Erweiterbarkeit (für Entwickler)

Der eInvoice Sender ist um eine kleine Menge von AL-Schnittstellen herum aufgebaut, sodass Partner und Kunden ihn erweitern können - Unterstützung für neue Belegarten hinzufügen, einen anderen Reader einbinden oder eigene Validierungsregeln ergänzen -, ohne die Basis-App zu verändern. Diese Seite richtet sich an **Entwickler**.

Schnittstelle 1 - DXP eInv Doc Exporter

Implementieren Sie **DXP eInv Doc Exporter**, um die Export-Payload für eine bestimmte Belegart zu erstellen. Sie besitzt eine einzige Methode, `CreateDocument(DocHeader: Variant): JsonObject` - sie wandelt eine Art von Business-Central-Beleg in die JSON-Payload um, die die Pipeline serialisiert und an die Generierung sendet.

Implementierungen werden über das erweiterbare Enum **Export Document** gebunden: Fügen Sie einen Wert für Ihre Belegart hinzu und verweisen Sie ihn auf Ihre implementierende Codeunit. Die Pipeline wählt den Exporter durch Auflösen dieses Enums aus, sodass das Registrieren eines neuen Exporters lediglich eine Erweiterung des Enums ist.

Schnittstelle 2 - DXP eInv Doc Reader V2

Implementieren Sie **DXP eInv Doc Reader V2**, um einen Reader über die CIM-Darstellung (Common Information Model) eines Belegs bereitzustellen - seine Methode `ReadDocument` füllt die CIM-Puffer (Beleg, Partei, Zeile, Zu-/Abschlag, Zahlung, Anhang), sodass Regeln und Generierung die Daten einheitlich verarbeiten können.

>

> Das Enum **Export Document** bindet *beide* Schnittstellen: jeder Enum-Wert verweist auf einen Exporter und einen Reader für diese Belegart. Wenn Sie eine neue Belegart hinzufügen, stellen Sie beide bereit.

Schnittstelle 3 - DXP eInv IValidation Rule

Implementieren Sie **DXP eInv IValidation Rule**, um Validierungsregeln hinzuzufügen. Sie besitzt zwei Methoden: `Evaluate` (führt die Prüfung aus und hängt Feststellungen an) und `GetMetadata` (liefert Kategorie, Standard-Schweregrad, Stufe, Beschreibung und Behebungshinweis der Regel zurück - was die Regel-Einrichtung befüllt). Eine einzige Codeunit kann anhand der Regel-ID verzweigen und viele Regeln bedienen, sodass das Budget für Objekt-IDs eingehalten wird.

Regeln werden über das erweiterbare Enum **Validation Rule** gebunden, das einen dedizierten Erweiterungspunkt **CUSTOM** für kunden- und partnereigene Regeln bereitstellt. Fügen Sie einen an Ihre implementierende Codeunit gebundenen Enum-Wert hinzu, und die Regel nimmt wie jede eingebaute Regel an der Validierung teil - sie erscheint in der Regel-Einrichtung, wo ihr Aktiviert-Kennzeichen, ihre Schweregrad-Übersteuerung und ihre Stufe konfiguriert werden können.

Das Ereignis OnCollectValidationRules

Die App veröffentlicht das Integrationsereignis **OnCollectValidationRules**. Abonnieren Sie es, um Ihre Regelimplementierungen zur Laufzeit bei der Validierungs-Engine zu registrieren. Dies ist der Einhängepunkt, über den eine erweiternde App Regeln zu der Sammlung beisteuert, die sowohl die Pipeline als auch die Live-Konformitätsprüfung verwenden.

Zwei Ereignisse für Feldzuordnungen

Eine Feldzuordnung mit der **Quellenart Erweiterung** fragt eine Erweiterung nach ihrem Wert. Die App veröffentlicht dafür zwei Integrationsereignisse:

- **OnResolveAssignmentValue** - `(AssignmentCode: Code[20]; FieldPath: Text; SourceRecRef: RecordRef; var Value: Text; var IsHandled: Boolean)`. Es wird für jede Zuordnung mit der Quellenart Erweiterung ausgelöst; setzen Sie `Value` und `IsHandled`. `SourceRecRef` ist der gebuchte Kopf bei einem Kopffeld und die gebuchte Zeile bei einem Zeilenfeld, und Zuordnungscode und Feldpfad sagen, welcher Wert gefragt ist. Verwenden Sie dieses Ereignis anstelle von **OnBeforeSetJsonValue**, das der Payload-Serializer nicht mehr auslöst.
- **OnLookupPostedDocument** - `(ExportDocType: Enum "DXP eInv Export Document"; var DocumentRecRef: RecordRef; var Picked: Boolean; var IsHandled: Boolean)`. Damit kann eine App, die eigene gebuchte Belege führt, diese dem Test der Feldzuordnungen anbieten. Ohne Subscriber kann der Test nur aus den Belegarten wählen, die diese App kennt.

> Die Aktion **Erweiterung erstellen** im Feldkatalog schreibt den Subscriber für Sie: Sie lädt eine kleine AL-Erweiterung für das markierte Feld herunter und legt die Zuordnung mit der Quellenart Erweiterung an, für die sie antwortet, falls es noch keine gibt, und schaltet sie ein. Bis die Erweiterung installiert ist, steht in der Payload-Herkunft des Belegs, dass kein Abonnent geantwortet hat.

Eine Belegart, die eine Erweiterung hinzufügt, braucht auch ihre gebuchten Tabellen, damit Bedingungs- und Wertregeln gegen sie ausgewertet werden können - reichen Sie sie über **OnGetPostedTablesForDocType** nach. Ohne das stehen für diese Belegart keine Regeln zur Verfügung.

> Zusammen decken diese drei Schnittstellen plus das Integrationsereignis die wichtigsten Erweiterungsszenarien ab: neue Belegart (Doc Exporter + Enum Export Document), benutzerdefiniertes Modell-Lesen (Doc Reader V2) und benutzerdefinierte Validierung (IValidation Rule + Enum Validation Rule, registriert über OnCollectValidationRules). Hinzu kommen für Feldzuordnungen die beiden oben beschriebenen Ereignisse, für einen Wert, der im Code berechnet werden muss.

Revision #3
Created 2026-07-22 13:20:33 UTC by Bernd Feddersen
Updated 2026-11-25 17:30:28 UTC by Bernd Feddersen