JET Inspector for Windows
Version 2.0

6047 Tampa Ave., Suite 209, Tarzana, CA 91356
818-345-2200 • Fax 818-345-8905 • BBS 818-343-8433
Internet: blagoon@netcom.com • CompuServe: 70152,1601

Contents

Disclaimer

Introduction

Thank you for purchasing **JET Inspector for Windows!**

Database access through interfaces such as Oracle OCI, Sybase SQL, and Microsoft ODBC are very powerful and require that the developer have an intricate understanding of the database implementation library. When Microsoft Visual Basic and Microsoft Access were released, this process was greatly simplified through new data-aware custom controls and a very simplified database access language. The Microsoft JET Database Engine, which is at the heart of this simplified database access, interfaces to local database files and also performs ODBC operations against any local or remote database that supports ODBC, thus making database access much easier to implement.

This product acts as a performance tuning and analysis tool for data access operations as they occur in the Microsoft JET Database Engine. In addition, if you also own the **ODBC Inspector,** it is possible through our Object Link Technology (OLT) also to show the ODBC operations which occur as a result of JET operations and a breakdown of the amount of execution time spent in JET and ODBC.

The JET Operations topic in this help file contains descriptions of the most commonly-used JET operations, and **JET Inspector's** output is self-explanatory enough for the average user. JET power users may require more complete, detailed information, however; our JET Operation Reference, available separately, has detailed information, including full parameter listings, for all JET operations traced by **JET Inspector**.

The version of **JET Inspector for Windows** you have purchased is to be used under Windows 3.1 and with any application written in Microsoft Visual Basic 3.0 (with the Access 2.0 compatibility layer) and/or Microsoft Access 2.0. The Inspector allows you to trace most JET operations simultaneously.

To use this product, we recommend that your computer have a 386 or higher processor and at least 8MB of RAM.

Menu commands

JET Inspector for Windows has five menus: Inspector, Marker, Find, Window, and Help. Accelerator keystrokes are provided to access many of the menu commands; these are listed in the table below:

Keystroke	Menu Command
Ctrl+I	Inspector On/Off
Ctrl+O	Options dialog
Ctrl+F	Find text
Ctrl+R	Find First
Ctrl+P	Find Previous
Ctrl+N	Find Next
Ctrl+L	Find Last
Ctrl+W	Clear Window

Inspector menu commands

Inspector On/Off
Toggles trace mode. When **JET Inspector** is tracing, the caption bar of the application will be set to "<< JET Inspector >>". When **JET Inspector** is not tracing, the caption bar of the application will be set to "JET Inspector".

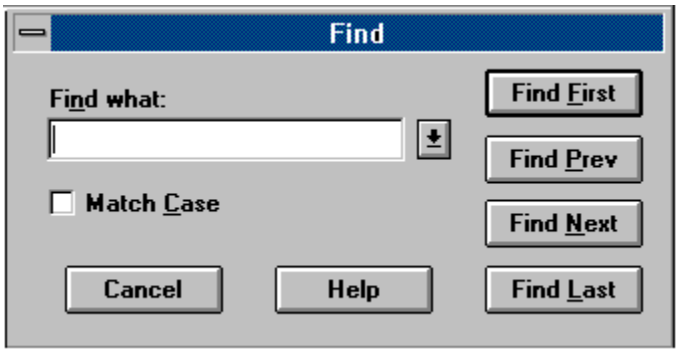
Options
Brings up the **JET Inspector** Options dialog box. It is within this dialog box that tracing options are setup.

Exit
Exits **JET Inspector.**

Marker! menu command
Brings up the Marker dialog box. The Marker dialog box is designed to allow user entry of a string of up to 255 characters which will be inserted into the trace log.

Find menu commands

Find Text...
Brings up the Find dialog box. This is a "floating" dialog box used to search for text within the Inspector's output window.



In the **Find what** box, enter the text you wish to search for. Check the Match Case box by clicking on it if you want to make the search case-sensitive. Then, click on the Find First button to find the first

occurrence of the text; you can also search for the next, previous or last occurrence by clicking on the appropriate button.
The Find dialog box remembers the last five text strings you have searched for in the current **JET Inspector** session; you can select one of these by clicking on the down-arrow next to the Find what box. You can switch back to the Inspector window by clicking on it, and the Find window will remain open.

Find First
Find Previous
Find Next
Find Last
These menu commands find the appropriate occurrence of the last text string specified in the Find dialog box.

Window menu commands

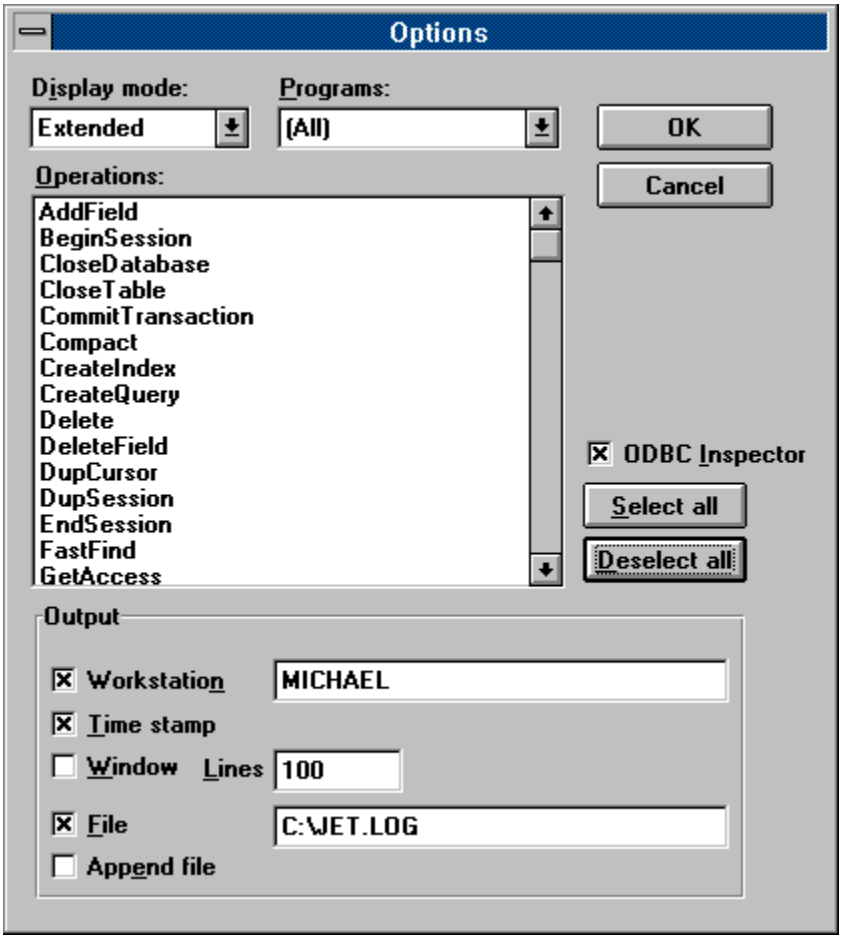
Clear Window
Clears the **JET Inspector** window. Any output in the **JET Inspector** window will be cleared. This has no effect on the log file, if one is being used.

Help menu commands

Contents
Brings up the contents portion of the help file.

About
Displays the application "about" box.

Options dialog box



Display mode combo box
The Display mode combo box contains a list of the different trace modes available. The **JET Inspector** currently supports “Basic Pseudo-Code” and “Extended” mode.

Programs combo box
The Programs combo box contains a list of all programs currently running under Windows. To trace a specific applications’ JET accesses, first make sure that the application is currently running, then select it from this combo box. **JET Inspector** will only display JET operations made by this application. Select the "(All)" option to display JET operations made by ALL applications.

Operations list box
The Operations list box is a multi-selection list box. Individual operations are selected to indicate that they should be displayed by the Inspector when they are occur. This allows you to filter out the operations you're not interested in tracing.

ODBC Inspector check box
When selected, this check box will establish an object link between the **JET Inspector** and the **ODBC Inspector**. If the **ODBC Inspector** is not installed on your system, then this check box will be grayed and disabled.

Select all push button
Selects ALL operations in the Operations list box.

Deselect all push button
Deselects ALL operations in the Operations list box.

Window check box
This check box should be selected to indicate that you want **JET Inspector** to display what it traces in the application area of the **JET Inspector's** application window.

Workstation check box
This check box should be selected to indicate that you want **JET Inspector** to include the workstation name on each line of the trace output. The workstation name is set by entering a 16 character string in the entry field adjacent to this check box.

Time stamp check box
This check box should be selected to indicate that you want **JET Inspector** to include the time stamp on each line of the trace output.

File check box
This check box should be selected to indicate that you want **JET Inspector** to create a log file which will contain all traced data. As **JET Inspector** traces, it will append its data in ASCII format to a log file whose path name is specified in the adjacent entry field.

Append file check box
This check box should be selected to indicate that you want **JET Inspector** to open the trace log file in append mode. If this check box is not selected and the file check box is selected, then each time the **JET Inspector** is started, the trace log file will be truncated to zero length.

OK push button
Accepts the changes made.

Cancel push button
Cancels the changes made.

Network multi-workstation tracing

The **JET Inspector's** log file tracing mechanism uses network file sharing. This means that by enabling the File and Append file features in the Options dialog box, all traced output will be appended to the trace log file using network file sharing. Thus, when more than one user has this feature enabled and the same network trace log file is used, then JET operations for these users will be traced into the same trace log file.

Naturally, if several users are sharing a network trace log file, the workstation name feature becomes very useful in identifying which JET operations occurred on a particular workstation. In addition, the time stamp is also very useful to help understand when an operation occurred.

By only tracing JET operations that are necessary, better performance will be available for the application that is being traced. In addition, with several users tracing to the same network trace log file, choosing JET operations becomes all the more important since each workstation running the Inspector will use file locking to prevent access to the file while it is writing its trace output.

See Also

Viewing Multi-Workstation Log Files

JET Operations

The following provides brief descriptions of the more common JET operations.

Be aware that the operation names shown here do not necessarily exist in the JET engine itself. Because the JET programming interface is undocumented, we have had to assign our own names to these operations to describe their actions.

In many cases, JET operation names correspond roughly or exactly to Visual Basic statement names. In other cases they do not, but we have tried to make the names describe as closely as possible the function performed.

For more complete, detailed information on all JET operations monitored by **JET Inspector**, please refer to our *JET Operation Reference*, available separately.

BeginTrans
Marks the beginning of a JET transaction in the specified JET Connection. All subsequent updates will be buffered by JET until the next CommitTrans operation (which writes all the updates as a single unit) or Rollback operation (which discards all the updates). The transaction applies to all databases, tables and dynasets opened under the specified JET Connection.

CommitTrans
Saves all record updates made since the last BeginTrans; ends the current transaction.

CreateDatabase
Creates a database and opens it.

DeleteRecord
Deletes the current record from a given table (or dynaset).

DoBackgroundProcessing
Microsoft Access and Visual Basic continually perform this JET operation while not performing any other operations in JET. It is usually best to shut this operation off in the **JET Inspector** to avoid degrading system performance and wasting space in the log file.

DoSearch
This JET operation is performed in response to Visual Basic FindFirst, FindNext, FindPrevious and FindLast methods. It performs a search on the given table using a given SQL-like query string.

EditRecord
Locks the current record in the specified table/dynaset for updating.

GetAccessRights
Retrieves access rights information for the specified user on the specified database, table or other JET object. See also SetAccessRights.

GetImmediateIndex
Retrieves the currently-selected table index for the specified table. The returned table index value is a zero-based index into the list of indexes defined for the table (i.e. 0 represents the first index, 1 is the second, etc.)

GetFieldData
Retrieves the data for the specified field (column) in the current row of the specified table. The returned data may be of any valid JET type, including byte, integer, long integer, floating-point, string or binary data. The JET Inspector attempts to determine the correct type of the data and display it in the proper format; in cases where this is not possible, all valid interpretations of the data are listed.

Move
Repositions the specified table (or dynaset or snapshot) ahead or behind a specified number of records.

MoveFirst

Repositions the specified table, dynaset or snapshot to the first record.

MoveLast
Repositions the specified table, dynaset or snapshot to its last record.

OpenDatabase
Opens the specified existing database using the given (optional) connect string.

OpenTable
Opens the specified existing table in the specified database.

Rollback
“Undoes” all updates in the given JET Connection since the last BeginTrans call. Cancels the current transaction.

Seek
Performs a search using a “key” defined with a previous SetSearchKey call. The search can be performed using various comparisons, such as “equal to” the key, “less than” the key, etc.

SetCurrentIndex
Makes the specified index the current index on the specified table, dynaset or snapshot.

SetFieldData
Sets the specified field (column) in the current record to the given data. The data type and layout depend on the field being set. The **JET Inspector** attempts to determine the correct type of the data; in cases where this is not possible, it lists all valid interpretations of the data.

SetSearchKey
Creates a search “key” which defines the data that subsequent Seek calls will try to match. The data passed to SetSearchKey is of a type dictated by the index used in the search; i.e. it can be a string, an integer or other data type. The Inspector attempts to determine the correct type of the data passed to SetSearchKey; in cases where this is not possible, it lists all valid interpretations of the data.

StartJETConnection
Opens a "JET Connection," associated with the current application; logs the user into JET using the given username and password. Note that the **JET Inspector** normally does not show password strings in its output, in the interest of preserving security. If you need to see the password information, please contact us.

StartJETEngine
Initializes JET processing for the current application; returns an “Application ID” which identifies the application to JET in StartJETConnection and other operations. StartJETEngine is always called near the beginning of an application’s processing with JET, and TerminateJETEngine is called near the end of JET processing.

TerminateJETConnection
Closes the specified JET Connection.

TerminateJETEngine
Terminates the specified application’s communications with JET. This is the counterpart to the StartJETEngine operation.

Update
Updates the current record with any changes that have been made with SetFieldData operation calls. Note that if a transaction has been started, the changes will not actually be saved until a CommitTrans operation occurs.

ValidateAccess
Returns access rights information for the specified object (database, table, etc.)

JET Inspector's traced data

It is possible to have **JET Inspector** trace data into **JET Inspector's** application window and/or a log file in ASCII format. The application window can hold approximately 3000 lines of Basic Mode output or about 8000 lines of Extended Mode output. The capacity of the log file is limited only by available disk space.

When in Basic Pseudo-Code Mode ("Basic Mode"), **JET Inspector** outputs a line as in the following example for each Basic Mode JET operation that occurs:

```
11:29:43 PAUL 000:099 MSACCESS: set db1 = OpenDatabase("C:\TEST.MDB", True, False, "[Unknown]")
```

The output of each operation is placed on a single line by itself. Each output line contains the following information:

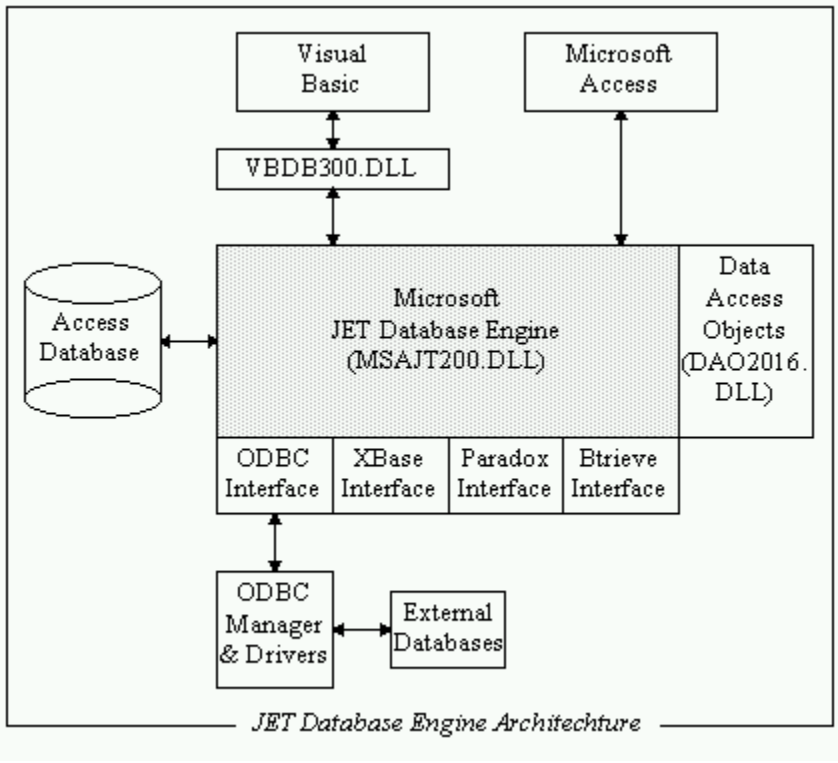
- Timestamp when the operation's execution began (if timestamps are enabled)
- Workstation name (if workstation names are enabled)
- Execution time of the call, in *seconds:*milliseconds
- Application name
- Pseudo-Basic code representing the JET operation that occurred.

In Extended Mode, the above information is given, but more detailed parameter information is included; also, a number of less-common JET operations (including those that do not directly translate to Visual Basic statements) only appear in Extended Mode.

Options exist within **JET Inspector** to allow the time stamp and/or workstation name to be included in each trace output line. The time stamp is very useful because it allows you to see what time a particular JET operation took place. The workstation name is also very useful when the trace log file is located on a network drive and is shared by many different users.

JET Overview

The Microsoft JET Engine is a set of related DLLs that are used by Microsoft Access and Visual Basic to perform database access. These applications communicate with JET as shown in the diagram below:

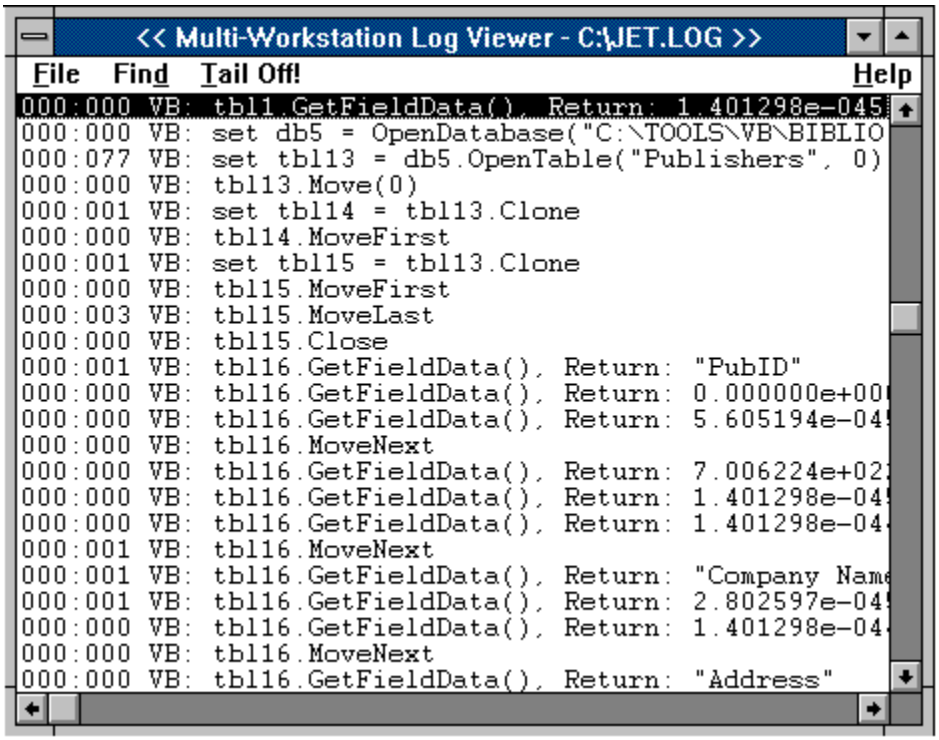


As can be seen from the diagram, both Visual Basic and Access use the JET engine for accessing databases in a variety of formats. Visual Basic communicates with JET through an interface layer in VBDB300..DLL. Access communicates directly with JET for performing many database functions, but also communicates via the Data Access Objects interface (DAO2016.DLL) when running Access Basic modules. The DAO layer provides an object-based data access language. Visual Basic programs (whether run from the IDE or compiled) do not seem to use this layer.

As the diagram shows, all Access and Visual Basic data access requests are eventually received by JET and are either processed by it directly or are dispatched to other modules. Therefore, by monitoring JET, the **JET Inspector** gives you a window into any data access done by Visual Basic or Access, whether against an Access database (.MDB format), other local file (XBase file, Paradox file, etc.) or a remote database accessed through ODBC.

Viewing Multi-Workstation Log Files

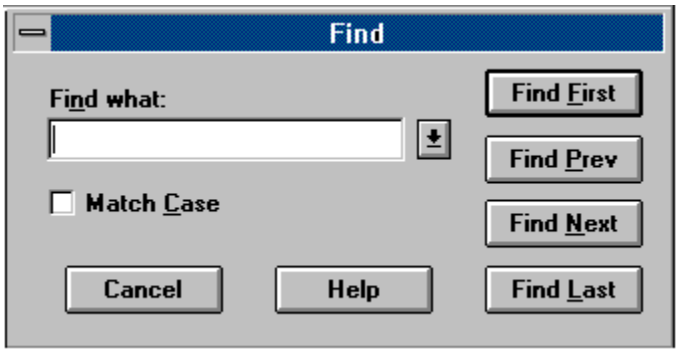
JET Inspector includes a utility, ***Multi-Workstation Log Viewer***, which allows a network administrator to view a shared log file. This utility can be started by selecting the Launch Log File Viewer item from the Inspector menu, or by double-clicking the Log Viewer's icon in the Program Manager.



The Viewer normally displays the end of the log file, and continually scrolls new data into view as it is written to the log file by any workstations.
Choosing the Tail Off! command from the menu causes the Viewer not to automatically show new data; this lets the user browse through the file without being interrupted by new data.

The Viewer normally brings up the log file as configured in the JET Inspector's Options dialog box. However, the user may open a different file by choosing the Open... command from the File menu.

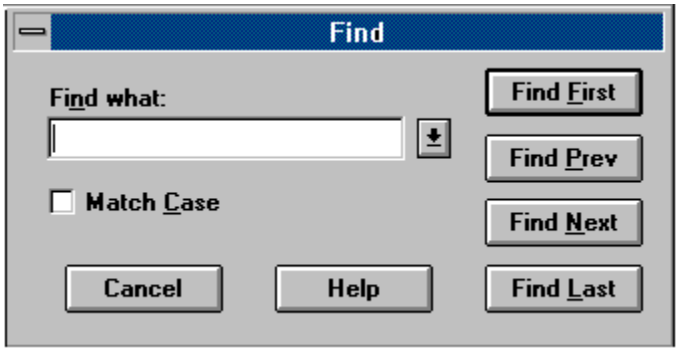
The Find menu commands bring up a dialog box that lets the user searches the log file for a given text string. As in the Inspector, the user can perform Find First, Find Previous, Find Next and Find Last operations to locate a specific occurrence of the search text.



The Find window remains active until it is closed, to allow repeating of searches.

Find Dialog Box

Selecting the **Find text...** menu command Brings up the Find dialog box. This is a "floating" dialog box used to search for text within the Inspector's output window.



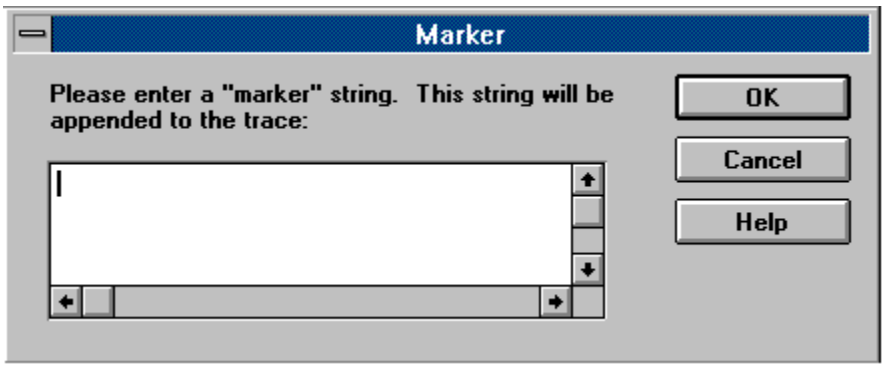
In the **Find what** box, enter the text you wish to search for. Check the Match Case box by clicking on it if you want to make the search case-sensitive. Then, click on the Find First button to find the first occurrence of the text; you can also search for the next, previous or last occurrence by clicking on the appropriate button.

The Find dialog box remembers the last five text strings you have searched for in the current SQL Inspector session; you can select one of these by clicking on the down-arrow next to the **Find what** box.

You can switch back to the Inspector window by clicking on it, and the Find window will remain open.

The **Find First, Find Previous, Find Next** and **Find Last** menu commands find the appropriate occurrence of the last text string specified in the Find dialog box.

Marker Dialog Box



This dialog box lets you enter a string of up to 255 characters to be written into the Inspector's window and log file output; this feature is useful for inserting annotations into the traced output.