Day 2 - Tasks

**1. What is the purpose of the core module in AEM?**

The core module is where all the **backend logic** resides. It typically contains **Java code** for business logic, **Sling Models**, **OSGi services**, and **servlets**. Think of it as the brain of your AEM application, where the heavy processing happens before displaying content to users.

**2. What kind of files and code can be found in the core folder?**

In the core folder, you'll find:

* **Java Classes**: Business logic, Sling Models, and OSGi services.
* **Bundle Configurations**: bnd.bnd file for setting bundle metadata.
* **POM.xml**: Maven configuration for building the core module.
* **Resources**: META-INF and OSGI-INF folders for service configurations.

**3. Explain the role of ui.apps in AEM projects.**

The ui.apps module holds all the **content that is deployed to the repository** (/apps folder in CRXDE). It includes:

* **Components**: The building blocks of pages.
* **Client Libraries**: CSS, JS, and assets.
* **Templates & Dialogs**: Configurations for authoring.
* **Sling Mappings**: Resource types and paths.

**4. How are components structured in the ui.apps folder?**

Components in ui.apps are structured as:

/apps/<project\_name>/components/<component\_name>

Each component typically contains:

* HTML (HTL files)
* JS/CSS (clientlibs)
* Dialogs (authoring interface)
* Sling Model or JavaScript logic (if needed)

**5. Hello World Component:**

**• Where is the Hello World component located in both core and ui.apps?**

* In **core**, the Java class might be in:

/core/src/main/java/com/project/models/HelloWorldModel.java

* In **ui.apps**, the component's files are in:

/ui.apps/src/main/content/jcr\_root/apps/<project\_name>/components/helloworld

**• Explain the Java class (in core) for the Hello World component.**

The Java class (HelloWorldModel.java) might look like this:

@Model(adaptables = Resource.class)

public class HelloWorldModel {

@Inject

private String message;

public String getMessage() {

return "Hello, " + message;

}

}

This class acts as a **Sling Model**, fetching data from the JCR repository.

**• How does the HTL script work in ui.apps for Hello World?**

The HTML file (helloworld.html) uses the Sling Model like:

<div data-sly-use.model="com.project.models.HelloWorldModel">

${model.message}

</div>

It pulls the data from the Java class and displays it in the component.

**• How are properties and dialogs defined for this component?**

* **Dialogs** are defined using .content.xml or cq:dialog files.
* They create a form-like interface for authors to enter data.
* These properties are stored in the JCR under /content.

**6. What are the different types of AEM modules (core, ui.apps, ui.content, etc.)?**

* **core**: Java code and backend logic.
* **ui.apps**: Components, templates, and client libraries (/apps folder).
* **ui.content**: Stores sample or starter content (/content folder).
* **ui.frontend**: Holds front-end build tools like Webpack, Node.js projects.
* **ui.tests**: Automated testing configurations.

**7. How does Maven build these modules?**

Maven uses the POM.xml to manage:

* **Modules**: Defines how each part of the project builds.
* **Dependencies**: Libraries needed by the project.
* **Build Plugins**: Tools to compile, package, and deploy AEM modules.

**8. Explain the build lifecycle of Maven in the context of AEM.**

Maven follows specific phases:

1. **clean**: Removes old builds.
2. **validate**: Checks if the project is correct.
3. **compile**: Converts source code to bytecode.
4. **test**: Runs unit tests.
5. **package**: Bundles code into JAR or ZIP.
6. **install**: Places the package into the local repository.
7. **deploy**: Deploys the package to the AEM instance.

**9. How are dependencies managed in pom.xml?**

In POM.xml, you define dependencies:

<dependencies>

<dependency>

<groupId>com.adobe.cq</groupId>

<artifactId>core.wcm.components.core</artifactId>

<version>2.17.0</version>

</dependency>

</dependencies>

Maven automatically downloads and manages these libraries.

**10. Why is Maven used instead of other build tools?**

* **Standardized Structure**: Predefined directory layout.
* **Dependency Management**: Handles external libraries seamlessly.
* **Automation**: Build, test, and deploy with a single command.
* **Integration**: Works well with AEM and **Sling**.

**11. What advantages does Maven offer for AEM development?**

* **Automated Builds**: One command to build the entire project.
* **Dependency Management**: No need to manually download libraries.
* **Consistency**: Ensures all developers build the project the same way.

**12. How does Maven help in managing dependencies and plugins in AEM projects?**

Maven uses:

* **Dependencies**: Defined in <dependencies> section.
* **Plugins**: For deployment, testing, and packaging:

<plugins>

<plugin>

<groupId>com.day.jcr.vault</groupId>

<artifactId>content-package-maven-plugin</artifactId>

</plugin>

</plugins>

**13. What does mvn clean install do in an AEM project?**

* **clean**: Deletes old build files.
* **install**: Builds the project, runs tests, packages it, and places it in the local Maven repository.

**14. How to deploy packages directly to AEM using Maven commands?**

Use:

mvn clean install -PautoInstallPackage

The **-PautoInstallPackage** profile deploys the package directly to the **author** instance.

**15. Explain the purpose of different Maven profiles in AEM (autoInstallPackage, autoInstallBundle).**

* **autoInstallPackage**: Deploys the full content package (ui.apps, ui.content).
* **autoInstallBundle**: Deploys only the **OSGi bundle** (core code) to the AEM instance.

**16. What is the purpose of dumplibs in AEM?**

dumplibs is used to:

* **Debug Client Libraries**.
* Shows all client libraries (clientlibs) loaded by a page.
* Helps in identifying **missing or incorrectly loaded resources**.

**17. How can you view client libraries using dumplibs?**

Append ?debugClientLibs=true to the page URL:

http://localhost:4502/content/my-site.html?debugClientLibs=true

Or use the **Web Console**:

http://localhost:4502/libs/granite/ui/content/dumplibs.html

**18. Explain how client libraries are structured in AEM.**

Client libraries (clientlibs) are organized under:

/apps/<project\_name>/clientlibs/<category\_name>

They contain:

* **CSS** and **JS** files.
* **dependencies.txt**: Defines **dependencies** on other client libraries.
* **js.txt** and **css.txt**: Lists the files to be included.