

# Develop using Eclipse

## Objectives:

- Install and configure Eclipse
- Install and configure the Eclipse AEM plugin
- Import an AEM project in Eclipse
- Sync AEM content



# Choose the right IDE

Integrated Development Environments (IDEs) are used to speed up development on projects. Ultimately, it's up to the developer and organization standards to choose.

Common Java IDEs used for AEM development

- Eclipse
- Visual Studio Code
- IntelliJ

Common Frontend IDEs for AEM development

- Visual Studio Code
- Brackets
- Basic text editors (Notepad++, bbedit, etc)

Depending on the project size and number of developers, it's not uncommon to use many different IDEs among different teams and initiatives.

# Key AEM Development features

Common features an IDE should have for AEM development

- Java Support
- Node/NPM support
- Maven
- Junit test runner
- Remote Java Debugging
- Dependency management and autocomplete
- AEM Project Sync

Nice to have

- JCR Properties view (.content.xml files)
- Git support

# Eclipse IDE

Eclipse offers many benefits

- Free and open source
- Auto completion
- Easy Code refactoring
- Highly customizable to specific projects
- Different perspectives depending on development activities
- Error debugging
- Supports many other languages beyond Java
- Many framework integrations and plugins available

Benefits for AEM

- m2e plugin for maven
- AEM sync plugin
- Git plugin





## **Exercise 1: Install and configure Eclipse (Local only)**

ADLS Readytech environments already have Eclipse installed

This exercise only needs completed if you are working locally.

# Eclipse | AEM Sync Plugin

Based on the Eclipse plug-in for Apache Sling.

Benefits include:

- Synchronization
  - Auto sync to AEM per saved file
  - Pull files from AEM
- Supports debugging and code hot-swapping
- Simple project wizard for AEM projects
- View and edit xml files as JCR properties
- Managed AEM server connection

## AEM Developer Tools

The Eclipse Plug-In that brings you the full connection to the Adobe Experience Manager.

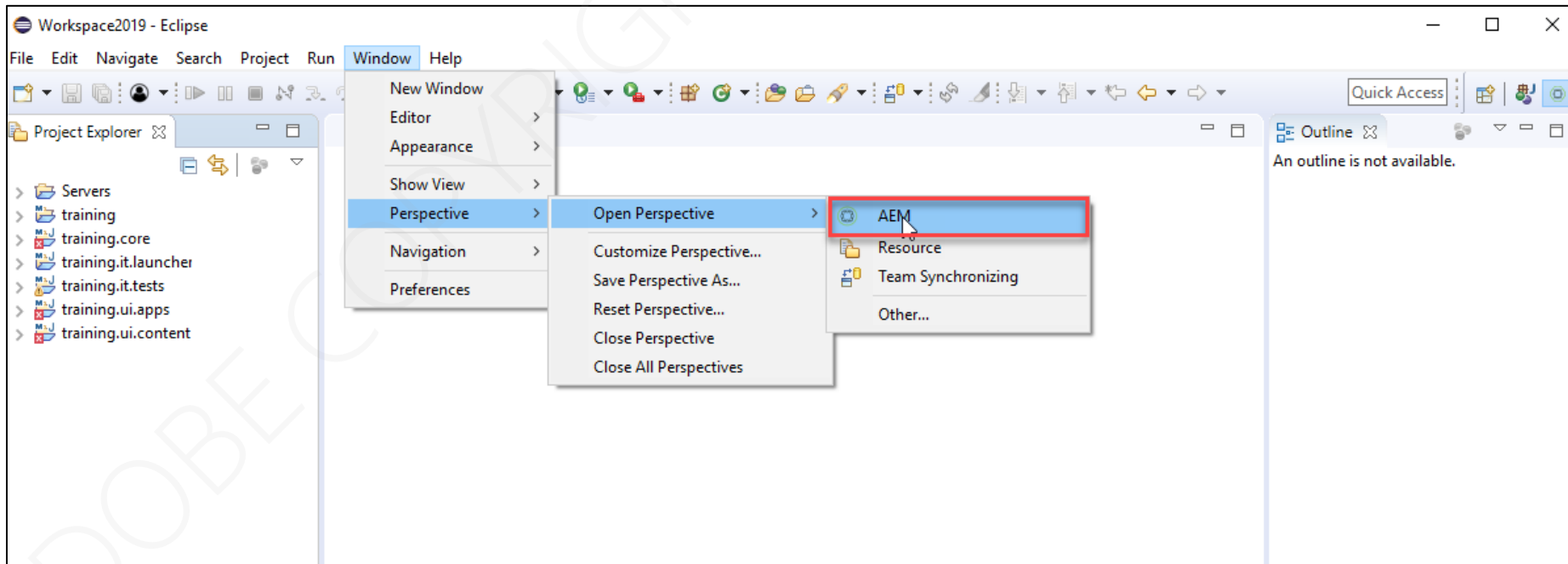
» [Online Documentation](#)



# AEM Sync Plugin | AEM Perspective

Navigate to Window > Perspective > Open Perspective and select AEM.

- JCR Property view with easy edit capabilities
- Virtualizes content packages into a JCR view
- Create Node option available based on common nodetypes



## AEM Sync Plugin | JCR Properties View

## AEM project store JCR properties in .content.xml files

Xml files can be viewed and edited in the JCR Properties panel

# Editing Supports

- All property types
- Value and value arrays
- Auto sync on modification to AEM

The screenshot shows the IntelliJ IDEA IDE. On the left, the Project Structure view displays the hierarchy: training.ui.apps [backend-training dev] > src/main/content/jcr\_root [nt:folder] > apps [nt:folder] > training [nt:folder] > config.author [nt:folder]. The file com.adobe.granite.csrf.impl.CSRFFilter [sling:OsgiConfig] is selected under config.author. On the right, the JCR Properties tool window is open, displaying the properties for /apps/training/config.author/com.adobe.granite.csrf.impl.CSRFFilter. The properties are listed in a table:

Name	Type	Value
jcr:primaryType	String	sling:OsgiConfig
filter.enable.safe.user.agents	Boolean	false
filter.excluded.paths	String[]	[/content/pagecreator]
filter.methods	String[]	[POST,PUT,DELETE]
filter.safe.user.agents	String[]	[Apache-HttpClient/*,*Jakarta\\\\\\\\\\\\\\\\\\\\ Commons-HttpCli

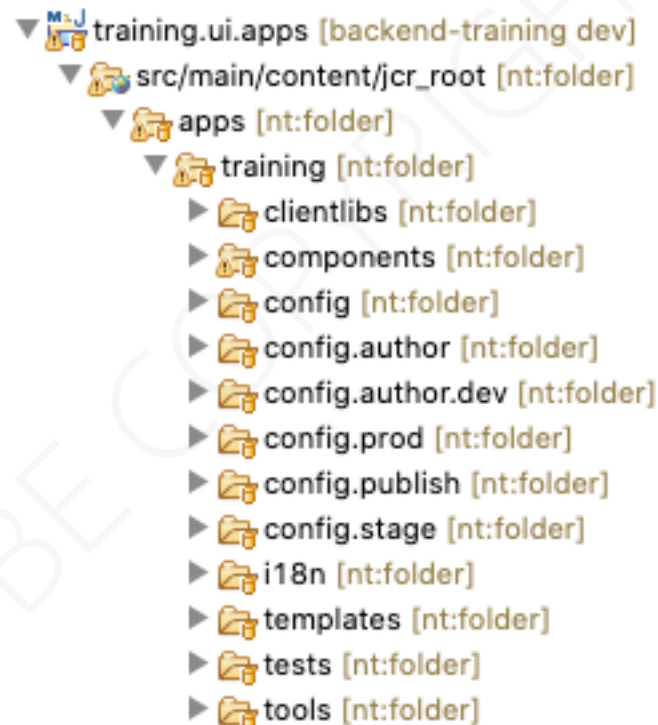


# AEM Sync Plugin | Virtual JCR view

To Enable right click on a content package maven module

- Select Properties > AEM
- Set the content sync root directory to the jcr\_root folder

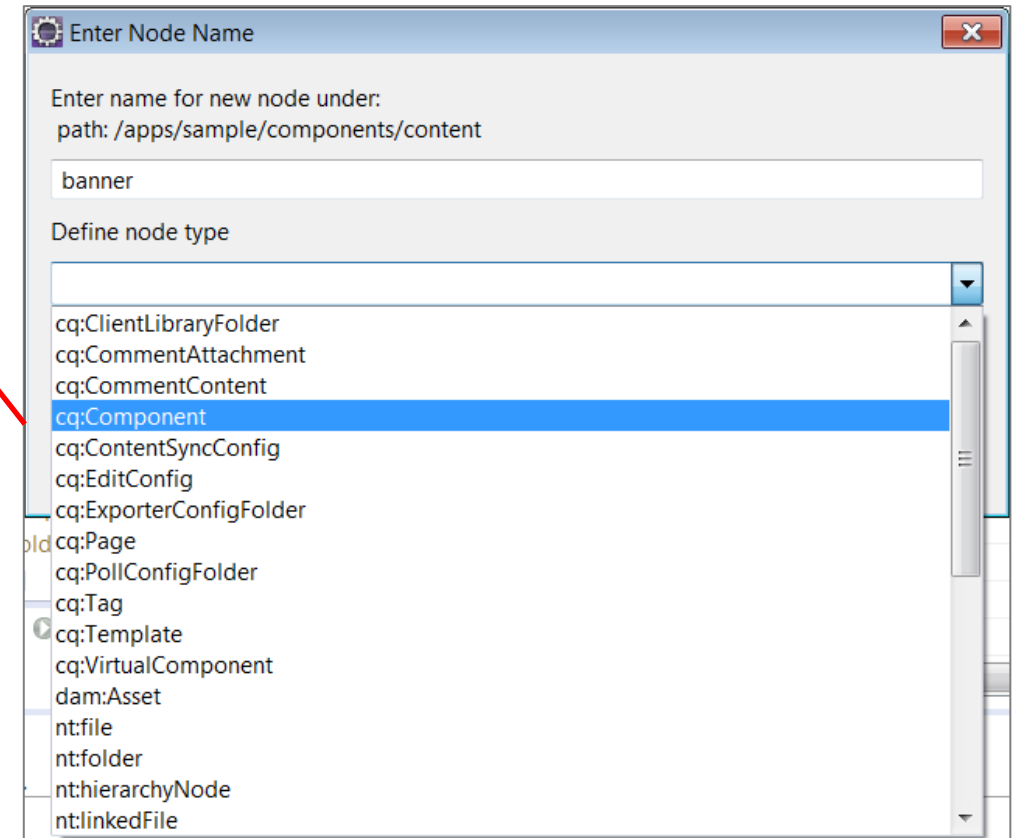
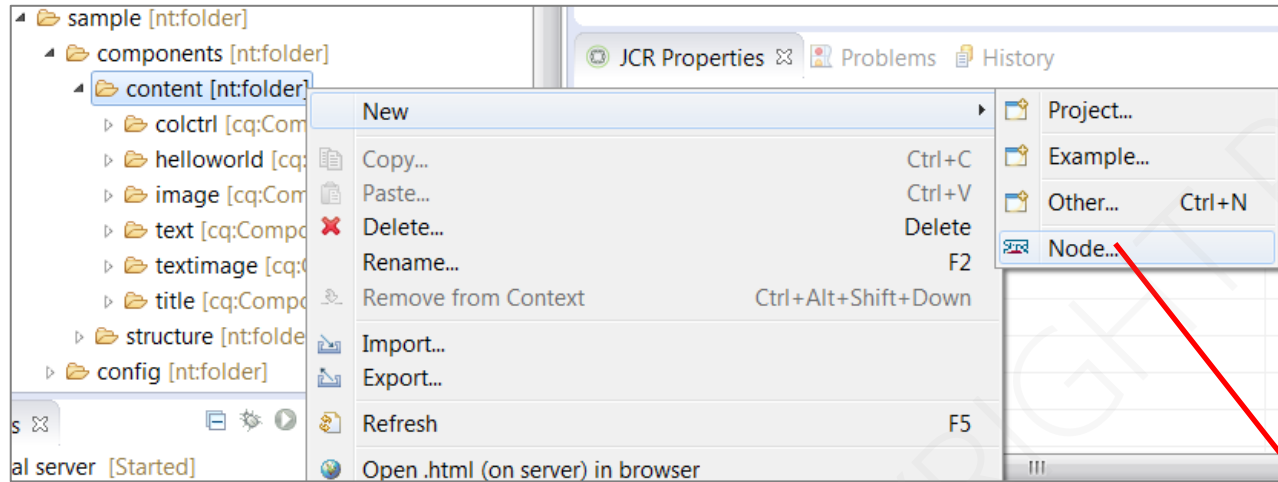
## AEM Project with Virtual JCR



## Typical Java Project



# AEM Sync Plugin | Create a Node





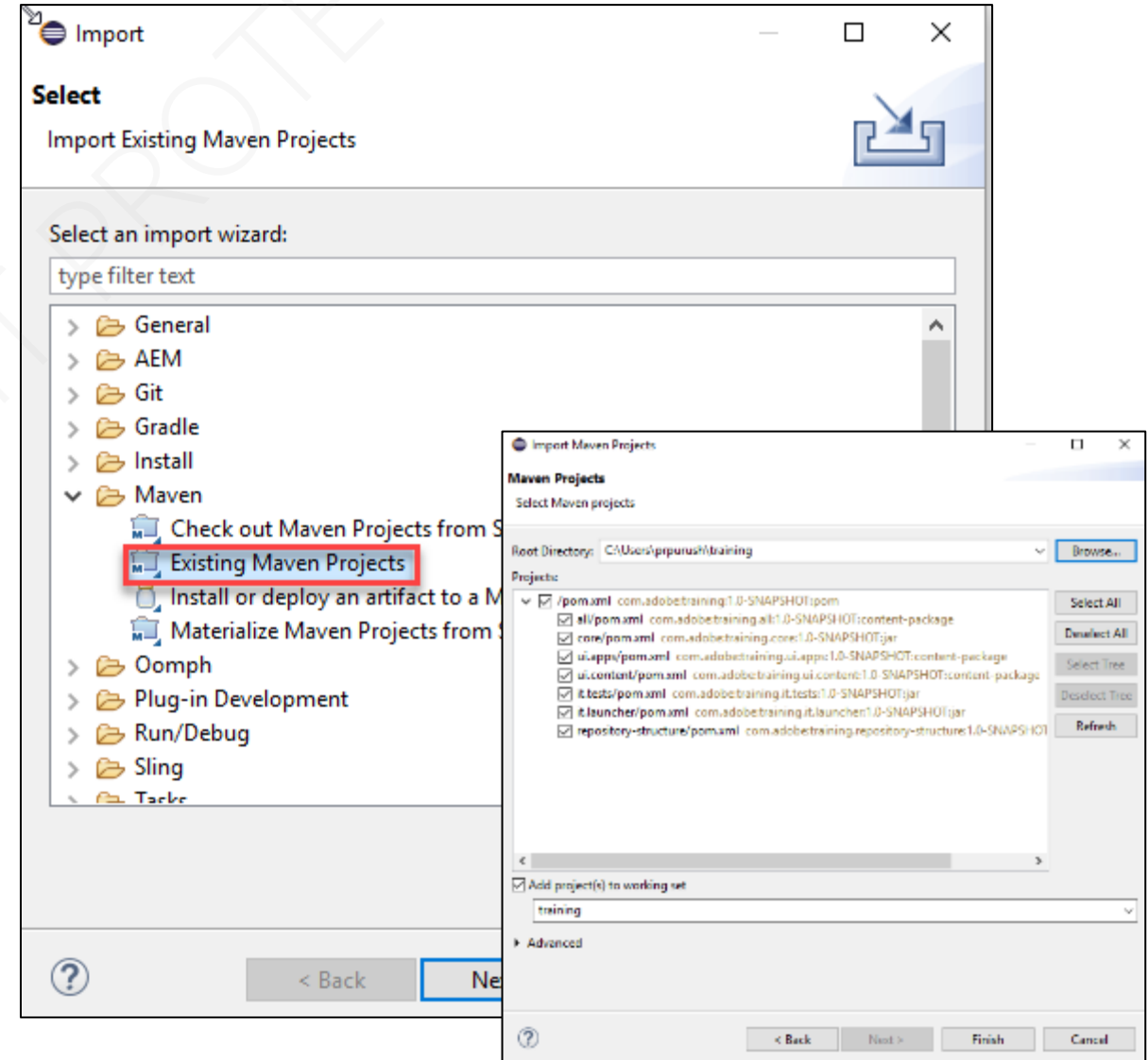
## Exercise 2 - Install and configure an Eclipse AEM plug-in (Local only)

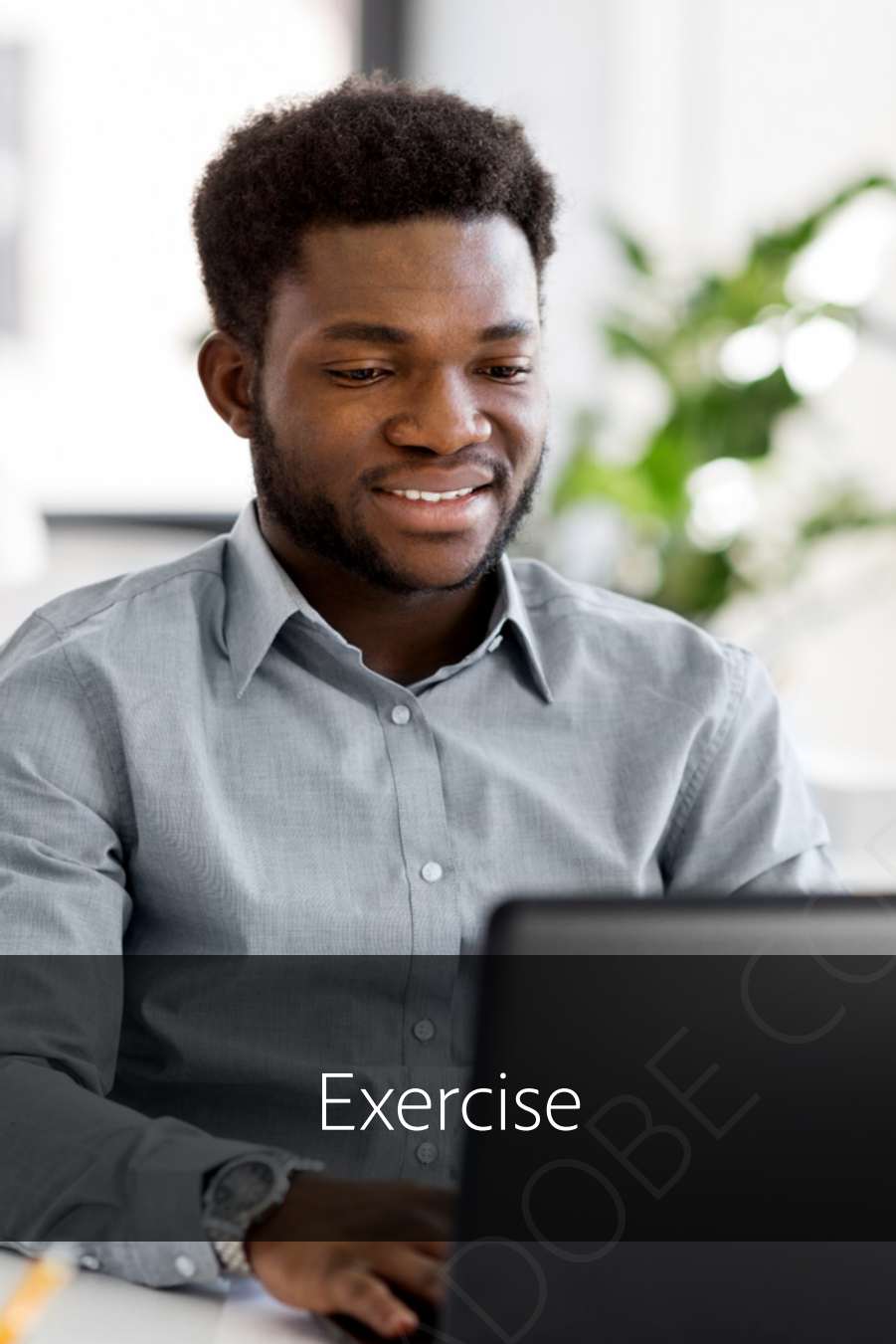
ADLS Readytech environments already have Eclipse installed

This exercise only needs completed if you are working locally

# Import a Maven project

1. In Eclipse, select File > Import from the menu.
2. Choose Existing Maven Projects in the Import wizard.
3. Browse and select the project you want to import.





## Exercise

### Exercise 3 - Import an AEM project in Eclipse

# Maven Use | Options

Deploying an entire project or module requires Maven. Two common options

- Maven command line outside Eclipse

```
[INFO] wetrain ..... SUCCESS [ 0.308 s]
[INFO] We.Train - Core ..... SUCCESS [ 4.234 s]
[INFO] We.Train - Repository Structure Package ..... SUCCESS [ 0.619 s]
[INFO] We.Train - UI apps ..... SUCCESS [ 3.957 s]
[INFO] We.Train - UI content ..... SUCCESS [ 1.594 s]
[INFO] We.Train - All ..... SUCCESS [ 6.018 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

- Maven Plugin



*For training purposes, maven command line will be used in training*



# M2Eclipse plugin

## Features of M2Eclipse

- Launch Maven builds from within Eclipse
- Dependency management for Eclipse build path based on Maven's pom.xml
- Resolve Maven dependencies from the Eclipse workspace without installing
- Automatic downloading of the required dependencies from the remote Maven repositories
- Quick search for dependencies in Maven remote repositories



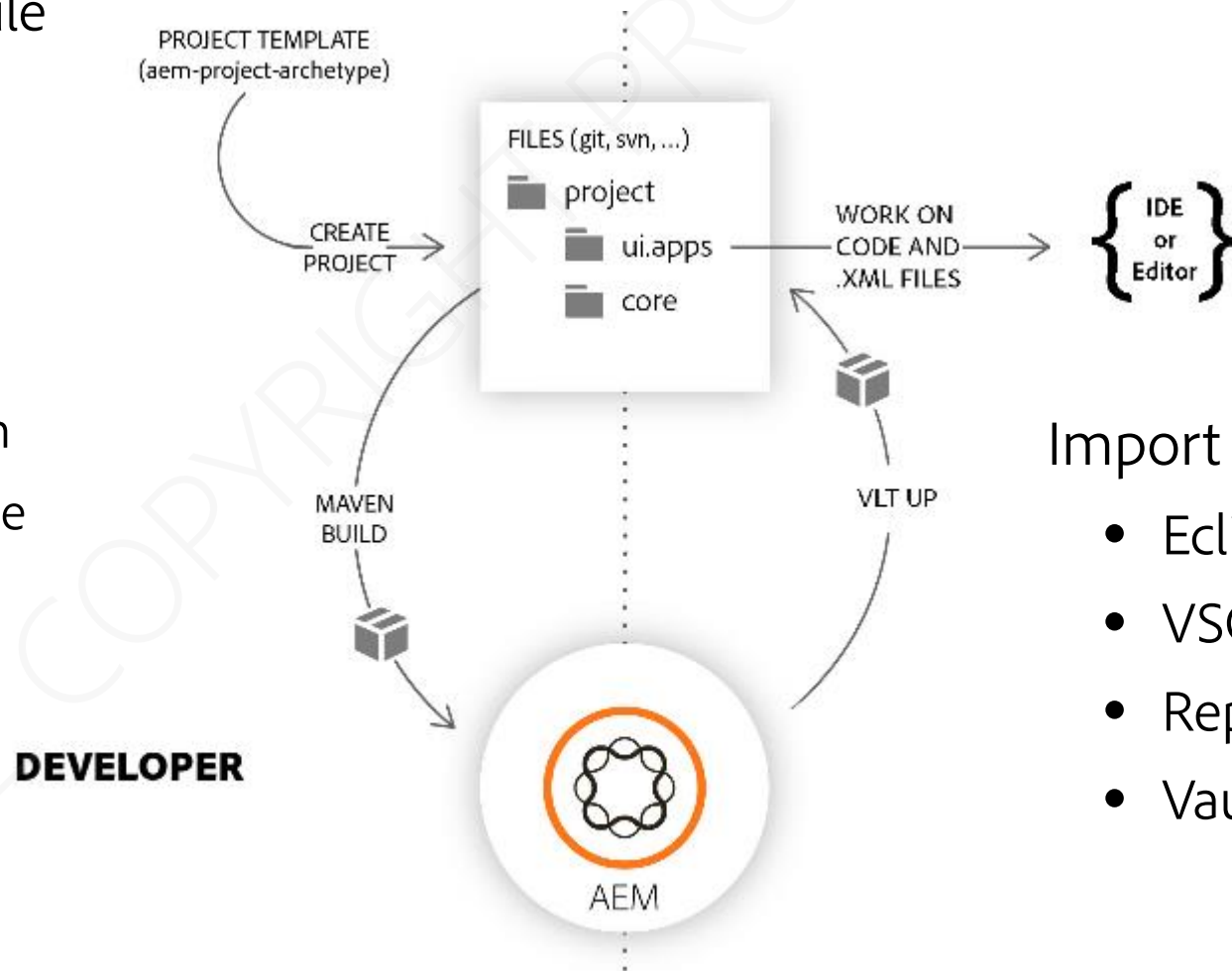
**M2Eclipse:** First-class Apache Maven support in the Eclipse IDE

*For training purposes, maven command line will be used in class*

# AEM Project Sync | Common Options

## Export to AEM

- Entire project or module
  - Maven command
  - M2Eclipse extension
- Individual Files
  - Eclipse sync plugin
  - VSCode sync extension
  - AEMFED command line
  - Repo command line



## Import from AEM

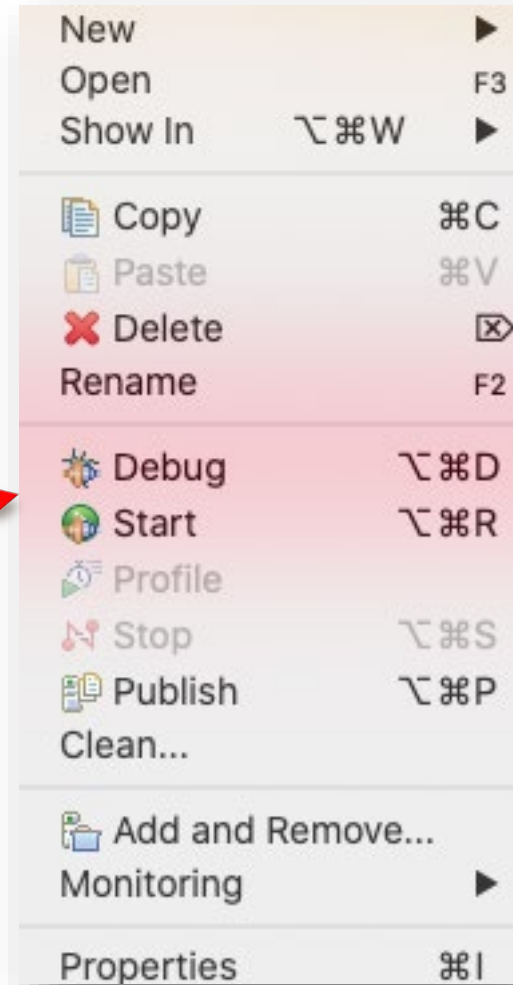
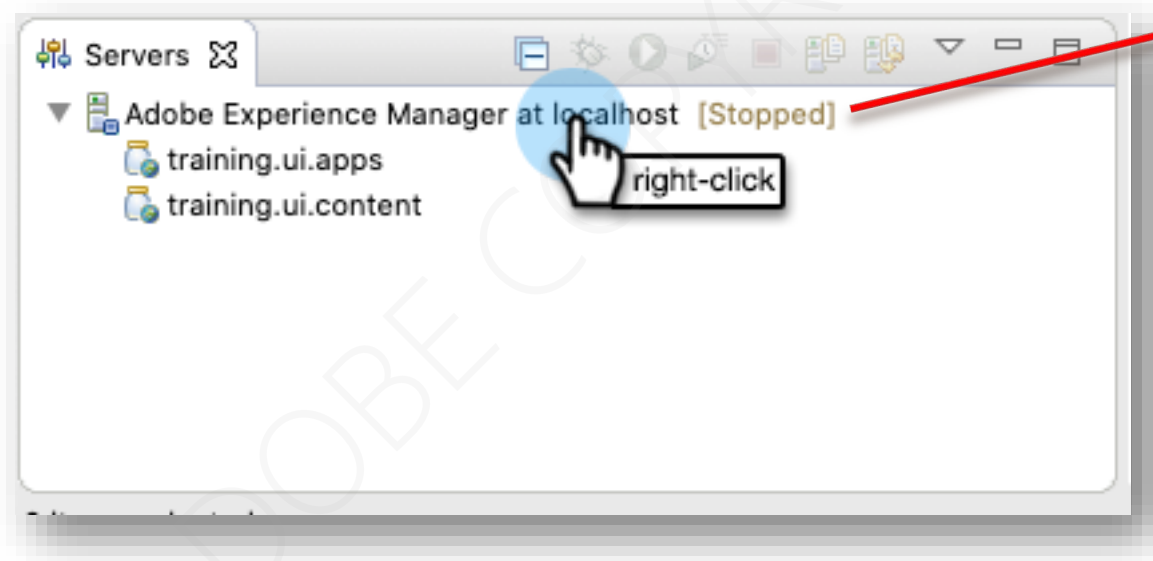
- Eclipse sync plugin
- VSCode sync extension
- Repo command line
- Vault command line



# Eclipse Sync Plugin | AEM Server

Server connection to an AEM server

- Choose different modules in different projects to sync
- Auto sync individual files to AEM on save
- Manual sync from AEM
- Start/stop/debug options



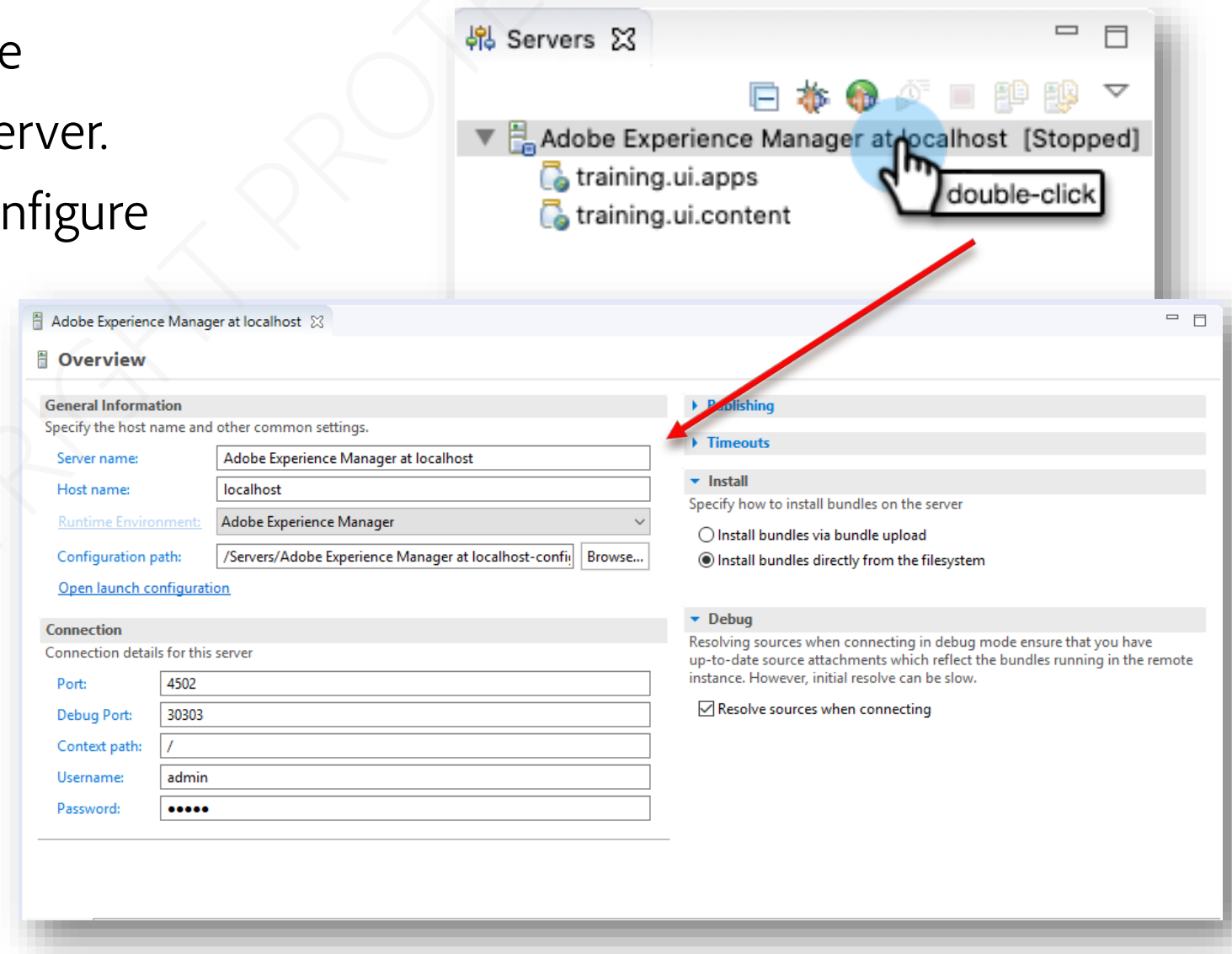
# AEM Server | Configure Options

Change to AEM perspective In Eclipse

On the Servers tab, define the new server.

Double click on the new server to configure

- Port
- Debug port
- Username/password
- Host
- Timeouts
- Auto publish rules



# AEM Server | Filter rules

Filter rules determine what can sync with AEM

- Applies to Maven
- Applies to the AEM Sync plugin

Defined in filter.xml files within a module

- `src/main/content/META-INF/filter.xml`

```
filter.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <workspaceFilter version="1.0">
3   <filter root="/apps/trainingproject"/>
4   <filter root="/apps/sling" />
5 </workspaceFilter>
6
```

Example filter for ui.apps

```
filter.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <workspaceFilter version="1.0">
3   <filter root="/conf/trainingproject" mode="merge"/>
4   <filter root="/content/trainingproject" mode="merge"/>
5   <filter root="/content/dam/trainingproject" mode="merge"/>
6 </workspaceFilter>
7
```

Example filter for ui.content

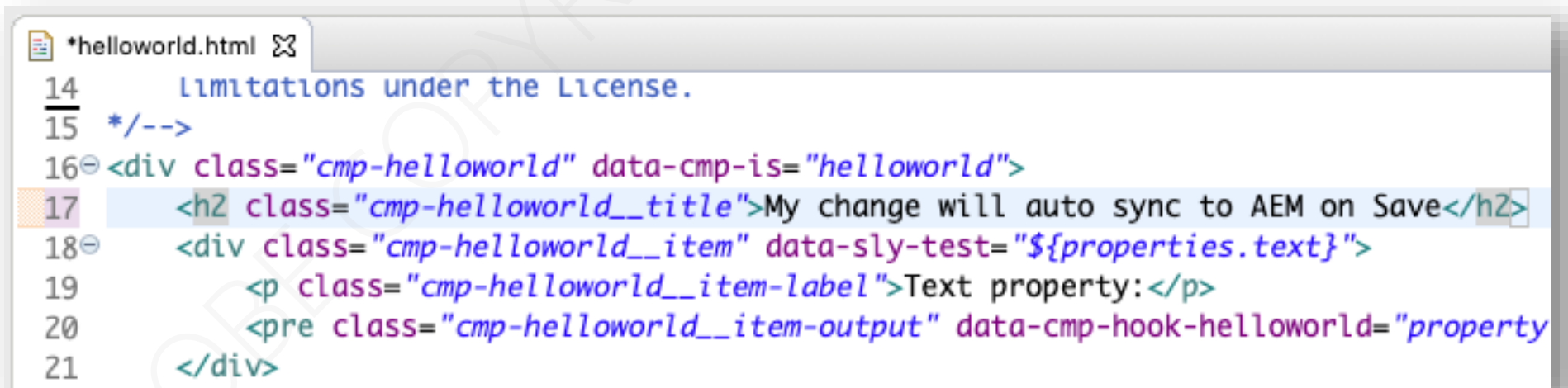
# AEM Server | Auto Sync Files on Save

Change to AEM perspective In Eclipse

Make sure you are using virtual JCR view

Make sure the AEM Server is started

- AEM server connections are set to auto sync by default



```
*helloworld.html
14      limitations under the License.
15  */-->
16  <div class="cmp-helloworld" data-cmp-is="helloworld">
17    <h2 class="cmp-helloworld__title">My change will auto sync to AEM on Save</h2>
18    <div class="cmp-helloworld__item" data-sly-test="${properties.text}">
19      <p class="cmp-helloworld__item-label">Text property:</p>
20      <pre class="cmp-helloworld__item-output" data-cmp-hook-helloworld="property
21    </div>
```

# AEM Server | Import from Server

Change to AEM perspective In Eclipse

Make sure you are using virtual JCR view

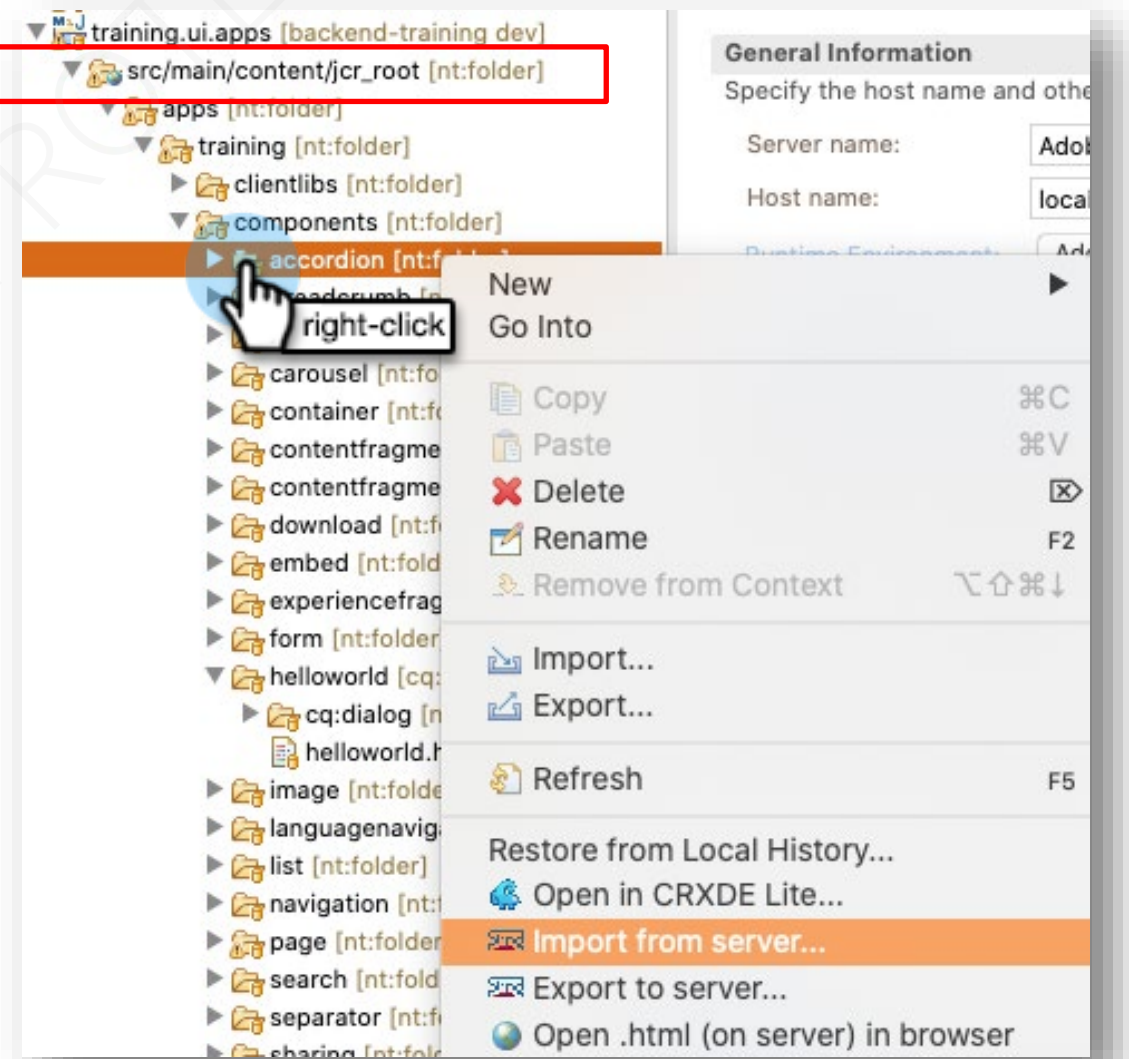
Make sure the AEM Server is started

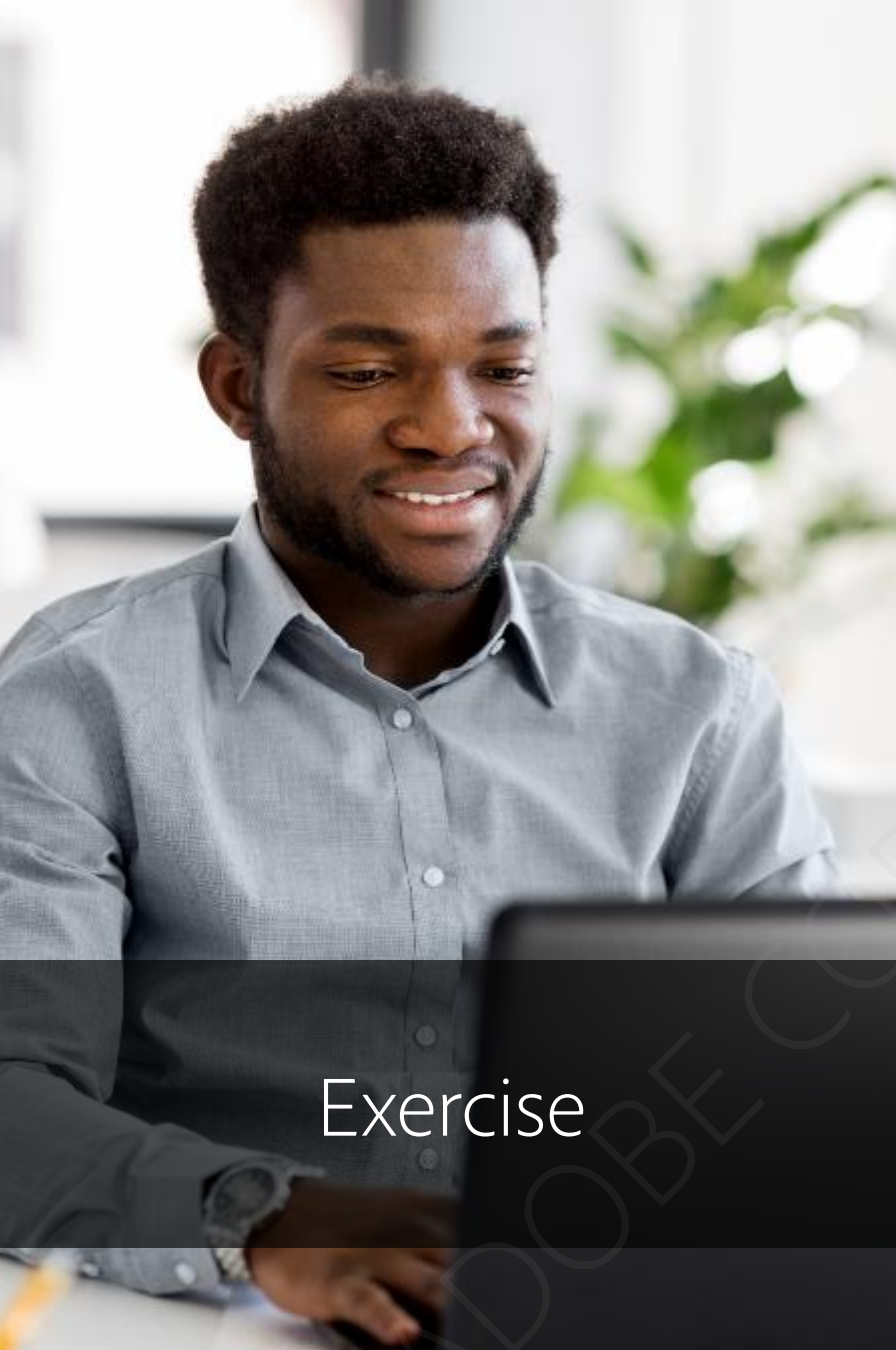
Commonly used for CRXDE Lite Activities

- Dialogs structures reused
- Overlay nodes from /libs

Importing skeleton content back to the project

- Pages, experience fragments, assets
- Configurations





## Exercise

### Exercise 4: Synchronization Tools for Eclipse

Tasks to perform:

1. Configure the AEM server
2. Sync AEM content





## Knowledge Check

Which of the following tools are used to configure the AEM development environment? (Select the three correct answers.)

- A. Maven ✓
- B. Eclipse ✓
- C. Apache Ant
- D. AEM Plug-in for Eclipse ✓