Installation and migration guide

This Installation and Migration guide contains the following information:

- Hardware and software prerequisites for VisualAge(R) for Java(TM), Professional and Enterprise Editions
- How to install VisualAge for Java, Professional and Enterprise Editions
- Prerequisites and supported platforms for the team repository server (EMSRV)
- How to install the team repository server (EMSRV)
- How to prepare for the team development environment
- Limitations and known problems with installation
- How to migrate from a previous version of VisualAge for Java (both general and component-specific information)

A PDF version of this guide is also available in the pdf directory on the product CD.

Where to find more information about VisualAge for Java

This file does not include detailed information about the specific components and features of VisualAge for Java, Enterprise Edition. For that information, you should refer to the product Release Notes which you can access by selecting **Start > Programs > VisualAge for Java > Release Notes**. For all languages other than English, the release notes can be found on the Web at http://www.ibm.com/vadd.

This file does not contain information about using VisualAge for Java. Refer to the Getting Started guide and to the online help for that information. Some of the online help has been grouped into PDF documents which you can view and print using Adobe Acrobat Reader (available from http://www.adobe.com/). Not all PDFs are available in all languages. The PDF files are available from the "pdf' directory. This can be found on the product CD or your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. If you did not download the part containing the PDFs, you will not have this directory.

See the PDF Index (in the Getting Started guide) for information on what each PDF file contains. The online help also contains a "Web Resources" section that contains links to VisualAge resources that are available on the Internet.

The VisualAge Developer Domain (VADD) Web site offers technical articles, tutorial, samples, and FAQs, along with easy access to support and product updates for VisualAge for Java. At this site, you can download the VisualAge for Java development tools, as well as reusable beans, wizards, and toolkits to complement your applet and servlet development. See http://www.ibm.com/software/vadd. You can also use this site to request features in upcoming releases of VisualAge for Java.

If you have already subscribed to VisualAge Developers Domain, you do not need to re-register. You can use your current ID and password to obtain ongoing information and code updates. If you are a new user, locate the subscription number in the box (or media kit) that you received. If you purchased VisualAge for Java, Enterprise Edition and do not have a subscription number, contact your IBM(R) sales representative.

The product home page for VisualAge for Java home page is at http://www.ibm.com/software/ad/vajava

Table of contents

Part A: VisualAge for Java, Professional Edition

Part B: VisualAge for Java, Enterprise Edition

Part C: Team repository server (EMSRV)

Part D: Component-specific migration information

* These components are not supported in VisualAge for Java, Version 3.5.

+ Enterprise Edition only

Part E: General information

Appendixes

Part A. Professional Edition

A.1.0 Prerequisites

Important: The installation code for VisualAge for Java makes use of VB script. If you encounter an installation error message that refers to "DoCosting" or VB script runtime errors, then you do not have VB script support installed on your system. VB script support is included in Windows 98 and Windows 2000, but not in Windows NT 4.0. You can download Windows Script 5.5 from http://msdn.microsoft.com/scripting.

This edition of VisualAge for Java, Version 3.5 has the following hardware and software prerequisites:

- Windows(R) 98, Windows 2000, or Windows NT (R) 4.0 with Service Pack 4 or higher.
- TCP/IP installed and configured
- Pentium (R) II processor or higher recommended. If you plan to work in the WebSphere(TM) Test Environment we recommend a minimum processor speed of 400 MHz.

- SVGA (800x600) display or higher (1024 x 768 recommended)
- 48 MB RAM minimum (96 MB recommended)
- To use the Distributed Debugger, you will need 128 MB RAM minimum (196 MB recommended)
- 128 MB RAM minimum is required if you wish to work in the VisualAge for Java WebSphere Test Environment. We strongly recommend 256 MB to avoid disk thrashing.
- Frames-capable Web browser such as Netscape Navigator 4.7 or higher, or Microsoft (R) Internet Explorer 5.0 or higher. You should not use anything lower than Netscape Navigator 4.7 or Internet Explorer 5.0 to view any online documentation.
- Disk space requirements: (based on NTFS) 350 MB minimum (400 MB or more recommended). Disk space on FAT depends on hard disk size and partitioning:
  If you are installing to a very large FAT drive, then the space required for VisualAge for Java is almost doubled (due to 32KB cluster overhead).

If you want to run the Websphere Application Server with DB2(R) and VisualAge for Java concurrently, then a minimum of 512 MB is recommended.

A.2.0 Installation

BEFORE installing VisualAge for Java, please go to http://www.ibm.com/vadd, select "VisualAge for Java Version 3.5, Professional Edition - Patch 1" and follow the instructions included with it.

This section contains information about installing VisualAge for Java, Version 3.5. **Important**: If you are migrating from a previous version of VisualAge for Java, refer to section 3.0 *before* installing Version 3.5. For special information about installing on Windows 2000, refer to section 2.3.

Please also refer to the README (which can be found in the root directory of the CD) for information about pervasive known problems and limitations.

A.2.1 Installing VisualAge for Java, Version 3.5

Before you install the product, check the following things:

- You must have at least 20 MB free on your Windows system drive, and your environment variable TEMP or TMP must point to a valid temporary directory with at least 6 MB free.
- The CLASSPATH must be less than 469 characters. VisualAge for Java inserts approximately 27 characters into the CLASSPATH during installation. These numbers are based on the assumption the target directory is 10 characters long, for example, IBMVJava35.

Installing VisualAge for Java, Version 3.5 from the product CD

1. Insert the CD-ROM into your CD drive. If you are migrating from a previous version of VisualAge for Java, read "Migrating from a previous version of VisualAge for Java" (section 3.0 of this document) BEFORE proceeding with the installation procedure.
2. If autorun is disabled on your system, run setup.exe from the root of the CD drive.
3. Select **Install Products**. Select **Install VisualAge for Java** to begin the installation of VisualAge for Java. If you intend to debug any classes are developed outside the VisualAge for Java IDE or debug programs running on a separate machine, select **Install Distributed Debugger**.
4. Follow the on-screen instructions.
5. Start the VisualAge for Java IDE.

Installing silently

To install VisualAge for Java, Version 3.5 silently, invoke the following command from the ivj35\setup directory:

setup /s /v/qn

VisualAge for Java will automatically be installed in the c:\Program Files\IBM\VisualAge for Java default directory.

To silently install to a different directory (for example, d:\IBMVJava35), invoke the following command from the ivj35\setup directory:

setup /s /v"INSTALLDIR=\"*d:\IBMVJava35*\" /qn"

where *d:\IBMVJava35* is the directory you want to install to.

Installing from the electronic image of VisualAge for Java, Version 3.5

To reduce download time, the electronic image of VisualAge for Java, Enterprise Edition for Windows, Version 3.5 is divided into parts.

- IDE (Integrated Development Environment) - comprises nine archive parts, of which only two are required. This allows you to download only those features that you want.
- Distributed Debugger - provides a separately downloadable part for each operating system that you will use as a target for debugging.
- IBM Developer Kit 1.2.2 - contains the IBM Developer Kit, Java Technology Edition, v 1.2.2, PTF 6b, for the Windows platform.

IDE

There are nine downloadable parts for the Integrated Development Environment. All nine parts are self-extracting archives. You must install the first two; the rest are optional. Refer to the list below for details about what each archive contains.

Once you have downloaded the first two parts plus the optional files that you want, run each self-extracting archive and ensure that each one is extracted into the same temporary directory. Once all the files have been extracted, go to the temporary directory and run `setup.exe`. Follow the on-screen instructions and start the IDE.

If you intend to work in any language other than English, you must download the correct part and run the self-extracting archive for your language *before* you run setup.exe. You cannot change or add a language after you have installed VisualAge for Java.

The following is a description of each archive part:

- VisualAge for Java - IDE 1 of 9 (English only) - REQUIRED. Includes the core Integrated Development Environment, Data Access Beans and Visual Composition Editor.
- VisualAge for Java - IDE 2 of 9 (English only) - REQUIRED. Includes the core Integrated Development Environment, Tools API, Servlet SmartGuide and XML for Java Parser.
- VisualAge for Java - IDE 3 of 9 (English only) - Enhanced Database Support (Stored Procedure Builder and SQLJ), JSP Development Environment, External Version Control.
- VisualAge for Java - IDE 4 of 9 - OPTIONAL. Deployment Environments and online documentation in PDF format.
- VisualAge for Java - IDE 5 of 9 - OPTIONAL. Brazilian and Spanish language pack. This contains the Brazilian and Spanish text for the IDE and all of its components.
- VisualAge for Java - IDE 6 of 9 - OPTIONAL. French and Italian language pack. This contains the French and Italian text for the IDE and all of its components.
- VisualAge for Java - IDE 7 of 9 - OPTIONAL. Japanese language pack. This contains the Japanese text for the IDE and all of its components.
- VisualAge for Java - IDE 8 of 9 - OPTIONAL. German and Korean language pack. This contains the German and Korean text for the IDE and all of its components.
- VisualAge for Java - IDE 9 of 9 - OPTIONAL. Simplified and Traditional Chinese language pack. This contains the Simplified and Traditional Chinese text for the IDE and all of its components.

Distributed Debugger

There is a separately downloadable part for each target operating system that is supported by the Distributed Debugger. Installation instructions vary, as described here:

- **VisualAge for Java - Distributed Debugger for Windows** contains the user interface and the debug engine for Windows. This part is a self-extracting archive. To install it, run this self-extracting archive, and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for AIX** contains the AIX debug engine. To install it, untar the file and follow the instructions in `readme-1st.txt`

- **VisualAge for Java - Distributed Debugger for OS/2** contains the OS2 debug engine. To install it, unzip the file and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for HP-UX** contains the debug engine for HP-UX. To install it, untar the file and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for Solaris** contains the debug engine for the Solaris Operating Environment. To install it, untar the file and follow the instructions in `readme-1st.txt`

IBM Developer Kit 1.2.2

VisualAge for Java - IBM Development Kit 1.2.2 contains the IBM Developer Kit 1.2.2, PTF 6b, for the Windows platform. This is a self-extracting archive. To install it, run this file, which will automatically launch the installation program after it has been extracted from the archive, and follow the instructions.

A.2.2 Installing additional components later

To install additional VisualAge for Java components any time after the initial installation, insert the CD-ROM into your CD drive, select to install VisualAge for Java, and select **Modify** in the **Program Maintenance** screen. If autorun is disabled on your system, you will have to run setup.exe from the root of the CD drive. If you have an electronic version of VisualAge for Java, you will also have to manually run setup.exe, and follow the same steps.

All the components will be listed in the Edit Features window, with an indication of their current installation states. A red 'X' indicates that a component is not currently installed. You can select to install these components. Do not select any components that are not marked with a red 'X'; they have already been installed.

You cannot install a second instance of VisualAge for Java, Version 3.5. You cannot change the installation language without uninstalling the product first.

A.2.3 Installation considerations for Windows 2000

This release of VisualAge for Java provides toleration support (as defined by Microsoft) for Windows 2000.

Beginning with this release, the default installation directory is <Program Files>\IBM\VisualAge for Java. For Windows 2000, by default, installation to <Program Files> can only be performed by Administrators and Standard (Power) users. Regular (Restricted) users cannot write to this location.

Due to the current design of VisualAge for Java, if the product is installed to this location and is to be used by a Regular (Restricted) user, you must change the security settings for either the IBM or IBM\VisualAge for Java directory under <Program Files> to allow write access by regular users. Failure to do so may result in failures when attempting to start the IDE.

A.2.4 Installing the IBM Developer Kit, Java Technology Edition, v1.2.2, PTF 6b

To install the IBM Developer Kit, Java Technology Edition, v1.2.2, PTF 6b, run install.exe from the IBM Developer Kit directory. This directory is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. For detailed information about the IBM Developer Kit, refer to the readme.txt file in the IBM Developer Kit directory.

A.3.0 Migrating from a previous version of VisualAge for Java

Refer to Part D and Part E for information about both component-specific and general migration information before beginning the migration process.

If you are currently using a VisualAge for Java, Version 2.0, 3.0 or 3.02 with JDK 1.1.x support, you cannot automatically migrate to VisualAge for Java, Version 3.5. These versions of VisualAge for Java can coexist with VisualAge for Java, Version 3.5. Refer to section 3.2 for information about migrating your repository contents and your resource files from Version 2.0 or 3.0x of VisualAge for Java.

You **can** automatically migrate from Version 3.0x, Early Adopters Environment for Java 2 Platform, Standard Edition, v1.2. This version of VisualAge for Java cannot coexist with VisualAge for Java, Version 3.5. Refer to section 3.1 for information about migrating from VisualAge for Java, Version 3.0x, Early Adopters Environment for Java 2 Platform, Standard Edition, v1.2.

A.3.1 Migrating from VisualAge for Java, VisualAge for Java, Version 3.0x Early Adopters

Environment for Java 2 Platform, Standard Edition, v1.2

Migration from VisualAge for Java, Version 3.0x Early Adopters Environment is automatic. The Version 3.5 installation program automatically upgrades an installed Version 3.0x, Early Adopters product to Version 3.5. Follow these steps to migrate to Version 3.5:

Automatic migration

1. Perform the following steps to back up your user data. It is strongly recommended you create backup copies of your repository and resource files in case problems occur during migration.
   a) Version your projects and packages. Only versioned projects and packages can be imported into the VisualAge for Java, Version 3.5 repository. Refer to your VisualAge for Java online help for versioning instructions.
   b) Save your repository to a new location outside your VisualAge for Java directory tree. The file name and path of the repository is x:\IBMVJava\ide\repository\ivj.dat, where x:\IBMVJava is the VisualAge for Java installation directory.
   c) Copy any resource files (such as images or sound files) used by your Java applications to a directory outside your VisualAge for Java directory tree. By default, resource files for each VisualAge for Java project are located in subdirectories called x:\IBMVJava\ide\project_resources\project, where x:\IBMVJava is the VisualAge for Java installation directory and project is the name of the project with which the resources are associated.
2. To install VisualAge for Java, Version 3.5, refer to the installation instructions in section 2.0.
3. Select to upgrade your current installation. You do not need to uninstall Version 3.0x, Early Adopters as it is automatically upgraded to Version 3.5. Version 3.5 will automatically be installed in the directory Version 3.0x, Early Adopters was installed in.
4. Upon the completion of a successful upgrade installation, all of your project resources and your repository data are automatically migrated to Version 3.5 when you start the Version 3.5 IDE for the first time. Your project resources are automatically versioned as well. For more information about versioning project resources, refer to the online help.

In the event that an installation failure occurs, you must manually migrate your user data. You also must manually migrate your user data if the IDE fails to start or if an error is encountered by the IDE when it is migrating your user data. Follow these steps to recover from a failure and manually migrate your user data:

Manual migration

1. Verify you have saved your repository and resource files *outside* the VisualAge for Java directory tree.
2. Uninstall the product completely. All Version 3.0x, Early Adopters and Version 3.5 subdirectories and files should be deleted.
3. Reboot and install Version 3.5.
4. Start the IDE.
5. You can now import packages and projects from your old repository into the new repository. In the Workbench, select **File** > **Import**, and then select the **Repository** radio button and click **Next**. In the Repository name field enter the path of your backup copy of ivj.dat. Then select the projects and packages that you want to import. You will not be able to import any projects or packages that have not been versioned.
6. To automatically add the selected projects to the workspace, select the **Add most recent project edition to the workspace** checkbox. This checkbox is only available when the **Projects** radio button is selected.
7. Click **Finish**.
8. Copy the backup copy of your resource files into the subdirectory x:\IBMVJava\ide\project_resources\project, where x:\IBMVJava is the VisualAge for Java, Version 3.5 installation directory and project is the name of the project with which the resources are associated.

A.3.2 Migrating from Version 2.0 or 3.0x of VisualAge for Java

You can manually migrate the contents of your repository and your resource files from Version 2.0 and Version 3.0x of VisualAge for Java. Refer to the online help file "Repairing class or package references" for information about repairing breakage.

You do not have to uninstall Version 2.0 or 3.0x; they can coexist with VisualAge for Java, Version 3.5.

Follow all the steps below *before* uninstalling Version 2.0 or 3.0x if you want to import your Version 2.0 or 3.0x repository and resource files into Version 3.5. If you are *not* uninstalling Version 2.0 or 3.0x, you do not need to perform steps 2 and 3, but you must perform all the other steps.

1. Version your projects and packages. Only versioned projects and packages can be imported into this version of VisualAge for Java. Refer to your VisualAge for Java online help for versioning instructions.
2. Save your repository to a new location *outside* your VisualAge for Java directory tree. The file name and path of the repository is x:\IBMVJava\ide\repository\ivj.dat, where x:\IBMVJava is the VisualAge for Java installation directory.
3. Copy any resource files (such as images or sound files) used by your Java applications to a directory *outside* your VisualAge for Java directory tree. By default, resource files for each VisualAge for Java project are located in subdirectories called x:\IBMVJava\ide\project_resources\project, where x:\IBMVJava is the VisualAge for Java installation directory and project is the name of the project with which the resources are associated.
4. You can now uninstall Version 2.0 or 3.0x, and install Version 3.5. You do not have to uninstall Version 2.0 or 3.0x; they can coexist with VisualAge for Java, Version 3.5.
5. Install VisualAge for Java Version 3.5 (see section 2.0) and start the Version 3.5 IDE.
6. Refer to the online help for information about importing packages and projects from your old repository into the new repository. You will not be able to import any projects or packages that have not been versioned.
7. Copy the backup copy of your resource files into the subdirectory x:\IBMVJava\ide\project_resources\project, where x:\IBMVJava is the VisualAge for Java, Version 3.5 installation directory and project is the name of the project with which the resources are associated.
8. Once you have verified that your manual migration is successful, you can delete your backup copies created in steps 2 and 3.

A.4.0 Known problems and limitations

Please also refer to the README (which can be found in the root directory of the CD) for information about pervasive known problems and limitations.

The following is a list of issues you should be aware of during installation:

A.4.1 Known problems and limitations with Installation

A.4.1.1 Disk limitations

- Do not install to an HPFS network or local drive since Windows 98, Windows 2000, and Windows NT 4.0 have trouble handling long file names on HPFS.
- Do not install to a Novell NetWare drive. Installation will fail on a Novell NetWare drive.

A.4.1.2 Environment space (Windows 98)

On Windows 98, you should edit your config.sys file to increase the environment space to 32000. Insert the following line:

SHELL=X:\COMMAND.COM /E:32000 /P
where "X:" is your boot drive. Reboot before you install the product. You must reboot after you install the product as well before you start the IDE.

A.4.1.3 User authorization

- You must have write access to the root "\" directory of the drive where VisualAge for Java is installed to install the NetQuestion Search Server.
- When you install the product on Windows 2000, you may receive a warning message if you are not an Administrator. The message indicates that some programs may not work correctly if they are not installed by an Administrator. You should log on as an Administrator before beginning the VisualAge for Java installation.
- When a Restricted User attempts to install VisualAge for Java, Version 3.5 on Windows 2000, they may incorrectly receive a "Setup Initialization Error" with the message "Setup has detected that unInstallShield is in use. Please close unInstallShield and restart setup. Error 432." Verify that you have Administrator or Standard User access rights before attempting to install the product again.
- If you do not have Administrator rights while installing on Windows NT, the online help search function will not

work.

A.4.1.4 TCP/IP considerations

- An "Autoconfigured" warning from the installation program indicates that your proxy exceptions are autoconfigured for your browser. Check with your system administrator to ensure that 127.0.0.1 is treated as a local address
- You must install and configure TCP/IP in order for IBM VisualAge for Java to install properly. For Windows 98 and Windows 2000, you must enable TCP/IP as follows:
  1. For a LAN Adapter configuration:
     - You must have DNS enabled with a valid host and domain name.
     - Your LAN DNS must resolve localhost to 127.0.0.1.
     - You cannot run disconnected with a LAN Adapter configuration.
  2. For a dial-up Adapter configuration:
     - You must have DNS disabled.
     - Your TCP/IP address must be obtained automatically.

  These configuration options will apply to all TCP/IP adapters, even though they have only been changed for this one. You will not be able to use both LAN and dial-up without reconfiguring.

  Dial-up networking TCP/IP properties for your Internet service provider (ISP) must be configured as documented by the ISP. The dial-up networking TCP/IP properties will override the properties in the dial-up Adapter TCP/IP properties configured via the Network icon in the Windows 98 or Windows 2000 Control Panel. The overriding of the properties will take place only if the dial-up Adapter TCP/IP properties are configured as above. You must not enable the DNS in the dial-up Adapter TCP/IP properties or set an IP address in the dial-up Adapter TCP/IP properties, because doing so will interfere with the dial-up networking configuration for the ISP.

  For Windows NT 4.0, you can use either of the TCP/IP configurations described above. If you are running standalone, you can also enable the Microsoft Loopback Adapter without the other two adapters.

A.4.1.5 Shell extension (Windows NT)

If you get a message that indicates that the installation program has detected a Shell Extension for Windows NT, the installation cannot proceed. You should then perform the following steps:

1. Make sure you have an Emergency Recovery Disk. Instructions for creating this are available in the Windows help documentation.
2. Invoke regedit.exe from a command prompt.
3. Expand the key \\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
4. Select the shell name in the name/data pairs for the above key. **Important!** Make a note of the data recorded for this name, because you will need it after installing IBM VisualAge for Java.
5. Select **Edit** > **Modify** from the menu bar for the shell name/data pair.
6. Set the value for the shell name to Explorer.exe. Click **OK**.
7. Select **Registry** > **Exit** from the menu bar.
8. Restart and complete IBM VisualAge for Java installation.
9. Once installation is complete, restore the previous registry entry as follows:
   a. Invoke regedit.exe from a command prompt.
   b. Expand the key \\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
   c. Select the shell name in the name/data pairs for the above key.
   d. Select **Edit** > **Modify** from the menu bar for the shell name/data pair.
   e. Restore the value for the shell name to the value that was recorded in Step 4. Click **OK**.
   f. Select **Registry** > **Exit** from the menu bar.

A.4.1.6 Recovering from failed installation

If your installation fails, you must remove any Version 3.5 files that have been installed. If the directory you

intended to install VisualAge for Java in is empty, then the installation process rolled back and removed any files that were installed. You can delete the empty directory if you want to, but it is not necessary. However, if the directory contains files, then you must start the installation process again. It will open in maintenance mode and you must select to remove your partial installation of Version 3.5 before you can try to install the product again.

You will also need to delete the registry entry:

\\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\VisualAge for Java for Windows

using the following instructions:

1. Make sure you have an Emergency Recovery Disk. Instructions for creating this are available in the Windows help documentation.
2. Invoke regedit.exe from a command prompt.
3. Expand and select the key \\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\VisualAge for Java for Windows\3.5
4. Select **Edit** > **Delete** from the menu bar for this key.
5. Select **Yes** when asked to confirm deletion of the key.
6. Select **Registry > Exit** from the menu bar.

If NetQuestion was installed before the installation failed, you must delete it as well.

There are three ways to check to see if NetQuestion has been installed:

1. A folder named imnnq_*XX* (where *XX* = NT, 95, or 98, depending on your version of Windows) has been created on the root of the drive you are install VisualAge for Java, Version 3.5 to. On Windows NT and Windows 2000, the directory will be IMNnq_NT and on Windows 98 it will be IMNnq_95 or IMNNQ_98.
2. There is a registry entry for NetQuestion under HKEY_LOCAL_MACHINE\SOFTWARE\IBM
3. There are environment variables set that match following naming pattern: IMN*

To remove NetQuestion, follow these steps. If NetQuestion had been installed by another product that uses it (for example, DB2), then only perform step 2 in the steps below; you should not perform any of the other steps.

1. Delete the registry entry using regedit.
   a. Invoke regedit.exe from a command prompt.
   b. Expand and select the key:
   HKEY_LOCAL_MACHINE/SOFTWARE/IBM/NetQuestion
   c. Select **Edit** > **Delete** from the menu bar for this key.
   d. Select **Yes** when asked to confirm deletion of the key.
   e. Select **Registry > Exit** from the menu bar.
2. From a command prompt, go to the imnnq_XX directory and enter the command uninstnq
3. Delete the environment variables matching the IMN* pattern.
4. Delete the imnnq_XX directory.
5. On Windows 98 only, reboot.

Back up your repository and resource files if you have not already done so. Refer to Part A, Section 3.1 for information on how to perform this task.

Reboot and re-install the product after you have performed all these steps.

A.4.2 Known problems and limitations with Uninstallation

The following is a list of items that you should be aware of during uninstall.

A.4.2.1 Disk space

You must have at least 2MB free on your Windows system drive, and your environment variable TEMP or TMP must point to a valid temporary directory with at least 5MB free.

A.4.2.2 Uninstalling the Distributed Debugger

If you installed the Distributed Debugger, you should uninstall VisualAge for Java BEFORE you uninstall the

Distributed Debugger. After you have uninstalled VisualAge for Java, you can uninstall the Distributed Debugger by going to the Debugger installation directory and running the uninstallation program.

If you have uninstalled VisualAge for Java, and you cannot uninstall the Distributed Debugger, you must delete the registry key:

HKEY_LOCAL_MACHINE/SOFTWARE/IBM/IBM Distributed Debugger/CurrentVersion/install/ParentProducts/Visual Age for Java

and then try to uninstall the Debugger again. DO NOT follow these steps if you are using the Distributed Debugger with any other products, as you will no longer be able to use the debugger after you delete the registry key.

Follow these steps:

1. Invoke regedit.exe from a command prompt.
2. Expand and select the key:
   HKEY_LOCAL_MACHINE/SOFTWARE/IBM/IBM Distributed Debugger/CurrentVersion/install/ParentProducts/Visual Age for Java
3. Select **Edit** > **Delete** from the menu bar for this key.
4. Select **Yes** when asked to confirm deletion of the key.
5. Select **Registry > Exit** from the menu bar.

A.4.2.3 Environment variables (Windows 98)

When you uninstall VisualAge for Java on Windows 98, some environment entries may be left in your autoexec.bat file. Normally these leftover entries do not cause any problems, but if you uninstall and re-install the product several times, two problems may occur. You may have conflicting path statements that can prevent the online help from working or you may run out of path space, which could prevent you from re-installing the product successfully.

To solve these problems:

1. Make a backup copy of your autoexec.bat file.
2. Determine if you have another program that requires the HTML search engine (such as DB2) on your system by following these steps:
   a) Uninstall VisualAge for Java and reboot your system.
   b) Run regedit.exe from a command prompt and expand the HKEY_LOCAL_MACHINE\SOFTWARE\ tree. If there is an IBM directory in this tree, expand it to see if there is an NetQuestion directory. If you see this directory, then you are probably using the search engine with another IBM product.
3. If you are *not* using the HTML search engine for another product, then remove any IMN or IMQ entries in your autoexec.bat file before re-installing VisualAge for Java.
4. If you *are* using the HTML search engine for another product, delete any duplicates of these entries from your autoexec.bat file:

   IMNINST
   IMNINSTSRV
   IMNNQ
   IMNNQ_95
   IMQCONFIGCL
   IMQCONFIGSRV
   Also delete the line IF EXIST X:\IMNNQ_05\IMNENV.BAT CALL IMNEV.BAT
5. Make sure that you do not remove the original entries when you remove the duplicates. If you are not sure which entries are the original ones, then you must determine where the system considers NetQuestion to be installed. Follow these steps:
   1. Start regedit.exe from a command prompt.
   2. Expand the key \\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\NetQuestion\Installation Directory
   3. The directory value inside of this key shows you the path where NetQuestion is installed. Certain environment variables must contain this directory as part of their value for NetQuestion to function correctly.

If you find any of the above environment variables that include a directory different from the one found in the registry as part of their value, delete them.

A.4.2.4 Uninstalling NetQuestion

When you are uninstalling VisualAge for Java, NetQuestion may not be uninstalled. Problems may occur if NetQuestion is not uninstalled and you later attempt to reinstall the product.

To remove NetQuestion, follow these steps. If NetQuestion had been installed by another product that uses it (for example, DB2), then only perform step 2 in the steps below; you should not perform any of the other steps.

1. Delete the registry entry using regedit.
   a. Invoke regedit.exe from a command prompt.
   b. Expand and select the key:
   HKEY_LOCAL_MACHINE/SOFTWARE/IBM/NetQuestion
   c. Select **Edit** > **Delete** from the menu bar for this key.
   d. Select **Yes** when asked to confirm deletion of the key.
   e.Select **Registry > Exit** from the menu bar.
2. From a command prompt, go to the imnnq_*XX* directory and enter the command uninstnq. On Windows NT and Windows 2000, the directory will be IMNnq_NT and on Windows 98 it will be IMNnq_95 or IMNNQ_98.
3. Delete the environment variables matching the IMN* pattern.
4. Delete the imnnq_XX directory.
5. On Windows 98 only, reboot.

Part B: VisualAge for Java, Enterprise Edition

B.1.0 Prerequisites

Important: The installation code for VisualAge for Java makes use of VB script. If you encounter an installation error message that refers to "DoCosting" or VB script runtime errors, then you do not have VB script support installed on your system. VB script support is included in Windows 98 and Windows 2000, but not in Windows NT 4.0. You can download Windows Script 5.5 from http://msdn.microsoft.com/scripting.

B.1.1 General prerequisites

This edition of VisualAge for Java, Version 3.5, has the following hardware and software prerequisites:

- Windows 98, Windows 2000, or Windows NT(R) 4.0 with Service Pack 4 or higher.
- TCP/IP installed and configured
- Pentium II (R) processor or higher recommended. If you plan to work in the WebSphere Test Environment we recommend a minimum processor speed of 400 MHz.
- SVGA (800x600) display or higher (1024x768 recommended)
- 96 MB RAM minimum (160 MB recommended)
- 128 MB RAM minimum is required if you wish to work in the VisualAge for Java WebSphere Test Environment. We strongly recommend 256 MB to avoid disk thrashing.
- To use the Distributed Debugger, you will need 128 MB RAM minimum (196 MB recommended)
- Disk space requirements: (based on NTFS) 400 MB minimum (750 MB or more recommended.) Disk space on FAT depends on hard disk size and partitioning. If you are installing to a very large FAT drive, then the space required for VisualAge for Java is almost doubled (due to 32KB cluster overhead). If you are installing the e-connectors (automatically installed with the Transactions Access Builder) you may need an extra 150-200 MB disk space free.
- Frames-capable Web browser such as Netscape Navigator 4.7 or higher, or Microsoft (R) Internet Explorer 5.0 or higher. **Important**: You must have the Windows Scripting Host on your system in order to install VisualAge for Java. The Host is included in Microsoft Internet Explorer (Version 3.0 or higher). You should not use anything lower than Netscape Navigator 4.7 or Internet Explorer 5.0 to view any online documentation, but you must have Internet Explorer (Version 3.0 or higher) installed to install VisualAge for Java.

If you want to run the Websphere Application Server with DB2 and VisualAge for Java concurrently, then a minimum of 512 MB is recommended.

You must use EMSRV, Version 7.0, if you are working in a team environment. For information on moving from Version 6.x to Version 7.0, refer to Part C.

B.1.2 Component-specific prerequisites

Certain components have specific prerequisites:

Domino(TM) AgentRunner

For development of the applications, Notes 5.0.5 (or higher) Client and the Domino 5.0.5 (or higher) Designer are required. For the execution of applications, Notes 5.0 (or higher) Client or Domino 5.0 (or higher) Server is required.

Domino Access Builder

Requires Notes 5.0.5 (or higher) Client or Domino 5.0.5 (or higher) Server

Enterprise Toolkit for AS/400 (R) (ET/400)

OS/400 Release V, V4R3M0 or higher is required

Enterprise Toolkit for OS/390(R) (ET/390)

On OS/390:

- NFS daemon running on OS/390
- REXEC server running on OS/390
- OMVS FTP server running on OS/390

On workstations, the NFS Maestro Client for Windows NT or Windows 2000.

Enterprise Access Builder for SAP R/3

Requires access to an SAP R/3 server.

Enterprise Access Builder for IMS(TM)

In order to run a Java application or servlet that uses IMS Connector for Java, the following separately packaged and installed products must be installed and running on the target host machine:

- IMS Connect V1.1.0 and IMS Version 7 (recommended) or
- IMS Connect V1.1.0 and IMS Version 5.1 or 6 or
- IMS TCP/IP OTMA Connection (IMS TOC) Version 2.1.3 and IMS Version 5.1 or above. (IMS TOC will continue to be supported for a limited time. See the IMS Web pages at http://www.ibm.com/software/data/ims for further details.)

B.2.0 Installation

BEFORE installing VisualAge for Java, please go to http://www.ibm.com/vadd, select "VisualAge for Java Version 3.5, Enterprise Edition - Patch 1" and follow the instructions included with it.

Please also refer to the README (which can be found in the root directory of the CD) for information about pervasive known problems and limitations.

B.2.1 Installing VisualAge for Java, Version 3.5, Enterprise Edition

This section contains information about installing VisualAge for Java, Version 3.5. **Important**: If you are migrating from a previous version of VisualAge for Java, refer to section 3.0 below *before* installing VisualAge for Java, Version 3.5. For special information about installing on Windows 2000, refer to section 2.5.

Before you install the product, check the following things:

- The CLASSPATH must be less than 359 characters. VisualAge for Java inserts approximately 141 characters into the CLASSPATH during the installation. These numbers are based on the assumption that the target directory is 10 characters long (for example, IBMVJava35).
- The PATH must be less than 390 characters on Windows 98 and Windows 2000 and less than 890 characters on Windows NT.
- You must have at least 20 MB free on your Windows system drive, and your environment variable TEMP or TMP must point to a valid temporary directory with at least 6 MB free.

You must perform the following instructions, regardless of whether you are installing the VisualAge for Java team

clients or installing a client with a local repository. Refer to section 2.3 for details about installing a team client or section 2.4 for details about installing a local repository.

Installing VisualAge for Java, Version 3.5 from the product CD

1. If you are migrating from a previous version of VisualAge for Java, read "Migrating from a previous version of VisualAge for Java", section 3.0 of this document, BEFORE proceeding with the installation procedure.
2. Insert the CD-ROM into your CD drive.
3. If autorun is disabled on your system, run setup.exe from the root of the CD drive.
4. Select **Install Products**. Select **Install VisualAge for Java** to begin the installation of VisualAge for Java. If you intend to debug any classes are developed outside the VisualAge for Java IDE or debug programs running on a separate machine, you can install the Distributed Debugger from **disk 2.** From the disk 2 Install screen select **Install Products**, then **Install Distributed Debugger**.
5. Follow the on-screen instructions.
6. Start the VisualAge for Java IDE.

Installing silently

To install VisualAge for Java, Version 3.5 silently, invoke the following command from the ivj35\setup directory:

setup /s /v/qn

VisualAge for Java will automatically be installed in the c:\Program Files\IBM\VisualAge for Java default directory.

To silently install to a different directory (for example, d:\IBMVJava35), invoke the following command from the ivj35\setup directory:

setup /s /v"INSTALLDIR=\"*d:\IBMVJava35*\" /qn"

where *d:\IBMVJava35* is the directory you want to install to.

Installing from the electronic image of VisualAge for Java, Version 3.5

To reduce download time, the electronic image of VisualAge for Java, Enterprise Edition for Windows, Version 3.5 is divided into parts.

- IDE (Integrated Development Environment) - comprises twelve archive parts, of which only two are required. This allows you to download only those features that you want.
- Distributed Debugger - provides a separately downloadable part for each operating system that you will use as a target for debugging.
- EMSRV - One archive part contains the team server code for all supported server platforms, which are in listed in Part C, Section 1.0 of this document.
- IBM Developer Kit 1.2.2 - contains the IBM Developer Kit, Java Technology Edition, v 1.2.2, PTF 6b, for the Windows platform.

IDE

There are twelve downloadable parts for the Integrated Development Environment. All twelve parts are self-extracting archives. You must install the first two; the rest are optional. Refer to the list below for details about what each archive contains.

Once you have downloaded the first two parts plus the optional files that you want, run each self-extracting archive and ensure that each one is extracted into the same temporary directory. Once all the parts have been extracted, go to the temporary directory and run `setup.exe`. Follow the on-screen instructions and start the IDE.

If you intend to work in any language other than English, you must download the correct part and run the self-extracting archive for your language *before* you run setup.exe. You cannot change or add a language after you have installed VisualAge for Java

The following is a description of each part:

- VisualAge for Java - IDE 1 of 12 (English only) - REQUIRED. Includes the core Integrated Development Environment, Data Access Beans and Visual Composition Editor.

- VisualAge for Java - IDE 2 of 12 (English only) - REQUIRED. Includes the core Integrated Development Environment, Tools API, Servlet SmartGuide and XML for Java Parser.
- VisualAge for Java - IDE 3 of 12 (English only) - OPTIONAL. Enhanced database support (Stored Procedure Builder and SQLJ), EJB/JSP Development Environment, Persistence Builder, External Version Control, ET/400, ET/390, Tivoli connection.
- VisualAge for Java - IDE 4 of 12 (English only) - OPTIONAL. Application Access Builders (Domino, SAP R/3, C++), Transactions Access Builder (requires part 5), XMI Toolkit, XML Generator, Domino Agent Runner, IDL Development Environment.
- VisualAge for Java - IDE 5 of 12 - OPTIONAL. e-connectors. If you install this part, you must install part 6 as well. **NOTE**: This is required if you are going to install the Transaction Access Builder
- VisualAge for Java - IDE 6 of 12 - OPTIONAL. e-connectors. If you install this part, you must install part 5 as well. **NOTE**: This is required if you are going to install the Transaction Access Builder
- VisualAge for Java - IDE 7 of 12 - OPTIONAL. Deployment Environments, Technology Previews and online documentation in PDF format.
- VisualAge for Java - IDE 8 of 12 - OPTIONAL. Brazilian and Spanish language pack. This contains the Brazilian and Spanish text for the IDE and all of its components.
- VisualAge for Java - IDE 9 of 12 - OPTIONAL. French and Italian language pack. This contains the French and Italian text for the IDE and all of its components.
- VisualAge for Java - IDE 10 of 12 - OPTIONAL. Japanese language pack. This contains the Japanese text for the IDE and all of its components.
- VisualAge for Java - IDE 11 of 12 - OPTIONAL. German and Korean language pack. This contains the German and Korean text for the IDE and all of its components.
- VisualAge for Java - IDE 12 of 12 - OPTIONAL. Simplified and Traditional Chinese language pack. This contains the Simplified and Traditional Chinese text for the IDE and all of its components.

Distributed Debugger

There is a separately downloadable part for each target operating system that is supported by the Distributed Debugger. Installation instructions vary, as described here:

- **VisualAge for Java - Distributed Debugger for Windows** contains the user interface and the debug engine for Windows. This part is a self-extracting archive. To install it, run this self-extracting archive, and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for AIX** contains the AIX debug engine. To install it, untar the file and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for OS/2** contains the OS2 debug engine. To install it, unzip the file and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for HP-UX** contains the debug engine for HP-UX. To install it, untar the file and follow the instructions in `readme-1st.txt`
- **VisualAge for Java - Distributed Debugger for Solaris** contains the debug engine for the Solaris Operating Environment. To install it, untar the file and follow the instructions in `readme-1st.txt`

EMSRV (team server)

VisualAge for Java - EMSRV 7.0 contains the repository server program for the team development environment. A single archive part contains the server code for Windows, AIX, OS/2, NetWare, HP-UX, Linux, and Solaris, in ZIP file format. To install, unzip this part and follow the instructions in `instmigr.htm`

IBM Developer Kit 1.2.2

VisualAge for Java - IBM Development Kit 1.2.2 contains the IBM Developer Kit 1.2.2, PTF 6b, for the Windows platform. This is a self-extracting archive. To install, run this file, which will automatically launch the installation program after it has been extracted from the archive, and follow the instructions.

B.2.2 Installing additional components later

To install additional VisualAge for Java components any time after the initial installation, insert the CD-ROM into your CD drive, select to install VisualAge for Java, and select **Modify** in the **Program Maintenance** screen. If

autorun is disabled on your system, you will have to run setup.exe from the root of the CD drive. If you have an electronic version of VisualAge for Java, you will also have to manually run setup.exe and then follow the same steps.

All the components will be listed in the Edit Features window, with an indication of their current installation states. A red 'X' indicates that a component is not currently installed. You can select to install these components. Do not select any components that are not marked with a red 'X'; they have already been installed.

You cannot install a second instance of VisualAge for Java, Version 3.5. You cannot change the installation language without uninstalling the product first.

B.2.3 Installing the VisualAge for Java team clients

Before each member of the development team can follow these steps, the EMSRV administrator must have set up and started the server, tested the client connection, and added the team developers to the repository user list. Refer to the Part C in this guide for information on performing these tasks. Part C also provides information on migrating a team repository.

In the following instructions, it is assumed that the shared repository that is installed on the server is called ivj.dat. EMSRV, Version 7.0 must have been installed on your server. You will receive an error message if you try to connect to a previous version of EMSRV.

1. Before you start to install VisualAge for Java, Version 3.5, ask your administrator for the following information:
   - The host name or IP address of the server.
   - The file name of the shared repository. By default, the file name is ivj.dat; you will not need to specify path information if the administrator specifies the repository's location as the EMSRV working directory when starting the server.
   - Your repository user name. You will need to select this name as the workspace owner when you start the VisualAge for Java client.
   - If password protection is enabled, the password for that workspace owner. (By default, password protection is not enabled.)
2. Start the VisualAge for Java, Version 3.5 installation. For details on installation, refer to section 2.1. If you are migrating from a previous version of VisualAge for Java, refer to section 3.0 for details about the migration process.
3. When prompted by the installation program, specify that you wish to use a repository that resides on a server. If, instead of always working as client connected to a shared repository, you would like to have a local repository on your workstation for working in standalone mode, see the alternative instructions below in section 2.4
   Provide the server's TCP/IP host name (or IP address) and the name of the repository, as given to you by your administrator. If the administrator did not specify a working directory when starting the EMSRV program, then you will need to fully qualify the repository's name with the server's path information for that file. The installation program will automatically insert the server address and repository information into the client's ide.ini file.
4. You will be asked to select a workspace owner name from the list of repository users that the administrator has set up. Select your user name. If password protection has been enabled, you will need to provide the user password.

   A progress bar indicates that the workspace is being connected to the repository. If, instead of a user list, you see an error message indicating that an unrecoverable error has occurred, one of the following situations may have occurred:
   1. The server is not active.
   2. You specified an incorrect server name when you installed VisualAge for Java on your workstation.
   3. You specified an incorrect repository name during installation.

   You can correct the server or repository information by editing the ide.ini file on the team client.

B.2.4 Installing a client that has a standalone repository

You may wish to have your own VisualAge for Java source code repository on your workstation, to use when you are not connected to the shared repository. In this case, when you are installing VisualAge for Java, Version 3.5 on

your workstation, specify that your repository will be on your local machine, rather than on the server. The installation program will create a repository for you.

When you want to use the shared repository on the team server, refer to the online help or the team.pdf file. The team.pdf file is in the pdf directory, which is on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. If you did not download the part containing the PDFs, you will not have this directory.

B.2.5 Installation and usage considerations for Windows 2000

This release of VisualAge for Java for Windows provides toleration support (as defined by Microsoft) for Windows 2000.

Beginning with this release, the default installation directory is <ProgramFiles>\IBM\VisualAge for Java. For Windows 2000, by default, installation to <ProgramFiles> can only be performed by Administrators and Standard (Power) users. Regular (Restricted) users cannot write to this location.

Due to the current design of VisualAge for Java, if the product is installed to this location, and is to be used by a Regular (Restricted) user, you must change the security settings for the IBM or IBM\VisualAge for Java directory under <ProgramFiles> to allow write access by regular users. Failure to do so may result in failures when attempting to start the IDE or while using some VisualAge for Java tools within the IDE.

The server list for the AS/400 Enterprise Toolkit is now stored as "<ProgramFiles>\IBM\shared files\fvdctcp.txt". Again, by default, this location is protected, and cannot be written to by Regular users. If a user without sufficient authority attempts to create or update the server list through any of the AS/400 SmartGuides' Add/Modify server list buttons, the file creation or update will fail with an io error in its Java code, which may or may not show up in the log or console.

The system Administrator must determine whether to give write access to the regular user for this location, or to keep it protected and load the file manually, preventing unauthorized users from updating the file.

B.2.6 Installing the IBM Developer Kit, Java Technology Edition, v1.2.2, PTF 6b

To install the IBM Developer Kit, Java Technology Edition, v1.2.2, PTF 6b, run install.exe from the IBM Developer Kit directory. This directory is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. For detailed information about the IBM Developer Kit, refer to the readme.txt file in the IBM Developer Kit directory.

B.3.0 Migrating from a previous version of VisualAge for Java

If you are migrating from a previous version of VisualAge for Java, Enterprise Edition, and you work in a stand-alone environment (rather than a team environment); and you will continue working in a stand-alone environment, follow the migration instructions for Professional Edition in Part A of this document.

If you are migrating from a team environment or to a team environment, consider the following issues before you begin the migration process.

- Will you continue to use your existing repositories? If yes, there are two approaches that you can take.
  - You can migrate the contents of the Version 3.5 repository into your existing repository. This will allow users of mixed development environments to access the same repository simultaneously, although differences in JDK levels may prevent you from treating all code as common. For more information on how to perform this task, refer to Section 3.0 in Part C.
  - You can continue using your existing repository and also install the Version 3.5 repository for separate use. In this case, you may have deal with issues like dual maintenance and planning the eventual, phased migration of your users to the Version 3.5 repository.
- What version of VisualAge for Java are you migrating from? If you decide to upgrade your JDK 1.1.x-based repository and you migrate your applications to work properly with the Java 2 Platform, v1.2.2, they may no longer work on v 1.1.x of the Java platform. For example, Swing classes are in a different package in the J2SDK v1.2.2 than they were in JDK v 1.1.x, and you must update the references to the Swing classes if you are migrating applications from 1.1.7 to 1.2.2. If you have many JDK-dependent applications, you may want to keep them in separate repositories.

- How many repositories are involved? In general, you need to consider all shared repositories and local Early Adopters Environment repositories. If you have *N* users of the Early Adopters Environment (all with local repositories) and *R* shared repositories, then there are *N* + *R* repositories to upgrade.

  The steps that you will need to follow when migrating depend on your situation and on your answers to the above questions.

  The following migration scenario is one of the most complicated. In this scenario, you have more than one shared repository and *N* developers with Early Adopters Environment local repositories. You want to use the new Version 3.5 repository, but also keep using all the data in your old shared repositories.

  Note: This scenario covers dealing with Early Adopters Environment (Java 2) local repositories, not JDK 1.1.x local repositories. If you want to import JDK 1.1.x local repository information into your Version 3.5 repository, follow the instructions in Section 3.2 of Part A.

  1. Upgrade EMSRV to Version 7.0. The team administrator must install EMSRV, Version 7.0. Refer to Section 2 of Part C for instructions on how to perform this task.
  2. Create backup copies of your user data.
     . Version your projects and packages. Only versioned projects and packages can be imported into the VisualAge for Java, Version 3.5 repository. Refer to your VisualAge for Java online help for versioning instructions.
     2. Save your repository to a new location outside your VisualAge for Java directory tree. The file name and path of the repository is x:\IBMVJava\ide\repository\ivj.dat, where x:\IBMVJava is the VisualAge for Java installation directory.
     3. Copy any resource files (such as properties, images or sound files) used by your Java applications to a directory outside your VisualAge for Java directory tree. By default, resource files for each VisualAge for Java project are located in subdirectories called x:\IBMVJava\ide\project_resources\projectname, where x:\IBMVJava is the VisualAge for Java installation directory and projectname is the name of the project with which the resources are associated.
  3. The team administrator performs a complete installation of VisualAge for Java, Version 3.5, including a *local* repository. The administrator then exports the entire contents of the local repository into all the shared repositories. Refer to Part C, Section 3.0 for details on how to perform this task.
  4. All Early Adopters Environment users install Version 3.5 locally, which will automatically upgrade the *N* local repositories.

  The migration process is now complete, and users can switch between a local repository or a shared repository as desired.

  Note: If you are migrating from a team environment to a local environment, you must manually export your projects from your old shared repository into your local one. As well, if you had a local repository you will have to import any projects you want to use from your old local repository into your new Version 3.5 local repository.

B.4.0 Known limitations and problems

Please also refer to the README (which can be found in the root directory of the CD) for information about pervasive known problems and limitations.

B.4.1 Known limitations and problems with Installation

B.4.1.1 Disk limitations

- Do not install to an HPFS network or local drive since Windows 98, Windows 2000, and Windows NT 4.0 have trouble handling long file names on HPFS.
- Do not install to a Novell NetWare drive. Installation will fail on a Novell NetWare drive.

B.4.1.2 Environment space (Windows 98)

On Windows 98, you should edit your config.sys file to increase the environment space to 32000. Insert the following line:

SHELL=X:\COMMAND.COM /E:32000 /P

where "X:" is your boot drive. Reboot before you install the product. You must reboot after you install the product as well before you start the IDE.

B.4.1.3 User authorization

- You must have write access to the root "\" directory of the drive where VisualAge for Java is installed to install the NetQuestion Search Server.
- When you install the product on Windows 2000, you may receive a warning message if you are not an Administrator. The message indicates that some programs may not work correctly if they are not installed by an Administrator. You should log on as an Administrator before beginning the VisualAge for Java installation.
- When a Restricted User attempts to install VisualAge for Java, Version 3.5 on Windows 2000, they may incorrectly receive a "Setup Initialization Error" with the message "Setup has detected that unInstallShield is in use. Please close unInstallShield and restart setup. Error 432." Verify that you have Administrator or Standard User access rights before attempting to install the product again.
- If you do not have Administrator rights while installing on Windows NT, the online help search function will not work.

B.4.1.4 TCP/IP considerations

- An "Autoconfigured" warning from the installation program indicates that your proxy exceptions are autoconfigured for your browser. Check with your system administrator to ensure that 127.0.0.1 is treated as a local address
- You must install and configure TCP/IP in order for IBM VisualAge for Java to install properly. For Windows 98 and Windows 2000, you must enable TCP/IP as follows:
  1. For a LAN Adapter configuration:
     - You must have DNS enabled with a valid host and domain name.
     - Your LAN DNS must resolve localhost to 127.0.0.1.
     - You cannot run disconnected with a LAN Adapter configuration.
  2. For a dial-up Adapter configuration:
     - You must have DNS disabled.
     - Your TCP/IP address must be obtained automatically.

  These configuration options will apply to all TCP/IP adapters, even though they have only been changed for this one. You will not be able to use both LAN and dial-up without reconfiguring.

  Dial-up networking TCP/IP properties for your Internet service provider (ISP) must be configured as documented by the ISP. The dial-up networking TCP/IP properties will override the properties in the dial-up Adapter TCP/IP properties configured via the Network icon in the Windows 98 or Windows 2000 Control Panel. The overriding of the properties will take place only if the dial-up Adapter TCP/IP properties are configured as above. You must not enable the DNS in the dial-up Adapter TCP/IP properties or set an IP address in the dial-up Adapter TCP/IP properties, because doing so will interfere with the dial-up networking configuration for the ISP.

  For Windows NT 4.0, you can use either of the TCP/IP configurations described above. If you are running standalone, you can also enable the Microsoft Loopback Adapter without the other two adapters.

B.4.1.6 Shell extension (Windows NT)

If you get a message that indicates that the installation program has detected a Shell Extension for Windows NT, the installation cannot proceed. You should then perform the following steps:

1. Make sure you have an Emergency Recovery Disk. Instructions for creating this are available in the Windows help documentation.
2. Invoke regedit.exe from a command prompt.
3. Expand the key \\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
4. Select the shell name in the name/data pairs for the above key. **Important!** Make a note of the data recorded for this name, because you will need it after installing IBM VisualAge for Java.
5. Select **Edit** > **Modify** from the menu bar for the shell name/data pair.

6. Set the value for the shell name to Explorer.exe. Click **OK**.
7. Select **Registry** > **Exit** from the menu bar.
8. Restart and complete IBM VisualAge for Java installation.
9. Once installation is complete, restore the previous registry entry as follows:
   a. Invoke regedit.exe from a command prompt.
   b. Expand the key \\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
   c. Select the shell name in the name/data pairs for the above key.
   d. Select **Edit** > **Modify** from the menu bar for the shell name/data pair.
   e. Restore the value for the shell name to the value that was recorded in Step 4. Click **OK**.
   f. Select **Registry** > **Exit** from the menu bar.

B.4.1.7 Recovering from failed installation

If your installation fails, you must remove any Version 3.5 files that have been installed. If the directory you intended to install VisualAge for Java in is empty, then the installation process rolled back and removed any files that were installed. You can delete the empty directory if you want to, but it is not necessary. However, if the directory contains files, then you must start the installation process again. It will open in maintenance mode and you must select to remove your partial installation of Version 3.5 before you can try to install the product again.

You will also need to delete the registry entry:

\\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\VisualAge for Java for Windows

using the following instructions:

1. Make sure you have an Emergency Recovery Disk. Instructions for creating this are available in the Windows help documentation.
2. Invoke regedit.exe from a command prompt.
3. Expand and select the key \\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\VisualAge for Java for Windows\3.5
4. Select **Edit** > **Delete** from the menu bar for this key.
5. Select **Yes** when asked to confirm deletion of the key.
6. Select **Registry > Exit** from the menu bar.

If NetQuestion was installed before the installation failed, you must delete it as well.

There are three ways to check to see if NetQuestion has been installed:

1. A folder named imnnq_*XX* (where *XX* = NT, 95, or 98, depending on your version of Windows) has been created on the root of the drive you are install VisualAge for Java, Version 3.5 to. On Windows NT and Windows 2000, the directory will be IMNnq_NT and on Windows 98 it will be IMNnq_95 or IMNNQ_98.
2. There is a registry entry for NetQuestion under HKEY_LOCAL_MACHINE\SOFTWARE\IBM
3. There are environment variables set that match following naming pattern: IMN*

To remove NetQuestion, follow these steps. If NetQuestion had been installed by another product that uses it (for example, DB2), then only perform step 2 in the steps below; you should not perform any of the other steps.

1. Delete the registry entry using regedit.
   a. Invoke regedit.exe from a command prompt.
   b. Expand and select the key: HKEY_LOCAL_MACHINE/SOFTWARE/IBM/NetQuestion
   c. Select **Edit** > **Delete** from the menu bar for this key.
   d. Select **Yes** when asked to confirm deletion of the key.
   e.Select **Registry > Exit** from the menu bar.
2. From a command prompt, go to the imnnq_XX directory and enter the command uninstnq
3. Delete the environment variables matching the IMN* pattern.
4. Delete the imnnq_XX directory.
5. On Windows 98 only, reboot.

Back up your repository and resource files if you have not already done so. Refer to Part B, Section 3.0 for information on how to perform this task.

After performing all of these steps, reboot and re-install the product.

B.4.1.8 CICS Transaction Gateway

If you have a version of the CICS Transaction Gateway installed on your machine when you install VisualAge for Java, then VisualAge will use this version instead of installing its own.

B.4.1.9 Rotated character set displayed in Japanese environment during e-connectors installation

When the e-connectors installation is run on a Japanese environment platform, the install initialization box displays a rotated character set. The initialization box is displayed for only a short period of time and signals that e-connectors are about to be installed. This does not effect the installation of the e-connectors.

B.4.2 Known limitations and problems with Uninstallation

The following is a list of items that you should be aware of during uninstallation.

B.4.2.1 Disk space

You must have at least 2MB free on your Windows system drive, and your environment variable TEMP or TMP must point to a valid temporary directory with at least 5MB free.

B.4.2.2 Uninstalling the Distributed Debugger

If you installed the Distributed Debugger, you should uninstall VisualAge for Java BEFORE you uninstall the Distributed Debugger. After you have uninstalled VisualAge for Java, you can uninstall the Distributed Debugger by going to the Debugger installation directory and running the uninstallation program.

If you have uninstalled VisualAge for Java, and you cannot uninstall the Distributed Debugger, you must delete the registry key:

HKEY_LOCAL_MACHINE/SOFTWARE/IBM/IBM Distributed Debugger/CurrentVersion/install/ParentProducts/Visual Age for Java

and then try to uninstall the Debugger again. DO NOT follow these steps if you are using the Distributed Debugger with any other products, as you will no longer be able to use the debugger after you delete the registry key.

Follow these steps:

1. Invoke regedit.exe from a command prompt.
2. Expand and select the key:
   HKEY_LOCAL_MACHINE/SOFTWARE/IBM/IBM Distributed Debugger/CurrentVersion/install/ParentProducts/Visual Age for Java
3. Select **Edit** > **Delete** from the menu bar for this key.
4. Select **Yes** when asked to confirm deletion of the key.
5. Select **Registry > Exit** from the menu bar.

As well, the value of the following registry key:

HKEY_LOCAL_MACHINE/SOFTWARE/IBM/IBM Distributed Debugger/CurrentVersion/install/refcount

should be set to 1.

B.4.2.3 Environment variables (Windows 98)

When you uninstall VisualAge for Java on Windows 98, some environment entries may be left in your autoexec.bat file. Normally these leftover entries do not cause any problems, but if you uninstall and re-install the product several times, two problems may occur. You may have conflicting path statements that can prevent the online help from working or you may run out of path space, which could prevent you from re-installing the product successfully.

To solve these problems:

1. Make a backup copy of your autoexec.bat file.
2. Determine if you have another program that requires the HTML search engine (such as DB2) on your system by following these steps:
   a) Uninstall VisualAge for Java and reboot your system.
   b) Run regedit.exe from a command prompt and expand the HKEY_LOCAL_MACHINE\SOFTWARE\ tree. If there is an IBM directory in this tree, expand it to see if there is an NetQuestion directory. If you see this directory, then you are probably using the search engine with another IBM product.
3. If you are *not* using the HTML search engine for another product, then remove any IMN or IMQ entries in your autoexec.bat file before re-installing VisualAge for Java.
4. If you *are* using the HTML search engine for another product, delete any duplicates of these entries from your autoexec.bat file:

   IMNINST
   IMNINSTSRV
   IMNNQ
   IMNNQ_95
   IMQCONFIGCL
   IMQCONFIGSRV
   Also delete the line IF EXIST X:\IMNNQ_05\IMNENV.BAT CALL IMNEV.BAT
5. Make sure that you do not remove the original entries when you remove the duplicates. If you are not sure which entries are the original ones, then you must determine where the system considers NetQuestion to be installed. Follow these steps:
   1. Start regedit.exe from a command prompt.
   2. Expand the key \\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\NetQuestion\Installation Directory
   3. The directory value inside of this key shows you the path where NetQuestion is installed. Certain environment variables must contain this directory as part of their value for NetQuestion to function correctly.

      If you find any of the above environment variables that include a directory different from the one found in the registry as part of their value, delete them.

B.4.2.4 Uninstalling NetQuestion

When you are uninstalling VisualAge for Java, NetQuestion may not be uninstalled. Problems may occur if NetQuestion is not uninstalled and you later attempt to reinstall the product.

To remove NetQuestion, follow these steps. If NetQuestion had been installed by another product that uses it (for example, DB2), then only perform step 2 in the steps below; you should not perform any of the other steps.

1. Delete the registry entry using regedit.
   a. Invoke regedit.exe from a command prompt.
   b. Expand and select the key:
   HKEY_LOCAL_MACHINE/SOFTWARE/IBM/NetQuestion
   c. Select **Edit** > **Delete** from the menu bar for this key.
   d. Select **Yes** when asked to confirm deletion of the key.
   e. Select **Registry > Exit** from the menu bar.
2. From a command prompt, go to the imnnq_*XX* directory and enter the command uninstnq. On Windows NT and Windows 2000, the directory will be IMNnq_NT and on Windows 98 it will be IMNnq_95 or IMNNQ_98.
3. Delete the environment variables matching the IMN* pattern.
4. Delete the imnnq_*XX* directory.
5. On Windows 98 only, reboot.

Part C - EMSRV

For information about installing the VisualAge for Java client, please refer to Part B of this guide. For information about setting up and administering the server, refer to the "Server setup and administration" file, emsrv70.htm, which can be found in the TeamServer\docs directory on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

You must use EMSRV, Version 7.0, if you are working with the Enterprise Edition of VisualAge for Java, otherwise, you cannot version your project resources. Refer to the online help for more information about versioning project resources.

C.1.0 Prerequisites

Please also refer to the section on "Known problems and limitations" at the end of Part C before installing EMSRV.

C.1.1 Supported Platforms

The EMSRV server is supported for the following operating system platforms:

- Novell NetWare 4.2
- Novell NetWare 5.1
- OS/2 (R) Warp Version 4.0
- OS/2 Warp Server for e-business
- Windows NT (R) Workstation Version 4.0 (with Service Pack 5)
- Windows NT Server Version 4.0 (with Service Pack 5)
- Windows (R) 2000 Professional
- Windows 2000 Professional Server
- Windows 2000 Advanced Server
- Sun Solaris (TM) Version 2.6 (with Patch 106757-05)
- Sun Solaris Version 7.0
- HP-UX Version 10.20 *
- HP-UX Version 11.0 *
- AIX (R) Version 4.3.2
- AIX Version 4.3.3
- Red Hat Linux 6.1
- Red Hat Linux 6.2

*** Note**: HP-UX is supported on 700-class workstation machines only. It has been tested on an HP-UX 9000/715/60 machine and an HP-UX 9000/782/200+ machine. Because 800-class (server) machines have a different architecture and require different binaries, EMSRV is not supported on 800-class machines.

IBM no longer supports EMSRV on Netware 4.11 or Netware 5.0, but EMSRV can be run on that platform if CLIBAUX.NLM is loaded before EMSRV.NLM is loaded. CLIBAUX.NLM is included with Novell's Support Pack 8a but is also available separately from Novell in the file CLIBAUX1.EXE, which can be found at the following location:

http://support.novell.com/cgi-bin/search/download?/pub/updates/nw/nw42/clibaux1.exe

C.1.2 TCP/IP

TCP/IP must be installed and configured on your server.

C.1.3 Novell patches required to run EMSRV

We recommend you obtain and apply the NetWare Minimum Patch List. These patch files are available at http://support.novell.com/misc/patlst.htm. You must select and apply the appropriate patches for the version of NetWare you are using.

C.1.4 Patch required to run EMSRV on Solaris

There is a bug in the Solaris, Version 2.6 implementation of PAM that prevents EMSRV from working correctly. Patch 106257-05 must be applied if you are using EMSRV on Solaris, Version 2.6. The patch is available at:

http://sunsolve.sun.com/pub-cgi/retrieve.pl?doc=fpatches%2F106257&zone_32=PAM

The specific bug which this patch fixes is:

4092227 pam_conv appdata_ptr member is not passed thru to conv() function as documented

The patch is not required if you are working with Version 7.0 of Solaris.

C.1.5 File systems supported

EMSRV has been tested and certified with the following file systems:

NetWare

- NWFS (NetWare File System) with DOS namespace
- NWFS with OS/2 or LONG namespace
- NSS (Novell Storage Services) with DOS namespace
- NSS with OS/2 or LONG namespace

OS/2

- HPFS (High-Performance File System)
- FAT

Windows NT and Windows 2000

- NTFS (New Technology File System)
- FAT32
- FAT

Solaris

- UFS (Unix File System)

HP-UX

- VxFS (Veritas File System)

AIX

- JFS (Journaled File System)

Linux

- EXT2FS (Second Extended File System)

EMSRV only supports locally-mounted file systems.

C.2.0 Installation

This section contains instructions for installing the EMSRV repository server program and a shared repository. For instructions for starting the server, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.

C.2.1 Installing EMSRV for Windows(R)

To install the EMSRV repository server program and a shared repository on a Windows NT or Windows 2000 server, follow these steps:

1. Copy the following three files from the TeamServer directory to the desired directory on the server:
   - emsrv.exe
   - emadmin.exe
   - emsrvmsg.dll

   Place these files in a subdirectory that is part of your PATH or create a subdirectory and add it to your PATH. We suggest placing them in a location with a lot of free space because the emsrv.log file will be written to the subdirectory in which you place these files.
2. Extract the following from the ide.zip file into the desired directory:
   - ivj.dat (repository file)
   - ivj.dat.pr directory (project resources directory)

The ide.zip file is located in the ivj35\backup directory, which is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

This directory is where you intend to store shared source code repositories. When you start the server later, you will specify this subdirectory as the EMSRV working directory, using the **-W** EMSRV startup parameter when you start the server.

EMSRV must have full rights to read, write and search the entire tree of directories in the ivj.dat.pr directory. The ivj.dat.pr directory should always be copied to the same directory as ivj.dat, or you will not be able to access your project resources.

Paths to repositories accessed via EMSRV for Windows NT must be specified at the command line as a FAT, NTFS or UNC path, relative to the EMSRV working directory. It is not possible to access a repository residing on a remote volume. Examples of this include drives shared using Microsoft (R) Networking that reside on other machines, and NetWare volumes accessible by the Gateway (and Client) Services for NetWare.

You may choose to rename the repository file, for example to team1.dat. If you rename the repository after copying it onto the server, you must rename the project resources directory accordingly. For example, if you rename the repository to team1.dat, you must change the name of the project resources directory to team1.dat.pr.

Team members will need to specify the repository file name when they install the VisualAge for Java client code. They also will need to specify the path on the server machine.

1. If you wish to enable password checking through the use of a passwd.dat file, copy the passwd.dat file from the TeamServer directory to the same directory where you copied ivj.dat. For more information on the types of password checking available, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.
2. Verify that TCP/IP is installed and correctly bound to a LAN adapter. You can verify the binding by using the **ping** command (for example, *ping IP address/host name)* to communicate with the server from a workstation on the LAN.
3. Proceed to install EMSRV in the Windows NT registry (optional), authorize the EMSRV user, and start EMSRV. These topics are described in the "Server setup and administration" file, emsrv70.htm.

C.2.1.1 Installing EMSRV as a service in the Windows registry

If you prefer to start EMSRV as a service rather than from a command prompt, you can install EMSRV in the Windows registry. There are two advantages to installing EMSRV as a service:

- You can specify automatic startup so that EMSRV will start whenever the repository server is booted.
- You can specify the default settings that you want EMSRV to use. For example, you might want to ensure that password validation is always enabled.

Tip: If EMSRV is started as a service, the default EMSRV working directory is the Windows NT or Windows 2000 system32\directory. *It is recommended that you change this default* by using the -W parameter when you install EMSRV as a service in the Windows NT or Windows 2000 registry.

To install EMSRV as a service:

1. From a command prompt, change to the directory where the EMSRV executable program is installed.
2. Issue the command emsrv -install [parameter2] [parameter3] ... The first parameter must be -install; the others are the EMSRV startup parameters that you have chosen for your environment.

   The following is an example of this command:

   emsrv -install -u joe -p donttell -W j:\sharedrep -rn

   This example installs EMSRV as a service in the Windows registry, with joe as the EMSRV user name and donttell as joe's password. By default, the EMSRV working directory will be j:\sharedrep and native password validation will be enforced.

   A message confirms that EMSRV has been installed.
3. Steps a and b are slightly different for Windows NT and Windows 2000.
   a) From the Windows NT Control Panel, double-click **Services**. The Services dialog box will appear. Select

EMSRV from the list of services.
b) From the Windows 2000 Control Panel, double-click **Administrative Tools**. Double-click **Services**. Double-click **EMSRV**.
You can start it manually from here. EMSRV is now installed as a service in the registry and the necessary DLLs have been copied to the system directory.

4. In the Startup Parameters text box, type the EMSRV startup parameters that you want to use. If you are specifying the working directory for EMSRV to use, you must type an *extra backslash* for each backslash in the path. Here is an example:

   -u emsrvacc -p secret -W d:\\javateam
5. Click **Start**. A message will appear, informing you that EMSRV is starting.

For more information about starting EMSRV, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.

The parameters that you provided will be used, by default, whenever EMSRV is started. You can also override or add to these parameters if you start EMSRV manually from the **Services** icon of the Windows Control Panel.

C.2. 2 Installing EMSRV for NetWare

EMSRV for NetWare requires the NetWare TCP/IP stack (TCPIP.NLM) to be loaded and configured on the server. When EMSRV for NetWare is started, if TCPIP.NLM is not loaded, it will automatically be loaded. The NWSNUT.NLM files, which are required by the EMSRV for NetWare user interface will also automatically be loaded.

To install the EMSRV repository server program and a shared repository on NetWare, follow these steps:

1. From the TeamServer directory, copy the following program files to the SYS:\SYSTEM directory on the server:
   - emsrv.nlm
2. Extract the following from the ide.zip file into the desired directory on the server:
   - ivj.dat (repository file)
   - ivj.dat.pr directory (project resources directory)

   The ide.zip file is located in the ivj35\backup directory, which is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

When you start the server later, you will specify this subdirectory as the EMSRV working directory, using the **-W** EMSRV startup parameter. EMSRV must have full rights to read, write and search the entire tree of directories in the ivj.dat.pr directory. The ivj.dat.pr directory should always be copied to the same directory as ivj.dat, or you will not be able to access your project resources.

You may choose to rename the repository file, for example, to team1.dat. If you rename the repository after copying it onto the server, you must rename the project resources directory accordingly. For example, if you rename the repository to team1.dat, you must change the name of the project resources directory to team1.dat.pr.

Team members will need to specify the repository file name and location when they install the VisualAge for Java client code.

1. Copy the sample password file, passwd.dat, from the TeamServer directory to the same directory where you copied ivj.dat.
2. Verify that the NetWare TCPIP.NLM is loaded and correctly bound to a LAN adapter. You can verify the binding by using the **ping** command (for example, *ping IP address/host name)* utility to communicate with the product from a workstation on the LAN.
3. For instructions for starting the product, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.

C.2.3 Installing EMSRV for OS/2 Warp

Note: OS/2 is no longer supported as a development platform. Refer to Part E for details.

To install the EMSRV repository server program and a shared repository on an OS/2 server, follow these steps:

1. Copy the following three files from TeamServer directory to the desired directory on the OS/2 computer:

- emsrv.exe
- emadmin.exe
- emclient.mod

Place these files in a subdirectory that is part of your PATH or create a subdirectory and add it to your PATH. We suggest placing them in a location with a lot of free space because the emsrv.log file will be written to the subdirectory you place these files in.

2. Extract the following from the ide.zip file into the subdirectory where you placed the files you copied in step 1:
   - ivj.dat (repository file)
   - ivj.dat.pr directory (project resources directory)

   The ide.zip file is located in the ivj35\backup directory, which is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

   This subdirectory is where you intend to store shared source code repositories. When you start the server later, you will specify this subdirectory EMSRV working directory, using the **-W** EMSRV startup parameter.

   EMSRV must have full rights to read, write and search the entire tree of directories in the ivj.dat.pr directory. The ivj.dat.pr directory should always be copied to the same directory as ivj.dat, or you will not be able to access your project resources.

   You may choose to rename the repository file, for example to team1.dat. If you rename the repository after copying it onto the server, you must rename the project resources directory accordingly. For example, if you rename the repository to team1.dat, you must change the name of the project resources directory to team1.dat.pr.

   Team members will need to specify the repository file name when they install the VisualAge for Java client code.
3. If you wish to enable password checking through the use of a passwd.dat file, copy the passwd.dat file from the TeamServer directory to the same directory where you copied ivj.dat. For more information on the types of password checking available, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.
4. Verify that TCP/IP is installed and correctly bound to a LAN adapter. You can verify the binding by using the **ping** command (for example, *ping IP address/host name)* to communicate with the server from a workstation on the LAN.
5. To start the server, refer to the instructions for starting EMSRV on OS/2, in the "Server setup and administration" file emsrv70.htm.

C.2.4 Installing EMSRV for AIX

In the steps below, "EMSRV user" refers to the user that starts the EMSRV program.

You must copy the following files from the TeamServer directory to your machine:

- emsrv
- emadmin
- emdevnum

Place these files in a subdirectory that is part of your PATH or create a subdirectory and add it to your PATH. We suggest placing them in a location with a lot of free space because the emsrv.log file will be written to the subdirectory in which you place these files.

Follow these steps for setting up EMSRV on an AIX machine:

1. The EMSRV user or the system administrator (root) sets aside disk space to store repositories.
2. An initial repository can be set up by extracting the following from the ide.zip file to the disk space set aside in step 1.
   - ivj.dat (repository file)
   - ivj.dat.pr directory (project resources directory)

   The ide.zip file is located in the ivj35\backup directory, which is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

EMSRV must have full rights to read, write and search the entire tree of directories in the ivj.dat.pr directory. The ivj.dat.pr directory should always be copied to the same directory as ivj.dat, or you will not be able to access your project resources. The directories must have rw and x (search) bits set for the EMSRV user.

3. Change the file owner to the EMSRV user or the system administrator. If you plan to have more than one repository, make copies of ivj.dat using different names but retain the same suffix (.dat). If you make duplicate copies of the repository, you must make duplicate copies of the ivj.dat.pr directory and change the name to match the repository it is associated with. For example, if you make a duplicate copy called "team.dat", you must make a duplicate project resources directory called "team.dat.pr"
   The EMSRV user or the system administrator should change to the directory where the repositories are stored and start the EMSRV program, using the appropriate flags. For more information on EMSRV flags, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.
4. The EMSRV user or the system administrator tells the team members the location and name of the team repository. This information is needed when team members install their client code.
5. To start the server, refer to the instructions in the "Server setup and administration" file emsrv70.htm.

Root access on UNIX platforms is required to authenticate users. EMSRV does NOT need to be started by the root user to accomplish this. Doing so would compromise security as EMSRV would then have complete access to all file systems.

Instead, you should change the owner of the EMSRV executable to 'root' and set the SUID bit of the executable. This can be accomplished as follows:

```
chown root emsrv
chmod u+s emsrv
```

When EMSRV attempts to authenticate a user, it will temporarily change the authority of the running EMSRV process to be the authority of the owner of the executable. Once authentication is complete, the authority of the running EMSRV process will be changed back to that of the user that started EMSRV. This happens on a per-process (per-client) basis so while a client is being authenticated, only the process serving that client has temporary root access.

Root access for authentication is required regardless of how EMSRV actually implements authentication. Interfaces such as PAM only provide a common API to permit applications to support multiple authentication methods, configuration specific to each method of authentication must still be correct.

C.2.5 EMSRV for HP-UX or Solaris

In the steps below, "EMSRV user" refers to the user that starts the EMSRV program.

You must copy the following files from the TeamServer directory to your machine:

For HP-UX:

- emsrv
- emsrv.shadow
- emadmin
- emdevnum

For Solaris:

- emsrv
- emadmin
- emdevnum
- pam.conf

On Solaris, PAM must be correctly configured on a machine running EMSRV otherwise it will not even be possible to shutdown EMSRV using EMADMIN. The /etc/pam.conf file must detail the 'emsrv' service. An example pam.conf file is included with this release.

Place these files in a subdirectory that is part of your PATH or create a subdirectory and add it to your PATH. We suggest placing them in a location with a lot of free space because the emsrv.log file will be written to the subdirectory in which you place these files.

Follow these steps for setting up EMSRV on a Solaris or HP-UX machine:

1. The EMSRV user or the system administrator (root) sets aside disk space to store repositories.
2. An initial repository can be set up by extracting the following from the ide.zip file to the disk space set aside in step 1.
   - ivj.dat (repository file)
   - ivj.dat.pr directory (project resources directory)

   The ide.zip file is located in the ivj35\backup directory, which is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

   EMSRV must have full rights to read, write and search the entire tree of directories in ivj.dat.pr. ivj.dat.pr should always be copied to the same directory as ivj.dat, or you will not be able to access your project resources. The directories must have rw and x (search) bits set for the EMSRV user.
3. Change the file owner to the EMSRV user or the system administrator. If you plan to have more than one repository, make copies of ivj.dat using different names but retain the same suffix (.dat). If you make duplicate copies of the repository, you must make duplicate copies of the ivj.dat.pr directory and change the name to match the repository it is associated with. For example, if you make a duplicate copy called "team.dat", you must make a duplicate project resources directory called "team.dat.pr"
   The EMSRV user or the system administrator should change to the directory where the repositories are stored and start the EMSRV program, using the appropriate flags. For more information on EMSRV flags, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.
4. The EMSRV user or the system administrator tells the team members the location and name of the team repository. This information is needed when team members install their client code.
5. To start the server, refer to the instructions in the "Server setup and administration" file, emsrv70.htm.

Root access on UNIX platforms is required to authenticate users. EMSRV does NOT need to be started by the root user to accomplish this. Doing so would compromise security as EMSRV would then have complete access to all file systems.

Instead, you should change the owner of the EMSRV executable to 'root' and set the SUID bit of the executable. This can be accomplished as follows:

chown root emsrv
chmod u+s emsrv

When EMSRV attempts to authenticate a user, it will temporarily change the authority of the running EMSRV process to be the authority of the owner of the executable. Once authentication is complete, the authority of the running EMSRV process will be changed back to that of the user that started EMSRV. This happens on a per-process (per-client) basis so while a client is being authenticated, only the process serving that client has temporary root access.

Root access for authentication is required regardless of how EMSRV actually implements authentication. Interfaces such as PAM only provide a common API to permit applications to support multiple authentication methods, configuration specific to each method of authentication must still be correct.

C.2.6 EMSRV for Linux

In the steps below, "EMSRV user" refers to the user that starts the EMSRV program.

You must copy the following files from the TeamServer directory to your machine:

- emsrv
- emadmin
- emdevnum
- pam/emsrv

Place these files in a subdirectory that is part of your PATH or create a subdirectory and add it to your PATH. We suggest placing them in a location with a lot of free space because the emsrv.log file will be written to the subdirectory in which you place these files.

PAM must be correctly configured on a machine running EMSRV otherwise it will not even be possible to shutdown EMSRV using EMADMIN. The PAM configuration file must be copied to /etc/pam.d/emsrv. A sample PAM configuration file is included with this release.

Follow these steps for setting up EMSRV on a Linux machine:

1. The EMSRV user or the system administrator (root) sets aside disk space to store repositories.
2. An initial repository can be set up by extracting the following from the ide.zip file to the disk space set aside in step 1.
   - ivj.dat (repository file)
   - ivj.dat.pr directory (project resources directory)

   The ide.zip file is located in the ivj35\backup directory, which is located on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java.

   EMSRV must have full rights to read, write and search the entire tree of directories in the ivj.dat.pr directory. The ivj.dat.pr directory should always be copied to the same directory as ivj.dat, or you will not be able to access your project resources. The directories must have rw and x (search) bits set for the EMSRV user.
3. Change the file owner to the EMSRV user or the system administrator. If you plan to have more than one repository, make copies of ivj.dat using different names but retain the same suffix (.dat). If you make duplicate copies of the repository, you must make duplicate copies of the ivj.dat.pr directory and change the name to match the repository it is associated with. For example, if you make a duplicate copy called "team.dat", you must make a duplicate project resources directory called "team.dat.pr"
   The EMSRV user or the system administrator should change to the directory where the repositories are stored and start the EMSRV program, using the appropriate flags. For more information on EMSRV flags, refer to the "Server setup and administration" file emsrv70.htm, which can be found in the TeamServer\docs directory.
4. The EMSRV user or the system administrator tells the team members the location and name of the team repository. This information is needed when team members install their client code.
5. To start the server, refer to the instructions in the "Server setup and administration" file emsrv70.htm.

Root access on UNIX platforms is required to authenticate users. EMSRV does NOT need to be started by the root user to accomplish this. Doing so would compromise security as EMSRV would then have complete access to all file systems.

Instead, you should change the owner of the EMSRV executable to 'root' and set the SUID bit of the executable. This can be accomplished as follows:

```
chown root emsrv
chmod u+s emsrv
```

When EMSRV attempts to authenticate a user, it will temporarily change the authority of the running EMSRV process to be the authority of the owner of the executable. Once authentication is complete, the authority of the running EMSRV process will be changed back to that of the user that started EMSRV. This happens on a per-process (per-client) basis so while a client is being authenticated, only the process serving that client has temporary root access.

Root access for authentication is required regardless of how EMSRV actually implements authentication. Interfaces such as PAM only provide a common API to permit applications to support multiple authentication methods, configuration specific to each method of authentication must still be correct.

C.3.0 Migration

You must perform the steps in section 3.1 before you can perform the steps in section 3.2.

C.3.1 Migrating from Version 6.x of EMSRV to Version 7.0

If you currently have Version 6.x of EMSRV installed and want to install Version 7.0 of EMSRV, you can either uninstall version 6.x of EMSRV and install Version 7.0 or keep your old version of EMSRV and install EMSRV 7.0.

You must install Version 7.0 to work with VisualAge for Java, Version 3.5.

To move from EMSRV, Version 6.x to EMSRV, Version 7.0, follow these steps:

1. Back up your repository
2. Shut down EMSRV 6.x.
3. Install EMSRV 7.0.

4. Start EMSRV 7.0.

EMSRV 7.0 is compatible with EMSRV 6.x. For example, if you are working in a previous edition of VisualAge for Java (that uses EMSRV 6.x), you can connect from your previous version to a shared repository running under EMSRV 7.0.

C.3.2 Migrating a shared repository from VisualAge for Java, Version 2.0/3.0x

You can upgrade your Version 2.0 or 3.0x (JDK 1.1-based) or 3.0x, Early Adopters (JDK 1.2 based) shared repository to work with VisualAge for Java, Version 3.5. In the steps below, the team administrator performs a complete installation of VisualAge for Java, Version 3.5 using a local repository. The administrator then exports the entire contents of the local repository into all the shared repositories.

To upgrade an existing repository on the server to work with VisualAge for Java, Version 3.5, follow these steps:

1. Install VisualAge for Java, Version 3.5 as a full install with a *local* repository. Refer to Section 2.0 in Part B for instructions on how to perform this task.
2. Start the Version 3.5 IDE. You will be connected to the local repository.
3. In the Workbench, select **File** > **Export**, then select the **Repository** radio button and click **Next**. Select **Shared repository with EMSRV server.** Type the IP address or host name of the server in the **Shared repository with EMSRV server address** field.
4. Click **Browse** to locate your shared repository or type the shared repository's path and file name in the **Repository name** field.
5. Select **Projects.** Click **Details** to see a list of versioned projects that can be exported from the source repository.
6. Select all the project editions and export them.
7. Click **Finish**.

All of the projects are copied into your shared repository. Your project resource files are also exported. If the repository you are exporting to is called sample.dat, then your project resources are exported to a folder called sample.dat.pr.

C.4.0 Preparing for team development

At this point, you have already installed the repository server programs and a shared source code repository. To finish setting up the team development environment, you must start the server, connect to it from a VisualAge for Java client, and add users to the repository user list.

If you have already installed the VisualAge for Java client on a workstation, you can refer to the online help for more information about setting up the team development environment. Otherwise, refer to the team.pdf in the pdf directory. The pdf directory is on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. If you did not download the part containing the PDFs, you will not have this directory.

In the following instructions, it is assumed that the shared repository that is installed on the server is called ivj.dat. When starting the repository server program, the administrator should provide the path of the ivj.dat file as the EMSRV working directory.

C.4.1 Preparing the team server

Before team developers can work with the shared repository, the administrator must set up the VisualAge for Java server and start the EMSRV repository server program. If some developers wish to start using VisualAge for Java, Version 3.5 before the server is ready, they can first install as standalone users and then connect to the shared repository later.

C.4.2 Testing a client connection

To confirm that the server has been successfully started, the administrator should connect to the shared repository from a VisualAge for Java, Enterprise Edition, Version 3.5 client. This action will confirm that the server's TCP/IP connection is working properly, that EMSRV has been started with the correct parameters, and that the administrator is aware of what server information must be provided during client installation.

For information on connecting to a shared repository, refer to "Connecting to a shared repository" in the VisualAge for Java online help or the team.pdf file. The team.pdf file is in the pdf directory, which is on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. If you did not download the part containing the PDFs, you will not have this directory.

C.4.3 Adding users to the repository user list

When a client first connects to the shared repository, the user is prompted to provide a workspace owner name. The user cannot start the IDE without first selecting a valid workspace owner name from a list of repository users.

By default, VisualAge for Java, Version 3.5 has a user called Administrator in its repository user list. Each user could initially choose Administrator as the workspace owner name; however, it is strongly recommended that every user provide a unique name to connect to the server, right away. In the VisualAge for Java team development environment, change control is based on defined user roles, which means each developer should be uniquely identified. To meet this objective, the administrator must add everyone to the list of repository users. (The only VisualAge for Java user who can add other names to a repository user list is Administrator.) The unique name belonging to each user should correspond to a system user name if native password verification is used.

For information on adding users to the repository list, refer to in the VisualAge for Java team online help or the team.pdf file in the pdf directory. The pdf directory is on the product CD or in your temporary directory (where you extracted your parts to) if you have an electronic version of VisualAge for Java. If you did not download the part containing the PDFs, you will not have this directory.

Now that the server is set up and ready, users should proceed to install their VisualAge for Java clients. Information on installing VisualAge for Java team clients can be found in Part B of this guide.

C.5.0 Limitations and known problems (EMSRV)

C.5.1 WAN support

IBM has not tested VisualAge for Java client/server connections across a wide area network (WAN). WAN connection is not supported or recommended.

You must have a high-speed local area network to run VisualAge for Java. Some users have successfully connected over wide-area networks which are rated at speeds less than 10 megabits/second, but such networks are not supported.

C.5.2 TCP/IP connection limitations

The default limit for client connections to an EMSRV is 256. This limit can be changed using the -M command-line option when you start EMSRV.

The number is bounded primarily by memory, but some TCP/IP stacks will run out of stream sockets before the limit of memory is reached. Typically, this number is greater than one hundred, but it varies with each stack.

C.5.3 Detecting unexpectedly dropped connections

EMSRV uses the TCP/IP KEEPALIVE timer to detect connections which have been unexpectedly dropped when a client has crashed or been rebooted. Some TCP/IP stacks allow the KEEPALIVE timeout to be changed. Typically, the default setting is 120 minutes.

EMADMIN may be used to list the connections and identify a connection that has not interacted with the server for a long time by the time of the last request. Such a connection may then be closed using the EMADMIN STOP command and the -kx option.

C.5.4 Interchanging different versions of EMSRV and EMSRV utilities

VisualAge for Java, Version 3.5, includes version 7.0 of the EMSRV and EMADMIN utilities.

You must use EMADMIN 7.0 with EMSRV 7.0. EMADMIN 7.0 will not work correctly with releases of EMSRV before 7.0.

C.5.5 Long filename support for the file prompter on NetWare

When using EMSRV for NetWare, long filenames may only be created and viewed on NetWare volumes to which

the LONG or OS/2 namespace has been added. Long filename support is required for the stored resource management feature used by VisualAge for Java, Version 3.5.

C.5.6 Long filename support on Windows

When using EMSRV for Windows NT or Windows 2000, long filenames may be created and viewed on FAT, FAT32, and NTFS volumes. Unlike earlier FAT implementations, the Windows NT or Windows 2000 FAT implementation has Long File Name support. Long filename support is required for the stored resource management feature used by VisualAge for Java, Version 3.5.

C.5.7 Long filename support on OS/2

When using EMSRV for OS/2, long filenames may be created and viewed on HPFS volumes. Long filename support is required for the stored resource management feature used by VisualAge for Java, Version 3.5.

C.5.8 PAM Limitations

On Linux and Solaris platforms, authentication is implemented using PAM (Password Authentication Modules). Although this would theoretically allow the use of any PAM (module) with EMSRV by changing the relevant PAM configuration file, in practice this is not possible.

EMSRV does not converse with clients in a manner that is entirely compatible with the PAM architecture. As a result, EMSRV authentication will only work where the module prompts initially for a text password (supplied initially by the client).

C.5.9 Large file support on UNIX Platforms

C.5.9.1 Introduction

On AIX, Solaris, and HP-UX, large file support needs to be enabled for both the server hosting a repository and the user starting the EMSRV process.

C.5.9.2 Building a filesystem on AIX

'-o bf=true' must be used with mkfs.

C.5.9.3 Building a Filesystem on Solaris

UFS file systems automatically have Large File Support

C.5.9.4 Building a Filesystem on HP-UX

'-o largefiles' must be used with newfs.

C.5.9.5 User Limits

On AIX, Solaris, and HP-UX, limits for the user starting the EMSRV process must be set. The -Hf and -Sf options require arguments specifying the number of 512 byte blocks. The following commands should be appropriate to enable up to 16 GB repositories:

ulimit -Hf 33554432
ulimit -Sf unlimited

C.5.10 Passwords containing non-ASCII characters cannot be used to authenticate with EMSRV

Due to a bug in the Microsoft C run-time libraries, any password containing non-ASCII characters entered in response to the prompt:

'Enter the password of the user who started EMSRV'

will not be correctly interpreted. The workaround is to supply the password with the -p option when running EMADMIN.

C.5.11 Menus and windows appear with garbage characters when running on Japanese NetWare

The EMSRV for NetWare NLM uses Novell's NLM User Interface Developer Components (NWSNUT). When running on Japanese NetWare, graphics characters used in the NWSNUT menus and windows are not available and will appear as 'garbage' characters. This is not a bug in the EMSRV NLM nor in NetWare, but rather it is a limitation of the Shift-JIS code page.

C.5.12 EMADMIN does not copy stored resources directory

When EMADMIN is used to copy a VisualAge for Java 3.5 repository, it does not copy the corresponding stored project resources directory.

C.5.13 Memory reported By EMSRV on NetWare and Windows NT/2000 platforms is not rounded up

The number of MB of installed system memory that is reported by EMSRV on NetWare and Windows NT/2000 machines is always rounded down to the nearest integer.

Part D. Component-Specific Migration Information

Please refer to section 14.0 for important Swing class migration information.

D.1.0 CICS Transaction Server (CICS TS)

This version of VisualAge for Java does not support the CICS(R) Transaction Server. The classes required to support CICS TS 1.3 and below are not included with this version. Any CICS TS applications that you attempt to migrate from earlier versions of VisualAge for Java will not work in Version 3.5 and should be deleted from your Version 3.5 workspace and repository.

If you want to work with CICS TS 1.3 or below, you must continue to use a previous version of VisualAge for Java. For the time being, you must also use a previous version of VisualAge for Java if you want to use the JCICS interface or need CICS support for IIOP. We do not support the CICS Transactions server because it is based on JDK 1.1.x.

D.2.0 Data Access Beans

Data Access Beans use Swing components and interfaces. Any applications that use the beans must be migrated from the old (JDK 1.1.x) Swing packages to the new (J2SDK v.1.2.2.) Swing packages. To do this, select the affected classes and packages after installing VisualAge for Java, Version 3.5, open the Fix/Migrate SmartGuide; and select the **Include JDK1.2 renamed packages** check box. This will add the appropriate From/To entries for Swing and enable you to automatically migrate Swing references to the Java 2 SDK. Any errors that occur while you are migrating will be logged in the Log window.

For more information on how to properly migrate your applications, refer to the Visual Composition Editor online help file "Fix/migrate guidelines for class or package references" and the related task file "Repairing class or package references".

D.3.0 Data Access Builder

The Data Access Builder is no longer included in VisualAge for Java. If you want to keep using the Data Access Builder, you will have to continue working in a previous version of VisualAge for Java.

There is no new feature in VisualAge for Java, Version 3.5 that directly replaces the functionality previously provided by the Data Access Builder, but there are three components in VisualAge for Java that provide similar functionality - Data Access Beans, Persistence Builder and the Enterprise JavaBeans Development Environment. Each feature offers a different approach to creating data access classes.

You cannot migrate your code over to VisualAge for Java, Version 3.5 and reuse it in any of these tools. You will have to recreate your applications if you want to use them in Version 3.5. Use the feature that best suits your main focus in code development and what you want your applications to do.

A comparison of the Data Access Builder, Data Access Beans, and Persistence Builder is provided as an appendix to this guide.

D.4.0 EJB Development Environment

D.4.1 Migrating enterprise beans from VisualAge for Java, Version 2.0, Enterprise Update

If you created enterprise beans in VisualAge for Java, Version 2.0, Enterprise Update, and you want to use them in VisualAge for Java, Version 3.5, you must follow the steps in Scenario 1 or Scenario 2 below, depending on whether you are importing from a repository (recommended) or from a JAR file.

Scenario One - Importing from a repository

If you are importing your beans from a repository follow these steps:

1. Add the projects that contain your enterprise beans, schemas, and maps from the repository into the workspace. Error icons will appear beside some of the loaded classes.
2. Create an open edition of each of the projects. Also create an open edition of any of the packages containing old generated classes.
3. Use the Schema browser and the Map browser to load all available schemas and maps into the workspace, then regenerate the deployed classes.
4. If you did not create and save a schema and map, then create a default schema and map, and regenerate your deployed code.

You can find more information on how to perform these steps in the VisualAge for Java online help for the EJB Development Environment.

Scenario Two - Importing from a JAR file

If you exported your enterprise beans to a JAR file, follow these steps:

1. Import the JAR file into your VisualAge for Java, Version 3.5 workspace.
2. Error icons will appear beside many of the imported classes, because the JAR file will not contain the required maps.
3. Regenerate the schemas, maps, deployed classes, and any test clients you have.

You can find more information on how to perform these steps in the VisualAge for Java online help for the EJB Development Environment.

D.4.2 Migrating enterprise beans from VisualAge for Java, Version 3.0

If you have an existing enterprise bean that has deployed code that was generated using VisualAge for Java, Version 3.0, and you now want to use VisualAge for Java, Version 3.5 to set method-level attributes for it, you must explicitly delete the existing deployed code for the enterprise bean and then regenerate it.

To migrate the beans code, perform the following steps. They *must* be followed in order.

1. Import the project into the workspace.
2. Create an open edition of the project. Also create an open edition of any of the packages containing deployed code you want to delete.
3. In the Workbench, click the EJB tab.
4. In the Enterprise Beans pane, select the enterprise bean or group for which you want to delete the deployed code.
5. Select **EJB > Delete**.
6. Click **Deployed Code** to delete the deployed code.
7. You must then migrate the associations. See section 4.4 for details on how to perform this task.
8. Check to make sure you bean class and bean class parents do not contain any errors.
9. Regenerate your access beans.
10. Regenerate the deployed code. Any TestClient class generated in a previous release of VisualAge for Java will automatically be removed from the EJB Types pane. If the TestClient class was generated by the EJB Development Environment tool, you may also delete the class from the Projects page in the Workbench. The new test client in Version 3.5 does not require generated code.

D.4.3 Migrating enterprise beans from VisualAge for Java, Version 3.02

If you have an existing enterprise bean that has deployed code that was generated using VisualAge for Java, Version 3.02, and you now want to work with that project in 3.5 perform the following steps. They *must* be followed in order.

1. Import the project into the workspace.

2. Create an open edition of the project. Also create an open edition of any of the packages containing deployed code you want to delete.
3. In the Workbench, click the EJB tab.
4. In the Enterprise Beans pane, select the enterprise bean or group for which you want to delete the deployed code.
5. Select **EJB > Delete**.
6. Click **Deployed Code** to delete the deployed code.
7. You must then migrate the associations. See section 4.5 for details on how to perform this task.
8. Check to make sure you bean class and bean class parents do not contain any errors.
9. Regenerate your access beans.
10. Regenerate the deployed code. Any TestClient class generated in a previous release of VisualAge for Java will automatically be removed from the EJB Types pane. If the TestClient class was generated by the EJB Development Environment tool, you may also delete the class from the Projects page in the Workbench. The new test client in Version 3.5 does not require generated code.

D.4.4 Migrating associations from VisualAge for Java, Version 3.0

When you add, edit, or delete an association in an EJB group that was created in Version 3.0, VisualAge for Java automatically converts all associations in the EJB group to the new association format. To complete the migration process, make the following changes manually:

- Add a call to this._initLinks() in the bean-class methods ejbCreate (all variations), ejbActivate, and ejbLoad. You must also remember to add this call to any future variations of ejbCreate that are added to the bean class.
- Add a call to this._removeLinks() in the bean-class method ejbRemove.

These methods were not converted automatically because of the high probability of handwritten modifications in these methods. VisualAge for Java will add the calls automatically when new beans are created in Version 3.5.

If you import a Version 3.0-or-older CMP entity bean into an EJB group whose associations have already been migrated and then add a new bean that inherits from the imported CMP entity bean, the new bean's bean class will display red Xs in several methods. This is because the imported bean's bean class will be missing the _initLinks, _getLinks,and _removeLinks methods. You must add these methods to the imported bean's bean class by hand.

When you are ready to deploy your enterprise bean code to a production application server, you should ensure that you also deploy the new run-time JAR file that contains the run-time code required by both associations and access beans. This JAR file should be deployed to your application server and should be included in the application server's classpath. Additional information about the run-time JAR file is found in the EJB Development Environment online help.

D.4.5 Migrating associations from VisualAge for Java 3.02

When you first open an EJB group with associations that were created in Version 3.02 of VisualAge for Java, the group will contain error icons next to generated link classes. When you add, edit, or delete an association in this kind of EJB group, VisualAge for Java automatically repairs the link classes for all associations in the EJB group. If you are not planning to make any changes to the association, you will still need to select the association in the **Properties** pane of the EJB page, and select **Edit** from its popup menu to open the association editor. You should then click **OK** to complete the migration process.

VisualAge for Java will automatically remove some association-related methods that were generated in VisualAge for Java 3.02. The methods that are removed are methods that, given the characteristics of the association, would not work correctly in Version 3.5. For example, if a role is part of the primary key, the set method for that role is not valid. The method would have been automatically generated in Version 3.02 and cannot be generated in Version 3.5.

D.4.6 Migrating generated test classes

If you attempt to migrate any generated test client classes from previous versions of VisualAge for Java to VisualAge for Java, Version 3.5, you will find that the classes will contain errors and will not work. However, since the test client employed in Version 3.5 does not employ any generated classes, you can safely delete your old generated test client classes if you plan to work solely with Version 3.5.

D.5.0 Enterprise Access Builder(EAB) and e-connectors

If you migrate your EAB applications from a previous version of VisualAge for Java to Version 3.5, you may want to regenerate your Records and Commands. They will perform better in Version 3.5 if you regenerate them.

If your EAB Command was created in Version 2.0x, you can not edit it in the Command Editor. However, you can edit it in the Visual Composition Editor. The current version of the runtime is backwards compatible, so you can run Version 2.0x Commands with it.

While most previous versions of VisualAge for Java can co-exist with Version 3.5, coexistence of different e-connector levels is not supported. If you have e-connectors installed, you need to follow one of the two following migration scenarios in order to successfully migrate from a previous version of VisualAge for Java to Version 3.5

The difference between the two migration scenarios is the stage at which the connectors are migrated.

In Scenario 1 the connectors are migrated during the initial installation process. After completing the steps in this scenario, you will have VisualAge for Java, Version 3.0x without connectors, coexisting with VisualAge for Java, Version 3.5 with connectors.

In Scenario 2, the connectors are migrated after the initial migration process. After completing the steps in this scenario, you will have VisualAge for Java, Version 3.5 without the connectors coexisting with VisualAge for Java, Version 3.0x, with the connectors. Later, you can uninstall the Version 3.0x connectors and install the Version 3.5 connectors.

Migration scenario 1:

1. Version all the Version 3.0x projects that contain connectors or EAB code.
2. Export these projects and related resources to a directory outside the VisualAge for Java install tree.
3. Delete all these projects from the 3.0x workspace.
4. Delete all CICS, Host-On-Demand, IBM Enterprise Access Builder, Encina(R), IMS(TM) and MQSeries(R) features from the workspace
5. Exit the Version 3.0x IDE.
6. Uninstall the connectors for VisualAge for Java Version 3.0x by following these steps:
    1. From the Windows Start Menu, select **Start > Settings > Control Panel**.
    2. Double-click **Add/Remove Programs.** Select the **Install/Uninstall** tab.
    3. Select **IBM Connectors.** Click **Add/Remove**.
    4. Confirm that you want to uninstall the connectors and their related components.
    5. Click **OK** in the Remove Programs window.
    6. To prevent conflicts and errors with future installations of connectors, the environment variables must not contain any reference to the removed connectors. To edit the environment variables:

        Windows NT and Windows 2000

        a. From the Start menu, select **Settings > Control Panel.**
        b. Double-click the **System** icon. Select the **Environment** tab.
        c. In the CLASSPATH, PATH, and NLS PATH delete any references to IBM Connectors or IBM\Connectors.

        Windows 98

        a. Back up the c:\autoexec.bat file
        b. At the command prompt, enter edit c:\ autoexec.bat
        c. In the SET CLASSPATH, SET PATH, and SET NLS PATH statement delete any references to IBM Connectors or IBM\Connectors.
7. Install VisualAge for Java, Version 3.5.
8. Add the desired CICS, Host-On-Demand, IBM Enterprise Access Builder, Encina, IMS and MQSeries features to the Version 3.5 workspace.
9. Import your projects into the Version 3.5 workspace to finish migrating your 1.1.x code. If you use the SAP R/3 Access Builder and Connector, refer to the note at the end of these steps.
10. You can uninstall VisualAge for Java, Version 3.0x once you have migrated any code you wish to preserve into VisualAge for Java 3.5.

If you are using the SAP Connectors you must regenerate classes that were generated with VisualAge for Java, Version 3.0x.

The Access Builder and Connector for SAP R/3 also provide some utility classes (for example, LogonBean) based on Swing 1.0.3. These classes are replaced by Swing 1.1- based classes When migrating from Version 3.0x to Version 3.5 you have to migrate your own Swing 1.0.3- based classes to the 1.1 level to continue using the utility classes provided by Access Builder and Connector for SAP R/3. You may use the IDE Fix/Migrate tool for this process.

Migration Scenario 2:

1. Install VisualAge for Java Version 3.5, without installing the Enterprise Access Builder for Transactions:
   a. Install VisualAge for Java, Version 3.5. Follow the on-screen instructions.
   b. In the Setup Type page, select **Custom**. Click **Next**.
   c. Select all the components that you want to install, but do not select the **Transactions Access Builder**. Click **Next** when you have finished selecting the components you want to install. If you were returned to the installation screen after attempting to install the Transactions Access Builder and do not want to install the connectors, deselect **Transactions Access Builder,** and click **Next**.
   d. Continue to follow the on-screen instructions to complete the installation.
   You can continue to work with the connectors for VisualAge for Java, Version 3.0x.
2. When you want to install the VisualAge for Java, Version 3.5 connectors, you must either uninstall Version 3.0x or uninstall the 3.0x connectors by following the steps outlined in Migration Scenario 1.
3. Once you have uninstalled the Version 3.0x connectors or VisualAge for Java, Version 3.0x, you can then install the connectors for Version 3.5 by following these steps:
   a. Install VisualAge for Java, Version 3.5. Follow the on-screen instructions.
   b. In the Program Maintenance page, select **Modify**. Click **Next**.
   c. Select **Transaction Access Builder**. Click **Next** and follow the on-screen instructions to continue installing the component.
   d. Add the connectors to the VisualAge for Java IDE. Refer to the online help for information on how to perform this task.

Uninstalling connectors

If you uninstall VisualAge for Java, Version 3.5, the connectors will automatically be uninstalled. To prevent conflicts and errors with future installations of connectors, the environment variables must not contain any reference to the removed connectors. To edit the environment variables, follow the appropriate steps below:

Windows NT and Windows 2000

1. From the Start menu, select **Settings > Control Panel.**
2. Double-click the **System** icon. Select the **Environment** tab.
3. In the CLASSPATH, PATH, and NLS PATH delete any references to IBM Connectors or IBM\Connectors.

Windows 98

1. Back up the c:\autoexec.bat file
2. At command prompt, enter edit c:\ autoexec.bat
3. In the SET CLASSPATH, SET PATH, and SET NLS PATH statements delete any references to IBM Connectors or IBM\Connectors.

Windows 98 only: You may have to manually delete the e-connectors directory (which, by default is C:\IBM Connectors) after you have uninstalled VisualAge for Java.

D.6.0 Enterprise Toolkit for AS/400

D.6.1 Migrating from Version 2.0 of VisualAge for Java

Classes generated using the Enterprise Toolkit for AS/400 with VisualAge for Java, Version 2.0 are not compatible with Java classes generated using the Enterprise Toolkit for AS/400 with VisualAge for Java, Version 3.5. The reason for this incompatibility is that the Version 2.0 ET/400 Java classes used the Abstract Windowing Toolkit (AWT) and the Version 3.5 Java ET/400 classes use Java 2 SDK Swing.

However, the classes that were provided in Version 2.0 can still be found in the package **com.ibm.ivj.et400.util.awt,** which is provided with VisualAge for Java, Version 3.5. If you want to use the classes generated in Version 2.0 in

VisualAge for Java, Version 3.5, you will have to change any references in the Version 2.0 classes from the **com.ibm.ivj.et400.util** package to the **com.ibm.ivj.et400.util.awt** package**.** You can do this using the Fix/Migrate SmartGuide provided with VisualAge for Java. Any errors that occur while you are migrating will be logged in the Log window. Refer to the online help for information about using this SmartGuide.

D.6.2 Migrating from Version 3.0 or 3.02 of VisualAge for Java

If you have generated Java classes using the ET/400 Convert Display File SmartGuide with VisualAge for Java, Version 3.0 or 3.02, they are not compatible with the Java classes generated using VisualAge for Java, Version 3.5 ET/400 Convert Display File SmartGuide.

The reason for this incompatibility is that the code generated in Version 3.0 and 3.02 uses deprecated classes such as AS400SVisualTextField and Subfile classes, whereas the code generated in Version 3.5 uses JFormatted beans. All the deprecated classes can still be found in the package **com.ibm.ivj.et400.util**, which is included with VisualAge for Java, Version 3.5. If you want to use the classes generated in Version 3.0 or 3.02, then you must use the Fix/Migrate tool to convert all the migrated classes so they refer to the new Java 2 SDK Swing package names. This can be done be selecting the affected classes, opening the Fix/Migrate tool and selecting the **Include JDK 1.2 renamed packages** check box. This will automatically migrate Swing references to the Java 2 SDK. Refer to the online help for more information about this task. Any errors that occur while you are migrating will be logged in the Log window.

The Create Subfile SmartGuide is not included with VisualAge for Java, Version 3.5. If you want to continue to using the Create Subfile SmartGuide to generate code, you must continue working with a previous version of VisualAge for Java. Any code generated by the Create Subfile SmartGuide in Version 3.0 or 3.02 can be migrated to Version 3.5 as the classes needed for the generated code are still supported**,** and can be found in the package **com.ibm.ivj.et400.util**. You must follow the same steps as documented in the previous paragraph to migrate your code.

D.7.0 Enterprise Toolkit for Workstations

The Enterprise Toolkit for Workstations (ET/WS) and the high-performance compiler (HPJ) are not included in this release of VisualAge for Java. If you want to keep using the Enterprise Toolkit for Workstations, you will have to continue working in a previous version of VisualAge for Java.

There is no new feature in VisualAge for Java, Version 3.5 that replaces the functionality previously provided by the ET/WS. Functionality similar to what was included with the ET/WS can be found in the "Just-in-time" compiler, which is part of the Java virtual machine. The Java virtual machine is included with the IBM Developer Kit, Java Technology Edition, v1.2.2, PTF 6b.

The JIT compiler takes bytecode and compiles it into direct execution code, then run the programs. The bytecodes are preserved, and still remain portable, however the code (after being compiled by the JIT) runs more quickly. For more information, go to the Sun(TM) web site http://java.sun.com.

D.8.0 IDL Development environment

If you have been using the IDL-to-Java compiler provided by either the IBM Component Broker Series or the CICS IIOP Server Support feature, you will need to manually define the IDL-to-Java compiler invocation string in the following dialogs:

- The IDL-to-Java Compile page of the Options dialog (accessible from the **Windows** menu). Modify the string in the **Command** field.
- The Change IDL-to-Java Compile Options dialog. In the IDL page, select **IDL > Change Compile Options.** Modify the string in the **Command** field.

For more information on modifying the string and opening the IDL page, refer to the online documentation for the IDL Development Environment.

Please see section 1.0 of Part D for information about CICS IIOP support.

D.9.0 Migration Assistant for ActiveX

The Migration Assistant for ActiveX is not included in this release of VisualAge for Java. You can migrate the code you created with the Migration Assistant in previous versions of VisualAge for Java to Version 3.5 by following the

general migration steps covered in Part B. There is no new feature in VisualAge for Java, Version 3.5 that replaces that functionality previously provided by the Migration Assistant for ActiveX.

D.10.0 Persistence Builder

You must use VisualAge for Java to regenerate code from any previous releases of Persistence Builder.

D.10.1 Upgrading specifically from Version 2.0

The following migration issues pertain to you if you are upgrading from VisualAge for Java, Version 2.0:

1. Version all of your projects pertaining to your Persistence Builder applications.
2. After you have migrated your code to VisualAge for Java, Version 3.5, add your projects to the Version 3.5 workspace. You will encounter problems with the services classes because certain converters are no longer included with VisualAge for Java. To correct these problems, generate the "Data Service Classes and Interfaces" for your object model and the problems will be resolved by the code generation services.
3. The superclass for converters has been moved to the project: VisualAge Persistence Common Runtime. You have to change the superclass of your composers to: com.ibm.vap.converters.VapAbstractConverter.
4. The composer superclass has also been moved to a new project. You will have to change the superclass of your composers to: com.ibm.vap.composers.VapAttributeComposer

For more information on performing these steps, refer to the online documentation for the Persistence Builder.

D.10.2 Upgrading from all previous versions (including Version 2.0)

When you load available models, maps, or schemas from the Model Browsers, the internals of the metadata is changed. You cannot then reliably load the metadata into a workspace that is at a lower release level than Version 3.5. The model, map, or schema will appear dirty. Save the model, map, or schema.

Classes created in previous releases for custom queries will have errors. The custom query framework now throws a Throwable object, to enable you to catch exceptions that occur when the custom query is called. As a result, any pre-existing custom queries will show up in the IDE as containing an unresolved error (with a red X), because they are not handling the thrown exception. To correct this situation, throw the exception from your custom query or catch the exception.

You can find more information on dealing with errors in the Visual for Java online help.

Changes to run-time support

The name of the project that contains the Persistence Builder run-time's prerequisite javax packages has changed names. This may require you to recalculate the project path for program elements that use Persistence Builder, as follows:

1. Select the class.
2. From the Selected menu or from the class's pop-up menu, select **Run > Check Class Path**.
3. Click the **Compute now** button near the Project path check box.
4. To save the recalculated path setting, select the **Save in repository (as default)** check box. If you save the setting, a new edition of the class will be created.
5. Select **OK**.

D.11.0 RMI Access Builder

The RMI Access Builder is not included with VisualAge for Java, Version 3.5. You can import your old generated classes and runtime library projects into the Version 3.5 workspace, but you will not be able to run the Remote Object Invocation Manager (ROIM) as it is not included. You should be familiar with the new J2SDK v1.2.2 security model and how its limitations may affect your RMI Access Builder applications.

Migrating your RMI Access Builder applications

If your RMI applications are not already in the Version 3.5 repository, follow these steps to import them into the workspace.

1. Version your RMI Access Builder classes and runtime library projects in your old version of VisualAge for Java. If you wish, you can maintain the classes simply as .java classes.
2. Import them into the VisualAge for Java, Version 3.5 IDE by following these steps:
   - In the Workbench, select **File > Import**.

- Select the Directory or Repository radio button.
- Enter the name the directory or repository you have stored your applications in.
- Select the type of files you want to import and import them.
- If you have not already done so, copy over any project resources associated with your RMI project into the Version 3.5 project resources directory. Create an open edition of the project and version it. In Version 3.5 of VisualAge for Java, your project resources are automatically versioned when you version the project they are contained in.

If you wish to run any of your server objects, you will first have to create a server mainline. For example, if you have a Sever-Side Server Proxy: Reverse1S, and ServerObj2S, you can write a server main to instantiate the server objects:

```
import...;
public class Servermain {
   public static void main(String arg[]) {
      try{
         Reverse1S myserver = new Reverse1S();
         ServerOvj2S obj2 = new ServerObj2S();
}
catch (Exception e) {
   e.printStackTrace();
   }
System.out.print ln("Server Objects Started.");
   }
}
```

As well, due to tighter security restrictions, you will have to define a policy file for servers and clients. For example, you could have a policy file called "My policy":

```
grant {
//Grant all permissions
permission java.security.AllPermission;
};
```

Or you could have a policy that allows anyone to listen on unprivileged ports:

```
grant {
// allows anyone to listen on un-privileged ports
permission java.net.SocketPermission "localhost:1024-", "listen";
permission java.net.SocketPermission "localhost:1024-", "resolve";
permission java.net.SocketPermission "pathfinder:1000-4000", "listen";
permission java.net.SocketPermission "pathfinder:1000-4000", "connect";
permission java.net.SocketPermission "pathfinder:1000-4000", "resolve";
};
```

where *pathfinder* is the user's machine name.

When you launch your client, you will have to run a command similar to the following in order to properly launch the client:

java-Djava.security.policy=<myPolicy.file>

For example if you are running main() within the IDE you would specify java.security.policy= in the Properties section when selecting the "run with" menu item.

You should maintain your code on your previous version of VisualAge for Java so you can continue to develop it or regenerate it.

D.12.0 Visual Composition Editor

You must use the IDE migration tool to repair metadata for visual composites. Refer to the online help topic "Fix/migrate guidelines for class or package references" for information on how to repair broken class or package

references due to migration of classes to the J2SDK v.1.2.2 or the renaming of user-defined program elements.

VisualAge for Java does not keep track of the dependencies between the classes being migrated. Classes which are referenced in a class and have not been migrated yet can have problems with class initialization or may introspect incorrectly.

Any errors that occur while you are migrating will be logged in the Log window. For example, suppose that the following message appears in the Log window while you are migrating a class called Sample.Samp:

Migrating class: Sample.Samp
Could not migrate Class:<Pkg>X::Y

Sample.Samp references X::Y; VisualAge for Java encountered problems in loading class Y. Either Y has not been migrated yet, or it is missing. (In the Visual Composition Editor, class Y appears as a problem class.) To fix this, migrate Y first or, if Y is missing, find a replacement class for Y. In extreme cases, you may have to open Samp or Y in your previous version of VisualAge for Java and reset beans or properties so that migration can continue.

In some cases, opening the class in the VCE will result in the Resolve Class References dialog opening. This dialog shows the problem classes that exist in the VCE. The Unresolved class reference column in the dialog specifies the class that VisualAge for Java used as a temporary replacement. This appears as follows:

com.ibm.uvm.abt.edit.DeletedClassView(X)

The X that appears in parentheses is the name of the problem class. It is likely that X was migrated correctly after the current class. To fix this, select the row to be fixed and click the **Replace** button. In the "Choose a valid class" dialog, select the class X as the replacement class. Do this for all the unresolved classes and then click **OK**.

D.13.0 Servlet Builder and Servlet Launcher

The Servlet Builder and Servlet Launcher are not included with VisualAge for Java, Version 3.5. A Servlet Builder run-time JAR file that is compatible with Version 3.5 is available at http://www.ibm.com/vadd. Follow the "Download" link.

There is no new feature in VisualAge for Java, Version 3.5 that directly replaces the functionality previously provided by Servlet Launcher. To test your servlets, you can either launch them explicitly by entering the URL in a web browser, or by writing HTML or JSP pages to invoke them. For development of new servlets, you can use the new Servlet SmartGuide, which will also create an HTML page which will launch your servlet.

You have two options for using the run-time file:

- You do not import your application into VisualAge for Java, Version 3.5, but upgrade it to the J2SDK v1.2.2 using a migration utility, compile it, and deploy it on the Websphere Application server. This option is covered in scenario 1.
- You import your application into VisualAge for Java, Version 3.5, upgrade it to the J2SDK v1.2.2 using the Fix/Migrate SmartGuide, manually modify it, compile it in the WebSphere Unit Test Environment, export it, and deploy it on the Websphere Application server. This option is covered in scenario 2.

Scenario 1

In this scenario, edit the code in your previous edition of VisualAge for Java, using the Servlet Builder before you export it.

1. Export your application code from your previous version of VisualAge for Java to a directory or repository outside the installation tree.
2. Use a utility such as WoodenChair's "Utility+ JDK Standard Edition" to migrate your java code (rename packages, handle changes in dependencies etc) to the J2SDK v1.2.2. To find out more about this product and its capabilities, go to WoodenChair's Web site at http://www.woodenchair.com/
3. Compile each of the .java class file and the JAR files using the J2SDK v1.2.2. The Servlet Builder 3.5 runtime JAR file must be in the compile classpath.

4. Deploy your code to the Websphere Application Server. The Servlet Builder 3.5 runtime JAR file must be in the WebSphere Application Server document root directory or the server's classpath.
5. If you need to make additional modifications, return to step 1. Repeat the other steps as necessary.

Scenario 2

In this scenario, you edit your Servlet Builder code within VisualAge for Java, testing it in the WebSphere Unit Test Environment.

Follow these steps:

1. Import the Servlet Builder 3.5 runtime JAR file into VisualAge for Java, Version 3.5
2. Import the project you want to work with into the VisualAge for Java, Version 3.5 workspace.
3. Use the Fix/Migrate SmartGuide to repair any broken references in the code.
4. Manually modify the code as desired, and compile and test it within the WebSphere Unit Test Environment.
5. Export the project to a JAR file.
6. Deploy your code to the Websphere Application Server. The Servlet Builder 3.5 runtime JAR file must be in the WebSphere Application Server document root directory or the server's classpath.

You can find more information on how to perform these steps in the VisualAge for Java online help.

The Servlet Builder feature, which was available in previous releases of VisualAge for Java, created servlets that directly generated an HTML user interface. This capability is useful for rapid application development, but has the drawback that it combines the business logic of the application with its user interface. If a change to the user interface is desired then the servlet must be modified. To make maintenance easier, it would be preferable to use JavaServer Pages (TM) (JSP) technology to separate the user interface from the business logic.

A JSP page is an HTML template that includes dynamically generated content. A JSP page can be modified using HTML editing tools like the PageDesigner feature in IBM WebSphere Studio. In IBM's WebSphere programming model, servlets are still used for Web interaction, but they delegate business logic to Java beans and the user interface to JSP. In this programming model, servlets and beans are developed using Java tools and JSP pages are developed using HTML tools.

This separation of the business logic and user interface allows you to assign development responsibilities to the team members who have the appropriate skills and tools, and it leads to easier maintenance.

In accordance with the WebSphere programming model, the function that was provided by the Servlet Builder has been replaced by Java tools in VisualAge for Java and by HTML tools in WebSphere Studio. VisualAge for Java, Version 3.5 provides a new SmartGuide (wizard) to get you started creating Web applications that follow the WebSphere model. You can use the Create Servlet SmartGuide to generate a servlet that invokes a bean for the business logic and a JSP page for the user interface. The SmartGuide also generates an HTML form that you can use to test the servlet. The servlet can be immediately tested in the WebSphere Test Environment of VisualAge for Java, and then refined in the IDE. The JSP and HTML files can be further edited using WebSphere Studio or another HTML tool.

The Create Servlet SmartGuide is modeled on the JavaBean wizard in WebSphere Studio. It also includes additional features that are required by Java programmers. If you already use WebSphere Studio for your HTML development, you will find that the SmartGuide simplifies the flow of work between that tool and VisualAge for Java. The development steps are:

1. Create a Java bean for the business logic of the application.
2. Run the SmartGuide to generate the servlet, JSP, and HTML form.
3. Test the servlet in VisualAge for Java
4. Continue development of the servlet in VisualAge for Java.
5. Refine the JSP and HTML files in WebSphere Studio or equivalent tool.

D.14.0 Swing classes

Swing classes are in a different package in the J2SDK v1.2.2 than they were for Java 1.1.x. If you need to update references to the swing classes you can use the Fix/Migrate SmartGuide.

To migrate your applications, you must select the affected classes and packages, open the Fix/Migrate SmartGuide (by right-clicking on the package or class, then selecting **Reorganize > Fix/Migrate**), and select the **Include JDK1.2 renamed packages** check box to automatically migrate Swing references to the J2SDK v1.2.2. This will add the appropriate From/To entries for Swing. Any errors that occur while you are migrating will be logged in the Log window.

Refer to the online help topics: "Fix/migrate guidelines for class or package references" and "Repairing class or package references" for information on how to properly migrate your applications.

You may not be able to use the Visual Composition Editor to migrate all Swing properties from JDK 1.1.7 to Java 2 because of serialization changes between Version 1.03 and 1.1.1 of Swing. After you have migrated your code to VisualAge for Java, Version 3.5, you may not be able to open some Swing application classes in the VCE. This will only occur when you have used the VCE property sheet to set the properties of a Javabean to a Swing object.

To work around this problem, follow these steps:

1. Re-open the class in your previous version of VisualAge for Java (in the VCE).
2. In the class' property sheet, click the Reset button. A window opens, listing all the bean properties you have modified. Any properties that have been set to Swing objects should be reset to their default setting.
3. Save the class and re-import it into the Version 3.5 IDE.

D.15.0 XMI Toolkit

For information on migrating from the Beta 1.2 version of the XMI Toolkit, please refer to the XMI Toolkit release notes.

If you have used any other release of the XMI Toolkit (for example, a technical preview, an alphaWorks release, or an earlier Beta), you should uninstall it and remove the environment updates performed during installation before using this release of the XMI Toolkit. Migration instructions are provided only for the 1.2 Release.

Part E: General Information

E.1.0 Dealing with project resources and resource management

In Version 3.5 of VisualAge for Java, you can now version and release your project resource files. Refer to the IDE and team online help for more information about this new feature.

If you have migrated projects from a previous version of VisualAge for Java to Version 3.5 of VisualAge for Java, you can follow these steps after you have completed the migration process. You do not have to perform them immediately after you have finished migration, but until you do, your project resources are not versioned in the repository.

1. Ensure that you have copied all your old project resources into the Version 3.5 project resources directory.
2. Add the projects into your workspace from the repository.
3. Create a new open edition of each project. You cannot add new resources to your project until you have created an open edition of it.
4. Version the project, which releases all the resources.

If you are working in the Enterprise Edition of VisualAge for Java, you must use EMSRV, Version 7.0. Resource management is not supported with earlier versions of EMSRV.

E.2.0 Migrating from OS/2 and AIX

OS/2 and AIX are not supported as development platforms for VisualAge for Java, Version 3.5. You can only install VisualAge for Java, Version 3.5 as a client on Windows NT, Windows 98, and Windows 2000. If you wish to use Version 3.5 with your OS/2 and AIX repository, you must connect to that repository from the Version 3.5 IDE. Refer to the online help for information on how to perform this task.

If you have written any platform-specific code, you will have to rewrite it so it works on Windows.

Two scenarios

Your ivj.dat OS/2 or AIX repository is on the same machine the Windows client is on:

1. Back up your old ivj.dat
2. Install Windows 98/2000/NT (for example, as a dual boot).
3. Install VisualAge for Java, Version 3.5.
4. Connect to your old repository (Enterprise) or import from your old repository (Professional). You should copy over any old project resources to your Version 3.5 project resources directory.

Your ivj.dat OS/2 or AIX repository is on a different machine the Windows client is on:

1. Your OS/2 or AIX machine must be running the EMSRV repository program (Version 7.0) provided with VisualAge for Java, Version 3.5. You can connect to repositories that were used with Version 3.02 or earlier of VisualAge for Java, but you must have Version 7.0 of EMSRV installed in order to connect to an old repository from Version 3.5.
2. The Windows client, on which you are running VisualAge for Java, Version 3.5 connects to OS/2 or AIX server as required. You should copy over any old project resources to your Version 3.5 project resources directory.

E.3.0 New security restrictions in J2SDK v1.2.2

Due to a change in the security policy for applets for Java 1.2.2, applets can no longer access local resources.

Several samples that were available with previous versions of VisualAge for Java are not included with VisualAge for Java, Version 3.5 because of this restriction. As well, some samples that are included can only be run as applications, otherwise, they will not work properly. Please refer to the online documentation for instructions on how to properly run samples.

You can change the default security policies by creating your own java.policy file and launching the AppletViewer with the following parameter: -Djava.security.policy=*someURL*

where *someURL* is the location of your new policy file. For more details on general security in Java, please refer to http://java.sun.com/security For specific information on security in Java 2 and policy file syntax, please refer to http://java.sun.com/products/jdk/1.2/docs/guide/security

E.4.0 New External Version control tool (replaces External SCM tool)

The new **External Version Control** tool enables you to connect to external source code management (SCM) providers such as ClearCase, PVCS Version Manager, TeamConnection(TM), and SourceSafe(TM) from VisualAge for Java. With this tool, you can add classes to your SCM provider, check classes and resources files in and out of the SCM system, and import the most recently checked-in version of a class from the SCM system.

This tool replaces the old **External SCM** tool and provides increased functionality.

E. 5.0 Working with third-party ORBs in VisualAge for Java

You can work with third party Object Request Brokers (ORBs) in VisualAge for Java if they are compatible with the J2SDK v1.2.2. You must import the ORB classes into the IDE before you can work with them.

When you import Java classes into the IDE, you can add ORB extension classes to the Java Class Libraries. You may also replace some of the ORB extension classes found in the existing Java Class Libraries as long as long as they are not part of the J2SDK v1.2.2 core classes. The J2SDK v1.2.2 core classes are the classes contained in the rt.jar file of J2SDK.

If you overwrite any classes contained in the J2SDK, VisualAge for Java may not function properly.

You can find an article that contains more details about working with third-party ORBs on the Web at:

http://www.ibm.com/software/vadd/Data/Document2175?OpenDocument&P=1&BCT=1Footer=1

Appendix A: A comparison of data access components

Provided below is a comparison of the Data Access Builder, Data Access Beans and Persistence Builder. A brief description of the EJB Development Environment is included, but this component is not included in the comparison.

The **Data Access Beans** feature offers a rapid way to access relational data visually. This feature is intended for use

with applications where a reusable object model is not required. You can use the Data Access Beans help create visual applications that need a direct view of tables inside the target database. You can also incorporate the Data Access Beans into your code manually.

[ENTERPRISE] The **EJB Development Environment** enables you to develop beans that implement Sun Microsystems' Enterprise JavaBeans (EJB) programming specifications. The EJB Development Environment provides all of the necessary run-time support for the IBM WebSphere Application Server. An incremental consistency checker ensures that enterprise beans conform to the EJB programming specification and indicates whether changes are needed to fix inconsistencies. Refer to the online help for more information on the EJB Development Environment.

[ENTERPRISE] The **Persistence Builder** provides scaleable persistence support for object models and is a first step towards EJB for many customers. Applications built using Persistence Builder can focus on the optimal, reusable object model and quickly map it to any supported relational database. Persistence Builder supports both bottom-up (Schema to Object), and top-down (Object to Schema) mapping. This feature maps objects, and relationships between objects, to data stored in relational databases.

The rest of this document describes the differences between the data access features - Data Access Beans, Data Access Builder, and Persistence Builder.

Implementation details of each feature are not included here. The online documentation covers the additional functionality that the Persistence Builder and Data Access Beans offer, such as relationships, visual palette parts, and advanced transactional capability, and the SQL Assist SmartGuide.

The list of core functions listed below represent an overview of Data Access Builder's main capabilities. Each core function has been implemented in different ways by each component.

Part 1: Object to Relational Mapping features

Database Schema to Object Mapping

- Section 1.0 - Using tables and views
- Section 2.0 - Using custom queries
- Section 3.0 - Using stored procedures

Code Generation

- Section 4.0 - CRUD Operations on objects and stored procedure support
- Section 5.0 - Custom query support

Part 2: Run-time features

- Section 1.0 - Datastore/Transactional support
- Section 2.0 - API LOB support
- Section 3.0 - Quick forms
- Section 4.0 - Current Date/Time/Timestamp support
- Section 5.0 - Cursor support
- Section 6.0 - Asynchronous Execution Concurrency issues

Part 3: Data Access Beans features

Implementations of the functions are explained in the following pages. The Data Access Builder and Persistence Builder functions are all compared consecutively, whereas the comparable Data Access Beans functions (for mapping) are listed at the end of this section.

Part 1: Object to Relational Mapping features

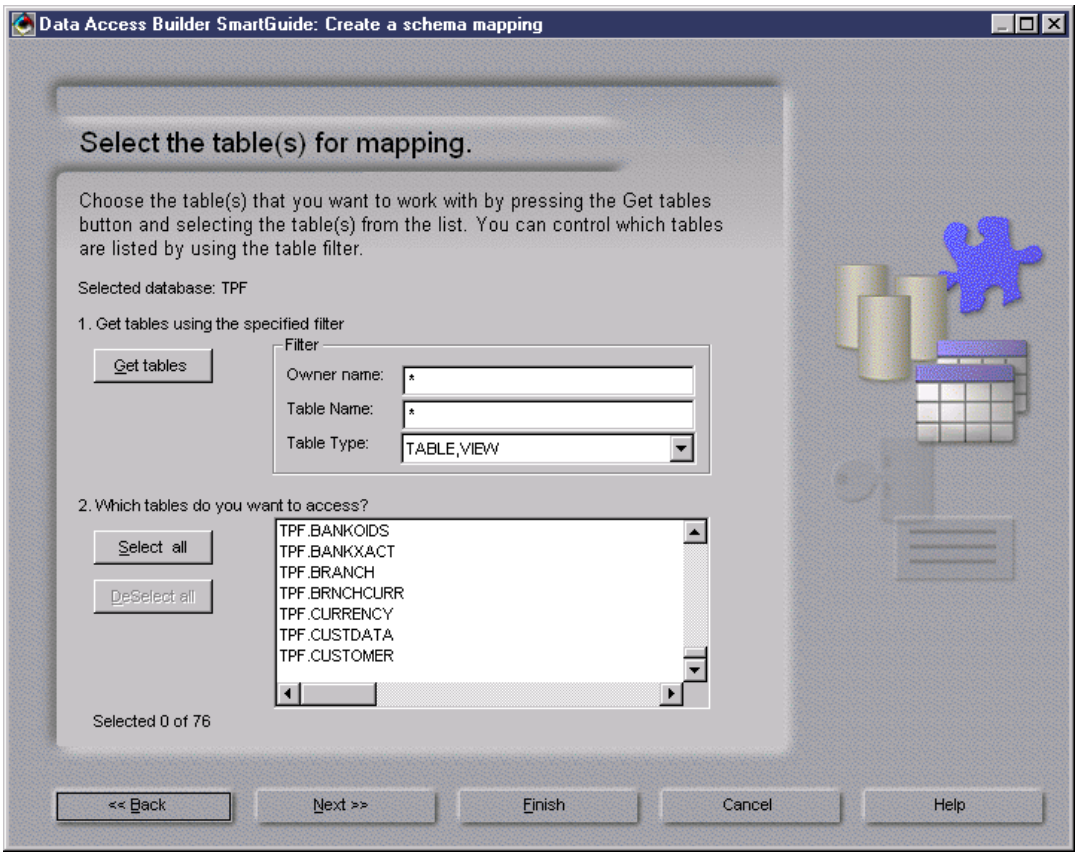
Database Schema to Object Mapping

1.0 Database Schema to Object Mapping Using Tables and Views

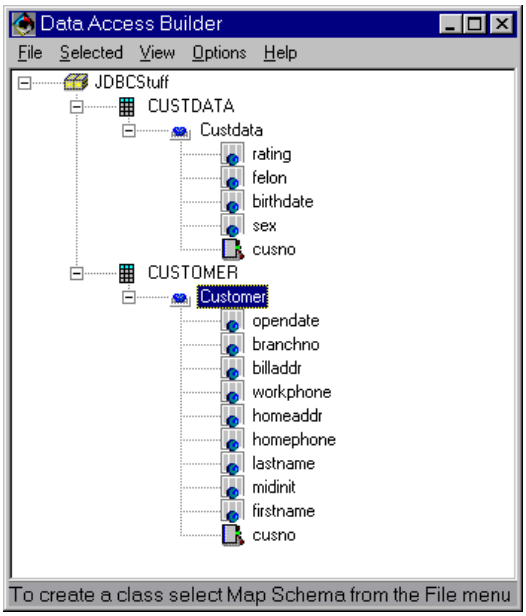
In Data Access Builder

Note: All Data Access Builder steps must be performed in a previous version of VisualAge for Java that includes the Data Access Builder.

In the Map Schema SmartGuide, select the DB2 or ODBC connection, then select the **Select database tables or view** radio button. Click **Next**. The following page opens, showing a list of tables:

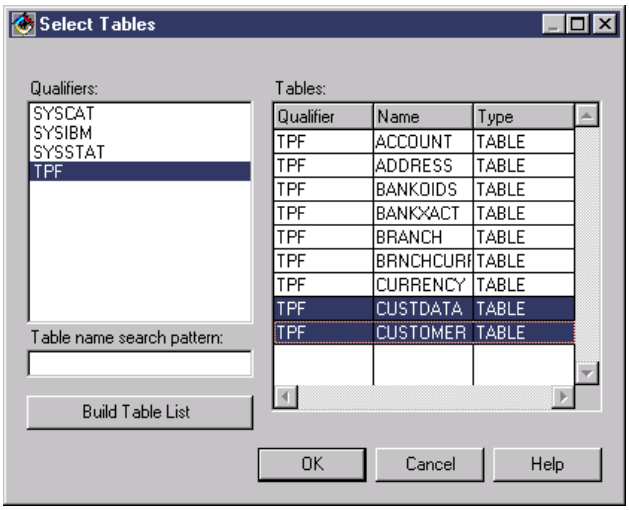


Select the tables you want to work with and click **Finish**. 1 to 1 object mappings between these tables and the resulting objects will be created. The Data Access Builder window shows the column to attribute mapping that automatically occurred.

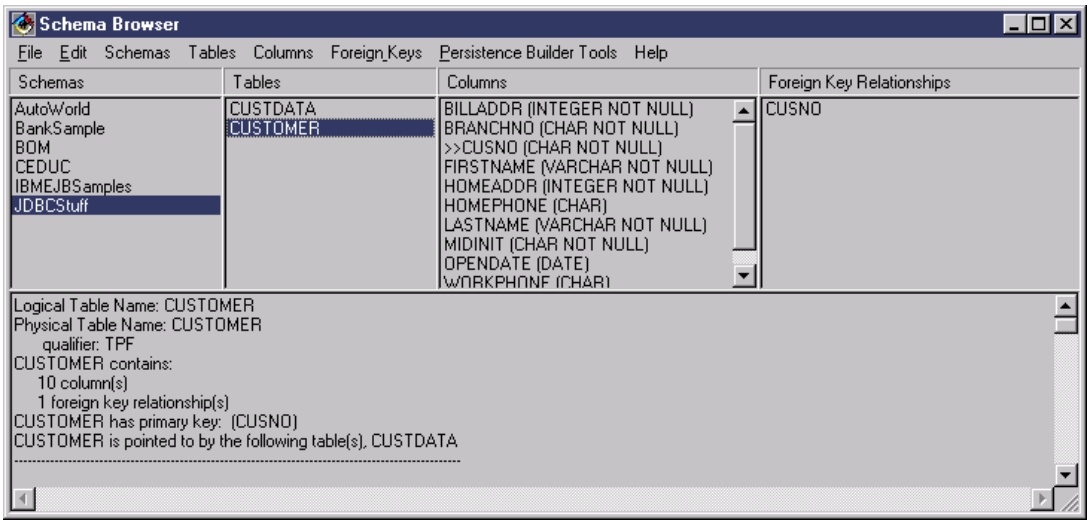


In Persistence Builder

In the Schema Browser, select **Schemas > Import Schema from Database**. Enter the connection information. The Select Tables dialog opens.



Select the desired tables or views, and click **OK**. The Schema Browser now contains the new schema and lists its tables, columns, and keys.

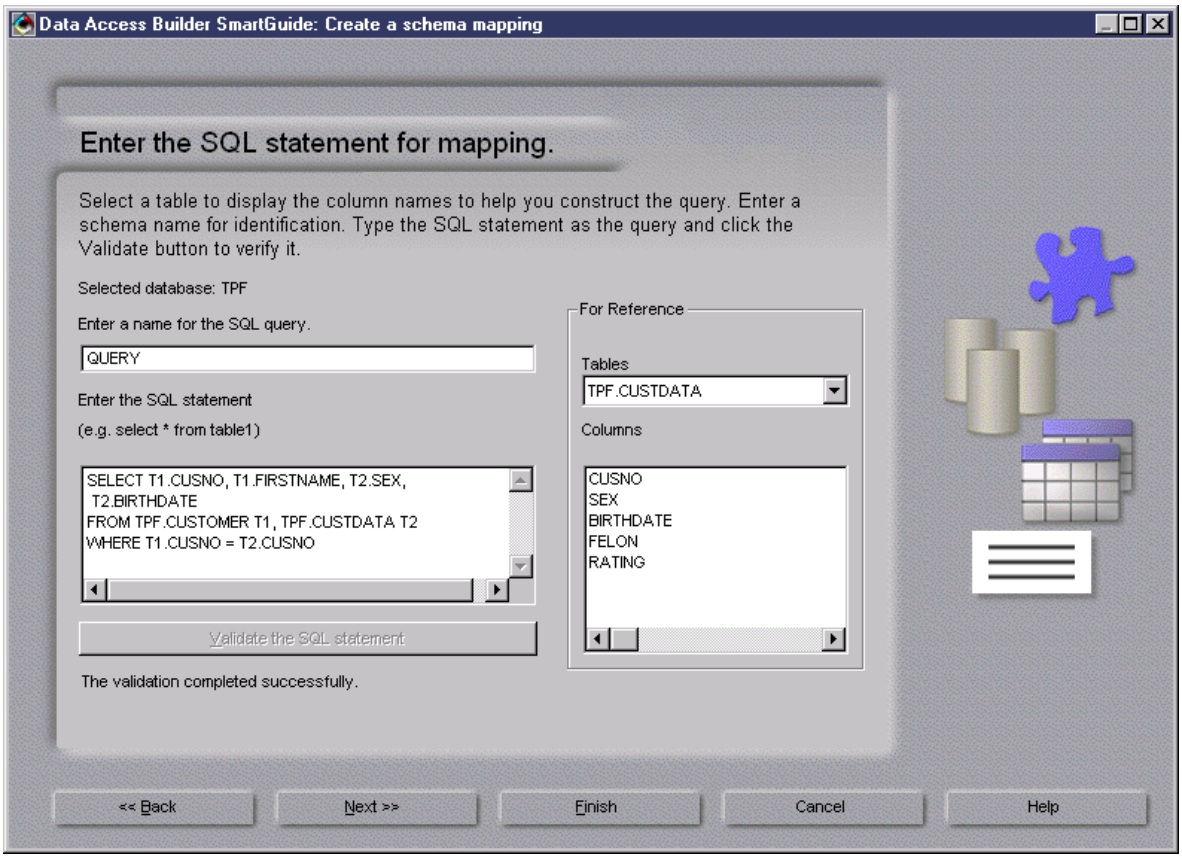


Select **Schemas > Generate Model from Schema**. This generates a simple one to one business model.

2.0 Database Schema to Object Mapping using Custom Queries

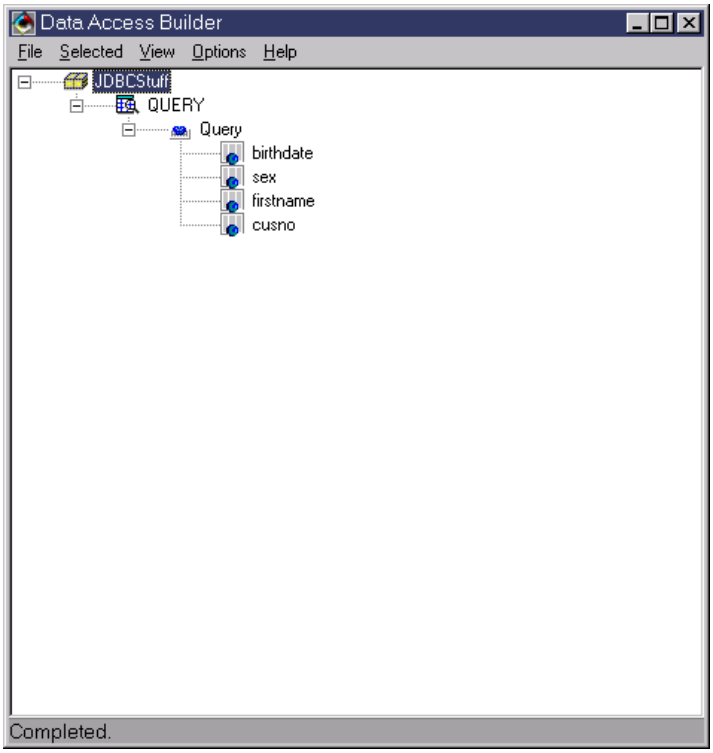
In Data Access Builder

In the Map Schema SmartGuide, select the DB2 or ODBC connection, then select the **Enter SQL Statement** radio button. Click **Next**. Select the tables you want to work with and click **Next**. The following page opens:



Enter your query and click **Finish**. The example above shows a two table join query. Queries with parameter values are also supported.

The resulting object contains the attributes from both tables as shown below:

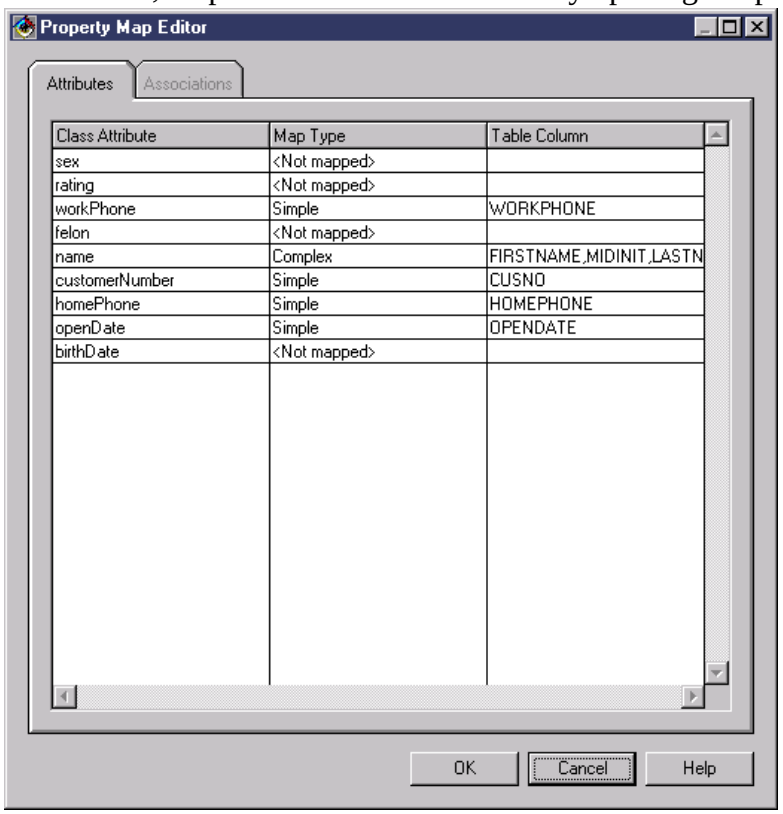


In Persistence Builder

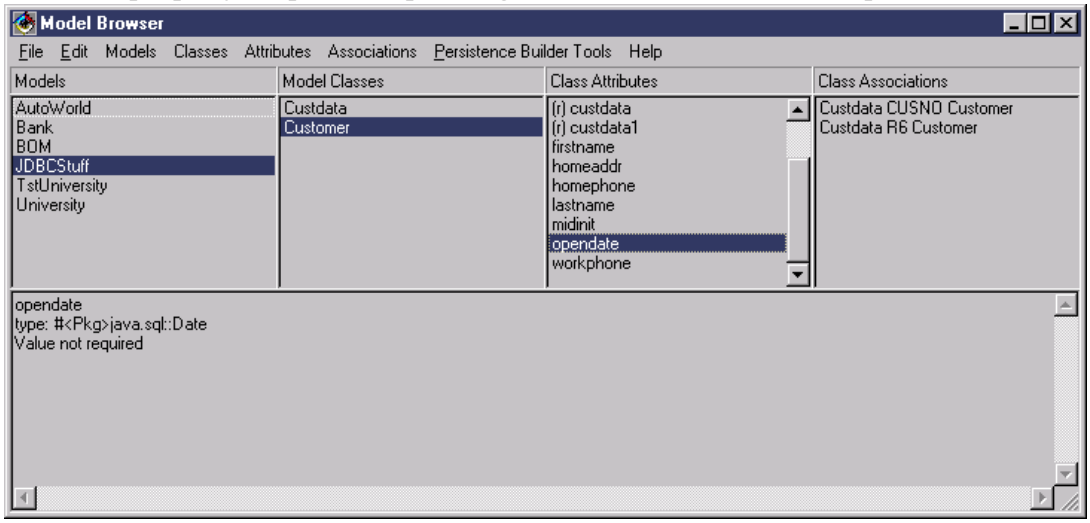
You can use the Persistence Builder to perform the join query mapping by mapping the Schema manually.

1. Import the schema using the Schema Browser.
2. Open the model browser and create the object model the tables will be mapped to.
3. Once the model is complete, open the Map browser, and create a new datastore map. Select the schema and model already created.

4. Select a model class, and add a cluster map by selecting **New Table Map > Add Cluster Map With No Inheritance**.
5. To add more join tables, select **New Table Map > Add secondary table map** menu item. Once the tables have been added, map the individual attributes by opening the property map editor on each table map.



6. Map each attribute to the tables columns, leaving the other tables columns unmapped. The Map browser will show the property maps corresponding to each table in the fourth pane.

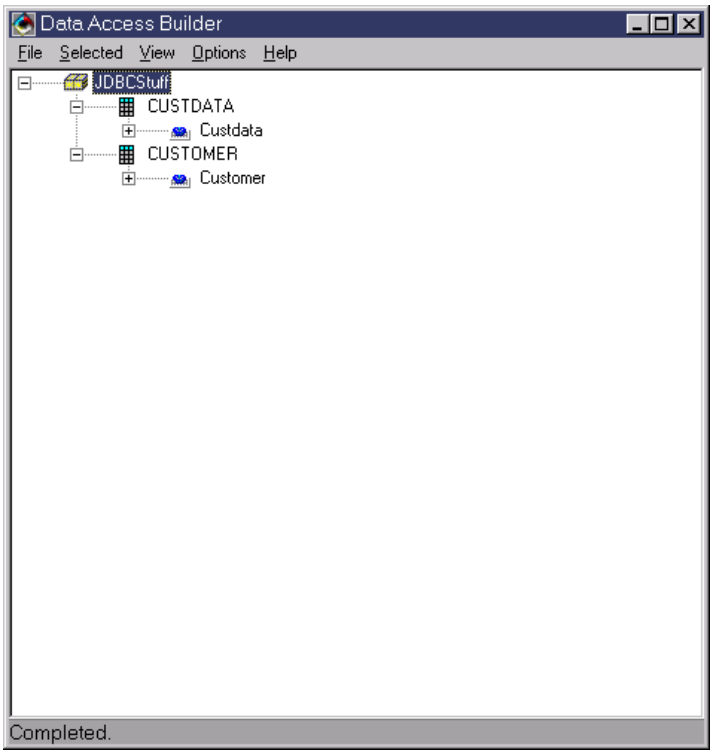


3.0 Database Schema to Object Mapping using Stored Procedures

In Data Access Builder

In the Map Schema SmartGuide, select the DB2 or ODBC connection, then select the **Stored procedure result set** radio button. The Stored Procedure page opens, showing a list of stored procedures. Enter a name for the mapping, select the stored procedure and click **Finish**.

The resulting object contains attributes mapped to the stored procedure's result columns. The queries or stored procedures for the CRUD operations are not pre-defined for this type of mapping.



In Persistence Builder

The Persistence Builder does not have any tools that support stored procedure mapping. You can generate a "Stub Schema" which creates skeleton service classes, where you must supply the supporting code. The all-instances method might appear like the following:

```
/**
* Return a query spec for the query called allInstances
* @return java.util.Vector
*/
public java.util.Vector allInstancesQuery() {
   Vector aSpecArray = new Vector();
     DatabaseCompoundType aCompoundType;
       DatabaseQuerySpec spec = new DatabaseCallableQuerySpec("{call getAllEmp ()}");
      aCompoundType = new DatabaseCompoundType();
       aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseDecimalField("COMM")).setAttributes(9,2,3,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseIntegerField("EMPNO")).setAttributes(4,0,4,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseDecimalField("SAL")).setAttributes(9,2,3,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseIntegerField("DEPTNO")).setAttributes(2,0,4,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseStringField("ENAME")).setAttributes(10,0,12,false));

((DatabaseSelectQuerySpec)spec).setOutputShape(aCompoundType);

aSpecArray.addElement(spec);
return aSpecArray;
}
```

Code Generation

4.0 CRUD Operations on Objects

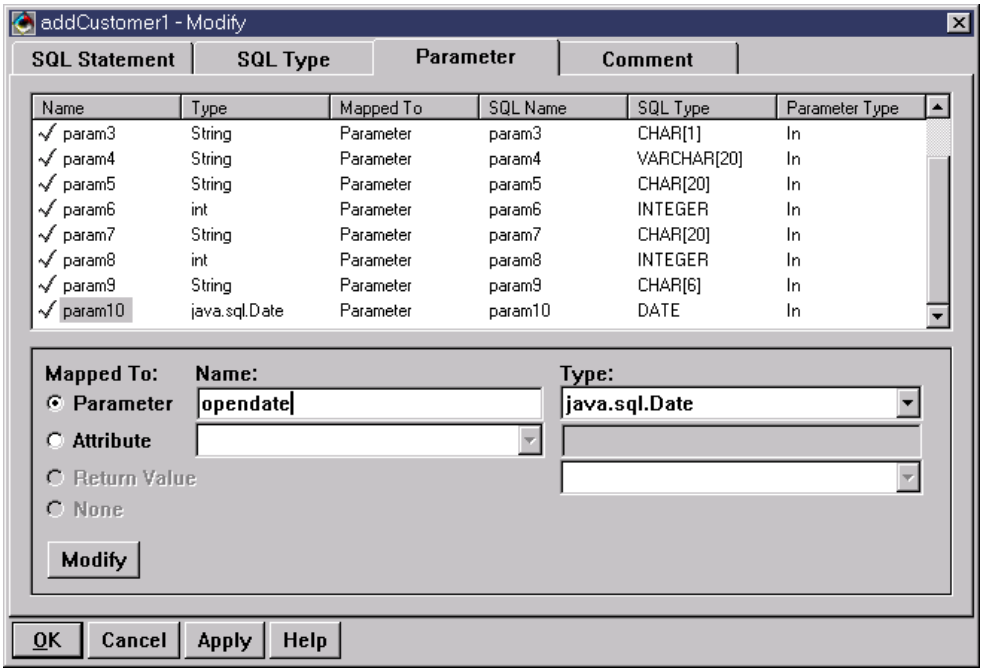
Data Access Builder

The basic CRUD operations (Create, Retrieve, Update, and Delete) are automatically generated with one table to one object mappings. If you use custom queries or stored procedures, these queries are missing, and you must manually define the queries using the Data Access Builder Tool. An example of this would be a join query that defines an object Customer.

1. In the Data Access Builder window, from a bean's pop-up menu, select **Methods**. Add a new method called *addCustomer1*.
2. Enter the following SQL statement:

```
INSERT INTO TPF.CUSTOMER (
CUSNO,
FIRSTNAME,
MIDINIT,
LASTNAME,
HOMEPHONE,
HOMEADDR,
WORKPHONE,
BILLADDR,
BRANCHNO,
OPEN DATE)
VALUES (?, ?, ?, ?, ?, ?, ?,?, ? ,?)
```

After Data Access Builder validates the query, you must map the parameters individually using the Parameter page.



You also need to define the query *addCustomer2* for the insert of the second join table CUSTDATA. The synchronization of the two queries for this atomic function is handled by the user.

Persistence Builder

Persistence Builder will generate all of the CRUD operations once a map has been created between the defined schema and model. In the case of multi-table joins, each table query is generated, and the service engine manages these operations as an atomic unit.

Stored Procedures are not supported in the tools yet, but "Stub Schemas" can be generated and extended. An example of an insert stored procedure query follows.

```
/**
*Return a query spec for the query called insert
```

```
* @return java.util.Vector
* @param args java.util.Vector
* @param anInjector com.ibm.vap.Persistence.BOInjector
public java.util.Vector insertQuery(java.util.Vector args,com.ibm.vap.Persistence.BOInjector anInjector) {
    Vector aSpecArray = new Vector();
DatabaseCompoundType aCompoundType;


 DatabaseQuerySpec spec = new DatabaseCallableQuerySpec("{call

    insertEmp (?,?,?,?)}");
Vector stringArgs;
a CompoundType = new DatabaseCompoundType();

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseIntegerField("EMPNO")).setAttributes(4,0,4,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseDecimalField("SAL")).setAttributes(9,2,3,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseIntegerField("DEPTNO")).setAttributes(2,0,4,false));

aCompoundType.addField((DatabaseTypeField)(new
com.ibm.ivj.db.base.DatabaseStringField("ENAME")).setAttributes(10,0,12,false));

StringArgs = new Vector();
stringArgs.addElement("EMPNO");
stringArgs.addElement("SAL");
stringArgs.addElement("DEPTNO");
stringArgs.addElement("ENAME");

spec.setInputShape(aCompoundType);
spec.setInputValues(args);
spec.setInputNamesWithDuplicates(stringArgs);
   aSpecArray.addElement(spec);
       return aSpecArray;
```

5.0 Custom Query Support

In Data Access Builder

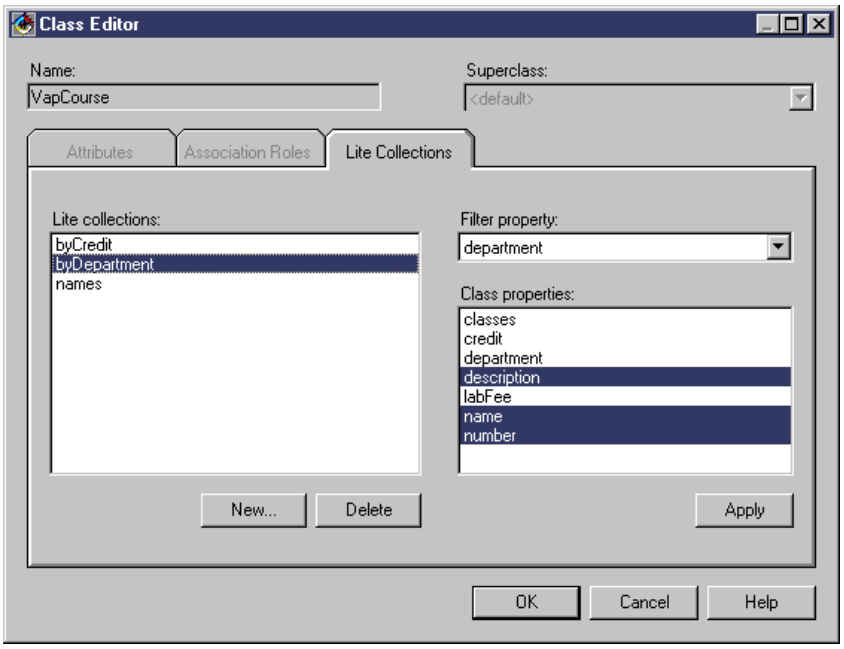
Custom queries in the Data Access Builder are defined the same way CRUD queries are defined. The user adds the named query, and uses the parameter page if necessary. The resulting query will appear as a method on the BusinessObjectMgr class.

The Data Access Builder also provides other methods of defining custom queries, such as SQL Predicate substitution.

```
TheEmpMgr.select('WHERE WORKDEPT = 'E11');
```

In Persistence Builder

There are two options for adding custom queries in Persistence Builder. Lite Collections can be defined at the class level using the class editor. The following page is used to search and filter on the specified class.



Lite collection queries are generally used in search lists or dialogs to "drill down" to the appropriate Business Object. The resulting methods return vectors of "Data Objects" that can be used to retrieve the corresponding persistent business object.

The following is a code example using the generated lite collection shown above:

```
VapCourseHomeImpl aHome = VapCourseHomeImpl.singleton();
   VapDepartmentKey aKey = new VapDepartmentKey("Sales");
VapLiteCollection liteCollection = aHome.getByDepartmentLiteCollection(aKey);

Enumeration enum = liteCollection.elements();
VapCourseDataObject aDO;
VapCourse aCourse;

while (enum.hasMoreElements()) {

    aDO = (VapCourseDataObject)enum.nextElement();
   aCourse = aHome.find(aDO.getNumber());
   //Fetching fully hydrated EJBObject
   }
```

Persistence Builder also provides a custom query framework, which can be used todefine various operations on a business class. These queries will return fully functional Business Objects.

The following is an example of a class student with a custom query "retrieveStudentsOver21". The Home class for students has the following method:

```
public Vector retrieveStudentsOver21() throws java.rmi.RemoteException,
com.ibm.vap.common.VapReadFailureException {
   return customQuery("studentsOver21Query");
}
```

The QueryPool for student contains the corresponding methods for the custom service:

```
/* Return a query spec for the query called studentsOver21
 * @return java.util.Vector */

public java.util.Vector studentsOver21Query() {
   Vector aSpecArray = new Vector();
   aSpecArray.addElement(new DatabaseSelectQuerySpec(studentsOver21SqlString()));
        return aSpecArray;

}

/* Return the SQL string for the query called studentsOver21Query
 * @return java.lang.String */
```

```
public java.lang.String studentsOver21SqlString() {
    return "SELECT T1.SNAME, T1.SNO, T1.SADVFNO, T1.SBDATE, T1.SADDR, T1.SPHNO, T1.SMAJ,
T1.SIQ FROM SPARKY.STUDENT  T1 WHERE T1.SBDATE <= (CURRENT DATE - 00210000.)";
}
```

This method is executed from the home, and has similar object caching behavior as the All-instances method. For more information about this method, refer to the Persistence Builder online help.

Part 2: Run-time features

1.0 Datastore/Transactional Capability

In Data Access Builder

Datastores in the Data Access Builder can be configured at many application levels:

- The Application Datastore is used as a default datastore that all generated business classes may use if one is not defined for it
- The Default Class Datastore specifies at a class level what the instances use as a datastore.
- The Object's Datastore specifies the datastore at the instance level.

Data Access Builder Datastores have a flat transactional model, in other words, if multiple transactions are needed to apply a business function, then a datastore representing each transaction would need to be activated.

Here is a simple scenario with an Employee and Department model:

```
EmployeeDatastore anEmpDS = new EmployeeDatastore().connect();
DepartmentDatastore aDeptDS = new DepartmentDatastore().connect();

Employee anEmp = new Employee();
Department aDept = new Department();

// Perform actions on anEmp and aDept

if (User Applied Employee Changes)
anEmpDS.commit()
else
anEmpDS.rollback();
aDeptDS.commit();
```

Persistence Builder

In VisualAge for Java, Version 3.5 Persistence Builder has the option of using the WebSphere Connection Pool using the JNDI name to lookup the Datasource. This option is available when generating the service classes.

Multiple datastores in Persistence Builder are only required if different databases are used to access model data. Multiple nested transactions are supported per datastore. Transactions are not implied as they are in Data Access Builder. User transactions are controlled by transaction objects that have views on the application business objects.

Here is a code snippet showing the same transactional capability in the Data Access Builder example.

```
DB2Datastore aDS = DB2Datastore.singleton().activate();
Transaction empTransaction = Transaction.begin();
Employee anEmp = EmployeeHomeImpl.create("EmpNum1");
Transaction deptTransaction = Transaction.begin();
Department aDept = DepartmentHomeImpl.create("DeptNum1");

// Do some actions on anEmp and aDept, always resuming the appropriate transaction before making
changes to the corresponding BO.

if (User Applied Employee Changes)
   empTransaction.commit();
else
   empTransaction.rollback();
deptTransaction.commit();
```

It is important to note that in Persistence Builder, no SQL is executed until a transaction is committed. The objects transactional state is maintained internally until an explicit rollback or commit occurs.

2.0 API support

The generated methods on the resulting business objects, and their companion management objects are slightly different between the three frameworks. The following table shows the classes and methods used for each function.

	Data Access Builder	**Persistence Builder**	**Data Access Beans**
Create	aBusinessObject.add();	aBusinessObjectHome.create();	aSelectBean.newRow();
Retrieve	aBusinessObject.retrieve();	aBusinessObjectHome.find (aKey);	aSelectBean.setParameter ("ColumnName", value); aSelectBean.execute();
Retrieve all	aBusinessObjectMgr.select();	aBusinessObjectHome.allInstances();	aSelectBean.execute();
Update	aBusinessObject.update();	(*)	Not supported
Delete	aBusinessObject.delete();	aBusinessObject.remove();	Not supported
Delete current	aBusinessObject.deleteCurrent();	Not supported (**)	aSelectBean.deleteRow();
Update current	aBusinessObject.updateCurrent();	Not supported (**)	aSelectBean.updateRow();
Commit	aBusinessObjectDS.commit();	currentTransaction.commit();	aSelectBean.commit();
Rollback	aBusinessObjectDS.rollback();	currentTransaction.rollback();	aSelectBean.rollback();
Activate	aBusinessObjectDS.connect();	aDataStore.activate();	aSelectBean.connect();
Reset	ABusinessObjectDS.disconnect();	aDataStore.reset();	aSelectBean.disconnect();

Code Example: Find an employee, and alter the phone number

```
Employee anEmp =
new Employee();
anEmp.setEmpno
("000130");
anEmp.retrieve();
anEmp.setPhoneno
("555-9988");
anEmp.update();
anEmp.getDefault
Datastore().
commit();
```

```
EmployeeHome aHome =
EmployeeHome.singleton();

Employee anEmp =
aHome.find("000130");

anEmp.setPhoneno("555-9988");
Transaction.
getCurrent().commit();
```

```
Retrieve all employees:

aSelectBean.execute();

positionToEmployee
("000130");
// User written method
// to position to
//correct row

aSelectBean.
setColumnValue
("PhoneNo", "555-9988");

aSelectBean.updateRow();

aSelectBean.commit();
// If connection is not
//autoCommit=true

Retrieve result set with
only one employee row.

aSelectBean.setParameter
```

			("EmployeeID", "000130");
			aSelectBean.execute();
			aSelectBean.setColumnValue
			("PhoneNo", "555-9988");
			aSelectBean.updateRow();
			
			aSelectBean.commit();
			// If connection is not
			//autoCommit=true

(*)Update operations are implied by altering the values on a business object, if the active transaction is committed, the changes will be synchronized with the datastore by issuing the appropriate update statements.

(**)The section Cursor Support shows alternative solutions.

LOB Support

Data Access Builder	Persistence Builder	Data Access Beans
Data Access Builder includes a class named DAIOStream, which can be used to retrieve LOB's from the database, without faulting the whole object into memory.	Persistence Builder currently does not support streaming LOB's. LOB objects are faulted into memory.	DAB supports JDBC 2.0 LOB data types. If you are using a JDBC 2.0 driver to retrieve a LOB, then you have the option to retrieve the entire LOB into memory, or just the LOB location.

3.0 Quickforms (RAD Features)

Data Access Builder	Persistence Builder	Data Access Beans
The quickform feature of Data Access Builder provides quick sample views, using popular AWT parts to represent the model provided.	A collection of visual parts on the VCE are intended to be used to help with the complexities of visual programming. These parts represent transactional classes, that help the user visually separate units of work.	Version 3.5 contains a Database Application Wizard that will generate an application that uses Swing components to represent the columns of a table obtained by a Select bean. You can also use the standard QuickForm capabilities of the VCE to quickly create a custom UI based on the properties of a Select bean.

4.0 Current Date/Time/Timestamp support

The Data Access Builder tools allow the user to specify "current" on date/time data types, which generates the appropriate SQL. Persistence Builder and Data Access Beans do not support this in their tools, but the SQL could be added manually.

5.0 Cursor Support

Data Access Builder supports cursor functions on result sets, such as *updateCurrent()* or *deleteCurrent()*.

Persistence Builder does not currently support these operations. A good alternative in this case would be to use the Data Access Beans feature.

The Data Access Beans support cursor functions with the DBNavigator visual part managing the cursor position. The following is a description of all the cursor features:

- Cache support for result sets
- Direct access to specific row in cache
- Scrolling backward and forward
- Support large result sets with user specified options

JDBC v1.0 does not support backward scrolling in a JDBC result set. The Select bean maintains a cache of the result set that allows you to scroll through the result set as well as directly access a specific row. The Select bean has properties that allow you to control the size of the cache. You can fetch the entire result set into the cache (with is the default), or fetch a packet at a time into the cache. The size and number of packets is controlled by the user.

- Provide update/insert/delete support on result set

Once a result set has been obtained, the Select bean provides methods to update, delete or insert rows into the result set. No new SQL has to be written by the user to perform these functions.

6.0 Asynchronous Execution

Data Access Builder supports asynchronous operation by providing runnable interfaces on its generated classes.

Persistence Builder also provides runnable interfaces for each CRUD operation and one for custom queries. The service object for each business class has the method *runAsynch(),* which determines the execution type for that class.

Data Access Beans supports asynchronous execution through the DBNavigator tool which spawns "DBAction" runnable instances.

Concurrency Issues (locking/isolation levels)

Data Access Builder does have API for setting the database isolation level on the generated datastore class *setTransactionIsolation(int).*

Persistence Builder maintains its own transactions, and supports the following isolation levels internally.

- Non-locking Repeatable Read
- Non-locking Unrepeatable Read
- Locking Repeatable Read
- Locking Unrepeatable Read.

The Transaction specifies either Repeatable Read or Unrepeatable Read isolation level. The service implementation for the Business class specifies either non-locking or locking implementation. Refer to the Persistence Builder online help for more details about the differences between the levels.

Data Access Beans does have API for setting the database isolation level. You can do this by specifying the desired isolation level when you supply the connection information through the custom property editor or via DatabaseConnection.setTransactionIsolation() method.

Data Access Beans (DAB) features

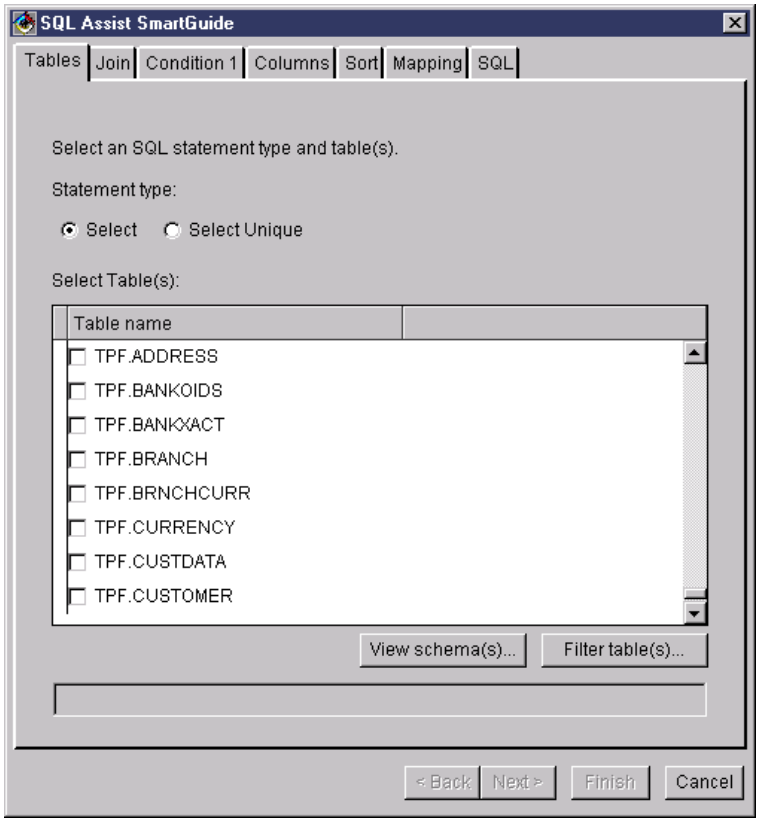
The mapping for DAB occurs between tables and Select beans. This mapping between a table and Select bean is not 1:1, and thus the Select bean does not represent the table. However, Select Beans can very useful for working with the current row and with a result set. You can create one or two Select Beans that can you can use to perform basic database programming operations on a table. Thus, if you are using DAX for database operations like queries, read, write, update, and delete in a simple, straight forward way, then DAB can be a good replacement for DAX.

You can interact with databases by setting the *query* property (which is made up of connection information and SQL Query information) of a **Select** bean**.** The connection information contained in the query property can be used by more than one select bean. You can incorporate select beans into your code either visually, using the Visual Composition Editor or manually.

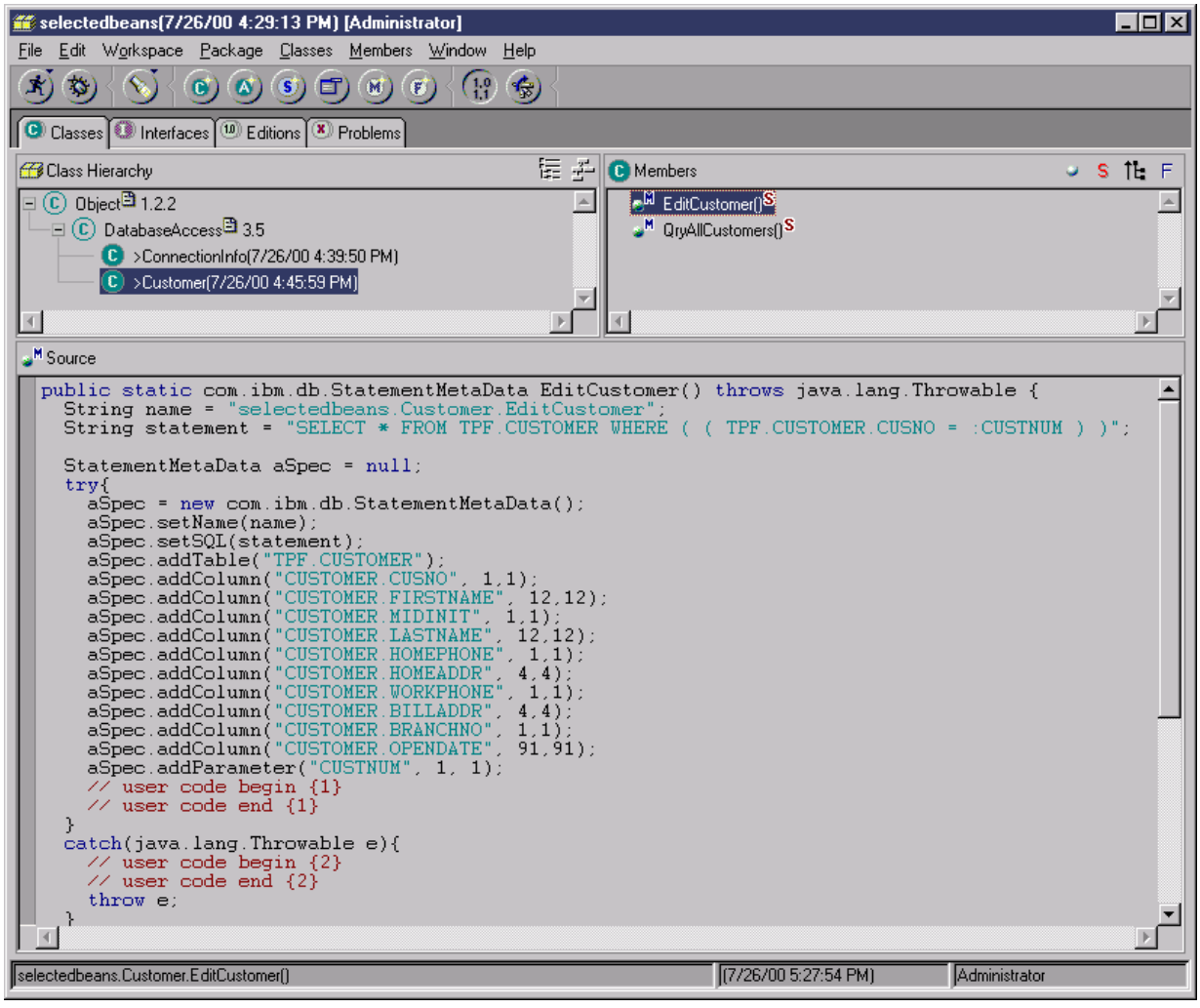
The following is a general outline on how to create a Select Bean to work with the current row of a customer table. Refer to the online help for more details about how to perform these steps:

1. Open a class in the VCE, create a Select bean, and open the Query property editor.
2. Enter the necessary connection information and generate a database access class.

3. Specify the Database Access class. Open the SQL Assist SmartGuide.



4. Selecting the customer table and specifying the condition that the customer number (cusno) is exactly equal to the parameter CUSTNUM. generates a Select bean for the customer. The Database Access class for this select bean would appear similar to the following:

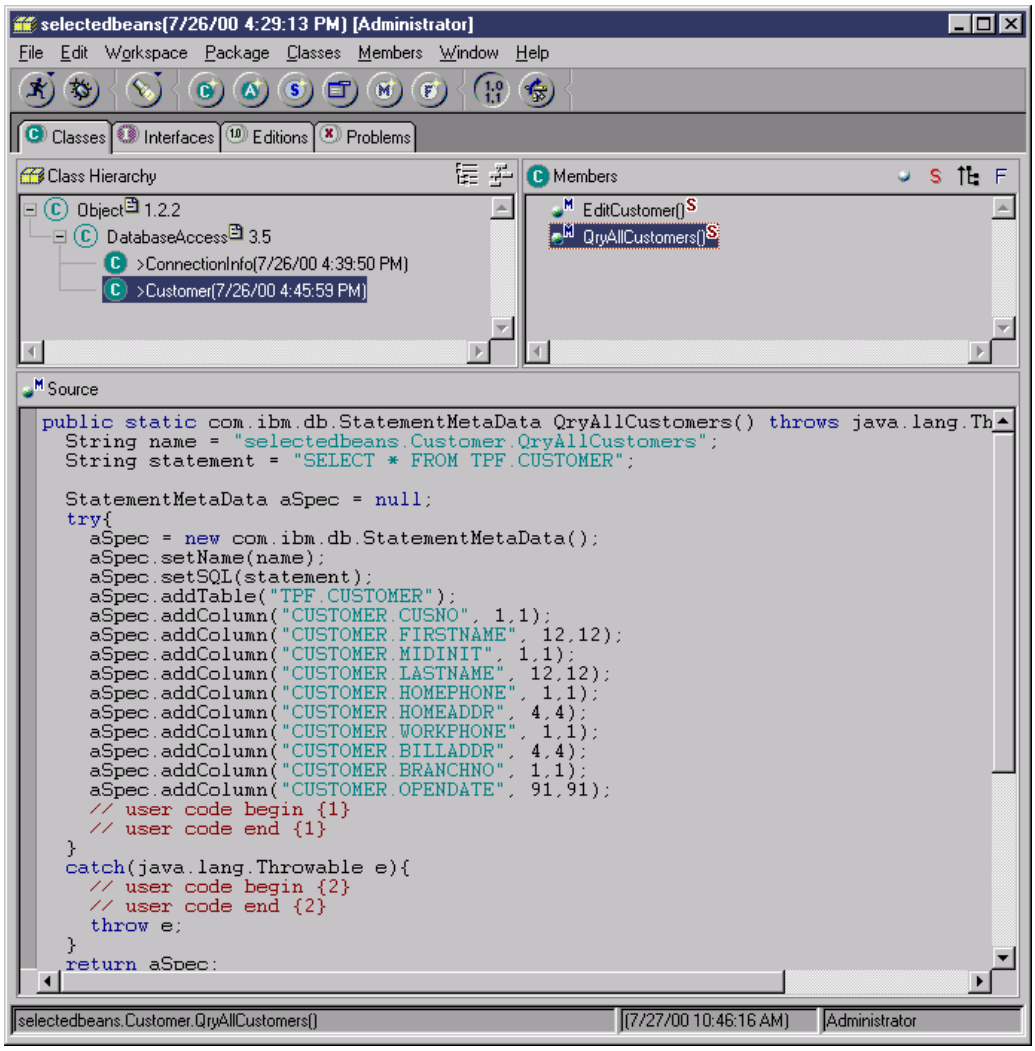


With this Select bean, you can perform basic database operations (read, write, update and delete), on a row in the customer table (when the customer number is specified).

Note: These types of changes makes the Select bean very difficult to use visually. These types of changes should only be performed when using DAB in hand written code.

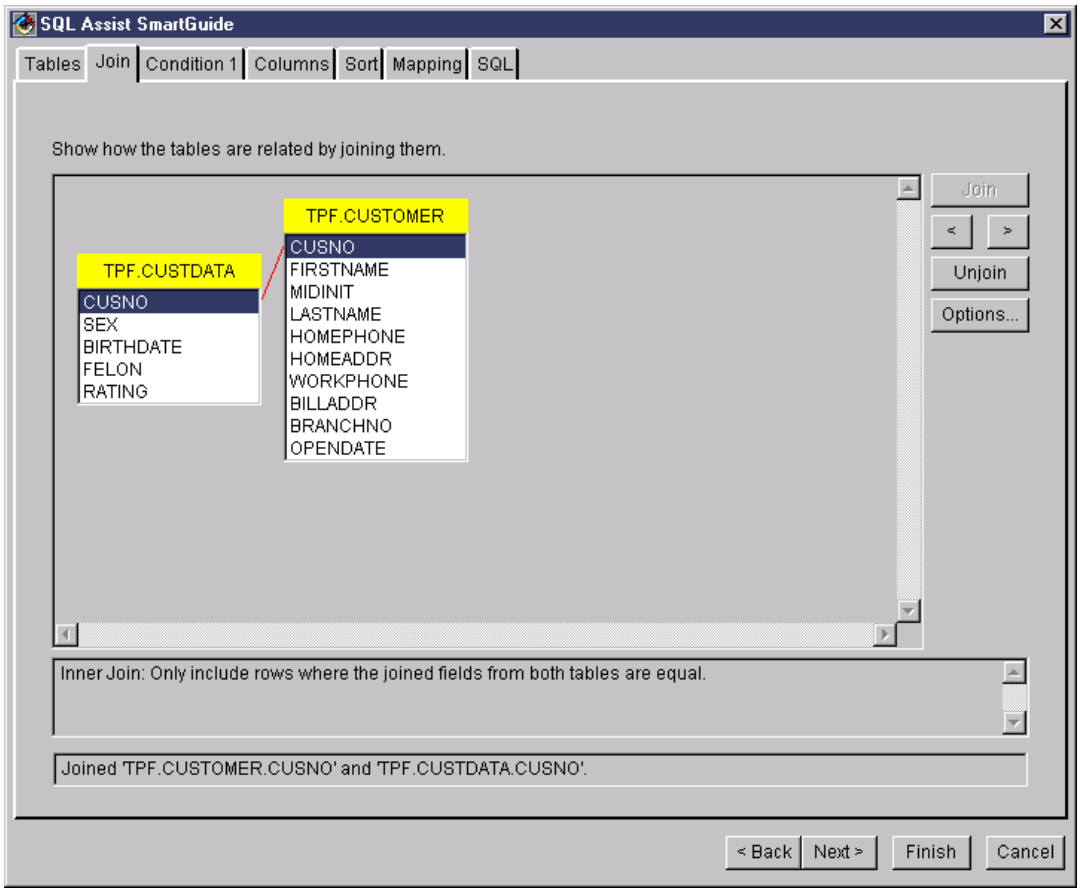
Creating a second select bean

Another Select bean can be created to work with a result set (that is, do a database query) by going through the same procedure and not specifying any conditions. This second Select bean would allow you take advantage of DAB result set functionality. The Database Access class for this Select bean would appear similar to the following:



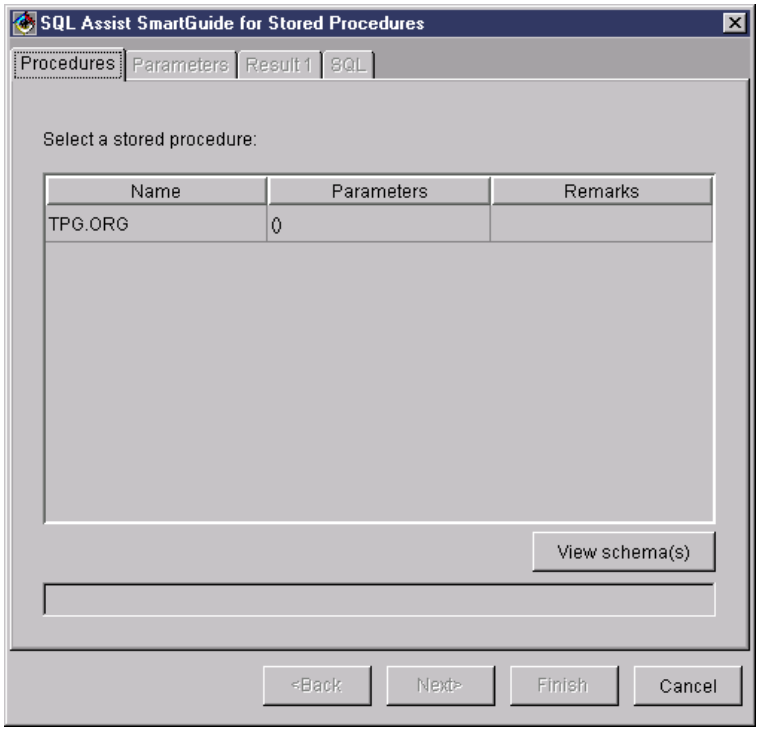
Select beans and custom queries

DAB provides strong support for creating Select beans that use custom queries. As shown below the SQL Assist SmartGuide can assist you in creating a join query, specifying conditions for a query, selecting columns, sorting columns and mapping fields.



Data Access Beans and Stored Procedures

The Procedure Call bean enables you to work with stored procedures. Creating a Procedure Call bean is very similar to creating a select bean. The SmartGuide for stored procedures lists the available stored procedures as shown below.



For more information on using the Procedure Call bean refer to the Data Access Beans online help.

Copyright and Notices

Contract with IBM Corp.

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you. This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites.

The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

IBM, AIX, AS/400, DB2, OS/390, OS/400, RS/6000, S/390, VisualAge and WebSphere are trademarks of IBM Corporation in the United States and/or other countries.

Lotus, Lotus Notes and Domino are trademarks of Lotus Development Corporation in the United States and/or other countries. Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries. Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States and/or other countries. Intel and Pentium are trademarks of Intel Corporation in the United States and/or other countries. Other company, product, and service names may be trademarks or service marks of others