Database Access by Applets

Create Database Table

Once you have access to a database, create a table in it for the examples in this lesson. You need a table with one text field for storing character data.

```
TABLE DBA (
        TEXT            varchar2(100),
        primary key (TEXT)
)
```

If we run the applet without a policy file, we will get a stack trace indicating permission errors. The Lesson 6 applet example provided the policy file and told you how to launch the applet with it. This lesson shows you how to read the stack trace to determine the permissions you need in a policy file.

To keep things interesting, this lesson has two versions of the database access applet: one uses the JDBC driver, and the other uses the the JDBC-ODBC bridge with an Open DataBase Connectivity (ODBC) driver.

Both applets do the same operations to the same database table using different drivers. Each applet has its own policy file with different permission lists and has different requirements for locating the database driver

JDBC Driver

The JDBC driver is used from a program written exclusively in the Java language (Java program). It converts JDBC calls directly into the protocol used by the DBMS. This type of driver is available from the DBMS vendor and is usually packaged with the DBMS software.

Starting the Applet: To successfully run, the DbaAppl.java applet needs an available database driver and a policy file. This section walks through the steps to get everything set up. Here is the `DbaAppl.html` file for running the `DbaAppl` applet:

```
import java.awt.Color;

import java.awt.BorderLayout;

import java.awt.event.*;

import java.applet.Applet;

import javax.swing.*;

import java.sql.*;

import java.net.*;

import java.io.*;

public class DbaAppl extends Applet implements ActionListener {

  JLabel text, clicked;

  JButton button, clickButton;

  JTextField textField;
```

```
  private boolean _clickMeMode = true;
  private Connection c;
  final static private String _driver = "oracle.jdbc.driver.OracleDriver";
  final static private String _url =
"jdbc:oracle:thin:username/password@(description=(address_list=(address=(protocol=tcp)(host=developer)(
port=1521)))(source_route=yes)(connect_data=(sid=ansid)))";

   public void init(){
    setBackground(Color.white);
    text = new JLabel("Text to save to file:");
    clicked = new JLabel("Text retrieved from file:");
    button = new JButton("Click Me");
    button.addActionListener(this);
    clickButton = new JButton("Click Again");
    clickButton.addActionListener(this);
    textField = new JTextField(20);
    setLayout(new BorderLayout());
    setBackground(Color.white);
    add(BorderLayout.NORTH, text);
    add(BorderLayout.CENTER, textField);
    add(BorderLayout.SOUTH, button);
  }
  public void start(){
    System.out.println("Applet starting.");
  }
  public void stop(){
    System.out.println("Applet stopping.");
  }
  public void destroy(){
```

```
 System.out.println("Destroy method called.");
}

public void actionPerformed(ActionEvent event){
  try{
     Class.forName (_driver);
     c = DriverManager.getConnection(_url);
   }catch (java.lang.ClassNotFoundException e){
    System.out.println("Cannot find driver");
    System.exit(1);
   }catch (java.sql.SQLException e){
    System.out.println("Cannot get connection");
    System.exit(1);
   }
  Object source = event.getSource();
  if(source == button){
    if(_clickMeMode){
      JTextArea displayText = new JTextArea();
      try{
  //Write to database
       String theText = textField.getText();
       Statement stmt = c.createStatement();
       String updateString = "INSERT INTO dba VALUES ('" + theText + "')";
       int count = stmt.executeUpdate(updateString);
 //Read from database
       ResultSet results = stmt.executeQuery("SELECT TEXT FROM dba ");
       while(results.next()){
         String s = results.getString("TEXT");
         displayText.append(s + "\n");
```

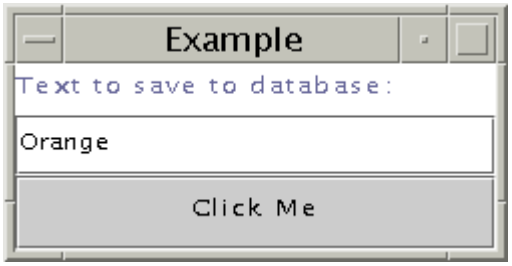
```
        }
        stmt.close();
        }catch(java.sql.SQLException e){
        System.out.println("Cannot create SQL statement");
        System.exit(1);
        }
//Display text read from database
        text.setText("Text retrieved from file:");
        button.setText("Click Again");
        _clickMeMode = false;
//Display text read from database
      } else {
        text.setText("Text to save to file:");
        textField.setText("");
        button.setText("Click Me");
        _clickMeMode = true;
      }
    }
  }
}
```

```
<HTML>
<BODY>

<APPLET CODE=DbaAppl.class
  WIDTH=200
  HEIGHT=100>
</APPLET>

</BODY>
</HTML>
```

Output:-



When Application Starts

And here is how to start the applet with appletviewer:

```
appletviewer DbaAppl.html
```

Locating the Database Driver: Assuming the driver is not available to the `DriverManager` for some reason, the following error generates when you click the `Click Me` button.

```
cannot find driver
```

This error means the DriverManager looked for the JDBC driver in the directory where the applet HTML and class files are and could not find it. To correct this error, copy the driver to the directory where the applet files are, and if the driver is bundled in a zip file, unzip the zip file so the applet can access the driver.

Once you have the driver in place, launch the applet again.

```
appletviewer DbaAppl.html
```

Reading a Stack Trace: Assuming the driver is locally available to the applet, if the DbaAppl.java applet is launched without a policy file, the following stack trace is generated when the end user clicks the `Click Me` button.

```
java.security.AccessControlException: access denied
(java.net.SocketPermission developer resolve)
```

The first line in the above stack trace tells you access is denied. This means this stack trace was generated because the applet tried to access a system resource without the proper permission. The second line means to correct this condition you need a `SocketPermission` that gives the applet access to the machine (`developer`) where the database is located.

You can use Policy tool to create the policy file you need, or you can create it with an ASCII editor. Here is the policy file with the permission indicated by the stack trace:

```
grant {
  permission java.net.SocketPermission "developer",
    "resolve";
  "accessClassInPackage.sun.jdbc.odbc";
};
```

Run the applet again, this time with a policy file named `DbaApplPol` that has the above permission in it:

```
appletviewer -J-Djava.security.policy=DbaApplPol
                                      DbaAppl.html
```

You get a stack trace again, but this time it is a different error condition.

```
java.security.AccessControlException: access denied
(java.net.SocketPermission
129.144.176.176:1521 connect,resolve)
```

Now you need a `SocketPermission` that allows access to the Internet Protocol (IP) address and port on the `developer` machine where the database is located.

Here is the `DbaApplPol` policy file with the permission indicated by the stack trace added to it:

```
grant {
  permission java.net.SocketPermission "developer",
                       "resolve";
  permission java.net.SocketPermission
  "129.144.176.176:1521", "connect,resolve";
};
```

Run the applet again. If you use the above policy file with the Socket permissions indicated, it works just fine.

```
  appletviewer -J-Djava.security.policy=DbaApplPol
                                          DbaAppl.html
```

JDBC-ODBC Bridge with ODBC Driver

Open DataBase Connectivity (ODBC) is Microsoft's programming interface for accessing a large number of relational databases on numerous platforms. The JDBC-ODBC bridge is built into the Solaris and Windows versions of the Java platform so you can do two things:

1. Use ODBC from a Java program
2. Load ODBC drivers as JDBC drivers. This example uses the JDBC-ODBC bridge to load an ODBC driver to connect to the database. The applet has no ODBC code, however.

The `DriverManager` uses environment settings to locate and load the database driver. For this example, the driver file does not need to be locally accessible.

Start the Applet: Here is the `DbaOdb.html` file for running the `DbaOdbAppl` applet:

```
<HTML>
<BODY>

<APPLET CODE=DbaOdbAppl.class
  WIDTH=200
  HEIGHT=100>
</APPLET>

</BODY>
</HTML>
```

And here is how to start the applet:

```
  appletviewer DbaOdb.html
```

Reading a Stack Trace: If the DbaOdbAppl.java applet is launched without a policy file, the following stack trace is generated when the end user clicks the `Click Me` button.

```
  java.security.AccessControlException: access denied
  (java.lang.RuntimePermission
  accessClassInPackage.sun.jdbc.odbc )
```

The first line in the above stack trace tells you access is denied. This means this stack trace was generated because the applet tried to access a system resource without the proper permission. The second line means you need a `RuntimePermission` that gives the applet access to the `sun.jdbc.odbc` package. This package provides the JDBC-ODBC bridge functionality to the Java[1] virtual machine (VM).

You can use Policy tool to create the policy file you need, or you can create it with an ASCII editor. Here is the policy file with the permission indicated by the stack trace:

```
grant {
  permission java.lang.RuntimePermission
    "accessClassInPackage.sun.jdbc.odbc";
};
```

Run the applet again, this time with a policy file named `DbaOdbPol` that has the above permission in it:

```
  appletviewer -J-Djava.security.policy=DbaOdbPol
                                          DbaOdb.html
```

You get a stack trace again, but this time it is a different error condition.

```
  java.security.AccessControlException:
  access denied (java.lang.RuntimePermission
  file.encoding read)
```

The stack trace means the applet needs read permission to the encoded (binary) file. Here is the `DbaOdbPol` policy file with the permission indicated by the stack trace added to it:

```
  grant {
    permission java.lang.RuntimePermission
       "accessClassInPackage.sun.jdbc.odbc";
    permission java.util.PropertyPermission
       "file.encoding", "read";
  };
```

Run the applet again. If you use the above policy file with the Runtime and Property permissions indicated, it works just fine.

```
  appletviewer -J-Djava.security.policy=DbaOdbPol
                                          DbaOdb.html
```
