Getting Started with JBuilder and Cloudscape

JBuilder is an integrated development environment, or IDE, for building Java applications. It comes with integrated database tools. This paper describes how to get started using JBuilder 2.0 to build Cloudscape applications.

JVM Overview

As you know, a Java application runs in a Java Virtual Machine, or JVM.

JBuilder has two distinct JVM environments:

- One that JBuilder itself uses while you design and build your application. (For the purposes of this white paper, let’s call this JBuilder’s embedded JVM environment.)
- One that it launches to run your application. (For the purposes of this white papre, let’s call this JBuilder’s runtime JVM environment.)

You don’t usually run your application while you are designing it and building. However, JBuilder allows you to test database connections and database queries while you are designing your application (“in design mode”). If you test your connections or queries in design mode, JBuilder uses the JBuilder embedded JVM environment to launch the JDBC driver you specify and to run the queries you specify. In addition, you must add the appropriate Cloudscape libraries to this JVM environment before testing the connections or queres and then restart JBuilder.

JBuilder also allows you to run applications that you are building. When you actually run your database application, JBuilder launches a new instance of a JVM for your application to run in. The application starts up a instance of the driver you specify when you run it. You must add the appropriate Cloudscape libraries to the project's properties in order to run the application.

In the most basic Cloudscape environment, in which the Cloudscape engine is embedded in the application itself, starting the embedded driver starts up the Cloudscape engine.

Initiated Cloudscape users will realize something right away; for an application in which the Cloudscape engine is embedded, you would not want to test connections and queries when you are in design mode *and* run the application itself within the same JBuilder session. Initiated Cloudscape users know that you cannot have two instances of Cloudscape access the same database at the same time; and if you have two separate JVMs each firing up Cloudscape, then you in essence have two instances of Cloudscape.

You can use one of the following strategies to avoid ever running into this problem:

- Avoid testing connections and queries in design mode. Instead, test your application by running it.
- Run Cloudscape inside the low-footprint server framework RmiJdbc while you are developing your application. Use the RmiJdbc driver and RmiJdbc flavor of the database connection URL while developing.

  RmiJdbc is included with Cloudscape at no extra cost. When you run Cloudscape inside the framework, multiple applications can connect to a database at the same time. This allows the multiple JVMs launched by JBuilder to access the same database.

  Later on, if you wish to deploy Cloudscape embedded in the application, simply change the driver name and database connection URL back to the embedded flavors before deploying.

This paper walks takes you through the steps for building an extremely simple UI for toursDB (Cloudscape's sample database) using both strategies.

- "Strategy One: No "Live" Connections in Design Mode"
- "Strategy Two: Live Connection to Cloudscape Running Inside RmiJdbc Server"

Strategy One: No “Live” Connections in Design Mode

Create a New Project

1. Start JBuilder and create a new project by choosing File->New Project. Call the project *tours.jpr*. It should be in a project folder called *toursGUI*.

Set up the Project Environment for the Runtime JVM

You will need to set up your project correctly so that you can run your application from JBuilder. Setting up the project involves specifying the class path to include the Cloudscape libraries and to set any needed Java VM parameters.

1. Choose Project Properties from the File menu.
2. Click Libraries.
3. Click New.
4. Name the library CloudscapeAndTours. Set its class path to include the primary Cloudscape library and the JBMSTours classes. For example:

   ```
   c:\cloudscape\lib\cloudscape.jar;c:\cloudscape\demo\programs\to
   urs
   ```

5. Click OK.
6. Click Add. Select CloudscapeAndTours..
7. Select the Run/Debug tab.
8. Set the *cloudscape.system.home* property to the location of the built-in *toursDB* demo database as a Java VM parameter, like so:

   ```
   -Dcloudscape.system.home=c:\cloudscape\demo\databases
   ```

9. Select Send Run Output to Execution Log.
10. Click OK.
11. Save your changes (choose Save from the File menu).

Begin the GUI Application

Now that the environment is set up correctly, create a new application.

1. Create a new application by selecting tours.jpr in the Navigation pane of the ApplicationBrowser, then choosing File-> New.

2 Select application and click OK.
3 Call the class *HotelDisplayer*. It should be in the *toursGUI* package
4 Click Finish.

Test the Environment

To test the new environment, we will use one of the JBMSTours applications, called *GenerateReport*. The application does not have a GUI, but it accesses toursDB, so if it runs successfully, the environment is set up correctly.

1 Click the folder icon with the plus sign on it in the AppBrowser Navigation pane to add a new file to the project.
2 Navigate to the location of the JBMSTours class files (*%cloudscape_install%/demo/programs/tours/JBMSTours*).
3 Select *GenerateReport.java*, and click Open.
4 Choose Run GenerateReport from the Run menu.
5 Wait a few seconds, then choose Execution Log from the View menu to examine the output. A successful run looks something like this (an excerpt):

```
D:\jbuilder2\java\bin\javaw.exe -
Dcloudscape.system.home=c:\cloudscape\demo\databases -
classpath
"D:\jbuilder2\myclasses;D:\jbuilder2\lib\swingall.jar;D:\jbui
lder2\lib\jbcl2.0.jar;D:\jbuilder2\lib\jbcl2.0-
res.jar;D:\jbuilder2\lib\jgl3.1.0.jar;c:\cloudscape\lib\cloud
scape.jar;c:\cloudscape\lib\tools.jar;c:\cloudscape\demo\prog
rams\tours;D:\jbuilder2\dependency\~CloudscapeAndTours;D:\jbu
ilder2\java\lib\classes.zip" JBMSTours.GenerateReport
AppAccelerator(tm) 1.1.034 for Java (JDK 1.1), x86 version.
Copyright (c) 1998 Borland International. All Rights
Reserved.
GenerateReport starting
Loaded the embedded JDBC driver
Connected to database toursDB
Hottest Average Temperature in each Region
Region: Africa 89.0
Region: Asia 97.0
Region: Australia and New Zealand 71.0
Region: Central Asia 90.0
Region: Europe 68.0
. . .
```

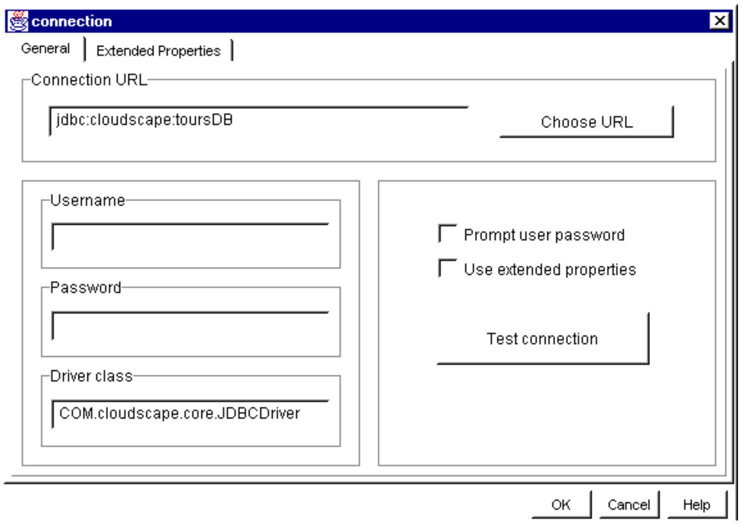
NOTE: If you have any errors, double-check that you have specified the class path for CloudscapeAndTours correctly in the project properties. You may see errors regarding DatabasePreloader. Ignore those errors.

Build the GUI

1. Select HotelDisplayer in the AppBrowser. JBuilder automatically included a Frame in that class called Frame1 when you created *HotelDisplayer*. Click the icon for *Frame1*, and select the Design tab.
2. Select the AWT tab in the Component Palette.
3. Click the *java.awt.scrollPanel* icon (third from the right), then click anywhere in the window.
4. Select the Data Express tab (at the top).
5. Click the Database icon (first icon on the left) to add a database connection and query, then click anywhere on the window.
6. JBuilder calls the connection *database1*. Rename this to *toursDB* by filling in the appropriate text in the Component Inspector.
7. Click the *connection* row (just below the *name* row in the Component Inspector). This is where you define the connection to the database. Click the “...” button.
8. The connection panel appears. Type in the database connection URL and the embedded Cloudscape driver name in the appropriate boxes.

   ```
   jdbc:cloudscape:toursDB
   ```

   ```
   COM.cloudscape.core.JDBCDriver
   ```



Connection URL: jdbc:cloudscape:toursDB

Username:

Password:

Driver class: COM.cloudscape.core.JDBCDriver

9 Click OK. Do not test the connection.

10 Select *borland.sql.QueryDataSet* in the Component Palette. Click anywhere in the window.

11 Type HotelInfo in the name field of the QueryDataSet's Component Palette.

12 Click query in the component palette, then click on the ... button.

13 Choose toursDB as the database.

14 Type the following text into the query window:

```
SELECT * FROM Hotels
```

15 Do not test the query.

16 Click OK.

17 Add a grid control to the window's scroll panel. This is a grid into which the results of the query will flow.

18 Select the JBCL tab in the Component Palette.

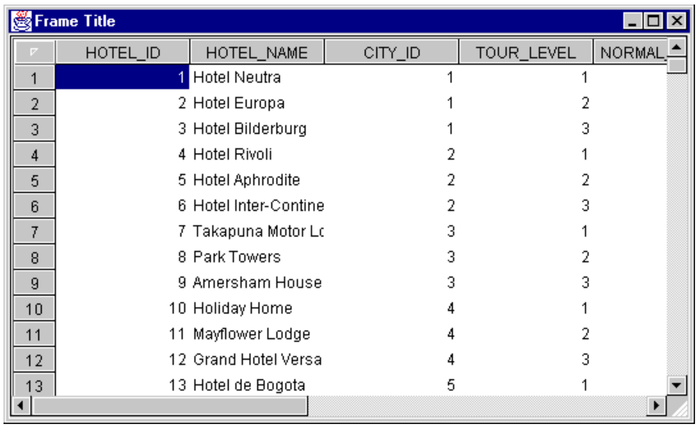
19 Click the borland.jbcl.control.GridControl icon, and click in the upper left-hand corner of the frame.

20 Click in the dataSet row.

21 Select HotelInfo as the source of the data. You will get an error message saying that the driver cannot be loaded; ignore this.

Run the GUI

1. Select HotelDisplayer in the Navigation pane of the AppBrowser.
2. Choose Run HotelDisplayer from the Run menu. The new mini-application should appear. It should display data from the *Hotels* table.



	HOTEL_ID	HOTEL_NAME	CITY_ID	TOUR_LEVEL	NORMAL_
1	1	Hotel Neutra	1	1	
2	2	Hotel Europa	1	2	
3	3	Hotel Bilderburg	1	3	
4	4	Hotel Rivoli	2	1	
5	5	Hotel Aphrodite	2	2	
6	6	Hotel Inter-Contine	2	3	
7	7	Takapuna Motor Lc	3	1	
8	8	Park Towers	3	2	
9	9	Amersham House	3	3	
10	10	Holiday Home	4	1	
11	11	Mayflower Lodge	4	2	
12	12	Grand Hotel Versa	4	3	
13	13	Hotel de Bogota	5	1	

Strategy Two: Live Connection to Cloudscape Running Inside RmiJdbc Server

Start the Server

Before you begin, start RmiJdbc server.

```
java -Dcloudscape.system.home=c:\cloudscape\demo\databases
```

Alter the *JBuilder.ini* File for JBuilder's embedded JVM

When working with live connections, ... it does not launch a new JVM. Instead, it uses . . .

1. With a text editor, open the file *JBuilder.INI*, which is in the /bin directory of the JBuilder directory.
2. Add the RmiJdbc classes, client.jar, and the JBMSTours classes to the IDEClassPath setting. For example:

```
c:\cloudscape\lib\client.jar;c:\cloudscape\frameworks\RmiJdbc\c
lasses\RmiJdbc.jar;c:\cloudscape\demo\programs\tours;
```

3 Save the file.

Create a New Project

1 Start JBuilder, and create a new project by choosing File->New Project. Call the project *tours.jpr*. It should be in a project folder called *toursGUI*.

Set up the Project Environment

You will need to set up your project correctly so that you can run your application from JBuilder. Setting up the project involves specifying the class path to include the Cloudscape libraries and to set any needed Java VM parameters.

1 Choose Project Properties from the File menu.

2 Click Libraries.

3 Click New.

4 Name the library CloudscapeAndTours. Set its class path to include the Cloudscape's client library, the RmiJdbc library, and the JBMSTours classes. For example:

```
c:\cloudscape\lib\client.jar;c:\cloudscape\demo\programs\tours;
c:\cloudscape\frameworks\RmiJdbc\classes\RmiJdbc.jar
```

NOTE: If you were going to deploy this as an embedded application, . . .

5 Click OK.

6 Click Add. Select CloudscapeAndTours.

7 Select the Run/Debug tab.

8 Set the *cloudscape.system.home* property to the location of the built-in *toursDB* demo database as a Java VM parameter, like so:

9 Select Send Run Output to Execution Log.

10 Click OK.

11 Save your changes (choose Save from the File menu).

Begin the GUI Application

Now that the environment is set up correctly, create a new application.

1 Create a new application by selecting tours.jpr in the Navigation pane of the ApplicationBrowser, then choosing File-> New.

2 Select application and click OK.

3 Call the class *HotelDisplayer*. It should be in the *toursGUI* package

4 Click Finish.

Test the Environment

To test the new environment, we will use one of the JBMSTours applications, called *GenerateReport*. The application does not have a GUI, but it accesses toursDB, so if it runs successfully, the environment is set up correctly.

1 Click the folder icon with the plus sign on it in the AppBrowser Navigation pane to add a new file to the project.

2 Navigate to the location of the JBMSTours class files (*%cloudscape_install%/demo/programs/tours/JBMSTours*).

3 Select *GenerateReport.java*, and click Open.

4 Choose Run GenerateReport from the Run menu.

5 Wait a few seconds, then choose Execution Log from the View menu to examine the output. A successful run looks something like this (an excerpt):

```
D:\jbuilder2\java\bin\javaw.exe -
Dcloudscape.system.home=c:\cloudscape\demo\databases -
classpath
"D:\jbuilder2\myclasses;D:\jbuilder2\lib\swingall.jar;D:\jbui
lder2\lib\jbcl2.0.jar;D:\jbuilder2\lib\jbcl2.0-
res.jar;D:\jbuilder2\lib\jgl3.1.0.jar;c:\cloudscape\lib\cloud
scape.jar;c:\cloudscape\lib\tools.jar;c:\cloudscape\demo\prog
rams\tours;D:\jbuilder2\dependency\~CloudscapeAndTours;D:\jbu
ilder2\java\lib\classes.zip" JBMSTours.GenerateReport
AppAccelerator(tm) 1.1.034 for Java (JDK 1.1), x86 version.
Copyright (c) 1998 Borland International. All Rights
Reserved.
GenerateReport starting
Loaded the embedded JDBC driver
Connected to database toursDB
Hottest Average Temperature in each Region
Region: Africa 89.0
Region: Asia 97.0
Region: Australia and New Zealand 71.0
Region: Central Asia 90.0
Region: Europe 68.0
. . .
```

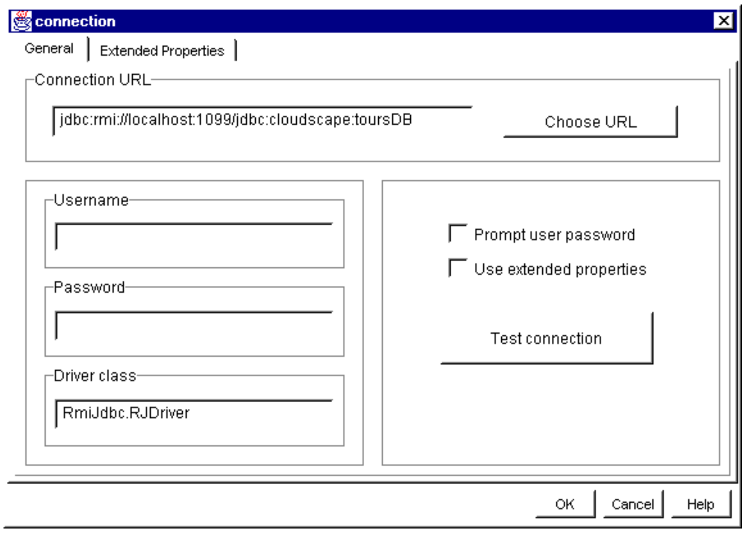
NOTE: If you have any errors, double-check that you have specified the class path for CloudscapeAndTours correctly in the project properties. You may see errors regarding DatabasePreloader. Ignore those errors.

Build the GUI

1. Select HotelDisplayer in the AppBrowser. JBuilder automatically included a Frame in that class called Frame1 when you created *HotelDisplayer*. Click the icon for *Frame1*, and select the Design tab.
2. Select the AWT tab in the Component Palette.
3. Click the *java.awt.scrollPanel* icon (third from the right), then click anywhere in the window.
4. Select the Data Express tab (at the top).
5. Click the Database icon (first icon on the left) to add a database connection and query, then click anywhere on the window.
6. JBuilder calls the connection *database1*. Rename this to *toursDB* by filling in the appropriate text in the Component Inspector.
7. Click the *connection* row (just below the *name* row in the Component Inspector). This is where you define the connection to the database. Click the “...” button.
8. The connection panel appears. Type in the database connection URL and the RmiJdbc client driver name in the appropriate boxes.

```
jdbc:rmi://localhost:1099/jdbc:cloudscape:toursDB
```

```
RmiJdbc.RJDriver
```



Connection URL: jdbc:rmi://localhost:1099/jdbc:cloudscape:toursDB

Username

Password

Driver class: RmiJdbc.RJDriver

Prompt user password

Use extended properties

9 Click OK.

10 Click Test Connection.

11 Select *borland.sql.QueryDataSet* in the Component Palette. Click anywhere in the window.

12 Type HotelInfo in the name field of the QueryDataSet's Component Palette.

13 Click query in the component palette, then click the ... button.

14 Choose toursDB as the database.

15 Type the following text into the query window:

```
SELECT * FROM Hotels
```

16 Click Test Query.

17 Click OK.

18 Add a grid control to the window's scroll panel. This is a grid into which the results of the query will flow.

19 Select the JBCL tab in the Component Palette.

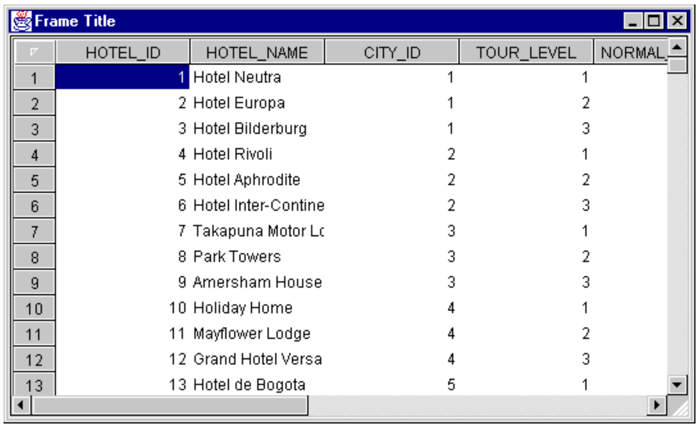
20 Click the borland.jbcl.control.GridControl icon, and click in the upper left-hand corner of the frame.

21 Click in the dataSet row.

22 Select HotelInfo as the source of the data..

Run the GUI

1. Select HotelDisplayer in the Navigation pane of the AppBrowser.
2. Choose Run HotelDisplayer from the Run menu. The new mini-application should appear. It should display data from the *Hotels* table.



	HOTEL_ID	HOTEL_NAME	CITY_ID	TOUR_LEVEL	NORMAL_
1	1	Hotel Neutra	1	1	
2	2	Hotel Europa	1	2	
3	3	Hotel Bilderburg	1	3	
4	4	Hotel Rivoli	2	1	
5	5	Hotel Aphrodite	2	2	
6	6	Hotel Inter-Contine	2	3	
7	7	Takapuna Motor Lc	3	1	
8	8	Park Towers	3	2	
9	9	Amersham House	3	3	
10	10	Holiday Home	4	1	
11	11	Mayflower Lodge	4	2	
12	12	Grand Hotel Versa	4	3	
13	13	Hotel de Bogota	5	1	