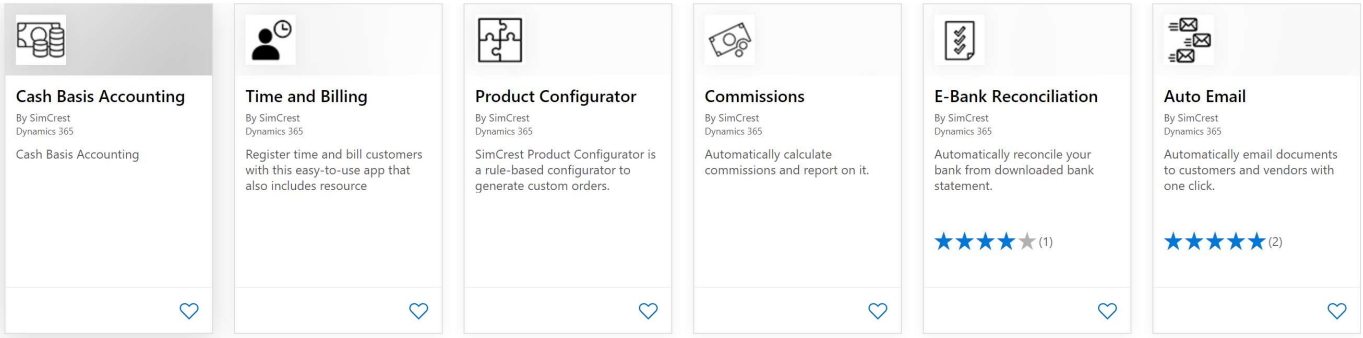
Business Central Migration Series 200: Migrate Dynamics NAV add-ons to BC Cloud.

For Business Central AppSource

Carsten Howitz, SimCrest, Inc.

Introduction – Carsten Howitz

- Started working with Navigator / Navision in 1990 in Denmark.
- Development, System Architect, Consulting, and Project Management of Navigator, Navision, Navision Financials, Dynamics NAV, and Business Central.
- Founded SimCrest, Inc. in 2001. Dallas, Miami, Los Angeles.
- Converted several customers from NAV to BC.
- Developed Six Apps currently on Business Central AppSource.



Agenda

- Using prefixes and making the best of it. Pitfalls when naming objects.
- Creating delta files from the old add-ons.
- Using Txt2AL tool to convert delta files to AL and fixing issues with code.
- Code compliance: permission sets, data classification, table and field names, upgrade and install code.
- How to use TempBlob for file management.
- Docker or Sandbox development?
- Standard use of Dates and Booleans.
- What about .Net variables? What to use instead.
- Making your own Apps extendable.
- Test apps. What do you need to cover?
- Monetization options.

Migration Checklist

1. Getting Object Range and Prefix
2. Convert Objects from C/AL to AL to create Extension
3. Create Test App

Full Microsoft Technical Validation Check List

https://docs.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-checklist-submission

Prefix and App Prefix

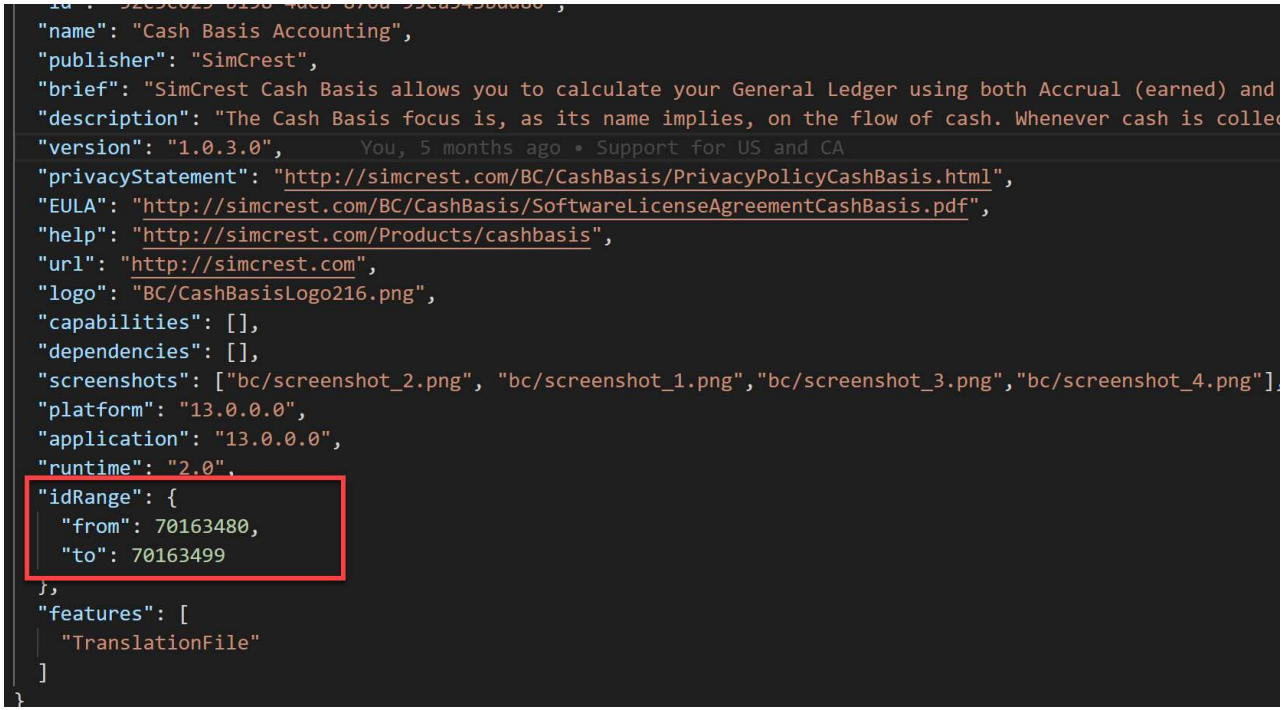
- Apply for Prefix
- Mandatory in Apps for
  - New Tables (name only)
  - New Pages (name only)
  - Table Extensions (name and fields)
  - Page Extensions (name and fields)
- Use App Prefix / Suffix
- Unique within company
- Avoid issues down the road

Tab-Ext70163481.SIMCCSHGLEntryExt.al

```
0 references
tableextension 70163481 "SIMC CSH G/L Entry Ext" extends "G/L Entry"
{

    fields
    {
        6 references
        field(70163480; "SIMC CSH Tax Entry"; Boolean)
        {
            Caption = 'Tax Entry';
            DataClassification = CustomerContent;
        }
        5 references
        field(70163481; "SIMC CSH Exclude In Cash"; Boolean)
        {
            Caption = 'Exclude In Cash';
            DataClassification = CustomerContent;
        }
    }
    keys
    {
        - reference
        Key(SIMCCSHKey1; "SIMC CSH Exclude In Cash", "SIMC CSH Tax Entry")
        {

        }
    }
}
```

Object Range

- Do not use whole range
- Specify per App
- Avoid overlaps



```
"name": "Cash Basis Accounting",
"publisher": "SimCrest",
"brief": "SimCrest Cash Basis allows you to calculate your General Ledger using both Accrual (earned) and
"description": "The Cash Basis focus is, as its name implies, on the flow of cash. Whenever cash is colle
"version": "1.0.3.0",      You, 5 months ago • Support for US and CA
"privacyStatement": "http://simcrest.com/BC/CashBasis/PrivacyPolicyCashBasis.html",
"EULA": "http://simcrest.com/BC/CashBasis/SoftwareLicenseAgreementCashBasis.pdf",
"help": "http://simcrest.com/Products/cashbasis",
"url": "http://simcrest.com",
"logo": "BC/CashBasisLogo216.png",
"capabilities": [],
"dependencies": [],
"screenshots": ["bc/screenshot_2.png", "bc/screenshot_1.png","bc/screenshot_3.png","bc/screenshot_4.png"],
"platform": "13.0.0.0",
"application": "13.0.0.0",
"runtime": "2.0",
"idRange": {
  "from": 70163480,
  "to": 70163499
},
"features": [
  "TranslationFile"
]
```

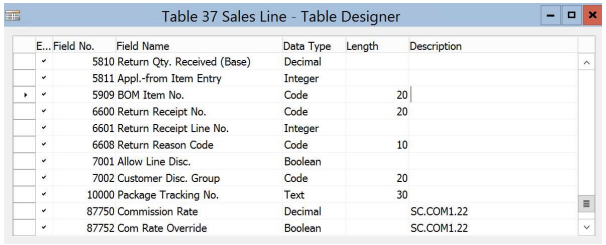
Check List: Object conversion from C/AL to AL

1. Export all objects from modified database into text file (modified.txt)
2. Export all objects from standard database into text file (original.txt)
3. Run script to split objects into separate files and compare
   Result: Delta files
4. Run txt2al tool to convert to AL files
5. Create new AL project in VS Code
6. Copy new AL files into VS Code project
7. Start fixing issues !!

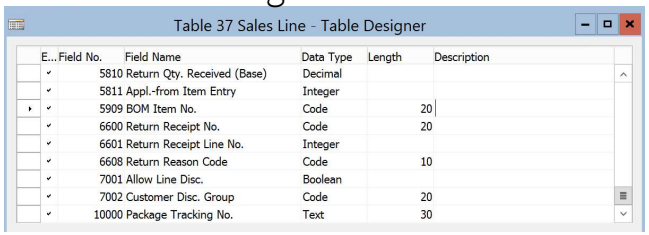
Delta Files

- Files containing the difference (delta) between the modified objects and the original objects

Modified



Original



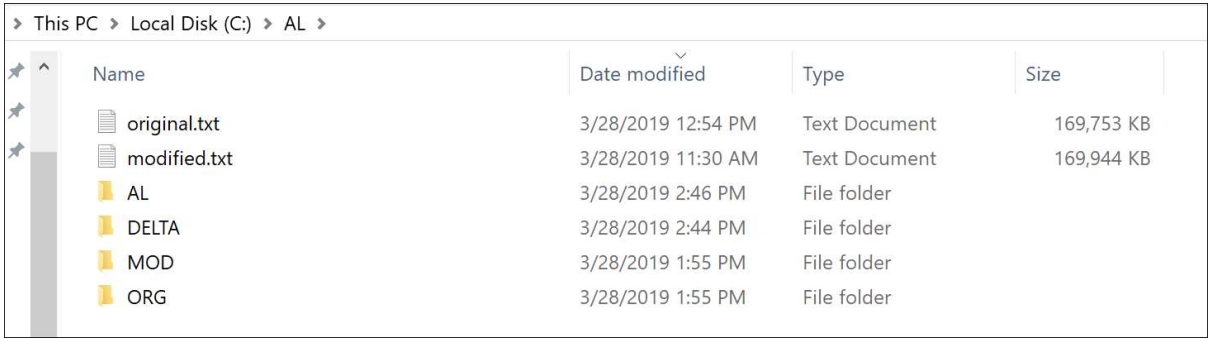
Delta

- We need the following objects in txt format:
  - Modified.txt, all objects from the database with the modifications
  - Original.txt, all objects from the database with no modifications
  - Must be the same NAV / BC version and CU

Delta File – Example Table 37

```
OBJECT Modification "Sales Line"(Table 37)
{
  OBJECT-PROPERTIES
  {
    Date=06/01/18;
    Time=10:46:53 AM;
    Modified=Yes;
    Version List=NAVW110.00,NAVNA10.00,SC.COM1.22;
  }
  PROPERTIES
  {
    Target="Sales Line"(Table 37);
  }
  CHANGES
  {
    { Insertion         ;InsertAfter="Package Tracking No."(Field 10000);
                         ChangedElements=FieldCollection
                         {
                           { 87750;  ;Commission Rate     ;Decimal       ;OnValidate=BEGIN
                                                                            TestStatusOpen;
                                                                            TESTFIELD("Quantity Invoiced",0);
                                                                            IF ("Commission Rate"<>xRec."Commission Rate") THEN
                                                                              "Com Rate Override":= TRUE;
                                                                          END;

                                                                           Description=SC.COM1.22 }
                           { 87752;  ;Com Rate Override   ;Boolean       ;OnValidate=BEGIN
                                                                            TestStatusOpen;
                                                                            TESTFIELD("Quantity Invoiced",0);
                                                                          END;

                                                                           Description=SC.COM1.22 }
                         }
```

Convert to Delta files - Script



Split and Compare to BC.ps1

```
# You may have to load this if running first time
# Set-ExecutionPolicy -ExecutionPolicy RemoteSigned

# Load Prerequisites
Import-Module 'C:\Program Files (x86)\Microsoft Dynamics 365 Business Central\130\RoleTailored Client\NavModelTools.ps1'

# Export objects from from base line
# Place original.txt as all baseline objects that are modified in MOD file
# Place modified.txt as baseline plus all modified objects that needs to be in extension
Split-NAVApplicationObjectFile -Source c:\al\original.txt -Destination 'c:\al\ORG' -Force
Split-NAVApplicationObjectFile -Source C:\al\modified.txt -Destination 'c:\al\MOD' -Force

# Compare
Compare-NAVApplicationObject -OriginalPath c:\al\ORG -ModifiedPath c:\al\MOD -DeltaPath c:\al\DELTA -Force -ExportToNewSyntax

# Create AL files
cd 'C:\Program Files (x86)\Microsoft Dynamics 365 Business Central\130\RoleTailored Client'
.\txt2al.exe --source=c:\al\delta --target=c:\al\al
```

Script is available on https://github.com/SimCrestInc/DirectionsLasVegas2019

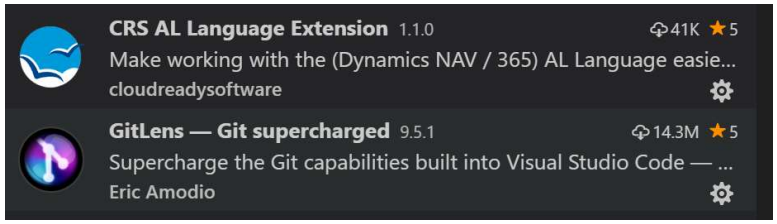
Fixing Issues in new AL project – VS Code

- Activate source code tracking (Git)
- .gitignore file (ignore *.al files, launch.json)
- Update app.json
- Copy and paste AL files into project
- Create settings.json
  - Auto Rename and Auto Prefix
- Create folders for related objects
- Fix tables first

Demo – Convert to AL and fix

Fixing issues – Time savers

- Install CRS AL Language
- Install Git Lens



- Settings.json prefix / rename

```
"CRS.ObjectNamePrefix": "SIMC ",
"CRS.OnSaveAlFileAction": "Rename"
```

- Ctrl-Space makes suggestions

Migration Scenario - Advanced

- Sales order pricing
  - Based on Shipment Date (not order or invoice date)
  - Should dependent on Check Mark in Sales & Receivable Setup

CU 7000: Sales Price Calc. Mgt.

```
CASE Type OF
  Type::Item:
    BEGIN
      Item.GET("No.");
      SalesLinePriceExists(SalesHeader,SalesLine,FALSE);
      CalcBestUnitPrice(TempSalesPrice);
      IF FoundSalesPrice OR
         NOT ((CalledByFieldNo = FIELDNO(Quantity)) OR
              (CalledByFieldNo = FIELDNO("Variant Code")))
      THEN BEGIN
        "Allow Line Disc." := TempSalesPrice."Allow Line Disc.";
        "Allow Invoice Disc." := TempSalesPrice."Allow Invoice Disc.";
        "Unit Price" := TempSalesPrice."Unit Price";
      END;
      IF NOT "Allow Line Disc." THEN
        "Line Discount %" := 0;
    END;
```

```
LOCAL SalesHeaderStartDate(SalesHeader : Record "Sales Header";VAR DateCaption : Text[30]) : Date
WITH SalesHeader DO
  IF "Document Type" IN ["Document Type"::Invoice,"Document Type"::"Credit Memo"] THEN BEGIN
    DateCaption := FIELDCAPTION("Posting Date");
    EXIT("Posting Date")
  END ELSE BEGIN
    DateCaption := FIELDCAPTION("Order Date");
    EXIT("Order Date");
  END;
```

Event Requests: https://github.com/microsoft/al/issues

```
608 [External] SalesLinePriceExists(SalesHeader : Record "Sales Header";VAR SalesLine : Record "Sales Line";ShowAll : Boolean) : Boolean
609 WITH SalesLine DO
610   IF (Type = Type::Item) AND Item.GET("No.") THEN BEGIN
611     IsHandled := FALSE;
612     OnBeforeSalesLinePriceExists(
613       SalesLine,SalesHeader,TempSalesPrice,Currency,CurrencyFactor,
614       SalesHeaderStartDate(SalesHeader,DateCaption),Qty,QtyPerUOM,ShowAll,IsHandled);
615     IF NOT IsHandled THEN BEGIN
616       FindSalesPrice(
617         TempSalesPrice,GetCustNoForSalesHeader(SalesHeader),SalesHeader."Bill-to Contact No.",
618         "Customer Price Group",'',"No.","Variant Code","Unit of Measure Code",
619         SalesHeader."Currency Code",SalesHeaderStartDate(SalesHeader,DateCaption),ShowAll);
620       OnAfterSalesLinePriceExists(SalesLine,SalesHeader,TempSalesPrice,ShowAll);
621     END;
622     EXIT(TempSalesPrice.FINDFIRST);
623   END;
624 EXIT(FALSE);
```

Extension Solution

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"Sales Price Calc. Mgt.", 'OnBeforeSalesLinePriceExists', '', true, true)]
0 references
local procedure BeforeSalesLinePriceIsCalculate(VAR SalesLine: Record "Sales Line"; VAR SalesHeader: Record "Sales Header";
                                                VAR TempSalesPrice: Record "Sales Price"; var InHandled: Boolean; ShowAll: Boolean)
var
    SalesPriceCalcMgmt: Codeunit "Sales Price Calc. Mgt.";
    SalesReceivableSetup: Record "Sales & Receivables Setup";
begin
    SalesReceivableSetup.Get();
    if not SalesReceivableSetup."Calculate Price From Shipment Date" then
        exit;

    // Use Shipment Date
    with SalesLine do begin
        SalesPriceCalcMgmt.FindSalesPrice(TempSalesPrice, SalesHeader."Sell-to Customer No.", SalesHeader."Bill-to Contact No.",
                                          "Customer Price Group", '', "No.", "Variant Code", "Unit of Measure Code",
                                          SalesHeader."Currency Code", "Shipment Date", ShowAll);
    end;
    InHandled := true;
end;
```

```
608 [External] SalesLinePriceExists(SalesHeader : Record "Sales Header";VAR SalesLine : Record "Sales Line";ShowAll : Boolean) : Boolean
609 WITH SalesLine DO
610   IF (Type = Type::Item) AND Item.GET("No.") THEN BEGIN
611     IsHandled := FALSE;
612     OnBeforeSalesLinePriceExists(
613       SalesLine,SalesHeader,TempSalesPrice,Currency,CurrencyFactor,
614       SalesHeaderStartDate(SalesHeader,DateCaption),Qty,QtyPerUOM,ShowAll,IsHandled);
615     IF NOT IsHandled THEN BEGIN
616       FindSalesPrice(
617         TempSalesPrice,GetCustNoForSalesHeader(SalesHeader),SalesHeader."Bill-to Contact No.",
618         "Customer Price Group",'',"No.","Variant Code","Unit of Measure Code",
619         SalesHeader."Currency Code",SalesHeaderStartDate(SalesHeader,DateCaption),ShowAll);
620       OnAfterSalesLinePriceExists(SalesLine,SalesHeader,TempSalesPrice,ShowAll);
621     END;
622     EXIT(TempSalesPrice.FINDFIRST);
623   END;
624 EXIT(FALSE);
```

Install Code

- Runs every time the extension is installed

```
codeunit 70163504 "SIMC CNF Install"
{
    Subtype = Install;

    var
        1 reference
        CNFSetup: Record "SIMC CNF Configurator Setup";

    trigger OnInstallAppPerCompany()
    begin
        CNFSetup.CreateStandardSetup();       You, a minu
    end;

    trigger OnInstallAppPerDatabase()
    begin
    end;
}
```

```
4 references
procedure CreateStandardSetup()       You, 6 months ago • cah
var
    NoSeries: Record "No. Series";
    NoSeriesLine: Record "No. Series Line";
    CNFSub: Codeunit "SIMC CNF Subscription";
begin
    If not Get then begin
        "Primary Key" := '';
        "Expiration Date" := CalcDate('30D', Today());
        "Trial Period" := true;
        "Activation Code" := CNFSub.CreateTrialActivationCode("Expiration Date");
        with NoSeries do begin
            Code := 'SIMC-CFITM';
            Description := 'Configurator Item-Bom No.';
            "Default Nos." := true;
            // Insert if possible
            if insert then;
        end;
        with NoSeriesLine do begin
            "Series Code" := NoSeries.Code;
```

Upgrade Code

- Runs every time the extension is upgraded

```
codeunit 70163505 "SIMC CNF Upgrade"
{
    Subtype = Upgrade;

    var
        1 reference
        CNFSetup: Record "SIMC CNF Configurator Setup";

    trigger OnUpgradePerCompany()
    begin
        CNFSetup.CreateStandardSetup();
    end;

    trigger OnUpgradePerDatabase()
    begin
    end;
}       You, 4 months ago • Updated install and update
```

```
4 references
procedure CreateStandardSetup()       You, 6 months ago • cah
var
    NoSeries: Record "No. Series";
    NoSeriesLine: Record "No. Series Line";
    CNFSub: Codeunit "SIMC CNF Subscription";
begin
    If not Get then begin
        "Primary Key" := '';
        "Expiration Date" := CalcDate('30D', Today());
        "Trial Period" := true;
        "Activation Code" := CNFSub.CreateTrialActivationCode("Expiration Date");
        with NoSeries do begin
            Code := 'SIMC-CFITM';
            Description := 'Configurator Item-Bom No.';
            "Default Nos." := true;
            // Insert if possible
            if insert then;
        end;
        with NoSeriesLine do begin
            "Series Code" := NoSeries.Code;
```

Permissions Sets

- Permissions related to the extension should be maintained in the extension itself:

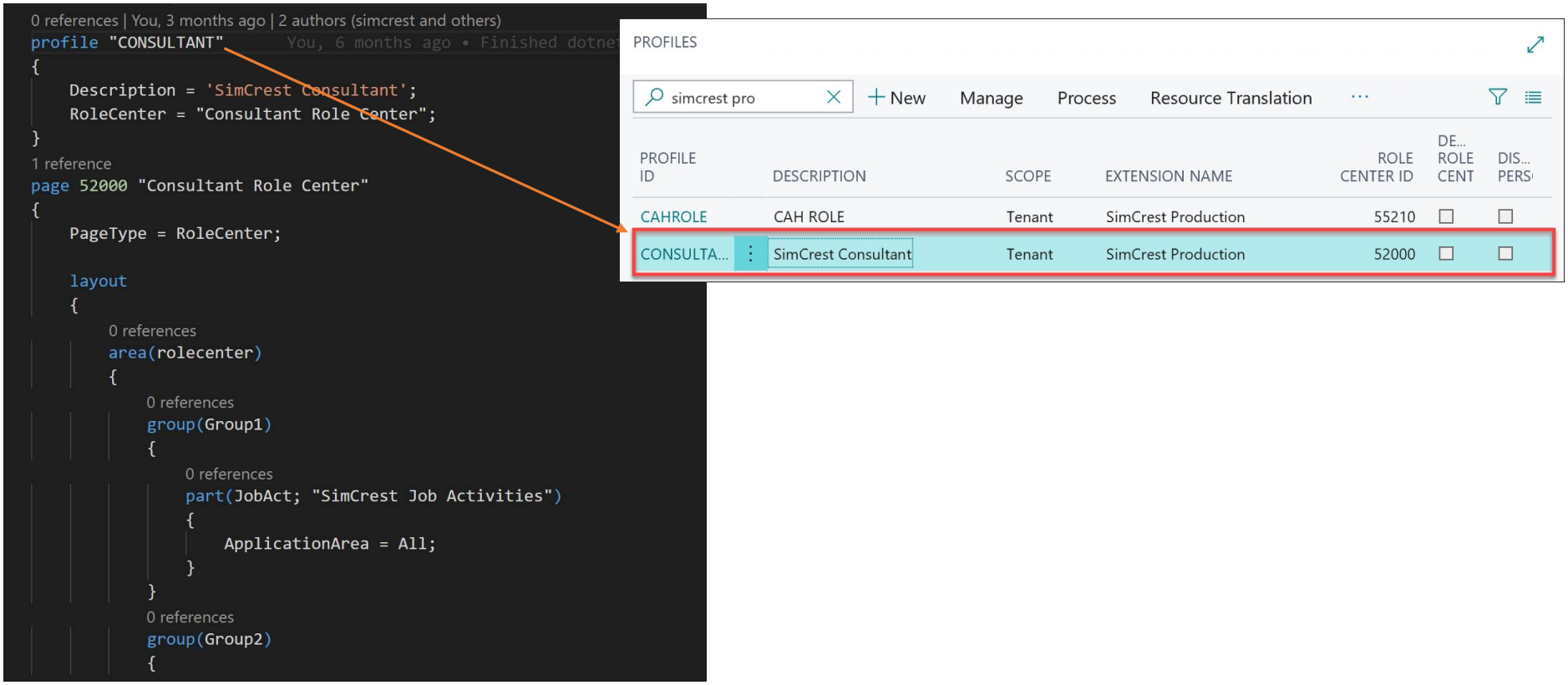
```
<PermissionSets>        You, 6 months ago • cah
    <PermissionSet RoleID="SIMC PROD CONFIG" RoleName="SimCrest Product Configurator">
        <Permission>
            <ObjectType>TableData</ObjectType>
            <ObjectID>70163500</ObjectID> <!-- Question -->
            <ReadPermission>1</ReadPermission>
            <InsertPermission>1</InsertPermission>
            <ModifyPermission>1</ModifyPermission>
            <DeletePermission>1</DeletePermission>
            <ExecutePermission>0</ExecutePermission>
            <SecurityFilter />
        </Permission>
        <Permission>
            <ObjectType>TableData</ObjectType>
            <ObjectID>70163501</ObjectID> <!-- Answer -->
            <ReadPermission>1</ReadPermission>
            <InsertPermission>1</InsertPermission>
            <ModifyPermission>1</ModifyPermission>
            <DeletePermission>1</DeletePermission>
            <ExecutePermission>0</ExecutePermission>
            <SecurityFilter />
        </Permission>
        <Permission>
            <ObjectType>TableData</ObjectType>
            <ObjectID>70163502</ObjectID> <!-- Trigger -->
```

simc | Page

PERMISSION SET LOOKUP

ROLE ID	NAME	EXTENSION NAME	SCOPE
SIMC PROD CONF SE...	SimCrest Product Configurator	Product Configurator	Tenant
SIMC PROD CONFIG	SimCrest Product Configurator	Product Configurator	Tenant
SIMC AUTOEMAIL SE...	SimCrest Auto Email Setup	Auto Email	Tenant
SIMCREST AUTOEMAIL	SimCrest Auto Email	Auto Email	Tenant

Profiles and Role Centers in Extensions



.Net in BC Cloud

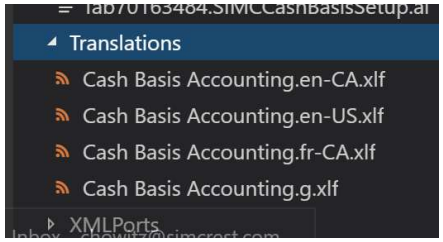
Available APIs:

- HTTP
- JSON
- TextBuilder
- XML

Class	Description
HttpClient	Provides a base class for sending HTTP requests and receiving HTTP responses from a resource identified by a Uniform Resource Identifier (URI).
HttpContent	A base class representing an HTTP entity body and content headers.
HttpHeaders	The HttpHeaders class contains a collection of headers and their values.
HttpRequestMessage	Represents an HTTP request message.
HttpReponseMessage	Represents an HTTP response message.
JsonArray	JsonArray is a container for any well-formed JSON array. A default JsonArray contains an empty JSON array.
JsonObject	JsonObject object is a container for any well-formed JSON object. A default JsonObject contains an empty JSON object.
JsonToken	JsonToken object is a container for any well-formed JSON data. A default JsonToken contains the JSON value of NULL.
JsonValue	JsonValue object is a container for any well-formed JSON object. A default JsonValue is set to the JSON value of NULL.
TextBuilder	TextBuilder can performantly concatenate multiple bigger strings together.
XmlAttribute	Represents an attribute.
XmlAttributeCollection	Represents a collection of attributes associated with an XmlElement.
XmlCData	Represents the CDATA section.
XmlComment	Represents the content of an XML comment.
XmlDeclaration	Represents the XML declaration node.
XmlDocument	Represents an XML document. This class can be used to load, validate, edit, add, and position XML in a document.
XmlDocumentType	Represents the document type declaration.
XmlElement	Represents an element.
XmlNamespaceManager	Resolves, adds, and removes namespaces to a collection and provides scope management for these namespaces.
XmlNameTable	Table of automized string object.
XmlNode	Represents a single node in the XML document.
XmlNodeList	Represents an ordered collection of nodes.
XmlProcessingInstruction	Represents a processing instruction, which XML defines to keep processor-specific information in the text of the document.
XmlText	Represents the text content of an element or attribute.

Translation Files

- Check requirements for each country
- Use MS Multilingual Editor with Translator Text (Azure)

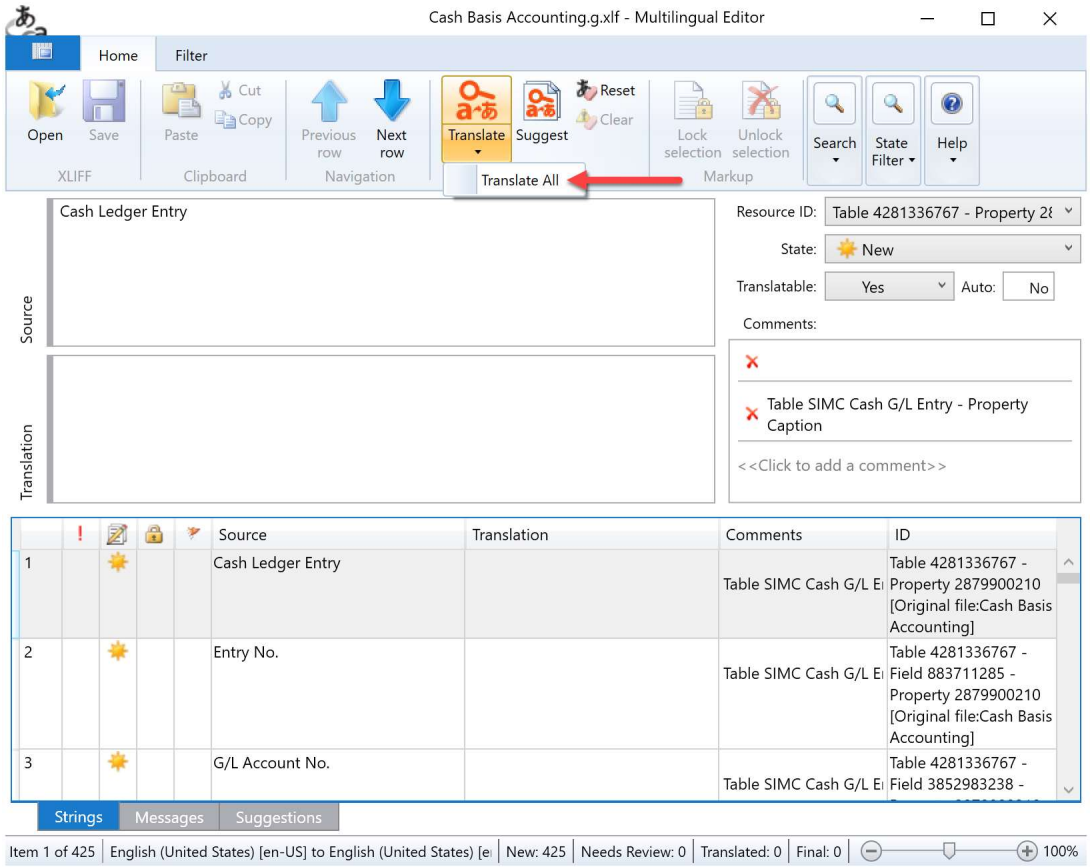


Country/Region	Language Code
Australia	en-AU
Austria	de-AT
Belgium	nl-BE fr-BE
Canada	en-CA fr-CA
Czechia **(on-premises only)**	cs-CZ
Denmark	da-DK
Germany	de-DE
Finland	fi-FI
France	fr-FR

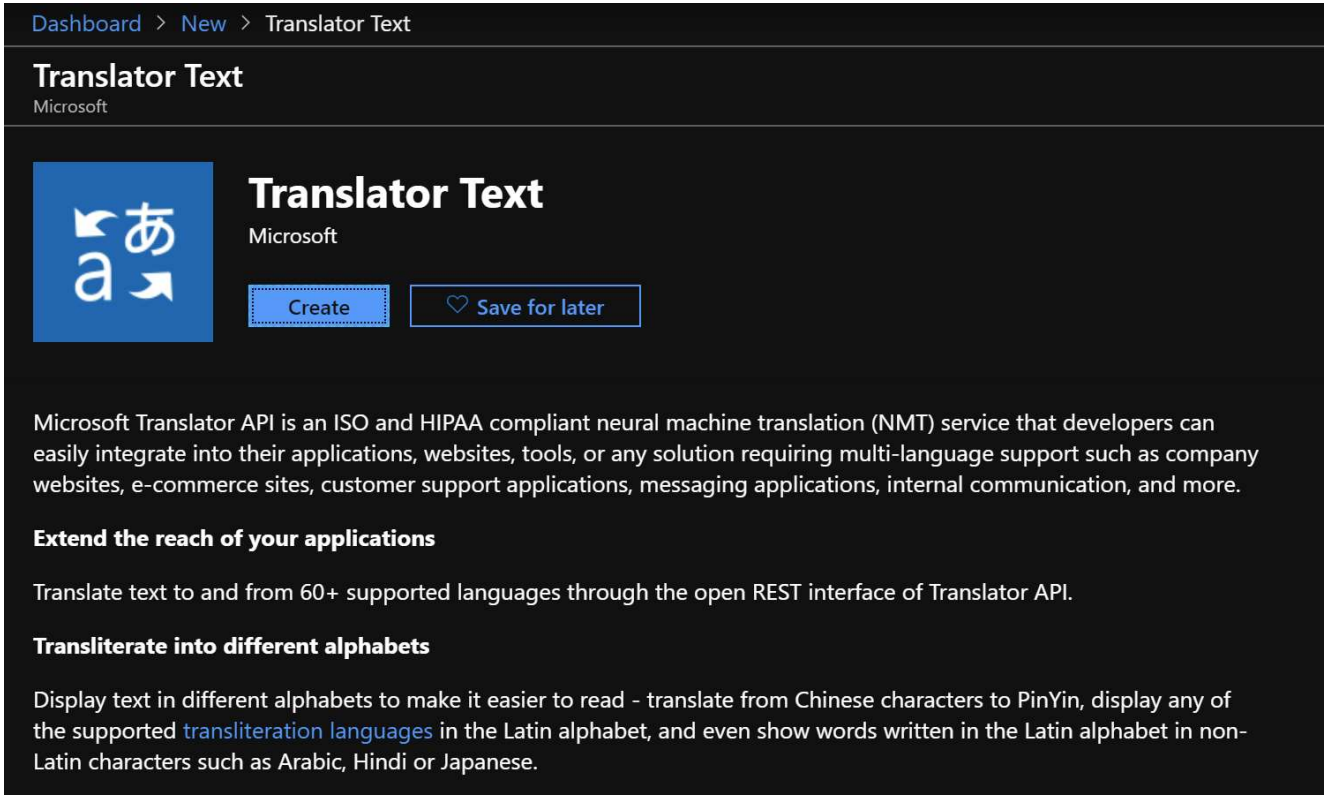
https://docs.microsoft.com/en-us/dynamics365/business-central/dev-itpro/compliance/apptest-countries-and-translations

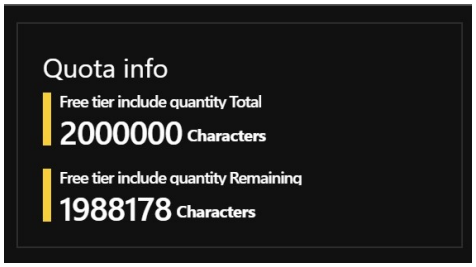
Multilingual Editor

- Use Translator Text (Azure)



Translator API (Azure)





Testing and Debugging (F5)

- Use Docker
  - Easy change of Country version. A must for Apps supporting multiple countries
  - Test against future versions
- Use Cloud Sandbox
  - Probably fastest and most accessible
  - Easy access from anywhere

Making your own App extendable

- You should create events in your own apps to make them extendable
- Find key parts of Apps you can allow partners to extend
- Allow partners to suggest events in your Apps
- Ideally your first release should have key events
- Write documentation for events and post example on Github

- See example for Auto Email App.

Auto Email Example

- Email Sales documents like quotes, orders, invoices, cr. Memos, statements etc. PO's also supported.
- Emails are logged to a log table and processed.
- What can be extended?
  - Pretty much anything with a unique document no. or record number (production order, fixed asset etc. etc.)

Auto Email: Document Type (Enum)

- Enums are extendable and works like Options

```
enum 70163325 "SIMC AEM Document Type"
{
    Extensible = true;

    value(0; SalesQuote) { Caption = 'Sales Quote'; }

    value(1; SalesOrder) { Caption = 'Sales Order'; }

    value(2; SalesReturnOrder) { Caption = 'Sales Return Order'; }

    value(3; SalesInvoice) { Caption = 'Sales Invoice'; }

    value(4; SalesCreditMemo) { Caption = 'Sales Credit Memo'; }

    value(5; ServiceInvoice) { Caption = 'Service Invoice'; }

    value(6; ServiceCrMemo) { Caption = 'Service Cr. Memo'; }

    value(7; PurchaseOrder) { Caption = 'Purchase Order'; }

    value(8; Statement) { Caption = 'Statement'; }

    value(9; Shortpaid) { Caption = 'Shortpaid'; }

    value(10; Collection) { Caption = 'Collection'; }       You, 4 m
}
```

```
1  enumextension 50100 "DocTypeExt" extends "SIMC AEM Document Type"
2  {
3      value(50000; Deposit)
4      {
5          Caption = 'Deposit';
6      }
7  }
```

Before Logging Email Event

```
    DocType::Collection:
        BEGIN
            Cust.GET(DocNo);
            SetupEmails(0, DocNo, Emailto, ccEmailTo, bccEmailTo, TriggerError);
            IF TriggerError THEN
                AutoEmailLog."Error Code" := 'Not set up to receive collections by email';
            EmailTemplate.Get(AutoEmailSetup."Collection Template");
            AutoEmailLog.Subject := StrSubstNo(EmailTemplate."Email Subject", Cust.Name);
            TriggerError := Cust.CalcOverdueBalance = 0;
            IF TriggerError AND ShowResult THEN
                ERROR('Customer has no overdue balance');
            IF TriggerError AND NOT ShowResult THEN
                AutoEmailLog."Error Code" := 'Customer has no overdue balance';
        END;
    else
        // All extended document types are handled here.
        OnBeforeLogSpecialDocType(DocType, DocNo, EmailTo, ccEmailTo, bccEmailTo, TriggerError, EmailTemplate, AutoEmailLog);
END;

AutoEmailLog."Entry No." := NextEntryNo;
AutoEmailLog."Document Type" := DocType;
AutoEmailLog."Document No." := DocNo;
AutoEmailLog."Email Template" := EmailTemplate.Code;
```

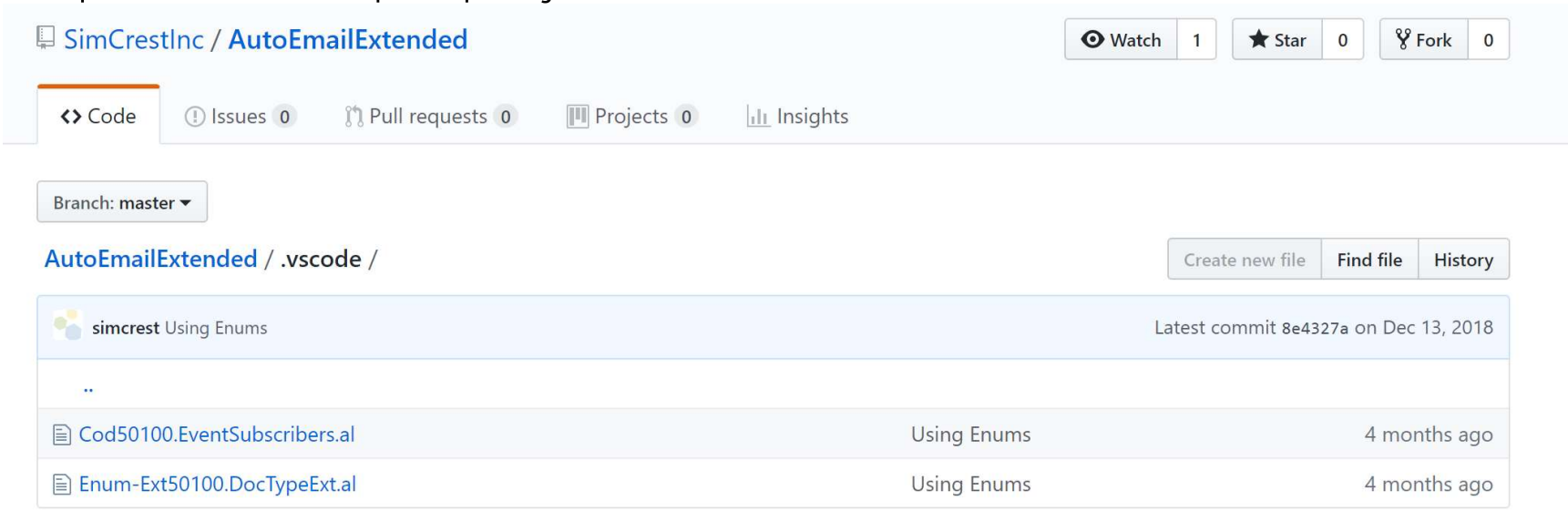
Before Document is Emailed Event

- Make sure RecRef, Email Template, Filename, and merge fields are set

```
                    FileIndex += 1;
                UNTIL CustLedgerEntry.NEXT = 0;
            MergeField1 := Customer."No.";
            MergeField2 := Customer.Name;
        END;
    else
        // Custom Enums
        begin
            OnBeforeDocumentEmailed(AutoEmailLog."Document Type", RecRef, AutoEmailLog, EmailTemplate, FileNameArray[1], MergeField1, MergeField2);
            TempBlob.Blob.CreateOutStream(Attachment);
            Report.SaveAs(EmailTemplate."Report No.", '', ReportFormat::Pdf, Attachment, RecRef);
            TempBlob.Blob.CreateInStream(AttachmentArray[1]);
        end;
END;

// Start load AEM attachments
AutoEmailAttachmentRec.SetRange("Email Type", AutoEmailLog."Document Type");
AutoEmailAttachmentRec.SetRange(Active, true);
If AutoEmailAttachmentRec.FindSet() then
```

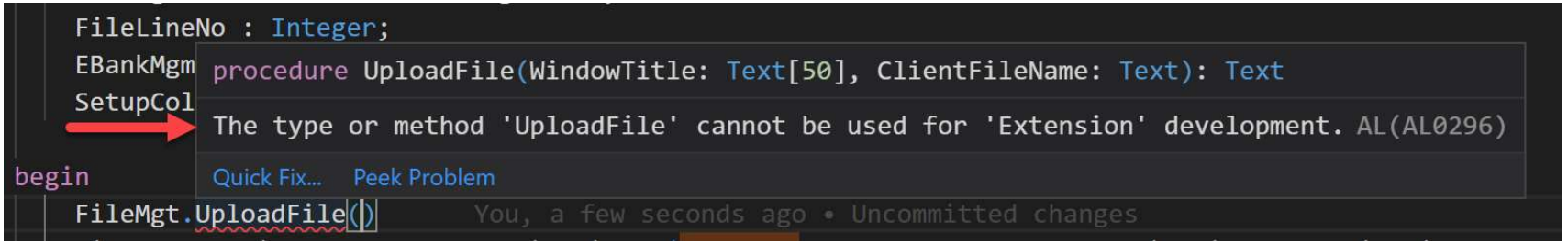
Suggestions

- Create a developer's manual listing all events available and how to use them.
- Post public example project on Github

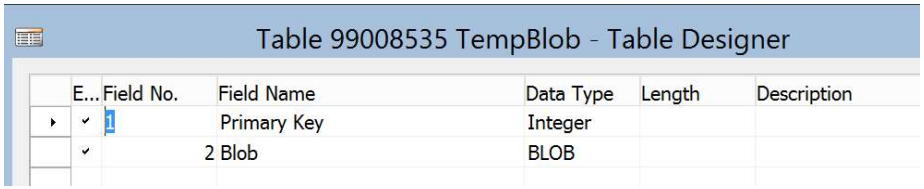


File Management in the Cloud

- Sending files to and from the Service Tier is not allowed in the Cloud



- So what can we use instead? TempBlob:

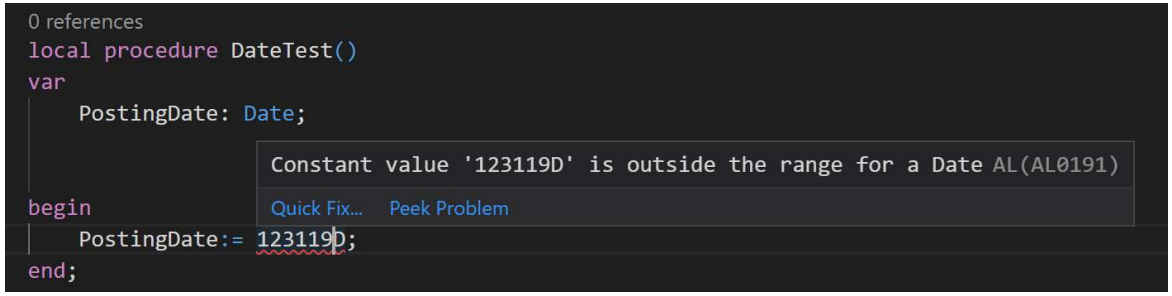


TempBlob Example

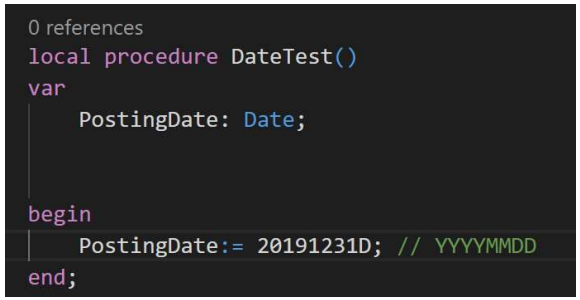
```
1 reference
procedure Import(BankAccRecon: Record "SIMC E-Bank Reconciliation")
var
    EbankSetup : Record "SIMC E-Bank Bank Setup";
    LineString: Text;
    NextLineNo: Integer;
    FileName: Text;
    ImportBankStmtTxt: Label 'Select a file to import';
    FileFilterTxt: Label 'Text Files(*.txt;*.csv)|*.txt;*.csv';
    FileFilterExtensionTxt: Label 'txt,csv';
    TempBlob: Record TempBlob;
    IStream: InStream;
    FileMgt: Codeunit "File Management";
    FileLineNo : Integer;
    EBankMgmt : Codeunit "SIMC E-Bank Management";
    SetupColumn : array [8] of Option "None","Date","Document No.",Description,Amount,Debit,Credit;

begin
    FileName:= FileMgt.BLOBImportWithFilter(TempBlob,ImportBankStmtTxt,'',FileFilterTxt,FileFilterExtensionTxt);
    If FileName='' then
      exit;
    EbankSetup.GET(BankAccRecon."Bank Account No.");
    NextLineNo:= 10000;
    FileLineNo:= 1;
    LoadSetupColumns(BankAccRecon."Bank Account No.",SetupColumn);
    EBankMgmt.DeleteEBankRecsForBank(BankAccRecon);
    TempBlob.Blob.CreateInStream(IStream);
    while IStream.ReadText(LineString)>0 do begin
      IF FileLineNo>EbankSetup."Ignore first X lines" THEN begin
        ImportCSV(LineString,BankAccRecon,NextLineNo,FileLineNo,SetupColumn);
        NextLineNo+= 10000;
      end;
      FileLineNo+= 1;
    end;
end;
```

New Date format in AL

- Converted Code:



- New Format:



New Booleans in Reports / XML Ports

- Now consistent use of true / false (instead of Yes / No):

```
report 50040 "Consultant Sales Statistics"
{
    DefaultLayout = RDLC;
    RDLCLayout = './Reports/Consultant Sales Statistics.rdl';
    UsageCategory = ReportsAndAnalysis;
    ApplicationArea = All;

    dataset
    {
        6 references
        dataitem("Job Ledger Entry";"Job Ledger Entry")
        {
            DataItemTableView = SORTING("No.","Posting Date","Entry Type",Type)
                                WHERE(Chargeable=CONST(Yes));
            RequestFilterFields = "Posting Date","No.";
            column(FORMAT_TODAY_0_4_;FORMAT(TODAY,0,4))
```

```
= SORTING("No.","Posting Date","Entry Type",Type)
  WHERE(Chargeable=CONST(true));
ds = "Posting Date","No.";
```

Test Apps

- You must submit a Test App with you Add-on App
- Must cover “80% of the code”
- They are a BIG hassle ☹ but

- Test Apps are excellent for internal testing
- Recurring and tedious testing can be done consistently and with ease once test scripts are developed

Test Initialization

```
17 references
local procedure Initialize();
var
    AEL: Record "SIMC Auto Email Log";
    Customer: Record Customer;
    AutoEmailSetup: Record "SIMC Auto Email Setup";

begin
    AutoEmailSetup.CreateStandardSetup(false);
    AutoEmailSetup.Get();
    AutoEmailSetup."Trial Period" := true;
    AutoEmailSetup."Expiration Date" := today + 30;
    AutoEmailSetup."Do not send emails for testing" := true;
    AutoEmailSetup.Modify();

    AEL.DeleteAll;

    Customer.get('10000');
    Customer."SIMC Email Documents" := true;
    Customer."SIMC Email To" := 'johndoe1@contoso.com';
    Customer.Modify();
    Customer.get('20000');
    Customer."SIMC Email Documents" := true;
    Customer."SIMC Email To" := 'johndoe2@contoso.com';
    Customer.Modify();
end;
```

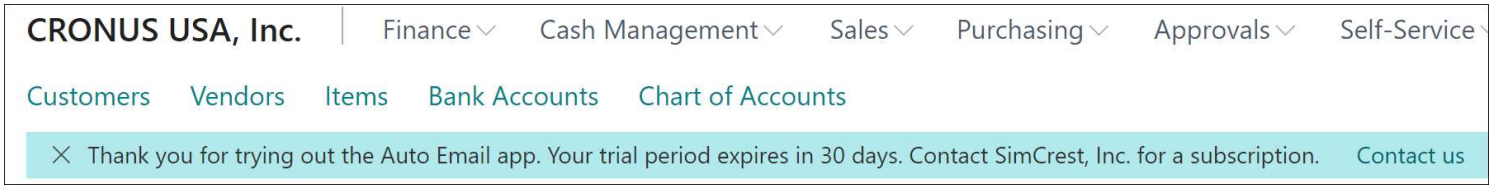
Test Example 1

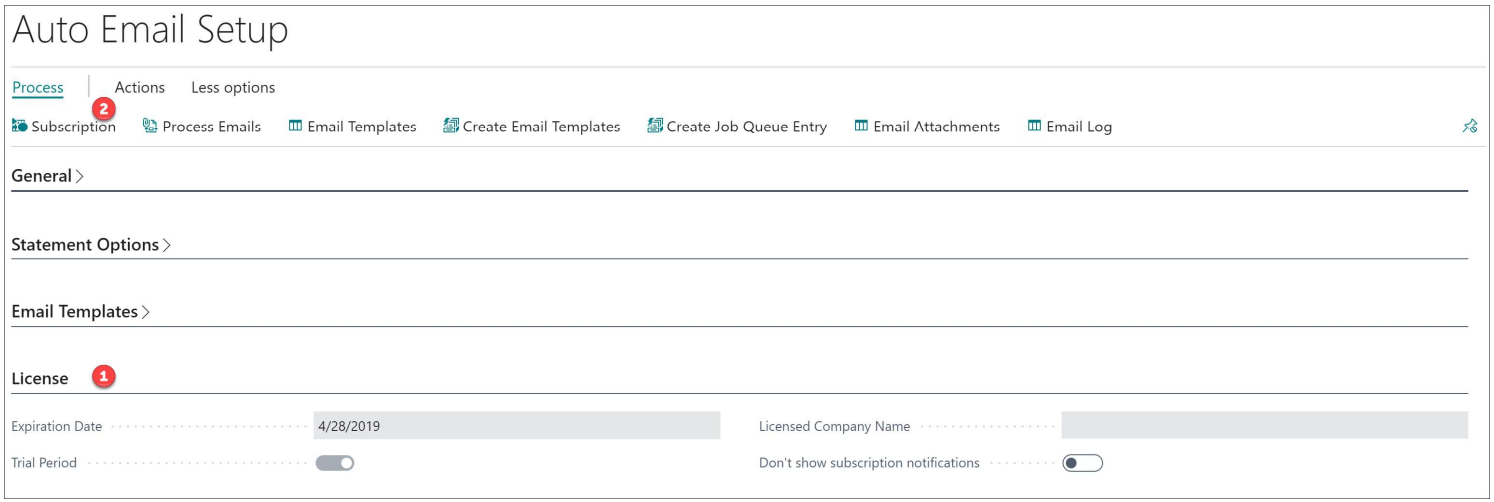
```
[Test]
0 references
procedure TestSalesOrderEmailed()
var
    SalesHeader: Record "Sales Header";
    AutoEmailMgmt: Codeunit "SIMC Auto Email Management";
    AEL: Record "SIMC Auto Email Log";
    FoundAEL: Record "SIMC Auto Email Log";

begin
    // Scenario 1 log entry created for sales order when logged to email
    Initialize();
    Commit;
    LibraryLowerPermissions.SetO365BusFull();
    // Given Create Sales Order
    LibrarySales.CreateSalesHeader(SalesHeader, SalesHeader."Document Type"::Order, '10000');
    // When Log Email
    AutoEmailMgmt.LogEmail(AEL."Document Type"::"Sales Order", SalesHeader."No.", false);
    AutoEmailMgmt.SendEmails();
    // Then 1 log entry with correct document type and number
    FoundAEL.FindFirst;
    Assert.AreEqual(1, FoundAEL.Count, 'Auto Email log must have 1 record.');
    Assert.AreEqual(AEL."Document Type"::"Sales Order",FoundAEL."Document Type", 'Auto Email Log Doc Type = Sales Order');
    Assert.AreEqual(SalesHeader."No.",FoundAEL."Document No.",'Auto Email Log Sales Order No.  = Found Sales Order No.')
end;
```

Test Example 2

```
[Test]
0 references
Procedure TestTrialExpiredSubsciption()
var
    AutoEmailSub: Codeunit "SIMC Auto Email Subscription";
    AutoEmailSetup: Record "SIMC Auto Email Setup";
    SubStatus: Option InTrial, TrialExpired, SubExpires, SubExpired, CompanyError, SubActive;
    SubStatus2: Option InTrial, TrialExpired, SubExpires, SubExpired, CompanyError, SubActive;
    DaysLeft: Integer;

begin
    Initialize();
    // Scenario Trial Subscription Enabled and expired
    // Given Today's date, check status
    AutoEmailSetup."Trial Period":= true;
    AutoEmailSetup."Expiration Date":= today-30;
    AutoEmailSetup.Modify();
    Commit;
    LibraryLowerPermissions.SetO365Full();

    // When Role Center is opened
    // Then Status=TrialExpired;
    SubStatus2:= SubStatus2::TrialExpired;
    AutoEmailSub.GetSubscriptionStatus(Today,SubStatus,DaysLeft);
    Assert.AreEqual(SubStatus,SubStatus2,'Status should be Trial Expired');
end;
```

Monetization

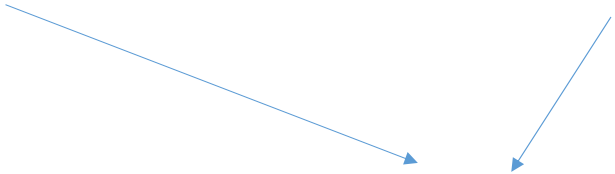




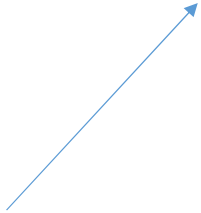
Monetization: Activation Code

App Code: AEM (Auto Email)

Company Name: CRONUS USA Inc.

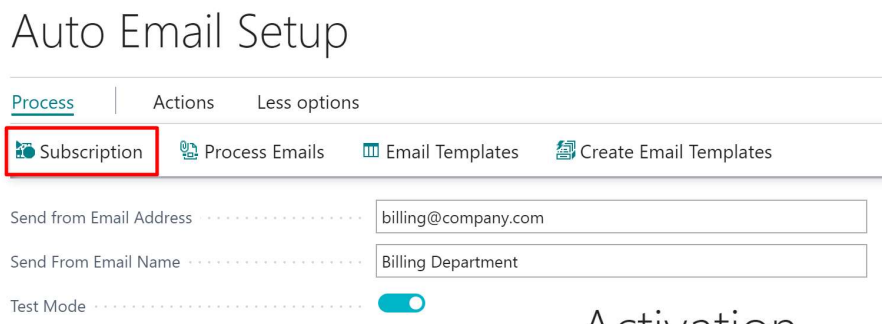


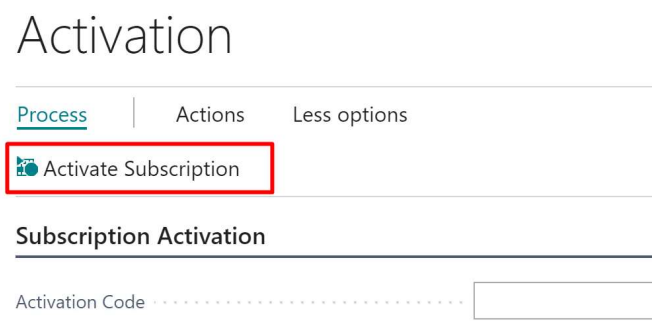
Activation code: 9384kskhf$3#2>,73J9aq0-6



Expiration Date: 05/05/2020

Monetization Activation





Monetization – Check if product enabled

```
2 references
procedure CheckProductEnabled(): Boolean;
var
    DaysLeft: Integer;
    ErrorMessage: Text;
    IssueDetected: Boolean;
    SubStatus: Option InTrial,TrialExpired,SubExpires,SubExpired,CompanyError,SubActive,ActivationError,TrialError;
begin
    GetSubscriptionStatus(Today(), SubStatus, DaysLeft);
    case SubStatus of
        SubStatus::TrialExpired:
            Error(TrialExpiredMsg);
        SubStatus::TrialError:
            Error(TrialCodeErrorMsg, AppName);
        SubStatus::SubExpired:
            Error(RegularExpiredMsg);
        SubStatus::CompanyError:
            begin
                ProductSetup.Get();
                Error(CompanyErrorMsg, AppName, ProductSetup."Licensed Company Name");
            end;
        SubStatus::ActivationError:
            CheckActivationCode(true, IssueDetected, ErrorMessage);
    end;
    exit(true);
end;
```

Let’s Stay Connected

chowitz@simcrest.com

This presentation is available at https://github.com/SimCrestInc/DirectionsLasVegas2019