Events beim Validieren eines SQUEEZE-Belegs (Aktion „Validieren")

Zielgruppe

Diese Dokumentation richtet sich an Drittanbieter-Entwickler, die eigene Logik ausführen möchten, wenn ein Anwender auf der Seite **DXP SQZ Document v2** (Page `70954659`) die Aktion **Validieren** auslöst – etwa zusätzliche eigene Prüfungen vor der Validierung, das Setzen weiterer Werte auf dem SQUEEZE-Beleg oder Folgeaktionen auf dem erzeugten Zielbeleg.

Die beschriebenen Events stehen ab **SQUEEZE für Dynamics 365 BC Version 2.50** zur Verfügung.

Einordnung: Was passiert beim Klick auf „Validieren"?

Die Aktion **Validieren** ruft nacheinander zwei Routinen der Belegseite auf:

1) `CheckPlausibility()`

Führt die **Plausibilitätsprüfung** der jeweiligen **Dokumentenklasse** aus und aktualisiert die **Plausibilitätseinträge** im unteren Bereich der Seite. Diese Routine läuft nicht nur beim Klick auf **Validieren**, sondern nach nahezu jeder Feldänderung. Sie ist deshalb **kein** geeigneter Aufhängepunkt für „einmalig beim Validieren".

Für Erweiterungen der Prüfung selbst existieren die bereits dokumentierten Events `OnBeforeDoPlausibilityChecks`, `OnAfterDoHeaderPlausibilityChecks`, `OnAfterDoLinePlausibilityChecks` und `OnAfterDoPlausibilityChecks`.

2) `FinishValidation()`

Bricht mit einer Fehlermeldung ab, solange noch Plausibilitätseinträge offen sind, behandelt gegebenenfalls die Dublettenprüfung und ruft anschließend die Methode `ValidateSQUEEZEDocument` der Codeunit **`DXP SQZ Document Mgt.`** (`70954632`) auf. Dort werden der verarbeitete JSON erzeugt, der Core-Beleg (**`DXP Document`**) weiterverarbeitet, der Zielbeleg erstellt und der SQUEEZE-Beleg abgeschlossen.

Beide hier beschriebenen Events werden in `DXP SQZ Document Mgt.ValidateSQUEEZEDocument` ausgelöst. Damit greifen sie einheitlich für:

- die Aktion **Validieren** auf **DXP SQZ Document v2** (`70954659`),
- die Aktion **Validieren** auf **DXP SQZ Document (Generic)** (`70954670`) und den älteren Belegseiten,
- die **automatische Validierung** (z. B. aus der Verarbeitungswarteschlange).

Es gibt bewusst kein Event direkt an der Aktion. Wenn Sie zwischen manuellem Klick und automatischer Validierung unterscheiden möchten, werten Sie den Parameter `AutomaticValidation` aus.

Event-Matrix (Kurzüberblick)

Bereich	Event	Zeitpunkt	Steuerung	Typische Use-Cases
`DXP SQZ Document Mgt.`	`OnBeforeValidateSQUEEZEDocument`	Nach der Plausibilitäts- und Dublettenprüfung, bevor der verarbeitete JSON erzeugt und der Zielbeleg angelegt wird	`var IsHandled: Boolean` bricht die Validierung ab; `var DocHeader` erlaubt Änderungen am Beleg	Eigene Zusatzprüfungen mit Abbruch, Felder nachbefüllen, Freigabeworkflow anstoßen, externe Systeme abfragen
`DXP SQZ Document Mgt.`	`OnAfterValidateSQUEEZEDocument`	Nachdem der Core-Beleg verarbeitet und der Zielbeleg erstellt wurde, vor dem Validierungsdialog	Kein Abbruch möglich	Zielbeleg nachbearbeiten, eigene Protokolltabellen schreiben, Benachrichtigungen versenden, Drittsysteme informieren

Abgrenzung zu `OnAfterWriteProcessedJsonToBlob`

Für Logik **nach** der Validierung existiert bereits ein Core-Event, beschrieben auf der Seite *Erstellung benutzerdefinierter Dokumente (Individuelle Verarbeitung)*:

```
[IntegrationEvent(false, false)]
local procedure OnAfterWriteProcessedJsonToBlob(var Document: Record "DXP Document"; var
ProcessedJSONObj: JsonObject)
```

Es wird in der Codeunit **`DXP Document Mgt.`** (Core) ausgelöst, nachdem der verarbeitete JSON in den Core-Beleg geschrieben wurde – innerhalb des Validierungsvorgangs, also ebenfalls beim Klick auf **Validieren**.

	`OnAfterWriteProcessedJsonToBlob` **(Core)**	**`OnAfterValidateSQUEEZEDocument`** **(SQUEEZE)**
Ausgelöst in	`DXP Document Mgt.`	`DXP SQZ Document Mgt.`
Gilt für	**alle** Core-Belege – auch aus anderen DEXPRO-Quell-Apps. Sie müssen selbst auf die Dokumentenklasse filtern.	ausschließlich SQUEEZE-Belege
Kontext	`DXP Document` und der verarbeitete JSON	zusätzlich der komplette **`DXP SQZ Document Header`** mit allen erkannten Werten, Bestellabgleich- und Splitting-Informationen
Manuell vs. automatisch	nicht unterscheidbar	über `AutomaticValidation` unterscheidbar
Zeitpunkt	direkt nach dem Schreiben des JSON, noch innerhalb der Core-Verarbeitung	am Ende der Validierung, nachdem der SQUEEZE-Beleg abgeschlossen wurde
JSON änderbar	ja (`var ProcessedJSONObj`)	nein

Empfehlung:

- Sie möchten den **verarbeiteten JSON** noch verändern oder einen Beleg in die **Individuelle Verarbeitung** übernehmen → `OnAfterWriteProcessedJsonToBlob`.
- Sie möchten SQUEEZE-spezifische Werte auswerten oder nur bei manueller Validierung reagieren → `OnAfterValidateSQUEEZEDocument`.
- Sie möchten **vor** der Erstellung des Zielbelegs eingreifen oder die Validierung abbrechen → `OnBeforeValidateSQUEEZEDocument`. Dafür gibt es im Core keine Entsprechung.

`OnBeforeValidateSQUEEZEDocument`

```
[IntegrationEvent(false, false)]
local procedure OnBeforeValidateSQUEEZEDocument(var DocHeader: Record "DXP SQZ Document
```

```
Header"; AutomaticValidation: Boolean; var IsHandled: Boolean)
```

Parameter	Beschreibung
`var DocHeader`	Der zu validierende SQUEEZE-Beleg. Änderungen an Feldern werden anschließend mitgeschrieben und fließen in den Zielbeleg ein.
`AutomaticValidation`	`false` beim Klick auf **Validieren**, `true` bei automatischer Validierung.
`var IsHandled`	Auf `true` gesetzt bricht die Validierung ab. Der SQUEEZE-Beleg bleibt unverändert offen.

Das Event wird ausgelöst, nachdem:

- der Beleg neu gelesen wurde und weder den Status **Gelöscht** noch **Abgelehnt** hat,
- bei automatischer Validierung die Plausibilitätsprüfung erfolgreich war,

und bevor der verarbeitete JSON erzeugt wird. Zu diesem Zeitpunkt existiert der Zielbeleg noch nicht.

`OnAfterValidateSQUEEZEDocument`

```
[IntegrationEvent(false, false)]
local procedure OnAfterValidateSQUEEZEDocument(DocHeader: Record "DXP SQZ Document Header";
CoreDocument: Record "DXP Document"; AutomaticValidation: Boolean)
```

Parameter	Beschreibung
`DocHeader`	Der validierte SQUEEZE-Beleg. Sein Status ist zu diesem Zeitpunkt bereits **Gelöscht** – das ist der reguläre Endzustand eines validierten Belegs und **kein** Fehler.
`CoreDocument`	Der zugehörige Core-Beleg (`DXP Document`), inklusive `Status` und `Linked-to Record Id` des erstellten Zielbelegs.
`AutomaticValidation`	`false` beim Klick auf **Validieren**, `true` bei automatischer Validierung.

Ob der Zielbeleg tatsächlich erstellt wurde, ermitteln Sie über den Core-Beleg:

- `CoreDocument.Status = CoreDocument.Status::Transferred` – der Zielbeleg wurde erstellt, `CoreDocument."Linked-to Record Id"` verweist darauf.
- `CoreDocument.Status = CoreDocument.Status::"Custom Processing"` – der Beleg wurde in die **Individuelle Verarbeitung** übergeben; ein Standard-Zielbeleg existiert nicht.

Wichtig: eigene Prüfungen gehören **nicht** in dieses Event

Für Prüfungen, die eine Validierung verhindern sollen, ist `OnBeforeValidateSQUEEZEDocument` der falsche Ort. Ein `Error` an dieser Stelle erscheint erst *nach* dem Klick auf **Validieren** als Dialog – der Anwender sieht das Problem also erst am Ende der Bearbeitung. Nutzen Sie stattdessen einen der beiden folgenden Wege. Beide sorgen dafür, dass das Problem als **Plausibilitätseintrag** angezeigt und die Aktion **Validieren** deaktiviert wird.

Weg 1 (ohne Entwicklung): Plausibilitätsregelgruppe

In der **Dokumentenklassen-Einrichtung** legen Sie unter **Plausibilitätsregelgruppe** Regeln an. Jede aktive und gültige Regel, deren Bedingungen **zutreffen**, beschreibt ein Problem und erzeugt automatisch einen Plausibilitätseintrag. Regeln, die sich auf die Belegzeilen beziehen, werden je Zeile ausgewertet, sodass der Eintrag auf die betroffene Zeile verweist.

Für Prüfungen wie „Shortcutdimensionscode 1 darf nicht leer sein" ist das der bevorzugte Weg – kein Code, keine Erweiterung, vom Anwender selbst pflegbar.

Weg 2 (mit Entwicklung): `OnAfterDoPlausibilityChecks`

Dieses Event wird von allen Dokumentenklassen mit `var TempPlausibilityCheckEntry` veröffentlicht und ist damit der zuverlässige Weg. Sie fügen Ihren Eintrag direkt in den temporären Datensatz ein:

```
codeunit 50100 "My SQZ Plausibility Checks"
{
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"DXP SQZ P. Inv/Crdt Memo Impl.",
        'OnAfterDoPlausibilityChecks', '', false, false)]
    local procedure OnAfterChecks(DocHeader: Record "DXP SQZ Document Header"; var
TempPlausibilityCheckEntry: Record "DXP Plausibility Check Entry" temporary)
    var
        CostCenterMissingLbl: Label 'Please enter a cost center.';
    begin
        if DocHeader."Shortcut Dimension 1 Code" <> '' then
            exit;
```

```
        AddEntry(TempPlausibilityCheckEntry, CostCenterMissingLbl, DocHeader.RecordId(), 0);
    end;

    local procedure AddEntry(var TempPlausibilityCheckEntry: Record "DXP Plausibility Check
Entry" temporary; MessageText: Text; LinkedToRecId: RecordId; LookupPageId: Integer)
    var
        NewEntryNo: Integer;
    begin
        NewEntryNo := 1;
        TempPlausibilityCheckEntry.Reset();
        TempPlausibilityCheckEntry.SetCurrentKey("Entry No.");
        if TempPlausibilityCheckEntry.FindLast() then
            NewEntryNo := TempPlausibilityCheckEntry."Entry No." + 1;

        TempPlausibilityCheckEntry.Init();
        TempPlausibilityCheckEntry."Entry No." := NewEntryNo;
        TempPlausibilityCheckEntry.Text := CopyStr(MessageText, 1,
MaxStrLen(TempPlausibilityCheckEntry.Text));
        TempPlausibilityCheckEntry."Linked to Record Id" := LinkedToRecId;
        TempPlausibilityCheckEntry."Lookup Page Id" := LookupPageId;
        if TempPlausibilityCheckEntry.Insert() then;
    end;
}
```

Das passende Publisher-Objekt richtet sich nach der **Dokumentenklasse**:

Dokumentenklasse	Codeunit
Einkaufsrechnung / Gutschrift	`DXP SQZ P. Inv/Crdt Memo Impl.`
Auftragsbestätigung	`DXP SQZ P. Order Conf. Impl.`
Generische Dokumentenklassen	`DXP SQZ Document Mgt.`

Setzen Sie `Linked to Record Id` und `Lookup Page Id` (für Zeilen: `Page::"DXP SQZ Document Subform"`), damit der Plausibilitätseintrag anklickbar wird und zum betroffenen Datensatz springt.

Alternative: `OnAfterDoHeaderPlausibilityChecks`

Mit `PlausibilityCheck.AddPlausibilityCheckEntry(...)` steht eine bequemere API zur Verfügung, die Ihnen das manuelle Einfügen in den temporären Datensatz erspart:

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"DXP SQZ P. Inv/Crdt Memo Impl.",
    'OnAfterDoHeaderPlausibilityChecks', '', false, false)]
local procedure OnAfterHeaderChecks(DocHeader: Record "DXP SQZ Document Header"; var
PlausibilityCheck: Codeunit "DXP Plausiblity Check Mgt.")
var
    CostCenterMissingLbl: Label 'Please enter a cost center.';
begin
    if DocHeader."Shortcut Dimension 1 Code" = '' then
        PlausibilityCheck.AddPlausibilityCheckEntry(CostCenterMissingLbl, 0);
end;
```

Für Zeilenprüfungen gilt `OnAfterDoLinePlausibilityChecks` analog; über die Überladung `AddPlausibilityCheckEntry(Text; LinkedToRecId; LookupPageId)` wird der Eintrag anklickbar.

> **Achtung bei der Dokumentenklasse Auftragsbestätigung:** In `DXP SQZ P. Order Conf. Impl.` wurden `OnBeforeDoHeaderPlausibilityChecks`, `OnBeforeDoLinePlausibilityChecks`, `OnAfterDoHeaderPlausibilityChecks` und `OnAfterDoLinePlausibilityChecks` ursprünglich ohne `var` auf dem Parameter `PlausibilityCheck` veröffentlicht. Ein Subscriber erhält dort eine Kopie, sodass hinzugefügte Einträge verloren gehen. Verwenden Sie in dieser Dokumentenklasse ab **Version 2.50** stattdessen die Varianten mit dem Suffix `V2` – zum Beispiel `OnAfterDoHeaderPlausibilityChecksV2`. Sie sind signaturgleich zu den Events der übrigen Dokumentenklassen. Die alten Events bleiben erhalten, sind aber als veraltet markiert.

Beispiel 1: Werte vor der Übergabe an den Zielbeleg ergänzen

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"DXP SQZ Document Mgt.",
    'OnBeforeValidateSQUEEZEDocument', '', false, false)]
local procedure OnBeforeValidateSetPostingDescription(var DocHeader: Record "DXP SQZ Document
Header"; AutomaticValidation: Boolean; var IsHandled: Boolean)
begin
    if DocHeader."Posting Description" = '' then
```

```
        DocHeader."Posting Description" := CopyStr(DocHeader."Document Reference", 1,
MaxStrLen(DocHeader."Posting Description"));
    // Kein Modify notwendig – der Beleg wird im Anschluss geschrieben.
end;
```

Zielbeleg nachbearbeiten: bitte die Core-Events verwenden

Wenn Sie den **erzeugten Zielbeleg** verändern möchten, ist keines der beiden Events auf dieser Seite der richtige Ort. Der Core stellt dafür Events bereit, die den Zielbeleg direkt übergeben – bei den meisten sogar als typisierten `Purchase Header`, sodass der Umweg über `Linked-to Record Id` und einen `RecordRef` entfällt. Diese Events sind auf der Seite *Events nach der Erstellung eines Core-Belegs (DXP Document)* beschrieben:

Zielbeleg	Codeunit	Event
Einkaufsrechnung / Gutschrift	`DXP P. Inv. CredMemo. Creation`	`OnAfterProcessStandardDocument(JObject; DocRecRef)`
Einkaufsbeleg allgemein	`DXP Document Transfer Mgt.`	`OnAfterCreatePurchaseDoc(var JObject; PurchaseHeader)`
Einkaufskopf	`DXP Document Transfer Mgt.`	`OnAfterCreatePurchaseHeader(var JObject; PurchaseHeader)`
Bestellung	`DXP P. Order Creation`	`OnAfterCreatePurchaseOrder(var JObject; var PurchaseHeader)`
Auftragsbestätigung	`DXP P. Order Confirmation`	`OnAfterProcessStandardDocument(JObject; var PurchaseHeader)`

`OnAfterValidateSQUEEZEDocument` ist erst dann die bessere Wahl, wenn Sie zusätzlich Werte aus dem **`DXP SQZ Document Header`** benötigen – etwa erkannte Werte, Bestellabgleich- oder Splitting-Informationen, die im JSON des Core-Belegs nicht enthalten sind.

Beispiel 2: Nur bei manueller Validierung reagieren

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"DXP SQZ Document Mgt.",
    'OnAfterValidateSQUEEZEDocument', '', false, false)]
local procedure OnAfterValidateNotifyUser(DocHeader: Record "DXP SQZ Document Header";
CoreDocument: Record "DXP Document"; AutomaticValidation: Boolean)
```

```
var
    ValidatedMsg: Label 'Document %1 has been forwarded.', Comment = '%1 = SQUEEZE document
no.';
begin
    if AutomaticValidation then
        exit;
    if not GuiAllowed() then
        exit;

    Message(ValidatedMsg, DocHeader."No.");
end;
```

Hinweise und Einschränkungen

- **Kein Dialog in der automatischen Verarbeitung.** Prüfen Sie vor `Message`, `Confirm` oder `Page.RunModal` immer `AutomaticValidation` und `GuiAllowed()`. Andernfalls läuft die Verarbeitungswarteschlange auf einen Fehler.
- `OnBeforeValidateSQUEEZEDocument` **läuft nach der Plausibilitätsprüfung.** Eigene Prüfungen, die in den **Plausibilitätseinträgen** der Seite erscheinen und die Aktion **Validieren** deaktivieren sollen, gehören stattdessen in `OnAfterDoHeaderPlausibilityChecks` bzw. `OnAfterDoLinePlausibilityChecks`.
- `IsHandled := true` **bricht vollständig ab.** Es wird kein Zielbeleg erstellt, der Core-Beleg bleibt unverändert und der SQUEEZE-Beleg behält seinen Status. Der Anwender erhält keine Meldung – geben Sie bei Bedarf selbst eine aus.
- **Kein Event bei gelöschten oder abgelehnten Belegen.** Hat der Beleg den Status **Gelöscht** oder **Abgelehnt**, wird die Validierung vorher beendet und keines der beiden Events ausgelöst.
- `OnAfterValidateSQUEEZEDocument` **läuft nach einem** `Commit`**.** Ein `Error` in einem Subscriber macht die Validierung nicht rückgängig, sondern hinterlässt einen validierten Beleg mit einer Fehlermeldung für den Anwender.

Verwandte Seiten

- Erstellung benutzerdefinierter Dokumente (Individuelle Verarbeitung)
- Events nach der Erstellung eines SQUEEZE-Belegs (CreateSource)
- Events nach der Erstellung eines Core-Belegs (DXP Document)
- Regelbasierter nächster Prozessschritt (Belege in Individuelle Verarbeitung steuern)

Revision #1
Created 2026-08-04 09:51:10 UTC by Bernd Feddersen
Updated 2026-08-04 09:52:54 UTC by Bernd Feddersen