DBSELECTOPTION

The DBSELECTOPTION control manages a single filter criterion of a database query. In contrast to the DBFIELD control, it allows the input of several values in a convenient way. Each value is assigned to an operator (e.g. "equals", "like", "between", etc.). The operator-value pairs are linked with a logical "OR" (represented by the string "||"). You can either directly input a string of operator-value pairs into the DBSELECTOPTION, or you invoke the value help. The resulting pop-up displays a grid of operator-value pairs. Maybe this still sounds rather difficult - wait for the following example.

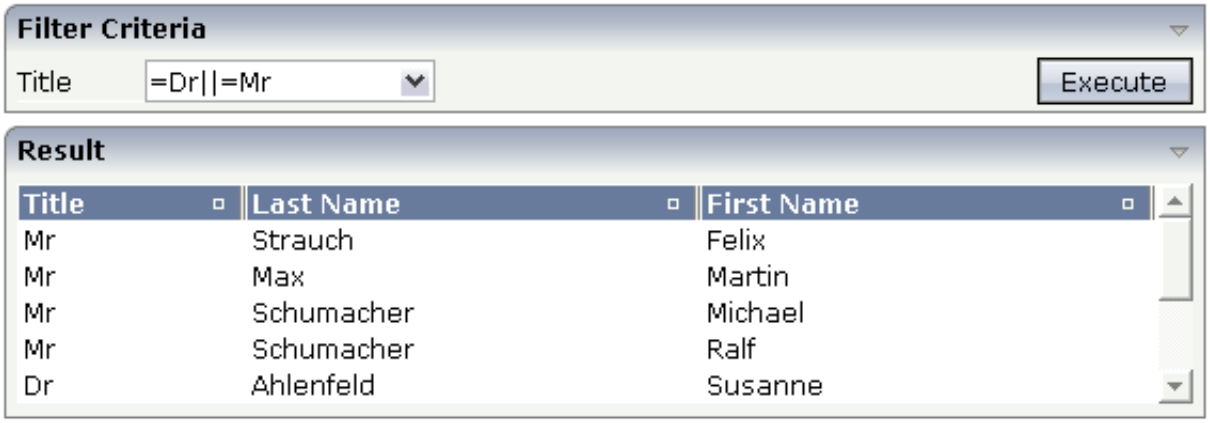
The grid pop-up itself offers a value help for entering appropriate values. The list of valid values (with an optional description) is read from database. The value help table/column (that contains the valid values) can differ from the table on which the query is executed.

This chapter covers the following topics:

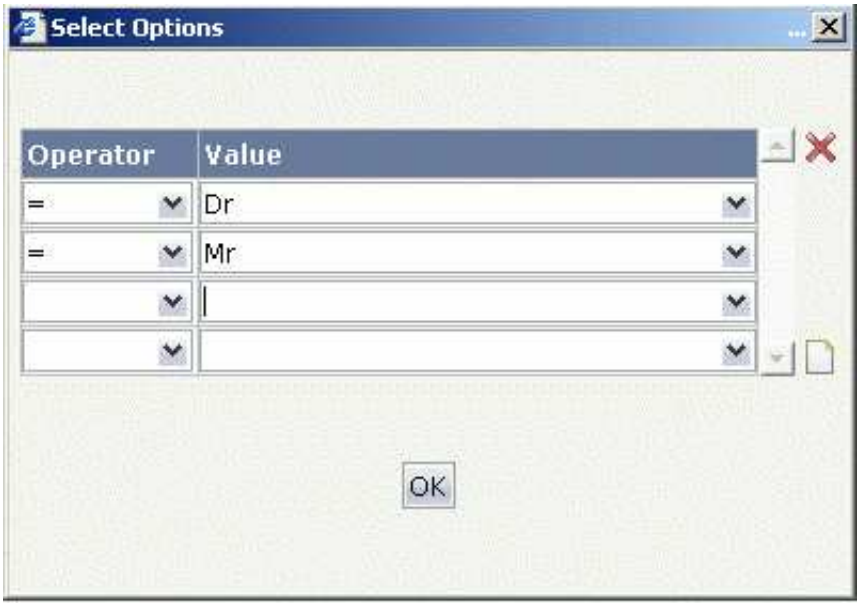
- Example
- Properties

Example

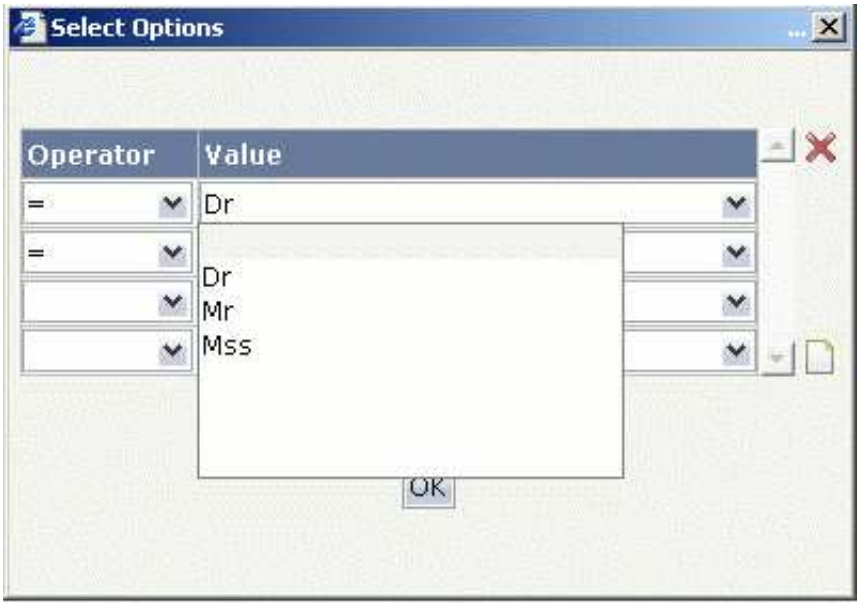
The following image shows an example in which the DBSELECTOPTION manages the filter criterion "Title" within a simple business partner report. The report shows partners that have a title equal to "Dr" or equal to "Mr".



On value help request, the following pop-up appears:



The first column defines the operator, the second the filter value. The value help of the first column returns a list of valid operators. The second column has value help, too (see properties `valuehelptable` and `valuehelpcolumn`). The following pop-up shows the list of valid values read from the table "TITLE".



Have a look at the XML layout definition:

```
<rowarea name="Filter Criteria">
    <itr>
        <label name="Title" width="60">
        </label>
        <dbselectoption querycolumn="TITLE" datasource="addressdb"
                        valueprop="dBSelectOptionTitle" valuehelptable="TITLE"
                        valuehelpcolumn="ID">
        </dbselectoption>
        <hdist width="100%">
        </hdist>
```

```
        <button name="Execute" method="onExecute">
        </button>
    </itr>
</rowarea>
<vdist>
</vdist>
<rowarea name="Result" height="140">
    <itr height="100%">
        <textgridsss2 griddataprop="result" rowcount="5" width="100%">
            <column name="Title" property="TITLE" width="20%">
            </column>
            <column name="Last Name" property="LASTNAME" width="40%">
            </column>
            <column name="First Name" property="FIRSTNAME" width="40%">
            </column>
        </textgridsss2>
    </itr>
</rowarea>
```

The corresponding adapter code is:

```
package com.softwareag.cis.test20;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;

import com.softwareag.cis.server.Adapter;
import com.softwareag.cis.server.util.DBSELECTOPTIONInfo;
import com.softwareag.cis.server.util.DBTEXTGRIDCollection;
import com.softwareag.cis.server.util.DBUtil;
import com.softwareag.cis.server.util.DelegateError;
import com.softwareag.cis.server.util.IDBCondition;
import com.softwareag.cis.server.util.IDBQUERYConnectionProvider;

public class DBSELECTOPTIONDemoAdapter
    extends Adapter
    implements IDBQUERYConnectionProvider
{
    // ------------------------------------------------------------------------
    // members
    // ------------------------------------------------------------------------

        Connection m_connection;

    // ------------------------------------------------------------------------
    // property access
    // ------------------------------------------------------------------------

    // property >dbselectOptionTitle<
    DBSELECTOPTIONInfo m_dBSelectOptionTitle = new DBSELECTOPTIONInfo(this, this);
    public DBSELECTOPTIONInfo getDBSelectOptionTitle() { return m_dBSelectOptionTitle; }

    // property >result<
    DBTEXTGRIDCollection m_result = new DBTEXTGRIDCollection();
    public DBTEXTGRIDCollection getResult() { return m_result; }
    public void setResult(DBTEXTGRIDCollection value) { m_result = value; }


    // ------------------------------------------------------------------------
    // public adapter methods
    // ------------------------------------------------------------------------

    /** implementation of interface IDBQUERYConnectionProvider */
    public Connection getDBConnection(String datasource)
    {
       if (m_connection == null)
       {
```

```
            try
            {
               Class.forName("org.hsqldb.jdbcDriver");
               m_connection = DriverManager.getConnection("jdbc:hsqldb:hsql://localhost", "sa", "");
            }
           catch (Exception exc)
           {
                throw new DelegateError(exc);
            }
        }
        return m_connection;
    }

    /** executes the query with current values of the filter criteria "Title". */
    public void onExecute()
    {
        try
        {
           StringBuffer sb = new StringBuffer();
           sb.append("SELECT * FROM BUSINESSPARTNER");
           DBUtil.addToQuery(sb, new IDBCondition[] { m_dBSelectOptionTitle}, true);

           String dataSource = m_dBSelectOptionTitle.getDataSource();
           Connection con = getDBConnection(dataSource);
           ResultSet rs = con.createStatement().executeQuery(sb.toString());
           m_result.initWithResultSet(rs);
        }
        catch (Exception exc)
        {
           throw new DelegateError(exc);
        }
    }
}
```

The adapter property `dbselectOptionTitle` is of type `DBSELECTOPTIONInfo` (from the package `com.softwareag.cis.server.util`). The `DBSELECTOPTIONInfo` implements (like all DB controls) the interface `IDBCondition` (`com.softwareag.cis.server.util`). On report execution, you may use the class `DBUtil` (`com.softwareag.cis.server.util`) to append the value of property `dbselectOptionTitle` to the SQL query. See the JavaDoc documentation of class `DBSELECTOPTIONInfo` and `DBUtil` for details.

The `DBSELECTOPTIONInfo` class does not open a database connection on its own (same to all DB controls). The embedding adapter provides for an implementation of interface `IDBQUERYConnectionProvider` (`com.softwareag.cis.server.util`) when creating a `DBSELECTOPTIONInfo` object. The interface method `getDBConnection` is called once - at the first time the `DBSELECTOPTIONInfo` has to access the database.

There are no updates (insert/update/delete) done with this connection. As the `DBSELECTOPTIONInfo` does not open the connection, it does not care about closing the connection.

For displaying the result, the class `DBTEXTGRIDCollection` (from the package `com.softwareag.cis.server.util`) is used. This class extends `TEXTGRIDCollection` by the ability to initialise the collection with a result set (`java.sql.ResultSet`). For each line of the result set, it creates an object of class `DBTEXTGRIDLine` (package `com.softwareag.cis.server.util`). Class `DBTEXTGRIDLine` implements the interface `IDynamicAccess`. With this, "normal" text grid controls can be used to visualize the data of the `DBTEXTGRIDCollection`.

Properties

Basic			
querycolumn	Name of the column in the query to that the filter criteria belongs to. This column may differ from the value help table/column (properties VALUEHELPTABLE, VALUEHELPCOLUMN). This name is used to build a SQL string in method "toSQLString".	Obligatory	
datasource	Logical identifier of the data source to use. This name is passed to the connection provider in method "IDBQUERYConnectionProvider.getDBConnection".	Obligatory	
valueprop	Property that returns a DBSELECTOPTIONInfo -instance. This instance provides for the value help read from database as well as for a convenient way to append the filter value to query string.	Obligatory	
valuehelptable	Name of the table from where the valid values of the DBSELECTOPTION control are stored.	Optional	
valuehelpcolumn	Name of the column from where the valid values of the DBSELECTOPTION control are stored.	Optional	
valuehelpcolumndescr	Name of a column where an additional description of the valid values is stored. The column must be inside the "value help table" (property VALUEHELPTABLE).	Optional	
width	Width of DBFIELD in pixels or as percentage value.	Optional	
length	Width of DBFIELD in amount of characters. WIDTH and LENGTH should not be used together.	Optional	
datatype	The DBSELECTOPTION control manages multiple "operator-value"-pairs. By default, each is managed as string. By the DATATYPE property, force the type of the data that is represented. As a consequence the DBFIELD controls inside the "operator-value"-popup is checking the data during input (e.g. if the DATATYPE is "int", it is not allowed to enter alphabetic characters) and adds a logic to transfer the data into various output formats (e.g. if the DATATYPE is "date", the date is formatted into the right date format). In addition it displays a standard "value help" popup dialog for some data types (e.g. if the DATATYPE is "date" then automatically a date input dialog pops up if invoking "value help").	Optional	int float date
flush	Flushing behaviour, please view "Common Rules" for details	Optional	screen server

displayonly	If set to "true", the DBFIELD will not be accessible for input. It is just used as an output field.	Optional	true false
align	Explicit Alignment	Optional	
valign	Explicit Alignment	Optional	
colspan	Number of columns occupied by this control.	Optional	
rowspan	Number of rows occupied by this control.	Optional	
fieldstyle	Explicit style information passed to the DBFIELD. Example: if you want the text inside the DBFIELD to be right aligned, define "text-align: right".	Optional	
helpid	Identifier that is used for building the URL of the online help page. Please refer to "Online Help Management" for details.	Optional	
comment	Comment without any effect on rendering and behaviour. The comment is shown in the layout editor’s tree view.	Optional	