Proxy Businessobjekt

- Funktionsprinzip
- How-to: Proxy Businessobjekt erstellen
  - Proxy-BO anlegen
  - Attribute definieren
  - Interface implementieren
  - Implementierung registrieren
  - Proxy-BO im Formular verwenden
- Ergebnis
- Typische Einsatzfälle

Ein **Proxy Businessobjekt (Proxy-BO)** wird verwendet, wenn Daten **nicht direkt aus der Nuclos-Datenbank stammen**, sondern über **Java-Code aus externen Quellen** bereitgestellt oder geschrieben werden sollen (z. B. externe Systeme, Dateien oder Webservices).

Dabei wird ein **Interface erzeugt**, das in einer Regel implementiert wird. Diese Implementierung übernimmt das Lesen, Schreiben oder Löschen der Daten.

Typische Einsatzfälle:

- Integration von **Drittsystemen**
- Zugriff auf **Dateisystemdaten**
- Daten aus **Webservices**
- Daten, die **nicht in der Nuclos-Datenbank gespeichert** sind

Funktionsprinzip

Ein Proxy-Businessobjekt besteht aus zwei Teilen:

1. **Businessobjekt-Definition in Nuclos**
2. **Java-Implementierung des Proxy-Interfaces**

Die Benutzeroberfläche zeigt Datensätze wie bei einem normalen Businessobjekt an.
Die Daten werden jedoch durch den **Java-Code der Proxy-Implementierung bereitgestellt oder verändert**.

Example

```
@Override
public Object insert(final VirtualBO pVirtualBO) throws BusinessException {
        AnyBO anyBo = new AnyBO();
        anyBo.save();
        return anyBo.getId();
}
```

How-to: Proxy Businessobjekt erstellen

Proxy-BO anlegen

1. Öffne
   Konfiguration Businessobjekt
2. Erstelle ein neues Businessobjekt.
3. Aktiviere die Option **Proxy Businessobjekt**.

Dadurch wird automatisch ein **Java-Interface** für dieses BO generiert.

Attribute definieren

Lege die benötigten Attribute an, z. B.:

- ID
- Name
- Datum
- Referenz auf ein Elternobjekt

Proxy-BOs werden häufig **als Unterformular eines anderen Businessobjekts** verwendet.

---

3 Interface implementieren

Nuclos erzeugt ein Interface wie z. B.:

org.nuclet.businessentity.RechnungProxy

Dieses muss in einer Java-Klasse implementiert werden.

Beispiel:

```
public class RechnungProxyImpl implements org.nuclet.businessentity.RechnungProxy {

public List<Rechnung> getAll() {
// Daten aus externem System laden
}

public Rechnung getById(Long id) {
// Datensatz aus externem System laden
}

public void insert(Rechnung rechnung) throws BusinessException {
// Datensatz im externen System speichern
}

public void update(Rechnung rechnung) throws BusinessException {
// Datensatz aktualisieren
}

public void delete(Long id) throws BusinessException {
// Datensatz löschen
}
}
```

Diese Methoden definieren, **wie Nuclos mit den externen Daten interagiert**.

---

4 Implementierung registrieren

Die Implementierung muss **voll qualifiziert mit Package angegeben werden**, damit Nuclos sie erkennt.

---

5 Proxy-BO im Formular verwenden

Das Proxy-BO kann anschließend:

- als **Unterformular**
- oder als **Datenquelle für externe Daten**

verwendet werden.

Ergebnis

- Benutzer sehen Datensätze im Proxy-BO wie in normalen Businessobjekten.
- Daten werden jedoch **dynamisch über Java-Code geladen oder gespeichert**.
- Änderungen werden **im externen System** durchgeführt.

Typische Einsatzfälle

Szenario	Lösung
Rechnungen liegen in Fremdsystem	Proxy-BO liest Daten über API
Dokumente im Dateisystem	Proxy-BO lädt Dateien
Daten aus Webservice	Proxy-BO ruft Service auf
Daten sollen nicht lokal gespeichert werden	Proxy-BO ersetzt DB-Speicherung