

Creating a project using Maven

Objectives:

- Describe Maven
- Install Maven command
- Create project using Maven
- Work with Maven project



Maven

Open-source development tool used to build and manage Java-based projects.

- Describes how project is built, plugins needed, as well as its dependencies
- Open-source artifacts are available on mavenrepository.com
- Adobe artifacts are available on repo.adobe.com

Advantages

- Build process is easier (after the project is set up properly)
- Provides a uniform build system
- Provides quality project information
- Provides guidelines for development best practices
- Allows transparent migration to new features

Project Object Model (POM) files are used as instructions for the Maven project

Understand POM Files

POM Files

- Named pom.xml in the project structure
- Exist at the project level (parent) and module level (sub-directories)
- Identify all project artifacts and their dependency hierarchy

A POM file contains:

- General project information
- Build settings
- Build environment
- POM relationships and dependencies
- Snapshot versions
- Property references
- Plug-ins

Sample POM file

```
<?xml version="1.0"?>
<project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://maven.apache.org/POM/4.0.0">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.adobe.training</groupId>
  <artifactId>parent</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <properties>
    <company.name>Training Portal</company.name>
    <cq.host>localhost</cq.host>
    <cq.port>4502</cq.port>
    <cq.user>admin</cq.user>
    <cq.password>admin</cq.password>
    <module.prefix>company</module.prefix>
  </properties>
  <packaging>pom</packaging>
  <name>Company Portal</name>
  <modules>
    <module>${module.prefix}-core</module>
    <module>${module.prefix}-ui</module>
  </modules>
  <build>
    <pluginManagement>
      <plugins>
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
```

Using Maven for AEM Projects

Packages the Single Package artifact

- Single Container for all bundles, content packages and 3rd party artifacts
- Author and Publish maven profiles for installation

Builds OSGi bundle artifacts

- Annotation driven java classes for OSGi
- Development maven profile for installation

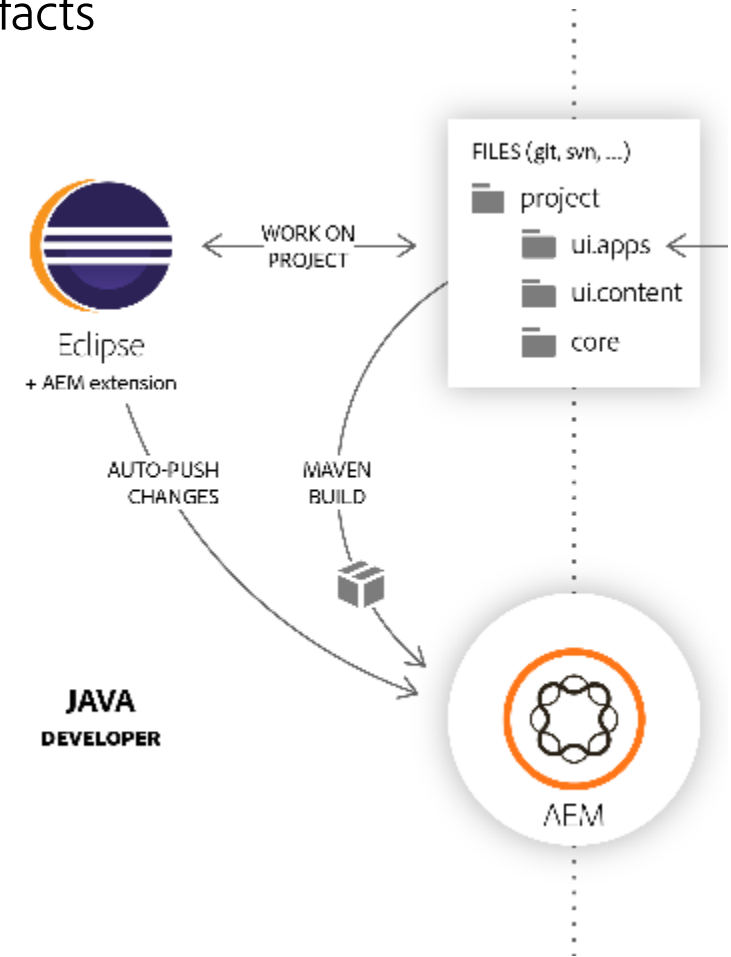
Builds Content Packages artifacts

- Files and folders packaged for AEM
- Development maven profile for installation

Manages artifacts for the project

- Dependencies
- Plugins
- AEM SDK (CS) or UberJar (6.5 and older)

Unit Tests and Sling Tests





Exercise

Exercise 1: Install Maven command

If you are using Readytech, you can skip this exercise.

Tasks to perform:

1. Install Maven
2. Configure Environment Variables
3. Install Node.js

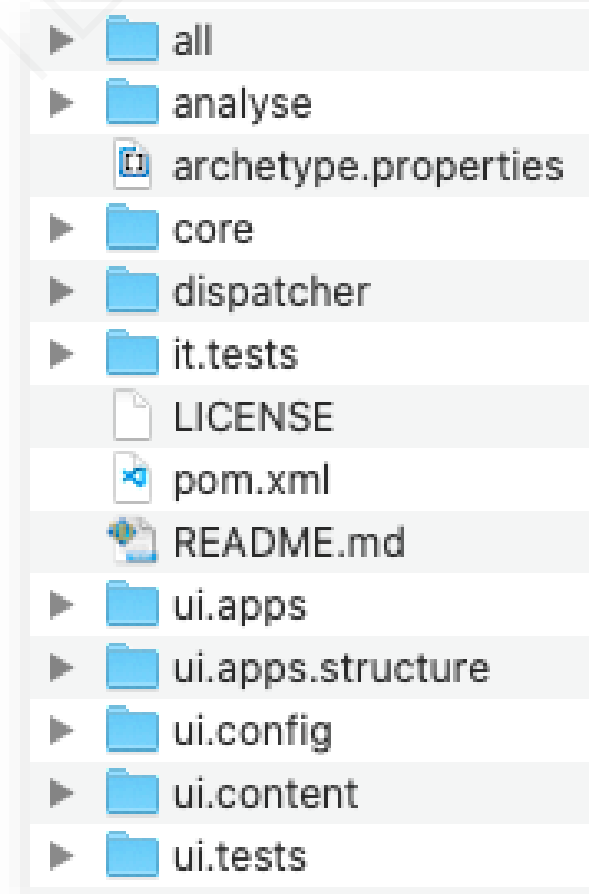
AEM Archetype

Used for rapid AEM project creation

Cloud Ready

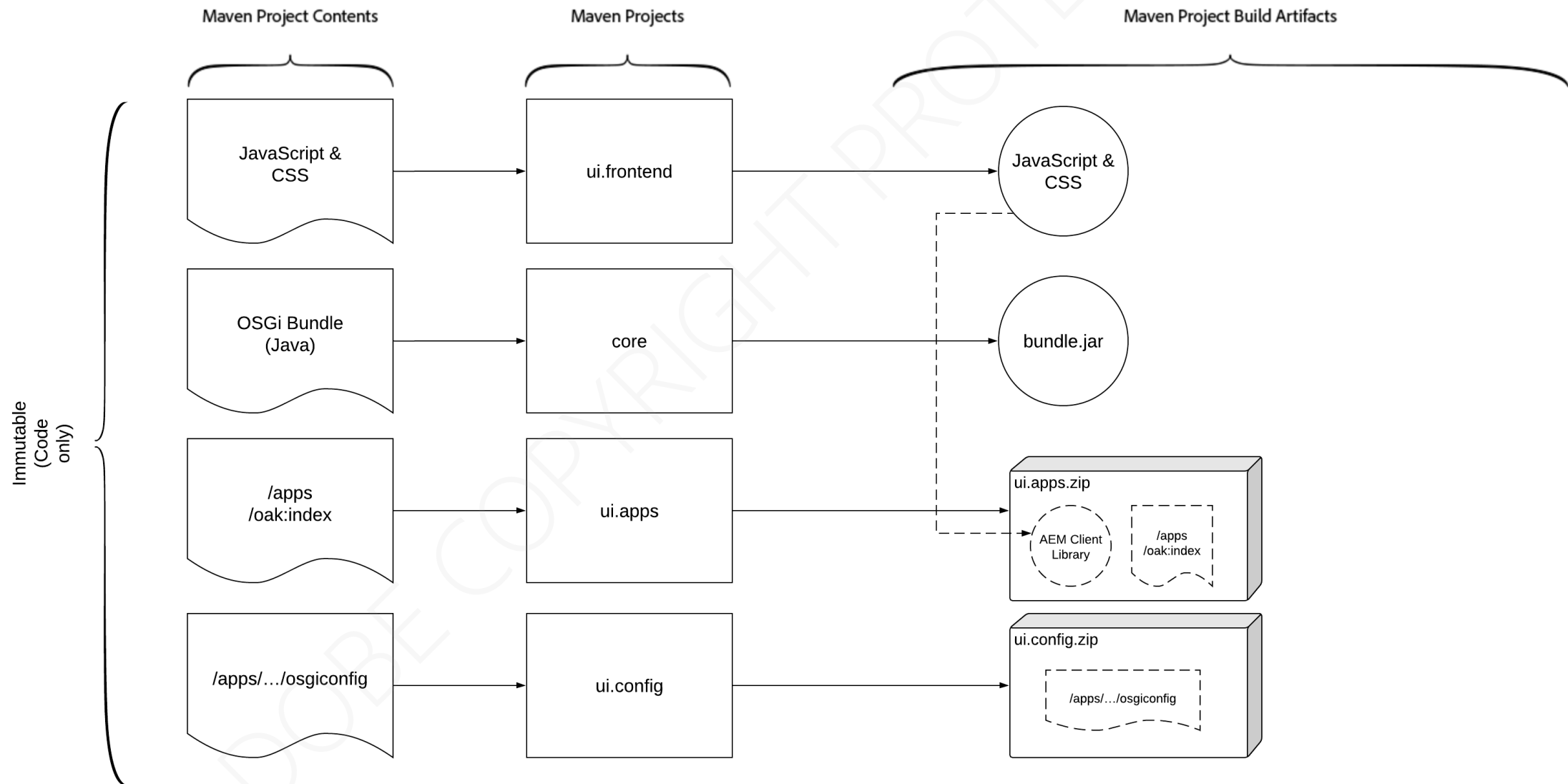
Standard for best practice

- Mutable/immutable separation
- Core Components
- Multi-Site
- Editable Templates
- XF header/footer concept
- Java example
- Dispatcher for Managed Services or Cloud Service
- Forms, Commerce, Webapp, Dynamic Media ready



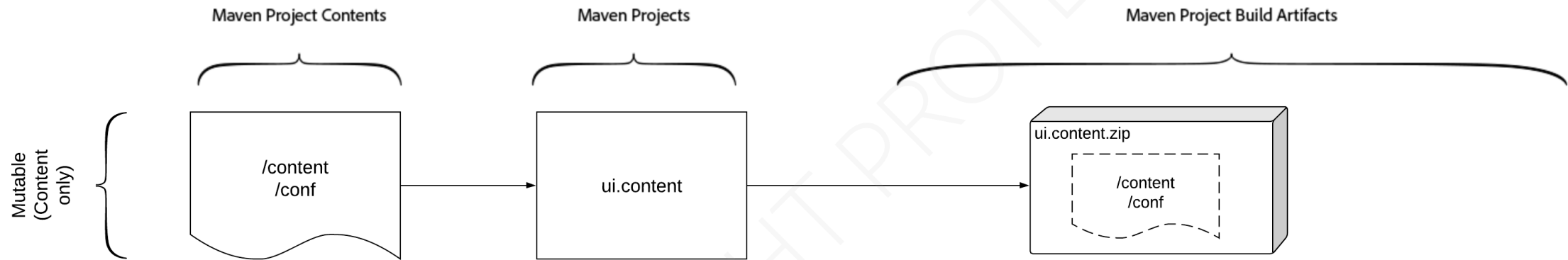
<https://github.com/adobe/aem-project-archetype#features>

AEM Archetype | Immutable Modules



<https://experienceleague.adobe.com/docs/experience-manager-cloud-service/assets/content-package-organization.png>

AEM Archetype | Mutable Modules

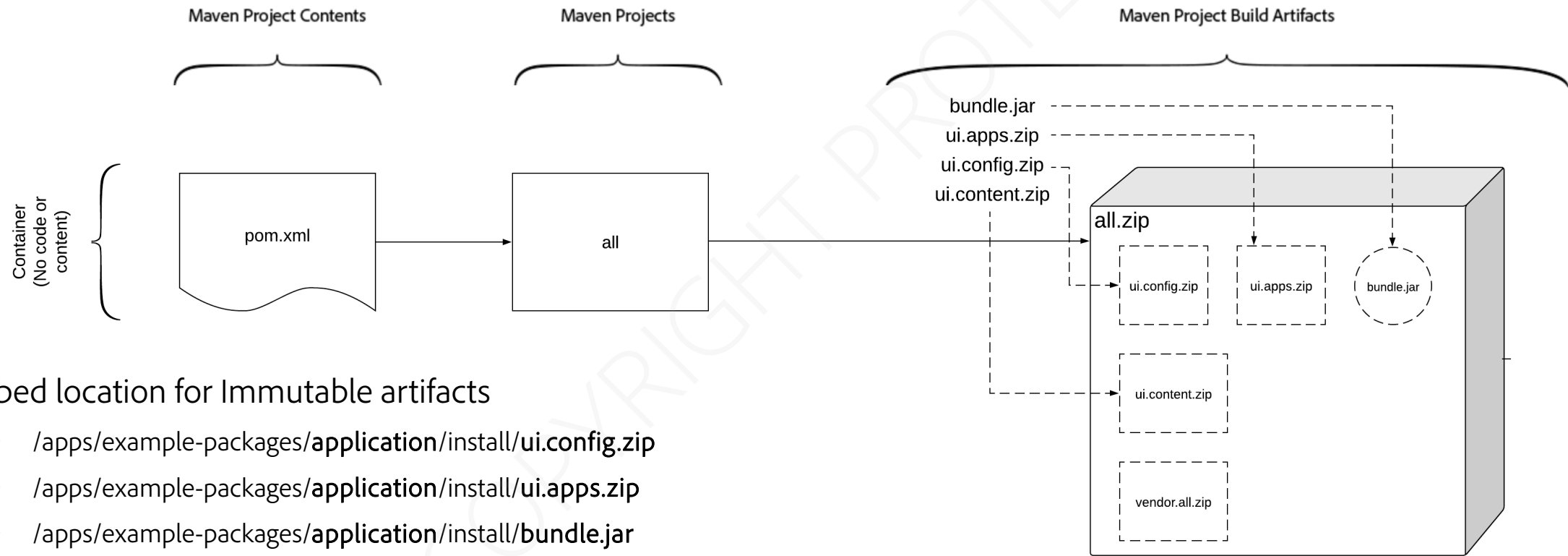


Mutable content in a project can be

- Minimal content for the project to build
- Initial templates and configurations
- Skeleton site structure that components depend on
- Initial folder structures for Experience Fragment and Assets
- Initial Content Fragment Modules

<https://experienceleague.adobe.com/docs/experience-manager-cloud-service/assets/content-package-organization.png>

AEM Archetype | Container Module



Embed location for Immutable artifacts

- `/apps/example-packages/application/install/ui.config.zip`
- `/apps/example-packages/application/install/ui.apps.zip`
- `/apps/example-packages/application/install/bundle.jar`

Embed location for Mutable artifacts

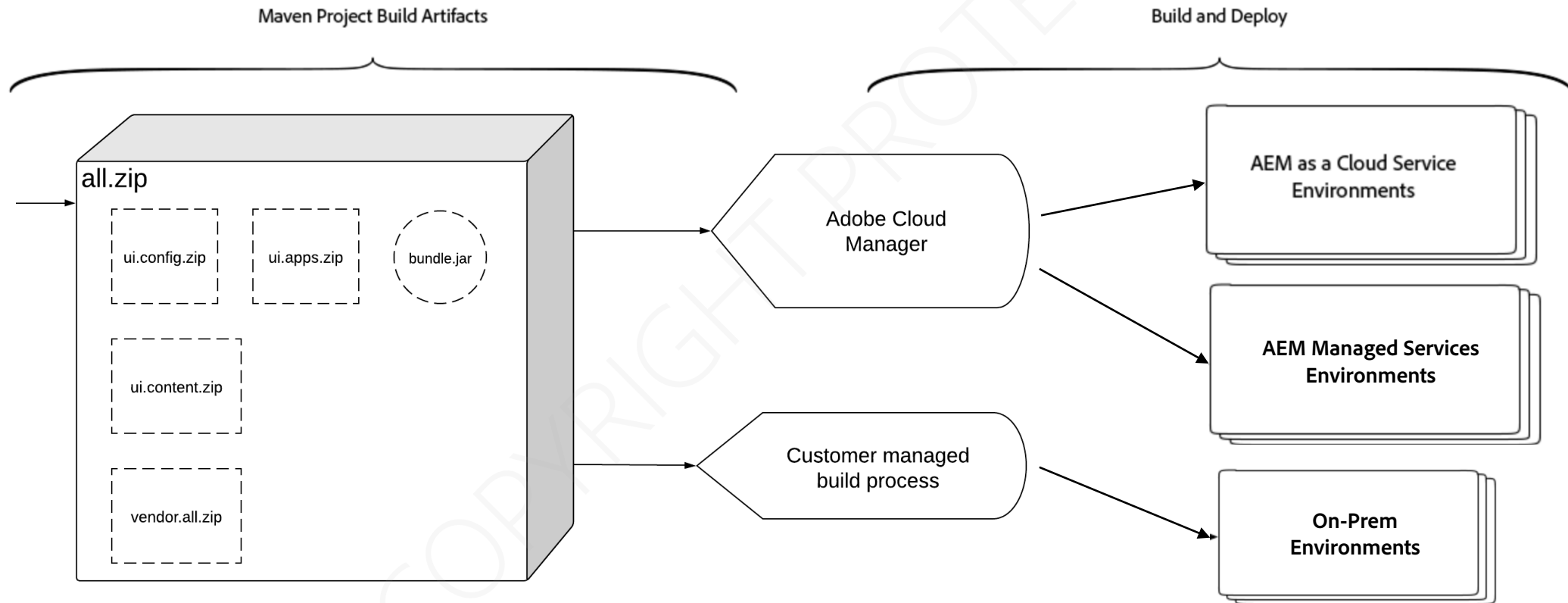
- `/apps/example-packages/content/install/ui.content.zip`

Embed location for 3rd party artifacts

- `/apps/example-vendor-packages/container/install/vendor.all.zip`

<https://experienceleague.adobe.com/docs/experience-manager-cloud-service/assets/content-package-organization.png>

Build and Deploy | Dev, Stage, Prod Environments



The **all** artifact is a single container package that goes through the CI/CD pipeline within Cloud Manager

- Cloud Manager then deploys the code to AEM environments

On-Prem deployments can also use the **all** container package

Build and Deploy | Local Development

Maven Project Build Artifacts

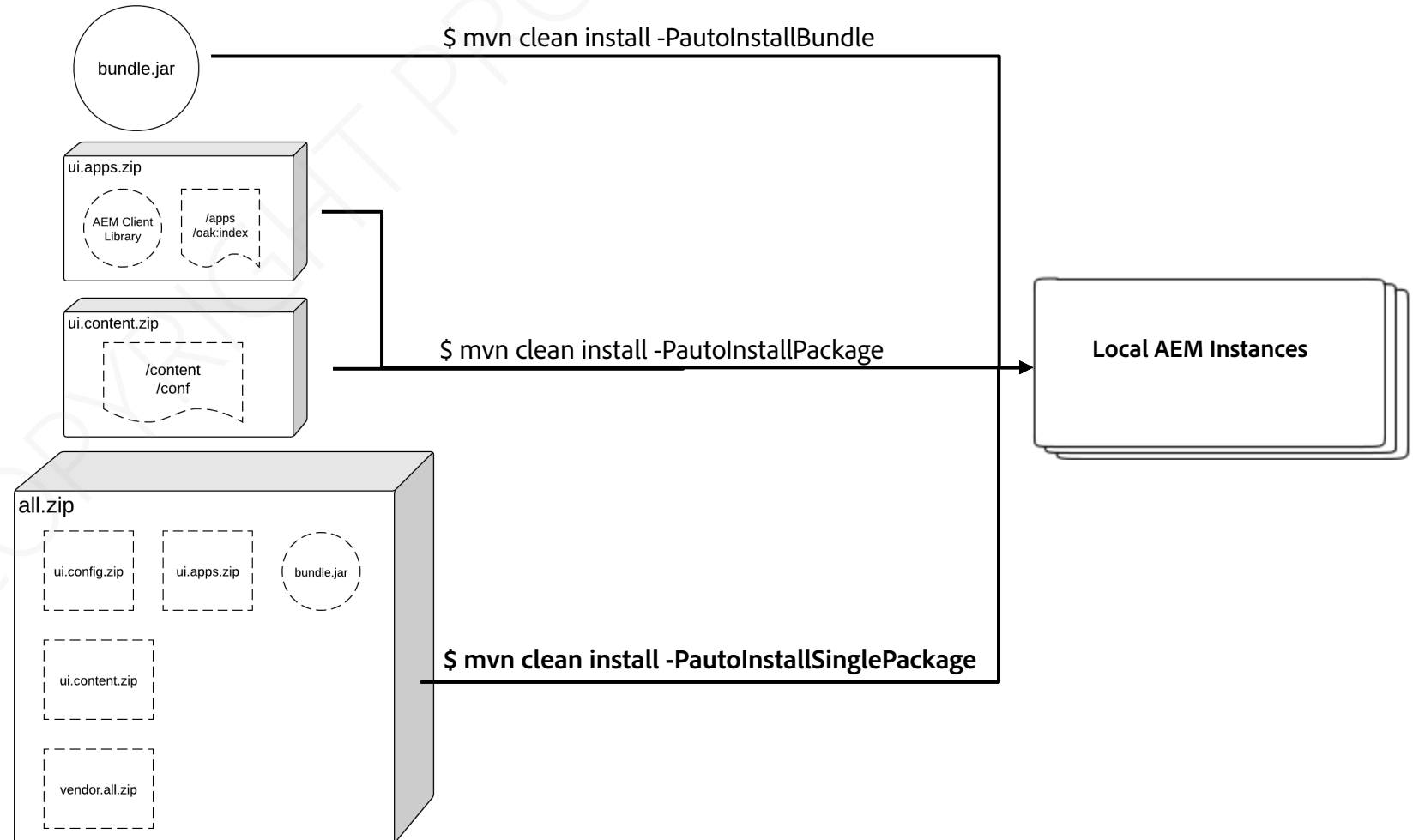
Build and Deploy

Developer Profiles

- Useful for targeted development
- Defined in parent/pom.xml

All Profile

- Installs all project artifacts
- Installs 3rd party artifacts
- Defined in all/pom.xml



Other maven modules

ui.apps.structure

- Defines the JCR repository roots this project uses
- Ensures installation of packages are ordered by JCR resource dependencies
- Generally, the default repository-structure module should handle most use cases in a project

it.tests

- Container for sling testing on an AEM instance

it.launcher

- Build it.tests with testing bundles into an AEM instance

Optional archetype inclusions

The archetype provides many different modules and configurations for a wide variety of AEM projects

- React or Angular frontend module
- Commerce Integration Framework (CIF) core components
- AEM Forms add-on
- Adobe Data layer
- AMP support
- Core components enabled with Dynamic Media
- Example Core Component Library

<https://github.com/adobe/aem-project-archetype#available-properties>

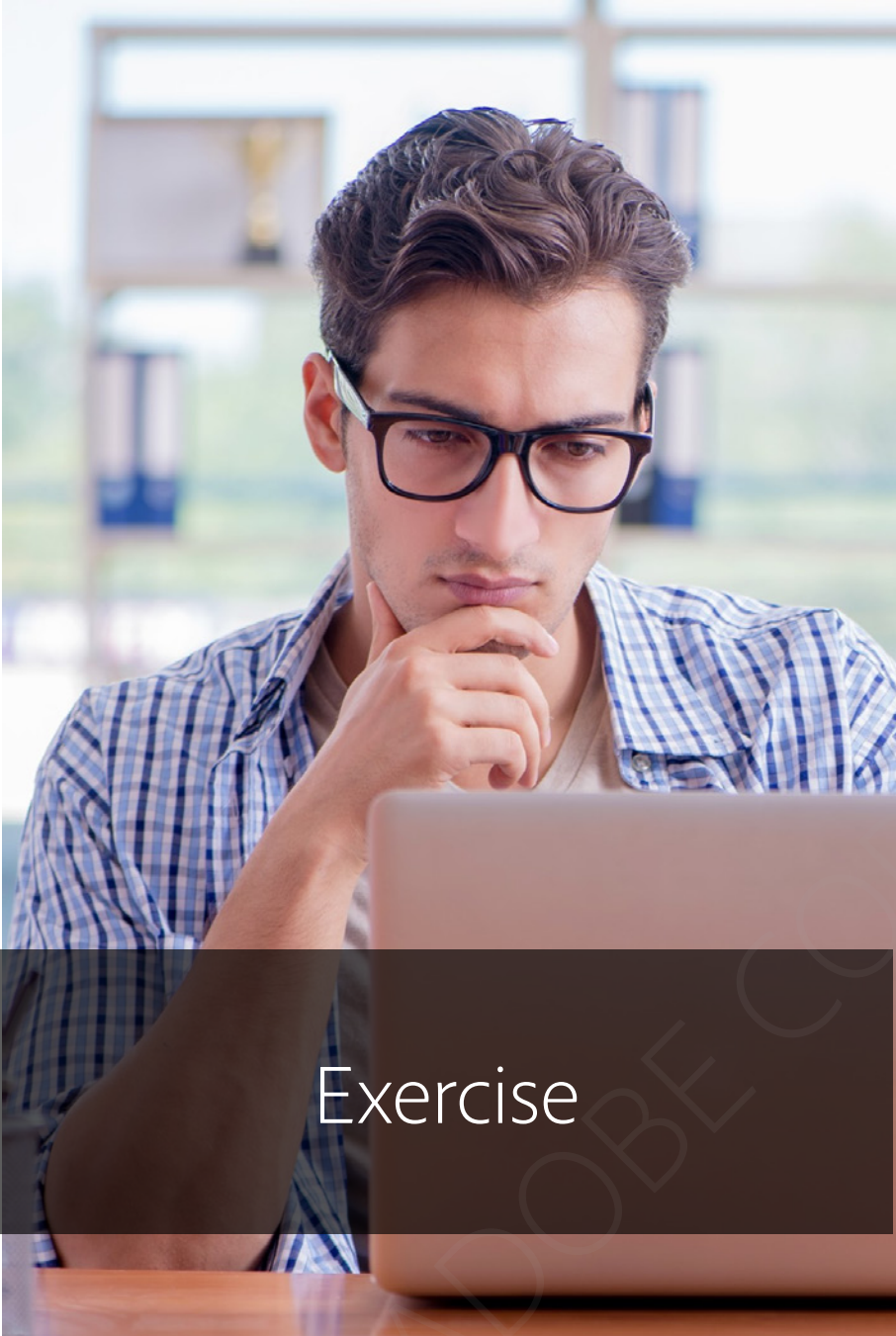
Create a Project with Maven

- Maven needs to be installed
- Adobe's nexus repo needs added to
username/.m2/settings.xml
- Project can then be installed into AEM

```
$ mvn clean install -Padobe-public  
-PautoInstallSinglePackage
```

```
mvn -B archetype:generate ^  
-D archetypeGroupId=com.adobe.aem ^  
-D archetypeArtifactId=aem-project-archetype ^  
-D archetypeVersion=25 ^  
-D appTitle=TrainingProject ^  
-D appId=training ^  
-D artifactId=training ^  
-D groupId=com.adobe ^  
-D package=com.adobe.training ^  
-D version=1.0-SNAPSHOT ^  
-D aemVersion=cloud ^  
-D sdkVersion=latest ^  
-D includeDispatcherConf ^  
-D frontendModule=none ^  
-D language=en ^  
-D country=us ^  
-D singleCountry=y ^  
-D includeExamples=y ^  
-D includeErrorHandler=n ^  
-D includeCommerce=n ^  
-D commerceEndpoint= ^  
-D includeForms=n ^  
-D sdkFormsVersion= ^  
-D datalayer=y ^  
-D amp=n ^  
-D enableDynamicMedia=n
```

Minimum number
of params needed



Exercise

Exercise 2: Synchronization tools for Eclipse

Tasks to perform:

1. Configure the profile for Maven
2. Create a project using Maven



Demonstration

Explore the POM files in the training project

training/pom.xml and observe

- global properties
- Dependencies, plugins, and profiles

training/all/pom.xml

- Observe the embedded content packages
- Observe the autoInstallSinglePackage profile

training/ui.apps.structure/pom.xml

- Observe the typical content roots defined

AEM's Dependency Jar

Special JAR file provided by Adobe to reduce the number of dependencies

Jar Contents:

- All public Java APIs exposed by AEM.
- Limited external libraries—all public APIs available in AEM that comes from Apache Sling, Apache Jackrabbit, Apache Lucene, Google Guava, and two libraries used for image processing.
- Interfaces and classes exported by an OSGi bundle in AEM.
- MANIFEST.MF file with the correct package export versions for all the exported packages.

Naming

- AEM 6.5 – Uberjar API
- Cloud Service - Java SDK API

Dependency Jar | Uberjar

AEM 6.5 and lower Only

AEM dependency is called the Uberjar

- Use the same AEM version and service pack

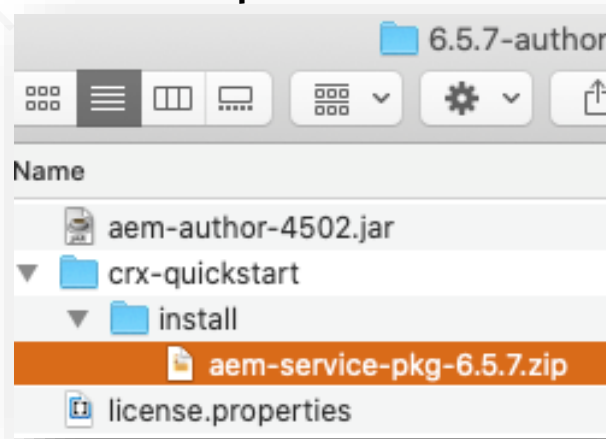
Javadocs can be downloaded using

- \$ mvn dependency:resolve -Dclassifier=Javadoc

Dependency version should be updated based on:

- Service Pack used

Example AEM 6.5.7 install



Maven Dependency element (parent)

```
<dependency>
  <groupId>com.adobe.aem</groupId>
  <artifactId>uber-jar</artifactId>
  <version>6.5.7</version>
  <classifier>apis</classifier>
  <scope>provided</scope>
</dependency>
```

Maven Dependency element (modules)

```
<dependency>
  <groupId>com.adobe.aem</groupId>
  <artifactId>uber-jar</artifactId>
  <classifier>apis</classifier>
</dependency>
```

Dependency Jar | SDK API

AEM dependency is called the Java SDK API

- Use the same version as the AEM SDK downloaded

Javadocs can be downloaded using

- \$ mvn dependency:resolve -Dclassifier=javadoc

Optional download of deprecated APIs from 6.5

- Artifact=aem-sdk-deprecated-6.5-api

Example SDK Zip file



Maven Dependency element (parent)

```
<dependency>
  <groupId>com.adobe.aem</groupId>
  <artifactId>aem-sdk-api</artifactId>
  <version>2020.01.1850.20200109T110957Z-191201</version>
  <scope>provided</scope>
</dependency>
```

Maven Dependency element (modules)

```
<dependency>
  <groupId>com.adobe.aem</groupId>
  <artifactId>aem-sdk-api</artifactId>
</dependency>
```

Refresh AEM SDK | Quickstart and API

Cloud Service Only

Cloud Service runs on continuous releases


- Daily maintenance releases will have minimal changes in the SDK
- Monthly maintenance releases will typically have more impactful changes

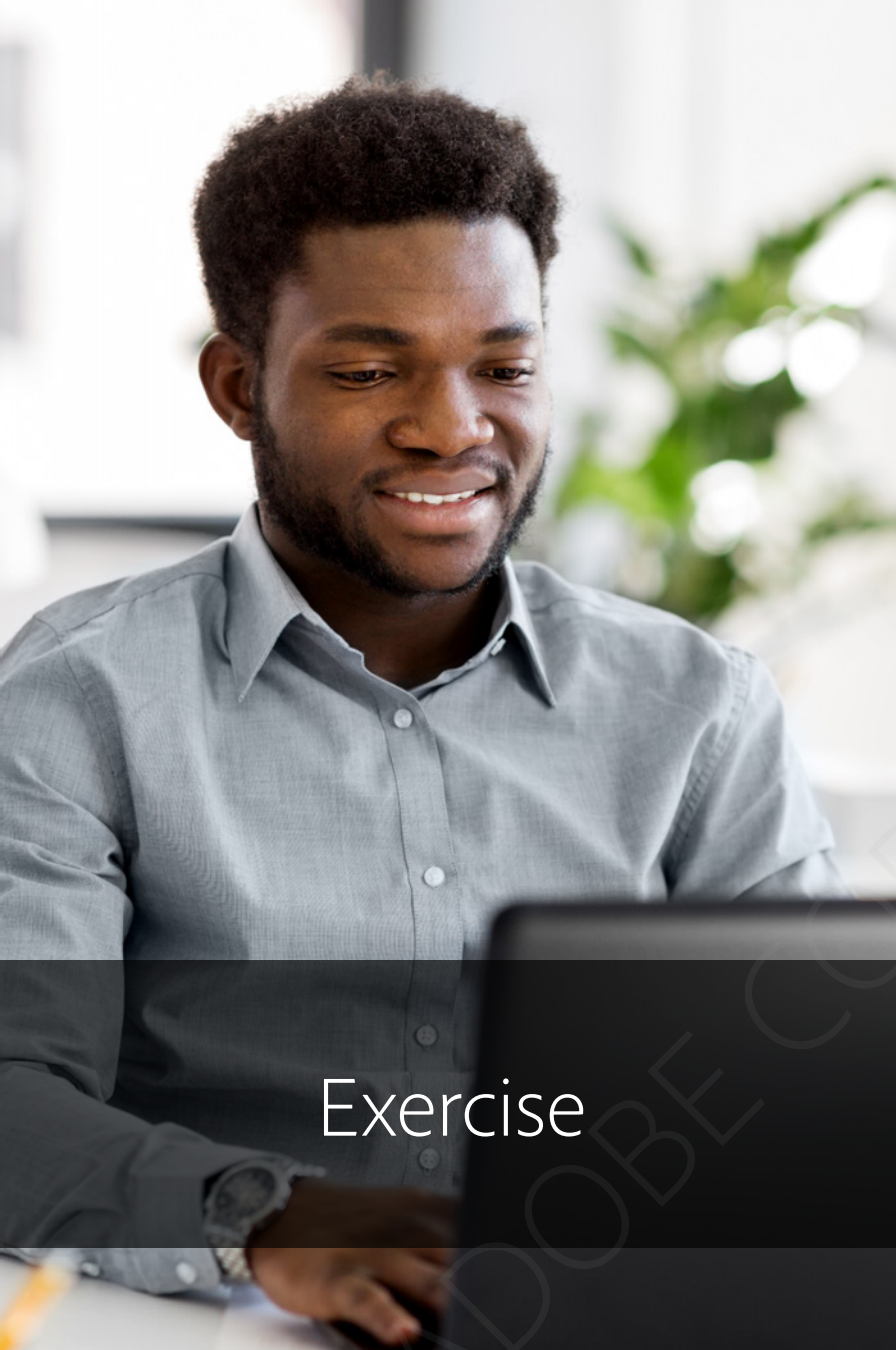
Adobe recommends refreshing the SDK at least after a monthly maintenance release

- When you download a new SDK, update your POM with the same release number

Optionally use same release as Production service

- Can easily be found in Cloud Manager

ENVIRONMENTS	STATUS
vlab7-us-adls-prod https://author-cm-p3955-e9924-ns-tea... AEM RELEASE: 2019.11.1401.20191121T204747Z REGION: EAST US	 RUNNING



Exercise

Exercise 3: Install the project into AEM

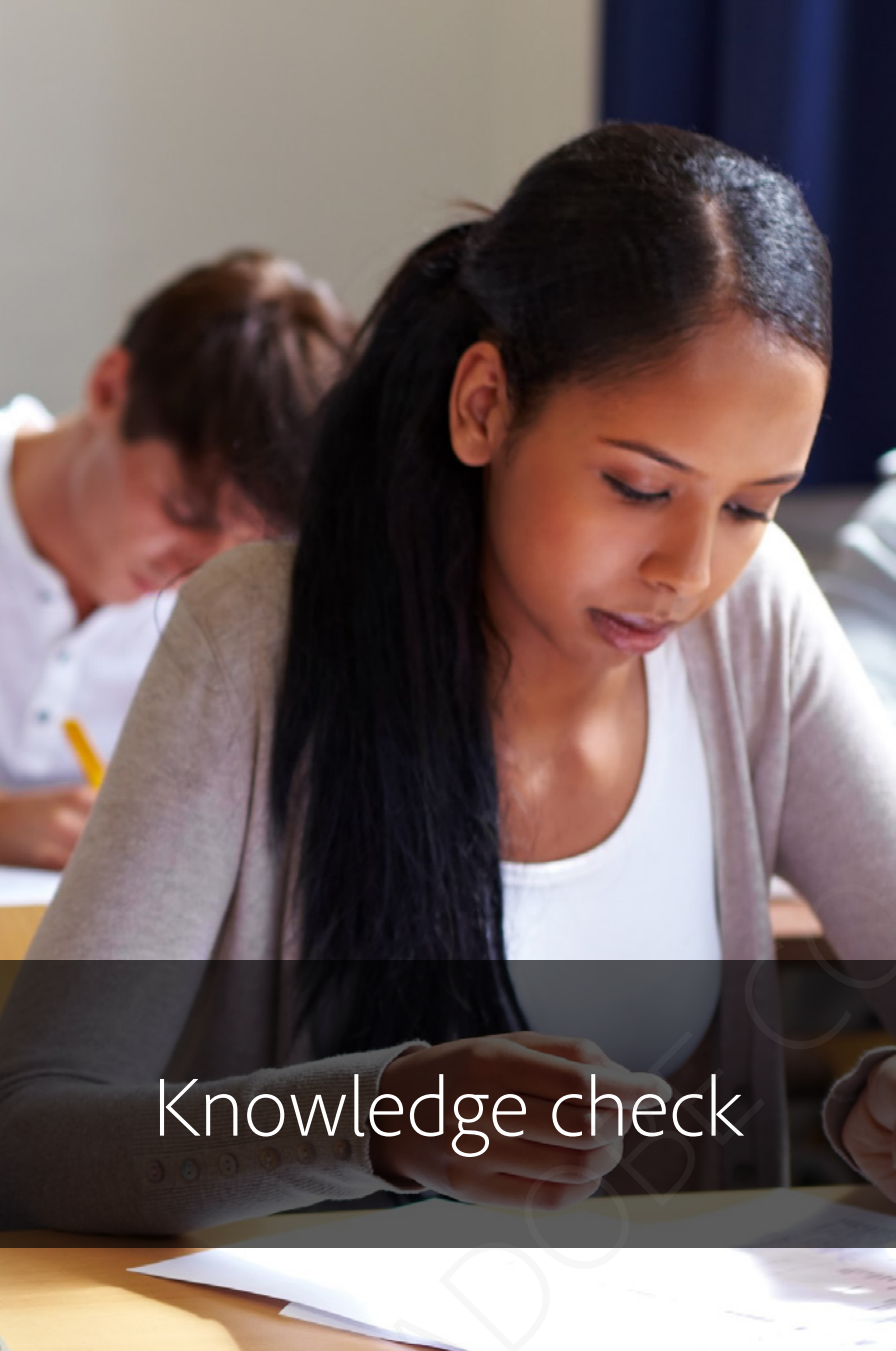
Tasks to perform:

1. Update the POM file
2. Install the project into AEM
3. Verify the installed content packages



Key takeaways

- Maven
 - Open source development tool used to build and manage Java-based projects.
 - Configured using Project Object Model (POM) files.
- AEM Archetype
 - Starter AEM project based on best practices
 - The archetype can be used for a wide variety of AEM projects
 - The All container package include all other artifacts
 - AEM dependencies are managed with the SDK API or Uberjar



Knowledge check

Which maven modules are created as part of the AEM Archetype? Select three answers.

- A. ui.apps.structure ✓
- B. code
- C. ui.apps ✓
- D. ui.content ✓