COSMO CONSULT

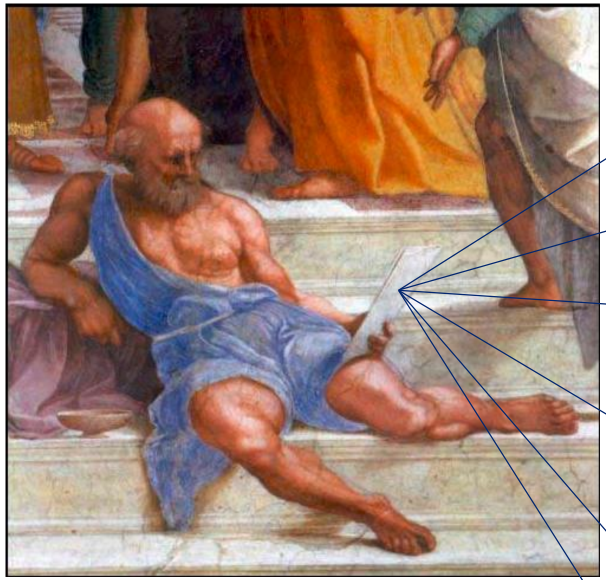
Business-Software für Menschen

Microsoft AL

Importants and Excercises

Core Statements on the subject of Programming AL

What the wise man says



Programming AL is almost like programming C/Side TXT files

Use MS Txt2AL Converter whenever you‘re helpless

What changes is the object structure: Fobs in Object Designer → AL (txt) Files in File System

It is crucial to setup a normalized file structure to
- become more efficient
- give „everyone“ the chance to „participate a development“ without the need of lengthy project individual introductions

It is lots more about getting familiar with guidelines and „doings“ than learning to code AL

if “actions are too cumbersome”, we will think about suitable tool-assistance

Programming in AL
IF you don’t know what to do

If you don‘t know, what to do - 2: Use Txt2AL Converter

If you don‘t know, what to do - 1: Press „Ctrl+Space“

If you don‘t know, what to do - 3: Press „t<context>“

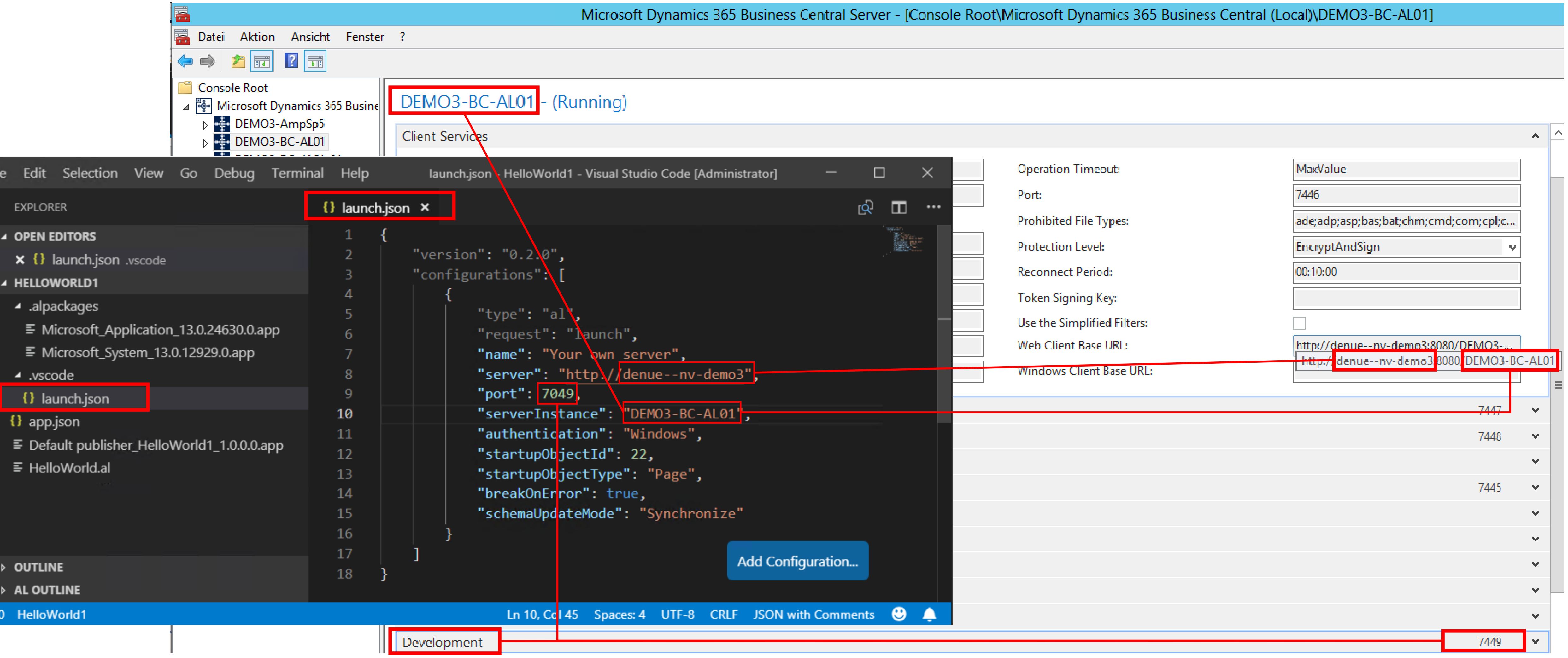
Literature / further Reading & Watching

- VS Code
  - https://code.visualstudio.com/docs/getstarted/introvideos
  - https://jpearson.blog/2019/01/31/vs-code-powershell-git-5-things/
  - https://code.visualstudio.com/docs/getstarted/tips-and-tricks: multi cursor editing …
- …

Boundary Conditions / Setups: HowTo handle a Project

Specifying BC Target → launch.json

- more detailed


Microsoft Dynamics 365 Business Central Server - [Console Root\Microsoft Dynamics 365 Business Central (Local)\DEMO3-BC-AL01]
Datei Aktion Ansicht Fenster ?
Console Root
Microsoft Dynamics 365 Busine
DEMO3-AmpSp5
DEMO3-BC-AL01
DEMO3-BC-AL01 - (Running)
Client Services
Development
e Edit Selection View Go Debug Terminal Help
launch.json - HelloWorld1 - Visual Studio Code [Administrator]
EXPLORER
OPEN EDITORS
launch.json .vscode
HELLOWORLD1
.alpackages
Microsoft_Application_13.0.24630.0.app
Microsoft_System_13.0.12929.0.app
.vscode
launch.json
app.json
Default publisher_HelloWorld1_1.0.0.0.app
HelloWorld.al
OUTLINE
AL OUTLINE
Add Configuration...
HelloWorld1
Ln 10, Col 45 Spaces: 4 UTF-8 CRLF JSON with Comments

```json
{
    "version": "0.2.0",
    "configurations": [
        {
            "type": "al",
            "request": "launch",
            "name": "Your own server",
            "server": "http://denue--nv-demo3",
            "port": 7049,
            "serverInstance": "DEMO3-BC-AL01",
            "authentication": "Windows",
            "startupObjectId": 22,
            "startupObjectType": "Page",
            "breakOnError": true,
            "schemaUpdateMode": "Synchronize"
        }
    ]
}
```

Operation Timeout:	MaxValue
Port:	7446
Prohibited File Types:	ade;adp;asp;bas;bat;chm;cmd;com;cpl;c...
Protection Level:	EncryptAndSign
Reconnect Period:	00:10:00
Token Signing Key:	
Use the Simplified Filters:	☐
Web Client Base URL:	http://denue--nv-demo3:8080/DEMO3-...
Windows Client Base URL:	http://denue--nv-demo3:8080/DEMO3-BC-AL01

7447
7448
7445
Development 7449

APP Organization

- „Namespace / Prefixing“
- Philosophy
- Enveloping Folder Structure
- Naming Conventions „Agent Folders / Files“
- Examples
- Documentation: logs and tags

Programming in AL

APP Organization - Unique Namespace: Prefixing

- Forced by the possibility of changing environment: suitable Prefixing to create uniqueness
  - (separated by a „Space“ from the object‘s name)
- Use the Prefix ***CCO*** (Cosmo Consult Operations) for the following:
  - Objects
    - Name
  - Object Codeunits
    - Name
  - Object Extensions
    - Name
    - Global Procedures
    - Fields (Table)
    - Keys / KeyNames
    - Controls (Page)
    - Actions (Page)
    - Values (Enum)
- Prefix „CCOT“ for Test Objects

Programming in AL

APP Organization - Philosophy

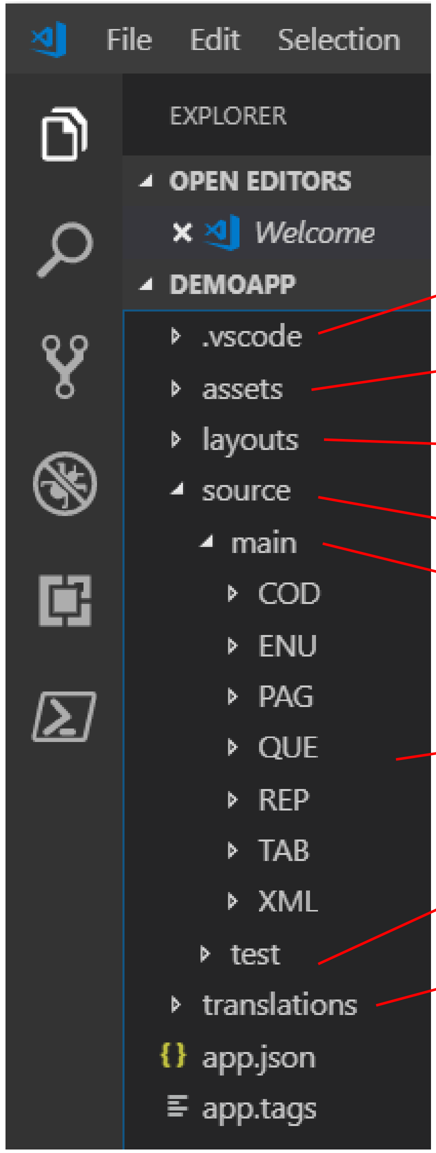
Philosophy

- Group together what belongs together
- Forget about IDs as sort criterion
- Create an understandable and easily enforceable rule on where code should be stored and how files should be named

Table x	or	Extension of Table x	Object Codeunit of Table x		*xliff ?*	*log of Table x*	„Agent Folder“	Table x
Page y	or	Extension of Page y	Object Codeunit of Page y		*xliff ?*	*log of Page y*	„Agent Folder“	Page y
Report z	or	*(Report Extension)*	Object CU of Report z	Layout	*xliff ?*	*log of Report z*	„Agent Folder“	Report z
Codeunit u			Object CU of Codeunit u		*xliff ?*	*log of Codeunit u*	„Agent Folder“	Codeunit u
XML Port v	or	*(?XML Port Extension?)*	Object CU of XML Port v		*xliff ?*	*log of XML Port v*	„Agent Folder“	XML Port v
Query w	or	*(?Query Extension?)*	Object CU of Query w		*xliff ?*	*log of Query w*	„Agent Folder“	Query w
Enum q	or	Enum Extension			*xliff ?*	*log of Enum q*	„Agent Folder“	Enum q
Control Add-In r			Object CU of *Ctrl. Add-In r*		*xliff ?*	*log of Ctrl Add-In r*	„Agent Folder“	Ctrl Add-In r

Programming in AL

APP Organization - Enveloping Folder Structure



contains „launch.json“

contains assets like a logo …

contains report layouts to be published (c.f. below.)

organizes the actual program files, distinguishes …

„main“: distinguishes object types via subfolder structure <type>

<type> folders: contains **agent object folders** (location of program files, c.f. below)

„test“: contains test codeunits per GAP

contains translation file of the app, if existing (c.f. below.)

app.json file

app.tags (file): contains collection of „GAP“-tags representing al GAPs implemented so far, c.f. below

Programming in AL

APP Organization - Naming Conventions Agent Folders / Files

Folder Names	<Type>	.	<Agent Object Name, abbreviated>	*[.<Origin>]*		
File Names	<Type>	.	<Agent Object Name, abbreviated>	*[.<Origin>]*	*[.<Subtype>]*	.al
Object Names	CCO	*[Agent Type]*	<Agent Object Name>	*[_+]*		
Log File	<Folder Name>			.log	(for docu-purposes, c.f. below)	

Type	denotes the Object Type of the source Object could be TAB, PAG, REP, COD, …
„abbreviated“	means without spaces …, as suggested as Variable Name by the Object Designer „Purchase Header → PurchaseHeader“
Origin	[„“,“cc“]; „“ if Object is app-external „cc“ if Object is app object
Subtype	[„“, „ext“, „code“]; „ext“ for extension objects; „code“ for Object Codeunits
„Agent Type“	Used for Object CUs only: indicates the type of the related object [T, P, R, X, C, …]
Appendix „+“	indicates in object names that the object extends a standard object.

Programming in AL APP Organization - Naming Conventions: Examples)

Object	Name of Object File	Name of Object	Object Folder
Tableextension *Example 1: Item* *Example 2: Purchase Header*	TAB.<Name of extended Tab, …>.ext.al *TAB.Item.ext.al* *TAB.PurchaseHeader.ext.al*	CCO_<Name of extended Tab>_+ *"CCO Item +"* *"CCO Purchase Header +"*	TAB.<Name Table, …> *TAB.Item* *TAB.PurchaseHeader*
Pageextension *Example: Item Card*	PAG.<Name of extended Page, …>.ext.al *PAG.ItemCard.ext.al*	CCO_<Name of extended Page>_+ *"CCO Item Card +"*	PAG.<Name of Page, …> *PAG.ItemCard*
Table (== new Table) *Example: My App Table*	TAB.<Name of new Table, …>.al *TAB.MyAppTable.cc.al*	CCO_<Name of new Table> *"CCO My App Table"*	TAB.<Name Table, …>.cc *TAB.MyAppTable.cc*
Page (== new Page) *Example: My App Page*	PAG.<Name of new Page, …>.al *PAG.MyAppPage.cc.al*	CCO_<Name of new Page> *"CCO My App Page"*	PAG.<Name of Page, …>.cc *PAG.MyAppPage.cc*
Report (== new Report) *Example: My App Report*	REP.<Name of new Report, …>.al *REP.MyAppReport.cc.al*	CCO_<Name of new Report> *"CCO My App Report"*	REP.<Name of Report, …>.cc *REP.MyAppReport.cc*
ObjectCU of APP-external Tabelle *Example 1: Item* *Example 2: Purchase Header*	TAB.<Name of Table, …>.code.al *TAB.Item.code.al* *TAB.PurchaseHeader.code.al*	CCO_T_<Name of Table>_+ *"CCO T Item +"* *"CCO T Purchase Header +"*	TAB.<Name of Table, …> *TAB.Item* *TAB.PurchaseHeader*
ObjectCU of APP-internal Tabelle *Example: My App Table*	TAB.<Name of new Table, …>.cc.code.al *TAB.MyAppTable.cc.code.al*	CCO_T_<Name of Table> *"CCO T My App Table"*	TAB.<Name of Table, …>.cc *TAB.MyAppTable.cc*
ObjectCU of APP-external Page *Example: Item Card*	PAG.<Name of Page>.code.al *PAG.ItemCard.code.al*	CCO_P_<Name of Page>_+ *"CCO P Item Card +"*	PAG.<Name of Page, …> *PAG.ItemCard*
ObjectCU of APP-internal Page *Example: My App Page*	PAG.<Name of Page>.code.al *PAG.MyAppPage.cc.code.al*	CCO_P_<Name of Page> *"CCO P My App Page"*	PAG.<Name of Page, …>.cc *PAG.MyAppPage.cc*
ObjectCU of APP-external Report *Example: Order Confirmation*	REP.<Name of Report>.code.al *REP.OrderConfirmation.code.al*	CCO_R_<Name of Report> *"CCO R Order Confirmation +"*	REP.<Name of Report, …> *REP.OrderConfirmation*
ObjectCU of APP-internal Report *Example: My App Report*	REP.<Name of Report>.code.al *REP.MyAppReport.cc.code.al*	CCO_R_<Name of Report> *"CCO R My App Report +"*	REP.<Name of Report, …>.cc *PAG.MyAppReport.cc*
…	…	…	…

Programming in AL

APP Organization - Documentation: log-files and app-tags

- Documentation via Log Files
  - One Log per Folder
  - Log file contains well known docu headers (per GAP)

```
<GAP-Tag>_<Date>_<Developer ID>: <Title of GAP>
        Information on changes …
```

A „GAP-Tag“ consists of the GAP Identifier and a two digit suffix counting the generations of implementations of the gap.

If a GAP Implementation tsarts, the suffix is „.01“, if is reopened after a first completion, it gets suffix „.02“ and so on.

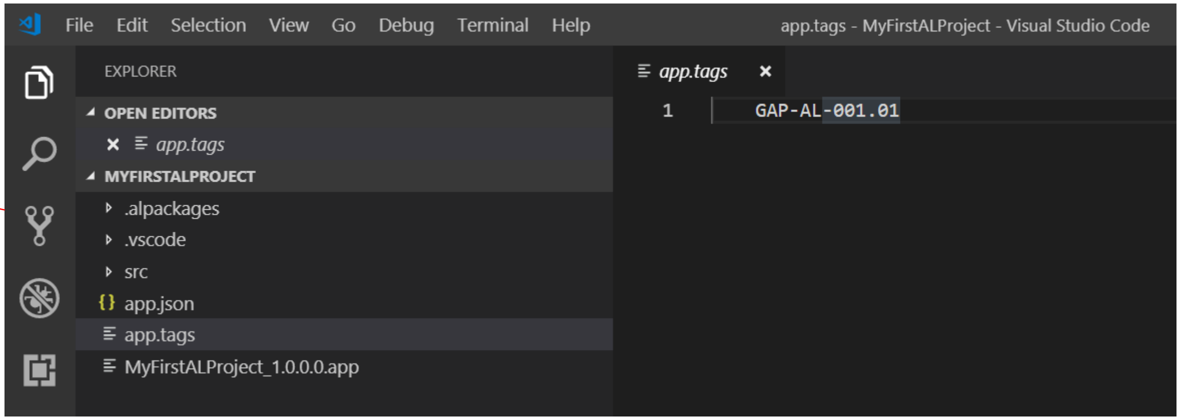
```
GAP-FI-001.01 01.01.2019 DENUE.MPRO: Default Description
        Created
                New field 55000 „Description 3“
        Modified
                …..
```

  - Remark: perhaps, we will be able to skip documentation in the future, using Azure DevOps …

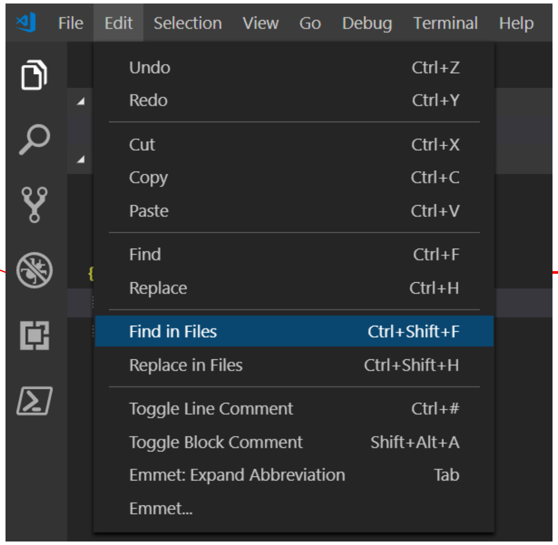
Programming in AL

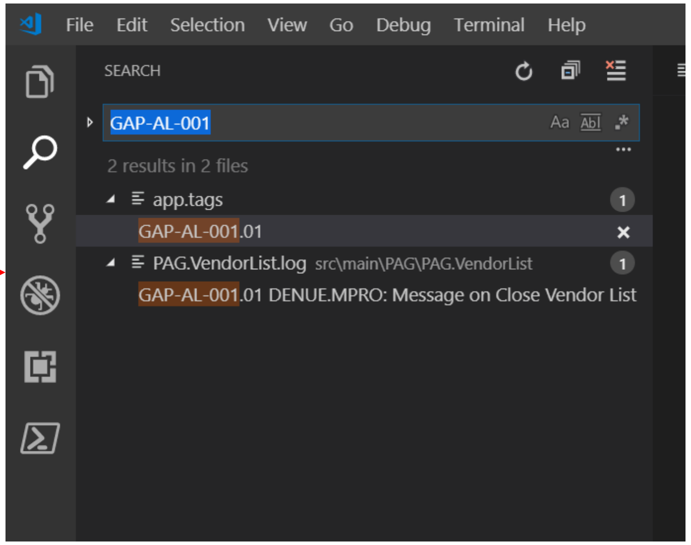
APP Organization - Documentation: log-files and app-tags

- The „app.tags“ file
  - Loctaed in the root folder of the app
  - Contains al list of the GAPs implemented up to the current project state
  - [GAP-ID].[xx]: the „xx“ denotes the „development phase“ of the GAP
  - Enables „app-wide“ search for GAP Implementations


File Edit Selection View Go Debug Terminal Help
app.tags - MyFirstALProject - Visual Studio Code
EXPLORER
OPEN EDITORS
app.tags
MYFIRSTALPROJECT
.alpackages
.vscode
src
app.json
app.tags
MyFirstALProject_1.0.0.0.app
app.tags

```
GAP-AL-001.01
```


File Edit Selection View Go Debug Terminal Help
Undo Ctrl+Z
Redo Ctrl+Y
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find Ctrl+F
Replace Ctrl+H
Find in Files Ctrl+Shift+F
Replace in Files Ctrl+Shift+H
Toggle Line Comment Ctrl+#
Toggle Block Comment Shift+Alt+A
Emmet: Expand Abbreviation Tab
Emmet...


File Edit Selection View Go Debug Terminal Help
SEARCH
GAP-AL-001
2 results in 2 files
app.tags 1
PAG.VendorList.log src\main\PAG\PAG.VendorList 1

```
GAP-AL-001.01
GAP-AL-001.01 DENUE.MPRO: Message on Close Vendor List
```

HandsOn

HandsOn – Exercise 1

Hello World
our very first page extension?

HandsOn – Exercise 1 | Hello World: our very first page extension

HandsOn

Understand the Example Extension „Hello World“

Configure your Environment / Run the Example

Load Template

Create „Hello World“
- Open VS Code
- Type F1 or Ctrl+Shift+P to open AL Command Line
- Type a fragment of „AL: Go!“
- Choose „AL: Go!“

Oder: „Alt+A, Alt+L“

- Enter (AL) Project Path
- Choose Environment (Cloud / OnPremise)

Configure Environment

Characterize your environment via proper values in the launch.json file

Define the APP

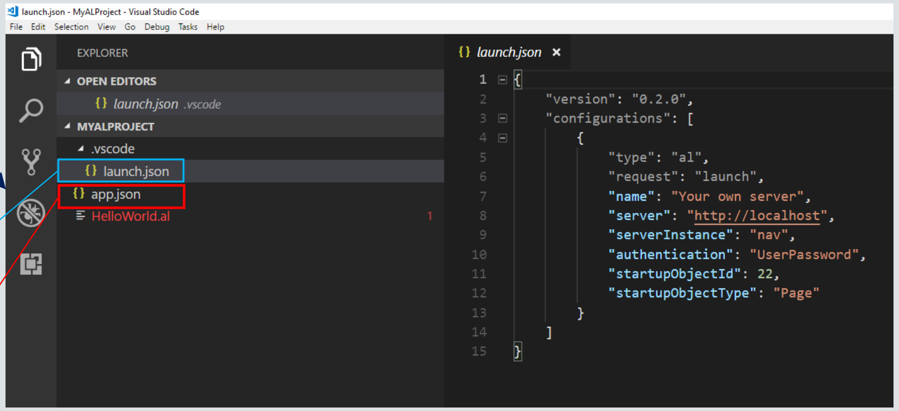
Characterize your APP via proper values in the app.json file

Introduce the Target DB‘s object structure to your APP

Download Symbols
- Type F1 or Ctrl+Shift+P to open AL Command Line
- Type a fragment of “Download Symbols”
- Choose “Download Symbols”

Compile / Publish / Install / Launch the APP

Hit „F5“ in VS Code ….



```
{
    "version": "0.2.0",
    "configurations": [
        {
            "type": "al",
            "request": "launch",
            "name": "Your own server",
            "server": "http://localhost",
            "serverInstance": "nav",
            "authentication": "UserPassword",
            "startupObjectId": 22,
            "startupObjectType": "Page"
        }
    ]
}
```

If things don‘t work as expected:
Restart VS Code
(e.g., if you‘ve changed the „Web Client Base URL“

HandsOn

HandsOn – Exercise 2

Deinstalling the app (via Client)

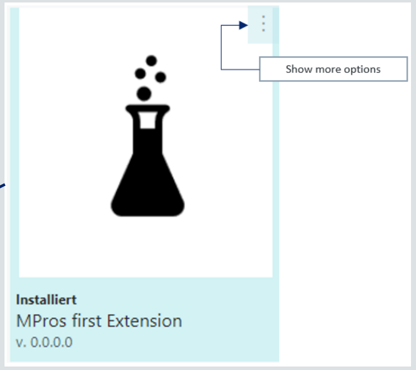
HandsOn – Exercise 2 | Deinstalling the app (via Client)

HandsOn

Open Client

Search „Extension Management“

Press “Show more options” and choose “Uninstall”



Extension is now „Uninstalled“ but not „Unpublished“

Press “Show more options” and choose “Unpublish”

Extension is now „no longer available on the system“

Learn - Why this 2 step procedure?
To be able to have an extension available on dedicated tenants only

→ publish into a data base
→ Install into a tenant

!! data remains !!

HandsOn

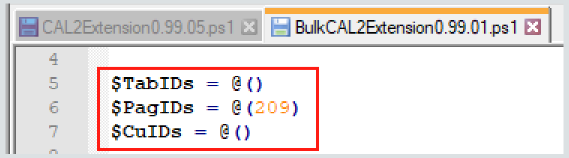
HandsOn – Exercise 3

Configuring the Txt2AL Converter

HandsOn – Exercise 3 | Configuring the Txt2AL Converter

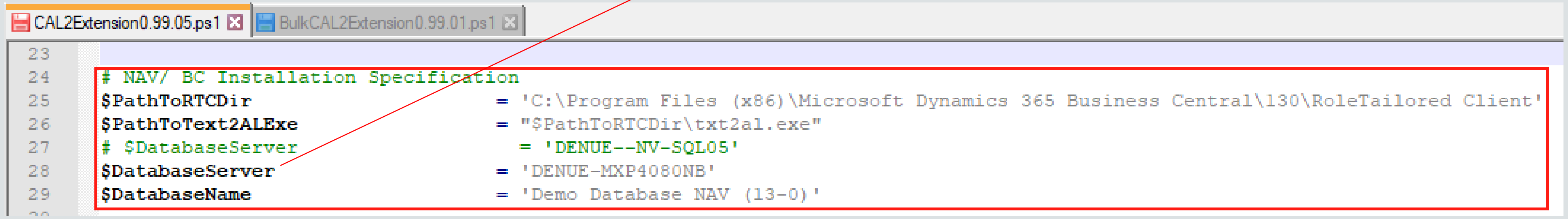
HandsOn

- Txt2AL Converter - HandsOn: Convert page „Units of Measure (209)“
  - Configure Converter



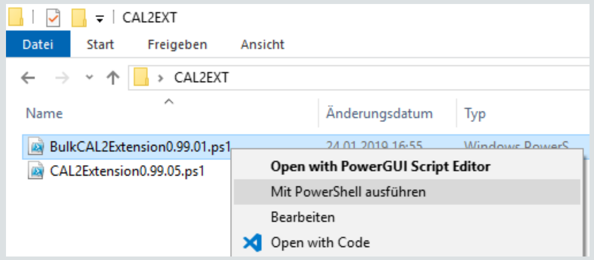
```
$TabIDs = @()
$PagIDs = @(209)
$CuIDs = @()
```

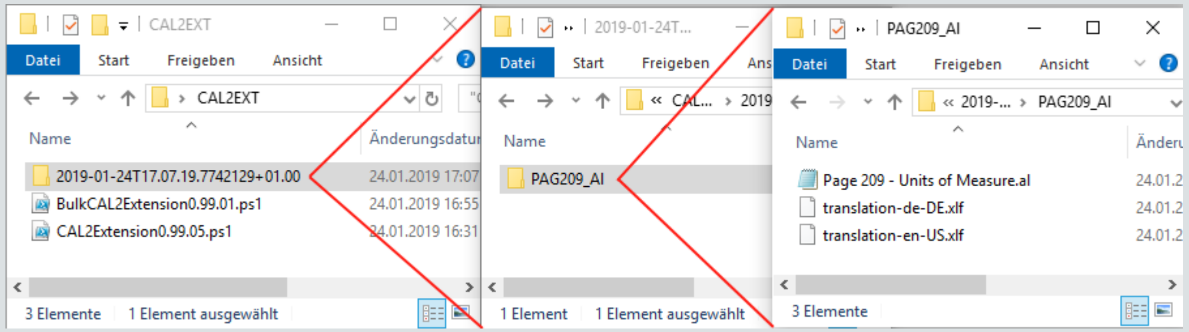
SQL-Server



```
# NAV/ BC Installation Specification
$PathToRTCDir                        = 'C:\Program Files (x86)\Microsoft Dynamics 365 Business Central\130\RoleTailored Client'
$PathToText2ALExe                    = "$PathToRTCDir\txt2al.exe"
# $DatabaseServer                      = 'DENUE--NV-SQL05'
$DatabaseServer                      = 'DENUE-MXP4080NB'
$DatabaseName                        = 'Demo Database NAV (13-0)'
```

  - Convert page „Units of Measure (209)“



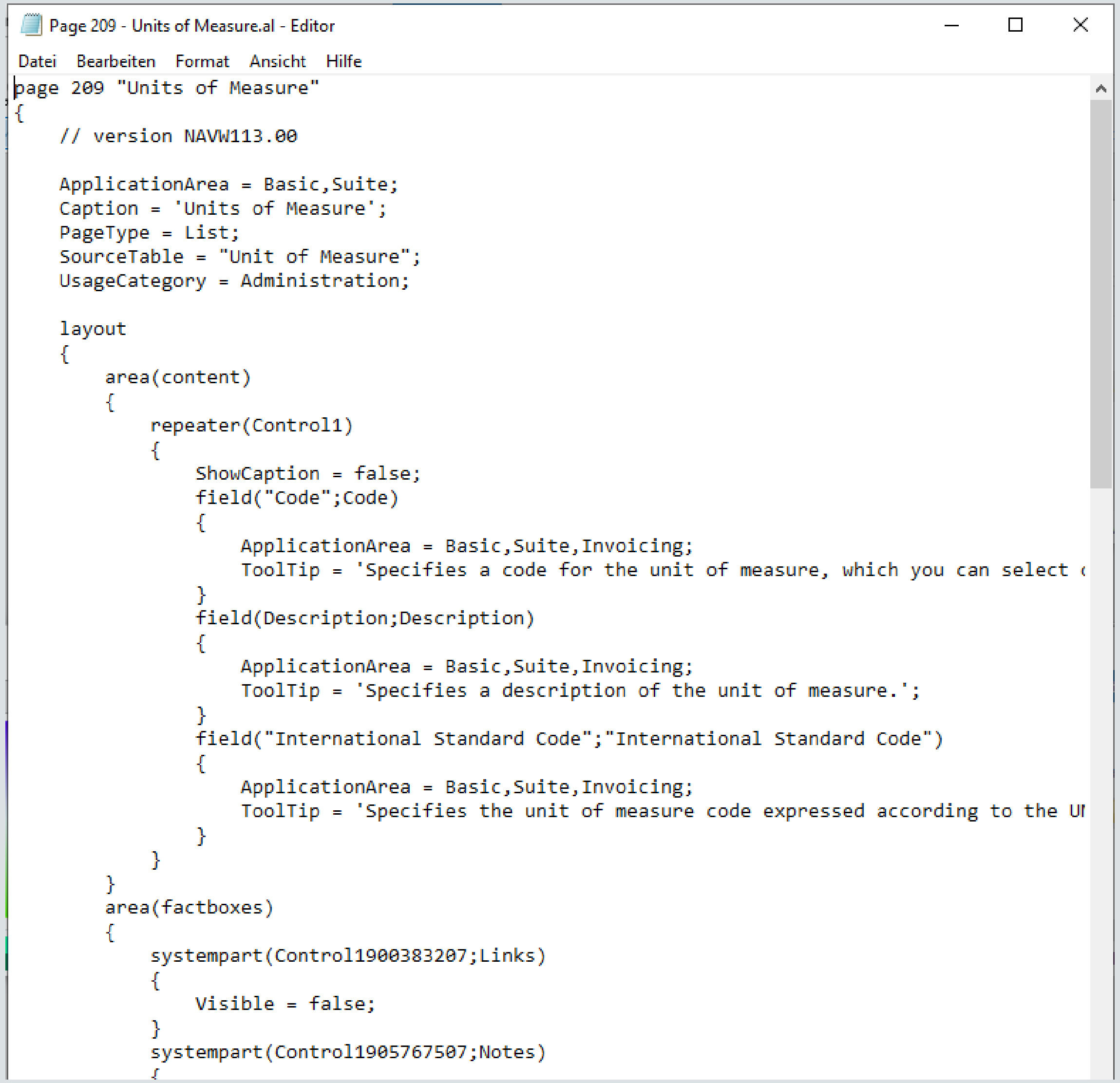


HandsOn – Exercise 3 Continued | Configuring the Txt2AL Converter

HandsOn

- Result

Page 209 – Units of Measure.al


Page 209 - Units of Measure.al - Editor
Datei Bearbeiten Format Ansicht Hilfe

```
page 209 "Units of Measure"
{
    // version NAVW113.00

    ApplicationArea = Basic,Suite;
    Caption = 'Units of Measure';
    PageType = List;
    SourceTable = "Unit of Measure";
    UsageCategory = Administration;

    layout
    {
        area(content)
        {
            repeater(Control1)
            {
                ShowCaption = false;
                field("Code";Code)
                {
                    ApplicationArea = Basic,Suite,Invoicing;
                    ToolTip = 'Specifies a code for the unit of measure, which you can select
                }
                field(Description;Description)
                {
                    ApplicationArea = Basic,Suite,Invoicing;
                    ToolTip = 'Specifies a description of the unit of measure.';
                }
                field("International Standard Code";"International Standard Code")
                {
                    ApplicationArea = Basic,Suite,Invoicing;
                    ToolTip = 'Specifies the unit of measure code expressed according to the UN
                }
            }
        }
        area(factboxes)
        {
            systempart(Control1900383207;Links)
            {
                Visible = false;
            }
            systempart(Control1905767507;Notes)
            {
```

HandsOn

HandsOn – Exercise 4

intelliSense

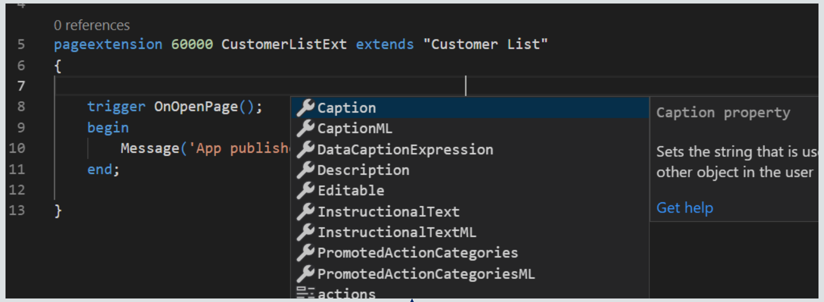
HandsOn – Exercise 4 | intelliSense

HandsOn

Open VS Code on „HelloWorld“ Project

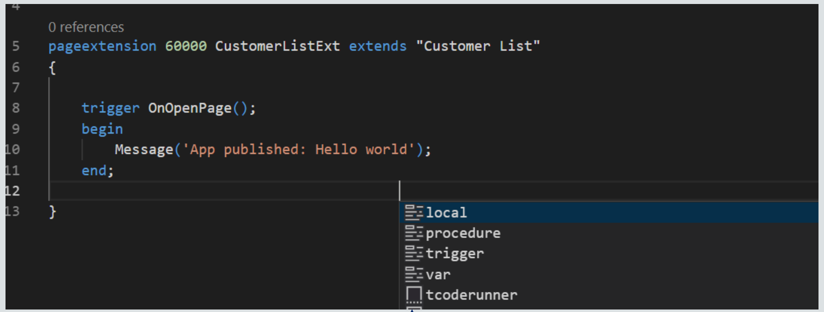
Open file „HelloWorld.al“

Place the cursor above “trigger OnOpenPage()” and press “Ctrl+Space” to activate Intellisense



```
0 references
pageextension 60000 CustomerListExt extends "Customer List"
{

    trigger OnOpenPage();
    begin
        Message('App publish
    end;

}
```

Place the cursor below “trigger OnOpenPage()” block and press “Ctrl+Space” again



```
0 references
pageextension 60000 CustomerListExt extends "Customer List"
{

    trigger OnOpenPage();
    begin
        Message('App published: Hello world');
    end;

}
```

Learn that
1) the suggestions depend on the actual context of your code – it’s “Intelli”
2) Elements of an object have to be placed in a certain order …. .

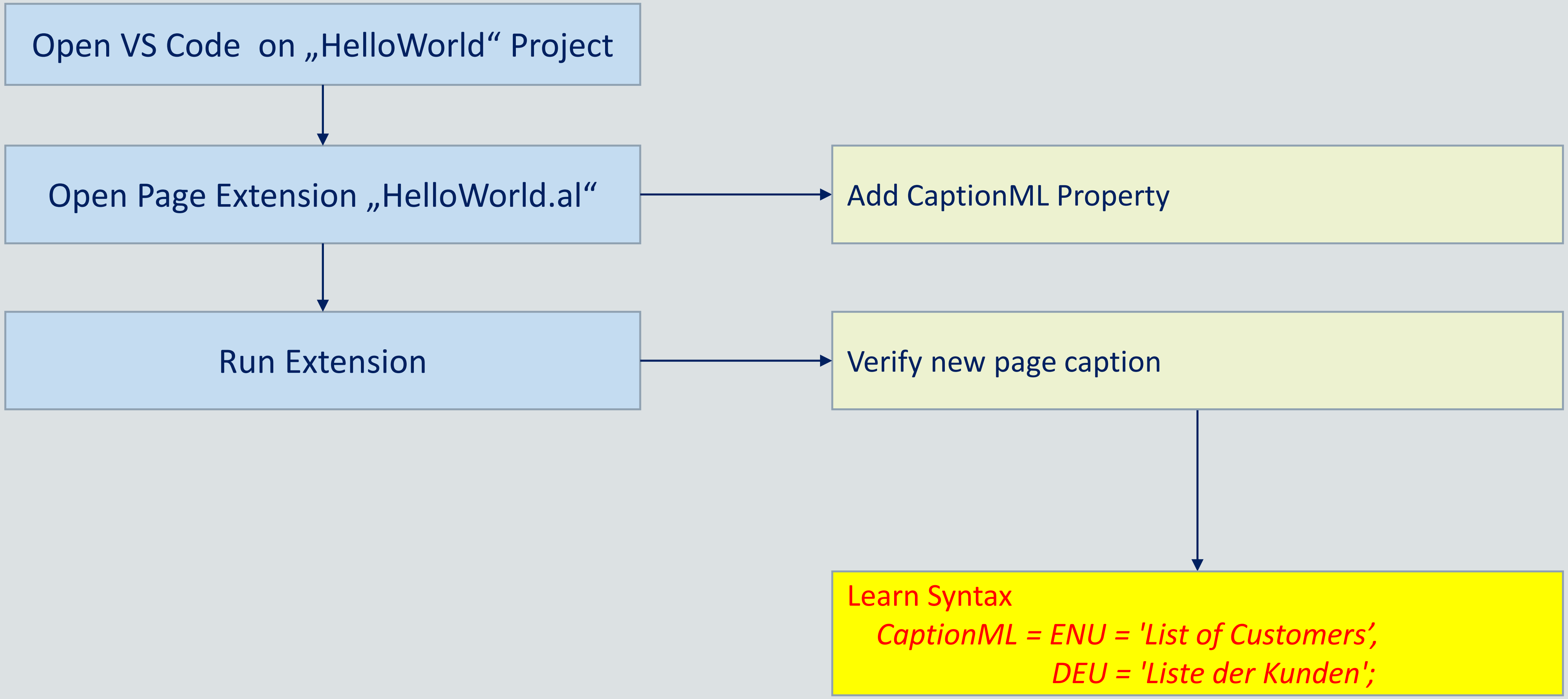
HandsOn

HandsOn – Exercise 5

AL Syntax: Caption ML

HandsOn – Exercise 5 | CaptionML

HandsOn


Open VS Code on „HelloWorld“ Project
Open Page Extension „HelloWorld.al“
Run Extension
Add CaptionML Property
Verify new page caption

Learn Syntax

```
CaptionML = ENU = 'List of Customers',
            DEU = 'Liste der Kunden';
```

HandsOn

HandsOn – Exercise 6

AL file structure

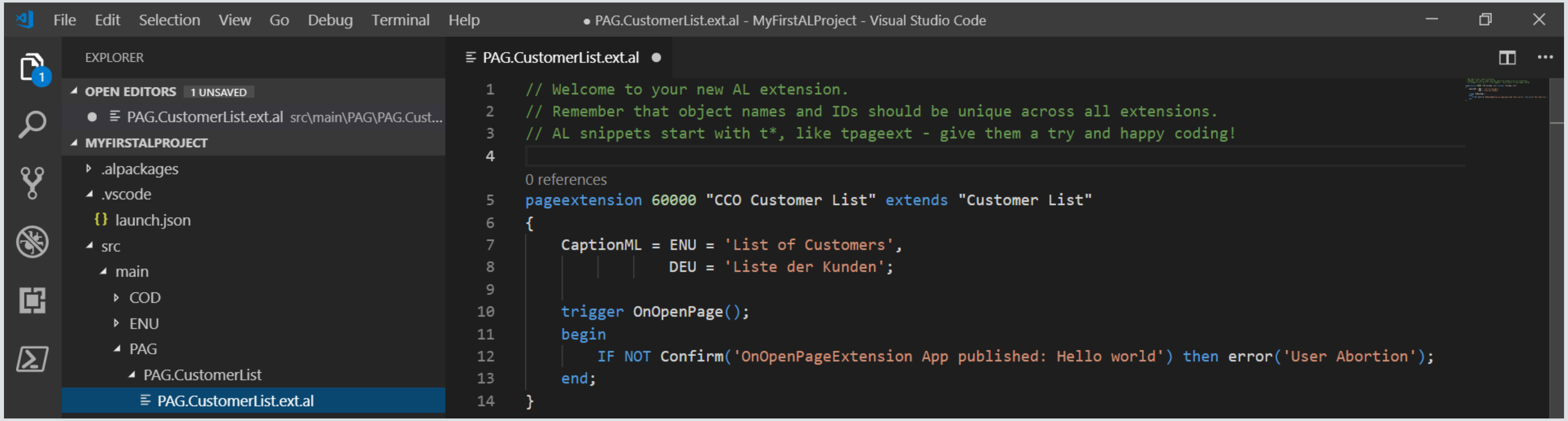
HandsOn – Exercise 6 | AL file structure

HandsOn

AL Project: Setting up the proper File / Folder structure

Clean up your project →

- Open „Hello World“ Project
- ↓ Setup a proper file / folder Structure
- ↓ Eventually rename your object files
- ↓ Eventually rename your objects



```
// Welcome to your new AL extension.
// Remember that object names and IDs should be unique across all extensions.
// AL snippets start with t*, like tpageext - give them a try and happy coding!

0 references
pageextension 60000 "CCO Customer List" extends "Customer List"
{
    CaptionML = ENU = 'List of Customers',
                DEU = 'Liste der Kunden';

    trigger OnOpenPage();
    begin
        IF NOT Confirm('OnOpenPageExtension App published: Hello world') then error('User Abortion');
    end;
}
```

HandsOn

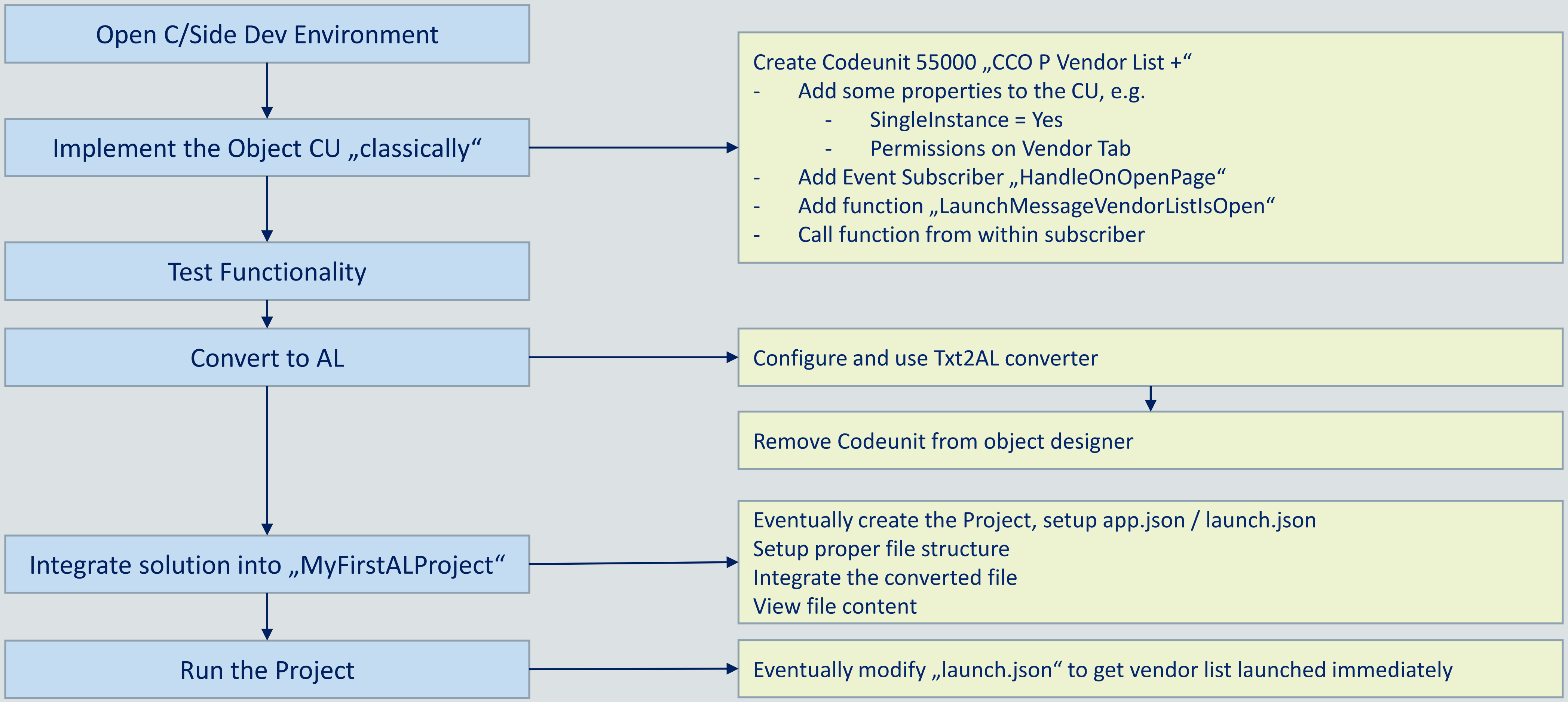
HandsOn – Exercise 7

Codeunits and the Txt2AL Converter

HandsOn – Exercise 7 | Codeunits and the Txt2AL Converter

HandsOn

Excercise CU programming via Txt2AL converter: Message on open Vendor List



Open C/Side Dev Environment

Implement the Object CU „classically“

Create Codeunit 55000 „CCO P Vendor List +“
- Add some properties to the CU, e.g.
  - SingleInstance = Yes
  - Permissions on Vendor Tab
- Add Event Subscriber „HandleOnOpenPage“
- Add function „LaunchMessageVendorListIsOpen“
- Call function from within subscriber

Test Functionality

Convert to AL

Configure and use Txt2AL converter

Remove Codeunit from object designer

Integrate solution into „MyFirstALProject“

Eventually create the Project, setup app.json / launch.json
Setup proper file structure
Integrate the converted file
View file content

Run the Project

Eventually modify „launch.json“ to get vendor list launched immediately

HandsOn

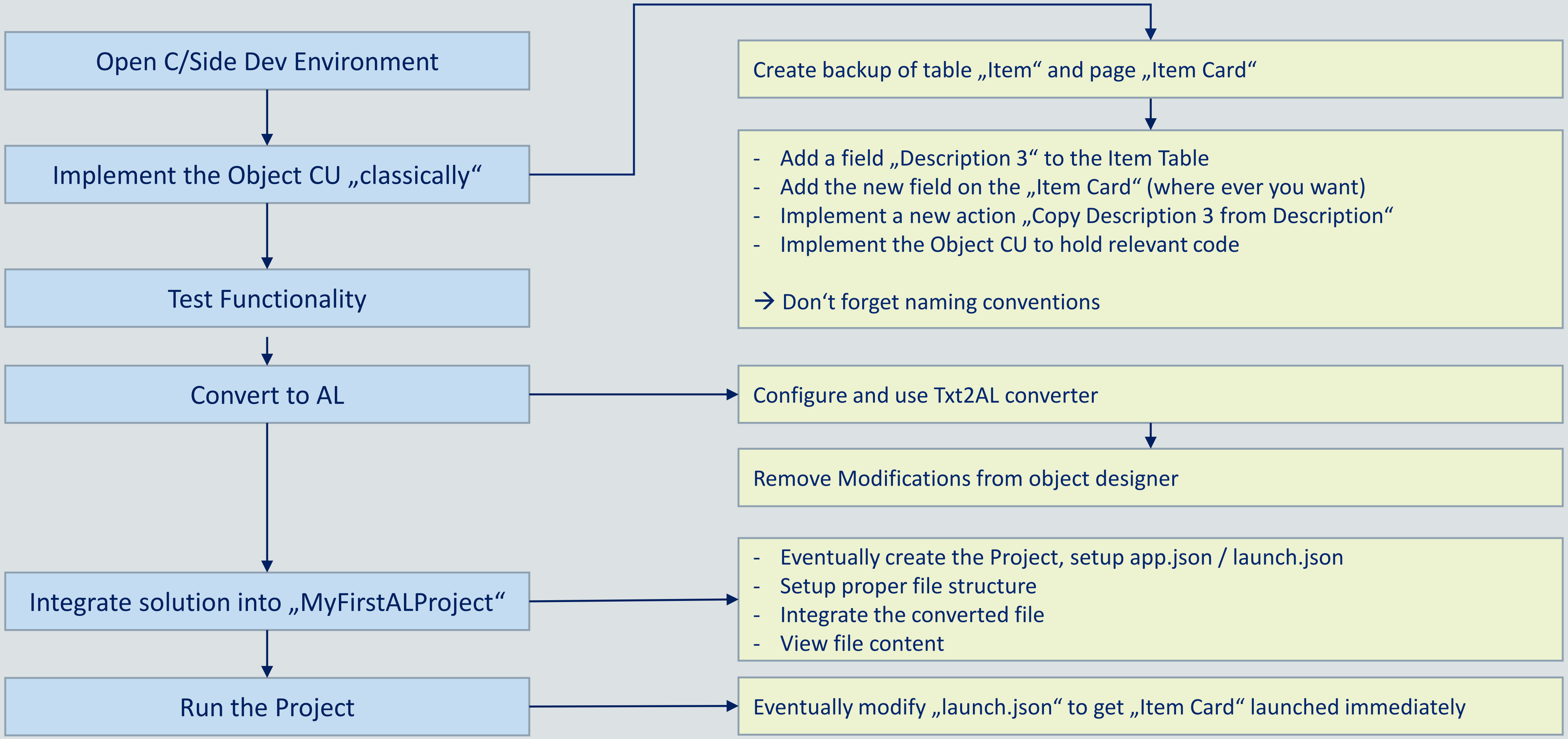
HandsOn – Exercise 7.5

Extension Programming and the Txt2AL Converter

HandsOn – Exercise 7.5 | Extension Programming and the Txt2AL Converter

HandsOn

Extension programming via Txt2AL converter – GAP-AL-000: „Description 3“ on „Item Card“



Open C/Side Dev Environment

Implement the Object CU „classically“

Test Functionality

Convert to AL

Integrate solution into „MyFirstALProject“

Run the Project

Create backup of table „Item“ and page „Item Card“

- Add a field „Description 3“ to the Item Table
- Add the new field on the „Item Card“ (where ever you want)
- Implement a new action „Copy Description 3 from Description“
- Implement the Object CU to hold relevant code

→ Don‘t forget naming conventions

Configure and use Txt2AL converter

Remove Modifications from object designer

- Eventually create the Project, setup app.json / launch.json
- Setup proper file structure
- Integrate the converted file
- View file content

Eventually modify „launch.json“ to get „Item Card“ launched immediately

HandsOn

HandsOn – Exercise 8

Procedures and Snippets

HandsOn – Exercise 8 | Procedures and Snippets

HandsOn

Implement „Message Vendor List is closed“ using Snippets

Open (My first) AL Project

Find and Open Object CU of Vendor List

Implement new Functionality

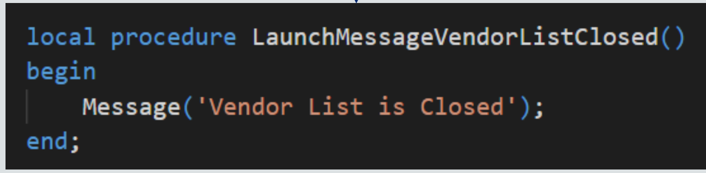
Run and Test the Project

Place cursor in new line (beyond existing Event Subscriber Procedure)

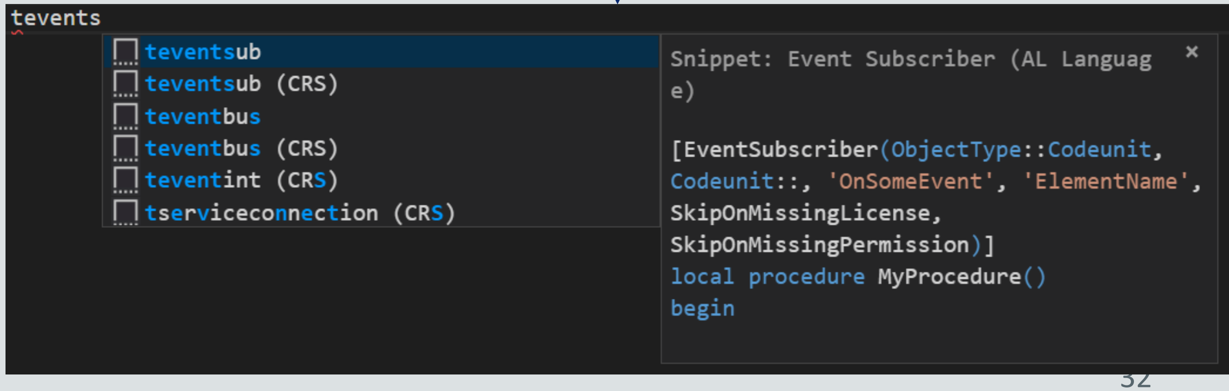
Type (leading) fragment of „teventsubscriber“ to start intelliSense

Implement the Desired Function „LaunchMessageVendorListClosed“

Choose the right snippet and implement the Event Subscriber



```
local procedure LaunchMessageVendorListClosed()
begin
    Message('Vendor List is Closed');
end;
```



```
tevents
```

```
teventsub
teventsub (CRS)
teventbus
teventbus (CRS)
teventint (CRS)
tserviceconnection (CRS)
```

```
Snippet: Event Subscriber (AL Languag
e)

[EventSubscriber(ObjectType::Codeunit,
Codeunit::, 'OnSomeEvent', 'ElementName',
SkipOnMissingLicense,
SkipOnMissingPermission)]
local procedure MyProcedure()
begin
```

HandsOn

HandsOn – Exercise 9

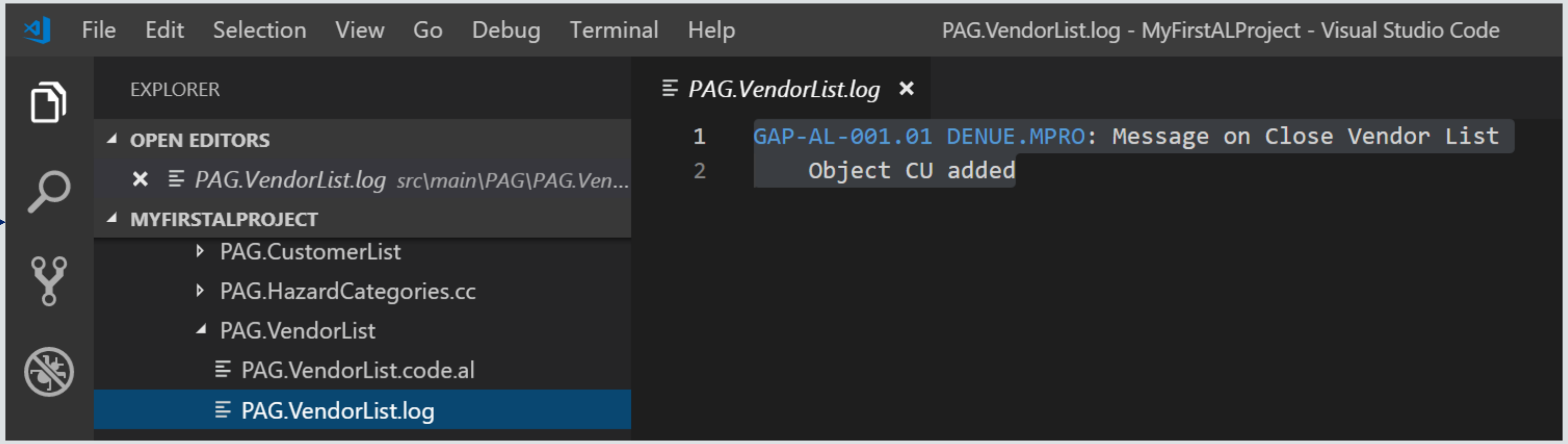
Documentation: “logs and tags”

HandsOn – Exercise 9 | Documentation „logs and tags“

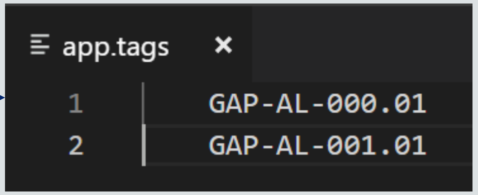
HandsOn

Finish „GAP-AL-000“: „Description 3 on Item Card“ and „GAP-AL-001: Message on Close Vendor List “

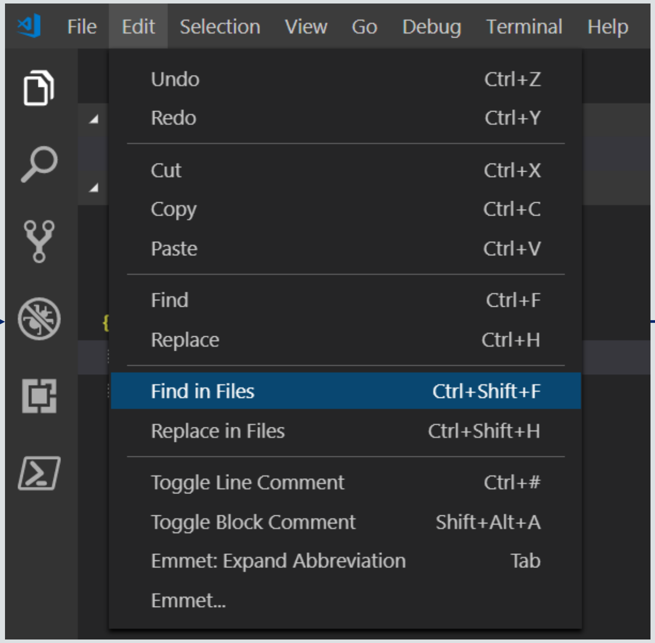
- Open (My first) AL Project
- Create Documentation (log-file(s))
- Log your GAPs within the „app.tags“-file
- Close open Editors and search for the GAP
- Run and Test the Project

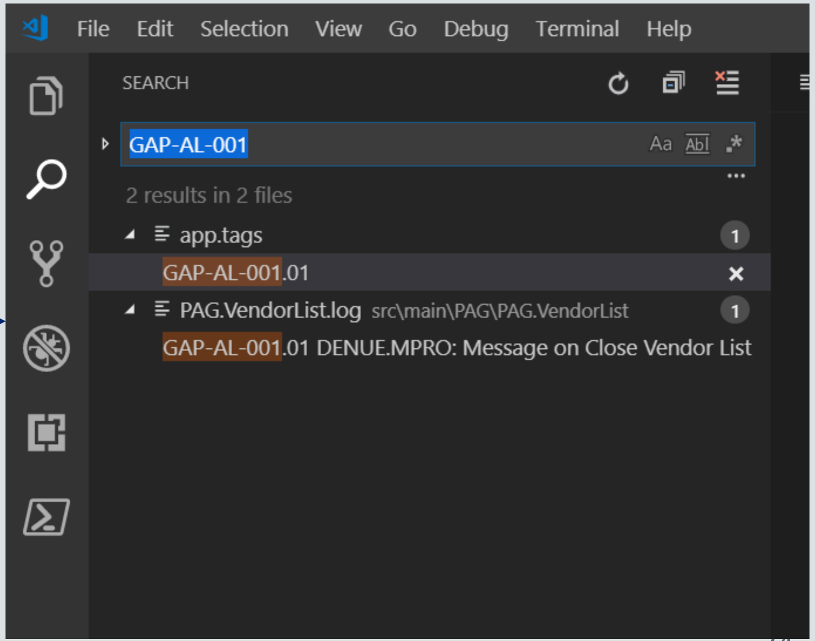


```
GAP-AL-001.01 DENUE.MPRO: Message on Close Vendor List
    Object CU added
```



```
GAP-AL-000.01
GAP-AL-001.01
```


File
Edit
Selection
View
Go
Debug
Terminal
Help
Undo Ctrl+Z
Redo Ctrl+Y
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find Ctrl+F
Replace Ctrl+H
Find in Files Ctrl+Shift+F
Replace in Files Ctrl+Shift+H
Toggle Line Comment Ctrl+#
Toggle Block Comment Shift+Alt+A
Emmet: Expand Abbreviation Tab
Emmet...



```
GAP-AL-001.01
GAP-AL-001.01 DENUE.MPRO: Message on Close Vendor List
```

HandsOn

HandsOn – Exercise 9.5

Pages 1, Structure

HandsOn – Exercise 9.5 | Pages

HandsOn

Convert a Page of your choice from C/Side to AL and analyze the „Layout“

HandsOn

HandsOn – Exercise 10

Pages 2, moving Elements

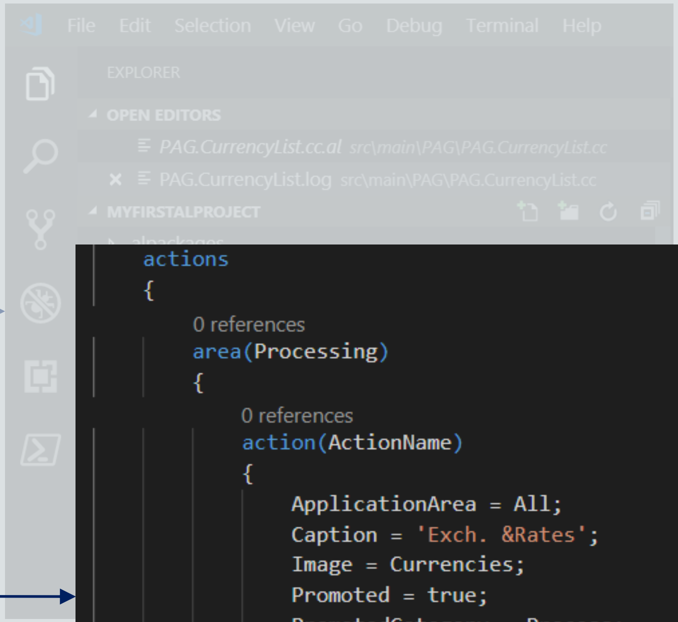
HandsOn – Exercise 10 | Pages

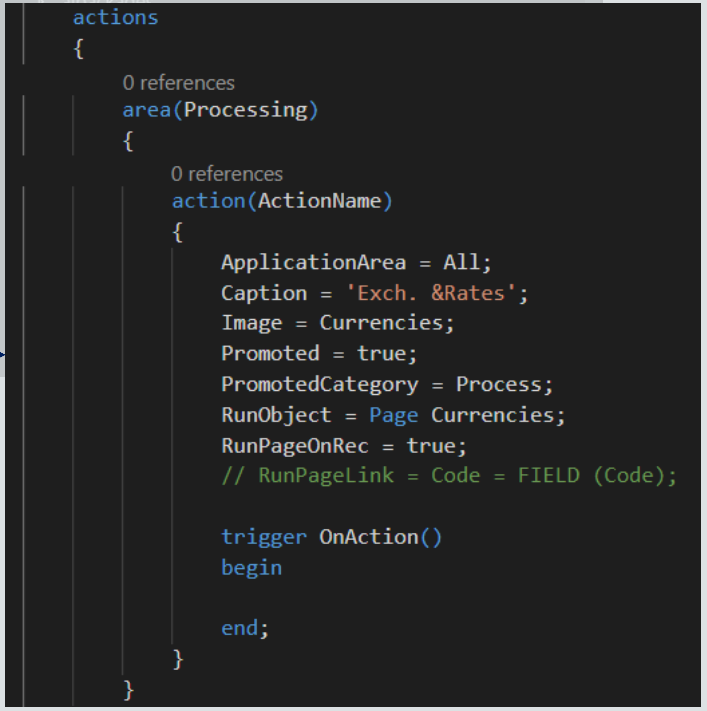
HandsOn

Implement „GAP-AL-002: Currency List“

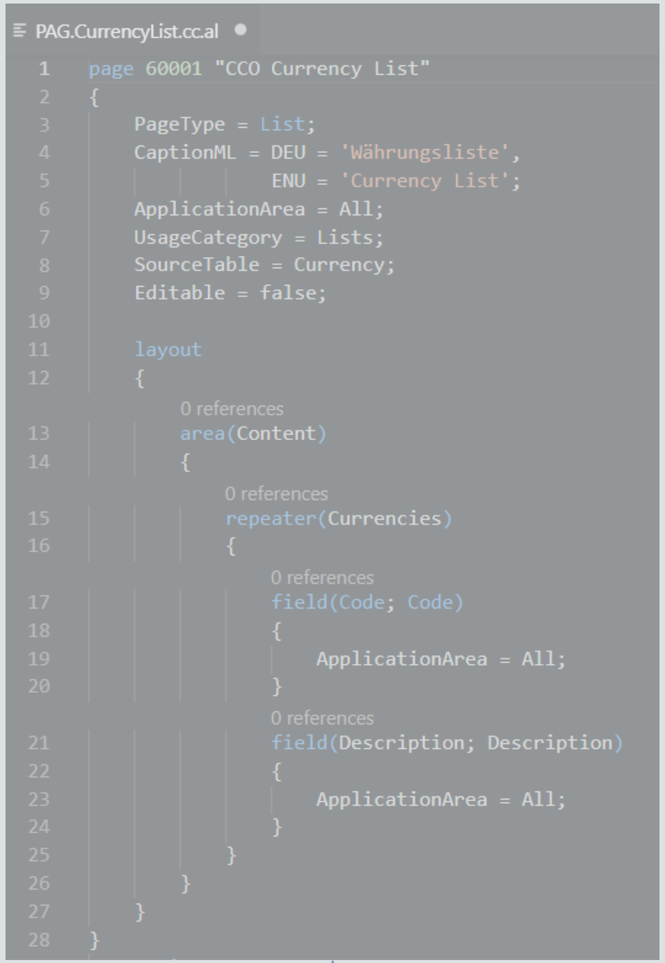
Create a simple List Page „Currency List“, which shows a list of all Currencies and their Descriptions. It shall not be possible to edit the list. Via an Action „Show extended List“, it shall be possible to open the standard currency list „Currencies“.

- Open (My first) AL Project
- Create the file structure needed
- Implement the page without the action
- Verify functionality
- Implement the desired action
- Run and Test the Project


File Edit Selection View Go Debug Terminal Help
EXPLORER
OPEN EDITORS
PAG.CurrencyList.cc.al src\main\PAG\PAG.CurrencyList.cc
PAG.CurrencyList.log src\main\PAG\PAG.CurrencyList.cc
MYFIRSTALPROJECT



```
actions
{
    0 references
    area(Processing)
    {
        0 references
        action(ActionName)
        {
            ApplicationArea = All;
            Caption = 'Exch. &Rates';
            Image = Currencies;
            Promoted = true;
            PromotedCategory = Process;
            RunObject = Page Currencies;
            RunPageOnRec = true;
            // RunPageLink = Code = FIELD (Code);

            trigger OnAction()
            begin

            end;
        }
    }
```



PAG.CurrencyList.cc.al

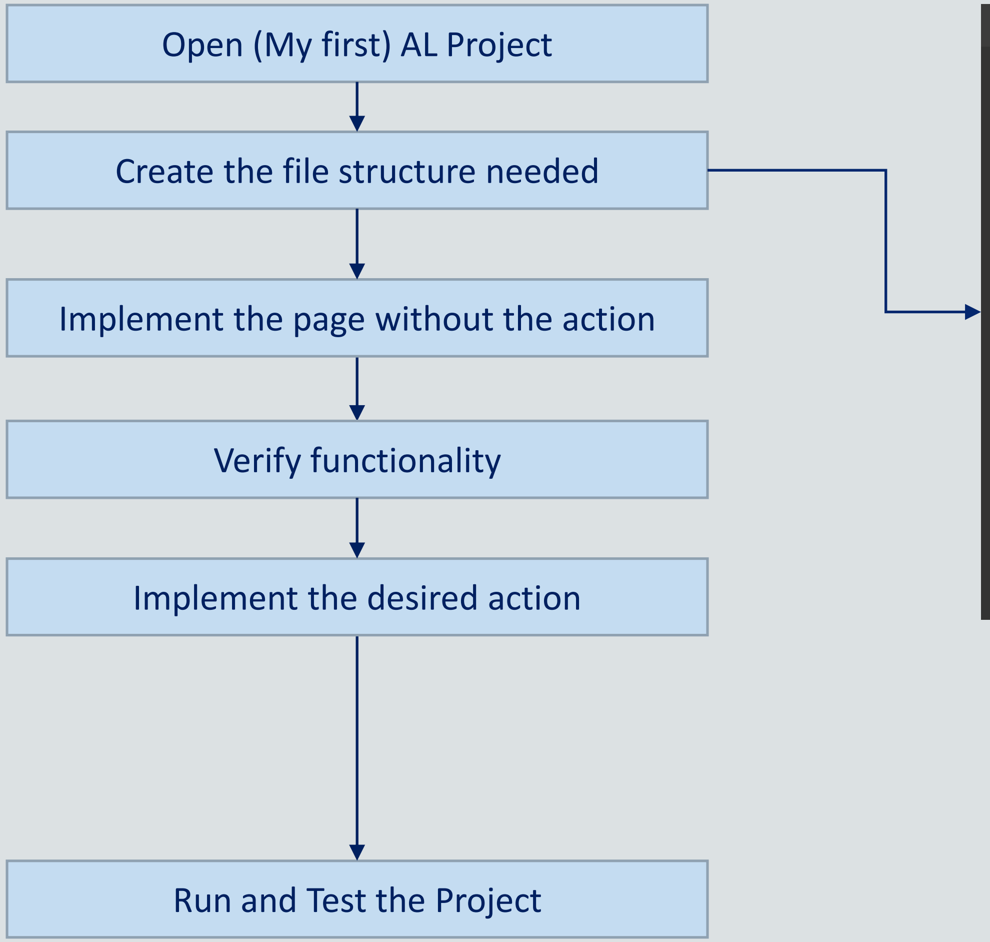
```
page 60001 "CCO Currency List"
{
    PageType = List;
    CaptionML = DEU = 'Währungsliste',
                ENU = 'Currency List';
    ApplicationArea = All;
    UsageCategory = Lists;
    SourceTable = Currency;
    Editable = false;

    layout
    {
        0 references
        area(Content)
        {
            0 references
            repeater(Currencies)
            {
                0 references
                field(Code; Code)
                {
                    ApplicationArea = All;
                }
                0 references
                field(Description; Description)
                {
                    ApplicationArea = All;
                }
            }
        }
    }
}
```

HandsOn – Exercise 10 | Pages

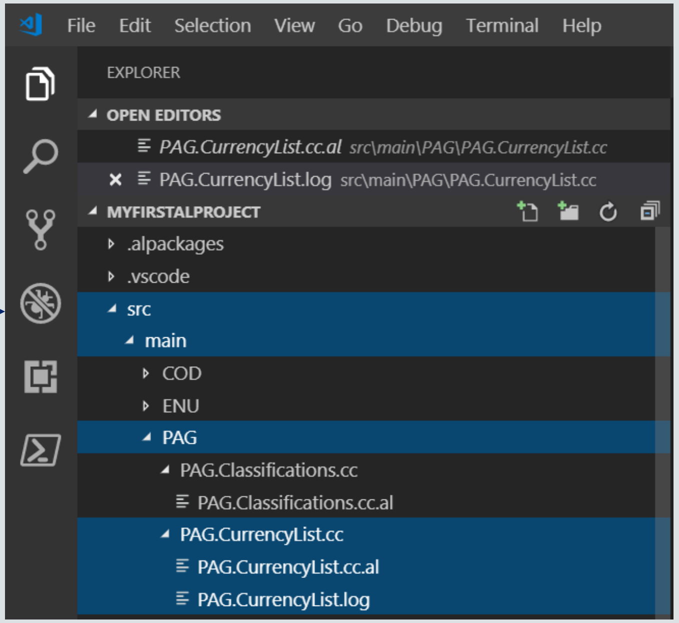
HandsOn

Implement „GAP-AL-002: Currency List“

Create a simple List Page „Currency List“, which shows a list of all Currencies and their Descriptions. It shall not be possible to edit the list. Via an Action „Show extended List“, it shall be possible to open the standard currency list „Currencies“.



Open (My first) AL Project
Create the file structure needed
Implement the page without the action
Verify functionality
Implement the desired action
Run and Test the Project



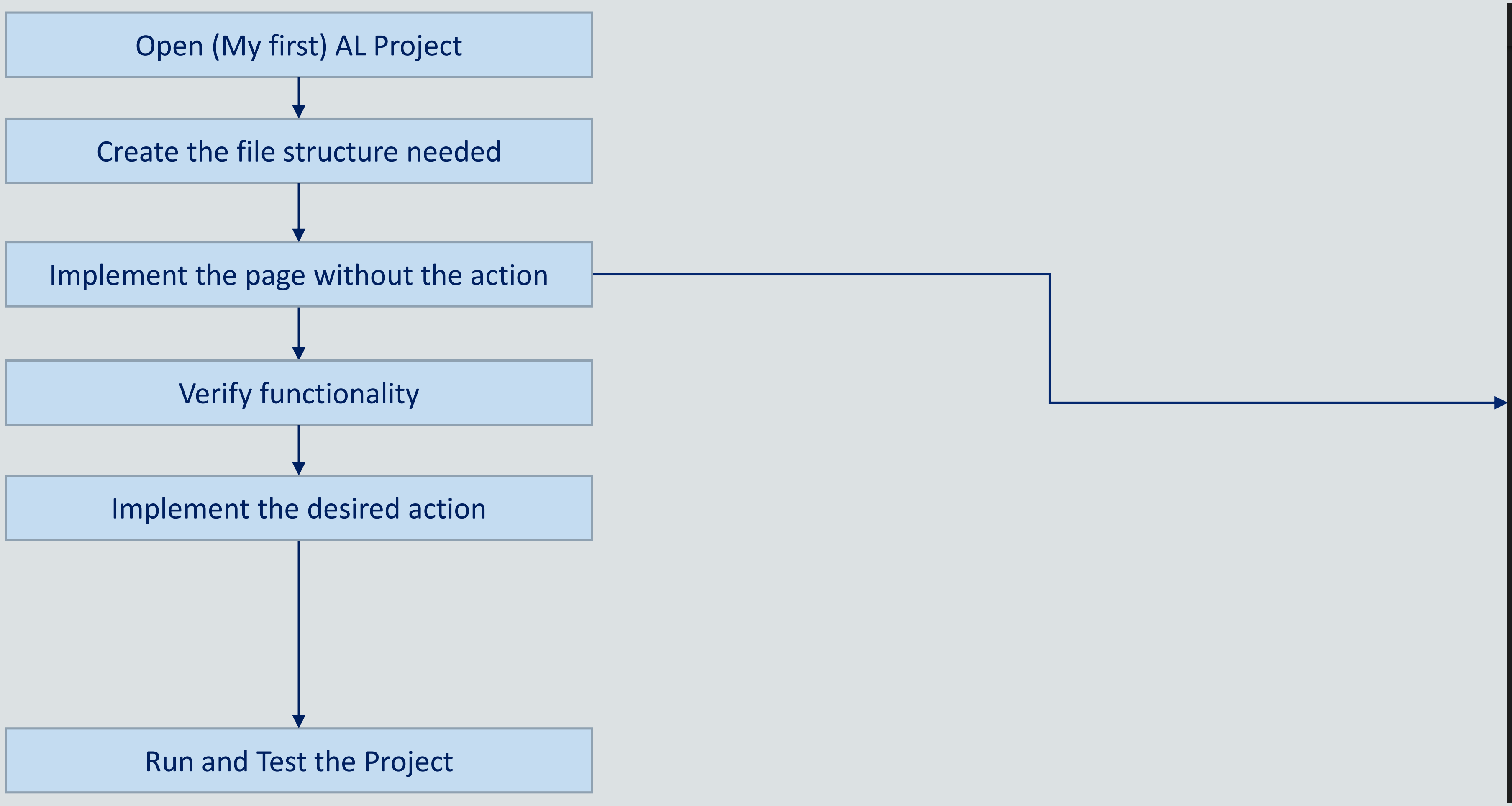
```
PAG.CurrencyList.cc.al  src\main\PAG\PAG.CurrencyList.cc
× PAG.CurrencyList.log  src\main\PAG\PAG.CurrencyList.cc
MYFIRSTALPROJECT
  .alpackages
  .vscode
  src
    main
      COD
      ENU
      PAG
        PAG.Classifications.cc
          PAG.Classifications.cc.al
        PAG.CurrencyList.cc
          PAG.CurrencyList.cc.al
          PAG.CurrencyList.log
```

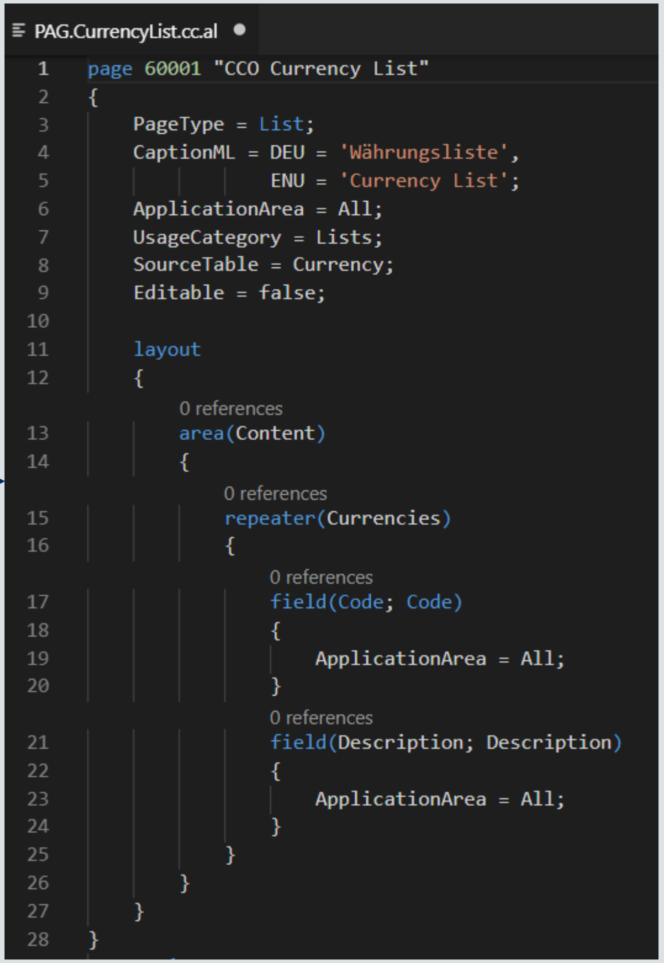
HandsOn – Exercise 10 | Pages

HandsOn

Implement „GAP-AL-002: Currency List“

Create a simple List Page „Currency List“, which shows a list of all Currencies and their Descriptions. It shall not be possible to edit the list. Via an Action „Show extended List“, it shall be possible to open the standard currency list „Currencies“.


Open (My first) AL Project
Create the file structure needed
Implement the page without the action
Verify functionality
Implement the desired action
Run and Test the Project



```
page 60001 "CCO Currency List"
{
    PageType = List;
    CaptionML = DEU = 'Währungsliste',
                ENU = 'Currency List';
    ApplicationArea = All;
    UsageCategory = Lists;
    SourceTable = Currency;
    Editable = false;

    layout
    {
        0 references
        area(Content)
        {
            0 references
            repeater(Currencies)
            {
                0 references
                field(Code; Code)
                {
                    ApplicationArea = All;
                }
                0 references
                field(Description; Description)
                {
                    ApplicationArea = All;
                }
            }
        }
    }
}
```

HandsOn – Exercise 10 | Pages

HandsOn

Implement „GAP-AL-002: Currency List“

Create a simple List Page „Currency List“, which shows a list of all Currencies and their Descriptions. It shall not be possible to edit the list. Via an Action „Show extended List“, it shall be possible to open the standard currency list „Currencies“.

Open (My first) AL Project

↓

Create the file structure needed

↓

Implement the page without the action

↓

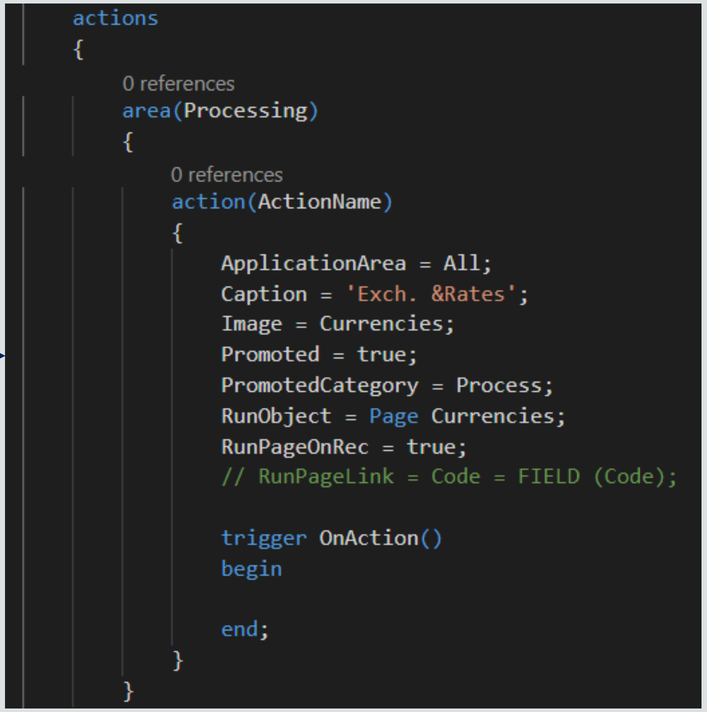
Verify functionality

↓

Implement the desired action →

↓

Run and Test the Project



```
actions
{
    0 references
    area(Processing)
    {
        0 references
        action(ActionName)
        {
            ApplicationArea = All;
            Caption = 'Exch. &Rates';
            Image = Currencies;
            Promoted = true;
            PromotedCategory = Process;
            RunObject = Page Currencies;
            RunPageOnRec = true;
            // RunPageLink = Code = FIELD (Code);

            trigger OnAction()
            begin

            end;
        }
    }
```

HandsOn

HandsOn – Exercise 11

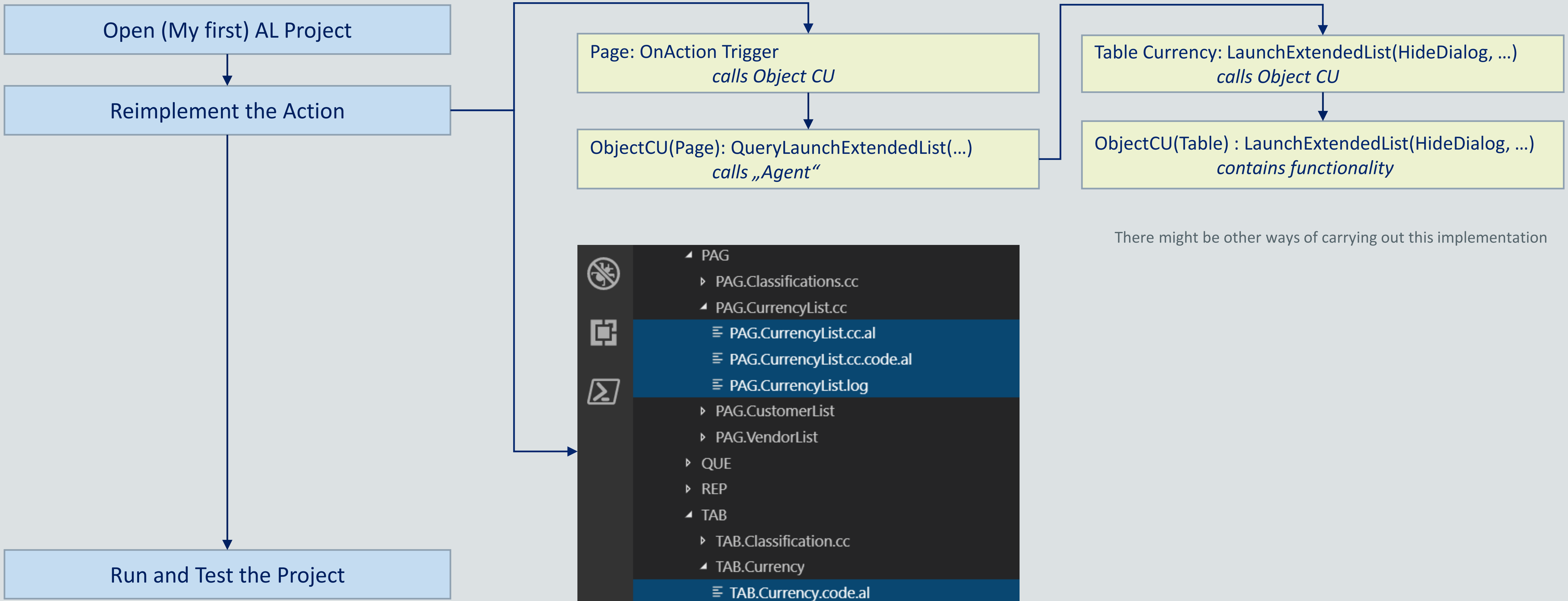
Object / File Structure

HandsOn – Exercise 11 | Object / File Structure

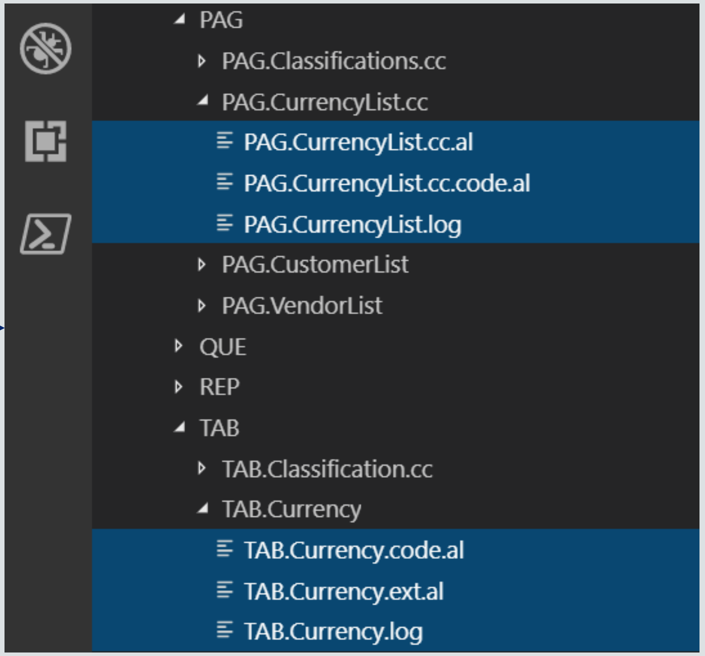
HandsOn

Implement „GAP-AL-003: Currency List advanced“

Extend the functionality of „GAP-AL-002 in a way that the user is prompted „Do you want to open the standard currency list“ when he hits the „Show extended List“ button.


Open (My first) AL Project
Reimplement the Action
Run and Test the Project
Page: OnAction Trigger *calls Object CU*
ObjectCU(Page): QueryLaunchExtendedList(…) *calls „Agent“*
Table Currency: LaunchExtendedList(HideDialog, …) *calls Object CU*
ObjectCU(Table) : LaunchExtendedList(HideDialog, …) *contains functionality*

There might be other ways of carrying out this implementation



```
◢ PAG
  ▷ PAG.Classifications.cc
  ◢ PAG.CurrencyList.cc
      PAG.CurrencyList.cc.al
      PAG.CurrencyList.cc.code.al
      PAG.CurrencyList.log
  ▷ PAG.CustomerList
  ▷ PAG.VendorList
▷ QUE
▷ REP
◢ TAB
  ▷ TAB.Classification.cc
  ◢ TAB.Currency
      TAB.Currency.code.al
      TAB.Currency.ext.al
      TAB.Currency.log
```

HandsOn

HandsOn – Exercise 11.5

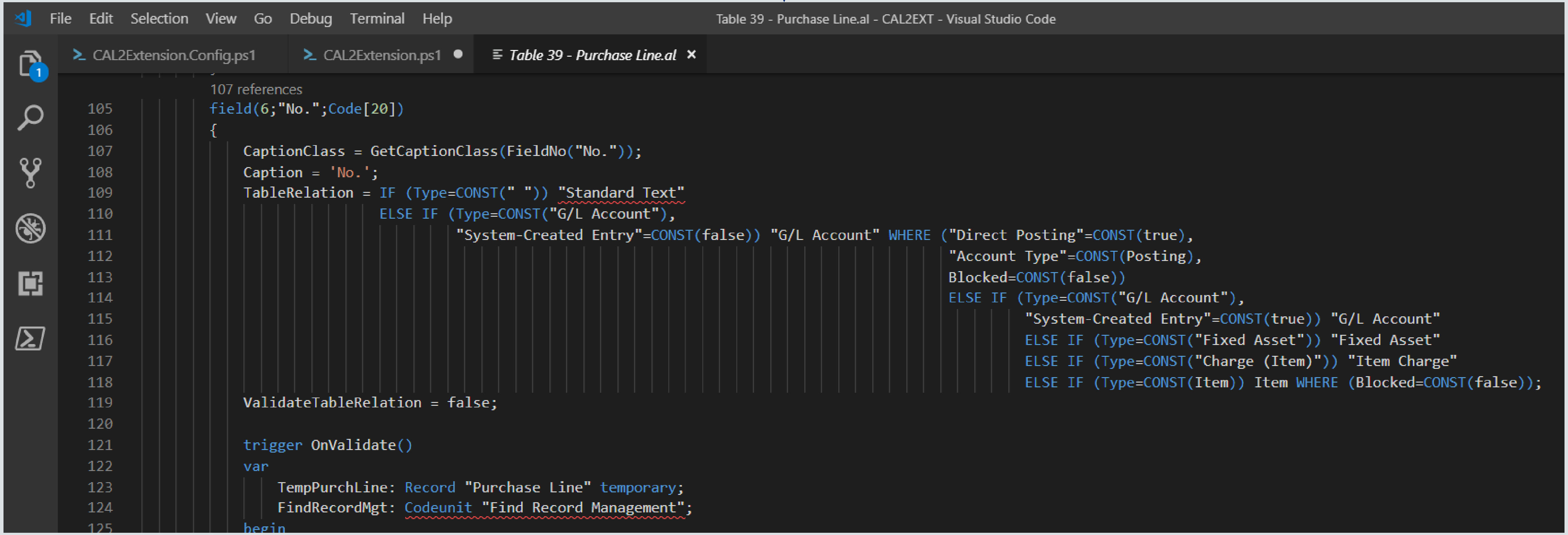
Table Relations and Txt2AL converter

HandsOn – Exercise 11.5 | Table Relations and the Txt2AL converter

Gain insight into the syntax of Table Relations by „Txt2AL Converting“ table 39 „Purchase Line“

HandsOn

Configure Txt2AL COnverter → Convert Table 39 „Purchase Line“ → Learn the syntax of „Table Relation Property“



```
field(6;"No.";Code[20])
{
    CaptionClass = GetCaptionClass(FieldNo("No."));
    Caption = 'No.';
    TableRelation = IF (Type=CONST(" ")) "Standard Text"
                    ELSE IF (Type=CONST("G/L Account"),
                             "System-Created Entry"=CONST(false)) "G/L Account" WHERE ("Direct Posting"=CONST(true),
                                                                                     "Account Type"=CONST(Posting),
                                                                                     Blocked=CONST(false))
                                                                                     ELSE IF (Type=CONST("G/L Account"),
                                                                                              "System-Created Entry"=CONST(true)) "G/L Account"
                                                                                              ELSE IF (Type=CONST("Fixed Asset")) "Fixed Asset"
                                                                                              ELSE IF (Type=CONST("Charge (Item)")) "Item Charge"
                                                                                              ELSE IF (Type=CONST(Item)) Item WHERE (Blocked=CONST(false));
    ValidateTableRelation = false;

    trigger OnValidate()
    var
        TempPurchLine: Record "Purchase Line" temporary;
        FindRecordMgt: Codeunit "Find Record Management";
    begin
```

HandsOn

HandsOn – Exercise 12

Module Hazard Category Part 1

HandsOn – Exercise 12 | Module Hazard Category Part 1

HandsOn

FRD

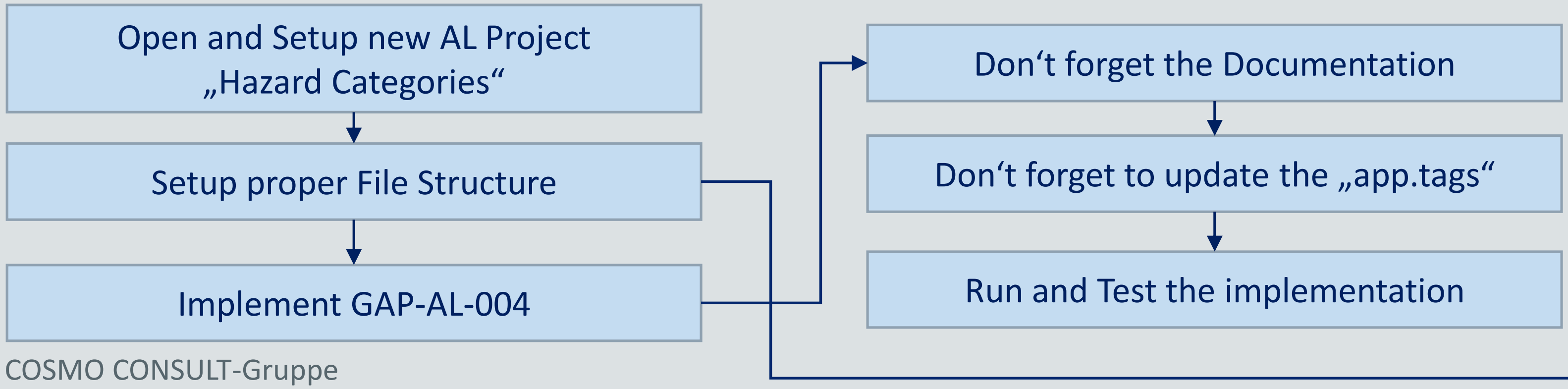
It shall be possible to group Items according to their "Hazard Category". "Hazard Categories" should be freely definable. When an article is purchased via ordering, its "Hazard Category" is to be displayed in the purchasing document and should be changeable there. During the course of item posting, the "Hazard Category" has to be transferred into the "Item Ledger Entries" and, if indicated, into the posted purchase documents.

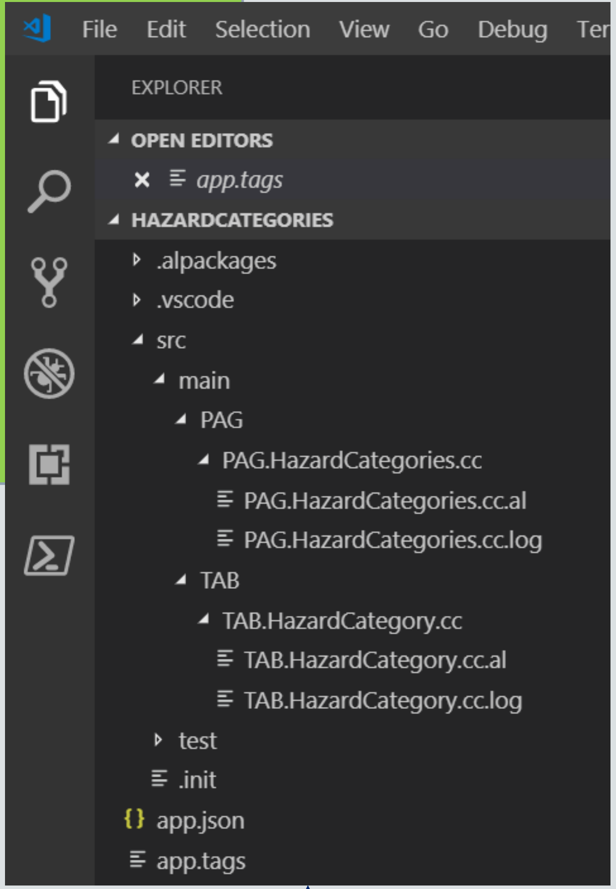
No other processes are to be respected.
No further details have to be respected concerning, e.g., manipulations of partially posted documents etc..

GAP-AL-004: Hazard Categories
A new Data Structure “Hazard Category” is to be implemented
(Table 60100 “Hazard Category” and List Page 60100 “Hazard Categories”).
Hazard Category

	Code	*Code 20*
	Description	*Text 80*

It shall not be possible to enter “empty” categories.
The new page shall be accessible via the search functionality.


Open and Setup new AL Project „Hazard Categories“
Setup proper File Structure
Implement GAP-AL-004
Don‘t forget the Documentation
Don‘t forget to update the „app.tags“
Run and Test the implementation



```
.alpackages
.vscode
src
  main
    PAG
      PAG.HazardCategories.cc
        PAG.HazardCategories.cc.al
        PAG.HazardCategories.cc.log
    TAB
      TAB.HazardCategory.cc
        TAB.HazardCategory.cc.al
        TAB.HazardCategory.cc.log
  test
  .init
app.json
app.tags
```

HandsOn

HandsOn – Exercise 13

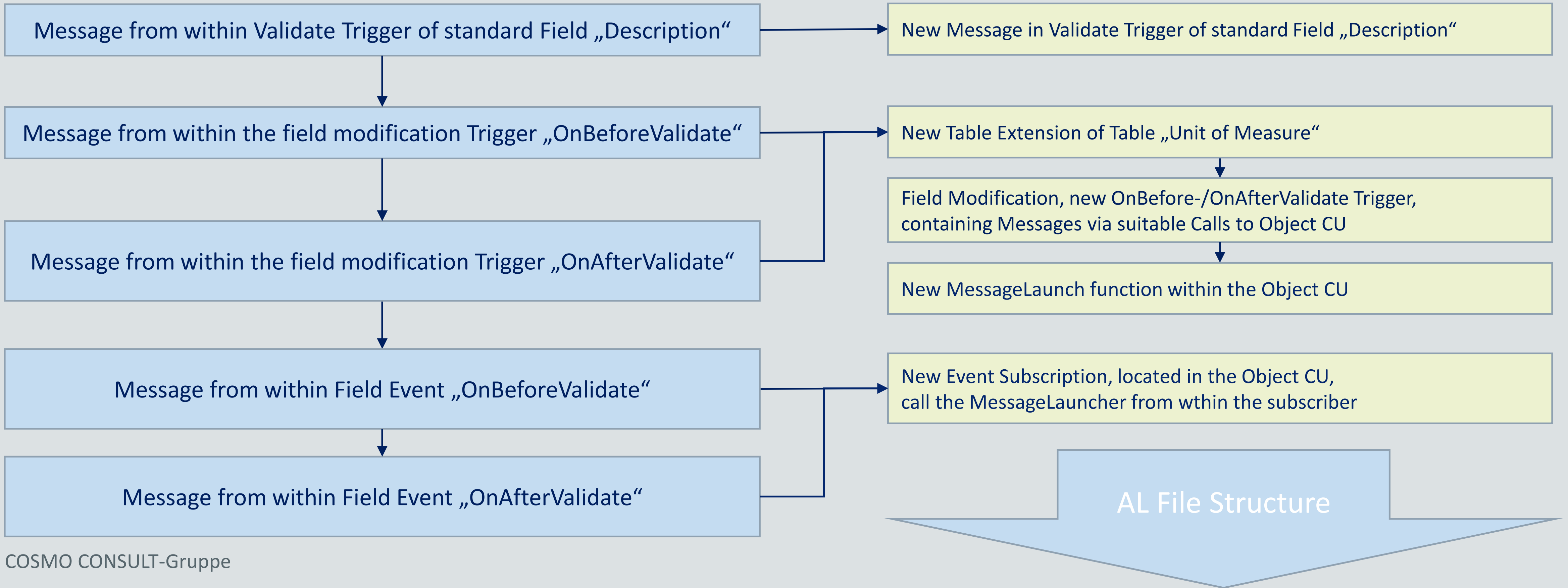
Table Extension (Table) Trigger

HandsOn – Exercise 13 | Table Extension (Table) Trigger

HandsOn

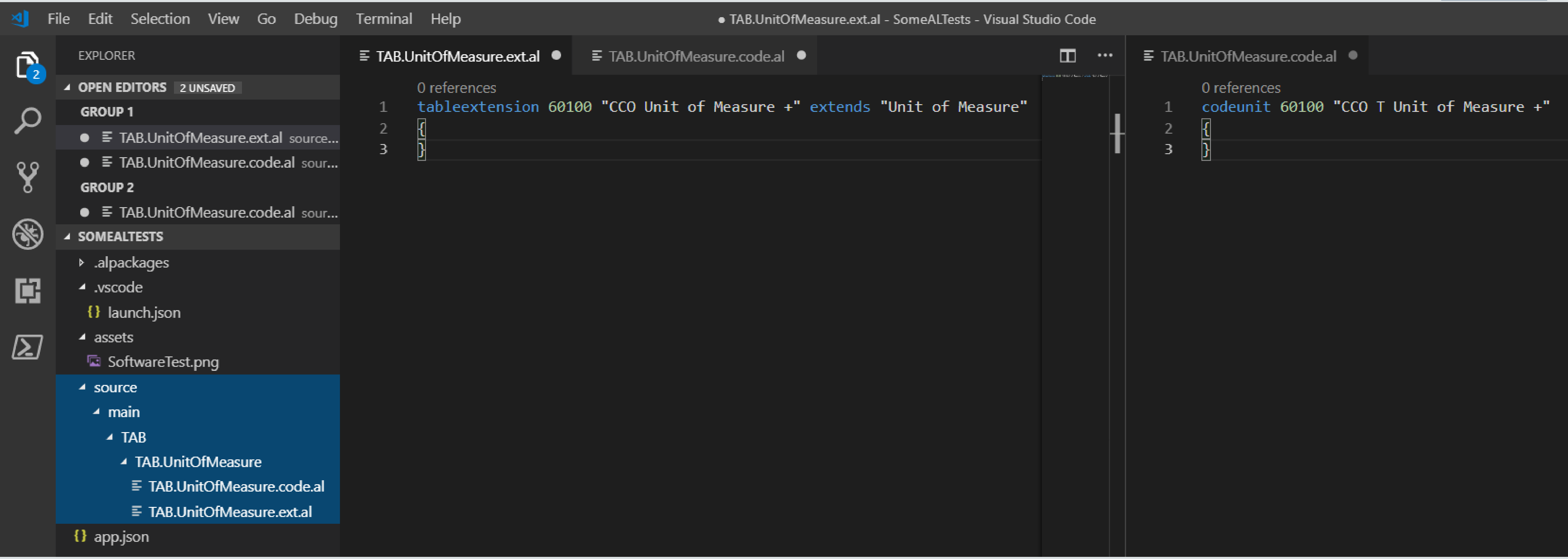
Sequence of Execution of table extension field triggers compared to relevant standard triggers and events.
Implement an Extension which is suited to show the order of execution of „OnBeforeValidate (Extension Trigger)“, „OnBeforeValidate (Table Field Event)“, „OnValidate (Table Field Trigger)“, „OnAfterValidate (Table Field Event)“ and „OnAfterValidate (Extension Trigger)“, use „Description“ field of „Unit of Measure“ as an example.
Implement a Message-Chain which is suited to show the order of execution.

Scetch of Design


Message from within Validate Trigger of standard Field „Description“
Message from within the field modification Trigger „OnBeforeValidate“
Message from within the field modification Trigger „OnAfterValidate“
Message from within Field Event „OnBeforeValidate“
Message from within Field Event „OnAfterValidate“
New Message in Validate Trigger of standard Field „Description“
New Table Extension of Table „Unit of Measure“
Field Modification, new OnBefore-/OnAfterValidate Trigger, containing Messages via suitable Calls to Object CU
New MessageLaunch function within the Object CU
New Event Subscription, located in the Object CU, call the MessageLauncher from wthin the subscriber
AL File Structure

HandsOn – Exercise 13 | AL File Structure

HandsOn


File
Edit
Selection
View
Go
Debug
Terminal
Help
● TAB.UnitOfMeasure.ext.al - SomeALTests - Visual Studio Code
EXPLORER
OPEN EDITORS 2 UNSAVED
GROUP 1
TAB.UnitOfMeasure.ext.al source...
TAB.UnitOfMeasure.code.al sour...
GROUP 2
TAB.UnitOfMeasure.code.al sour...
SOMEALTESTS
.alpackages
.vscode
launch.json
assets
SoftwareTest.png
source
main
TAB
TAB.UnitOfMeasure
TAB.UnitOfMeasure.code.al
TAB.UnitOfMeasure.ext.al
app.json
TAB.UnitOfMeasure.ext.al
TAB.UnitOfMeasure.code.al
TAB.UnitOfMeasure.code.al

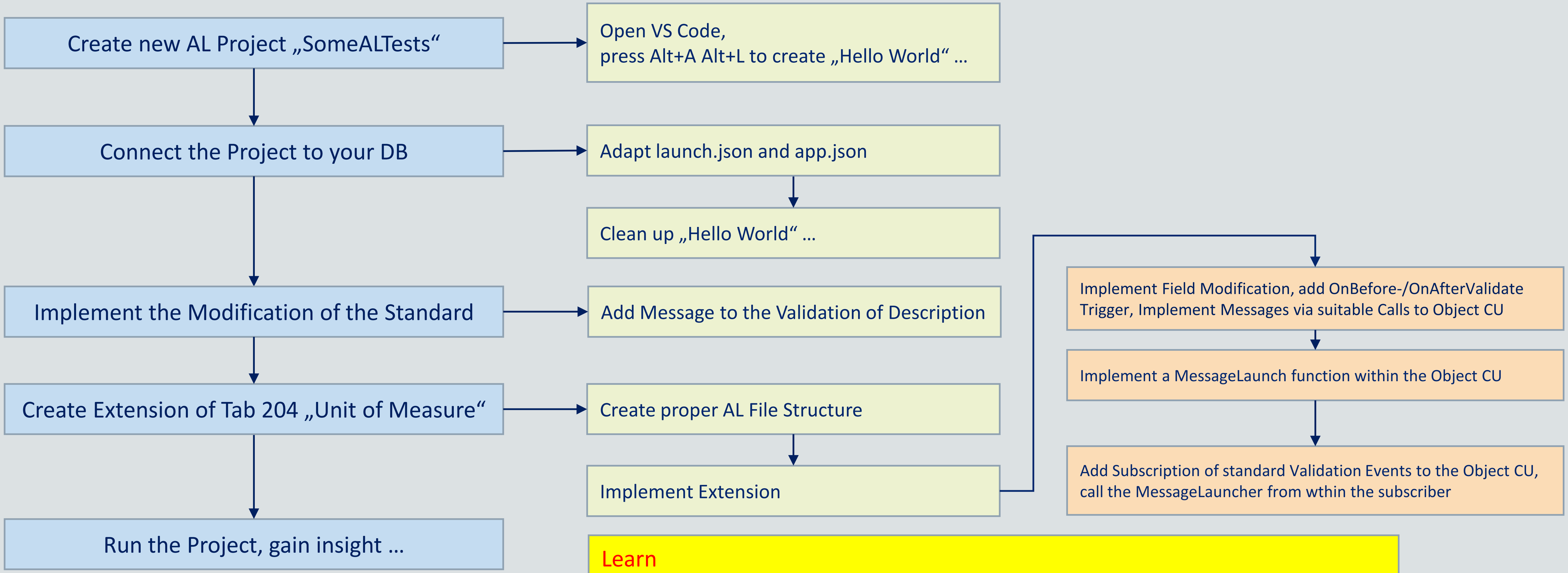
```
0 references
tableextension 60100 "CCO Unit of Measure +" extends "Unit of Measure"
{
}
```

```
0 references
codeunit 60100 "CCO T Unit of Measure +"
{
}
```

HandsOn – Exercise 13 | Doing

HandsOn

Sequence of Execution of table extension field triggers compared to relevant standard triggers and events.
Implement an Extension which is suited to show the order of execution of „OnBeforeValidate (Extension Trigger)“, „OnBeforeValidate (Table Field Event)“, „OnValidate (Table Field Trigger)“, „OnAfterValidate (Table Field Event)“ and „OnAfterValidate (Extension Trigger)“, use „Description“ field of „Unit of Measure“ as an example.
Implement a Message-Chain which is suited to show the order of execution.


Create new AL Project „SomeALTests“
Open VS Code,
press Alt+A Alt+L to create „Hello World“ …
Connect the Project to your DB
Adapt launch.json and app.json
Clean up „Hello World“ …
Implement the Modification of the Standard
Add Message to the Validation of Description
Create Extension of Tab 204 „Unit of Measure“
Create proper AL File Structure
Implement Extension
Implement Field Modification, add OnBefore-/OnAfterValidate Trigger, Implement Messages via suitable Calls to Object CU
Implement a MessageLaunch function within the Object CU
Add Subscription of standard Validation Events to the Object CU, call the MessageLauncher from wthin the subscriber
Run the Project, gain insight …

Learn
1) how extension field trigger fit into the total field trigger / event scenario
2) that invoking intelliSense is always a good idea

HandsOn – Exercise 13 | Hybrid Implementation

HandsOn

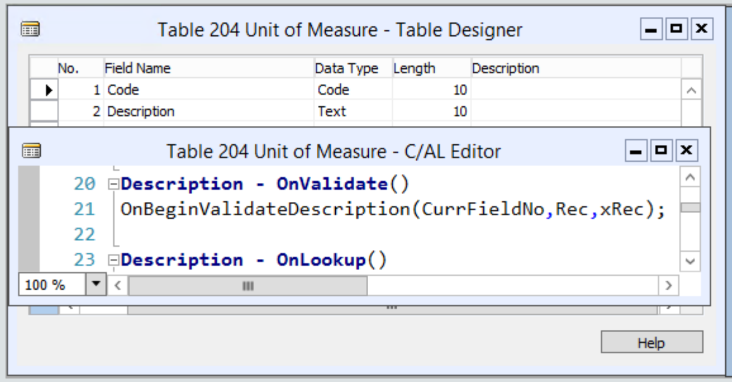
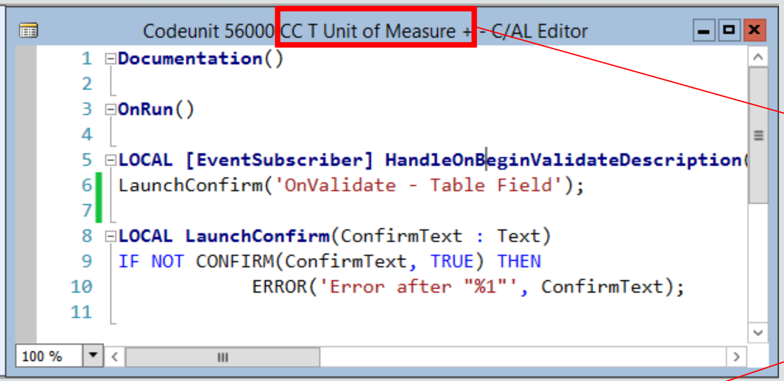


Table 204 Unit of Measure - Table Designer

No.	Field Name	Data Type	Length	Description
1	Code	Code	10	
2	Description	Text	10	

Table 204 Unit of Measure - C/AL Editor

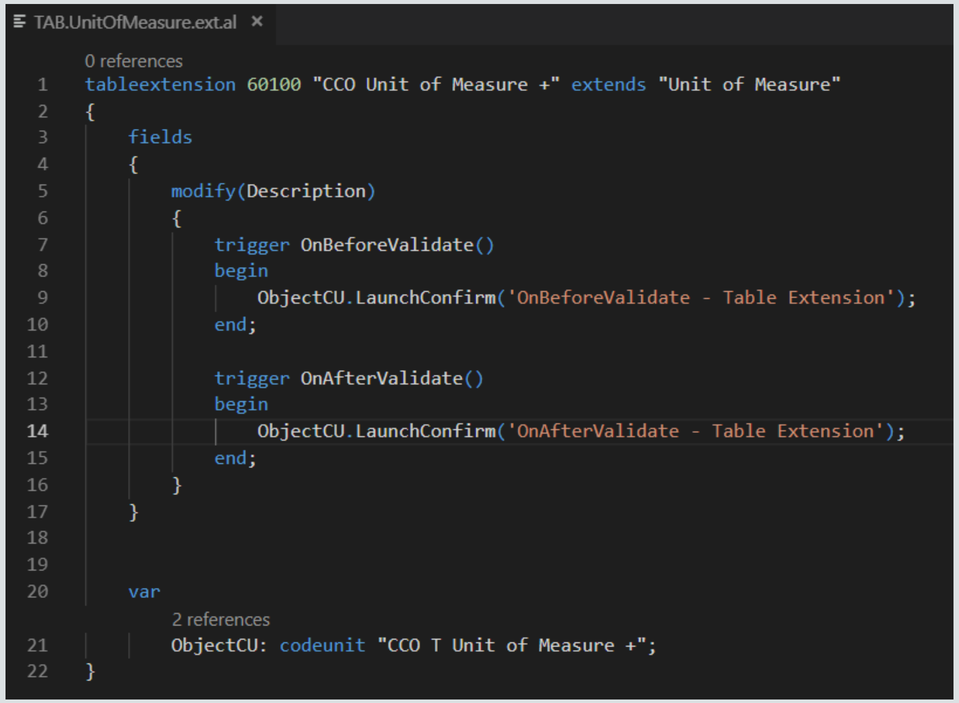
```
Description - OnValidate()
OnBeginValidateDescription(CurrFieldNo,Rec,xRec);

Description - OnLookup()
```



Codeunit 56000 CC T Unit of Measure + - C/AL Editor

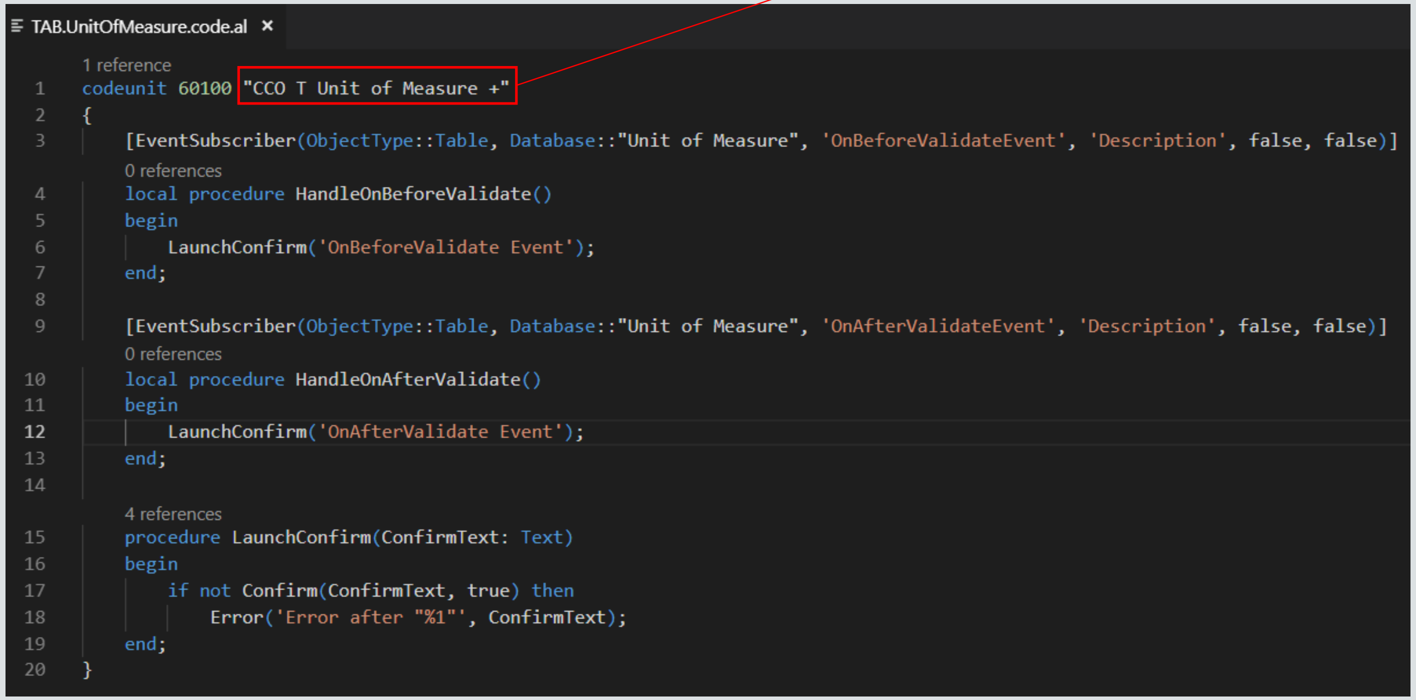
```
Documentation()

OnRun()

LOCAL [EventSubscriber] HandleOnBeginValidateDescription(
LaunchConfirm('OnValidate - Table Field');

LOCAL LaunchConfirm(ConfirmText : Text)
IF NOT CONFIRM(ConfirmText, TRUE) THEN
          ERROR('Error after "%1"', ConfirmText);
```

!!!



TAB.UnitOfMeasure.ext.al

```
0 references
tableextension 60100 "CCO Unit of Measure +" extends "Unit of Measure"
{
    fields
    {
        modify(Description)
        {
            trigger OnBeforeValidate()
            begin
                ObjectCU.LaunchConfirm('OnBeforeValidate - Table Extension');
            end;

            trigger OnAfterValidate()
            begin
                ObjectCU.LaunchConfirm('OnAfterValidate - Table Extension');
            end;
        }
    }


    var
        2 references
        ObjectCU: codeunit "CCO T Unit of Measure +";
}
```



TAB.UnitOfMeasure.code.al

```
1 reference
codeunit 60100 "CCO T Unit of Measure +"
{
    [EventSubscriber(ObjectType::Table, Database::"Unit of Measure", 'OnBeforeValidateEvent', 'Description', false, false)]
    0 references
    local procedure HandleOnBeforeValidate()
    begin
        LaunchConfirm('OnBeforeValidate Event');
    end;

    [EventSubscriber(ObjectType::Table, Database::"Unit of Measure", 'OnAfterValidateEvent', 'Description', false, false)]
    0 references
    local procedure HandleOnAfterValidate()
    begin
        LaunchConfirm('OnAfterValidate Event');
    end;

    4 references
    procedure LaunchConfirm(ConfirmText: Text)
    begin
        if not Confirm(ConfirmText, true) then
            Error('Error after "%1"', ConfirmText);
    end;
}
```

HandsOn

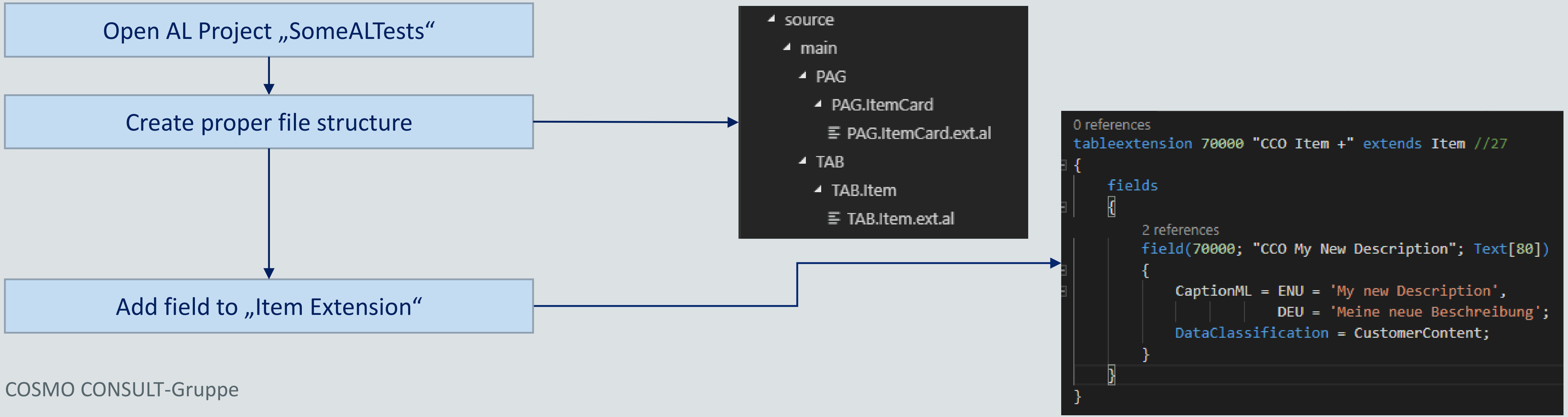
HandsOn – Exercise 14

Page Extension Layout

HandsOn – Exercise 14 | Page Extension Layout

HandsOn

Create an Item (Card) extension
- Item Table: Add a new field „My new Description“ (Text 50)
- Item Card
  - Add a label „Here comes the Item No.“ on first place of fast tab „Item“
  - Move the „Replenishment System“ from where ever it is at the right bottom of the „Item“ fast tab.
  - Add a „Description Group“ at the right bottom of the „Item“ fast tab, containing all three Item Descriptions
  - Rename „Item“ fast tab to „Item Fast Tab“ („Artikelregister“)
  - Add a new first fast tab „Descriptions“, place all three „Descriptions“ on this fast tab
  - Add an action to the „Sales“ section of the „Navigate“ menu which launches the currency list.


Open AL Project „SomeALTests“
Create proper file structure
Add field to „Item Extension“

```
◢ source
  ◢ main
    ◢ PAG
      ◢ PAG.ItemCard
          PAG.ItemCard.ext.al
    ◢ TAB
      ◢ TAB.Item
          TAB.Item.ext.al
```

```
0 references
tableextension 70000 "CCO Item +" extends Item //27
{
    fields
    {
        2 references
        field(70000; "CCO My New Description"; Text[80])
        {
            CaptionML = ENU = 'My new Description',
                        DEU = 'Meine neue Beschreibung';
            DataClassification = CustomerContent;
        }
    }
}
```

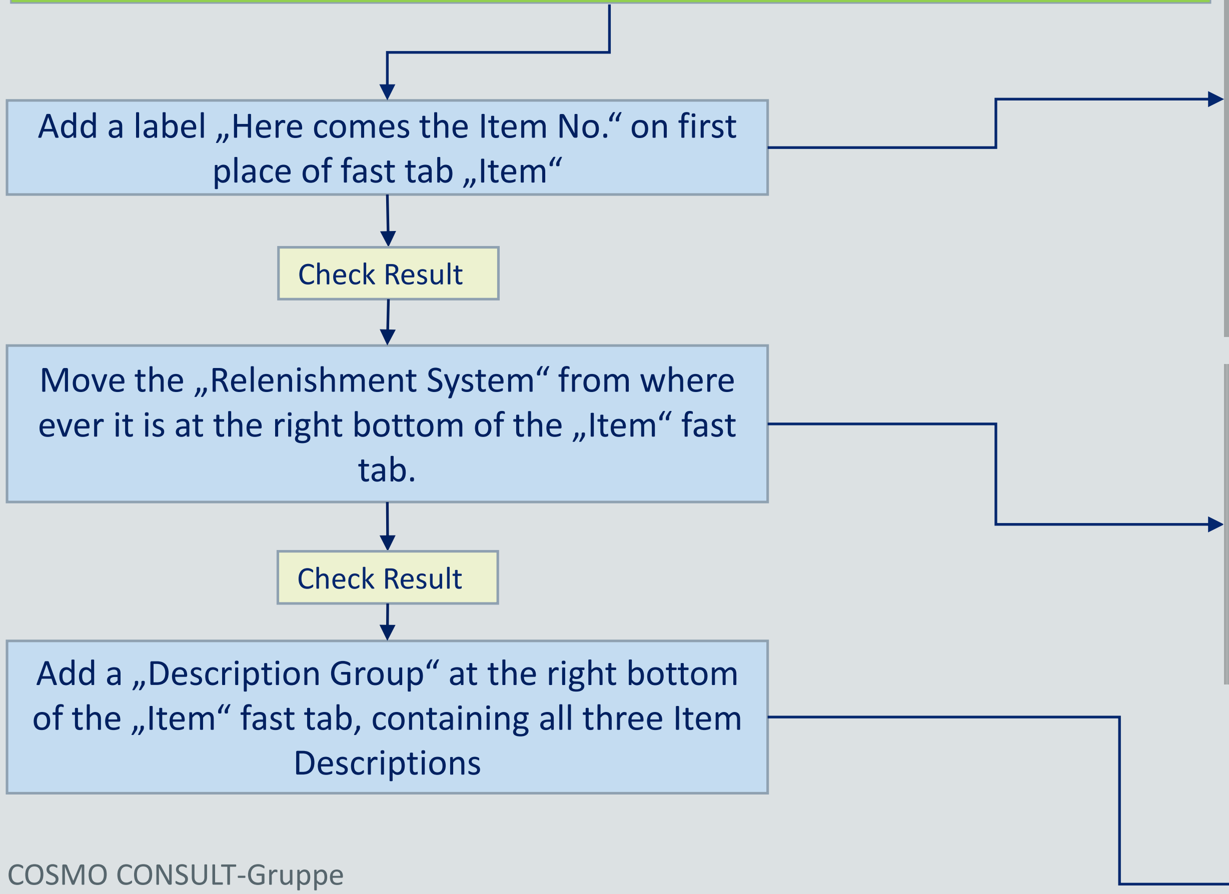
There might be other ways of carrying out this implementation

HandsOn – Exercise 14 | Page Extension Layout, continued 1

HandsOn

Create an Item (Card) extension

- Item Table: Add a new field „My new Description“ (Text 50)
- Item Card
  - Add a label „Here comes the Item No.“ on first place of fast tab „Item“
  - Move the „Relenishment System“ from whereever it is at the right bottom of the „Item“ fast tab.
  - Add a „Description Group“ at the right bottom of the „Item“ fast tab, containing all three Item Descriptions
  - Rename „Item“ fast tab to „Item Fast Tab“ („Artikelregister“)
  - Add a new first fast tab „Descriptions“, place all three „Descriptions“ on this fast tab
  - Add an action to the „Sales“ section of the „Navigate“ menu which launches the currency list.



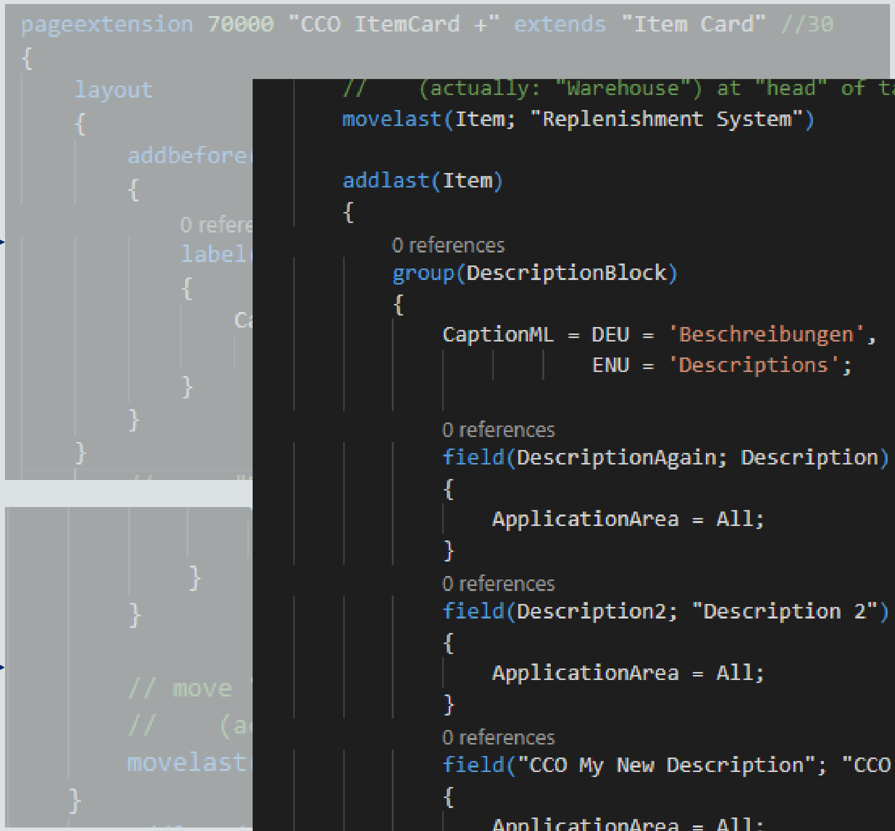
Add a label „Here comes the Item No.“ on first place of fast tab „Item“

Check Result

Move the „Relenishment System“ from where ever it is at the right bottom of the „Item“ fast tab.

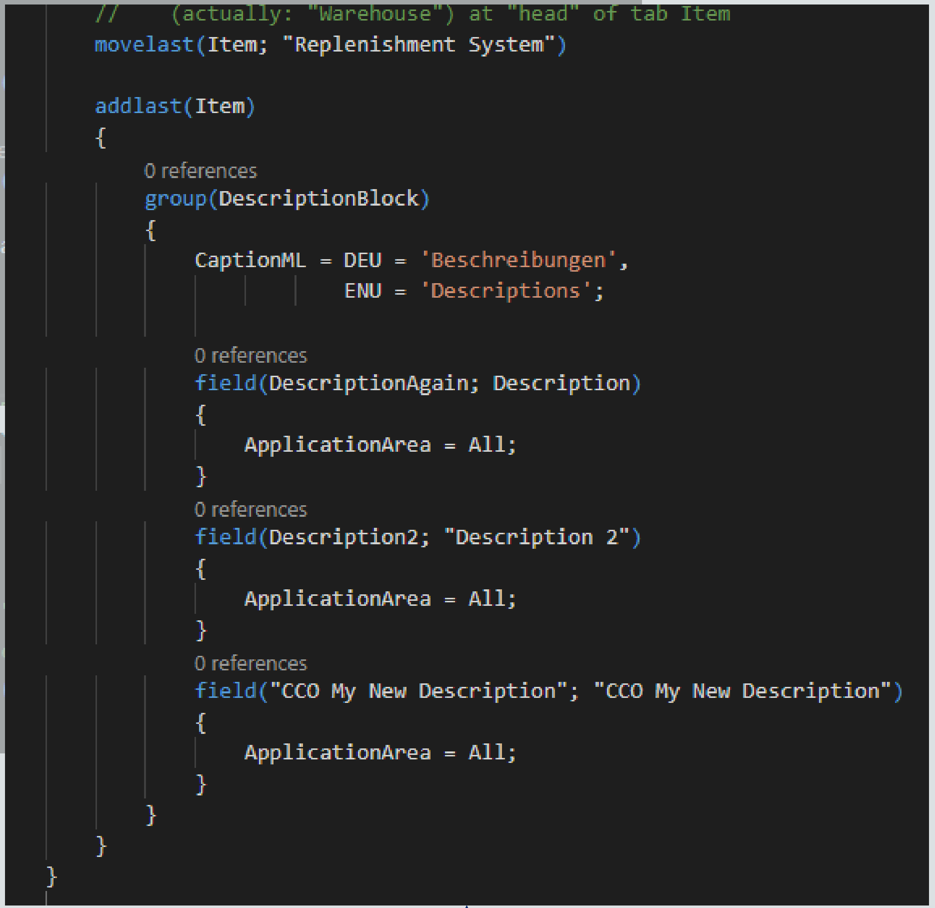
Check Result

Add a „Description Group“ at the right bottom of the „Item“ fast tab, containing all three Item Descriptions



```
pageextension 70000 "CCO ItemCard +" extends "Item Card" //30
{
    layout
    {
        addbefore
        {
            0 refere
            label
            {
                Ca

            }
        }
    }
```

```
            }
        }

        // move
        //     (a
        movelast
    }
```



```
        //    (actually: "Warehouse") at "head" of tab Item
        movelast(Item; "Replenishment System")

        addlast(Item)
        {
            0 references
            group(DescriptionBlock)
            {
                CaptionML = DEU = 'Beschreibungen',
                            ENU = 'Descriptions';

                0 references
                field(DescriptionAgain; Description)
                {
                    ApplicationArea = All;
                }
                0 references
                field(Description2; "Description 2")
                {
                    ApplicationArea = All;
                }
                0 references
                field("CCO My New Description"; "CCO My New Description")
                {
                    ApplicationArea = All;
                }
            }
        }
    }
```

There might be other ways of carrying out this implementation

HandsOn – Exercise 14 | Page Extension Layout, continued 2

HandsOn

Create an Item (Card) extension

- Item Table: Add a new field „My new Description“ (Text 50)
- Item Card
  - Add a label „Here comes the Item No.“ on first place of fast tab „Item“
  - Move the „Relenishment System“ from whereever it is at the right bottom of the „Item“ fast tab.
  - Add a „Description Group“ at the right bottom of the „Item“ fast tab, containing all three Item Descriptions
  - Rename „Item“ fast tab to „Item Fast Tab“ („Artikelregister“)
  - Add a new first fast tab „Descriptions“, place all three „Descriptions“ on this fast tab
  - Add an action to the „Sales“ section of the „Navigate“ menu which launches the currency list.

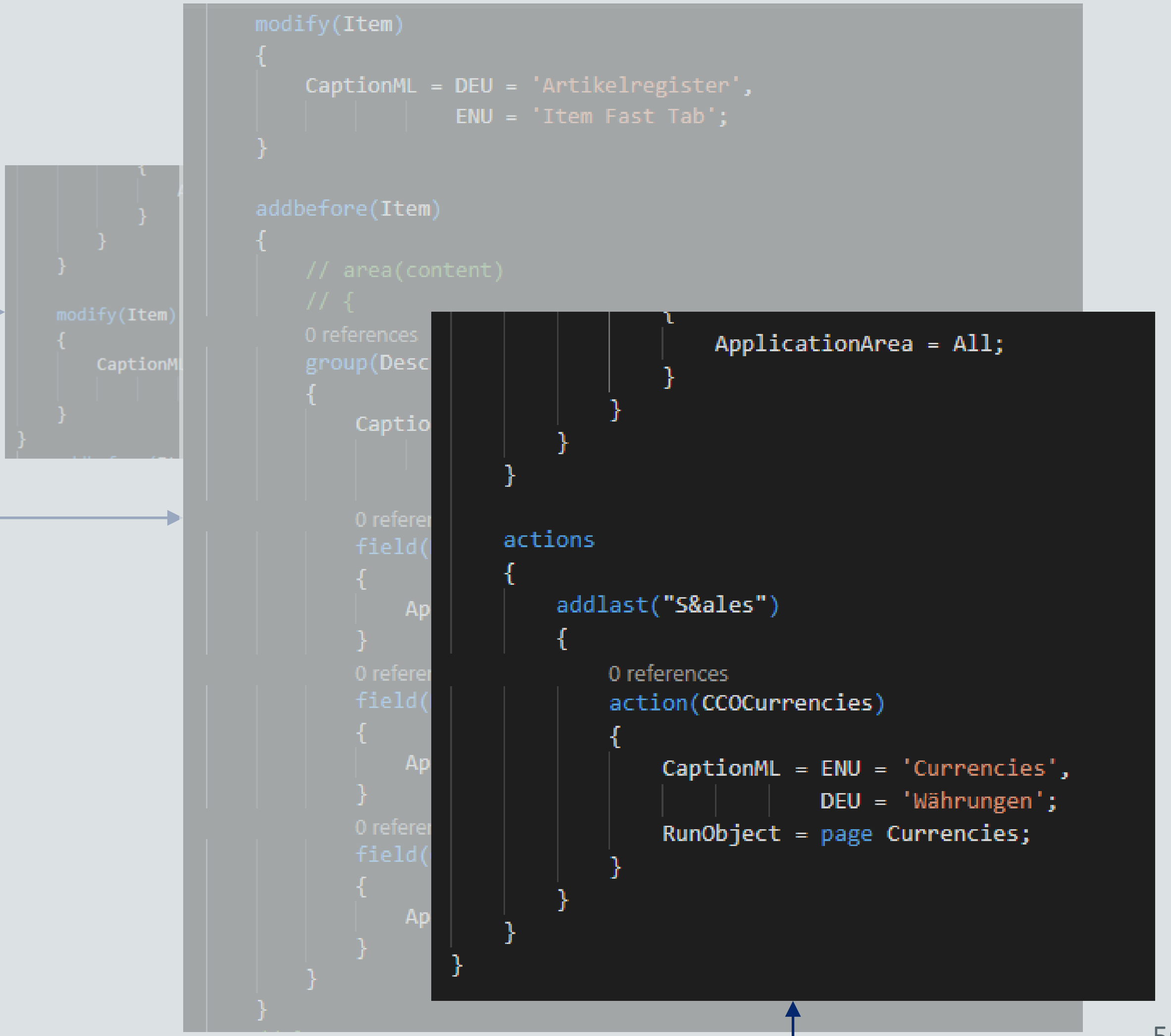
Rename „Item“ fast tab to „Item Fast Tab“ („Artikelregister“)

Check Result

Add a new first fast tab „Descriptions“, place all three „Descriptions“ on this fast tab

Check Result

Add an action to the „Sales“ section of the „Navigate“ menu which launches the currency list.



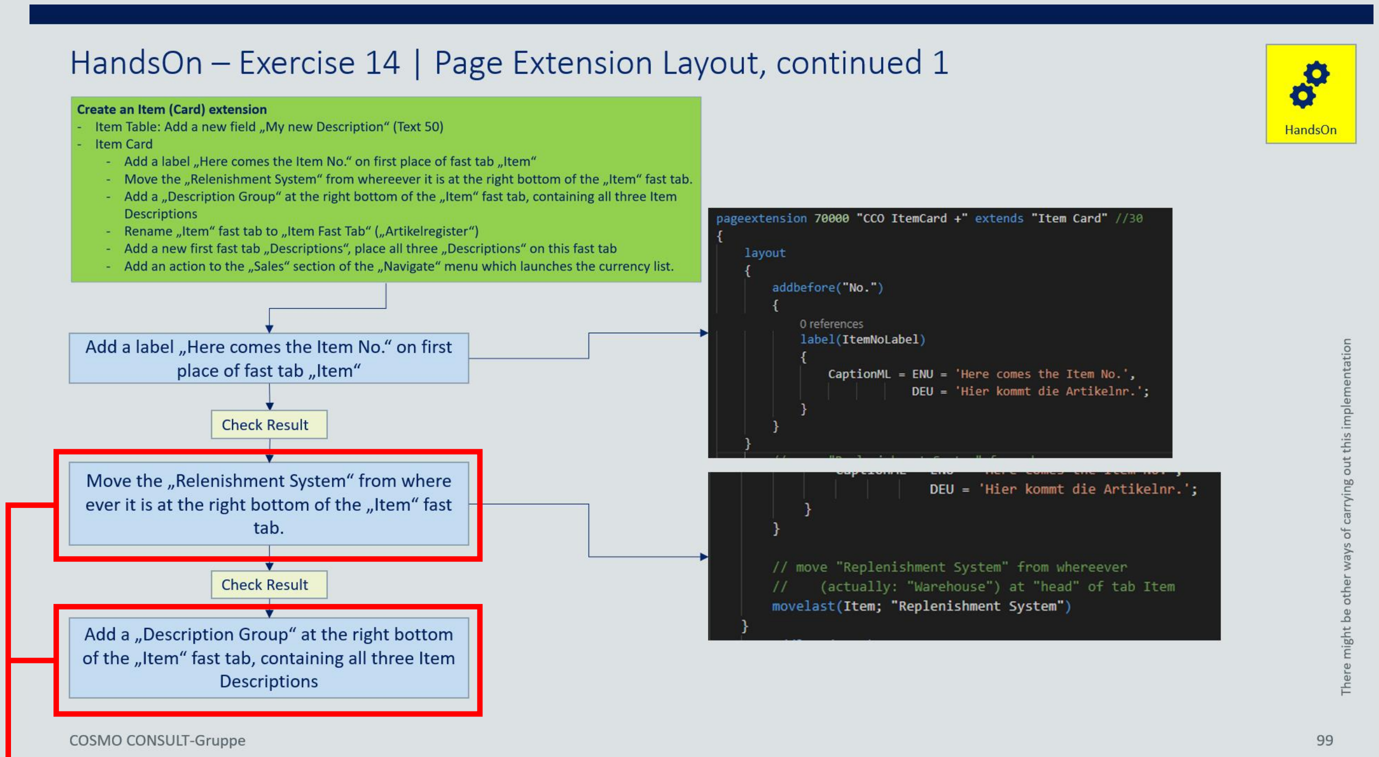
```
modify(Item)
{
    CaptionML = DEU = 'Artikelregister',
                ENU = 'Item Fast Tab';
}

addbefore(Item)
{
    // area(content)
    // {
    0 references
    group(Desc
    {
        Captio

        0 referer
        field(
        {
            Ap
        }
        0 referer
        field(
        {
            Ap
        }
        0 referer
        field(
        {
            Ap
        }
    }
}
```

```
                    ApplicationArea = All;
                }
            }
        }
    }

    actions
    {
        addlast("S&ales")
        {
            0 references
            action(CCOCurrencies)
            {
                CaptionML = ENU = 'Currencies',
                            DEU = 'Währungen';
                RunObject = page Currencies;
            }
        }
    }
}
```

There might be other ways of carrying out this implementation

HandsOn – Exercise 14 | Page Extension Layout, continued 4

HandsOn



HandsOn – Exercise 14 | Page Extension Layout, continued 1

HandsOn

Create an Item (Card) extension
- Item Table: Add a new field „My new Description“ (Text 50)
- Item Card
  - Add a label „Here comes the Item No.“ on first place of fast tab „Item“
  - Move the „Relenishment System“ from whereever it is at the right bottom of the „Item“ fast tab.
  - Add a „Description Group“ at the right bottom of the „Item“ fast tab, containing all three Item Descriptions
  - Rename „Item“ fast tab to „Item Fast Tab“ („Artikelregister“)
  - Add a new first fast tab „Descriptions“, place all three „Descriptions“ on this fast tab
  - Add an action to the „Sales“ section of the „Navigate“ menu which launches the currency list.

Add a label „Here comes the Item No.“ on first place of fast tab „Item“

```
pageextension 70000 "CCO ItemCard +" extends "Item Card" //30
{
    layout
    {
        addbefore("No.")
        {
            0 references
            label(ItemNoLabel)
            {
                CaptionML = ENU = 'Here comes the Item No.',
                            DEU = 'Hier kommt die Artikelnr.';
            }
        }
    }
```

Check Result

Move the „Relenishment System“ from where ever it is at the right bottom of the „Item“ fast tab.

```
                            DEU = 'Hier kommt die Artikelnr.';
            }
        }

        // move "Replenishment System" from whereever
        //    (actually: "Warehouse") at "head" of tab Item
        movelast(Item; "Replenishment System")
    }
```

Check Result

Add a „Description Group“ at the right bottom of the „Item“ fast tab, containing all three Item Descriptions

There might be other ways of carrying out this implementation

COSMO CONSULT-Gruppe 99

Learn
1) how to influence the layout of a page
2) Page extensions are „executed“ from top to bottom … „the last one wins“

HandsOn

HandsOn – Exercise 15

Module Hazard Category Part 2

HandsOn – Exercise 15 | Module Hazard Category Part 2

HandsOn

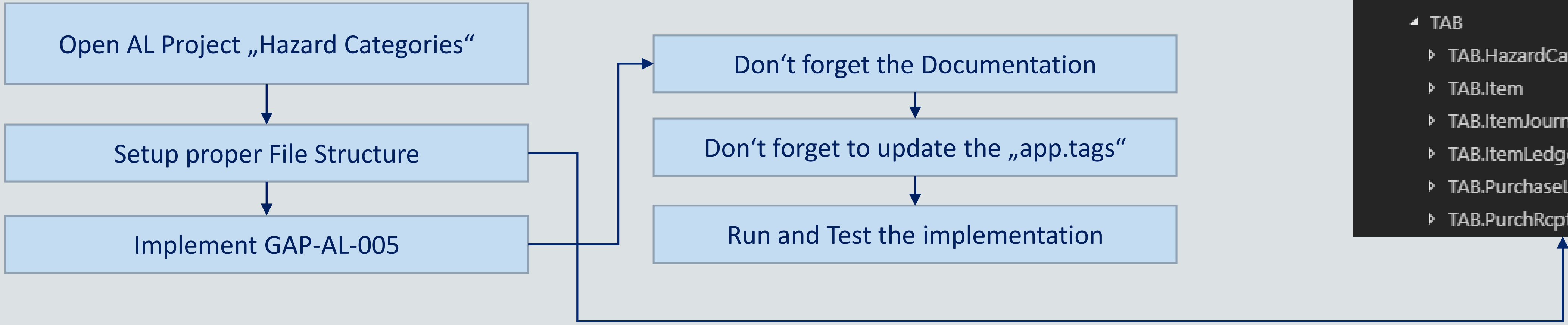
FRD

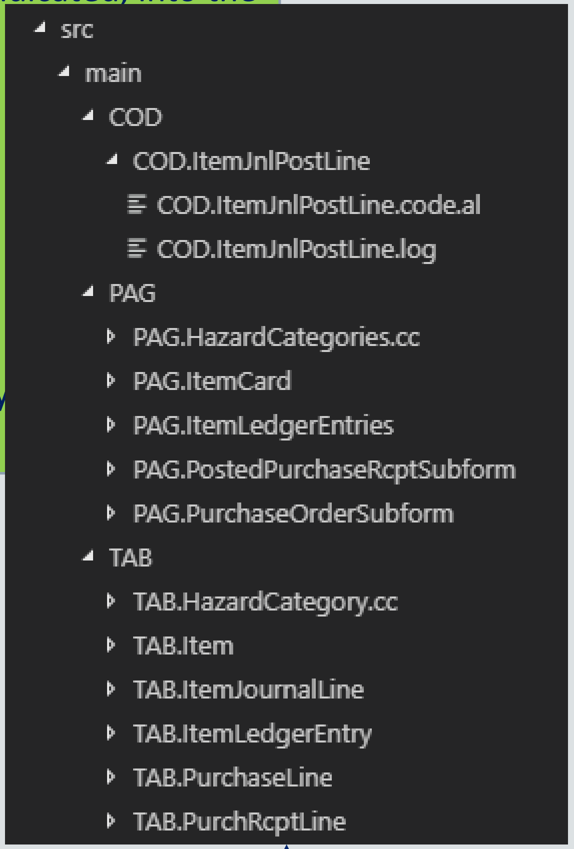
It shall be possible to group Items according to their "Hazard Category". "Hazard Categories" should be freely definable. When an article is purchased via ordering, its "Hazard Category" is to be displayed in the purchasing document and should be changeable there. During the course of item posting, the "Hazard Category" has to be transferred into the "Item Ledger Entries" and, if indicated, into the posted purchase documents.

No other processes are to be respected.
No further details have to be respected concerning, e.g., manipulations of partially posted documents etc..

Remember: New data structure was setup with GAP-AL-004

GAP-AL-005: Hazard Categories – Functionality
Implement new fields in tables "Item", "Purchase Line", "Purch. Rcpt. Line", "Item Journal Line" and "Item Ledger Entry
Implement functionality.


Open AL Project „Hazard Categories“
Setup proper File Structure
Implement GAP-AL-005
Don‘t forget the Documentation
Don‘t forget to update the „app.tags“
Run and Test the implementation



```
◢ src
  ◢ main
    ◢ COD
      ◢ COD.ItemJnlPostLine
          COD.ItemJnlPostLine.code.al
          COD.ItemJnlPostLine.log
    ◢ PAG
      ▷ PAG.HazardCategories.cc
      ▷ PAG.ItemCard
      ▷ PAG.ItemLedgerEntries
      ▷ PAG.PostedPurchaseRcptSubform
      ▷ PAG.PurchaseOrderSubform
    ◢ TAB
      ▷ TAB.HazardCategory.cc
      ▷ TAB.Item
      ▷ TAB.ItemJournalLine
      ▷ TAB.ItemLedgerEntry
      ▷ TAB.PurchaseLine
      ▷ TAB.PurchRcptLine
```

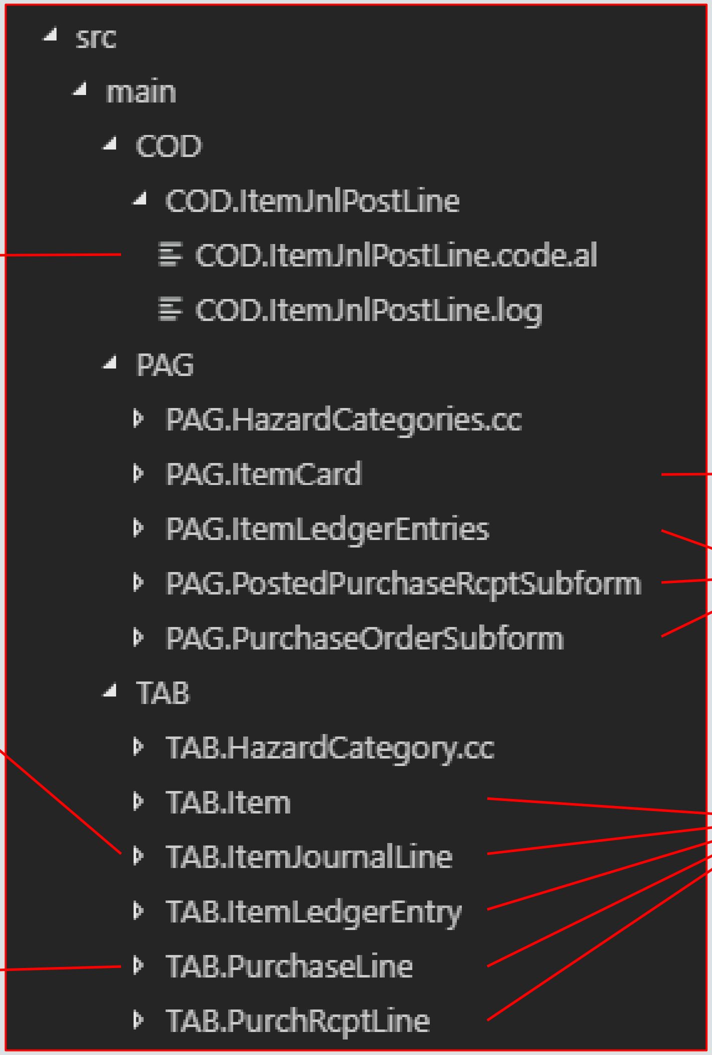
HandsOn – Exercise 15 | Module Hazard Category Part 2, continued

HandsOn

Functionality

Involved Objects

Data Structure & GUI



```
◢ src
  ◢ main
    ◢ COD
      ◢ COD.ItemJnlPostLine
          COD.ItemJnlPostLine.code.al
          COD.ItemJnlPostLine.log
    ◢ PAG
      ▷ PAG.HazardCategories.cc
      ▷ PAG.ItemCard
      ▷ PAG.ItemLedgerEntries
      ▷ PAG.PostedPurchaseRcptSubform
      ▷ PAG.PurchaseOrderSubform
    ◢ TAB
      ▷ TAB.HazardCategory.cc
      ▷ TAB.Item
      ▷ TAB.ItemJournalLine
      ▷ TAB.ItemLedgerEntry
      ▷ TAB.PurchaseLine
      ▷ TAB.PurchRcptLine
```

Posting:
Transfer of Hazard Category from Jnl Line to Entry
(via *OnAfterInitItemLedgEntry*)

Posting:
Transfer of „Hazard Category“ from Purchase Line to „Jnl Line“
(via *OnAfterCopyItemJnlLineFromPurchLine*)

Preassignment & Consistecy Checks:
- Copy „Hazard Category“ from Item if not yet filled
(via *OnAfterAssignItemValues*)
- Ensure „Hazard Category“ only present with Items …

New field „Hazard Category Code“ on Fast Tab „Item“

New field „Hazard Category Code“

New field „Hazard Category Code“ Code[20]

HandsOn

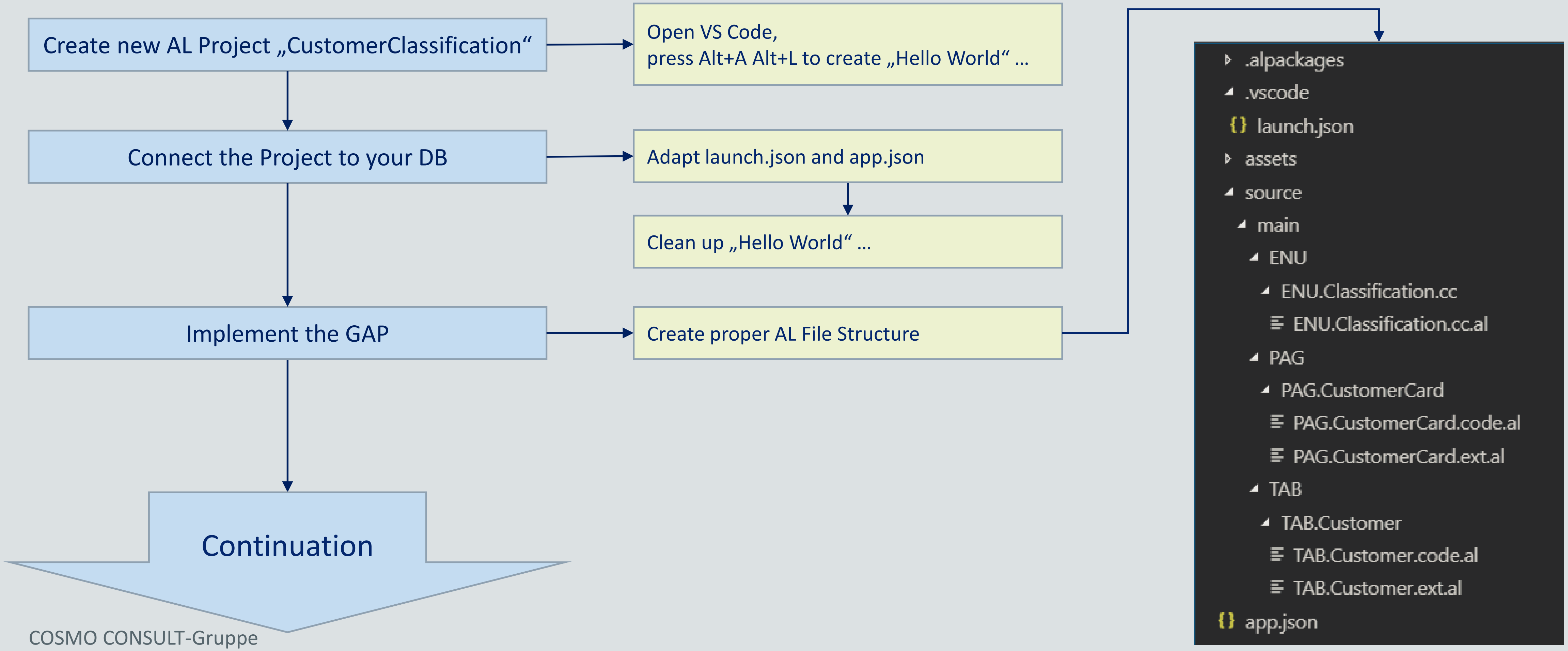
HandsOn – Exercise 16

Customer Classification, Enums

HandsOn – Exercise 16 | Customer Classification, Enums

HandsOn

GAP-AL-006: Customer Classification
It shall be possible to setup a classification of customers by assigning them one of the „values“ „---“, „Bronze Customer“, „Silver Customer“ or „Gold Customer“. An Action „Suggest Salutation“ in the Customer Action Group shall suggest a salutation depending on the particular classification. The Action starts a respective message „Hello <name>“, „Dear bronze <Name>“, „Dear silver <name>“ or „Dear most revered gold <name>“.


Create new AL Project „CustomerClassification“
Open VS Code,
press Alt+A Alt+L to create „Hello World“ …
Connect the Project to your DB
Adapt launch.json and app.json
Clean up „Hello World“ …
Implement the GAP
Create proper AL File Structure
Continuation

```
▹ .alpackages
◢ .vscode
  {} launch.json
▹ assets
◢ source
  ◢ main
    ◢ ENU
      ◢ ENU.Classification.cc
        ENU.Classification.cc.al
    ◢ PAG
      ◢ PAG.CustomerCard
        PAG.CustomerCard.code.al
        PAG.CustomerCard.ext.al
    ◢ TAB
      ◢ TAB.Customer
        TAB.Customer.code.al
        TAB.Customer.ext.al
{} app.json
```

There might be other ways of carrying out this implementation

HandsOn – Exercise 16 | Customer Classification, Enums continued 1

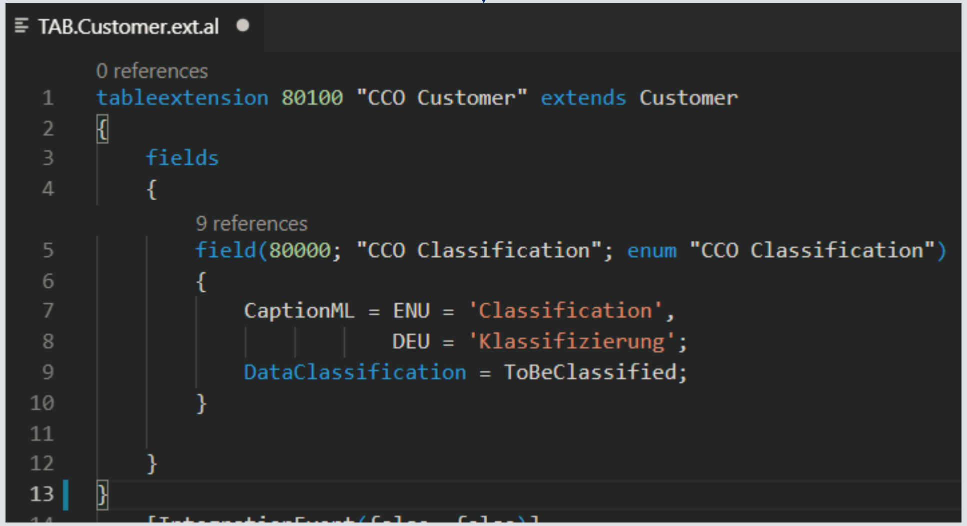
HandsOn

GAP-AL-006: Customer Classification
It shall be possible to setup a classification of customers by assigning them one of the „values“ „---“, „Bronze Customer“, „Silver Customer“ or „Gold Customer“. An Action „Suggest Salutation“ in the Customer Action Group shall suggest a salutation depending on the particular classification. The Action starts a respective message „Hello <name>“, „Dear bronze <Name>“, „Dear silver <name>“ or „Dear most revered gold <name>“.

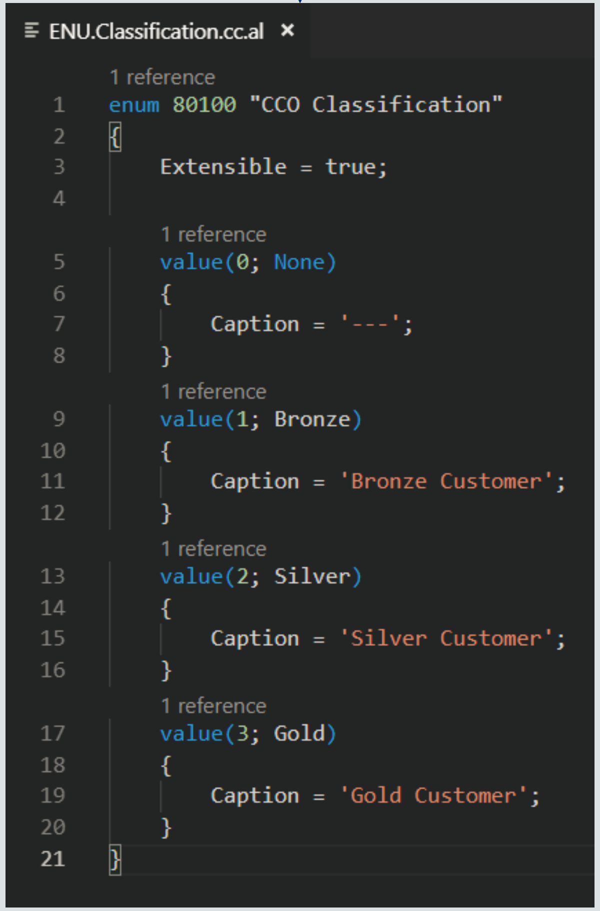
Implement the GAP

- Create proper AL File Structure
- Create Enum Object
- Create Table Extension of Customer Table with new Classification Field
- Show the field on Customer Card
- Add Action functionality
- Don‘t forget about Documentation (logs and tags)

Run / Test the Project


TAB.Customer.ext.al

```
0 references
tableextension 80100 "CCO Customer" extends Customer
{
    fields
    {
        9 references
        field(80000; "CCO Classification"; enum "CCO Classification")
        {
            CaptionML = ENU = 'Classification',
                        DEU = 'Klassifizierung';
            DataClassification = ToBeClassified;
        }

    }
}
    [IntegrationEvent(false, false)]
```


ENU.Classification.cc.al

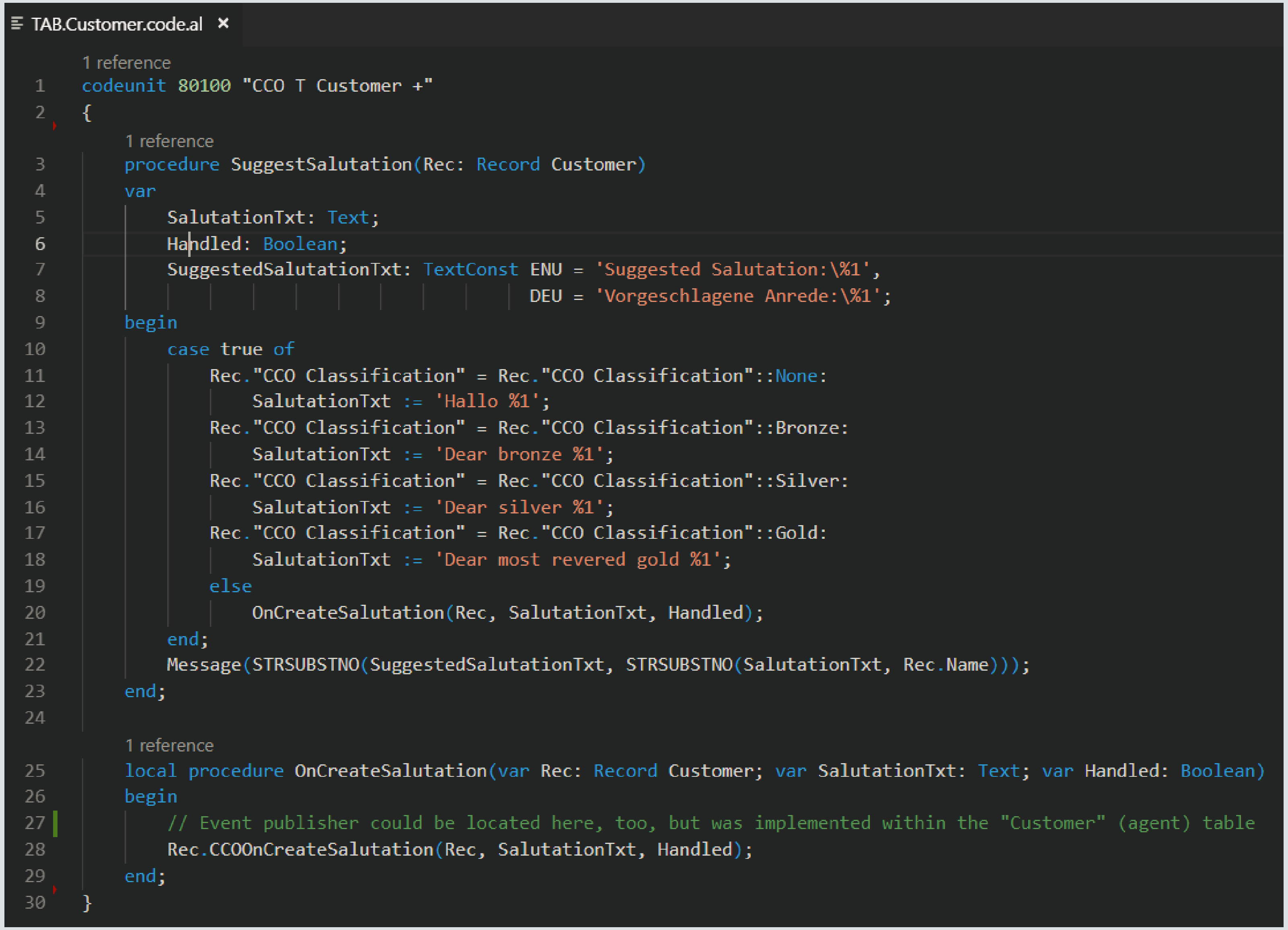
```
1 reference
enum 80100 "CCO Classification"
{
    Extensible = true;

    1 reference
    value(0; None)
    {
        Caption = '---';
    }
    1 reference
    value(1; Bronze)
    {
        Caption = 'Bronze Customer';
    }
    1 reference
    value(2; Silver)
    {
        Caption = 'Silver Customer';
    }
    1 reference
    value(3; Gold)
    {
        Caption = 'Gold Customer';
    }
}
```

Learn Enum Syntax

There might be other ways of carrying out this implementation

HandsOn – Exercise 16 | Customer Classification, Enums continued 2

GAP-AL-006: Customer Classification : Hints



```
1 reference
codeunit 80100 "CCO T Customer +"
{
    1 reference
    procedure SuggestSalutation(Rec: Record Customer)
    var
        SalutationTxt: Text;
        Handled: Boolean;
        SuggestedSalutationTxt: TextConst ENU = 'Suggested Salutation:\%1',
                                          DEU = 'Vorgeschlagene Anrede:\%1';
    begin
        case true of
            Rec."CCO Classification" = Rec."CCO Classification"::None:
                SalutationTxt := 'Hallo %1';
            Rec."CCO Classification" = Rec."CCO Classification"::Bronze:
                SalutationTxt := 'Dear bronze %1';
            Rec."CCO Classification" = Rec."CCO Classification"::Silver:
                SalutationTxt := 'Dear silver %1';
            Rec."CCO Classification" = Rec."CCO Classification"::Gold:
                SalutationTxt := 'Dear most revered gold %1';
            else
                OnCreateSalutation(Rec, SalutationTxt, Handled);
        end;
        Message(STRSUBSTNO(SuggestedSalutationTxt, STRSUBSTNO(SalutationTxt, Rec.Name)));
    end;

    1 reference
    local procedure OnCreateSalutation(var Rec: Record Customer; var SalutationTxt: Text; var Handled: Boolean)
    begin
        // Event publisher could be located here, too, but was implemented within the "Customer" (agent) table
        Rec.CCOOnCreateSalutation(Rec, SalutationTxt, Handled);
    end;
}
```

There might be other ways of carrying out this implementation

HandsOn

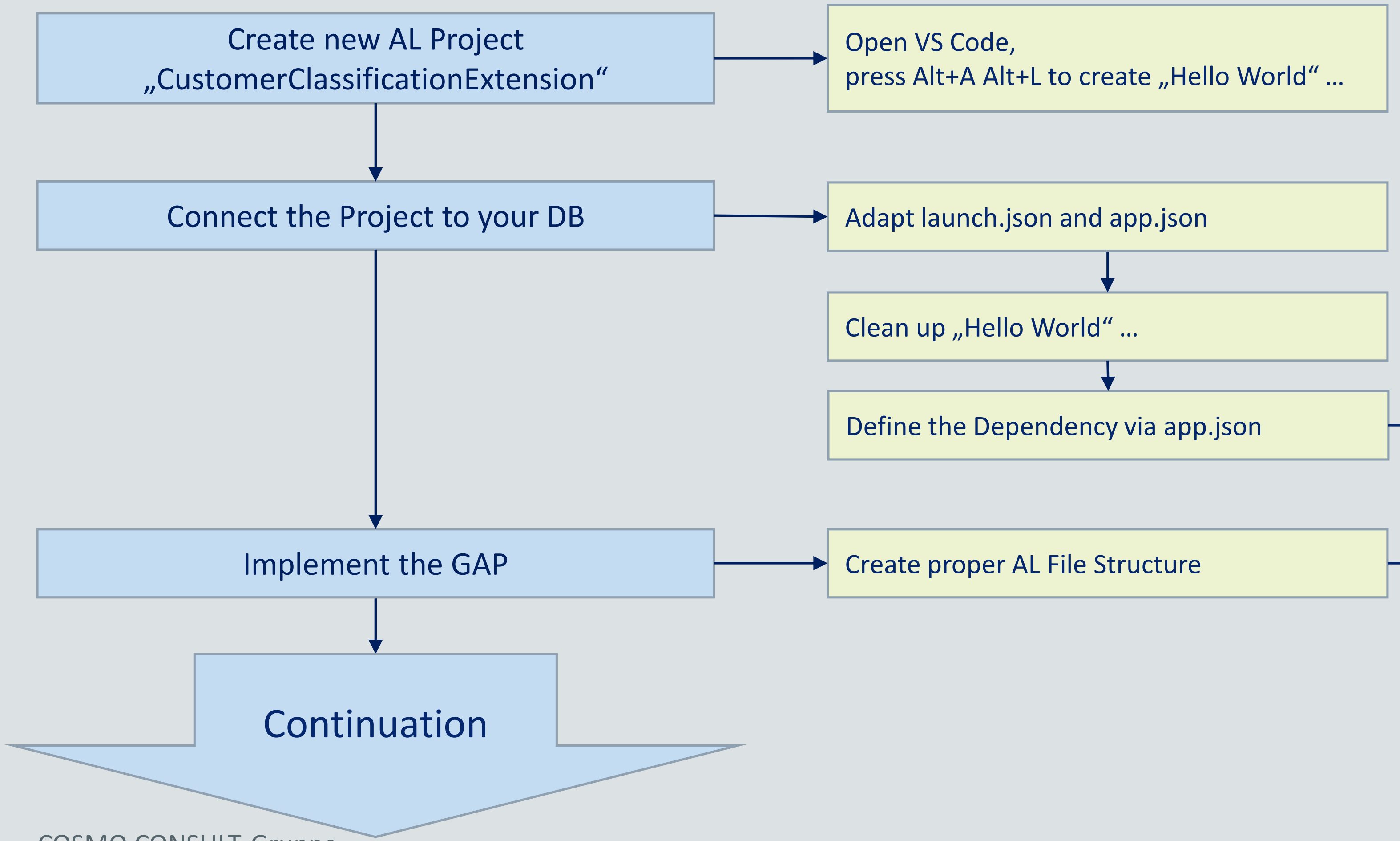
HandsOn – Exercise 17

Extended Customer Classification, Enum Extensions App Dependencies

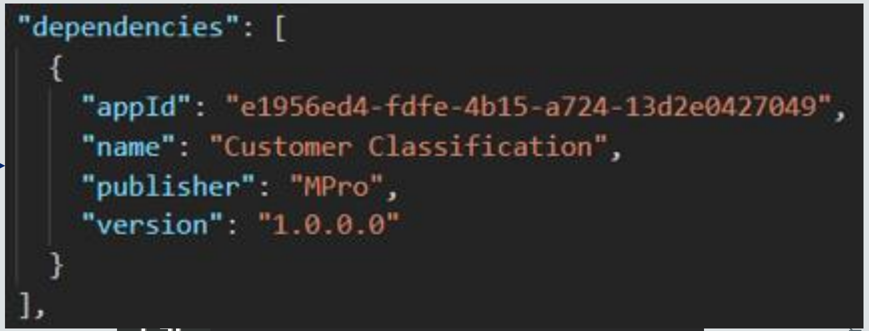
HandsOn – Exercise 17 | Extending Customer Classification, Extended Enums

HandsOn

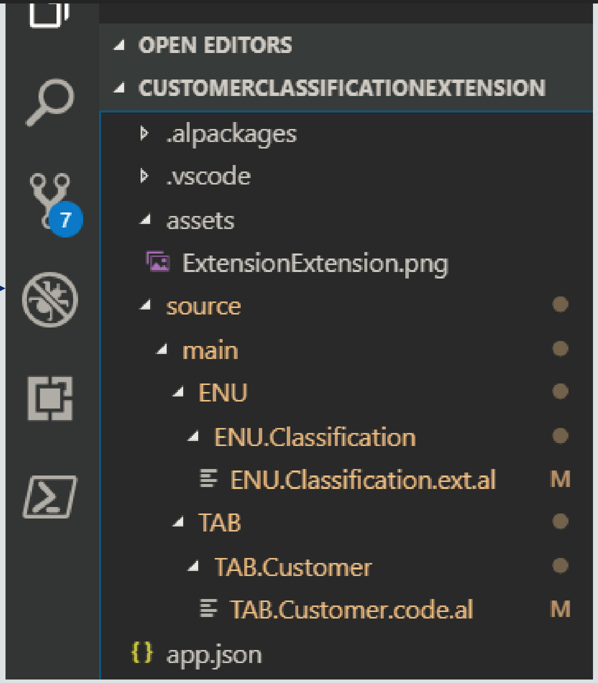
GAP-AL-007: Extended Customer Classification
The Customer Classification system shall be extended by a new category „Platinum Customer“.
The corresponding salutation suggestion shall be „Hallo supreme most revered <name>“



Create new AL Project „CustomerClassificationExtension“
Open VS Code, press Alt+A Alt+L to create „Hello World“ …
Connect the Project to your DB
Adapt launch.json and app.json
Clean up „Hello World“ …
Define the Dependency via app.json
Implement the GAP
Create proper AL File Structure
Continuation



```
"dependencies": [
  {
    "appId": "e1956ed4-fdfe-4b15-a724-13d2e0427049",
    "name": "Customer Classification",
    "publisher": "MPro",
    "version": "1.0.0.0"
  }
],
```



```
OPEN EDITORS
CUSTOMERCLASSIFICATIONEXTENSION
  .alpackages
  .vscode
  assets
    ExtensionExtension.png
  source
    main
      ENU
        ENU.Classification
          ENU.Classification.ext.al      M
      TAB
        TAB.Customer
          TAB.Customer.code.al           M
  app.json
```

There might be other ways of carrying out this imple

HandsOn – Exercise 17 | Extending Customer Classification, Extended Enums

HandsOn

GAP-AL-007: Extended Customer Classification
The Customer Classification system shall be extended by a new category „Platinum Customer“.
The corresponding salutation suggestion shall be „Hallo supreme most revered <name>“

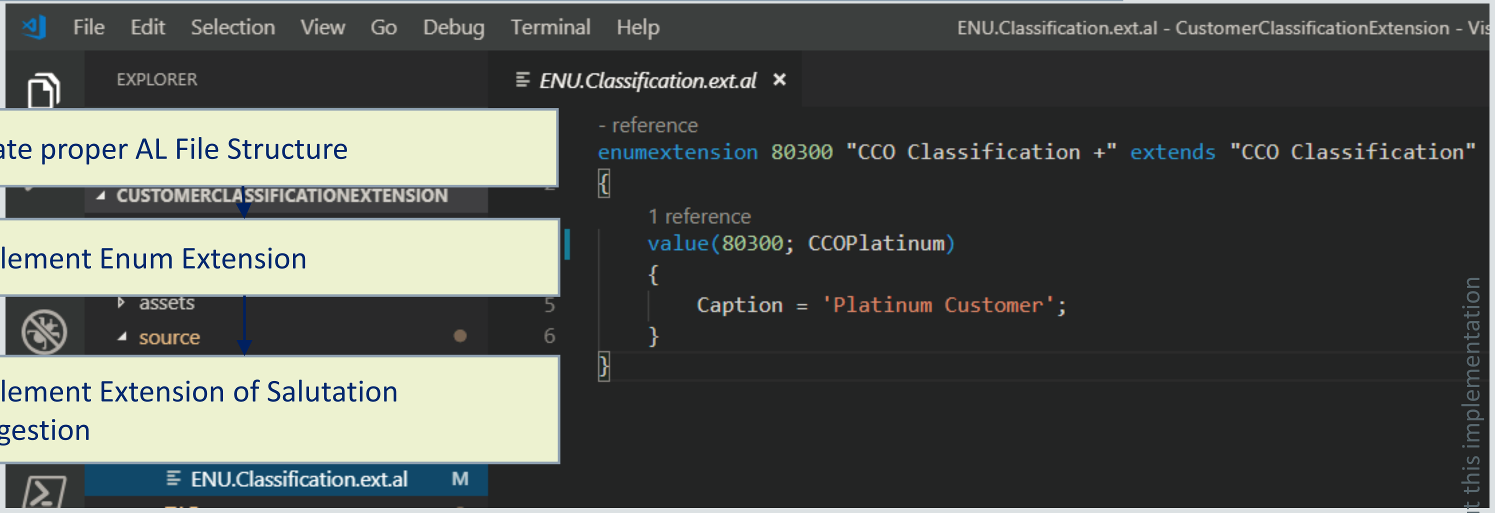
Implement the GAP

Create proper AL File Structure

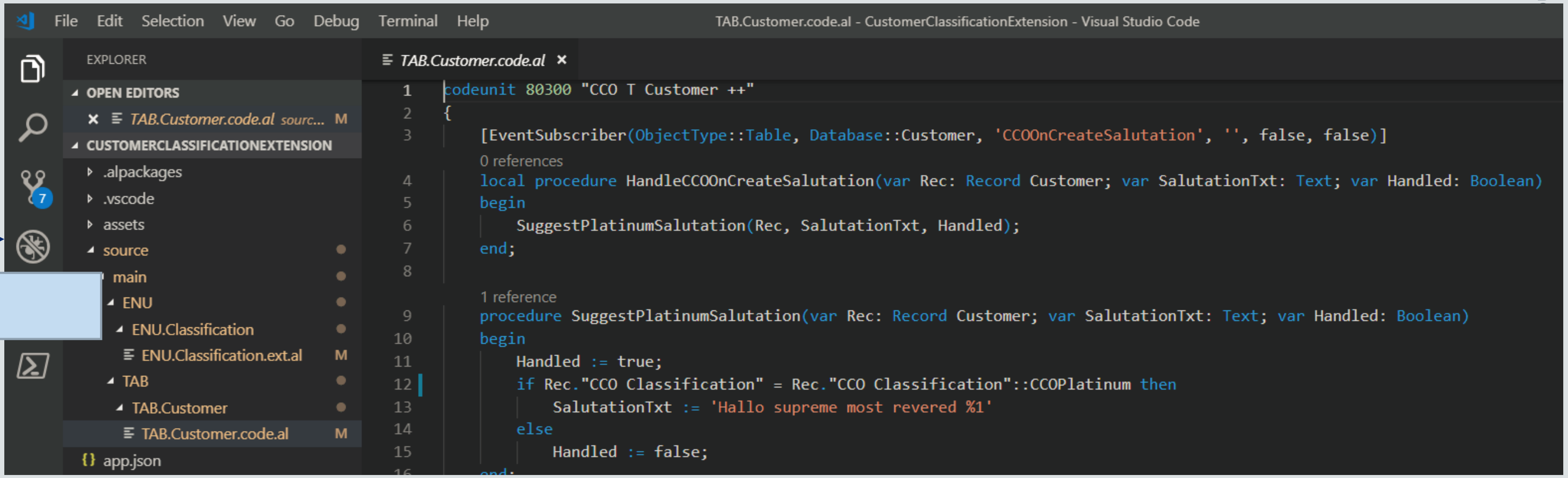
Implement Enum Extension

Implement Extension of Salutation Suggestion

Run / Test the Project


File Edit Selection View Go Debug Terminal Help
ENU.Classification.ext.al - CustomerClassificationExtension - Vis
EXPLORER
CUSTOMERCLASSIFICATIONEXTENSION
assets
source
ENU.Classification.ext.al M
ENU.Classification.ext.al

```
- reference
enumextension 80300 "CCO Classification +" extends "CCO Classification"
{
    1 reference
    value(80300; CCOPlatinum)
    {
        Caption = 'Platinum Customer';
    }
}
```


File Edit Selection View Go Debug Terminal Help
TAB.Customer.code.al - CustomerClassificationExtension - Visual Studio Code
EXPLORER
OPEN EDITORS
TAB.Customer.code.al sourc... M
CUSTOMERCLASSIFICATIONEXTENSION
.alpackages
.vscode
assets
source
main
ENU
ENU.Classification
ENU.Classification.ext.al M
TAB
TAB.Customer
TAB.Customer.code.al M
app.json
TAB.Customer.code.al

```
codeunit 80300 "CCO T Customer ++"
{
    [EventSubscriber(ObjectType::Table, Database::Customer, 'CCOOnCreateSalutation', '', false, false)]
    0 references
    local procedure HandleCCOOnCreateSalutation(var Rec: Record Customer; var SalutationTxt: Text; var Handled: Boolean)
    begin
        SuggestPlatinumSalutation(Rec, SalutationTxt, Handled);
    end;

    1 reference
    procedure SuggestPlatinumSalutation(var Rec: Record Customer; var SalutationTxt: Text; var Handled: Boolean)
    begin
        Handled := true;
        if Rec."CCO Classification" = Rec."CCO Classification"::CCOPlatinum then
            SalutationTxt := 'Hallo supreme most revered %1'
        else
            Handled := false;
```

COSMO CONSULT-Gruppe

HandsOn

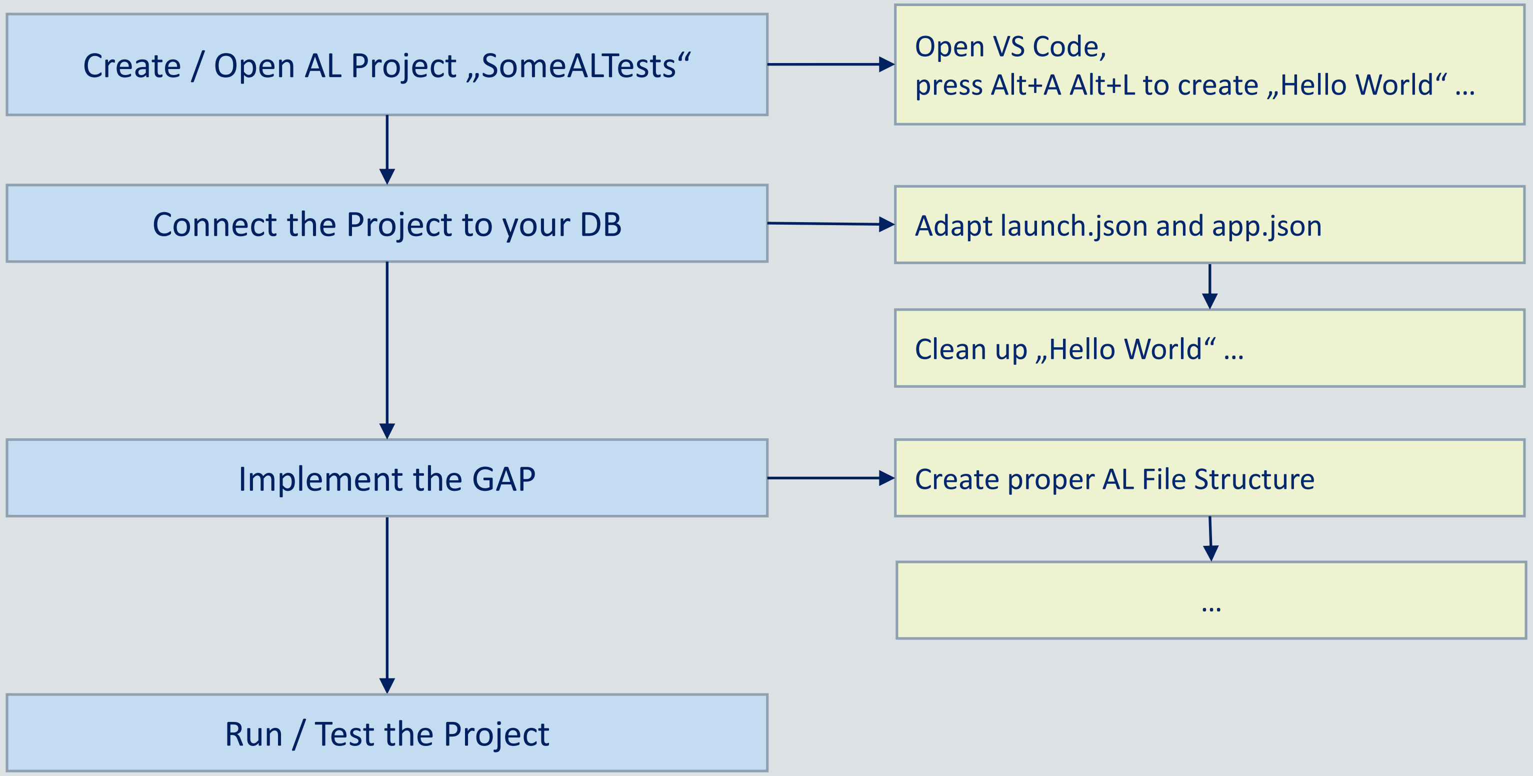
HandsOn – Exercise 18

CU Launcher

HandsOn – Exercise 18 | Tools: CU Launcher

HandsOn

GAP-AL-008: CU Launcher
A functionality is to be implemented, which allows to start particular codeunits via GUI.
Design
- Page „CU Launcher“
  - with control „CU ID to launch“
  - Action „Launch CU“ which launches CU (after proper confirmation)


Create / Open AL Project „SomeALTests“
Open VS Code,
press Alt+A Alt+L to create „Hello World“ …
Connect the Project to your DB
Adapt launch.json and app.json
Clean up „Hello World“ …
Implement the GAP
Create proper AL File Structure
…
Run / Test the Project

There might be other ways of carrying out this implementation