Table of Contents

Metadata are an important element of JVx' model. Every column used in DataBook or Editor has a ColumnDefinition internally. This controls which editor will be used (date, text, number, selection, etc.), the datatype of the column, how many characters can be entered, and the like. A ColumnDefinition is created from metadata. The metadata are generated on server side.

In the simplest case, the metadata are read from the database, if a database is used. The metadata creation is done automatically and usually requires no changes.

However, it may be necessary in certain situations to adapt the standard mechanism to your own needs, e.g., add additional information, define standard values with your own algorithm, or set specific labels.

Working with your own metadata means that you understand the default behavior:

The class DBAccess is responsible for creating metadata from database tables and views. It also enables you to change default metadata creation and to implement your own requirements.

DBAccess is a server-side class. The metadata thus generated are transferred to the client as needed, and the model takes care of the conversion.

The creation of custom metadata is explained by the following example.

We want to extend the metadata with an additional field. It contains detail information about the field.

1. Create your own implementations of ColumnMetaData and ColumnDefinition

```
public static class MyColumnMetaData extends ColumnMetaData
{
    private String columnInfo;

    public String getColumnInfo()
    {
        return columnInfo;
    }

    public void setColumnInfo(String pColumnInfo)
    {
        columnInfo = pColumnInfo;
    }

    public ColumnDefinition createColumnDefinition() throws ModelException
    {
        MyColumnDefinition columnDef = new MyColumnDefinition();
        initializeColumnDefinition(columnDef);

        columnDef.setColumnInfo(columnInfo);

        return columnDef;
    }
}

public static class MyColumnDefinition extends ColumnDefinition
```

```
{
    private String columnInfo;

    public String getColumnInfo()
    {
        return columnInfo;
    }

    public void setColumnInfo(String pColumnInfo)
    {
        columnInfo = pColumnInfo;
    }
}
```

As can be seen already, both classes are linked. The class ColumnMetaData creates a new instance of ColumnDefinition.

In **Step 2*, we're going to configure a DBAccess object. It should use our new classes:

```
DBAccess dba = DBAccess.getDBAccess("<jdbcurl>");
dba.setUsername("<username>");
dba.setPassword("<password>");
dba.open();

dba.setColumnMetaDataCreator(new IColumnMetaDataCreator()
{
    public ColumnMetaData createColumnMetaData(DBAccess pDBAccess,
                                               ResultSetMetaData
pResultSetMetaData,
                                               int pResultSetColumnIndex)
    {
        MyColumnMetaData cmd = new MyColumnMetaData();
        pDBAccess.initializeColumnMetaData(pResultSetMetaData,
pResultSetColumnIndex, cmd);

        cmd.setColumnInfo(cmd.getName() + "(" + cmd.getLabel() + ")");

        return cmd;
    }
});
```

We have to implement the interface IColumnMetaDataCreator and simply return our own class. We set the new ColumnInfo before we return the object.

From now on all metadata instance are created using the `ColumnMetaDataCreator`.

The newly created MyColumnMetaData instance is transferred to the client as usual. The model continues to generate automatically our MyColumnDefinition instance via createColumnDefinition. No further action needed.

From:
http://doc.sibvisions.com/ - **Documentation**

Permanent link:
http://doc.sibvisions.com/jvx/server/storage/metadata

Last update: **2020/06/26 13:21**