Squeeze Invoice

Diese Lösung ist für den Fall gedacht, wenn Squeeze eine bestehende Beleganalyse ablöst aber die bestehende Documents Invoice-Lösung beibehalten werden soll. In Documents soll die Validierung stattfinden.

Installation

Dieses Kapitel beschreibt die Installation für den Documents-Part für Squeeze Invoice. Die Installation kann abhängig von der aktuellen Kundenlösung sehr unterschiedlich ausfallen.

Einleitung

Diese Dokumentation beschreibt die Integration der Validierung inklusive des Einsatzes des Squeeze-Viewers in Documents.

Die Basis-Installation besteht aus diversen unverschlüsselten Portal-Skripten und dem Mappentypen "SqueezeInvoice" sowie externen Documents-Resourcen. Die Beleglesung findet in Squeeze statt und pro Rechnung wird eine Mappe vom Mappentypen "SqueezeInvoice" erstellt.

Die Umstellung sollte zunächst auf einem Testsytsem erfolgen. Documents sollte vorab auf eine stabile aktuelle Version aktualisiert werden.

Voraussetzungen

In jedem Fall wird eine Lizenz für einen zusätzlichen Mappentypen benötigt. Die Scripting-Lizenz müsste bei einem bestehenden Rechnungs-Workflow vorhanden sein. Es wird eine "named"-Lizenz für den Import-Benutzer vorausgesetzt. Da die Installation eine bestehende Konfiguration ablöst sollte die Lizenz automatisch vorhanden sein. Falls für die evtl. neuen Anwender noch keine Documents-Lizenz zur Verfügung steht, müssen

Wenn die Squeeze-Konfiguration über Documents gesteuert werden soll wird eine Gadget-Lizenz für die Konfigurations-Ordner benötigt. Die Konfiguration kann optional komplett in Squeeze erfolgen. Dadurch ist die Gadget-Lizenz keine zwingende Voraussetzung.

Die Lösung benötigt keinen separaten Workflow! Die Steuerung erfolgt über öffentliche Filter-Ordner und Portal-Skripte. Eine Implementation beim Kunden erfordert daher Kenntnisse im Portal-Skripting und Kenntnisse im Umgang mit den externen Ressourcen.

Aufbau

Die Auslieferung beinhaltet den Documents-Mappentypen "SqueezeInvoice" sowie eine Reihe von unverschlüsselten Portal-Skripten, öffentlichen Ordnern und externe Ressourcen mit einigen hilfreichen Konfigurations-Beispielen.

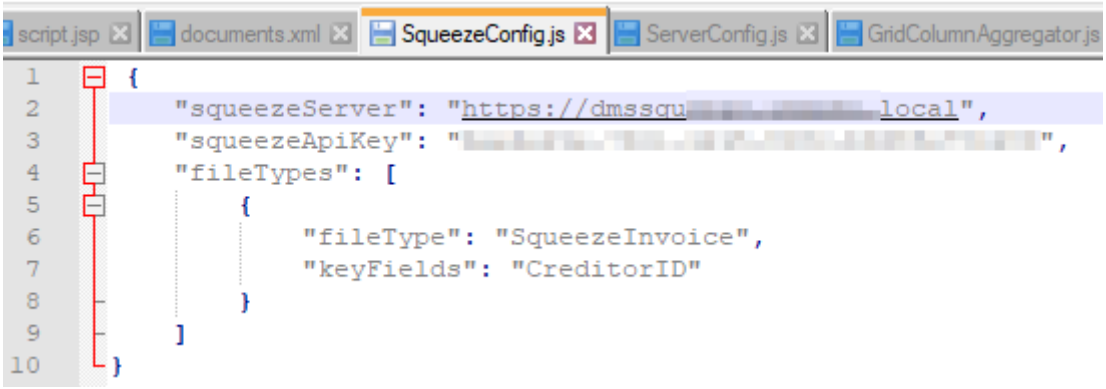
Nach der Beleganalyse werden die Mappen vom Typen "SqueezeInvoice" erzeugt. Diese werden den Anwendern über die öffentlichen Filter-Ordner angezeigt. Nach der Validierung wird ein Mappentyp-Wechsel auf den bestehenden Mappentypen ausgeführt. Die Feldwerte müssen entsprechend zugeordnet werden.

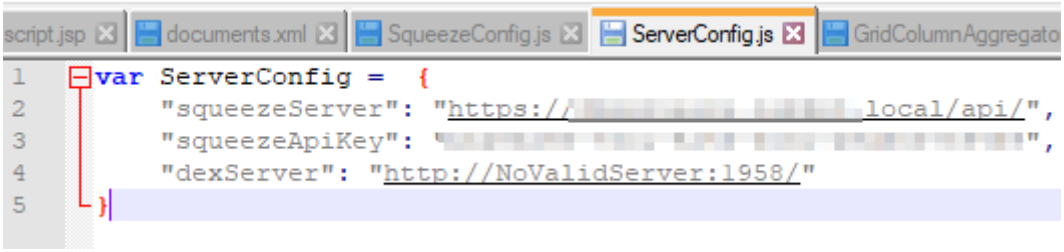
Eine genauere Beschreibung der Abläufe wird auf den folgenden Seiten beschrieben.

Anbindung Squeeze

Für die Anbindung von Squeeze an Documents existiert bereits die Dokumentation "Invoice Squeeze Anbindung". Aus diesem Grund wird die Anbindung in dieser Dokumentation nur rudimentär erklärt.

Bei vorhandener Gadget-Lizenz kann die Anbindung über die WEB-Konfiguration erfolgen. Andernfalls muss die Anbindung manuell über die Dateien "SqueezeConfig.js" und "ServerConfig.js" erfolgen.





XML-Importe

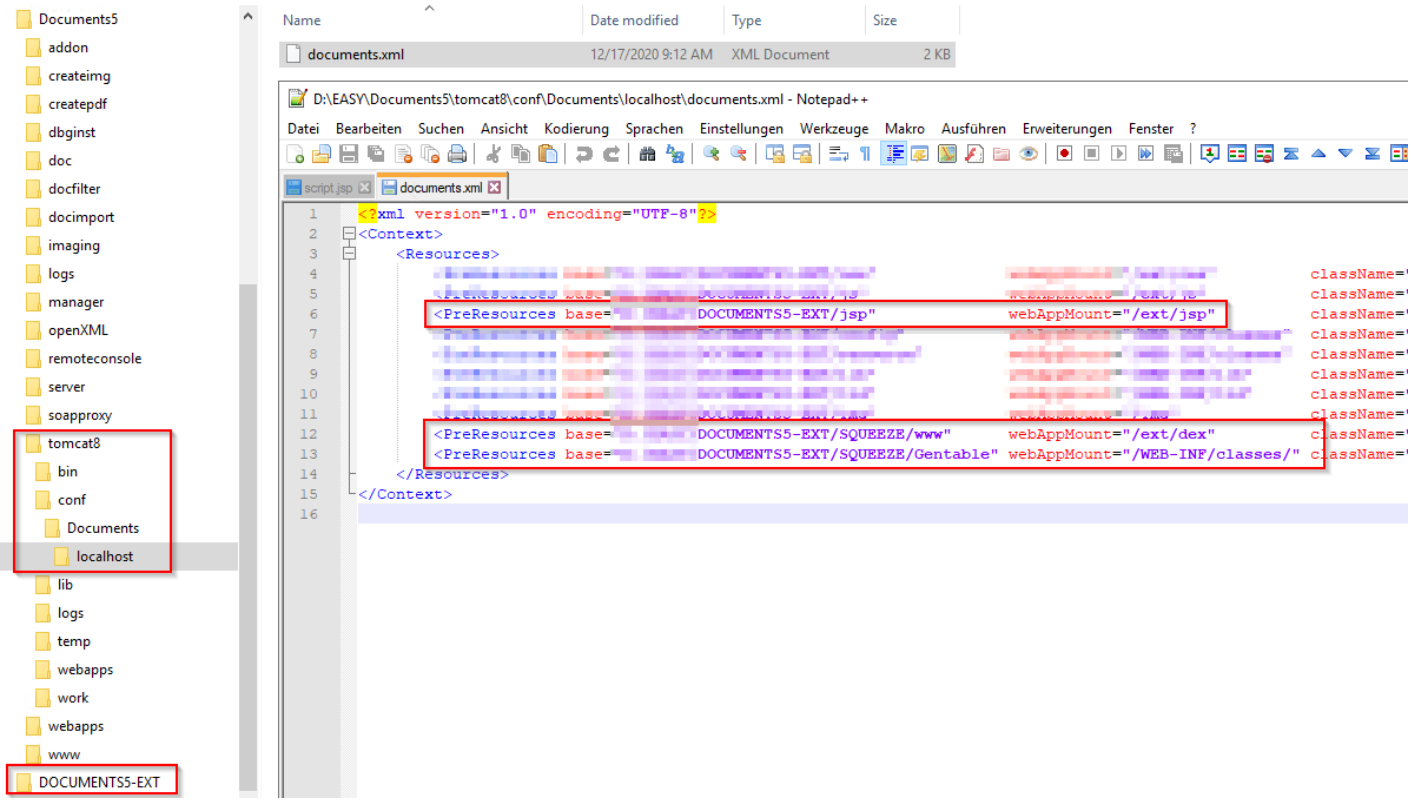
Der Mappentyp "SqueezeInvoice", sowie die öffentlichen Ordner als auch die Portal-Skripte können via XML-Import hinzugefügt werden. Die importierten Dateien können abgesehen von den verschlüsselten Skripten beliebig angepasst werden.

Das Installationspaket enthält nur einer einzigen XML-Datei. Diese XML beinhaltet den Mappentyp; alle Skripte und die öffentlichen Ordner sowie einen Nummernkreis "Invoice".

Externe Ressourcen

Externe Ressourcen werden in Documents verwendet, um möglichst alle projektspezifischen Anpassungen an definierten Orten zu verwalten, damit zum Beispiel Documents-Updates möglichst unabhängig von den Anpassungen ausgeführt werden können. Bei einer bestehenden Rechnungseingangs-Lösung ist davon auszugehen, dass bereits externe Ressourcen existieren. Die Anpassungen müssen somit in die bestehende Lösung integriert werden. Aus diesem Grund ist es schwierig eine einheitliche Vorgabe für die Umsetzung zu geben.

Der Einstiegspunkt ist die Datei "**documents.xml**" unter "**...\Documents5\tomcat8\conf\Documents\localhost**". In der Datei werden die Ablageorte für die externen Ressourcen definiert und diese Datei dürfte bereits existieren. Falls die Datei noch nicht existiert muss Sie erstellt werden. Der folgende Screenshot zeigt eine bereits angepasste Datei.



In dem Beispiel befinden sich alle externen Ressourcen im Ordner "**DOCUMENTS5-EXT**". Die Datei sollte bereits einen Pfad zur Datei "**script.jsp**" enthalten. Im Screenshot ist es der oberste markierte Pfad.

Die unteren beiden Einträge wurden hinzugefügt und müssen auf den mitgelieferten Squeeze-Ordner verweisen.

script.jsp

Vermutlich existiert bereits eine Datei "**script.jsp**", welche wiederum diverse js-Dateien importiert. Diese Datei muss um mindestens 2 Einträge erweitert werden. Zum einen muss die "**SqueezeFunctions.js**" importiert werden, um Zugriff auf spezifische Funktionen zu erhalten. Zum anderen sollte die "**GridColumnAggregator.js**" importiert werden, um im Gentable eine Summenzeile zu erzeugen.

Die folgenden Funktionen können entweder in eine eigene JSP-Dateien ausgelagert werden oder können in eine bereits verwendete Datei hinzu kopiert werden.

Mappentyp im Bearbeitungs-Modus öffnen

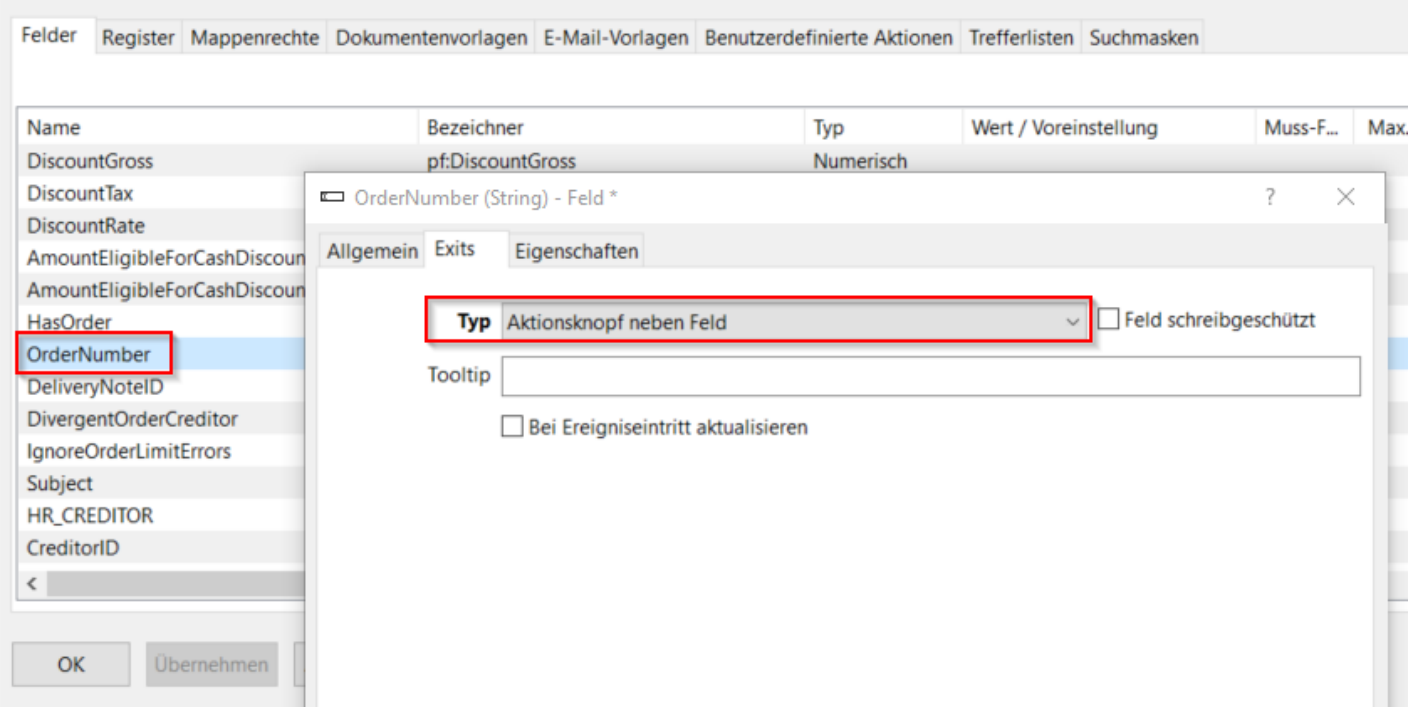
Über den Workflow kann eine Aktion derart konfiguriert werden, dass sich die Mappe immer direkt im Bearbeitungs-Modus öffnet. Bei der Validierung wird dieses Verhalten in der Regel gewünscht. Um das Verhalten nachzuahmen kann die folgende Funktion eingebunden werden.

```
/** Öffnet den Beleg immer im Bearbeitungs-Modus
 **/
(function(){
    var autoEditEnabled = true;

documents.sdk.exitRegistry.registerFileExitCallback("SqueezeInvoice","File.afterFileOpen",
function(documentsContext, options){
        if(autoEditEnabled){
            documentsContext.getFileContext().startFileEditMode();
```

```
            autoEditEnabled = false;
        }
    });

documents.sdk.exitRegistry.registerFileExitCallback("SqueezeInvoice","File.afterFileEditCommit,
function(documentsContext, options){
        autoEditEnabled = true;
    });
})();
```

Stammdaten-Auswahl via Pop-Up

Damit keine umständlichen Datenbank-Abfragen konfiguriert werden müssen können die Daten direkt aus den Squeeze-Stammdaten-Tabellen gezogen werden. Der Aufruf "**documents.sdk.exitRegistry.registerFileFieldExitCallback()**" benötigt den technischen Namen des Mappentypen sowie den technischen Feldnamen. Am Mappentyp-Feld muss auf dem Register "**Exits**" der Typ "**Aktionsknopf neben dem Feld**" gesetzt werden.



Bei Positionsfeldern lautet der Aufruf "**documents.sdk.gentable.gentableRegistry.registerCallback()**". Hier muss als erster Parameter der technische Name der Gentable-Definition angegeben werden und als zweiter Parameter muss der Funktionsname am Gentable-Button eingetragen werden.

In den Funktionen wird ein SqueezePopup-Objekt erzeugt. Die Funktion "**addMasterdataID()**" benötigt hierbei nicht den Namen der Tabelle, sondern die eindeutige ID der Tabelle. Für das Pop-

Up kann eine Größe definiert werden und es kann eine Überschrift angegeben werden. Es muss zwingend angegeben werden ob es sich um ein Pop-Up für ein Kopf- oder Positionsfeld handelt. Bei der "**set()**" Funktion wird zuerst der technische Feldname angegeben, dann der Name der Tabellen-Spalte und beim dritten Parameter kann angegeben werden ob die Spalte im Pop-Up angezeigt werden soll oder nicht. Über "**showOnly()**" können Spalten im Pop-Up angezeigt werden, welche später nicht gesetzt werden und über "**processData**" kann nach dem Setzen der Werte weiterer Code ausgeführt werden. In dem Beispiel wird aus einer weiteren Squeeze-Stammdaten-Tabelle die Zahlungsbedingung zum Kreditor ermittelt. Der Wert wird über ein Portal-Script ermittelt.

Die "**open()**" Funktion öffnet das Pop-Up.

```
/** Kreditor-Auswahl **/
documents.sdk.exitRegistry.registerFileFieldExitCallback("SqueezeInvoice", "CreditorID",
function(d5Context, options){
	var popup = new SqueezePopup(d5Context, options);
	popup.width = 1500;
	popup.height = 800;
	popup.type = "head";
	popup.title = "Kreditoren";
	popup.addMasterdataID("2");
	popup.set("CreditorID", "CreditorId", true);
	popup.set("CreditorName", "Name1", true);
	popup.set("Creditor_PostCode", "Zip", true);
	popup.set("Creditor_City", "City", true);
	popup.set("Creditor_Country", "CountryCode", true);
	popup.set("Creditor_TaxID", "EUTaxId", true);
	popup.set("Creditor_NationalTaxID", "NationalTaxId", true);
	//popup.set("Creditor_Mail", "Email", true);
	popup.set("IBAN", "IBAN", true);
	//popup.showOnly("ColumnName");

	// Wird nach dem Setzen der Werte aufgerufen
	popup.processData = function(data, d5Context, options){
		var fileContext = d5Context.getFileContext();
		var creditorID  = fileContext.getFileFieldValue("CreditorID");
		var hasOrder    = fileContext.getFileFieldValue("HasOrder");
		console.log("hasOrder: ", hasOrder);
		if( hasOrder==="0" || hasOrder==="false" || hasOrder===0 || hasOrder===false ){
			var params = {};
			params.pCredID = creditorID;
```

```
    var PPC = squeeze_callScript("SqueezeInvoice_UDA_GetPPCByCreditor", params);
    console.log("PPC: ", PPC);

    if (PPC !== ""){
        fileContext.setFileFieldValue("ConditionsOfPayment_ID", PPC);
    }
  }
};

popup.open();
});
```

Es werden zusätzliche Beispiele für die Bestellnummer, die Zahlungsbedingung sowie die Bestelldatenauswahl auf Positionsebene ausgeliefert.

Automatische Berechnungen

Bei einigen Betragsfeldern sollen automatische Berechnungen ausgeführt werden. Zum Beispiel kann aus dem Netto- und Brutto-Wert automatisch der Steuerbetrag berechnet werden. An den Feldern muss auf dem Exit-Register der Typ "**Bei Wertänderung**" gesetzt werden. Funktionen und Beispiele können der Hersteller-Dokumentation entnommen werden.

HTML Style

Für die korrekte Darstellung muss zwingend der folgender Style hinzugefügt werden. Der Part kann optional direkt in die "**script.jsp**" hinzugefügt werden.

```
 <style>
 .dexValidEntry
 {
    color: green;
 }

 .dexInvalidEntry
 {
    color: red;
 }
</style>
```

Properties

Die Feldbezeichnungen am Mappentypen sowie die Bezeichnungen der öffentlichen Ordner, etc. sind alle internationalisiert. Die Übersetzungen befinden sich in properties-Dateien, welche im Documents-Installations-Ordner unter "...\server\locale\" abgelegt werden müssen.

- **Dexpro_de.properties**
- **Dexpro_en.properties**
- **Squeeze_de.properties**
- **Squeeze_en.properties**

Mappentyp "SqueezeInvoice"

Die Feldstruktur des Mappentypen "SqueezeInvoice" entspricht zum großen Teil dem "Invoice" Mappentypen. Es wurden einige Felder, die rein zur Workflow-Steuerung dienen, entfernt. Dafür sind einige neue Felder hinzugekommen und bei vielen Betrags-Feldern wurde ein User-Exit bei Wertänderung gesetzt, damit automatische Berechnungen durchgeführt werden können.

Zugriffsrechte

Der Mappentyp hat in der Auslieferung keinen aktiven Mappenklassenschutz. Es werden lediglich fixe Mappenrechte gesetzt. Ein Mappenklassenschutz wird erst erforderlich, wenn Benutzer aus einer Gruppe nicht die Rechnungen aus der anderen Gruppe sehen dürfen.

Register

Der Mappentyp enthält 4 Register. Auf dem Register "**Squeeze**" wird bei der Anlage über Squeeze eine sqz-Datei erzeugt und auf dem Register abgelegt. Die Dateiendung ist intern mit dem Squeeze-Viewer verknüpft, welcher den passenden Rechnungs-Beleg aus Squeeze ausruft. Die Datei muss vor der Übergabe an den weiteren Rechnungs-Workflow wieder gelöscht werden.

Auf dem zweiten Belege-Register "**Documents**" wird die PDF-Datei abgelegt. Auf dem Feldregister "**Workflow**" befinden sich technische Felder zur Steuerung der Anzeige bei den dynamischen Filter-Ordnern und auf dem Register "**Gentable**" befinden sich die Felder zur Steuerung des Gentable.

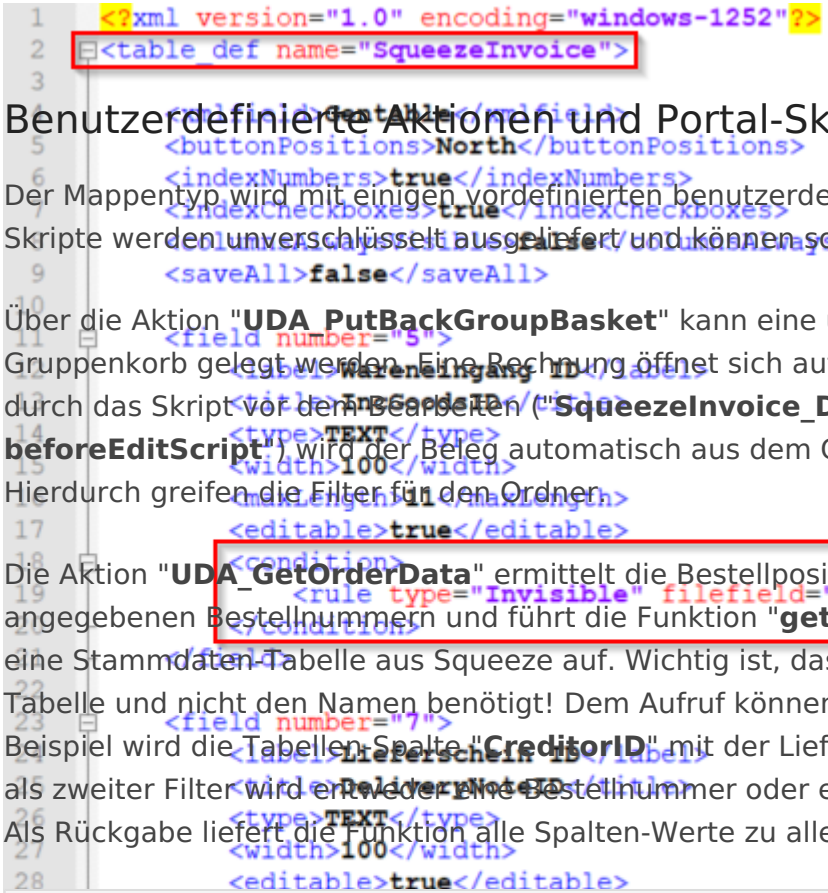
Gentable

Beim Gentable sollte die aktuelle Kundenlösung übernommen werden, so dass der Feldwert unverändert in das Gentable-Feld des finalen Mappentyps geschrieben werden kann. Es kann sein, dass ein Kunde unterschiedliche Gentable verwendet, um zum Beispiel Rechnungen mit und ohne Bestellbezug unterschiedlich behandeln zu können. Hier muss in Abstimmung mit dem Kunden entschieden werden welche Positions-Spalten im Gentable für die Validierung enthalten sein müssen.

Die Gentable XML wird über das Mappenfeld "**GentableXML**" geladen. Die Steuerung erfolgt über

das Skript "**SqueezeInvoice_DF_GentableDefScriptName**", welches über die Eigenschaft "**gentableDefScriptName**" aufgerufen wird. Das Skript kann optional auch andere XML aus anderen Feldern laden. Bei Bedarf kann die Eigenschaft auch entfernt werden und die XML-Struktur kann alternativ über eine Server-XML-Datei aufgerufen werden.

Der Name der Gentable-Definition sollte in "**SqueezeInvoice**" umbenannt werden, damit Überschneidungen zum bestehenden Gentable vermieden werden. Falls dies explizit gewollt ist muss der Name in den Funktionsaufrufen entsprechend angepasst werden. Das ausgelieferte Beispiel enthält Spalten, welche über eine Bedingung ("**condition**") nur bei Bestellbezug angezeigt werden. Bei einer Wertänderung am Bool-Feld "**HasOrder**" wird die Mappe aktualisiert. Dadurch werden die Gentable-Spalten entsprechend ein- oder ausgeblendet.



Benutzerdefinierte Aktionen und Portal-Skripte

Der Mappentyp wird mit einigen vordefinierten benutzerdefinierten Aktionen ausgeliefert. Die Skripte werden unverschlüsselt ausgeliefert und können somit uneingeschränkt angepasst werden.

Über die Aktion "**UDA_PutBackGroupBasket**" kann eine übernommene Rechnung zurück in den Gruppenkorb gelegt werden. Eine Rechnung öffnet sich automatisch im Bearbeitungs-Modus und durch das Skript vor dem Bearbeiten ("**SqueezeInvoice_DF_BeforeEdit**" - siehe Eigenschaft "**beforeEditScript**") wird der Beleg automatisch aus dem Gruppenkorb-Ordner entnommen. Hierdurch greifen die Filter für den Ordner.

Die Aktion "**UDA_GetOrderData**" ermittelt die Bestellpositionen. Das Skript iteriert die angegebenen Bestellnummern und führt die Funktion "**getOrderData()**" aus. Die Funktion ruft eine Stammdaten-Tabelle aus Squeeze auf. Wichtig ist, dass "**getMasterData()**" die ID der Tabelle und nicht den Namen benötigt! Dem Aufruf können Filter mitgegeben werden. In dem Beispiel wird die Tabellen-Spalte "**CreditorID**" mit der Lieferanten-ID zur Rechnung gefiltert und als zweiter Filter wird entweder eine Bestellnummer oder eine Lieferscheinnummer angegeben. Als Rückgabe liefert die Funktion alle Spalten-Werte zu allen Treffern in einem Objekt.

```
/** Ermittelt die Bestelldaten aus den Squeeze-Stammdaten
 * @param {string} vendorID Lieferanten ID für Filterung
 * @param {string} addColumnFilter Zusätzliche Filter-Spalte
 * @param {string} addColumnValue Spaltenwert
 **/
function getOrderData(vendorID, addColumnFilter, addColumnValue){
    var sqz = new Squeeze();
        sqz.addFilter("CreditorID", vendorID, "eq");
        sqz.addFilter(addColumnFilter, addColumnValue, "eq");
    var mdObj = sqz.getMasterdata("11"); /* Die ID der Stammdaten-Tabelle und nicht den
Namen! */
```

```
    if( mdObj.error===true ){
        return mdObj.errorMessage;
    }
    return mdObj.data;
}
```

Das Beispiel kann verwendet werden um weitere Stammdaten-Abfragen gegen die Squeeze-Stammdaten zu generieren. Durch die Abfragen an Squeeze kommt die Lösung ohne eigene Stammdaten aus.

Am Ende des Skripts wird aus den Bestelldaten ein Gentable erstellt. Diese Stelle muss projektspezifisch angepasst werden. In dem Beispiel wird das Gentable im XML-Format gespeichert.

```
/* Erstelle neues Gentable */
var gentable = "<table>\n";
for( var md=0; md<masterDataObj.length; md++ ){
    var masterDataEntry = masterDataObj[md];
    var articleText     = parseHtml(masterDataEntry.PosArticleText);
    if( masterDataEntry.id ){
        gentable += "    <tr>\n";
        gentable += "        <td title=\"OrderNumber\">"    +
masterDataEntry.OrderNumber        + "</td>\n";
        gentable += "        <td title=\"OrderPos\">"       +
masterDataEntry.Position           + "</td>\n";
        gentable += "        <td title=\"DeliveryNoteId\">" +
masterDataEntry.DeliveryNoteNumber + "</td>\n";
        gentable += "        <td title=\"IncGoodsID\">"     +
masterDataEntry.PosWENr            + "</td>\n";
        gentable += "        <td title=\"Quantity\">"       +
masterDataEntry.PosQuantity        + "</td>\n";
        gentable += "        <td title=\"QuantityUnit\">"   +
masterDataEntry.Unit               + "</td>\n";
        gentable += "        <td title=\"Price\">"          +
masterDataEntry.PosSinglePrice     + "</td>\n";
        gentable += "        <td title=\"PriceUnit\">"      +
masterDataEntry.PosPricingUnit     + "</td>\n";
        gentable += "        <td title=\"NetAmount\">"      +
masterDataEntry.PosTotalAmount     + "</td>\n";
        gentable += "        <td title=\"ArticleNumber\">"  +
masterDataEntry.PosArticleNumber   + "</td>\n";
        gentable += "        <td title=\"ArticleDesc\">"    +
```

```
articleText                          + "</td>\n";
        gentable += "    </tr>\n";
    }
}
gentable += "</table>";
docFile.Gentable = gentable;
if( docFile.sync() ){
    context.returnType = "showEditFile";
    return docFile.getid();
}
```

Die Aktion "**UDA_StartWorkflow**" startet den Rechnungs-Workflow. Bevor die Übergabe gestartet werden kann sollten einige Prüfungen erfolgen. Im ausgelieferten Skript wird zum Beispiel die Summe aus Netto und Steuer mit dem Brutto-Betrag verglichen und die Summe der Bestellpositionen wird gegen den Kopfbetrag geprüft. Es können beliebige Prüfungen hinzugefügt werden.

Im nächsten Schritt werden alle wichtigen Feldwerte in Variablen zwischengespeichert, denn beim später folgenden Mappentypwechsel gehen Feldwerte verloren, wenn ein Feld nicht im neuen Mappentypen enthalten ist. Zudem werden alle Eigenschaften an der Mappe entfernt.

Nach dem Mappentypwechsel müssen die Werte auf die neuen Felder gespeichert werden. Alle Belege werden automatisch auf das erste Dokumenten-Register geschoben. Die sqz-Datei muss gelöscht werden und ggf. müssen Eigenschaften neu gesetzt werden. Am Ende wird der passende Workflow gestartet.

Anpassung von Farben der Markierungen

Ab der Invoice-Version (V.1.1.115) sind die folgenden Parameter in der Datei **Documents5\DEXPRO_ClientExits\DexClientExits.jsp** bereits vorhanden. Über diese Parameter kann die Standard-Farbe der Markierungen angepasst werden. Ist der Wert leer oder einkommentiert, werden die Standardfarben genommen

Außerhalb von Invoice können diese Parameter innerhalb einer beliebigen Datei hinterlegt werden, welche in dem jeweiligen Projekt in den externen Ressourcen hinterlegt ist. Ggf. muss nach einer solchen Änderungen der Browser-Cache geleert werden. Sind die Parameter nicht vorhanden, wird automatisch die Standardfarbe verwendet.

```
SqueezeParam.fieldValidColor = ""; // Markierungsfarbe für von Squeeze erkannte Felder
SqueezeParam.fieldNotValidColor = ""; // Markeriungsfarbe für nicht von Squeeze erkannte Felder
SqueezeParam.fieldAlternativesColor = ""; // Markierungsfarbe für Felder mit Alternativen
```

> Gültig sind nur in HTML gültige Farbangaben im hexadezimalen Format (z.B. **#006B00**)

> Damit diese Parameter greifen, muss die Datei SqueezeFunctionsLib.js (**Documents5\SQUEEZE\www\SqueezeViewer\SqueezeFunctions.js**) mindestens in der Version **2.0.27** vorliegen. Vorherige Varianten unterstützen diesen Parameter nicht. Die aktuelle Version der Datei ist im oberen Teil der Datei ersichtlich.

Vorgehensweise für SqueezeFunctions.js-Versionen kleiner 2.0.27

In der Datei **Documents5\SQUEEZE\www\SqueezeViewer\SqueezeFunctions.js** kann die Farbe der Markierungen in Documents angepasst werden. Die Farbcodes entsprechen Standard HTML Color Codes.

Folgende Farben können gesetzt werden

1. **gFieldValid**: Farbe für gefüllte Felder (Standard: **#59E817**)
2. **gFieldNotValid**: Farbe für nicht gefüllte Felder (Standard: **#FF0000**)
3. **gFieldAlternatives**: Farbe für Felder mit Alternativen (Standard: **#FFA500**)

```
36
37  var gFieldValid = "#59E817";
38  var gFieldNotValid = "#FF0000";
39  var gFieldAlternatives = "#FFA500";
```

Nach Änderung der Datei muss ggf. der Browser-Cache geleert werden

Änderungen in dieser Datei werden nach Invoice-Updates eventuell überschrieben und müssen erneut vorgenommen werden.