

View mit Dokumenten erstellen

🌐 Sprache: **Deutsch** · [English](#)

🎓 View mit Dokumenten erstellen

Dokumente aus dem Dateisystem in einer View anzeigen – per Proxy-BO, Datenquelle und serverseitiger Regel.

TUTORIAL

ANWENDUNGSENTWICKLER

STAND: JUL 2026

GILT FUER NUCLOS 4.2026.X

Auf dieser Seite

- [Warum ein Proxy-BO?](#)
- [1. Proxy-BO anlegen](#)
- [2. Datenquelle für die View](#)
- [3. Interface in einer Regel implementieren](#)
- [4. Ins Layout einbinden](#)
- [Verwandte Seiten](#)

Warum ein Proxy-BO?

Für eine View, die **Dokumente** anzeigt, reicht ein virtuelles oder generisches Businessobjekt nicht: Die Dateien liegen nicht in der Datenbank, sondern im Serververzeichnis `[nuclos-home]/data/documents`. In der Datenbank stehen nur Name und ID.

Deshalb nutzen wir ein **Proxy-BO**. Damit wird die View per **Java-Code** erzeugt, sodass wir die Dokumente aus dem Dateisystem laden können.



Hinweis

Proxy-BOs lassen sich nur als Subform an ein Eltern-Objekt hängen. Soll es eigenständig sein, braucht es ein Schreib-Proxy-BO ([Beispiele](#)).

1. Proxy-BO anlegen

Erstelle über den BO-Wizard ein Proxy-BO mit mindestens einem Feld für den **Dokumentenanhang** und einer **Referenz auf das Eltern-Objekt**. Dabei wird automatisch ein Interface generiert, das wir später implementieren.

Attribute bearbeiten

Fügen Sie Attribute hinzu, ändern Sie vorhandene oder löschen Sie welche. Die Reihenfolge der Attribute wird in Listen und Unterformularen als Standard für die Spalteneinstellungen verwendet.

Anzeigename	Beschreibung	Datentyp	Feldbreite	Nachkommastellen	Eindeutig?	Historie?	Pflichtfeld?	Feldname	Javatype	Berechnet	Defaultwert	Attributgruppe
ElternObjekt	ElternObjekt	Referenzfeld	255		☐	☐	☒	elternobjekt	String	☐		Grunddaten
Dokument	Dokument	Dokumentenanhang	255		☐	☐	☐	dokument	GenericObjectDocumentFile	☐		Grunddaten

Proxy-BO im Wizard.

2. Datenquelle für die View

Lege eine Datenquelle an, die diese Felder liefert und per Parameter auf das Eltern-Objekt eingeschränkt werden kann:

- `elternid` – Referenz auf das Eltern-Objekt
- `dokumentid` – ID des Dokuments (Tabelle `t_ud_documentfile`)
- `dokumentname` – Name des Dokuments (Tabelle `t_ud_documentfile`)

Name
Beschreibung
Regelrelevant? ☐

Modell SQL **Vorschau**

Statement wieder herstellen

+

📁

Parameter...	Datentyp	Pflichtpara...
elternid	Long	☐

```

select
    --Referenz auf Eltern-Objekt
    a.intid,

    --Dokument-ID
    pd.struid_dokument,

    --Dokument-Name
    doc.strfilename,

    --Weitere Felder
    [...]

from elternobjekt a
--Weitere Joins
[...]
--Objekt mit Referenz auf Dokument
inner join tabellex x on [JOIN]
inner join t_ud_documentfile doc on doc.struid = x.struid_dokument
where [BEDINGUNGEN]
--Einschränkung auf Eltern-Objekt
and $elternid = a.intid

```

Datenquelle der View.

3. Interface in einer Regel implementieren

Lege über *Regelwerke* [Regeln \(serverseitig\)](#) eine Regel an (am besten nach dem Interface benannt). Für den Standardfall genügt `getByElternObjekt (Long)` – sie wird aufgerufen, wenn das Proxy-BO über das Layout des Eltern-Objekts angezeigt wird. Alle übrigen Methoden dürfen null zurückgeben.



Download aus dem Proxy-BO

Soll das Dokument direkt aus dem Proxy-BO heruntergeladen werden, **muss** zusätzlich `getId( . )` implementiert werden – mit einer je Dokument eindeutigen ID.

```

package org.nuclet.[nuclet];

import java.io.File;
import java.io.FileFilter;
import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
import java.nio.file.Path;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.apache.commons.io.filefilter.WildcardFileFilter;
import org.nuclos.api.common.NuclosFile;
import org.nuclos.api.exception.BusinessException;
import org.nuclos.api.provider.DatasourceProvider;
import org.nuclos.api.provider.FileProvider;

public class BeispielObjektProxyImpl implements org.nuclet.[nuclet].BeispielObjektProxy{
    /*

```

```

    * Getter-Methode für das Eltern-Objekt.
    * Wir nutzen den Parameter, um das Ergebnis der Datenquelle einzuschränken.
    * Es muss immer ein neues Proxy-Objekt erstellt werden, welches mit den Daten
    * aus der Datenquelle gefüttert wird.
    * Alternativ können die Daten auch von einer anderen Stelle kommen (Remote-Verbindung, QueryProvider
etc.).

    * Für den Dokumentenanhang holen wir uns mit Hilfe von dokumentid und dokumentname
    * über die Methode getDocument() die Datei aus dem Installationsverzeichnis.
    * @param pAngebotId = Id des Eltern-Objekts.
    * @return Alle Zeilen des Proxy-BOs, die im Eltern-Objekt angezeigt werden sollen.
    */
@Override
public List<BeispielObjekt> getByElternObjekt(java.lang.Long pElternobjektId) {
    List<BeispielObjekt> result = new ArrayList<>();
    /*
    * Setze das Verzeichnis, in dem die Datei zu finden ist.
    */
    File dir = new File(getHomeDir() + "/data/documents/");
    try{
        /*
        * Setze die Parameter für die Datenquelle und führe sie aus.
        */
        Map<String, Object> params = new HashMap<>();
        params.put("elternid", pElternobjektId);
        for (Object[] row : DataSourceProvider.run(BeispielDatenquelleDS.class, params).
getRows()){
            /*
            * Erstelle neues Proxy-BO und füttere es mit Daten.
            */
            BeispielObjekt b = new BeispielObjekt();
            Long elternid = (Long) row[x];
            b.setElternId(elternid);
            String dokumentid = (String) row[y];
            String dokumentname = (String) row[z];
            b.setDokument(getDocument(dokumentid, dokumentname, dir));
            /*
            * Setze weitere Felder
            */
            b.set...(..);

            result.add(b);
        }
    } catch (BusinessException e) {
        e.printStackTrace();
    }
    return result;
}

/*
* Ermittelt das Nuclos-Installationsverzeichnis.
*/
private String getHomeDir() {
    String home = "";
    try {
        Class<?> clz = Class.forName("org.nuclos.server.common.NuclosSystemParameters");
        Method getNuclosHome = clz.getMethod("getNuclosHome");
        home = ((Path) getNuclosHome.invoke(null)).toString();
    } catch (IllegalAccessException | IllegalArgumentException |
InvocationTargetException | ClassNotFoundException | NoSuchMethodException |
SecurityException e) {
        e.printStackTrace();
    }
    return home;
}

/*
* Suche das Dokument im gegebenen Verzeichnis, packe es in eine NuclosFile und gib
* dieser den gegebenen Namen (andernfalls würden die Dokumente in der View unter ihrer
* ID angezeigt werden).
* @param documentId = ID des Dokuments.
* @param anzeigename = Gewünschter Anzeigename des Dokuments.

```

```

    * @param dir = Verzeichnis, in dem das Dokument gesucht wird.
    * @return Eine NuclosFile oder null, wenn kein passendes Dokument gefunden wurde.
    */
    private NuclosFile getDocument(String documentId, String anzeigename, File dir) throws
BusinessException {
        FileFilter fileFilter = new WildcardFileFilter(documentId + ".*");
        File[] files = dir.listFiles(fileFilter);
        if (files.length == 1) {
            NuclosFile file = FileProvider.newFile(files[0]);
            file.setName(anzeigename);
            return file;
        }
        return null;
    }

    /*
    * Alle anderen Methoden müssen nicht weiter implementiert werden.
    */

    @Override
    public void setUser(org.nuclos.api.User user) {
    }
    @Override
    public List<BeispielObjekt> getAll() {
        return null;
    }
    @Override
    public List<java.lang.Long> getAllIds() {
        return null;
    }
    @Override
    public BeispielObjekt getById(java.lang.Long id) {
        return null;
    }
    @Override
    public List<BeispielObjekt> getByDokument(org.nuclos.api.UID pNuclosdocumentfileId) {
        return null;
    }
    @Override
    public void insert(BeispielObjekt pweitereDokumente) throws org.nuclos.api.exception.BusinessException {
    }
    @Override
    public void update(BeispielObjekt pweitereDokumente) throws org.nuclos.api.exception.BusinessException {
    }
    @Override
    public void delete(java.lang.Long id) throws org.nuclos.api.exception.BusinessException {
    }
    @Override
    public void commit() {
    }
    @Override
    public void rollback() {
    }
}

```

4. Ins Layout einbinden

Zuletzt das Proxy-BO wie eine normale Subform ins Layout des Eltern-Objekts einfügen.

Verwandte Seiten

Proxy Businessobjekt

Proxy-BO.

[Öffnen](#)

Regeln (serverseitig)

Regeln.

[Öffnen](#)

Datenquellen

Datenquellen.

[Öffnen](#)

