



# What's new in developer tools



# Agenda

- VS Code Extension
- Namespace
- ALDoc
- Investigate and troubleshoot
- AL Language
- Code authoring
- GitHub Copilot Chat
- AppSource

# Agenda

VS Code Extension

Namespace

ALDoc

Investigate and troubleshoot

AL Language

Code authoring

GitHub Copilot Chat

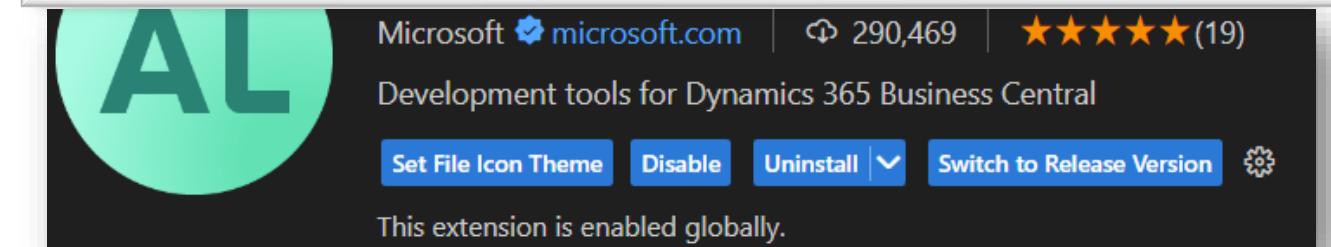
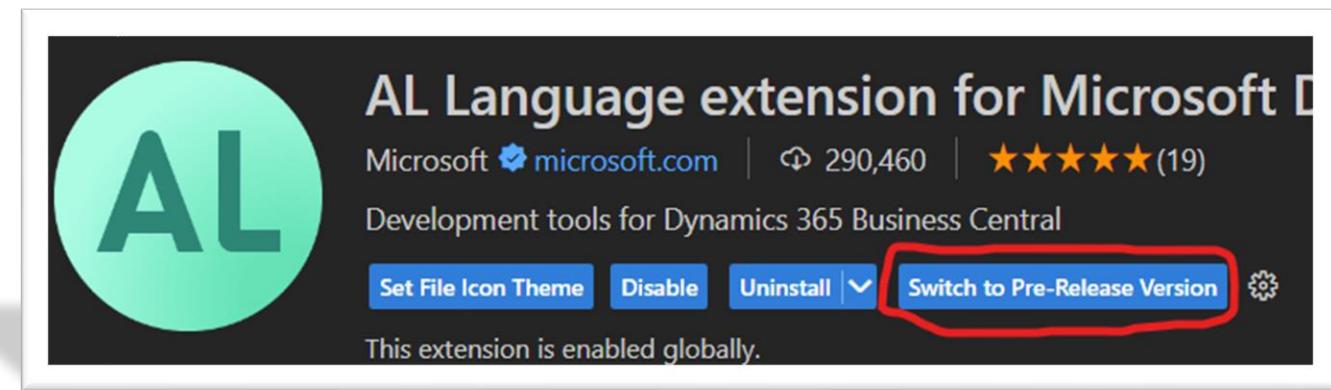
AppSource

# AL Language Extension - Pre-Release

Use Pre-Release to test or preview future functionality

Avoid manual install from BC insider/preview builds

Early access to bug fixes



# AL Language Extension on Linux (preview)

Use AL Language extension your favorite Linux distribution

Including Windows Subsystem for Linux (WSL)

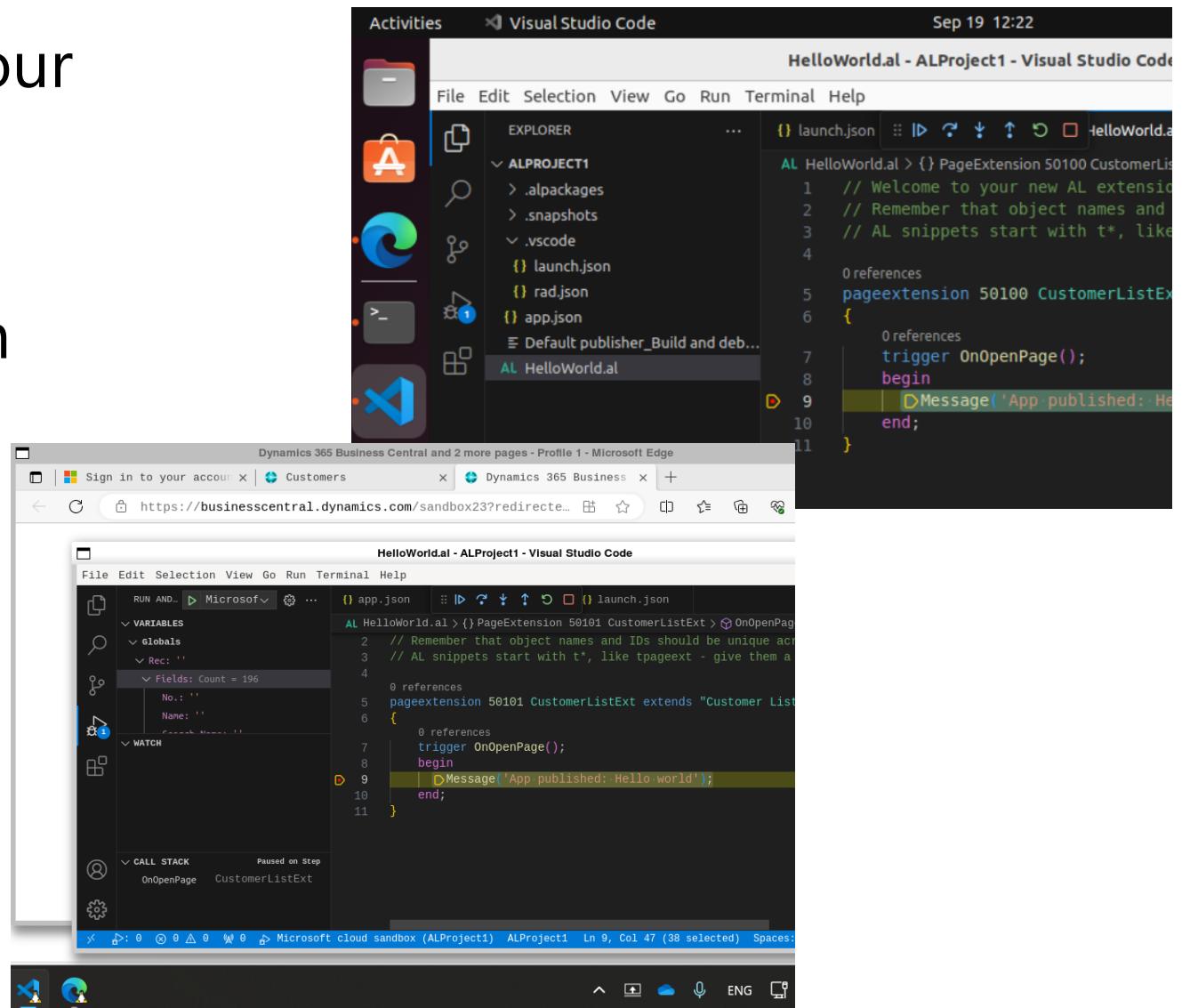
[github.com/al/issues](https://github.com/al/issues)

Requires .NET

[Install .NET on Linux distributions - .NET | Microsoft Learn](https://learn.microsoft.com/en-us/dotnet/learn/distributions/)

[Running Visual Studio Code on Linux](#)

[Download Microsoft Edge](#)



The screenshot shows two instances of Visual Studio Code running on a Linux desktop. The top instance is titled "HelloWorld.al - ALProject1 - Visual Studio Code" and displays an AL code editor with the following content:

```
// Welcome to your new AL extension
// Remember that object names and IDs should be unique across
// AL snippets start with t*, like tpageext - give them a
// meaningful name instead.
0 references
pageextension 50100 CustomerListExt
{
    0 references
    trigger OnOpenPage();
    begin
        Message('App published: Hello-world');
    end;
}
```

The bottom instance is also titled "HelloWorld.al - ALProject1 - Visual Studio Code" and shows the Explorer, RUN AND\_, VARIABLES, WATCH, and CALL STACK panes. The VARIABLES pane lists a "Rec" variable with a value of an array containing 196 items. The WATCH pane shows the same array. The CALL STACK pane indicates the code is paused on a step.

# Syntax highlighting in DevOps

Syntax highlighting for AL in Azure DevOps can be enabled by installing the following free extension in Azure DevOps:

<https://marketplace.visualstudio.com/items?itemName=Dynamics365BusinessCentral.allanghighlights>

The screenshot shows the product page for the 'AL Language Syntax Highlights for Microsoft Dynamics 365 Business Central' extension. It features a large green circular icon with the letters 'AL'. The title is 'AL Language Syntax Highlights for Microsoft Dynamics 365 Business Central'. Below the title, it says 'Reports | Manage' and 'Microsoft microsoft.com'. It has a star rating of '(0)' and is labeled 'Free'. A description states: 'Enables syntax highlighting for \*.al files in Azure DevOps. The extension should be compatible with the latest version of the AL Language extension for Microsoft Dynamics 365 Business Central.' A green button at the bottom left says 'Get it free'.

```
1 namespace MyAwesomeAIApp.Demo;
2
3 using Microsoft.Sales.Customer;
4
5 pageextension 54321 "Customer Card Page Ext" extends "Customer Card"
6 {
7     layout
8     {
9         addlast(General)
10        {
11            group("Group with Rich Text Editor")
12            {
13                // Remember to check out all the other fantastic developer additions, like the rich text editor
14                field(Spoilers; GlobalText)
15                {
16                    Multiline = true;
17                    ExtendedDataType = RichContent;
18
19                    trigger OnValidate()
20                    begin
21                        // Here you should persist your GlobalText variable by using the Streams pattern
22                    end;
23                }
24            }
25        }
26    }
27
28    actions
29    {
30        addlast("&Customer")
31        {
32            action("Show Greeting")
33            {
34                Image = CheckDuplicates;
35                ApplicationArea = All;
36                RunObject = codeunit HelloWorld;
37
38                trigger OnAction()
39                var
40                    TextVar: Text;
41                    IntegerVar: Integer;
42
43                    begin
44                        IntegerVar := 5;
45                        IntegerVar += 12345;
46                        TextVar := 'Hello, Release Note reader!';
47                        Message(TextVar + 'This is the Customer Card');
48                    end;
49                }
50
51            }
52        }
53    }
54
55    var
56        GlobalText: Text;
57        StreamIn: InStream;
58        StreamOut: OutStream;
```

# Agenda

VS Code Extension  
Namespace

ALDoc

Investigate and troubleshoot

AL Language

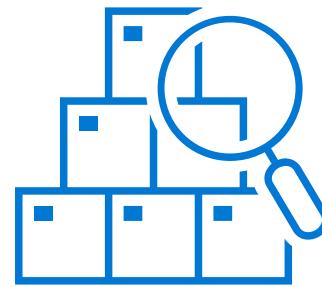
Code authoring

GitHub Copilot Chat

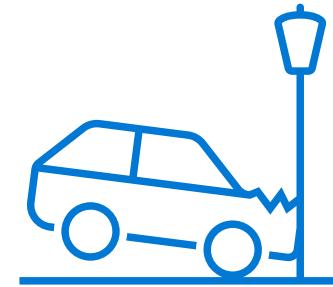
AppSource

# Namespaces – What is it and why does it make sense

A namespace is a way of organizing different names or identifiers that belong to a certain group or context



Grouping objects based of functionality  
to improve discoverability



Reduce/prevent name collisions

# DEMO: Namespaces

# Namespaces - Syntax

- First statement in source file
- Covers all objects in the file
- Namespaces can be nested with a dot
- Namespace names must be CLS compliant
- The same namespace can be used in multiple files
- The same namespace can be used in multiple modules

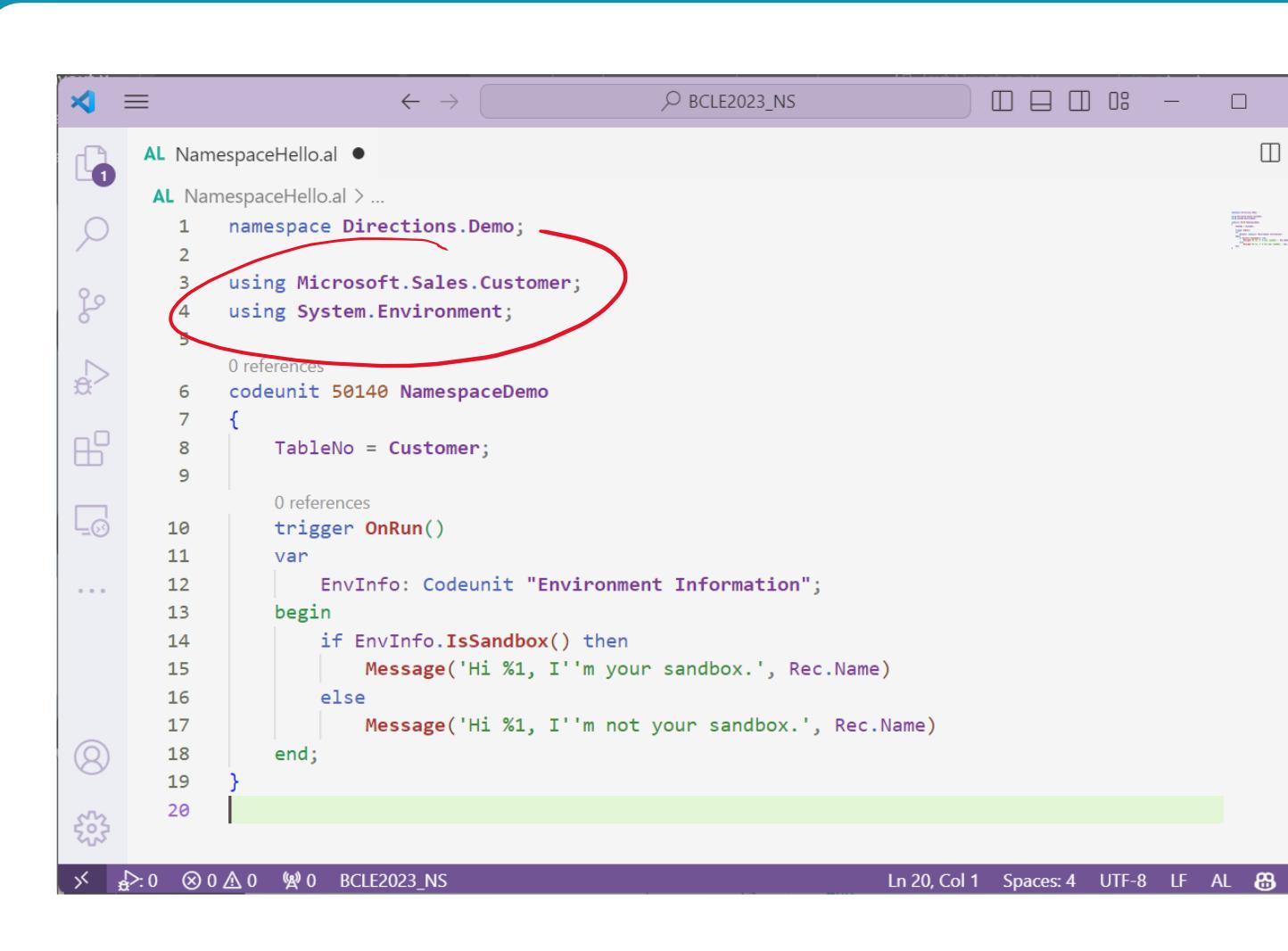
AL EnvironmentInformationImpl.Codeunit.al X

System > Environment Information > src > AL EnvironmentInformationImpl.Codeunit.al > ...

```
1 // -----
2 // Copyright (c) Microsoft Corporation. ALL rights reserved.
3 // Licensed under the MIT License. See License.txt in the project root for license information.
4 // -----
5
6 namespace System.Environment;
7
8 using System;
9 using System.Environment.Configuration;
10
11 codeunit 3702 "Environment Information Impl."
12 {
13     Access = Internal;
```

# Namespaces – Using directive

- Using directives brings all objects from a namespace in scope
- There is no ordering



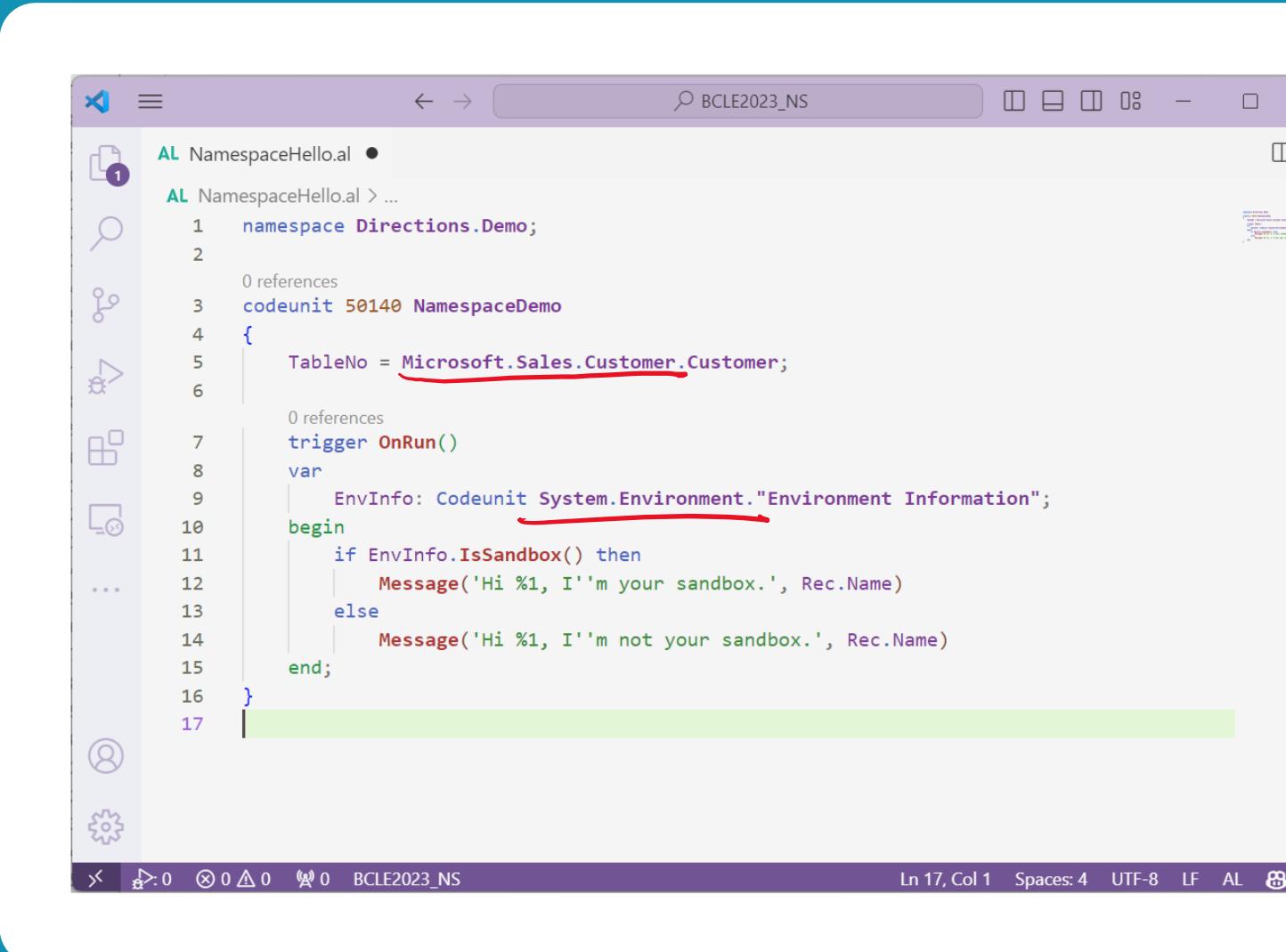
The screenshot shows the Microsoft Dynamics 365 Application Lifecycle (AL) code editor. The current file is `NamespaceHello.al`. The code contains the following:

```
AL NamespaceHello.al
AL NamespaceHello.al > ...
1  namespace Directions.Demo;
2
3  using Microsoft.Sales.Customer;
4  using System.Environment;
5
6  codeunit 50140 NamespaceDemo
7  {
8      TableNo = Customer;
9
10     trigger OnRun()
11     var
12         EnvInfo: Codeunit "Environment Information";
13     begin
14         if EnvInfo.IsSandbox() then
15             Message('Hi %1, I''m your sandbox.', Rec.Name)
16         else
17             Message('Hi %1, I''m not your sandbox.', Rec.Name)
18     end;
19 }
20
```

A red oval highlights the two `using` statements at the top of the codeunit. The status bar at the bottom right indicates the code is 20 lines long, in column 1, with 4 spaces per line, in UTF-8 encoding, and using LF line endings.

# Namespaces – Fully qualified names

- Alternative to using directives
- Specific to a single object
- Removes risk of name clashes



The screenshot shows the Microsoft Dynamics 365 AL code editor interface. The title bar says "BCLE2023\_NS". The left sidebar has icons for file, search, and code navigation. The main area shows a codeunit named "NamespaceHello.al" with the following content:

```
AL NamespaceHello.al
AL NamespaceHello.al > ...
1  namespace Directions.Demo;
2
3  codeunit 50140 NamespaceDemo
4  {
5      TableNo = Microsoft.Sales.Customer.Customer;
6
7      0 references
8      trigger OnRun()
9      var
10         EnvInfo: Codeunit System.Environment."Environment Information";
11         begin
12             if EnvInfo.IsSandbox() then
13                 Message('Hi %1, I''m your sandbox.', Rec.Name)
14             else
15                 Message('Hi %1, I ''m not your sandbox.', Rec.Name)
16         end;
17 }
```

Two specific parts of the code are highlighted with red underlines: "Microsoft.Sales.Customer.Customer" and "System.Environment."Environment Information". The status bar at the bottom shows "Ln 17, Col 1" and other standard editor information.

# Namespaces – 2 rules

#1 Only one object of a kind with the same name in a module

#2 Only one object of a kind with the same name in a namespace

# Namespaces – Affixes

- ⌚ Affxes are still there
- ⌚ We plan to lift some Affix restrictions for new namespaced objects in the future
- ⌚ New namespaced objects can have names without affixes

# Namespace – How to proceed

Do not namespace your own code until all your dependencies are namespaced

Your code will fail to compile if/when dependencies are namespaced

Choose a unique name for your namespaces

E.g. {Company}.{Product}.{Area}.{Feature}

Choose a stable and non-version specific name

Renaming a namespace is a breaking change

Your root should be your own

E.g. do not use System, Microsoft, Utility, Common, etc as root namespaces.

Over time, not switching to namespaces will increase risk of collisions

# Namespace – AL Explorer

The screenshot shows the AL Explorer interface within a Microsoft Dynamics 365 application window titled 'ALProject6'. The interface includes a navigation bar with File, Edit, Selection, View, Go, and other options. Below the bar, there are tabs for 'Explorer' (selected), 'Extension Management.dal', and 'Advanced Settings Impl..dal'. A project dropdown shows 'ALProject6 (Default publisher)'. The main area is titled 'AL Explorer' and contains tabs for 'OBJECTS' (selected), 'EVENTS', 'APIS', and 'EXTENSIBLE ENUMS'. A search bar and group by dropdown are also present. A red circle highlights the 'Namespace' column in the table below. The table lists various objects with their details:

Signature	Type	Object Name ↑	Module	Namespace
OnPostJnlLineOnBeforeGenJnlPostLineRun(var ...)	Report	Adjust Exchange Rates	base Application	MICROSOFT.FINANCE.CURRENCY
OnPostGenJnlLineOnBeforeGenJnlPostLineRun(var ...)	Report	Adjust Exchange Rates	Base Application	Microsoft.Finance.Currency
OnVendorAfterGetRecordOnAfterFindVendLedger...	Report	Adjust Exchange Rates	Base Application	Microsoft.Finance.Currency
OnBeforeGenJnlLineInsert(var NewGenJnlLine: Rec...	Codeunit	Adjust Gen. Journal Balance	Base Application	Microsoft.Finance.GeneralLedger.Journal
OnInsertCorrectionLinesOnBeforeNewGenJnlLineI...	Codeunit	Adjust Gen. Journal Balance	Base Application	Microsoft.Finance.GeneralLedger.Journal
OnRunOnBeforeGenJnlLineFind(var GenJnlLine: Rec...	Codeunit	Adjust Gen. Journal Balance	Base Application	Microsoft.Finance.GeneralLedger.Journal
OnPostMultipleAdjustmentsToItemLedgerOnAfterL...	Codeunit	Adjust Item Inventory	Base Application	Microsoft.Inventory.Item
OnPostMultipleAdjustmentsToItemLedgerOnBeforeL...	Codeunit	Adjust Item Inventory	Base Application	Microsoft.Inventory.Item
OnBeforeOpenGeneralSetupExperience(var PageID: Int...	Codeunit	Advanced Settings	System Application	System.Environment.Configuration
OnBeforeOpenCompanySettings(var PageID: Inte...	Codeunit	Advanced Settings Ext.	Base Application	System.Environment.Configuration
OnTempCustLedgEntryGetRecordOnAfterSetDetail...	Report	Aged Accounts Receivable	Base Application	Microsoft.Sales.Reports
OnLookupBreakdownAccountName(var AllocAcco...	Table	Alloc. Account Distribution	Base Application	Microsoft.Finance.AllocationAccount
OnLookupBreakdownAccountNumber(var AllocAc...	Table	Alloc. Account Distribution	Base Application	Microsoft.Finance.AllocationAccount
OnBeforeValidateCity(var AlternativeAddress: Rec...	Table	Alternative Address	Base Application	Microsoft.HumanResources.Employee
OnBeforeValidatePostCode(var AlternativeAddress...)	Table	Alternative Address	Base Application	Microsoft.HumanResources.Employee
OnLookupTotalingRangeOnElse(var AnalysisLine: ...)	Table	Analysis Line	Base Application	Microsoft.Inventory.Analysis
OnAfterCalcIndirectCost(var ItemStatisticsBuffer: R...	Codeunit	Analysis Report Management	Base Application	Microsoft.Inventory.Analysis
OnAfterCalcItemStatistics(var ItemStatisticsBuf: Re...	Codeunit	Analysis Report Management	Base Application	Microsoft.Inventory.Analysis
OnAfterFilterItemAnalyViewBudgEntry(var ItemSta...	Codeunit	Analysis Report Management	Base Application	Microsoft.Inventory.Analysis
OnAfterFilterItemAnalyViewEntry(var ItemStatistic...	Codeunit	Analysis Report Management	Base Application	Microsoft.Inventory.Analysis

At the bottom, there are buttons for 'Source' and 'Subscribe'. The status bar at the bottom shows the file path 'esbenk/ALDocV2\_ModelAndObsoletionList\*' and the project name 'ALProject6'.

# Agenda

VS Code Extension  
Namespace  
**ALDoc**  
Investigate and troubleshoot  
AL Language  
Code authoring  
GitHub Copilot Chat  
AppSource

# Reference Documentation



Discoverability



Ramping up new  
developers



Help other partners



Anchor point for  
information

# Source as Documentation

```
/// <summary>
/// Triggers the document sharing flow.
/// </summary>
/// <param name="DocumentSharingRec">The record to invoke the share with.</param>
/// <example>
/// The following example shows how to share a document using the Document Sharing codeunit.
/// <code>
/// TempDocumentSharing.Name := 'My Shared Document.pdf';
/// TempDocumentSharing.Extension := '.pdf';
/// TempDocumentSharing.Data := "Document Blob";
/// TempDocumentSharing.Insert();
/// DocumentSharing.Share(TempDocumentSharing);
/// </code>
/// </example>
procedure Share(var DocumentSharingRec: Record "Document Sharing")
begin
    Codeunit.Run(Number: Codeunit::"Document Sharing Impl.", Record: DocumentSharingRec) :Boolean;
end;
```

```
DOCUMENTSHARINGREC. RECORD - DOCUMENT SHARING
begin
    DocumentSharing.Sh
end;
```

Triggers the document sharing flow.

0 references

```
trigger OnRun()
var
    DocumentSharing: Record "Document Sharing"
    DocumentSharingR: Record "Document Sharing"
begin
    DocumentSharing.Share(DocumentSharingRec);
end;
```

Triggers the document sharing flow.

# Source to Documentation

```
/// <summary>
/// Triggers the document sharing flow.
/// </summary>
/// <param name="DocumentSharingRec">The record to invoke the share with.</param>
/// <example>
/// The following example shows how to share a document using the Document Sharing codeunit.
/// <code>
/// TempDocumentSharing.Name := 'My Shared Document.pdf';
/// TempDocumentSharing.Extension := '.pdf';
/// TempDocumentSharing.Data := "Document Blob";
/// TempDocumentSharing.Insert();
/// DocumentSharing.Share(TempDocumentSharing);
/// </code>
/// </example>
procedure Share(var DocumentSharingRec: Record "Document Sharing")
begin
    Codeunit.Run(Number: Codeunit::"Document Sharing Impl.", Record: DocumentSharingRec) : Boolean;
end;
```

## Methods

### Share

Triggers the document sharing flow.

```
procedure Share(var DocumentSharingRec: Record "Document Sharing" temporary)
```

#### Parameters

Name	Description
DocumentSharingRec	Record "Document Sharing" temporary

#### Examples

The following example shows how to share a document using the Document Sharing codeunit.

```
TempDocumentSharing.Name := 'My Shared Document.pdf';
TempDocumentSharing.Extension := '.pdf';
TempDocumentSharing.Data := "Document Blob";
TempDocumentSharing.Insert();
DocumentSharing.Share(TempDocumentSharing);
```

# ALDoc - Two purposes, one tool

## Microsoft

Publish reference documentation for Microsoft 1<sup>st</sup> party apps on [learn.microsoft.com](https://learn.microsoft.com).

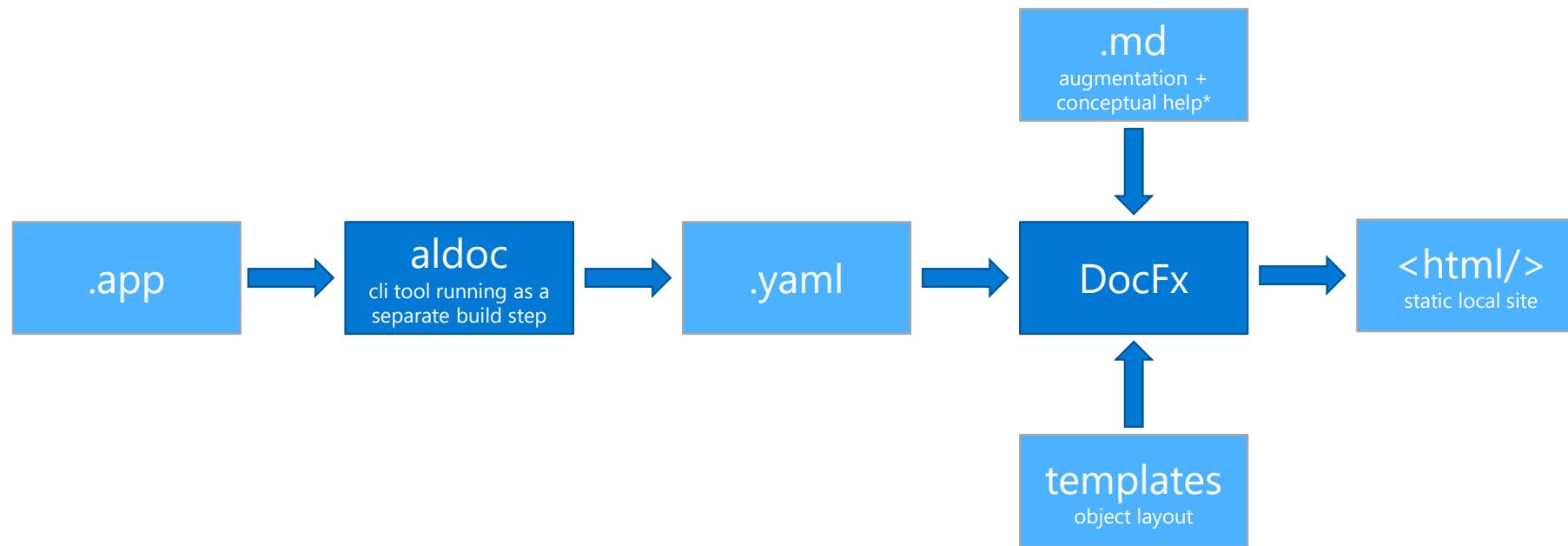
## Partners

Allow partners to document their code base to help their partners (and sometimes internal developers) understand the code.

# ALDoc – transformation flow

1. aldoc init
2. aldoc build

3. docfx build



# DEMO: ALDoc

# ALDoc - Resources

Getting started with ALDoc

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/help/help-aldoc-generate-help>

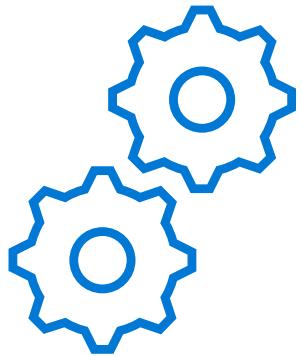
Information on DocFX

<https://dotnet.github.io/docfx/>

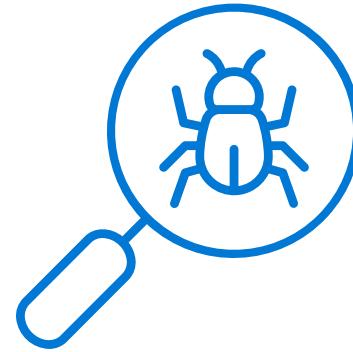
# Agenda

VS Code Extension  
Namespace  
ALDoc  
**Investigate and troubleshoot**  
AL Language  
Code authoring  
GitHub Copilot Chat  
AppSource

# Imagine you want to...



Investigate how a certain  
functionality works



Troubleshoot an issue in a production  
or sandbox environment

# That involves knowledge and manual work...

Open VS Code

Create a workspace or project

Find details on environments

Figure out what the session id is, if wanting to attach to client

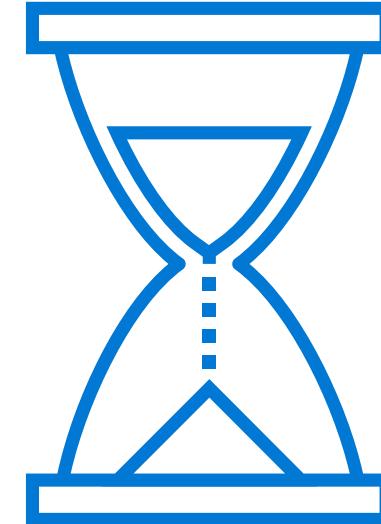
Create one or more launch configurations for the target environments

Determine what apps are involved in the scenario

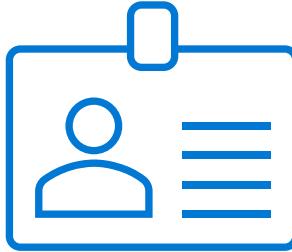
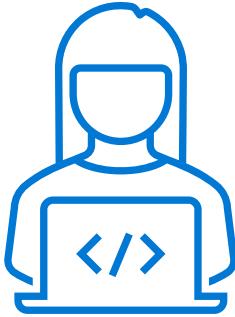
Create app.json with fake dependencies

Find details to authenticate

Download symbols



# Imagine you could automate this!



Increase developer productivity

Empower tech savvy consultants  
to investigate apps and do light  
troubleshooting with limited VS  
Code knowledge

Empower support to  
troubleshoot faster and  
lower time-to-mitigate

**DEMO: Explore and Debug in VS Code from  
web client**

# Overview of Explore and Attach options

The screenshot shows the Microsoft Dynamics 365 Business Central interface. On the left, the 'Vendor Card' for '10000 · Fabrikam, Inc.' is displayed. The 'General' section shows details like No. (10000), Balance (2,071.13), Name (Fabrikam, Inc.), and Blocked status. Below it, the 'Address & Contact' section shows Address, Phone No. (4255550101), Mobile P..., and Email (krystal.york@co). A circular 'Vendor Picture' placeholder is shown. To the right, a modal window titled 'Page Inspection' provides information about the page. It includes tabs for 'Page', 'Table', and 'Table Fields'. Under 'Table Fields', there is a list of fields: 'Phone No. (9, Text[30])' with value '4255550101' (Base Application), and 'Mobile Phone No. (508, Text[50])' with value '(Blank)' (Base Application). Two buttons are highlighted with pink boxes: 'Explore page in Visual Studio Code' (under Page) and 'Explore field in VS Code' (under Table Fields).

Extensions: Those listed in Page Inspection Extensions tab

Source focus: Specific table field in table

Extensions: Base and System

Source focus: Nothing

Extensions: Those listed in Page Inspection Extensions tab

Source focus: Page

## Help & Support

### Find it

Learn more about your current task or page ([Vendor Card](#))

Find answers in the [Help](#)

Read the [blog](#)

See which capabilities are [coming soon](#)

### Do more with your trial

[Request partner contact](#) to start a subscription

Explore pre-built solutions from our partners that [extend](#) the capabilities of Business Central

Get started with [Business Central consulting services](#) and find partners with expertise in your industry or region.

The demonstration and setup data in CRONUS evaluation companies and My Company, including the predefined configuration packages, are not designed to satisfy local requirements in any country or region.

### Give feedback

Do you have a great idea that you would like to see in Business Central?

Register your idea and get others in the community to vote for it at [aka.ms/BusinessCentralideas](#)

### Troubleshooting

Version: US Business Central 23.0 (Platform 23.0.12468.0 + Application 23.0.12034.12474)

[View the last known error](#)

[Inspect pages and data](#)

[Analyze performance](#)

[Enable additional logging](#)

[Attach debugger to this session](#)

# Permissions

None for Attach in Help and Support

"Troubleshooting tools" for Page Inspector options

Permission Set ↑	Name	Type ↑	Extension Name
→ <a href="#">TROUBLESHOOT TOOLS</a>	⋮ Troubleshoot Tools	System	System Application

# Overview of Visual Studio Code flow

## Open Visual Studio Code

If already open, user asked whether to create new project or reuse existing  
If AL VSIX not installed, it will suggest it and stop

## Generate empty project

app.json with dependencies to extensions  
Snapshot and debug launch configurations matching environment and session  
Asks to login  
Download symbols

Asks whether initiate snapshot debug (and profiling), regular debug or nothing

# “Best practices”

Be patient, downloading symbols can take time

You still need to have access to extension source to view

Delete generated projects occasionally

Reuse project and optionally symbols for longer running flows

Consider using a shared symbols path setting

Both production and sandboxes supported (only debug in latter)

Use for easy AL profiling with snapshots and instrumentation

# Future ideas, no commits

Open in Codespaces

Open with code from GitHub

Specifically, from source and build metadata in extension

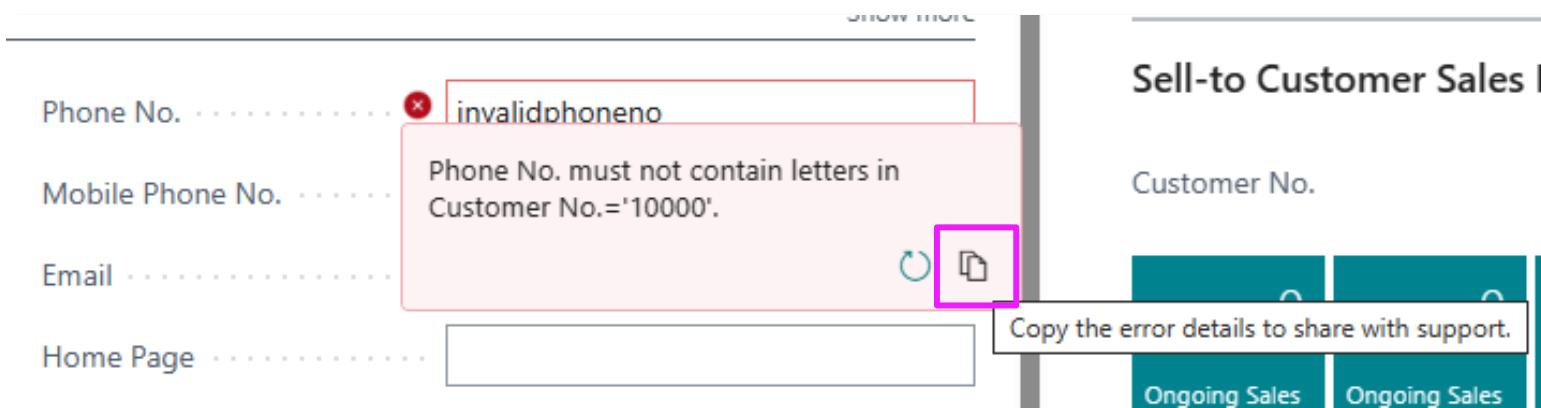
Pull latest for extensions that supports it

Get symbols for all extensions

Open for specific extension (and dependencies)

**Give us feedback on Ideas and Viva (Yammer)!**

# New: Error details including call stack



If requesting support, please provide the following details to help troubleshooting:

Phone No. must not contain letters in Customer No.=''10000''.

Internal session ID:  
f4c3969c-5e8b-4114-80d4-cb5081a5494e

Application Insights session ID:  
041b521f-9027-42f3-a494-466b6af9f065

Client activity id:  
f46c3e4b-4cf6-d6d6-74f8-ed61150ac3dc

Time stamp on error:  
2023-09-19T19:41:58.7499959Z

User telemetry id:  
22e667cd-6607-4fc2-aa39-ca4ba6db4e91

AL call stack:  
Customer(Table 18)."Phone No. -  
OnValidate"(Trigger) line 7 - Base Application by  
Microsoft

# IntelliSense in debugger console

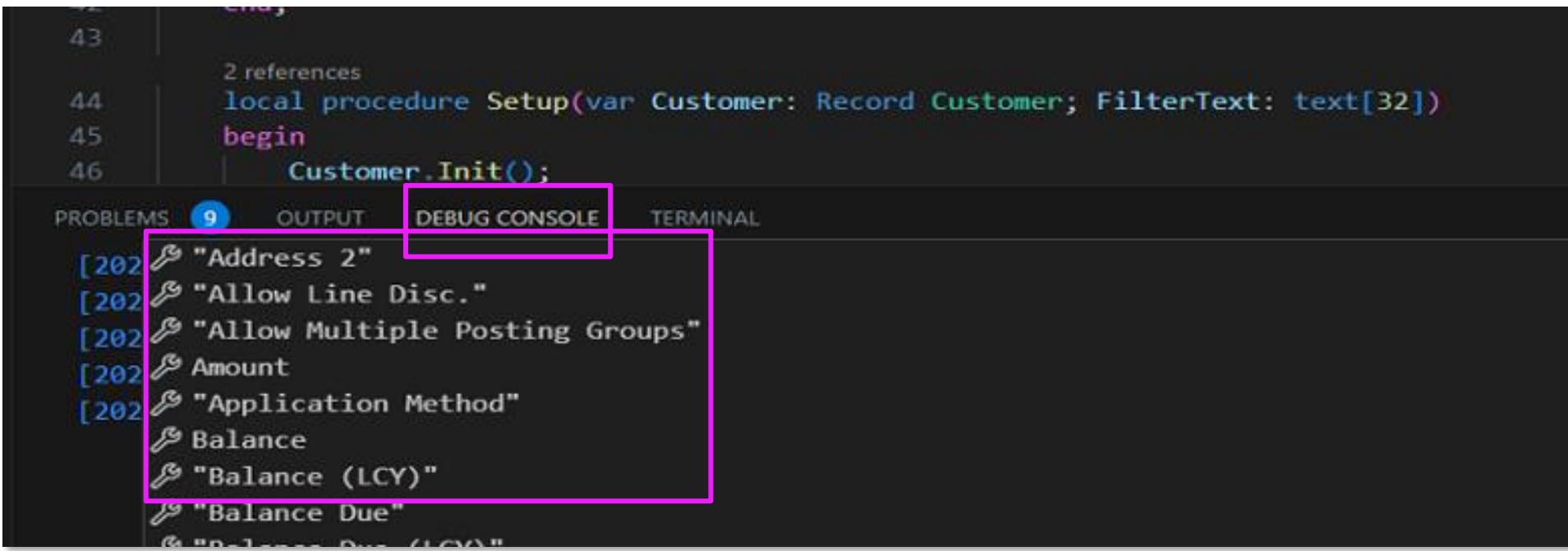
On object types and members that can be evaluated in debugger

Fields

Locals

Globals

Member symbols that are not methods on a data type (evaluation of methods not supported)



```
42
43      2 references
44      local procedure Setup(var Customer: Record Customer; FilterText: text[32])
45      begin
46          Customer.Init();
```

PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL

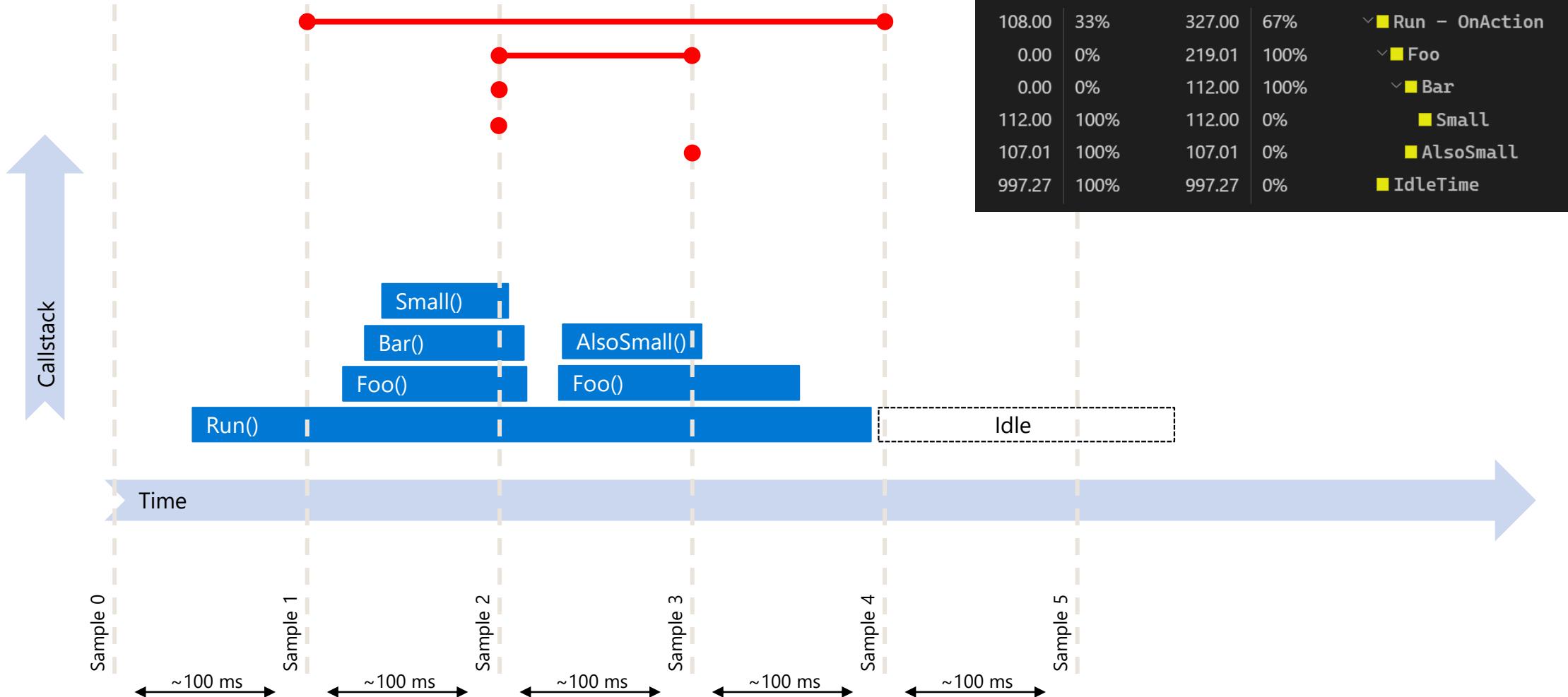
```
[202] ⚒ "Address 2"
[202] ⚒ "Allow Line Disc."
[202] ⚒ "Allow Multiple Posting Groups"
[202] ⚒ Amount
[202] ⚒ "Application Method"
[202] ⚒ Balance
[202] ⚒ "Balance (LCY)"
[202] ⚒ "Balance Due"
[202] ⚒ "Balance Due (LCY)"
```

# Recap: In-client profiler and AL performance profiler

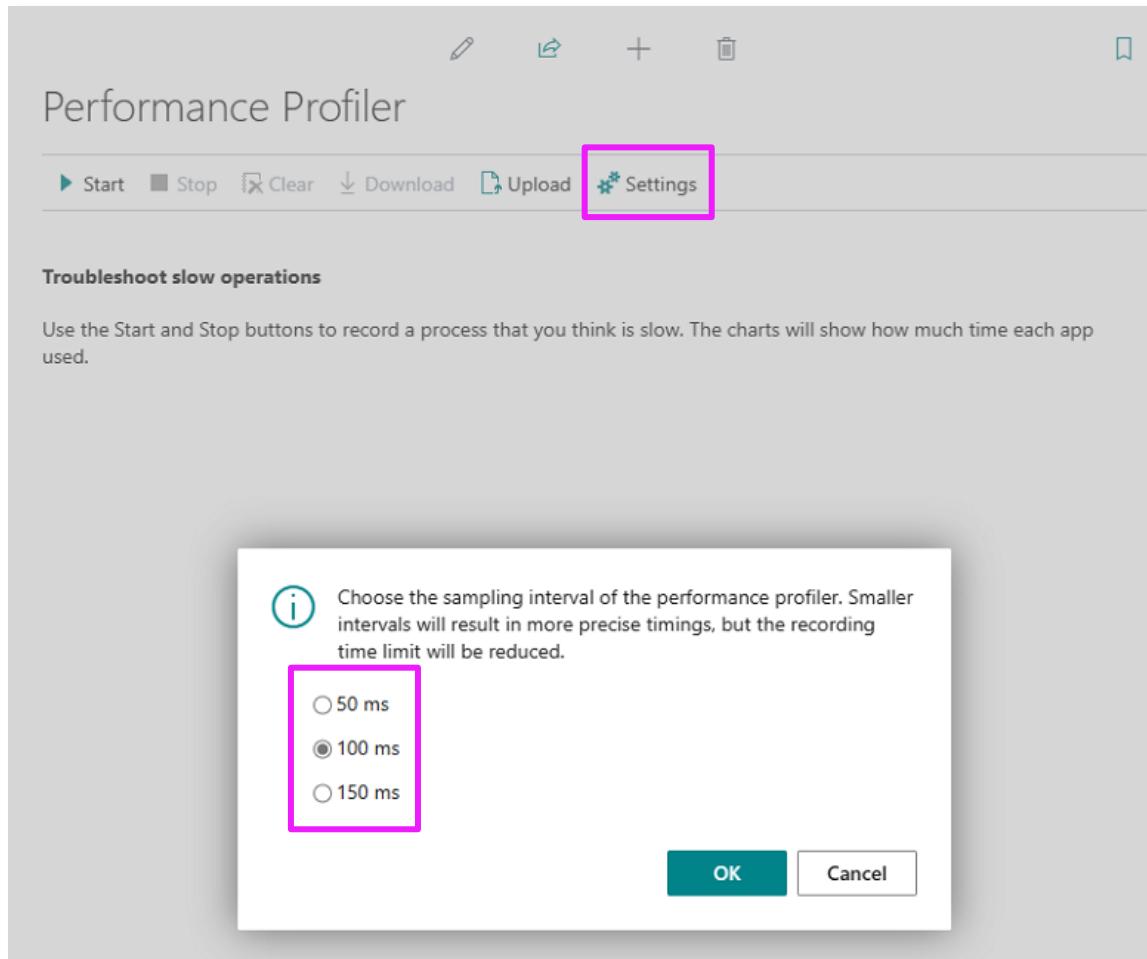
The screenshot shows the Dynamics 365 Performance Profiler interface. The main area displays a 'Call Tree' with various application components and their performance metrics. A detailed timeline overlay is shown for a specific section of the code, highlighting execution times for different functions.

Method Name	Object Type	Object Name	Self Time	Total Time	App Name
OnOpenPage	Page	Sales Invoice		402 milliseconds	Base Application
ActivateFields	Page	Sales Invoice			Base Application
SetDocNoVisible	Page	Sales Invoice	182 milliseconds	182 milliseconds	Base Application
IsPaymentServiceVisible	Table	Payment Service Setup		109 milliseconds	Base Application
OnRegisterPaymentServiceP...	Table	Payment Service Setup	109 milliseconds	109 milliseconds	Base Application
CheckShowBackgrValidationN...	Page	Sales Invoice	111 milliseconds	111 milliseconds	Base Application
OnOpenSalesInvoice	Codeunit	LP Subscribers		109 milliseconds	Late Payment Prediction
ShowL	Self Time in ms	Total Time in ms	Function Name		
Show	108.00	33%	Run – OnAction		
Show	0.00	0%	Foo	109 milliseconds	109 milliseconds
SalesInv	0.00	0%	Bar		4 seconds 987 ...
→ Proces	112.00	100%	Small	4 seconds 987 ...	4 seconds 987 ...
OnInit	112.00	100%	AlsoSmall	111 milliseconds	221 milliseconds
FillLo	107.01	100%	IdleTime	110 milliseconds	Base Application
FillBu	997.27	100%		110 milliseconds	Base Application
OnOpenPage				110 milliseconds	Base Application

# Recap: Sampling during profiling



# New: Additional sampling intervals in profilers



The screenshot shows a portion of a JSON configuration file. The relevant code is:

```
13     "name": "AL: Generated Snapshot request",
14     "request": "snapshotInitialize",
15     "type": "al",
16     "environmentType": "Sandbox",
17     "environmentName": "bcle",
18     "sessionId": 165998,
19     "breakOnNext": "WebClient",
20     "executionContext": "DebugAndProfile",
21     "profilingType": "Sampling",
22     "profileSamplingInterval": 100
23 }
```

A pink box highlights the 'profileSamplingInterval' field and its value '100'. A tooltip for this field provides the following information: 'Specifies the sampling interval in milliseconds when "Sampling" profiling type is specified. Default is 100ms.' The tooltip also lists other values: '150' and '50'. To the right of the tooltip, the text 'Default value' is visible.

# Agenda

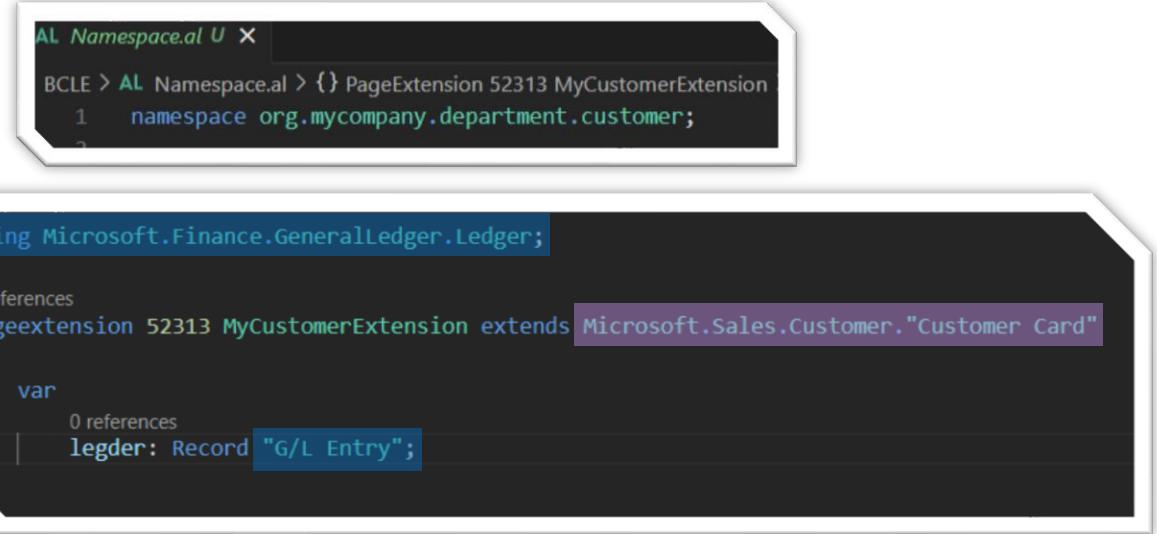
VS Code Extension  
Namespace  
ALDoc  
Investigate and troubleshoot  
**AL Language**  
Code authoring  
GitHub Copilot Chat  
AppSource

# AL Language

## Namespace

Add using statements  
Fully qualified name

```
procedure GetFromMapping(DictKey: Variant; Mapping: Variant; &  
var  
    ByName: Dictionary of [Text, Text];  
    ByIntegerId: Dictionary of [Integer, Text];  
    ByCode: Dictionary of [Code[100], Text];  
begin  
    case OperationType of  
        OperationType::ByNameMapping:  
            begin  
                ByName := Mapping;  
                exit(ByName.Get(DictKey));  
            end;  
        OperationType::CodeMapping:  
            begin  
                ByCode := Mapping;  
                exit(ByCode.Get(DictKey));  
            end;  
        OperationType::IntegerMapping:  
            begin  
                ByIntegerId := Mapping;  
                exit(ByIntegerId.Get(DictKey));  
            end;  
end;
```



## Variant

Variant to Dictionary/List  
New variant type methods  
IsDictionary  
IsList

# AL Language

## SecureText

New datatype for variables and parameters

Protects data from being inspected in the AL Debugger

The screenshot shows the AL Debugger interface. On the left, the Locals window displays local variables: secretProvider (Codeunit), headerValue (<HiddenValue>), and a highlighted Message: GET object. Inside the Message object, Method is set to GET, RequestUri is a <HiddenValue>, and privateValue is also a <HiddenValue>. In the code editor on the right, line 40 shows the declaration of variable Ok: Boolean; and the beginning of a begin block. Line 41 shows the call to secretProvider.Setup. Line 42 contains the assignment of headerValue := secretProvider.GetAPIKey() : SecretText; and the call to GetAPIKeyFromBing. Line 43 shows the exit of the block with privateValue. Line 44 shows the end of the block.

```
37     Ok: Boolean;
38     begin
39         secretProvider.Setup(allowSet: false, allowAPIKey: true, allowQueryKeyValue: true);
40         headerValue := secretProvider.GetAPIKey() : SecretText;
41         Message := CreateHttpMessageForBing(apiKeyValue: headerValue) : HttpRequestMessage;
42         Ok := GetAPIKeyFromBing(message: Message, Result: privateValue) : Boolean;
43         exit(privateValue);
44     end;
```

Set SecureText in methods attributed with [NonDebuggable]  
Pass on SecureText internally in your extension

# Fieldgroup

Add fields to fieldgroup - even if it does not exist

The image shows a dual-pane development environment and a Dynamics 365 application interface.

**Code Editor (Left):**

```
AL FieldGroupExt.al M X
BCLE > AL FieldGroupExt.al > TableExtension 52310 "G/L Entry Extension"
4
0 references
5 tableextension 52310 "G/L Entry Extension" extends "G/L Entry"
6 {
7     fieldgroups
8     {
9         addlast(Brick; Description, Amount, "Debit Amount") { }
10    }
11 }
```

**Code Editor (Right):**

```
AL GL Entry.dal X
518
519
520 fieldgroups
521 {
522     fieldgroup(DropDown; "Entry No.", Description, "G/L Account No.")
523     {
524     }
525 }
526
```

**Dynamics 365 Application (Bottom):**

CRONUS USA, Inc. | Finance ▾ Cash Management ▾ Sales ▾ Purchasing ▾ Shopify ▾ All Reports ▾

GLEntryBrick: All ▾ | Search Analyze + New Delete Edit List

Category	Description	Amount	Category	Description	Amount	Category	Description	Amount
Direct Cost	20000 on 01/18/22	-128.40	Invoice	S-INV102001	128.40	Invoice	S-INV102001	-164.70
Invoice	S-INV102001	-4.94	Invoice	S-INV102001	-1.65	Invoice	S-INV102001	-1.65
Invoice	S-INV102001	172.94	Invoice	S-INV102001	172.94	Invoice	S-INV102001	-172.94

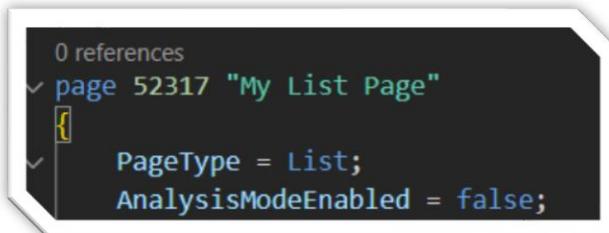
Appends fields, excluding duplicates

# AL Language

## AllowInCustomizations

Always allow (default)

Never allow



A screenshot of the Dynamics 365 customization interface. On the left, there is a code editor window showing AL code for a table extension:

```
tableextension 52315 MyCustomerExtension extends Customer
{
    fields
    [
        0 references field(52315; "Field1"; Integer)
        {
            DataClassification = CustomerContent;
            AllowInCustomizations = Always;
        }
        0 references field(52316; "Field2"; Text[20])
        {
            DataClassification = SystemMetadata;
            AllowInCustomizations = Never;
        }
    ]
}
```

To the right, a modal dialog titled "Add Field to Page" is open, instructing users to "Place fields by dragging from the list to a position on the page". It lists several fields with their types and "Allow In Customizations" settings:

Type	Field Name	Allow In Customizations
Text	Fax No.	Always
Integer	Field1	Never
Decimal	Fin. Charge Memo Amounts ...	Never

## AnalysisModeEnabled

On by default

Disable analysis mode for specific list page

# AL Language

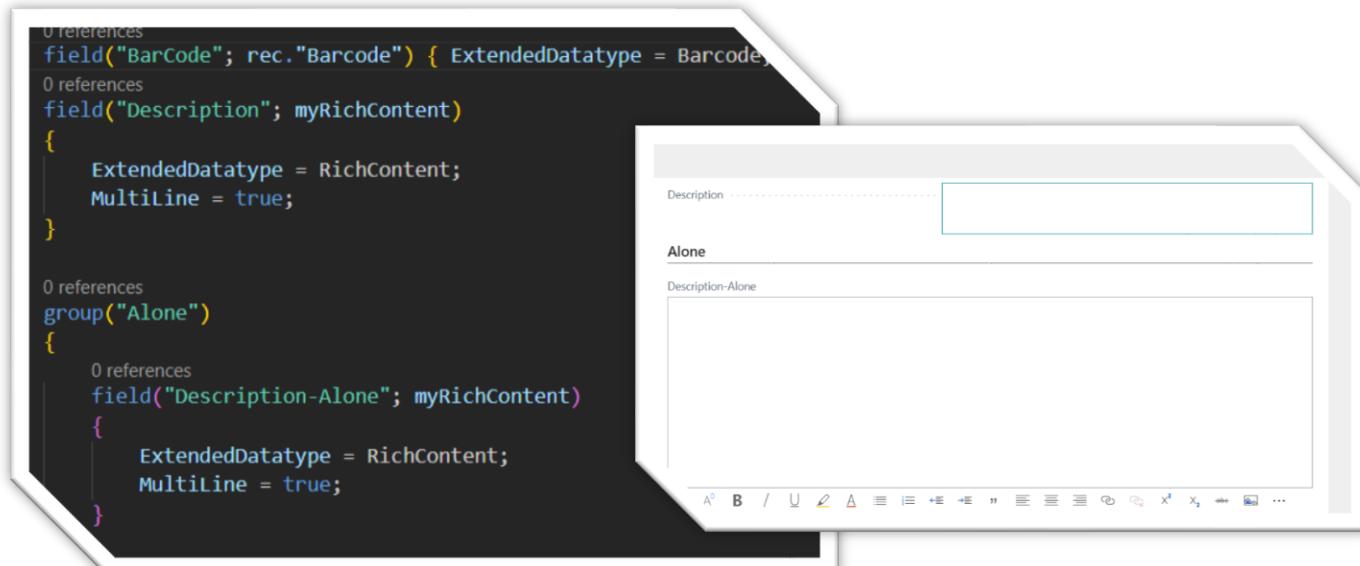
## ExtendedDataType **Barcode**

Set on table or page field to hint value is a barcode

Open device camera and scan

## ExtendedDataType **RichContent**

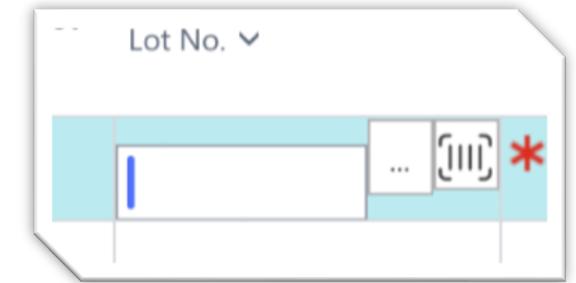
Add to page field for rich content controls.



The screenshot shows a portion of AL Language code in a code editor. The code defines a RichContent control for a 'Description' field within a 'Description-Alone' group. The 'MultiLine' property is set to true, indicating it's a rich text area.

```
0 references
field("BarCode"; rec."Barcode") { ExtendedDatatype = Barcode;
0 references
field("Description"; myRichContent)
{
    ExtendedDatatype = RichContent;
    MultiLine = true;
}

0 references
group("Alone")
{
    0 references
    field("Description-Alone"; myRichContent)
    {
        ExtendedDatatype = RichContent;
        MultiLine = true;
    }
}
```



Requires Multiline = true  
Must be the only control  
in a group

# Rich Content fields

A native part of the platform, including

- Toolbar with keyboard shortcuts
- Support for formatting, images, links, tables
- Copy & Paste
- Multi-line, moderate sizing
- HTML output

Ideal for authoring multi-media content, such as an email body, social media post or rich notes.

The screenshot shows the Dynamics 365 Business Central interface. On the left, there's a list view of items for 'CRONUS USA, Inc.' under the 'Inventory' module. On the right, a modal window titled 'Social Media post' is open. It contains a rich text editor toolbar at the top with icons for bold, italic, underline, etc. Below the toolbar, there are buttons for 'Send', 'Discard draft', and 'Use Word template'. A section labeled 'Target platforms' has a dropdown menu. The main content area displays a promotional message: 'Introducing the ANTWERP conference table'. It mentions the launch of a new product, the ANTWERP conference table, after years of research and development. It includes links to '2023 Brochure.pdf' and 'Technical details.docx'. A green call-to-action button says 'Join us at the Green Room live webinar as we show you around - sign up is now open.' Below the message is a photograph of a modern conference table surrounded by chairs. At the bottom of the modal, there are more toolbar icons and some status information: 'MUNICH Swivel Chair, yellow', 'Inventory', '0 No', 'No PCS', and '148.10'.

# AL Language

## Query from RunObject

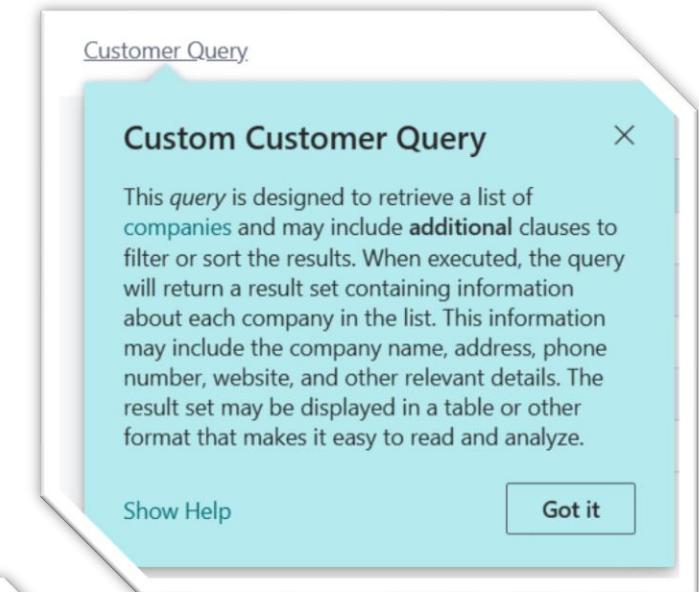
```
action("Run my Query")
{
    Caption = 'My Action';
    RunObject = query MyQuery;
}
```

## Additional Query metadata

UsageCategory to find the Query in Tell Me

AboutTitle/AboutText (and ML) for  
Teaching Tips - including markup

*Italic* by either \_Italic\_ or \*Italic\*, **Bold** by either \_Bold\_ or \*\*Bold\*\*  
[Internal links](#) by [Internal links](?page=<pageId>)



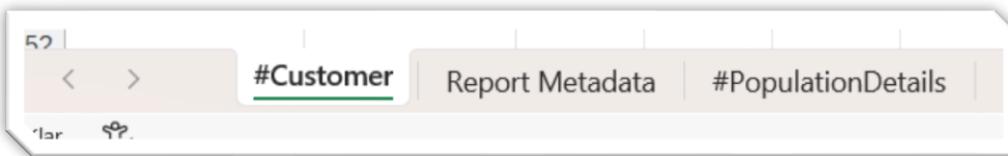
```
query 52318 MyQuery
{
    AboutTitle = 'Custom Customer Query';
    AboutText = 'This *query* is designed to retrieve a list of [companies](?page=22) and may include **additional** clauses to filter or sort the results. When executed, the query will return a result set containing information about each company in the list. This information may include the company name, address, phone number, website, and other relevant details. The result set may be displayed in a table or other format that makes it easy to read and analyze.';
    UsageCategory = ReportsAndAnalysis;
```

# AL Language

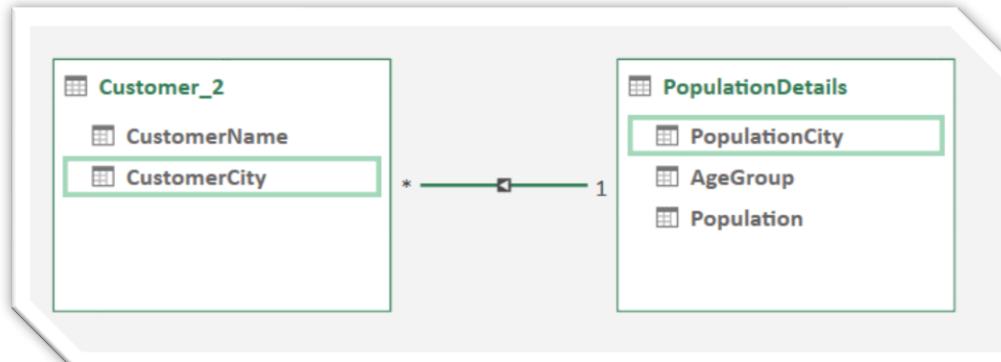
## ExcelLayoutMultipleDataSheets

Multiple Data Sheets in Excel

Simplifies complex reporting with multiple datasets in a single Excel report



Power Pivot/Query in Excel to model



```
report 52310 MyReport
{
    Caption = 'Customer potential impact report';
    ApplicationArea = All;
    UsageCategory = ReportsAndAnalysis;
    ExcelLayoutMultipleDataSheets = true;

    dataset
    {
        2 references
        dataitem("Customer"; Customer)
        {
            0 references
            column("CustomerName"; Customer.Name) { }
            0 references
            column("CustomerCity"; Customer."city") { }
        }

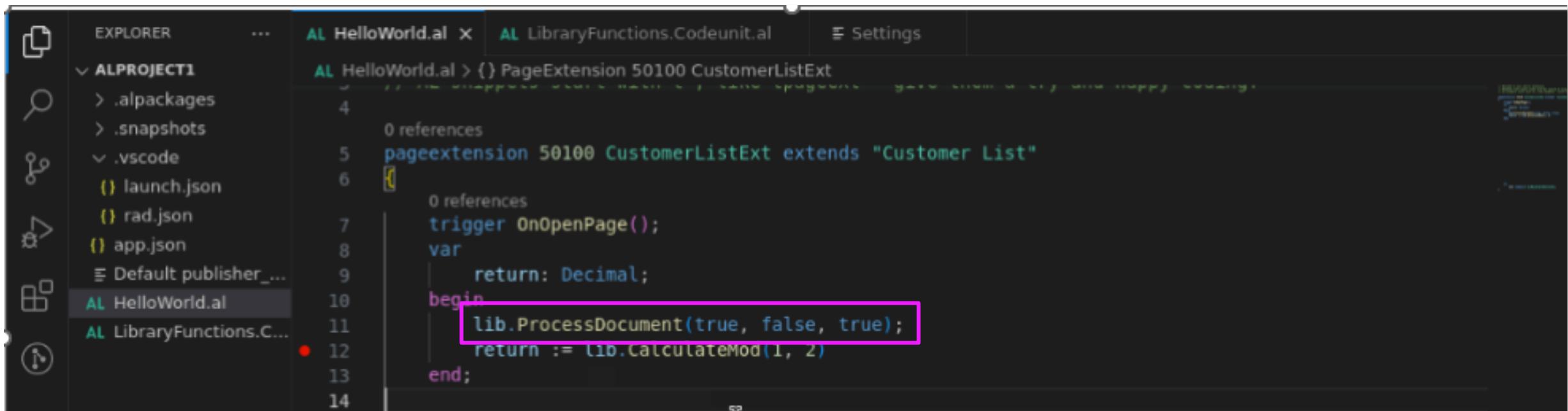
        3 references
        dataitem("PopulationDetails"; PopulationDetail)
        {
            0 references
            column("PopulationCity"; PopulationDetails."City") { }
            0 references
            column("AgeGroup"; PopulationDetails."Age Group") { }
            0 references
            column("Population"; PopulationDetails.Population) { }
        }
    }
}
```

# Agenda

VS Code Extension  
Namespace  
ALDoc  
Investigate and troubleshoot  
AL Language  
**Code authoring**  
GitHub Copilot Chat  
AppSource

# IntelliSense: Inlay hints on parameters and return types

Add **inline information to source code with Ctrl+Alt** to help you understand code better



The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure with files like `.alpackages`, `.snapshots`, `.vscode`, `launch.json`, `rad.json`, `app.json`, and `Default publisher...`. The file `AL HelloWorld.al` is currently selected.
- Editor View:** Displays the AL (Action Language) code for a page extension. The code includes:

```
AL HelloWorld.al > {} PageExtension 50100 CustomerListExt
4
5    0 references
6    pageextension 50100 CustomerListExt extends "Customer List"
7    {
8        0 references
9        trigger OnOpenPage();
10       var
11           return: Decimal;
12           begin
13               lib.ProcessDocument(true, false, true);
14               return := lib.CalculateMod(1, 2)
15           end;
```
- IntelliSense:** An inlay hint is displayed over the call to `lib.ProcessDocument` at line 13, highlighting the parameters `true`, `false`, and `true`.

# IntelliSense: Inlay hints on parameters and return types

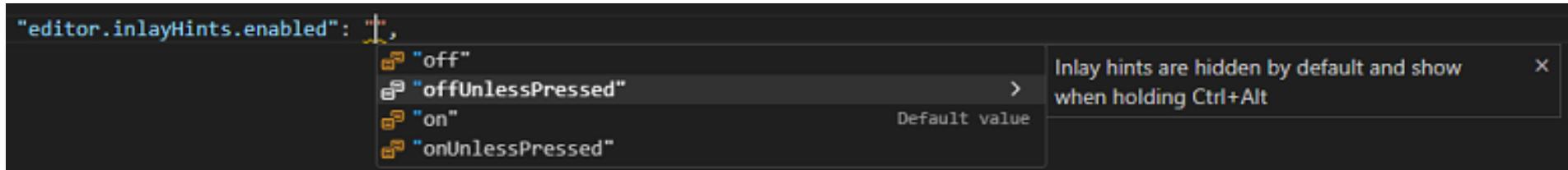
Add **inline information to source code with Ctrl+Alt** to help you understand code better

```
AL HelloWorld.al > () PageExtension 50100 CustomerListExt
4
5    0 references
6    pageextension 50100 CustomerListExt extends "Customer List"
7
8        0 references
9        trigger OnOpenPage();
10       var
11           return: Decimal;
12           begin
13               lib.ProcessDocument(includeStartDate: true, includeEndDate: false, deleteWhenDone: true);
14               return := lib.CalculateMod(number: 1, dividedBy: 2) :Integer
15           end;
16
17
18
19
20
21
```

# IntelliSense: Inlay hints on parameters and return types

Enable in Visual Studio Code settings

Decide on standard behavior



And look

```
"editor.inlayHints.fontFamily": "",  
"editor.inlayHints.fontSize": 8,  
"editor.inlayHints.padding": false,
```

Enable for AL

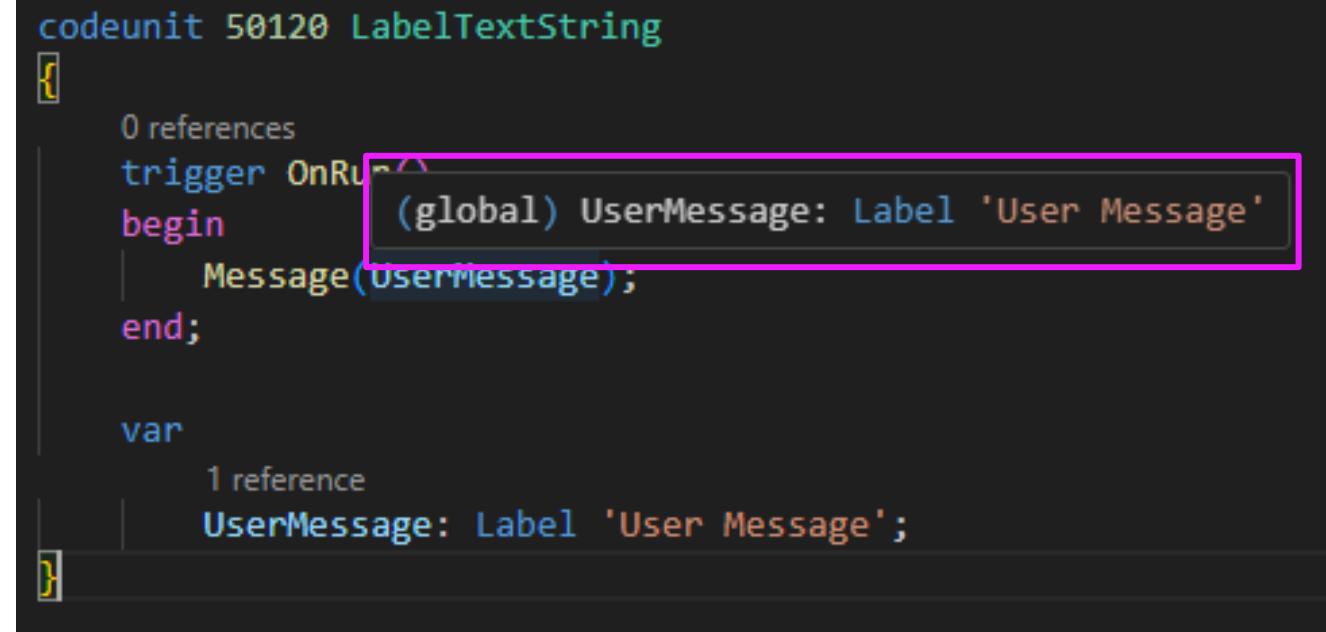
```
"al.inlayhints.functionReturnTypes.enabled": true,  
"al.inlayhints.parameterNames.enabled": true
```

# IntelliSense: Hover on label variable for string value

Want to see label string value?

Before: Navigate to definition or use Peek Definition (Alt+F12)

Now: Just hover over variable to see content



A screenshot of a code editor showing a tooltip for a variable. The code is:

```
codeunit 50120 LabelTextString
{
    0 references
    trigger OnRun()
    begin
        Message(UserMessage);
    end;

    var
        1 reference
        UserMessage: Label 'User Message';
}
```

The variable `UserMessage` is highlighted with a pink rectangle. A tooltip is displayed above it, showing the value `(global) UserMessage: Label 'User Message'`.

# IntelliSense: Smarter signature recommendations

Before: Developer had to select overload manually

Now: First valid overload recommended, active parameter highlighted.

Remaining overloads sorted in order of relevance based on provided parameters.

The screenshot shows a code editor window for a PowerApp named 'SupermarketPayments'. The code is as follows:

```
procedure Process(PaymentId: Guid; PaymentAmount: Decimal; TaxKind: Enum "Tax Calculation Type");
}

0 references
codeunit 50888 SupermarketPayments
{
    0 references
    procedure IncomingPayment(PaymentId: Code[20]; Amount: Decimal; TaxKind: Enum "Tax Calculation Type");
        begin
            Processor.Process(PaymentId, Amount, TaxKind);
        end;

    var
        1 reference
        Processor: Interface PaymentProcessor;
}
```

An IntelliSense dropdown is open at the cursor position, showing the following suggestion:

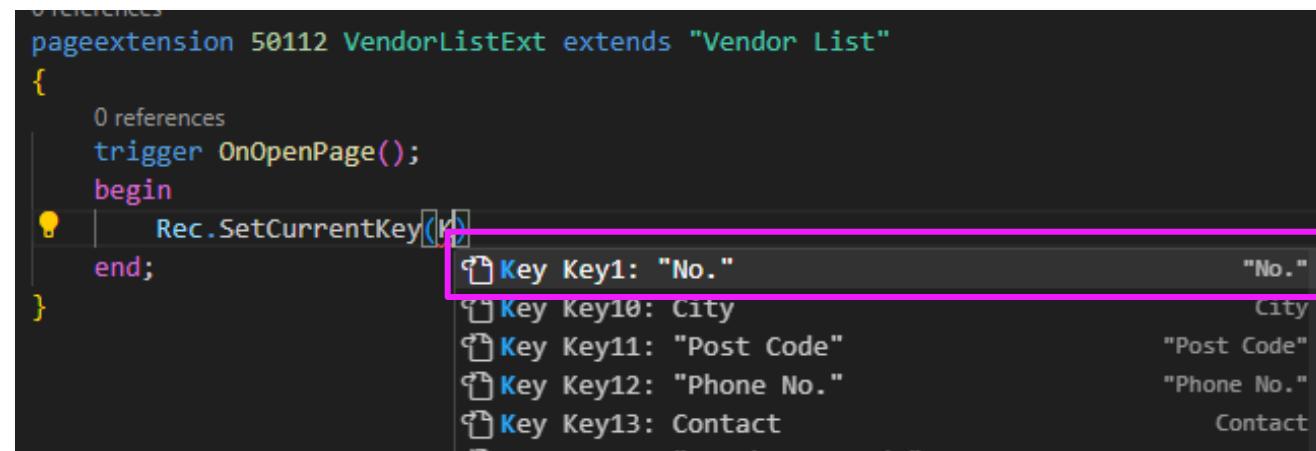
```
procedure Process(PaymentId: Guid, PaymentAmount: Decimal, TaxKind: Enum "Tax Calculation Type")
^
1/6 No documentation is available.
```

The 'TaxKind' parameter in the suggested signature is highlighted in blue, indicating it is the active parameter. A pink rectangle highlights the entire dropdown menu.

# IntelliSense: Available keys

Methods accepting keys, e.g., SetCurrentKey(), now lists available keys in IntelliSense.

Fields still suggested, e.g., for sorting on specific fields



```
0 references
pageextension 50112 VendorListExt extends "Vendor List"
{
    0 references
    trigger OnOpenPage();
    begin
        Rec.setCurrentKey(K)  
| Key Key1: "No." "No."
        | Key Key10: City City
        | Key Key11: "Post Code" "Post Code"
        | Key Key12: "Phone No." "Phone No."
        | Key Key13: Contact Contact
    end;
}
```

# Navigate: “Find all references” improvements

Now works on triggers, system methods, and trigger events

On page and table fields, page actions, and table methods/triggers/events.

Finds only references which are relevant

```
1777 17 references
trigger OnInsert()

MiniCustomerTemplate.Table.al C:\git\Platform-Core\Apps\BaseApp - References (18)
421     DimensionsTemplate: Record "Dimensions Template";
422     ConfigTemplateMgt: Codeunit "Config. Template Management";
423     RecRef: RecordRef;
424     begin
425         OnBeforeInsertCustomerFromTemplate(MiniCustomerTemplate: Rec, ConfigTemplateHeader: ConfigTemplateHeader);
426         Customer.SetInsertFromTemplate(FromTemplate: true);
427         InitCustomerNo(Customer: Customer, ConfigTemplateHeader: ConfigTemplateHeader);
428         Customer.Insert(RunTrigger: true) : Boolean;
        RecRef.GetTable(Rec: Customer);
429         ConfigTemplateMgt.UpdateRecord(ConfigTemplateHeader: ConfigTemplateHeader,
430         RecRef.SetTable(Rec: Customer));
431
432         DimensionsTemplate.InsertDimensionsFromTemplates(ConfigTemplateHeader: ConfigTemplateHeader);
433         Customer.Find : Boolean;
434         OnAfterInsertCustomerFromTemplate(MiniCustomerTemplate: Rec, ConfigTemplateHeader: ConfigTemplateHeader);
435     end;

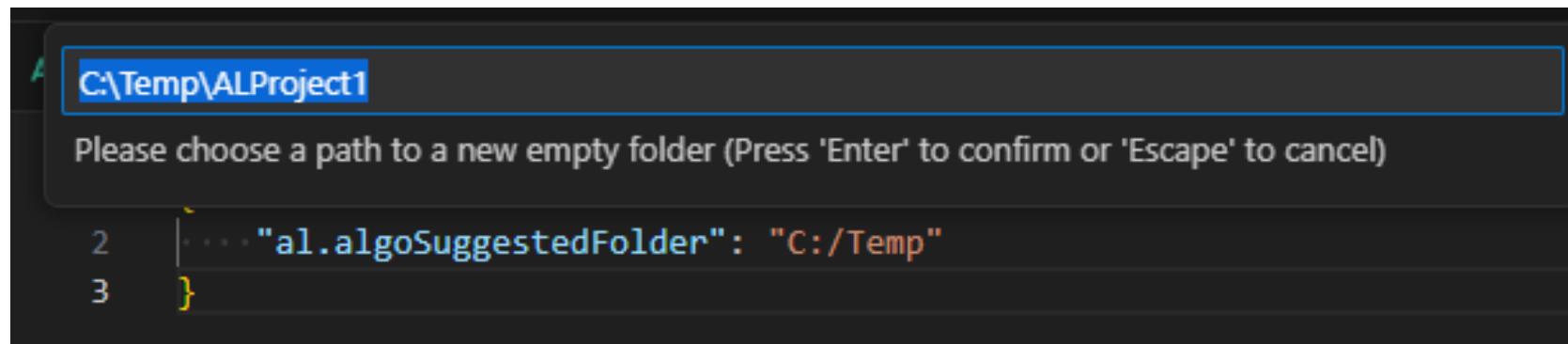
var
```

- > Contact.Table.al 2
- ✓ Customer.Table.al 2
  - trigger OnInsert()
  - Customer.Insert(true)
- > CustomerDataMigrationFacade.Codeunit.al 1
- > CustomerEntity.Page.al 2
- > CustomerPicture.Page.al 1
- > CustomerTempMgt.Codeunit.al 2
- ✓ MiniCustomerTemplate.Table.al 1
  - Customer.Insert(true);
- > NativeCustomerEntity.Page.al 2
- > O365ExcelImportManagement.Codeunit.al 1

# Project: Default folder location for new projects.

Before: AL:Go! for project creation used fixed folder suggestion

Now: Custom root folder location can be defined in settings



# Project: Output folder when building apps

Before: Built app always created in project folder

Now: Support for defining output folder in settings

```
"al.compilationOptions": {  
    "outFolder": "C:/myappfolder"  
},
```

Useful, e.g., for sharing apps across projects, even across workspaces

Project will still contain node for app file, but physical location outside of project

# Agenda

VS Code Extension  
Namespace  
ALDoc  
Investigate and troubleshoot  
AL Language  
Code authoring  
**GitHub Copilot Chat**  
AppSource

# What is GitHub Copilot Chat?

The screenshot shows the GitHub Copilot Chat interface integrated into the Visual Studio Code (VS Code) environment. On the left, the Chat sidebar displays a conversation between GitHub Copilot and the user @pboring\_microsoft. The user asks for an explanation of a provided code snippet, which Copilot provides. A pink box highlights the explanatory text. On the right, the main editor window shows a code file named `ConvertTemperature.CodeUnit.al`. A pink box highlights the code itself. A context menu is open over the code, with another pink box highlighting the "Explain This" option under the "Copilot" section.

CHAT: GITHUB COPILOT

GitHub Copilot  
Hi @pboring\_microsoft, how can I help you?

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and [share feedback](#) so that we can learn and improve.

pboring\_microsoft  
`/explain`

GitHub Copilot  
This code defines a procedure called `ConvertCelsiusToFahrenheit` that takes a single input parameter, `Celsius`, which is of type `Decimal`. The procedure converts the temperature in Celsius to Fahrenheit and returns the result as a `Decimal` value.

The procedure first declares a local variable called `Fahrenheit` of type `Decimal`. It then calculates the Fahrenheit temperature by multiplying the Celsius temperature by 9/5 and adding 32. The result is stored in the `Fahrenheit` variable. Finally, the procedure returns the value of `Fahrenheit`.

This code is a simple implementation of a temperature conversion function. It takes a temperature in Celsius and returns the equivalent temperature in Fahrenheit. The code is easy to read and understand, and it follows good coding practices such as declaring local variables and using descriptive variable names.

To improve the code's readability, we could add comments to explain the purpose of the procedure and the calculations being performed. We could also use more descriptive variable names to make the code easier to understand. In terms of performance, this code is already quite efficient, but we could consider using a more optimized algorithm if we needed to convert temperatures at a very high frequency.

AL ConvertTemperature.CodeUnit.al

```
2 references
1  codeunit 70074169 MS_ConvertTemperature
2  {
3      0 references
4      trigger OnRun()
5      begin
6          Message('Yes, access to the codeunit!');
7      end;
8
9      1 reference
10     procedure ConvertFahrenheitToCelsius(Fahrenheit: Decimal): Decimal
11     var
12         Celsius: Decimal;
13         begin
14             Celsius := (Fahrenheit - 32) * 5 / 9;
15             exit(Celsius);
16         end;
17
18     1 reference
19     procedure ConvertCel
20     var
21         Fahrenheit: Deci
22         begin
23             Fahrenheit := Ce
24             exit(Fahrenheit)
25         end;
26 }
```

Go to Definition F12 imal

Go to Implementations Ctrl+F12

Go to References Shift+F12

Peek >

Find All References Shift+Alt+F12

Find All Implementations

Show Type Hierarchy

Copilot >

- Start Code Chat Ctrl+I
- Explain This
- Fix This
- Generate Docs
- Generate Tests

PROBLEMS 1 OUTPUT DEBUG CONSOLE

# What is GitHub Copilot Chat?



A chat interface that lets you interact with GitHub Copilot, to ask and receive answers to coding-related questions within VS Code



Can answer a wide range of coding-related questions on topics including syntax, programming concepts, test cases, debugging, and more.



Use cases: Generating unit test cases, explaining code, proposing code fixes, answering coding questions



It is an assistance tool, not a replacement for human programming.  
Remember to review generated content.



<https://docs.github.com/en/copilot/github-copilot-chat/about-github-copilot-chat>

# DEMO: GitHub Copilot Chat

# How to get started

Requires GitHub Copilot

<https://docs.github.com/en/copilot>

<https://docs.github.com/en/billing/managing-billing-for-github-copilot/about-billing-for-github-copilot>

Install and configure GitHub Copilot Chat

<https://docs.github.com/en/copilot/github-copilot-chat/using-github-copilot-chat>

# Agenda

VS Code Extension  
Namespace  
ALDoc  
Investigate and troubleshoot  
AL Language  
Code authoring  
GitHub Copilot Chat  
**AppSource**

# The transactability roadmap for Business Central apps



Enabling transactability for Business Central Apps with a '[per-user](#)' pricing model in a Microsoft AppSource [direct sales](#) channel motion.

Current public preview program



Adding transactability capabilities which enables '[per-user](#)' selling through a '[managed \(CSP\) reseller network](#)'.

Private preview from mid Sept 2023



# "AppSource Transact" session from 2023 Release Wave 1

[aka.ms/BCPartnerPortal](http://aka.ms/BCPartnerPortal)



Choose your path

Build a Practice

Drive Readiness

Go to Market

Events

## New marketing campaign for Business Central

Achieve your demand generation goals for Business Central and learn how marketing automation can help you grow your business.

[Watch the Business Central partner webinar](#)

[Download the webinar presentation](#)

### Business Central Launch Event (2023 Release wave 1)

- [Slide decks for BCLE rw1 2023](#)
- [Session list for BCLE rw1 2023](#)
- [Introducing best practices for building Power Apps based solutions for Business Central](#)
- [Introducing master data management](#)
- [What's new in financial Management - GL Account Review](#)
- [What's new in financial Management - Intercompany Postings](#)
- [What's new in financial Management - Statistical Accounts](#)
- [What's new in Power Automate and approval workflows](#)

### Directions NA and Directions ASIA (April 2023)

Watch the Microsoft keynotes and get access to the Microsoft breakout session content.

Directions conferences in April 2023

- [Microsoft keynote day 1 – Business Central, D365 Sales, what's new and roadmaps](#)
- [Microsoft keynote day 2 – Partner announcements, programs, incentives, tooling, and services](#)
- [Microsoft keynote day 3 – Power Platform](#)
- [Directions NA – Microsoft breakout session decks](#)
- [Directions ASIA – Microsoft breakout session decks](#)
- [Microsoft keynote slides from Directions NA 2023](#)

Microsoft Dynamics 365

### Transact capabilities for Business Central apps



# ConvertTemperature sample app on GitHub

<https://github.com/BusinessCentralDemos/ConvertTemperature/blob/main/ConvertTemperature/ConvertTemperature.al>

The image displays two side-by-side screenshots. On the left is the Microsoft AppSource page for the 'ConvertTemperature' app. The app's logo features a blue-to-orange gradient background with '-40°' in white. It is categorized under 'Dynamics 365 Business Central'. Below the logo are buttons for 'Buy now' and 'Contact me'. The 'Plans + Pricing' tab is selected, showing three plans: Gold (\$0.03/user/month), Silver (\$0.02/user/month), and Bronze (First month free, then \$0.01/user/month). The 'Overview' and 'Ratings + reviews' tabs are also present. On the right is a screenshot of the GitHub repository for 'BusinessCentralDemos / ConvertTemperature'. The repository has 0 forks and 0 stars. The code editor shows the file 'ConvertTemperature.al' with the following content:

```
permissionset 70074169 "MS_SamplePS"
{
    Assignable = true;
    Caption = 'sample permission set';
    IncludedPermissionSets = MS_DefaultPS;
    Permissions = codeunit MS_ConvertTemperature = X;

    permissionset 70074170 "MS_DefaultPS"
    {
        Assignable = true;
        Caption = 'sample permission set';
        Permissions = page MS_ConvertTemperature = X;
    }

    entitlement "MS_Gold"
    {
        Type = PerUserOfferPlan;
        Id = 'mdcc1667400477212.bc_converttemp_sample.gold';
        ObjectEntitlements = "MS_SamplePS";
    }

    entitlement "MS_Silver"
    {
        Type = PerUserOfferPlan;
```

# Entitlement examples on Microsoft Learn

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-entitlement-object#entitlement-examples>

## Entitlement examples

This example illustrates a simple entitlement object with the `Type` property set to `Role`, which means that the entitlement is associated with an AAD role. When `Type` is set to `Role`, the `RoleType` property is used to distinguish between local and delegated assignments of the role, in this case it is `Delegated`. The `ObjectEntitlements` property defines the list of permissions that the entitlement includes.

```
AL

entitlement BC_Role_Delegated
{
    Type = Role;
    RoleType = Delegated;
    Id = '1a2aaaaa-3aa4-5aa6-789a-a1234567aaaa';
    ObjectEntitlements =
        "D365 BUS PREMIUM - BaseApp";
}
```

An example of an entitlement where `Type` is `PerUserServicePlan`:

```
AL      ("AppSource Transact" session f..)

entitlement BC_PerUserServicePlan
{
    Type = PerUserServicePlan;
    Id = '1a2aaaaa-3aa4-5aa6-789a-a1234567aaaa';

    ObjectEntitlements = "D365 BASIC";
}
```

# New developer resources

## HttpClient

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/methods-auto/httpclient/httpclient-data-type>

## Reporting

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-reports>

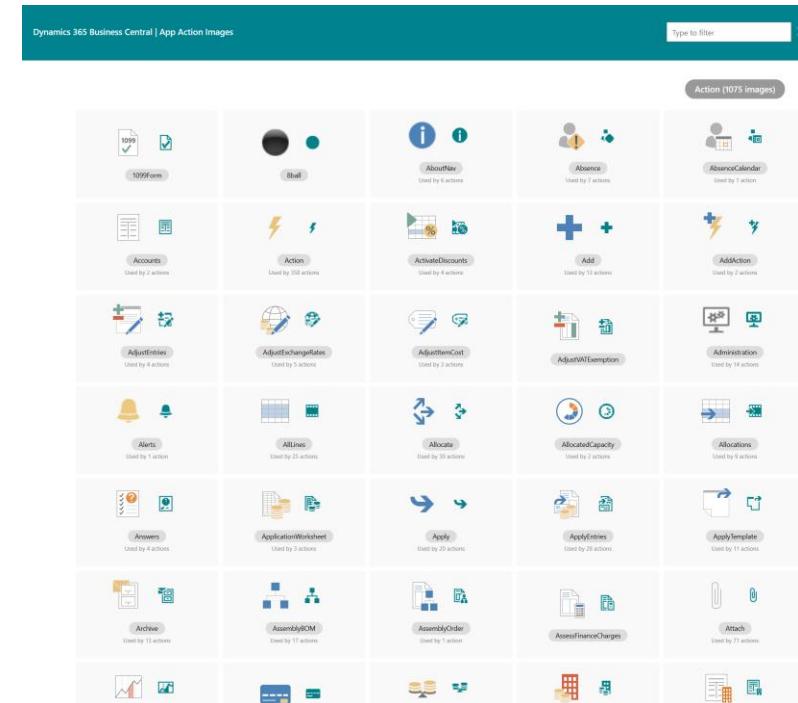
## Extensibility

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-extensibility-overview>

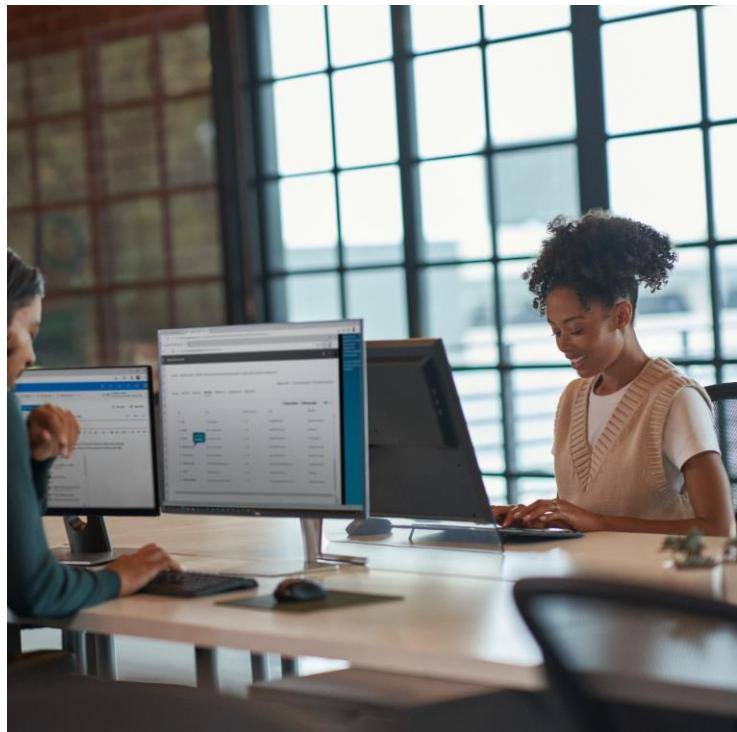
## BCPT

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-performance-toolkit>

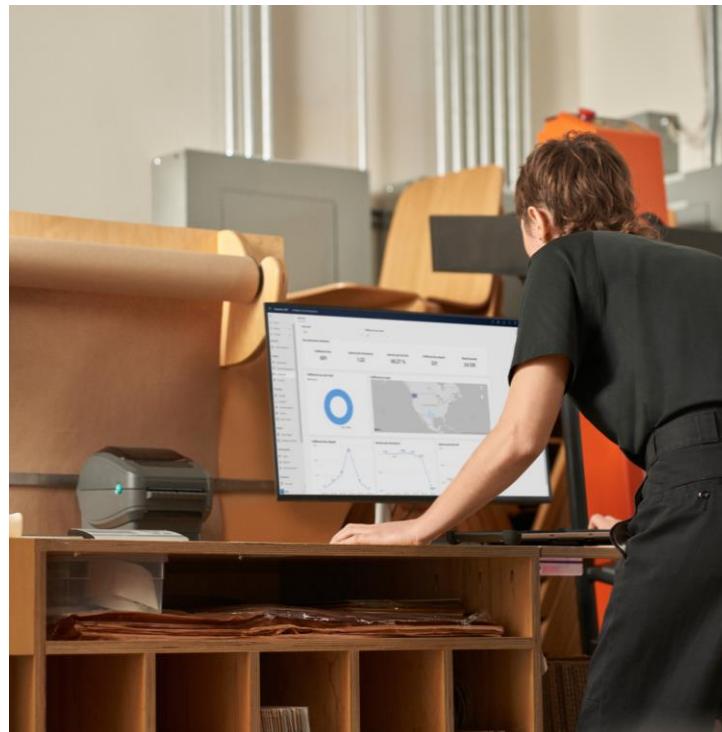
aka.ms/BCIcons – Overview of Icons for developers



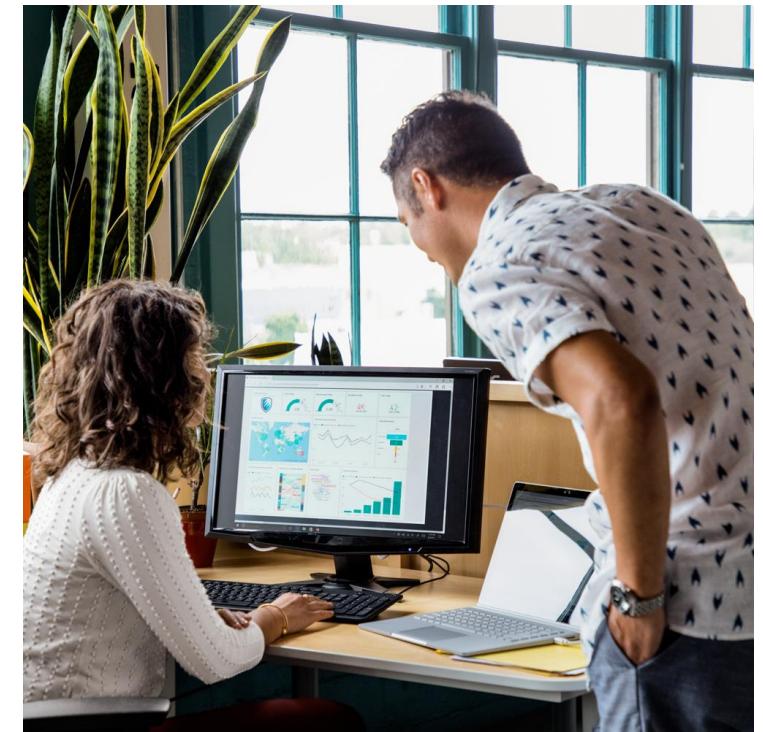
# Watch other relevant launch event sessions



What's new: AL-Go for GitHub



What's new: Performance testing before go-live with Performance Toolkit

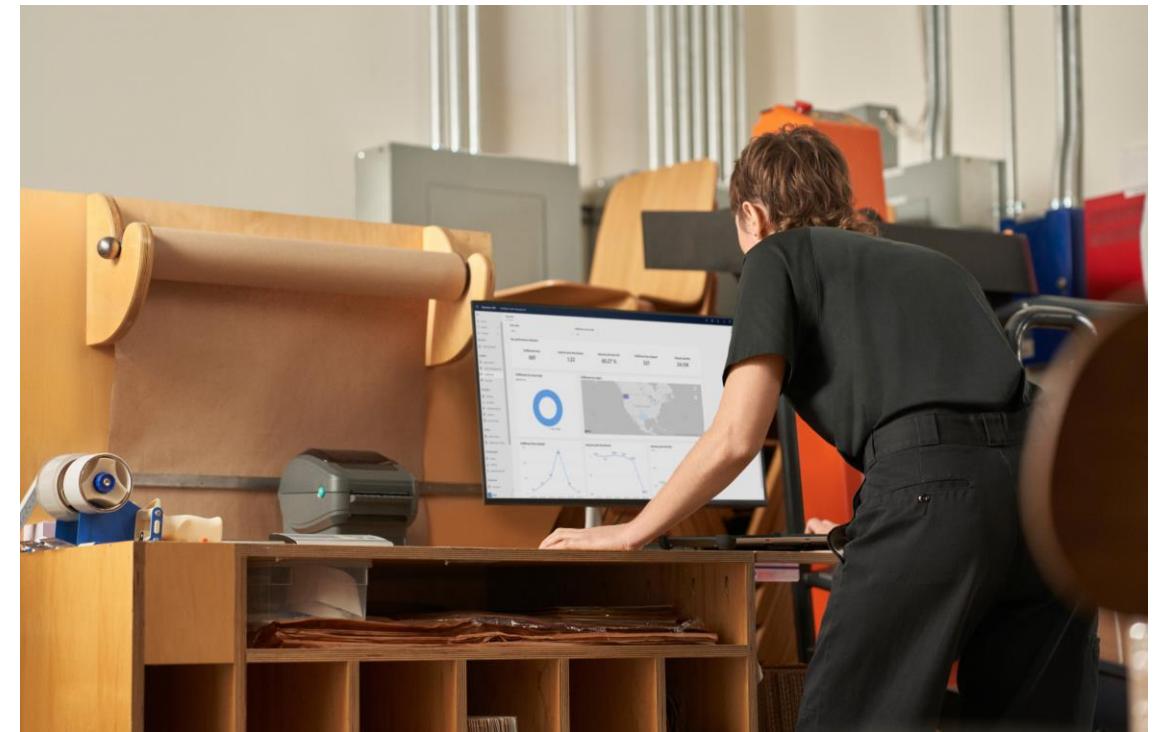


What's new: Telemetry

# Watch other relevant launch event sessions



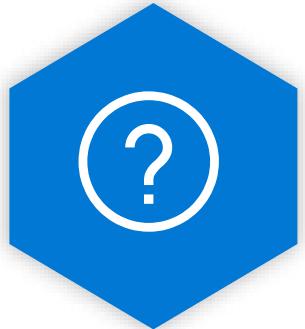
What's new: Creating customer centric onboarding experiences



What's new: Power BI and reporting (for developers)

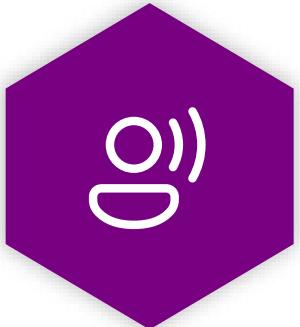
# General Business Central resources, learn more!

Join the  
conversation  
[twitter.com/  
MSDyn365BC](https://twitter.com/MSDyn365BC)



Have a  
question?

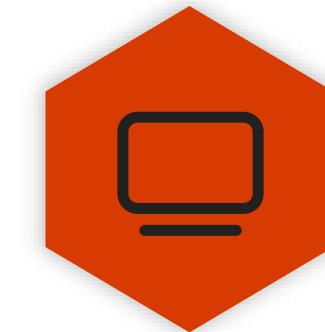
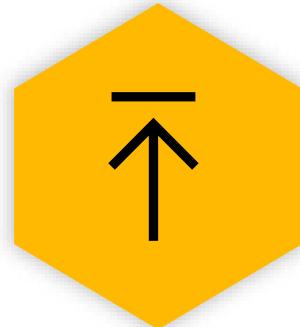
[aka.ms/BCYammer](https://aka.ms/BCYammer)



Looking for  
resources?

[aka.ms/BCAll](https://aka.ms/BCAll)

Submit  
your ideas  
[aka.ms/BCIdeas](https://aka.ms/BCIdeas)



Join the office  
hours

[aka.ms/BCTeamOfficeHours](https://aka.ms/BCTeamOfficeHours)

Thank you



# Syntax highlighting in DevOps

Syntax highlighting for AL in Azure DevOps can be enabled by installing the following free extension in Azure DevOps:

<https://marketplace.visualstudio.com/items?itemName=Dynamics365BusinessCentral.allanghighlights>

```
1 namespace MyAwesomeAIApp.Demo;
2
3 using Microsoft.Sales.Customer;
4
5 pageextension 54321 "Customer Card Page Ext" extends "Customer Card"
6 {
7     layout
8     {
9         addlast(General)
10        {
11            group("Group with Rich Text Editor")
12            {
13                // Remember to check out all the other fantastic developer additions, like the rich text editor
14                field(Spoilers; GlobalText)
15                {
16                    Multiline = true;
17                    ExtendedDataType = RichContent;
18
19                    trigger OnValidate()
20                    begin
21                        // Here you should persist your GlobalText variable by using the Streams pattern
22                    end;
23                }
24            }
25        }
26    }
27
28    actions
29    {
30        addlast("&Customer")
31        {
32            action("Show Greeting")
33            {
34                Image = CheckDuplicates;
35                ApplicationArea = All;
36                RunObject = codeunit HelloWorld;
37
38                trigger OnAction()
39                var
40                    TextVar: Text;
41                    IntegerVar: Integer;
42
43                    begin
44                        IntegerVar := 5;
45                        IntegerVar += 12345;
46                        TextVar := 'Hello, Release Note reader!';
47                        Message(TextVar + 'This is the Customer Card');
48                    end;
49            }
50        }
51
52        var
53            GlobalText: Text;
54            StreamIn: InStream;
55            StreamOut: OutStream;
56    }
```