

Extension Entwicklung

In diesem Kapitel werden wichtige Integration Events und Beispiele aufgeführt, die bei der Entwicklung einer Extension für Breeze Interface helfen.

Wichtig:

Anpassungen dürfen nur durch entsprechend geschulte Berater/Entwickler durchgeführt werden. Diese befinden sich außerdem immer außerhalb des Standard-Supports durch DEXPRO.

- [Leitfaden zur Implementierung einer individuellen Belegerstellung](#)

Leitfaden zur Implementierung einer individuellen Belegerstellung

Überblick

Diese Dokumentation bietet eine Anleitung zur Implementierung benutzerdefinierter Dokumentenerstellungsprozesse unter Verwendung des Integration Event `OnDocumentCreation` im DEXPRO Core-Framework. Während die Standardimplementierung Einkaufsbelege (Tabelle 38,39) erstellt, zeigt dieser Leitfaden, wie Sie Ihre eigene Dokumentenerstellungslogik mithilfe der bereitgestellten JSON-Daten implementieren können.


Voraussetzungen

Folgende Einstellung ist vor der Validierung eines Belegs in der [Dokumentenklassen Einrichtung](#) vorzunehmen:

- **Nächster Prozessschritt:** Breeze Interface
- **Zielvorlage:** Keine

Rechnung / Gutschrift

Start | Navigation | Weitere Optionen


↓ Feldzuordnung herunterladen 

Allgemein

Dokumentenklasse Rechnung / Gutschrift  **1** Nächster Prozessschritt Breeze Interface 

Freeze >

Breeze Interface

Zielvorlage **2** Keine 

Sie haben einen Beleg mit der App [DEXPRO Squeeze](#) validiert und in den nächsten Prozessschritt [Breeze Interface](#) übertragen? Dann sehen Sie diesen jetzt im [Breeze Interface](#) in einer Wartestellung (*WaitingForRetrieval*). An diesem Punkt besteht die Möglichkeit, den validierten Beleg in einen Genehmigungsworkflow zu überführen, um diesen nach erfolgter Genehmigung zurückzumelden und in den Prozess der Zielbelegerstellung zu übergeben - weitere Information zur Breeze Interface API finden sie [hier](#).

Breeze Belege ✓ Gespe

Externe Belegnr.	Belegdatum	Buchungsdatu...	Eink. von Kredi-Nr.	Eink. von Name	Belegart	Nettobetrag	Steuerbetrag	Steuersatz	Gesamtbetrag	Status	Prozess ID	Hinweis	Doublette	Zielvorlage	Nr. 1	Core Belegnr.	API Beleg ID
→ B308154	13.07.2023	23.01.2025	K00170	Media-Hannes	Rechnung	7.894,71	1.499,99	19,00	9.394,70	WaitingForRetrieval			Einkaufsk...	Keine	BRZ00000000015	CGRE00000000012	52

Sobald Sie den Status des Breeze Belegs in "Processed" geändert haben, wird dieser an den [DEXPRO Core](#) zur weiteren Verarbeitung gemeldet - dies ist der [Integrationspunkt](#), an dem Sie ansetzen müssen.

Integrationspunkt

Der zentrale Integrationspunkt ist das `OnDocumentCreation`-Ereignis in *Codeunit 70954599 "DXP Doc. Creation Def. Impl."* in der App *DEXPRO Core*. Dieses Ereignis wird während der Dokumentenverarbeitung ausgelöst und ermöglicht die Implementierung einer benutzerdefinierten Dokumentenerstellungslogik.

Ereignis-Signatur

```
[IntegrationEvent(false, false)]
```

```
local procedure OnDocumentCreation(JObject: JsonObject; var CreatedDocumentRecordId: RecordId)
```

Parameter

- `JObject`: Enthält die vollständigen JSON-Daten mit den Dokumentinformationen
- `CreatedDocumentRecordId`: Muss mit der RecordId des erstellten Dokuments gesetzt werden

Implementierungsleitfaden

Schritt 1: Subscriber-Codeunit erstellen

Zunächst erstellen Sie eine Codeunit, die das OnDocumentCreation-Ereignis abonniert:

```
codeunit 50100 "Custom Document Creation"
{
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"DXP Doc. Creation Def. Impl.",
    'OnDocumentCreation', '', false, false)]
    local procedure OnDocumentCreation(JObject: JsonObject; var CreatedDocumentRecordId:
RecordId)
    begin
        // Your implementation here
        CreateCustomDocument(JObject, CreatedDocumentRecordId);
    end;
}
```

Schritt 2: Dokumentenerstellungslogik implementieren

Hier ein Beispiel für die Implementierung der benutzerdefinierten Dokumentenerstellung:

```
local procedure CreateCustomDocument(JObject: JsonObject; var CreatedDocumentRecordId:
RecordId)
var
```

```

CustomHeader: Record "Custom Document Header";
CustomLine: Record "Custom Document Line";
JsonHelper: Codeunit "DXP Json Helper";
HeaderJObject: JsonObject;
LinesJArray: JsonArray;
LineJToken: JsonToken;
LineNo: Integer;
begin
    // 1. Create Header
    CustomHeader.Init();
    CustomHeader."Document Type" := GetDocumentType(JObject);
    CustomHeader."Vendor No." := CopyStr(JsonHelper.ValAsTxt(JObject, 'vendorNo', true), 1,
MaxStrLen(CustomHeader."Vendor No."));
    CustomHeader."Document Date" := JsonHelper.ValAsDate(JObject, 'docDate', true);
    CustomHeader."Posting Date" := JsonHelper.ValAsDate(JObject, 'postingDate', true);
    CustomHeader."Document Reference" := CopyStr(JsonHelper.ValAsTxt(JObject, 'docReference',
true), 1, MaxStrLen(CustomHeader."Document Reference"));
    CustomHeader.Insert(true);

    // 2. Process Lines
    LinesJArray := JsonHelper.ReadJArrayFromObj(JObject, 'lines');
    LineNo := 10000;

    foreach LineJToken in LinesJArray do begin
        CustomLine.Init();
        CustomLine."Document No." := CustomHeader."No.";
        CustomLine."Line No." := LineNo;
        CustomLine."Item No." := CopyStr(JsonHelper.ValAsTxt(LineJToken.AsObject(), 'no',
true), 1, MaxStrLen(CustomLine."Item No."));
        CustomLine.Quantity := JsonHelper.ValAsDec(LineJToken.AsObject(), 'qty', true);
        CustomLine."Unit Price" := JsonHelper.ValAsDec(LineJToken.AsObject(), 'unitPrice',
true);
        CustomLine.Insert(true);

        LineNo += 10000;
    end;

    CreatedDocumentRecordId := CustomHeader.RecordId;
end;

```

JSON-Struktur

Die JSON-Eingabe folgt dieser Struktur:

```
{
  "type": "Invoice",
  "vendorNo": "K00170",
  "iban": "DE58520503530052599766",
  "vatRegNo": "DE540026784",
  "barcode": "123456789",
  "docDate": "2023-07-13",
  "serviceDate": "2023-07-13",
  "postingDate": "2025-01-23",
  "postingDesc": "Die ist eine Buchungsbeschreibung.",
  "docReference": "R308154",
  "orderNo": "B23106029",
  "netAmount": 7894.71,
  "netAmount2": 0.0,
  "netAmount3": 0.0,
  "taxRate": 19.0,
  "taxRate2": 0.0,
  "taxRate3": 0.0,
  "taxAmount": 1499.99,
  "taxAmount2": 0.0,
  "taxAmount3": 0.0,
  "totalAmount": 9394.7,
  "currency": "EUR",
  "assignedTo": "BERND.FEDDERSEN",
  "note": "Die ist eine Bemerkung!",
  "dimensions": {
    "ABTEILUNG": "PROD",
    "EINKÄUFER": "BF",
    "KOSTENTRÄGER": "IT"
  },
  "dimensionSetID": 27,
  "customFields": {
    "docClass": "DXP Invoice / Credit Memo",
    "namesAndValues": [
      {
```

```
        "name": "Customer_No",
        "value": "01121212"
    },
    {
        "name": "Custom_Date",
        "value": "2025-01-26"
    }
]
},
"orderMatchDifference": "false",
"lines": [
    {
        "orderNo": "B23106029",
        "orderLineNo": 10000,
        "receiptNo": "ELIEF107004",
        "receiptLineNo": 10000,
        "vendorItemNo": "",
        "type": "Item",
        "no": "90002",
        "description": "Jogitek G 1337 Gaming-Headset",
        "qty": 17.0,
        "uom": "STÜCK",
        "unitPrice": 99.99,
        "lineDisc": 0.0,
        "netAmount": 1699.83,
        "totalAmount": 2022.8,
        "taxRate": 19.0,
        "vatBusPostingGroup": "INLAND",
        "vatProdPostingGroup": "MWST.19",
        "dimensions": {
            "ABTEILUNG": "PROD",
            "BEREICH": "30",
            "EINKÄUFER": "BF",
            "KOSTENTRÄGER": "IT",
            "VERKAUFSKAMPAGNE": "WINTER"
        },
        "dimensionSetID": 53,
        "genProdPostingGroup": "HANDEL",
        "genBusPostingGroup": "INLAND",
```

```
"deferralCode":"","
"quantityDifference":"false",
"unitPriceDifference":"false",
"discountDifference":"false",
"receiptDateDifference":"false",
"customFields":{
  "docClass":"DXP Invoice / Credit Memo",
  "namesAndValues":[
    {
      "name":"Custom_Field_1",
      "value":"Text in Zeile 10000"
    },
    {
      "name":"Custom_Field_2",
      "value":"2025-01-31"
    }
  ]
}
},
{
  "orderNo":"B23106029",
  "orderLineNo":30000,
  "receiptNo":"ELIEF107004",
  "receiptLineNo":30000,
  "vendorItemNo":"","
  "type":"Item",
  "no":"90005",
  "description":"Sumsing Galaxy 23",
  "qty":12.0,
  "uom":"STÜCK",
  "unitPrice":349.99,
  "lineDisc":0.0,
  "netAmount":4199.88,
  "totalAmount":4997.86,
  "taxRate":19.0,
  "vatBusPostingGroup":"INLAND",
  "vatProdPostingGroup":"MWST.19",
  "dimensions":{
    "ABTEILUNG":"PROD",
```

```
    "BEREICH": "50",
    "EINKÄUFER": "BF",
    "KOSTENTRÄGER": "IT",
    "VERKAUFSKAMPAGNE": "WINTER"
  },
  "dimensionSetID": 55,
  "genProdPostingGroup": "HANDEL",
  "genBusPostingGroup": "INLAND",
  "deferralCode": "",
  "quantityDifference": "false",
  "unitPriceDifference": "false",
  "discountDifference": "false",
  "receiptDateDifference": "false",
  "customFields": {
    "docClass": "DXP Invoice / Credit Memo",
    "namesAndValues": [
      {
        "name": "Custom_Field_1",
        "value": "Text in Zeile 20000"
      },
      {
        "name": "Custom_Field_2",
        "value": "2025-01-30"
      }
    ]
  }
},
{
  "orderNo": "B23106029",
  "orderLineNo": 20000,
  "receiptNo": "ELIEF107004",
  "receiptLineNo": 20000,
  "vendorItemNo": "",
  "type": "Item",
  "no": "90000",
  "description": "Y-Phone Ultra Pro 23",
  "qty": 5.0,
  "uom": "STÜCK",
  "unitPrice": 399.0,
```

```
"lineDisc":0.0,
"netAmount":1995.0,
"totalAmount":2374.05,
"taxRate":19.0,
"vatBusPostingGroup":"INLAND",
"vatProdPostingGroup":"MWST.19",
"dimensions":{
  "ABTEILUNG":"PROD",
  "BEREICH":"70",
  "EINKÄUFER":"BF",
  "KOSTENTRÄGER":"IT",
  "VERKÄUFER":"JH",
  "VERKAUFSKAMPAGNE":"WINTER"
},
"dimensionSetID":58,
"genProdPostingGroup":"HANDEL",
"genBusPostingGroup":"INLAND",
"deferralCode":"",
"quantityDifference":"false",
"unitPriceDifference":"false",
"discountDifference":"false",
"receiptDateDifference":"false",
"customFields":{
  "docClass":"DXP Invoice / Credit Memo",
  "namesAndValues":[
    {
      "name":"Custom_Field_1",
      "value":"Text in Zeile 30000"
    },
    {
      "name":"Custom_Field_2",
      "value":"2025-01-29"
    }
  ]
}
]
}
}
```

Best practice

1. **Fehlerbehandlung:** Implementieren Sie eine ordnungsgemäße Fehlerbehandlung für JSON-Parsing und Datenbankoperationen:

```
local procedure CreateCustomDocument(JObject: JsonObject; var CreatedDocumentRecordId: RecordId)
var
    CustomHeader: Record "Custom Document Header";
begin
    if not DoesJsonHaveRequiredFields(JObject) then
        Error('Required fields are missing in the JSON payload');

    if not TryCreateCustomHeader(JObject, CustomHeader) then
        Error('Failed to create document header');

    // ... rest of the implementation
end;
```

2. **Validierung:** Implementieren Sie eine ordnungsgemäße Validierung vor der Dokumentenerstellung:

```
local procedure ValidateDocument(JObject: JsonObject): Boolean
var
    Vendor: Record Vendor;
    VendorNo: Code[20];
begin
    VendorNo := CopyStr(JsonHelper.ValAsTxt(JObject, 'vendorNo', true), 1,
MaxStrLen(VendorNo));
    if not Vendor.Get(VendorNo) then
        Error('Vendor %1 does not exist', VendorNo);
    // Add more validation as needed
    exit(true);
end;
```

Häufige Szenarien

Szenario 1: Verschiedene Dokumenttypen erstellen

```
local procedure GetDocumentType(JObject: JsonObject): Enum "Custom Document Type"
var
    DocType: Text;
begin
    DocType := JsonHelper.ValAsTxt(JObject, 'type', true);
    case DocType of
        'Invoice':
            exit("Custom Document Type"::Invoice);
        'CreditMemo':
            exit("Custom Document Type"::"Credit Memo");
        else
            Error('Unknown document type: %1', DocType);
    end;
end;
```

Szenario 2: Benutzerdefinierte Felder verarbeiten

```
local procedure ProcessCustomFields(JObject: JsonObject; var CustomDoc: Record "Custom Document Header")
var
    CustomFieldsObj: JsonObject;
    CustomFieldsArr: JsonArray;
    FieldToken: JsonToken;
begin
    if not JsonHelper.TokenExists(JObject, 'customFields') then
        exit;

    CustomFieldsObj := JsonHelper.ReadJsonObjectFromObj(JObject, 'customFields');
    CustomFieldsArr := JsonHelper.ReadJsonArrayFromObj(CustomFieldsObj, 'namesAndValues');

    foreach FieldToken in CustomFieldsArr do
        ProcessCustomField(FieldToken.AsObject(), CustomDoc);
    end;
end;
```

Fehlerbehebung

1. **RecordId ist leer:** Stellen Sie sicher, dass Sie den CreatedDocumentRecordId-Parameter setzen:

```
CreatedDocumentRecordId := CustomHeader.RecordId;
```

2. **JSON-Parsing-Fehler:** Verwenden Sie die JsonHelper-Funktionen für sicheres Parsing:

```
if not JsonHelper.TokenExists(JObject, 'vendorNo') then  
    Error('Vendor number is missing in JSON');
```

3. **Datenvalidierung:** Implementieren Sie eine ordnungsgemäße Datenvalidierung:

```
if JsonHelper.ValAsDec(LineJToken.AsObject(), 'qty', true) <= 0 then  
    Error('Quantity must be greater than 0');
```