ReporterPE – **Quickstart** Guide

(Version 1.5 vom 23.01.2024)

Vielen Dank für Ihr Interesse an ReporterPE.

Dieser PostExecuter ermöglicht Ihnen die Erstellung von PDF, XHTML und ODT Dokumenten aus Lobster _data, ohne auf die Nutzung von Jasper Reports angewiesen zu sein. Stattdessen nutzen Sie ein ODT Dokument (Open Office Format) als Template für die Erzeugung.

Systemvoraussetzungen:
ReporterPE erfordert Lobster _data 4.6.x oder höher. Diese Version (1.2.3) von ReporterPE wurde in Lobster _data 4.6.1 bis 4.6.5 getestet. Bitte beachten Sie, dass wir den reibungslosen Betrieb des PosteExecuters in anderen Versionen des _data oder bei anderen gegebenenfalls installierten Erweiterungen in /extlib nicht garantieren können.

Installation:

Entpacken Sie das heruntergeladene Archiv ReporterPE.zip und legen Sie die im Ordner /extlib enthaltene .jar Datei im Ordner /extlib Ihrer Lobster _data Installation ab.
Im Ordner /etc/admin/datawizard Ihrer Lobster _data Installation editieren Sie die Datei **custom_post_executer.properties**. Falls diese noch nicht existiert legen Sie sie an. Fügen Sie den Eintrag **de.derbrill.reporter.ReporterPE** hinzu.

Danach ist ein Neustart Ihres _data Systems erforderlich. Auf einem Lobster _data TEST System bedarf es keiner weiteren Aktion um mit ReporterPE zu arbeiten. Für ein Produktivsystem benötigen Sie einen Lizenzschlüssel.

Generelle Funktionsweise

ReporterPE befüllt ein von Ihnen erstelltes ODT Dokument anhand einer von Ihnen im Mapping erzeugten Struktur des Zielbaums.

Um ReporterPE als Postexecuter nutzen zu können, muss Ihr Mapping zuerst eine JSON Datei erzeugen.
Dementsprechend ist in Phase 5 **immer** die JSON Integration Unit zu nutzen. Die Struktur des Zielbaums in Phase 3 ist von dem von Ihnen erstellten Template abhängig. Die Feldnamen müssen den im Template angegebenen Variablen entsprechen.

Dorfstraße 30, 24790 Schülldorf
https://www.derbrill.de/ReporterPE

Lizenz für das Produktivsystem bereitstellen

Wenn Sie bereits einen Lizenzschlüssel für Ihr Lobster Produktivsystem von uns erhalten haben, legen sie diesen im Verzeichnis **./conf/ReporterPE/** Ihrer Lobster _data Prod Instanz ab. Die Lizenz ist an Ihre Lobster _data Installations-ID gebunden. Sollten Sie mehrere Lobster _data Produktiv Instanzen betreiben, so benötigen Sie eine Lizenz pro genutzter Installations-ID.

Wenn Sie einen Lizenzschlüssel erwerben möchten, kontaktieren Sie uns per eMail unter: helpdesk@derbrill.de

Erste Schritte – das Demo Profil

ReporterPE wird Ihnen mit einem Demo-Profil ausgeliefert. Wenn Sie die Package-Datei in Ihr Lobster _data System importieren können Sie den Funktionsumfang sofort testen. Die benötigten Konfigurationsdateien werden durch den Profilimport automatisch angelegt. Explizit wird in Verzeichnis conf der Ordner ReporterPE mit dem Unterordner Examples angelegt. Die benötigten Testdaten sind auch im Package enthalten.

Mapping Einstellungen

Um Variablen im Template zu befüllen, muss die Zielstruktur bestimmte Anforderungen erfüllen:

Einfache Variablen werden auf Wurzelebene des Zielbaums angelegt (genauer in dem Knoten den Sie in der IntegrationUnit als „Start at Node“ angeben).

Ein Feld mit dem Namen myField verweist im Template auf die Variable ${myField}

Felder in Unterknoten mit gesetztem Attribut maximum > 1 werden als **Listeneinträge** gewertet. Geben sei ein Knoten mit dem Namen myList. Dieser enthält die Felder myField1 und myField2.

Diese Felder werden im Template durch Angabe der Variablen ${myList.myField1} beziehungsweise $(myList.myField2} referenziert.

Verpflichtende Felder

Zwei Felder sind in Ihrem Mapping verpflichtend.
templatefile muss den Pfad zu Ihrem Template ODT file enthalten. Hierbei ist es möglich Dateien aus dem (erreichbaren) Dateisystem oder einer im Netzwerk erreichbaren Resource zu benutzen. Die Angabe erfolgt unter Angabe des gewünschten Protokolls (File: / http:// oder https://) gefolgt von der URL. Z.B. File:./conf/ReporterPe/examles/demo.odt oder
https://example.org/reporter/demo.odt

outputformat muss das gewünschte Ausgabeformat enthalten. Hierbei sind die Werte pdf, xhtml und odt möglich.

Magic extensions

Endet Ihr Feldname (und der Variablenname im Template) mit zwei Unterstrichen, gefolgt von styled (z.B. myField__styled) wird der Textinhalt als Markup gestylter Text interpretiert.

Unterstützt werden folgende Formatierungsauszeichnungen:
Fett
italic
<u>underline</u>
_{subscript}
^{superscript}

Farben können über <p> und tags gesetzt werden:

My mother has blue eyes.

Bitte beachten Sie, dass im Falle der Nutzung von styled Text einfache Zeilenumbrüche durch das tag <br/> ersetzt werden müssen. Hierbei ist essentiell das TAG XML wohlgeformt zu schließen. Anderenfalls kann es zu Darstellungsfehlern kommen.

Verweise auf Bilder:
Um die Eigenschaften eines Bildes im Template zu setzen erzeugen Sie einen Knoten mit gesetztem Attribut Maximum = 1. Der Name des Knotens muss mit zwei Unterstrichen, gefolgt von img enden. Zum Beispiel myImage__img. Der Knoten muss mindestens zwei Felder enthalten:

url: mit Verweis auf eine URL im Web, oder einem Pfad im (erreichbaren) Dateisystem. Wird eine Datei im Dateisystem referenziert, so muss der Inhalt des Felds mit dem Ausdruck **File:** beginnen. Beispiel: **File:./conf/images/myImage.png**
resize: true oder false

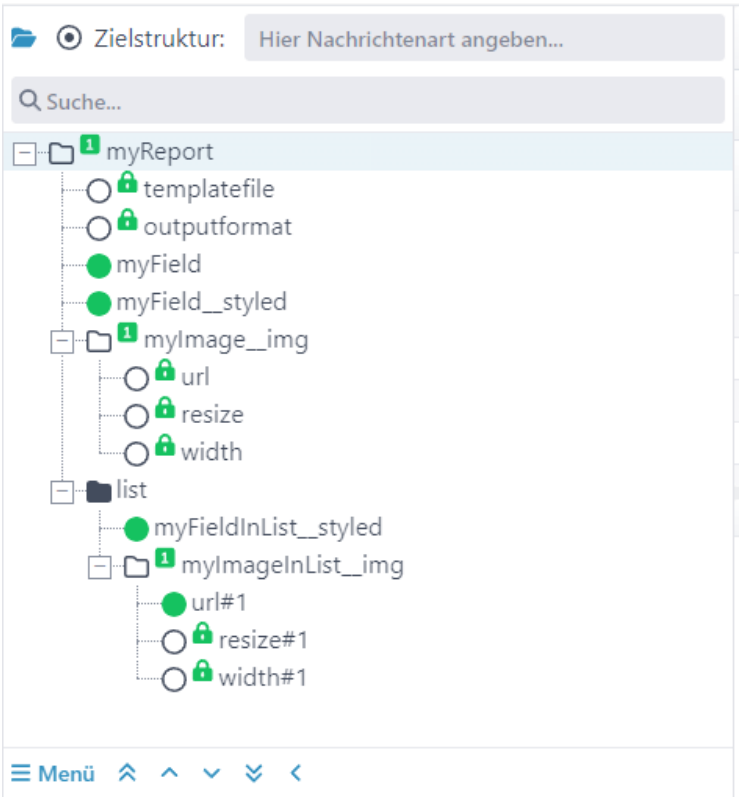
Neu in version 1.5:
Anstatt einer url kann ein Feld **data** angegeben werden. Mit Hilfe dieses Felds ist es möglich base64 kodierte Daten an ReporterPe zu übergeben. Die Daten müssen kompatibel zu dem Ergebnis eines create barcode(a, b, c, d, e) Funktionsaufrufs übergeben werden. Zum Beispiel:

https://www.lobster-world.com/online/_data/docs/46/de/BasicCreateBarcode.html?c1705860075748

Enthält das Feld **resize** den Wert true, ist mindestens eins der Felder **width** oder **height** verpflichtend. Wird nur eins der beiden Felder angegeben, so wird das Bild proportional skaliert. Sind sowohl width, als auch height angegeben, so wird das Bild anhand der gesetzten Werte skaliert und gegebenenfalls verzerrt dargestellt.

Es ist möglich Bilder in Listen einzubinden. Hierbei ist darauf zu achten, dass der Knoten der die Eigenschaften des Bilds festlegt dem Knoten der Liste untergeordnet ist.

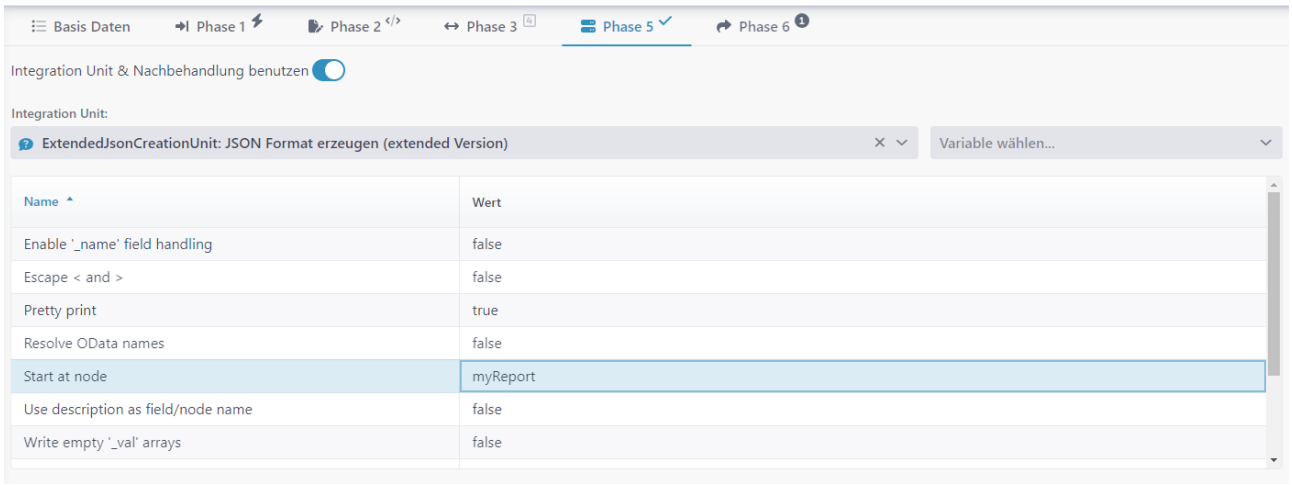
Der Ziel-Baum zum Befüllen eines Template kann beispielsweise so aussehen:



Nachdem Ihr Mapping abgeschlossen ist bedarf es nur noch zwei kleiner weiterer Schritte.

Phase 5 – IntegrationUnit

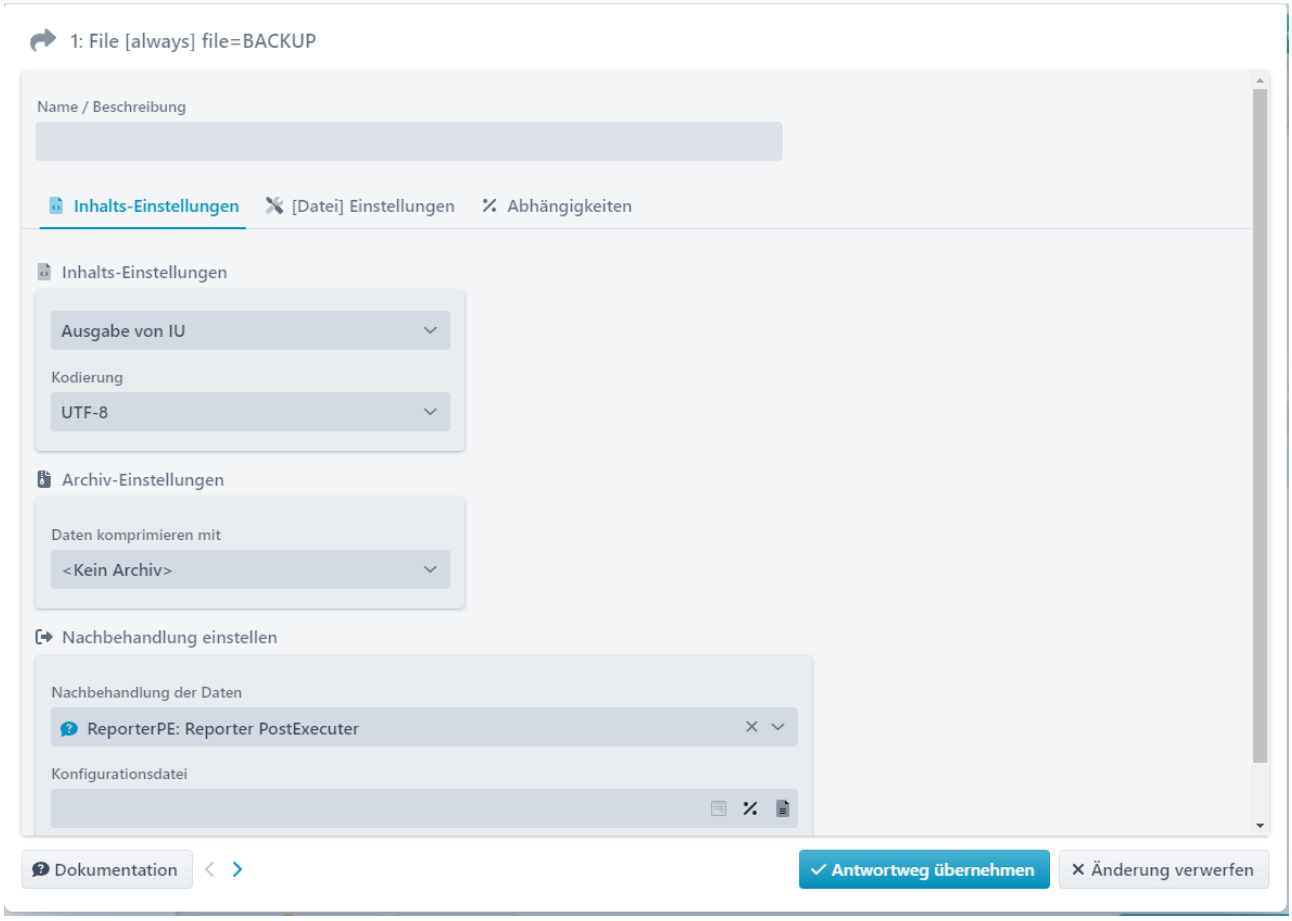
Wählen Sie hier die ExtendedJsonCreationUnit aus. Setzen Sie den Ihrem Mapping entsprechenden Wert für Start at node.



Phase 6 – Ausgangswege

Erzeugen Sie einen Ausgangsweg Ihrer Wahl. Für die Inhaltseinstellungen wählen Sie Ausgabe von IU. Kodierung: UTF-8.

Unter Nachbehandlung der Daten wählen Sie ReporterPE: Reporter PostExecuter.



Anlegen des ODT Templates

Für diesen Quickstart Guide gehen wir davon aus, dass Sie LibreOffice (https://de.libreoffice.org/) zur Erstellung Ihrer Templates nutzen. Wir nutzten für die Erstellung der Templates LibreOffice in Version: 6.3.4.2. Bestimmte Funktionen können in abweichenden Versionen an anderer Stelle zu finden sein.

Erste Schritte – Werte aus dem Profil an das Template übergeben

Die Übergabe an das Template erfolgt über Platzhalter (sog. Template-Variablen). Diese werden dem Template durch die Nutzung des $ - Symbols bekannt gegeben. Dem Symbol folgt der Feldname aus dem Profil in geschweiften Klammern, z.B. **${myField}**. Die Template-Variable kann an jeder Stelle im Template eingefügt werden. Eine Mehrfachnutzung der Variable ist möglich.

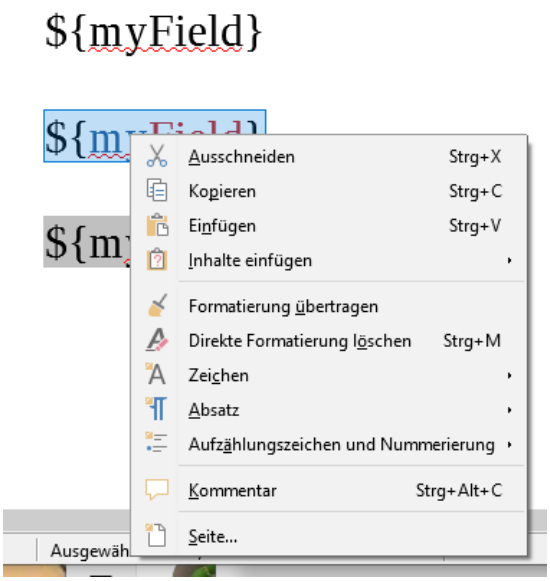
Um dem dynamisch eingefügten Text eine bestimmte Formatierung zu verpassen, formatieren Sie den gesamte Template-Variablen Ausdruck. Beispielsweise so: **${myField}**
Bitte beachten Sie, dass ReporterPE nur die Schriftarten im PDF unterstützt, die auf dem _data Server installiert sind und dem User, der den Lobster _data Service gestartet hat zur Verfügung stehen! Wird ein gewählter Font nicht gefunden fällt das System auf die Standard-Fonts zurück. Siehe: https://de.wikipedia.org/wiki/Portable_Document_Format

Das klingt doch eigentlich so, als könnte nichts schief gehen, oder? – Leider nein.

Für das ODT Dokument ist der Verweis auf die Variable ein Text, der sich auch formatieren lässt. Enthält der Verweis auf die Variable verschiedene Anweisungen zur Formatierung, zum Beispiel etwas schön buntes wie **${myField}** wird der Konvertierungsprozess im _data abgebrochen. Der Postexecuter erkennt im Falle mehrerer Formatierungsanweisungen die Angabe der Variablen nicht mehr.

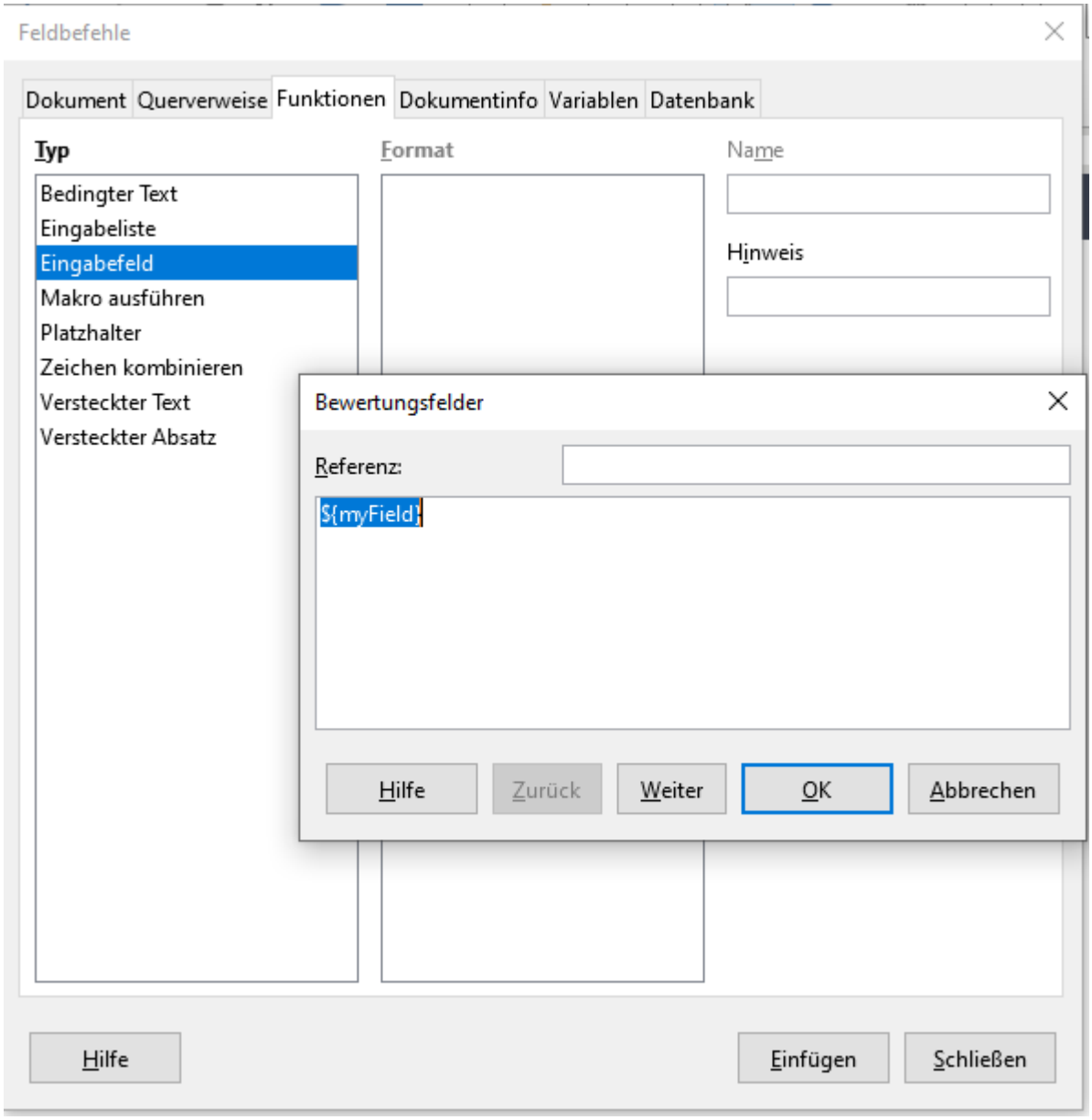
Leider lassen sich diese Formatierungsversuche im ODT nicht immer erkennen. Und meistens passiert dies, oftmals für Sie unsichtbar, wenn Sie an den Template-Variablen kleine Änderungen vornehmen.

Der erste Weg dies zu lösen ist im Template für die Variablen die direkte Formatierung zu löschen (markieren, dann Strg + M, oder über das per Rechtsklick erreichbare Kontextmenü. Nehmen Sie danach die gewünschte Formatierung auf den gesamten Variablenausdruck vor.



Auf Nummer sicher gehen Sie, wenn Sie ein wenig mehr Arbeit investieren. 😊 Statt den Variablennamen direkt in das Template zu tippen Erzeugen Sie ein Eingabefeld über

Einfügen-> Feldbefehl -> Weitere Feldbefehle oder die Tastenkombination **STRG+F2**

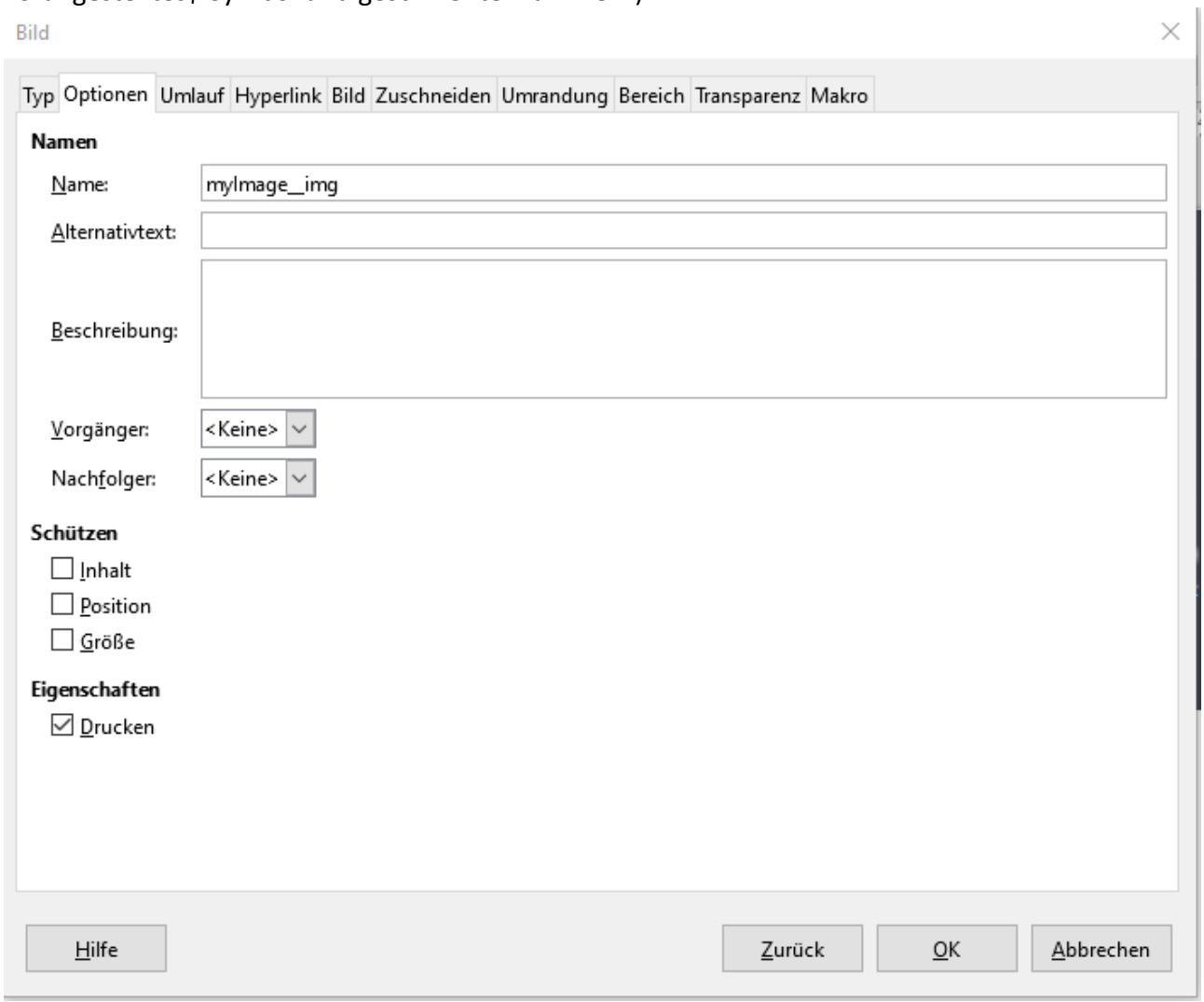


Wichtig: Alle in Ihrem Template angegeben Template-Variablen sind per Voreinstellung **Pflichtfelder**. Wird eine angegebene Template-Variable durch die im Mapping erzeugte JSON Struktur nicht befüllt, so bricht das Mapping mit einer Fehlermeldung ab.

Sie haben jedoch die Möglichkeit eine Template-Variable als **optional** zu markieren, indem sie ein Ausrufezeichen an den Variablennamen anhängen.
${myField!}

Bilder einfügen

Eine wichtige Aufgabe ist das einfügen von Bildern. Diese Aufgabe lösen Sie, indem Sie ein Platzhalterbild in das Template einfügen. Hierbei ist zu empfehlen, dass das von Ihnen gewählte Bild möglichst genau so dimensioniert ist, wie die durch den _data einzubindenden Bilder. Nachdem Sie das Platzhalter-Bild in das Template eingebunden haben öffnen Sie den Eigenschaftendialog (entweder durch Doppelklick auf das Bild, oder über Format->Bild->Eigenschaften. Hier finden Sie unter dem Reiter Optionen die Eigenschaft Name. Diese muss dem Knotennamen für die Bilddaten im Mapping entsprechen (ohne vorangestelltes$ Symbol und geschweifte Klammern).



Hierbei ist zu beachten, dass Bildobjekte im Template (leider) einen eindeutigen Namen benötigen. Es ist also nicht möglich dieselben Bilddaten an verschiedenen Stellen im Template zu nutzen. Wenn Sie denselben Namen zweimal vergeben ändert (zumindest LibreOffice) den Namen der zweiten Instanz. Und das stillschweigend.

Arbeiten mit Listen – Tabellen

Wenn Sie in Ihrem Mapping ein JSON-Array erzeugen, kann dieses als Liste im Template genutzt werden, um damit Tabellen zu befüllen.

Legen Sie eine Tabelle an. Ab der Zeile, in die Sie Ihre Liste schreiben möchten fügen Sie als Templatevariable den Wert ${Knotenname.Feldname} ein. Beispielsweise ${myList.myField} in jede Zelle ein, die im Template befüllt werden soll.

Statische Überschrift 1	Statische Überschrift 2
${myList.myField1}	${myList.myField2}

Wenn Sie Bilder in den Zellen nutzen möchten, so binden Sie ein Platzhalter-Bild ein und vergeben als Namen Knotenname.Feldname, z.B. myList.myImage__img. Hierbei gelten die im Bereich Mapping beschriebenen Regeln.

Arbeiten mit Listen – ohne Tabellen

Sollten Sie eine Liste direkt im Text benötigen (z.B. für Aufzählungen), müssen Sie dem Template bekannt geben, dass an einer bestimmten Stelle eine Liste genutzt werden soll. Hierbei benötigt der PostExecuter eine Deklaration für den Beginn und das Ende einer genutzten Liste. Die Deklaration startet mit dem Ausdruck **[#list** gefolgt von einem **Knotennamen** aus Ihrem Mapping und wird mit dem Ausdruck **as internerListenname]** abgeschlossen.Beispiel: **[#list myList as list1]**

Innerhalb der List werden Felder aus Ihrem Mapping durch den Variablennamen **${internerListenname.Feldname}** referenziert. Beispiel: **${list1.myField}**

Das Ende der Listen-Nutzung muss dem Template durch den Ausdruck **[#list]** bekannt gegeben werden.

Ein komplettes Beispiel kann dann so aussehen:

[#list myList as list1]
${list1.myField}
[#list]

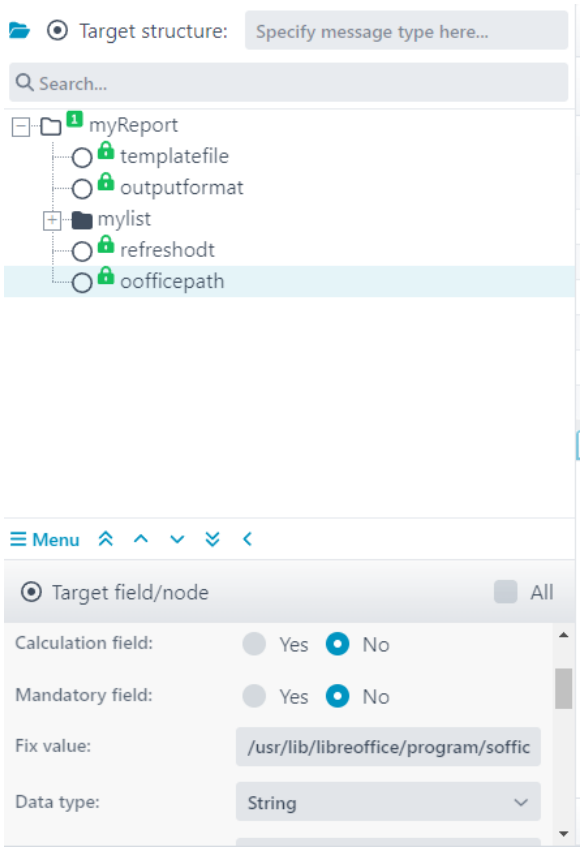
Dynamisches Auffrischen des erzeugten ODT Dokuments

Manchmal ist es notwendig das gefüllte ODT Template vor der PDF Erzeugung inhaltlich aufzufrischen. Ein gutes Beispiel hierfür ist das Erzeugen von Inhaltsverzeichnissen. ReporterPE erlaubt diesen Prozess zu automatisieren.

Hierfür ist es notwendig, dass eine LibreOffice Instanz serverseitig installiert ist. Der Pfad zur Installation muss bekannt sein und im JSON übergeben werden.

Um das Serverseitige Auffrischen des gefüllten ODT-Template auszulösen, erweitern das Mapping um 2 Felder:

refreshodt	mit dem Inhalt true
oofficepath	mit dem Pfad zu dem installierten LibreOffice auf Ihrem Server. (beispielsweise: /usr/lib/libreoffice/program/soffice.bin)



Dorfstraße 30, 24790 Schülldorf
https://www.derbrill.de/ReporterPE