What is the purpose of the core module in AEM?

This is used  in AEM contains the Java backend logic for components and services. It includes:

1 Java class for business logic, OSGi services, and Sling models.

2 Unit tests for testing the Java code.

**3** For managing service settings can use OSGI configurations

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

The core folder contains Java classes for backend logic, OSGi configurations for services, unit tests for Java code, and the POM file for Maven build configuration of the core module.

Explain the role of ui.apps in AEM projects.

The ui.apps module contains component definitions (HTL scripts, dialog definitions, and component configurations), client libraries (CSS, JavaScript, and other static resources), and content structure that defines how components are stored in the JCR.

How are components structured in the ui.apps folder?

Components in the ui.apps folder are structured as /apps/<project-name>/components/<component-name>

**Hello World Component:**

* Where is the Hello World component located in both core and ui.apps?
* Core: /core/src/main/java/com/<project>/core/models/HelloWorldModel.java
* ui.apps: /apps/<project>/components/helloworld
* Explain the Java class (in core) for the Hello World component.

The Java class defines the backend logic for the component. It typically implements the Sling Model interface and contains methods to fetch and process data for the component.

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

The HTL script (helloworld.html) renders the component’s HTML. It uses data from the Java class to display dynamic content.

* How are properties and dialogs defined for this component?

Properties are defined in .content.xml under the component folder, and dialogs are defined in the dialog folder using XML or Touch UI configurations.

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

The different types of AEM modules include core (backend logic and OSGi services), ui.apps (component definitions and client libraries), ui.content (content structure and templates), it.tests (integration tests), and all (aggregates all modules into a single package).

How does Maven build these modules?

Maven builds modules using the POM (Project Object Model) file. Each module has its own POM file, and the parent POM coordinates the build process.

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

Maven’s build lifecycle includes clean (removes previous build artifacts), compile (compiles the source code), test (runs unit tests), package (packages the code into a deployable artifact), install (installs the artifact in the local repository), and deploy (deploys the artifact to a remote repository).

How are dependencies managed in pom.xml?

Dependencies are declared in the <dependencies> section of the pom.xml file. Maven downloads and manages these dependencies from repositories.

Why is Maven used instead of other build tools?

Maven is used because it provides a standardized build process, simplifies dependency management, supports modular project structures, and integrates well with AEM’s development workflow.

What advantages does Maven offer for AEM development?

Maven offers consistency in the build process, automates dependency management, supports modularity in project structures, and integrates seamlessly with AEM tools and plugins.

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

Maven uses the pom.xml file to declare dependencies and plugins, resolve and download dependencies from repositories, and configure plugins for tasks like packaging and deployment.

What does mvn clean install do in an AEM project?

mvn clean install cleans the project by removing previous build artifacts, compiles, tests, and packages the code, and installs the artifact in the local Maven repository.

How to deploy packages directly to AEM using Maven commands?

Use the AEM Maven Plugin with the command mvn clean install -PautoInstallPackage to build and deploy the package to the AEM instance.

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

The autoInstallPackage profile automatically installs the content package to AEM, and the autoInstallBundle profile automatically installs the OSGi bundle to AEM.

What is the purpose of dumplibs in AEM?

Dumplibs is a tool to inspect and debug client libraries in AEM. It provides details about the libraries loaded on a page.

How can you view client libraries using dumplibs?

Access http://<aem-host>/libs/granite/ui/content/dumplibs.html to view a list of client libraries and their details.

Explain how client libraries are structured in AEM.

Client libraries are structured as /apps/<project>/clientlibs/<library-name>. They include CSS stored in the css folder, JavaScript stored in the js folder, and categories defined in .content.xml to group libraries.