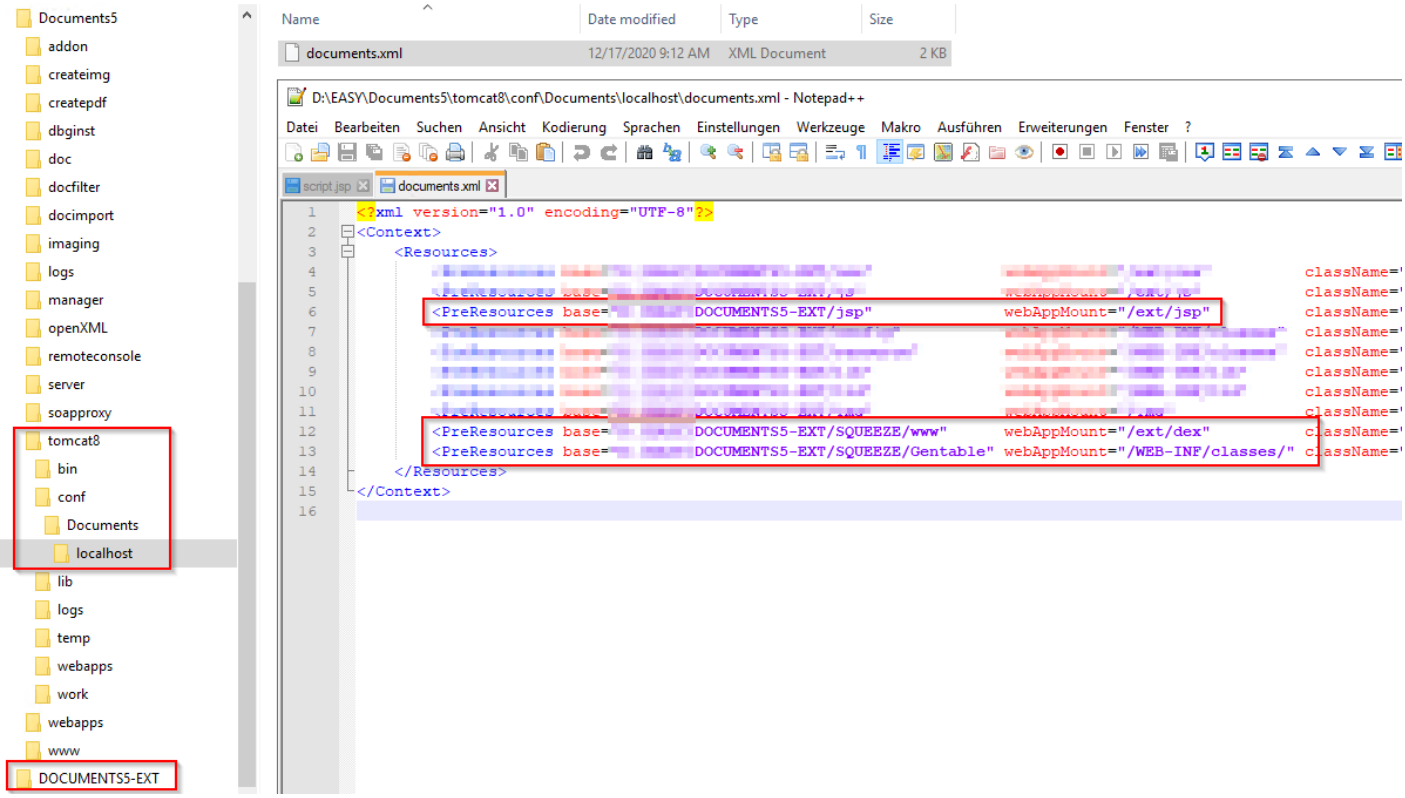
Externe Ressourcen

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

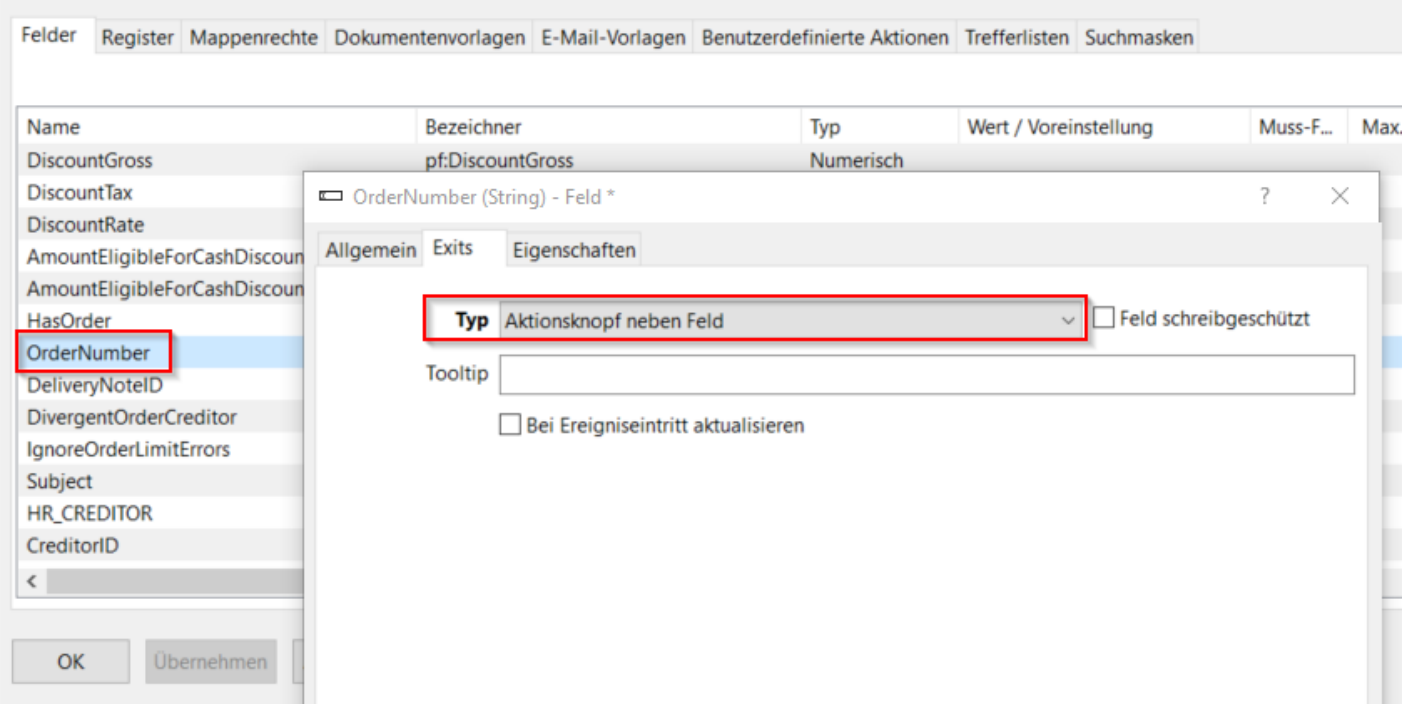
Über den Workflow kann eine Aktion derart konfiguriert werden, dass sich die Mappe immer direkt im Bearbeitungs-Modus öffnet. Bei der Validierung wird dieses Verhalten in der Regel gewünscht. Um das Verhalten nachzuahmen kann die folgende Funktion eingebunden werden.

```
/** 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
,File.afterFileEditCancel", 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>
```

Revision #5
Created 2021-02-09 13:08:38 UTC by Markus Meisner
Updated 2021-02-24 11:35:06 UTC by Markus Meisner