



ReporterPE – Quickstart Guide

(Version 2.0 vom 20.10.2024)

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 (2.0) von ReporterPE wurde in Lobster _data 4.6.9 bis 4.6.11 getestet. Bitte beachten Sie, dass wir den reibungslosen Betrieb des PostExecuters in anderen Versionen des _data oder bei anderen gegebenenfalls installierten Erweiterungen in /extlib nicht garantieren können.

Parallelbetrieb mit Jasper Reports

Dies gilt insbesondere für den Parallelbetrieb mit älteren Jasper Reports Installationen. Ältere Installationen nutzen die iText 1.2.7.jar. Diese Library ist nicht mit ReporterPE kompatibel. ReporterPE ist abhängig von OpenPDF (einem Fork von iText), der inzwischen auch bei der Nutzung von JasperReports eingesetzt wird. Sollte in Ihrem /extlib Verzeichnis eine iText_1.2.7.jar liegen empfehlen wir Ihnen ein Update der Jasper Libraries.

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. Entfernen Sie alle älteren Versionen von ReporterPE.

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.



Um das beste Resultat bei der Erzeugung von PDF Dateien zu erreichen, ist empfohlen zusätzlich eine aktuelle LibreOffice Version auf Ihrem Server zu installieren. Diese kann dann von ReporterPE genutzt werden, um die PDF Erzeugung anzustoßen und den vollen Funktionsumfang von LibreOffice zu nutzen. Unicode Text (z.B. arabisch) und Textrahmen werden von ReporterPE nur unterstützt, wenn LibreOffice statt der internen PDF Engine zur Erzeugung der PDFs genutzt wird. Nach der LibreOffice Installation ist empfohlen Ihr _data System neu zu starten. Wenn Sie ReporterPE bereits benutzt haben und dann LibreOffice installieren ohne den _data neu zu starten, kann es ansonsten bei der ersten Nutzung von LibreOffice als Render-Engine zu Funktionsstörungen kommen.

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.

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 – die Demo Profile

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

Hierbei werden Profile zur Verfügung gestellt, die zum Teil zwingend die Installation von LibreOffice auf dem Server erfordern. Diese sind in einem Extra Ordner zu finden.

Mapping Einstellungen

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

Einfache Variablen werden auf Wurzelebene des Zielbaums angelegt (genauer gesagt: 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. Gegeben 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 Nennung erfolgt unter Angabe des gewünschten Protokolls (File:/ http:// oder https://) gefolgt von der URL. Z.B. File:./conf/ReporterPe/examples/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`

`<i>italic</i>`

`<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 `
` ersetzt werden müssen. Hierbei ist essentiell das tag wohlgeformt (im Sinne von XML) 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 einen 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

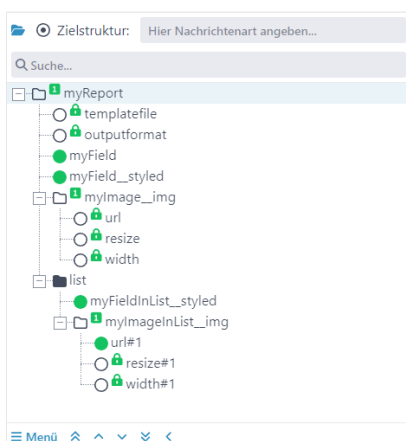
Anstatt einer url kann ein Feld **data** angegeben werden. Mit Hilfe dieses Felds ist es möglich, base64 kodierte Daten an ReporterPe zu übergeben. Die Daten müssen kompatibel zu dem Ergebnis eines create barcode(a, b, c, d, e) Funktionsaufrufs übergeben werden. Zum Beispiel:

<https://www.lobster-world.com/online/ data/docs/46/de/BasicCreateBarcode.html>

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:

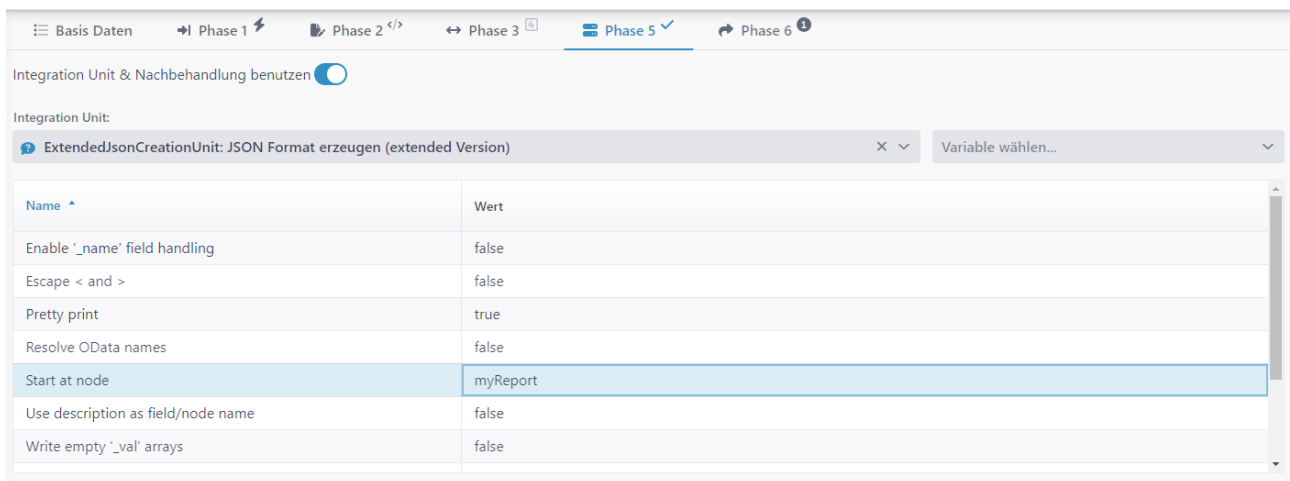


Nachdem Ihr Mapping abgeschlossen ist, bedarf es nur noch zweier weiterer Schritte.



Phase 5 – IntegrationUnit

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



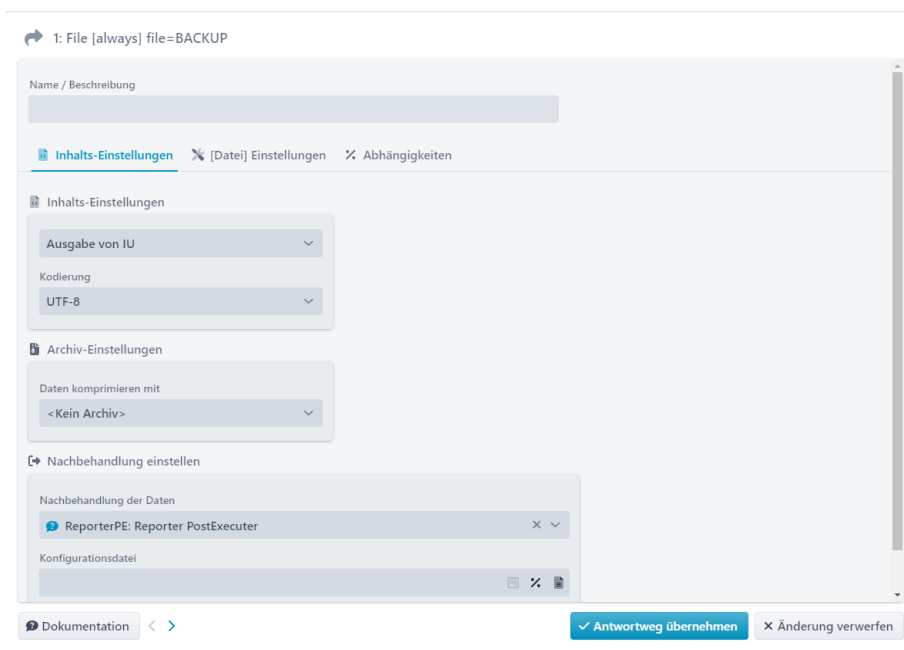
The screenshot shows the 'Integration Unit & Nachbehandlung benutzen' toggle set to 'on'. Below it, the 'Integration Unit:' dropdown is set to 'ExtendedJsonCreationUnit: JSON Format erzeugen (extended Version)'. A table lists various configuration options:

Name	Wert
Enable '_name' field handling	false
Escape < and >	false
Pretty print	true
Resolve OData names	false
Start at node	myReport
Use description as field/node name	false
Write empty '_val' arrays	false

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**.



The screenshot shows the configuration for a file output path. The 'Inhalts-Einstellungen' (Content Settings) section is active, showing 'Ausgabe von IU' (Output of IU) selected for 'Ausgabe von IU' and 'UTF-8' for 'Kodierung' (Encoding). The 'Archiv-Einstellungen' (Archive Settings) section shows '<Kein Archiv>' (No Archive) selected for 'Daten komprimieren mit' (Compress data with). The 'Nachbehandlung einstellen' (Post-processing) section shows 'ReporterPE: Reporter PostExecuter' selected for 'Nachbehandlung der Daten' (Post-processing of data). The 'Konfigurationsdatei' (Configuration file) field is empty. At the bottom, there are buttons for 'Antwortweg übernehmen' (Take over answer path) and 'Änderung verwerfen' (Reject change).



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: 7.6.7.2. 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 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 **gesamten** Template-Variablen Ausdruck.

Beispielsweise so: **\$myField**

Der aus dem Mapping übergebene Literal wird dann fett und in blau angedruckt.

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

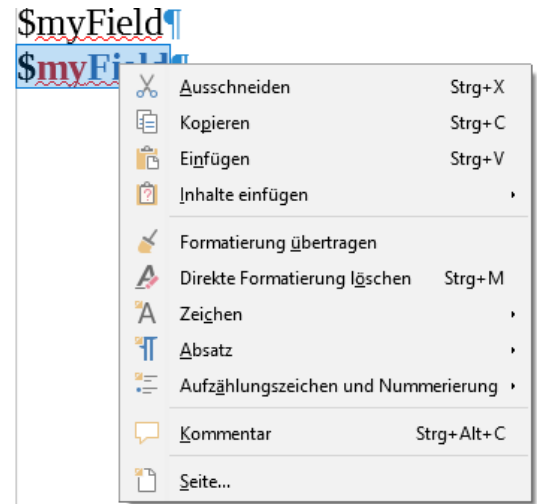
Das klingt doch eigentlich so, als könnte nichts schief gehen, oder? – Leider nein.

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 die Variable nicht als solche erkannt, sondern als \$myfield angedruckt. 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.

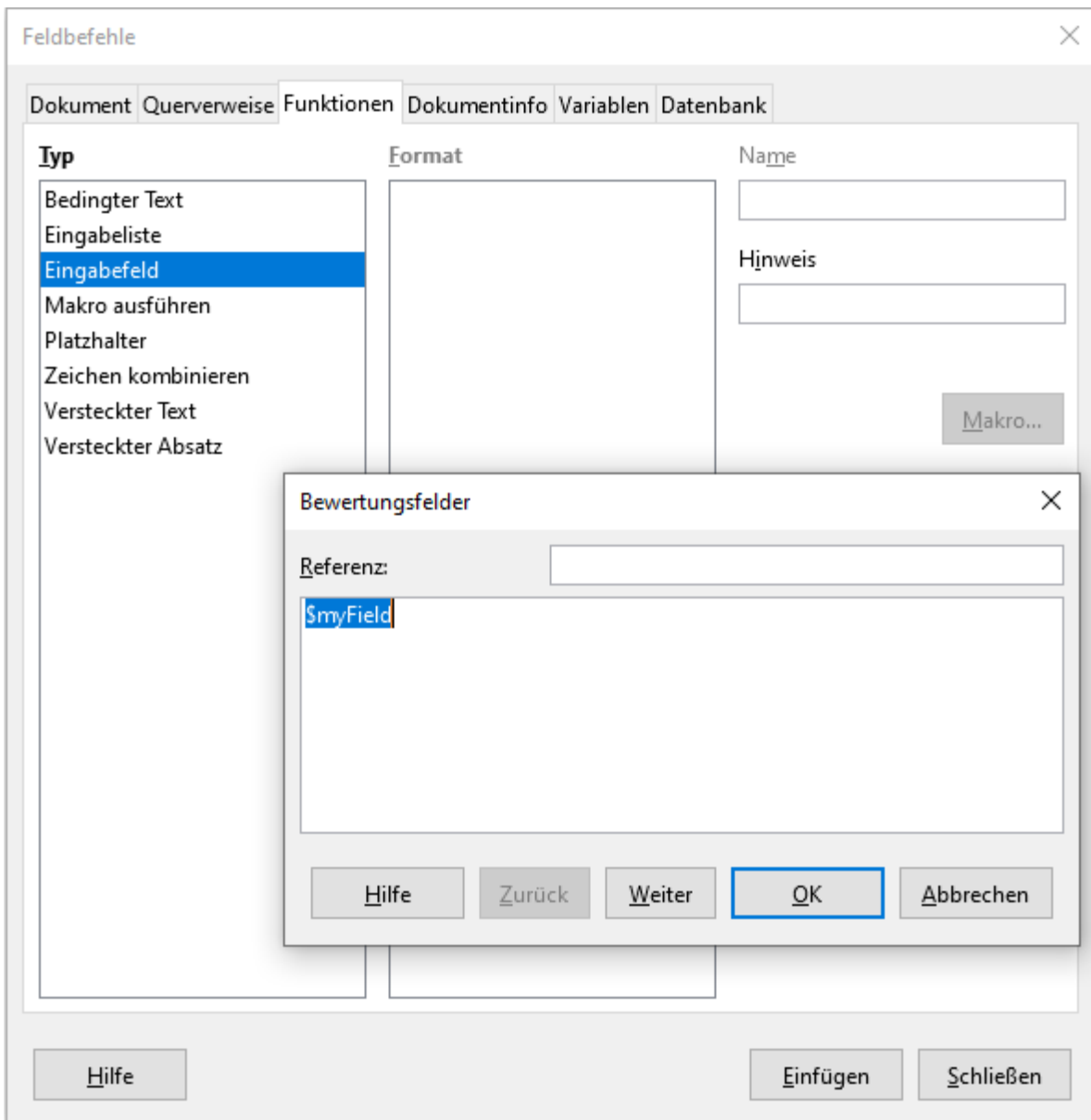




Nutzung von Feldbefehlen in LibreOffice

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**





Templatevariablen als optional deklarieren

Wichtig: Alle in Ihrem Template angegebenen Template-Variablen sind per Voreinstellung **Pflichtfelder**. Wird eine angegebene Template-Variable durch die im Mapping erzeugte JSON Struktur nicht befüllt, so wird der Variablenname angedruckt.

Sie haben jedoch die Möglichkeit, eine Template-Variable als **optional** zu markieren, indem sie ein Ausrufezeichen hinter das \$ Symbol schreiben.

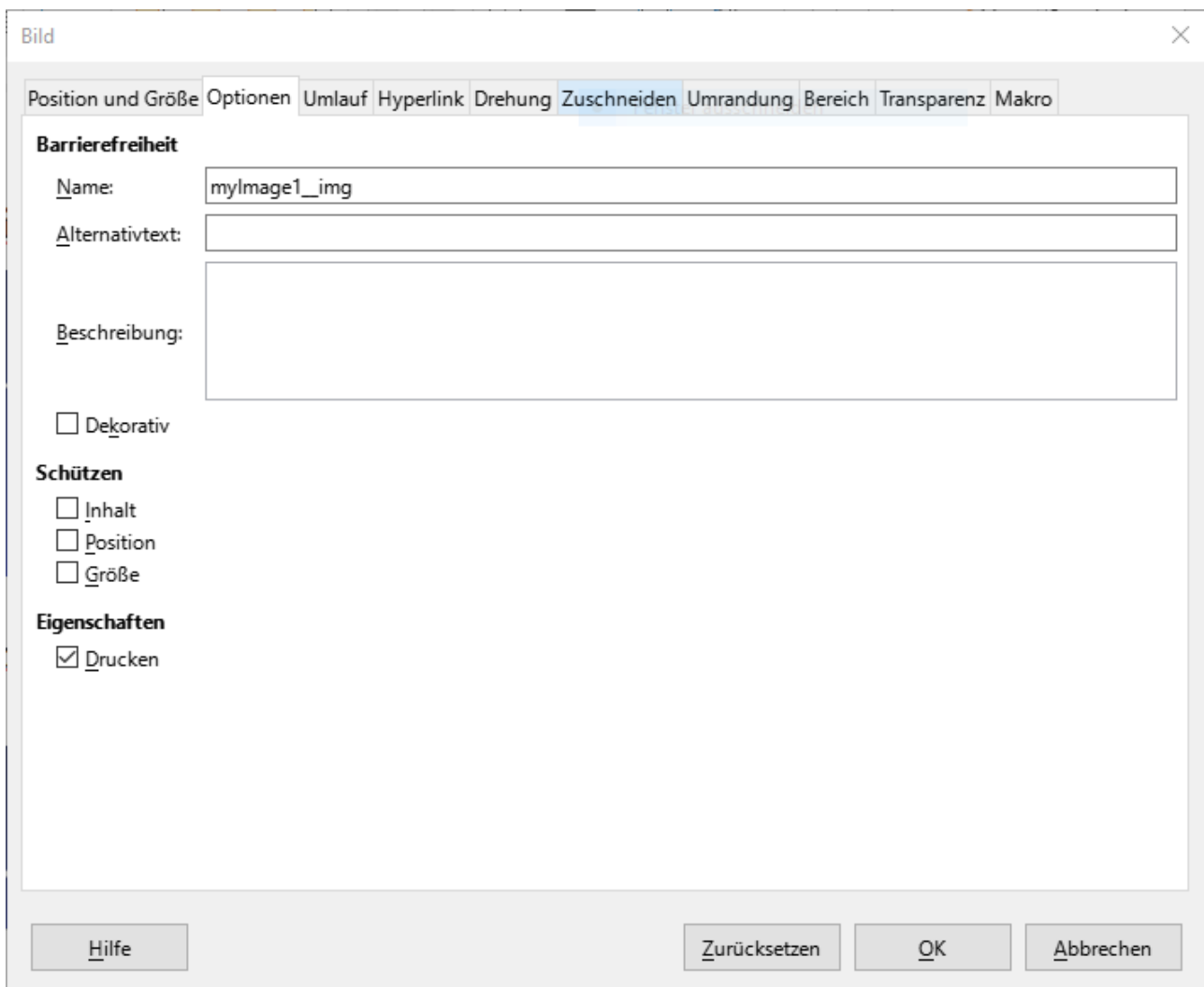
\$!myField

In diesem Fall wird der Variablenname nicht angedruckt, wenn kein Wert für das Feld myField im Mapping übergeben wird.



Bilder einfügen

Eine wichtige Aufgabe ist das Einfügen von Bildern. Diese Aufgabe lösen Sie, indem Sie ein Bild als Platzhalter 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).



Bild

Position und Größe Optionen Umlauf Hyperlink Drehung **Zuschneiden** Umrandung Bereich Transparenz Makro

Barrierefreiheit

Name: myImage1_img

Alternativtext:

Beschreibung:

☐ Dekorativ

Schützen

☐ Inhalt

☐ Position

☐ Größe

Eigenschaften

☒ Drucken

Hilfe Zurücksetzen OK Abbrechen



Hierbei ist zu beachten, dass Bildobjekte im Template (leider) einen eindeutigen Namen benötigen. Es ist also nicht möglich, identische 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
<code>\$myList.myField1</code>	<code>\$myList.myField2</code>

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.

#foreach(\$indexVariable in \$myList)

• `$indexVariable.feldName`

#end

Die Deklaration startet mit dem Ausdruck **#foreach** gefolgt von einem Ausdruck in Klammern, der dem PostExecuter Anweisungen gibt, wie die Liste abgearbeitet werden muss. Dieser Ausdruck besteht aus einer **Indexvariablen**, die von Ihnen vergeben wird, gefolgt von der Angabe in welcher **Liste** gearbeitet werden soll. Hierbei entspricht der Name der Liste dem Namen Ihres Knotens im Mapping, dem ein `$` Zeichen vorangestellt wird.



Innerhalb der Liste werden Felder aus Ihrem Mapping durch den Variablennamen **\$indexVariable.feldName** referenziert. Beispiel: **\$myIndex.myField**

Das Ende der Listen-Nutzung muss dem Template durch den Ausdruck **#end** bekanntgegeben werden.

Ein komplettes Beispiel kann dann so aussehen:

```
#foreach($myIndex in $myList)
$myIndex.myField
#end
```

Listen in Listen (ohne Tabellen)

Prinzipiell ist es möglich, Listen ineinander zu verschachteln.

```
#foreach($myIndex in $myList)
$myIndex.myField
#foreach($myIndex2 in $myIndex.myList2)
$myIndex2.myField2
#end
#end
```

Hierbei erfolgt die Abarbeitung der Schleifen genau so, wie sie auch im Mapping erfolgt. Dies ist in einem Kontext außerhalb von Tabellen auch recht einfach umzusetzen.

Listen in Listen (in Tabellen)

#hierLauernDrachen

Deutlich komplizierter wird es, wenn Sie verschachtelte Listen in Tabellen nutzen wollen. Tabellen sind in Ihrer Natur zweidimensionale Objekte. Hier weitere Dimensionen einzuführen ist nicht leicht und widerspricht eigentlich dem, was wir mit ReporterPE erreichen wollen – eine möglichst unkomplizierte Herangehensweise an PDF-Erzeugung.

Eindeutige Empfehlung unsererseits ist, die inneren Schleifen bereits im Mapping aufzulösen und die entsprechenden Werte als Literals in einer Variablen zu übergeben!

Dies gesagt: Es geht trotzdem...



Die Nutzung der Direktive `@before-row` und `@after-row` erlaubt das Auflösen von Listeneinträgen innerhalb einer Tabelle. Hierbei muss dann JEDE Liste, die in der Tabelle genutzt werden soll, in einer **EIGENEN** Tabelle deklariert werden.

Das heißt, wenn Sie verschachtelte Listen in einer Tabelle nutzen wollen, ist es notwendig, entsprechend Ihrer Datenstruktur auch im Template Tabellen ineinander zu verschachteln. Klingt kompliziert? Ist es auch...

Die nachfolgende Tabelle bildet zwei Listen ab. Hierbei hat die Haupttabelle einen schwarzen Rahmen, die eingebundene Tabelle einen blauen.

@before-row#foreach(\$l1 in \$loop1)\$l1.value1	\$l1.value2
\$l1.value3	<div>@before-row#foreach(\$l2 in \$l1.loop2)\$l2.value@after-row#end</div> @after-row#end

Um nun Zellen mit Werten aus den verschachtelten Listen zu füllen, ist es notwendig, in jeder der einzelnen Tabellen die Direktive `@before-row` und am Ende die Direktive `@after-row` einzufügen. Der Direktive folgt die Deklaration der Liste über `#foreach`, genauso wie Sie es außerhalb einer Tabelle tun würden.



Erzwingen von Seitenumbrüchen

Wenn Sie in Ihrem Template aus dem Mapping heraus einen Seitenumbruch erzwingen wollen, so ist dies über ein gestyltes Feld möglich.

Erzeugen Sie in Ihrem Mapping ein Feld, welches die Magic Extension `__styled` im Namen trägt. Beispielsweise `pgbrk__styled`. Als Fixwert übergeben Sie dem Feld folgenden Inhalt:

```
<p style="page-break-before:always;"></p>
```

Nun wird jedesmal, wenn Sie in Ihrem Template die Templatevariable `$pgbrk__styled` einfügen an dieser Stelle ein Seitenumbruch forciert.

Scripting innerhalb des Templates

Seit Version 2.0 werden grundlegende Skript Befehle innerhalb des Templates unterstützt. Prinzipiell ist bereits das Nutzen einer Templatevariablen ein Skript-Befehl. Weitere Optionen:

Wenn-Dann-Sonst Konstrukte werden über die Befehle `#if` `#else` und `#end` geregelt. Die zu prüfende Bedingung wird dem `#if` Befehl in runden Klammern übergeben. Eine Prüfung ob dem Template die Variable `oofficepath` übergeben wurde (inklusive der Prüfung ob diese leer ist) kann beispielsweise so umgesetzt werden:

```
#if( "$!oofficepath"=="")  
oofficepath not set.  
#else oofficepath: $oofficepath  
#end
```

Zur Analyse der Datenlage bei Listen stehen Ihnen folgende Ausdrücke zur Verfügung:
`$myList.size()` liefert Ihnen die Anzahl der Listeneinträge in der Liste `$myList` zurück.
`$myList.isEmpty()` liefert Ihnen `true` oder `false` zurück, je nach dem ob die Liste mit mindestens einem Eintrag gefüllt ist, oder nicht.

Innerhalb einer `#foreach` Schleife können Sie folgende Werte prüfen:

`$foreach.hasNext` liefert Ihnen zurück, ob es nach dem gerade bearbeiteten Listeneintrag einen weiteren gibt.

`$foreach.index` liefert Ihnen die aktuelle Iteration des Listendurchlaufs zurück, gezählt von 0

`$foreach.counter` liefert Ihnen die aktuelle Iteration des Listendurchlaufs zurück, gezählt von 1



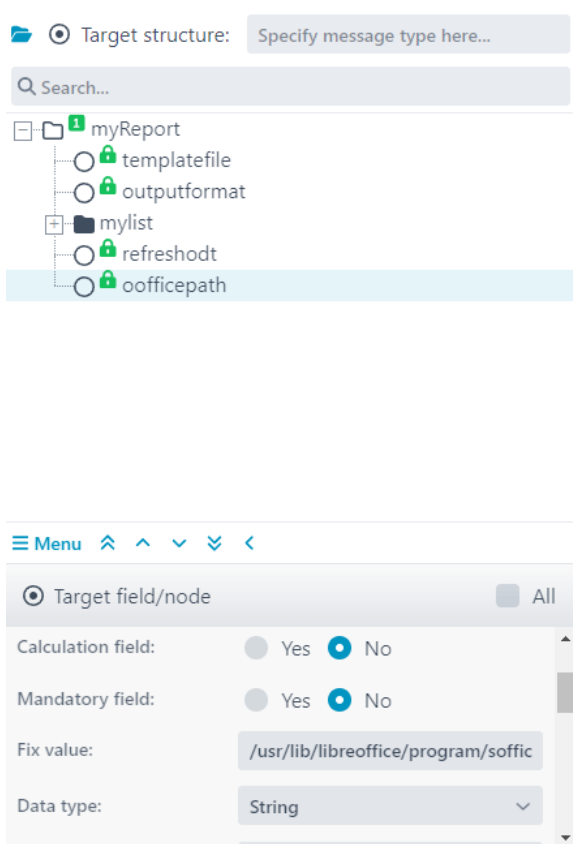
Dynamisches Auffrischen des erzeugten ODT Dokuments

Manchmal ist es notwendig, das gefüllte ODT Template vor der Erzeugung der Zielfelddatei 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 Sie 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)



The screenshot shows the ReporterPE configuration interface. At the top, there is a 'Target structure' section with a dropdown menu set to 'Specify message type here...'. Below this is a search bar labeled 'Search...'. The main area displays a tree structure for 'myReport' with the following nodes: 'templatefile', 'outputformat', 'mylist', 'refreshodt', and 'oofficepath'. The 'refreshodt' and 'oofficepath' nodes are highlighted in blue. Below the tree is a 'Menu' section with a 'Target field/node' dropdown set to 'All'. The configuration options are as follows:

Target field/node	Value
Calculation field:	<input type="radio"/> Yes <input checked="" type="radio"/> No
Mandatory field:	<input type="radio"/> Yes <input checked="" type="radio"/> No
Fix value:	/usr/lib/libreoffice/program/soffice
Data type:	String



Renderengines im Vergleich

ReporterPE erlaubt Ihnen zwischen der internen PDF Erzeugung und der PDF Erzeugung über LibreOffice zu wählen. Die qualitativ hochwertigeren PDFs erhalten Sie, wenn Sie LibreOffice die Ausgabe übernehmen lassen. Die interne PDF Erzeugung sollten Sie dementsprechend nur bei der Verarbeitung von Massendaten, also vielen PDFs in vielen Jobs, nutzen.

Die interne PDF Erzeugung setzt voraus, dass Sie sich bei der Erstellung der Templates viel Mühe geben, wenn Sie mit dynamischen Bildern arbeiten. Wenn die Platzhalterbilder in den Abmessungen nicht genau der Größe der eigentlich zu nutzenden Bilddaten entsprechen, so kann es zu Überlagerungen mit folgenden Absätzen kommen. Auch kann es bei tief verschachtelten Listen in Tabellen zu Darstellungsfehlern der Tabellenrahmen kommen. Weiterhin werden bei der Nutzung der internen PDF-Erzeugung nicht alle Features von LibreOffice unterstützt.

Lassen Sie Ihre PDF Dateien durch LibreOffice erzeugen, so ist die Ausführung der Jobs etwas langsamer.

Feature	Interne PDF-Erzeugung	LibreOffice PDF Erzeugung
Tabstops	Nein	Ja
Unicode Fonts	Nein	Ja
Frei positionierbare Textrahmen	Nein	Ja



Wir wünschen Ihnen viel Freude, wenn Sie ReporterPE ausprobieren. Fragen beantworten wir Ihnen gerne unter helpdesk@derbrill.de

Ihr derbrill Team.
<https://www.derbrill.de>