Oracle + JDBC

A JDBC egy Java alapú kapcsolatot biztosít az adatbázis szerver és egy Java alapú kliens között.

Oracle Express Edition beállítások (Linuxon):

```
JAVA_HOME=/usr/java/jdk1.6.0_24/
ORACLE_HOME=/usr/lib/oracle/xe/app/oracle/product/10.2.0/server/
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
CLASSPATH=$ORACLE_HOME/jdbc/lib/ojdbc14.jar:$ORACLE_HOME/jlib/orai18n.jar:
$JAVA_HOME/src.zip
```

Példaprogram:

```
import java.sql.*;
import oracle.jdbc.*;
import oracle.jdbc.pool.OracleDataSource;

import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;

class JDBCTestGUI_xe extends JFrame implements ActionListener, ItemListener
{
  private JTable result_table;
  private JButton search_button;
  private TextField name_field;
  private JPanel input_panel;
  private JPanel output_panel;
  private JPanel static_main_panel;
  private DefaultTableModel table_model;
  private JScrollPane scroll_pane;
  private Vector column_names_vector;
  private Choice field_choice;
  private String choice_string;
  private JTabbedPane tabbed_pane;

  private ResultSet rs;
  private Statement stmt;


  /**
   *  Constuctor. Creating and initializing objects.
   * */
  public JDBCTestGUI_xe(String title) {
        super( title );
        this.setSize( 600, 400 );
        this.static_main_panel = new JPanel();
        this.output_panel = new JPanel();
        this.input_panel = new JPanel();
        this.name_field = new TextField( 50 );
        this.search_button = new JButton( "Search" );
        this.field_choice = new Choice();
```

Konstruktor, beállítások

```


        createGUI();

        try {

            /* Connect to the Oracle Database and using the "HR" user's schema */
            OracleDataSource ods = new OracleDataSource();
            ods.setURL("jdbc:oracle:thin:hr/hr@localhost:1521/XE");
            Connection conn = ods.getConnection("HR","HR");
            stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_READ_ONLY);
```

Kapcsolódás az adatbázishoz

```
      } catch ( Exception ex ) {
          ex.printStackTrace();
      }



 }
 /**
  *    Creating the graphical user interface for the program.
  **/
 public void createGUI() {

     tabbed_pane = new JTabbedPane(JTabbedPane.TOP); // the tabbed pane will be show more sheets
     tabbed_pane.addTab( "Static SQL", this.static_main_panel );  // actually only one sheet is used

     this.static_main_panel.setLayout( new BorderLayout() ); // the main panel for demonstrates the static SQL queries
     this.input_panel.setLayout( new GridLayout(3,3) );
     this.output_panel.setLayout( new BorderLayout() );
```

Grafikus kezelői felület,
elrendezés beállítása

```
     // the labels and input fields
     this.input_panel.add( new Label("Content:") );
     this.input_panel.add( this.name_field );
     this.input_panel.add( new Label("Search in field:") );
     this.input_panel.add( this.field_choice );
     this.input_panel.add( this.search_button );

     this.getContentPane().add( tabbed_pane );
     this.static_main_panel.add( input_panel, BorderLayout.NORTH );
     this.static_main_panel.add( output_panel, BorderLayout.CENTER);


     // adding event listeners
     this.field_choice.addItemListener(this);
     this.search_button.addActionListener(this);
```

Eseménykezelés
hozzárendelése

```
     // column names in a vector
     this.column_names_vector = new Vector();
     this.column_names_vector.add( "First name" );
     this.column_names_vector.add( "Last name" );
     this.column_names_vector.add( "Email" );
     this.column_names_vector.add( "Salary" );
     this.column_names_vector.add( "Department" );

     // need to use a table model for dynamic table handling
     this.table_model = new DefaultTableModel( column_names_vector, 0 );
     this.result_table = new JTable( table_model );
   this.scroll_pane = new JScrollPane(result_table);
     this.output_panel.add( scroll_pane );
```

Tábla beállítása az
eredményekhez

```
     // the column names listed into a drop-down list
     this.field_choice.add( "First name" );
     this.field_choice.add( "Last name" );
     this.field_choice.add( "Email" );
     this.field_choice.add( "Department" );

     choice_string = "First name";

 }


 public void actionPerformed( ActionEvent e ) {
     if ( e.getSource() == this.search_button ) {

          /*************** Starting queries  ***************/

          String sql = "";
          if ( this.name_field.getText().equals("") ) {


               sql = "SELECT first_name, last_name, email, salary, department_name FROM employees, departments WHERE
employees.department_id = departments.department_id ORDER BY last_name";
```

```
        } else {
            if ( choice_string == "Last name" ) {
                sql = "SELECT first_name, last_name, email, salary, department_name FROM employees, departments WHERE
                        employees.department_id = departments.department_id AND last_name LIKE '"+ name_field.getText() +
                        "' ORDER BY last_name";
            } else if (choice_string == "First name") {
                sql = "SELECT first_name, last_name, email, salary, department_name FROM employees, departments WHERE
                        employees.department_id = departments.department_id AND first_name LIKE '"+ name_field.getText() +
                        "' ORDER BY last_name";
            } else if (choice_string == "Email") {
                sql = "SELECT first_name, last_name, email, salary, department_name FROM employees, departments WHERE
                        employees.department_id = departments.department_id AND email LIKE '"+ name_field.getText() +
                        "' ORDER BY last_name";
            } else if (choice_string == "Department") {
                sql = "SELECT first_name, last_name, email, salary, department_name FROM employees, departments WHERE
                        employees.department_id = departments.department_id AND department_name LIKE '"+
                        name_field.getText() +"' ORDER BY last_name";
            }

        }

        try {
            System.out.println( sql );
            rs = stmt.executeQuery( sql );

            // removing all rows from the table
            int count = table_model.getRowCount();
            for ( int i = count-1; i>=0; i-- ) {
                table_model.removeRow(i);
            }
            repaint();

            while (rs.next()) {
                String row[] = {rs.getString(1), rs.getString(2), rs.getString(3), rs.getInt(4) + "", rs.getString(5)};

                this.table_model.addRow( row ); // adding new row into the table
            }
            repaint();
        }catch ( SQLException ex ) {
            ex.printStackTrace();
        }

    }
}

public void itemStateChanged( ItemEvent e ) {
      this.choice_string = field_choice.getSelectedItem();
}


public static void main (String args[]) throws SQLException
{
  OracleDataSource ods = new OracleDataSource();
  ods.setURL("jdbc:oracle:thin:hr/hr@localhost:1521/XE");
  Connection conn = ods.getConnection("HR","HR");

  // Create Oracle DatabaseMetaData object
  DatabaseMetaData meta = conn.getMetaData();


  /**************/

  JDBCTestGUI_xe gui = new JDBCTestGUI_xe( "Test application to try JDBC with Oracle" );

  gui.setVisible(true);
  gui.show();

   // this is an implementation of window listener
    // the program will be stopped if you close the main frame
    WindowListener listener = new WindowAdapter() {
      public void windowClosing(WindowEvent we) {
        System.exit(0);
      }
      };
      gui.addWindowListener(listener);

    /**************/



  }
}
```