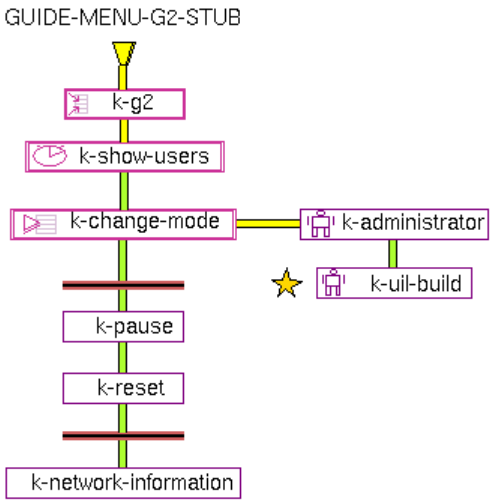
4 Clone the gms-subpanel for the menu that you want to modify.

5 Rename the connection stub on the cloned gms-subpanel to match the name given to the main menu item stub.

   For example, use the name guide-menu-g2-stub for the G2 menu for both the connection stub and the main menu stub.

6 Use the GMS palette to make modifications to the menu in the gms-subpanel.

The following figure shows the gms-subpanel for the G2 menu in GUIDE. GUIDE adds the user-mode, uil-build, to the User Mode cascade menu:



Removing Menu Items

The gms-inactive-keys attribute of the gms-global-settings object associated with the G2 menu bar contains a list of GMS menu keys for those items that should be temporarily removed from the menus. The name of the gms-global-settings object for the G2 menu bar is called g2uimenu-globle-settings (the misspelling is correct).

To remove menu items:

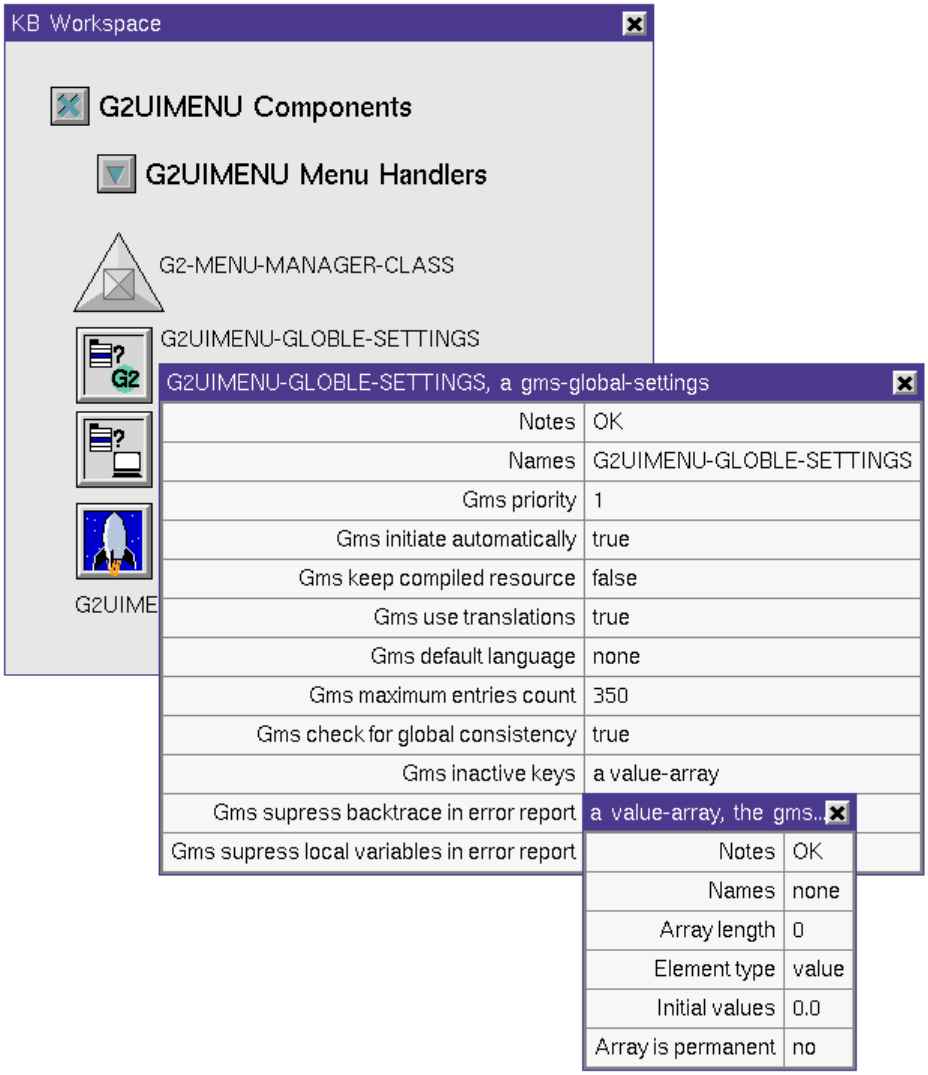
1 Enter the following command in Inspect:

    go to g2uimenu-globle-settings

2 Choose table to display its table.

3 Choose subtable on the gms-inactive-keys attribute.

Here is the g2uimenu-global-settings object and its table and subtable:



4 To specify inactive keys, enter them in initial-values as a list of symbols separated by commas, for example, the symbol K-FILE-MENU.

When the GDI menu resource is compiled, GMS looks at the list of inactive keys.

This list can be dynamically updated during a G2 session. For example, menu items can be dynamically added or removed if the mode of G2 changes.

Note If you want to permanently remove menu options, delete them from your cloned copy of the GDI menu resource.

Enabling and Disabling Menu Items

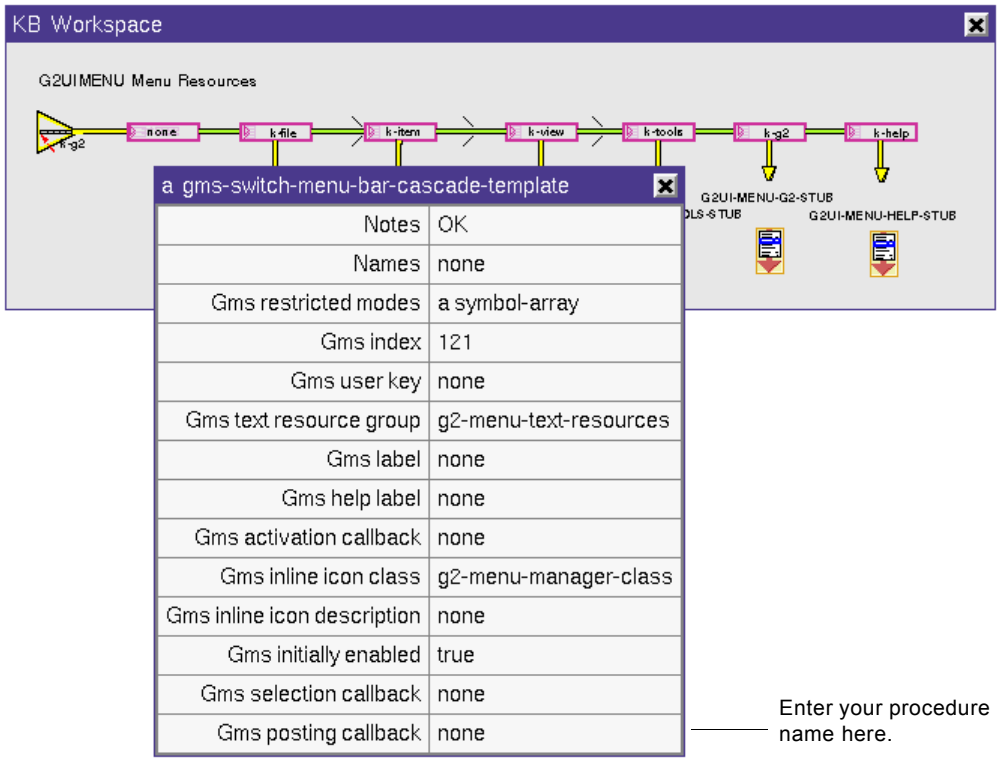
If you need to enable or disable the menu item(s) added to the G2 menu, specify your own posting callback for either the menu item or cascade menu for the menu item. The posting callback is called or invoked just before the menu is displayed.

To enable and disable menu items:

1. Display the table for a GMS cascading entry, such as a gms-cascade-template.
2. Edit the gms-posting-callback attribute to specify a callback procedure such as the following:

```
example-posting-callback (handle: integer, activation-path: item-or-value,
    menu-index: integer, display-status: truth-value)
index: integer;
begin
    {–foo is value of the gms-user-key attribute of the menu item that
        you wish to enable or disable –}
    index = call gem-get-index-for-key (handle, the symbol foo);
    {– check if index is valid because menu option may not exist –}
    if (index >= 0) then begin
        if (specified_condition_is_true) then
            call gms-enable-entry (handle, index);
        else
            call gms-disable-entry (handle, index);
    end;
end
```

Here is the table for a cascade entry in the G2 menu bar:



Changing the Behavior of Menu Items

GMS menus execute procedures when the user chooses leaf entries from menus. These procedures are activation or posting callback procedures that determine menu creation, posting, and activation for the G2 menu bar.

Note GDI refers to callback procedures as menu handlers.

For more information on writing callback procedures, see the chapter on writing and using callback procedures in the *G2 Menu System User's Guide*.

Programming Hints

By following these programming hints, you can create menus that are easier to maintain and use, and, more importantly, that can coexist with GMS menus of other applications:

- Use GFR for all menu text. Set the gfr-use-default-language attribute of the gfr-text-resource-group to true.
- Use a gms-global-settings object and set gms-use-translations to true.

- If you are not using the G2 menu provided by g2uimenu, make sure you have a gms-switch-menu-bar-cascade-template, shown in the following figure, on your menu specification, to the right of the Menu Bar root template.

The gms-switch-menu-bar-cascade-template enables other modules to register their menus with GMS. GMS creates a menu item on the gms-switch-menu-bar-cascade-template for every menu found during initialization. Using the G2 Logo menu, you can switch between menu bars in a G2 session.

- Use gms-reusable-panel-templates for attaching menus to gms-cascade-templates. This provides greater flexibility and extensibility by allowing multiple menus to share the same menu panel.

Modifying Dialog Layout and Actions

The dialogs can be used in your application just as they are or with slight modifications made with a dialog settings object.

Note If you want to use unmodified dialogs in your application, you need to launch the specified dialogs. GDI includes a launch procedure for each dialog in its API. By using the launch procedure, you can return the information from the dialog to your application.

Modifying a Dialog

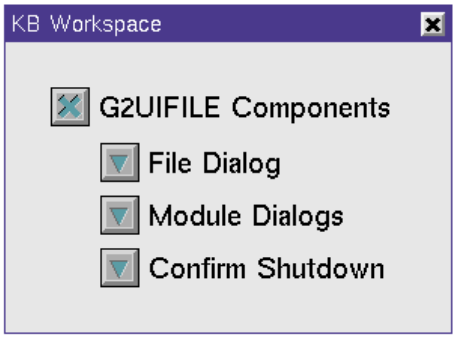
To modify a dialog:

1 Display the GDI top-level workspace.

See Displaying GDI Top-Level Workspaces.

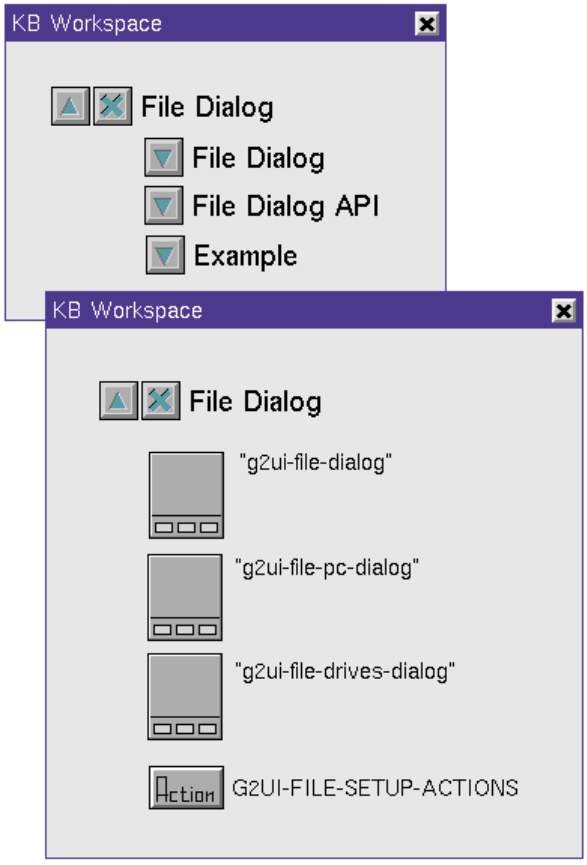
2 Click the G2UIFILE Top Level button.

3 Click the G2UIFILE Components button to display this workspace:



4 Locate the master dialog most like the dialog you are trying to create.

For example, here is how you would navigate to the File master dialog:



5 Click on the dialog icon and choose the clone. option.

6 Transfer the cloned dialog to a workspace in your own module.

7 Edit the attribute display that contains the dialog ID to indicate its purpose.

The ID must be changed and must not clash with the IDs of any of the other dialogs supplied by other required modules.

8 Show the subworkspace of the dialog.

9 Delete the controls that you do not need.

10 Create a dialog settings object of the type associated with the dialog you are modifying and make customizations.

11 Use the supplied API for launching the applicable type of dialog.

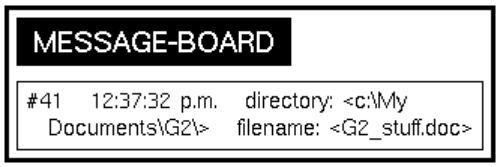
For example, if you started with the basic Open File dialog, you can call g2ui-file-launch-dialog with the ID of your master dialog and your copy of a g2ui-file-dialog-settings object.

The following procedure uses the g2ui-file-launch-dialog to return the values for the names of a file and its containing directory as specified in the Open File dialog.

```
example-launch-file-dialog (window: class g2-window, settings: class g2ui-file-
   dialog-settings)
directory, filename: text;
type: symbol;
begin
  type = call g2-type-of-file-system ();
  if (type = the symbol win32) then
    filename, directory = call g2ui-file-launch-dialog ("g2ui-file-pc-dialog",
       window, settings)
  else
    filename, directory = call g2ui-file-launch-dialog ("g2ui-file-dialog", window,
       settings);

  inform the operator that "directory: <[directory]>  filename: <[filename]>";
end
```

Note This procedure is included in the File Dialog examples for g2uifile.

The filename and directory names specified in the Open File dialog appear in the Message Board, as shown in the following figure. Of course, in your application, you can use the return values for your own purposes.



If you want to use a dialog but with a slight modification, clone a dialog settings object from the g2uifile palette. Change the attribute values as needed. For

example, you can rename the dialog title from Open File to something else like Specify Filename.

There are many other attributes in the dialog settings object that you can change. For example, you can specify an different validation procedure, which checks for errors when a dialog is invoked.

For more information on settings objects, see Modifying Dialog Text.

Adding and Removing Controls

You can add controls to a GDI dialog or remove existing controls. Once you have added or removed a control, use the dialog settings object to customize all text and behaviors for the default controls.

Use the following procedure to add or remove controls. The Open File dialog is used as an example.

To add a control to the Open File dialog:

1. Locate the master dialog for the Open File dialog.
2. Click on the dialog icon and select the clone. option.
3. Use transfer to move the cloned dialog to a workspace containing your own module.
4. Edit the attribute display that contains the dialog ID.

   The default ID for the PC version of the Open File dialog is g2ui-file-pc-dialog.
5. Edit the ID to indicate its purpose in your module, for example, my-save-module-dialog.

   The ID must be changed and must not clash with the IDs of any of the other dialogs supplied by other required modules.
6. To display the cloned master dialog for the Open File dialog, click on the dialog icon and select show subworkspace.
7. To add controls, either clone them from the GUIDE palette or select them from the GUIDE Objects option in the Item menu in the GUIDE menu bar.

Note To use GUIDE UIL controls, you must have GUIDE loaded into G2, switched to the GUIDE menu bar, and selected uil-build mode in the Change Mode option in the G2 menu.

---

8. Drag and drop each control onto the cloned master dialog.

Make sure to give each control a unique ID so that you can refer to it programmatically, for example, save-required-button.

To remove a control from the Open File dialog:

➔ Click on the controls to delete and select delete.

To use your own setup handler:

➔ Create a dialog settings object and override the value of the file-dialog-setup-handler attribute to name a user-defined setup handler.

The default file-dialog-setup-handler is g2ui-file-setup-dialog.

Note Each variation of dialog settings objects has a setup handler.

The following setup handler initializes the controls that were added to the default Open File dialog. The setup handler is called from the dialog event queue that handles the events for the dialog while is it being launched and managed. The following setup handler code sample calls the default setup handler, g2ui-file-setup-dialog, then specifies a new control, the save-required button.

```
my-save-module-setup-dialog
    (dialog: class uil-dialog, this-initiating-item: item-or-value,
     window: item-or-value, original-initiating-item: item-or-value,
     dialog-actions: class symbol-list)

    my-button: class uil-icon-toggle-button;
    begin
        call g2ui-file-setup-dialog (dialog, this-initiating-item, window,
            original-initiating-item, dialog-actions);
        my-button = call uil-get-grobj-from-id-on-dlg-or-wksp
            (“save-required-button”, dialog);
        if (my-button exists) then
        begin
            conclude the the toggle-state of my-button is on;
            call uil-configure-grobj-method (my-button);
        end
    end
```

Hint To specify new controls, call the default setup handler first to pick up existing control specifications.

The my-save-module-setup-dialog code sample accesses a toggle-button on the dialog by calling the procedure, uil-get-grobj-from-id-on-dlg-or-wksp (see the *G2 GUIDE/UIL Procedures Reference Manual*). The toggle-button is located by the value in its ID attribute. The toggle button is then set to be ON and configured to reflect its current state.

To launch your dialog:

➔ Create a dialog launch procedure for invoking your dialog.

Note Your launch procedure must handle the information from all the controls on the dialog.

The following sample launch procedure handles the initialization of the control added to the dialog and a procedure to get the information from *all* the controls when the user clicks on the OK button.

```
sample-user-defined-dialog-launch-procedure
    (dialog-id: text, window: class g2-window,
     settings: class g2ui-file-dialog-settings)

    dialog, button: item-or-value;
    edit-box: item-or-value
    my-button: class uil-icon-toggle-button;
    dir, file: text;
    save-required: truth-value = false;
    begin
    {-- launch the dialog and let it start processing events in the dialog
        queue --}
    dialog, button = call uil-start-dialog-processing (dialog-id, settings,
        window, _g2ui-file-setup-actions, false, false, the symbol none,false);

    {-- user has clicked on either the OK or Cancel button and the dialog has
        been unmanaged, process the controls on the dialog before releasing
        it. --}
    dir = “unspecified”;
    file = “unspecified”;
    if (button exists and button is an item) then
        begin
        if not (the uil-cancel-button of button is true) then
            begin
            edit-box = call uil-get-grobj-from-id-on-dlg-or-wksp
                (“filename-edit-box”, dialog);
            if (edit-box exists and edit-box is an item) then
                file = the message-contents of edit-box;

            edit-box = call uil-get-grobj-from-id-on-dlg-or-wksp
                (“directory-edit-box”, dialog);
            if (edit-box exists and edit-box is an item) then
                dir = the message-contents of EB;

            my-button = call uil-get-grobj-from-id-on-dlg-or-wksp
                (“save-required-button”, dialog);
            if (my-button exists and my-button is an item) then
                begin
                if (the toggle-state of my-button is on) then
                    save-required is true
                else
                    save-required is false;
                end;
```

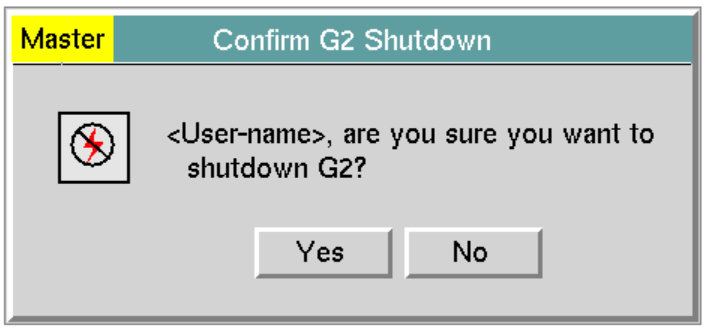
```
        end;
    end

{-- release the dialog back to the dialog pool --}
if (dialog exists and dialog is an item) then
call uil-release-dialog (dialog, window, false);
{-- Now that you have the information, you would do something to it
(dir, file, save-required) --}
end
```

Example of Modifying a Master Dialog

The following steps walk you through the modification of the Confirm Shutdown dialog. If you need to modify one of the GDI dialogs for your own purposes, you will perform actions similar to the ones specified here. The basic steps in master dialog modification are:

- Clone the appropriate master dialog so that you can work on your own copy.
- Give your copy of the master dialog its own unique ID.
- Add and remove controls as needed.
- Create a subclass of the dialog settings object so that you can add attributes and your own default values.
- Write a setup handler to initialize your controls and default values.
- Write a launch procedure that uses your dialog settings object and carries out the task of your modified master dialog.

Before making your modifications, you must find a master dialog to modify. Look through the master dialogs to find one that nearly matches what you need. For example, if you wanted to add the user's name to the shutdown dialog, you would start with the GDI Confirm Shutdown dialog to create a new master dialog, as shown in the following figure:



Note To make the following modifications, you must load *g2cuidev.kb* and *guide.kb* into G2.

To modify the Confirm Shutdown dialog:

1 Clone the Confirm Shutdown master dialog and place the copy on a workspace belonging to your module.

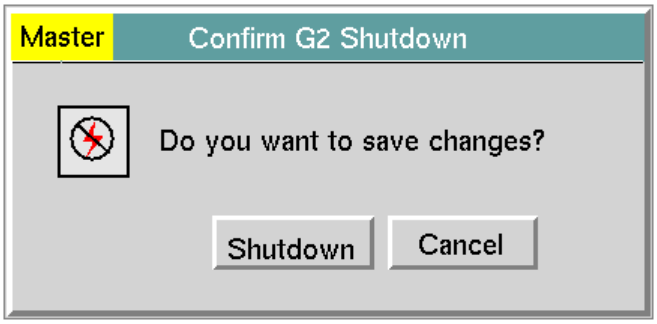
For more information on the Confirm Shutdown master dialog, see Close and Shutdown Dialogs.

2 Because each master dialog in an application must have a unique ID, change the ID of the copy on your workspace.

The default ID of the Confirm Shutdown dialog is g2ui-file-confirm-shutdown-dialog. You might enter my-confirm-shutdown-dialog as the ID.

3 Remove the control you do not need and rearrange the remaining ones.

For example, delete the Save button and move the remaining buttons, as shown in the following figure:



4 Create a new class of the dialog settings object by subclassing g2ui-file-confirm-shutdown-dialog-settings.

In the example, you would name the new class my-confirm-shutdown-dialog-settings.

5 To the my-confirm-shutdown-dialog-settings class, add an attribute for user-name to class-specific attributes and change the attribute initializations for attributes that should have new default values.

For example:

MY-CONFIRM-SHUTDOWN-DIALOG-SETTINGS

Class specific attributes user-name initially is ""

Attribute initializations shutdown-button-label initially is "Yes";
cancel-button-label initially is "No";
confirm-shutdown-dialog-setup-handler
initially is my-confirm-shutdown-dialog-setup;
dialog-text initially is "Are you sure you want
to shutdown?"

6 Create a new setup handler that updates the message on the dialog to include the user's name.

This procedure calls the default setup handler and then performs the additional setup actions. For example:

```
my-confirm-shutdown-dialog-setup (dialog: class uil-dialog, this-initiating-item:
   item-or-value, window: item-or-value, original-initiating-item: item-or-value,
   dialog-actions: class symbol-list)
MsgTxt : item-or-value;
begin
  call g2ui-file-confirm-shutdown-dialog-setup (dialog, this-initiating-item,
     window, original-initiating-item, dialog-actions);
  if the user-name of this-initiating-item /= "none" then begin
    MsgTxt = call uil-get-grobj-from-id-on-dlg-or-wksp ("dialog-text", dialog);
    call uil-update-text(MsgTxt, "[capitalize-words("[the user-name of this-
       initiating-item]")], are you sure you want to shutdown?");
  end;
end
```

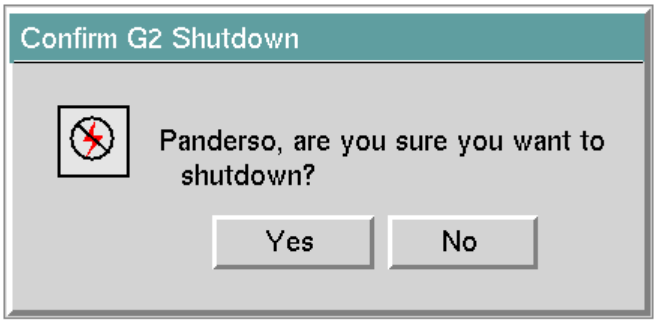
Note The value of the confirm-shutdown-dialog-setup-handler attribute in the dialog settings object is the name of the new setup handler, my-confirm-shutdown-dialog-setup.

7 Create a launch procedure, my-launch-confirm-shutdown-dialog.

The launch procedure creates the dialog settings object, concludes a value for the user-name attribute, calls the g2ui-file-confirm-shutdown-launch-dialog, and receives the result. For example:

```
my-launch-confirm-shutdown-dialog(Win : class g2-window)
settings : class my-confirm-shutdown-dialog-settings;
result : symbol;
begin
  create a my-confirm-shutdown-dialog-settings settings;
  conclude that the user-name of settings = the text of the g2-user-name
    of Win;
  result = call g2ui-file-confirm-shutdown-launch-dialog ("my-confirm-
    shutdown-dialog", Win, settings);
  inform the operator that "[result]";
  {Replace the inform the operator line with the following line of code to
    really do the shutdown.

  if result = the symbol shutdown-g2 then shut down g2;}
end
```

When you invoke the dialog from a GMS menu item or an action button, the dialog includes the user name when it asks for confirmation, as shown in the following figure:



4

Customizing the Dialogs

Describes the file, module, and G2 session management capabilities provided by G2UIFILE.

Introduction

The g2uifile utility delivers a set of fully customizable and localizable dialogs on file and module manipulation. These dialogs support UNIX and Windows file systems. You can customize these dialogs using the UIL IDs, attributes, setup actions, and API calls described in this chapter.

Note Only advanced users of GUIDE/UIL should customize the dialogs provided with GDI.

Displaying the GDI Master Dialogs

You can navigate to the master dialogs through the top-level workspace for g2uifile.

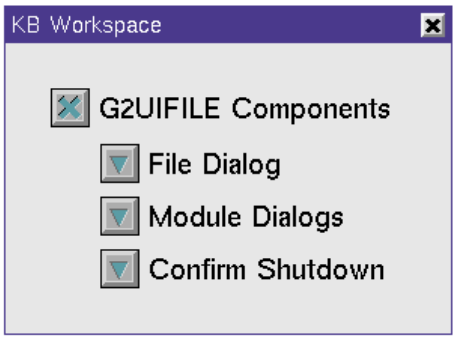
To navigate to the GDI master dialogs:

1. Display the GDI top-level workspace.

   See Displaying GDI Top-Level Workspaces.

2. Click the G2UIFILE Top Level button.

3. Click the G2UIFILE Components button to display this workspace:



Subworkspaces exist for the File, Module, and Shutdown dialogs. These subworkspaces describe the master dialogs that you can clone and customize.

For each dialog, this chapter presents information in the following four categories:

- **Master dialogs** – Each dialog has a master dialog and an API that supports launching, initializing, and dismissing the dialog. The API consists of dialog settings objects, action description arrays, and procedures. To customize a dialog, first clone the master dialog and transfer it to your application. Using the settings object, you can change the appearance and text of a dialog. For more complex customizations, you can add or remove controls and override default handlers.

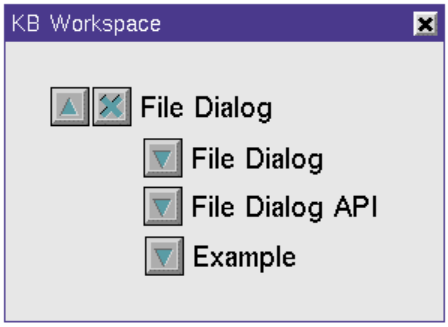
- **Dialog settings objects** – A dialog settings object is an argument passed to the API call that launches a dialog. The settings object is the initializing object of the dialog and contains attributes for the dialog's title, the button labels, localized text, and default values of controls. It specifies setup, validation, and

handlers. To customize a settings object, clone it from the GDI palette. To add additional attributes, you can subclass the settings object class.

- **Setup actions** – Most dialogs have an associated uil-description-array that describes the actions that are executed when the dialog is launched. Modify the action description array only to remove actions. To add actions, use your copy of a dialog settings object.
- **API calls** – Each dialog has supporting API calls for launching, initializing, validating, and dismissing a dialog. The API calls handle the user interface and do not perform operations. They return information necessary to perform operations. For example, the g2ui-file-create-module-launch-dialog returns the name of the module to create; it does not create the module.

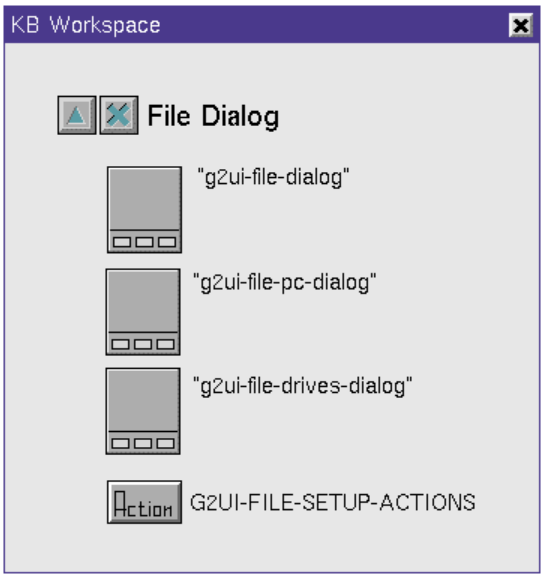
File Dialog

The File dialog enables you to clone and customize dialogs for file management. You can view the master dialogs for the PC and UNIX, the procedures for the API, and an example of customizing the File dialog.



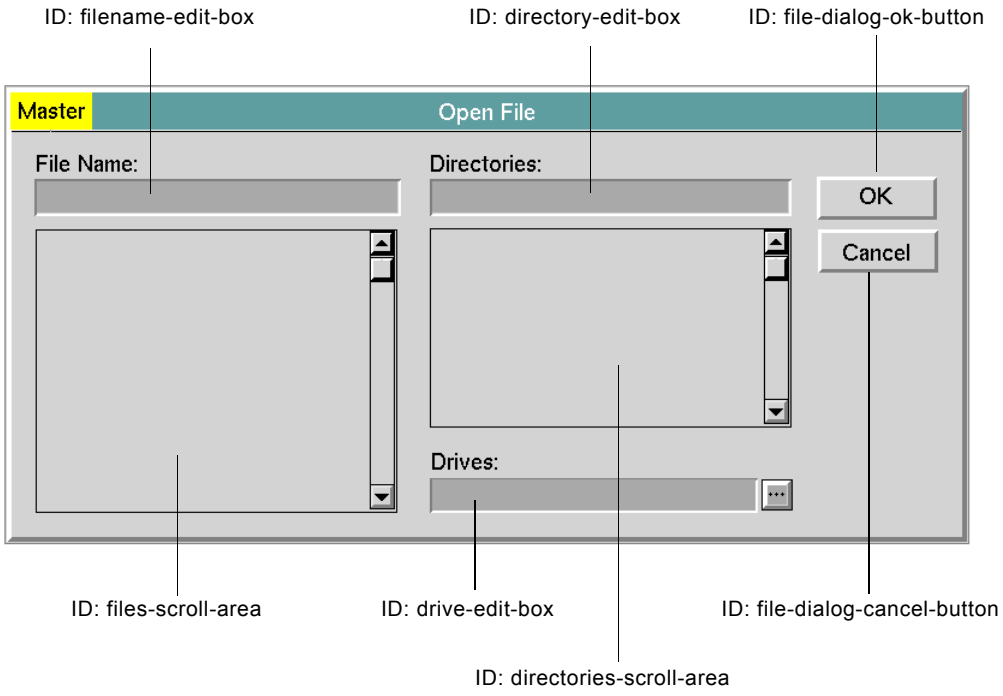
The top-level workspace for each type of file dialog is organized with subworkspaces containing master dialog(s), examples, and the public API.

Selecting the File dialog navigation button displays the workspace containing the master dialogs, as shown in the following figure:



Caution Application developers are free to clone the master dialogs, remove controls, and add their own controls. *However, it is extremely important that the IDs for the controls that come with the master dialog are not changed.* If IDs change, the underlying code that coordinates the update and display of files, directories and modules will fail because the ID maps the code to the user interface.

The following figure shows the master dialog for basic file management. This master is for a PC, but master dialogs also exist for the UNIX environment.



The IDs of the edit boxes and buttons are:

- filename-edit-box
- files-scroll-area
- directory-edit-box
- directories-scroll-area
- drive-edit-box
- file-dialog-ok-button
- file-dialog-cancel-button

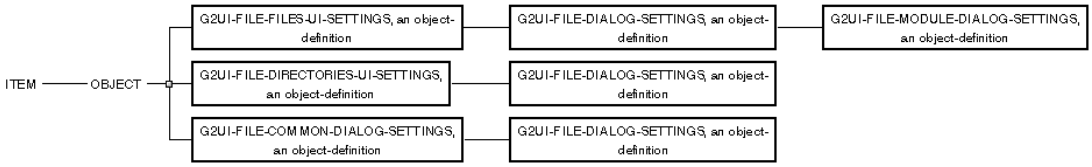
Attributes

The File dialog attributes belong to five classes. The settings object for the File dialog inherits attributes from g2ui-file-common-dialog-settings, g2ui-file-directories-ui-settings, and g2ui-file-ui-settings. The Module and Shutdown dialogs also use these attributes.

- g2ui-file-common-dialog-settings
- g2ui-file-directories-ui-settings

- g2ui-file-ui-settings
- g2ui-file-dialog-settings
- g2ui-file-module-dialog-settings

The following figure depicts the class hierarchy of the dialog settings objects for the file and module dialogs:



For attributes applicable only to a specified dialog, see the description of that dialog and Attributes of g2ui-file-module-dialog-settings. To change the text for labels and buttons on a dialog, see the Modifying Dialog Text.

Attributes of g2ui-file-common-dialog-settings

Attribute	Description
use-module-dialog-preferences	Specifies that the colors of all dialogs are taken from the preferences object, g2uifile-dialogs-module-settings, highest in the module hierarchy. A value of false means that you can configure the colors of a particular master dialog to whatever you want.
Allowable values:	true or false
Default value:	true
dialog-title	Specifies the text to use as the dialog's title.
Allowable values:	Any text value
Default value:	"Open File"
Notes:	For localization, set the value of this attribute through the use of GFR techniques.

Attribute	Description
dialog-title-justification	Specifies the justification for the text of the dialog's title.
Allowable values:	center, left, right
Default value:	left
dialog-title-configuration	Specifies a uil-text-configuration-class object to use to configure the dialog's title text.
Allowable values:	The name of a uil-text-configuration-class object
Default value:	uil-dialog-title-configuration
ok-button-label	Specifies the text to use for the OK button's label.
Allowable values:	Any text value
Default value:	"OK"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
apply-button-label	Specifies the text to use for the Apply button's label.
Allowable values:	Any text value
Default value:	"Apply"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
cancel-button-label	Specifies the text to use for the Cancel button's label.
Allowable values:	Any text value
Default value:	"Cancel"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
button-layout-style	Specifies the layout style for the dialog's OK, Apply and Cancel buttons.
Allowable values:	motif-style or windows-style
Default value:	"windows-style"

Attributes of g2ui-file-directories-ui-settings

Attribute	Description
directory-edit-box-label	Specifies the text to use for label of the edit-box that contains the directory information.
Allowable values:	Any text value.
Default value:	"Directories:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
default-directory-for-unix	Specifies the default directory to use on a UNIX platform.
Allowable values:	A valid UNIX directory path.
Default value:	""
Notes:	The null string defaults to the current G2 directory.

Attribute	Description
default-directory-for-win32	Specifies the default directory to use on a Win32 platform.
Allowable values:	A valid Win32 directory path
Default value:	""
Notes:	The null string defaults to the current G2 directory.
default-drive	Specifies the default drive to use on a Win32 platform.
Allowable values:	A valid Win32 drive
Default value:	"c:"
reset-settings-with-current-directory-on-ok	A truth-value specifying whether or not to update the settings object with the current directory, and drive if win32, when the user selects the OK button.
Allowable values:	true or false
Default value:	true
Notes:	This is handled by the procedure, g2ui-file-dialog-conclude-action.
user-entered-new-directory-handler	Handles what happens when the user types a new directory specification into the directory edit-box.
Allowable values:	A procedure with the correct argument list (*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window, *EditConfirmed*: truth-value)
Default value:	g2ui-file-user-entered-new-directory

Attribute	Description
user-selected-new-directory-handler	Handles what happens when the user selects a directory in the directories scroll-area.
Allowable values:	A procedure with the correct argument list (*message*: item-or-value, *scroll-area*: item-or-value, *window*: class g2-window)
Default value:	g2ui-file-user-selected-new-directory
user-entered-new-drive-handler	Handles what happens when the user specifies a new drive in the drive edit-box.
Allowable values:	A procedure with the correct argument list (*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window, *EditConfirmed*: truth-value)
Default value:	g2ui-file-user-entered-new-drive

Attributes of g2ui-file-files-ui-settings

Attribute	Description
file-name-edit-box-label	Specifies the text to use for label of the edit-box that contains the file (or module) information.
Allowable values:	Any text value
Default value:	"File Name:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
default-filter	Specifies the default filter text.
Allowable values:	Any text value
Default value:	"*.*"

Attribute	Description
user-entered-new-filter-handler	Handles what happens when the user types a new filter into the file edit-box.
Allowable values:	A procedure with the correct argument list (*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window, *EditConfirmed*: truth-value)
Default value:	g2ui-file-user-entered-new-filter
Notes:	If filtering modules, use g2ui-file-user-entered-new-filter-for-modules.
user-selected-new-file-handler	Handles what happens when the user selects a file, or module, in the files scroll-area.
Allowable values:	A procedure with the correct argument list (*message*: item-or-value, *scroll-area*: item-or-value, *window*: class g2-window)
Default value:	g2ui-file-user-selected-new-file

Attributes of g2ui-file-dialog-settings

Attribute	Description
dialog-text-updated-from	Specifies the text to use for buttons and titles. You can specify the text in the dialog settings object or use a GFR text resource group.
Allowable values:	settings-object or gfr-resource-group
Default value:	settings-object
gfr-resource-group-for-dialog	Specifies the GFR text resource group for the text in buttons and labels.
Allowable values:	Any GFR resource group
Default value:	g2uifile-text-resources

Attribute	Description
file-dialog-setup-handler	Handles the initialization of the controls on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-setup-dialog
Notes:	Called during the dialog launching process by the procedure, g2ui-file-check-for-user-setup-action.
file-dialog-validation-handler	Handles the validation of the information selected and entered by the user on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-validation-for-open-file
Notes:	Called by the procedure, g2ui-file-validation-action, when the user selects the OK or Apply button. Other provided validation handlers are: g2ui-file-validation-for-delete-module g2ui-file-validation-for-merge-module g2ui-file-validation-for-open-file g2ui-file-validation-for-save-file g2ui-file-validation-for-save-module

Attributes of g2ui-file-module-dialog-settings

The following attributes are defined for the following dialogs:

- Save Module
- Save Snapshot
- Warmboot from Snapshot
- Merge Module
- Delete Module

Attribute	Description
module-edit-box-label	Specifies the text to use for label of the edit-box that contains the module file name.
Allowable values:	Any text value
Default value:	"Module File Name:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
remove-references-to-module-in-hierarchy-on-delete	A truth-value specifying whether or not to remove references to the deleted module from the directly-required-modules attributes of all module-information objects.
Allowable values:	true or false
Default value:	true
Notes:	Users are notified when the module hierarchy cannot be resolved automatically.

Attribute	Description
module-filename-text	Specifies the text to use for label of the edit-box that contains the module file name.
Allowable values:	Any text value
Default value:	"Module File Name:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
show-file-progress-display-text	Specifies the text to use for the toggle button for showing the progress of saving a module or snapshot.
Allowable values:	Any text value
Default value:	"Show file progress display"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
save-current-workspace-layout-of-window-text	Specifies the text to use for the toggle button for saving or saving a snapshot of the current layout of workspaces in a module.
Allowable values:	Any text value
Default value:	"Save current workspace layout of window"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
save-module-including-required-modules-text	Specifies the text to use for the toggle button for saving a module with its required modules.
Allowable values:	Any text value
Default value:	"Save module, including required modules"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
resolve-conflicts-automatically-text	Specifies the text to use for the toggle button for resolving conflicts automatically when merging modules.
Allowable values:	Any text value
Default value:	"Resolve conflicts automatically"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
bring-formats-up-to-date-text	Specifies the text to use for the toggle button for bringing up to date the formats of a merged module.
Allowable values:	Any text value
Default value:	"Bring formats up to date"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
install-system-tables-of-merged-kb	Specifies the text to use for the toggle button for installing the system tables of a merged KB.
Allowable values:	Any text value
Default value:	"Install system tables of merged KB"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
delete-associated-workspaces-text	Specifies the text to use for the toggle button for deleting the associated workspaces of the deleted module.
Allowable values:	Any text value
Default value:	"Delete associated workspaces"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
remove-references-to-module-hierarchy-text	Specifies the text to use for the toggle button for removing references to the deleted module in the current module hierarchy.
Allowable values:	Any text value
Default value:	"Remove references to module in hierarchy"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
run-in-catch-up-mode-text	Specifies the text to use for the toggle button for running in catch-up mode when warmbooting.
Allowable values:	Any text value
Default value:	"Run in catch-up mode:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Setup Actions for File Dialog

The object that contains the procedures called while launching the File dialog is g2ui-file-setup-actions. G2ui-file-setup-actions specifies the following procedures in this order:

1. uil-simulate-play-mode
2. g2ui-file-check-for-user-setup-action
3. g2ui-file-clear-ui
4. uil-show-dialog

A user-defined setup handler specified in the g2ui-file-dialog-settings object handles the initialization of these controls. The g2ui-file-check-for-setup-action procedure looks for setup actions in the procedure named in the file-dialog-setup-handler attribute of the associated settings object.

Note For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

File Dialog API

g2ui-file-check-for-user-setup-action

(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure checks the settings object associated with the dialog for a user supplied setup procedure named by the file-dialog-setup-handler attribute.

g2ui-file-clear-ui

(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure clears out the contents of the controls on the file dialog.

g2ui-file-create-file-dialog-settings-object

(*window:* class g2-window)
-> *dialog-settings:* class g2ui-file-dialog-settings

Returns a g2ui-file-dialog-settings object. This object contains attributes for initializing controls on the file dialog, as well as for specifying procedures to override its default behavior.

g2ui-file-dialog-conclude-action

(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called when the user selects the OK or Cancel button on the dialog. The procedure handles post processing for the dialog.

g2ui-file-filter-directories-for-drive

(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called when the user selects the OK button on the Select Drive dialog. The procedure updates the directories scroll-area to show the sub-directories for the current directory path.

g2ui-file-launch-dialog

(*Id*: text, *window:* class g2-window, *settings:* class g2ui-file-dialog-settings)
-> *file-name:* text, *directory-path:* text

Handles launching the dialog whose id matches the specified id on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, g2ui-file-launch-dialog returns the file and directory specified by the user in the dialog. If the user failed to specify either value, the symbol unspecified is returned in its place.

g2ui-file-setup-dialog

(*dialog*: class uil-dialog, *this-initiating-item:* item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization after g2ui-file-clear. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-dialog-settings object that was passed into g2ui-file-launch-dialog.

g2ui-file-user-entered-new-directory

(*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window,
EditConfirmed: truth-value)

Called when the user enters a new directory in the Directories edit-box, and the edit-box then loses focus, either by tabbing out, clicking elsewhere to transfer focus, or selecting the OK button. A side effect of calling this procedure is that the directories scroll-area is updated to reflect the sub-directories of the current directory.

g2ui-file-user-entered-new-drive

(*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window,
EditConfirmed: truth-value)

Called when the user enters a new drive in the Drives edit-box, and the edit-box then loses focus, either by tabbing out, clicking else where to transfer

focus, or selecting the OK button. A side effect of calling this procedure is that the directory edit-box is updated to reflect the new drive.

g2ui-file-user-entered-new-filter
(*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window, *EditConfirmed*: truth-value)

Called when the user enters a new filter in the File edit-box, and the edit-box then loses focus, either by tabbing out, clicking else where to transfer focus, or selecting the OK button. A side effect of calling this procedure is to update the files scroll-area based on the current directory and filter.

g2ui-file-user-selected-new-directory
(*message*: item-or-value, *scroll-area*: item-or-value, *window*: class g2-window)

Called when the user selects a message in the directories scroll-area. A side effect of calling this procedure is that the current directory path is updated, and the files listed in the files scroll-area are updated for the new directory. The current filter is applied to the update.

g2ui-file-user-selected-new-file
(*message*: item-or-value, *scroll-area*: item-or-value, *window*: class g2-window)

Called when the user selects a message in the files scroll-area. A side effect of calling this procedure is that the File edit-box is updated with the selected file name.

g2ui-file-validation-for-open-file
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)

Does validation on the specified directory and file when the OK button is selected on the file dialog.

g2ui-file-validation-for-save-file
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)

Does validation on the specified directory and file when the OK button is selected on the file dialog.

g2ui-file-validation-action
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)

Calls the validation procedure named by the file-dialog-validation-handler attribute of the settings object associated with the dialog.

Save Module Dialog

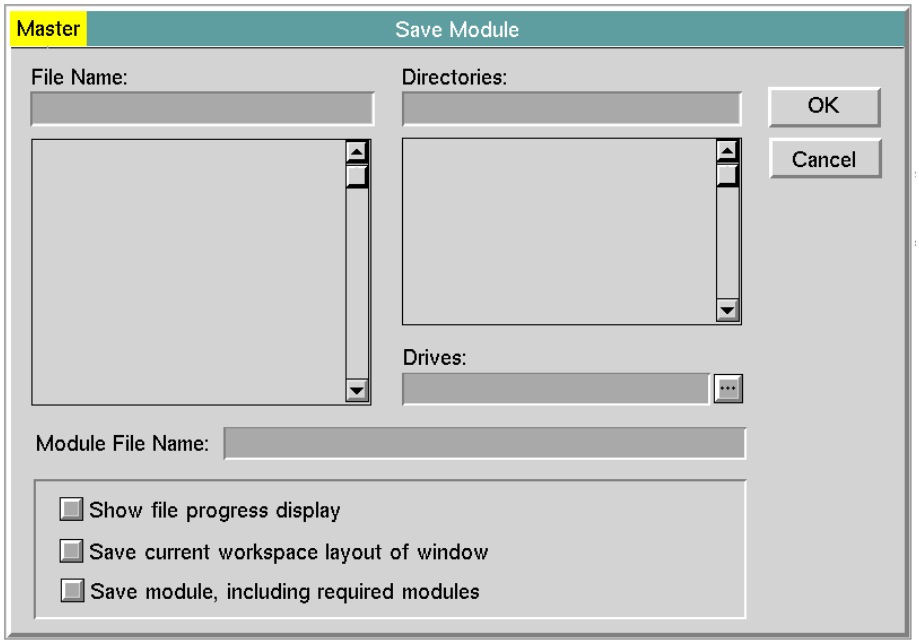
The IDs for the Save Module dialog are:

- g2ui-file-save-module-dialog
- g2ui-file-save-module-pc-dialog

The Save Module dialog enables you to save a module by specifying drive, directory, and file name.

Note If you are using these dialogs on a UNIX system, the field for selecting connected network drives does not appear. The field for connected drives appears only in dialogs for Win32 systems.

The following figure shows the master dialog for the Save Module dialog for PCs:



The IDs of the three Save Module buttons are, in order:

- file-progress-button
- save-workspace-button
- save-required-button

Note Only those IDs for controls that have been added to the basic file dialog control set have been called out.

Save Module Dialog API

g2ui-file-create-file-module-dialog-settings-object

(*window:* class g2-window)
-> *dialog-settings:* class g2ui-file-module-dialog-settings

Returns a g2ui-file-module-dialog-settings object. This object contains attributes for initializing controls on the module dialogs, as well as for specifying procedures to override its default behavior.

g2ui-file-save-module-launch-dialog

(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-module-dialog-settings)
-> *file-name:* text, *directory-path:* text, *module:* symbol,
save-required-modules: truth-value,
save-window-settings-or-not: item-or-value,
display-progress: truth-value)

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, the procedure returns the file, directory and module specified by the user in the dialog, along with specifications of whether to:

- Save the required modules.
- Save the current workspace layout of the window.
- Show the file-progress display.

If the user failed to specify values for directory or file, the symbol *unspecified* is returned in its place. If the user fails to specify a module, the symbol *none* is returned in its place.

g2ui-file-save-module-setup

(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-module-dialog-settings object that was passed into g2ui-file-save-module-launch-dialog.

g2ui-file-user-entered-new-filter-for-modules

(*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window,
EditConfirmed: truth-value)

Called when the user enters a new filter in the Module Name edit-box, and the edit-box then loses focus, either by tabbing out, clicking elsewhere to transfer focus, or selecting the OK button. A side effect of calling this procedure is to update the module's scroll-area based on the current directory and filter.

g2ui-file-validation-for-save-module
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Does validation on the specified directory and module file name when the OK button is selected on the save module dialog.

Other Procedures That Support Save Module Dialog

The following list contains other procedures that operate on Save Module dialog. For more information on these procedures, see File Dialog API.

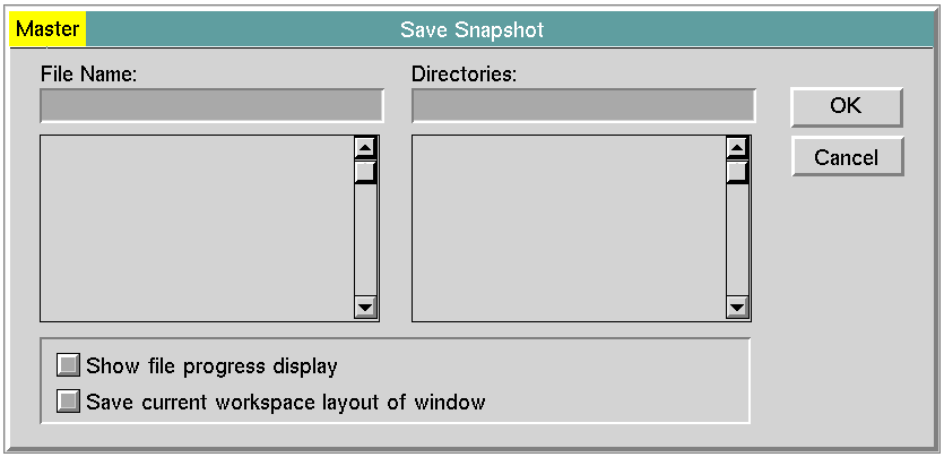
- g2ui-file-clear-ui
- g2ui-file-filter-directories-for-drive
- g2ui-file-setup-dialog
- g2ui-file-user-entered-new-directory
- g2ui-file-user-entered-new-drive
- g2ui-file-user-selected-new-directory
- g2ui-file-user-selected-new-file

Save Snapshot Dialog

The IDs of the Save Snapshot dialog are:

- g2ui-file-save-snapshot-dialog
- g2ui-file-save-snapshot-pc-dialog

The following figure shows the master dialog for the Save Snapshot dialog for PCs:



The IDs of the two Save Snapshot buttons are, in order:

- save-workspace-button
- file-progress-button

Note Only those IDs for controls that have been added to the basic file dialog control set have been called out.

Save Snapshot Dialog API

g2ui-file-save-snapshot-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-module-dialog-settings)
-> *file-name:* text, *directory-path:* text,
display-progress: truth-value, *save-window-settings-or-not*: item-or-value

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in settings. When the dialog is dismissed, the procedure returns the file and directory specified by the user in the dialog, along with specifications of whether to:

- Show the file-progress display.
- Save the current workspace layout of the window.

If the user failed to specify values for directory or file, the symbol unspecified is returned in its place.

Other Procedures That Support Save Snapshot Dialog

The following list contains other procedures that operate on Save Snapshot dialog.

For more information on these procedures, see File Dialog API.

- g2ui-file-clear-ui
- g2ui-file-filter-directories-for-drive
- g2ui-file-setup-dialog
- g2ui-file-user-entered-new-directory
- g2ui-file-user-entered-new-drive
- g2ui-file-user-selected-new-directory
- g2ui-file-user-selected-new-file

For more information on these procedures, see Save Module Dialog API.

- g2ui-file-create-file-module-dialog-settings-object

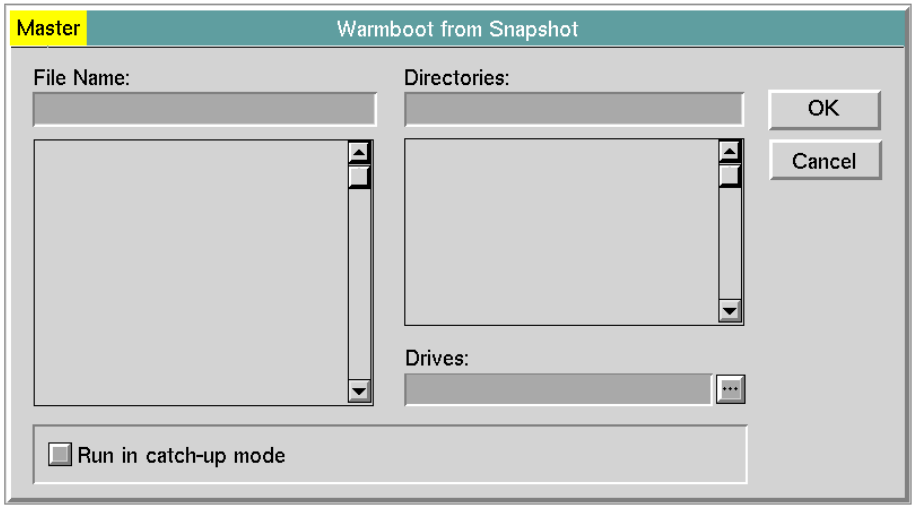
- g2ui-file-user-entered-new-filter-for-modules

Warmboot from Snapshot Dialog

The IDs of the Warmboot from Snapshot dialog are:

- g2ui-file-warmboot-dialog
- g2ui-file-warmboot-pc-dialog

The following figure shows the master dialog for the basic Warmboot from Snapshot dialog for PCs:



The ID of the Warmboot button is:

- catch-up-mode-button

Note Only those IDs for controls that have been added to the basic file dialog control set have been called out.

Warmboot from Snapshot Dialog API

g2ui-file-warmboot-launch-dialog

(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-module-dialog-settings)
-> *file-name:* text, *directory-path:* text,
catch-up-mode: truth-value

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, the procedure returns the file and directory specified by

the user in the dialog, along with specifications of whether to run in catch-up mode.

If the user failed to specify values for directory or file, the symbol *unspecified* is returned in its place.

Other Procedures That Support Warmboot from Snapshot Dialog

The following list contains other procedures that operate on Warmboot from Snapshot dialog.

For more information on these procedures, see File Dialog API.

- g2ui-file-clear-ui
- g2ui-file-filter-directories-for-drive
- g2ui-file-setup-dialog
- g2ui-file-user-entered-new-directory
- g2ui-file-user-entered-new-drive
- g2ui-file-user-selected-new-directory
- g2ui-file-user-selected-new-file

For more information on these procedures, see Save Module Dialog API.

- g2ui-file-create-file-module-dialog-settings-object
- g2ui-file-user-entered-new-filter-for-modules

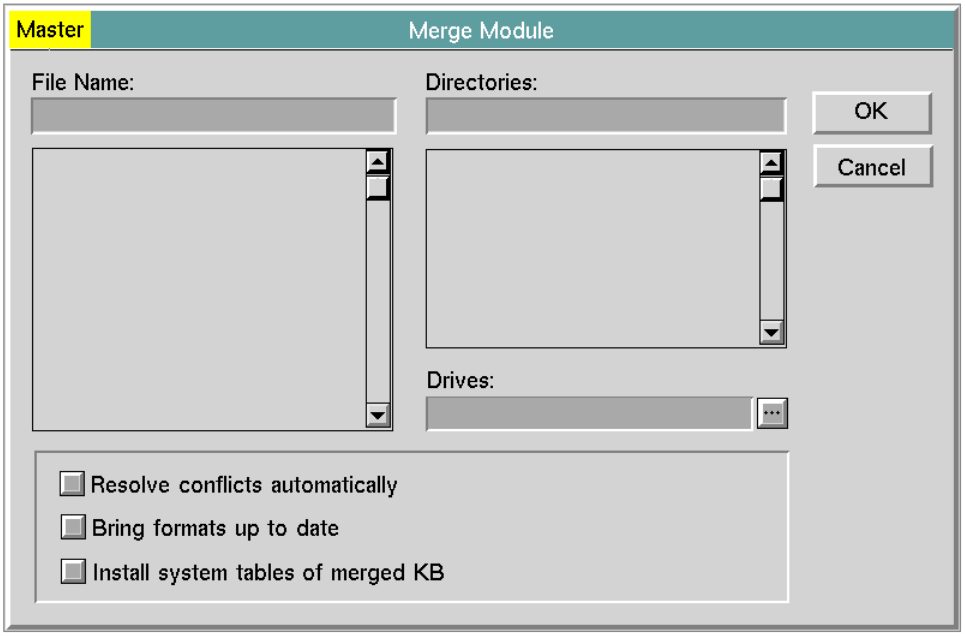
Merge Module Dialog

The IDs for the Merge Module master dialogs are:

- g2ui-file-merge-dialog
- g2ui-file-merge-module-pc-dialog.

Note If you are using these dialogs on a UNIX system, the field for selecting connected network drives does not appear. The field for connected drives appears only in dialogs for Win32 systems.

The following figure shows the master dialog for Merge Module:



The IDs of the three buttons are, in order:

- conflicts-button
- formats-button
- system-tables-button

Note Only those IDs for controls that have been added to the basic file dialog control set have been called out. For information regarding the ids of the other controls, see File Dialog.

Merge Module API

g2ui-file-merge-module-launch-dialog

(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-module-dialog-settings)
-> *file-name:* text, *directory-path:* text, *resolve-conflicts*: truth-value,
update-formats: truth-value, *install-system-tables*: truth-value)

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, g2ui-file-merge-module-launch-dialog returns the file and directory specified by the user in the dialog, along with specifications for whether to:

- Resolve conflicts automatically.
- Bring internal formats up-to-date.
- Install the system tables of the merged module.

If the user failed to specify values for directory or file, the symbol unspecified is returned in its place.

g2ui-file-merge-module-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-module-dialog-settings object that was passed into g2ui-file-merge-module-launch-dialog.

g2ui-file-validation-for-merge-module
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Does validation on the specified directory and module file name when the OK button is selected on the Merge Module dialog.

Other Procedures That Support Merge Module Dialogs

The following list contains other procedures that operate on merge module dialogs.

For more information on these procedures, see File Dialog API.

- g2ui-file-clear-ui
- g2ui-file-filter-directories-for-drive
- g2ui-file-setup-dialog
- g2ui-file-user-entered-new-directory
- g2ui-file-user-entered-new-drive
- g2ui-file-user-selected-new-directory
- g2ui-file-user-selected-new-file

For more information on these procedures, see Save Module Dialog API.

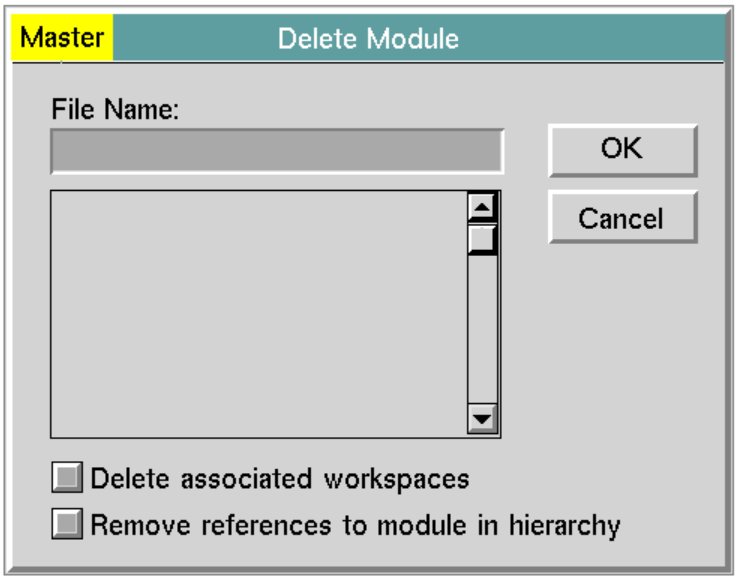
- g2ui-file-create-file-module-dialog-settings-object
- g2ui-file-user-entered-new-filter-for-modules

Delete Module Dialog

The ID for the Delete Module master dialog is:

- g2ui-file-delete-module-dialog

The following figure shows the master dialog for Delete Module:



The IDs of the Delete Module dialog buttons are:

- delete-workspaces-button
- remove-references-button

Note Only those IDs for controls that have been added to the basic file dialog control set have been called out. For information regarding the ids of the other controls, see File Dialog.

Delete Module API

g2ui-file-delete-module-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-module-dialog-settings)
-> *module-name:* text, *delete-workspaces:* truth-value,
resolve-hierarchy: truth-value

Launches the dialog whose id matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, the procedure returns:

- The module to delete.
- A value which specifies whether or not to delete associated workspaces for the deleted module as well.
- A value which specifies whether or not to remove references to the deleted module from the module hierarchy.

If the user failed to specify a value for the module, the symbol *unspecified* is returned in its place.

g2ui-file-delete-module-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-module-dialog-settings object that was passed into g2ui-file-delete-module-launch-dialog.

g2ui-file-validation-for-delete-module
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Does validation on the specified module name when the OK button is selected on the delete module dialog.

Other Procedures That Support Delete Module Dialogs

The following list contains other procedures that operate on delete module dialogs. For more information on these procedures, see File Dialog API.

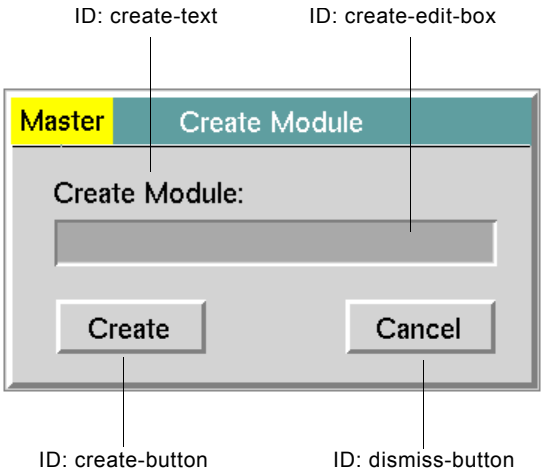
- g2ui-file-clear-ui
- g2ui-file-setup-dialog
- g2ui-file-user-selected-new-file

For more information on these procedures, see Save Module Dialog API.

- g2ui-file-create-file-module-dialog-settings-object
- g2ui-file-user-entered-new-filter-for-modules

Create Module Dialog

The ID for the Create Module master dialog is g2ui-file-create-module-dialog. The following figure shows the master dialog for Create Module:



Attributes of g2ui-file-create-module-dialog-settings

Attribute	Description
use-module-dialog-preferences	Specifies that the colors of all dialogs are taken from the preferences object, g2uifile-dialogs-module-settings, highest in the module hierarchy. A value of false means that you can configure the colors of a particular master dialog to whatever you want.
Allowable values:	true or false
Default value:	true
dialog-text-updated-from	Specifies the text to use for buttons and titles. You can specify the text in the dialog settings object or use a GFR text resource group.
Allowable values:	settings-object or gfr-resource-group
Default value:	settings-object

Attribute	Description
gfr-resource-group-for-dialog	Specifies the GFR text resource group for the text in buttons and labels.
Allowable values:	Any GFR resource group
Default value:	g2uifile-text-resources
dialog-title	Specifies the text to use as the dialog's title.
Allowable values:	Any text value
Default value:	"Create Module"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
dialog-title-justification	Specifies the justification for the text of the dialog's title.
Allowable values:	center, left, right
Default value:	left
dialog-title-configuration	Specifies a uil-configuration object to use to configure the dialog's title.
Allowable values:	The name of a uil-text-configuration-class object
Default value:	uil-dialog-title-configuration
create-text	Specifies the text to use as the edit-box prompt.
Allowable values:	Any text value
Default value:	"Module Name:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
create-button-label	Specifies the text to use for the Create button label.
Allowable values:	Any text value
Default value:	"Create"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
cancel-button-label	Specifies the text to use for the Cancel button label.
Allowable values:	Any text value
Default value:	"Cancel"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
button-layout-style	Specifies the layout style for the dialog's Create and Cancel buttons.
Allowable values:	motif-style or windows-style
Default value:	windows-style
create-module-dialog-setup-handler	Handles the initialization of the controls on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-create-module-setup

Setup Actions for Create Module Dialog

The object that contains the procedures called while launching the Create Module dialog is g2ui-file-create-module-setup-actions. G2ui-file-create-module-setup-actions specifies the following procedures:

- g2ui-file-create-module-check-for-user-setup-action (calls procedure named in settings object)
- uil-simulate-play-mode
- uil-show-dialog

For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

Create Module API

g2ui-file-create-create-module-dialog-settings-object
(*window:* class g2-window)
-> *dialog-settings:* class g2ui-file-create-module-dialog-settings

Returns a g2ui-file-create-module-dialog-settings object. This object contains attributes for initializing controls on the create module dialog, as well as for specifying procedures to override its default behavior.

g2ui-file-create-module-check-for-user-setup-action
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure checks the settings object associated with the dialog for a user supplied setup procedure named by the create-module-dialog-setup-handler attribute.

g2ui-file-create-module-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-create-module-dialog-settings)
-> *module-name:* symbol

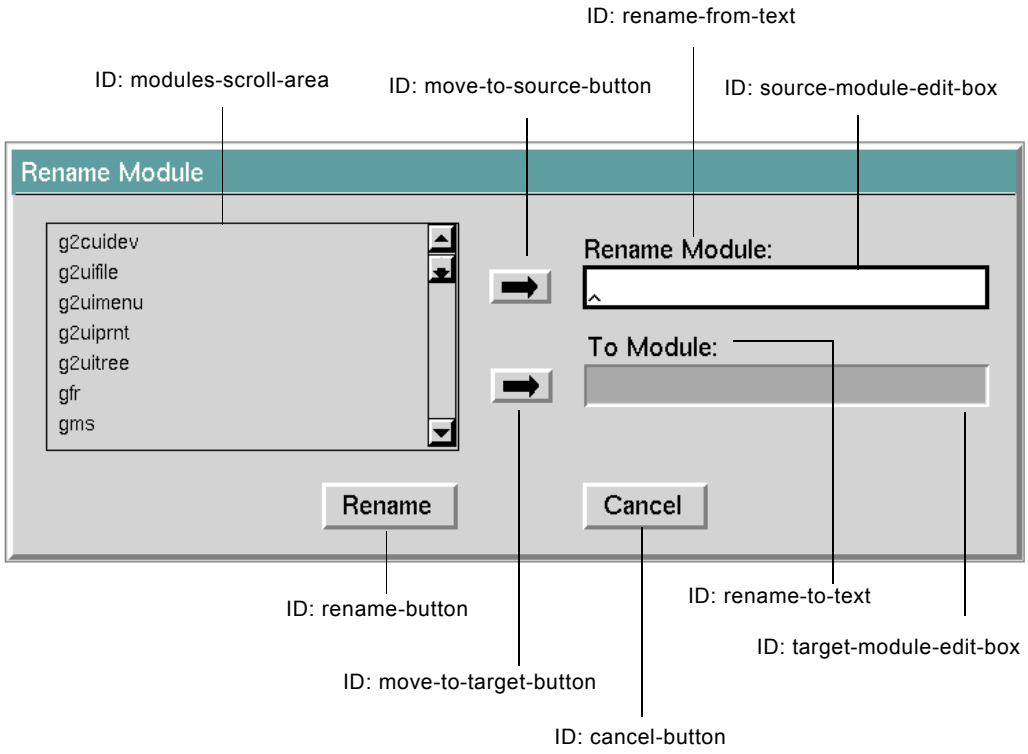
Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, g2ui-file-create-module-launch-dialog returns the name of the module to create. If the user failed to specify a value for the module, the symbol *unspecified* is returned in its place.

g2ui-file-create-module-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-create-module-dialog-settings object that was passed into g2ui-file-create-module-launch-dialog.

Rename Module Dialog

The ID for the Rename Module master dialog is g2ui-file-rename-module-dialog. The following figure shows the master dialog and IDs for Rename Module:



Attributes of g2ui-file-rename-module-dialog-settings

Attribute	Description
use-module-dialog-preferences	Specifies that the colors of all dialogs are taken from the preferences object, g2uifile-dialogs-module-settings, highest in the module hierarchy. A value of false means that you can configure the colors of a particular master dialog to whatever you want.
Allowable values:	true or false
Default value:	true
dialog-text-updated-from	Specifies the text to use for buttons and titles. You can specify the text in the dialog settings object or use a GFR text resource group.
Allowable values:	settings-object or gfr-resource-group
Default value:	settings-object
gfr-resource-group-for-dialog	Specifies the GFR text resource group for the text in buttons and labels.
Allowable values:	aNy GFR resource group
Default value:	g2uifile-text-resources
dialog-title	Specifies the text to use as the dialog's title.
Allowable values:	Any text value
Default value:	"Rename Module"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
dialog-title-justification	Specifies the justification for the text of the dialog's title.
Allowable values:	center, left, right
Default value:	left
dialog-title-configuration	Specifies a uil-configuration object to use to configure the dialog's title.
Allowable values:	The name of a uil-text-configuration-class object
Default value:	uil-dialog-title-configuration
source-edit-box-label	Specifies the text to use as the source module's edit-box label.
Allowable values:	Any text value
Default value:	"Rename Module:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
target-edit-box-label	Specifies the text to use for the target module's edit-box label.
Allowable values:	Any text value
Default value:	"To Module:"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
rename-button-label	Specifies the text to use for the Rename button label.
Allowable values:	Any text value
Default value:	"Rename"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
cancel-button-label	Specifies the text to use for the Cancel button label.
Allowable values:	Any text value
Default value:	"Cancel"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
button-layout-style	Specifies the layout style for the dialog's Create and Cancel buttons.
Allowable values:	motif-style or windows-style
Default value:	windows-style
prompt-for-creating-new-module-if-doesnt-exist	A truth-value specifying whether or not to prompt the user for creating the target module if it does not already exist.
Allowable values:	true or false
Default value:	true
Notes:	If false, and the target module does not exist, the module will be created without a confirmation from the user.

Attribute	Description
move-selected-module-handler	Handles what happens when the user clicks on either of the right arrow icon buttons.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-rename-module-move-selected-module
rename-module-dialog-setup-handler	Handles the initialization of the controls on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-rename-module-setup
rename-module-dialog-validation-handler	Handles the validation of the information selected and entered by the user on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-validation-for-rename-module
Notes:	Called by the procedure, g2ui-file-rename-module-validation-action, when the user selects the Rename button.

Setup Actions for Rename Module Dialog

The g2ui-file-rename-module-setup-actions object contains the procedures called while launching the Rename Module dialog. G2ui-file-rename-module-setup-actions specifies the following procedures:

- g2ui-file-rename-module-check-for-user-setup-action (calls procedure named in settings object)
- uil-simulate-play-mode
- uil-show-dialog

For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

Rename Module API

g2ui-file-rename-module
(*source-module*: symbol, *target-module*: symbol, *window*: class g2-window)

Reassigns all the top-level workspaces that reside in *source-module* to reside in *target-module*. When done, the user is prompted as to whether the source-module, and its associated workspaces, should be deleted.

g2ui-file-create-rename-module-dialog-settings-object
(*window:* class g2-window)
-> *dialog-settings:* class g2ui-file-rename-module-dialog-settings

Returns a g2ui-file-rename-module-dialog-settings object. This object contains attributes for initializing controls on the Rename Module dialog, as well as for specifying procedures to override its default behavior.

g2ui-file-rename-module-check-for-user-setup-action
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure checks the settings object associated with the dialog for a user supplied setup procedure named by the rename-module-dialog-setup-handler attribute.

g2ui-file-rename-module-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-rename-module-dialog-settings)
-> *source-module:* text, *target-module:* text

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, g2ui-file-rename-module-launch-dialog returns the names of the source and target modules.

g2ui-file-rename-module-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-rename-module-dialog-settings object that was passed into g2ui-file-rename-module-launch-dialog.

g2ui-file-rename-module-move-selected-module
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called when one of the arrow buttons is clicked on by the user. The procedure copies the text of the selected message into the corresponding edit field.

g2ui-file-rename-module-validation-action
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Calls the validation procedure named by the rename-module-dialog-validation-handler attribute of the settings object associated with the dialog.

g2ui-file-validation-for-rename-module
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

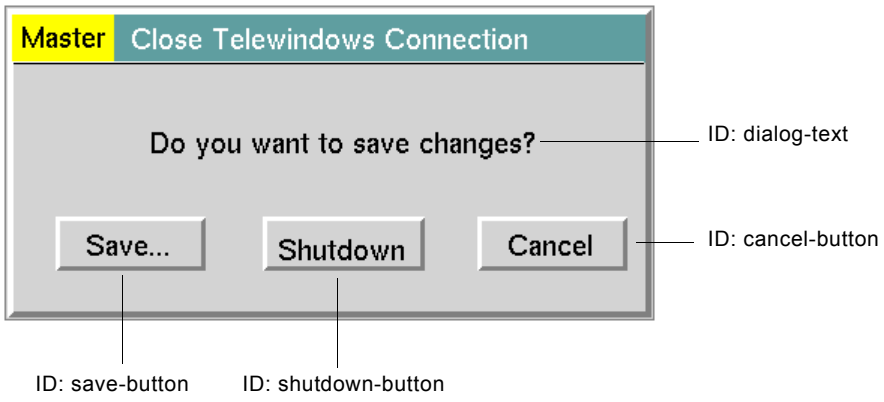
Does validation on the specified module names when the Rename button is selected on the rename module dialog.

Close and Shutdown Dialogs

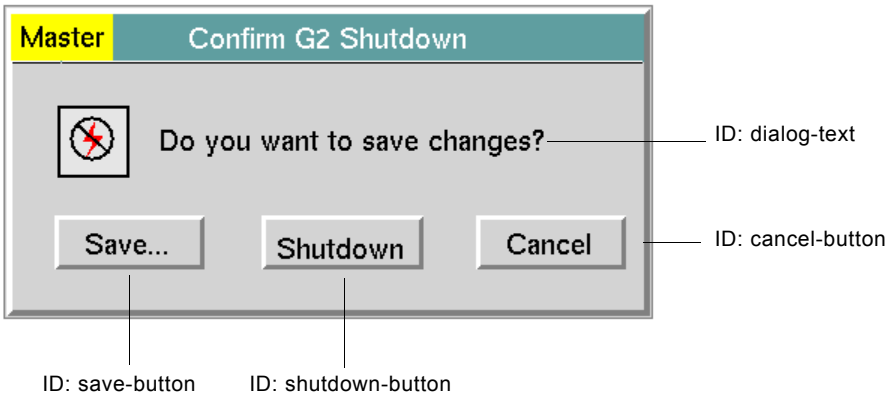
The IDs for the Close Telewindows Connection and Confirm G2 Shutdown master dialogs are:

- g2ui-file-confirm-tw-shutdown-dialog
- g2ui-file-confirm-shutdown-dialog

The master dialog for closing a Telewindows connection is identical to the G2 shutdown dialog, except that it does not contain the icon symbolizing connection termination:



The following figure shows the master dialog for confirming a G2 shutdown:



Attributes of g2ui-file-confirm-shutdown-dialog-settings

Attribute	Description
use-module-dialog-preferences	Specifies that the colors of all dialogs are taken from the preferences object, g2uifile-dialogs-module-settings, highest in the module hierarchy. A value of false means that you can configure the colors of a particular master dialog to whatever you want.
Allowable values:	true or false
Default value:	true
dialog-text-updated-from	Specifies the text to use for buttons and titles. You can specify the text in the dialog settings object or use a GFR text resource group.
Allowable values:	settings-object or gfr-resource-group
Default value:	settings-object
gfr-resource-group-for-dialog	Specifies the GFR text resource group for the text in buttons and labels.
Allowable values:	any GFR resource group
Default value:	g2uifile-text-resources
dialog-title	Specifies the text to use as the dialog's title.
Allowable values:	Any text value
Default value:	"Confirm G2 Shutdown"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
dialog-title-justification	Specifies the justification for the text of the dialog's title.
Allowable values:	center, left, right
Default value:	left
dialog-title-configuration	Specifies a uil-configuration object to use to configure the dialog's title.
Allowable values:	The name of a uil-text-configuration-class object
Default value:	uil-dialog-title-configuration
dialog-text	Specifies the text to use for the dialog prompt.
Allowable values:	Any text value
Default value:	"Do you want to save changes?"
Notes:	.For localization, set the value of this attribute through the use of GFR's techniques.
save-button-label	Specifies the text to use for the Save button's label.
Allowable values:	Any text value
Default value:	"Save..."
Notes:	.For localization, set the value of this attribute through the use of GFR's techniques.

Attribute	Description
shutdown-button-label	Specifies the text to use for the Shutdown button's label.
Allowable values:	Any text value.
Default value:	"Shutdown"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques .
cancel-button-label	Specifies the text to use for the Cancel button's label.
Allowable values:	Any text value
Default value:	"Cancel"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques .
button-layout-style	Specifies the layout style for the dialog's OK, Apply and Cancel buttons.
Allowable values:	motif-style or windows-style
Default value:	windows-style
confirm-shutdown-dialog-setup-handler	Handles the initialization of the controls on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)
Default value:	g2ui-file-confirm-shutdown-dialog-setup

Setup Actions for Confirm Shutdown Dialog

The g2ui-file-confirm-shutdown-setup-actions object contains the procedures called while launching the Confirm Shutdown dialog. G2ui-file-confirm-shutdown-setup-actions specifies the following procedures:

- g2ui-file-confirm-shutdown-check-for-user-setup-actions (calls procedure named in settings object)
- uil-simulate-play-mode
- uil-show-dialog

For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

Confirm Shutdown API

g2ui-file-confirm-shutdown-check-for-user-setup-action
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure checks the settings object associated with the dialog for a user supplied setup procedure named by the confirm-shutdown-dialog-setup-handler attribute.

g2ui-file-confirm-shutdown-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-confirm-shutdown-dialog-settings)
-> *user-requested-action:* symbol

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, g2ui-file-confirm-shutdown-launch-dialog returns a symbol indicating the requested user action: save-kb, shutdown-g2, or cancel-shutdown.

g2ui-file-confirm-shutdown-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: class g2-window, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during dialog initialization. The procedure initializes the controls on the dialog based on the values found in the g2ui-file-confirm-shutdown-dialog-settings object that was passed into g2ui-file-confirm-shutdown-launch-dialog or g2ui-file-confirm-tw-shutdown-launch-dialog.

g2ui-file-confirm-tw-shutdown-launch-dialog
(*dialog-id*: text, *window*: class g2-window,

settings: class g2ui-file-confirm-shutdown-dialog-settings)
-> *user-requested-action:* symbol

Handles launching the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in *settings*. When the dialog is dismissed, g2ui-file-confirm-tw-shutdown-launch-dialog returns a symbol indicating the requested user action: save-kb, shutdown-g2, or cancel-shutdown.

g2ui-file-create-confirm-shutdown-dialog-settings-object
(*window:* class g2-window)
-> *dialog-settings:* class g2ui-file-confirm-shutdown-dialog-settings

Returns a g2ui-file-confirm-shutdown-dialog-settings object. This object contains attributes for initializing controls on the confirm shutdown dialog.

5

Customizing the Tree View Control Dialog

Describes the tree-view capabilities provided by the G2UITREE module.

Introduction

The g2uitree G2 module delivers to the G2 developer a fully customizable and localizable tree-view control as well as class and workspace browser dialogs. The tree-view control can live on a workspace or on a dialog.

The g2uitree utility is built from the following GUIDE/UIL classes:

- uil-scroll-area
- uil-message-object
- uil-appended-item

Note Only advanced users of GUIDE/UIL should customize the dialogs provided with GDI.

Displaying the G2UITREE Palette and Dialogs

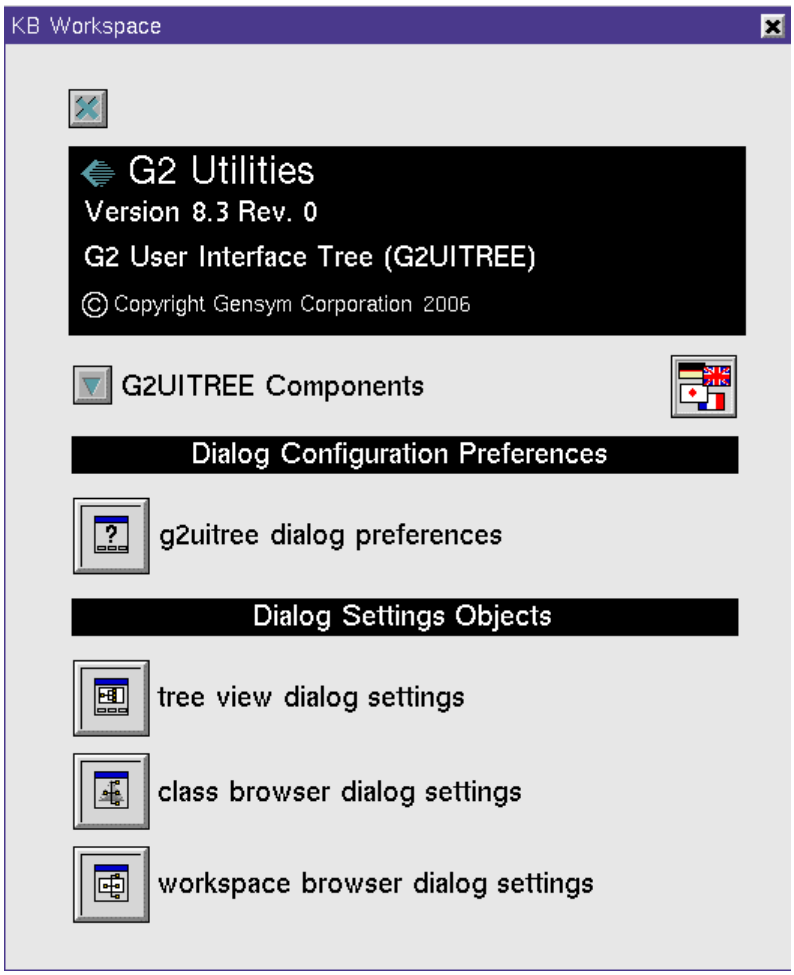
To navigate to the top-level workspace for g2uitree:

1 Display the GDI top-level workspace.

See Displaying GDI Top-Level Workspaces.

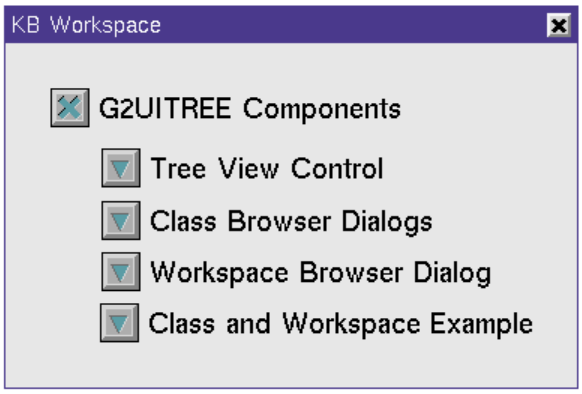
2 Click the G2UITREE Top Level button.

The following figure shows the top-level workspace for g2uitree.



To access the master dialogs for the tree dialogs:

➔ Click the G2UITREE Components button to display this dialog:



There are subworkspaces for the Tree View Control, Class Browser dialogs, and Workspace Browser dialog. The subworkspaces contain the master dialogs that you can clone and customize. The Class and Workspace Example workspace contains examples of the Class and Workspace Browsers.

For each dialog, this chapter presents information in the following four categories:

- **Master dialogs –** Each dialog has master dialog and an API that supports launching, initializing, and dismissing a dialog. The API consists of dialog settings objects, action description arrays, and procedures. To customize a dialog, first clone the master dialog and transfer it to your application. Using the settings object, you can change the appearance and text of a dialog. For more complex customizations, you can add or remove controls and override default handlers.
- **Dialog settings objects –** A dialog settings object is an argument passed to the API call that launches the dialog. The settings object is the initializing object of the dialog and contains attributes for the dialog's title, the button labels, localized text, and default values of controls. It specifies setup, validation, and handlers. To customize a settings object, clone it from the GDI palette. To add additional attributes, you can subclass the settings object class.
- **Setup actions –** Most dialogs have an associated uil-description-array that describes the actions that are executed when the dialog is launched. Modify the action description array only to remove actions. To add actions, use your copy of a dialog settings object.
- **API calls –** Each dialog has supporting API calls for launching, initializing, validating, and dismissing a dialog. The API calls handle the user interface and do not perform operations. They return information necessary to perform operations. For example, the g2ui-file-create-module-launch-dialog returns the name of the module to create; it does not create the module.

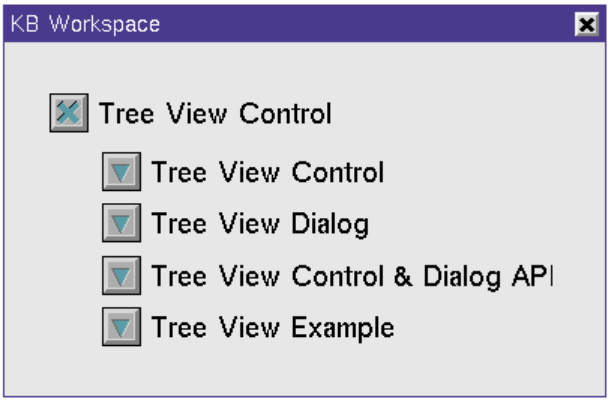
Caution Application developers are free to clone the master dialogs, remove controls, and add their own controls. *However, it is extremely important that the IDs for the controls that come with the master dialog are not changed.* If IDs change, the underlying code that coordinates the update and display of files, directories, and modules will fail because the ID maps the code to the user interface.

Tree View Control and Dialog

The Tree View Control dialog enables you to present information to users in a hierarchical view. For example, you can use the tree view control to view a hierarchy of modules, classes, or workspaces.

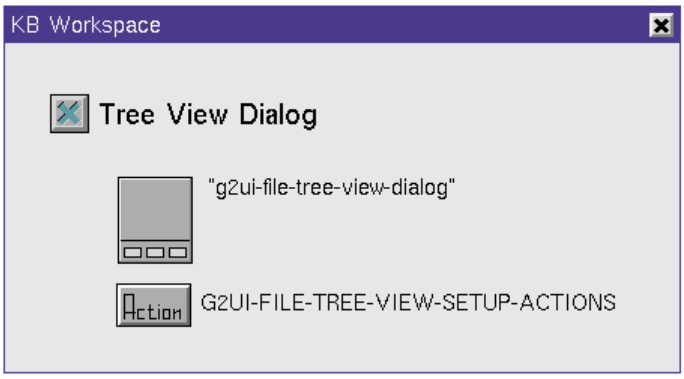
The Tree View Control dialog is similar to the file and module dialogs in the g2uifile module. An API launch procedure uses a settings object that specifies default values and the names of handler procedures.

The following figure shows the top-level workspace that enables you to navigate to the components of the Tree View Control.

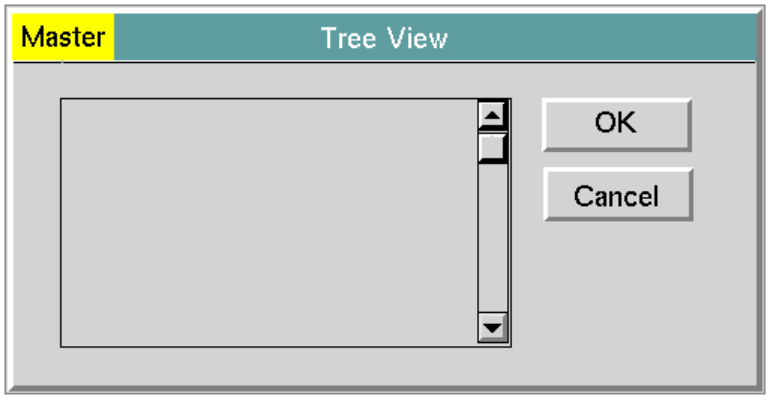


The top-level workspace enables you to navigate to subworkspaces containing the tree view control, the master dialog, the API, and an example.

Selecting the Tree View Control navigation button displays the workspace containing the tree-view control itself. Selecting the Tree View Control Dialog navigation button displays the workspace containing the master dialog, as shown in the following figure:



Displaying the subworkspace for the dialog gives you the master dialog for the tree view control, as shown in the following figure:



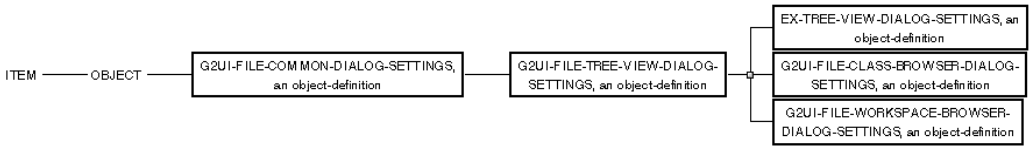
To modify the Tree View Control or dialog, see Modifying GDI Menus and Dialogs, especially the sections on palettes, using dialogs "as is," and adding and removing controls. To see how you can use the Tree View Control, see Class Browser Dialogs and Workspace Browser Dialog.

Attributes

The Tree View Control dialog attributes belong to three classes of settings objects. The g2ui-file-tree-view-dialog-settings and ex-tree-view-dialog-settings settings objects inherit attributes from g2ui-file-common-dialog-settings:

- g2ui-file-common-dialog-settings
- g2ui-file-tree-view-dialog-settings
- ex-tree-view-dialog-settings

The following figure shows the class hierarchy of the dialog settings objects for the Tree View Control dialog. The settings objects for the class and workspace browser dialogs are also depicted.



For more information on dialog settings objects, see Modifying Dialog Text.

Attributes of g2ui-file-common-dialog-settings

Attribute	Description
use-module-dialog-preferences	Specifies that the colors of all dialogs are taken from the preferences object, g2uitree-dialogs-module-settings, highest in the module hierarchy. A value of false means that you can configure the colors of a particular master dialog to whatever you want.
Allowable values:	true or false
Default value:	true

Attribute	Description
dialog-title	Specifies the text to use as the dialog's title.
Allowable values:	Any text value
Default value:	"Open File"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
dialog-title-justification	Specifies the justification for the text of the dialog's title.
Allowable values:	center, left, right
Default value:	left
ok-button-label	Specifies the text to use for the OK button's label.
Allowable values:	Any text value
Default value:	"OK"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
apply-button-label	Specifies the text to use for the Apply button's label.
Allowable values:	Any text value
Default value:	"Apply"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques .

Attribute	Description
cancel-button-label	Specifies the text to use for the Cancel button's label.
Allowable values:	Any text value
Default value:	"Cancel"
Notes:	For localization, set the value of this attribute through the use of GFR's techniques.
button-layout-style	Specifies the layout style for the dialog's OK, Apply and Cancel buttons.
Allowable values:	motif-style or windows-style
Default value:	windows-style

Attributes of g2ui-file-tree-view-dialog-settings

Attribute	Description
dialog-text-updated-from	Specifies the text to use for buttons and titles.
Allowable values:	settings-object or gfr-resource-group
Default value:	settings-object
gfr-resource-group-for-dialog	Specifies the GFR test resource group.
Allowable values:	any GFR resource group
Default value:	g2uitree-text-resources

Attribute	Description
tree-view-dialog-setup-handler	Handles the initialization of controls on the dialog.
Allowable values:	A procedure with the correct argument list (*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: class g2-window, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list).
Default value:	g2ui-file-tree-view-setup
clear-tree-view-control-on-setup	A truth-value specifying whether or not to update the settings object with the default node, when the user selects the OK button.
Allowable values:	true or false
Default value:	true
user-selected-new-node-handler	Handles what happens when the user selects a node in the node scroll-area.
Allowable values:	A procedure with the correct argument list (*message*: item-or-value, *scroll-area*: item-or-value, *window*: class g2-window)
Default value:	g2ui-file-tree-view-user-selected-new-node

Attributes of ex-tree-view-dialog-settings

Attribute	Description
node-tree	Specifies the type of node tree.
Allowable values:	Any object
Default value:	a g2ui-file-tree-view-object

Setup Actions for Tree View Control Dialog

The object that contains the procedures called while launching the File dialog is g2ui-file-tree-view-setup-actions. G2ui-file-tree-view-setup-actions specifies the following procedures:

- uil-simulate-play-mode
- uil-show-dialog
- g2ui-file-tree-view-check-for-user-setup-action

A user-defined setup handler specified in the g2ui-file-tree-view-dialog-settings object handles the initialization of these controls. The g2ui-file-tree-view-check-for-setup-action procedure looks for setup actions in the procedure named in the tree-dialog-setup-handler attribute of the associated settings object.

Note For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

Tree View Control Dialog API

g2ui-file-tree-view-check-for-user-setup-action
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: item-or-value, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)

Called during initialization. The procedure checks the settings object associated with the dialog for a user supplied setup procedure named by the tree-view-dialog-setup-handler attribute.

g2ui-file-tree-view-collapse-leaves-for-node
(*node*: class g2ui-file-tree-view-object, *message-object*: item-or-value, *window*: class g2-window)

The g2ui-file-tree-view-collapse-leaves-for-node call is a method on g2ui-file-tree-view-object. The settings object for dialogs containing tree-view controls is g2ui-file-tree-view-dialog-settings. This settings object contains an attribute, node-tree, which is used to track the actual information displayed in the tree-view control. The default method for g2ui-file-tree-view-object does a recursive delete of the uil-message-objects in the tree-view control from the specified node to its branches.

g2ui-file-tree-view-cleanup-action dialog
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: item-or-value, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)

Calls g2ui-file-tree-view-initialize-settings-object on the settings object that is the *initiating-object* for the specified dialog. This is called automatically when the user selects the OK or Cancel button on any of the tree-view dialogs.

g2ui-file-tree-view-get-node

(*id*: text)
-> *node*: item-or-value

Returns the node from the node-tree of the settings object whose ID matches the specified ID. The user-data attribute of the selected message in the tree-view control provides the ID value.

g2ui-file-tree-view-get-leaves-for-node

(*node*: class g2ui-file-tree-view-object, *window*: class g2-window)

The g2ui-file-tree-view-get-leaves-for-node call is a method on g2ui-file-tree-view-object. The settings object for dialogs containing tree-view controls is g2ui-file-tree-view-dialog-settings. This settings object contains an attribute, node-tree, which is used to track the actual information displayed in the tree-view control.

The default method for g2ui-file-tree-view-object checks the node-type of the specified node and expects *one* of the following values:

- The symbol class
- The symbol workspace

The g2ui-file-tree-view-get-leaves-for-node call then dispatches to either g2ui-file-tree-view-get-leaves-for-class or g2ui-file-tree-view-get-leaves-for-workspace.

g2ui-file-tree-view-initialize-settings-object

(*Settings*: class g2ui-file-tree-view-dialog-settings)
-> *Settings*: class g2ui-file-tree-view-dialog-settings

Resets the *Settings* object associated with a dialog to its initial state.

g2ui-file-tree-view-launch-dialog

(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-tree-view-dialog-settings)
-> *selected-text*: text

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in settings. When the dialog is dismissed, g2ui-file-tree-view-launch-dialog returns the text of the selected message in the tree-view control.

g2ui-file-tree-view-setup

(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: item-or-value, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

Called during initialization. This procedure initializes the controls on the dialog based on values found in the g2ui-file-tree-view-dialog-settings object that was passed into g2ui-file-tree-view-launch-dialog.

g2ui-file-tree-view-user-selected-node
(*message-object*: item-or-value, *scroll-area*: item-or-value, *window*: class g2-window)

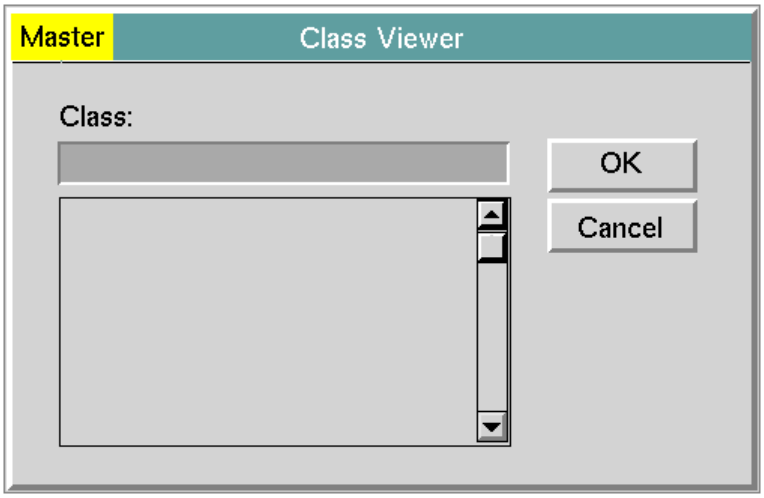
Called when the user selects a message in the tree-view control. Depending upon the state of the selected message in the tree-view control, the message is either expanded or collapsed.

Expanding and collapsing messages is methodized and handled by the following methods:

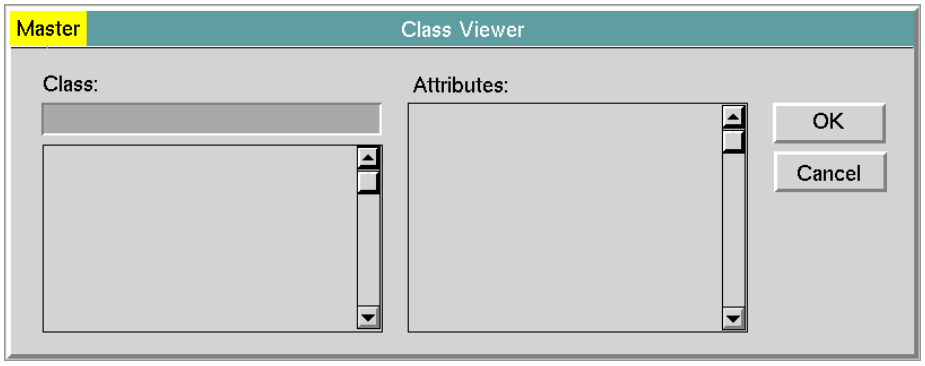
- g2ui-file-tree-view-get-leaves-for-node
- g2ui-file-tree-view-collapse-leaves-for-node

Class Browser Dialogs

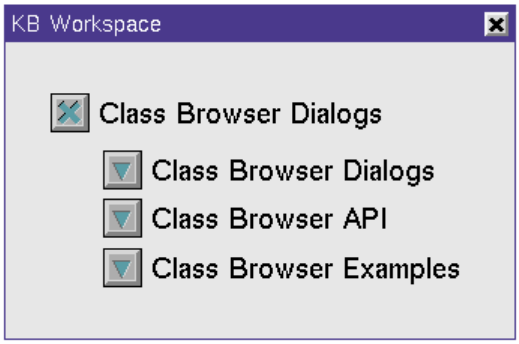
The Class Browser enables you to select classes, as shown in the following figure:



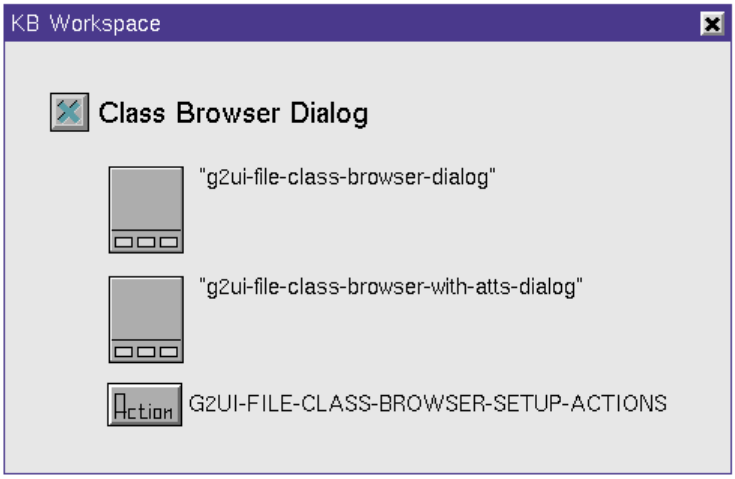
The Class Browser can also display attributes. The following figure shows the dialog that enables you to display attributes of a specified class:



The following figure shows the top-level workspace that enables you to navigate to the components of the Class Browser:



Selecting the Class Browser Dialogs navigation button displays the workspace containing the master dialogs for Class Browser and Class Browser with attributes, as shown in the following figure.



Attributes

The Class Browser dialog attributes belong to these classes:

- g2ui-file-common-dialog-settings
- g2ui-file-tree-view-dialog-settings
- g2ui-file-class-browser-dialog-settings

The g2ui-file-class-browser-dialog-settings class inherits from the g2ui-file-tree-view-dialog-settings class, which inherits from the g2ui-file-common-dialog-settings class.

For information on the attributes for g2ui-file-common-dialog-settings, see Attributes of g2ui-file-common-dialog-settings. For more information on the attributes for g2ui-file-tree-view-dialog-settings, see Attributes of g2ui-file-tree-view-dialog-settings. For more information on dialog settings, see the Modifying Dialog Text.

Attributes of g2ui-file-class-browser-dialog-settings

Attribute	Description
initial-class	Specifies the class at the top of the tree.
Allowable values:	any class
Default value:	g2
class-name-edit-box-label	The label for the class edit-box, which enables you to type in the class name
Allowable values:	any text
Default value:	"Class:"
attributes-scroll-area-label	The label for the scroll area where you can scroll through the list of attributes
Allowable values:	any text
Default value:	"Attributes:"

Attribute	Description
user-entered-new-class-handler	Handles what happens when the user types a new class specification into the class edit-box
Allowable values:	A procedure with the correct argument list (*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window, *EditConfirmed*: truth-value)
Default value:	g2ui-file-class-browser-user-entered-new-node

Setup Actions for Class Browser Dialog

The object that contains the procedures called while launching the Class Browser dialog is g2ui-file-class-browser-setup-actions. G2ui-file-class-browser-setup-actions specifies the following procedures:

- uil-simulate-play-mode
- uil-show-dialog
- g2ui-file-tree-view-check-for-user-setup-action

A user-defined setup handler specified in the g2ui-file-tree-view-dialog-settings object handles the initialization of these controls. The g2ui-file-tree-view-check-for-setup-action procedure looks for setup actions in the procedure named in the tree-dialog-setup-handler attribute of the associated settings object.

Note For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

Class Browser API

g2ui-file-class-browser-create-settings-object
(*window*: class g2-window)
-> *settings*: class g2ui-file-class-browser-dialog-settings

Returns a g2ui-file-class-browser-dialog-settings object. This object contains attributes for initializing controls on the class browser dialog.

g2ui-file-class-browser-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-file-class-browser-dialog-settings)
-> *root-class*: text, *selected-text*: text, *selected-attribute*: text

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in settings. When the

dialog is dismissed, g2ui-file-class-browser-launch-dialog returns three text values, the root class for the selected class, the selected class, and the selected attribute. If no class or attribute is selected, g2ui-file-class-browser-launch-dialog returns empty strings.

g2ui-file-class-browser-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value, *window*: item-or-value, *original-initiating-item*: item-or-value, *dialog-actions*: class symbol-list)

Called during initialization. This procedure initializes the controls on the dialog based on values found in the g2ui-file-class-browser-settings object that was passed into g2ui-file-class-browser-launch-dialog. This procedure first calls g2ui-file-tree-view-setup.

g2ui-file-class-browser-user-entered-new-node
(*edit-box*: item-or-value, *dialog*: item-or-value, *window*: class g2-window, *edit-confirmed*: truth-value)

Called when the edit-box loses focus because the user enters a new class name in the Class edit-box and performs one of the following actions:

- Tabs out.
- Clicks elsewhere to transfer focus.
- Selects the OK button.

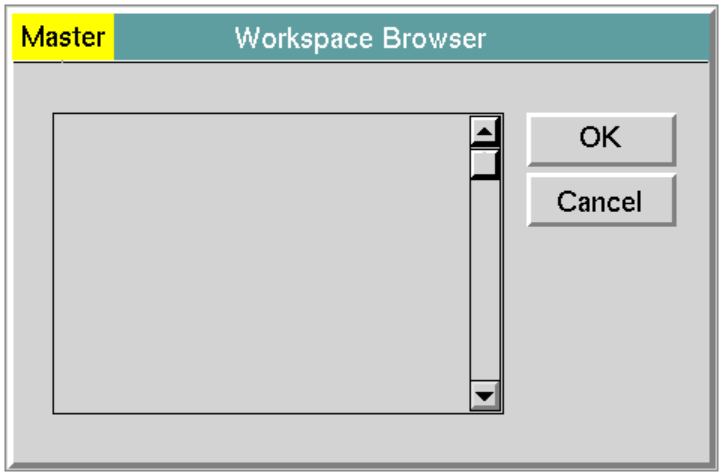
The g2ui-file-class-browser-user-entered-new-node procedure then updates the tree-view displaying the class hierarchy for the new class.

g2ui-file-tree-view-get-leaves-for-class
(*node*: class g2ui-file-tree-view-object, *window*: class g2-window)

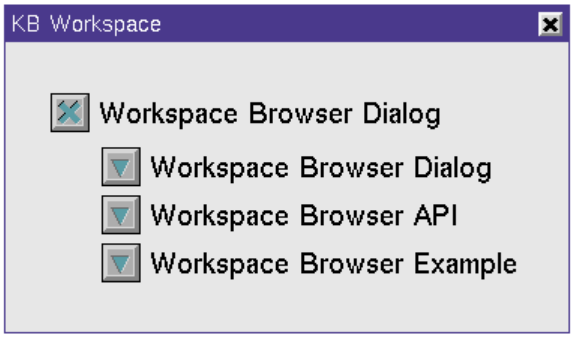
Creates the leaves for the specified node of the class hierarchy tree. The node-label attribute of node contains the name of the class to expand. This procedure iterates over all inferior classes and creates leaves for each one.

Workspace Browser Dialog

The Workspace Browser enables you to select workspaces, as shown in the following figure:



The following figure shows the top-level workspace that enables you to navigate to the components of the Workspace browser.



Selecting the Workspace Browser Dialog navigation button displays the workspace containing the master dialog for the Workspace Browser, as shown in the following figure.

Attributes

The Workspace Browser dialog attributes belong to these classes:

- g2ui-file-common-dialog-settings
- g2ui-file-tree-view-dialog-settings
- g2ui-file-workspace-browser-dialog-settings

The g2ui-file-workspace-browser-dialog-settings class inherits from the g2ui-file-tree-view-dialog-settings class, which inherits from the g2ui-file-common-dialog-settings class.

For information on the attributes for g2ui-file-common-dialog-settings, see Attributes of g2ui-file-common-dialog-settings. For more information on the attributes for g2ui-file-tree-view-dialog-settings, see Attributes of g2ui-file-tree-view-dialog-settings. For more information on dialog settings, see the Modifying Dialog Text.

Attributes of g2ui-file-workspace-browser-dialog-settings

Attribute	Description
initial-workspace	Specifies the workspace at the top of the tree.
Allowable values:	a symbol for any workspace name, or a class name
Default value:	g2-workspace-hierarchy

Setup Actions for Workspace Browser Dialog

The object that contains the procedures called while launching the Workspace Browser dialog is g2ui-file-workspace-browser-setup-actions. G2ui-file-workspace-browser-setup-actions specifies the following procedures:

- uil-simulate-play-mode
- uil-show-dialog
- g2ui-file-tree-view-check-for-user-setup-action

A user-defined setup handler specified in the g2ui-file-tree-view-dialog-settings object handles the initialization of these controls. The g2ui-file-tree-view-check-for-setup-action procedure looks for setup actions in the procedure named in the tree-dialog-setup-handler attribute of the associated settings object.

Note For information on actions, uil-action-description-arrays, and launching dialogs, refer to the *G2 GUIDE User's Guide*.

Workspace Browser API

g2ui-file-tree-view-add-leaf-node-for-workspace
(*node*: class g2ui-file-tree-view-object, *workspace*: class kb-workspace, *window*: class g2-window)

Creates a leaf node for the specified node and adds the specified workspace to the internal list of workspaces associated with the tree-view. The name of the relation between the node and the workspace list is the-workspace-list-for-tree-view-node.

g2ui-file-tree-view-get-leaves-for-workspace
(*node*: class g2ui-file-tree-view-object, *window*: class g2-window)

Creates the leaves for the specified node of the workspace tree. Leaves are gathered as long as the node-label attribute of the specified node contains one of the following values:

- The symbol knowledge-base
- A named workspace
- The name of a module

If node-label is the symbol knowledge-base, then G2UITREE creates an outline of the top-level workspaces for the loaded modules. If node-label is the named of a workspace, then the workspace hierarchy is limited to just those immediate "child" workspaces of the named workspace. If node-label is the name of a module, the workspace hierarchy contains just the top-level workspaces for the specified module.

g2ui-file-workspace-browser-create-settings-object
(*window*: class g2-window)
-> *settings*: class g2ui-file-workspace-browser-dialog-settings

Returns a g2ui-file-workspace-browser-dialog-settings object. This object contains attributes for initializing controls on the workspace browser dialog.

g2ui-file-workspace-browser-launch-dialog
(*dialog-id*: text, *window*: class g2-window,
settings: class g2ui-workspace-browser-view-dialog-settings)
-> *workspace*: item-or-value

Launches the dialog whose ID matches the specified ID on the given window. The dialog is initialized to reflect the attribute settings in settings. When the dialog is dismissed, g2ui-file-workspace-browser-launch-dialog returns the workspace that was selected or the symbol not-found.

g2ui-file-workspace-browser-setup
(*dialog*: class uil-dialog, *this-initiating-item*: item-or-value,
window: item-or-value, *original-initiating-item*: item-or-value,
dialog-actions: class symbol-list)

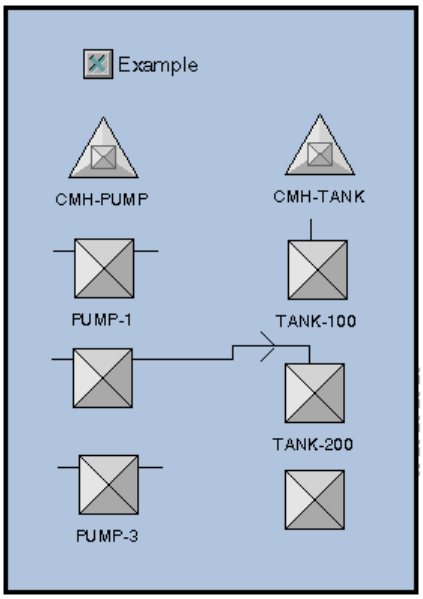
Called during initialization. This procedure initializes the controls on the dialog based on values found in the g2ui-file-workspace-browser-settings object that was passed into g2ui-file-workspace-browser-launch-dialog. This procedure first calls g2ui-file-tree-view-setup.

Example of Modifying a Master Dialog

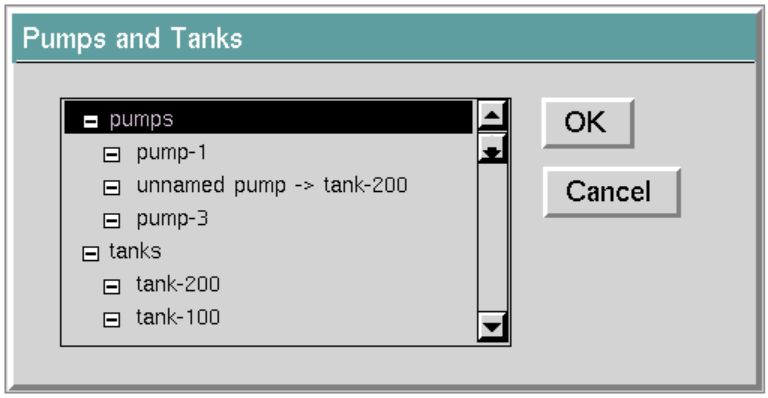
The following steps walk you through the modification of the Tree View Control dialog. If you need to modify the Tree View Control dialog for your own purposes, you will perform actions similar to the ones specified here. The basic steps are:

- Clone the appropriate master dialog so that you can work on your own copy.
- Give your copy of the master dialog its own unique ID.
- Create a class for each node that needs a method to collect its leaves.
- Write methods that open the leaves of the tree when the user double clicks on the "+" icon.
- Create a subclass of the dialog settings object.
- Write your own setup handler.
- Create a subtable for the node-tree attribute to specify the node class.
- Launch the Tree View Control dialog with your dialog settings object, which specifies the nodes to display.

This example shows you how to browse a hierarchy of pumps and tanks, whose structure appears in the following figure:



If you want to display the various pumps and tanks that you have created, you would start with the basic Tree View Control dialog to create a new master dialog, as shown in the following figure.



Note To make the following modifications, you must load *g2cuidev.kb* and *guide.kb* into G2.

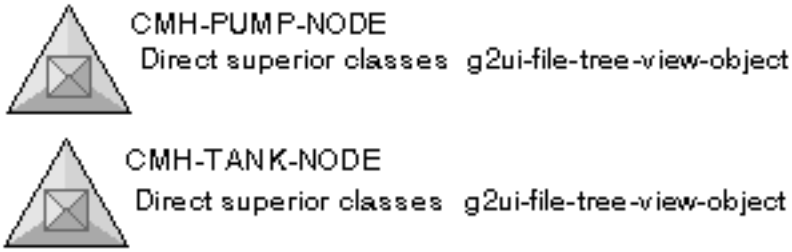
To modify the Tree View Control dialog:

1 Clone the Tree View Control master dialog and place the copy on a workspace belonging to your module. For more information on the Tree View Control master dialog, see Tree View Control and Dialog.

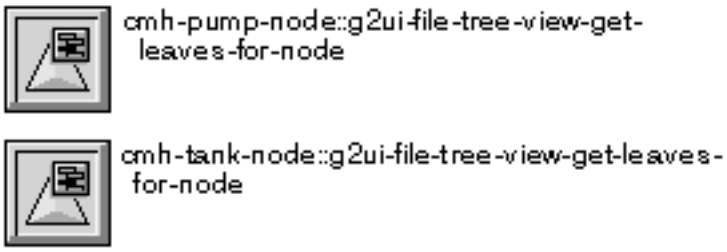
2 Because each master dialog in an application must have a unique ID, change the ID of the copy on your workspace.

The default ID of the Tree View Control dialog is g2ui-file-tree-view-dialog. Enter your own ID, such as my-tree-view-dialog.

3 Create a class for each node that needs a method to collect the leaves of its tree. The direct superior class is g2ui-file-tree-view-object. In this example, both pumps and tanks require such a class.



4 Write a method for each node class, cmh-pump-node and cmh-tank-node. These methods use the g2ui-file-tree-view-get-leaves-for-node API call and are executed when the user double-clicks on the "+" icon.



The following method, cmh-pump-node::g2uifile-tree-view-get-leaves-for-node, gets the leaves for the instances of the Pump class:

```
g2ui-file-tree-view-get-leaves-for-node (Node: class cmh-pump-node, Win : class
    g2-window)
Pump : class cmh-pump;
Tank : class cmh-tank;
Leaves : class g2ui-file-tree-view-node-leaves;
NewNode : class g2ui-file-tree-view-object;
C : integer = 0;
Id, Label : text;
begin
  Leaves = the node-leaves of Node;
  for Pump = each cmh-pump do
    C = C + 1;
    create a g2ui-file-tree-view-object NewNode;
    Id = call uil-utility-unique-id ("node-");
    conclude that the node-id of NewNode = Id;
    conclude that the node-parent of NewNode = the node-id of Node;
    conclude that the tab-num of NewNode = the tab-num of Node + 1;
    if the name of Pump exists then Label = "[the name of Pump]"
    else Label = "unnamed pump";
    if there exists a cmh-tank Tank connected to Pump and the name of Tank exists
       then Label = "[Label] -> [the name of Tank]";
    conclude that the node-label of NewNode = Label;
    insert NewNode at the end of Leaves;
  end;

  if C > 0 then conclude that the node-has-leaves of Node = true;
  conclude that the number-of-leaves of Node = C;
  conclude that the node-state of Node = the symbol expanded;
  conclude that the node-leaves-have-been-collected of Node is true;
end
```

This method gets information on the parent-child relationships in a particular node tree of pumps and uses each pump's name as a label. If there is another level of the node tree, it is indented. If the pump is connected to a tank, the connection is included in the label.

The following method, cmh-tank-node::g2uifile-tree-view-get-leaves-for-node, is similar to the method for displaying pumps. The method gets the leaves for the instances of the Tank class:

```
g2ui-file-tree-view-get-leaves-for-node (Node: class cmh-tank-node, Win : class
    g2-window)
Tank : class cmh-tank;
Leaves : class g2ui-file-tree-view-node-leaves;
NewNode : class g2ui-file-tree-view-object;
C : integer = 0;
Id, Label : text;
begin
  Leaves = the node-leaves of Node;
  for Tank = each cmh-tank do
    C = C + 1;
    create a g2ui-file-tree-view-object NewNode;
    Id = call uil-utility-unique-id ("node-");
    conclude that the node-id of NewNode = Id;
    conclude that the node-parent of NewNode = the node-id of Node;
    conclude that the tab-num of NewNode = the tab-num of Node + 1;
    if the name of Tank exists then Label = "[the name of Tank]"
    else Label = "unnamed tank";
    conclude that the node-label of NewNode = Label;
    insert NewNode at the end of Leaves;
  end;

  if C > 0 then conclude that the node-has-leaves of Node = true;
  conclude that the number-of-leaves of Node = C;
  conclude that the node-state of Node = the symbol expanded;
  conclude that the node-leaves-have-been-collected of Node is true;
end
```

5 To call your own setup handler, create a new class of the dialog settings object by subclassing g2ui-file-tree-view-dialog-settings.

Name the new class cmh-object-manager-settings-spec:

CMH-OBJECT-MANAGER-SETTINGS-SPEC

6 Create a new setup handler, cmh-tree-view-object-manager-setup-root-nodes, that displays the tree for pumps and tanks.

CMH-TREE-VIEW-OBJECT-MANAGER-SETUP-ROOT-NODES

The cmh-tree-view-object-manager-setup-root-nodes setup handler calls the default setup handler and then performs the additional setup actions. The following code is the cmh-tree-view-object-manager-setup-root-nodes setup handler.

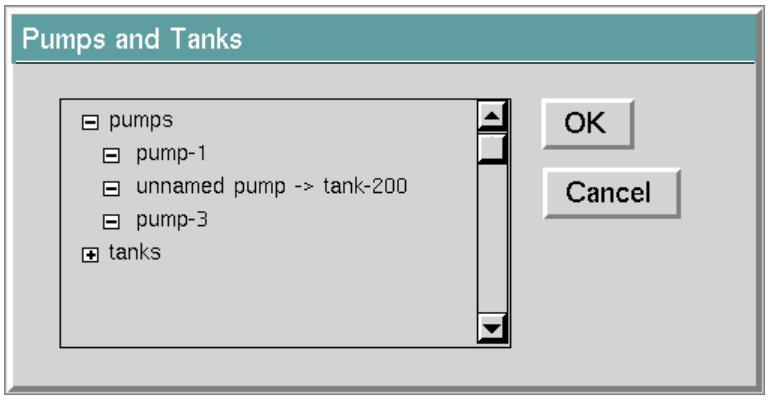
```
cmh-tree-view-object-manager-setup-root-nodes (Dlg: class uil-dialog,
     ThisInitiatingItem: item-or-value, Win: item-or-value,
     OriginalInitiatingItem: item-or-value, DialogActions: class symbol-list )
InitObj, SA, MO: item-or-value;
InitialNodes : class item-array;
Root : class g2ui-file-tree-view-object;
PumpNode, TankNode: item-or-value;
Leaves : class g2ui-file-tree-view-node-leaves;
Id, MsgText: text;
I : integer;

begin
     call g2ui-file-tree-view-setup (Dlg, ThisInitiatingItem, Win,
          OriginalInitiatingItem, DialogActions);
     SA = call uil-get-grobj-from-id-on-dlg-or-wksp (“tree-view-scroll-area”,
     Dlg);
     if (SA exists and SA is an item) then begin
          InitObj = call uil-find-initiating-object (Dlg);
          if (InitObj exists and InitObj is an item) then begin

               {create Root node}
               Root = the node-tree of InitObj;
               Id = call uil-utility-unique-id (“node-”);
               conclude that the node-id of Root = Id;
               conclude that the node-type of Root = the symbol
                    object-manager-root-node;
               conclude that the node-has-leaves of Root is true;
               Leaves = the node-leaves of Root;

               {create Root node for Pumps}
               create a cmh-pump-node PumpNode;
               Id = call uil-utility-unique-id (“node-”);
               conclude that the node-id of PumpNode = Id;
               conclude that the node-parent of PumpNode = the node-id of
                    Root;
               conclude that the node-label of PumpNode = “Pumps”;

               MsgText = “    [the node-label of PumpNode]”;
               MO, SA = call uil-create-message-with-appended-item (SA,
                    MsgText, 1, the symbol g2ui-file-node-has-leaves-icon );
               conclude that the user-data of MO = the node-id of PumpNode;
               conclude that the uil-appended-item-x-offset of MO = 5;
               insert PumpNode at the end of Leaves;
```

```
                {create Root node for Tanks}
                create a cmh-tank-node TankNode;
                Id = call uil-utility-unique-id (“node-”);
                conclude that the node-id of TankNode = Id;
                conclude that the node-parent of TankNode = the node-id of
                    Root;
                conclude that the node-label of TankNode = “Tanks”;

                MsgText = “    [the node-label of TankNode]”;
                MO, SA = call uil-create-message-with-appended-item (SA,
                    MsgText, 1, the symbol g2ui-file-node-has-leaves-icon );
                conclude that the user-data of MO = the node-id of TankNode;
                conclude that the uil-appended-item-x-offset of MO = 5;
                insert TankNode at the end of Leaves;

                call uil-display-message-list (SA,0);
            end;
        end;
    end
```

Note The value of the tree-view-dialog-setup-handler attribute in the cmh-object-manager-settings-spec dialog settings object is the name of the new setup handler, cmh-tree-view-object-manager-setup-root-nodes.

7. Enter the new setup handler named cmh-tree-view-object-manager-setup-root-nodes as the value of the attribute tree-view-dialog-setup-handler.
8. Create a subtable for the node-tree attribute in the cmh-object-manager-settings-spec dialog settings object and choose the cmh-pump-node class as the attribute value.
9. Launch the Tree View Control dialog by using g2ui-file-tree-view-launch-dialog API call. The launch procedure causes the Tree View Control dialog to display and uses the cmh-object-manager-settings-spec dialog settings object, which specifies that the nodes for pumps and tanks be displayed.

The following figure shows the Tree View Control dialog, displaying the node tree for pumps:



6

Customizing the Print Dialogs

Describes how the print dialogs interact with the Printer Setup system table through attribute settings and how you can use the print API.

Introduction

The G2 utility g2uiprnt provides a general purpose printing tool for use by G2 applications. The print dialogs set attributes that the Printer Setup system table uses to start a print job.

To use the print dialogs, see Printing G2 Workspaces.

Note The g2uiprnt module delivers a componentized print spooling dialog. It replaces the existing module guiprint.

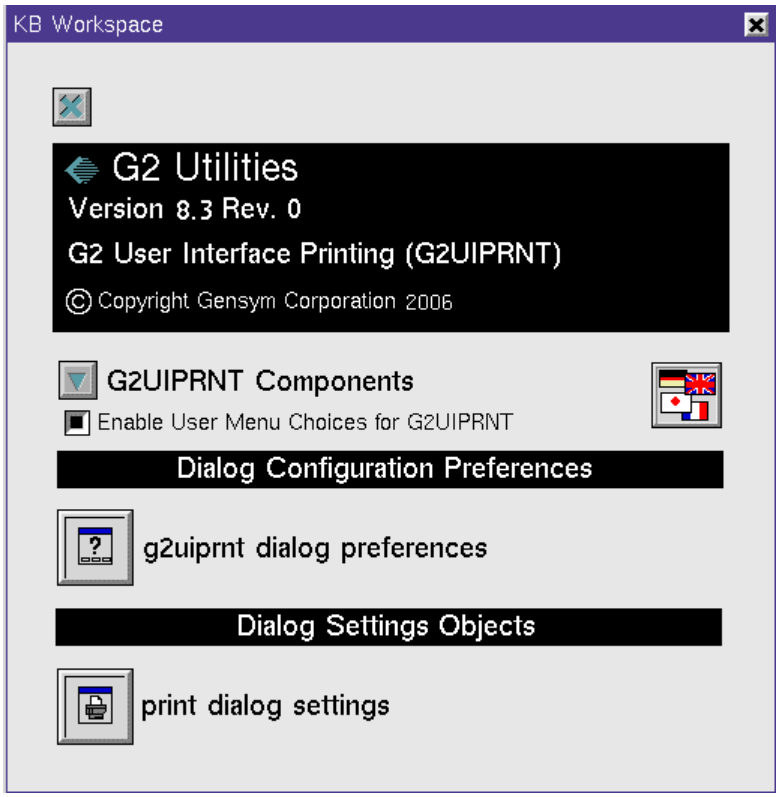
To navigate to the top-level workspace for g2uiprnt, do one of the following:

1 Display the GDI top-level workspace.

See Displaying GDI Top-Level Workspaces.

2 Click the G2UIPRINT Top Level button.

The following figure shows the top-level workspace for g2uiprnt:



Note To enable the Print user menu choice, make sure that the Enable User Menu Choice for g2uiprnt toggle button is selected.

Using Dialog Preferences and Settings

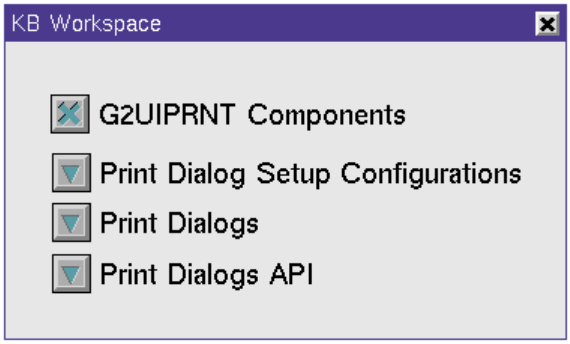
The g2uiprnt palette enables you to clone the preferences or settings icons and place them in your module. You can use the dialog preferences to change the colors used in the dialogs. For more information on modifying dialog preferences, see Modifying Dialog Colors.

You can use the dialog settings objects to modify print attributes, although these also can be modified using the print dialogs. For information on the print attributes, see The Dialog Settings Object.

For more information on using preferences and settings objects, see Modifying Dialog Appearance.

G2UIPRNT Components

Use the navigation button for G2UIPRNT Components to display the components menu, as shown in the following figure:



From the components menu, you can navigate to workspaces containing icons for:

- UIL configurations
- Master dialogs
- Print API

Print Dialog Setup Configurations

The Print Dialog Setup Configurations option enables you to specify the various background, border, button, and text colors on the print dialogs.

For more information on using UIL configurations, see the *G2 GUIDE User's Guide*. For additional information on modifying dialog configurations, see Modifying Dialog Colors.

Print Dialogs

The Print Dialogs option enables you to navigate to the workspace containing the master dialogs for the three print dialogs:

- Main Print Dialog
- Printer Selection Dialog
- Printer Options Dialog

The workspace also contains the text parameters for the lists of printers and servers and the pathname to printer:

- g2ui-print-printer-file-filename (default value is "printers.txt")
- g2ui-print-server-file-filename (default value is "servers.txt")
- g2ui-print-printer-file-pathname (default value is "")

For more information on using these text parameters to specify a printer, see Specifying a Printer.

Print Dialogs API

The Print Dialogs API option enables you to view the API calls that enable you to modify the action of the print dialogs. For more information on the API calls, see Print API.

The Printer Setup System Table

Printing in G2 is controlled exclusively by the state of the Printer Setup system table of the top-level module at the time printing occurs. The print dialogs and print API interact with the Printer Setup system table to carry out printing. The print dialogs are, in effect, user interfaces to the Printer Setup system table, which in turn provides G2 with the information it needs to carry out printing.

The following figure is an example of the information stored in the Printer Setup system table:

PRINTER-SETUP

Notes	OK
Authors	none
Change log	0 entries
Printing details	Workspace scaling: 200 workspace units per inch; Color conversion: black-and-white
Page layout	Paper size: letter; Paper orientation: portrait; Left margin: 0.5 inch; Top margin: 0.5 inch; Right margin: 0.5 inch; Bottom margin: 0.5 inch
Print spooling	Spooled filename template: "/bt/guide/guide/print-*.ps"; Spool file to printer: no; Printer identification: "unknown"
Printing priority	8
Printing file format	postscript

For more information Printer Setup system table, see the *G2 Reference Manual*.

The Dialog Settings Object

The print dialogs create or use an existing g2ui-print-dialog-settings object to store the values entered by the user from the print dialogs. The g2ui-print-dialog-settings object is then passed along to the printing procedures as the specification for how printing is carried out.

After the user enters these values into the print dialogs, these values are remembered for the next print session. Each time you want to print a workspace and there is no variation in the control settings, you need only select the Print button.

These settings apply to the G2 window, not G2 globally. For example, a user on a Telewindows session would have their own unique settings remembered for their window.

The print object is attached to a G2 window through a G2 relation named the-default-printer-settings-for. These settings can be viewed by using Inspect, looking for all of the G2 windows, and describing the appropriate window. There can be only one g2ui-print-dialog-settings per window.

After printing is finished, the print file is removed from your file system unless you indicate that the print job was to be directed to a file, in which case, the print file persists.

Attributes of g2ui-print-dialog-settings

Attribute	Description
user-name	Specifies the name of the user logged into the G2 window the print job originates.
Allowable values:	Any symbol
Default value:	unspecified
selected-printer	Specifies the name of the printer selected by the user on the main print dialog.
Allowable values:	Any symbol
Default value:	unspecified
selected-server	Specifies the name of the server selected by the user on the main print dialog.
Allowable values:	Any symbol
Default value:	unspecified
spooled-file-name	Specifies the name of the print file sent to the printer.
Allowable values:	Any text
Default value:	“temp-print-job.ps”
pathname-for-print-spooling	Specifies the pathname of the printer. If value is default, the default printer is used.
Allowable values:	Any text
Default value:	“default”

Attribute	Description
continuous-update-of-status-display	Specifies that the status of the print job be updated continuously.
Allowable values:	true or false
Default value:	false
echo-print-command	Specifies that the exact print command used to send the print job to the printer be displayed in a dialog. This attribute is intended as a debugging aid and should not be necessary in routine circumstances.
Allowable values:	true or false
Default value:	false
printer-command	Specifies the print command used to send the print job to the printer.
Allowable values:	Any symbol
Default value:	lpr
printer-copies	Specifies the print command line directive that indicates the number of copies to print.
Allowable values:	Any text
Default value:	"#"
printer-dispatch	Specifies the print command line directive that indicates the name of the printer.
Allowable values:	Any text
Default value:	"-P"

Attribute	Description
server-dispatch	Specifies the print command line directive that indicates the name of the server.
Allowable values:	Any text
Default value:	"-S"
scale	Specifies the scale for printing in workspace units per inch; 100 means one-to-one scaling.
Allowable values:	Any number
Default value:	100
print-to-file	Specifies that the print be sent to a file rather than printed immediately.
Allowable values:	true or false
Default value:	false
print-to-local-printer	Specifies that the print job be sent to the local printer.
Allowable values:	true or false
Default value:	false
number-of-copies	Specifies the number of copies of the print job to print.
Allowable values:	1 to 99
Default value:	1

Attribute	Description
paper-size	Specifies the size of the paper, such as letter, legal or A4, that the selected printer accept.
Allowable values:	letter, legal, ledger, a0, a1, a2, a3, a4, a5, b0, b1, b2, b3, b4, b5
Default value:	letter
orientation	Specifies whether length is greater than width (portrait) or width is greater than length (landscape).
Allowable values:	portrait or landscape
Default value:	portrait
print-format	Specifies the format of the print job, such as PostScript or Encapsulated PostScript.
Allowable values:	“postscript” or “encapsulated postscript”
Default value:	“postscript”
scale-to-fit-single-page	Specifies that a large print job be scaled to fit onto one of the selected paper sizes.
Allowable values:	true or false
Default value:	false
color-conversion	Specifies that the print job be color, black and white, or grayscale.
Allowable values:	full-color, black-and-white, grays
Default value:	grays

Attribute	Description
left-margin	Specify the left page margin in the selected units. The default is 0.5 inches.
Allowable values:	Any number
Default value:	0.5
top-margin	Specify the top page margin in the selected units. The default is 0.5 inches.
Allowable values:	Any number
Default value:	0.5
right-margin	Specify the right page margin in the selected units. The default is 0.5 inches.
Allowable values:	Any number
Default value:	0.5
bottom-margin	Specify the bottom page margin in the selected units. The default is 0.5 inches.
Allowable values:	Any number
Default value:	0.5
units	Specify the units for setting page margins.
Allowable values:	inches or centimeters
Default value:	inches

Attribute	Description
printing-details-text	Specifies a cached copy of the Printer Setup system table settings prior to the new settings. These settings are restored after the current print job completes.
Allowable values:	Any text
Default value:	""
page-layout-text	Specifies a cached copy of the Printer Setup system table settings prior to the new settings. These settings are restored after the current print job completes.
Allowable values:	Any text
Default value:	""
printer-spooling-text	Specifies a cached copy of the Printer Setup system table settings prior to the new settings. These settings are restored after the current print job completes.
Allowable values:	Any text
Default value:	""
printer-format-text	Specifies a cached copy of the Printer Setup system table settings prior to the new settings. These settings are restored after the current print job completes.
Allowable values:	Any text
Default value:	""
print-subworkspace	Specifies that all the workspaces below the selected workspace in the workspace hierarchy be printed.
Allowable values:	true or false
Default value:	false

Attribute	Description
print-in-reverse-order	Specifies that the workspaces be printed depth-first in reverse order.
Allowable values:	true or false
Default value:	false
print-status	Specifies the status of the print job: • initialized – Print job is ready for transmission to printer. • executing – Printing operation being attempted. • spooling – Transmission to printer and and its print queue. • printing– Print job is printing. • complete – Print job is finished.
Allowable values:	initialized, executing, spooling, printing, or complete
Default value:	unspecified
Notes:	If the user chooses the Cancel button in the main print dialog, the value of the printer-status attribute is initialized.
print-succeeded	Specifies that the previous print job was successfully printed.
Allowable values:	true or false
Default value:	false

Print API

The Print API provides a procedural interface to the print dialogs and the g2ui-print-dialog-settings object that stores the printer settings.

g2ui-print-prompt-user-for-print-dialog-settings

(*printer-settings-object*: class g2ui-print-dialog-settings,
window:class g2-window)
-> *printer-settings-object*: class g2ui-print-dialog-settings

Launches the print dialog. After the user enters the desired values into the controls on the dialog, *printer-settings-object* is returned to the calling procedure with those values.

g2ui-print-workspace

(*WS-or-list*: class item, *window* :class g2-window,
printer-settings-object: class g2ui-print-dialog-settings)
-> *status*: symbol, *filename*: text

Executes the print function as specified by the values in the *printer-settings-object*. The *status* is the status of the print job and is one of: SUCCEED or FAIL. The *filename* is a text value of the file name used to store the print file.

g2ui-print-create-print-dialog-settings-object

(*window*: class g2-window)
-> *printer-settings-object*: class g2ui-print-dialog-settings

Creates an instance of the class g2ui-print-dialog-settings, configures its attributes to match those currently set in the G2 system table Printer Setup, and returns the object to the calling procedure.

g2ui-print-find-print-dialog-settings-for-window

(*window*: class g2-window)
-> {*printer-settings-object*: item-or-value | no-default-settings-found: symbol}

Locates the g2ui-print-dialog-settings assigned to the G2 window passed in as the argument. This procedure locates the g2ui-print-dialog-settings (if any) through the relation the-default-printer-settings-for. If an object could not be found, a symbol is returned having the value no-default-settings-found.

g2ui-print-set-print-dialog-settings-as-default-for-window

(*printer-settings-object*: class g2ui-print-dialog-settings,
window: class g2-window)

Attaches the g2ui-print-dialog-settings assigned to the G2 window passed in as the argument.

g2ui-print-conclude-system-table-from-print-dialog-settings
(*printer-settings-object*: class g2ui-print-dialog-settings,
window: class g2-window)

Stores the attribute values in *printer-settings-object* into the Printer Setup system table for the current top-level module.

Before updating the system table, the procedure saves the table's current values to an internal storage area of *printer-settings-object*. Such storage is necessary because G2's system tables are global, and any action on the part of one user to these tables affects all users. To restore the previous attribute values to the Printer Setup table, use g2ui-print-restore-printer-setup-from-print-dialog-settings.

g2ui-print-restore-printer-setup-from-print-dialog-settings
(*printer-settings-object*: classg2ui-print-dialog-settings,
window: class g2-window)

Stores the values in the internal storage area of *printer-settings-object* into the Printer Setup system table for the current top-level module. This undoes the effect of calling g2ui-print-conclude-system-table-from-print-dialog-settings with that setup object.

Index

A

B

C

D

E

F

G

H

I

K

T

U

V

W