

**NAV  
TECH  
DAYS  
2017**

mibuso.com

# DEEP DIVE INTO THE NEW DEVELOPMENT TOOLS

STANISLAW STEMPIN, JESPER SCHULZ-WEDDE, ESSEN NYHUUS KRISTOFFERSEN  
MICROSOFT DEVELOPMENT CENTER COPENHAGEN

**WHEN YOU ARE PASSIONATE ABOUT MICROSOFT DYNAMICS NAV | [www.navtechdays.com](http://www.navtechdays.com)**

**NAV  
TECH  
DAYS  
2017**

mibuso.com

# Disclaimer

This presentation is provided for informational and training purpose only. It represents Microsoft's view as of the presentation date. Microsoft cannot guarantee the accuracy of any information after the presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted as a commitment on Microsoft's part. This presentation is provided "as-is".

Sample code included in this presentation is made available AS IS.

Navision\_main - Microsoft Dynamics NAV Development Environment

File Edit View Tools Window Help

Object Designer

Type	ID	Name	Modified	Version List
Table	71	Purch.-Disc. (Yes/No)		NAVW110.0
Page	73	Purch.-Explode BOM		NAVW111.0
Report	74	Purch.-Get Receipt		NAVW111.0
	76	Purch.-Get Drop Shpt.		NAVW111.0

Codeunit 80 Sales-Post - C/AL Editor

```

1 Documentation()
2
3 OnRun(VAR Rec : Record "Sales Header")
4   OnBeforePostSalesDoc(Rec);
5
6   ValidatePostingAndDocumentDate(Rec);
7
8   IF PreviewMode THEN BEGIN
9     CLEARALL;
10    PreviewMode := TRUE;
11  END ELSE
12    CLEARALL;
13
14  GetGLSetup;
15  GetCurrency("Currency Code");
16
17  SalesSetup.GET;
18  SalesHeader := Rec;
19  FillTempLines(SalesHeader);
20  TempServiceItem2.DELETEALL;
21  TempServiceItemComp2.DELETEALL;
22
23  // Header
24  CheckAndUpdate(SalesHeader);
25

```

100 %

OnRun: Ln 11 Col 12

EUROPE\esbenk

# To-do list for a new Developer Experience

- State of the art development experience
- Better development story for extensions
- Protect existing partner investments
- Support for standard SCM – file based
- Ability to add new features quickly
  - Faster compile time
  - Componentized compiler
- Support for modularization
- Cloud deployment
- Appeal to new developers



Visual Studio Code - Cc X +

code.visualstudio.com

Visual Studio Code

Docs

Updates

Blog

Community

Extensions

FAQ

Search Docs

Download

Version 1.15 is now available! Read about the new features and fixes in July.

Code editing.  
Redefined.

Free. Open source. Runs everywhere.

Download for Windows  
Stable Build

Other platforms and Insiders Edition

By using VS Code, you agree to its  
license and privacy statement.

www.ts - node-express-ts - Visual Studio Code

File Edit View Goto Help

EXTENSIONS

@popular

C# 1.2.2 356K ★★★★★

C# for Visual Studio Code (p...

Microsoft

Install

Python 0... 211K ★★★★★

Linting, Debugging (multi-t...

Don Jayamanne

Install

Debugger for Chrome 148

Debug your JavaScript code...

Microsoft JS Diagno...

Install

C/C++ 0.7... 143K ★★★★★

Complete C/C++ language ...

Microsoft

Install

Go 0.6.39 99K ★★★★★

Rich Go language support f...

lukehoban

Install

ESLint 0.10... 88K ★★★★★

Integrates ESLint into VS Co...

Dirk Baeumer

Install

app.ts

1 import app from './app';

2 import debugModule = require('debug');

3 import http = require('http');

4

5 const debug = debugModule('node-express-typescript:server');

6

7 // Get port from environment and store in Express.

8 const port = normalizePort(process.env.PORT || '3000');

9 app.set('port', port);

10

11 // create

12 const ser

13 server.li

14 server.on

15 server.on

16

17 /\*\*

18 \* Normal

19 \*/

20 function normalizePort(val: any): number|string|boolean {

21 let port = parseInt(val, 10);

22

package.json

README.md

Ln 9, Col 21

Spaces: 2

UTF-8

LF

TypeScript

IntelliSense

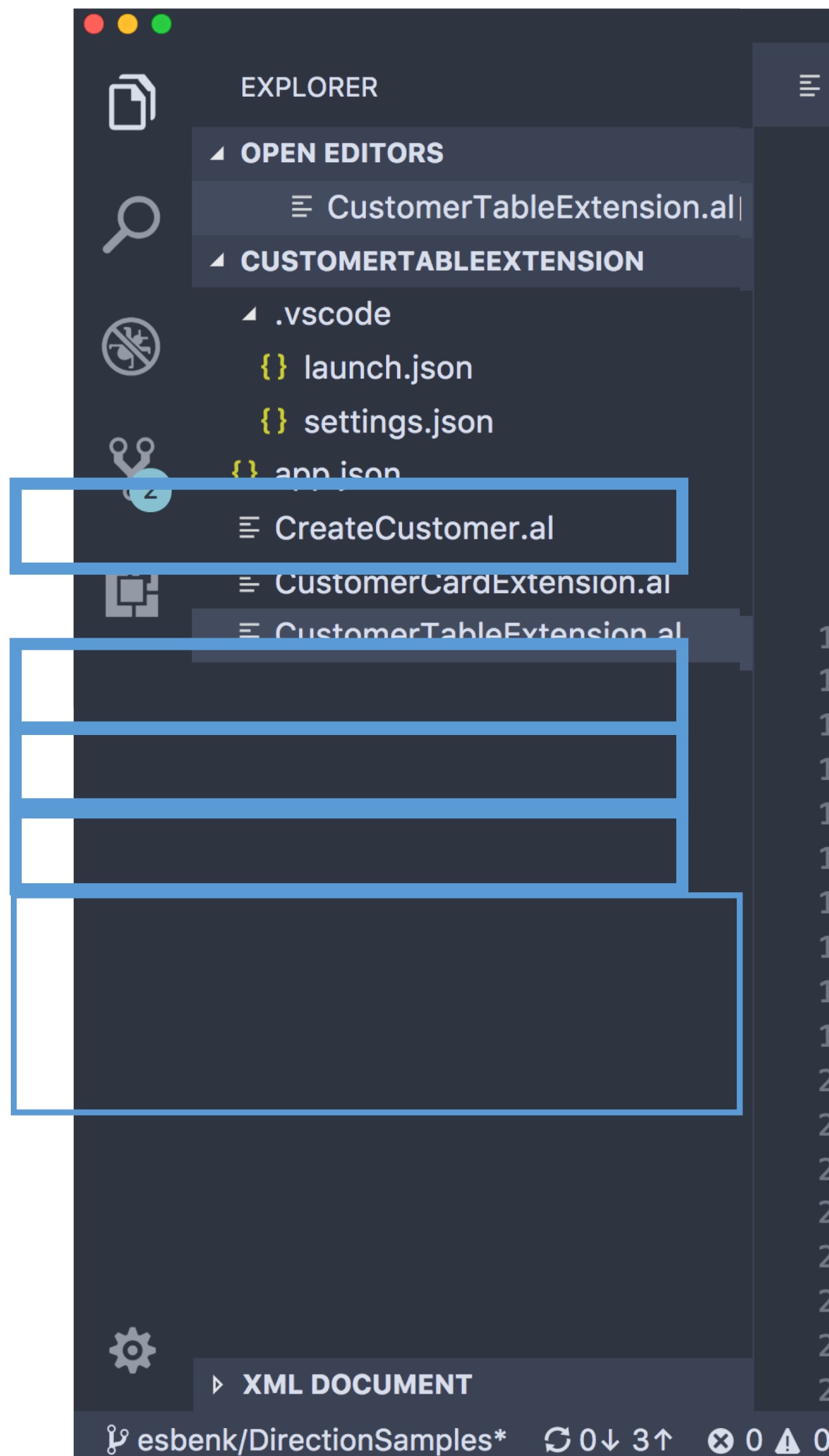
Debugging

Built-in Git

Extensions

# Demo – Hello World

# Project = Folder



## Project = Folder

Visual Studio Code file folders use the information related to launching and debugging the app, as well as the app.json file for publishing to AppSource. Our configuration contains: All files in the folder are also saved in the app, as in Visual Studio Code. Authentication typed in SCM

It should NOT be included in SCM

# JSON (JavaScript Object Notation)

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

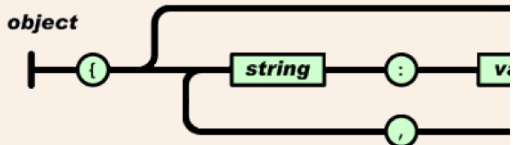
JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is called a *dictionary*, *hash*, *map*, or *record*.
- An ordered list of values. In most languages, this is called a *list*, *array*, or *vector*.

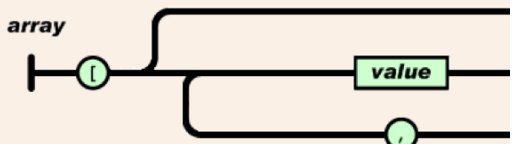
These are universal data structures. Virtually all modern programming languages support them.

In JSON, they take on these forms:

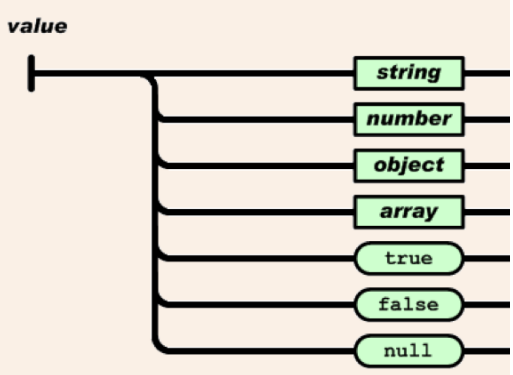
An *object* is an unordered set of name/value pairs. An object is written as follows:



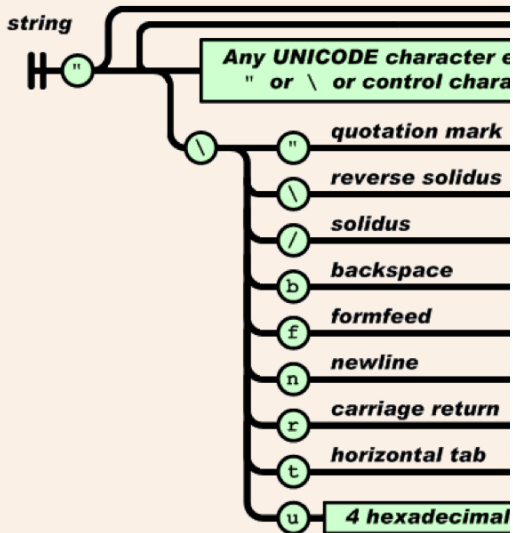
An *array* is an ordered collection of values. An array is written as follows:



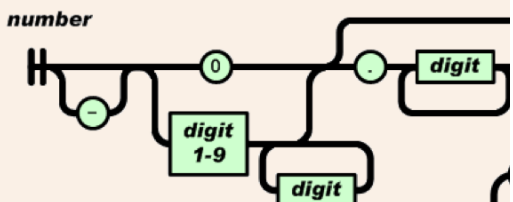
A *value* can be a *string* in double quotes, or a *number*, or an *object*, or an *array*, or the *true* value, or the *false* value, or the *null* value.



A *string* is a sequence of zero or more Unicode characters. A string is written as follows:



A *number* is very much like a C or Java number, except that it may not have leading zeros. A number is written as follows:



```
{
  "id": "1d6b6366-cc04-4b3e-a426-4daca84b85e1",
  "name": "ALProject1",
  "publisher": "Default publisher",
  "brief": "",
  "description": "",
  "version": "1.0.0.0",
  "compatibilityId": "1.0.0.0",
  "privacyStatement": "",
  "EULA": "",
  "help": "",
  "url": "",
  "logo": "",
  "capabilities": [],
  "dependencies": [],
  "screenshots": [],
  "platform": "11.0.0.0",
  "application": "11.0.0.0",
  "idRange": {
    "from": 50100,
    "to": 50149
  }
}
```

**object**

- { }
- { *members* }

**members**

- pair*
- pair* , *members*

**pair**

- string* : *value*

**array**

- [ ]
- [ *elements* ]

**elements**

- value*
- value* , *elements*

**value**

- string*
- number*
- object*
- array*
- true**
- false**
- null**

**string**

- " "
- " *chars* "

**chars**

- char*
- char* *chars*

**char**

- any-Unicode-character-except-"-or-\-or-control-character*
- \ "
- \\
- \/
- \b
- \f
- \n
- \r
- \t
- \u *four-hex-digits*

**number**

- int*
- int frac*
- int exp*
- int frac exp*

**int**

- digit*
- digit1-9 digits*
- *digit*
- *digit1-9 digits*

**frac**

- . *digits*

**exp**

- e digits*

**digits**

- digit*
- digit digits*

**e**

- e**
- e+**
- e-**
- E**
- E+**
- E-**



# Demo – Hello World, Continued

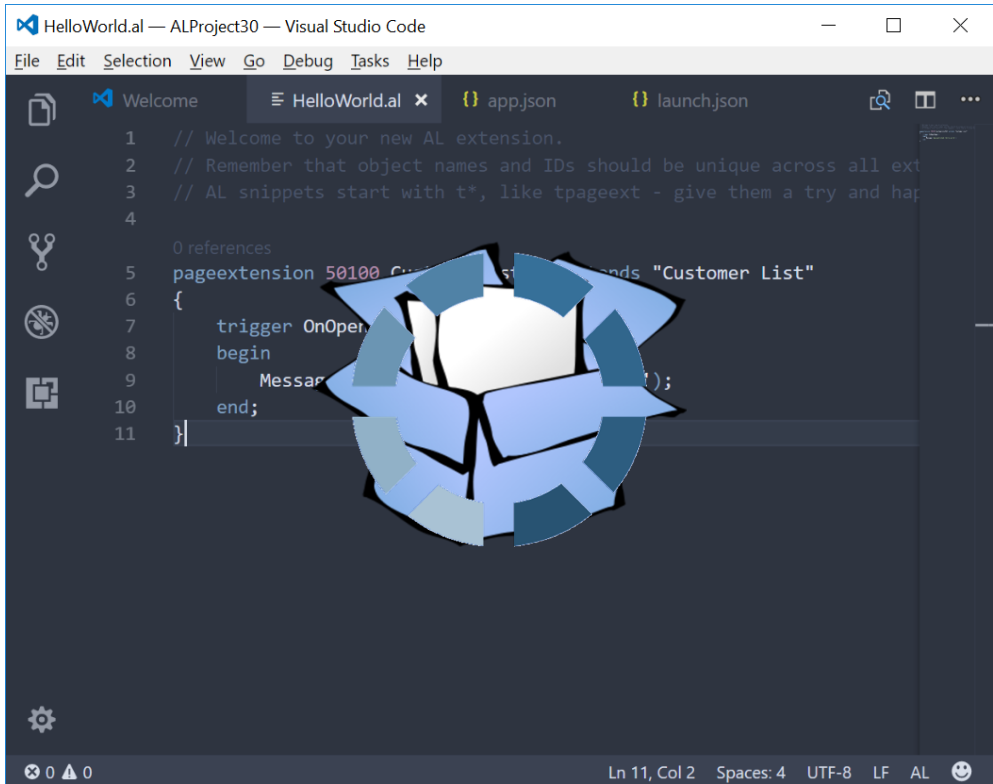
# What did we just see

Visual Studio Code with the AL Language extension

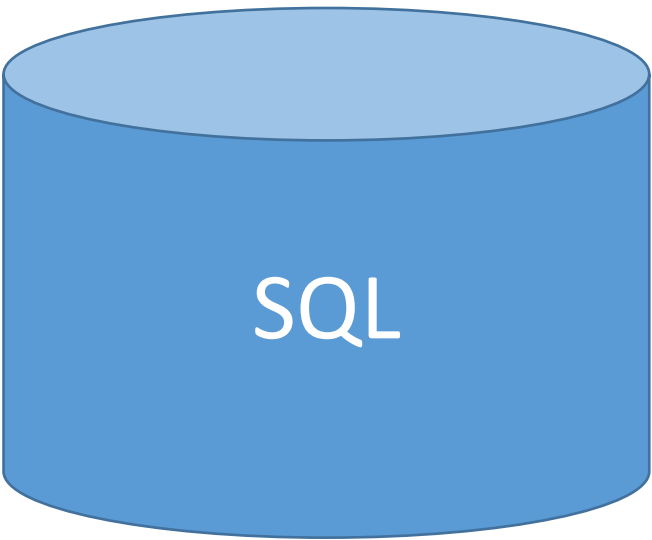
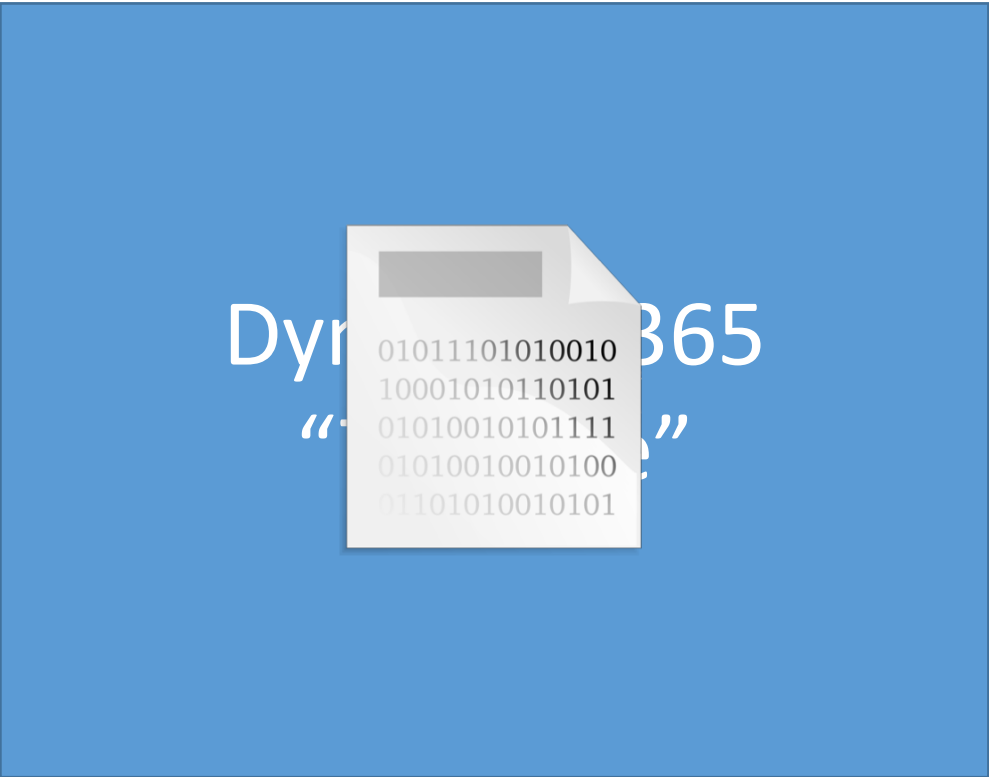
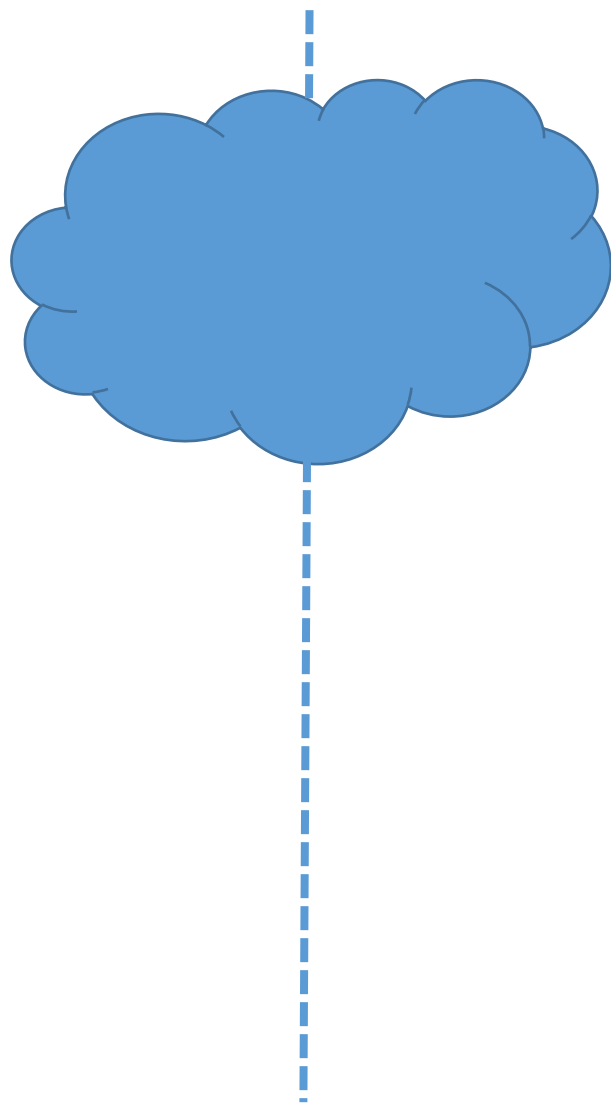
App deployed to a Dynamics 365 in Azure/On-Prem

A simple page extension

# How does the deployment work

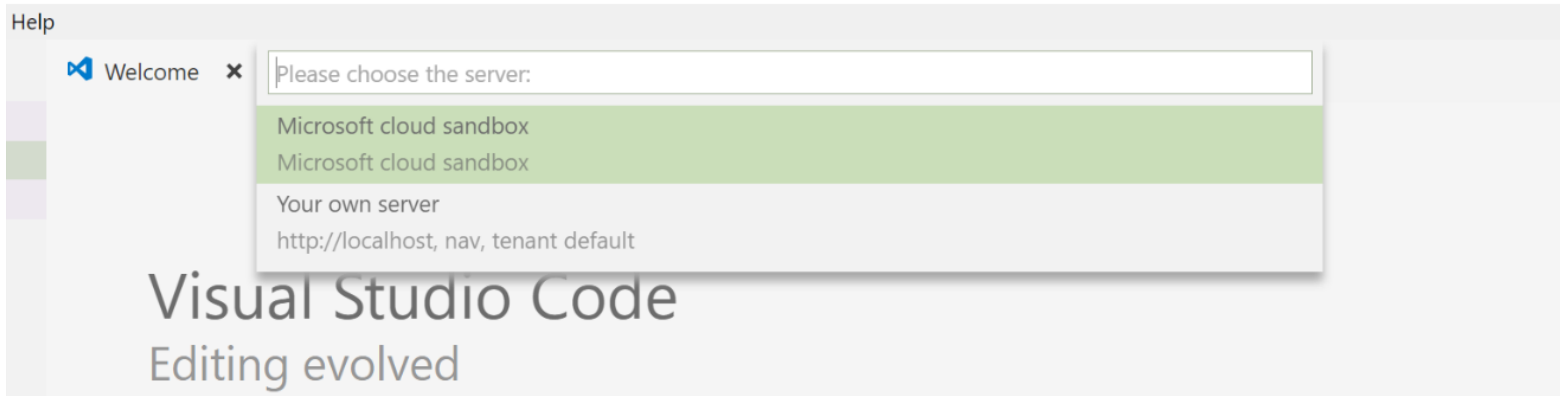


Ctrl+F5 (⌘+F5)



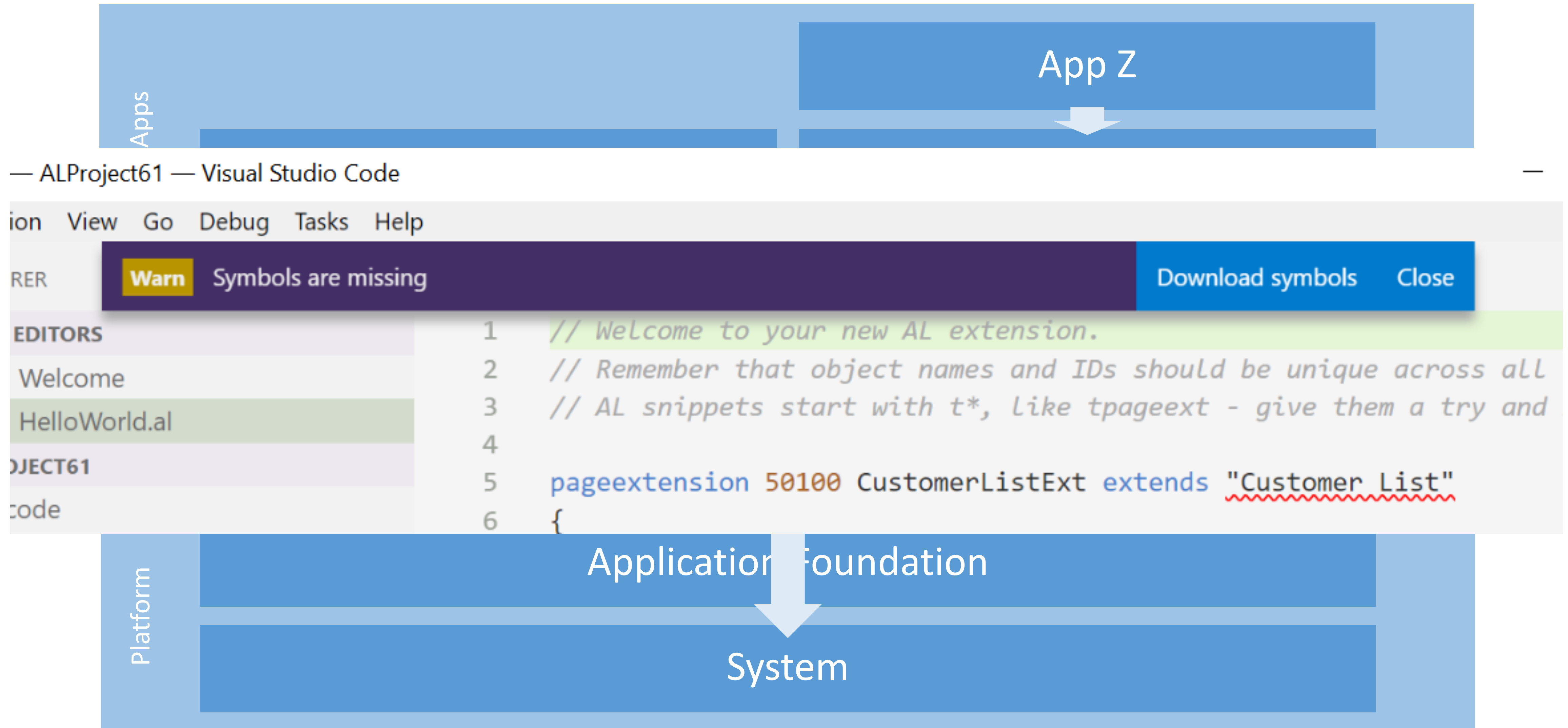
# Deployment Options

launch.json





# Symbols and Modules



# The syntax

```
49  |  dr  |  Codeunit.al  |  Page.al  |  Table.al
50  |  1  |  page 50100 MyPage
6   |  layout {
7   |      area(content) {
8   |          group(GroupName) {
9   |              Caption = 'Nice Group';
10  |
11  |              field(Name;Name)
12  |              {
13  |                  Caption = 'Customer Name';
14  |                  Visible = true;
15  |                  ToolTip = 'This is the Customer name';
16  |
17  |                  trigger OnValidate();
18  |                  begin
19  |                      // Some validation
20  |                  end;
21  |              }
22  |          }
23  |      }
24  |  }
25  |
73  |  procedure CalcMe();
74  |  begin
75  |  end;
```

Properties

Triggers

INITIALIZATION

# Where are my designers

Address.al — getAddress — Visual Studio Code

File Edit Selection View Go Debug Tasks Help

Address.al x Service Connection Setup.al Address Serv

1 // -----

2 // Copyright (c) Microsoft Corporation. All right

3 // Licensed under the MIT License. See License.tx

4 // -----

5

6 table 70051000 Address

7 {

8 fields

9 {

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

19

20

21

field(2; Service URL ;Text[250])

{

Description = 'The Service URL of the service you want to connect to.'

ExtendedDataType = URL;

}

f

{

fields

{

PerfectUI.al — Debugger

Velcome {} app.json {} settings.json {} launch.json CustomerListExt.al FantasticCo

3

1 tp

tpage, Page of type card

tpage, Page of type list

tpagecust

tpageext

tprocedure

tprofile

Page of type card (al)

page Id PageName

{

PageType = Card;

SourceTable = TableName;

layout

{

area(content)

{

group(GroupName)

umbers property

as whether the system will clear a range of

s as it formats them.

Column CompanyAddr[1] CompanyAddr1

Column CustAddr[2] CustAddr2

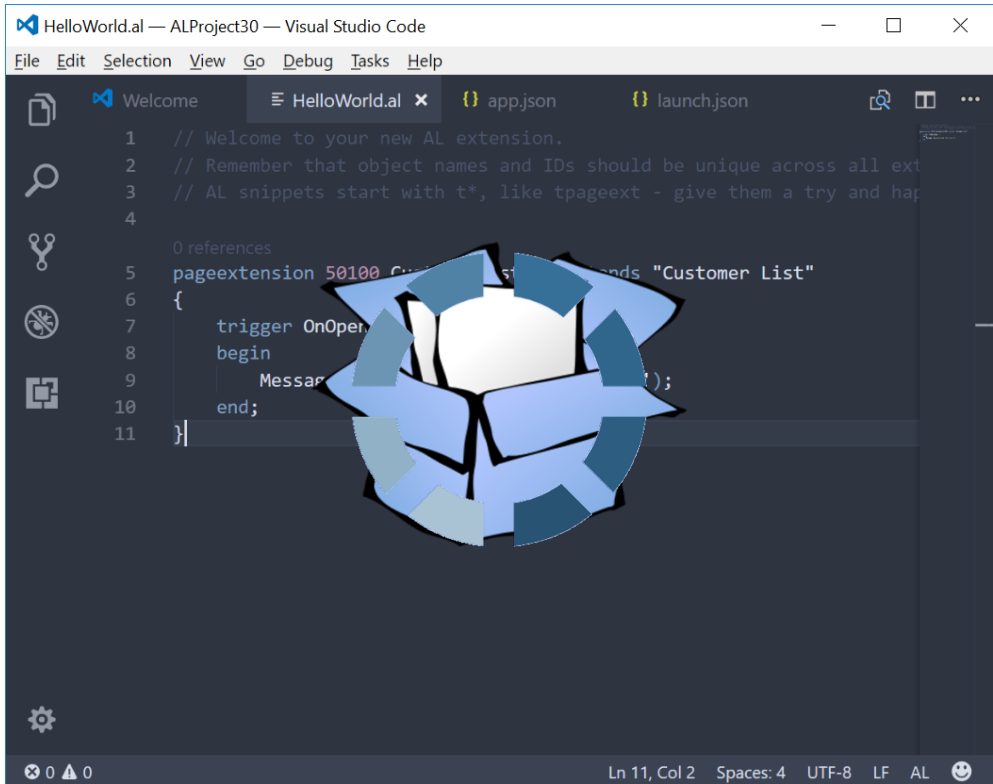
← → ↑ ↓

Help

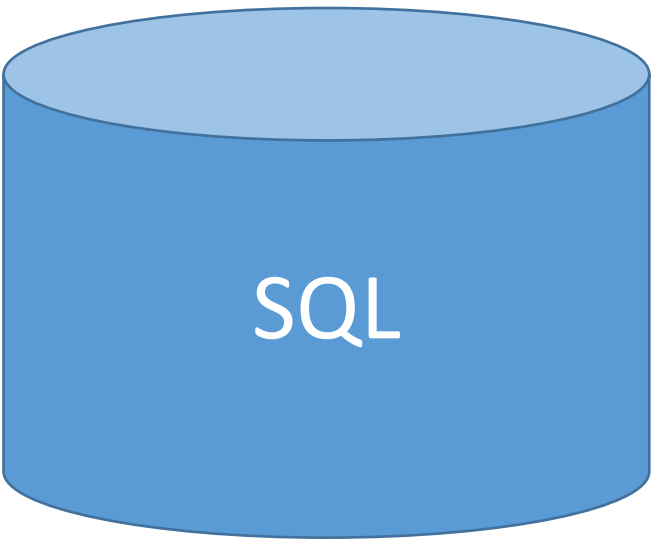
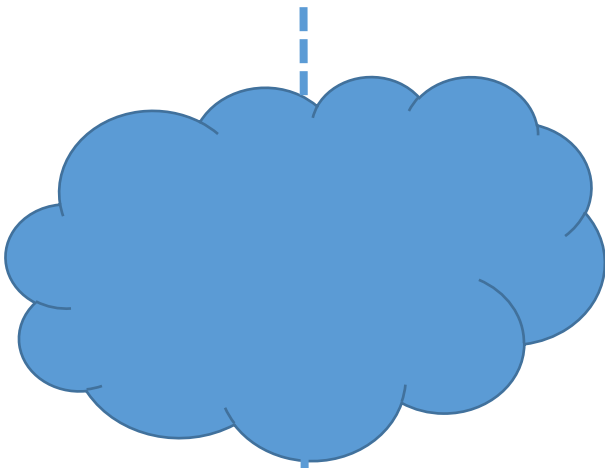
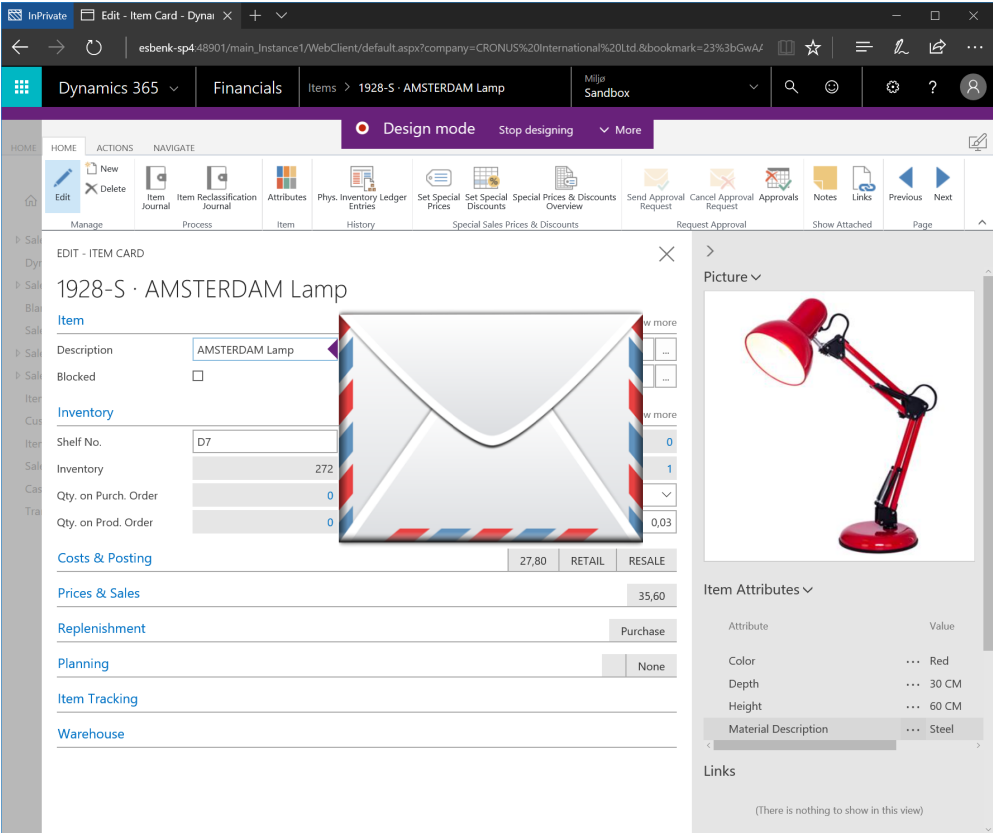
# Demo – Designer



# Designer



F7 extension=219165b6-9b1f-4204-9475-01657a0172ee



# Developing for the cloud

You are not alone in the cloud

Resources are shared => Resource Governance

Data is not shared => Sandboxing

## Restrictions

- Some platform API's are blocked
- Certain application methods are blocked
- .net interop is not supported

# What about .net

## New native AL types

JSON

XML

TextBuilder

HttpClient

List

Dictionary

String  
Functions

## Move .net functionality to Azure functions

Use native HttpClient to call Azure functions (or other web-services)

## Submit pull request to the C/AL Open Library

Requests will be sanitized and if accepted be included in the base application

<https://github.com/Microsoft/cal-open-library>

# JSON support

```
1  {
2    "books": [
3      {
4        "title": "Pride And Preju
5        "price": "24.95",
6        "genre": "novel",
7        "ISBN": "1-861001-57-8"
8      },
9      {
10       "title": "The Handmaid's
11       "price": "29.95",
12       "genre": "novel",
13       "ISBN": "1-861002-30-1"
14     },
15     {
16       "title": "Sense and Sensi
17       "price": "19.95",
18       "genre": "novel",
19       "ISBN": "1-861001-45-3"
20     }
21   ]
22 }
```

```
LOCAL PROCEDURE CreateBooks@7(
|   VAR JObject@1001 : DotNet "'Newtonsoft.Json, ...'.Newtonsoft.Json.Linq.JObject");
VAR
|   JArray@1000 : DotNet "'Newtonsoft.Json, ...'.Newtonsoft.Json.Linq.JArray";
|   'Newtonsoft.Json.Linq.JToken';
|   local procedure CreateBooks() JObject: JObject;
|   var
|       JArray: JSONArray;
|   begin
|       JArray.Add(
|           CreateBook('Pride And Prejudice', 24.95, 'novel', '1-861001-57-8'));
|       JArray.Add(
|           CreateBook('The Handmaid''s Tale', 29.95, 'novel', '1-861002-30-1'));
|       JArray.Add(
|           CreateBook('Sense and Sensibility', 19.95, 'novel', '1-861001-45-3'));
|       JObject.Add('books', JArray);
|   end;

|   local procedure CreateBook(
|       Title: Text; Price: Decimal;
|       Genre: Text; ISBN: Text) JObject: JObject;
|   begin
|       JObject.Add('title', Title);
|       JObject.Add('price', Format(Price));
|       JObject.Add('genre', Genre);
|       JObject.Add('ISBN', ISBN);
|   end;

|   JObject.Add(JProperty.JProperty('genre', Genre));
|   JObject.Add(JProperty.JProperty('ISBN', ISBN));
|   JToken := JObject;
END;
```



# XML Support

```
1  <?xml version="1.0" enc
2  <books>
3      <book genre="novel"
4          <title>Pride An
5          <price>24.95</p
6      </book>
7      <book genre="novel"
8          <title>The Hand
9          <price>29.95</p
10     </book>
11     <book genre="novel"
12         <title>Sense an
13         <price>19.95</p
14     </book>
15 </books>
```

```
LOCAL PROCEDURE CreateBooks@2(
    VAR Document@1003 : DotNet "'System.Xml, ...'.System.Xml.XmlDocument");
VAR
    Book@1000 : DotNet "'System.Xml, ...'.System.Xml.XmlElement";
    Books@1001 : DotNet "'System.Xml, ...'.System.Xml.XmlElement";

local procedure CreateBooks() : XmlDocument;
var
    Document : XmlDocument;
    Node: XmlNode;
begin
    Document.Add(
        XmlElement.Create('books',
            CreateBook('Pride And Prejudice', 24.95, 'novel', '1-861001-57-8'),
            CreateBook('The Handmaid's Tale', 29.95, 'novel', '1-861002-30-1'),
            CreateBook('Sense and Sensibility', 19.95, 'novel', '1-861001-45-3')));
    exit(Document);
end;

local procedure CreateBook(Title: Text; Price: Decimal;
    Genre: Text; ISBN: Text) Element: XmlElement;
begin
    Element := XmlElement.Create('book',
        XmlElement.Create('title', XmlText.Create(Title)),
        XmlElement.Create('price', XmlText.Create(Format(Price))));

    Element.Attributes.Set('genre', Genre);
    Element.Attributes.Set('ISBN', ISBN);
end;

Book.AppendChild(Element);

Book.SetAttribute('genre', Genre);
Book.SetAttribute('ISBN', ISBN);
END;
```



# StringBuilder

```
procedure BuildUsingText() Result : Text;  
var  
    i : Integer;  
begin  
    for i := 1 to 100000 do  
        Result += Format(i);  
    end;
```

25 sec

```
procedure BuildUsingBigText() Result : Text;  
var  
    i : Integer;  
    bt : BigText;  
begin  
    for i := 1 to 100000 do  
        bt.AddText(Format(i));  
    bt.GetSubText(Result, 1);  
end;
```

24 sec

```
procedure BuildUsingStringBuilder() : Text;  
var  
    i : Integer;  
    tb : StringBuilder;  
begin  
    for i := 1 to 100000 do  
        tb.Append(Format(i));  
    Exit(tb.ToText());  
end;
```

72 ms

# New Text Functions

```
procedure TextFunctions();
var
  t : Text;
  idx : Integer;
  b : Boolean;
  t2 : Text;
  list : List of [Text];
begin
  t := 'Hello World';

  b := t.Contains('llo'); // b = true

  b := t.EndsWith('Odd'); // b = false

  b := t.StartsWith('Hello'); // b = true

  idx := t.IndexOf('World'); // idx = 7

  idx := t.LastIndexOf('l'); // idx = 10

  t := 'Hi';
  t2 := t.PadLeft(4); // t2 = '  Hi'
  t2 := t.PadRight(4, '-'); // t2 = 'Hi--'

  t2 := t.ToLower(); // t2 = 'hi';

  t := 'Mon,Tue,Wed,Thu,Fri,Sat,Sun';
  list := t.Split(','); // list contains 7 weekdays

  // Fluent interface
  t := 'Hello World';
  b := t.ToUpper().Contains('WORLD'); // b = true
end;
```

# Generic types in AL – List, Dictionary

```
0 references
procedure CountCharactersInCustomerName(customerName : Text; var counter : Dictionary of [Char, Integer]);
var
    i : Integer;
    c : Integer;
begin
    clear(counter);

    for i := 1 to StrLen(customerName) do begin
        if counter.Get(customerName[i], c) then
            counter.Set(customerName[i], c + 1)
        else
            counter.Add(customerName[i], 1);
        end;
    end;
end;
```

```
1 reference
procedure PrintCustomerNames(customerNames : List of [Text]);
var
    customerName : Text;
begin
    foreach customerName in customerNames do
        Message(customerName);
    end;
end;
```

```
1 reference
procedure WorkWithListOfCustomers();
var
    customerNames : List of [Text];
begin
    // adding an element to a list
    customerNames.Add('John');

    // checking if the list contains an element
    if customerNames.Contains('John') then
        Message('John is in the list.')
    else
        Message('John is not in the list.');
```

```
    // getting an element using its index
    Message('Element retrieved at index - 1 : ' + customerNames.Get(1));

    // setting an element at an index
    customerNames.Set(1, 'Marc');

    // getting an element using its index
    Message('Element retrieved after updating element at index - 1 : ' + customerNames.Get(1));

    // inserting a new element at index 1
    customerNames.Insert(1, 'Simon');

    // getting an element using its index
    Message('Element retrieved after inserting element at index - 1 : ' + customerNames.Get(1));

    // removing an element at index 1
    customerNames.RemoveAt(1);

    // getting an element using its index
    Message('Element retrieved after deleting element at index - 1 : ' + customerNames.Get(1));

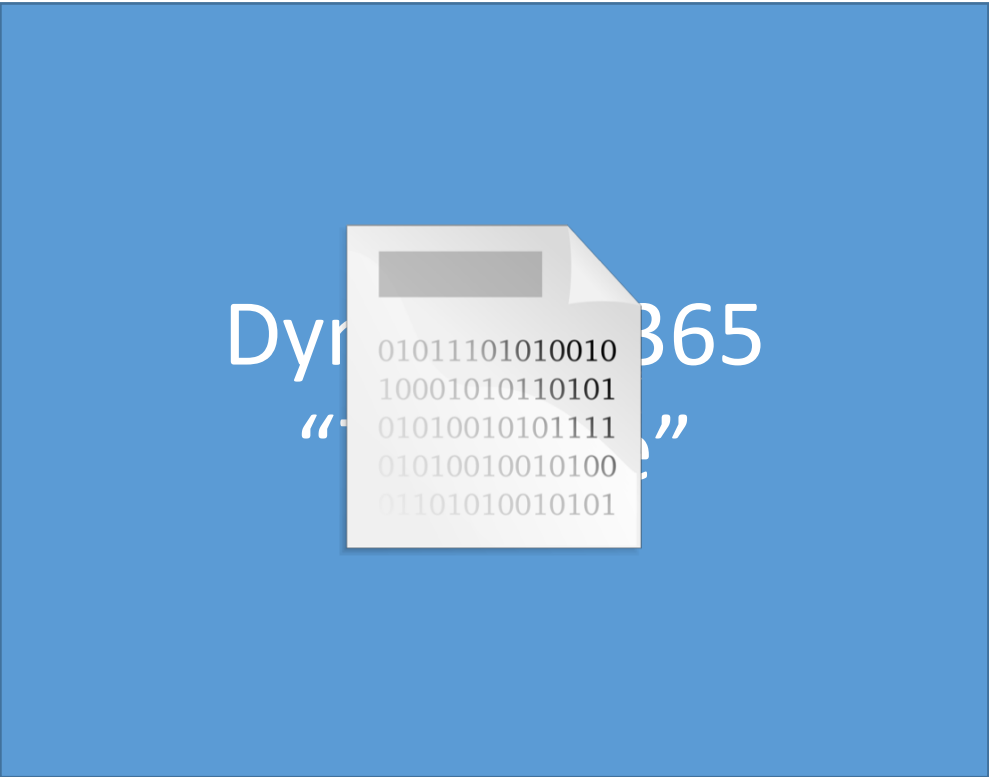
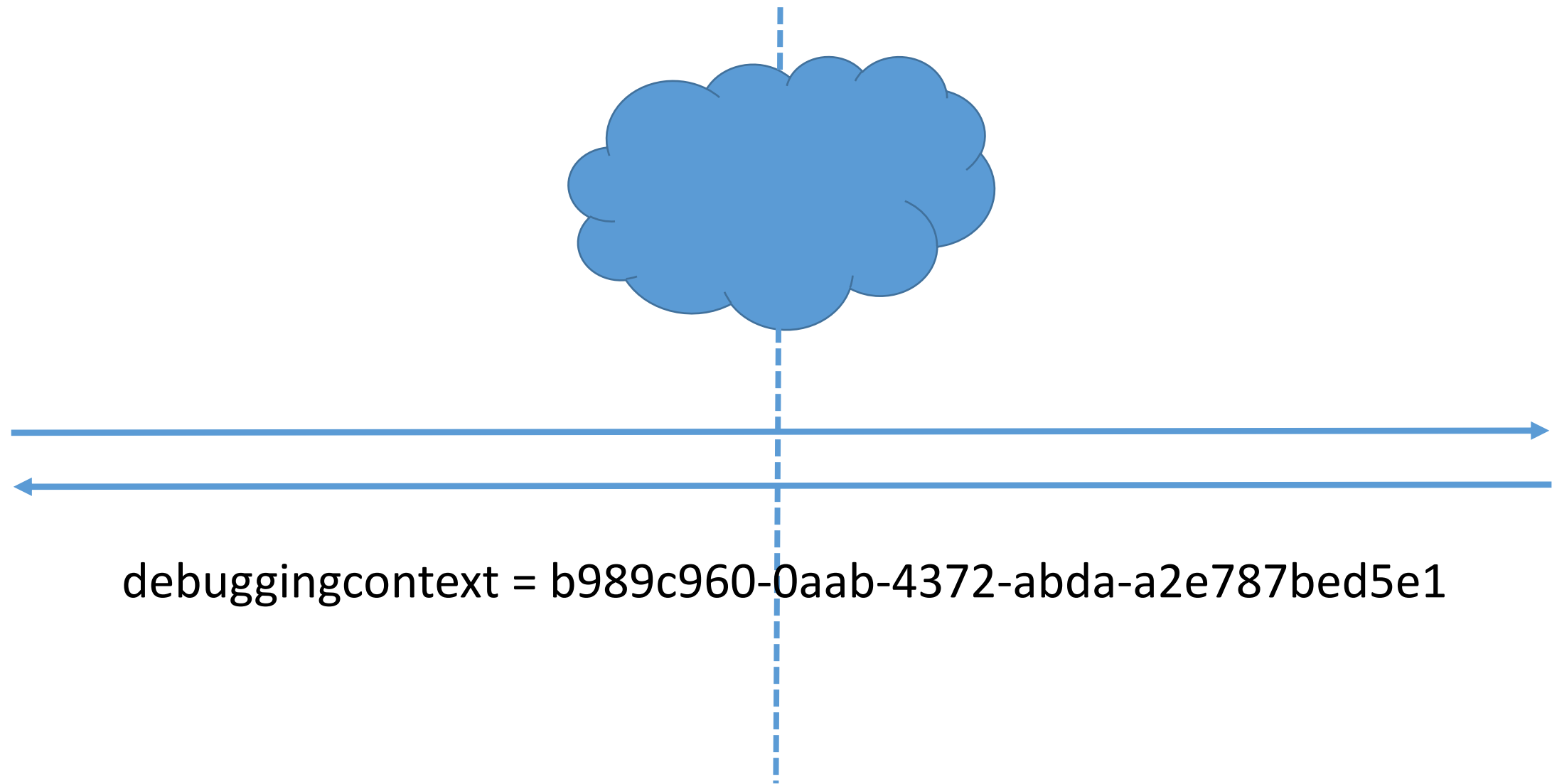
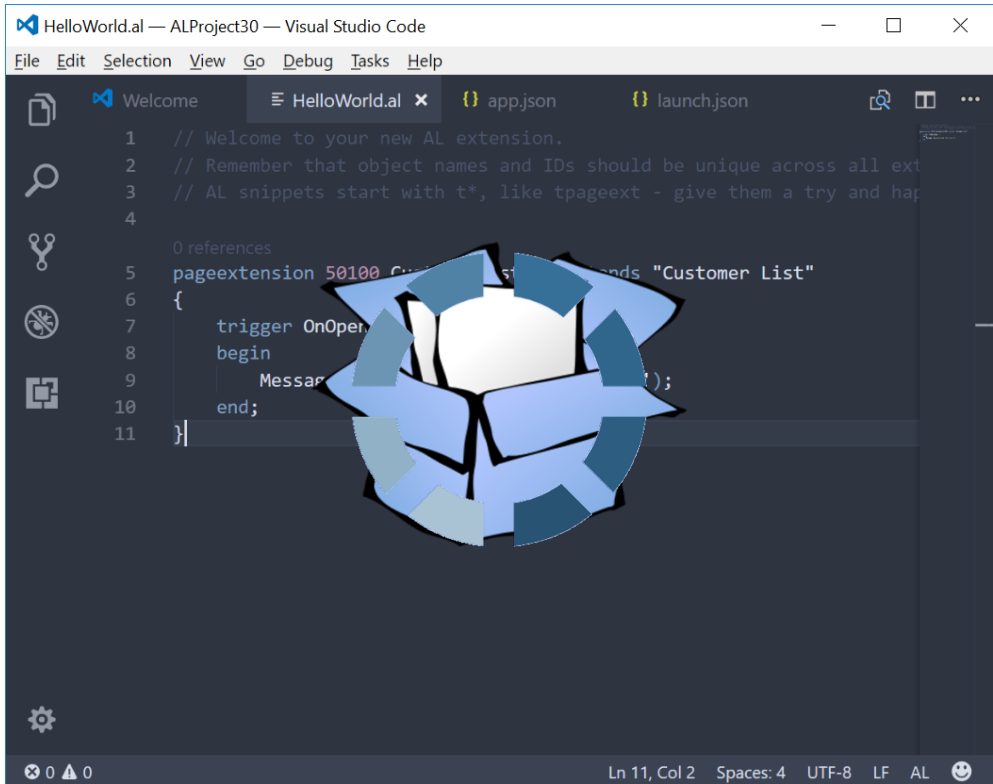
    // checking whether list has got any elements
    Message('Is List empty: ' + Format(customerNames.Count = 0));
end;
```

# Demo – Putting it together

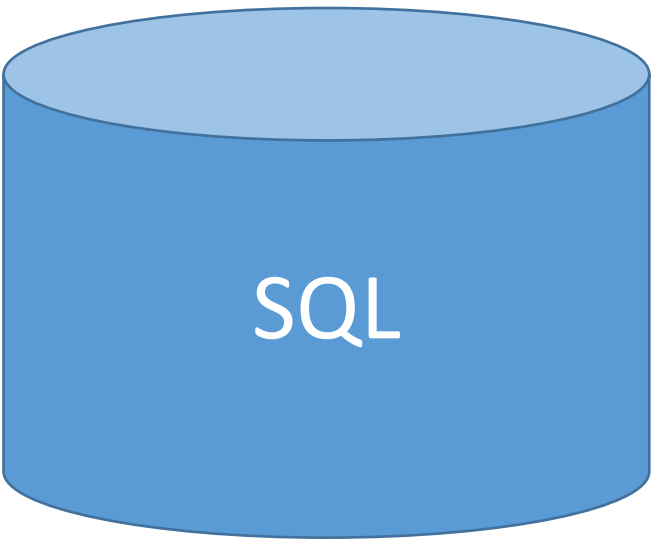
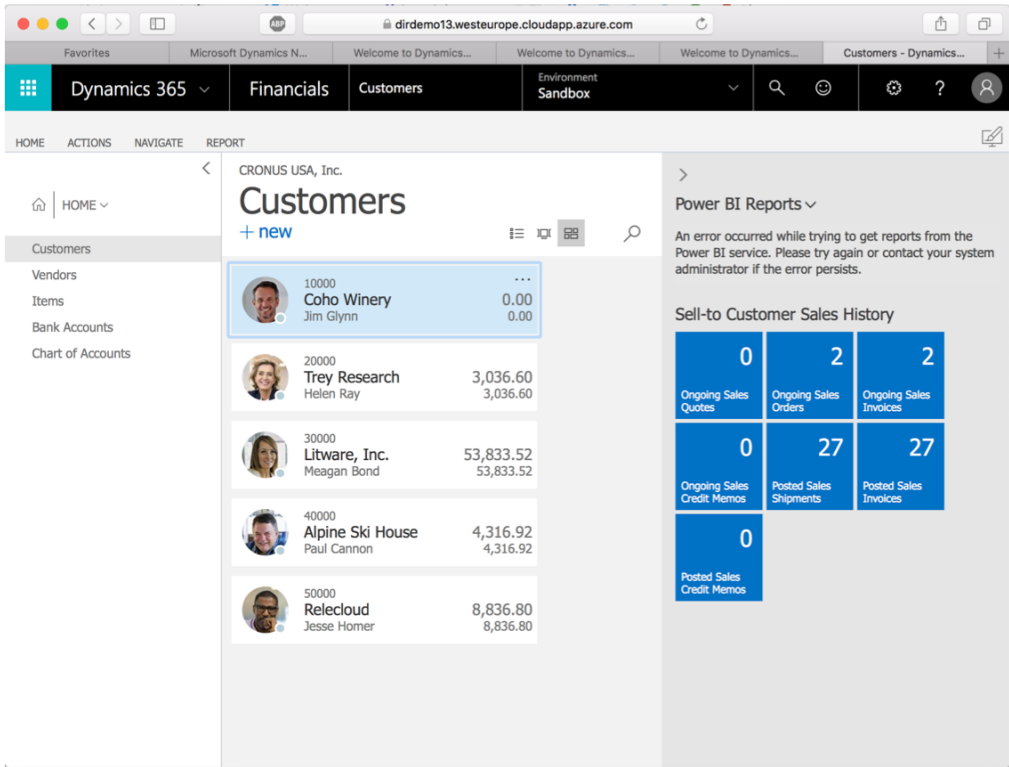
# Demo – Debugger



# Debugging flow



debuggingcontext = b989c960-0aab-4372-abda-a2e787bed5e1



# Dependencies

Apps can depend on apps

Dependencies are specified in app.json

```
"dependencies": [ {  
  "appId": "4805fd15-75a5-46a2-952f-39c1c4eab821",  
  "name": "WeatherLibrary",  
  "publisher": "Microsoft",  
  "version": "1.0.0.0"  
},
```

Symbols are the key

# Demo – Client Add-ins

# Embedded resources

## PermissionSet

Export-NAVAppPermissionSet

## Translations

C/SIDE → Export translation

## Extension table data

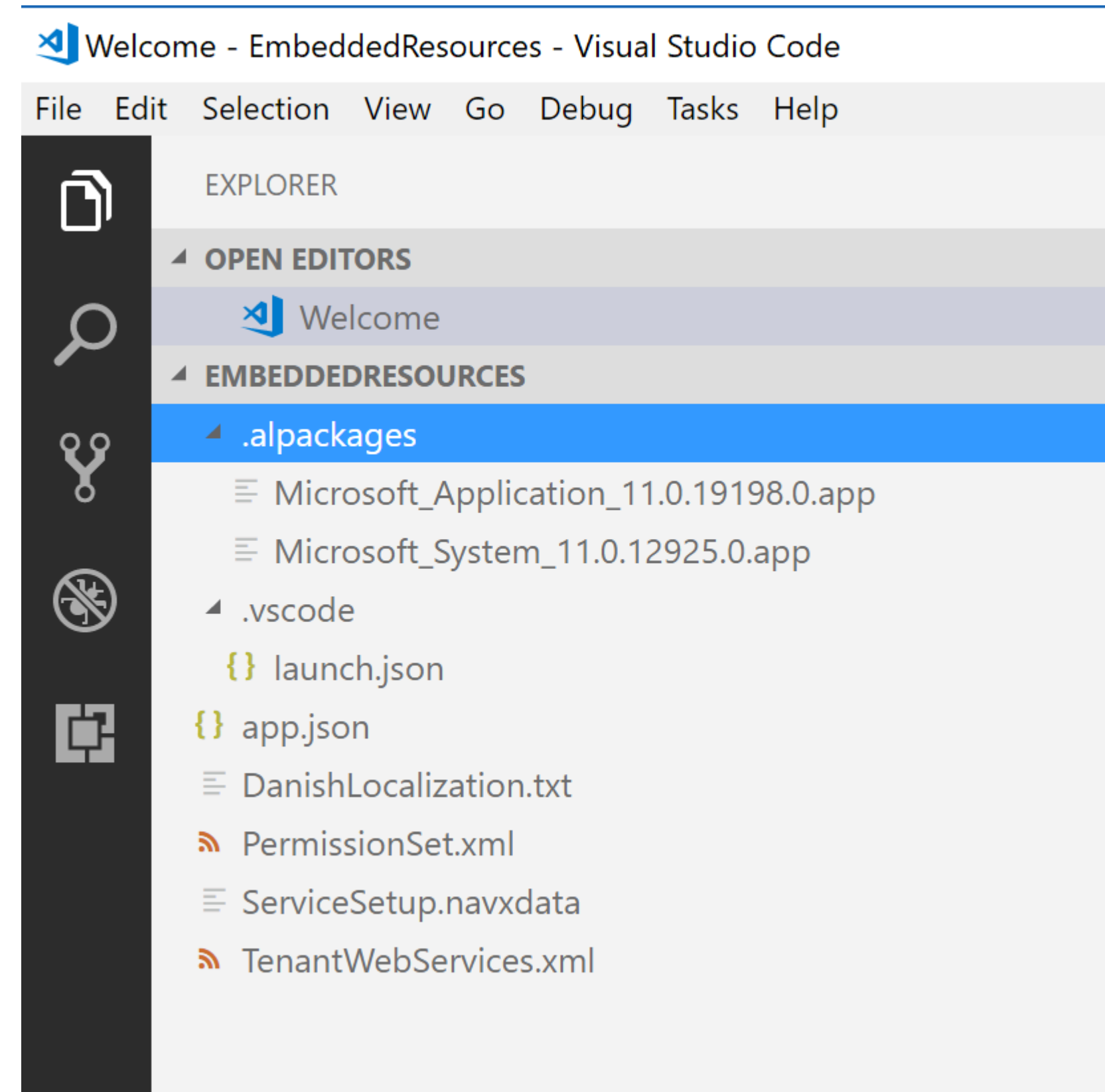
Export-NAVAppTableData

## Web service definitions

Export-NAVAppTenantWebService

## Custom report layouts

Export-NAVAppReportLayout



# New approach to translations

## New syntax for labels

```
field(51; CurrencyAmount; Decimal)
{
    Caption = 'Outstanding amount in %1', Comment = '%1 - currency', MaxLength = 40;
}

var
    0 references
    ErrorMessage : label 'Web service configuration missing', Locked = true;
```

## Generation of XLIFF files

```
<trans-unit id="Table 892477493 - Field 3208725453 - Property 2879900210" max-width="40" :
    <source>Outstanding amount in %1</source>
    <note from="Developer" annotates="general" priority="2">%1 - currency</note>
    <note from="Xliff Generator" annotates="general" priority="3">Table - Field</note>
</trans-unit>
```



# Demo – XLIFF based translations

# What about MenuSuites?

```
pageextension 50102 ExtendNavigationPane extends "Business Manager Role Center"
{
    actions
    {
        addlast(Embedding)
        {
            0 references
            action("Open my new page")
            {
                RunObject = page "Customer Bank Account List";
                ApplicationArea = All;
            }
        }
    }
}
```

# Install experience

## New Install codeunits and new triggers

```
codeunit 50103 InstallCodeunitForExtension
```

```
{
```

```
    Subtype = Install;
```

```
    trigger OnInstallAppPerCompany();
```

```
    var
```

```
        context : ExecutionContext;
```

```
        info : ModuleInfo;
```

```
    begin
```

```
        context := GetCurrentModuleExecutionContext();
```

```
        NavApp.GetCurrentModuleInfo(info);
```

```
        info.
```

```
    end;
```

```
}
```

AppVersion

DataVersion

Dependencies

Id

Name

Publisher

procedure AppVersion() : Versi  
on

# Demo – Installation & Navigation

# Profile Objects

```
profile SalesRepresentative
{
    Description = 'Sales Representative Profile';
    RoleCenter = "Sales & Relationship Mgr. RC";
    Customizations = SimplifiedCustomer;
}
```

1 reference

```
pagecustomization SimplifiedCustomer customizes "Customer Card"
{
    layout
    {
        modify("Balance (LCY)")
        {
            Visible = false;
        }

        moveafter(Address;"E-Mail")
    }

    //Variables, procedures and triggers are not allowed on Page Customizations
}
```



# Move from C/AL to AL

Open the source code

```
finsql.exe Command=ExportToNewSyntax
```

```
SubType = Upgrade;
```

```
txt2al.exe --source --target --rename --type --extensionStartId
```

```
var
```

```
    archiveRef : RecordRef;
```

```
begin
```

```
    if (NavApp.GetArchiveVersion() <> '') then begin
```

```
        // Compatible schema
```

```
        NavApp.RestoreArchiveData(50102);
```

```
        // Incompatible schema or partial upgrade
```

```
        NavApp.GetArchiveRecordRef(50102, archiveRef);
```

```
    end;
```

```
end;
```

```
}
```

# Modern Development for NAV 2018



## Customize the installation



## Specify parameters

Microsoft Dynamics NAV

Client

Modern Development Environment

Administration Tool

Server

SQL Server Database Components

Microsoft Office Outlook Add-In

Automated Data Capture System

Web Server Components

Help Server

ClickOnce Installer Tools

Development

Allowed Extension Target Level:

Internal

HttpClient AL Function Maximum Timeout v...

00:05:00

Enable Developer Service Endpoint:

☒

HttpClient AL Function Response Size:

15

Enable loading application symbol references at server start...

☒

Port:

7049

Enable SSL:

☐

Developer Services Port

\* 7049

Developer Services Enabled

\* Yes

Installs the Microsoft AL Language extension for \

# Demo – Symbol generation from C/SIDE

# Testability

Testability is still supported.

Test codeunits, Page testability,

Tests can be written in modules

Separating tests from production code.

Running the tests is still work in progress

We want to provide an integrated experience

# What is next



# Demo – Last one

# Not only a technical change

Important new member of our team – you, the community!

GitHub – be part of shaping the future of Modern Dev

GitHub – contribute samples and snippets

GitHub – suggest new events

C/AL Open Library – create .NET wrappers to help move into extensions V2

VS Code extensions around AL Language

# THANK YOU!!

<https://aka.ms/GetStartedWithApps>

<https://github.com/microsoft/AL/issues>

stanst@microsoft.com