

Sanscript™

Fully Visual Scripting

Environment

User's Guide

This guide provides information on using the Sanscript Fully Visual Scripting Environment Version 2.0 for Windows 95, Windows 98, and Windows NT 4.0 or higher.

January 1999

Northwoods Software Corporation

142 Main St.

Nashua, NH 03060

<http://www.nwoods.com>

Copyright © 1995-1999 Northwoods Software Corporation

All rights reserved

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without the prior written permission of the publisher.

Northwoods Software Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Northwoods or an authorized sublicensor.

Neither Northwoods Software Corporation nor its employees are responsible for any errors that may appear in this publication. The information in this publication is subject to change without notice.

The following are trademarks of Northwoods Software Corporation: Northwoods, Sanscript, Flowgram, the Northwoods logo, and Fully Visual Programming.

The following are third-party trademarks:

Adobe and POSTSCRIPT are registered trademarks of Adobe Systems Incorporated.

Microsoft, MS, MS-DOS, Windows, Windows NT, and Windows 95 are registered trademarks, and NT is a trademark, of Microsoft Corporation.

All other trademarks and registered trademarks are property of their respective holders.

This product includes software licensed from Digital Equipment Corporation.

© Digital Equipment Corporation, 1993-1995. All rights reserved.

CONTENTS

Preface	vii
1. Introduction	1
What is Sanscript?	1
Starting and stopping the Sanscript tool	1
Using online help.....	2
2. Getting started	3
What you can do with Sanscript.....	3
The Sanscript screen	3
Supplied building blocks.....	5
Programming functions.....	6
The Sanscript Toolbar.....	6
Example functions.....	8
Hello World	8
Do you want to say hello	9
Fibonacci	11
Templates	13
Read File Template	13
3. Using the ready-to-run programs	17
What is a program? What is an application?	17
Locating programs in the Catalog	17
Running programs within the Sanscript environment	19
Running a program as an application	19
Introducing the ready-to-run programs	20
4. Creating your own functions	25
Creating the Hello World Example.....	25
Constructing functions from templates and examples.....	26
Understanding datatypes	27
General editing rules for functions	28
Using Sanscript language functions.....	32
Comment	33
Constant	33
Forms.....	34
Package.....	36
Pick One	37
Repeat	40
Errors	42
Error Handling.....	43
Ordering links.....	44
Transforming Language Functions	44
Compound datatypes	46
Lists	46
Records	46
Global Records	49
Viewing and printing Flowgrams	50
Viewing large Flowgrams.....	50

Viewing and editing function properties	50
Improving the appearance of Flowgrams	53
Printing Flowgrams	53
5. Running and testing functions.....	55
Running, pausing, and stopping a function	55
Running, pausing, and stopping a program:	55
Browsing a running program	55
Stepping through a function	56
Setting and clearing pause points	56
Examining values in a paused program	57
6. Working with the Catalog	59
Customizing the Catalog	59
Creating folders	59
Moving or copying functions or folders	60
Deleting functions or folders	62
Changing properties of a folder, or a function, or a file cabinet	62
Sharing functions with others	63
Using file cabinets.....	63
Defined file cabinet locations	63
Creating cabinets.....	64
Deleting cabinets	64
Exporting cabinets	64
Importing cabinets	65
7. Customizing options, toolbars, and windows	67
Customizing options.....	67
Customizing toolbars.....	69
Customizing windows.....	70
8. Programming functions	71
About programming functions	71
9. Using Sanscript Professional Edition	73
About Sanscript Professional Edition	73
Importing and using components from another application	73
Importing components from an application.....	74
Generating documentation for functions	75
Setting access controls and other properties for a cabinet.....	75
Glossary	77
Index	81

FIGURES

Figure 1. The Sanscript screen	4
Figure 2. Folders of the General file cabinet	5
Figure 3. Functions contained in the System folder	6
Figure 4. Toolbar	6
Figure 5. Flowgram for the "Hello World" example	9
Figure 6. Dialog box displayed by "Hello World"	9
Figure 7. Flowgram for "Do you want to say hello?"	10
Figure 8. Result of running Ask Yes/No in "Do you want to say hello?"	10
Figure 9. Flowgram for TRUE case in "Do you want to say hello?"	10
Figure 10. Flowgram for FALSE case in "Do you want to say hello?"	11
Figure 11. Flowgram for Fibonacci function	11
Figure 12. Flowgram for the Repeat in Fibonacci	12
Figure 13. Show and ask to stop in Fibonacci	12
Figure 14. Result of expanding the "Read File" template.....	13
Figure 15. Contents of Repeat within "Read File"	14
Figure 16. Completed outer flowgram of "Read Line"	14
Figure 17. Completed Repeat of "Read File"	15
Figure 18. Find Item dialog box.....	18
Figure 19. Make Application dialog box	19
Figure 20. Viewing an application using Windows Explorer.....	20
Figure 21. Pick One Edit Choices dialog box.....	38
Figure 22. Edit Repeat Inlet dialog box	41
Figure 23. Error function	42
Figure 24. Properties for <function> dialog box.....	51
Figure 25. All Functions Called dialog box	52
Figure 26. References To <function> dialog box	52
Figure 27. Drag-Drop Operation dialog box	61
Figure 28. Confirm Dependent Functions dialog box.....	61
Figure 29. Editing the properties of a file cabinet.....	62
Figure 30. Export Cabinet Information dialog box.....	65
Figure 31. Example of importing components	74

TABLES

Table 1. Online Help access	2
Table 2. Toolbar symbols and functions	6
Table 3. Simple examples folder.....	21
Table 4. System examples folder.....	21
Table 5. Examples functions folder.....	22
Table 6. Match quiz folder	22
Table 7. Batch examples folder.....	23
Table 8. Templates folder	23
Table 9. Datatype definitions	27
Table 10. Datatype conversion formats	28
Table 11. Inlet, outlet, and connector rules	31
Table 12. List of form fields	35
Table 13. Valid Error input datatypes	43

Table 14. Transforming language functions.....	45
Table 15. Drag and Drop operations.....	60
Table 16. Folder names and descriptions	64
Table 17. Overview options.....	67
Table 18. Catalog options	68
Table 19. Canvas options.....	68
Table 20. Window options.....	69
Table 21. General options	69

Purpose of this guide

This guide describes how to use the Northwoods Software Sanscript tool. The Sanscript tool provides a way to construct programs using pictures instead of text.

Who should use this guide

This guide is intended for a broad audience, including system administrators, application programmers, and any other user of the Sanscript tool.

Structure of this guide

This guide is organized as follows:

- **Introduction** provides an overview of what the tool can do, how to start it, and how to get help.
- **Getting started** introduces the parts of the Sanscript screen and basic concepts.
- **Using the ready-to-run programs** describes how to run programs that are already in the Sanscript Catalog.
- **Creating your own functions** explains how to construct your own functions, programs, and applications.
- **Running and testing functions** describes how to run and test functions or programs that you have constructed using the tool.
- **Working with the catalog** describes how to store and manage functions you create. This chapter also describes how to create, export, and import file cabinets containing functions.
- **Customizing options, toolbars, and windows** describes how to set options and adapt the appearance of the screen to suit your needs.
- **Programming functions** introduces the functions supplied with the tool.

- **Using Sanscript Professional Edition** describes the features of the Professional Edition.

Assumptions

This manual assumes you are familiar with Windows and Windows NT concepts and terminology. If you are not, please refer to your Windows documentation or online help.

Conventions

This manual uses the following conventions.

General conventions

The list below shows special kinds of formatting used in this manual.

<u>Note:</u>	Clarification or exception.
<u>Caution:</u>	Warning about conditions that could cause unexpected damage to data, software, or equipment.
<i>Courier</i>	Text you enter from the keyboard or view on the screen
Arial bold	Command
Key+Key	Key combination. Hold down the first key and press the second key.

Mouse conventions

The list below defines key words associated with mouse operations and movements in this manual.

Point	Position the mouse cursor so that the tip rests directly on the screen object.
Click	Press and release the left mouse button.
Double-click	Quickly press and release twice the left mouse button.
Right-click	Press and release the right mouse button.
Drag	Point, press and hold the left mouse button, move the cursor to the new location, release the left mouse button.

1. INTRODUCTION

What is Sanscript?

Sanscript is a revolutionary new programming tool. Unlike many tools that have been called “visual” but still require conventional programming, Sanscript is fully and truly visual—you don’t need to “write code” to create programs. This technological breakthrough allows both programmers and non-programmers to construct computer applications quickly.

With Sanscript, you create applications simply by drawing pictures:

- 1 Locate each function you want to use in the Catalog.
- 2 Drag each function out of the Catalog and drop it onto a canvas.
- 3 Link the functions.

Your mouse does most of the work.

The Sanscript Catalog includes over 200 assembled functions you can use immediately to build programs.

Starting and stopping the Sanscript tool

To start Sanscript:

- On the Windows **Start** menu, click **Programs**, then click **Northwoods Software**, then click **Sanscript**.

To stop using the Sanscript tool:

- 1 On the **File** menu, click **Exit** to close the program.
If you made changes while running the program, a dialog box appears with the text **Save Changes?**.
- 2 Click **Yes** to save changes and exit, click **No** to exit without saving changes, or click **Cancel** to simply close the dialog box.

Using online help

Online help is a convenient way to look up information while using the product. Get help these ways:

Table 1. Online Help access

Do this...	To get this kind of help...
Press F1	Help Table of Contents
Click Help on a dialog box (where it appears)	Specific help about the dialog box
Click to select an object on the Canvas window, then press F1	Specific help about the object
On the Help menu, click On Context , then click an object you see on the screen	Specific help about the object

2. GETTING STARTED

What you can do with Sanscript

The hundreds of functions that come with Sanscript allow you to create applications to do specific tasks, such as:

- Find all folders on a computer that have had a lot of recent activity
- Determine how full a disk drive is
- Automatically delete files of a certain kind in certain folders

Functions that come with Sanscript include, among others:

- Plus, minus, and other integer and decimal arithmetic functions
- Search, extract, and other text processing functions
- Open, close, read, write and other file functions
- Create path and other directory functions
- Launch application, Send Keys to an application, and other system functions
- Send message, and other Dynamic Data Exchange (DDE) functions

The Sanscript screen

When you start Sanscript, three windows open: the Overview, the Catalog, and the Canvas.

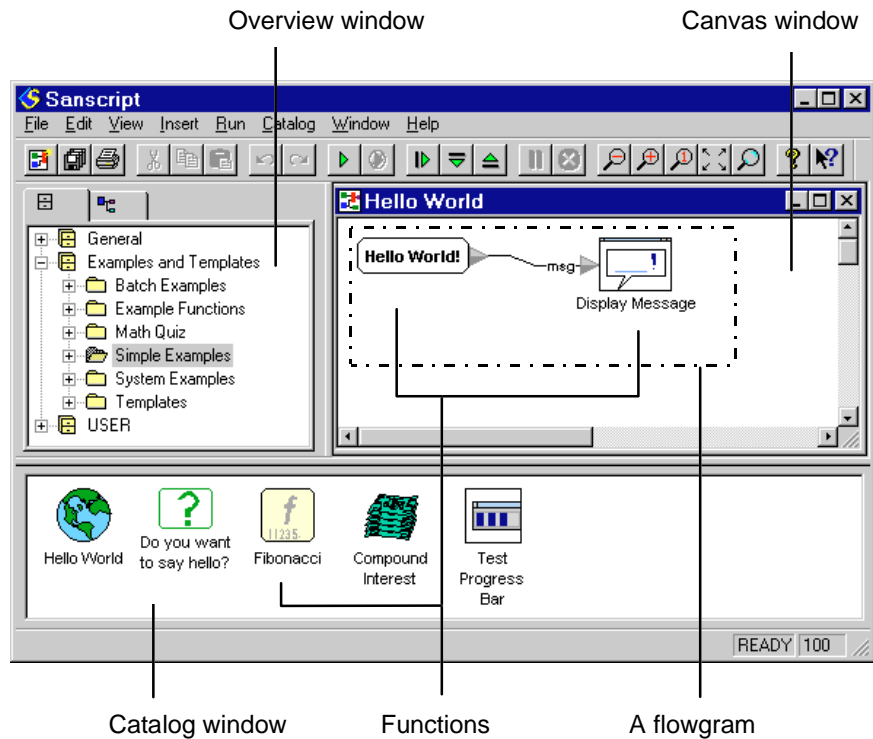


Figure 1. The Sanscript screen

The **Overview** window (on the left) displays all the file cabinets available in Sanscript. Initially, the Overview window shows the General, Examples and Templates, and User file cabinets. The Overview window works like the left half of Windows Explorer.

- Click a file cabinet (or folder) icon to display its contents in the Catalog window.
- Double-click a file cabinet (or a folder) to list its folders in the Overview window.

The **Catalog** window (on the bottom) displays the contents of the file cabinet or folder open in the Overview window.

- Double-click a function in the Catalog to open that function and show its flowgram in the Canvas window.

Note: If you double-click a function and it does not open, the function does not have a flowgram.

- Select functions in the Catalog window and drag them onto the Canvas window to create flowgrams.

- Right click a function in the Catalog window to view and edit the function's properties.

The **Canvas** (on the right) displays the flowgram for a function. Data in a flowgram flows over links from one function to another. At startup, the Canvas displays the flowgram for the “Hello World” function; later, it shows the flowgram for the function last worked with (if the “Remember Open Windows on Exit” option is checked).

Note: A function receives data, performs some calculation, and then outputs some data. For example, the asterisk (*) key on a calculator is a function that takes two numbers as inputs and produces their product as its result. The calculation part of a function is defined using the Sanscript tool's special flowgram data-flow diagram. The flowgram for a function is defined as some set of functions linked to other functions.

Supplied building blocks

The Sanscript tool comes with hundreds of functions to use as building blocks in making other functions. Most of these are stored in folders (see Figure 2) in the General file cabinet. Double-click the General cabinet to display the folders in the Catalog window.

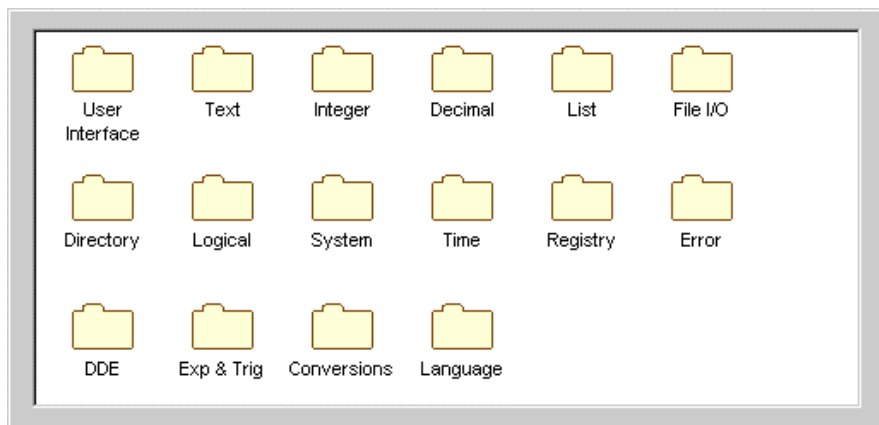


Figure 2. Folders of the General file cabinet

The *language functions*, in the Language folder and on the **Insert** menu, are special functions needed to complete the programming language itself. (The language functions are described in Chapter 4.). All the other functions, the *programming functions*, are the building blocks you use to construct your own programs.

Programming functions

Several hundred programming functions are provided with Sanscript. These functions are distributed among the folders listed in the Catalog.

As an example, Figure 3 shows the functions contained in the System folder.

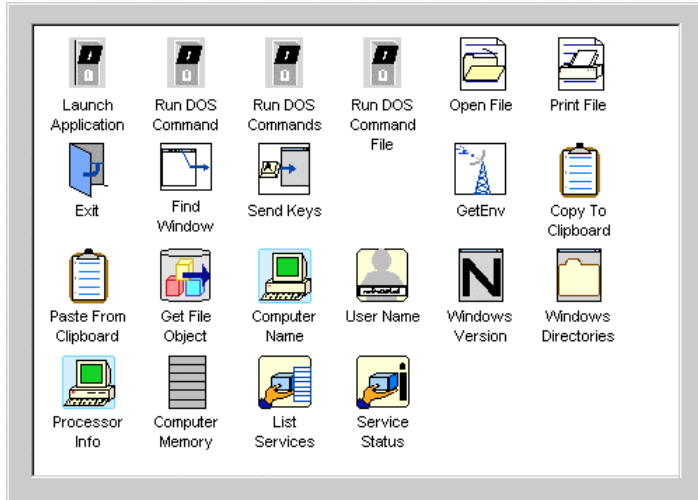


Figure 3. Functions contained in the System folder

Included with the programming functions are *templates* and *examples*. Templates are partially constructed programs with missing portions that you fill in. You can use templates and examples to speed up the construction of your own programs.

The Sanscript Toolbar




The toolbar provides buttons for frequently performed operations.



Figure 4. Toolbar




















See Table 2 for an explanation of toolbar symbols and functions.

Table 2. Toolbar symbols and functions

Click ...	To do the following...
	Create a new function
	Save the contents of the active Canvas window
	Print the contents of the active Canvas window

(Continued on next page)

Table 2. Toolbar symbols and functions (continued)

Click ...	To do the following...
	Cut selected items and place them on the clipboard
	Copy selected items and place them on the clipboard
	Paste items from the clipboard
	Undo recent edits
	Redo edits previously undone
	Run the current program
	Check the current program for errors
	Step - run the next function in the current program
	Step into - prepare to run inside the flowgram of the next function
	Step out of - leave the flowgram of the current function and prepare to run the following function
	Pause - suspend a running program
	Stop - terminate a running program
	Zoom out to view more of a large flowgram
	Zoom in to view just a portion of a large flowgram
	Restore the zoom magnification to its normal setting
	Set the zoom magnification so that the active flowgram just fits within the Canvas window
	Bird's eye zoom - zoom out to select a portion of the flowgram to see
	Look up information about how to use Sanscript
	Look up information about a selected menu command, toolbar button, or other object

Example functions

The example functions, stored in the Simple Examples folder of the Examples and Templates file cabinet, help you get started using Sanscript. This section introduces the three example functions, “Hello World,” “Do you want to say hello?”, and “Fibonacci”. The example functions “Compound Interest” and “Test Progress Bar” are not described in this manual. “Compound Interest” illustrates how to do a more extensive arithmetic using decimal numbers, and “Test Progress Bar” illustrates how to add a progress bar to an application.

The behavior for a function such as “Hello World” is defined by its flowgram. As shown in Figure 5, the flowgram for the function “Hello World,” a flowgram consists of a set of functions and links. The functions receive data on their inlets and produce data on their outlets. Data flows over links which connect the outlets of functions to the inlets of other functions.

All functions work according to the following rules:

- 1 Any function that has data present on all of its inlets is runnable; if a function has no inlets, it is runnable.
- 2 The Sanscript environment can choose any runnable function to run next.
- 3 When a function finishes running, it produces data on its outlets.
- 4 Data on the outlets of finished functions flows to the inlets of all the functions linked to those outlets.
- 5 The cycle repeats at step 1 until all functions in a program have run.

In keeping with these rules, and because inlets appear on the left of functions and outlets on the right, functions typically operate from left to right (but not necessarily from top to bottom).

Hello World

The “Hello World” example supplied with Sanscript is an example of a very basic function.

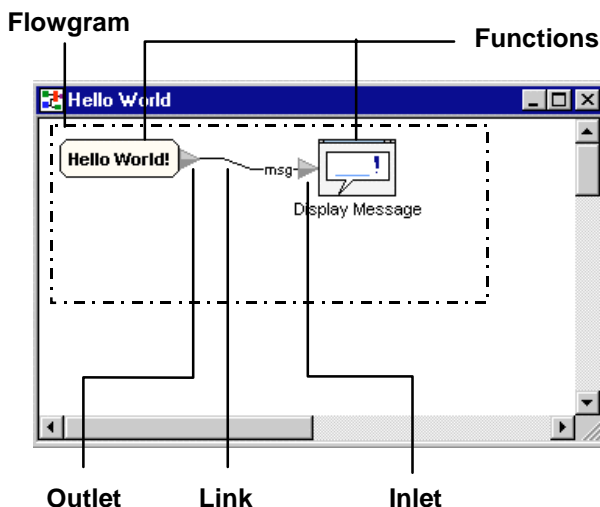


Figure 5. Flowgram for the “Hello World” example

The flowgram for Hello World has only two functions:

- a constant
- a Display Message function

The constant runs and produces the text “Hello World” on its outlet. A link connects this data to the inlet of a Display Message function. When Display Message runs, it opens a window and displays its data in a dialog box.

Thus, when you run the “Hello World” function, data flows in the flowgram of Figure 5, and the dialog box of Figure 6 appears.



Figure 6. Dialog box displayed by “Hello World”

Do you want to say hello

The “Do you want to say hello?” example illustrates “conditional programming” and uses the Sanscript Pick One language function to make a decision.

First, an Ask Yes/No function asks the user “Do you want to say hello?” The Ask Yes/No function displays the dialog box shown in Figure 8. The user’s answer then goes to the selection inlet of a Pick One function.

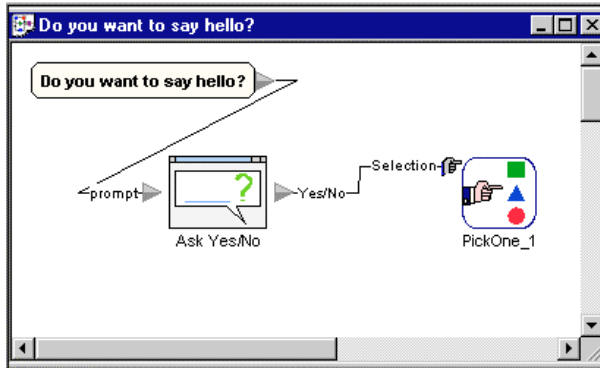


Figure 7. Flowgram for “Do you want to say hello?”

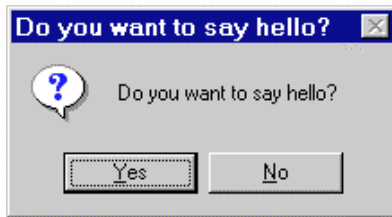


Figure 8. Result of running Ask Yes/No in "Do you want to say hello?"

Inside the Pick One function, there are two separate flowgrams, one for the TRUE (or Yes) case (see Figure 9) and one for the FALSE (or No) case (see Figure 10). If the user clicks Yes, the flowgram selected by the True tab runs, and “Hello World” runs. If the user clicks No, the flowgram selected by the False tab runs, and the message “That’s too bad!” is displayed.



Figure 9. Flowgram for TRUE case in “Do you want to say hello?”

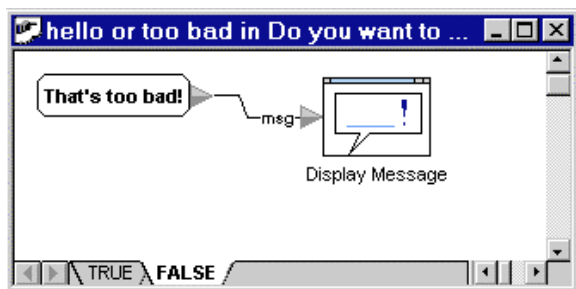


Figure 10. Flowgram for FALSE case in “Do you want to say hello?”

To view the alternative flowgrams within the Pick One:

- 1 Double-click the Pick One, or right-click the Pick One and click **Edit Definition**.
- 2 Click the tab (“TRUE” or “FALSE”) for the choice whose flowgram you want to view.

Fibonacci

The “Fibonacci” example illustrates “repetitive programming” and uses the Sanscript Repeat language function to repeat a calculation.

The Fibonacci function is mathematically defined as a function that produces the series of numbers 1, 2, 3, 5, 8, 13, ... The calculation for Fibonacci starts with 1 and 2 and produces the series at each step by summing the previous two numbers.

The flowgram (Figure 11) for the Fibonacci function starts off the calculation. The initial sum (2) and the first number of the series (1) flow into a Repeat function.

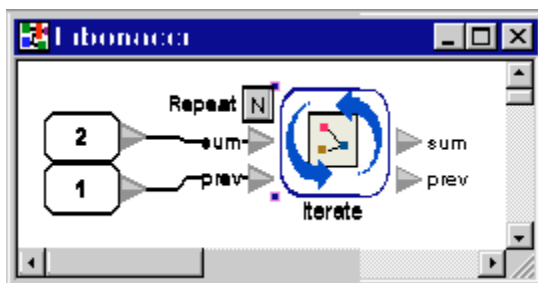


Figure 11. Flowgram for Fibonacci function

The flowgram within the Repeat function (double-click the Repeat to view, Figure 12) shows the calculation performed at each step in calculating the Fibonacci series.

Flowgrams for Repeats can have special inlet and outlet connectors (see Figure 12). During the first repetition, the inlet connectors supply the data from the inlets of the Repeat; during the second and later repetitions, the inlet connectors supply the data placed on the outlet connectors during the previous repetition. After the last repetition, the outlet connectors send their values to the outlets of the Repeat function, where they can flow to other functions.

In the flowgram for the Repeat in Fibonacci (Figure 12), the supplied function “Plus” produces the next number in the series on the “sum” outlet connector by adding the current number of the Fibonacci series (on the “sum” inlet connector) to the previous number in the series (on the “prev” inlet connector). The flowgram also moves the previous value of the “sum” to the “prev” outlet connector, which makes it the value of “prev” for the next iteration.

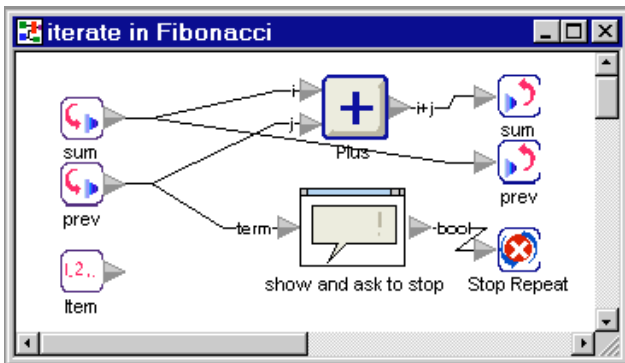


Figure 12. Flowgram for the Repeat in Fibonacci

At each iteration, the “show and ask to stop” function (double-click to see its definition, Figure 13) displays a dialog box with the current sum and asks if the user wants to “Compute Another?” If the user clicks No in the dialog box displayed by the Ask Yes/No function, that answer is converted to a “Yes” by the “Not” function. After the “show and ask to stop” function completes, the “Yes” flows to the “Stop” function, and stops the Repeat, and the calculation of Fibonacci.

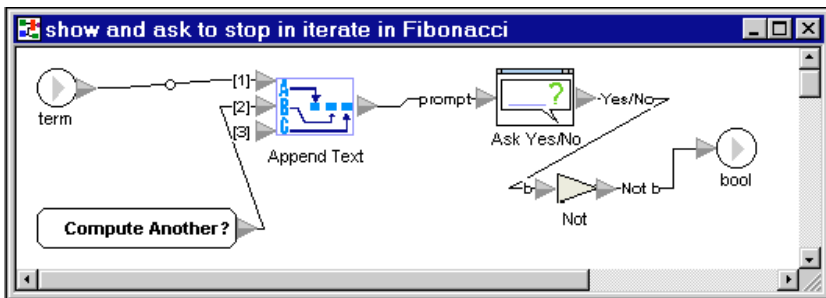


Figure 13. Show and ask to stop in Fibonacci

Templates

The templates, stored in the Templates folder of the Examples and Templates file cabinet, are partially completed functions that you can fill in to construct functions of your own. Templates speed up the construction of functions by eliminating the extra work needed to recreate common patterns.

This section illustrates the use of templates with an example using the “Read File” template.

The templates “Modify File”, “Write File”, “Build List” and “Filter List” are not described in this manual. Use “Modify File” for updating a file with new information, “Write File” for creating and writing a new file, “Build List” for constructing a list, and “Filter List” is revising an existing list.

Read File Template

When a template is dropped onto the Canvas window, it *expands* to a flowgram. The Read File template expands to a flowgram that is the pattern for opening and closing a file (see Figure 14).

The expanded Read File flowgram has three functions, which have already been linked: an Open File for Read, a Repeat, and a Close Input File. The red color of the `filename` inlet of Open File for Read means that it must be linked to a constant, or a function, that provides the name of the file. The `any?` outlet of the Open File For Read, which is TRUE if there are any lines in the file and FALSE otherwise, is connected to the `N` inlet of the Repeat and thereby ensures that the Repeat will not run if the file is empty.

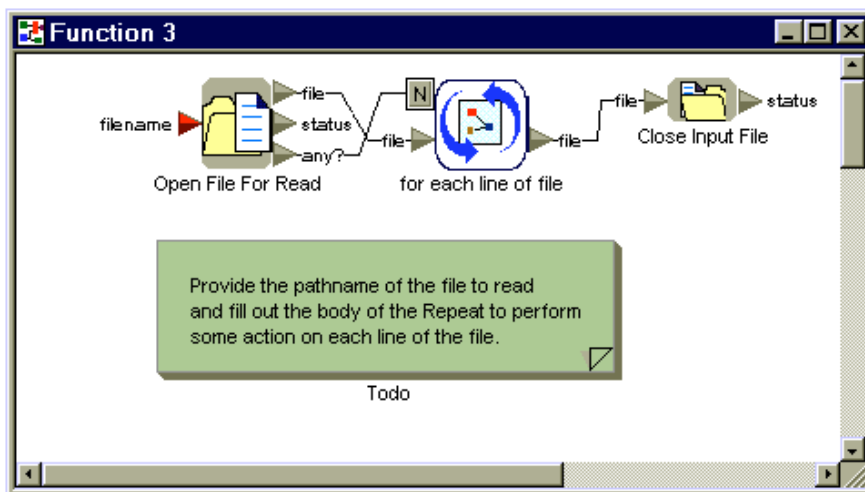


Figure 14. Result of expanding the "Read File" template

The flowgram within the Repeat (see Figure 15) is the pattern for reading each line of a file: a Read Line function receives the file on its `file` inlet and produces the file, the next line, and an indication (`eof`) of whether the current line is at the end of the file. A Stop function ensures that the Repeat stops when the last line has been read.

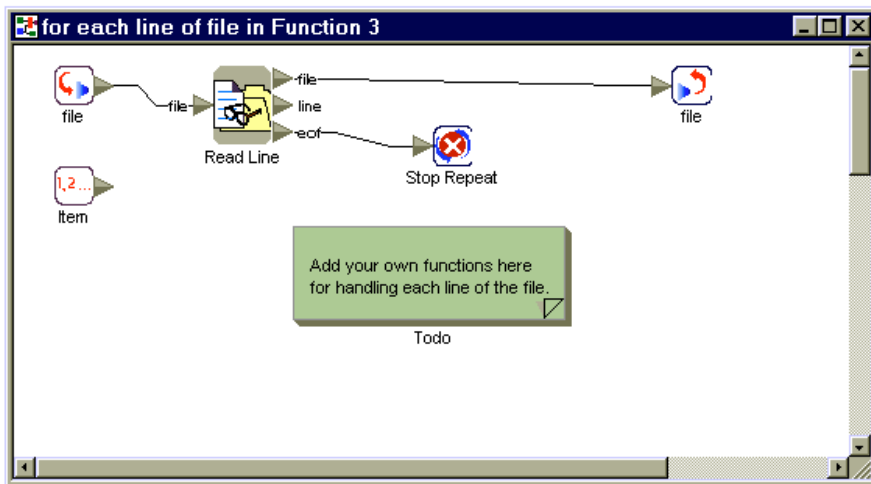


Figure 15. Contents of Repeat within "Read File"

To make a program using the Read File template, after expanding the template, simply insert a function to provide the file name, and insert one or more functions inside the Repeat to process each line. One way to complete the program is to use a constant for the file name (see Figure 16), and to use a Display Message function to display each line (see Figure 17).

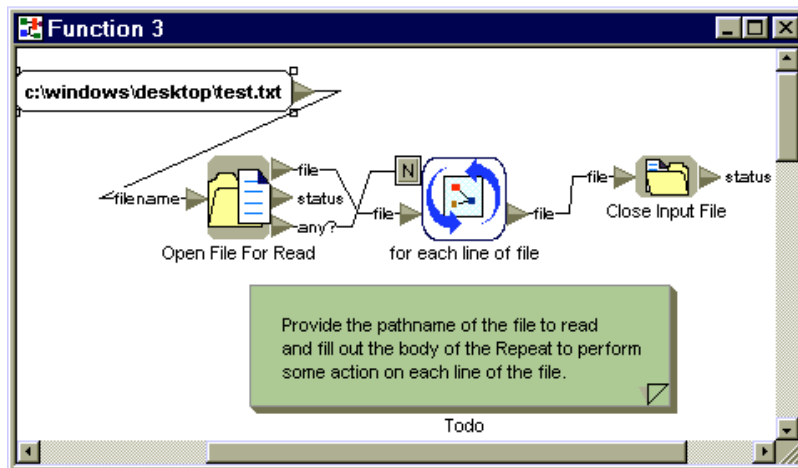


Figure 16. Completed outer flowgram of "Read Line"

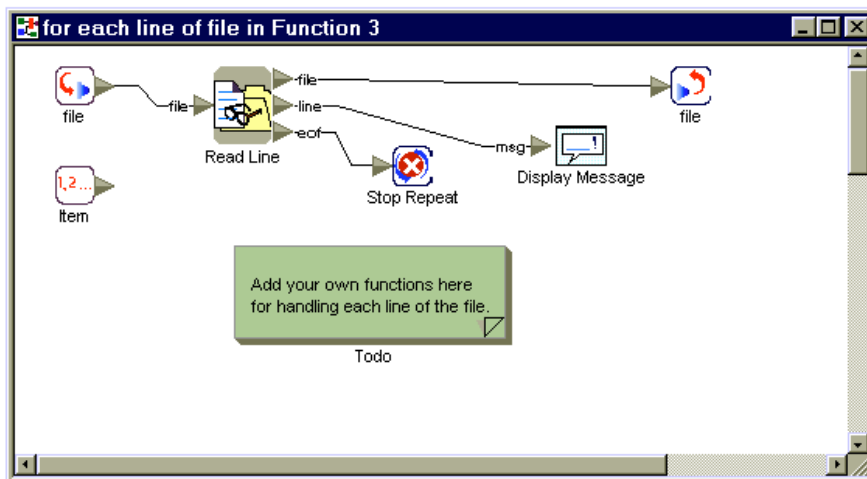


Figure 17. Completed Repeat of "Read File"

The resulting program, called Function 3 in this example, displays each line of the file `c:\windows\test.txt`, one line at a time, and stops after the last line is displayed.

3. USING THE READY-TO-RUN PROGRAMS

What is a program? What is an application?

A **function** takes inputs, performs a calculation, and produces outputs. Sanscript functions are stored in the Catalog and are represented by symbols.

A **program** is a function whose flowgram does not require inputs — that is, it has no inlet connectors that require a value (it can have inlet connectors that have default values). Because a program does not need inputs from another function, a program can run by itself. You can run a program either within the Sanscript tool, or independent of the tool (by double-clicking a Windows shortcut). A Sanscript program run outside of the Sanscript tool is called an **application**.

Locating programs in the Catalog

Programs—flowgrams without required inputs—can be found anywhere in the Sanscript Catalog. When you start using Sanscript, all the programs are located in the Examples and Templates cabinet. Functions you create are put in the User file cabinet by default. If you *import* a file cabinet created by someone else, that cabinet may contain programs you can run.

To browse the Catalog to locate a program:

- 1 In the **Overview** window, double-click the file cabinet you want to browse.
- 2 In the file cabinet, double-click the folder you want to browse.
You can open folders that you see in either the **Overview** window or the **Catalog** window.
- 3 Double-click the function you want to browse.
If the function has no inlet connectors, it is a program you can run; otherwise, it is a function you can use to construct other functions and programs.

You can also use this procedure to locate programs in the **Examples and Templates** file cabinet.

To search the Catalog to locate a program:

- 1 On the **Catalog** menu, click **Find First**.

The system displays the **Find Item in Folder** dialog box.

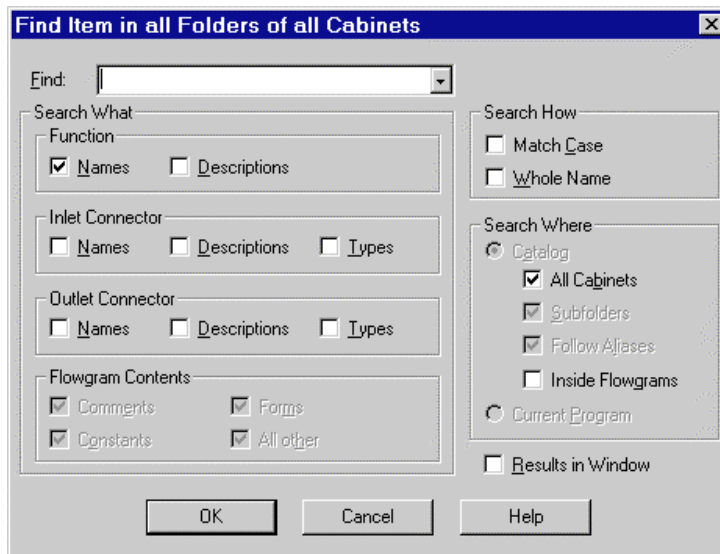


Figure 18. Find Item dialog box

- 2 Fill in the **Find** drop-down list box with a portion of the name of the function you want to locate.
- 3 Fill in the **Search What** group box to identify what parts of the function you want to search.
- 4 Fill in the **Search How** group box to restrict the number of things that will match.
- 5 Fill in the **Search Where** group box to specify what parts of the Catalog you want to search.
- 6 Fill in the **Flowgram Contents** group box when you select the **Current Program** in the **Search Where** box.
- 7 Check **Results in Window** if you want to see a list of all items that match.
- 8 Click **OK** to begin the search.
- 9 Double-click any item shown in the **Results** window to open that item.
- 10 Double-click to open a function that you want to browse.
If the function has no inlet connectors, it is a program that you can run; otherwise, it is a function that can be used to construct other functions and programs.

Running programs within the Sanscript environment

After you have located a program in the Catalog, you can run (and stop) the program from within Sanscript.

To run a program:

- 1 In the **Catalog** window, double-click the program's icon.
If the **Run** button, located on the toolbar, is lit (green), the function is a program and can be run.
- 2 Click the **Run** button on the toolbar.
The **Run** button becomes gray and the **Stop** button lights.

To stop running a program:

- Click the **Stop** button on the toolbar.

Running a program as an application

To run a program using a Windows shortcut, the program must first be made into an application.

To make a program into an application:

- 1 On the **File** menu, click **Make Application**.
- 2 In the **Make Application** dialog box, review the directory where Sanscript has chosen to store the application.
By default, applications are stored in the **Apps** folder of the **User** folder. Click **Browse** to place the application elsewhere.



Figure 19. Make Application dialog box

- 3 Click **OK**.
A copy of the program and all the other functions it needs has now been stored in the selected folder and a shortcut is available on your desktop. The

program is now a separate application, so that changes you make to the program within Sanscript will not affect the application. See Figure 16 to see what the application files look like to Windows Explorer.

Note: By default, the folder containing the applications is: Program Files\Northwoods Software\Sanscript\User\Apps. Use Windows Explorer to view the three files in the folder that contains the application: **.BAT** file, **shortcut** file, and **LIB.fvl** file. The LIB.fvl file contains the program itself. Use the .BAT file or shortcut file to run the application.

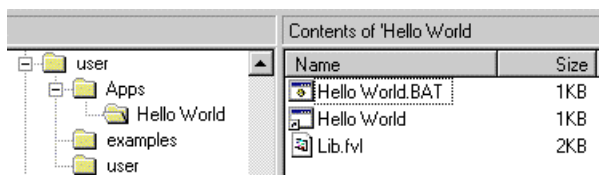


Figure 20. Viewing an application using Windows Explorer

To run an application:

- 1 Using Windows Explorer, locate the shortcut created by **Make Application**.
- 2 Double-click the shortcut.

Introducing the ready-to-run programs

This section introduces the ready-to-run programs. To locate and run the function in the Catalog, use the procedures described in “Locating programs in the Catalog” (page 17) and “Running programs within the Sanscript environment” (page 19).

If you find a function especially useful, make it into an application so you can run it more conveniently from your Windows desktop. For instructions on making a function into an application, see “General editing rules for functions” (page 28).

Tables 3 through 8 show names and descriptions of ready made functions, grouped by various categories into folders.

Table 3. Simple examples folder

Name	Description
Hello World	A very simple program that demonstrates how to display a message.
Do you want to say hello?	A simple program that demonstrates a Pick One.
Fibonacci	A program that computes the Fibonacci sequence: 1, 2, 3, 5, 8, 13... It demonstrates the use of a Repeat.
Compound Interest	Demonstrates Repeats and Packages.
Test Progress Bar	Demonstrates use of a progress bar and a status line.

Table 4. System examples folder

Name	Description
Disk Space Used	Find out how much disk space is used by a set of files
Directory Tree Listing	Show a listing of all the files matching a given name.
Show Disk Drives	Show a listing of the disk drives available on the system.
Show System Info	Present information about this computer system.
Show Memory Info	Present information about the amount of memory on this computer.
Show Services	Show a list of the services on this NT system.

(Continued on next page)

Table 4. System examples folder (continued)

Name	Description
Test Send Keys	Play an AVI file by controlling the Media Player using Send Keys.
Show Registry File Types	Present a list of the file extensions under HKEY_CLASSES_ROOT.

Table 5. Examples functions folder

Function	Operation
InRange?	Determine if an integer is within a given range.
StringSubst	Function to substitute some text for another in a bigger text string. The input text must contain a question mark ("?"), which will be replaced with the SUBST text.
Round to Decimal Fraction	Rounds decimal numbers to some fraction. By default rounds to 1/100, or two decimal places.
Fill with Blanks	Produce a fixed length text. Given an input text string, this function pads with blanks (space characters) until it is SIZE characters long. If the input text is longer than SIZE, it is returned without truncation and without any appended blanks.

Table 6. Match quiz folder

Function	Operation
Math Quiz	Simple arithmetic drill

Table 7. Batch examples folder

Function	Operation
Report File Space Used	Produce a report listing all the files with a given extension, their sizes, modified dates, and creation dates. The report is placed in the given directory, with a name that is the name of the computer.
Report Full Disks	Produce a report listing all disks that have less than some percentage space free. The report is placed in the given directory, with a name that is the computer name. The default free percentage is 5%.
Change Sounds in Registry	A sample program for changing the registry. The process must have permission for making the changes.

Table 8. Templates folder

Function	Operation
Read File Template	A template for reading the lines of a file. Drop this into your flowgram and complete the body of the Repeat.
Modify File Template	A template for copying a file while changing some or all of its lines. Drop this into your flowgram and complete the body of the Repeat.
Write File Template	A template for creating a file line-by-line. Drop this into your flowgram and complete the body of the Repeat.
Build List Template	A template for creating a list an item at a time. Drop this into your flowgram and complete the body of the Repeat.
Filter List Template	A template for creating a list by selecting items from an existing list. Drop this into your flowgram and complete the body of the Repeat.

4. CREATING YOUR OWN FUNCTIONS

This chapter describes how to use Sanscript to create your own functions.

The first section shows how to construct the Hello World example step by step, gives an overview of datatypes and gives general editing rules for functions.

The second section discusses the use of Language Functions.

The third section discusses the compound datatypes of List, Record, and Global Record.

The fourth section describes how to view and print flowgrams.

Creating the Hello World Example

This section presents a step-by-step procedure to create the Hello World program.

Step 1. Create a function and its flowgram

- 1 On the **File** menu, click **New**, and then **New Flowgram**.
- 2 In the **Properties for Function** dialog box, enter “Hi There” in the **Name** field.
- 3 In the **Description** field, enter “Says Hello.”
- 4 Click the default icon to specify the function’s icon. Choose crankit.wmf as the new icon.
- 5 Click **Open**, then click **OK**.

Step 2. Create a Constant “Hello World”

- 1 Double-click the **Canvas** to create a constant.
- 2 Click “<No Value>” to select it. Enter “Hello World.”
- 3 Press Enter to finish editing the constant’s text string.

Step 3. Insert a “Display Message” on the Canvas

- 1 In the **Overview** Window double-click the **General** file cabinet.
- 2 In the **General** file cabinet double-click the **User Interface** folder.
- 3 In the Catalog, drag the **Display Message** function to the **Canvas** to the right of the constant.
- 4 Click the **Display Message** function on the **Canvas** to select it. Use the mouse to move it where you want it and adjust its size. Then, click the **Canvas** to unselect it.

Step 4. Link the outlet of the constant to the inlet of Display Message

- Drag the outlet of the constant to the inlet of the **Display Message**.
A link from outlet to inlet appears.

Step 5. Run your new function

- On the Run menu, click Start. The “Hi There” function (and program) runs and displays its message.

Congratulations! You’ve created your first flowgram!

Constructing functions from templates and examples

You can quickly construct new functions that are similar to an example or a template, or insert a template onto an existing Canvas. (Because “Hello World” already happens to exist in the Examples and Templates cabinet, the procedures of this section are a faster way to create that particular function than the procedure described earlier in *Creating the Hello World Example*.)

To create a new function from an existing example or template:

- 1 On the **File** menu, click **Program Wizard...**
- 2 In the **Create New Program from Template or Existing Program** dialog box, click the name of the template or example you want to use to create your new function.
- 3 In the **Name** field, keep the name that appears, or enter a new name for your new function.
- 4 In the **Create in** field, keep the name of the file cabinet that appears, or select a different file cabinet.
- 5 Click **OK**.

6 Fill in and edit the new flowgram as needed.

To expand a template in an existing Canvas window:

- 1 Open a function, or otherwise make sure that there is an active **Canvas** window.
- 2 On the **Insert** menu, click **Template**.
- 3 In the **Choose a Template** dialog box, click the name of the template you want to expand onto the active **Canvas**.
- 4 Click **OK**.
- 5 Fill in and edit the expanded template as needed.

Understanding datatypes

Simple datatypes

Data is central to the use of functions and flowgrams. Data is produced and consumed by functions and flows on the links of flowgrams. Sanscript comes with several datatypes built in. A datatype is a name that specifies the kind of information that can appear on an outlet or inlet. Table 9 shows datatypes and their definitions.

Table 9. Datatype definitions

Datatype	Definition	Examples
Integer	Whole numbers	-1, 0, 2, 3, 4 ...
Decimal	Numbers that can have a fractional part appearing after a decimal point	-1.1, 0.0, 3.14159, ...
Text	Letters, numbers, and special symbols that you can type on a keyboard	a, b, c, Hello, \$, %, ...
Boolean	The logical values used in combination with AND, OR, NOT	TRUE, FALSE, Yes, No

The easiest way to create a value of one of the simple datatypes is to create a Constant. A later section discusses compound datatypes, such as Lists, Records, and Global Records.

Datatype gravity and automatic conversions

In the construction of the “Hi There” function, the inlet of the Display Message captured the link from the outlet of the constant. This Sanscript feature, which simplifies flowgram editing, is called “datatype gravity” or simply “gravity.” Gravity identifies the nearest inlet that can accept data from the outlet being

linked. Gravity suggests a connection whenever the datatypes are “compatible.” Two datatypes are compatible if they are the *same* datatype or there is an *automatic conversion* from one datatype to the other.

If Sanscript uses an automatic conversion to make the link, a circle appears on the link. The circle indicates that a conversion is involved, and reminds you to consider the type of conversion. Table 10 lists the simple datatypes and whether they are compatible.

Table 10. Datatype conversion formats

From/To	Boolean	Integer	Decimal	Text
Boolean	Same	—	—	C
Integer	—	Same	C	C
Decimal	—	L	Same	C
Text	L	L	L	Same

Table symbols and meaning

— Types are incompatible; no conversion exists

Same Types are identical and can always be linked

C Conversion from type on left to type on right

L Conversion, but information can be lost

When text is converted to Boolean: “yes,” “true,” and “1” are converted to TRUE, and “no,” “false,” and “0” are converted to FALSE. All other text values are converted to FALSE.

When Decimal is converted to Integer, the number before the decimal point becomes the Integer, and the numbers after the decimal point are lost.

Datatype conversion is a convenience, but the circle is a reminder that not all data can be converted. In the case of the lossy conversions (Text to Boolean, Integer, or Decimal; and Decimal to Integer) a program may not run correctly if critical data is lost during conversion.

General editing rules for functions

This section contains procedures for creating and editing functions.

Methods to view function properties

In general, there are three methods to edit a function’s properties. Use the method that is easiest for you.

- Double-click the function itself.

- On the **Edit** menu, click an editing action.
- Right-click the function, and choose from the context menu.

Creating a function and editing its definition

This section presents the general rules for creating a new function and editing its definition in a flowgram.

To create a function:

- 1 On the **File** menu, click **New**, and then click **New Flowgram**.
- 2 Enter a name for the function in the **Properties for Function** dialog box.
- 3 Select the folder for the function (if the function will go in a folder other than **User**).
- 4 Enter a description for the function (optional).
- 5 Click the default symbol to choose a different symbol for the function.(optional)
The system opens a new canvas for the function.

In order for the function to do something when it runs, it must have a flowgram on its canvas. (A function's flowgram consists of other linked functions on its canvas.)

To find a function and place it on the canvas:

- 1 Double-click the symbols (file cabinets or folders) in the **Overview** window until the function you want appears in the **Catalog** window.
- 2 Drag the function from the **Catalog** to the **Canvas**.
- 3 Click the **Canvas** to deselect the function.

If you make a mistake, use the **Undo** command on the **Edit** menu to go backwards in editing history. Used repeatedly, **Undo** can remove any changes made since the previous **Save** command.

To undo and redo edits:

- On the **Edit** menu, click **Undo**.
- On the **Edit** menu, click **Redo**.

To link functions:

- Drag a link from the outlet of the first function to the inlet of the second function.

To increase or reduce the size of a function:

- 1 Select the function you want to resize.
Resize handles appear to indicate the function is selected.
- 2 Drag the handle in the direction you want it to move.

Note: The red resize handle on some functions is used to expand the area occupied by the function but not its picture. Use the red resize handle to see more inlets or outlets when a function has many of them. Some functions are not resizable. Double-clicking them has no effect.

To move a function:

- 1 Click the function to select it.
Resize handles appear to indicate the function is selected.
- 2 Drag the function to position it where you want it on the flowgram.

To delete a function:

- 1 Click the function to select it.
Resize handles appear to indicate the function is selected.
- 2 On the **Edit** menu, click **Delete** or **Cut**, to remove it.

To edit the definition of a function:

- Double-click the function.
The flowgram of the function is displayed and can be edited.

Note: Some functions cannot be edited. For those functions, double-clicking has no effect. Most of the functions in the General cabinet, for example, cannot be edited.

To save a function

Because flowgrams link to other flowgrams that you may have edited, Sanscript can only save all functions; it cannot save a single function.

To save all open and edited flowgrams:

- On the **File** menu click **Save All**.

Note: The language functions do not have a flowgram to edit.

Using inlets, outlets, and connectors

Use inlets to bring data into a function, and use outlets to bring data out of a function. For each inlet, there is an inlet connector inside the function, and for each outlet there is an outlet connector inside the function. Use the connectors to bring data into or out of the flowgram for the function.

Depending on the kind of function, there are special rules for adding inlets, outlets, and connectors. Table 11 describes these special rules.

Table 11. Inlet, outlet, and connector rules

If the function is:	You can add:				
	Inlets	Outlets	Inlet/ Outlet Pairs	Inlet Connectors	Outlet Connectors
Constant, or Comment, or Stop Repeat	—	—	—	—	—
Form	—	—	—	—	—
Package	✓	✓	—	✓	✓
Pick One	✓	✓	—	✓	✓
Repeats	✓	—	✓	✓	—
User-defined functions	—	—	—	✓	✓

To add an inlet connector or outlet connector to a flowgram:

- 1 Open the function and view its flowgram.
- 2 On the **Insert** menu, click **Connector**, then click **Inlet Connector** or **Outlet Connector**.
The inlet connector or outlet connector appears on the flowgram and the inlet or outlet appears on the outside of the function.

To add an inlet or outlet to a function:

- 1 Click the function to select it.
- 2 On the **Insert** menu, click **Inlet/Outlet**, then click either **Inlet** or **Outlet**.
The inlet or outlet appears on the outside of the function, and the inlet connector or outlet connector appears in its flowgram (if any).

To add an inlet/outlet pair to Repeats (only):

- 1 Click the Repeat to select it.
- 2 On the **Insert** menu, click **Inlet/Outlet**, and then click **Inlet Outlet Pair**.
The paired inlet/outlet appears on the outside of the function, and the paired inlet connector and outlet connector appears on the inside. The connectors are initially linked so that data will automatically circulate each cycle of the repeat.

To view or change the datatype and default value of an inlet or outlet:

- 1 Double-click the function to open it.
- 2 Double-click the inlet connector or outlet connector.
- 3 Select a datatype from the list.
- 4 On inlet connectors, check the box if you want the inlet to have a default value, and then enter the value in the Default Value box.
- 5 Click **OK**.

Using Sanscript language functions

This section explains how to use the language functions. The language functions are fundamental to constructing programs.

To insert a language function in a flowgram:

Use one of the following methods to insert language functions.

- On the **Insert** menu, click the language function you want to insert.

- Right click a blank area of the **Canvas**, and select a command from the context menu.
- Locate the **Language** folder in the **Catalog** and drag a language function onto the **Canvas**.

Comment



Use comments in a flowgram to describe the flowgram: what it does, how it works, where its results appear, special inputs it may need, limitations on its behavior, changes pending, and so on.

To insert a comment in a flowgram:

- 1 On the **Insert** menu, click **Comment**.
- 2 Select the text “Enter comment here. “
- 3 Press the Delete or Backspace key to remove the “Enter comment here” text.
- 4 Enter your comment text.
Press the Enter key to create multiple lines of comments.
- 5 To end a Comment, click the **Canvas**.

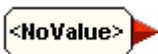
There are three classes of Comment: Comment, Bug, and Todo. The only difference between these is the format (font, color, label) of the comment.

To change the style of a Comment:

- 1 Double-click the Comment (but not its text) .
- 2 In the **Edit Comment Style** dialog box, click the arrow to select **Comment**, **Bug**, or **Todo**.
- 3 Click **OK**.

To move or delete a Comment, see “ General editing rules for functions” on page 28. Comments cannot be resized.

Constant



Use a constant in a flowgram to generate a constant value. The following list shows valid constant values:

Valid constant values

Integer	Decimal
Text	Boolean
Time	Pathname

List of text

List of Pathname

To insert a constant in a flowgram:

- 1 On the **Insert** menu, click **Constant**; or, from the **Language** folder, drag a Constant symbol to the **Canvas**.
- 2 Click **No Value** to select the value field.
- 3 Enter the value of the constant. The datatype of the value is chosen automatically by Sanscript based on what you type. For example, type:
 - 0 for integer zero
 - 0.0 for decimal zero
 - letters, numbers, or typographical symbols for a text value
 - 12/21/99 for 31 December 1999

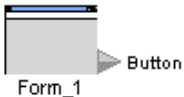
Constants can have several datatypes. To be sure you have given the value the correct datatype, edit the definition of the constant.

To edit the definition of a Constant:

- 1 Double-click the Constant (but not the value text).
- 2 In the **Edit Value** dialog box, click the arrow next to the datatype field to select the proper datatype for the value. You can also change the value using this dialog box.
- 3 Click **OK**.

To move or delete a Comment, use the general editing procedures described on page 28. You cannot resize a comment.

Forms



Use a form in a flowgram to generate a user interface form, a dialog box.

To insert a Form in a flowgram:

- On the **Insert** menu, click **Form**, or, from the **Language** folder, drag a **Form** icon onto the **Canvas**.

Table 12 shows the valid form fields and their uses:

Table 12. List of form fields

Form Field	Uses and Behavior
Push button	When clicked by the user, sends its button name to the Button outlet of the form and dismisses the form.
Check box	Receives TRUE or FALSE on its inlet and checks or unchecks the box accordingly. When the form is dismissed, sends TRUE to its outlet if the box is checked. Sends FALSE to the outlet if the box is not checked.
Label	Displays its text on the form. There is no inlet or outlet.
Text box	Receives text value on its inlet and displays it in the text box. The user can type into the text box while the form is displayed. Sends the value of the text box to its outlet when the form is dismissed.
List box	Receives on its inlet and displays on the form a list of text values. The user can click to select one of the values. Sends a list of selected text values to its outlet. If the user did not select any value, sends the empty list string to the outlet.

To insert a Form Field into a Form:

- 1 Click the form to select it.
- 2 On the **View** menu, click **Options**, then click the **Flowgram** tab.
- 3 Confirm that the **View Labels** option is on, and click **OK**.
- 4 On the **Insert** menu, click **Form Field**, then click the field you want to insert in the form.
- 5 Click each inlet or outlet label to specify the name of the field or data.
- 6 Repeat step 5 for each field.

To make a Form Field display a value:

- Link Constants or other functions to the form's inlets.

To modify the position or size of the fields on the form:

- 1 Double-click the form to open the form editor.
- 2 Use your mouse to resize and position the objects in the form.

To move, delete, or resize a Form, use the general editing procedures described on page 28. Double-click objects in a Form to edit their properties.

Package



Package_1

A package is a language function that contains a flowgram. Use a package to simplify a complex flowgram. You can either replace a portion of the flowgram with a package in one step, or you can insert a package into a flowgram and then build up the contents of the package. In either case, use inlets and outlets to get data into or out of the package.

To replace a portion of a flowgram with a package:

- 1 Click each function on the flowgram that you want in the package. Alternatively, click the canvas and drag a selection box around all the functions to include them all in the package.
- 2 On the **Insert** menu, click **Replace**, then click **with Package**.

To undo the replacement with a package at any time:

- 1 Click the package.
- 2 On the **Insert** menu, click **Replace**, then click **Expansion**.

To insert a package into a flowgram without replacing a portion of the flowgram:

- 1 Open the flowgram in which you want to place the package.
- 2 On the **Insert** menu, click **Package**.

To view or change the contents of a package:

- 1 Double-click the package to open it.
- 2 Edit the contained flowgram to change its layout or its calculation.
- 3 Close the window to close the package.

To move, resize, or delete a Package, use the general editing procedures on page 28.

Pick One



A **Pick One** is a language function that chooses among several flowgrams contained within it based on data arriving on its special selector inlet. Use a Pick One in a flowgram to make a decision among different calculations.

Setting a Pick One datatype

Before using a Pick One, you must set the datatype of the selector inlet. The selector is colored red when it has no datatype set, and gray when the datatype has been set. Pick One datatypes are limited to:

- Boolean
- Decimal
- Integer
- Pathname
- Text

A Pick One can “learn” its datatype from any data that it receives. You can set a Pick One datatype by linking it to the correct outlet, or by editing its properties.

To insert a Pick One into a flowgram:

- 1 Open the flowgram into which you want to place the Pick One.
- 2 On the **Insert** menu, click **Pick One**.

To automatically set the datatype of a Pick One:

- 1 Insert a new Pick One into a flowgram.
- 2 Drag a link from some outlet with a defined datatype (but not datatype “***any***”) to the selector inlet of the Pick One.

To manually set the datatype of a Pick One:

- 1 Insert a new Pick One into a flowgram.
- 2 Click the Pick One to select it.
- 3 On the **Edit** menu, click **Choices**.
The **Edit Choices** dialog box appears.

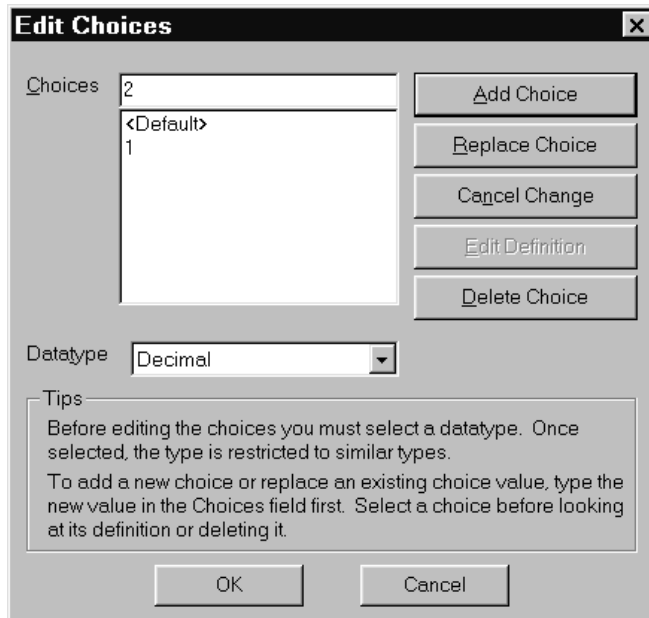


Figure 21. Pick One Edit Choices dialog box

- 4 In the **Edit Choices** dialog box, click the arrow on the datatype box and choose the datatype of the Pick One and its selector inlet.
- 5 Click **OK**.

Setting Pick One choices

After choosing the datatype of the Pick One, you must specify the number and value of the choices for the Pick One. (The number of choices is the same as the number of flowgrams within the Pick One.) The value of a choice is the value of the selector inlet that will cause the flowgram for that choice to run.

If you set the datatype of the Pick One to a datatype with a small number of possible values, the Pick One automatically creates a choice for each value of that datatype. For example, if you link a Boolean outlet to the selector inlet of a Pick One, the Pick One will not only have the datatype Boolean, but it will also automatically have all possible choices: TRUE and FALSE.

For other datatypes, such as decimal, integer, or text, the number of possible values is high, so the Pick One will initially show only one choice: <Default>. You must specifically add other choices for these datatypes.

To add choices to a Pick One:

- 1 Click the Pick One to select it.
- 2 On the **Edit** menu, click **Edit Choices**, or right-click the Pick One and click **Edit Choices** from the context menu.
- 3 Set the datatype of the Pick One.
For directions on setting the datatype, see page 37.
- 4 In the **Edit Choices** dialog box, enter the value for a new choice in the **Choices** entry box.
- 5 Click **Add Choice**.
- 6 Repeat steps 4 and 5 to add more new values.

To delete choice values in a Pick One:

- 1 Click the Pick One to select it.
- 2 On the **Edit** menu, click **Edit Choices**, or right-click the Pick One and click **Edit Choices** on the Context menu.
- 3 In the **Edit Choices** dialog box, click the value that you want to delete in the **Choices** list box.
- 4 Click **Delete Choice**.
- 5 Repeat steps 3 and 4 to delete existing choice values.

Note: When you delete a choice value from a Pick One, the flowgram for that choice is deleted. You may want to copy the flowgram to a new Package to preserve it.

To replace choice values in a Pick One:

- 1 Click the Pick One to select it.
- 2 On the **Edit** menu, click **Edit Choices**, or right-click the Pick One and click **Edit Choices** from the context menu.
- 3 In the **Edit Choices** dialog box, click the value in the **Choices** list box that you want to replace with a new value.
- 4 Enter the new value for that choice in the **Choices** entry box.
The new value must not already be a choice.
- 5 Click **Replace Choice**.
- 6 Repeat steps 3 and 4 to replace existing choice values with new values.

When you replace a choice value with a new value, the flowgram for the original choice is preserved and becomes the flowgram for the new choice.

To edit the flowgram for an existing choice in a Pick One:

- 1 Set the datatype and choice values for the Pick One, as described on page 37.
- 2 Double-click the Pick One, or right-click the Pick One and click **Edit Definition**.
- 3 Click the tab for the choice whose flowgram you want to edit. If the tab for the choice does not appear, click the right or left arrows to slide the tabs over.
- 4 Edit the flowgram for the choice.
- 5 Close the window showing the flowgram.

Repeat



A Repeat is a language function that repeatedly runs one flowgram for some number of times. The data value that arrives at the function's special repeat determines the number of times the flowgram repeats. Use this language function to repeat a calculation: Examples of repeat functions include:

- a fixed or variable number times equal to the value of an integer
- once for each item in a list of items
- an unknown number of times controlled by the repeating flowgram itself

Before using a Repeat, set the datatype of the repeat inlet. The best way to set the datatype of the Repeat is to link its repeat inlet to the outlet of another function which has the correct datatype. Doing so causes the repeat inlet of the Repeat to automatically learn its datatype from the other outlet.

Another way to set the datatype of the Repeat is to open it before connecting any link to its repeat inlet. Inside the repeat is a special repeat Item connector. You can change the datatype of the Repeat by changing the datatype of this connector.

If you do not set the datatype of the Repeat, or you do not connect a link to its repeat inlet, then the Repeat becomes an “infinite loop.” To prevent such a loop from occurring, insert a function somewhere in the flowgram for the Repeat that will eventually stop it. The functions that can do this are:

- Stop Repeat
- Exit
- Error!

To insert a Repeat into a flowgram:

- 1 Open the flowgram into which you want to place the Repeat.
- 2 On the **Insert** menu, click **Repeat**.

To manually set the datatype of a Repeat:

- 1 Insert a new Repeat into a flowgram.
- 2 Double-click the Repeat to open it.
- 3 Double-click the “Item” connector to edit its definition.

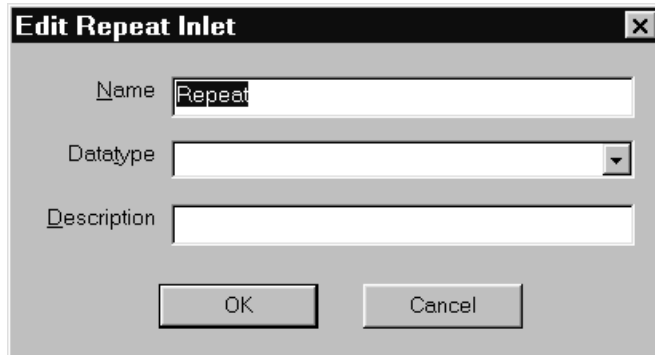


Figure 22. Edit Repeat Inlet dialog box

- 4 Click the arrow on the **Datatype** box to choose the datatype to be connected to the repeat inlet.
- 5 Enter a name in the **Name** box, and a description in the **Description** box.

To automatically set the datatype of a Repeat:

- 1 Insert a new Repeat into a flowgram.
- 2 Drag a link from an outlet to the repeat inlet of the Repeat.

When the datatype of a repeat is set, the symbol for the repeat changes to indicate the datatype of the repeat.

To view or edit the flowgram of a Repeat:

- 1 Double-click the Repeat.
A window opens to display the flowgram for the Repeat.
- 2 Edit the flowgram.
- 3 Close the editing window.

To cause a Repeat to stop:

- 1 Double-click the Repeat.
- 2 On the **Insert** menu, click **Stop Repeat**.

- 3 Link the inlet of the Stop to the outlet of a calculation that generates the value TRUE when the Repeat should stop.
- 4 Alternatively, to stop the whole program as well as the Repeat, from the System folder insert function Exit, or, from the Error folder insert function Error!.

Errors

An Error is a special value that defines a particular error. An error value includes information fields to identify the error that occurred. Figure 23 shows how to use the Error! function to create an error value.

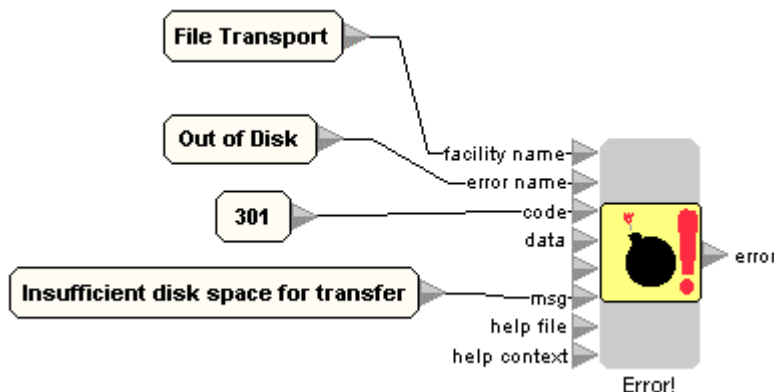


Figure 23. Error function

If an error value occurs on an unlinked outlet of a function, Sanscript signals the error. (You can think of the error on an unlinked outlet as having “dropped on the canvas”.) The error then starts traveling, seeking an Error Handler. If there is no handler in the flowgram, it travels backwards from the flowgram to its function, to the flowgram that the function is in, and so on, continuing to all outer functions and flowgrams. If the error is not handled by any of these, the Error Handler terminates the application or program and displays a message.

Note: An error behaves like an “exception” in other languages.

To create an error value:

- 1 Use the **Overview** to open the **General** file cabinet, and then the **Error** folder.
- 2 Drag the **Error!** function into a flowgram.
- 3 Create the following inputs to Error! to identify the error:

Table 13. Valid Error input datatypes

Inlet	Datatype	Description
Facility	Text	The name of the software declaring the error.
Error Name	Text	Name of this kind of error.
Code	Integer	Number you can use to exactly identify the error.
Msg	Text	Message text that identifies the error in more detail. This can be constructed and provide any information you'd like.
Help File	Text	Name of a help file that provides more information about the error.
Help Context	Text	A "help id" that identifies a particular topic in the help file.

Note: If you create a special function to declare an error or all your errors, you can easily reuse error functions, including inside Error Handlers, to compare error values.

Error Handling



Error Handler

An Error Handler is a language function in a flowgram that tracks and corrects errors. When an error occurs, the Error Handler substitutes its own internal flowgram, including corrections, for the flowgram on which it is placed. The internal flowgram has the same inlets and outlets as the parent flowgram.

Programs may contain several Error Handlers. If no Error Handler handles (corrects) an error, the error bubbles up through all the functions running when the error occurred and terminates the program or application.

To insert an error handler in a flowgram:

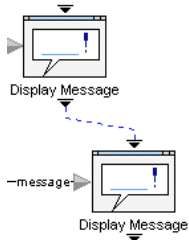
- 1 Open the function where you want to place the Error Handler.
- 2 On the **Insert** menu, click **Error Handler**, or drag an **Error Handler** from the **Language** folder of the Catalog to the canvas.

To specify corrective actions inside an error handler:

- 1 Double-click an Error Handler to open it.
- 2 Use the special **Error** inlet connector to get the value of the error, and use the functions in the Error folder of the Catalog to compare the error to various error values you can correct.

- 3 Drag and drop functions onto the flowgram of the Error Handler to perform the corrective calculation.
- 4 Link to the outlets of the Error Handler to substitute values for the outlets of the function.
- 5 Close the Error Handler.

Ordering links



An *ordering link* links two functions and ensures the second function runs only after the first completes its operation.

Use ordering links to ensure that functions with side effects (such as opening windows) are done before a second function, which depends on that side effect, begins operation.

An ordering link connects from the ordering outlet of one function to the ordering inlet of another. The ordering inlet is a triangle on the top of function icons. The ordering outlet is the triangle on the bottom of function icons.

Ordering links are not shown by default.

To view ordering links:

- 1 On the **View** menu, click **Options**.
- 2 Click **Flowgram**, then check **View Ordering Links**.

To draw an ordering link:

- Drag a link from the ordering outlet of the first function to the ordering inlet of the second function.

Transforming Language Functions

The Sanscript tool supports certain language transformations for ease in editing and to help scale up to solve large problems. Table 14 shows problems and how they are solved using language transformations.

Table 14. Transforming language functions

Problem/Goal	Solution
Simplify a large flowgram	Replace a portion of a flowgram with a package
Turn a portion of a flowgram into a reusable function	Replace a portion of a flowgram with a function
Eliminate a package or function usage without changing the calculation	Replace a function or package with its expansion
Copy the contents of a package or function into more than one flowgram	Replace a function or package with its expansion

To replace a portion of a flowgram with a package:

- 1 Click each function on the flowgram that you want to go into the package, or drag a selection box around all the functions.
- 2 On the **Edit** menu, click **Replace**, then click **with Package**

To replace a portion of a flowgram with a function:

- 1 Click each function on the flowgram that you want to go into the package, or drag a selection box around all the functions.
- 2 On the **Edit** menu, click **Replace**, then click **with Flowgram**.
A new function is created and placed in the User file cabinet.

To expand a function or package in place:

- 1 In the flowgram where you want to expand the function or package, click the function or package.
- 2 On the **Edit** menu, click **Replace**, then click **with Expansion**.
The use of the package or function is replaced with contents of the function or package.

Compound datatypes

In addition to the simple data types of integer, decimal, text, and pathname, the Sanscript tool can group simple datatypes to form complex datatypes of *List* and *Record*. These compound datatypes add power and simplicity.

Lists

Use a list to group a set of similar items. For example, use a list to represent a series of numbers, or a series of lines in a file, or a list of filenames, and so forth.

To create a list from one value and a count:

- 1 In the **General** file cabinet, open the **Lists** folder.
- 2 Drag the **List Of** function onto the **Canvas**.
- 3 Link an outlet with an integer value to the **Length** inlet.
- 4 Link an outlet with the value to be used for each item in the list, to the **Init** inlet.

To create a list from a set of values:

- 1 In the **General** file cabinet, open the **Lists** folder.
- 2 Drag the **Make List of Items** function onto the **Canvas**.
- 3 Link an outlet whose value is to be the first item in the list to the **items** inlet. A new outlet appears.
- 4 Repeat step 3 for each item that is to go into the list.

Records

Use a *record* to group dissimilar but related items. For example, use a record to represent a set of information about an employee, such as the employee's name, social security number, phone number, age, office number, and so forth. These are dissimilar datatypes (integer and text) but are related by a common external reference.

To create a new record datatype:

- 1 On the **File** menu, click **New**, then click **Record**.
- 2 In the **Define a new record type** dialog box, enter a **Name** for the new type.
- 3 Select the **Cabinet** where the datatype information belongs. By default, this is the **User** cabinet.

- 4 Click **OK**.
- 5 In the **Record Fields** dialog box, click **New Field**.
A new row appears in the record definition area.
- 6 Specify the **Name** for the field, the **Type** of the field, and the default value.
- 7 Enter **Yes** in the **Set?** column if the field can be changed by a running flowgram.
Enter **No** in the **Set?** column if the field cannot be changed.
(If the field can be changed an inlet will be added to set the field.)
- 8 Enter a **Description** of the field.
- 9 Enter the name of a help file and the help context key which provides information about this field.
This optional entry supplies context-sensitive help.
- 10 Click **External Data** if the data is sent to or received from a function external to Sanscript. Then, specify the **External Type**, **Byte**, and **Size** fields to match the external data type.
- 11 Click **OK**.

When you create a record datatype, two new functions are available in the file cabinet where they are kept: **Make** and **Fields**. Use the **Make**<record name> function to create records of this new datatype. Use the <record name>**Fields** function to get set the fields of the record or get the fields of a record.

Managing records and their fields

To create a new record:

- 1 Open the file cabinet holding the **Make**<record name> function.
- 2 Drag the **Make**<record name> function onto a flowgram.
- 3 Connect outlets having the proper datatypes to the inlets of the **Make** function.
The outlet of the **Make** function generates the record value.

To set the fields of a record:

- 1 Open the file cabinet holding the **Make**<record name> function.
- 2 Drag the **Make**<record name> function onto a flowgram.
- 3 Link a record value to the record (top) inlet of **Make**<record name>.
(Leave the record inlet unlinked to create a record with default values.)

- 4 Link the fields inlets of the **Make**<record name> function for each field that you want to modify to outlets having the proper datatypes.
The outlet of the **Make**<record name> function generates the new record value.

To get the fields of a record:

- 1 Open the file cabinet holding the <record name>**Fields** function.
- 2 Drag the <record name>**Fields** function onto a flowgram.
- 3 Link a record value to the record (top) inlet of <record name>**Fields** .
(Leave the record inlet unlinked to create a record with default values.)
- 4 Link to the outlets of <record name>**Fields** to obtain the values of the fields of the record.
The outlet of the Fields function generates the record value.

Some records have many fields, and you may only need to set or get a few fields at any given point in your flowgram. To make this easier, you can create custom record access functions that have just the inlets and outlets you need. Use the custom record access functions just as you would the default **Make** or **Fields** functions.

To create a custom record accessing function:

- 1 Open the file cabinet holding the <record name>**Fields** function.
- 2 Drag the <record name>**Fields** function to a blank area of the Catalog.
- 3 Select **Copy** in the **Drag-Drop Operations** dialog box.
- 4 Double-click the new copy of the **Fields** function to open its definition.
- 5 Select the **Set** option in the row for any field you need to set.
(Set causes an inlet to appear on the customized function).
- 6 Select the **Get** option in the row for any field you need to get.
(Get causes an outlet to appear on the customized function).
- 7 Right mouse click the new fields function.
- 8 Choose **Properties....**
- 9 Change the name of custom fields function, as well as any other properties you need to change.
- 10 Click **OK**.
Drag the custom access function into any flowgram in which you want to use it.

Note: To create custom access functions, start with the Make function (to set many fields) or with the Fields function (to set many fields).

Global Records

A global record is one whose values can be set or obtained anywhere in a program. With (non-global) records, each use of the Make function creates a new record. With global records, one copy of the global record is shared by the entire program.

To create a new global record:

- 1 On the **File** menu, click **New**, then click **New Global**.
The **Define a New Global Record** dialog box appears.
- 2 Type a name for the global record in the **Name** box.
- 3 Choose a file cabinet to hold the global record in the **Cabinet** box.
- 4 Click **OK**.
- 5 Specify the **Name**, **Initial Value**, **Type**, and **Description** for the first field of the global record.
- 6 Click **New Field** to add additional fields.
- 7 Close the global record field definition dialog box.

To set or get the values of a global record:

- 1 Use the **Overview** and **Catalog** windows to view the global record in its file cabinet.
- 2 Drag the global record to a flowgram.
- 3 To set fields of the global record, link functions to the inlets of the global record function.
- 4 To get fields of the global record, link functions to the outlets of the global record function.

Note: Global record properties make it easy to get the data in a global record from any function in the program without having to pass the record through outlets, links, and inlets. However, this capability demands careful design: because any function in your program can set the global record, and any function can get its values, you must use other means of ordering the functions to ensure that their sets and gets occur in the correct order. Often, other data flowing through your functions provides the needed ordering. When this is not the case, you can use ordering links. Because global records generate this additional design work, use them sparingly.

Viewing and printing Flowgrams

Viewing large Flowgrams

Zoom controls change the magnification of the Canvas so you can see more or less of a large flowgram. You can:

- Zoom out (decrease magnification)
- Zoom in (increase magnification)
- Set zoom magnification back to normal
- Zoom so that the entire flowgram fits within the Canvas window
- Get a bird's eye view and then select a portion of the flowgram to view

To zoom out to see more of a large flowgram:



- On the toolbar, click the zoom-out button.

To zoom in to see more details:



- On the toolbar, click the zoom-in button.

To set zoom magnification back to normal:



- On the toolbar, click the normal size button.

To set zoom magnification so that the entire Flowgram fits in the Canvas:



- On the toolbar, click the zoom-to-fit button.

To get a bird's eye view and then zoom in:



- 1 On the toolbar, click the bird's eye zoom button.
- 2 Find the portion of the flowgram you want to zoom into.
- 3 Use the mouse to drag a rectangle around the portion of the flowgram that you want to zoom into.

Viewing and editing function properties

Functions have a default symbol. You can change the symbol to one that is more appropriate. You can either choose existing symbols in the Clipart folder, or you can create your own symbols using any drawing tool that can create Windows metafiles (file type .wmf).

To change the symbol for a function:

- 1 Select the function in the **Catalog** or on the **Canvas**.
- 2 Right-click the function, then click **Properties**, or, on the **Edit** menu, click **Properties** to open the **Properties for <function name>** dialog box.

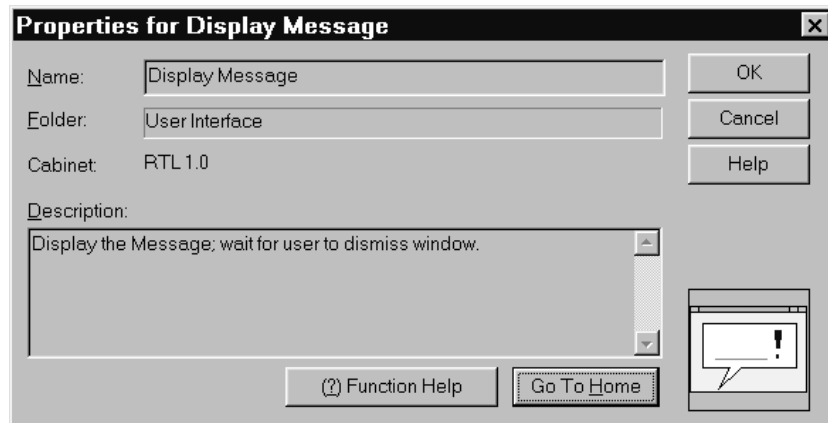


Figure 24. Properties for <function> dialog box

Note: If you are not allowed to change the function the properties dialog box may be disabled.

- 3 Click the symbol now being used to represent the function.
The **Open** dialog box shows the contents of the Clipart folder.
- 4 In the **Open** dialog box select the filename of a symbol appropriate for the function, then click **Open**.

To change the name, description of a function:

- 1 Select the function in the **Catalog** or on the **Canvas**.
- 2 Right-click the function, then click **Properties**, or, on the **Edit** menu, click **Properties** to open the **Properties for <function name>** dialog box.
- 3 Enter the new name or description in the text boxes.

The “home” location for a function is the folder in the Catalog where the function is stored. Go to a function’s home location if you want to see what file cabinet and folder the function resides in, or to delete the function.

To open the “home” folder for a function:

- 1 Select the function in the **Catalog** or on the **Canvas**.
- 2 Right-click the function, then click **Properties**, or, on the **Edit** menu, click **Properties** to open the **Properties for <function name>** dialog box.
- 3 Click **Go to Home**.

To view all the functions used by a function:

- 1 Select the function in the **Catalog** or on the **Canvas**.
- 2 Click the function to select it.
- 3 On the **View** menu, click **Functions Used** to open the **All Functions Called** dialog box.



Figure 25. All Functions Called dialog box

- 4 Select a function you want to view in detail, and click **View**, or click **Close**.

To view all the functions using a particular function:

- 1 Select the function in the **Catalog** or on the **Canvas**.
- 2 Click the function to select it.
- 3 On the **View** menu, click **Functions Using** to open the **References To** dialog box.



Figure 26. References To <function> dialog box

- 4 Select a function you want to view in detail, then click **View**.

Improving the appearance of Flowgrams

You can make complex flowgrams, with many crossing links, easier to view by rearranging the order of inlets or outlets on certain functions. Rearrangement can be either manual or automatic.

To rearrange inlet or outlet order:

- 1 Double-click the function to open its flowgram on the canvas.
- 2 Move the inlet connectors or outlet connectors in the vertical dimension in the order in which you want their inlets or outlets to appear on the outside of the function.
- 3 On the **Edit** menu, click **Inlet/Outlet Order**.

To get an automatic layout of a flowgram:

- 1 Double-click the function to open its flowgram on the canvas.
- 2 On the **Edit** menu, click **Arrange**.

Printing Flowgrams

To print a flowgram:

- 1 Open the function, or otherwise make sure that the flowgram's canvas is the active canvas window.
- 2 On the **File** menu click **Print Setup**. Specify the printer settings.
- 3 On the **File** menu click **Print**.

5. RUNNING AND TESTING FUNCTIONS

Running, pausing, and stopping a function

You can start a program running and stop it. You can step one function at a time through a running program to watch every function it calls and monitor all the data. You can also set pause points within a program so that the program will freeze at those points.

When a program pauses, either because it has hit a pause point or because you are stepping through it, you can examine the values of the data on inlets or outlets by placing your mouse pointer over the inlet or outlet.

By using these methods, you can test your program's functions.

Running, pausing, and stopping a program:

To run a function:

- 1 Open the function and view its flowgram.
- 2 On the **Run** menu, click **Start**.

To pause a running function immediately:

- On the **Run** menu, click **Pause**.

To continue running a paused function:

- Click **Run**.

To stop a running function:

- On the **Run** menu, click **Stop**.

Browsing a running program

When you run a program, the function you choose to start the program is shown in the Overview window under the second (or Program) tab. The function

remains under the Program tab until you run a different program or until you use Run-Set Current Program.

You can view all the functions used by the most-recently run program by clicking the name of the program and any other names that appear in the Overview window. Use the Program area of the Overview window to browse the program just as you use it to browse file cabinets and folders.

Stepping through a function

To step through the running of a function:

- 1 Open the function and view its flowgram.
- 2 On the **Run** menu, click **Step**, or press F10.
- 3 Click **Step** as necessary to execute functions in the program.
- 4 Place your mouse over inlets or outlets of functions that have run or are about to run to monitor the values being created in the flowgram.

To step into the running of a function:

- On the **Run** menu, click **Step Into** or press F8.

To step out to the calling function:

- On the **Run** menu, click **Step Out Of** or press Shift-F7.

Setting and clearing pause points

When you are testing a program, you can cause the program to pause at particular points to see what values the program is generating.

Set as many pause points in your program as you want. Each time you run the program, it pauses at each pause point.

To set a pause point on a particular function:

- 1 In the flowgram containing the function reference, click the function at which you want the running program to pause.

2 On the **Run** menu, click one of the following.

Pause Before Function

Pause Before Any Call

Pause After Function

Pause After Any Calls

To clear pause points:

- 1 In the flowgram containing the function reference, click the function that has the pause set.
- 2 On the **Run** menu, click **Clear Pause on Function**.

To clear all pauses:

- On the **Run** menu, click **Clear All Pauses**.

Examining values in a paused program

You can examine the data values on the inlets or outlets in a paused program. You can examine values can be examine only if the function that generates them has run. Use stepping or pause points described on page 56 to force the program to pause.

To examine the values in a paused program:

- Place your mouse pointer over any inlet or outlet.
The value at that inlet or outlet appears in a floating box and in the **Status** bar.

Sometimes the data values being examined are long and take up too much space to be displayed in a hover box.

To examine long values in a paused program:

- Right-click the inlet or outlet to open a dialog box to examine the value.

6. WORKING WITH THE CATALOG

Customizing the Catalog

The file cabinets of the Catalog store all the functions you can use to construct programs. You can organize and reorganize the contents of the Catalog and file cabinets to:

- Create new folders.
- Copy or move functions from one folder to another.
- Copy or move functions or folders from one file cabinet to another.
- Create aliases (like Windows shortcuts) that let functions appear in more than one folder for easy access.
- Delete functions, folders, and cabinets.
- Change properties of functions, folders, and cabinets.

An alias in the Catalog is like a Windows shortcut. An alias displays a function or folder in a folder other than its original (or home) folder. Use aliases to place frequently-used functions in your frequently used folders. If the name of a symbol in the Catalog appears in italic type, the symbol is an alias.

Creating folders

To create a new folder:

- 1 Use the **Overview** or **Catalog** to open the file cabinet and/or folder in which you want to create the new folder.
- 2 On the **File** menu, click **New**, then click **New Folder**.

Note: As a shortcut, right-click the Catalog window, then click **New Folder** from the context menu.

Moving or copying functions or folders

Five kinds of move and copy are available to allow you to rearrange folders or functions in the Catalog. Table 15 lists those options, related shortcuts, and their functions.

Table 15. Drag and Drop operations

Operation	Shortcut	Description
Move	Shift+drag	Moves the function from the source folder to the destination folder
Copy	Ctrl+drag	Makes a copy of the function in the destination folder.
Create Alias	Ctrl+Shift+drag	Creates an alias (pointer) in the destination folder that points to the source folder.
Move Dependent	Alt+Shift+drag	Moves the function and selected functions it calls (except for the Sanscript run-time system) to the destination folder.
Copy Dependent	Alt+Ctrl+drag	Copies the function and selected functions it calls (except for the Sanscript run-time system) to the destination folder.

Note: Each function or datatype is unique, different even from datatypes or functions of the same name. Move preserves the unique identity. Copy creates a new identity for the new function or datatype. A copied datatype is different and incompatible with its source datatype. Move removes the function from its source folder; Move Dependent can remove *many* functions from their source folders.

To move or copy a folder or function to a different folder or file cabinet:

- 1 Use the **Overview** to expand the file cabinet and folders to view the folder or file cabinet where you want to move the item.
- 2 Use the **Overview** again to view the folder or function you want to move.
- 3 Drag the folder or function to the destination folder in the **Catalog** window, or to the destination folder or file cabinet shown in the **Overview** window, to open the **Drag-Drop Operation dialog box**.

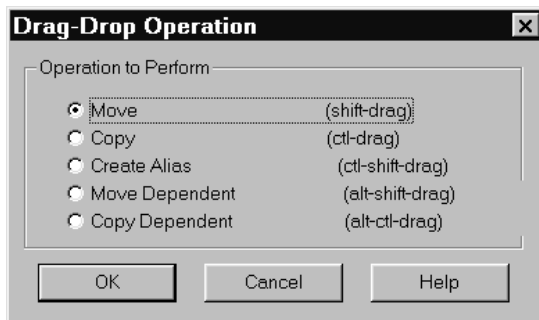


Figure 27. Drag-Drop Operation dialog box

- 4 Click the operation you want to perform. When you select Move Dependent, or Copy Dependent, the **Confirm Dependent Functions** opens. Figure 28 shows the dependent functions for the “Do you want to say hello?” example.

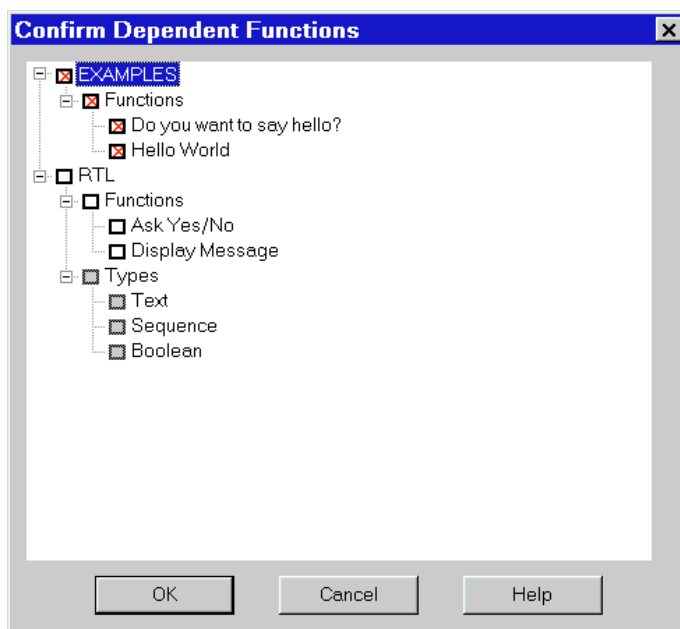


Figure 28. Confirm Dependent Functions dialog box

- 5 Select check boxes for those functions you want to copy or move.

Note: Deselect common functions or datatypes that are present on a destination computer system. For example, the RTL (run time library) is present by default on all systems that have Sanscript installed.

Deleting functions or folders

To delete a folder or function:

- 1 Use the **Overview** or **Catalog** to view the folder or function that you want to delete.
- 2 Select the folder or function.
- 3 On the **Catalog** menu, click **Delete**, or, right-click the folder or function and on the context menu click **Delete**.

Note: If you cannot delete an item, the Delete command is not available.

Changing properties of a folder, or a function, or a file cabinet

To examine or change properties of a folder or function:

- 1 Select the folder or function in the **Catalog** that you want to change.
- 2 On the **Catalog** menu, click **Properties**, or, right-click the folder or function and on the context menu click **Properties**.

To examine or change properties of a cabinet:

- 1 Select the file cabinet in the **Overview** that you want to change.
- 2 On the **Catalog** menu, click **Properties**, or, right-click the white space in the Catalog window and on the context menu click **Properties**. Figure 29 shows where to click to on the white space of the **Catalog**.

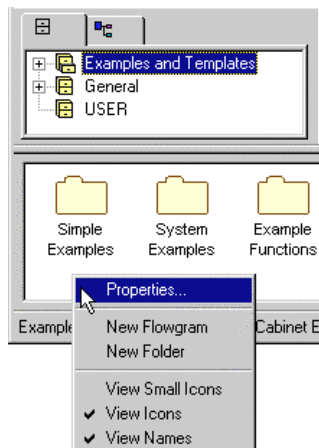


Figure 29. Editing the properties of a file cabinet

Sharing functions with others

Using file cabinets

Use File Cabinets to share functions with others. File cabinets are stored in separate Windows folders (directories) and can be copied to other computers. Operations on file cabinets are:

- Creating cabinets—to form groups of functions.
- Deleting cabinets.
- Exporting Cabinets—to package a file cabinet for use by someone else.
- Importing Cabinets—to get a copy of a cabinet from someone else.

Defined file cabinet locations

By default, file cabinets are stored in folders within the User folder Sanscript created for you at installation. The user folder is typically located in

```
C:\Program Files\Northwoods Software\Sanscript
```

Table 16 describes the folders within the User folder that Sanscript uses to hold file cabinets and applications.

Table 16. Folder names and descriptions

Folder name	Description
User\User	Stores the User file cabinet, the default cabinet where functions and datatypes you create are stored.
User\Examples	Stores the Examples and Templates file cabinet.
User\Imports	Stores any file cabinets that you have imported.
User\Cabinets	Stores any new file cabinets that you create using the New Cabinet command on the File menu.
User\Exports	Folder used by default to hold file cabinets that you export. The contents of this folder are considered external to Sanscript – use Windows Explorer to view, manage, move, or copy the files in this folder. You can explicitly export a cabinet into any other folder (except those created by Sanscript).
User\Apps	Folder used by default to hold applications you create (with command Make Application). The contents of this folder are considered external to Sanscript – use Windows Explorer to view, manage, move, or copy the files in this folder. You can explicitly make an application in any other folder (except those created by Sanscript).

Creating cabinets

To create a new file cabinet:

- 1 On the **File** menu, click **Cabinets**, then click **New Cabinet**.
- 2 Enter a name for the cabinet.

Cabinets are stored in the ... \user\cabinets\

Deleting cabinets

To delete a file cabinet:

- 1 Use the **Overview** window to open the file cabinet you want to delete.
- 2 On the **File** menu, click **Cabinets**, then click **Delete Cabinet**.
- 3 Confirm deletion of the cabinet.

Exporting cabinets

To export a file cabinet:

- 1 Use the **Overview** window to open the file cabinet you want to export.

- 2 On the **File** menu, click **Cabinets**, then click **Export Cabinet**.
- 3 Enter an **Author** name and **Description** for the cabinet.
This information documents the cabinet's source and its function.
- 4 If you are releasing a new version of the file cabinet that is compatible with prior versions, specify a **Minor** version number one higher than the previous cabinet you published.
- 5 If you are releasing a new version of the file cabinet that is not compatible with prior versions (you deleted functions or changed the way existing functions are used), specify a **Major** version number that is one higher than the previous cabinet you published.
- 6 Select a directory in which to store the cabinet.
You can create new folders using the "Create new folder" button in the File dialog box.

This procedure copies the cabinet to `... \user\exports\<exported cabinet name>` for other users to Import.

Figure 30. Export Cabinet Information dialog box

Note: use importing and exporting rather than direct copying of directories to share file cabinets with others.

Importing cabinets

To import a file cabinet:

- 1 On the **File** menu, click **Cabinets**, then click **Import Cabinet**.
- 2 In the **Location of Cabinet to Import** dialog box, browse directories until you locate the cabinet to import.

3 Select the `lib.fv1` file holding the definition of the file cabinet.

This procedure will import a copy of the cabinet in

```
...\user\imports\<<imported cabinet name>.
```

Note: Use importing and exporting rather than direct copying of directories to share file cabinets with others. Visit Northwoods' web site at

<http://www.nwoods.com/sanscript/cabinets.htm> to obtain cabinets for popular software applications.

7. CUSTOMIZING OPTIONS, TOOLBARS, AND WINDOWS

Customizing options

Sanscript provides many customization options.

To view the Options :

- On the **View** menu, click **Options**.

To view or change the Overview options:

- 1 On the **View** menu, click **Options**.
- 2 Click **Overview**, then select the appropriate check boxes. Table 17 lists options and their functions.

Table 17. Overview options

To	Check or uncheck
Hide or show the Overview window	Display Overview Window
Hide or show functions in the Catalog tree view	Show functions
Hide or show language functions that are not flowgrams in the Program tree view	Exclude simple function calls

To view or change the Catalog (folder) options:

- 1 On the View menu, click **Options**.
- 2 Click **Folder**, then select the appropriate check boxes. Table 18 lists options and their functions.

Table 18. Catalog options

To	Check or uncheck
Hide or show the Catalog window	Display Catalog Folder Window
Hide or show function names	View names
Hide or show the icons for functions	View icons
Show functions with large or small icons	Small icons, Large icons

To view or change the Canvas (flowgram) options:

- 1 On the **View** menu, click **Options**.
- 2 Click **Flowgram**, then select the appropriate check boxes. Table 20 lists options and their functions.

Table 19. Canvas options

To	Check or uncheck
Hide or show the names of inlets and outlets	View Inlet/Outlet Labels
Automatically show inlet and outlet labels only during linking	Auto-View Labels
See ordering link inlets and outlets in flowgrams	View Ordering Links

To view or change the window options:

- 1 On the **View** menu, click **Options**.
- 2 Click **Window**, then select the appropriate check boxes. Table 20 lists options and their functions.

Table 20. Window options

To	Check or uncheck (or click)
Make sure the current window configuration comes up the next time you start Sanscript	Remember Open Windows Now
Have Sanscript restore windows to those open the last time you exited.	Remember Open Windows on Exit

To view or change the general options:

1. On the View menu, click Options.
2. Click **General**, then select the appropriate check boxes. Table 21 lists options and their functions.

Table 21. General options

To	Check or uncheck
Emit a sound (or not) when a program completes or pauses. (You may need to enable sounds in the Windows Control Panel Sounds properties dialog box.)	Sound on Program Pause or Completion
Enable or disable a confirm-delete dialog box when you delete functions in the Catalog.	Prompt on Delete in Catalog
Enable or disable the recording of all your individual edits so they can be redone by Edit Redo or after a crash or power failure.	Journaling (for Crash Recovery and Redo)
Enable or disable the recording of all your individual edits so they can be undone by Edit Undo	Undo Logging

Customizing toolbars

You can hide the toolbar or move it.

Move the toolbar next to the menu, the status bar, or the left or right edge of the Sanscript window to dock it with (attach it to) those items. Move the toolbar to any other position in the window to have it remain a floating toolbar.

To hide or display the toolbar:

- On the View menu, check or uncheck the Toolbar command.

To move the toolbar to a new location:

- Click the toolbar next to, but not on a button, and drag it to another location.

Customizing windows

You can resize and reposition the Catalog or Overview window of Sanscript.

Drag a window next to the menu, the status bar, or the left or right edge of the Sanscript window to dock it with (attach it to) those items. A docked window becomes attached. Move the Catalog to the middle of another window or outside Sanscript to create a floating Catalog.

To move the Overview, Catalog, or Canvas to a new location:

- Click the window on its gray border, and drag it to another location.

8. PROGRAMMING FUNCTIONS

About programming functions

Over 200 programming functions are available in the Sanscript Catalog.

For detailed documentation on using these, refer to either:

- Sanscript Function Reference Guide
- Sanscript Help

9. USING SANSRIPT PROFESSIONAL EDITION

About Sanscript Professional Edition

The Sanscript Professional Edition is intended for programmers or sophisticated end users who want to integrate Sanscript with other software applications or who want to use Sanscript to integrate other software applications with each other. “Integrating with an application” means sending data to the application, receiving data from the application, or controlling the application.

The additional capabilities that Sanscript Professional Edition provides beyond those of Sanscript Standard Edition are the ability to:

- Import components from another application into a Sanscript file cabinet.
- Generate documentation for the functions in a Sanscript file cabinet.
- Set the access control properties of file cabinets.

Importing and using components from another application

In order to integrate with an external application, the application must export interface information for programming tools such as Sanscript. The application must publish a programming interface using Microsoft COM (“Component Object Model”). An application that uses COM to publish components does so by providing a type library, either as part of its executable file (.EXE) or in a separate type library file (with extension .TLB or .OLB). Contact the software vendor to determine if the application publishes a programming interface using COM.

Once you have determined that the external application supports COM, an overview of the steps to integrate with it are as follows (full details are listed in the next section below):

1. Install the application.
2. Import components from the application into Sanscript Professional Edition.
3. Optionally, use Sanscript to create your own higher-level functions based on the imported components. Your higher level functions may be simpler to use or specific to solving a certain problem.
4. Optionally, export the imported cabinet and your higher-level cabinet so they can be used by other users.

See Figure 31 for an example of using the OLE/COM wizard to import components from Peachtree Office Accounting.

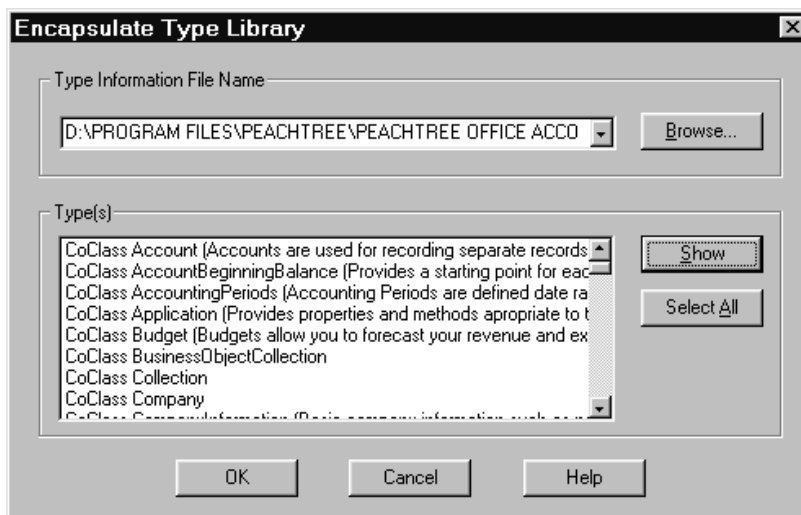


Figure 31. Example of importing components

Importing components from an application

To import components from an application:

- 1 Install the application on the same computer as Sanscript Professional Edition. (Doing so ensures that your computer's registry has complete information exported by the application.)
- 2 On the **File** menu, click **OLE/COM Wizard**.
- 3 In the `Encapsulate Type Library` dialog box, click **Browse** to locate a .EXE file or .TLB file containing type information for the application.
- 4 Click **Show** to display all the components that the application exports.

5 Click **Select All** to indicate which components to import, and then click **OK**.

Sanscript imports the components into a new file cabinet. When importing is complete, you can use the imported components to construct an application or function. You can also export the imported cabinet for use by other users.

Note: The Encapsulate Type Library dialog box may show you many more exported objects than Sanscript can import. Be sure to import only files of type .EXE, .OLB, or .TLB. You may need to contact the vendor of the application to obtain documentation or an SDK (software development kit) that explains that vendor's programming interface. Visit Northwoods' web site at <http://www.nwoods.com/sanscript/cabinets.htm> for additional information on what applications can be imported and to obtain already encapsulated cabinets.

Generating documentation for functions

Sanscript can generate a Rich Text File describing all the functions in any file cabinet. The generated Rich Text File can be used as the basis for generating manuals or help files. The generated file lists each function, the function's inlets and outlets, and any descriptive text that has been entered by the user (see page 62, Changing properties of a folder, or a function, or a file cabinet). The *Sanscript Function Reference Guide*, which is generated from the General cabinet, illustrates the kind of documentation that can be generated.

To generate the Rich Text File describing a file cabinet:

- 1 Open the file cabinet.
- 2 On the **Help** menu, click **Generate**.
- 3 In the *Generate Documentation* dialog box, specify a folder and filename to hold the Rich Text File.
- 4 Click **OK**.
- 5 Import the resulting Rich Text File into a word processor or help file processor.

Note: The Rich Text File is designed primarily for use in creating a help file. You may need to edit it in order to create a manual.

Setting access controls and other properties for a cabinet

Use access controls to define the kind of changes that can be made to a file cabinet. To change access controls, the file cabinet must not have a password, or you must know the password.

To change access controls and other properties for a cabinet:

- 1 Right-click the file cabinet in the Catalog.
- 2 In the Properties for... dialog box, click **More**.
- 3 If the **Set Password** button appears, the file cabinet does not require a password. If the **Enter Password** button appears, click **Enter Password** and enter the password for the cabinet. If the password is correct, the properties fields become enabled.
- 4 Check **Allow Modify** (located in the Options group).
- 5 Modify whatever cabinet properties you wish. Examples are: the **Description**, the **Allow Delete**, **Allow Modify**, or **Allow Copy From** access controls, and the **Help File** name and context.
- 6 Uncheck **Allow Modify**.
- 7 Click **OK**.

Note: If you set a password for a file cabinet, be sure to record it, otherwise you might no longer be able to modify the cabinet. Rather than using a password, consider preventing unintentional modification to a file cabinet by keeping **Allow Modify** unchecked. Doing so will eliminate the need to remember a password.

GLOSSARY

application

Program run outside of the Sanscript tool.

Canvas

Window that provides an area to view and edit the flowgram for a function.

Catalog

The folder window on the lower part of the Sanscript main window that displays the contents of the file cabinet or folder open in the Overview window. The Catalog contains the functions that can be dragged onto the canvas to create a flowgram.

comment

Language function that inserts a remark into a flowgram.

constant

Language function that generates a constant value. Constant values include integer, decimal, text, Boolean, and lists of these.

datatype

Name for the kind of information that can appear on an outlet or inlet. Examples of datatypes are integer, decimal, text, and Boolean.

flowgram

Diagram of functions linked to other functions. Its purpose is to specify the calculation performed by a function.

form

Language function that causes a window (dialog box) to appear

function

Basic building block for programs. Inputs data, performs a calculation, and produces outputs. Sanscript functions are stored in the Catalog and are represented by icons.

global

Record whose values are available to any function in a program.

icon

Picture used to represent a function. Icons can be created using popular drawing tools, such as Microsoft Powerpoint and Corel Draw, that can save a picture as a Windows metafile.

inlet

Arrow on the left side of a function's icon that receives data from a link with another function.

inlet connector

Language function used within the flowgram of a function to get data from an inlet of the function.

language functions

Special functions needed to complete the programming language itself. All the other functions, the programming functions, are the building blocks you use to construct your own programs.

link

Line drawn from an outlet of one function to the inlet of another and used to carry data from one to the other.

outlet

Arrow on the right side of a function's icon that sends data over links to other functions.

outlet connector

Language function used within the flowgram of a function to send data to the outlet of the function.

Overview

The window that displays the program tree and all the file cabinets available in Sanscript. The Overview window works like the left half of Windows Explorer.

Similarly, the Overview is used to explore and locate all the functions provided in Sanscript.

package

Language function used within a flowgram to hide a portion of the flowgram. Packages allow a flowgram to be very large and complex without also taking up a lot of screen space.

Pick One

Language function used to make a decision based on a prior calculation. A Pick One uses data arriving on its selector inlet to choose which flowgram stored within it should run. A Pick One can have any number of flowgrams within it, one for each value that could arrive on its selector inlet.

program

Specific kind of function that does not need inputs from another function. Because it does not need inputs from another function, a program can run all by itself. In traditional programming languages a program is sometimes called the “main program.” A program can run within the Sanscript tool, or it can be made into an application, which runs outside of the tool.

record

Language construct used to create a new datatype composed of other datatypes. Similar to records in a database, a record typically has several types of data stored within it.

Repeat

Language function that repeatedly runs the flowgram contained within the Repeat. There are several kinds of repeat depending on what is connected to the Repeat inlet. If an integer number is connected to the inlet, the number of repetitions is the number. If nothing is connected, the repetitions continue until the flowgram inside the repeat sends TRUE to the Stop function. If a list is connected, there is one repetition for each item in the list.

stop

Language function placed inside a Repeat. Stops the repeat if a TRUE is sent to its inlet.

template

Partially completed function that you can use as a starting point for creating a new function. A template automatically expands to a flowgram when dropped on a canvas. Comments within the template indicate the portions of the template that need to be filled in.

INDEX

A

alias

definition, 59

application

definition, 17

making an, 19

running an, 20

C

cabinets

generating automatic documentation, 75

setting access controls for, 76

Canvas, 5

Catalog, 4

browsing the, 17

customizing the, 70

floating and docking the, 70

General folder of, 5

COM

encapsulating, 73

comments

using, 33

connectors

using, 32

constants

using, 33

D

datatype

definition, 27

datatypes

automatic conversion, 27

gravity, 27

integer, decimal, text, Boolean, 27

lists, 46

records, 46

E

editing, 29

deleting functions, 30

editing function definitions, 30

linking functions, 30

moving functions, 30

resizing functions, 30

undoing and redoing edits, 29

error handlers

using, 43

errors

using, 42

examples

Do You Want to Say Hello, 9

Fibonacci, 11

Hello World, 8

Hello World, creating, 25

Pick One, 9

external applications

importing components from, 73

integrating, 73

F

file cabinet

editing properties of, 62

file cabinets

creating new, 64

- deleting, 64
- exporting, 64
- importing, 65
- using, 63
- where they are stored, 63

flowgrams

- viewing large, 50

folders

- creating, 59
- deleting, 62
- editing properties of, 62
- moving, copying, 60

forms

- using, 34

function, 5

- definition, 17

functions, 3

- controlling the order in which they run, 68
- creating, editing, 25, 29
- deleting, 30, 62
- editing definitions of, 30
- editing properties of, 62
- exporting, 64
- finding in the Catalog, 17
- how they work, 8
- importing, 65
- Language Functions, 5
- linking, 30
- moving, copying, 60
- printing, 53
- Programming Functions, 5
- saving, 31
- searching for, 17
- seeing more details of, 67
- sharing, 63
- simplifying, 44, 53
- using Language Functions, 32
- viewing and editing properties of, 50

I

inlets

- editing, 32
- using, 31

L

labels

- hiding, 68

Language Functions

- transforming, 44
- using, 32

O

options

- customizing, 67

ordering link

- definition, 44

ordering links

- using, 44

outlets

- editing, 32
- using, 31

Overview, 4

P

packages

- using, 36

Pick Ones

- using, 37

program

- definition, 17

program wizard

- using the, to create new functions, 26

programs

- browsing, 55
- finding in the Catalog, 17
- ready-to-run, 20
- running, stopping, 19
- running, stopping, pausing, 55
- searching for, 17
- testing, 56

R

Repeats

- inlet/outlet pair, 32
- using, 40

T

template

definition, 13

templates

introduction to, 13

using, to create new functions, 26

Toolbar

buttons on the, 6

customizing the, 69

floating and docking the, 69

type libraries

importing, 74

Z

zooming, 50