

VisualAge for Java, Version 2.0



# Domino AgentRunner



---

# Contents

|  |       |
|--|-------|
| <b>Legal Notices.</b>  | v     |
| Notices  | v     |
| <br><b>Chapter 1. The Domino AgentRunner</b>   | <br>1 |
| The Domino AgentRunner   | 1     |
| Using the AgentRunner  | 1     |
| Setting up for the AgentRunner   | 1     |
| Import or Write your Agent   | 2     |
| Generate an AgentContext Document  | 3     |
| Running and Debugging Your Agent.  | 4     |
| Creating the Production Agent  | 5     |
| <br><b>Chapter 2. Samples</b>  | <br>7 |
| Domino Samples Overview  | 7     |
| Sample: Intro Agent Using the Domino AgentRunner   | 9     |
| Sample: An Agent that Searches Your Database for Documents with a<br>Particular Subject. | 10    |
| Sample: Creating a To Do Agent   | 13    |
| Sample: Query a Domino or DB2 Database   | 15    |
| Sample: Using Data Access Beans to Access Notes.   | 16    |
| Sample: Using Swing 1.0.2 JTable   | 18    |



---

# Legal Notices

---

## Notices

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785, USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Canada Ltd., Department 071, 1150 Eglinton Avenue East, North York, Ontario M3C 1H7, Canada. Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

This publication may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

**Programming Interface Information** Programming interface information is intended to help you create application software using this program.

General-use programming interfaces allow the customer to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

**Warning:** Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

### **Trademarks and Service Marks**

The following terms are trademarks of the International Business Machines Corporation in the United States or other countries or both:

- AIX
- AS/400
- DB2

- CICS
- IBM
- OS/2
- OS/390
- RS/6000
- San Francisco
- VisualAge
- Visual Servlet
- WebSphere

Lotus, Lotus Notes and Domino are trademarks or registered trademarks of Lotus Development Corporation in the United States and/or in other countries.

Tivoli Management Environment, TME 10, and Tivoli Module Designer are trademarks of Tivoli Systems.

Encina and DCE Encina Lightweight Client are trademarks of Transarc Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks or registered trademarks of Microsoft Corporation in the U.S. and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks or registered trademarks of Intel Corporation in the U.S. and/or other countries.

Other company, service, and product names, which may be denoted by a double asterisk(\*\*), may be trademarks or service marks of their respective owners.

---

# Chapter 1. The Domino AgentRunner

---

## The Domino AgentRunner

The Domino AgentRunner is a tool used to help you build, run and debug Domino agents in VisualAge for Java. It uses a set of debug classes that access Notes context information so you can run and debug an agent in the IDE.

To use the AgentRunner, create an agent that extends `DebugAgentBase` in VisualAge for Java then export the class file to the file system. Create an agent in Notes and import the class file to run the agent. When you run the agent, it will generate an `AgentContext` document in the `AgentRunner.nsf`. Now you can run the agent inside the IDE with the AgentRunner. It accesses the `AgentContext` document to make the program behave as though it were running as a real agent in a Notes environment.

When you have finished debugging your agent using the integrated debugger, you can create the production agent simply by editing your class so it extends `AgentBase` and reimporting the class file to your agent in Notes.

See also: Sample: Using the AgentRunner.

---

## Using the AgentRunner

The AgentRunner makes it easy to run and debug your Domino agents in the VisualAge for Java IDE. To use the AgentRunner you must have Notes 4.6 installed on your system. Your task flow will be like this:

1. Set up for using the AgentRunner.
2. Create an agent that extends `DebugAgentBase`.
3. Generate an `AgentContext` document in Notes.
4. Debug your agent in VisualAge for Java.
5. Create the production agent.

See also: Sample: Using the AgentRunner.

---

## Setting up for the AgentRunner

Follow these steps to set up your Notes 4.6 environment to support the AgentRunner:

1. Add the `IVJAgentRunner.jar` file to the `JavaUserClasses` statement in your `notes.ini` file. If you do not have a `JavaUserClasses` statement in your `notes.ini` file already, you can cut and paste the following statement to the end of your `notes.ini` file:

```
JavaUserClasses=X:\VAJava\ide\runtime\IVJAgentRunner.jar
```

where X is the drive on which you installed VisualAge for Java.

Because you have edited the `notes.ini` file you must shut down and restart Notes so that your changes take effect.

2. Set your `PATH` to point to your Notes directory by entering

```
set path=%path%X:/path/Notes
```

on a command line. X is the drive on which Notes is installed. *path* is the path to your Notes directory. Your Notes directory will be temporarily added to your path. When you close your command window, this setting will be erased. Do not close the command window on which you set your path until you have finished using the AgentRunner.

If you want to permanently add this pointer, set the path to Notes in your computer's systems settings.

3. Copy the AgentRunner.nsf file from X:\VAJava\ide\runtime (where X is the drive on which you installed VisualAge for Java) to your notes\data directory.

Your Notes environment is now set to support the AgentRunner. Next, you have to set up your VisualAge for Java Environment.

### Add the Domino Java Classes to the Workbench

To set up your VisualAge for Java IDE to use the AgentRunner you have to add the Domino Java class library to the Workspace.

1. From the Workbench, select the **File** menu, then **Quick Start**.
2. Select **Features** from the list of categories on the left side of the window.
3. Select **Add Feature** from the list of tasks on the right side of the window.
4. Click **OK**.
5. In the Selection Required window, select **Domino Java class library**.
6. Click **OK**.

A new project will appear on the All Projects view of the Workbench called Domino Java class library. This project contains a lotus.notes package with all the Java Classes for Notes Object Interface/Domino 4.6 and additional Debug classes that support the AgentRunner tool. You can now use these classes when running or debugging an agent in the IDE.

Now you are ready to write or import an agent.

---

## Import or Write your Agent

Create a project in the Workbench into which you will import your agent.

### To create a project:

1. Go to the Projects page of the Workbench.
2. On an empty space in the All Projects view, click mouse button 2.
3. Select **Add - Project**. The Add Project SmartGuide will open.
4. Select the **Create a new project named** radio button.
5. Enter a name such as *Domino Agents* in the text field.
6. Click **Finish**.

When the SmartGuide closes you will have a new project in the workbench called Domino Agents.



You are now ready to import an agent from Notes, or create a new agent in the IDE.

**To import an agent from Notes into your Domino Agents project:**

1. Select the **Domino Agents** project with mouse button 2.
2. Select **Import**. A SmartGuide will open.
3. Select **Directory** as your import source.
4. Click **Next**.
5. Fill in the **Directory** field with the path that contains your agent's java file.
6. Select the **.java** checkbox.
7. Click the **Details** button beside **.java** and select the java file that you want to import. Click **OK**.
8. Click **Finish**

The imported Java code is compiled and any unresolved problems that are introduced are added to the All Problems page.

Your .java file will appear as in a package under your Domino Agents project in the Workbench.

**To create a new agent in VisualAge for Java:**

1. Click on your **Domino Agents** project with mouse button 2.
2. Select **Add**, then **Class**. The Create Class SmartGuide will open.
3. Enter a package name in the **Package** text field.
4. Select the **Create a new class** radio button. The Create Class SmartGuide opens.
5. Enter a class name for you agent, and select **DebugAgentBase** as your Superclass.
6. Click **Finish**.

Your new package and class will appear under your Domino Agents project in the Workbench. Click on your class with mouse button 1. In the Source window of the workbench, write the code for your agent (see the *Java Programmer's Guide* in Notes for instructions).

When you have finished writing your agent, you should generate an AgentContext document so you can run your agent in the IDE.

---

## Generate an AgentContext Document

Before you can generate the AgentContext document you have to export your agent's .class file from the IDE to the file system so it can be read by Notes.

**Export your class file to the file system**

To export your agent's .class file, from the Workbench:

1. Select your agent with mouse button 2.
2. Select **Export**.
3. Select the **Directory** radio button. The Export to a directory SmartGuide opens.
4. Enter the path where you want to export the file in the text field.

5. Select the **.class** check box.
6. Click the **Details** button beside the **.class** check box.
7. Select the .class file that you want to export.
8. Click **OK**.
9. Click **Finish**.

Your agent's .class file is now in the file system. Next, you should generate your AgentContext document in a Notes database.

### Generating an AgentContext document

To create a Java agent in a Notes database:

1. Open Lotus Notes 4.6 (or higher).
2. Create your agent in the appropriate database.
3. Fill in the particulars for your agent. Select the **Java** radio button for What should this agent run
4. Click **Import class Files** and select file you exported from VisualAge for Java.
5. Run the agent.

The AgentContext document is automatically generated in the AgentRunner.nsf when you run your agent in Notes from an agent class that extends DebugAgentBase. A call to getSession() will, after generating an AgentContext document, return null. Any use of the returned session will result in a thrown exception, but since the purpose of running the agent is only to generate the context document (and not to run any of the agent code), you can ignore the exception.

When you have generated an AgentContext document, you are ready to run and debug your agent in the VisualAge for Java IDE.

---

## Running and Debugging Your Agent

Follow these steps to run and debug your agent in the IDE:

1. Set one or more breakpoints in the NotesMain() method of your agent.
2. Select your agent with mouse button 2.
3. Select **Tools - Domino AgentRunner**

You have two options.

1. Select **Properties** if you want to modify your AgentContext or select a different AgentContext. The AgentRunner window will open. Select the AgentContext that you wish to use and then click **Run Agent**. To modify your agent, click **Update Agent Context**. You can change the Agent Runs on and Search Criteria fields to generate the UnprocessedDocuments collection that you would like to use for debugging purposes. You must supply this information because it cannot be determined from running the agent. When you have finished updating your AgentContext, click the **Update AgentContext Document** button and close the window. Click **Run Agent** to run the agent or click **Save Selection** first to make your new AgentContext the default choice.

2. Select **Run** if you wish to run your agent with the default AgentContext document. This is either the last AgentContext that you ran or the last one you

saved. If you have set any breakpoints or have errors in your code, the Debugger window will open and allow you to step through your code.

When your code is error-free, you can create the production agent.

---

## Creating the Production Agent

When your development of the agent is complete and you want to move your agent to Notes, follow these steps in VisualAge for Java:

1. Change the Base class for your agent to extend `lotus.notes.AgentBase` instead of `DebugAgentBase`.
2. Export the `.class` file to the file system.

Open the agent in Notes and reimport the `.class` file from the file system. You can now run your agent in Notes.



---

## Chapter 2. Samples

---

### Domino Samples Overview

These samples demonstrate how to use agents, applications and applets to connect to your Domino or DB2 database.

#### Before You Begin

To run all of the Domino samples you must follow these steps to set up your environment:

1. Add the projects from the repository to your Workbench. The easiest way to add a package to your Workbench is by using Quick Start:
  1. From the Workbench, select the **File - Quick Start**.
  2. Select **Feature** from list of categories on the left side of the Quick Start window.
  3. Select **Add Feature** from the list of options on the right side of the Quick Start window.
  4. Click **OK**. The Selection Required window will open.
  5. To select multiple features hold down the the Ctrl key and select the following features with mouse button 1 and select:
    - Domino Java class library**
    - IBM Data Access Beans**
    - IBM Domino Examples**
    - Java class libraries**
    - JFC class libraries**
  6. Click **OK**. An information window with a progress bar will open to show you what you are adding to your Workbench. When it is finished, the features you selected will appear as projects in your Workbench.
  7. Open the IBM Domino Examples project to see the Domino samples available.
2. You must have the DeptDesc.nsf set up in your Domino environment. Using the command line, copy this file from the X:\IBMJava\ide\samples directory (where X is the drive onto which you installed VisualAge for Java) into your Y:\notes\data directory (where Y is the drive onto which you installed Notes).
3. Ensure that JavaDlgSettings is in your notes.ini file and that JavaUserClasses includes a path to your SSQLIB directory. If you are missing these settings you can cut the following lines and paste them at the end of your notes.ini file. Make sure that you change the JavaUserClasses path to that of your directory.

```
JavaDlgSettings=*.class|*.jar,*.cab,*.zip|*.jpg,*.gif,*.au|*.java|1
JavaUserClasses=d:\sqllib\java\db2java.zip
```
4. After you have changed your notes.ini file, you must close and reopen Notes for the changes to take effect.
5. Make sure that that Notes is in your PATH statement. To look at your classpath, enter set classpath on a command line. If the path to your Notes directory is not in your classpath statement, enter set classpath = %classpath%path/Notes. Do not close the command window while running this sample.

If you want to add the pointer to notes permanently, set your PATH statement in your computer's system settings.

6. If you are using the AgentRunner you must also follow any other set up steps in Setting Up For the Agent Runner.

### Preparation for DB2

For the samples that access DB2, you must add the DB2 class library to the IDE and to your classpath.

To create a project for the DB2 class library:

1. Create a project for the DB2 class library by selecting a project or package in the Workbench with mouse button 2.
2. Select **Add - Project**. The Add Project SmartGuide will open.
3. Select the **Create a new project named:** radio button and enter a name, such as DB2 class library in the text field.
4. Click **Finish**. The project will be added to the Workbench.

To import the classes to the repository:

1. Select **File - Import**. The Import SmartGuide will open.
2. Select the **Jar file** radio button.
3. Click **Next**.
4. Type the path to your db2java.zip file or use the Browse option to find the file.
5. Select the **class** radio button, then click the **Details** button beside it.
6. Select all of the classes and click **OK**.
7. Click **Finish**.

To add the classes to the Workbench:

1. Select the DB2 class library project with mouse button 2.
2. Select **Add - Package**. The Add Package SmartGuide will open.
3. Select the **Add packages from the repository** radio button.
4. Select the db2java package(s).
5. Click **Finish**.

When you use DB2 you must make sure that the DB2 class library is in the classpath for that sample.

### The Samples

Click on a sample name to link to the run and build instructions.

Sample: Using the Domino AgentRunner. This sample demonstrates how to use the AgentRunner with a simple agent.

Sample: Creating a To Do Agent. This sample demonstrates how to create an agent that uses Java AWT and NOI.

Sample: Creating an Agent that Searches Your Database. This sample is an agent which runs against a database retrieving all documents which have the same subject as the document you invoke the agent against.

Sample: Query your Domino or DB2 Database. This sample shows how an agent can access multiple databases.

Sample: Using Data Access Beans to Access Notes. This sample demonstrates how to use Data Access Beans to access a Domino database.

Sample: Using Swing JTable. This sample shows the use of the database support in Swing 1.0.2 using the JTable class and associated classes.

---

## Sample: Intro Agent Using the Domino AgentRunner

### Objective

The objective of this sample is to demonstrate how to use the AgentRunner.

### Time Required

You should allow 20 minutes to build and run this sample.

### Before You Begin

To build and run this sample you need to have Notes 4.6 or higher installed.

Set up your Workbench for using the samples by following the steps in the Domino Samples Overview.

Set up your Lotus Notes and VisualAge for Java environments to support the Domino AgentRunner by following the steps in the task file Setting up for the AgentRunner and return to this sample.

### Description

This sample is a simple agent that will tell you the title of the database on which the agent is running.

### Running the Sample

Once you have set up your Lotus Notes and VisualAge for Java environments you have to:

1. Export your class file to the file system.
2. Create and run the agent in notes to create an AgentContext document.
3. Return to the AgentRunner to run and debug the agent.

#### 1. Export the .class File

To export the .class file from the com.ibm.ivj.examples.domino.intro package in VisualAge for Java to the file system so it can be read by Notes:

1. Open the **IBM Domino Examples** project in the Workbench and expand the **com.ibm.ivj.examples.domino.intro** package.
2. Click the **Intro** class with mouse button 2.
3. Select **Export**. The Export SmartGuide will open.
4. Select the **Directory** radio button as your export destination. Click **Next**.
5. In the **Directory** text field, enter a location in the file system where you want to export the .class file.
6. Select the **.class** checkbox and click **Details**.
7. Select the **IBM Domino Examples** and the **Intro::com.ibm.ivj.examples.domino.intro** checkboxes. Click **OK**.
8. Deselect the **.java** and **resource** checkboxes if they are selected.
9. Click **Finish**.

When you export files to the file system, the directory path of the file is maintained. For example, if you entered X:\VAJexport as your export destination, the Intro.class file will be in X:\VAJexport\com\ibm\ivj\examples\domino\intro.

## 2. Create the Agent in Notes

For the AgentRunner to work in the IDE you have to provide it with a context for running your agent. To create an AgentContext document:

1. Open any database
2. From the **Create** menu, select **Agent**.
3. Enter a name for your agent, such as *MyIntro*.
4. Select **Manually From Actions Menu** for **When should this agent run**.
5. Select **All documents in database** for **Which document(s) should it act on**.
6. Select the **Java** radio button for **What should this agent run**.
7. Click the **Import Class Files** button.
8. Locate and select the Intro.class file that you exported from VisualAge for Java and click **Add**.
9. Click **OK**.
10. Press the Esc key to close the Create Agent window. You will be asked if you want to save your changes. Click **Yes**.

## Running the Agent in Notes

From the **Actions** menu of the Domino database, select **MyIntro**. The agent will run but the session will return Null. An AgentContext document will be created in the AgentRunner.nsf

## 3. Using the AgentRunner

Now you are ready to run the agent in the AgentRunner.

To run your agent from the Workbench:

1. Open the **Console** window by selecting **Window - Console**. This window will show the status of your agent.
2. Select the **Intro** class with mouse button 2.
3. Select **Tools - Domino AgentRunner - Run**.

The Domino AgentRunner will run your agent using the AgentContext document that you created in Notes. The Console will give you a message that the Intro Agent is running. When it has run against your Notes database, it will return the title of the database in which you created the agent in Notes.

---

## Sample: An Agent that Searches Your Database for Documents with a Particular Subject

### Objective

The objective of this sample is to create an agent that searches the documents in a Domino database.

### Time Required

Allow half an hour to create and run this agent.

### Before You Begin

To build and run this sample you need to have Notes 4.6 or higher installed.



To prepare for the sample, follow the steps in the Domino Samples Overview.

Set up your Lotus Notes and VisualAge for Java environments to support the Domino AgentRunner by following the steps in Setting up for the AgentRunner and return to this sample.

### Description

This sample agent runs against a database retrieving all documents which have the same subject as the document you invoke the agent against. The agent will send you a mail message containing a list of links to all documents related to a common subject. For example, if you select a document with the subject Hello World and then invoke the agent, a list of all the documents that contain the string 'Hello World' in the subject field will be returned in a mail message, with a link to each document.

### Export the .class File

To export the .class file from the com.ibm.ivj.examples.simpleagent package in VisualAge for Java to the file system so it can be read by Notes:

1. Open the **IBM Domino Examples** project in the Workbench and expand the **com.ibm.ivj.examples.domino.simpleagent** package.
2. Click the **SimpleAgent** class with mouse button 2.
3. Select **Export**. The Export SmartGuide will open.
4. Select the **Directory** radio button as your export destination. Click **Next**.
5. In the **Directory** text field, enter a location in the file system where you want to export the .class file.
6. Select the **.class** checkbox and click **Details**.
7. Select the **IBM Domino Examples** and the **SimpleAgent::com.ibm.ivj.examples.domino.simpleagent** checkboxes. Click **OK**.
8. Deselect the **.java** and **resource** checkboxes if they are selected.
9. Click **Finish**.

When you export files to the file system, the directory path of the file is maintained.

For example, if you entered X:\VAJexport as your export destination, the SimpleAgent.class file will be in

X:\VAJexport\com\ibm\ivj\examples\domino\simpleagent.

### Create the Agent in Notes

For the AgentRunner to work in the IDE you have to provide it with a context for running your agent. In this samples you run the agent in your Notes mail database but you could create the agent in any database. To create an AgentContext document:

1. Open your mail database.
2. Select **Agent** from the **Create** menu.
3. Enter a name for the agent, such as *CheckMail*
4. Select **Manually from Actions Menu** from the drop-down list **When should this agent run**.
5. Select **All documents in database** from the **Which documents should it act on** pull-down menu.
6. Select the **java** radio button for **what should this agent run**.
7. Click the **import class files** button.
8. Locate the SimpleAgent class file, click the **Add** button.

9. Click **OK**.
10. Press the **Esc** key to close the Create Agent window. You will be asked if you want to save your changes.
11. Click **Yes**.

### Running the Agent in Notes

Open the Java Console in Notes by selecting **File - Tools - Show Java Debug Console** so you can monitor your agent. Select a document that has as its subject the text string you want your agent to search for. From the **Actions** menu of the Domino database, select **SimpleAgent**. The agent will run but the session will return Null. An AgentContext document will be created in the AgentRunner.nsf

### Using the AgentRunner

Now you are ready to run the agent in the AgentRunner.

To run your agent from the Workbench:

1. Open the **Console** window by selecting **Window - Console**. This window will show the status of your agent.
2. Select the **SimpleAgent** class with mouse button 2.
3. Select **Tools - Domino AgentRunner - Run**.

The Domino AgentRunner will run your agent using the AgentContext document that you created in Notes.

### Update your AgentContext (optional)

If you want to change which documents you want the agent to run against:

1. Select the **SimpleAgent** class with mouse button 2.
2. Select **Tools - Domino AgentRunner - Properties**. The Properties window will open.
3. Click the **Update Agent Context** button.
4. Select which documents you want the agent to run against like you would in Notes.
5. Add a description so you can identify your context.
6. Click **Update Agent Context Document**.
7. Close the window.
8. Select your new agent context.
9. Click the **Run Agent** button from the AgentRunner window.

The Domino AgentRunner will run your agent with your updated agent context.

### Create the Production Agent

When your agent runs without errors, you can create the production agent by taking it out of 'Debug' mode.

Open your SimpleAgent class file in the Workbench. In the Source window, change what the class extends from *lotus.notes.DebugAgentBase* to *lotus.notes.AgentBase*. Click your **SimpleAgent** class in the All Projects window. You will be asked if you want to save your changes. Click **Yes**.

### Export the .class File

Your agent is now in it's completed state and can be run in Notes. Export the

.class file to the file system using the Export SmartGuide. Open your agent in Notes and reimport the SimpleAgent.class file.

### Run the Agent in Notes

To run the agent, select a mail document with a common subject. Select your agent from the **Actions** menu. The agent will run and send you a mail message that contains all the documents in the database that contain the subject of the document you ran the agent against. Open the mail message and select the links to the related documents.

---

## Sample: Creating a To Do Agent

### Objective

To create a Notes agent that uses Java AWT and NOI.

### Time Required

Allow half an hour to run this sample with its supplied content and look over the output.

### Before You Begin

To run this sample you need the following products:

- VisualAge for Java
- Notes 4.6 or higher running as a client
- Notes 4.6 or higher running as a server - this is not needed if you initiate your agent manually, for example from the Actions menu.

Follow the steps in the Domino Samples Overview to prepare for the sample.

Set up your Lotus Notes and VisualAge for Java environments to support the Domino AgentRunner by following the steps in Setting up for the AgentRunner and return to this sample.

### Description

This sample automatically checks your To do list once per day for any items that are due. A mail message will then be sent to the owners of the due items reminding them that a task is due that day.

### Export the .class File

To export the .class file from the com.ibm.ivj.examples.domino.todoagent package in VisualAge for Java to the file system so it can be read by Notes:

1. Open the **IBM Domino Examples** project in the Workbench and expand the **com.ibm.ivj.examples.domino.todoagent** package.
2. Click the **ToDoAgent** class with mouse button 2.
3. Select **Export**. The Export SmartGuide will open.
4. Select the **Directory** radio button as your export destination. Click **Next**.
5. In the **Directory** text field, enter a location in the file system where you want to export the .class file.
6. Select the **.class** checkbox and click **Details**.
7. Select the **IBM Domino Examples** and the **ToDoAgent::com.ibm.ivj.examples.domino.todoagent** checkboxes. Click **OK**.
8. Deselect the **.java** and **resource** checkboxes if they are selected.

9. Click **Finish**.

When you export files to the file system, the directory path of the file is maintained.

For example, if you entered X:\VAJexport as your export destination, the ToDoAgent.class file will be in

X:\VAJexport\com\ibm\ivj\examples\domino\todoagent.

#### **Follow These Steps in Notes 4.6 to Set Up the Agent.**

1. Open your mail database.
2. Select **Agent** from the **Create** menu.
3. Enter a name for the agent, such as CheckTodo
4. Select **On Schedule Daily** from the drop-down list **When should this agent run**.
5. Click the **Schedule** button.
6. From the Schedule window, enter a time for the agent to run, select the **Don't run on weekends** check box, and have the agent run on your Local database.
7. Click **OK**.
8. From the **Which documents should it act on** pull-down menu, select **All documents in database**.
9. Click the **Add Search** button and select **By Form Used** as the condition
10. Select **Task** as the form type and click **OK**.
11. Select the **java** radio button for **what should this agent run**.
12. Click the **import class files** button.
13. Select the **ToDoAgent.class** file.
14. Click **Add**.
15. Click **OK**.
16. Press the Esc key to close the Create Agent window. You will be asked if you want to save your changes.
17. Click **Yes**.

#### **Running the Agent in Notes**

Open the Java Console in Notes by selecting **File - Tools - Show Java Debug Console** so you can monitor your agent. From the **Actions** menu of the Domino database, select **ToDoAgent**. The agent will run but the session will return Null. An AgentContext document will be created in the AgentRunner.nsf

#### **Using the AgentRunner**

Now you are ready to run the agent in the AgentRunner.

To run your agent from the Workbench:

1. Open the **Console** window by selecting **Window - Console**. This window will show the status of your agent.
2. Select the **ToDoAgent** class with mouse button 2.
3. Select **Tools - Domino AgentRunner - Run**.

The Domino AgentRunner will run your agent using the AgentContext document that you created in Notes.

#### **Create the Production Agent**

When your agent runs without errors, you can create the production agent by taking it out of 'Debug' mode.

Open your `ToDoAgent` class file in the Workbench. In the Source window, change what the class extends from `lotus.notes.DebugAgentBase` to `lotus.notes.AgentBase`. Click your **ToDoAgent** class in the All Projects window. You will be asked if you want to save your changes. Click **Yes**.

### Export the .class File

Your agent is now in its completed state and can be run in Notes. Export the .class file to the file system using the Export SmartGuide. Open your agent in Notes and reimport the `ToDoAgent.class` file.

### Running the Agent

You now have an agent that will check your To Do list for any tasks that are due today.

To test the agent, create a task and set the due date as today's date. Create another task with some other day as the due date. You can even assign tasks to other people. When the agent runs manually or at the time you specified, the owner of the task will receive a mail reminder that the task is due that day.

---

## Sample: Query a Domino or DB2 Database

### Objectives

The objectives of this sample are:

- To use the Data Access Builder to connect to DB2 through JDBC.
- To use the Data Access Builder to connect to NotesSQL through NOI.

### Time Required

Allow 30 minutes to create and run this sample.

### Before You Begin

To build and run this sample you need:

- VisualAge for Java
- DB2
- Notes 4.6 or higher running as both client and server

Follow the steps in the Domino Samples Overview to prepare for the sample. If you want to use the AgentRunner you also have to follow the steps in Setting Up For the AgentRunner.

### Preparation for DB2

To set your DB2 userid and password in the source code:

1. Expand the IBM Domino Examples project.
2. Expand the `com.ibm.ivj.examples.domino.mailagent` package.
3. Expand the MailAgent class
4. Select the `processDB2(String)` method.
5. Find the line `connectObject = DriverManager.getConnection(url, 'USERID', 'PASSWORD');`
6. Change the parameters 'USERID' and 'PASSWORD' to the userid and password with which you connect to DB2.

To set your classpath:

1. Select the MailAgent class with mouse button 2.

2. Select **Properties**. The Properties for UppercaseDeptname window will open.
3. Select the Class Path tab.
4. Select the **Project path** checkbox and click **Edit**.
5. Select the **DB2 class library** project. Click **OK**.
6. Click **OK** to close the Properties window.

### Description

This sample shows a Lotus Notes agent that runs automatically when new mail arrives. It checks the incoming mail to see if it is a special mail request that matches a certain format. If so, the mail is read and a database query is run. The results of the query are send back to the original sender of the mail message.

To setup Notes to run the agent, do the following:

1. Open your mail database.
2. From the **Create** menu, select **Agent**.
3. Enter a name for the agent, such as CheckMail
4. From the **When should this agent run** dropdown list, select **if new mail has arrived**.
5. Select the **java** radio button for **what should this agent run**.
6. Click the **import class files** button.
7. Locate the AgentQuery.class file, click **Add**.
8. Click **OK**.
9. Press the Esc key to close the Create Agent window. You will be asked if you want to save the agent.
10. Click **Yes**.

### Running the Agent

To query your Notes database, send yourself a mail message with DBQUERY as the subject and

DATABASE: NOTES DeptDesc.nsf

QUERY: SELECT Deptno,Deptname WHERE Deptno='E11'

as the body of the message.

To query your DB2 database, send yourself a mail message with DBQUERY as the subject and

DATABASE: DB2 JDBC:DB2:SAMPLE

QUERY: Select \* from Employee

as the body of the message.

The results of the query will be sent to you as a mail message.

---

## Sample: Using Data Access Beans to Access Notes

### Objective

This sample shows how to use Data Access Beans to access a Domino database.

### Time Required

To run the sample with its supplied content and look over the output, you will probably want to allow about 20 or 30 minutes.

### Before You Begin

You need to have the following tools:

- VisualAge for Java
- Notes 4.6 or higher
- ODBC
- Notes SQL - this can be downloaded from the Lotus website.

Follow the steps the Domino Samples Overview to set up your Lotus Notes and VisualAge for Java environments.

### Description

This sample retrieves information from a Domino database using the NotesSQL ODBC driver that is accessed with Java using the Data Access Beans, and displayed using Swing beans. This sample is in the IBM Domino Examples project and is called `com.ibm.ivj.examples.domino.dasample`.

### Check your Path

To ensure that your path includes all of the necessary pointers:

1. Expand your `com.ibm.ivj.examples.domino.dasample` package.
2. In the Workbench, select the **DAApplet** class with mouse button 2.
3. Select **Properties**. The Properties for DAApplet notebook will open.
4. Select the **Extra directories path** checkbox. Click **Edit**.
5. Click **Add Jar/Zip**.
6. Locate and add the path to `sqlib\java\db2java.zip`.
7. Click **OK** to close the VisualAge window
8. Click **OK** to close the notebook.

### Run as an Applet

1. Select the **DAApplet** class in the Workbench with mouse button 2.
2. Select **Run - In Applet Viewer**. The Applet viewer will open.
3. Use the toolbar at the bottom of the applet to scroll through the members of the department. On the toolbar, the button with a picture of a person on it is used to get the data from the database. The arrow buttons will help you navigate through the database and the final button is used to refresh the information.

### Run as an Application

1. Select the **DAApplet** class in the Workbench with mouse button 2.
2. Select **Run - Run main**. The application will open.
3. Use the toolbar at the bottom of the applet to scroll through the members of the department. On the toolbar, the button with a picture of a person on it is used to get the data from the database. The arrow buttons will help you navigate through the database and the final button is used to refresh the information.

---

## Sample: Using Swing 1.0.2 JTable

### Objective

This sample shows the use of the database support in Swing 1.0.2 using the JTable class and associated classes.

### Time Required

Allow half an hour to run this sample.

### Before You Begin

To run this sample you need to have the following tools installed and configured for your machine.

- VisualAge for Java
- DB2 - This sample uses data from the SAMPLE database in DB2.

### Description

This sample retrieves information from a DB2 database and displays it in a window created with JTable.

### Running the sample

Open the IBM Domino Examples project and expand the com.ibm.ivj.examples.domino.jtable package. To run this example on your machine you need to make a few small changes. In the Workbench, select the EmplTable class. The source code of this class will appear in the source window. Change the following statements to your own machine variables:

1. `static String uid = 'userid';` where 'userid' is the userid that you use to connect to DB2
2. `static String pwd = 'PASSWORD';` where 'PASSWORD' is the password that you use to connect to DB2

### Select your Parameters

Because this sample runs as both an application and an applet you can add the properties for both.

From the Properties for JTableExample notebook:

1. Select the Applet tab to set the properties for the applet
2. In the **Width** text field enter *450*.
3. In the **Height** text field enter *200*.
4. In the **Parameters** text field enter  
`<param name=FirstName value=DANIEL>`  
`<param name=LastName value=SMITH>`
5. Select the Program tab to set the properties for the application.
6. In the **Command line** arguments text field enter *DANIEL SMITH*
7. Click **OK**.

Valid names for the parameters are any names in the DB2 SAMPLES database.

### Run as an Applet

1. Select the **JTableExample** class in the Workbench with mouse button 2.
2. Select **Run - In Applet Viewer**. The Applet viewer will open with the information for Daniel Smith.



### **Run as an Application**

1. Select the **JTableExample** class in the Workbench with mouse button 2.
2. Select **Run - Run main**. The application will open with the information for Daniel Smith.