JAM 7

Read Me First

August 1995

Send suggestions and comments regarding this document to:
Technical Publications Manager
JYACC, Inc.
116 John Street
New York, NY 10038
(212) 267–7722

Or via email to:
`release@jyacc.com`

Table of Contents

About this Guide

JAM's *Read Me First* consists of four sections:

Section One: What's New in JAM
New features and changes in JAM 7.

Section Two: Installation Guide
Installation notes for installing JAM 7 on your system, whether for the first time or as an upgrade. Read this section thoroughly before beginning installation.

Section Three: License Manager Installation
Licensing options and instructions for installing the License Manager (used on many UNIX and VMS platforms).

Section Four: Appendixes
Additional installation notes for specific platforms, and database driver information, as well as a password request form for licensing JAM.

There are other documents supplied with JAM which you might find of interest:

- *Getting Started* contains useful information for orienting you to JAM, a description of the JAM environment and features, and a quick start to using JAM to build applications.

- *Tutorial* consists of 11 lessons that introduce you to JAM and guide you through the creation of a mini-JAM database application.

- Database-specific release notes detailing the setup of your JAM application as a database client. The release notes, `readme.*`, are supplied online in the `notes` subdirectory.

- The list of bugs fixed in JAM 7. This document, `fix700.txt` is provided online in the `notes` subdirectory.

- For those platforms (VMS and few UNIX platforms) which cannot support JAM’s online documentation, a complete set of printed manuals are provided in lieu of the online documentation set.

SECTION ONE

What's New in JAM

1

What’s New in JAM 7

Updated Tutorial

The tutorial has been enhanced to demonstrate new JAM 7 features. It shows how to use the screen wizard to create screens. It also shows how to:

- Create menus and toolbars
- Create and use graph and grid widgets
- Use JPL to change widget properties at runtime

Refer to Chapter 3 in *Getting Started*.

Updated Sample Application

The VideoBiz sample application has been enhanced to include new JAM 7 features, including graph and grid widgets, and new JPL syntax to access runtime properties.

Refer to Chapter 2 in the *Application Development Guide*. for a description of features and instructions on how to use the sample application.

Development Environment

Screen Wizard

The screen wizard is a tool that can help you design a transaction manager screen—that is, one that works with the JAM transaction manager. It collects the basic design information by prompting you with a series of simple dialogs and quickly guiding you through the design process. While it is simple to use, you can also use the screen wizard to make a first cut of a complex screen.

The screen wizard can make three different types of screens: master, master-detail, and master-detail-subdetail.

Refer to the *Editors Guide*, page 67 for details on using the screen wizard.

Undo

Most editor actions, such as widget creation, deletion, positioning, and property changes, can be undone and redone via the menu bar or toolbar. You can traverse the undo stack and thereby reverse and restore multiple edits. The size of the undo stack is configurable.

Refer to the *Editors Guide*, page 51.

Editor Interface

The screen and menu bar editors have been enhanced with a number of improvements. These include:

- Toolbars that contain frequently used commands.
- Selection dialogs for a number of properties, including file name selection and font properties.
- Widget list now allows extended selection.
- Options⇒Set Inherit Warnings lets you suppress inheritance warnings.
- Ctrl+click on a toolbox icon to enter multiple create mode for the selected widget type.
- Double clicking is now supported on all platforms, including character-mode—for example, for file selection.

- Edit⇒Grid Align aligns selected widgets to the grid.
- GUI editors now snap a widget to the grid at the widget's anchor points.
- Context-sensitive help is available from all editor properties and menu items.
- Options⇒Enable Debugger automatically enables the debugger when you enter test mode.
- File dialogs in character mode contain a list of most recently used directories.
- Before entering test mode, the editor asks whether to save all unsaved screens except the current one.
- Context-sensitive help is available from all editor properties and dialogs.

JISQL

JISQL is a graphical tool for creating JDB databases and for writing and executing interactive SQL scripts. Refer to page 33 in the *Database Guide*.

Menu Bar Editor

The menu bar editor interface is significantly reworked; new properties support tool bar creation. Refer to page 213 in the *Editors Guide*.

Debugger

The JAM debugger has been enhanced with these improvements:

- Event filtering — You now have finer control over events types when tracing through your application. Refer to page 515 in the *Application Development Guide*.
- Complex breakpoint setting — The Edit Breakpoints screen offers better control of breakpoint setting. Breaks can be set on an event, sub-event, or source code location, or change in an expression. Refer to page 512 in the *Application Development Guide*.
- Expert mode — Expert mode offers access to advanced features such as the Application Data window and Call Installed Function. Refer to page 495 in the *Application Development Guide*.

JPL

- New syntax options allow easy access to all application components and their properties. Refer to page 24 in the *Language Reference*.

- JPL’s maximum line length is now 252 characters.
- The `msg` command has an `%Mt` option that forces temporary display of message to the status line. JAM automatically dismisses the message after the specified timeout elapses. Refer to page 58 in the *Language Reference*.

Grid Widgets

You can create and modify grid widgets in the editor. A grid frame widget is a two-dimensional array, displayed as columns and rows, that allows users to scroll data both vertically and horizontally. It can be particularly useful for displaying data in a spreadsheet fashion and for displaying database query results in a tabular arrangement. In addition, you can carry out database operations, such as inserting and deleting records.

With a grid frame widget, you can display an unlimited number of rows associated with a specified set of columns.

Refer to page 181 in the *Editors Guide*.

Graph Widgets

You can present data on your application screens in graph or chart format by using graph widgets. Graph data can be generated at runtime or obtained from static sources and can be displayed in a variety of formats: pie chart; bar/line graph; XY plot; and high/low chart.

Refer to page 121 in the *Editors Guide*.

DLL Support

JAM now provides library functions that let you load and install dynamic link libraries (DLLs) under Windows: `sm_slib_load` and `sm_slib_install`. You can now call functions located in a DLL library directly from JPL without editing `funclist.c` and recompiling.

You can bundle all bitmaps, cursors, and icons into a DLL. JAM searches for resources in a DLL before looking on disk.

Toolbars

You can create toolbars through the menu bar editor that display alongside screen menus. A screen’s menu bar and toolbar share the same menu definition; each toolbar item corresponds to a menu bar item and vice versa. You can toggle display of menu items so that they appear on the toolbar or the menu bar, on both, or on neither.

For information about creating toolbars, refer to page 219 in the *Editors Guide*,; for information about runtime options, refer to page 92 in the *Application Development Guide*.

Runtime Properties Access

All properties of the application and its screens, and widgets are accessible through JPL and library functions. For information about accessing properties in JPL, refer to page 28 in the *Language Reference*; for access through library functions, refer to `sm_prop_get` and `sm_prop_set`.

Database Interface

Database Importer

If the database engine supports table views and synonyms, those database objects can be imported to a JAM repository.

Database Drivers

- All database drivers support a new `DBMS` command, `DBMS COLUMN_NAMES`. If specified, a SQL `SELECT` statement returns the column names to the JAM variables named in the command.
- The database driver for SYBASE now supports CT-Library.

JDB

JDB has a new graphical interactive SQL editor, JISQL.

New Library Functions

- `dm_gen_change_select_suffix` — appends text to a SQL `SELECT` statement.
- `dm_is_connection` checks whether the specified connection is open.
- `dm_is_engine` checks whether the specified engine is initialized.
- `sm_tm_continuation_validity` checks which `CONTINUE` commands are available in transaction manager.

SQL Generation

- Optimistic locking can be specified with one of the following methods:
  - Set the Version Column to Yes and the transaction manager supplies the column values automatically.
  - Set In Update Where and In Delete Where to Yes to add the value in a widget to the `WHERE` clause in `UPDATE` and `DELETE` statements.

  Refer to page 370 in the *Application Development Guide*.
- The Use In Where property now supports the following operators:

  ```
  in
  like%
  %like%
  not in
  not like%
  not %like%
  ```

  Refer to page 273 in the *Application Development Guide*.

Transaction Manager Commands

- The transaction manager has two new commands that let you change the transaction mode:

  `COPY_FOR_UPDATE` *Application Development Guide*, page 430

  `COPY_FOR_VIEW` *Application Development Guide*, page 432
- The transaction manager supports non-sequential scrolling with these commands:

`TM_CONTINUE_BOTTOM` *Application Development Guide*, page 412

`TM_CONTINUE_DOWN` *Application Development Guide*, page 416

`TM_CONTINUE_TOP` *Application Development Guide*, page 420

`TM_CONTINUE_UP` *Application Development Guide*, page 424

Transaction Manager Processing

- `sm_tm_inquire` has four new parameters:

  ```
  TM_QUERY_ACTION
  TM_PARENTING_OCC
  TM_SV_SEL_REQUEST
  TM_CONTINUATION
  ```

- `sm_tm_set` has two new parameters:

  ```
  TM_QUERY_ACTION
  TM_SV_SEL_REQUEST
  ```

- `sm_tm_pinquire` has two new parameters:

  ```
  TM_COMMAND_PARM
  TM_PREVIOUS_HOOK
  ```

Transaction Models

The standard transaction models for all engines support the new commands.

GUI Runtime Behavior

The following changes affect GUI runtime behavior. Some of these may be implemented by changing the JAM behavior.

- Resize properties specify automatically to shrink or expand certain widgets such as list boxes in response to screen size changes at runtime. Refer to page 99 in the *Editors Guide*.
- The Size To Contents property automatically resizes certain widget types such as push buttons and labels to their contents.

- Better word wrap behavior on GUI platforms. Multiline text widgets whose Wordwrap property is set to Yes no longer are shifting and do not display horizontal scrollbars.
- Extended selection in list boxes.
- Default button now moves to button with focus.
- Space key now activates push buttons and toggle buttons.
- Windows dialog screens are automatically centered if location parameters are omitted.
- `sm_system` processes are hidden in Windows unless bracketed by calls to `sm_leave` and `sm_return`.

GUI Enhancements

- Wallpaper Pixmap property. Motif supports a pixmap image as the screen background; Windows supports it as a screen and MDI frame background. The specified image can be centered or tiled.
- The configuration map file now contains a section that tells JAM which fonts and font sizes to display in the drop-down menus in the screen editor; it also defines default display fonts, and maps system-specific font names to JAM font aliases. Refer to page 147 in the *Configuration Guide*.
- XBM and XPM image file formats are supported for Motif; GIF and JPEG formats are supported for Motif and Windows.
- In Windows, the Icon property now accepts the name of an icon (`.ico`) file.

File I/O API

A new set of library functions let you write directly from JAM screens to disk and vice versa:

- `sm_fio_a2f` writes the contents of an array to a file.
- `sm_fio_close` closes an open file stream.
- `sm_fio_editor` invokes an external text editor for an array.

- `sm_fio_error` gets the error returned by the last call to a file I/O function.
- `sm_fio_error_set` sets the file I/O error.
- `sm_fio_f2a` writes a file's contents to an array.
- `sm_fio_getc` Reads the next byte from the specified file stream.
- `sm_fio_gets` reads a line from a file.
- `sm_fio_handle` gets a handle to an open file.
- `sm_fio_open` opens the specified file and returns a handle to the JPL caller.
- `sm_fio_putc` writes a single byte to an open file.

Copying and Deleting Widgets

Two new functions let you copy and delete widgets at runtime, `sm_obj_copy` and `sm_obj_delete`.

Mouse Event Processing

- Double Click is a property of dynamic labels, single-line and multiline text widgets, list boxes, and combo boxes. Refer to page 305in the *Editors Guide*.
- You can also query the application for these mouse runtime properties: `mouse_field_name`, `mouse_field_occ`, `mouse_form_name`, and `mouse_form`. Refer to page 555 in the *Application Development Guide*.
- `sm_ms_inquire` gets information about the mouse's current state: the position of the last mouse click on the physical or JAM screen, whether other keys are pressed in combination with it, and which mouse buttons have been pressed and how recently.
- `sm_mus_time` gets the system time of the last mouse click.

List Box Selection

A list box widget's Listbox Type property is by default set to Select Any. Users can choose zero to any number of items. Consequently, list boxes need to be made into a selection group only if you want to limit users to one choice. Refer to page 200 in the *Editors Guide*.

Password Character

The Password Char property lets you specify any character to hide user input to a field. This property is available only if the Password property is set to Yes. Refer to page 234 in the *Editors Guide*.

General Library Functions

New Functions

- `sm_setsibling` forces sibling status onto the next screen opened as a window.
- `sm_ww_length` gets the number of characters in a word wrap field.
- `sm_ww_write` puts text into a wordwrap field.
- `sm_ww_read` gets word-wrapped text from a multiline text widget.
- In Presentation Manager only, `sm_pm_res_add_map` installs tables that map string resource identifiers to integer identifiers.

Changes to Existing Functions

- Message functions like `sm_femsg` now accept a `%Mt` option in the message text that forces temporary display of message to the status line. JAM automatically dismisses the message after the specified timeout elapses.
- `sm_message_box` now accepts these additional arguments to specify button combinations:

  ```
  SM_MB_YESALLNOCANCEL
  SM_MB_OKALL
  SM_MB_YESALLNOALLCANCEL
  ```

- `sm_mnitem_get` and `sm_mnitem_set` now access these menu item properties:

  `MNI_DISPLAY_ON` specifies whether to display the menu item on the menu and/or the tool bar.

MNI_ORDER: The order in which a menu item appears on the toolbar.

MNI_ACT_PIXMAP: The name of an image file whose contents are shown for an active toolbar item—that is, accessible but not pressed.

MNI_ARM_PIXMAP: The name of an image file whose contents are shown for an armed toolbar item

MNI_EXT_HELP_TAG: A help context identifier that specifies the help to invoke from an external help program.

MNI_INACT_PIXMAP: The name of an image file whose contents are shown for an inactive or unavailable (grayed) item.

MNI_NAME: The menu item's name.

MNI_TOOL_TIP: The balloon help to display when the cursor remains over the toolbar item.

Undocumented Functions

A number of functions have been removed from the documentation set; almost all directly access specific properties that are now generally accessible through `sm_prop_get` and `sm_prop_set` and JPL. `sm_query_msg` was removed in favor of `sm_message_box`.

All undocumented functions continue to be supported in JAM 7.

Table 2. JAM library functions that are no longer documented

sm_achg	sm_getcurno	sm_protect
sm_apply_prop	sm_get_prop	sm_query_msg
sm_ascroll	sm_gp_inquire	sm_rscroll
sm_base_fldno	sm_isselected	sm_sc_max
sm_bitop	sm_n_ldb_fldno	sm_set_prop
sm_chg_attr	sm_length	sm_sibling
sm_create_id	sm_max_occur	sm_size_of_array
sm_destroy_id	sm_name	sm_t_scroll
sm_finquire	sm_novalbit	sm_t_shift
sm_fldno	sm_num_occurs	sm_unprotect
sm_fset	sm_oshift	sm_viewport
sm_ftype	sm_prop_errno	

Configuration

New Options

- KeepInMDI option in Windows initialization file determines whether newly opened screens are forced within MDI frame. *Configuration Guide*, page 157.
- Setup variable CLOSELAST_OPT can be set to OK_CLOSE_LAST so users can exit the base form without exiting the application. *Configuration Guide*, page 35. For compatibility with JAM 6 behavior, set to NO_CLOSE_LAST.
- TOOLBAR_DISPLAY enables or disables tool bar display. *Configuration Guide*, page 26..
- TOOLTIP_DISPLAY enables or disables tool tip text display. *Configuration Guide*, page 26.
- Splash screens that appear during application startup can be set as follows:
  - In the Windows initialization file through the IntroPixmap option. *Configuration Guide*, page 157.
  - In the Motif resource file through the IntroPixmap resource. *Configuration Guide*, page 174.
- Setup variable STARTSCREEN specifies the JAM screen that the application first displays. *Configuration Guide*, page 35.
- For extended selection list boxes, setup variable LISTBOX_SELECTION can be set to EXTENDED_SELECTION or SIMPLE_SELECTION. EXTENDED SELECTION is the default. For compatibility with JAM 6, set to SIMPLE_SELECTION.
- Word-wrapped text has setup variable WW_COMPATIBLE, with the settings WW_COMPATIBLE_ON and WW_COMPATIBLE_OFF. WW_COMPATIBLE_OFF is the default behavior; if set to WW_COMPATIBLE_ON word wrapping reverts to the JAM 6 behavior.

Changes

- Setup variable DECIMAL_PLACES default is now PLACES_VARIABLE.
- SMVARS now defaults to $SMBASE/config/smvars.bin
- DA_CENTBREAK's default value is now 50.
- 3D's default value is now Yes in the Windows initialization file.

SECTION TWO

Installation Guide

2

Installation Notes

These notes describe the installation of JAM 7 on your system, whether for the first time or as an upgrade to earlier versions of JAM. You should read these notes thoroughly before beginning installation.

The Installation Guide provides you with the following important information:

- The requirements to develop applications with JAM.
- Directions on how to install JAM.
- Instructions on (re–)configuring JAM.
- A manifest of files installed with JAM.
- Instructions on trouble-shooting installation problems.

Database-specific release notes detailing the setup of your JAM application as a database client are available in the online `readme.*` in the `notes` subdirectory. The list of bugs fixed in JAM 7 is provided online in the `notes` subdirectory in the file `fix700.txt`.

For those platforms which cannot support JAM's online documentation (VMS and a few less popular UNIX platforms), a complete printed set of manuals is also provided in lieu of the online documentation.

Contacting Product Support

In the unlikely event that you encounter a problem during installation, JYACC's Product Support group can help. To contact JYACC Product Support, do one of the following:

- Call 1-212-267-7722 and ask to speak with a Product Support representative. Be sure to mention your problem is related to installation.
- Send email to `support@jyacc.com`
- Send a fax to 1-212-608-4250.

If you are writing to JYACC via email or fax, include the following information:

- Your name and company name.
- Phone number, fax number, and email address.
- Customer number (if you have a maintenance agreement).
- Best time/way to contact you.
- Version of JAM (and/or other JYACC products you are installing).
- Serial number of the software you are installing.
- Type of machine you are installing on, including operating system revision, GUI environment name and version (for example, Motif 1.2.3, Microsoft Windows for Workgroups 3.11).

Information about JYACC and its products are also available on the World Wide Web at URL `http://www.jyacc.com`.

Requirements for JAM 7

The complete JAM 7 development environment requires the following:

Under Microsoft Windows

- 80386 or better processor.
- 56 Megabytes of disk space (of which 30 is for the online documentation).
- Microsoft Window 3.1 or higher.
- 8 megabytes of memory.
- MFC (Microsoft Foundation Class) must be installed.

To add your own C functions to JAM: Purchase the Microsoft Visual C 1.5 compiler.

To use JAM's PVCS interface under Windows: Purchase the Microsoft Visual C 1.5 compiler and the *PVCS Version Manager for Windows*.

Under UNIX or VMS

There are no special requirements to run JAM 7 under character mode, other than access to the standard C compiler (DEC C 4.0 for VMS, "cc" for UNIX).

To run under Motif, you need Motif equivalent to OSF version 1.2.3 or higher.

All Platforms

If you are using a database, you will need the database vendor's client and network software. For more information consult the database-specific release notes online.

Getting JAM 7 Running

In order to install and use JAM 7, you need to perform these simple steps:

1. Verify that your system has the required hardware and software
2. Consider where you want to install the software and how the files should be protected as outlined in the following pages.
3. Install and configure the JAM software from the supplied media onto your system.
4. Install any JAM database drivers from the supplied media onto your system.
5. On UNIX and VMS platforms only, configure the License Manager. Once these steps are done, you are ready to begin using JAM!

Installing JAM 7 on Windows

If you are upgrading from JAM 6, consider installing JAM 7 in a separate directory from your JAM 6 installation; for example, `C:\JAM7`.

JAM 7 is supplied in compressed form on diskettes. These distribution diskettes come with a Windows–based install program on the first disk. To run the install program:

1. Run Windows.
2. Insert the first diskette in your disk drive.
3. From the File Manager's File/Run command, type `A:INSTALL` (or `B:INSTALL`).

   The installation program walks you through the rest of the process, which ends with you rebooting to set new DOS environment variables.

As part of the install program, you are asked to select which components of JAM you want to install. By default, all components are selected. The components are:

❍ Program Files — These are mandatory to run JAM. This component contains all configuration files, utilities, and DLLs necessary to run the baseline JAM development environment.

❍ C Development Files — Necessary if you want to add your own C code, link out certain options, or link statically with a database driver (which you usually don't want to do).

❍ Online Help and Manuals — These can be quite large, and if you are short of space on your system you can skip these or share a copy across a network.

❍ Tutorial and Sample Files — This option installs the JAM Tutorial and the Videobiz sample application. Installation sets up a separate Tutorial working directory location so that you can work on and modify a personal copy of the tutorial without disturbing the original. A JAM Tutorial program group is created to make it easy to launch the copy of the tutorial.

❍ PVCS Support — Necessary if you are using PVCS version control software.

❍ JAM 5 Conversion Files — Only needed if upgrading from JAM 5. Contains `f5to6`, `dd5to6`, and `sm5api` samples.

If you are upgrading from earlier versions of JAM, the install program unsets any existing `SMVARS`, `SMPATH`, `SMTERM`, `SMMSGS`, `SMKEY`, and `SMVIDEO` in your `autoexec.bat` file. The old `autoexec.bat` is copied to `autoexec.bak`.

Note: *In order for JAM to operate correctly, JAM's* `util` *directory must be on your DOS PATH. Normally, this is not an issue, as the installation process appends the JAM* `util` *directory to the end of your DOS PATH environment variable. However, if you have installed a previous version of JAM, the DOS PATH environment variable may still contain that version's* `util` *directory. If this is the* *case, you must manually edit your* `autoexec.bat` *file to remove the older version's* `util` *directory from the PATH.*

Installing a Database Driver

The JAM database drivers are supplied separately in compressed form on diskettes and provide a Windows-based install program on the first disk. Once you have installed JAM 7 and rebooted, you are ready to begin the database driver installation.

To run the install program:

1. Run Windows.
2. Insert the first diskette in your disk drive.
3. From the File Manager's File/Run command, type `A:INSTALL` (or `B:INSTALL`).

The database driver install program presents two options:

- ❍ Complete Install — Copies all the driver files to your PC and configures `JAM7.INI` for your database version. If you have not previously installed database driver software, choose this option.
- ❍ Configuration — Simply updates `JAM7.INI` for another version of the database. If you already installed the Version 7 database driver and wish to modify the database version in `JAM7.INI`, choose this option.

The install program prompts you to select or confirm the version of your database software. JAM provides DLLs for the currently available versions of the database client software. If your version is not listed, choose the option `Other` and build a new executable using a C compiler.

INFORMIX Users

The options for the Informix install are:

Informix Version 4
Informix Version 5 lower than 5.01 WF1
Informix Version 5.01 WF1 or higher
Other

If your version is not listed, choose `Other`. In this case, you must statically build a new executable using the makefile in ***smbase***`\informix\win` when the installation completes. For more information about building a new executable, refer to Creating a New JAM Executable on page 40 and consult the `readme.inf` in the JAM `notes` subdirectory.

If you don't know your Informix version, check the release number listed in the text files provided in `INFORMIXDIR\release` where `INFORMIXDIR` is the root Informix installation.

ODBC Users

The options for ODBC are:

Microsoft Open Database Connectivity version 2
Other

If your ODBC version is 2.x or earlier, choose Version 2. Note that ODBC version 1 is compatible with version 2.

If your version is not listed, choose `Other`. In this case, you must build a new executable using the makefile in ***smbase***`\odbc\win` when the installation completes. For more information about building a new executable, refer to Creating a New JAM Executable on page 40 and consult `readme.odb` in the JAM `notes` subdirectory.

ORACLE Users

The options for Oracle are:

Oracle Version 6 using OCI
Oracle Version 7 using OCI
Oracle Version 6 using Pro*C
Oracle Version 7 using Pro*C
Other

Oracle supports two development interfaces: a C language API called OCI and an embedded SQL language named Pro*C. Most applications can use JAM's OCI or Pro*C interfaces interchangeably. Typically, most JAM developers use JAM's OCI interface unless they are linking their own custom Pro*C functions with JAM. If you wish to use Oracle 7's stored procedures, you must use the OCI interface.

If your version is not listed, choose `Other`. In this case, you must build a new executable using the makefile in ***smbase***`\oracle\win` when the installation completes. For more information about building a new executable, refer to Creating a New JAM Executable on page 40 and consult `readme.ora` in the JAM `notes` subdirectory.

If you do not know your Oracle version, check the entry in `ORACLE_HOME\dbs\register.ora` or `ORACLE_HOME\orainst\windows.rgs` where `ORACLE_HOME` is the root of the Oracle installation. The installation will want to know if you are using Oracle 6 or Oracle 7. Note that Pro*C 1.3.x is equivalent to Oracle 6 and Pro*C 1.5.x is equivalent to Oracle 7.

SYBASE Users

The options for Sybase are:

Sybase Version 4 using DB-Library
Sybase Version 10 using DB-Library
Sybase Version 10 using CT-Library
Other

For versions of Sybase earlier than 10, JAM/Sybase requires the DB-Library interface. Now with JAM/Sybase's support for Sybase OpenClient Version 10, two interfaces are supported, DB-Library and CT-Library.

If your version is not listed, choose `Other`. In this case, you must build a new executable using the makefile in ***smbase***`\sybase\win` when the installation completes. For more information about building a new executable, consult `readme.syb` in the JAM `notes` subdirectory.

If you do not know your Sybase version, run the Sybase Windows program `sqlver.exe` or refer to the About menu option of any of the Sybase Windows program (e.g., – `sybping.exe`).

Installing JAM 7 on UNIX

First Time Installations of JAM

If this is the first version of JAM to be installed on your system, take a moment to carefully consider the issues of file ownership and protections.

Ideally, JAM files are owned by the system administrator (root) or by a specially created `jam` login, and have "read-only" protection for JAM users. This is because, in normal usage, the distributed JAM files do not need to be modified and should not be modified except during configuration. Three approaches to effect this are presented here. Which is best depends upon your needs.

- The first approach is to login as `root` to install the files. After installation is complete, set the permissions so that only `root` can modify the files but all others can read and/or execute them. See `chmod` in your system manual, or type `man chmod` for details on setting permissions.
- The second approach is offered to accommodate systems for which access to the root account is tightly controlled. Create a dummy login ID (e.g., `jam`), then install JAM while logged in as `jam`. This allows whoever has access to the `jam` login account to control the ownership, permissions, and modifications. The permissions on the files may need to be modified as in the first approach.
- The third approach is to create a special group, and allow members of that group special access to the files.

Any of these three approaches requires the assistance of the system administrator (or somebody who has access to the root login). A fourth approach is to load the files into a given user's account. While this may sound convenient, it is the approach most likely to cause installation and maintenance headaches for the user.

After deciding who is going to own the JAM files (`root`, `jam`, or a group), consider the question of where they will be installed. This document usually refers to this directory as ”the JAM parent directory”.

If your system layout does not permit you to put the files in `/usr/jam`, your system may allow a symbolic link from `/usr/jam` to the actual directory using the `ln` command. At the very least, use a directory with an easily-remembered name. The worst place for the JAM parent directory is ”deep down” in a particular user's directory.

Experience has shown that once a directory has been used to hold JAM, it becomes increasingly difficult to move JAM files because users embed JAM directory names in makefiles, shell scripts, etc.

For JAM 7, it is recommended that you install the product separately from your previous JAM installations.

Proceed with installation after deciding who will own the JAM parent directory and where it will be located. The procedure described here assumes that `/usr/jam` is going to be the JAM parent directory, and that `root` or `su` is doing the installation. If you are not using `/usr/jam`, substitute the name of the directory you are actually using.

Performing the Installation

The installation process creates a JAM distribution bundled with your Database drivers enabling you to create and run JAM client applications.

1. As `root` or `su`, type

   ```
   mkdir /usr/jam
   ```

   This creates the JAM parent directory.

2. Then type

   ```
   cd /usr/jam
   ```

   This makes `/usr/jam` your current working directory.

JAM is distributed in either `tar` or `cpio` format. Read the label on the media to find out which format was used for your distribution.

Before proceeding with this section, refer to page 105 for information on installing on specific platforms. Try to find an installation command that matches your system, media type, and distribution format. If you cannot find a match, use one of the two commands below. Substitute the name of the device you will use to install JAM (i.e., the name for your media) for [*device*].

Only diskette distributions of JAM require multiple media to accommodate JAM. For distributions on other media, only one medium is required.

tar distributions:

If the JAM distribution is in `tar` format, use the following command:

```
tar xvf /dev/[device]
```

`tar` reports reaching the end of each diskette and asks that the next be inserted. When this happens, replace the diskette with the next diskette, then press Enter (or Return).

If a diskette is inserted out of order, `tar` will report an error; you must start over, beginning with the first diskette.

When `tar` is finished loading JAM, your normal prompt is displayed.

cpio distributions:

If the JAM distribution is in `cpio` format, use the following command:

```
cpio –ivdBm < /dev/[device]
```

`cpio` reports reaching the end of each diskette and asks for the name of the device to continue with. When this happens, replace the diskette with the next diskette, type in the same device name as before (`/dev/[`*device*`]`), then press Enter (or Return).

If a diskette is inserted out of order, `cpio` will report an error; you must start over, beginning with the first diskette.

When `cpio` is finished loading JAM, your normal prompt is displayed.

Database Drivers

If your Database drivers have been supplied on a separate media, then:

1. Move to the JAM parent directory (`cd /usr/jam`).
2. Repeat the extraction procedure used for the JAM distribution.

Troubleshooting

Be aware that `cpio` and `tar` are not always the friendliest of utilities. It may appear that JAM diskettes or cartridges are bad even though the problem is that an incorrect option or the wrong device name was specified in the `cpio` or `tar` command. If you encounter problems with `cpio` or `tar`,:

- Check first that you are using the correct command.
- Check that the correct options for your system were used.
- Check that the correct device name was used.

Configuration

JAM 7 has a very simple configuration on UNIX systems. The following steps must be performed:

1. On those platforms that require it, the license manager must be configured. Refer to page 67 for instructions on installing the License Manager.
2. On some installations, the appropriate `makevars.inc` file must be selected before recompiling JAM.
3. Each developer needs to access two JAM configuration files:
   - `.ebtrc` — Can either be accessed by setting the `EBTRC` environment variable to point to it (e.g., `$SMBASE/config/.ebtrc`), or it should be copied from the JAM `config` directory to the user's home directory. This file is used by the online help system.
   - `XJam` — Should be copied to the user's `app-defaults` directory, if there is one, or to the user's home directory. This file contains the X resources used by JAM at runtime.

Configuring the License Manager

The license manager is used on many UNIX systems and networks to authorize usage of the JAM authoring system. The license manager is used on the following platforms:

- x86 UNIX:
  - SCO UNIX (SVR3.2)
  - SVR4 UNIX
    - NCR 3000 Series
    - Univel Unixware
    - Solaris x86 2.x
    - Misc x86 SVR4s
- DEC
  - DEC Ultrix (MiPs RISC)
  - DEC Alpha/AXP UNIX & OSF/1 2.0

- Data General
  DG/UX 5.4.x
- HP/9000
  300/400 HP/UX
  700/800 HP/UX
- IBM RS/6000 AIX
- Siemens Nixdorf RM Series
- Silicon Graphics IRIX 4 & 5
- SUN
  x86 Solaris 2.x
  Sparc SunOS 4 (Solaris 1.x)
  Sparc SunOS 5 (Solaris 2.x)

Alternately, to find out whether you need to configure the License Manager, see if the file `lmhostid` exists in the `util` directory.

If the License Manager was installed with your JAM software, it must be configured before you can run JAM applications. Refer to the License Manager Installation on page 67 for instructions. Then return here to continue with the rest of the configuration process.

Selecting the Correct makevars.inc

The `makevars.inc` file is used by JYACC's makefiles to select operating system or C compiler-specific options. In most environments, there is only one operating system and C compiler, so JYACC needs to supply only one `makevars.inc`. However, in two ports, there are various options for compilers, and you need to chose the proper `makevars.inc` file. These two ports are for SunOS 4 (which has several C compilers) and for x86 System V Release 4 (of which there are several brands of UNIX).

On SunOS4

Because (under SunOS4) Suns can have two different compilers ("cc" and "acc"), you need to tell JAM which compiler you plan on using to build new programs.

To do this, change to the include directory:

```
cd /usr/jam/include
```

If you are using "cc" as your compiler, issue the command:

```
ln m.SUN4_4 makevars.inc
```

If you are using “acc” as your compiler, issue the command:

```
ln m.SUN4_4_SC2 makevars.inc
```

And that’s it!

On System V Release 4

The System V Release 4 (SVR4) distribution of JAM has an install script, `install.jam`, distributed with it (except for UNIXWARE). It can be found in the JAM parent directory. The script file must be run to initialize the `include` directory’s `makevars.inc` file. The script prompts for the particular System V Release 4 operating system being used. If the operating system is ESIX, choose that selection. All other platforms fall into the generic SVR4 (System V Release 4) category. To run the script, type

```
sh install.jam
```

and follow the directions.

Making a Local Copy of the Tutorial

Each user who runs the Tutorial should make a local copy of the `$SMBASE/samples/tutorial` directory to avoid inadvertent file conflicts.

Setting SMBASE, PATH, SMTERM and EBTRC

Each user’s environment must contain properly set `SMBASE`, `PATH`, and `SMTERM` environment variables in order to run JAM.

SMBASE

`SMBASE` must be set to equal the name of the JAM parent directory. If you have installed JAM as recommended, then `SMBASE` should be set to equal `/usr/jam`. If you used a different directory for JAM, replace `/usr/jam` with that directory’s name when you use the appropriate following command:

Bourne or Korn Shell users:

```
SMBASE=/usr/jam/;export SMBASE
```

C Shell users:

```
setenv SMBASE /usr/jam/
```

PATH

The environment variable `PATH` tells the operating system which directories to search when looking for executable programs. (Usually, `PATH` is initialized in the startup file `.login`, `.profile`, or `.cshrc`.) `PATH` should also contain the path to

JAM's `util` directory so that the system can find `jamdev` and the JAM utilities. In addition, if JAM's `util` directory is not on the `PATH`, business graphics will not work, nor will online help or documentation. To modify `PATH`, use the appropriate following command:

Bourne or Korn Shell:

```
PATH=$PATH:${SMBASE}/util ; export PATH
```

C Shell:

```
setenv PATH=$PATH:${SMBASE}/util
```

SMTERM

The environment variable `SMTERM` tells JAM what type/model console you are using (i.e., which video file and which key file JAM should use). For instance, if you were running a JAM application under Motif, you would set this variable as follows:

Bourne or Korn Shell:

```
SMTERM=X;export SMTERM
```

C Shell:

```
setenv SMTERM X
```

If you are running a character mode terminal with JAM 7, then you need to have appropriate video and key files (which instruct JAM on how to drive your terminal). JAM comes supplied with files for several popular terminal types (which are in turn often emulated by other brands of terminals). To find the appropriate setting of `SMTERM` (which selects the video and key files), refer to Chapter 7 in the *Configuration Guide* for detailed information on video files. Then examine the file `$SMBASE/config/smvars`. There you should find a setting of `SMTERM` which meets your needs.

Note: *You can use other environment variables instead of* `SMTERM` *to specify video and keys files separately. Refer to the Configuration Guide for more information on the environment variables* `SMVIDEO` *and* `SMKEY`*.*

EBTRC

The `EBTRC` environment variable is used under UNIX to point to the `.ebtrc` file which is distributed in the JAM `config` directory. It can be set as follows:

Bourne or Korn Shell users:

```
EBTRC=$SMBASE/config/.ebtrc; export EBTRC
```

C Shell users:

```
setenv EBTRC $SMBASE/config/.ebtrc
```

Setting SMBASE, PATH, and SMTERM Automatically

The JAM 7 UNIX releases provide two shell script files in the config directory:

- `setup.sh` for Bourne and Korn shell users.
- `setup.csh` for C–shell users.

These serve as guides for setting up the JAM environment, and can be executed directly from the startup script or merged into it. (You use the Bourne (standard) or Korn shell if you use the file `.profile` in your `login` directory. You use the C–shell if you use the file `.cshrc` in your `login` directory. If you are not sure which you are using, ask your system administrator.)

If all JAM users on the system use the same type/model console, the installer should make sure that `SMTERM` is set appropriately in the `setup.*` files. Then, if you use the C–shell, include the following line in your .cshrc script file:

```
source /usr/jam/config/setup.csh
```

If you use the Korn or Bourne shell, include the following line in your `.profile` script file:

```
. /usr/jam/config/setup.sh
```

Notice that these lines assume that JAM is installed in `/usr/jam`.

If JAM 7 is **not** installed in `/usr/jam`, the installer should modify the definition of `SMBASE` in the setup scripts. Of course, each user should correct the path to the `config` directory when including the lines above in their setup scripts.

If your terminal type is different from the value `SMTERM` is set to in `setup.csh` and `setup.sh` and you do not want to modify those files, you can easily override that value. Simply add a line to your `.cshrc` or `.profile` to set `SMTERM` correctly. The corrected line should be placed immediately following the line illustrated above.

Alternatively, you can merge the appropriate setup script file into your startup script file (`.profile` or `.cshrc`), and modify the value of `SMTERM` as needed.

Setting the XNLSPATH Environment Variable

Some implementations of Motif require you to set the `XNLSPATH` environment variable before JAM can be successfully used. Please check the documentation for your implementation of Motif to see if this is the case for you.

Installing JAM 7 on VMS

The installation process creates a JAM distribution bundled with your database drivers enabling you to create and run JAM client applications.

1. Before installing, decide where the JAM parent directory will be located. It is recommend that you use `[JAM]` as the JAM parent directory.

2. Create and set default to the JAM parent directory.

   If you are upgrading from a previous version of JAM, and the JAM parent directory is not empty, ensure that you have Read, Write, Execute, and Delete privileges for the directory and for all files and directories in it by typing:

   ```
   set file/prot=(o:wred) [...]*.*;*
   ```

   However, it is strongly recommend that the JAM parent directory be empty.

3. To install the new JAM software, place the TK50 cartridge, DAT tape, or 9–track tape in the appropriate drive and type the following:

   ```
   mount/foreign msa0:jyacc

   backup msa0:jam.bck/sav/select=[JYACC...] –
   [...]/new_version/log
   ```

   where `msa0` is the name of your tape device.

If your database drivers have been supplied on a separate media then follow the instructions below:

1. Go to the JAM parent directory (`SET DEF [JAM]`).

2. Repeat the extraction procedure used for the JAM distribution.

This completes the installation of JAM. However, the software must be configured and the JAM license must be verified before JAM applications can be developed.

VMS Configuration

Before the installed product is usable, it must be configured. Some of the steps can be skipped if the product is installed in [JAM], but most must be performed regardless.

If you find that some of the JAM utilities do not run on your system, it may be because you are not using the same versions of VMS and DEC C that were used to generate the utilities. In this case, the utilities need to be relinked on your system. Refer to page 102 for more details.

Running SETUP.COM

Once you have loaded JAM onto your system, you should set default to the JAM parent directory and examine the file `SETUP.COM`. This file is run to set up the JAM runtime environment. JAM users need to run it before running JAM applications.

In `SETUP.COM`, check the definition of JAM. If JAM is not defined correctly to be the name of the JAM parent directory, edit `SETUP.COM` and correct the symbol definition.

Later, if you decide not to put the JAM licensing information file in the JAM `[.CONFIG]` directory, define `LM_LICENSE_FILE` in this file also.

Execute the command file `SETUP.COM`. This sets up the JAM environment for you:

```
@SETUP
```

`SETUP.COM` sets the JAM environment variable `SMTERM` by asking for the type of your terminal; VT100, VT200 and VT220 are acceptable responses, assuming you have one of those types of terminals. If you have another type of terminal, examine the file `SMBASE:[CONFIG]SMVARS` to see what other terminals JAM supports.

`SETUP.COM` defines the logical name `SMBASE` as a concealed device. Note that JAM uses this logical name extensively. The files `[CONFIG]SMVARS.`, `[CONFIG]SMVARS.BIN`, and `JAM.COM` depend on the value of `SMBASE`.

`SETUP.COM` adds JAM utilities to the user environment by executing `@JAM`.

`SETUP.COM` defines the JAM environment variable `SMVARS`. `SMVARS` points to `[CONFIG]SMVARS.BIN` in JAM. JAM reads the binary file `SMVARS.BIN` to find out the values of the variables `SMKEY` and `SMVIDEO`, among others. `SMKEY` and `SMVIDEO`, in turn, point to a binary key translation file and a binary video file. (Key translation files are referred to in this document as key files.) If there are video and key files listed in the ASCII file `SMBASE:[CONFIG]MSGFILE.BIN`, then that type of terminal is supported by JAM. `SMMSGS` is the variable that points to `SMBASE:[CONFIG]MSGFILE.BIN`, and `SMPATH` is the variable that points to the directory where the JAM screen libraries reside. These variables must be set before running JAM. There are other JAM variables that you can set; refer to the *Configuration Guide* for details. Variables defined as logical names in your environment always supersede the ones in `SMVARS.BIN`. If you modify `SMVARS.`, you need to run `VAR2BIN` to compile it into binary format:

```
SET DEF SMBASE:[CONFIG]
VAR2BIN SMVARS
```

This creates a new `SMVARS.BIN` file.

`SETUP.COM` defines a symbol, MAKE, to represent a DCL command file, `SMBASE:[000000]MAKE.COM`. Type

```
MAKE –H
```

to learn more about it.

Licensing JAM 7

The License Manager is used to authorize usage of the JAM authoring system. It is installed with your JAM software and must be configured before you can run JAM applications. Refer to the License Manager Installation on page 67. Then return here to continue with the rest of configuration.

Preparing the VideoBiz Sample Application

Before the VideoBiz sample application can be used, a few configuration steps need to be taken.

1. Run the makefile in `...[SAMPLES.VIDEOBIZ.CONFIG]`. For example:

   ```
   SET DEF SMBASE:[SAMPLES.VIDEOBIZ.CONFIG]
   @MAKE
   ```

   This will compile VideoBiz's configuration map file (`CMAP.`), setup file (`VBSETUP`) and generate a Motif resource file (`XJAM.DAT`). These files will be picked up when the users run `VBSETUP.COM` to configure for VideoBiz.

2. Each user needs to make a local copy of the JAM configuration file, `XJAM.DAT`. The file `XJAM.DAT` should be copied to the user's `app–defaults` directory, if there is one, or to the user's home directory.

Making Local Copies of SETUP.COM

It is recommended that users run `SETUP.COM` from their `LOGIN.COM` files, or that it be run from the system startup file.

Troubleshooting an Installation

In order to verify that you have installed JAM 7 properly, you should set up a user as described above. Then, with the environment variables properly set, you should start the `jamdev` executable, either by typing `jamdev` in UNIX or VMS, or by double clicking the `jamdev` icon under Windows.

The installation was successful if you see the screen editor brought up on your display. Note that in Windows you must reboot your computer after finishing the installation before trying to start JAM (as JAM's environment variables are set in `autoexec.bat`).

An error message appears if you attempt to run JAM without configuring, or if you have configured improperly. But the message does not always tell how to correct the problem because JAM can only see the problem from its perspective. The installation was not successful if you receive an error message and the program terminates. The following sections describe some of the most common error messages and how to resolve them.

Error Messages

`SMVARS: Environment variable missing`

Cause: `SMBASE` (or `SMVARS`, if you are using one) is not set, or not set properly.

Action: Normally, JAM looks for the `smvars` file in the `config` directory under the directory pointed to by `SMBASE`. If `SMBASE` is not set, you must have an `SMVARS` variable to tell JAM explicitly where the `smvars` file can be found. If `SMBASE` is not set property, and JAM cannot find an `smvars` variable, the above message is given.

You should correct (or set) the `SMBASE` environment variable, or determine why setting it is not taking effect.

Windows users: Check to see that it's being set in `autoexec.bat` and that you have not run out of environment space.

UNIX users: Make sure that the variable is exported (via the `export sh/ksh` command or `setenv csh` command).

`SMVARS: No such file or directory`

Cause: `SMVARS` is not set correctly. The most common cause of an incorrect value for `SMVARS`.

Action: Check that the full pathname (including drive letter under Windows) of the file is included and correct.

`SMVARS: Bad file format`

Cause: `SMVARS` is set to be the name of an existing file, but it is not a binary `SMVARS` file. The most common incorrect value for `SMVARS` in this case is the path and name of the source `SMVARS` file instead of the binary `SMVARS` file.

Action: Point to the binary file that has the `.bin` extension.

```
Please enter terminal type or <RETURN> to exit.
```

Cause: `SMTERM` is not set (and, under UNIX, `TERM` is also not set) or the value could not be found in `smvars.bin`.

Action: See the left column of `smvars`), in the environment, or, in Windows, in `JAM7.INI`. Press Enter (or Return), and set `SMTERM` correctly and then try to run `jamdev` again.

```
(Filename): No such file or directory
```

Cause: The file name in parenthesis is specified as the value of the `SMVIDEO`, `SMKEY`, `SMMSGS`, or some similar configuration variable in the environment or in `smvars.bin`, but the file could not be found.

Action: Correct the name, remembering to use the full pathname of the file. If you correct it in `smvars`, `smvars.bin` must be recompile with the `var2bin` utility.

```
SMMSGS: Environment variable missing
```

Cause: (or the same message with some other configuration variable). JAM could not find an entry for the cited variable in the `SMVARS` file or in the environment. It can be defined in either (refer to the following note). Getting this error usually means that you have some configuration variables defined in your environment, but not enough, and `SMVARS` is not defined.

Note: *Most JAM configuration variables can be set in the environment rather than depending on values set in an* `SMVARs`*-defined file. If JAM can't find a required configuration variable in either the environment or in the* `SMVARS` *file, that variable will be cited as missing. Refer to the Configuration Guide for more details.*

If you are running in character mode and the screen appears disorganized, with all the text bunched together along with special characters, check the value of `SMTERM`; you probably only need to change its value, but you may also need to create a new video file.

Recurring errors: Occasionally, one of the errors described above seems to recur even though the file `smvars` is in order. More often than not, the problem is that the file `smvars` in the `config` directory, although correct, has not been converted to `smvars.bin`, the binary representation that JAM needs.

REMEMBER: if you modify `smvars`, be sure to run `var2bin` to convert it to binary format. Otherwise JAM is not aware of the changes, and it will seem as though `smvars` was not updated.

DLL Not Found Messages

```
Cannot find LINF4DMW.DLL
Cannot find LINF5DMW.DLL.
Cannot find TMINF1W.DLL
```

Cause: Windows could not find the JAM/Database Driver DLLs.
Action: Check that the JAM 7 `util` directory is on the `PATH`. If you do not wish to use the JAM/Database Driver DLLs, edit `JAM7.INI` and remove the database name from the "installed" entry.

```
Cannot find LDLLSQLW.DLL
Cannot load DLL linf4dmw.dll; LoadLibrary error 2.
```

```
Cannot find ISQLI501.DLL
Cannot load.DLL linf5dmw.dll; LoadLibrary error 2.
```
or
```
Cannot load DLL li501dmw.dll; LoadLibrary error 2.
```

Cause: Windows could not find the Informix 4.10 or 5.0 client software.
Action: Verify that `INFORMIXDIR` is set in the environment and that `%INFORMIXDIR%\bin` is in the `PATH`.

```
Cannot find LODBDMW.DLL
Cannot find TMODB1W.DLL
```

Cause: Windows could not find the JAM/Database Driver DLLs.
Action: Check that the JAM 7 `util` directory is on the `PATH`. If you do not wish to use the JAM/Database Driver DLLs, edit `JAM7.INI` and remove the database name from the "installed" entry.

```
Cannot find ODBC.DLL
Cannot load DLL lodbdmw.dll; LoadLibrary error 2.
```

Cause: Windows could not find the ODBC Driver Manager.
Action: This software is often installed in the Windows directory. If it was installed in another location, make sure the installation's `bin` directory is in the `PATH`.

```
Cannot find LORA6MW.DLL
Cannot find LORA7MW.DLL
Cannot find LEMB6MW.DLL
Cannot find LEMB7MW.DLL
Cannot find TMORA1W.DLL
```

Cause: Windows could not find the JAM/Database Driver DLLs.
Action: Check that the JAM 7 `util` directory is on the `PATH`. If you do not wish to use the JAM/Database Driver DLLs, edit `JAM7.INI` and remove the database name from the "installed" entry.

```
Cannot find ORA6WIN.DLL
Cannot load DLL lora6dmw.dll; LoadLibrary error 2.
```

```
Cannot find ORA7WIN.DLL
Cannot load DLL lora7dmw.dll; LoadLibrary error 2.
```

```
Cannot find SQL13WIN.DLL
Cannot load DLL lemb6dmw.dll; LoadLibrary error 2.
```

```
Cannot find SQL15WIN.DLL
Cannot load DLL lemb7dmw.dll; LoadLibrary error 2.
```

Cause: Windows could not find the Oracle client software.
Action: If you are using Oracle 6 client, verify that `ORACLE_HOME` is set in the environment; if you are using Oracle 7 client, verify that `ORACLE_HOME` is in your `ORACLE.INI` file. Check that `%ORACLE_HOME%\bin` is in the `PATH`.

```
Cannot find LSYB4DMW.DLL
Cannot find LSDBDMW.DLL
Cannot find LSCTDMW.DLL
Cannot find TMSYB1W.DLL
```

Cause: Windows could not find the JAM/Database Driver DLLs.
Action: Check that the JAM 7 `util` directory is on the `PATH`. If you do not wish to use the JAM/Database Driver DLLs, edit `JAM7.INI` and remove the database name from the “installed” entry.

```
Cannot find W3DBLIB.DLL
Cannot load DLL lsyb4dmw.dll; LoadLibrary error 2.
```

Cause: Windows could not find the Sybase 4 software.
Action: Verify that the `SYBASE DLL` directory is in your `PATH`. If it is not, exit Windows, correct the environment variable, and restart Windows.

```
Cannot find WCTLIB.DLL
Cannot find WCSLIB.DLL C
annot load DLL lsctdmw.dll; LoadLibrary error 2.
```

```
Cannot find WSYBDB.DLL
Cannot load DLL lsdbdmw.dll; LoadLibrary error 2.
```

Cause: Windows could not find the Sybase 10 software.
Action: Verify that the `SYBASE DLL` directory is in your `PATH`. If it is not, exit Windows, correct the environment variable, and restart Windows.

Cause: Windows could not find the Sybase client software.
Action: Verify that `SYBASE` is set in the environment and that `%SYBASE%\dll` is in the `PATH`.

LINK: warning L4051: lafxcw.lib: cannot find library

JAM is dependent on Microsoft Foundation Class (MFC) for toolbars under Windows. Therefore MFC should be installed prior to installing JAM.

Cause: MFC is not installed.
Action: Install MFC and add the `mfc/lib` directory to the `LIB` environment variable.

Warning: chart <Begin> failed: –1

and all graphs remaining blank indicates that the `gdsp` program (which is installed in the `util` directory) was not found on the path. (Unix only.)

Warning: missing graph files in $SMPATH

and all graphs remaining blank indicates that:

UNIX:	The `grafcap` file was invalid or not found in `$SMPATH`
Windows:	The IPT setting in the file `LIBSTI.INI` does not point to the directory holding the `grafcap` file.

License Manager Error Messages

The License Manager Installation section fully describes the errors the License Manager generates. Below are some of the more common messages:

`Invalid Encryption Code`

Cause: You may have typed incorrect characters when you created the password file.
Action: Compare the hardcopy that JYACC faxed you to the online file. Make sure you entered a zero rather than the letter O, and the number 1 rather than a lower case L (l). Then remove all unnecessary spaces.

`Cannot communicate with the license server`

Cause: This error can be caused by a variety of problems.
Action: Begin troubleshooting by referring to the `license.log` file, which should have been updated when `lmgrd` was started. The log file should indicate whether the server has started up properly, and if not, the source of the problem.

`No such FEATURE exists.`

Cause: Most likely, this error means that the License Manager is unable to locate the correct JYACC license file (usually `license.dat`).

Troubleshooting Business Graphics

Because JAM's business graphics capabilities rely upon external programs (.DLLs in Windows), it can be easy to get into a situation where all of JAM except for business graphics is working. To troubleshoot business graphics, you should keep the following in mind:

Under Unix / VMS:

- There needs to be a copy of the `grafcap` file (which is distributed in the `config` directory) in one of the directories pointed to by `SMPATH`.
- The files `gdsp` and `swsdrvr` (which are distributed in the `util` directory) must be found along the search `PATH` for executables (that is, JAM must be able to `exec()` these programs).

Under Windows:

- `libsti.ini` needs to be copied to the Windows main directory and contain the correct paths in it for the `IPT` variable (this is done automatically by the installation program).
- The `libsti.dll` (which is distributed in the `util` directory) file needs to be found in a directory along the `PATH`.

Troubleshooting Online Help and Manuals

JAM's online help and manuals are built using a text retrieval package called DynaText produced by Electronic Book Technologies. (Hence you will see `ebt` appear in many filenames).

Note that you can run the `dtext` program in the `util` directory (or double-click its icon under Windows) to verify that things are working. `dtext` will open the top level view of the manuals available online, including a manual about DynaText itself. This also allows access to all of JAM's manuals, not just the *Editors Guide* which is available from within `jamdev` and serves as JAM's online help facility. In order for the online help system to work:

Under UNIX:

- The JAM `util` directory has to be in the `PATH` environment variable.

- The help engine needs to find the file `.ebtrc`. This can be done either by placing a copy of it in your home directory (and each user’s home directory), or by having an `EBTRC` environment variable which points to a copy elsewhere (usually, in the JAM `config` directory).
- The contents of the `.ebtrc` file must be correct. Since `.ebtrc` relies on the `SMBASE` environment variable, that must be set and exported.

Under Windows:
Note: *The installation program normally performs all of these steps for you automatically.*

- The JAM `util` directory has to be in the `PATH` environment variable.
- The `dynatext.ini` file must be copied to the main Windows directory.
- The `SMBASE` environment variable must be set.

Creating a New JAM Executable

Note: *The procedures and files described here assume that your C compiler is installed and working properly.*

Windows users should note that `jamdev.exe` and `jam.exe` in the `util` subdirectory are already enabled for the database(s) selected during the installation process.

It is not necessary to build a new executable unless you are:

- Adding new C code to JAM.
- Using a database driver which does not provide DLL support.

If you do need to build a new JAM executable follow steps 1, 2, 3 and 4 below.

STEP 1

Choose the appropriate makefile to build your application, where `smbase` is your JAM parent directory, and database-name is your database engine:

	UNIX	Windows	VMS
JAM	`/smbase/link/makefile`	`\smbase\link\makefile`	`smbase:[link]makefile`
JAM enabled for JYACC's JDB database	`/smbase/link/makefile`	`\smbase\link\makefile`	`smbase:[link]makefile`
Character-mode JAM enabled for database-name	`smbase/database-name/unix/makefile`	N/A	`smbase:[database-name.vms]makefile`
Motif JAM enabled for database-name	`smbase/database-name/xwin/makefile`	N/A	`smbase:[database-name.xwin]makefile`
Windows JAM enabled for database-name (utilizing DLLs)	N/A	`\smbase\link\makefile`	N/A
Windows JAM enabled for database-name (statically built)	N/A	`\smbase\database-name\win\makefile`	N/A

Note that the make process creates a new JAM editor executable, `jamdev.exe`; you may wish to give it a unique name to distinguish it from the distributed `jamdev.exe` or others that you have built differently.

STEP 2

To use any makefile, first copy it from the appropriate directory to your JAM application development directory. This should be a local directory owned by the developer (or the team, if several people are working on the same project). You do not want to work on the files directly in the JAM distribution directories; these should be kept pristine for future use.

STEP 3

Edit the makefile as needed. When editing the makefile, note that the continuation character is '\'. No blanks, tabs, or other characters should be on the line immediately following the '\'. On some systems, the continuation character also continues comment lines (lines that have a `#` as the first character), so be careful that any lines you comment out in the future do not end with the '\' character. Also, commented lines should not be placed between continued lines.

A.

Select or deselect what to make by commenting or uncommenting the lines in the makefiles. The following are the defaults:

`JAM = JAM`	(`jam.exe` under Windows)
`JAMDEV = jamdev`	(`jamdev.exe` under Windows)
`#RWRUN = rwrun`	(`rwrun.exe` under Windows)

If you do not have Report Writer installed, leave the Report Writer lines as comments.

B.

If you rename your makefile make sure to modify the `MAKEFILE` macro.

```
MAKEFILE = makefile
```

For example, if you rename the makefile to foo, change the macro to `MAKEFILE = foo`.

C.

If you are adding your own C modules, you must instruct the makefile to compile and link in your modules. To do this, add the filenames to the `SRCMODS` macro. For example, if your application's C source code modules are `mymod1.c`, `mymod2.c`, and `mymod3.c`, you would change the `SRCMODS` line of the makefile to the following lines:

Windows	UNIX
`SRCMODS = \`	`SRCMODS = \`
`mymod1.obj \`	`mymod1.o \`
`mymod2.obj \`	`mymod2.o \`
`mymod3.obj \`	`mymod3.o \`
`funclist.obj`	`funclist.o`

D.

Some of your software components may be installed in directories with names different than the default. If `SMBASE` is not set in the environment, you can add a line in the makefile to define it, for example:

Windows	UNIX
`SMBASE=\jam7`	`SMBASE=/usr/jam`
`DBIBASE=$(SMBASE)\`*database-name*	`DBIBASE=$(SMBASE)/`*database-name*

E.

To add Microsoft Codeview debugging information for Windows applications, un-comment the following block:

```
# ---------------- START OF BLOCK ------------------ #
#DEBUG_CFLAGS= –Od –f– –Zi –Fduser.pdb
#DEBUG_LDFLAGS= /co
# ---------------- END OF BLOCK -------------------- #
```

If you have added your own C modules and need a debugger to resolve a problem in your source code, uncomment this block and recompile. For some technical support issues, a JYACC Product Support representative may ask you to uncomment this block and recompile.

F.

The JAM debugger allows you to trace JPL and JAM screen events. It is installed by default, however if you do not wish to use the JAM debugger, comment the following block:

Windows	UNIX
`#JAMDBGCFLAGS= –DJAMDEBUG=1`	`#JAMDBGCFLAGS= –DJAMDEBUG=1`
`#JAMDBGRFLAGS= –DJAMDEBUG=1`	`#JAMDBGMODS= smdbinit.o`
`#JAMDBGMODS= smdbinit.obj` `#JAMDBGLIBS= $(SMBASE)\lib\llibsmdb.lib`	`#JAMDBGLIB= $(SMBASE)/lib/libsmdb.a`

G.

If you have installed JAM/ReportWriter, and wish to rebuild `jam` and `jamdev` with it, verify the setting of `RWBASE`. If you did not install ReportWriter in the `smbase` directory, set `RWBASE` appropriately and uncomment the Report Writer block:

Windows	UNIX
`RWBASE=$(SMBASE)\rw` `RWLIB = $(RWBASE)\rwwin\llibrww.lib` `RWCFLAGS = –DJAMRW=1`	`RWBASE=$(SMBASE)/rw` `RWLIB = $(RWBASE)/librw.a` `RWCFLAGS = –DJAMRW=1`

H.

The section `DATABASE INSTALLATION PARAMETERS` defines the databases for the executables. Using this makefile you can build an executable with both JYACC's JDB database and *database-name* or you can build an executable with just *database-name*.

To build for your specific database, the flag `<db-extension>_INIT` must be set to one of the following: `d`, `l`, `u`, `p`. These flags control the handling for case sensitivity. The default is `d`. To find out what the JAM default is for your database engine, refer to the database-specific release notes online. You can also consult the *Database Guide* for more information about case handling.

To use JDB, the flag `JDB_INIT` must be set to one of the following: `d`, `l`, `u`, `p`. If you do *not* wish to use JDB, set `JDB_INIT=n`. The flags `<db-extension>_ENGNAME` and `JDB_ENGNAME` set the default engine names. Consult the *Database Guide* before changing these values.

I.

An important section to pay attention to is `DATABASE PARAMETERS`. This section defines the paths to the database engine's header files and libraries. Verify your database engine's version. Uncomment the appropriate block of parameters based upon this version. Also, verify and correct the pathnames if necessary.

STEP 4

Run the compiler utility to build the executable.

Windows	UNIX
`nmake`	`make`

You do not have to understand everything in the makefile for it to serve your needs. But if you would like to learn more about makefiles, refer to your compiler's documentation on the `make` or `nmake` utility.

The database-specific release notes in the JAM `notes` subdirectory provide additional information about the database makefiles.

Things to remember:

- You need `SMBASE` set in your environment.
- You need the appropriate database version selected.
- The database environment variables should be setup.
- The compiler environment variables should be setup.

Distributed JAM Software

The successful installation of JAM results in the creation of the following directories in the JAM parent directory:

Directory/File	Contents	Notes
database-name	Files needed to (re)build `jam` and `jamdev` that will serve as a database-name client application.	
`install.com`	DCL file to be run as part of the licensing process	VMS only
`jam.com`	DCL symbol definitions for JAM utilities	VMS only
`make.com`	DCL file for program development	VMS only
`make.hlp`	TEXT file for `MAKE.COM`	VMS only
`setup.com`	DCL file for setting the JAM environment	VMS only
`upgrade.com`	DCL file that ensures correct “makes” if upgrading	VMS only
`config`	Configuration and setup files and screen libraries	
`docs`	Online documentation	
`include`	Header files for C program development	
`jdb`	Tools and utilities for JDB database	
`lib`	Libraries for C program development	
`link`	Files for building a custom JAM executable	
`notes`	Additional JAM-related documents	
`patch`	Patches to released JYACC products (if any)	
`samples`	Videobiz sample database application and Tutorial	
`util`	Utility programs and files including `jamdev`	

The following sections describe the contents of the directories installed with JAM 7:

The config Directory

The `config` directory contains configuration files needed by JAM applications at execution time. Refer to the *Configuration Guide* to learn how to convert the text files into their JAM-usable binary formats.

Because there are so many files in this directory (especially on UNIX and VMS installations), it is not possible to provide a file by file description. Instead, some important files are described individually, and others are described in general terms:

File	Description	Notes
`*cmap`, `*cmap.bin`	JAM configuration map files. These files enhance JAM's portability by allowing it to deal with arbitrary color names, fonts, and line styles in all environments. The `*cmap` files also contain JAM's color scheme, which tells JAM what colors to use by default. The utility `cmap2bin` converts the text file to binary format.	
`dmdatabase`, `dmdatabase.jpl`	Used by the Importer to access table information in databases.	
`dmjdb`	Binary JPL file used by Importer to access JDB.	
`dmjdb.jpl`	Source to `dmjdb`.	
`dminf`	Binary JPL file used by Importer to access Informix	Informix driver only
`dminf.jpl`	Source to `dminf`	Informix driver only
`dmodb`	Binary JPL file used by Importer to access ODBC	ODBC driver only
`dmodb.jpl`	Source to `dmodb`	ODBC driver only
`dmora`	Binary JPL file used by Importer to access Oracle	Oracle driver only
`dmora.jpl`	Source to `dmora`	Oracle driver only
`dmsyb`	Binary JPL file used by Importer to access Sybase	Sybase driver only
`dmsyb.jpl`	Source to `dmsyb`	Sybase driver only

File	Description	Notes
`dynatext.ini`	Initialization file for the online help engine. It is copied to the Windows directory automatically by the installation process.	WIN only
`.ebtrc`	Setup file for the online help engine. Either the `EBTRC` environment variable should be set to point to this file, or it should be copied to each user's home directory.	UNIX only
`*.fnt`	Business graphics font files. If you use any of these fonts in your application, you will need to include them in your distribution.	
`grafcap`	Initialization file for business graphics package.	
`jam7.ini`	Initialization file for `jam.exe` and `jamdev.exe`. It is copied to the Windows directory automatically by the installation process.	WIN only
`*keys`, `*keys.bin`	Files describing the key mapping used by JAM. The `*keys` files are ASCII text files. The `*keys.bin` files are the binary files used by JAM. The utility `key2bin` translates text key files into binary format. The utility `showkey` can be used to identify key mappings.	
`*vid`, `*vid.bin`	JAM video files. Describe the capabilities and attributes of terminals used by JAM. The `*vid` files are ASCII text files, while the `*vid.bin` files are the binary compiled files used by JAM. Each text file contains a comment indicating the kind of terminal that the file is suitable for. The utility `vid2bin` converts the ASCII text files to the binary files.	
`hiback.c`	Source to DOS program to see background highlight (for use with J*term*)	UNIX & VMS
`jam7.lib`	JAM's runtime support library	
`jamdev7.lib`	JAM's development time support library	
`jicmap.bin`, `jisetup.bin`, `jisql.ini`, `jivars.bin`	Various internal configuration files used for the JDB ISQL tool.	
`*.lib`	Various other internal JAM libraries.	

File	Description	Notes
`libsti.ini`	Initialization file for the business graphics engine. It is copied to the Windows directory automatically by the installation process.	WIN only
`mbedit.ini`	Initialization file for the menu bar editor. It is copied to the Windows directory automatically by the installation process.	WIN only
`msgfile`, `msgfile.bin`	The text and binary representations of JAM's message file (which contains the text of prompts and messages used in JAM). The utility `msg2bin` converts the text file into binary format.	
`net*.dat`	Sample license manager configuration files.	UNIX only
`odbcgbls`	JPL include to define ODBC constants.	WIN only
`smvars`, `smvars.bin`	The text and binary representations of JAM's configuration file. The utility `var2bin` converts configuration files (e.g., `smvars` and `smsetup`) from text into binary format.	
`*setup`, `*setup.bin`	Special JAM settings used in various environments.	
`setup.sh`, `setup.csh`	Sample setup scripts for Bourne Shell and for C Shell.	UNIX only
`smjgbls`	JPL include file to define all JAM constants. Used by default in all JAM applications. This file is in text, and can be edited to limit the number of globals loaded into JAM.	
`smwizard.jpl`	JPL public file for the JAM screen wizard's generated screens. Handles all processing for screen's buttons and grid row functions.	
`smwzmenu.mnu`, `smwzmenu`	Menu bar for JAM screen wizard's generated screens.	
`styles.sty`	Default/sample transaction manager style file.	
`symbold`, `symbols1`	Font files for business graphics.	
`tpimsgs`	Messages for JAM/TP*i*.	
`XJam`	Sample `XJam` file for Motif users. This should be copied to the home directories of each user.	UNIX & VMS

The database Directories

Files to support JAM database drivers are provided in subdirectories named after the database they correspond to. These directories supply all the necessary header files, source files, and libraries needed to build JAM 7 programs with the JAM database driver and with the transaction manager.

File/Directory	Description
database directory:	
`tm`*xxx*`1.c`	Source to *xxx* model for the transaction manager, where *xxx* is a three letter abbreviation for the database name.
`*.ec`, `*.sc`, `*.c`, `*.pc`	Database-specific source files.
database`\win` directory	Windows Only: Holds database-specific makefile for Windows executables.
database`/unix` directory	UNIX Only: Holds database-specific makefile for character mode executables.
database`/xwin` directory	UNIX Only: Holds database-specific makefile for Motif executables.
[*database*`.vms`] directory	VMS Only: Holds database-specific makefile for character mode executables.
[*database*`.xwin`] directory	Holds database-specific makefile for Motif executables.

Note that JDB has a different directory structure, since JYACC provides the database as well as the interface.

The docs Directory

The `docs` directory holds JAM's online documentation. It is not present on those systems which do not support JAM's help engine, including VMS.

The include Directory

The header files in the `include` directory are used in the rebuilding of your JAM executable. They are:

File	Description
`dmerror.h`	Definitions for JAM database driver errors.
`dminf.h`	
`dmmach.h`	Platform-specific definitions for JAM database drivers.
`dmuproto.h`	Prototypes for user-callable JAM database driver functions.
`m.*` or `makevars.inc`	macro definitions for the makefile that is distributed in the `link` directory. This should not be modified or moved. Refer to configuration section on page 26.
`smaltsc.h`	Structures and definitions for custom scrolling.
`smascii.h`	Definitions of ASCII control characters.
`smattrib.h`	Definitions of attributes (colors, highlight, reverse).
`smbitops.h`	File needed for compatibility with previous versions of JAM.
`smcmprs.h`	UNIX & VMS Only: J*term* data compression definitions.
`smcommon.h`	Included in `smmach.h`
`smdbdefs.h`	Database definitions.
`smdefs.h`	Includes all commonly used JAM header files.
`smedits.h`	Field edits definitions.
`smerror.h`	Error code definitions for use with library routines.
`smglobs.h`	Definitions for variable inquiry functions.
`sminstfn.h`	Definitions for function list installation.
`smio.h`	Definitions for system independent terminal I/O.
`smkeys.h`	Definitions of the JAM key codes. Needed to handle returns from keyboard processing routines.
`smmach.h`	Definitions of machine-specific characteristics. Helps make JAM applications portable.
`smmenu.h`	JAM 5 menu API definitions.
`smmenu6.h`,	JAM menu API definitions.
`smmwuser.h`,	JAM Windows interface-specific definitions.
`smmwwids.h`	JAM Windows interface-specific definitions.
`smpi.h`	For compatibility; includes `smpiprot.h`.
`smpiinit.h`	JAM GUI interface initialization functions definitions.
`smpiprot.h`	JAM GUI interface common library function definitions.
`smproto.h`	Prototypes for documented functions.

File	Description
`smsetup.h`	Definitions for the various setup variables that can be defined in `smvars` and `smsetup`.
`smumisc.h`	Miscellaneous definitions.
`smuprapi.h`	Header file for the properties API.
`smuprdb.h`	
`smuprdef.h`	
`smuprops.h`	
`smusrdbi.h`	Definitions for JAM database driver initialization.
`smvalids.h`	Definitions for installed functions.
smvideo.h	Definitions for video file entries.
`smwin.h`	Replacement for `windows.h`
`smxmuser.h`	JAM Motif interface-specific definitions.
`tmsubs.h`	Definitions and prototypes for transaction manager.

The jdb Directory

JAM 7 supplies a single-user database called JDB to help developers learn the product and prototype applications. The JDB files are supplied in the `jdb` directory.

File/Directory	Description
`bin` subdirectory:	
`isql`[2][`.exe`[1]]	Command line utility for creating JDB databases and executing SQL statements.
`mksql`[`.exe`[1]]	Utility for making SQL.
`tbldata`[`.exe`[1]]	Utility for importing and exporting JDB data.
`files` subdirectory:	
`jdberr.txt`	Text file of JDB error and warning messages.
`magic`	File definitions for `magic` system utility
`version`	Text file containing JDB version number and porting date.
`include` subdirectory:	
`api.h`	Prototypes for JDB application interface.
`jdbcmn.h`	Definitions for JDB error codes.

[1]Executables in VMS and Windows have the .exe extension·
[2]Note that the JISQL utility, which is a visual version of ISQL, is to be found in the util directory.

File/Directory	Description
`lib` subdirectory:	
`jdbw.lib`	Windows Only: JDB SQL library
`jdbfmw.lib`	Windows Only: JDB File Management library
`libjdb.a`	UNIX Only: JDB SQL library
`libjdbfm.a`	UNIX Only: JDB File Management library
`libjdb.olb`	VMS Only: JDB SQL library.
`libjdbfm.olb`	VMS Only: JDB File Management library.

[1]Executables in VMS and Windows have the .exe extension·
[2]Note that the JISQL utility, which is a visual version of ISQL, is to be found in the util directory.

The lib Directory

The `lib` directory contains the JAM object library files necessary for compiling and linking your applications with JAM. Note that under Windows, libraries have the `.lib` extension, under UNIX the `.a` extension, and under VMS the `.olb` extension. The files are:

Library	Description
SUN only:	
`ansistub`	Library of ANSI routines for use with "cc" compiler
UNIX & VMS only:	
`libdm`	JAM database driver Common Support Library
`libjdbdm`	JDB Support and Model Library
`libjpeg`	JPEG format reader library. N.B.: This software is based in part on the work of the Independent JPEG Group.
`libjx`	Library of screen editor routines
`libpi`	JAM GUI interface common code library
`libpixm`	JAM Motif interface code library (there may also be other versions for specific motif implementations, e.g., `libpixm_ixi` for IXI Motif).
`libsm`	Screen manager routines
`libsmdb`	JAM/Debugger library.
`libsti`	Business graphics library
`libtm`	JAM transaction manager common library
`libXpm`	Bitmap support library

Library	Description
Windows only:	
`cktbl16.lib`	Grid widget
`ctl3ds.lib`,	3D controls
`libsti.lib`	Business graphics
`llibdmw.lib`	JAM database driver Common Support Library
`ljdbdmw.lib`	JDB Support and Model Library
`llibjxw.lib`	Screen editor routines
`llibmw.lib`	JAM Windows interface code library
`llibpiw.lib`	JAM GUI interface common code library
`llibsmdb.lib`	JAM/Debugger library
`llibsmw.lib`	Screen manager routines
`llibtmw.lib` `llibwc.lib`	JAM transaction manager common library
`pvcsbwi.lib`, `pvcsvmw.lib`	PVCS support libraries

If you are using the Informix driver, the lib directory also contains the following:

File	Description
Windows:	
`linf4dmw.lib`	Informix 4 Support and Model Library
`linf5dmw.lib`	Informix 5 Support and Model Library
UNIX:	
`libinf50dm.a`	Informix Support and Model Library
`libinf60dm.a`	Informix Support and Model Library
VMS:	
`libinf41dm.olb`	Informix Support and Model Library
`libinf50dm.olb`	Informix Support and Model Library

If you are using the ODBC driver, the `lib` directory also contains the following:

File	Description
`lodbdmw.lib`	ODBC Support and Model Library

If you are using the Oracle driver, the `lib` directory also contains the following:

File	Description
Windows:	
`lora6dmw.lib`	Oracle 6 OCI Support and Model Library
`lemb6dmw.lib`	Oracle 6 Pro*C Support and Model Library
`lora7dmw.lib`	Oracle 7 OCI Support and Model Library
`lemb7dmw.lib`	Oracle 7 Pro*C Support and Model Library
UNIX:	
`libora6dm.a`	Oracle 6 OCI Support and Model Library
`libora7dm.a`	Oracle 7 OCI Support and Model Library
`libembdm.a`	Oracle Pro*C Support and Model Library
VMS:	
`liboradm.olb`	Oracle OCI Support and Model Library
`libembdm.olb`	Oracle Pro*C Support and Model Library

If you are using the Sybase driver, the lib directory also contains the following:

File	Description
Windows:	
`lsyb4dmw.lib`	Sybase 4 DB-Library Support and Model Library
`lsdbdmw.lib`	Sybase 10 DB-Library Support and Model Library
`lsctdmw.lib`	Sybase 10 CT-Library Support and Model Library
UNIX:	
`libsyb4dm.a`	Sybase DB-Library Support and Model Library
`libsybcldm.a`	Sybase CT-Library Support and Model Library
VMS:	
`libsyb4dm.olb`	VMS: Sybase Support and Model Library

The link Directory

The `link` directory contains files to rebuild JAM development and runtime executables. Refer to Creating a New JAM Executable on page 40 for information on how to modify the makefile from this directory to accomplish this.

The files in the directory are:

File, Description

File	Description
`3dcheck.bmp`	Windows: Bitmap used in rebuilding JAM
`funclist.c`	Source file can be used as the foundation of your functions; it contains demo functions that show the calling sequence for C functions linked with `jam` and `jamdev`.
`jam.ico`	Windows: JAM's icon
`jmain.c`	Source for the `jam` main routine. It is provided to allow you to modify it. Notice the calls to `sm_jinitcrt()` and `sm_jresetcrt()`, which handle all startup and shutdown processing. These two routines must not be modified or removed!
`jxmain.c`	Source for the `jamdev` main routine. If you are creating a version of `jamdev` that includes your own functions, you should read through and understand this source file.
`make.com`	VMS: A DCL file which takes makefile. as its input to perform a make.
`makefile`	This file is easily modified to compile your code and link it into new versions of `jam` and `jamdev` .
`memmove.c`	Windows: Misc replacement source routine
`mwjam.def`, `mwjxform.def`	Windows: Linking definition files
`mwjam.rc`, `mwjxform.rc`	Windows: Resource definition files
`piinit.c`	JAM GUI interface initialization code for `jam`.
`pijxinit.c`	JAM GUI interface initialization code for `jamdev` (`jam` + screen editor).
`smdbinit.c`	Initialization code for the debugger.
`stdalloc.c`	Replacement source module for some C RTL routines
`system.c`	File specific to OS (`msdossys.c`, `posix.c`, `vms.c`, etc.)
`xfunclst.c` or `xfunclist.c`	Used to configure editor support functions

The notes Directory

The `notes` directory may contain documents, appendices, and notes on JAM installation, configuration, and development. This information is specific to your system.

File	Contents
`ver*`	describe the versions of the various components that make up the JAM distribution
`readme.*`	additional documentation for the components
`fix*.*`	contains descriptions of bugs that have been fixed

The patch Directory (not always present)

After a version of JAM has been released, JYACC may release a patch to it to correct problems. Patches are released in patch levels; each patch level contains one or more patches and any previous patch levels.

The `patch` directory contains one or more subdirectories, each named for a version of the product (e.g., `jam700`). Those directories in turn have further subdirectories, usually named `level1`, `level2`, etc., which contain the patches. Each of those directories contains a `readme` file which describes the problem(s) the patch level addresses and how to incorporate the patch into JAM.

The samples Directory

The `samples` directory contains three sets of sample programs:

- `tutorial` — Contains the source code that goes with the documentation's tutorial.
- `videobiz` — A complete JAM application that expands upon the tutorial.
- `sm5api` — Now obsolete, but is provided to help in supporting legacy JAM 5 applications.

The directory also contains some sample source code (`*.jpl`, `*.c`) for use with JAM.

The util Directory

The `util` directory holds all JAM executables (and DLLs, under Windows). Under VMS, due to the restriction that DCL verbs be unique to four characters, some utilities have shortened DCL verb names. The name displayed in parentheses after the full executable name is the shortened DCL verb name.

The utilities (which have the `.exe` extension in Windows and VMS) in this directory are:

Utility	Description
`bin2c` (`b2c`)	Converts binary files to a form suitable for inclusion in C programs.
`bin2hex` (`b2h`)	Converts binary files to and from a hexadecimal ASCII representation. Useful when porting screens and other binary data to other systems.
`binh`	Windows front end to `binherit`.
`binherit`	Batch mode inheritance program.
`cmap2bin`	Converts configuration map files (usually `*cmap`) to binary versions.
`dd2rec`	Converts JAM 5 data dictionary records to a format usable with `sm5api` samples.
`dd5to6`	Converts data dictionaries from release 5 format to release 6.
`dtext`, `dtextrw.exe`	JAM's help engine. You can run this to open the manuals outside of `jamdev`. `dtextrw.exe` is the name under Windows; others use `dtext`.
`f2asc`, `f2ascl` (Windows)	Converts a binary JAM screen file to an ASCII listing of the screen's contents and edits, and vice versa. `f2ascl` is a Windows front end that prompts for parameters.
`f5to6`	Converts forms from JAM 5 format to JAM 6 format.
`formlib`	Maintains libraries of screens, JPL procedures, etc.
`gdsp`	Business graphics support program (UNIX & VMS)
`jam`	The runtime (and distributable) version of JAM tool.
`jamdev`, `jamdev.sym` (Windows)	The main JAM 7 development utility with authoring, debugger, and JDB linked in. A `.sym` file is available from product support and is useful in catching and reporting GPFs, should they occur, but is not included in the standard distribution.

Utility	Description
jisql	JDB ISQL Tool
jiutil	UNIX & VMS: Internal Support routine for jisql.
jpl2bin	Converts JPL code to binary format.
jyaccd	UNIX: Part of License Manager system.
key2bin	Converts text key files (*keys) into binary key files (*keys.bin).
lmchksum, lmdown, lmgrd, lmhostid, lmremove, lmreread, lmstat, lmswitchr	UNIX: Part of License Manager system.
m2asc	Converts ASCII menu definitions to and from binary format.
mbedit	Menu bar editor.
mkinit	Utility for building JAM database driver initialization files.
msg2bin (m2bin)	Converts text message files into binary message files.
msg2hdr (m2hdr)	Converts text message files into C header files.
s2asc	Style to/from ASCII conversion program.
showkey	Utility to show raw key sequences generated by a terminal.
swsdrvr	Business graphics support program (UNIX and VMS).
term2vid	UNIX: Converts termcap and terminfo entries into videofiles.
var2bin	Converts configuration files in text format (e.g., smvars and smsetup) to the binary format used by JAM applications.
vid2bin	Converts text video files into binary video files.
what	Utility to show versions of source files used to build executables.
xmjxhelp	UNIX: Help engine for JYACC's online documentation. Not directly runnable.
*.pif	Windows PIF files for DOS based utilities.

Utility	Description
`*.ico`	Icons for DOS-based utilities.
other files	Utilities that are part of help system, etc. (See below for information on DLLs)

On Windows, the directory will also contain some or all of the following DLLs:

DLL	Supplied with	Compatible Database Version
`cktbl16.dll`	JAM	
`isqli501.dll`	Informix Client	5.0
`jamres.dll`	JAM	
`ldllsqlw.dll`	Informix Client	4.10
`lemb6dmw.dll`	JAM/Oracle Driver	6
`li501dmw.dll`	JAM/Informix Driver	5.01WF1 and later
`libsti.dll`	JAM	
`linf4dmw.dll`	JAM/Informix Driver	4.10
`linf5dmw.dll`	JAM/Informix Driver	5.01WE1 and earlier
`lodbdmw.dll`	JAM/ODBC Driver	1.0 and 2.0
`lora6dmw.dll`	JAM/Oracle Driver	6.0
`lora7dmw.dll`	JAM/Oracle Driver	7.0
`lsctdmw.dll`	JAM/Sybase Driver	10.0
`lsdbdmw.dll`	JAM/Sybase Driver	10.0
`lsyb4dmw.dll`	JAM/Sybase Driver	4.2
`odbc.dll`	ODBC Client	1.0 and 2.0
`ora6win.dll`	Oracle Client	6.0
`ora7win.dll`	Oracle Client	7.0
`sql13win.dll`	Oracle Client	6.0
`sql15win.dll`	Oracle Client	7.0
`tminf1w.dll`	JAM/Informix Driver	
`tmodb1w.dll`	JAM/ODBC Driver	
`tmora1w.dll`	JAM/Oracle	
`tmsyb1w.dll`	JAM/Sybase Driver	

DLL	Supplied with	Compatible Database Version
`w3dblib.dll`	Sybase Client	4.2
`wcslib.dll`	Sybase Client	10.0
`wctlib.dll`	Sybase Client	10.0
`wecjlib.dll`	JAM	
`wsybdb.dll`	Sybase Client	10.0

SECTION THREE

License Manager Installation

3

Introduction

The License Manager Installation section of *Read Me First*:

- Describes the licensing options available for JAM.
- Explains how to customize and install the license file that activates your JAM token.
- Provides explanations of licensing-related error messages that can appear when starting or running JAM.

In addition, several license management utilities are provided with the product; these are also described.

OpenVMS Throughout this section there are places where instructions for OpenVMS users diverge from the directions for installing on a UNIX system. Wherever such passages occur, they are demarcated with "*OpenVMS*" in the left margin. If, to avoid ambiguity, it is necessary to indicate a passage explicitly for UNIX users, such passages will be similarly demarcated with "*UNIX*".

License Tokens

The licensing options available for JAM apply only to the product used as a development tool. End-user software developed with JAM is not restricted by the license manager.

Network Tokens

***Note:** To take advantage of network licensing on UNIX, a TCP/IP network is required.*

OpenVMS DECnet is required for OpenVMS users.

A network license token specifies a maximum number of simultaneous users of the software on a network. A network token allows JAM to be run on any number of network nodes, as long as the total number of copies of JAM in use at any time does not exceed the token limit.

Suppose, for example, a customer has six developers using separate workstations on the same network. They do not all use JAM full-time, but each does require access to the software at various times from his/her own workstation. Suppose, also, that no more than three of these developers are likely to be using JAM at a given time. Rather than purchase six CPU-locked tokens, one for each machine, this customer could purchase three network tokens; permission to use JAM would “float” on the network, allowing any three users concurrent access to the software.

A *user* in this context represents a single copy of JAM running on the network. Thus, if one individual has two processes running JAM simultaneously, the license manager counts two users, even though only one individual or one workstation is involved. In the context of network licenses, the term *token* is used interchangeably with *number of running incidences of the jamdev executable*. A token represents permission to use one copy of JAM.

Customers who plan to run JAM on network hosts of binary incompatible types must purchase a binary copy for each such host type; if JAM will be run on hosts that are all of the same type, only one copy of the JAM binary for that host type need be purchased.

Password-Managed Licensing

A password is simply a character string into which all relevant information regarding the license(s) you have purchased is encoded. Based on the password, the license management software, embedded in JAM, can determine whether or not a request to use the software should be honored. For example, will running an additional copy of JAM exceed the maximum number of users specified for a network token?

JYACC computes the required password based on the type of license you have purchased and on your system configuration. When you obtain your password, as described on page 72, you must enter it into the license file as part of the JAM installation process. JAM will run only after the password and all other required information have been properly entered into the file.

In most circumstances, JAM license management is transparent to the developer. End users are not affected at all by the licensing system.

Pre-existing Users of CPU-Locked Licensing

Note: *The following information on CPU (or node) -locked tokens is presented for existing JAM customers who use this licensing scheme, which is not available to new JAM 7 customers.*

CPU-Locked Tokens

A CPU-specific token is issued for JAM running on a single host. Each simultaneous usage of the JAM binary on the host requires its own CPU-locked token. Separate tokens are purchased for each host (node) on which JAM will run.

In any configuration, each CPU-locked token can be administered on a stand-alone basis by the licensed host running JAM. Alternatively, for licensed hosts on a network, any or all CPU-locked tokens can be administered centrally by license server(s) (one or three) on the network.

OpenVMS

Only one server is permitted for OpenVMS users.

4

Installing the License Manager

Password-managed licensing for JAM uses the Flexible License Manager (FLEX*lm*), a product of GlobeTrotter Software, Inc.

The instructions in this chapter show you how to create the appropriate license file. If you are running other software products that use FLEX*lm* for license management, you can add the applicable entries to the existing license file rather than create a new license file on the server. It is important that you use the latest version of the license manager daemon (`lmgrd`), supplied by JYACC or by the vendor using FLEX*lm*. You can determine the version by running (under UNIX):

```
strings lmgrd | grep FLEX
```

OpenVMS

OpenVMS licensing does not use `lmgrd`; clients connect directly with the JYACC daemon `jyaccd`.

License Servers

If you have a network license, you can designate one or three hosts as license servers. In general, one license server is adequate unless the network or servers go down frequently. If this is the case, you can designate three servers so that you can still run JAM even if one is down.

If you have multiple license servers, each server can continue to dispense tokens (up to the limit available) as long as it is able to communicate with a quorum of the servers. A quorum of servers is the smallest (whole) number that is greater than half the number of servers. For three servers, two is a quorum. JAM developers who are connected through a license server that has gone down will automatically be connected to one of the remaining servers, if possible. Note that if developers lose the connection to the license server and are not automatically reconnected, they will always be able to save their work.

OpenVMS For OpenVMS users, only one license server is permitted.

License File

The license file, by default, is named `license.dat` and is usually located in the `config` directory of the JAM installation (`$SMBASE/config`). `jamdev` and other development utilities will also search for the license file at `/usr/local/flexlm/licenses` and in the directories specified for `SMPATH`. Also, license manager utilities can optionally specify the location of the license file with the `–c` option.

Specifying the File Location

If you choose to locate the license file in a directory other than `config` or if you are adding the JAM entries to an existing FLEX*lm* file, set the environment variable `LM_LICENSE_FILE` to the full path of the file (including the filename).

OpenVMS After running `SMBASE:[UTIL]LMHOSTID`, copy the OpenVMS template JAM license file `SMBASE:[CONFIG]LICENSE.VMS` to `SMBASE:[CONFIG]LICENSE.DAT`. (After you have filled out the password request form and faxed it to the JYACC password desk, you will have the information necessary to edit this file and enable licensing.)

The default license file on OpenVMS is `SYS$COMMON:[SYSMGR]flexlm.dat`. You can override this setting with either the `–c` command-line option, or using the logical name `LM_LICENSE_FILE`. It is recommended that you store the license file in `SMBASE:[CONFIG]LICENSE.DAT`. Setting the logical `LM_LICENSE_FILE` as follows will allow this:

```
$ define LM_LICENSE_FILE SMBASE:[CONFIG]LICENSE.DAT
```

Sample License Files

Sample license files are provided in `$SMBASE/config`. They are:

- `net1.dat` — Network JAM license with a single server.
- `net3.dat` — Network JAM license with three servers.

The files are examples, commented to assist you in creating the appropriate entries for your configuration. You will receive complete license files from the password desk. These files can be used as-is or incorporated into your existing license file from other vendors. The formats are explained in more detail below.

OpenVMS

The sample license file for OpenVMS users is found at `SMBASE:[CONFIG]LICENSE.VMS`.

License File Description

License files require a `SERVER` line for each server plus a `DAEMON` line, a `FEATURE` line, and a `FEATURESET` line. The formats of these lines are shown below. If you are an existing JAM customer using the CPU-locked scheme, refer to page 74 for information on the format of the `FEATURE` line.

Host ID

The `SERVER` line and the `FEATURE` line require a *host ID*. You can obtain the host ID in the following manner:

UNIX

Run the `lmhostid` utility provided in `$SMBASE/util`. The output displays the host type followed by the host ID. Host ID is not case-sensitive.

OpenVMS

For OpenVMS users, use the ethernet address as the host ID. You can obtain the ethernet address via `ncp`, or, preferably from the `lmhostid` utility found in `SMBASE:[util]`:

```
$ run smbase:[util]lmhostid
```

or

```
$ mcr ncp
NCP>show known lines
Known Line Volatile Summary as of 6–NOV–1989 14:41:48
Line       State
SVA–0        on
```

(Take the line from above, and enter in the next command)

```
NCP>show line SVA–0 char
```

(This command will list the ethernet address as “`Hardware address`”.)

SERVER line

`SERVER` *sname hostid port*

where:

hostid is the host ID of the server, as described above.

port is the TCP/IP port number used by this server; the default is `1700` for this parameter, but you can use any unused port number.

sname is the server's name; if you do not know this, run the UNIX `uname -n` command; enter the server name exactly as it is displayed. This parameter is case-sensitive.

OpenVMS

On OpenVMS, the server node name can be entered with the trailing "`::`", if desired, as is the practice of many OpenVMS system administrators.

DAEMON line

`DAEMON jyaccd` *path*

where:

path is the full path for the daemon; normally this is `$SMBASE/util/jyaccd`

OpenVMS

For OpenVMS, the `DAEMON` line has an extra field: the DECnet object number. The syntax for a `DAEMON` line is:

`DAEMON jyaccd` *path object* `[`*options_file*`]`

where:

path is the full file specification of the daemon's `.exe` file (i.e., `smbase:[util]jyaccd.exe`).

object is the DECnet object number:

- 0: daemon uses object number 0, task name is `jyaccd`.
- 128–255: daemon uses this object number (200 is recommended).

options_file is the (optional) daemon option file location.

For example:

```
DAEMON jyaccd smbase:[util]jyaccd.exe 200

DAEMON jyaccd smbase:[util]jyaccd.exe 200
smbase:[config]jyaccd.opts
```

FEATURE line

The format of a `FEATURE` line for a JAM network token is as follows:

`FEATURE` *product* `jyaccd 7.0` *dd–mmm–yyyy nn password* `“”`

where:

product is the product name, one of `jam`, `rw`, `case_twk` or `jamtpi-tux-server`, etc.

dd–mmm–yyyy is the expiration date of the token or `01–jan–0000` if the token has no expiration date. The date must be entered in the format shown; the month is not case-sensitive.

`“”` serves as a placeholder for the *hosttype* parameter which is not applicable for network tokens.

nn is the number of users (tokens) allowed under this license.

password is the password you obtain from the JYACC password desk; not case-sensitive.

Note: *The quotation marks act as a placeholder for the hosttype, which is not applicable for network tokens. However, these empty quotation marks (no space between them) are required. Also, there must be a space between the password.and the quotation marks.*

FEATURESET line

`FEATURESET jyaccd` *featureset–code*

featureset–code is the code you obtain from the JYACC password desk; it is not case-sensitive. This code encrypts the codes of all features this daemon supports, so no `FEATURE` lines can be added or removed in the license file.

If you have an existing FLEX*lm* license file, add only the `FEATURE`, `DAEMON` and `FEATURESET` lines to the file. If you are already using FLEX*lm*, the `SERVER` lines, supplied by JYACC, must be equivalent to the `SERVER` lines already present.

OpenVMS

The `FEATURESET` line is not used for OpenVMS users.

Starting the Daemon

Regardless of licensing scheme used, UNIX users must use the latest available version of the license daemon (`lmgrd`) in order to run JAM. Instructions for OpenVMS users are described below. (The version of the license daemon can be determined by typing `strings lmgrd | grep –i flexlm`.)

To start the daemon, enter the command:

`lmgrd –c` *path* `>` *log* `&`

where:

path is the full path for the license file.

log is the name of a log file to which the output is redirected.

Typically, the command will appear as:

```
$SMBASE/util/lmgrd –c $SMBASE/config/license.dat >
 license.log &
```

It is recommended that you add this command to the system startup scripts file (for Sun systems: `/etc/rc.local`) so that the daemon will run automatically when the system is rebooted.

If you have other software running under FLEX*lm*, you can stop and restart the license daemon or run `lmreread` before you will be able to run JAM.

OpenVMS

OpenVMS licensing does not use `lmgrd`; clients connect directly with the `jyaccd` daemon. It is recommended that the license daemon be started in the system startup command file.

To start the daemon on OpenVMS, enter the command:

`jyaccd [–c` *license_file_path*`] [–l` *log_file_path*`]`

where:

license_file_path is an alternate license file, if the logical `LM_LICENSE_FILE` is not set or if you want to override the setting of `LM_LICENSE_FILE`.

log_file_path is an optional log file name. If this option is not selected, the logging information will be written to the screen.

Note: *On OpenVMS, the* `SYSNAM` *privilege is required to run the* `jyaccd.exe` *daemon.*

Obtaining Your Password

In most cases, your license password is not shipped with your copy of JAM. Password request forms are located at page 109 in Appendix C. Fill out the form that applies to the type of license you have purchased, and fax the completed form to the JYACC password desk at (212) 608-6753, email the form to JYACC at *license@jyacc.com*, or mail it to JYACC, Inc., Attn.: Password Desk, 116 John Street, New York, NY 10038.

To obtain the password for your license, you must install the products and supply the following information about your configuration:

- Serial number of the JAM binary shipped to you (on the media).
- JAM version you have purchased (on the media).
- Server IDs (run `lmhostid`).

If you have purchased more than one license, you must fill out and submit a separate request form for each password. Photocopies of the request form are acceptable.

To ensure that your password is generated correctly and is accurately communicated to you, JYACC recommends that you fax or email your password request.

Installing Your License: OpenVMS

These are the specific directions for installing your license on OpenVMS.

1. Fully install JAM as described on page 30, up to and including the section on running `SETUP.COM` (which sets the logical `SMBASE`).
2. Run `SMBASE:[UTIL]LMHOSTID`.
3. Fill out the license request form found at the end of this document.
4. Fax the form to the JYACC password desk (refer to the form for details).
5. Copy `SMBASE:[CONFIG]LICENSE.VMS` to `SMBASE:[CONFIG]LICENSE.DAT`.
6. Edit `SMBASE:[CONFIG]LICENSE.DAT` using the information faxed to you by the password desk.
7. Run `@INSTALL.COM`. This installs `LICSHR.EXE` as a shared image and defines `LM_LICENSE_FILE`.

You are now licensed to run `jamdev.exe`. If you experience problems starting up `jamdev`, you may need to run the DCL file `@smBASE[UTIL]RELINK.COM`. Refer to page 102 for more details.

Updating Your License

Any change in your configuration or in the terms of your license that affects data entered into the license file requires that you obtain a new password. Call the JYACC password desk for your new password.

Note: *The host ID, for example, can change as a result of hardware maintenance. This does not change the terms of your license, but it does require that you obtain a new password and update your license file.*

Changes that affect the terms of your license are:

- Expiration date
- Number of users (for network tokens and CPU-locked tokens)
- Host type (for CPU-locked tokens)
- Host id (for CPU-locked tokens)

Note: *CPU-locked tokens are no longer sold. If you are an existing JAM customer using this licensing scheme, refer to the following section for information specific to CPU-locked licensing.*

Contact your JYACC sales representative to make any of theses changes. The password desk is not able to issue passwords for these changes without instruction from the sales department.

Installation for CPU-Locked Licenses

Note: *The following information on CPU (or node) -locked tokens is presented for existing JAM customers who use this licensing scheme, which is not available to new JAM customers.*

If you are a user of the CPU-locked licensing scheme, you still need to run the license manager daemon `lmgrd` in order to run JAM.

License File FEATURE Line

CPU-locked licenses require a `FEATURE` line in the license file. The format of the `FEATURE` line is:

`FEATURE` *product* `jyaccd 7.0` *dd–mmm–yyyy nn password "hosttype" hostid*

where:

product is the product name, one of `jam`, `rw`, `case_twk` or `jamtpi-tux-client`, etc.

dd–mmm–yyyy is the expiration date of the token or `01–jan–0000` if the token has no expiration date. The date must be entered in the format shown; the month is not case-sensitive.

nn is the number of CPU-locked tokens.

password is the password you obtain from the JYACC password desk; not case-sensitive.

Note: *A separate password is issued for each machine with CPU-locked tokens, whether the tokens are administered on a stand-alone basis or centrally over a network.*

hosttype is the processor type of the licensed host; this parameter must be within double quotes, as shown. To determine the host type, run the `lmhostid` utility provided in `$SMBASE/util`. The output displays both the host type and the host ID.

hostid is the ID of the licensed host as described above.

5

License-Related Error Messages

In most circumstances, JAM license management is transparent to the developer. This chapter lists and describes the error messages that can be generated from JAM and from the underlying license manager software, FLEX*lm*.

JAM Error Messages

Startup Messages

The messages described in this section can appear when `jamdev` is started.

All License Types

`JAM License Manager: Problem with license file for jam.`

Action: Notify your system administrator.

`JAM License Manager: Cannot communicate with license server.`

Cause: There is a problem communicating with the `lmgrd` daemon.
Action: Notify your system administrator.

Network Licenses

`JAM License Manager: All jam licenses are currently in use.`

Cause: All licenses are in use. This message is followed by a list of all JAM users currently on the network.
Action: Wait until a license becomes available (when one of these users terminates `jamdev`). Consider purchasing more licenses.

Note: *The following JAM startup error information are specific to CPU (or node) -locked tokens and is presented for those JAM customers who use this licensing scheme. CPU-locked licenses are not available to new JAM customers.*

CPU-Locked Licenses

```
JAM License Manager: jam not authorized for this computer.
```

Cause: JAM is not licensed to run on this node.
Action: Notify your system administrator.

`JAM License Manager: jam not authorized for host type` *xxxxxx*`.`

Cause: JAM is licensed to run on this node, but this is not the host type expected.
Action: Notify your system administrator.

Runtime Message

The following message can appear when you exit the screen editor and attempt to continue processing the current screen or begin processing another screen. The message applies only to network licenses.

```
JAM License Manager: Cannot communicate with license server
or your jam license was obtained by another user when the
license server was restarted.
```

Cause: The license server or the connection to the license server has gone down. The message indicates one of two situations:

– The license server or the connection between this host and the server is currently down.

– The server or connection was down and has been restarted; within the first minute after restart, another user started `jamdev` and received the last available token. This is a very rare situation: Since `jamdev` checks in with the license server at regular intervals, there is only a brief period after reestablishment of the connection when the server would not know that a particular host was using one or more of the licenses. This means someone has started up `jamdev` during that brief interval, and all other licenses on the network are in use.

Action: You can save the form you are currently editing but you cannot return to editing that or any other form. Notify your system administrator.

FLEX*lm* Messages

The following messages are generated by FLEX*lm*, the underlying license management software, and is provided courtesy of the *FLEXlm End User Manual*.

Informational Messages

`Connected to` *node*

Cause: This daemon is connected to its peer on *node*.

`CONNECTED, master is` *name*

Cause: The license daemon logs this message when a quorum is achieved and everyone has selected a master.

`DENIED:` *N feature* `to` *user* `(`*mm/dd/yy hh:mm*`)`

Cause: *user* was denied access to *N* licenses of *feature*.

`EXITING DUE TO SIGNAL` *nnn*

`EXITING WITH CODE` *nnn*

Cause: An interrupt signal has been intercepted. All daemons list the reason that the daemon has exited.

`EXPIRED:` *feature*

Cause: *feature* has passed its expiration date.

`IN:` *feature* `by` *user* `(`*N* `licenses) (used:` *d:hh:mm:ss*`)`

Cause: *user* at *d:hh:mm:ss*.

`IN server died:` *feature* `by` *user* `(`*N* `licenses) (used:` *d:hh:mm:ss*`)`

Cause: *user* has checked in *N* licenses of *feature* by virtue of the fact that his server died.

`License Manager server started`

Cause: The license daemon has been started.

`Lost connection to` *host*

Cause: A daemon can no longer communicate with its peer on node *host*, which can cause the clients to have to reconnect, or cause the number of daemons to go below the minimum number, in which case clients may start exiting. If the license daemons lose the connection to the master, they will kill all the vendor daemons; vendor daemons will shut themselves down.

`Lost quorum`

Cause: There are not enough servers to satisfy the quorum number. The daemon will process only connection requests from other daemons.
Action: Check network connection between servers.

`MASTER SERVER died due to signal` *nnn*

Cause: The license daemon received fatal signal *nnn*.

`MULTIPLE` *xxx* `servers running.`
`Please kill, and restart license daemon`

Cause: The license daemon has detected multiple copies of vendor daemon *xxx* are running.
Action: Kill all *xxx* daemon processes and re-start the license daemon.

`OUT: feature` *feature* `by` *user* `(`*N* `licenses) (used:` *d:hh:mm:ss*`)`

Cause: *user* has checked out *N* licenses of *feature* at *d:hh:mm:ss*.

`Removing clients of children`

Cause: The top-level daemon logs this message when one of the child daemons dies.

`RESERVE` *feature* `for HOST` *name*

`RESERVE` *feature* `for USER` *name*

Cause: A license of *feature* is reserved for either user *name* or host *name*.
Action: None.

`Restarted` *xxx* `(internet port` *nnn*`)`

Cause: Vendor daemon *xxx* was restarted at internet port *nnn*.

`Retrying socket bind (address in use)`

Cause: The license server try to bind their sockets for approximately 6 minutes if they detect `address in use` errors.
Action: Check for multiple `lmgrd`s using the same TCP port number.

`Selected (EXISTING) master` *node*

Cause: This license daemon has selected an existing master (*node*) as the master.

`SERVER shutdown requested`

Cause: A daemon received a request to shut down from a user-generated kill command.

`[NEW] Server started for:` *feature-list*

Cause: A (possibly new) server was started for the features listed.

`Shutting down` *xxx*

Cause: The license daemon is shutting down the vendor daemon *xxx*.

`SIGCHLD received. Killing child servers.`

Cause: A vendor daemon logs this message when a shutdown was requested by the license daemon.

`Started` *name*

Cause: The license daemon logs this message whenever it starts a new vendor daemon.

`Trying connection to` *node*

Cause: The daemon is attempting a connection to *node*.

Configuration Problem Messages

hostname`: Not a valid server host, exiting`

Cause: This daemon was run on an invalid hostname.
Action: Run `lmgrd` on the host(s) specified in the `SERVER` lines.

hostname`: Wrong hostid, exiting`

Cause: The hostid is wrong for *hostname*.
Action: Check the license file and ensure the hostnames match the hostids.

`BAD CODE for` *feature-name*

Cause: The specified feature name has a bad encryption code.
Action: Check the password received from JYACC.

`CANNOT OPEN options file` *file*

Cause: The options file specified in the license file could not be opened.
Action: Check the path for the options file on the `DAEMON` line in the license file.

`Couldn’t find a master`

Cause: The daemons could not agree on a master.
Action: Kill and then restart `lmgrd` on servers.

`Feature does not yet exist`

Cause: Machine date is prior to start date of license.
Action: Correct the machine time.

`license daemon: lost all connections`

Cause: This message is logged when all the connections to a server are lost, which often indicates a network problem.
Action: Check the network and restart the daemons.

`lost lock, exiting`

Cause: Error closing lock file.

`Unable to re-open lock file`

Cause: The vendor daemon has a problem with its lock file, usually because of an attempt to run more than one copy of the daemon on a single node.

Action: Locate the other daemon that is running via a `ps` command, and kill it with `kill -9`.

`NO DAEMON line for` *daemon*

Cause: The license file does not contain a `DAEMON` line for *daemon*.

Action: Add `DAEMON` line for *daemon* in the license file.

`No license data for` *feature*`, feature unsupported`

Cause: There is no feature line for *feature* in the license file.

Action: Edit the license file.

`No features to serve!`

Cause: A vendor daemon found no features to serve. This could be caused by bad data in the license file.

Action: Inspect the license file for bad data.

`UNSUPPORTED FEATURE request:` *feature* `by` *user*

Cause: *user* has requested a feature that this vendor daemon does not support. This can happen for a number of reasons: the license file is bad, the feature has expired, or the daemon is accessing the wrong license file.

`Unknown host:` *hostname*

Cause: The *hostname* specified on a `SERVER` line in the license file does not exist in your host's database.

Action: Check with your system administrator for the correct hostname. FLEX*lm* uses standard network services to find the host: Domain Name Server (DNS), Network Information Services (NIS or YP) or in `/etc/hosts`.

`lm_server: lost all connections`

Cause: This message is logged whenever all the connections to a server are lost. This probably indicates a network problem.

`NO DAEMON lines, exiting`

Cause: The license daemon logs this message if there are no `DAEMON` lines in the license file. Since there are no vendor daemons to start, there is nothing to do.

Action: Edit license file.

`NO DAEMON line for` *name*

Cause: A vendor daemon logs this error if it cannot find its own `DAEMON` name in the license file.

Action: Edit license file.

Daemon Software Error Messages

`ATTEMPT TO START VENDOR DAEMON` *xxx* `with NO MASTER`

Cause: A vendor daemon was started with no master selected. This is an internal consistency error in the daemons.
Action: Report error to JYACC technical support.

`BAD PID message from nnn:` *xxx* `(`*msg*`)`

Cause: A top-level vendor daemon received an invalid PID message from one of its children (daemon number *xxx*).

`BAD SCONNECT message: (`*message*`)`

Cause: An invalid `server connect` message was received.

`Cannot create pipes for server communication`

Cause: The `pipe` system call failed.
Action: Report error to JYACC technical support.

`Can’t allocate server table space`

Cause: A malloc error.
Action: Check swap space

`Connection to` *node* `TIMED OUT`

Cause: The daemon could not connect to *node*.
Action: Check the network.

`Error sending PID to master server`

Cause: The vendor server could not send its PID to the top-level server in the hierarchy.
Action: Report error to JYACC technical support.

`f-do-notify called with no valid feature`

Cause: This is an internal inconsistency error.
Action: Report error to JYACC technical support.

`Illegal connection request to` *DAEMON*

Cause: A connection request was made to *DAEMON*, but this vendor daemon is not *DAEMON*.
Action: Report error to JYACC technical support.

`Illegal server connection request`

Cause: A connection request came in from another server without a `DAEMON` name.
Action: Report error to JYACC technical support.

`KILL of child failed, errno =` *`nnn`*

Cause: A daemon could not kill its child.
Action: Get PID of daemon and kill with `kill –9`.

`No internet port number specified`

Cause: A vendor daemon was started without an internet port.
Action: Specify an Internet port on the `SERVER` line.

`Not enough descriptors to re-create pipes`

Cause: The top-level daemon detected the death of one of its sub-daemons. In trying to restart the chain of sub-daemons, it was unable to get the file descriptors to set up the pipes to communicate.
Action: This is a fatal error. The daemons must be restarted.

`read: error message`

Cause: An error in a `read` system call was detected.

`recycle_control BUT WE DIDN'T HAVE CONTROL`

Cause: The hierarchy of vendor daemons has become confused over who holds the control token. This is an internal error.

`return_reserved: can't find feature listhead`

Cause: When a daemon is returning a reservation to the free reservation list, it could not find the listhead of features.

`select: message`

Cause: An error in a `select` system call was detected.
Action: Report error to JYACC technical support.

`Server exiting`

Cause: The server is exiting. This is normally due to an error.
Action: Report error to JYACC technical support.

`SHELLO for wrong DAEMON`

Cause: This vendor daemon was sent a `server hello` message that was destined for a different `DAEMON`.

`Unsolicited msg from parent!`

Cause: Normally, the top-level vendor daemon sends no unsolicited messages. If one arrives, this message is logged. This is a bug.
Action: Report error to JYACC technical support.

`WARNING: CORRUPTED options list (o–>next == 0)`
`Options list TERMINATED at bad entry`

Cause: An internal inconsistency was detected in the daemon's option list.
Action: Report error to JYACC technical support.

6

Network License Administration

The following license management utilities are provided in `$SMBASE/util` and described in this chapter:

- ❍ `jyaccd` — License daemon. On UNIX, started by `lmgrd`; on OpenVMS, started instead of `lmgrd`.
- ❍ `lmdown` — Allows for the graceful shutdown of all license daemons on all nodes.
- ❍ `lmgrd` — Start the license manager daemon.
- ❍ `lmremove` — Allows the license administrator to remove a single user's license for a specified feature (useful if the node running the software has crashed).
- ❍ `lmreread` —Causes the license daemon to reread the license file and start any new vendor daemons that have been added.
- ❍ `lmstat` —Allows the system administrator to monitor license management operations including:
  - Which daemons are running.
  - Users of individual features.

- Users of features served by a specific daemon.

Refer to page 92 for a description of the license daemon options file, which allows the system administrator to customize JAM license management.

The following typographical conventions are used throughout this chapter:

`literal`
This font is used for text that should be typed verbatim, such as examples, filenames and directory names.

italics
Italics show where screen, file, and variable names should appear. Replace these with the appropriate names in your applications.

[x] **or** `[x]`
In this notation, the brackets indicate that *x* is an optional element. Do not type the brackets.

`{KEYWORD1|KEYWORD2|KEYWORD3}`
Curly braces indicate that only one of the keywords shown should be entered; the choices are separated by vertical bars. Do not type the braces or the bars.

License Administration Utilities

This section describes FLEX*lm* license administration utilities provided with JAM.

lmdown

Graceful shutdown of all license daemons

`lmdown [–c` *license_file*`] [–q]`

`–c` *license_file*	Use the specified license_file. If this switch is not specified, `lmdown` looks for the environment variable `LM_LICENSE_FILE`. If the environment variable is not set, `lmdown` looks for the file `/usr/local/flexlm/licenses/license.dat`.
`–q`	Quiet mode. If this switch is not specified, `lmdown` asks for confirmation before asking the license daemons to shut down. If this switch is specified, `lmdown` does not ask for confirmation.

Description

`lmdown` sends a message to every license daemon asking it to shut down. The license daemons write out their last messages to the log file, close the file, and exit. All licenses which have been given out by those daemons will be rescinded, so that the next time a client program goes to verify his license, it will not be valid.

The end-user system administrator should protect the execution of `lmdown` since shutting down the servers will cause loss of licenses.

Note: *This command can be used only by a "FLEXlm administrator" (i.e., a member of group* `lmadmin` *or, if the* `lmadmin` *group does not exist, a member of group* `0`*).*

lmgrd

The flexible license manager daemon

`lmgrd [–2] [–b] [–c` *license_file*`] [–d] [–l` *logfile*`] [–p] [–s` *interval*`] [–t` *timeout*`]`

`–2`	Specifies V2 startup arguments, in contrast to the `–b` switch. Note that this switch is required if you intend to use the `–p` switch (available in `lmgrd` v2.4 and later).
`–b`	Specifies backward compatibility mode. Use this switch if you are running a v2.1 or later `lmgrd` with a v1.5 or earlier vendor daemon. This is the default switch in FLEX*lm* v2.4 and later.
`–c` *license_file*	Use the specified *license_file*. If this switch is not specified, `lmgrd` looks for the environment variable `LM_LICENSE_FILE`. If the environment variable is not set, `lmgrd` looks for the file `/usr/local/flexlm/licenses/license.dat`.
`–d`	Specifies that hostnames which are read from the license file should have the local domain name appended to them before sending to a client. Useful when clients are accessing licenses from another domain. (Available in `lmgrd` v2.4 and later.)
`–l` *logfile*	Specifies the output log file to use.
`–p`	Specifies that the `lmdown` and `lmremove` utilities can only be run by a license administrator. A license administrator is a member of the `lmadmin` group, or, if the `lmadmin` group does not exist, a member of group `0`. (This is available in `lmgrd` v2.4 and later.)
`–s` *interval*	Specifies the logfile timestamp interval, in minutes. The default is 360 minutes.
`–t` *timeout*	Specifies the timeout interval, in seconds, during which daemons must complete their connections to each other. The default value is 10 seconds. A larger value may be preferable if the daemons are being run on busy systems or a very heavily loaded network.

Environment

If no `–c` option is specified, `lmgrd` looks for the environment variable `LM_LICENSE_FILE` in order to find the license file to use. If that environment variable is not set, `lmgrd` looks for the file `/usr/local/flexlm/licenses/license.dat`.

Description

`lmgrd` is the main daemon program for the FLEX*lm* distributed license management system. When invoked, it looks for a license file containing all required information about vendors and features.

Note: *This utility does not apply to OpenVMS.*

lmremove

Remove specific licenses and return them to the license pool

`lmremove [–c` *license_file*`]` *feature user host display*

`–c` *license_file*	Use the specified license file. If this switch is not specified, `lmremove` looks for the environment variable `LM_LICENSE_FILE`. If that environment variable is not set, `lmremove` looks for the file `/usr/local/flexlm/licenses/license.dat`.

Description

`lmremove` allows the system administrator to remove a single user's license for a specified feature. This could be required in the case where the licensed user was running the software on a node that subsequently crashed. This situation will sometimes cause the license to remain unusable. `lmremove` will allow the license to return to the pool of available licenses.

`lmremove` removes all instances of *user* on node *host* on display *display* from usage of *feature*. The end-user system administrator should protect the execution of `lmremove` since removing a user's license can be disruptive.

Note: *This command can be used only by a "FLEXlm administrator" (i.e., a member of group* `lmadmin` *or, if the* `lmadmin` *group does not exist, a member of group* `0`*).*

lmreread

Tells the license daemon to reread the license file

`lmreread [–c` *license_file*`]`

`–c` *license_file*	Use the specified license file. If this switch is not specified, `lmreread` looks for the environment variable `LM_LICENSE_FILE`. If the environment variable is not set, `lmreread` looks for the file `/usr/local/flexlm/licenses/license.dat`.

Description

`lmreread` allows the system administrator to tell the license daemon to reread the license file. This can be useful if the data in the license file has changed; the new data can be loaded into the license daemon without shutting down and restarting it.

`lmreread` uses the license file from the command line (or the default file, if none specified) only to find the license daemon to send it the command to reread the license file. The license daemon will always reread the original file that it loaded. If you need to change the path to the license file read by the license daemon, then you must shut down the daemon and restart it with that new license file path.

You can not use `lmreread` if the SERVER node names or port numbers have been changed in the license file. In this case, you must shut down the daemon and restart it in order for those changes to take effect.

`lmreread` does not change any option information specified in an options file. If the new license file specifies a different options file, that information is ignored. If you need to reread the options file, you must shut down the daemon and restart it.

lmstat

Report status on license manager daemons and feature usage

`lmstat` [`–a`] [`–A`] [`–c` *license_file*] [`–f` *[feature]*] [`–l` *[reg_expression]*] [`–s` *[server]*] [`–S` *[daemon]*]
[`–t` *timeout*]

`–a`	Display everything.
`–A`	List all active licenses.
`–c` *license_file*	Use the specified license file. If this switch is not specified, `lmstat` looks for the environment variable `LM_LICENSE_FILE`. If the environment variable is not set, `lmstat` looks for the file `/usr/local/flexlm/licenses/license.dat`.
`–f` *[feature]*	List all users of the specified features.
`–l` *[reg_expression]*	List all users of the features matching the given regular expression.
`–s` *[server]*	Display the status of the specified server nodes.
`–S` *[daemon]*	List all users of the specified daemon's features.
`–t` *timeout*	Specifies the timeout interval, in seconds, during which daemons must complete their connections to each other. The default value is 10 seconds. A larger value may be desirable if the daemons are being run on busy systems or a very heavily loaded network.

Environment

If no `–c` option is specified, `lmgrd` looks for the environment variable `LM_LICENSE_FILE` in order to find the license file to use. If that environment variable is not set, `lmgrd` looks for the file `/usr/local/flexlm/licenses/license.dat`.

Description

`lmstat` provides information about the status of the server nodes, vendor daemons, vendor features, and users of each feature. Information can optionally be qualified by specific server nodes, vendor daemons, or features.

License Options File

This section describes the daemon options file, which allows the system administrator to customize JAM license management.

jyaccd.opt

Site administrator options file for FLEX*lm* licensed applications

```
/usr/local/flexlm/options/jyaccd.opt
```

Description

The `jyaccd.opt` file contains optional information supplied by the system administrator at the end-user site. This information can be used to tailor the behavior of the license daemons. The options file can contain the following information:

- reserved license information
- logfile control options
- license timeout control

Lines beginning with a pound sign (#) are ignored, and can be used as comments.

There is no default location or name for the options file; it is only active if it has been specified in the `license.dat` file as the fourth argument on the `DAEMON` line. Note that if there are multiple `DAEMON` lines in the `license.dat` file, then there can be multiple options files, one for each `DAEMON` line. Not all of the lines in an options file refer to a feature, so the site administrator *must* set up separate options files in order to use the `NOLOG` and `REPORTLOG` features.

Each line in the options file starts with a keyword which identifies the information on that line. The keyword is one of `RESERVE`, `NOLOG`, `GROUP`, `INCLUDE`, `EXCLUDE`, `TIMEOUT`, `LINGER`, or `REPORTLOG`. Their descriptions follow:

RESERVE

`RESERVE` *numlic feature* `{USER|HOST|DISPLAY|GROUP}` *name*

The `RESERVE` command reserves the specified number of licenses for the specified user, host, display, or group. Note that reserving a license decreases the number of generally available licenses.

NOLOG

`NOLOG {IN|OUT|DENIED|QUEUED}`

`NOLOG` causes messages of the specified type to be filtered out of the daemon's log file. Specifying a `NOLOG` option reduces the amount of output to the log file, which can be useful in those cases where the log file grows too quickly.

GROUP

`GROUP` *group–name member–list*

The `GROUP` command is used to define collections of users, which can then be used in `RESERVE`, `INCLUDE`, or `EXCLUDE` commands.

INCLUDE/EXCLUDE

`{INCLUDE|EXCLUDE}` *feature* `{USER|HOST|DISPLAY|GROUP}` *name*

`INCLUDE` and `EXCLUDE` are used to specify which users (or hosts, displays, or groups) are allowed to use a particular feature. Any user who is `EXCLUDE`d from a feature will not be able to use that feature. Specifying an `INCLUDE` line has the effect of excluding everyone else from that feature; thus, only those users specifically `INCLUDE`d will be able to use that feature.

TIMEOUT

`TIMEOUT` *feature idletime*

The `TIMEOUT` command is used to set up a minimum idle time after which a user will lose his license if he is not using it. This allows the site administrator to prevent users from wasting a license (by keeping it checked out when they are not using it) when someone else wants a license.

REPORTLOG

`REPORTLOG filename`

`REPORTLOG` tells the daemon that it should create a log file suitable for use with the FLEX*lm* report writing tools. This log file maintains more detailed information than the standard log file, but is not meant to be human readable. If the filename starts with a plus character (+), the file will be opened in append mode.

Example

The following is an example of an options file:

```
REPORTLOG /usr/adm/gsi.replog
RESERVE compile USER pat
RESERVE compile USER less
RESERVE compile HOST terry
NOLOG QUEUED
```

SECTION FOUR

Appendixes

A

Installation Notes for Specific Platforms

JAM is installed on over one hundred platforms, each with its own set of commands, command options, and conventions for naming drives. In addition, systems can be configured differently, so it might be difficult for JYACC to predetermine the exact command necessary to install JAM on a specific system. Nevertheless, there are some platforms for which specific details can be provided. Search the following list for your system and media type. If you do not find a match, use the general instructions in the sections “Installing JAM 7 on....”

In some of the commands listed here, you must substitute the name of the device you are using to install JAM (i.e., the device name for your media), for [*device*].

UNIX Platforms

386 UNIX System V Release 4

JAM is distributed on 5.25” high-density diskettes [5H], and 3.5” high-density diskettes [3H] diskettes. Check the label on the diskette to see which format was used. The command you use also depends on the number of the drive you are using to install JAM.

(5H, device 0)	`cpio –ivdBm < /dev/rdsk/f0q15dt`
(5H, device 1)	`cpio –ivdBm < /dev/rdsk/f1q15dt`
(3H, device 0)	`cpio –ivdBm < /dev/rdsk/f03ht`
(3H, device 1)	`cpio –ivdBm < /dev/rdsk/f13ht`

Hewlett Packard HP–UX

X is logical device #

(cartridge)	`tcio –i /dev/rct/device | cpio –ivdBm`
(reel tape)	`tar xvf /dev/rmt/`*X*`m`
(8mm tape)	`tar xvf /dev/rmt/`*X*`h`

IBM RS6000 Systems

(diskettes)	`cpio –ivdBcm < /dev/rfd0`
(8mm tape)	`tar xvf /dev/rmt0`

Due to incompatibilities between system settings under AIX it may be necessary to use the following command when reading the tape, for example:

```
tctl –b0 –p0 –f<device–name> read | tar xvf –
```

DEC UNIX (formerly OSF/1)

(4mm (DAT) tape)	`tar xv`
	or
	`tar xvf /dev/rmt0h`
	or
	`tar xvf /dev/[`*device*`]`

Interactive UNIX System V and SCO UNIX System V

JAM is distributed in `cpio` format on 5.25” high density diskettes [5H] or 3.5” high density diskettes [3H]. The command you use to install JAM also depends on the logical device number of the device you are using.

For `cpio` distributions:

(5H, device 0)	`cpio –ivdBm < /dev/rdsk/f0q15dt`
(5H, device 1)	`cpio –ivdBm < /dev/rdsk/f1q15dt`
(3H, device 0)	`cpio –ivdBm < /dev/rdsk/f0q18dt`
(3H, device 1)	`cpio –ivdBm < /dev/rdsk/f1q18dt`

Microsoft Windows

ECL DLL (WECJLIB.DLL)

`WECJLIB.DLL` is a dynamic link library which performs JPEG decoding. There are two versions of the DLL. The version that comes with JAM can be freely distributed. An enhanced version, which has added functionality, can be order from Express Compression Labs.

In the version of the DLL provided with this package, images are rendered using ordered dithering. The enhanced version of the DLL supports Hi-Color and Tru-Color display hardware. With such hardware, dithering is not necessary and the best possible picture quality can be achieved. Two-pass color quantization and improved dithering are also supported in the enhanced version, which results in better picture quality on 256-color display devices.

For ordering information on single and multiple user licenses for the enhanced version of the DLL, email to:

```
ecl@netcom.com
```

or write to:

Dr. Y. Shan
P.O. Box 367
Caulfield East VIC 3145, Australia

Environment Space

If `COMMAND.COM` runs out of environment space when you issue a `SET` command, you can add the following line to your `CONFIG.SYS` file:

```
SHELL=C:\COMMAND.COM /E:1000 /P
```

If your `COMMAND.COM` resides on a different drive or directory, modify the line accordingly.

MS-Visual C++ Floating-Point Options

For Microsoft Visual C++ distributions, all the distributed libraries created in the current version of JAM are compiled with the /FPc switch, so that you can choose at link time which floating-point library to use. You can use either the math coprocessor library (`LLIBC7.LIB`), the emulator library (`LLIBCE.LIB` — the default), or the alternate math library (`LLIBCA.LIB`).

Using Visual Workbench

JYACC supplies makefiles for creating JAM executables. It is not necessary to use Microsoft Visual C++'s Visual Workbench to create new JAM executables. Instead, you can invoke the `nmake` utility directly from the MS-DOS command line to create executables.

If you want to use Visual Workbench, take the following steps:

1. In the Visual Workbench, choose Open from the Project menu. The Open Project dialog box opens.
2. Select the drive and directory where executables will be built. (You should have already put the makefile you intend to use there.) You can specify the file extension to focus your search in the List Files of Type box. Then select the makefile.
3. Choose OK. When the makefile opens, Visual Workbench prompts you to confirm that it is an external makefile. Visual Workbench displays the External Project Options dialog box instead of the Project Options dialog box if you choose the Project command from the Options menu.
4. From the External Project Options dialog box, select the Release radio button in the Build Mode box.

   Note: *The makefile supported by JYACC does not support DEBUG mode.*
5. In the Release Build box, add the names of the executables you want to build after the makefile name. By default, the makefile supplied by JYACC builds all the Windows executables.
6. Choose OK. The External Project Options dialog box closes.
7. Now you can choose Build from the Project menu of the WorkBench.

The following are questions and answers to the most common problems related to building JAM executables under Visual Workbench:

Q: What should I do if I get a message saying `ERROR: You must define SMBASE. See the instructions in the installation notes.` while building a JAM application?

A: The JYACC makefile requires you to specify where JAM is installed—that is the purpose of the macro `SMBASE`. You can overcome this error by either declaring it on the `nmake` command line (`smbase=c:\jam7` — or whatever the correct value is), or by editing the makefile to set the macro directly in the makefile.

For the first option:

– Choose Project from the Options menu.

– Edit the Build Release box to add the macro definition for `SMBASE`. For details on using macro definitions on the `nmake` command line, run `qh nmake` under the MS-DOS shell.

For the second option, examine the makefile distributed with JAM for details.

Q: Can I add the name of a source code module to my project through the Visual Workbench?

A: No. Visual Workbench does not allow you to edit the external makefile. If you want to add the name of a new source module to your project, use whatever editor you prefer to directly edit the makefile supplied by JYACC. The comments in the makefile should point you in the right direction.

Using JAM Utilities under Windows

Because of various limitations of MS-Windows, it can be challenging to use JAM's utilities when launched from the program manager. The biggest problem arises because Window's notion of the current directory is hidden from the user.

By default, all of JAM's utilities are designed to be run from the MS-DOS command line (with the exception of `BINHERIT.EXE`, refer to the note below). In the MS-DOS environment, there is a current directory (which people usually see by having a prompt of `$P$G` or similar), and when the utilities run they operate on files in that directory by default. Under Windows, however, the current directory is usually set to the directory holding the executable, for example, `c:\jam7\util`.

This means that if you launch a utility, such as `f2asc`, from Program Manager and type `–a foo.asc screen.jam` into the parameters window, `f2asc` will look for `screen.jam` in the `util` directory and create its output `foo.asc` there as well.

Since this is usually not what you want, consider one of the following approaches:

- Enter full paths in the parameters window (for example, `–a d:\myproj\foo.asc d:\myproj\screen.jam`).
- Edit the `.PIF` files to set the directory elsewhere than the `util` directory.
- Run the programs in DOS.

For the `jamdev` executable, you can use the File Manager to associate the extension `.JAM` with `jamdev`. This way, you can double-click on JAM files in the File Manager to launch JAM.

Note: *The* `binherit` *and* `isqlw` *utilities are distributed as Microsoft C QuickWin applications. They cannot be run under DOS. If you are interested in the behavior of QuickWin applications, refer to the QuickWin chapter in the Programming Techniques manual that comes with Visual C/C++.*

VMS Platforms

DEC C Version 4.0 vs. VAX C

This installation of JAM was compiled using DEC C version 4.0. Therefore, this installation of JAM is incompatible with VAX C.

Other Versions of VMS or DEC C

JAM was compiled under VMS version 5.5 using DEC C version 4.0. Since JAM uses VMS shared libraries, any other combination of operating system and DEC C compiler may be a problem. To solve this problem you must relink the utilities. The DCL command file `RELINK.COM` is provided in the `SMBASE:[UTIL]` directory. Executing `@MAKEEXE` relinks all the utilities and `JAMDEV` in the `SMBASE:[UTIL]` directory. Depending on your system, relinking can take from thirty minutes to an hour.

The SMBASE: logical

The `SMBASE:` logical is a concealed device name. It must be used with a colon and directory name (e.g., `SMBASE:[SAMPLES.WELCOME]`, `SMBASE:[LIB]`, `SMBASE:[000000]`). This logical must be defined (as in `SETUP.COM`) to use JAM as described in these installation notes. The default value in `SETUP.COM` is `DKA200:[JAM]`.

In some past versions of JAM for VMS, the `JYACC:` logical was used.

B

JAM7.INI and Databases

The installation procedure for Windows automatically modifies `JAM7.INI` to work with your database driver(s).

Corresponding with the database and version you are using the documented section will be added:

Database and Version	Entry in `JAM7.INI` file

Sybase Version 4 using DB-Library	[databases]
	installed=sybase
	
	[dbms sybase]
	driver= lsyb4dmw.dll
	model= tmsyb1w.dll

Sybase Version 10 using DB-Library	[databases]
	installed=sybase
	
	[dbms sybase]
	driver= lsdbdmw.dll
	model= tmsyb1w.dll

Sybase Version 10 using CT-Library	[databases]
	installed=sybase
	
	[dbms sybase]
	driver= lsctdmw.dll
	model= tmsyb1w.dll

Database and Version	Entry in `JAM7.INI` file

Oracle Version 6 using OCI	[databases]
	installed=oracle
	[dbms oracle]
	driver= lora6dmw.dll
	model= tmora1w.dll

Oracle Version 7 using OCI	[databases]
	installed=oracle
	[dbms oracle]
	driver= lora7dmw.dll
	model= tmora1w.dll

Oracle Version 6 using Pro*C	[databases]
	installed=oracle
	[dbms oracle]
	driver= lemb6dmw.dll
	model= tmora1w.dll

Oracle Version 7 using Pro*C	[databases]
	installed=oracle
	[dbms oracle]
	driver= lemb7dmw.dll
	model= tmora1w.dll

Informix Version 4	[databases]
	installed=informix
	[dbms informix]
	driver= linf4dmw.dll
	model= tminf1w.dll

Informix Version 5 lower than 5.01 WF1	[databases]
	installed=informix
	[dbms informix]
	driver= linf5dmw.dll
	model= tminf1w.dll

Informix Version 5.01 WF1 or higher	[databases]
	installed=informix
	[dbms informix]
	driver= li501dmw.dll
	model= tminf1w.dll

Database and Version	Entry in `JAM7.INI` file
Microsoft Open Database Connectivity (ODBC) version 2	[databases] installed=odbc [dbms odbc] driver= lodbdmw.dll model= tmodb1w.dll

C

Password Request Form

Fill out the following form and fax the completed form to the JYACC password desk at (212) 608-4250.

Refer to the main body of this guide, particularly Chapter 4, for detailed information about JAM licenses. Refer to page 72 for an explanation on how to obtain the password for your license(s). If you require further assistance with any of the information needed on the form or with installing the JAM license file, call JYACC technical support at (800) 826-0050.

JYACC, Inc., grants permission to reproduce at will the password request form provided in this appendix.

Password Request Form Instructions

1. Install all products on the machine or machines that you intend to run therm on.
2. Run the utility `lmhostid` provided in the `$SMBASE/util` directory. This will display the Host Type followed by the Host id. JYACC cannot issue your password without this information.
3. Fill out all requested information on the password request form.
4. Select a method of return: fax or email. Be sure to include all required routing information.
5. Include the serial number as indicated on the software media. For JAM, it is a #JXXXXXXX. For JAM/ReportWriter it is a #RXXXXXXX.
6. Include the version of the product as indicated on the label of the software media.
7. If you are moving a previously installed license, contact your salesperson before requesting your password.
8. If you are requesting more than one password, copy the form as many times as required.
9. If you are faxing: Submit the form to the JYACC Password Desk at 212-608-4250. You do not need a separate cover sheet. A response containing your complete license file will be returned to you.

JYACC Password Request

Name:	____________________
Company:	____________________
Customer ID:	____________________
Phone:	____________________
Return password via:	
Fax:	____________________
Email:	____________________

	Version	Serial #
JAM	________	____________________
JAM/RW	________	____________________
JAM/TPi	________	____________________
JAM/CASE	________	____________________

Host Type	(from lmhostid)	____________________
Host id	(from lmhostid)	____________________
Host id #2	(for a 3 Server License):	____________________
Host id #3	(for a 3 Server License):	____________________

Note: To get host ids #2 and #3, run the program `lmhostid` on the machines that you wish to use as servers #2 and #3 for the three-server configuration described in Section 3 of Read Me First.