AS/400 Data Access Using VisualAge for Java Enterprise Toolkit for AS/400

Lab 6

Fall Common 1998

Pramod Patel and VA Java Enterprise Toolkit for AS/400 Team

IBM Canada Ltd

Disclaimer

This package is available for use AS IS. There is no support or service to the documentation and the code shipped with the package. IBM reserves all the rights to the lab material. This self-study material is provided for personal use. Reproduction of the material for commercial use is prohibited unless written agreement is provided by IBM.

AS/400 Data Access Using VisualAge for Java Enterprise Toolkit for AS/400 Introduction

This lab introduces you to the **IBM Enterprise Toolkit for AS/400 (ET/400)** feature of **VisualAge for Java (VA Java) Enterprise Edition Version 2.0**.   Formerly known as **AS/400 Feature** in **VisualAge for Java Enterprise Edition Version 1.0.1** and available as a standalone component, this component has been fully integrated in Version 2.0.

Goal

In this lab, you will be introduced to some of the subfile examples shipped as part of **Enterprise Toolkit for AS/400** feature. You will learn how to use the new **Create Subfile SmartGuide**. This SmartGuide (wizard) generates the client Java code using **AS/400 Toolbox for Java** record-level access classes to access data from AS/400 database files. You will develop GUI (*Graphical User Interface)* applications that will make use of the code generated by the Create Subfile SmartGuide to retrieve and update data in both sequential and keyed database files.

Tool

This lab requires **VisualAge for Java Enterprise Edition 2.0** to develop Java code. You will use the **Integrated Development Environment** (**IDE**), **Visual Composition Editor** (**VCE**) commonly known as **Visual Builder**, **Enterprise Toolkit for AS/400** feature and **AS/400 Toolbox for Java** (shipped as part of **Enterprise Toolkit for AS/400** feature).

On the AS/400 host, you need to install the ADTSLAB library by applying one of the following PTF (*Program Temporary Fix*) to the Application Development ToolSet/400 (ADTS/400) Licensed Program Product (LPP).

OS/400 Release	LPP Number	PTF Number
V3R2M0	5763PW100	SF45556
V3R6M0	5716PW100	SF45554
V3R7M0	5716PW100	SF51022
V4R2M0	5769PW100	SF49832
V4R3M0	5769PW100	SF49832

Exercise 1. Overview of Enterprise Toolkit for AS/400 Examples

Introduction

This section will introduce you to:
- **VisualAge for Java Integrated Development Environment (IDE)**
- **IBM Enterprise Toolkit for AS/400** Feature
- Online help on **IBM Enterprise Toolkit for AS/400** feature
- Online help on **AS/400 Toolbox for Java**
- Some of **IBM Enterprise Toolkit for AS/400** Examples

Goal

At the end of this exercise, you will be able to:
- Start the **VisualAge for Java Integrated Development Environment (IDE)**
- Understand the **IBM Enterprise Toolkit for AS/400** feature
- Start help on the **IBM Enterprise Toolkit for AS/400** feature
- Start help on the **AS/400 Toolbox for Java**
- Run the **IBM Enterprise Toolkit for AS/400** examples

What are we waiting for! Let us begin...

Introduction to VA Java Integrated Development Environment (IDE)

NOTE: **"Mouse Button 1"** is the left mouse button, and **"Mouse Button 2"** is the right mouse button. Click always means click **Mouse Button 1**, unless the mouse button is explicitly specified.

Starting VA Java Integrated Development Environment (IDE)

The **VA Java IDE** is also called **Workbench**. It is started as follows.

1. If there is no **Workbench** entry on the task bar at the bottom of your (screen) Windows desktop, then to start **VisualAge for Java:**
- Click on the **Start** button on the left side of the task bar.
- Select the **Programs -> IBM VisualAge for Java for Windows -> IBM VisualAge for Java.**

 After a short delay, the **Workbench** window pops up, and you will see a **"What do you want to Do"** dialog. Select the **Go to the workbench** radio button. Click **OK**.

2. If there is already a **Workbench** entry on the task bar at the bottom of your (screen) Windows desktop, then move the cursor to it and double click **Mouse Button 1** to restore the window and bring it into focus.
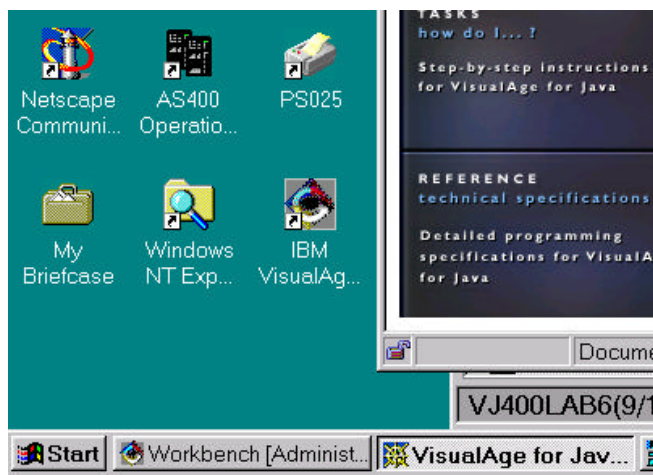
NOTES:

The **Workbench** is an Integrated Development Environment (**IDE**). It is a multiple pane window with a menu bar, a tool bar and a multi-pane notebook for rapid Java development. The notebook is divided into various sections using tabs.

- **Projects** are like AS/400 libraries. They allow you to partition your applications into manageable units. Projects contain multiple packages. Projects are a **VA Java** unique concept.
- **Packages** are similar to AS/400 ILE service programs. They allow you to divide your application components into easily reused units. Packages contain multiple classes. Packages are Java language constructs.
- **Classes** are similar to AS/400 ILE RPG modules. Classes allow you to divide your Java source code into logical reusable groups that contain multiple fields (variables) and methods (similar to RPG procedures or subroutines).
- **Methods** are similar to AS/400 ILE RPG procedures and subroutines. Methods contain the actual Java code and logic to perform a task.

If your **Workbench** window is not modified, then you will notice two horizontally divided panes. The top pane is a hierarchy browser displaying the project hierarchy, when the project page is selected. The bottom pane is a source browser displaying the description of the selected (focused) projects or packages on the top pane. If a class or a method is selected, then its source is displayed in the bottom pane.
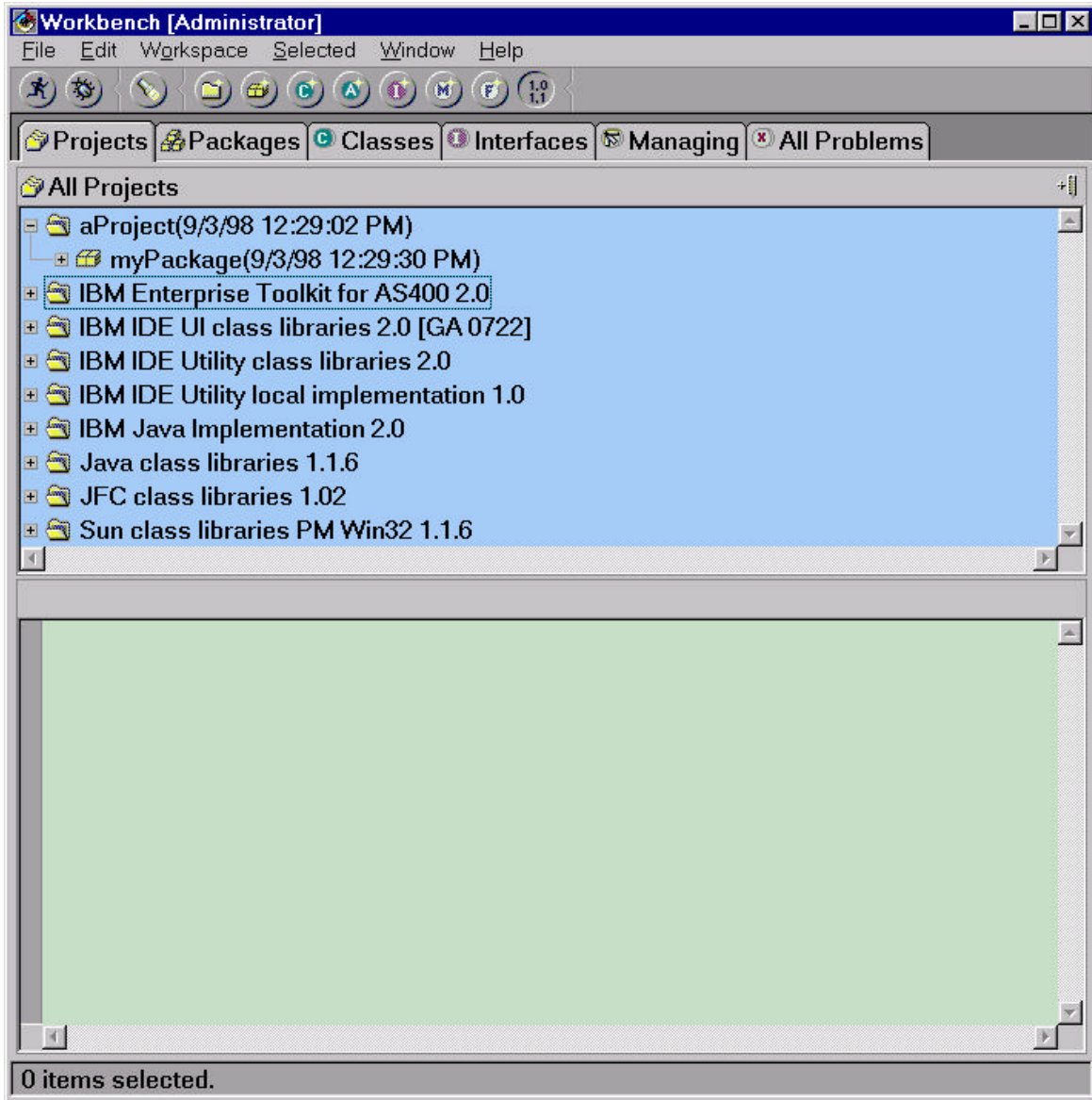
Overview of IBM Enterprise Toolkit for AS/400

This section will show you how the **IBM Enterprise Toolkit for AS/400** feature is organized on the **VA Java Workbench**.

1. Click on the ⊞ symbol beside the **IBM Enterprise Toolkit for AS400** project to expand it. (If you see ⊟ symbol, then someone has already expanded it before you).
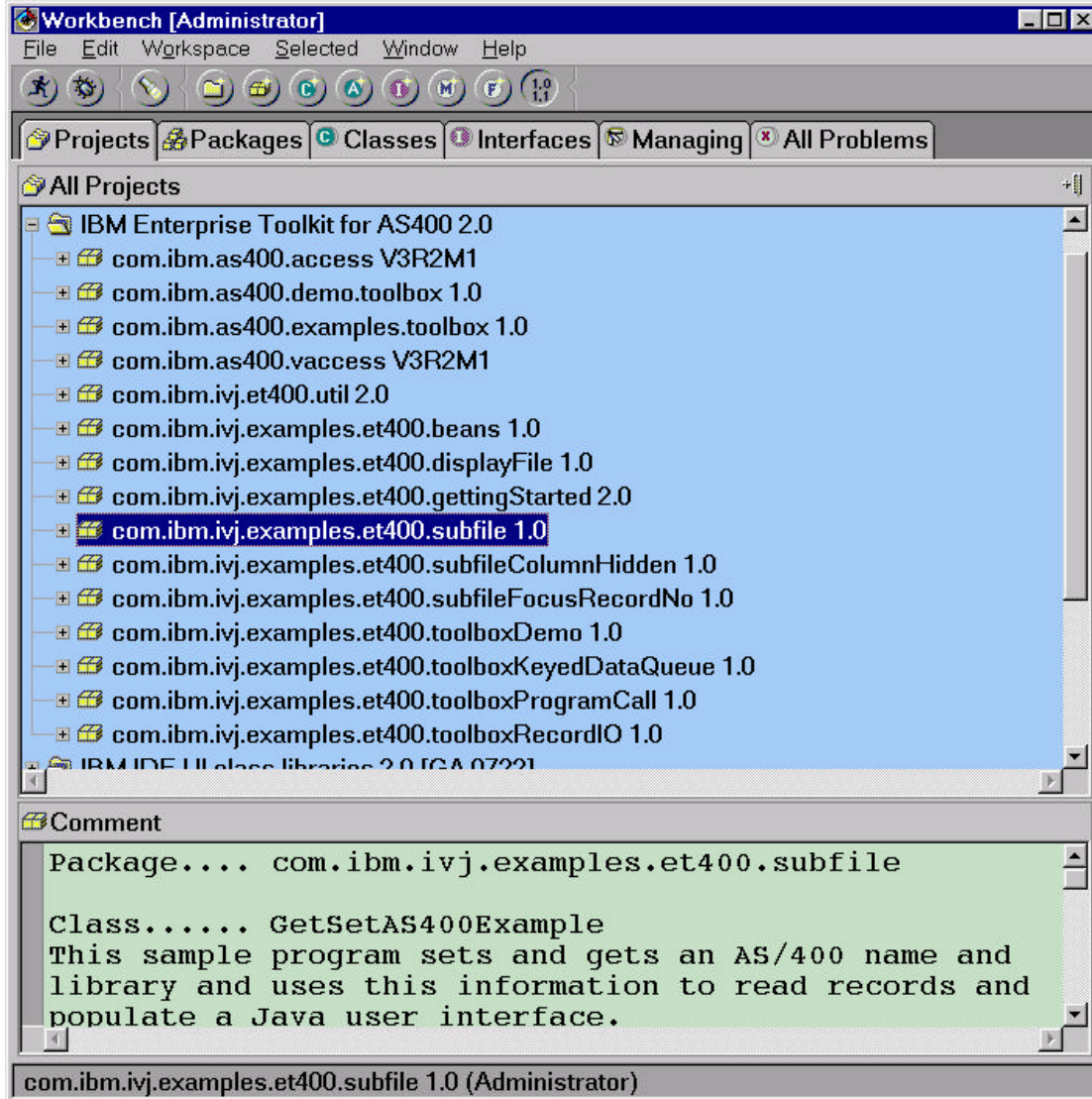
You will notice a list of packages under **IBM Enterprise Toolkit for AS400** project.

com.ibm.as400.access	AS/400 Toolbox for Java access classes
com.ibm.as400.demo.toolbox	AS/400 Toolbox for Java demo programs
com.ibm.as400.examples.toolbox	AS/400 Toolbox for Java on-line help examples
com.ibm.as400.vaccess	AS/400 Toolbox for Java visual access classes
com.ibm.ivj.et400.util	ET/400 runtime utility classes
com.ibm.ivj.examples.et400.bean	Edit Word/Edit Code example
com.ibm.ivj.examples.et400.displayFile	ET/400 Display File example
com.ibm.ivj.examples.et400.gettingStarted	ET/400 Getting Started example
com.ibm.ivj.examples.et400.subfile	ET/400 Subfile examples
com.ibm.ivj.examples.et400.subfileColumnHidden	Subfile column hiding example
com.ibm.ivj.examples.et400.subfileFocusRecordNo	Subfile record number example
com.ibm.ivj.examples.et400.toolboxDemo	Demo accessing various Toolbox resources
com.ibm.ivj.examples.et400.toolboxKeyedDataQueue	Keyed dataqueue example
com.ibm.ivj.examples.et400.toolboxProgramCall	Parameter passing example
com.ibm.ivj.examples.et400.toolboxRecordIO	Data access using record I/O examples

2. Click on any package to review its description in the bottom pane. Click on the ⊞ symbol beside the package name to expand it to see all the classes it contains.

Starting IBM Enterprise Toolkit for AS/400 On-line Help

Before you can start your work you should know how to start the help in case you need it. **VA Java** makes it easy for you to start help. **VA Java** uses HTML (*HyperText Markup Language*) and relies on the default web browser of the workstation to display it. There are different ways you can start **VA Java** help.

- From the Help menu of the **Workbench**
- By pressing F1 from any SmartGuide (wizard)
- By clicking on IDE Help in Release Notes

In this section, you will start the help using the first method.

1. Select **Help -> Help Home Page** from the Help menu on the **Workbench** to open the main home page of the **VA Java** help. **VA Java** home page is displayed in the default web browser. Browse the topics.

---
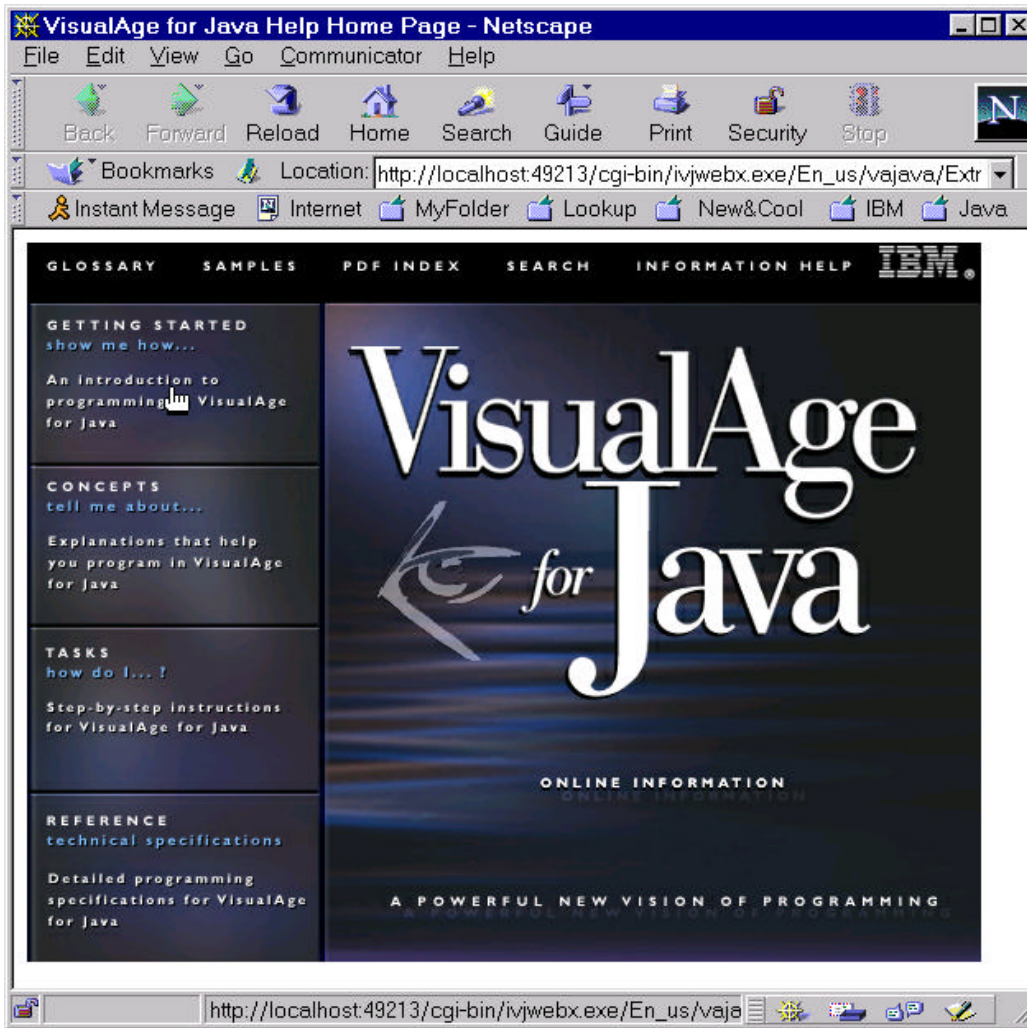
2. Click on the **Getting Started** link (button) on the left hand column. This will open the **Getting Started** page. Note that this page has two vertical panes.
3. Click on the ▶ symbol beside **Enterprise ToolKits** in the left pane to expand it. You will notice a list of all the toolkits.

4. Click on the **Enterprise ToolKit for AS/400** to open its Getting Started web page. Browse the topics.
5. You can click on other links, such as, **Concepts, Tasks, References**, from the menu bar on top of the page to view them.

   Leave the browser running.

10/16/98   Course material may not be reproduced in whole or in part
                                without the prior written permission of IBM.   Trademarks are the property of their respective owners.

Starting AS/400 Toolbox for Java On-line Help

Since this lab and **Enterprise Toolkit for AS/400** make extensive use of the **AS/400 Toolbox for Java** classes, it is worthwhile knowing how to start the Toolbox help.

1. Move the focus onto the **Workbench** by clicking on the **Workbench** on the task bar at the bottom of your Windows desktop.
2. Select **Help -> Tools -> AS/400 Toolbox for Java** from the Help menu to open the main page of the toolbox help. Browse the topics.

---
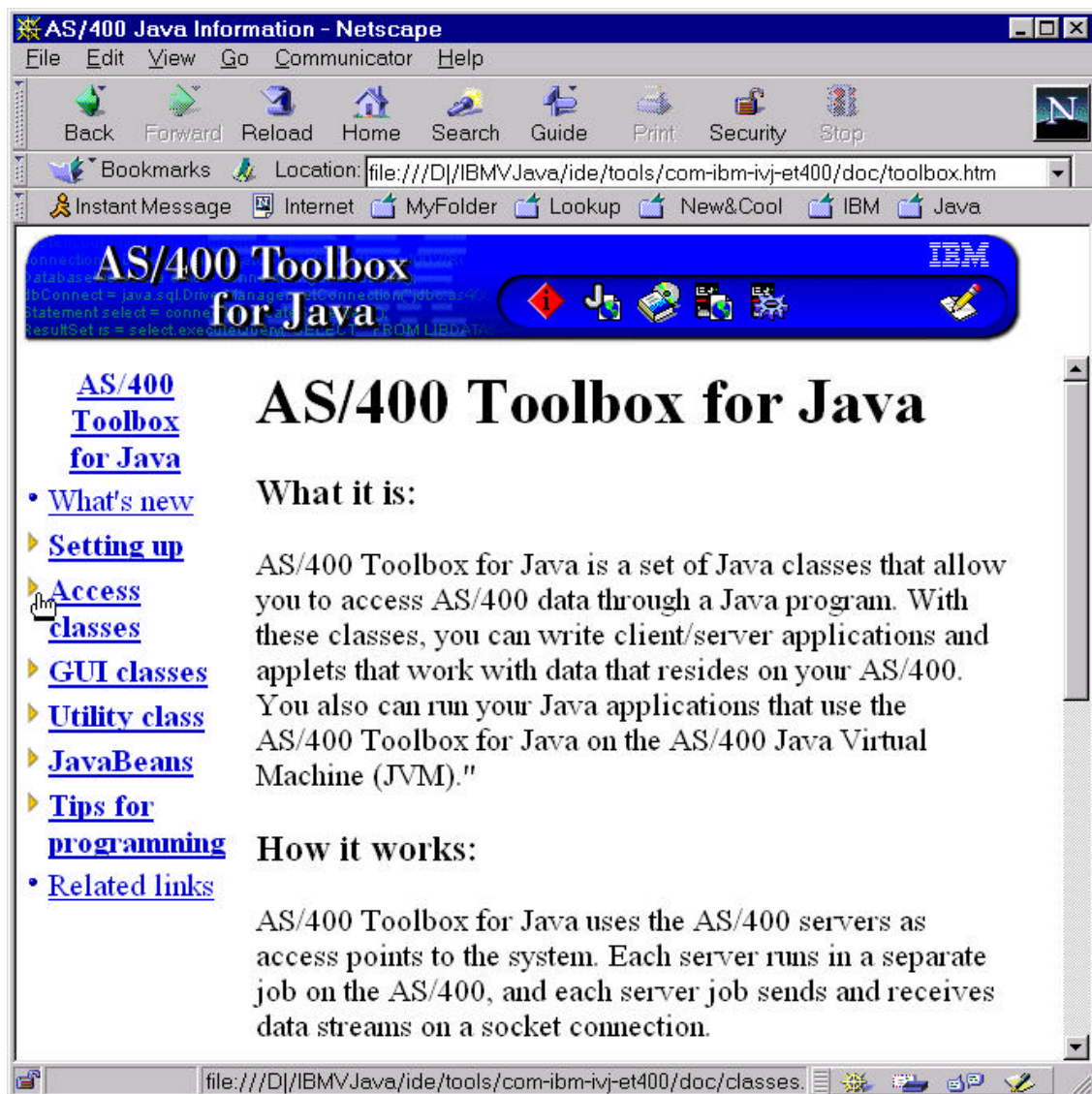
3. Since you will be working with Record-level access classes, click on the ▶ symbol beside the **Access classes** (or on the **Access classes**) in the left column to expand the list.
4. Click on the **Record-level access** from the expanded list to open its web page. Browse the topics.
5. Leave the browser running.

Running IBM Enterprise Toolkit for AS/400 Examples

The purpose of this section is to make you familiar with the examples shipped with the **IBM Enterprise Toolkit for AS/400**. They are provided to make some of the concepts easier to understand and to help you become productive quickly. You will concentrate on the subfile examples here.
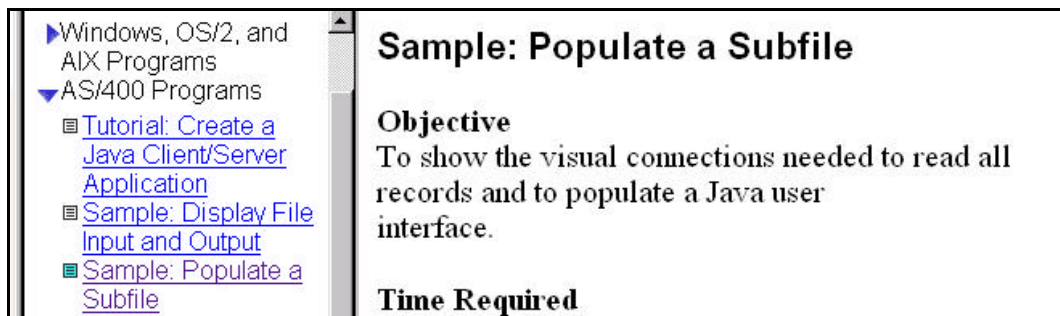
1. Move the focus onto the **Workbench** by clicking on the **Workbench** on the task bar at the bottom of your Windows desktop.
2. Click on the ⊞ symbol beside **com.ibm.ivj.examples.et400.subfile** package to expand it. Browse the description in the bottom pane.

Running SubfileMethodsExample

This example demonstrates how to change the visual subfile attributes, such as colors and horizontal and vertical separator lines.

Before you start running the example, you may look at the help for the example.
1. Click on the **Help -> Samples**. The browser moves into focus and the help Samples page is displayed. (If the browser is not in focus, then click on the **IBM VisualAge for Java - Netscape** on the task bar at the bottom of your Windows desktop).
2. Click on the ▶ symbol beside **AS/400 Programs** on the left column of the web page.

3. Click on the **Sample: Set Subfile Presentation**. The description on the **SubfileMethodsExample** is displayed on the right frame of the browser.

    Now you are ready to run the example.
4. Move the focus to the **Workbench** by clicking on the **Workbench** on the task bar at the bottom of your Windows desktop.
5. Click on the **SubfileMethodsExample** class to select it. This example reads all records from the PRODDTL database file located in the ADTSLAB library on your AS/400 host. Review the class description in its comments in the bottom pane.

---

6. Click on the ⟨🏃⟩ symbol on the toolbar to run it. (Alternately, once the class is selected, you can click **Mouse Button 2** to bring up a pop-up menu and then select "**Run -> Run main**" from it). A GUI window running the **SubfileMethodsExample** class is displayed.
7. Click on the **ReadAll Records** button. The **Signon to AS/400** dialog is displayed.
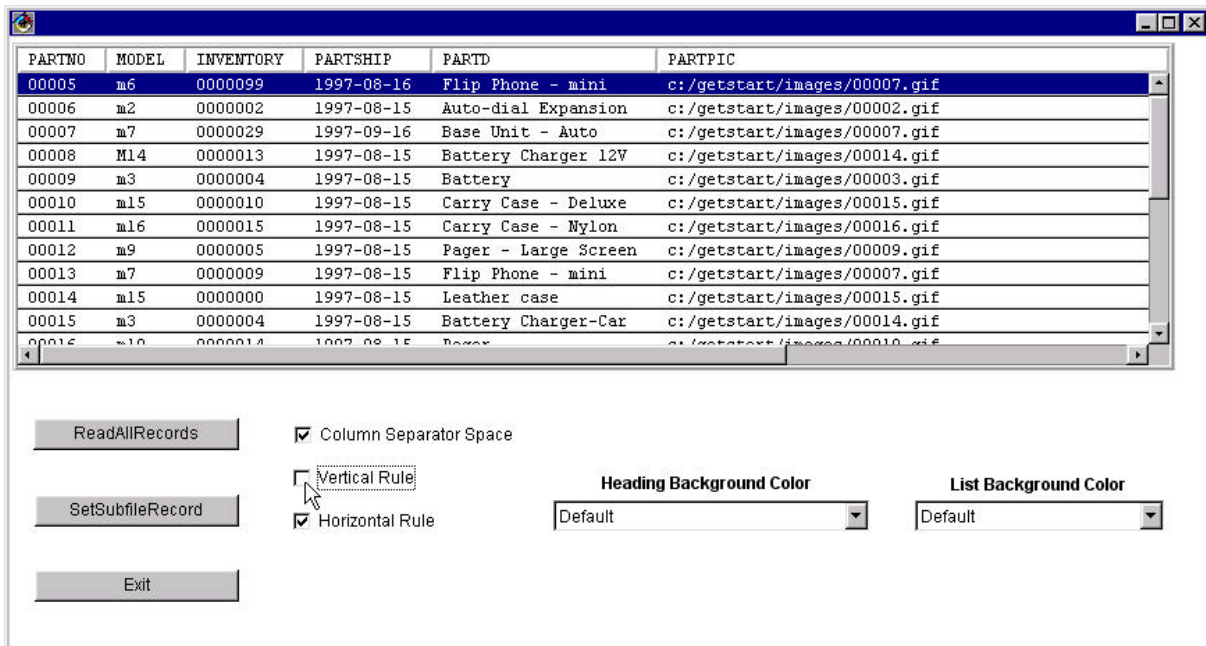
Enter the System name, Userid and Password provided to you leaving the **Default User ID** and **Save password** check boxes checked. Click **OK** on the dialog. The subfile is populated with data.

The actions performed during this step are: an AS/400 object is created and a connection is made to the AS/400 host, the database file is opened, all records are retrieved and the visual subfile is populated.

8. Click on the **Vertical Rule** check box to uncheck it. Notice that the vertical column separator lines disappear.

PARTNO	MODEL	INVENTORY	PARTSHIP	PARTD	PARTPIC
00005	m6	0000099	1997-08-16	Flip Phone - mini	c:/getstart/images/00007.gif
00006	m2	0000002	1997-08-15	Auto-dial Expansion	c:/getstart/images/00002.gif
00007	m7	0000029	1997-09-16	Base Unit - Auto	c:/getstart/images/00007.gif
00008	M14	0000013	1997-08-15	Battery Charger 12V	c:/getstart/images/00014.gif
00009	m3	0000004	1997-08-15	Battery	c:/getstart/images/00003.gif
00010	m15	0000010	1997-08-15	Carry Case - Deluxe	c:/getstart/images/00015.gif
00011	m16	0000015	1997-08-15	Carry Case - Nylon	c:/getstart/images/00016.gif
00012	m9	0000005	1997-08-15	Pager - Large Screen	c:/getstart/images/00009.gif
00013	m7	0000009	1997-08-15	Flip Phone - mini	c:/getstart/images/00007.gif
00014	m15	0000000	1997-08-15	Leather case	c:/getstart/images/00015.gif
00015	m3	0000004	1997-08-15	Battery Charger-Car	c:/getstart/images/00014.gif

9. Click on the **Horizontal Rule** checkbox to uncheck it. Notice that the horizontal separator lines between each records disappear.
10. Click on the **Column Separator Space** checkbox to uncheck it. What did you notice?

    Hint: If you want to find the answer review the comments in the class header for the **SubfileMethodsExample** class in the source browser of the **Workbench**.
11. Click on the down arrow of the **Heading Background Color** choice list. Select any color of your choice. Notice that the header background color has been changed to the selected color.
12. Click on the down arrow of the **List Background Color** choice list. Select any color of your choice. Notice that the list background color has been changed to the selected color.
13. Click on the **Exit** button to close this example and return to the **Workbench**.

Running SubfileSelectionMethodsExample

This example demonstrates the different record selection modes supported on the visual subfile.

Before you start running the example, let us bring up the help on it.
1. Click on the **IBM VisualAge for Java - Netscape** on the task bar at the bottom of your Windows desktop to move the focus on to the browser. (If you are not on the **Samples** page, then click on the **Samples** on the menu bar to display the **Samples** page. Click on the ▶ symbol beside **AS/400 Programs** on the left column of the web page).
2. Click on the **Sample: Set Subfile Selection Mode**. The description on the **SubfileSelectionMethodsExample** is displayed on the right frame of the browser.

   Now you are ready to run the example.
3. Move the focus to the **Workbench** by clicking on the **Workbench** on the task bar at the bottom of your Windows desktop.
4. Click on the **SubfileSelectionMethodsExample** class to select it. This example reads all the records from the PRODDTL database file located in the ADTSLAB library on your AS/400 host. View the class description in the comments in the bottom pane.
5. Click on the ⟨🏃⟩ symbol on the toolbar to run it. (Alternately, once the class is selected, you can click **Mouse Button 2** to bring up a pop-up menu and then select "**Run -> Run main**" from it). A GUI window running the **SubfileSelectionMethodsExample** class is displayed.
6. Click on the **ReadAll Records** button. The **Signon to AS/400** dialog is displayed. Enter the System name, Userid and Password provided to you, and click **OK**. The subfile is populated with the data.

   The actions performed during this step are: an AS/400 object is created and a connection is made to the AS/400 host, the database file is opened, all records are retrieved and the visual subfile is populated.

---
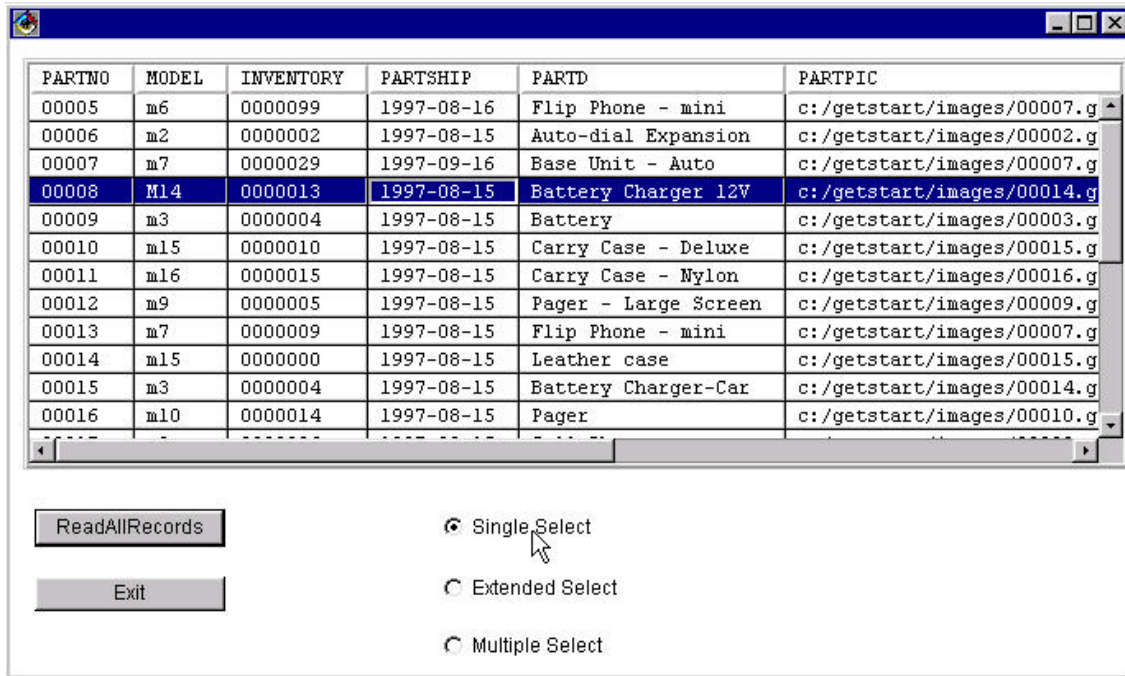
7. By default, the first radio button **Single Select** is selected. If it is not selected, click on it to select it.
8. Now let us try the record selection. Click on a record to select it. Then click on a different record holding the CTRL key down. Did both the records get selected or only the last one? Only the last record will be selected.
9. Click on a record to select it. Then click on a different record holding the SHIFT key down. Did multiple records get selected?
Again only the last record will be selected.

   NOTE: The single select mode ignores CTRL and SHIFT keys and allows only a single record to be selected at any time.
10. Click on the **Extended Select** radio button to select it.
11. Try the record selection. Click on a record to select it. Then click on a different record holding the CTRL key down. Did both the records got selected or only the last one?
This time both records are selected. You can select multiple records by holding the CTRL key down and clicking on other records. Try it.
12. Click on a record to select it. Then click on a different record holding the SHIFT key down. Multiple records between the two selected records including them will be selected.
13. While you have multiple records selected, click on a record to select it without holding the CTRL or the SHIFT key down. What happened?

> **Tips:** In Extended Select mode:
> - If you have multiple records selected and if you want to deselect any records from the selection, keep the CTRL key down and click on the selected record to deselect it.
> - If you have multiple records selected and you click on any record without holding the CTRL or the SHIFT key down, all the records are deselected except the one you clicked on.

14. Click on the **Multiple Select** radio button to select it.
15. Try the record selection. Click on a record to select it. Then click on a different record without holding the CTRL or the SHIFT key down. Did both the records get selected or only the last one?

    Both records will be selected. In this mode, you can select multiple records this way. Try it.
16. Click on an already selected record. It will deselect it.
17. Click on a record to select it. Then click on a different record holding the CTRL key down. This time both records are selected.

    This behavior is the same as what you noticed in the step 15. That means the CTRL key has no special effect.
18. Click on a record to select it. Then click on a different record holding the SHIFT key down. Did it select multiple records? No, but the last record clicked is also selected in addition to previously selected records.

    This behavior is the same as what you noticed in the step 15. That means the SHIFT key has no special effect.

    > **Tip:** **Create Subfile SmartGuide** only allows you to set the **Single Select** or the **Multiple Select** mode on the generated subfile code. The **Multiple Select** of the SmartGuide actually sets the **Extended Select** mode in the generated code. You can change the selection mode in the class constructor of the generated code.
19. Click on the **Exit** button to exit the application and return to the **Workbench**.

In this exercise, using the examples shipped with **IBM Enterprise Toolkit for AS/400**, you learned how to use the subfile to access AS/400 data, modify visual attributes, and use different record selection modes. Now you will learn how to design an application using a subfile.

Exercise 2. Accessing Data From Sequential Files

Introduction

This section will introduce you to:

* The **Create Subfile SmartGuide** to generate the subfile code for sequential database files
* The Accessory Panel application to quickly verify the generate subfile code
* Working with sequential database files on an AS/400 host using the generated subfile code

Goal

At the end of this section you will be able to:

* Use the **Create Subfile SmartGuide** to generate the subfile code for sequential database files
* Use the Accessory Panel application to quickly verify the generate subfile code
* Work with sequential database files on an AS/400 host.

Let us go for it...

Generate Subfile Code

In this section, you will use the **Create Subfile SmartGuide** to generate the Java classes needed to access sequential database file, PRODDTL, on the AS/400 host. These classes use the record-level access classes of the **AS/400 Toolbox for Java**. Before you begin, you need to setup the development environment on the **Workbench.** This is already done for you. A project **VA Java for AS400 Labs** and a package **VJ400LAB6** has been already created for you.

1. Click on the ⊞ beside the **VA Java for AS400 Labs** project to expand it.
2. Click on the **VJ400LAB6** package to select it.
3. Click **Mouse Button 2**. Select **Manage -> Create Open Edition**. This will create an open edition for the selected package. This is necessary to create new classes in it.
4. From the **Workbench** menu bar select **Selected -> Tools -> ET/400 -> Create Subfile...**.
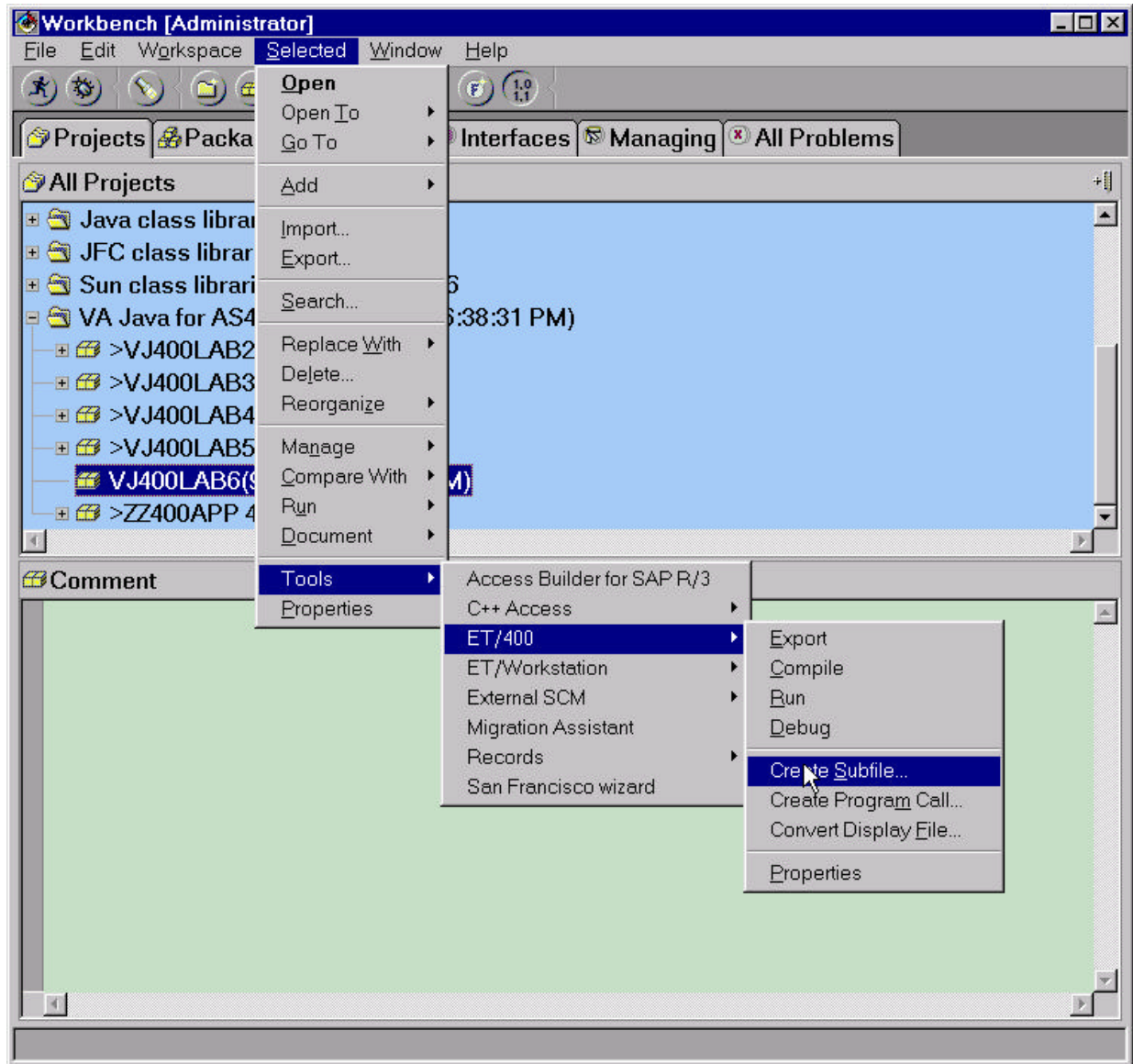
The first page of the **Create Subfile SmartGuide** is displayed.
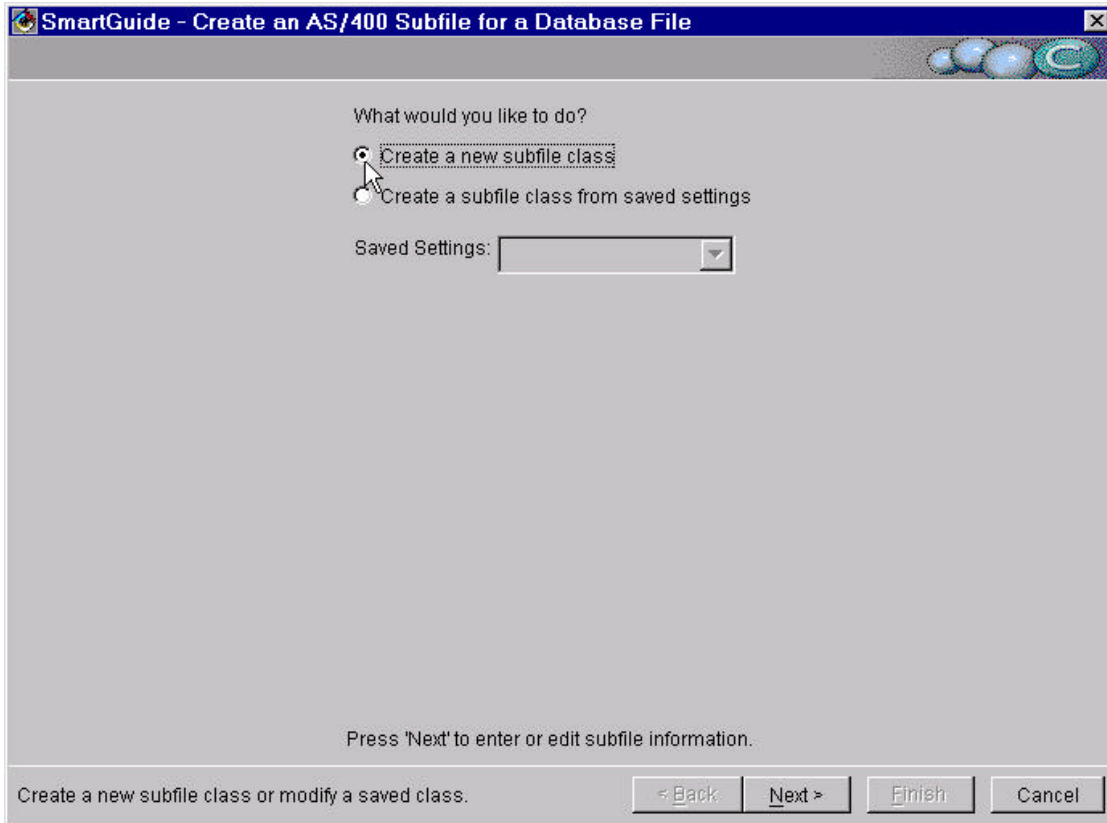
5. On the first page of the **Create Subfile SmartGuide**, select the **Create a new subfile class** radio button. Click the **Next** button. The second page of the **Create Subfile SmartGuide** is displayed.

6. On the second page of the SmartGuide, enter the following data:
   - In the **Server** field select the Server name given to you.

     If the **Server** field is empty, then click on the **Add/Modify...** button. Click on the **Add...** button on the **Add/Modify Servers** dialog. Enter the given server name in the **Server** name field on the **Add Server** dialog. Then click the **OK** and **Close** buttons to close the dialogs.
   - Enter PRODDTL in the **File** field.

     NOTE: Alternately, you can click on the **Browse...** button and select the database file from the **Select AS/400 Database File** dialog. It is left for you to experiment!
   - Enter JAVAxx (where xx is the workstation number between 01 to 99) in the **Library** field.
   - Leave the default settings for the **Class, Project and Package** fields. Note that these values are automatically filled in for you.
   - Click the **Next** button. The **Signon to AS/400** dialog is displayed. Enter the server name, userid and password and click **OK** button.

   The SmartGuide makes an AS/400 connection and retrieves the record format information on the file. Then the third page of the **Create Subfile SmartGuide** is displayed.
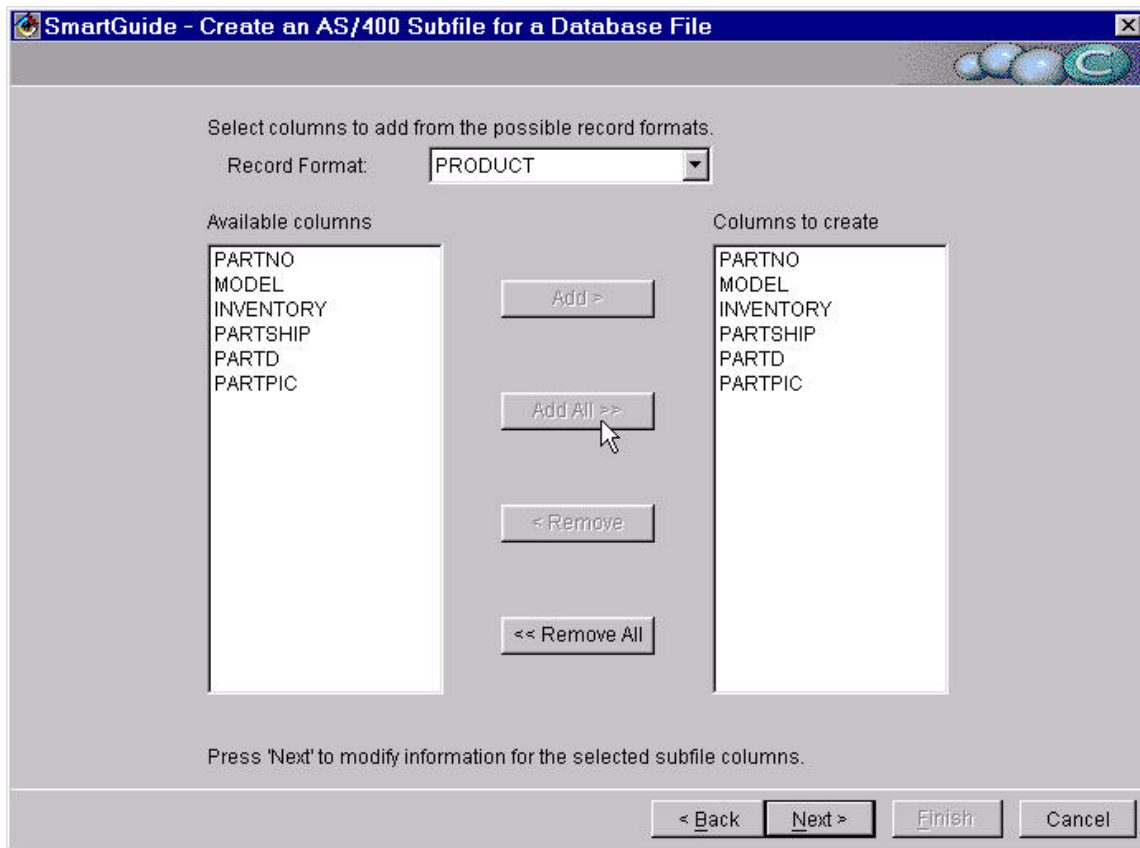
7. On the third page of the SmartGuide, enter the following data:
   - Leave the Record Format field unchanged, since you have only one record format.

     If the database file contains more than one record format, then this allows you to select one of them. The Subfile SmartGuide supports only one record format at present.
   - Click on **Add All>>** button to add all available columns to the **Columns to create** list.

     <u>**NOTE:**</u> Alternately, you can select columns and then click on the Add button. Remember that only columns added to the **Columns to create** list will be visible in the final application. So add all columns for this exercise.
   - Click the **Next** button.

   The SmartGuide retrieves the file field information. The fourth page of the **Create Subfile SmartGuide** is displayed.

8. To better understand all the columns on the fourth page, press **F1** key. This will bring up the help for this page of the SmartGuide in the browser. Look up the help on the Validity, Comparison, Edit Code, and Edit Word related columns.
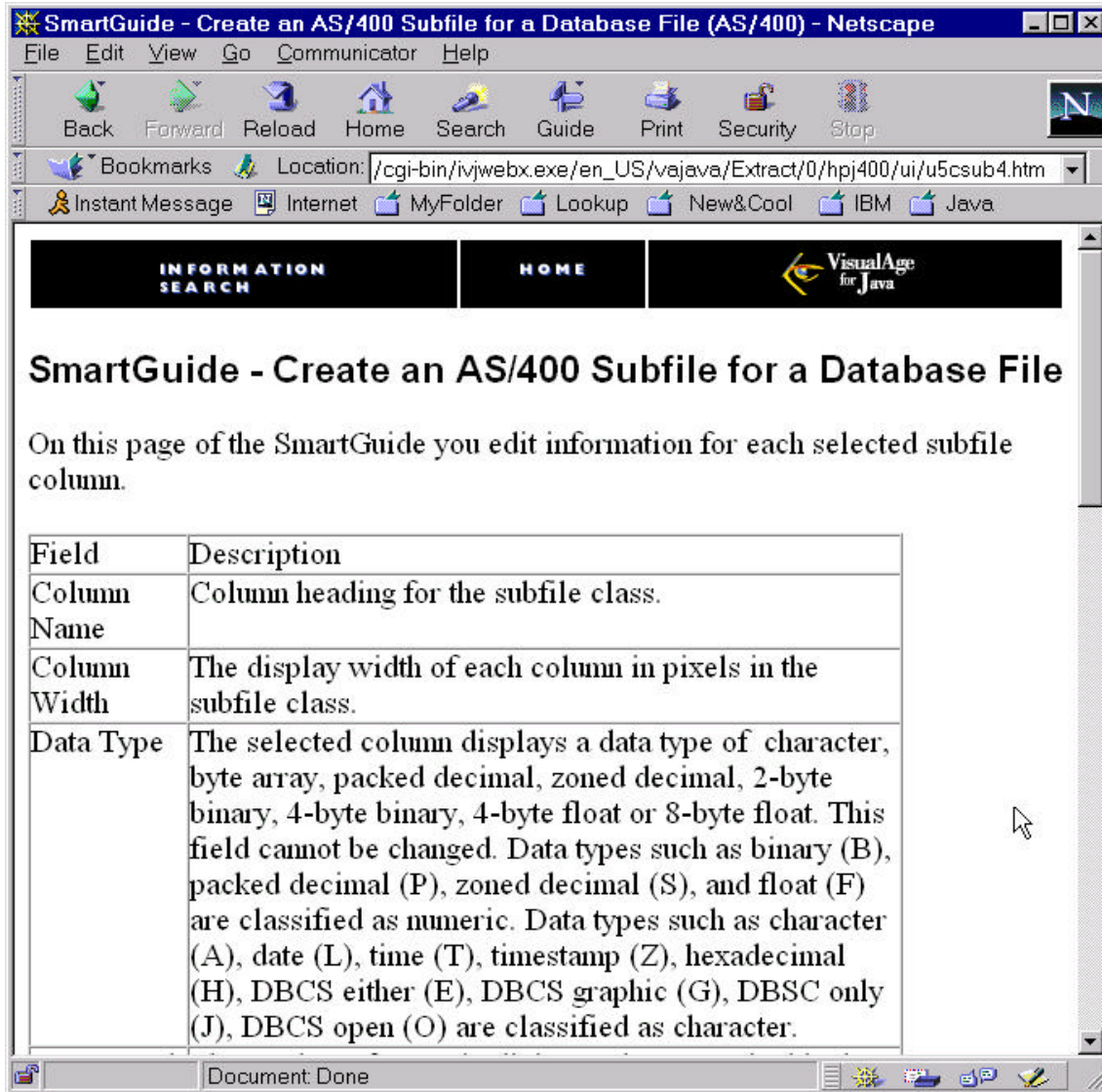
9. On the fourth page of the SmartGuide, enter the following:
    *   Select the row for the INVENTORY field (column name). Click on the **Validity Type** (column) cell for INVENTORY. A choice list is displayed. Select **Comparison** from the list.

        NOTE: Minimize the browser window if required.
    *   Click on the **Comparison Type** cell for the selected row. A choice list is displayed. Select **Greater Than Equals** from the list.
    *   Double Click on the **Comparison Value** cell for the selected row. Enter "5" in the cell.

        The last three steps will ensure that the INVENTORY field must be greater than or equal to 5. This validity checking will be performed when you run the generated subfile and try to update the value in this field.

- Click the **Next** button. The fifth page of the **Create Subfile SmartGuide** is displayed.
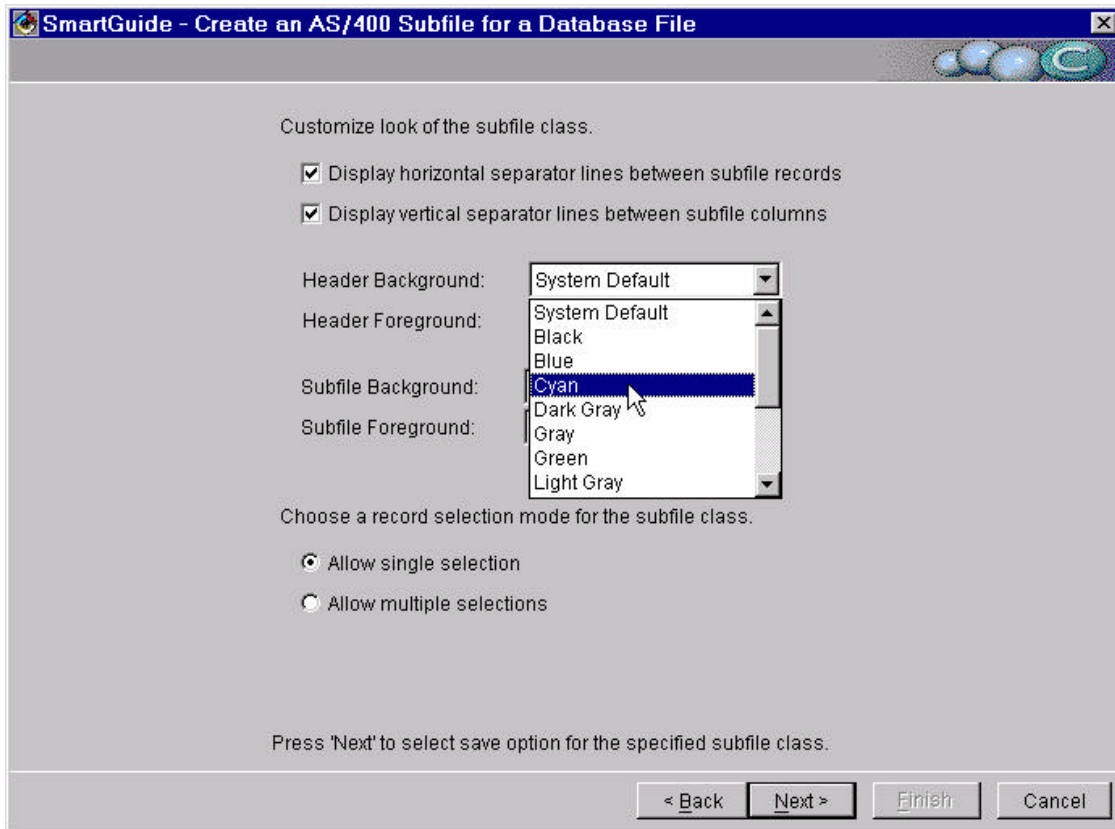
10. On the fifth page of the SmartGuide, do the following:
   - Click on the down arrow on the **Header Background:** field. A choice list is displayed. Select the color of your choice from the list. Choose a light color, otherwise you will not be able to view the headers displayed in black (System default foreground color) properly.
   - Click on the down arrow on the **Subfile Background:** field. A choice list is displayed. Select the color of your choice from the list. Choose a light color, otherwise you will not be able to view the subfile information displayed in black (System default foreground color) properly.
   - Leave other values unchanged as defaults. Note that the horizontal and vertical separator lines will be displayed and the single selection mode will be set in the generated code. Click the **Next** button. The sixth page of the **Create Subfile SmartGuide** is displayed.
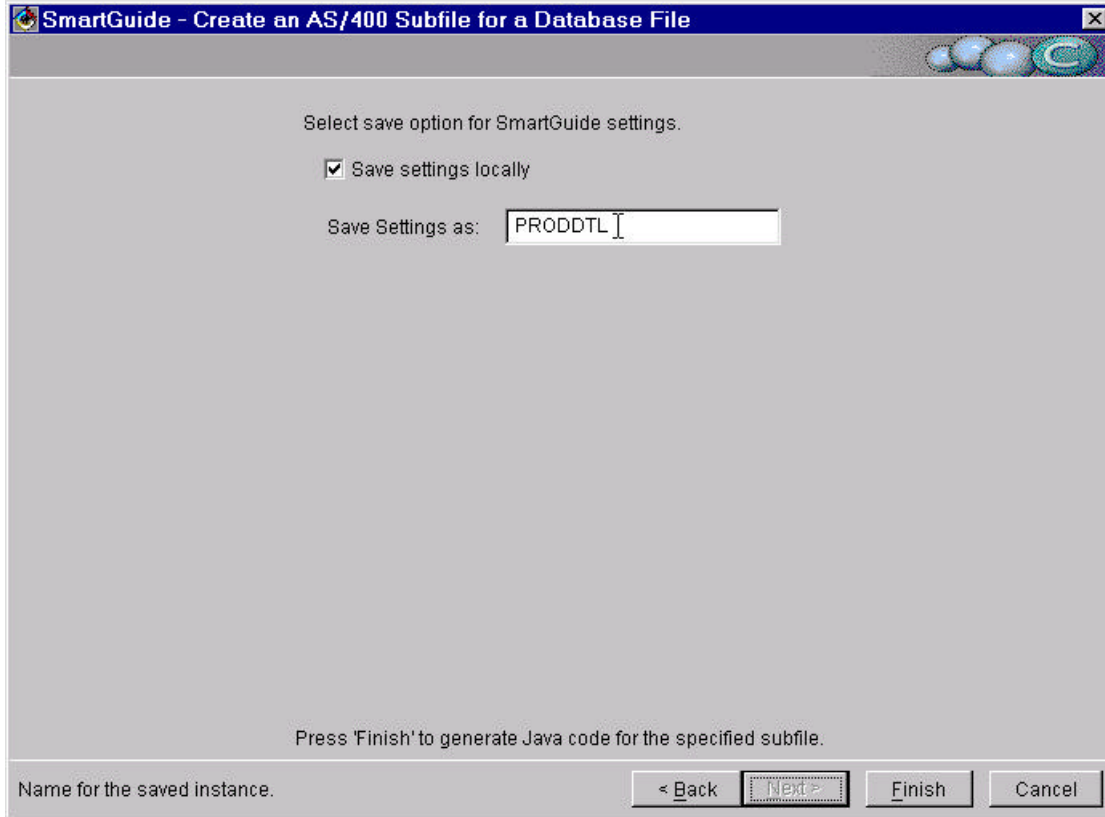
11. On the sixth page of the SmartGuide, leave the values unchanged. By default, the SmartGuide settings are saved in a file with the same name as the file name. These settings can be reused later. Click the **Finish** button.

    The SmartGuide is closed and the focus returns to the **Workbench**. The SmartGuide generates the code in the selected package, in this case in the VJ400LAB6 package.

12. Click on the ⊞ beside the **VJ400LAB6** package. Notice that the following three classes were generated and added to the package.

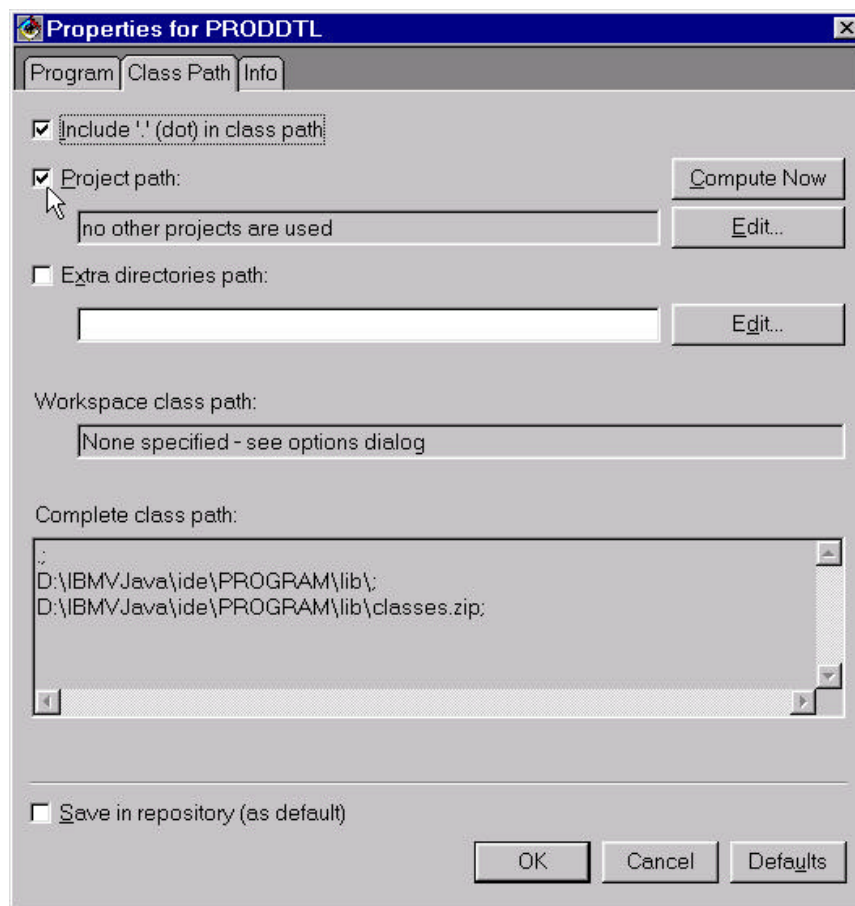| **PRODDTL**	Runable class that contains all the methods to perform open, close, read, update, and delete functions on the file
PRODDTLAccessoryPanel	Helper GUI application to test the main subfile class
PRODDTLWindowListener	Helper class that implements the WindowListener

    Expand the classes and review the generated code.

Run the Subfile

The main subfile class is a runable Java GUI application. This section shows you how to start and use it to verify the generated code quickly and easily.

To run the application, you need to setup its classpath to ensure that it can access the required resources from the **IBM Enterprise Toolkit for AS/400** and **JFC class libraries** (*Java Foundation Classes*) projects.

1. Click on the **PRODDTL** class to select it.
2. Click **Mouse Button 2**. Select **Properties** from the pop-up menu. The **Properties** window is displayed.
3. Click on the **Class Path** tab on the Properties window. The **Class Path** page is displayed.

```
┌─────────────────────────────────────────────────────────────┐
│ ◆ Properties for PRODDTL                              ✕ │
├─────────────────────────────────────────────────────────────┤
│ │Program│ Class Path │ Info │                                │
│                                                               │
│  ☑ Include '.' (dot) in class path                           │
│                                                               │
│  ☑ Project path:                        ┌─ Compute Now ─┐     │
│    ┌──────────────────────────────┐     └───────────────┘     │
│    │ no other projects are used   │     ┌──── Edit... ──┐     │
│    └──────────────────────────────┘     └───────────────┘     │
│  ☐ Extra directories path:                                   │
│    ┌──────────────────────────────┐     ┌──── Edit... ──┐     │
│    │                              │     └───────────────┘     │
│    └──────────────────────────────┘                          │
│                                                               │
│  Workspace class path:                                       │
│    ┌──────────────────────────────────────────────────┐      │
│    │ None specified - see options dialog              │      │
│    └──────────────────────────────────────────────────┘      │
│                                                               │
│  Complete class path:                                        │
│    ┌──────────────────────────────────────────────────┐▲     │
│    │ .;                                               │      │
│    │ D:\IBMVJava\ide\PROGRAM\lib\;                    │      │
│    │ D:\IBMVJava\ide\PROGRAM\lib\classes.zip;         │      │
│    │                                                  │▼     │
│    │◄                                              ►│      │
│    └──────────────────────────────────────────────────┘      │
│                                                               │
│  ☐ Save in repository (as default)                           │
│                          ┌─ OK ─┐ ┌ Cancel ┐ ┌ Defaults ┐    │
│                          └──────┘ └────────┘ └──────────┘    │
└─────────────────────────────────────────────────────────────┘
```

4. Click on the **Project path:** check box to check it.
5. Click on the **Edit...** button. The **Class Path** window is displayed.

---
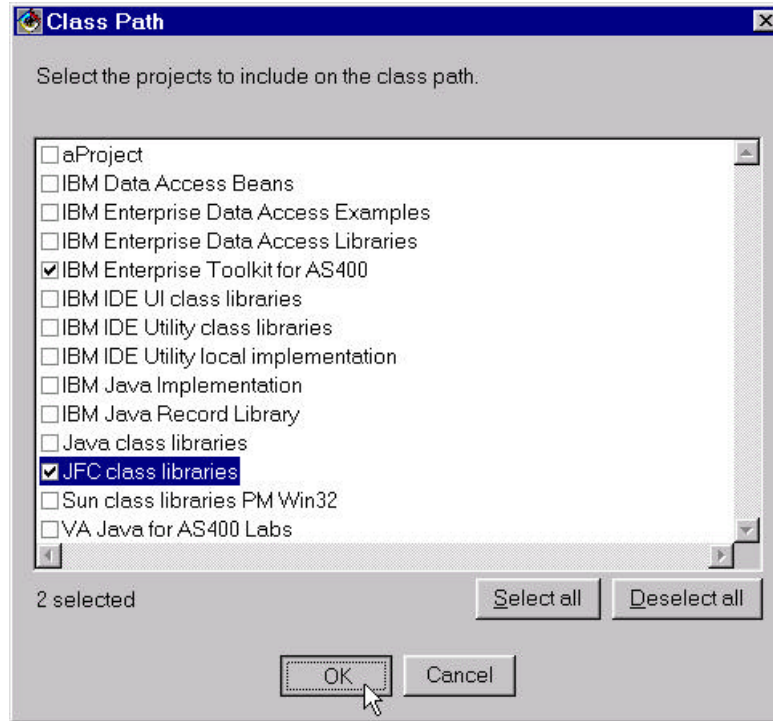
6. Click on the **IBM Enterprise Toolkit for AS400** check box to check it.
7. Click on the **JFC class libraries** check box to check it.
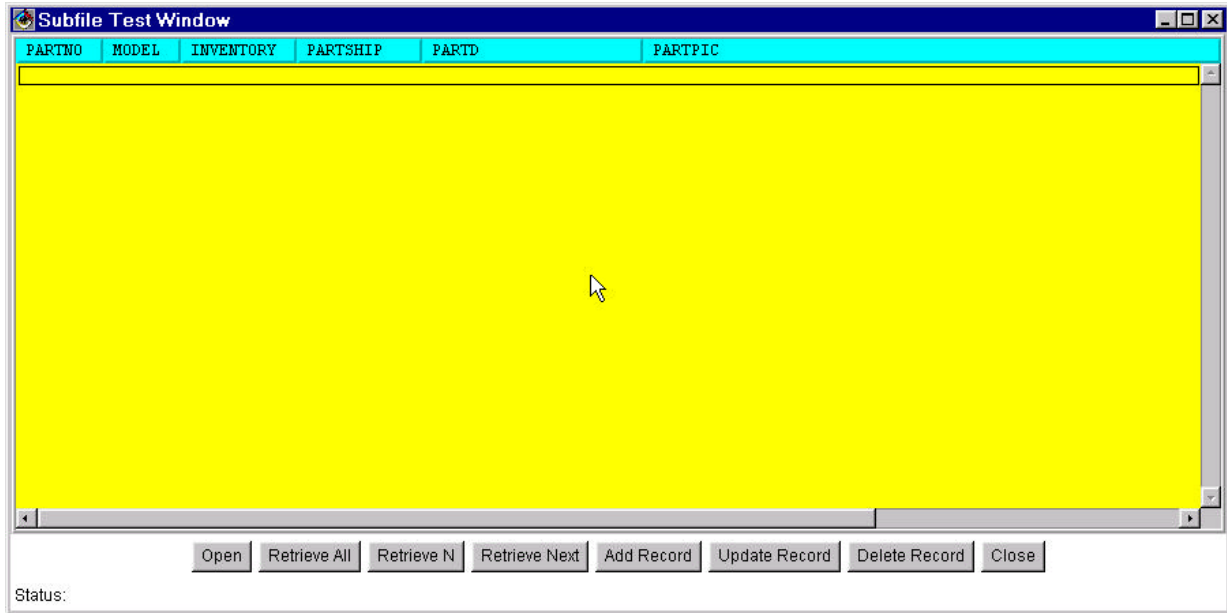
   <u>**NOTE:**</u> You don't need to click on the **VA Java for AS400 Labs** project. **VA Java** automatically adds the current project in the classpath.
8. Click **OK**. Click **OK** again to close the **Properties** window.

   Now you are ready to run the application.
9. From the **Workbench** menu bar, select **Selected -> Run -> Run main...**. (Make sure that PRODDTL class is selected). The **Subfile Test Window** is displayed.

Tip: If you receive

```
Exception occurred in main():java.lang.NoClassDefFoundError:
com.ibm.ivj.et400.util.SubfileHeading
```

in the **VA Java Console** window, then repeat previous steps 1 to 8. This occurs when the classpath is not setup correctly.

10. Click the **Open** button. The **Signon to AS/400** dialog is displayed. Enter the Userid and Password provided to you. Note that the system name is already filled in. Click **OK**.

   A connection to the AS/400 host is made. The database file on the host is opened. A message: "*Status: File opened*." is displayed in the Status bar at the bottom of the application window.

11. Click the **Retrieve All** button.

   A message: "*Status: Retrieving all records...*" is displayed in the Status bar at the bottom of the application window. The subfile displays all the records once they are retrieved. The status message changes to: "*Status: Records Retrieved.*".

12. Click on a record to select it. Now click on another record to select it holding the CTRL or SHIFT key. Does it allow you to select multiple records? If not, then why?

13. Write the PARTNO for the selected record _____. Click the **Delete Record** button.

The record is deleted from the file. A message: "*Status: Record deleted.*" is displayed in the Status bar at the bottom of the application window. Verify that the record is deleted from the visual subfile.

14. Click on a record to select it. Click the **Update Record** button. The **Add/Update Dialog** displays the selected record values.

---

15. Change the inventory value. Click **OK**. The **Add/Update Dialog** is closed.

The record is updated in the file. A message: "*Status: Record updated.*" is displayed in the Status bar at the bottom of the application window.

NOTE: Add/Update Dialog is designed to handle only 15 fields. This number is the total number of fields in the file including hidden fields.

16. Click the **Add Record** button. The **Add/Update Dialog** is displayed.
17. Enter appropriate values for a new record. Click **OK**. The **Add/Update Dialog** is closed.

The record is added to the file. A message: "*Status: Record added.*" is displayed in the Status bar at the bottom of the application window.

NOTE: This file doesn't prevent you from entering duplicate records.

18. Click the **Close** button.

The file is closed on the AS/400 host. The AS/400 connection is not closed. All records are cleared from the visual subfile. A message: "*Status: File closed.*" is displayed in the Status bar at the bottom of the application window.

19. Click the **Open** button to reopen the file.

The file is reopened on the AS/400 host.

20. Click **Retrieve N** button.

The first 5 records of the file are displayed in the visual subfile.

NOTE: The value of N is stored in the **retrievalBlock** variable in the PRODDTL class definition. It is set to 5 at the time of code generation. It can be changed.

21. Click on **Retrieve Next** button.

The next (sixth) record is displayed in the visual subfile.

22. Click on "X" on the right hand top corner of the visual subfile to close the application.

Bravo! You generated subfile classes and verified them successfully.

Exercise 3. Build GUI Application Using the Subfile (Optional)

Introduction

In this exercise, you will build a GUI application that makes use of the generated subfile using **VA Java's Visual Composition Editor (VCE),** commonly known as **Visual Builder**.

Goal

At the end of this section you will be able to:

* Design a GUI application that makes use of the generated subfile code
* Work with sequential database files on an AS/400 host using GUI application

Let us design it...

Creating a Frame for Your Visual Application

1. Select the VJ400LAB6 package.
2. Click on the ⓒ **Create Class** icon on the tool bar of the **Workbench**. The **Create Class SmartGuide** is displayed.
3. Leave the **Create a new class radio button** selected. Enter InventoryList in the **Class name** field.
4. Enter java.awt.Frame in the **Superclass** field.
5. Click **Compose the class visually** check box to check it.
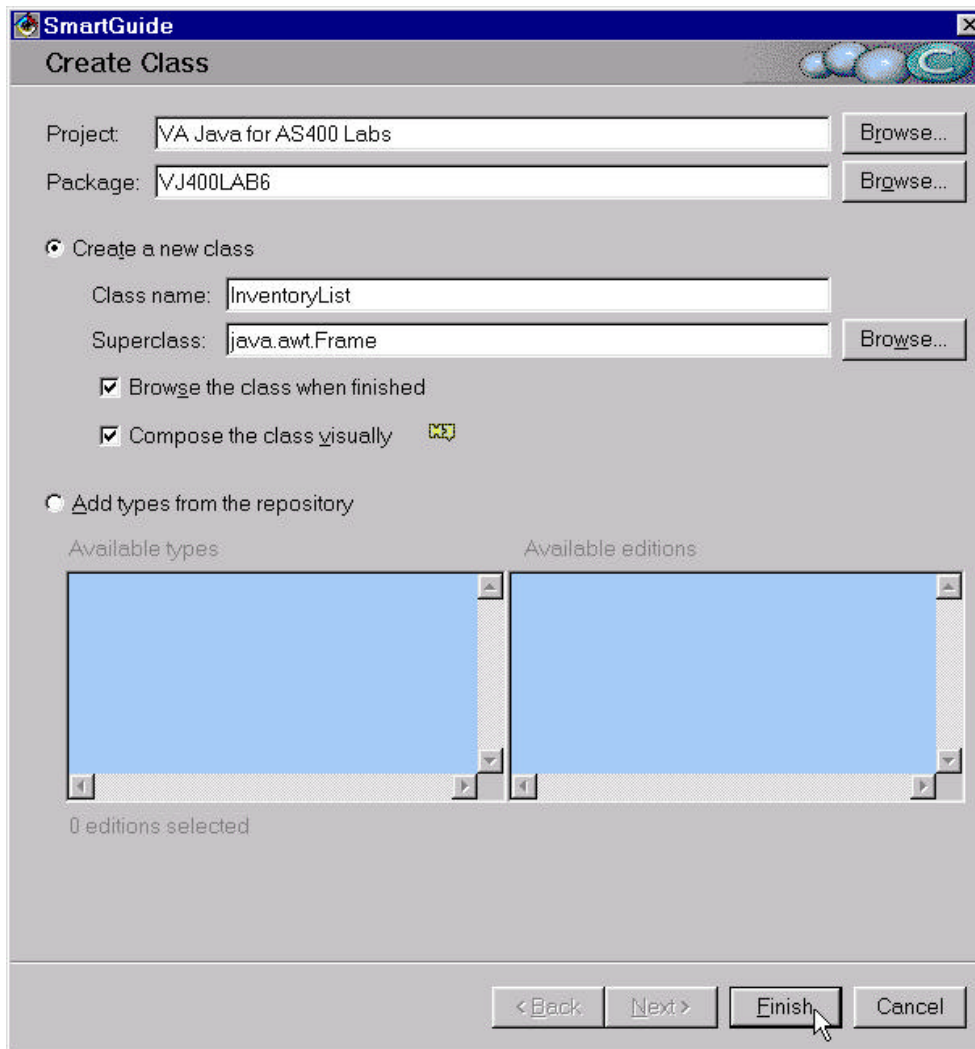
6. Click the **Finish** button. The SmartGuide disappears and the **Visual Composition Editor** appears.
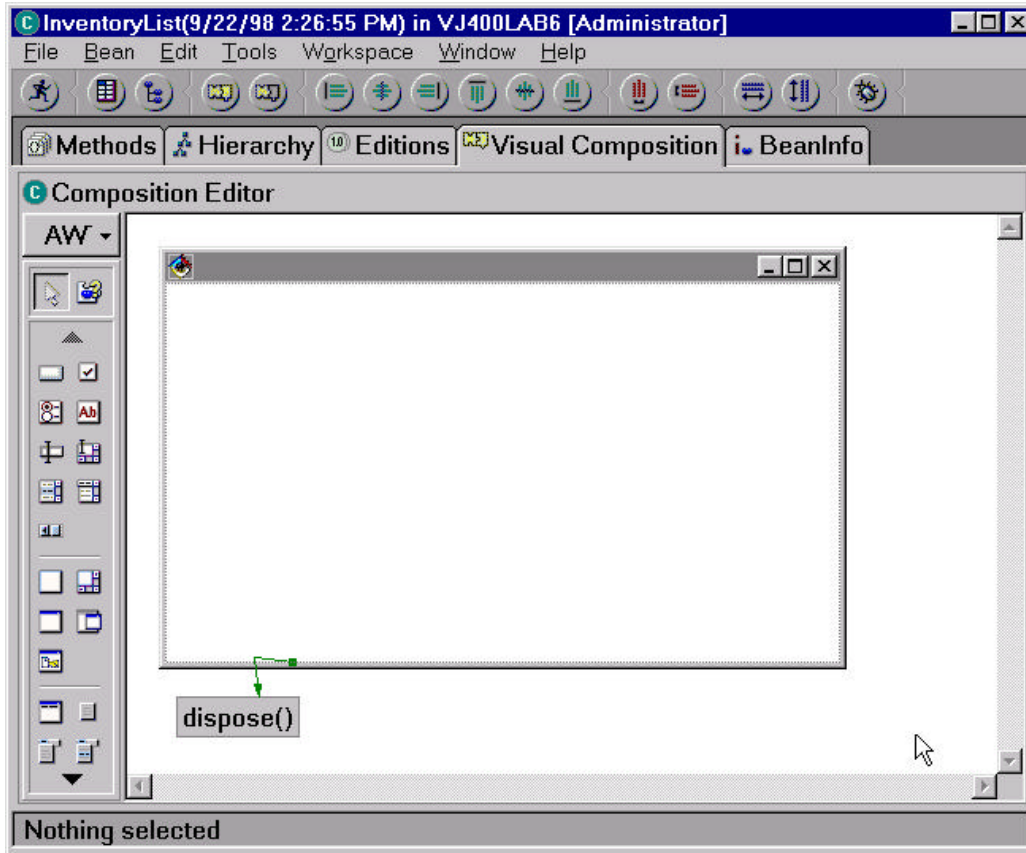
7. Maximize the **VCE** window to make it easier to design.
8. Click on the title bar of the Frame window to select it. Hold the corner handles and drag them to increase the size of the frame. Cover approximately ¾ of the white space (free form surface).

Creating a Title for the Application

In this section you will use a label bean to create a title for the user interface. The title will be in bold font and centered on top of the window.

1. Select the **AWT** category from the Beans Palette list, if not selected.
2. Click on the **Label** bean. Drop the Label bean onto the upper center area of the frame by moving the "+" cursor to the desired position and single clicking **Mouse Button 1**.

   Hint: To find the name of any bean on the Beans Palette, bring the cursor onto the bean and slowly move it over it. Fly over (bubble) help will show the bean name.
3. Click **Mouse Button 2** on the **Label1** bean and select **Properties**. The **Properties** window is displayed.

4. On the **Properties** window, click the value field to the right of the **alignment** and select **CENTER** from the list.
5. Click the value field to the right of the font. Click the ⬚ (list) button to open the font window. Select **Bold** from the **Style** list. Click **OK** to close the dialog.
6. Click the field to the right of the **text** and type Part Inventory.
7. Click "X" on the upper right corner to close the **Properties** window.
8. Click on the **Label1** label to select it. Use the corner handles to increase the size of the label to display the complete text.

Adding PRODDTL Class as a Bean

In this section, you will add the PRODDTL class as a visual bean to the GUI application you are designing.

1. Select **Choose Bean...** from the Beans Palette. The **Choose Bean** window is displayed.
2. Enter a few letters of the class name PRODDTL in the **Class name** field. Click **Browse...** button. **Choose a valid class** window is displayed.
3. Select the **PRODDTL** class from the **Class Names** list. Select **VJ400LAB6** from the bottom **Package Names** list. Click **OK**. The **Choose a valid class** window is closed.
4. **VJ400LAB6.PRODDTL** is added to the **Class name** field of the Choose Bean window. Leave the **Name** field blank.
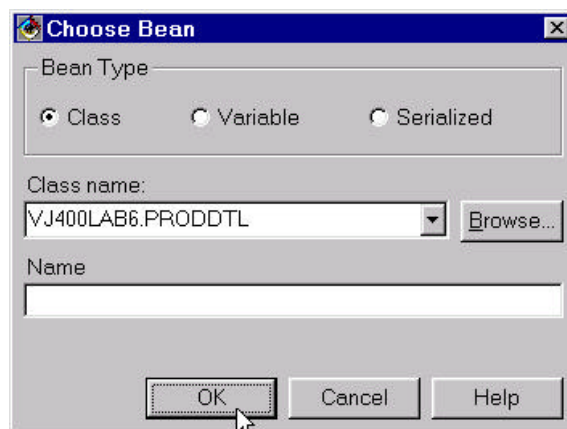
Click **OK**. The **Choose Bean** window is closed.
5. Move the cross hairs inside the frame and click **Mouse Button 1** to drop the bean.

NOTE: Make sure that the cross hairs are inside the frame before you click, otherwise the bean will not be visible when you run the application.

The PRODDTL bean is added to the frame.

6. Click on the PRODDTL bean to select it. Move it to the inside left hand side edge of the frame.   Use the corner handles to re-size it.
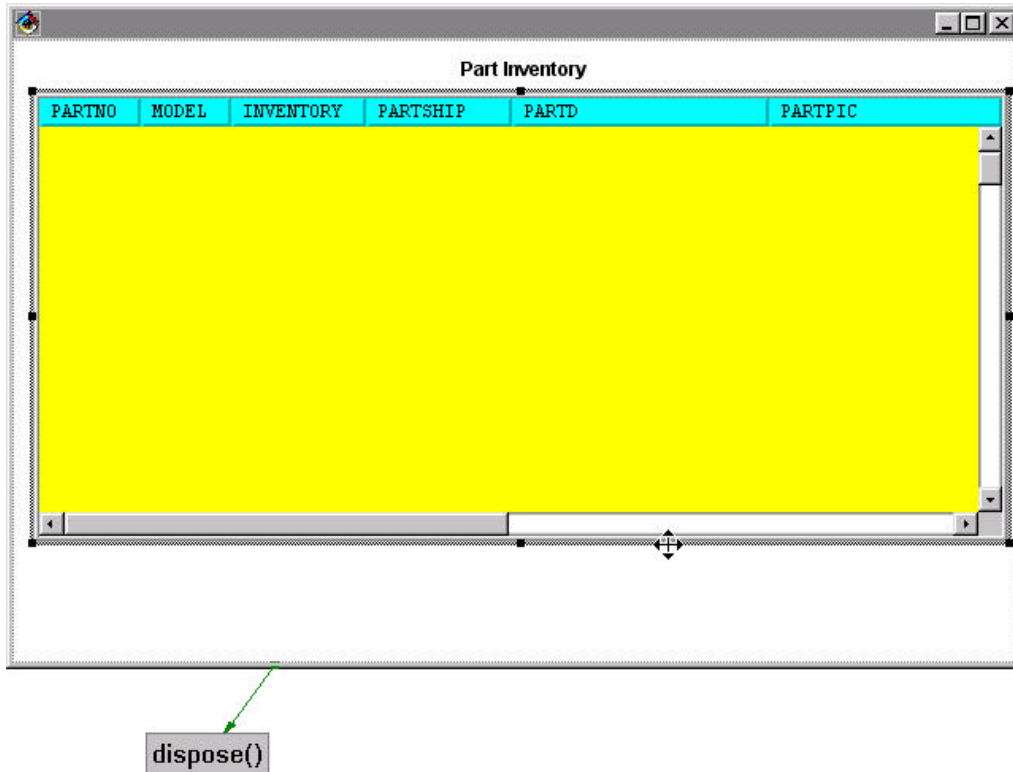
Adding an Open Button

You will add the **Open** action button here.

1. Click on the **Button** bean from the **AWT** Bean Palette. Drop it just below the left of the frame.
2. Click **Mouse Button 2**. Select **Properties** from the pop-up menu.
3. On the **Properties** window, click to the right of the **beanName**. Enter ButtonOpen.
4. Click to the right of the **label**. Enter Open. Click on the "X" on top right hand corner to close the **Properties** window.

Adding a ReadAll Button

You will add the **ReadAll** action button here.

1. Click on the **Button** bean from the **AWT** Bean Palette. Drop it to the right of the **Open** button.

2. Click **Mouse Button 2**. Select **Properties** from the pop-up menu.
3. On the **Properties** window, click to the right of the **beanName**. Enter ButtonReadAll.
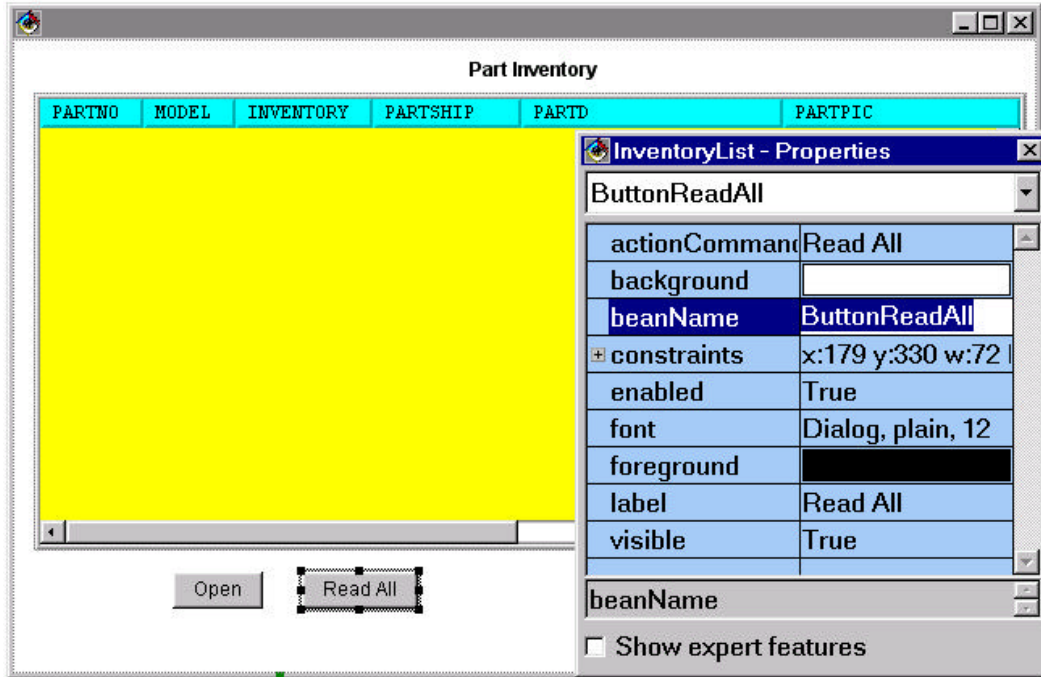4. Click to the right of the **label**. Enter Read All.
5. Click on the "X" on top right hand corner to close the **Properties** window.

Adding a Status Label

You will add a status bar at the bottom of the application window. It will be used to display the status of the major actions.

1. Click on the **Label** bean. Drop the bean at the bottom left corner of the frame. Click **Mouse Button 1**. Re-size the label to span the entire width of the frame window.
2. Click **Mouse Button 2** on the **Label2** bean and select **Properties**. The **Properties** window is displayed.
3. On the **Properties** window, click to the right of the **background**. Click the ▣ (list) button. The **Background** window is displayed. Click on the **Basic** radio button to select it. Select a light color of your choice. Click **OK** to close the **Background** window.
4. Click to the right of **text.** Delete the existing value to make it blank.
5. Click on the "X" on top right hand corner to close the **Properties** window.

Adding a Connection to Open the Database File

You will add a connection to open PRODDTL file for read and write when the **Open** button is clicked.

1. Click on the **Open** button to select it. Click **Mouse Button 2.** Select
   Connect -> actionPerformed from the pop-up menu.
2. The cursor turns into a spider. Drag and drop it onto the PRODDTL1 subfile bean. Click
   Mouse Button 1. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Method** radio button selected.
   Select **openSequentialFileReadWrite()** method from the list. Click **OK** to close the
   window. A connection shown with a solid green line displays. It means the connection is
   complete.
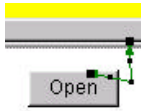
Displaying File Opened Status Message

You will add a connection to display a "File opened" status message, when the file is opened
successfully.

1. Click on the previous connection **(connEtoM1)** to select it. Click **Mouse Button 2.** Select
   Connect -> normalResult from the pop-up menu.
2. Drag and drop the spider onto the status bar (**LabelStatus** bean). Click **Mouse Button 1**.
   Select **text** from the pop-up menu. A connection is shown with a broken green line. This
   means the connection is incomplete and requires one or more parameters.
3. Click **Mouse Button 1** on the incomplete connection to select it. Click **Mouse Button 2.**
   Select **Properties** from the pop-up menu. An **Event-to-method connection - Properties**
   window is displayed.
4. Click on the **Set parameters...** button. A **Constant Parameter Value Settings** window is
   displayed.
5. Click to the right of **value**. Enter "File opened" (without quotes). Click **OK** to close the
   Constant Parameter Value Settings window.
6. Click **OK** to close the **Event-to-method connection - Properties** window. The connection
   turns solid green.

Displaying Exception Messages

You will add a connection to display any exception messages, if the file open fails.

1. Click on the previous connection **(connEtoM1: ButtonOpen, actionPerformed -->PRODDTL1,openSequentialFileReadWrite())** to select it. Click **Mouse Button 2.** Select **Connect -> exceptionOccured** from the pop-up menu.
2. Drag and drop the spider onto the status bar (**LabelStatus** bean). Click **Mouse Button 1**. Select **text** from the pop-up menu. A connection is shown with a broken green line. This means the connection is incomplete and requires one or more parameters.
3. Click **Mouse Button 1** on the incomplete connection to select it. Click **Mouse Button 2.** Select **Properties** from the pop-up menu. An **Event-to-method connection - Properties** window is displayed.
4. Click on the **Pass event data** check box to check it.
5. Click **OK** to close the **Event-to-method connection - Properties** window. The connection turns solid green.

Adding a Connection to Read All Records from the File

You will add a connection to read all records in the visual subfile bean when the **Read All** button is clicked.

1. Click on the **Read All** button to select it. Click **Mouse Button 2.** Select **Connect -> actionPerformed** from the pop-up menu.
2. The cursor turns into a spider.   Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Method** radio button selected. Select the **ReadAllRecords()** method from the list. Click **OK** to close the window. A connection is shown with a solid green line.

Clearing the Status Message

When all the records are read successfully, they are displayed in the visual subfile. There is no need to provide a visual clue using a status message. You will add a connection to clear the status message in this case.

1. Click on the previous connection **(connEtoM4: ButtonReadAll, actionPerformed -->PRODDTL1, ReadAllRecords())** to select it. Click **Mouse Button 2.** Select **Connect -> normalResult** from the pop-up menu.
2. Drag and drop the spider onto the status bar (**LabelStatus** bean). Click **Mouse Button 1**. Select **text** from the pop-up menu. A connection is shown with a broken green line. This means the connection is incomplete and requires one or more parameters. Leave the connection as is. This will use the default value of blank for the text.

Displaying Exception Messages

You will add a connection to display any exception messages, if the file read fails.

1. Click on the previous connection **(connEtoM4: ButtonReadAll, actionPerformed -->PRODDTL1, ReadAllRecords())** to select it. Click **Mouse Button 2.** Select **Connect -> exceptionOccured** from the pop-up menu.
2. Drag and drop the spider onto the status bar (**LabelStatus** bean). Click **Mouse Button 1**. Select **text** from the pop-up menu. A connection is shown with a broken green line. This means the connection is incomplete and requires one or more parameters.
3. Click **Mouse Button 1** on the incomplete connection to select it. Click **Mouse Button 2.** Select **Properties** from the pop-up menu. An **Event-to-method connection - Properties** window is displayed.
4. Click on the **Pass event data** check box to check it.
5. Click **OK** to close the **Event-to-method connection - Properties** window. The connection turns solid green.

Save the GUI Application

You will save the application you are designing.

Click on the **Save Bean** from the **Bean** menu of the **VCE**. Information window displays messages. The InventoryList bean is saved in the VJ400LAB6 package.

Test the GUI Application

You will test the application you have designed. However, before you can run your application you need to set the correct class path to find the required classes when the application is run.

1. Click on the **Run -> Check Class Path...** from the **Bean** menu of the **VCE**. The **Properties for InventoryList** window is displayed. Click on the **Project path:** check box to check it. Click on the **Edit...** button.
2. Click on the **IBM Enterprise Toolkit for AS400** check box to check it.
3. Click on the **JFC class libraries** check box to check it.

   **NOTE:** You don't need to click on the **VA Java for AS400 Labs** project. VA Java automatically adds the current project in the classpath.
4. Click **OK**. Click **OK** again to close the **Properties** window.
5. Click on the **Run -> Run main...** from the **Bean** menu of **VCE**. The **Part Inventory** application window is displayed. Re-size it to see all the visual components.

---
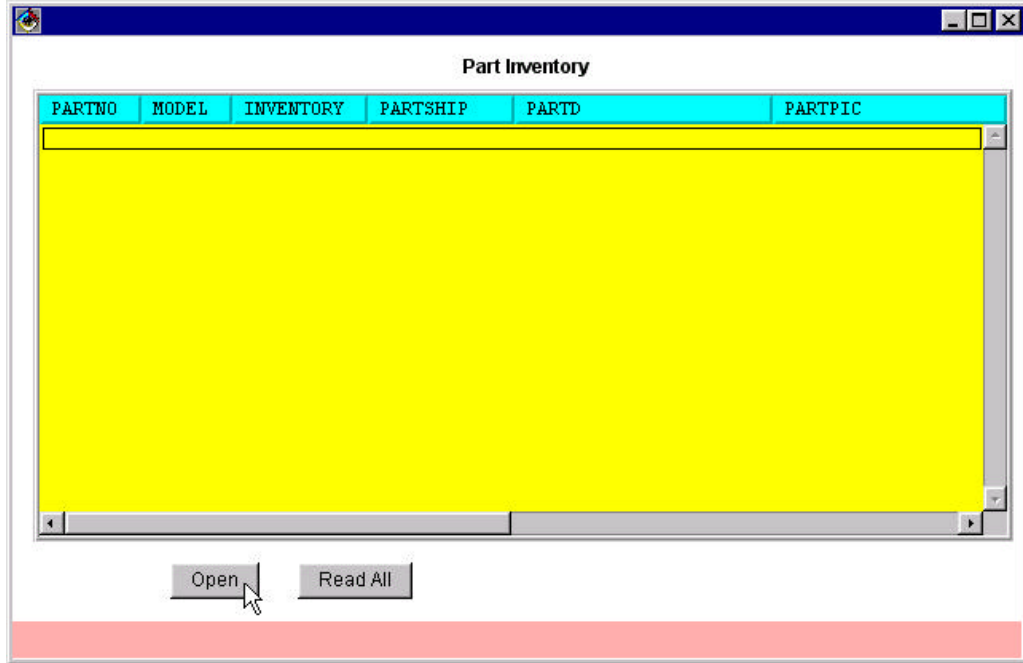
6. Click on the **Open** button. The **Signon to AS/400** dialog is displayed. Enter the Userid and Password given. Leave the **Default User ID** and **Save password** check boxes checked.
7. Click **OK** to close the **Signon to AS/400** dialog, make the AS/400 connection and open the file. "File opened." status message is displayed in the status bar.
8. Click on the **Read All** button. All records are retrieved from the database file and displayed in the visual subfile. The old message from the status bar is deleted.
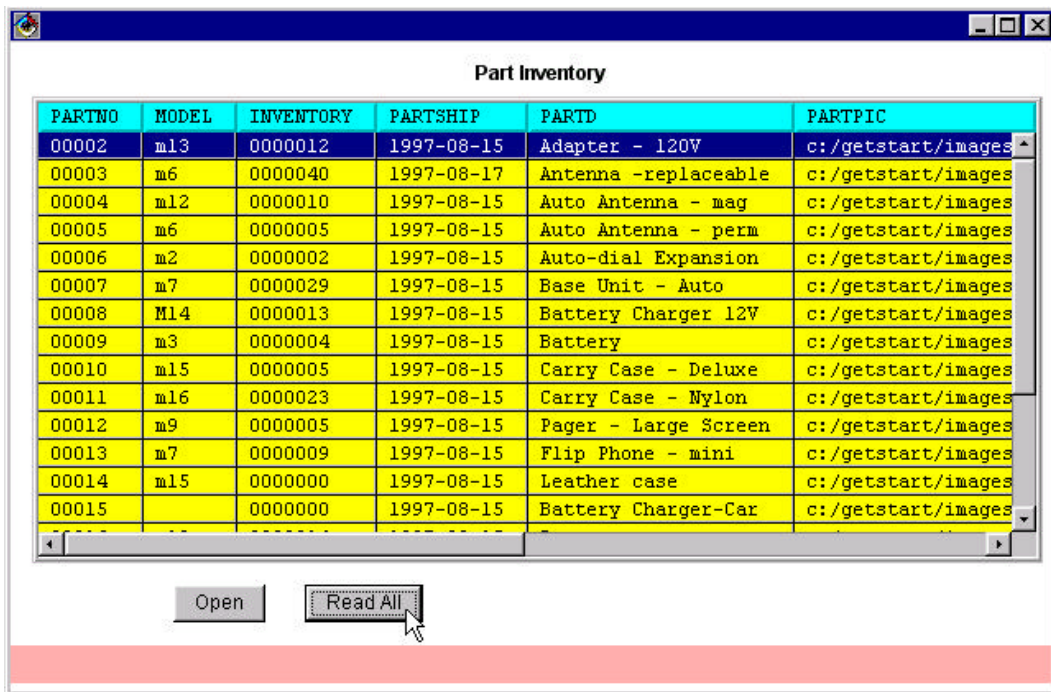
9. Click on the "X" on top right hand corner to close the application window.

Isn't it great to see that the application you designed actually works?

Adding a Delete Button

You will add the **Delete** action button here.

1. Click on the **Button** bean from the **AWT** Bean Palette. Drop it to the right of the **Read All** button.
2. Click **Mouse Button 2**. Select **Properties** from the pop-up menu.
3. On the **Properties** window, click to the right of the **beanName**. Enter ButtonDelete.
4. Click to the right of the **label**. Enter Delete.
5. Click on the "X" on top right hand corner to close the **Properties** window.

Adding an Update Button

You will add an **Update** action button here.

1. Click on the **Button** bean from the **AWT** Bean Palette. Drop it on the left of the **Delete** button.
2. Click **Mouse Button 2**. Select **Properties** from the pop-up menu.
3. On the **Properties** window, click to the right of the **beanName**. Enter ButtonUpdate.
4. Click to the right of the **label**. Enter Update.
5. Leave the **Properties** window open.

Adding a Connection to Delete a Record in the File

You will add a connection to delete the selected record from the visual subfile bean when the **Delete** button is clicked.

1. Click on the **Delete** button to select it. Click **Mouse Button 2.** Select **Connect -> actionPerformed** from the pop-up menu.
2. The cursor turns into a spider.   Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Method** radio button selected. Select the **deleteRecord(int)** method from the list. Click **OK** to close the window. An incomplete connection is shown with a dashed green line.

Adding a Connection to Get the Focus Record Number

The **deleteRecord(int)** method called in the previous connection required an *int* parameter leaving the connection incomplete. You will add a connection to pass the selected (focused) record number as the parameter to it.

1. Click on the previous connection (**connEtoM7)** to select it. Click **Mouse Button 2.** Select **Connect -> recordNumber** from the pop-up menu.
2. The cursor turns into a spider.   Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Properties** radio button selected. Select the **focusRecordNumber** property from the list. Click **OK** to close the window. A connection is shown with a solid purple line. The previous connection turns solid green.

Adding a Connection to Delete the Record from the Subfile

The **deleteRecord(int)** method called in the previous connection deletes the selected record from the database file on the AS/400. But it does not refresh the visual subfile automatically. You have two choices: click on the **Read All** button to read all records, or delete the selected record from the visual subfile. You will add a connection to perform the later.

1. Click on the connection (**connEtoM7: buttonDelete, actionPerformed -->PRODDTL1, deleteRecord(int)**) to select it. Click **Mouse Button 2.** Select **Connect -> normalResult** from the pop-up menu.
2. The cursor turns into a spider.   Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Method** radio button selected. Select the **RemoveSubfileRecord(int)** method from the list. Click **OK** to close the window. An incomplete connection is shown with a dashed green line.

Adding a Connection to Get the Focus Record Number

The **RemoveSubfileRecord(int)** method called in the previous connection required an *int* parameter leaving the connection incomplete. You will add a connection to pass the selected record number as the parameter to it.

1. Click on the previous connection (**connEtoM8)** to select it. Click **Mouse Button 2.** Select **Connect -> arg1** from the pop-up menu.
2. The cursor turns into a spider.   Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Properties** radio button selected. Select the **focusRecordNumber** property from the list. Click **OK** to close the

window. A connection is shown with a solid purple line. The previous connection turns solid green.

Adding a Connection to Update a Record in the File

You will add a connection to update the selected record in the visual subfile bean when the **Update** button is clicked.

1. Click on the **Update** button to select it. Click **Mouse Button 2.** Select **Connect -> actionPerformed** from the pop-up menu.
2. The cursor turns into a spider. Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Method** radio button selected. Select the **updateRecord(int, java.lang.String[])** method from the list. Click **OK** to close the window. An incomplete connection is shown with a dashed green line.

Adding a Connection to Get the Focus Record Number

The **updateRecord(int, java.lang.String[])** method called in the previous connection required two parameters. You will add a connection to pass the selected record number as the *int* parameter to it.

1. Click on the previous connection (**connEtoM9: buttonUpdate, actionPerformed -->PRODDTL1, updateRecord(int,java.lang.String[])**) to select it. Click **Mouse Button 2.** Select **Connect -> recordNumber** from the pop-up menu.
2. The cursor turns into a spider. Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Properties** radio button selected. Select the **focusRecordNumber** property from the list. Click **OK** to close the window. A connection is shown with a solid purple line. The previous connection is still a dashed green line.

Adding a Connection to Get the Selected Record as Strings

The **updateRecord(int, java.lang.String[])** method called in the previous connection required two parameters. You will add a connection to pass the selected record data as the **String[]** to it.

1. Click on the previous incomplete connection (**connEtoM9:buttonUpdate, actionPerformed -->PRODDTL1, updateRecord(int,java.lang.String[])**) to select it. Click **Mouse Button 2.** Select **Connect -> recordData** from the pop-up menu.
2. The cursor turns into a spider. Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.

3. On the **End connection to (PRODDTL1)** window, select the **Method** radio button selected. Select the **getRecordAsStrings(int)** method from the list. Click **OK** to close the window. A connection is shown with a dashed purple line.

Adding a Connection to Get the Focus Record Number

The **getRecordAsStrings(int)** method called in the previous connection **connPfromM1** required an *int* parameter. You will add a connection to pass the selected record number as the *int* parameter to it.
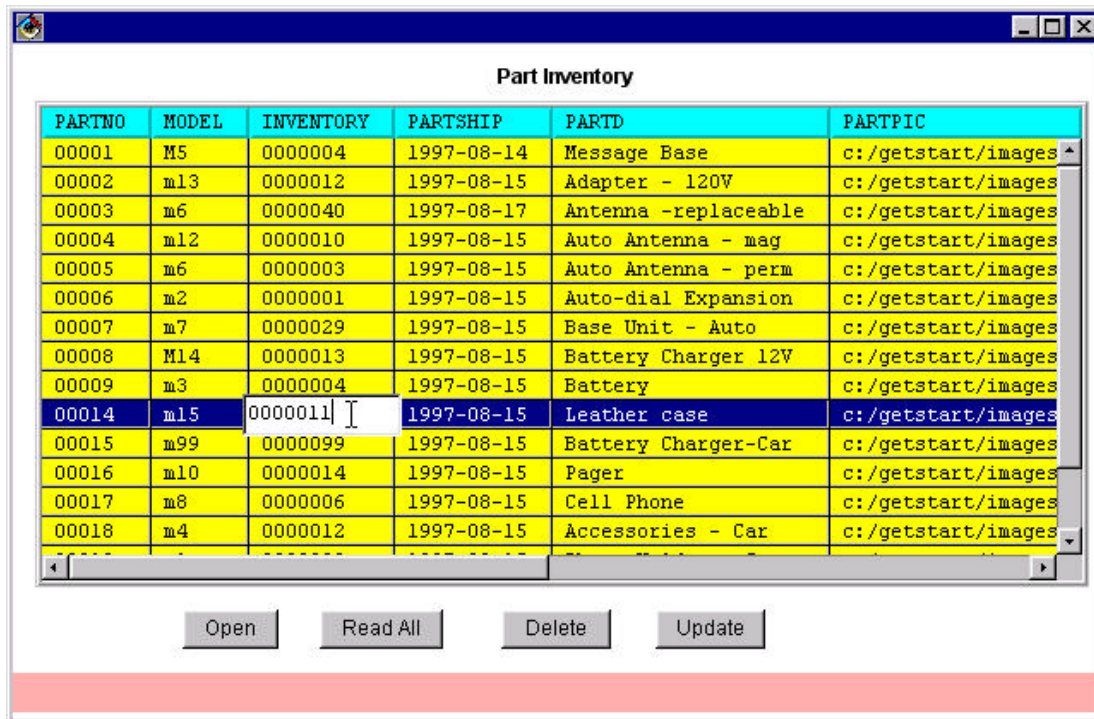
1. Click on the previous connection (**connPfromM1**) to select it. Click **Mouse Button 2.** Select **Connect -> arg1** from the pop-up menu.
2. The cursor turns into a spider. Drag and drop it onto the PRODDTL1 subfile bean. Click **Mouse Button 1**. Select **Connectable Features...** from the pop-up menu.
3. On the **End connection to (PRODDTL1)** window, leave the **Properties** radio button selected. Select the **focusRecordNumber** property from the list. Click **OK** to close the window. A connections is shown with a solid purple line. The previous connection (**connEtoM9**) is now a solid green line.
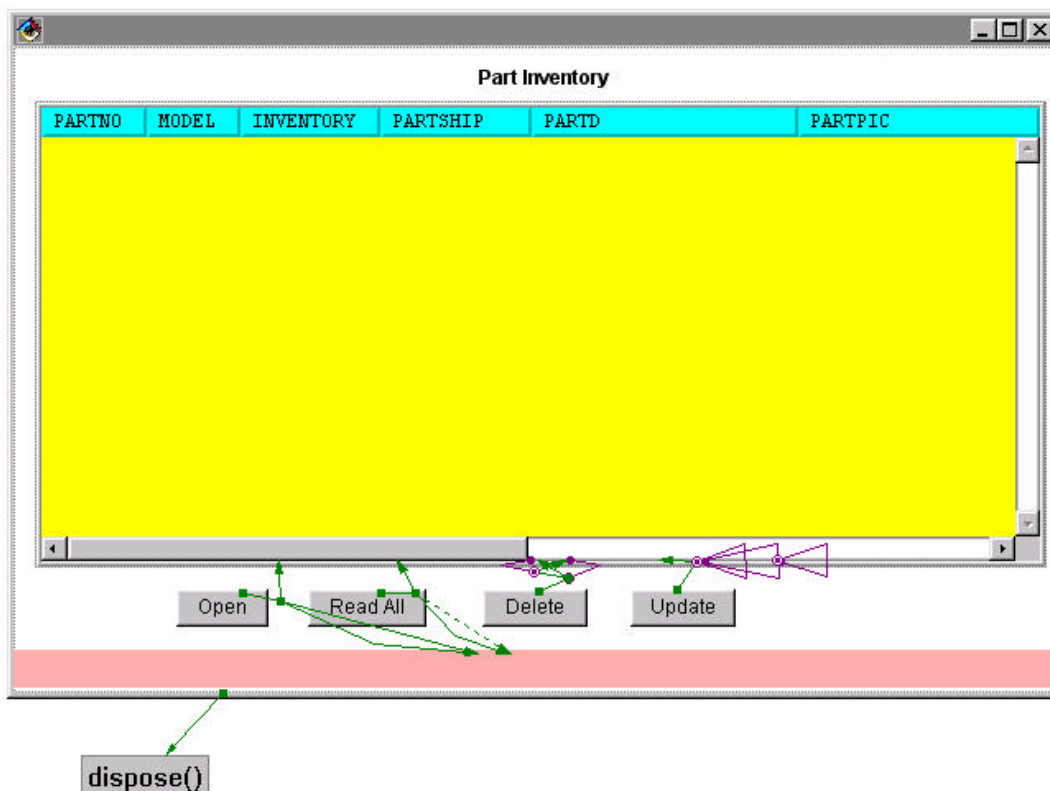
Test the GUI Application

Now you can test your application. Since you have setup the classpath earlier, you don't need to do it again.

1. Click on the 🏃 from the tool bar of the **VCE**. Your application is saved. **Part Inventory** application window is displayed. Re-size it to see all visual components.
2. Click on the **Open** button. The **Signon to AS/400** dialog displays. Enter the Userid and Password given. Leave the **Default User ID** and **Save password** check boxes checked.
3. Click **OK** to close the **Signon to AS/400** dialog, make the AS/400 connection and open the file. "File opened." status message is displayed in the status bar.
4. Click on the **Read All** button. All records are retrieved from the database file and displayed in the visual subfile. The old message from the status bar is deleted.

5. Select a record on the subfile. Click on the **Delete** button. The selected record is deleted and removed from the visual subfile.
6. Select another record on the subfile. Move the cursor on to the INVENTORY cell of the selected record and click **Mouse Button 1** while holding the ALT key. Now the cell is editable. Enter any value less than 5 and press the ENTER key. Does it allow you? If not, why not?

   Hint: Do you remember setting the validity checking on the INVENTORY column?

Enter a valid INVENTORY value. Press the ENTER key. Click on the **Update** button. The record is updated on the visual subfile and in the AS/400 file.

7. Click on the **Read All** button to refresh the data. Verify your previous delete and update actions.

8. Click on the "X" on top right hand corner to close the application window.

You are very persistent. Now you are a great VisualAge Java programmer!

Exercise 4. Accessing Data From Keyed Files (ADVANCED - OPTIONAL)

Introduction

This section will introduce you to:

- The **Create Subfile SmartGuide** to generate the subfile code for keyed database files
- Building a GUI application to work with a keyed file
- Working with keyed database files on an AS/400 host using the generated subfile code

Goal

At the end of this section you will be able to:

- Use the **Create Subfile SmartGuide** to generate the subfile code for keyed database files
- Work with keyed database files on an AS/400 host.

Let us do it...

Generate Subfile Code

In this section, you will use the **Create Subfile SmartGuide** to generate the Java classes that allow you to access keyed database file PRODDTL on the AS/400 host. These classes use the record-level access classes of the **AS/400 Toolbox for Java**. Notice that until now you accessed the PRODDTL file as a sequential database file. However, the PRODDTL file is a keyed file with PARTNO as the key. So in this exercise we will access it as a keyed file. You do not need to create the subfile classes again, since the keyed file access methods are already created for you in the generated PRODDTL classes. However, you will create these classes again to make only few fields visible.

1. Make sure the **VJ400LAB6** package is selected on the **Workbench**. From the **Workbench** menu bar select **Selected -> Tools -> ET/400 -> Create Subfile....** The first page of the **Create Subfile SmartGuide** is displayed.
2. On the first page of the **Create Subfile SmartGuide**, select **Create a new subfile class** radio button. Click the **Next** button. The second page of the **Create Subfile SmartGuide** is displayed.
3. On the second page of the SmartGuide, enter the following data:
    - In the **Server** field select the Server name given to you.
    - Enter PRODDTL in the **File** field.
    - Enter JAVAxx (where xx is the workstation number between 01 to 99) in the **Library** field.
    - Enter PRODDTLKY in the **Class** field.
    - Leave the default settings for the **Project and Package** fields. Notice that these values are automatically filled in for you.
    - Click the **Next** button. The **Signon to AS/400** dialog is displayed. Enter the Server name, Userid and Password and click **OK** button.

      The SmartGuide makes an AS/400 connection and retrieves the record format information on the file. Then the third page of the **Create Subfile SmartGuide** is displayed.

---
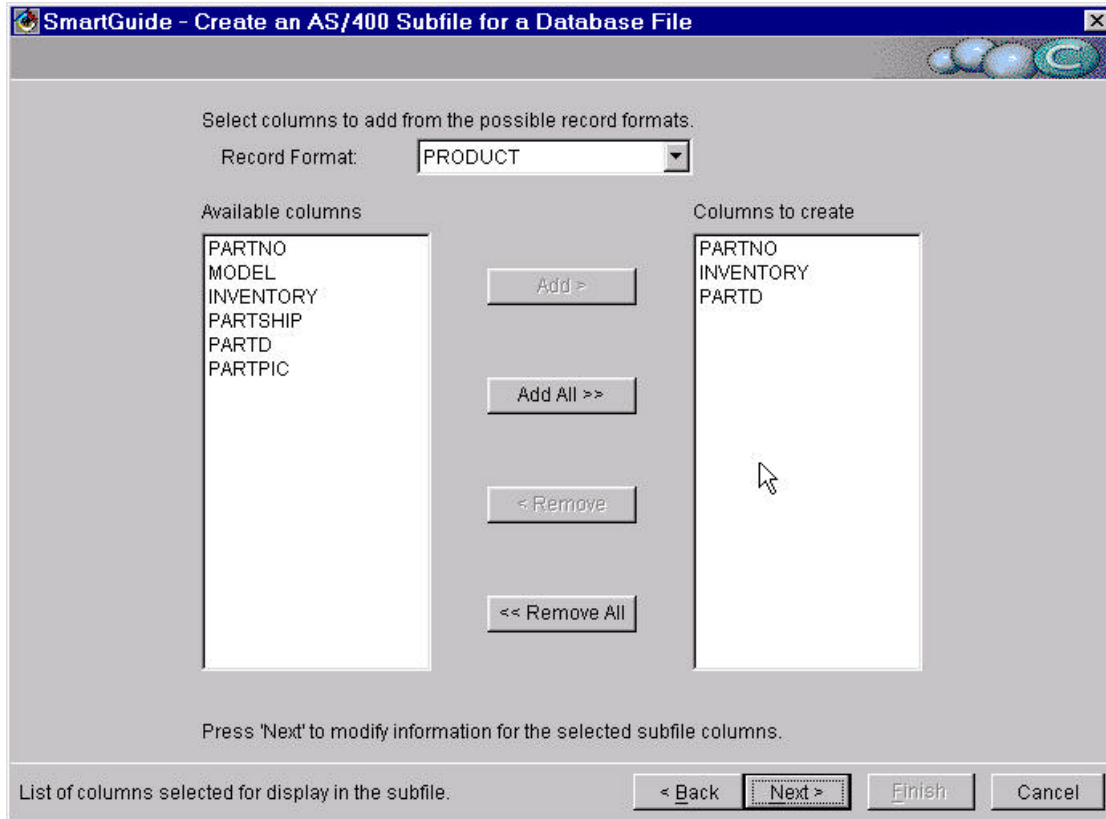
4. On the third page of the SmartGuide, do the following:
    *   Leave the Record Format field unchanged, since you have only one record format.
    *   Select PARTNO, INVENTORY and PARTD columns from the **Available columns** list. Click on the **Add** button to add selected columns to the **Columns to create** list.
    *   Click the **Next** button.

    The SmartGuide retrieves the file field information.
5. On the fourth page of the SmartGuide, leave the default values unchanged. Click the **Next** button.
6. On the fifth page of the SmartGuide, leave the default values unchanged. Click the **Next** button.
7. On the sixth page of the SmartGuide, leave the default values unchanged. Click the **Finish** button.

    The SmartGuide is closed and PRODDTLKY classes are generated in the VJ400LAB6 package.

Build a GUI Application to Work with the Keyed Subfile

In this section, you will build a GUI application that allows you to work with the PRODDTLKY keyed subfile code.

1. Select the VJ400LAB6 package.
2. Click on the ⓒ **Create Class** icon on the tool bar of the **Workbench**. The **Create Class SmartGuide** displays.
3. Leave the **Create a new class radio button** selected. Enter InventoryKeyed in the **Class name** field.
4. Enter java.awt.Frame in the **Superclass** field.
5. Click the **Compose the class visually** check box to check it.
6. Click the **Finish** button. The SmartGuide disappears and the **Visual Composition Editor** appears.
7. To add a title, select the **AWT** category from the Beans Palette list on the **VCE**, if not selected. Click on the **Label** bean. Drop the Label bean onto the upper center area of the frame by moving the "+" cursor to the desired position and clicking **Mouse Button 1**. Click **Mouse Button 2** on the **Label1** bean and select **Properties**. The **Properties** window is displayed. On the **Properties** window:
    *   Click the value field to the right of the **alignment** and select **CENTER** from the list.
    *   Click the value field to the right of the **font**. Click the ⬚ (list) button to open the **font** window. Select **Bold** from the **Style** list. Click **OK** to close the dialog.
    *   Click the field to the right of the **text** and type Keyed Inventory.
    *   Click "X" on the upper right corner to close the **Properties** window.
    *   Click on the **Label1** label to select it. Use the corner handles to increase the size of the label to display the complete text.

8. To add the PRODDTLKY bean, select **Choose Bean...** from the

    Beans Palette. The **Choose Bean** window is displayed.
9. Enter a few letters of the class name PRODDTLKY in the **Class name** field. Click the **Browse...** button. The **Choose a valid class** window is displayed.
10. Select the **PRODDTLKY** class from the **Class Names** list. Select **VJ400LAB6** from the bottom **Package Names** list. Click **OK**. The **Choose a valid class** window is closed.
11. **VJ400LAB6.PRODDTLKY** is added to the **Class name** field of the Choose Bean window. Leave the **Name** field blank. Click **OK**. The **Choose Bean** window is closed.
12. Move the cross hairs inside the frame and click **Mouse Button 1** to drop the bean.

    NOTE: Make sure that the cross hairs are inside the frame before you click, otherwise the bean will not be visible when you run the application.
13. To add the "Enter PARTNO" label, click on the **Label** bean. Drop the Label bean below the PRODDTLKY subfile bean and single click **Mouse Button 1**. Click **Mouse Button 2** on the **Label2** bean and select **Properties**. The **Properties** window is displayed.

On the **Properties** window:
- Click the field to the right of the **text** and type Enter PARTNO:.
- Click "X" on the upper right corner to close the **Properties** window.
- Click on the **Label2** label to select it. Use the corner handles to increase the size of the label to display the complete text.

14. To create a status bar, click on the **Label** bean. Drop the Label bean just above the bottom of the application frame and click **Mouse Button 1**. Click **Mouse Button 2** on the **Label3** bean and select **Properties**. The **Properties** window is displayed. On the **Properties** window:
    - Click the field to the right of the **text** and delete the existing value to make it blank.
    - Click "X" on the upper right corner to close the **Properties** window.
    - Click on the **Label3** label to select it. Use the corner handles to increase the size of the label to display the complete text.

15. To add a text entry field, click on the **TextField** bean from the **AWT** Bean Palette. Drop it to the right of **Enter PARTNO:** label. Re-size it to hold five digits.

16. To add a **Read Record** button, click on the **Button** bean from the **AWT** Bean Palette. Drop it to the right of the **textField1**. Click **Mouse Button 2**. Select **Properties** from the pop-up menu. On the **Properties** window, click to the right of the **label** and enter Read Record. Click on the "X" on top right hand corner to close the **Properties** window.

17. To add a connection to open the file in read only mode do the following:
    - Select the application frame window. Click **Mouse Button 2** and select **Connect**->**windowOpened** from the pop-up menu.
    - Drag and drop the spider on the PRODDTLKY subfile bean. Select **Connectable Features...** from the pop-up menu.
    - Leave the **Method** radio button selected and select the **openKeyedFileReadOnly()** method from the list. Click **OK.**

    A connection is created.

18. To add a connection to read the record do the following:
    - Select the **Read Record** button. Click **Mouse Button 2** and select **Connect**->**actionPerformed** from the pop-up menu.
    - Drag and drop the spider on the PRODDTLKY subfile bean. Select **Connectable Features...** from the pop-up menu.
    - Leave the **Method** radio button selected and select the **readKeyedRecord(java.lang.String[])** method from the list. Click **OK.**

    An incomplete connection is created.

19. To add a connection to pass the parameter to the previous connection select the previous incomplete connection. Click **Mouse Button 2** and select **Parameter from code...** from the pop-up menu. An **Event-to-code connection** dialog is displayed. Replace the pre-filled code with the following code exactly as shown and click **OK** to close the dialog**.** Click **Yes** on the **Save changes** message box.

```
/**
 * Build a single element string array with the PARTNO string.
 */
public String[] connEtoM2_KeyValues(java.lang.String aPARTNO) {
    String[] keyValues = {""};
    keyValues[0] = aPARTNO;
    return keyValues;
}
```

An incomplete connection is created.

20. To add a connection to pass the parameter to the previous connection select the previous incomplete connection. Click **Mouse Button 2** and select **Connect->aPARTNO** from the pop-up menu. Drag and drop the spider on the **textField1** and select **text** from the pop-up menu.

Now all the incomplete connections should be complete.

21. To add a connection to display any exception messages if the read fails, do the following:
    - Click on the previous connection **(connEtoM2: Button1, actionPerformed -->PRODDTLKY1,readKeyedRecord(java.lang.String[]))** to select it. Click **Mouse Button 2.** Select **Connect -> exceptionOccured** from the pop-up menu.
    - Drag and drop the spider onto the status bar (**LabelStatus** bean) and click **Mouse Button 1**. Select **text** from the pop-up menu. A connection is shown with a dashed green line. This means the connection is incomplete and requires one or more parameters.
    - Click **Mouse Button 1** on the incomplete connection to select it. Click **Mouse Button 2.** Select **Properties** from the pop-up menu. The **Event-to-method connection - Properties** window is displayed.
    - Click on the **Pass event data** check box to check it.
    - Click **OK** to close the **Event-to-method connection - Properties** window. The connection turns solid green.
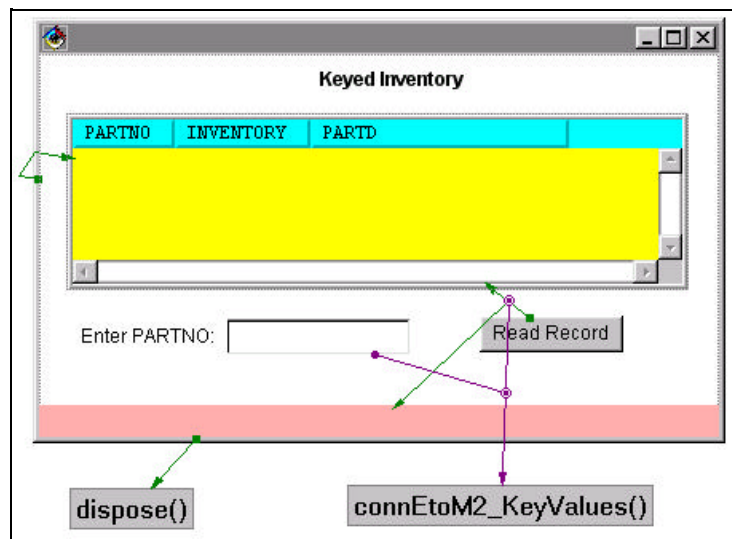
Test the GUI Application

Now you can test your application.

1. To setup the classpath, click on the **Run -> Check Class Path...** from the **Bean** menu of the **VCE**. The **Properties for InventoryList** window is displayed. Click on the **Project path:** check box to check it. Click on the **Edit...** button. Click on the **IBM Enterprise Toolkit for AS400** and the **JFC class libraries** check boxes to check them. Click **OK** and **OK** again to close the dialogs.

2. Click on the 🏃 from the tool bar of the **VCE**. Your application is saved. The **Keyed Inventory** application window is displayed. Re-size it to see all visual components.
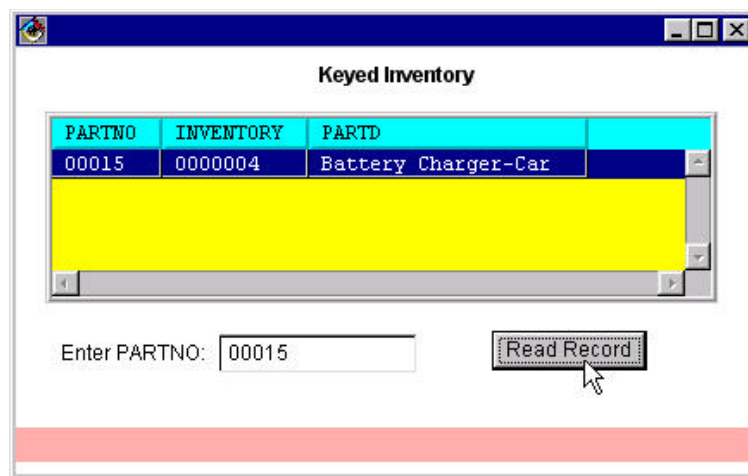
3. The **Signon to AS/400** dialog is displayed. Enter the Userid and Password given. Leave the **Default User ID** and **Save password** check boxes checked.
4. Click **OK** to close the **Signon to AS/400** dialog, make the AS/400 connection and open the file.
5. Enter 00015 (or the valid part number for the existing record) in the text field and click the **Read Record** button. The record should be displayed in the subfile.

   Now you know how to work with a keyed file. You can make connections to update, delete or add records.

<u>Congratulations!!! You completed the entire lab...</u>

Appendix 1. Adding a Feature into the Workbench

This section will show you how you can install a feature after you install **VA Java** for the first time.

- Select **File -> Quick Start** from the menu bar on the **Workbench**. The **Quick Start** dialog is displayed.
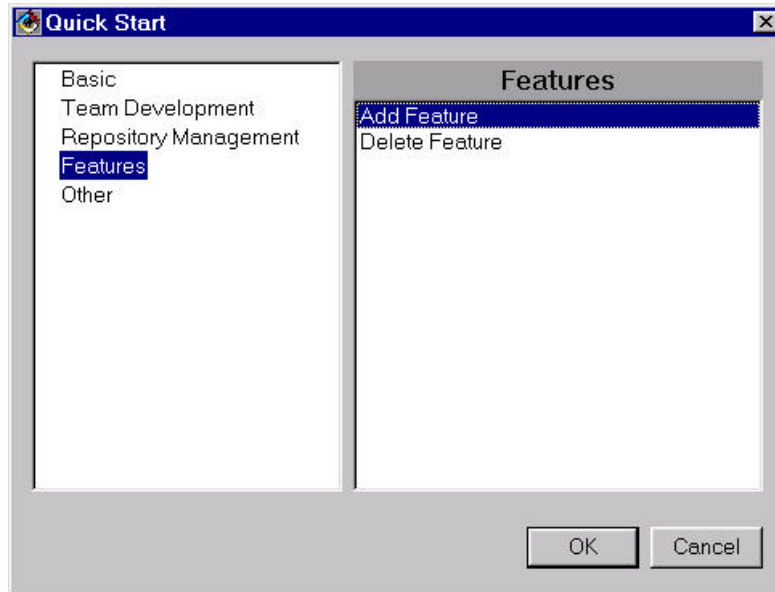
- Click on **Features** from the left hand list and then on **Add Feature** from the right hand list. Now click **OK**. The **Selection Required** dialog is displayed.
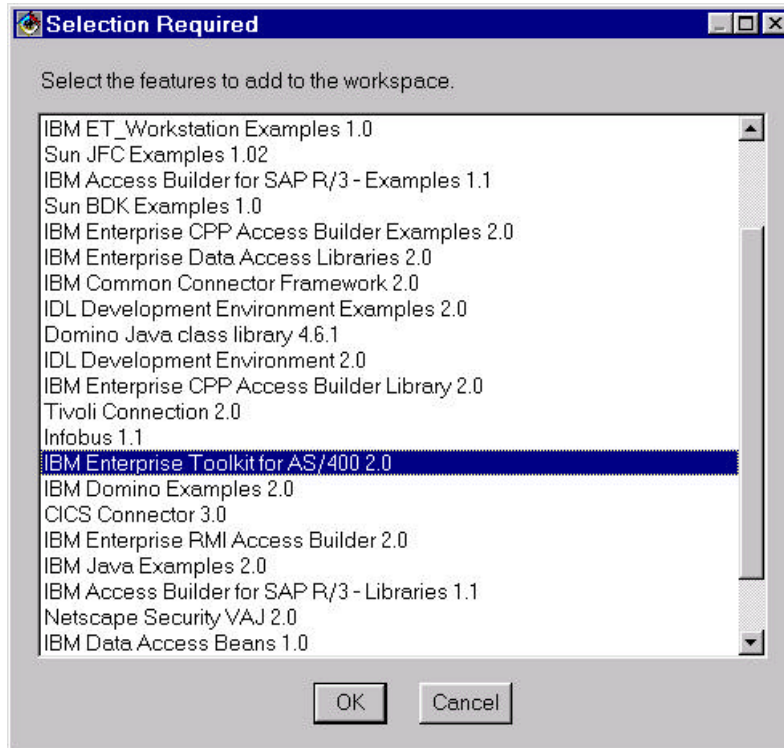
- Scroll down the list on the **Selection Required** dialog. You may not see the entry **IBM Enterprise Toolkit for AS/400 2.0,** since it is already added to the **Workbench**.
- Click **Cancel** to close the dialog. **<u>DON'T CLICK OK</u>** (it will overwrite any fixes that might have been applied to the feature and it is time consuming).