Visual Studio Code and AL Language

Instructions for beginners

If you are attending a Visual Studio Code and AL Language course that assumes pre-existing knowledge of Visual Studio Code and AL Language, or you have never before worked with Visual Studio Code or AL Language, then these instructions will help you with the most basic tasks.

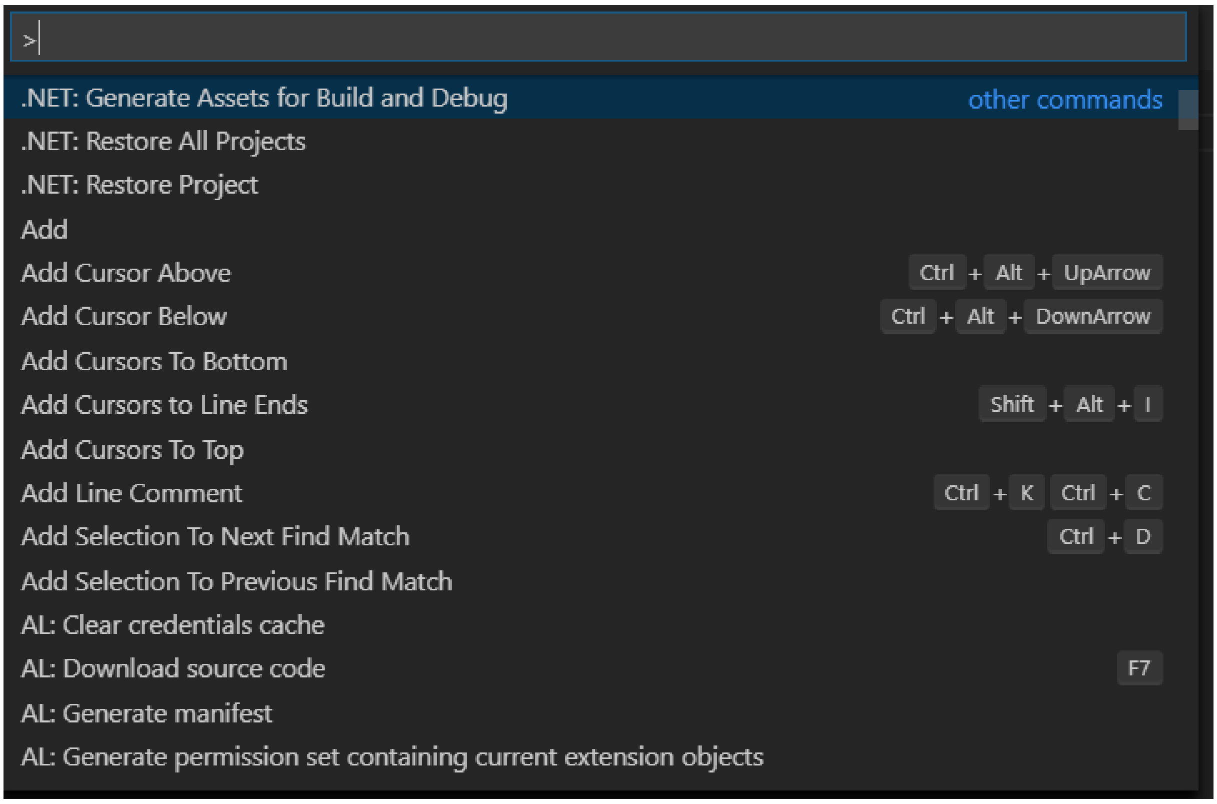
Remember, you always have an instructor in the room during your hands-on workshop. If you need any help, don't hesitate to talk to your instructor.

Contents

Create a new AL workspace

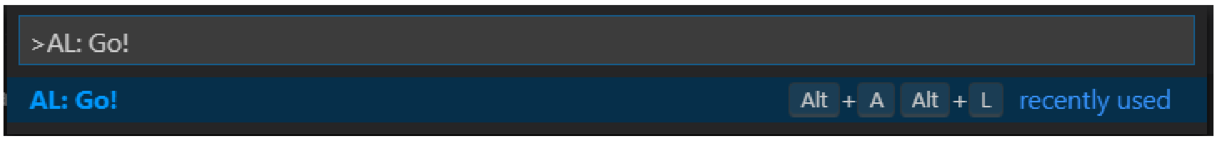
In this exercise you create a new workspace to develop the Classification extension. With this extension, you'll explore the capabilities of the classification prediction functionality of Azure Machine Learning.

1. Start Visual Studio Code.
2. Press Ctrl+Shift+P to access the Command Palete:



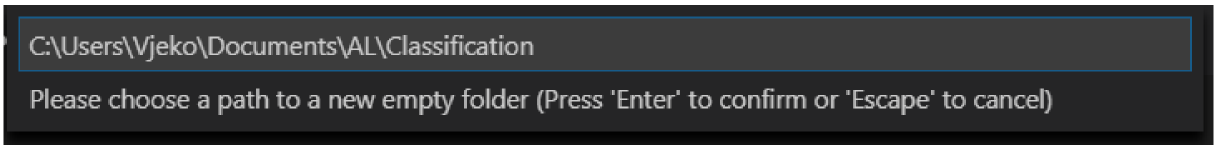
```
>|
.NET: Generate Assets for Build and Debug                          other commands
.NET: Restore All Projects
.NET: Restore Project
Add
Add Cursor Above                                         Ctrl + Alt + UpArrow
Add Cursor Below                                         Ctrl + Alt + DownArrow
Add Cursors To Bottom
Add Cursors to Line Ends                                 Shift + Alt + I
Add Cursors To Top
Add Line Comment                                         Ctrl + K  Ctrl + C
Add Selection To Next Find Match                         Ctrl + D
Add Selection To Previous Find Match
AL: Clear credentials cache
AL: Download source code                                 F7
AL: Generate manifest
AL: Generate permission set containing current extension objects
```

3. In the Command palette prompt, enter "AL: Go!":



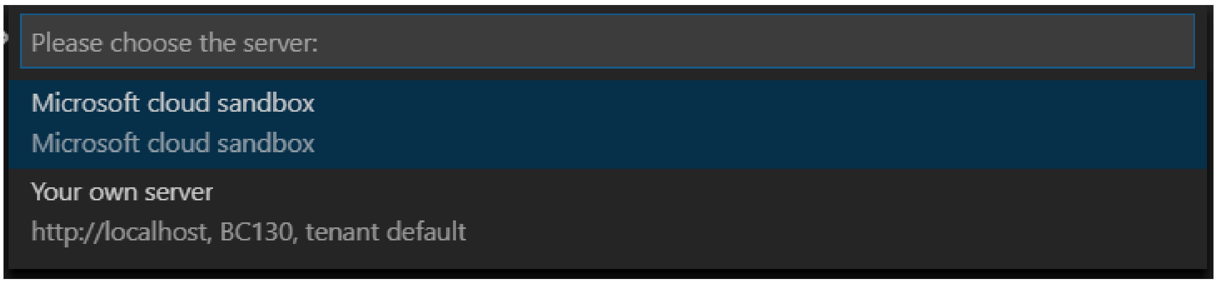
```
>AL: Go!
AL: Go!                          Alt + A  Alt + L   recently used
```

4. Press Enter (or click on the command) to execute the command.
5. Change the name of the workspace by replacing the last part of the suggested path with the name of your choosing. For example, to call your workspace "Classification", change the suggested path like this:.



```
C:\Users\Vjeko\Documents\AL\Classification
Please choose a path to a new empty folder (Press 'Enter' to confirm or 'Escape' to cancel)
```

   Very important: If you click anywhere outside the prompt, or if you switch to another application (for example, to Windows Explorer, to copy a path that you want to paste in this prompt) the prompt will disappear, and you'll have to start over. This applies to all prompts in Visual Studio Code. Therefore, if you want to store your workspace in an entirely different path, you'll first have to copy that path into clipboard, and then start the "AL: Go!" command, or you'll have to type the entire path into the prompt.
6. Choose your preferred server launch configuration:


Please choose the server:

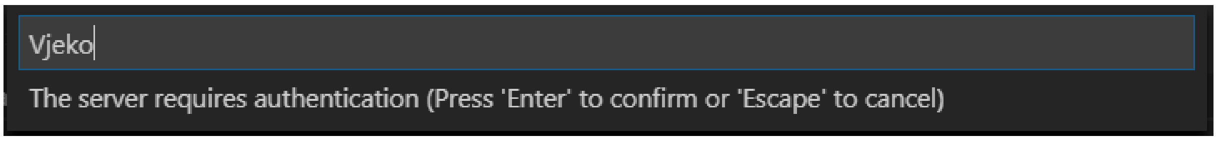
Microsoft cloud sandbox
Microsoft cloud sandbox

Your own server
http://localhost, BC130, tenant default

7. Wait a few seconds until Visual Studio Code prepares your workspace. After workspace creation finishes, Visual Studio Code opens your new workspace.
8. If you chose **Microsoft cloud sandbox**, then move to step 15.

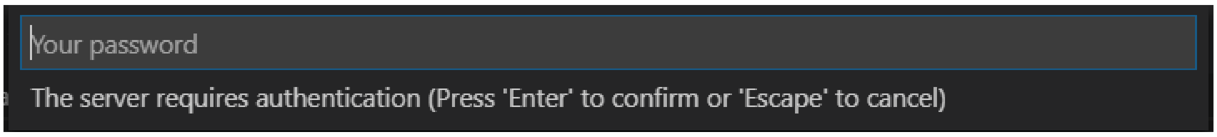
Configuring Your own server

9. If you chose **Your own server**, Visual Studio Code prompts you for your username.
10. If you are using Windows authentication with your local instance of Business Central, just press Esc on your keyboard, and then proceed at step 13.
11. Enter your username:


Vjeko

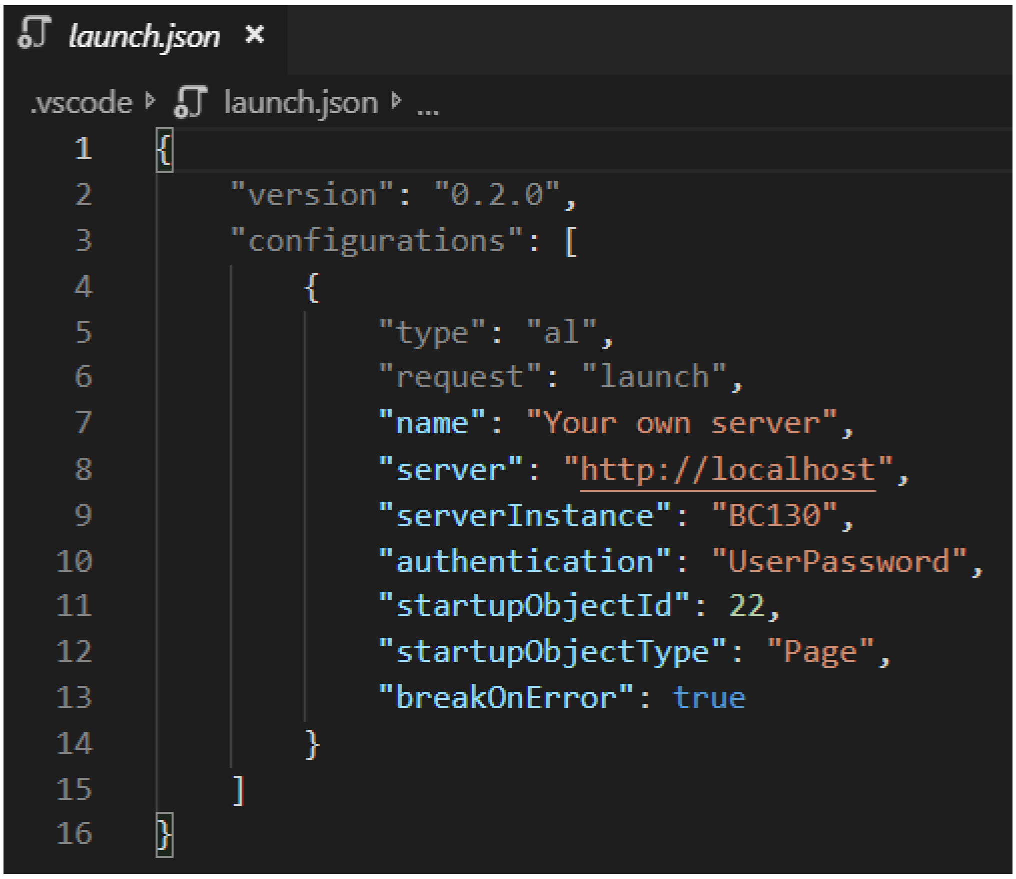
The server requires authentication (Press 'Enter' to confirm or 'Escape' to cancel)

12. Visual Studio Code prompts you for your password. Enter your password:


Your password

The server requires authentication (Press 'Enter' to confirm or 'Escape' to cancel)

13. After you finish entering username and password, Visual Studio Code automatically opens the launch.json file.

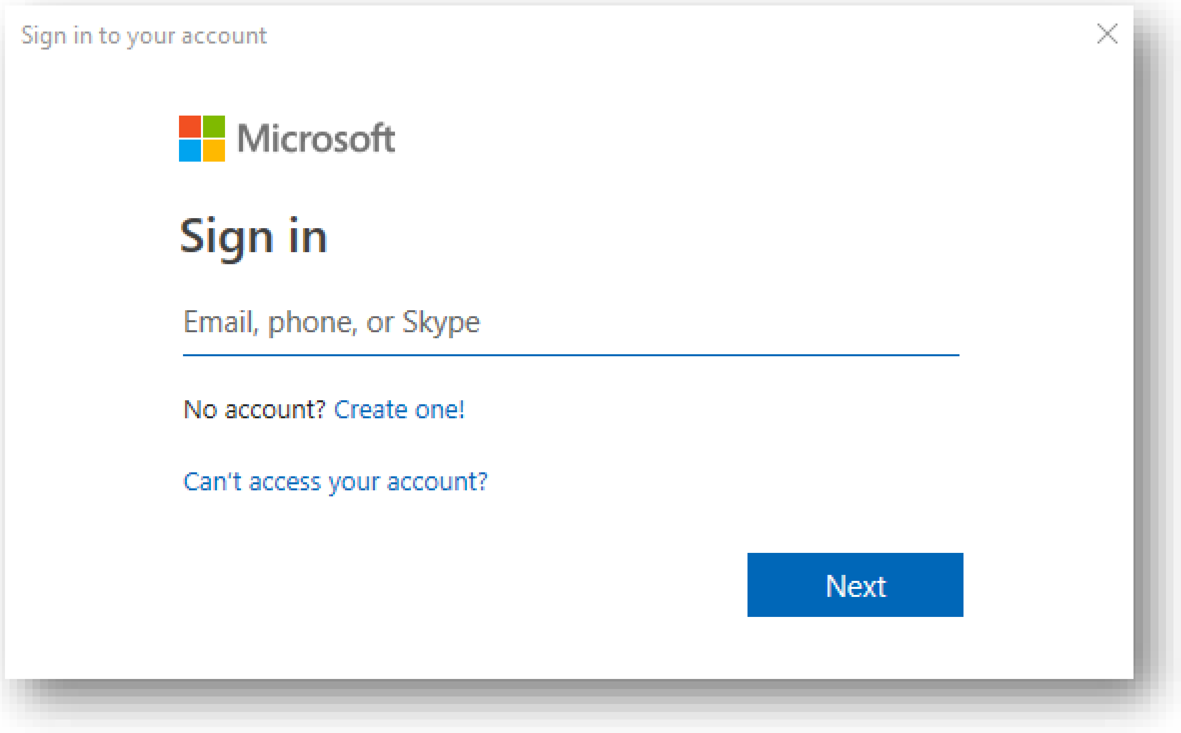


```
{
    "version": "0.2.0",
    "configurations": [
        {
            "type": "al",
            "request": "launch",
            "name": "Your own server",
            "server": "http://localhost",
            "serverInstance": "BC130",
            "authentication": "UserPassword",
            "startupObjectId": 22,
            "startupObjectType": "Page",
            "breakOnError": true
        }
    ]
}
```

14. Modify the “server” and “serverInstance” properties to match your own configuration. If you are using an Azure virtual machine sandbox, then you will find this information in your landing page. If you are unsure what to enter here, ask your instructor.

Configuring Microsoft cloud sandbox

15. If you chose Microsoft cloud sandbox, Visual Studio Code opens the sign-in dialog for your Business Central.


Sign in to your account

Microsoft

Sign in

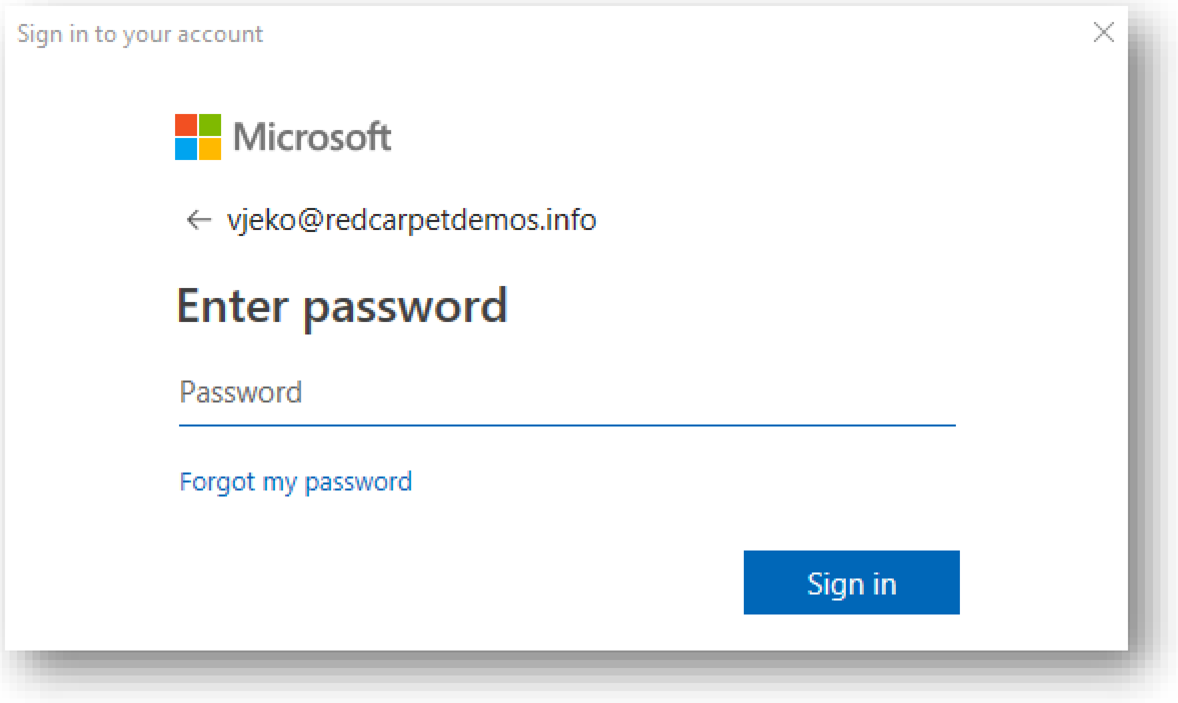
Email, phone, or Skype

No account? Create one!

Can't access your account?

Next

16. Enter your e-mail address, and then click **Next**. You will be prompted to enter password:


Sign in to your account

Microsoft

vjeko@redcarpetdemos.info

Enter password

Password

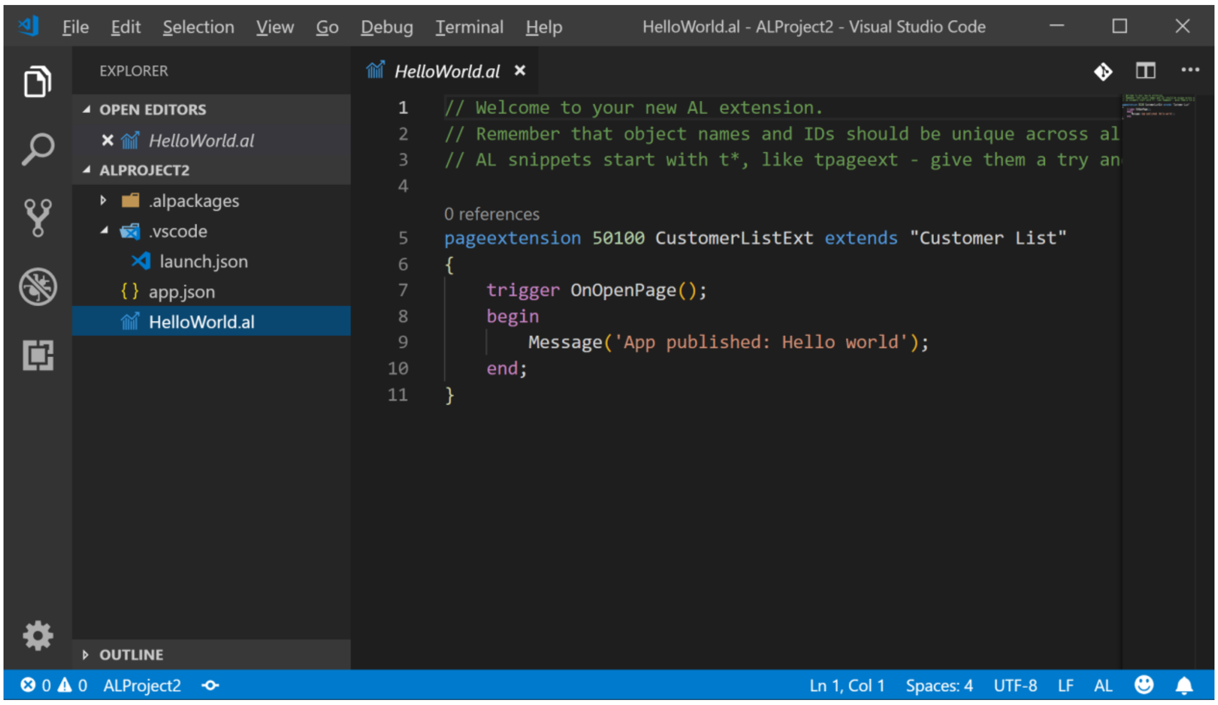
Forgot my password

Sign in

17. Enter your password, and then click **Sign in**.
18. If you have successfully signed in, the Sign-in dialog will automatically close.

Cleaning up your workspace

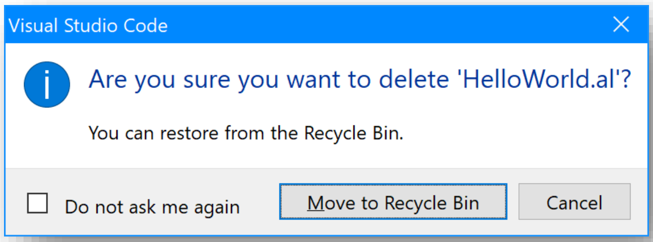
Your demo workspace contains a demo file called “HelloWorld.al”. This file contains a demo page extension object that shows a hello-world type of message when the application starts. You don’t need this file.


File Edit Selection View Go Debug Terminal Help
HelloWorld.al - ALProject2 - Visual Studio Code
EXPLORER
OPEN EDITORS
HelloWorld.al
ALPROJECT2
.alpackages
.vscode
launch.json
app.json
HelloWorld.al
OUTLINE
ALProject2
Ln 1, Col 1 Spaces: 4 UTF-8 LF AL

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

0 references
pageextension 50100 CustomerListExt extends "Customer List"
{
    trigger OnOpenPage();
    begin
        Message('App published: Hello world');
    end;
}
```

19. In the root of your workspace, right click the HelloWorld.al file and then click **Delete**. A confirmation dialog will ask you if you are sure:



Visual Studio Code

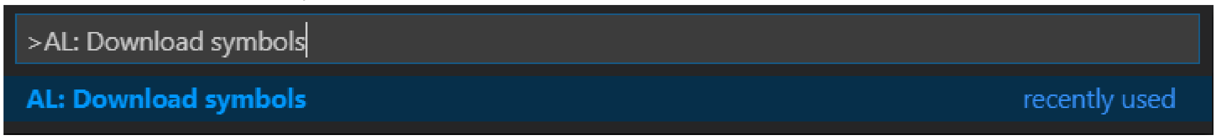
Are you sure you want to delete 'HelloWorld.al'?

You can restore from the Recycle Bin.

20. If you never want to be alerted about deleting files again, select the **Do not ask me again** checkbox.
21. Click **Move to Recycle Bin**.
    Hint: Visual Studio Code always moves files to Recycle Bin, never deletes them from disk. If you accidentally delete a file, you can always restore it from Recycle Bin.

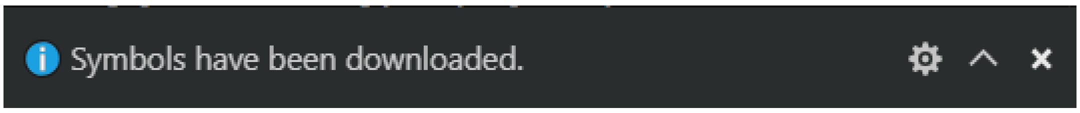
Finalizing configuration

22. Press Ctrl+Shift+P to access the Command Palette again.
23. Run the “AL: Download symbols” command:



```
>AL: Download symbols
```

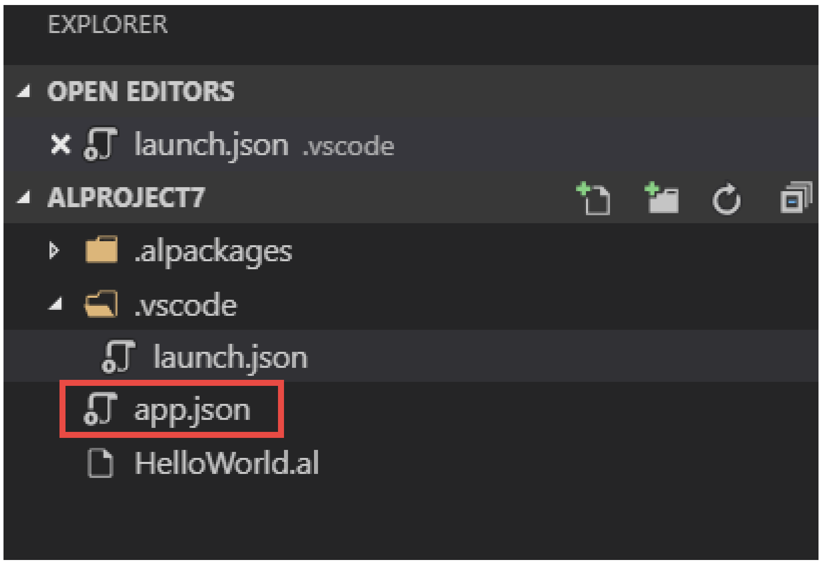
24. If you have successfully entered your credentials earlier, after a short while, Visual Studio will show this notification:


Symbols have been downloaded.

Important: You must resolve any issues you might have at this point. Otherwise, you will not be able to continue using Visual Studio Code and AL Language to develop extensions for Business Central. If you cannot resolve the issues yourself, then ask your instructor for help.

Configure your app manifest

Every extension written in AL Language must contain the app manifest file. It's the file named **app.json** that resides in the root of your workspace.



While in most situations you won't have to access or modify this file during workshops, you need to understand what this file is and how to use it.

This file defines the metadata configuration for your app (extension), and is used by Business Central when publishing and installing new extensions. Proper configuration of this file ensures smooth development and publishing process.

1. Press Ctrl+Shift+E to access the **Explorer** pane.
2. In the Explorer pane, click app.json to open it in the editor.
3. Define the "name" property, enter your name. This is the name that your extension will have when deployed to Business Central.
4. Define the "publisher" property. This is the publisher name under which you will be publishing extensions, and is especially important for the AppSource publishing. Together, the publisher and the name properties will become parts of the extension output file name during the building process.
   This is an example of the configured file:

```json
{
  "id": "4e05d8a8-348d-4cad-aeab-10e2225d77e1",
  "name": "Extension demo",
  "publisher": "Vjeko.com",
  "brief": "",
  "description": "",
  "version": "1.0.0.0",
  "privacyStatement": "",
  "EULA": "",
  "help": "",
  "url": "",
  "logo": "",
  "capabilities": [],
  "dependencies": [],
  "screenshots": [],
  "platform": "13.0.0.0",
  "application": "13.0.0.0",
  "idRange": {
    "from": 50100,
    "to": 50149
  },
  "runtime": "2.3"
}
```

5. Do not change the “id” property. This is the most important value for identifying your extension. If you change your guid and you have already deployed your extension to a Business Central instance, if you deploy a new version with a changed id, Business Central will recognize it as a new extension, rather than an update of the old one.
6. If you want to change the “id” for whatever reason, do not randomly change the value of the “id” by typing arbitrary numbers and letters. If you have to change the “id”, and you know what you are doing, use a webpage (such as https://www.guidgen.com/) or another tool that can create genuine guids for you.
7. Sometimes, you may want to change the “idRange” property to indicate the valid range that your extension works with. This is specially important during development of extensions that you intend to publish on the AppSource. Sometimes, during development, you may want to change your range, to limit the likelihood of causing a conflict of another extension you already deployed on the same development server. If you want to modify the range, change the “from” and “to” properties of the “idRange” object:

```
  "idRange": {
    "from": 50020,
    "to": 50029
  },
```

Visual Studio Code and AL Language basics

There are a lot of basic tasks in Visual Studio Code, and the hands-on scripts will assume that you are familiar with these tasks.

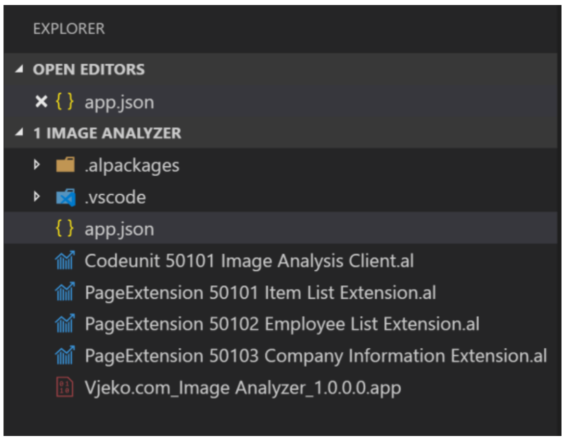
If this is your first time using Visual Studio Code and AL Language, then these instructions should help.

Creating a new file

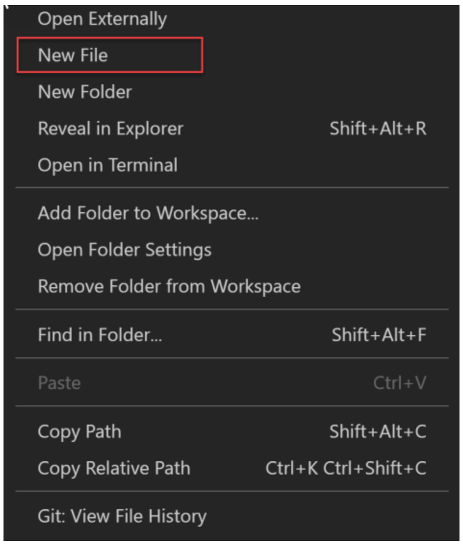
There are several ways to create a new file in Visual Studio Code. For example, you could start by pressing Ctrl+N on your keyboard – this opens a new editor tab and allows you to start writing code immediately. However, when you create a file this way, you must always save it first, and then assign it the correct extension, so that Visual Studio Code can detect which language to use to provide syntax checking and highlighting rules. Even though you can always change the language mode later, it is much simpler to create a file with a specific extension directly.

Whenever the hands-on instructions ask you to create a new file (and tell you which file to create), this is the simplest way:

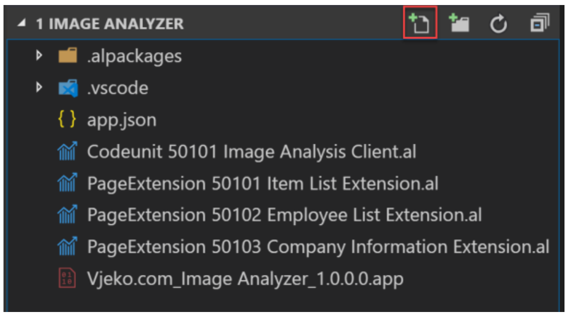
1. If you are not in the Explorer pane of Visual Studio Code, press Ctrl+Shift+E to access the Explorer pan. This is the explorer pane:


EXPLORER
OPEN EDITORS
app.json
1 IMAGE ANALYZER
.alpackages
.vscode
app.json
Codeunit 50101 Image Analysis Client.al
PageExtension 50101 Item List Extension.al
PageExtension 50102 Employee List Extension.al
PageExtension 50103 Company Information Extension.al
Vjeko.com_Image Analyzer_1.0.0.0.app

2. Right-click on an empty space anywhere in the Explorer area, and then click **New file** in the popup menu.

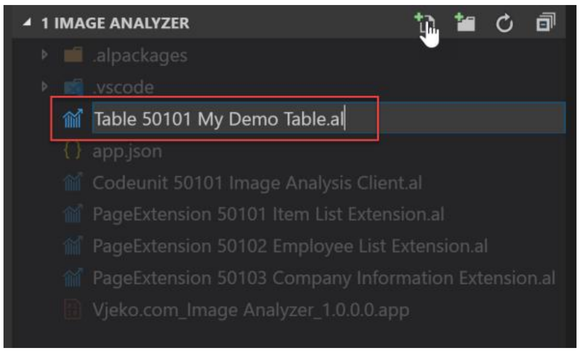

Open Externally
New File
New Folder
Reveal in Explorer Shift+Alt+R
Open in Terminal
Add Folder to Workspace...
Open Folder Settings
Remove Folder from Workspace
Find in Folder... Shift+Alt+F
Paste Ctrl+V
Copy Path Shift+Alt+C
Copy Relative Path Ctrl+K Ctrl+Shift+C
Git: View File History

If you want to create a new file in an existing subfolder in your workspace, then just right-click that subfolder's name.
Alternatively, when you are hovering the explorer area with your mouse, the context icons appear to the right of your workspace name. Click the **New file** icon:



```
1 IMAGE ANALYZER
  .alpackages
  .vscode
  app.json
  Codeunit 50101 Image Analysis Client.al
  PageExtension 50101 Item List Extension.al
  PageExtension 50102 Employee List Extension.al
  PageExtension 50103 Company Information Extension.al
  Vjeko.com_Image Analyzer_1.0.0.0.app
```

3. When you clicked New file, the file name box appears in the Explorer area, with the cursor inside the box. Type the file name and extension (the extension is very important!), for example:



```
1 IMAGE ANALYZER
  .alpackages
  .vscode
  Table 50101 My Demo Table.al
  app.json
  Codeunit 50101 Image Analysis Client.al
  PageExtension 50101 Item List Extension.al
  PageExtension 50102 Employee List Extension.al
  PageExtension 50103 Company Information Extension.al
  Vjeko.com_Image Analyzer_1.0.0.0.app
```

4. When you press enter, the file automatically opens in a new editor tab and allows you to start writing code.

Building, deploying, and running the extension

When you are done writing code, or simply when you want to test and see how the stuff you have written actually behaves, you will want to build, deploy, and run the extension. Even though this sounds like a lot of steps, it's actually a very simple command.

To build, deploy and run your extension, simply press Ctrl+F5 on your keyboard.

If you also want to debug your extension while it's executing, press F5 instead. Keep in mind that unless you really want to use debugger, it is much better to run without debugging (Ctrl+F5).