BC/NAV TECH DAYS 2022

mibuso.com

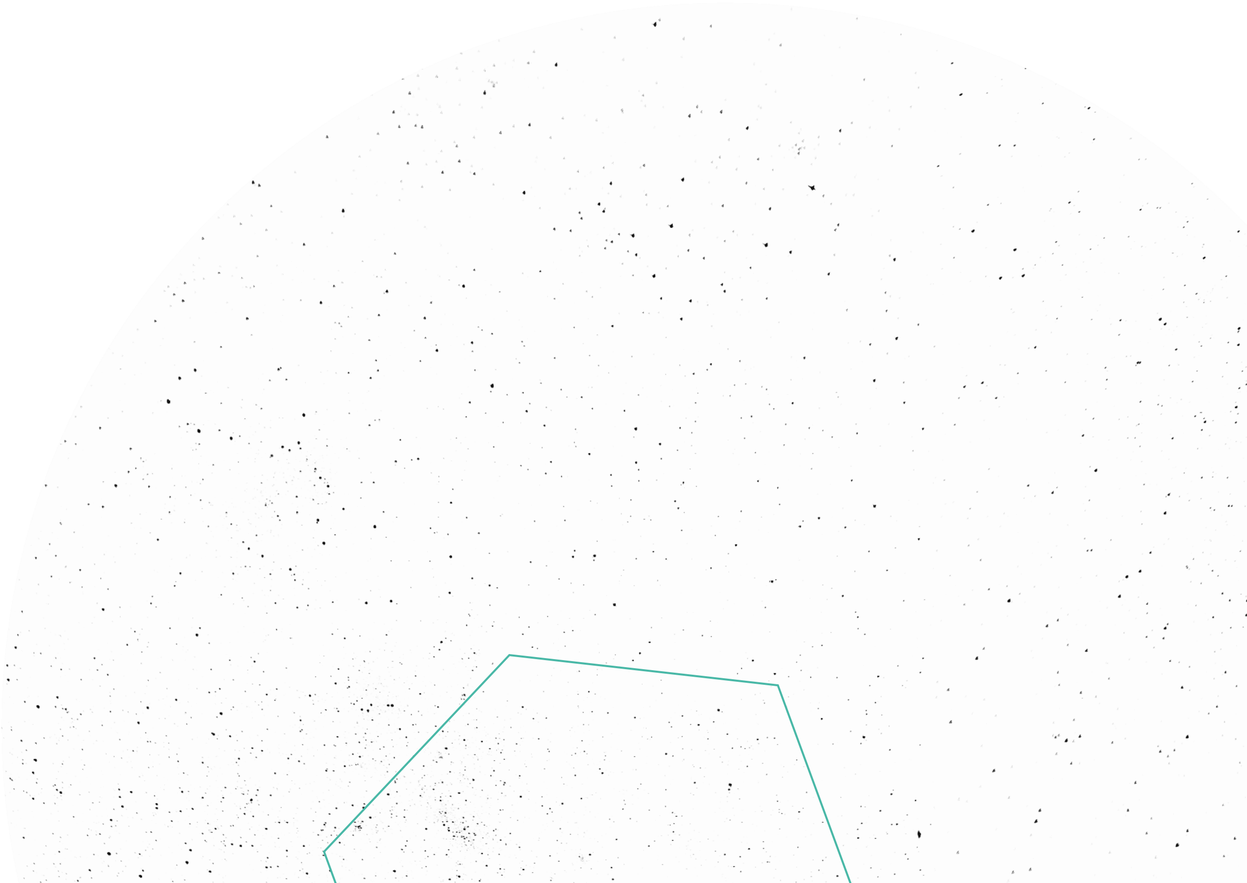
Customizing the developer experience

Kalman Beres, Nikolay Dobrev
MICROSOFT

www.bctechdays.com

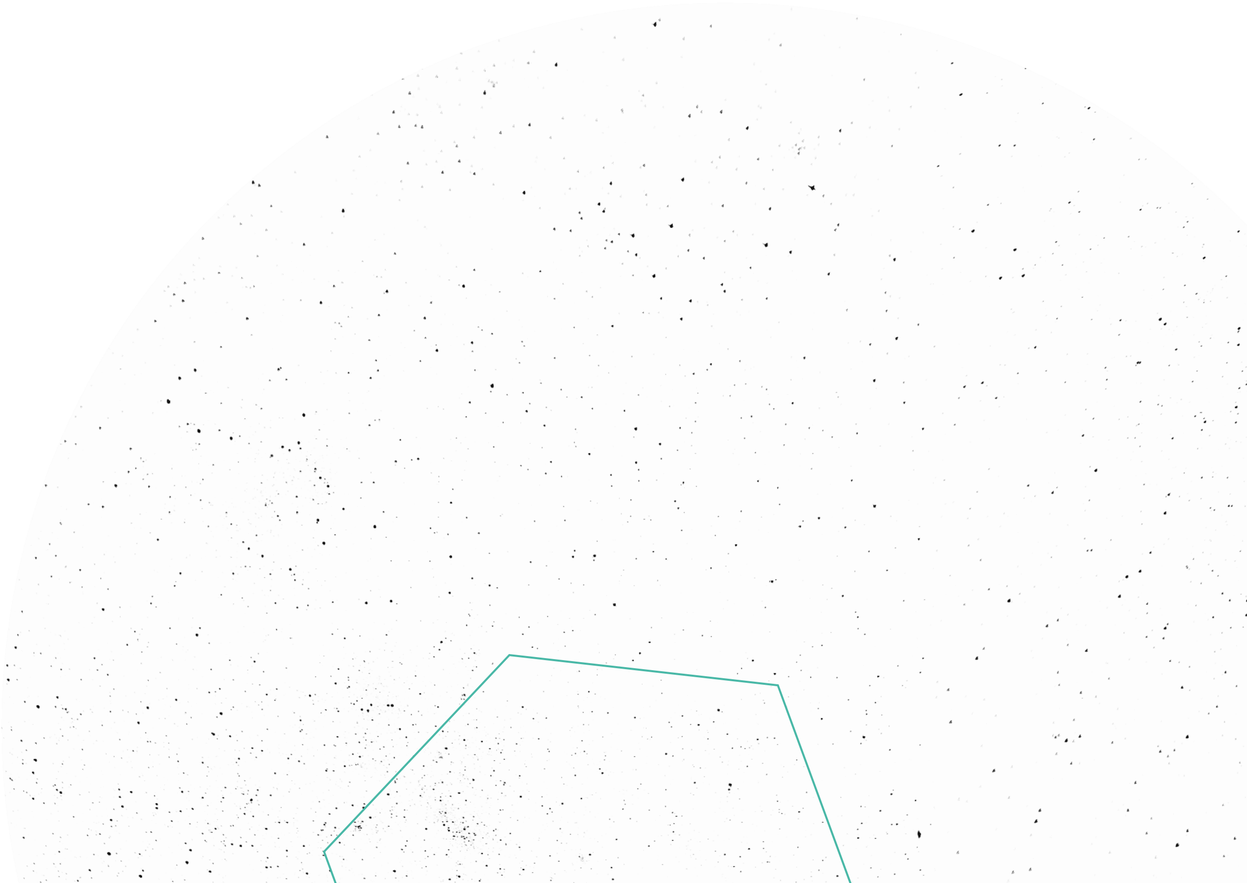
Agenda

- AL Settings
- Workspaces
- Dependency publishing
- Rapid Application Development (RAD)

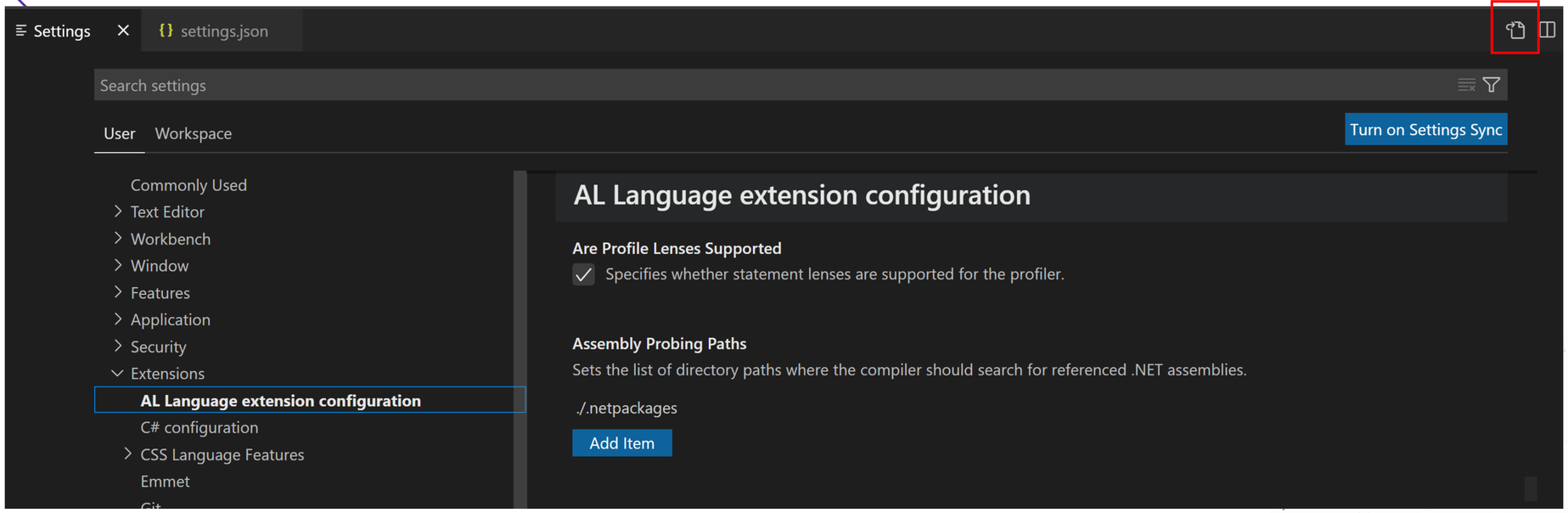


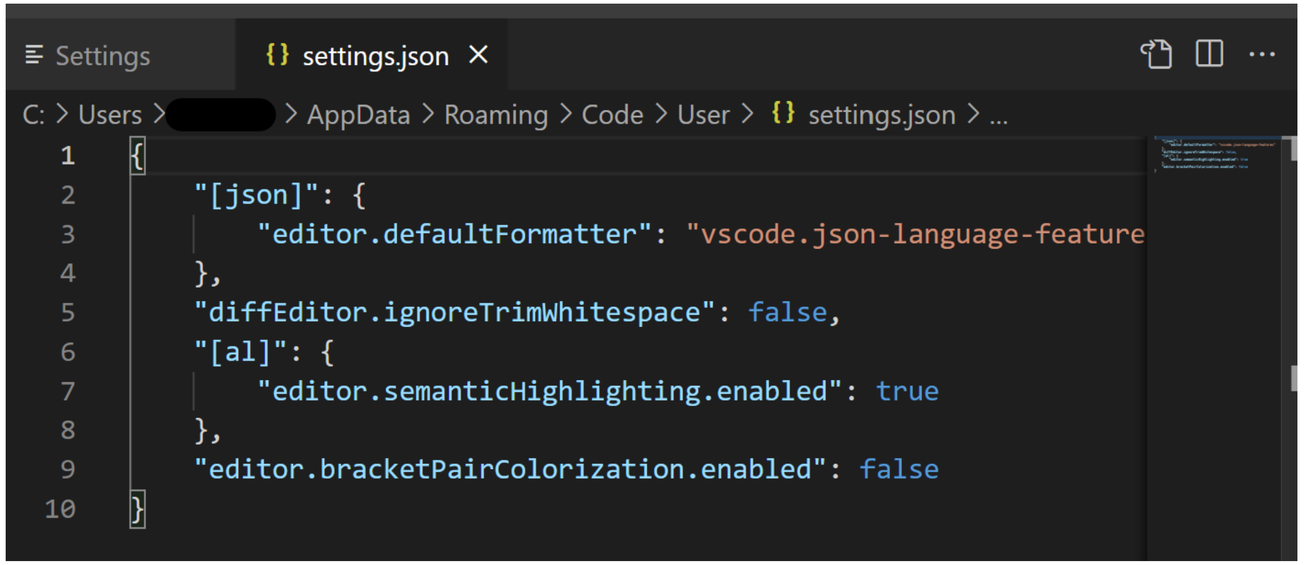
Agenda

- **AL Settings**
- Workspaces
- Dependency publishing
- Rapid Application Development (RAD)



AL Settings

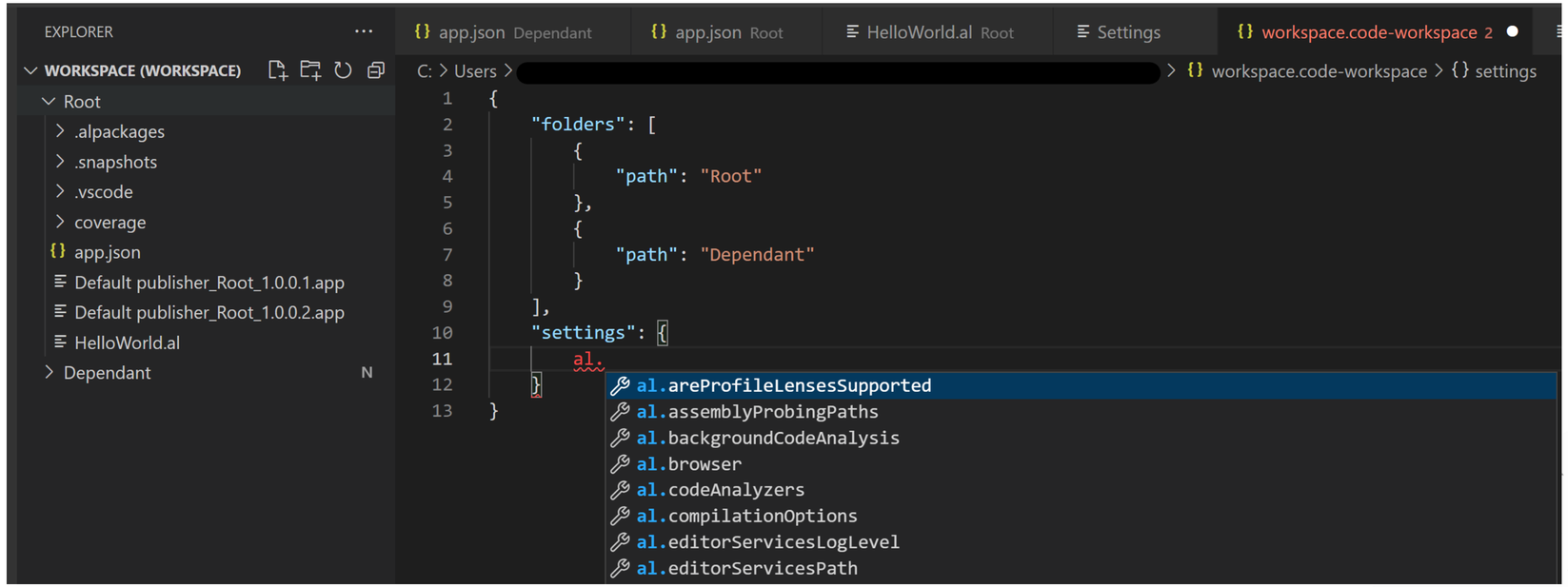




[Visual Studio Code User and Workspace Settings]

- ***User/Global settings*** *- Settings that apply globally to any instance of VS Code you open.*
- **Workspace Settings** - Settings stored inside your workspace and only apply when the workspace is opened. Overrides User settings.
- **Project folder Settings** - Apply to a specific folder, it overrides User/Global settings.

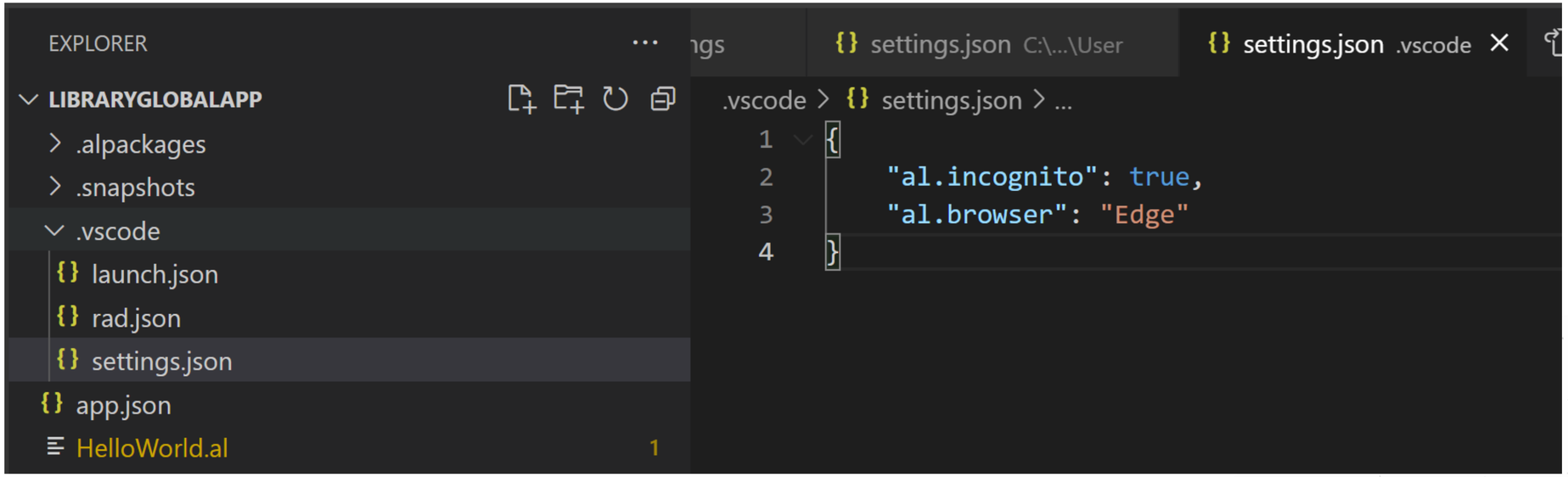
AL Settings



- **User/Global settings** - Settings that apply globally to any instance of VS Code you open.
- ***Workspace Settings*** - *Settings stored inside your workspace and only apply when the workspace is opened. Overrides User settings.*
- **Workspace Folder Settings** - Apply to a specific folder of a multi-root workspace. Overrides Workspace and User settings.

Visual Studio Code User and Workspace Settings

AL Settings



- **User/Global settings** - Settings that apply globally to any instance of VS Code you open.
- **Workspace Settings** - Settings stored inside your workspace and only apply when the workspace is opened. Overrides Global settings.
- ***Project Settings*** - *Apply to a specific project and it overrides the Global settings.*

Visual Studio Code User and Workspace Settings

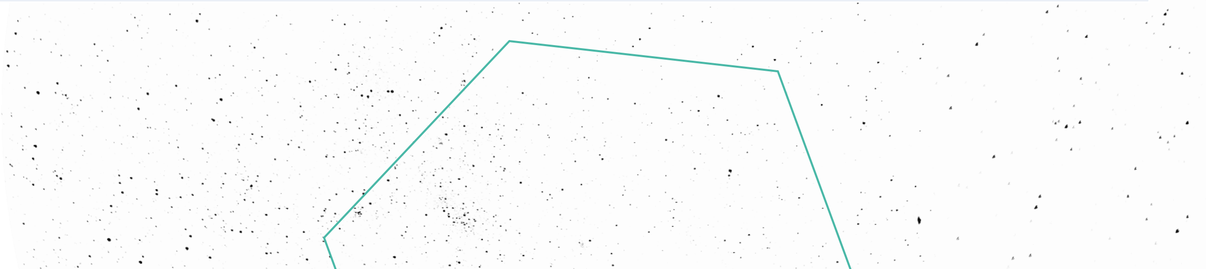
Nonobvious AL Settings

BC/NAV TECH DAYS 2022
mibuso.com

Setting name	Description
al.incrementalBuild	Specifies whether the compiler should reuse the existing background compilation for creating the package. If it is set to false then Ctrl + Shift+ B is equivalent to building the AL project with alc.exe.
al.editorServicesLogLevel	Sets the logging verbosity level for the AL Language Editor Services host executable. Possible values are 'Verbose', 'Normal', 'Warning', and 'Error'. Controls the verbosity output of the C:\Users\<user>\.vscode\extensions\ms-dynamics-smb.al-<version>\bin\win32\EditorServices.log and DebuggerServices.log. If it is set to option other than 'Normal' remember to revert it back to 'Normal' because the files can grow exponentially fast.
al.browser	The AL developer can specify the browser in which to open the Business Central client when launching the application from Visual Studio Code. The available options are 'Edge', 'Edge Beta', 'Chrome', 'Firefox' and 'SystemDefault'.
al.incognito	The AL developer to open the browser in Incognito/InPrivate mode when launching the application from Visual Studio Code. This option will take effect only if the 'al.browser' option is set to a non-default value.

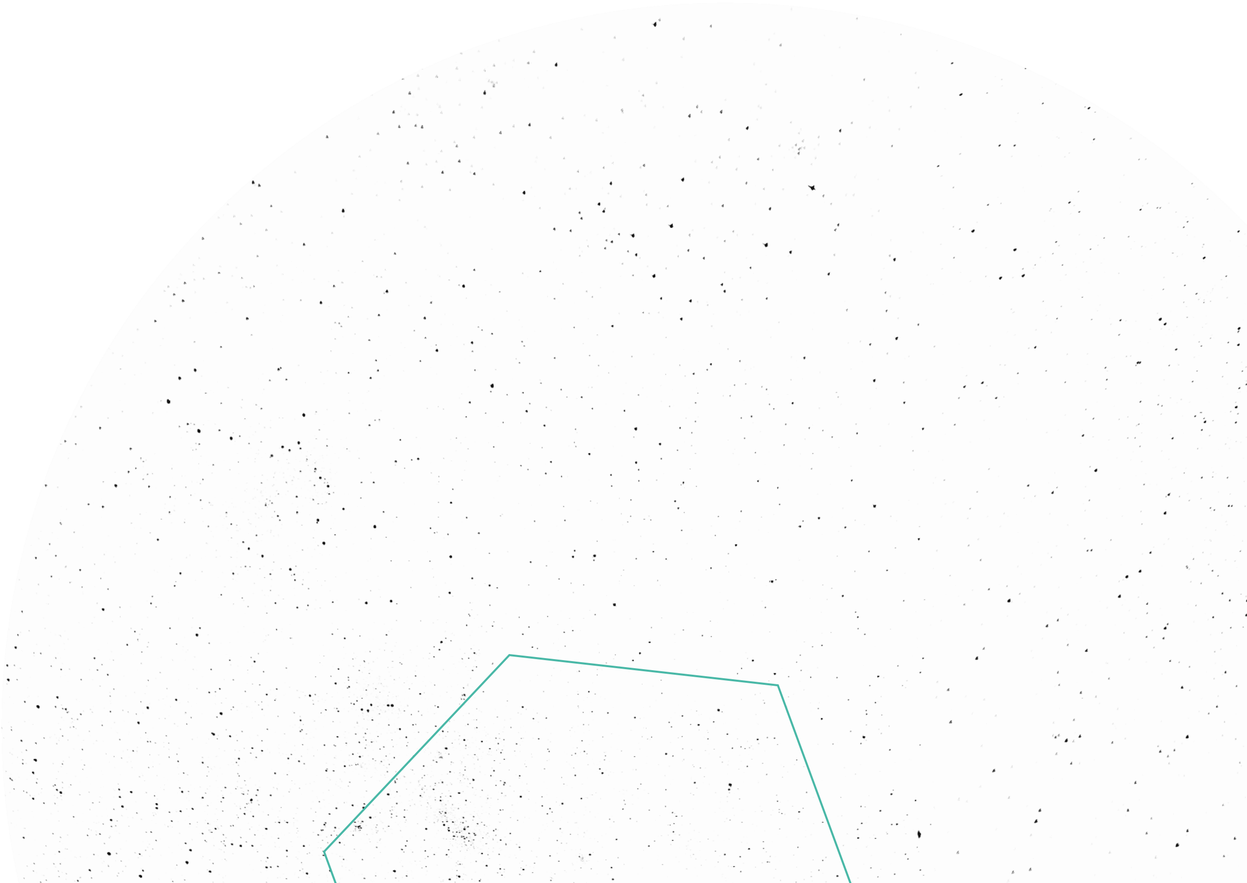
al.compilationOptions

Option	Description
generateReportLayout	Controls whether the compiler will generate Report Layout files when building the package. Default value is true.
parallel	Specifies whether the compiler should use multiple threads when building the project. Default value is true
maxDegreeOfParallelism	Specifies the maximum number of concurrent tasks the compiler should use when compiling the project. Default: 2
delayAfterLastDocumentChange	Specifies the number of milliseconds before getting document diagnostics. Default:800 ms
delayAfterLastProjectChange	Specifies the number of milliseconds to wait for a project and all that depend on the project background compilation to re-start. Default: 4 sec
continueBuildOnError	Specifies if build should continue even if errors are found. It requires "al.incrementalBuild" to be false.



Agenda

- AL Settings
- **Workspaces**
- Dependency publishing
- Rapid Application Development (RAD)



Workspaces

What is an AL workspace?

- It is the collection of one or more AL projects.

What is an AL project?

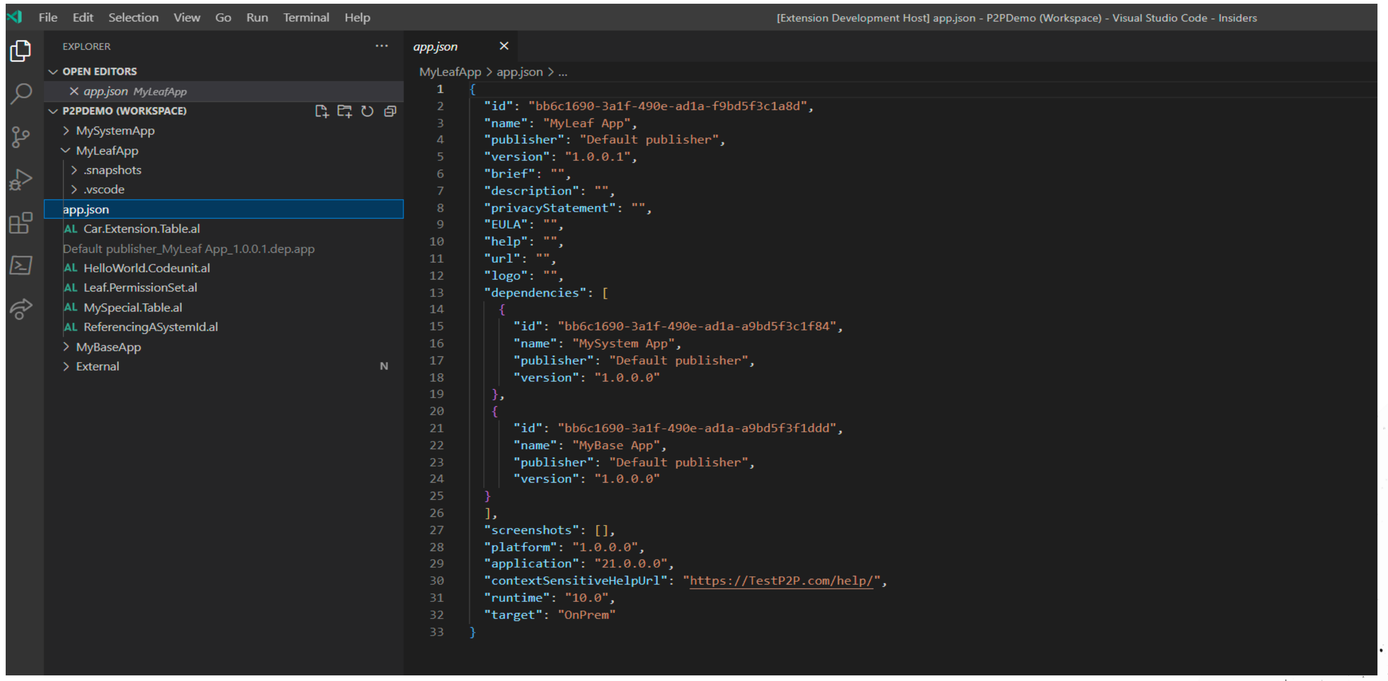
- It is a collection of files(al, json, xml) that produces an AL app file.
- It is defined by the app.json file.

- A workspace can be a standalone single workspace or a multi root workspace defined by .code-workspace.json file

- The biggest advantage of a workspace is that it allows one to work on multiple projects on the same VsCode instance.

Projects with project references

- Projects in a workspace can define a dependency relation.



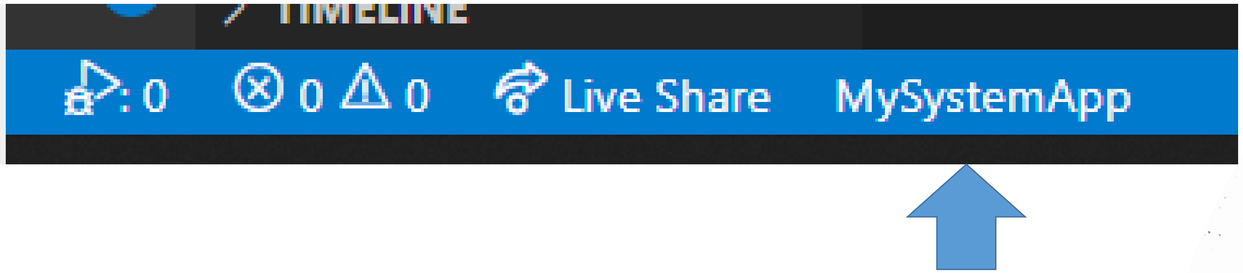
"MySystem App" is a project in the workspace
"MyBase App" is also a project in the workspace

- Within a dependency relation symbols will be resolved from the project and not from the app file.

The active project

- The active project is determined by looking at:
  - The project that has an opened AL file in focus
  - The app.json file opened defining the AL project .

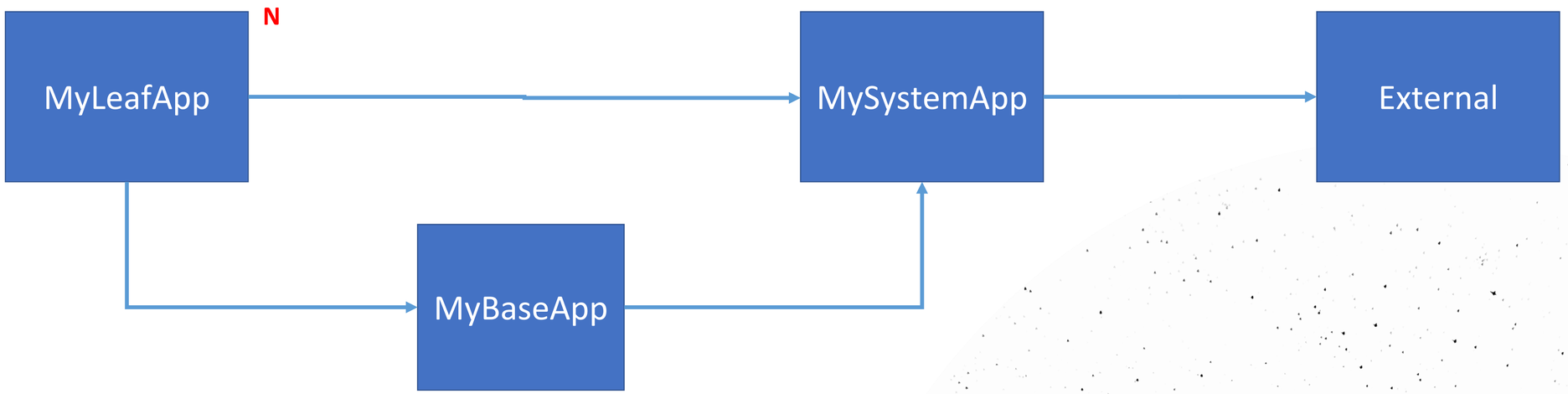
- The active project is shown on the taskbar.



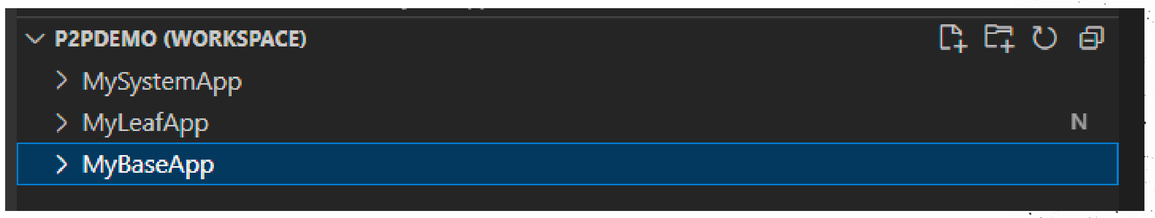
- All operations (commands) execute in the context of the active project.

Loading a workspace with project references

- The active project and all the project references (forward closure) are loaded

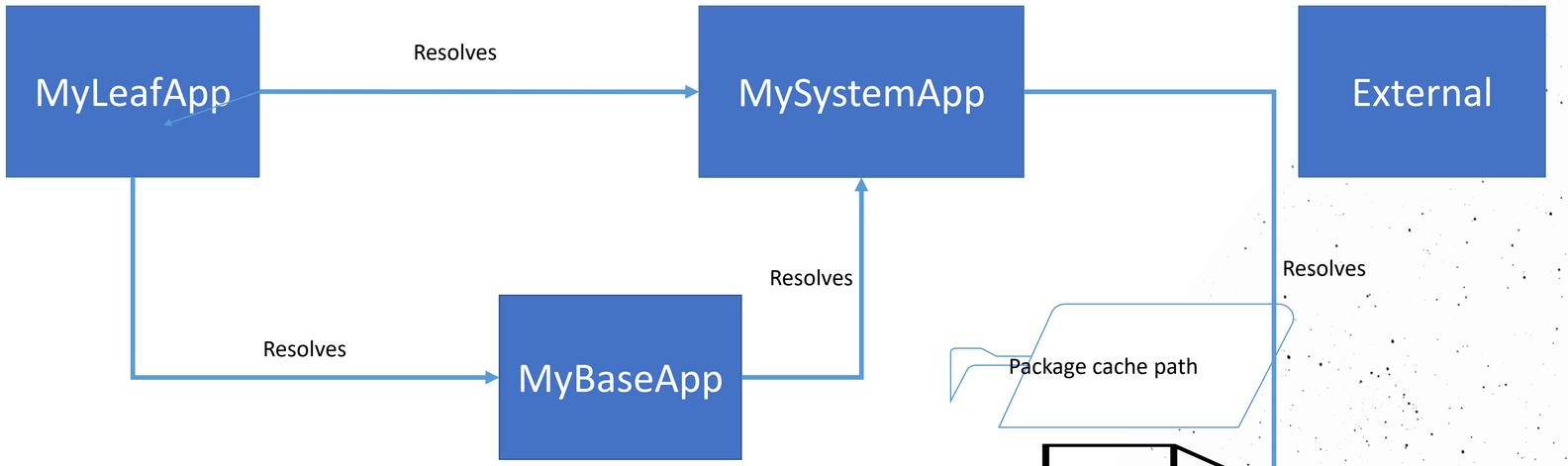


- Projects that are not loaded are marked with the letter N.



Resolving symbols

- A project reference acts as a symbol resolver instead of an app reference.

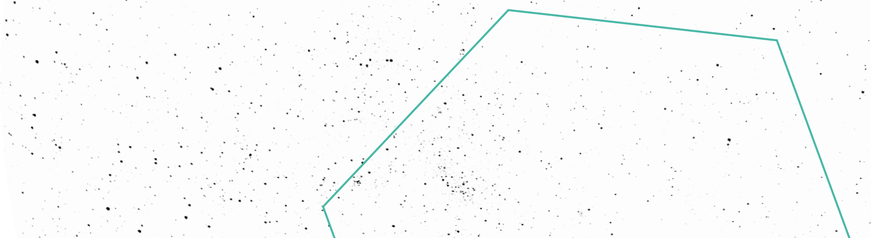


Building projects with project references

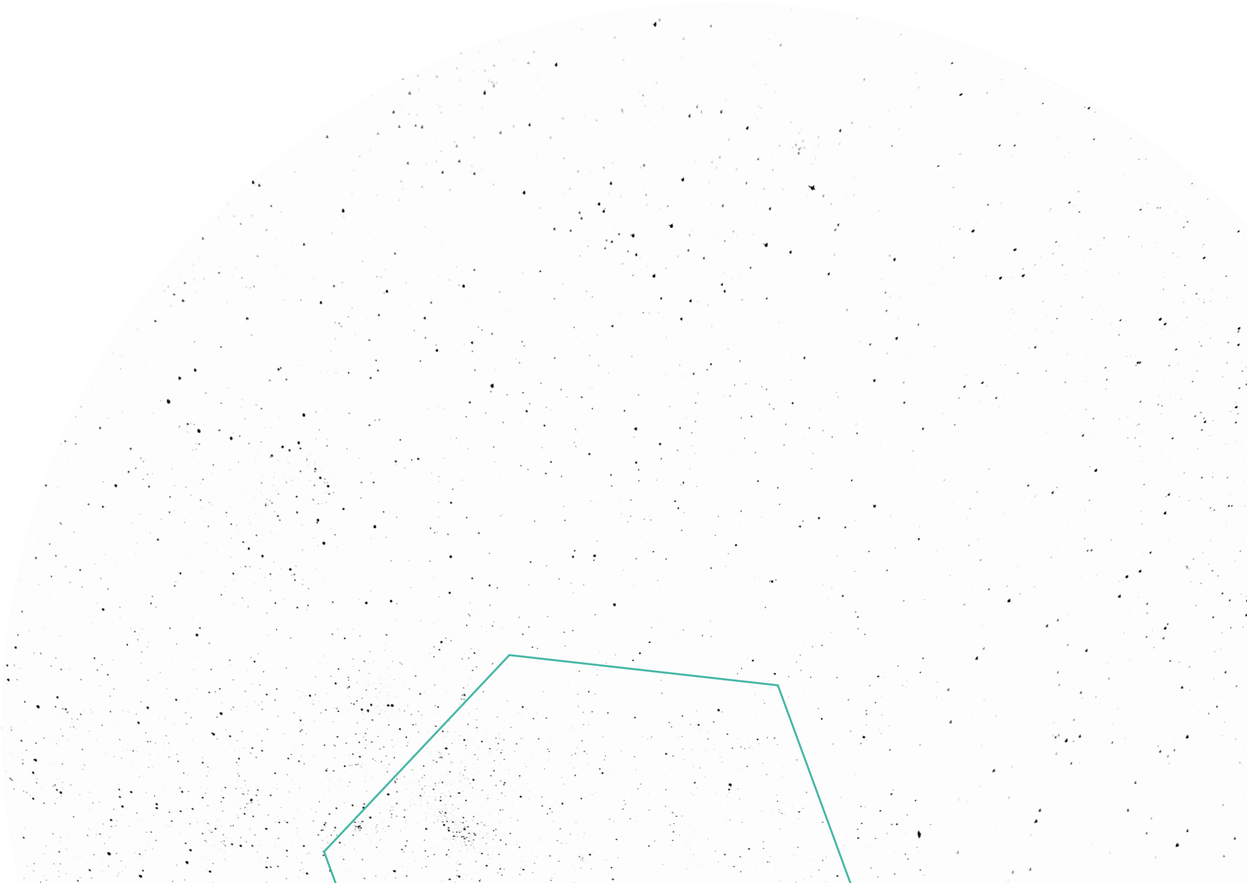
- The active project is built (Ctrl+ Shift+ B)

- In order that dependency publishing works all changed (dirtied) project references are also built. A project is dirtied if an AL file is changed in the project. Dirtying loses scope only after publishing.

- Hint: Do build often since we do a GC collect after each build sequence.

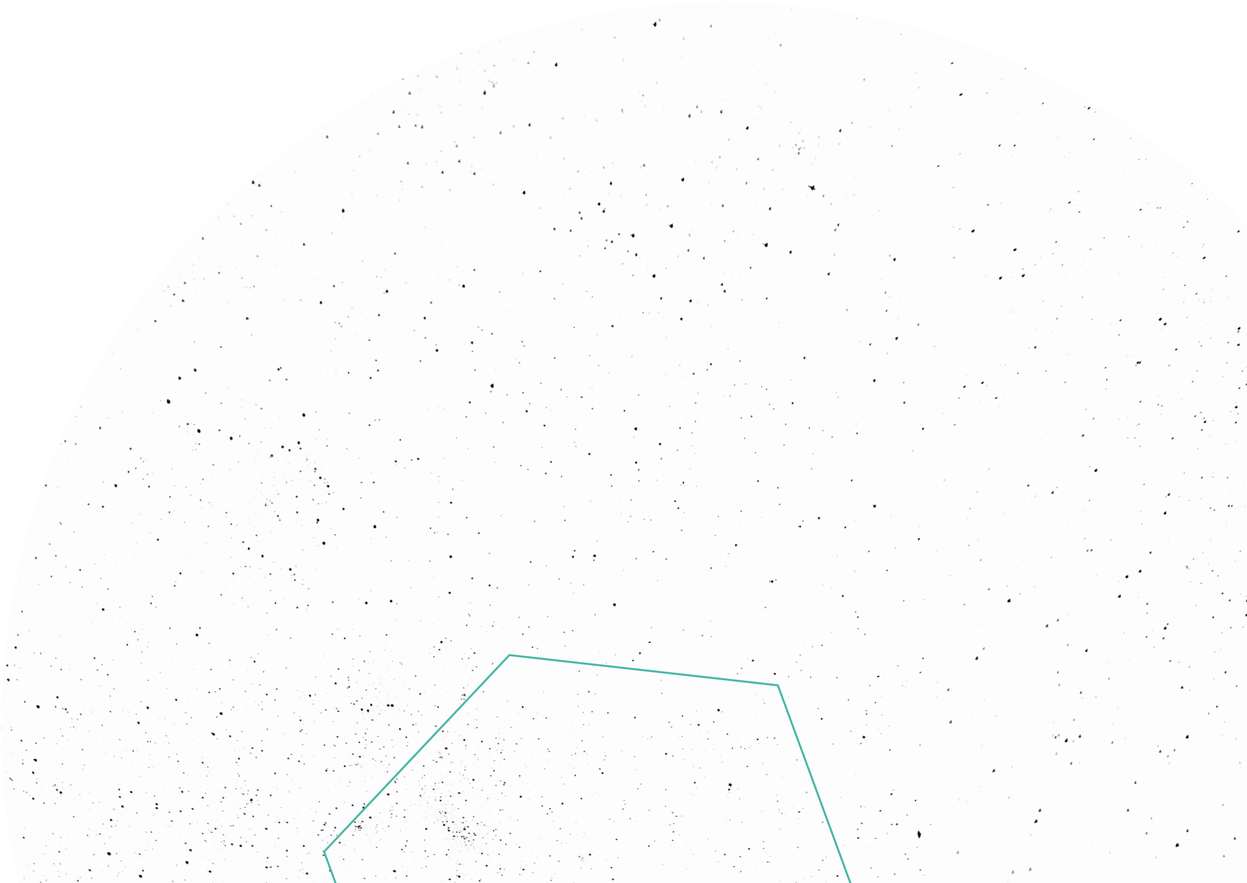


Workspaces Demo

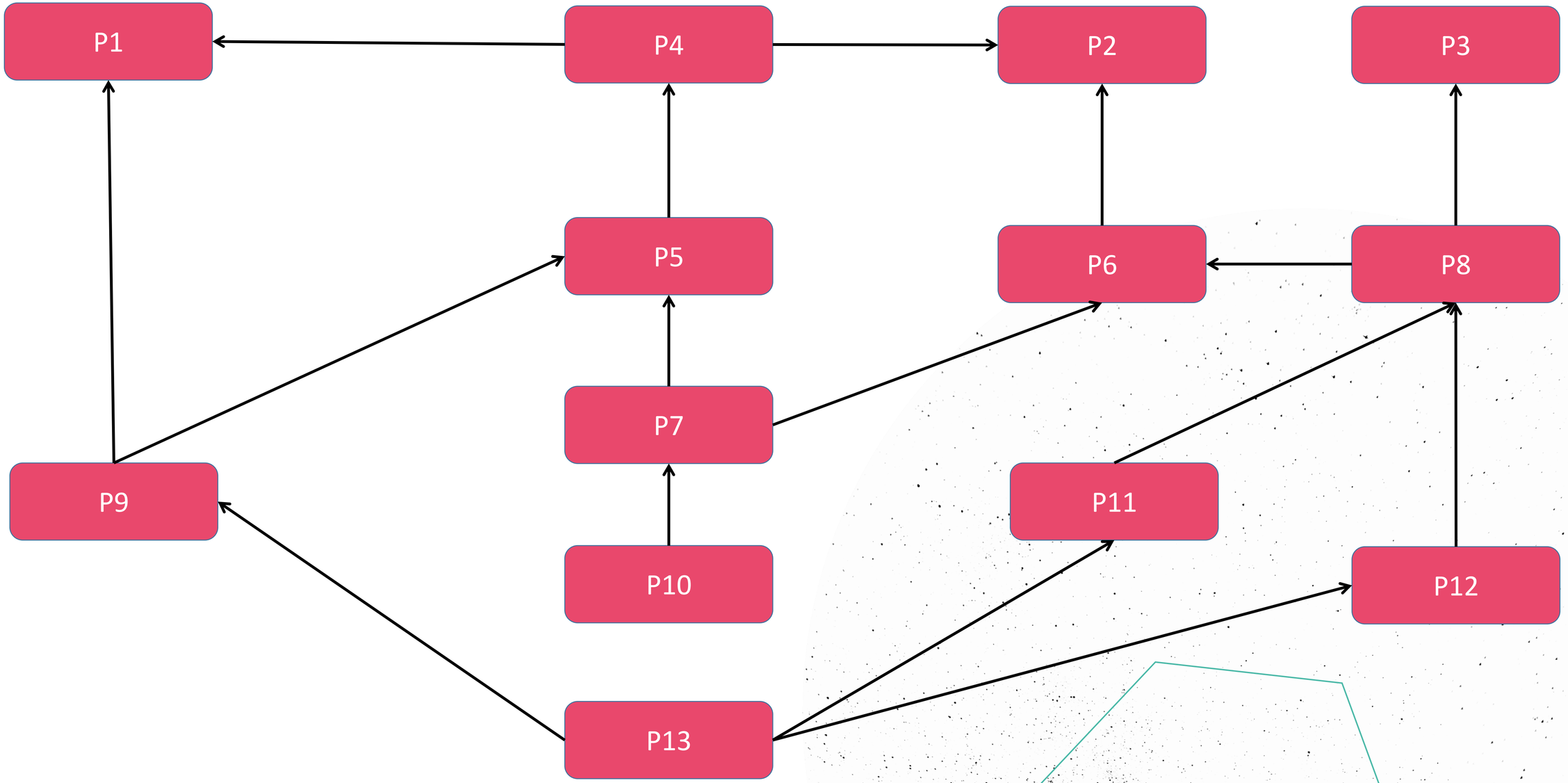


Agenda

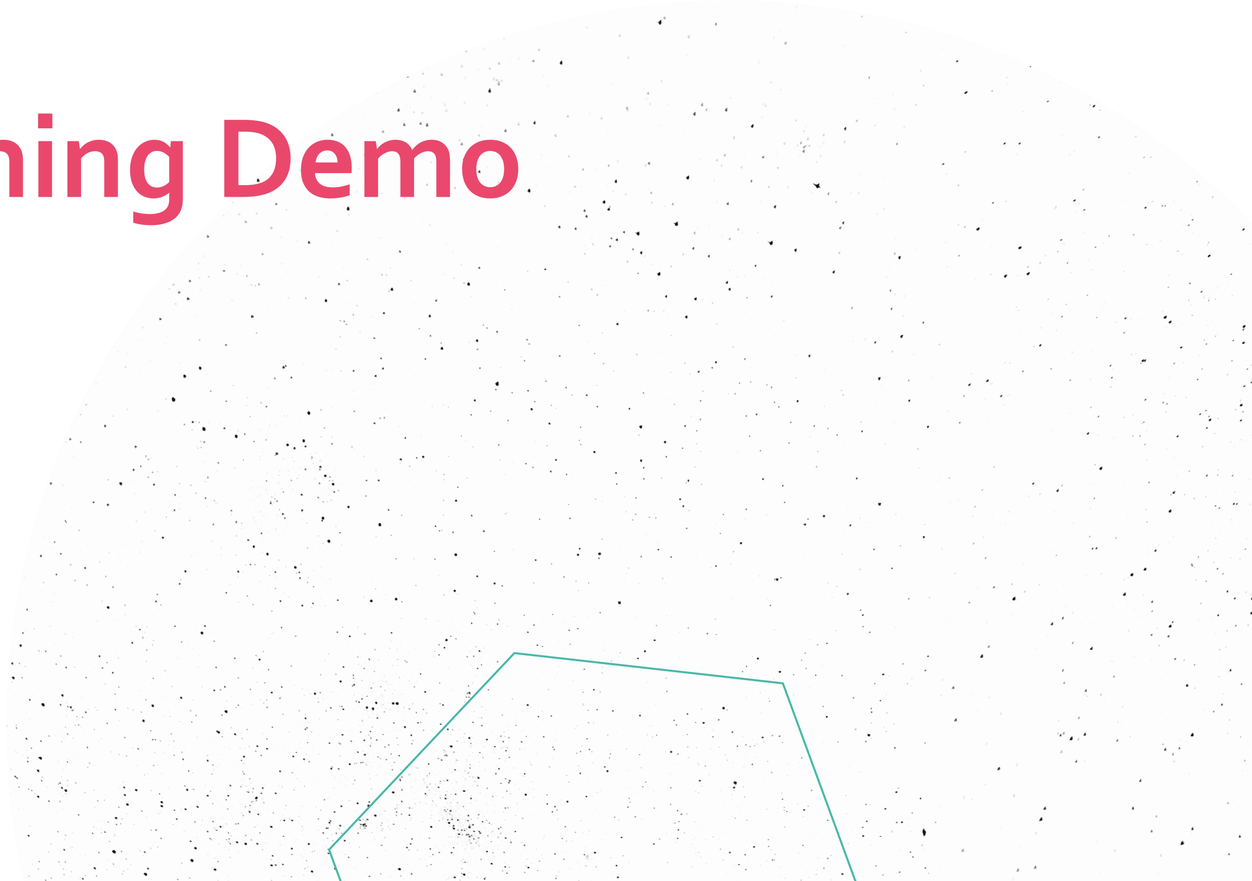
- AL Settings
- Workspaces
- **Dependency publishing**
- Rapid Application Development (RAD)



Dependency publishing



Full Dependency Publishing Demo



Full dependency publishing

BC/NAV TECH DAYS 2022
mibuso.com

- Dependency graph for an active project within a workspace is calculated when the workspace is loaded.

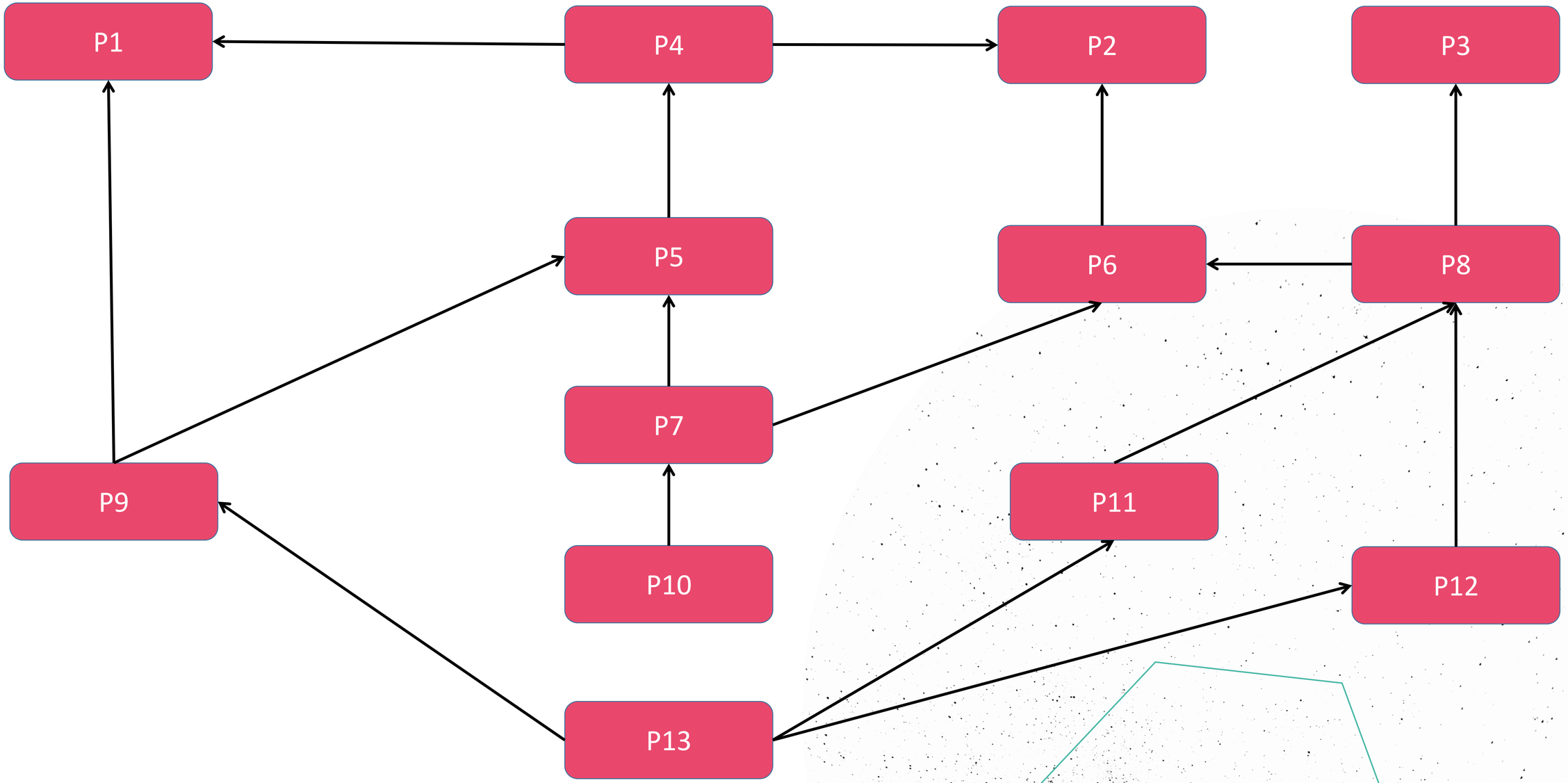
  Note: The dependency graph is built for a workspace meaning if you have a dependency chain [A -> B -> C] and only projects A and C are part of the workspace then the graph won't have a link between A and C.

- Full compilation is done for the whole dependency graph built for the current active project.

  This ensures we have the most up-to-date compiled artifacts. If the compilation of any of the projects fails, the whole operation is aborted.

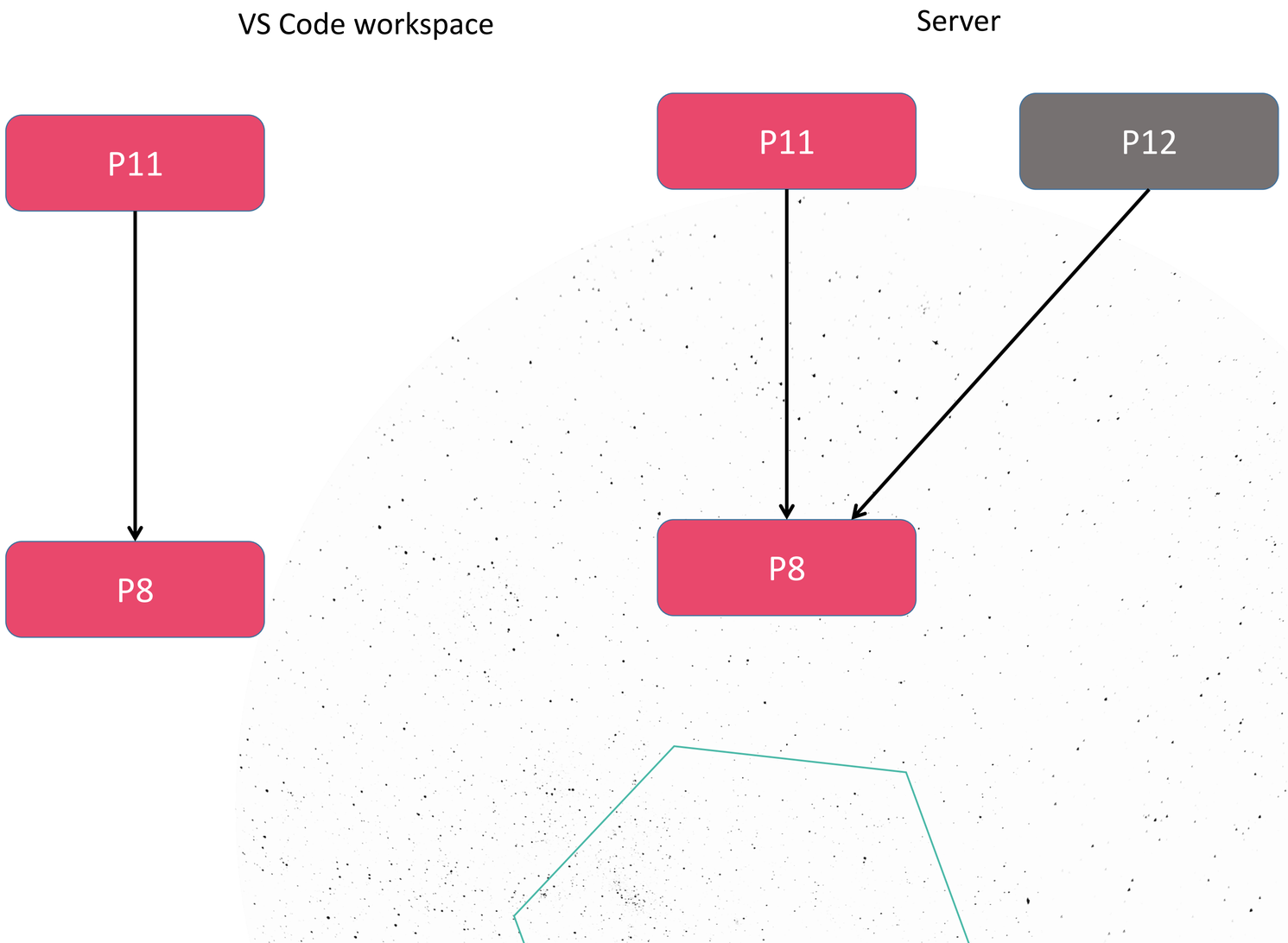
- We publish in topological order.
  - The launch.json configuration of the **current active project** is used to determine the publish parameters.
  - It publishes the whole dependency graph meaning even if the server already contains one of the projects it will be still published.
  - If a publish operation fails because of an error on the server, the whole operation fails.

Dependency publishing



Dependency publishing

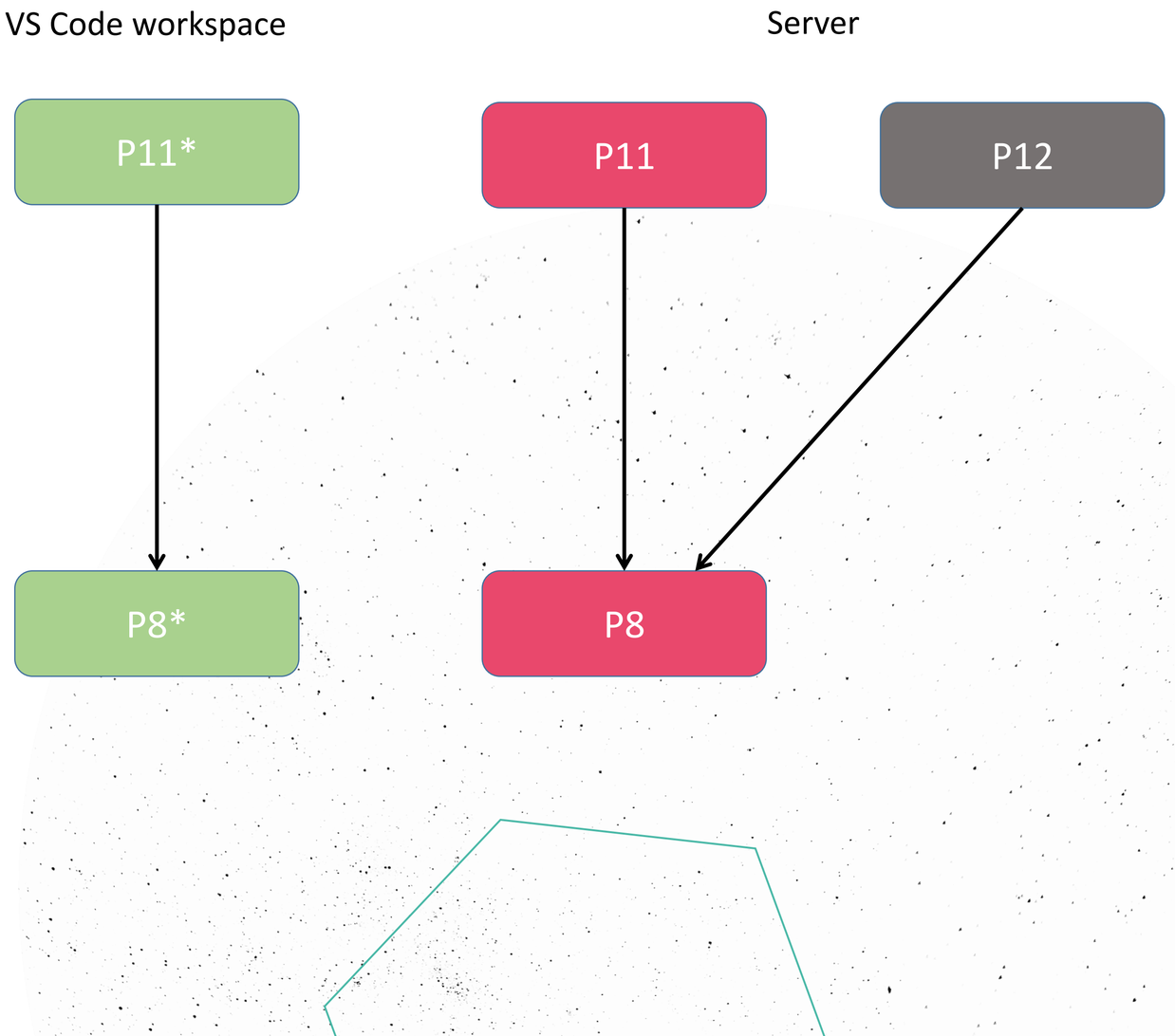
- **P11** and **P8** are part of the same workspace.
- **P11**, **P8** and **P12** are already published to the server.



Dependency Publishing

Both Workspace projects are edited.

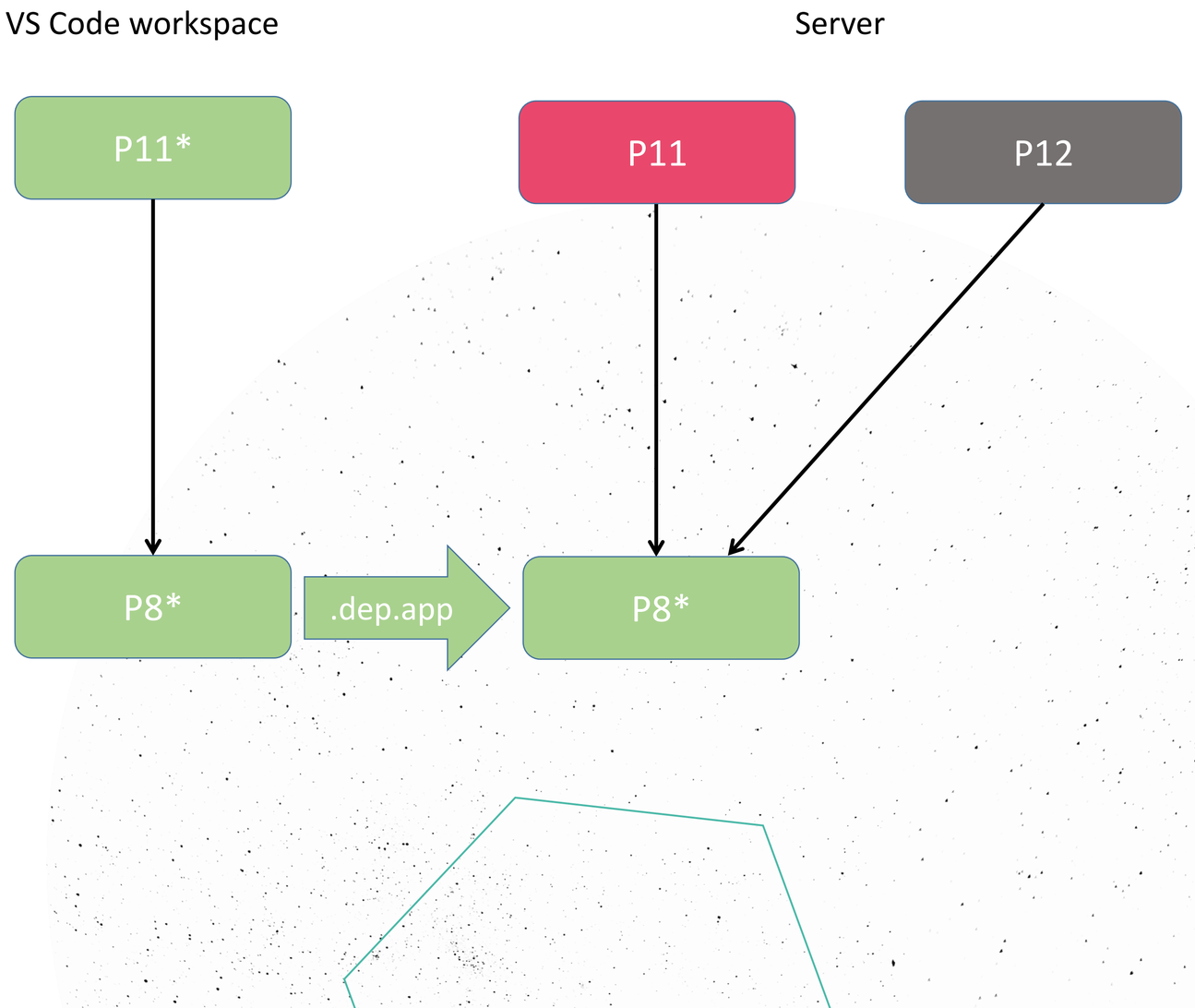
Denoted with Asterix *



Dependency Publishing

Deploying P8 to the server will result in the following.

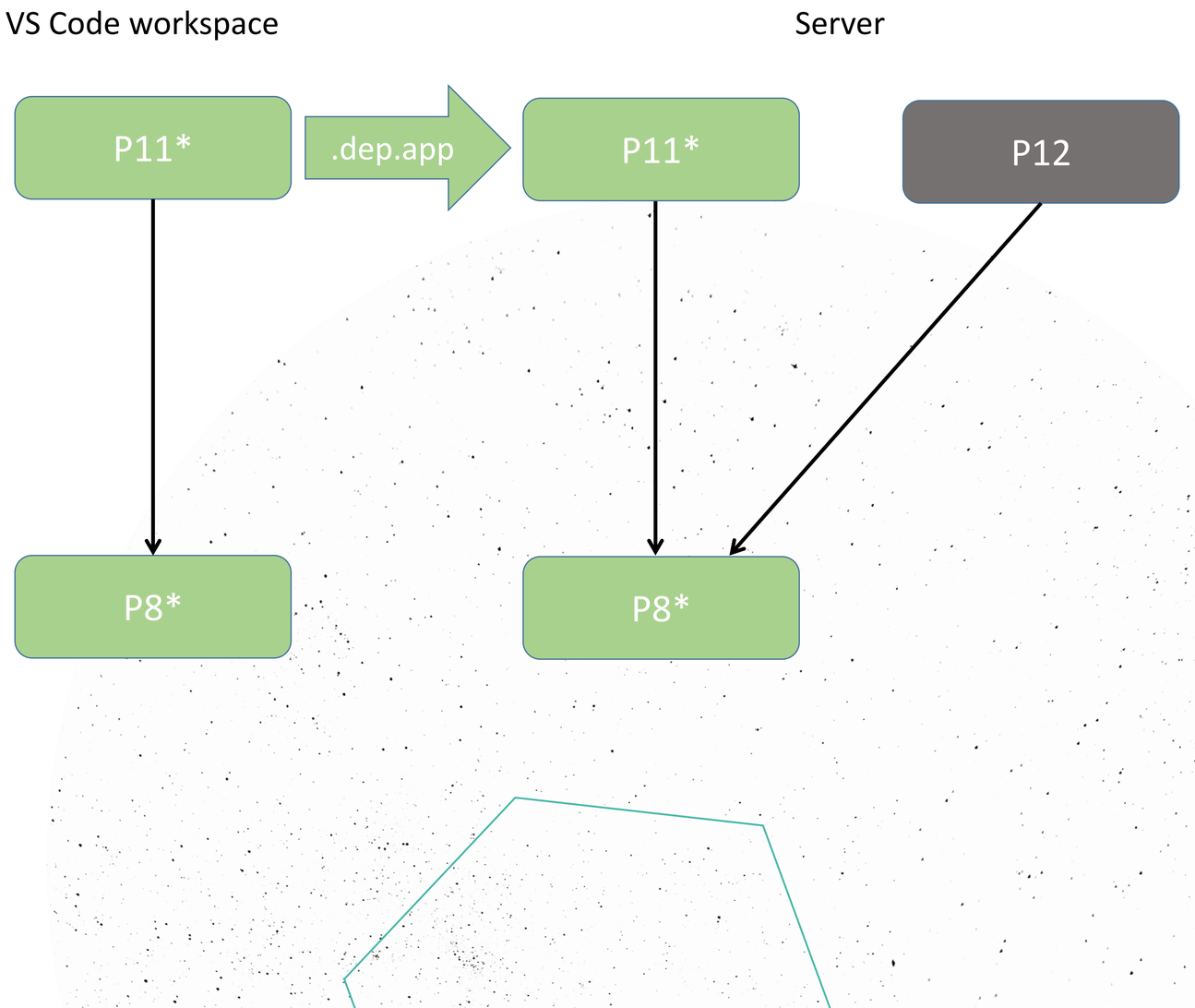
1. Uninstalling every dependency app available on the server- **P11** and **P12**.
2. Publishing **P8**.
3. Installing back every dependency app- **P11** and **P12**.



Dependency Publishing

Now if we deploy **P11** we will have the following scenario.

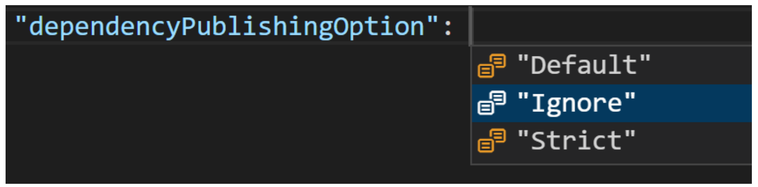
1. All extensions in the graph are uninstalled (including the ones not in the VS Code Workspace).
2. In this case both **P11** and **P8** are deployed.
3. All extensions available on the server part of the graph are reinstalled.

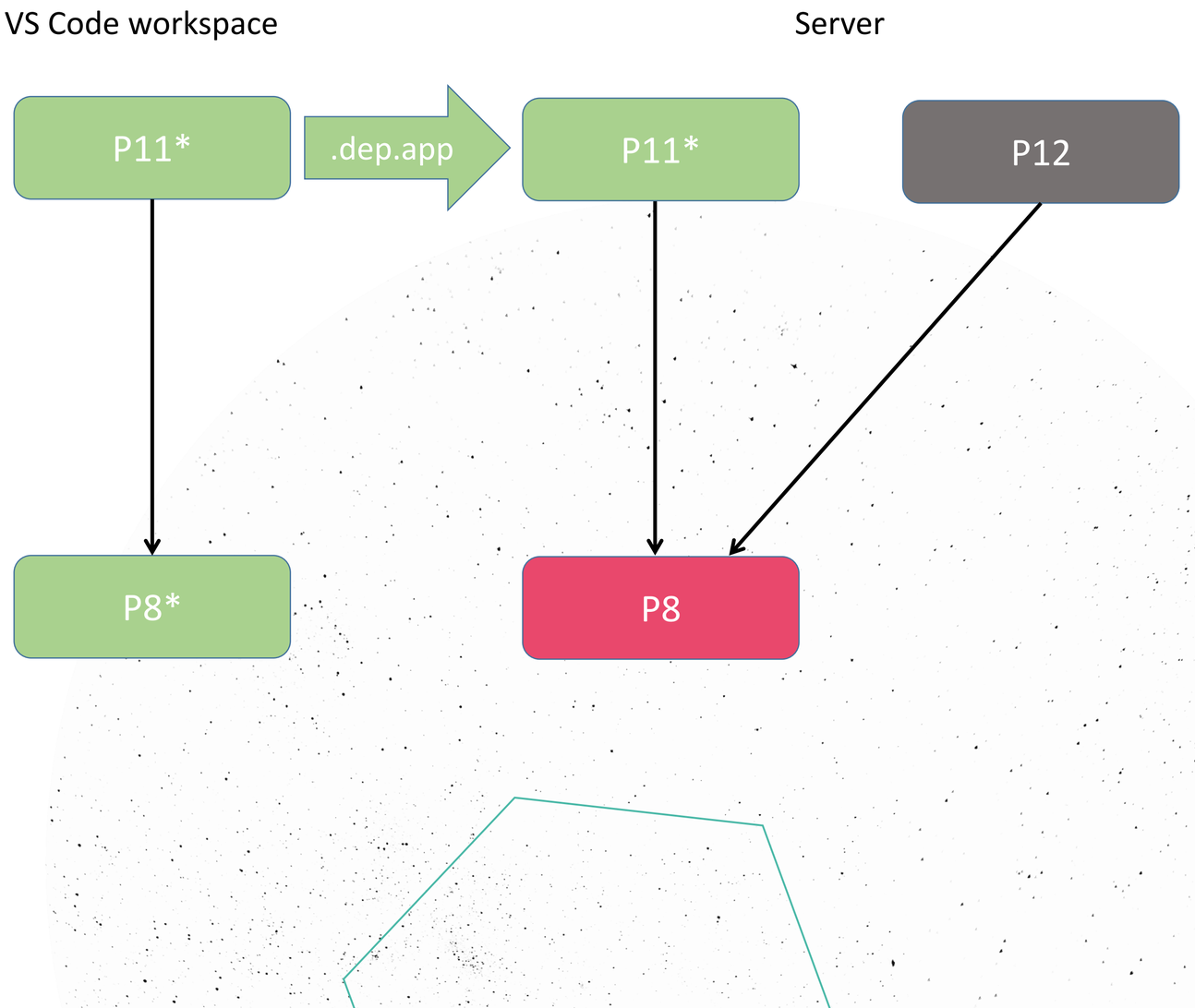


Dependency Publishing with Ignore

- When **P8** is big and it takes long time to recompile.
- When we know **P11** will work with the older version of **P8** and recompilation is not needed.

Specifying dependencyPublishingOption in launch.json file.



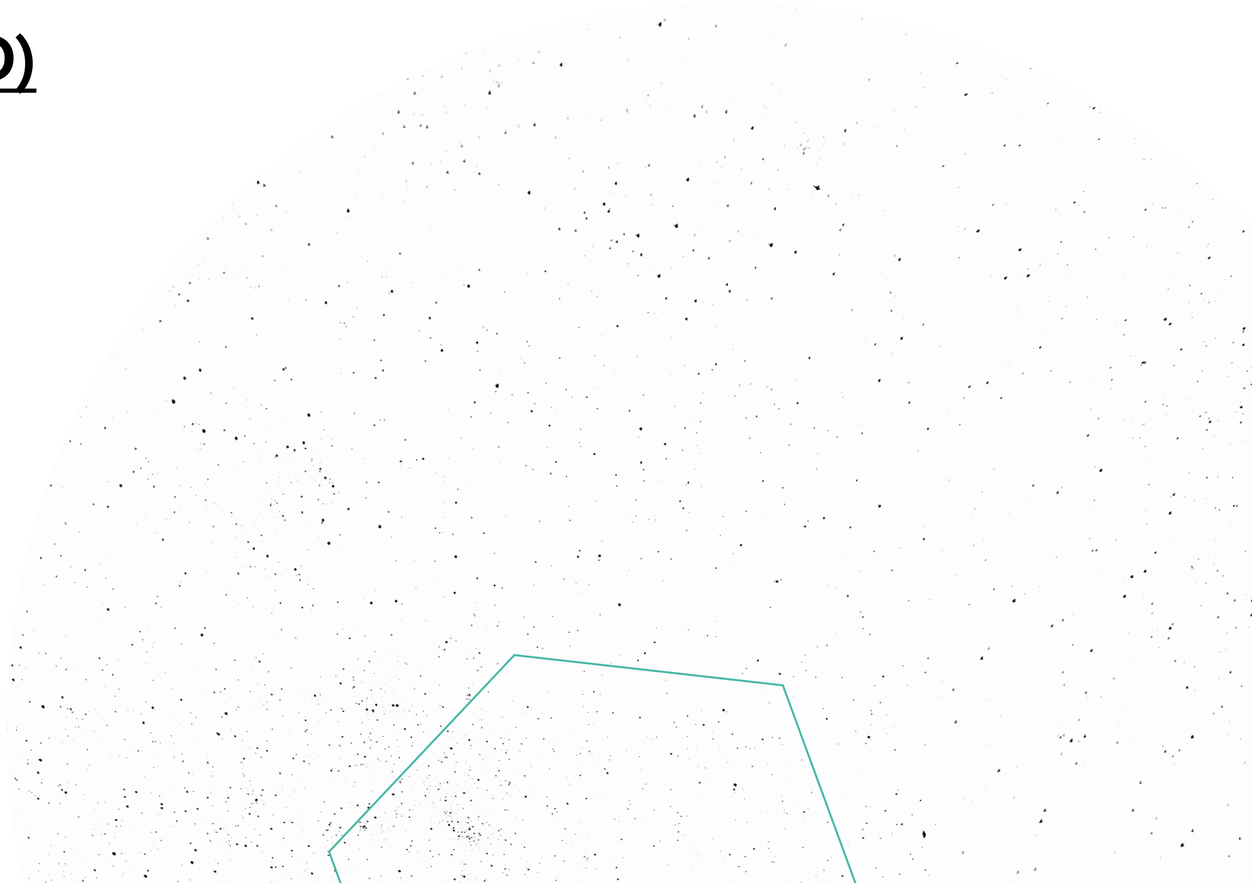


[Launch.json file options]

Dependency Publishing Demo

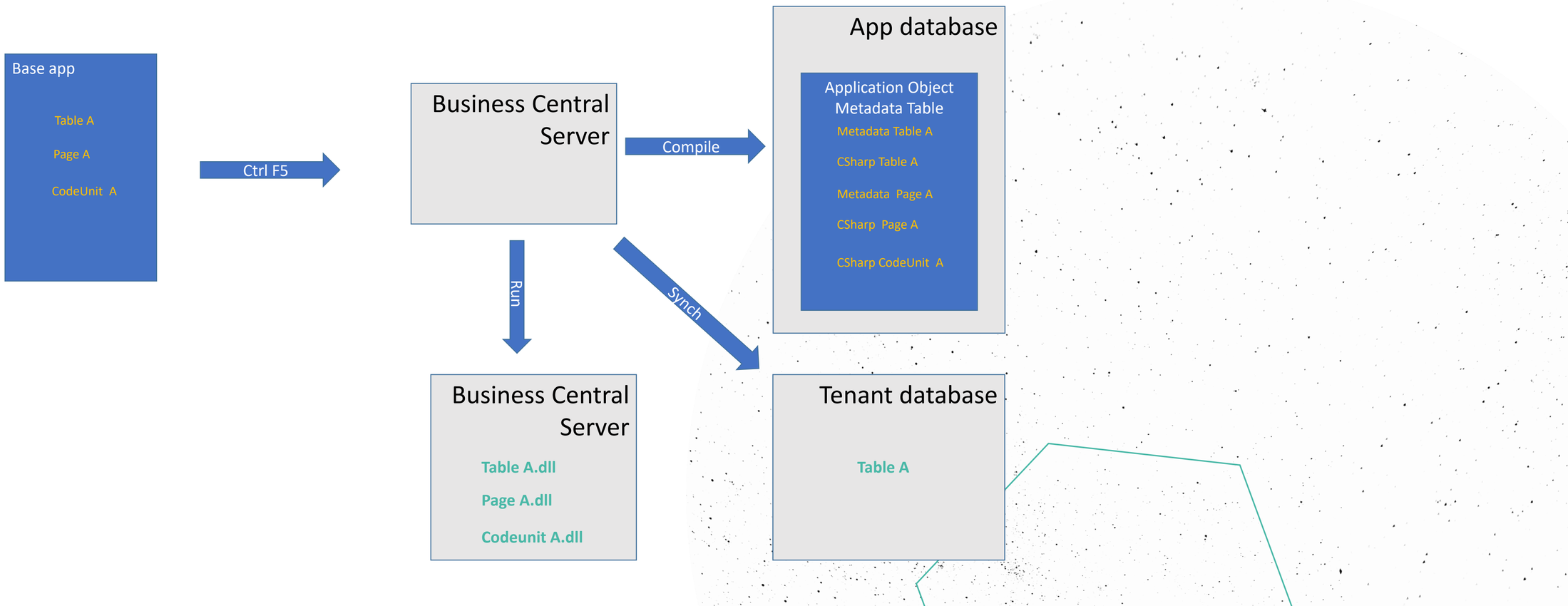
Agenda

- AL Settings
- Workspaces
- Dependency publishing
- **Rapid Application Development (RAD)**



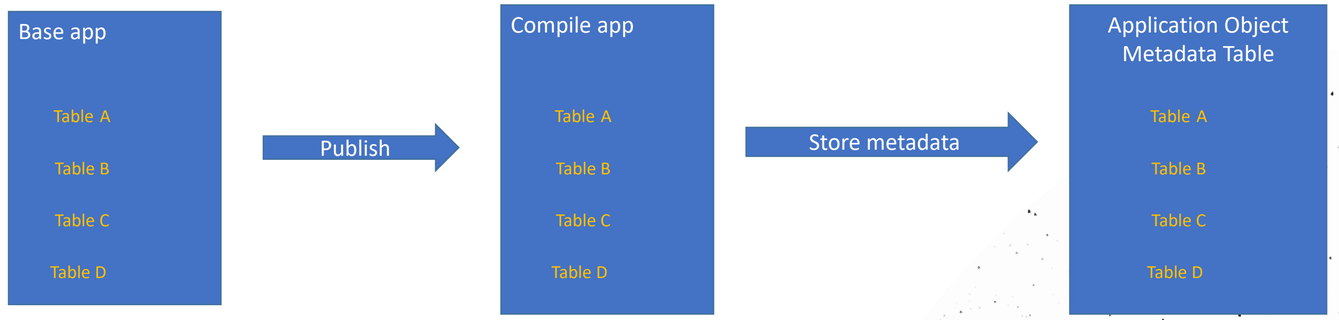
Rapid application development aka RAD

- It is a fast server incremental compilation and deployment step.
- A very much simplified drawing on how Business Central runs AL code:

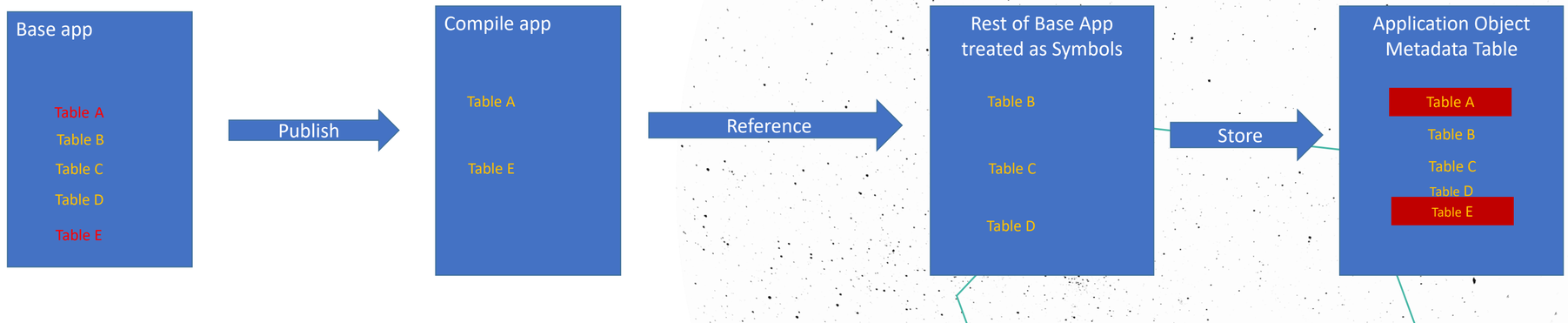


RAD explained

- How does it work? We need a baseline of a published app.

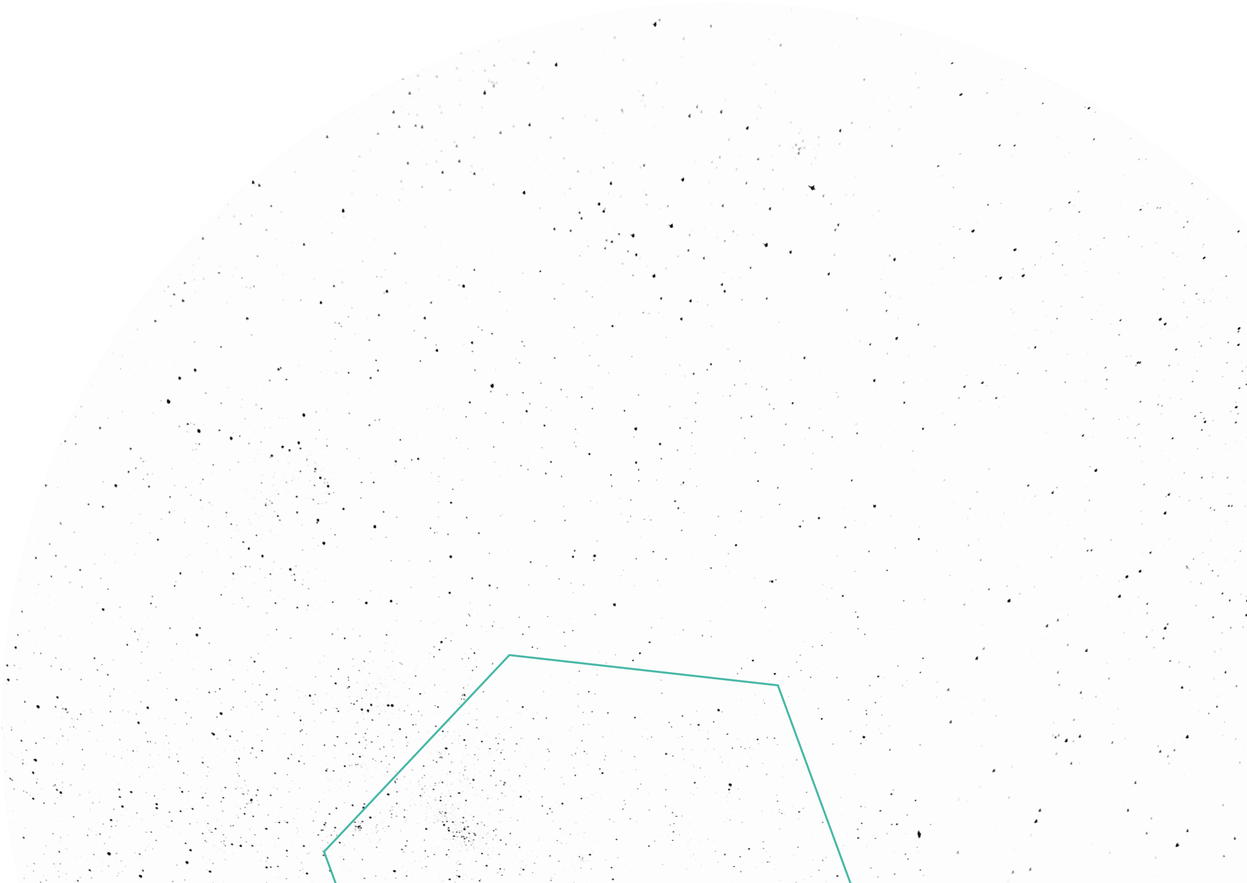


- Modify Table A and add Table E. The NST compilation then will look like:



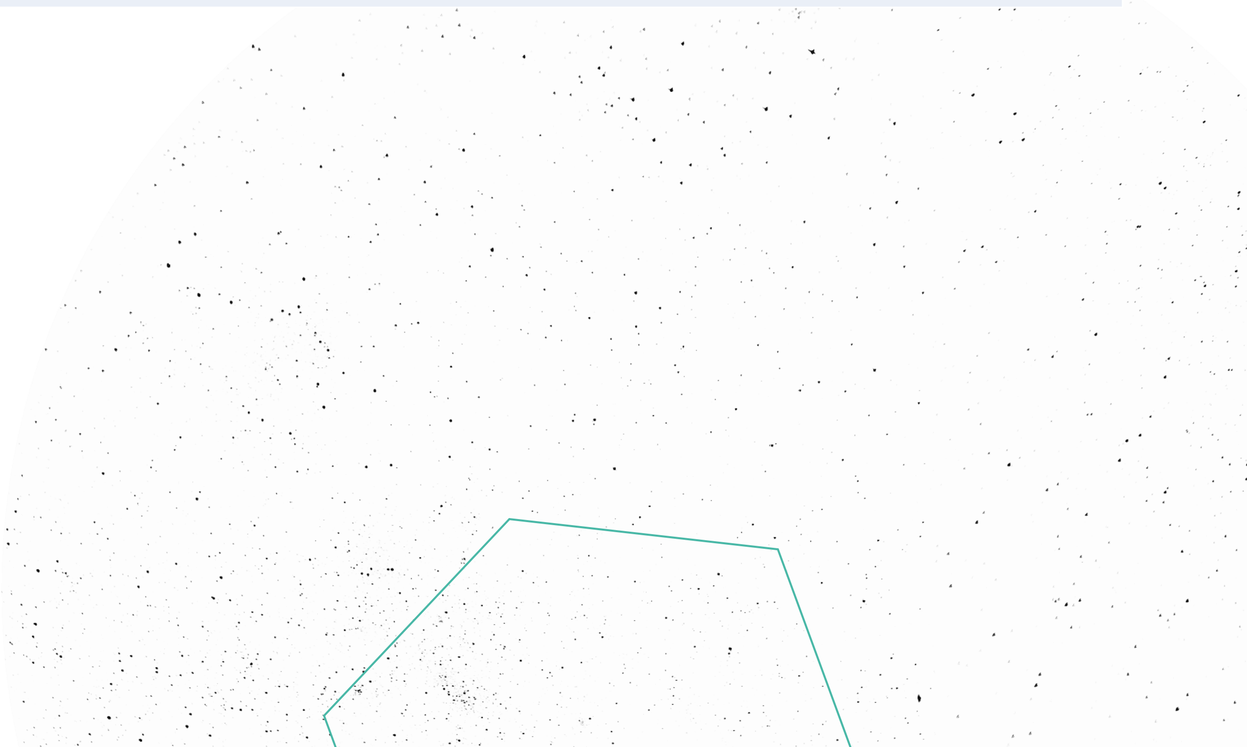
BC/NAV TECH DAYS 2022
mibuso.com

RAD Demo



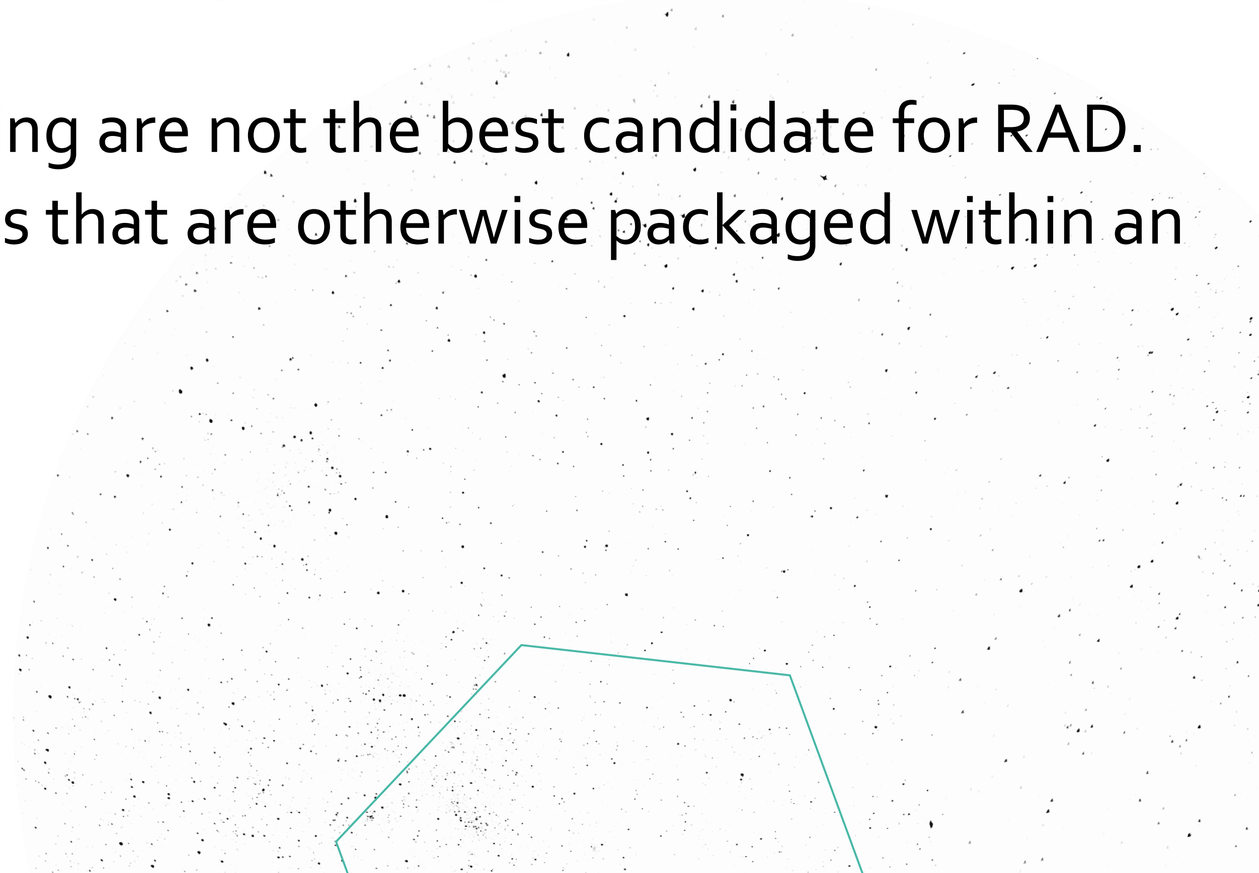
Visual Studio Code commands:

Shortcut	Command
Ctrl + Alt+ F5	Rapid Application Publish without debugging
Alt + F5	Rapid Application Publish with debugging



Things to be aware:

- RAD publishing requires a baseline
- Changes in the manifest (name, publisher, version) are not supported for RAD
- If RAD publishing fails a full publishing is needed
- RAD state (Rad.json) is kept until a successful publishing is performed within an instance of VSCode.
- Application Object ID rename refactoring are not the best candidate for RAD.
- RAD publishing would not package files that are otherwise packaged within an app file:
  - Translation files
  - Permission files
  - Layout files
  - Web service definitions



Q&A

Any Questions?

BC/NAV TECH DAYS 2022
mibuso.com

Thank You!