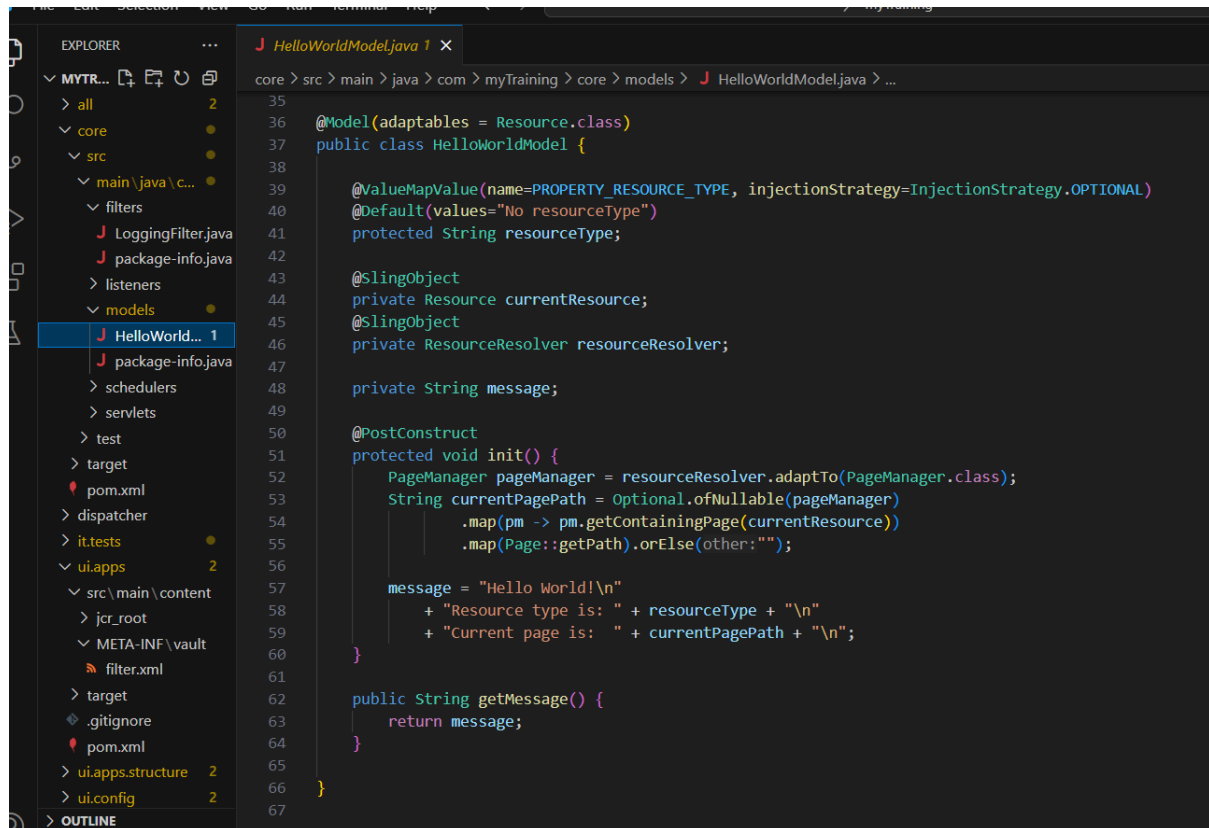


What is the purpose of the core module in AEM?

In an AEM project, the core module is responsible for handling the backend logic, business services, and custom functionality. It is implemented as an OSGi bundle and contains Java-based logic.

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

- Sling Models (Java Classes)



- OSGi Services
- Servlets

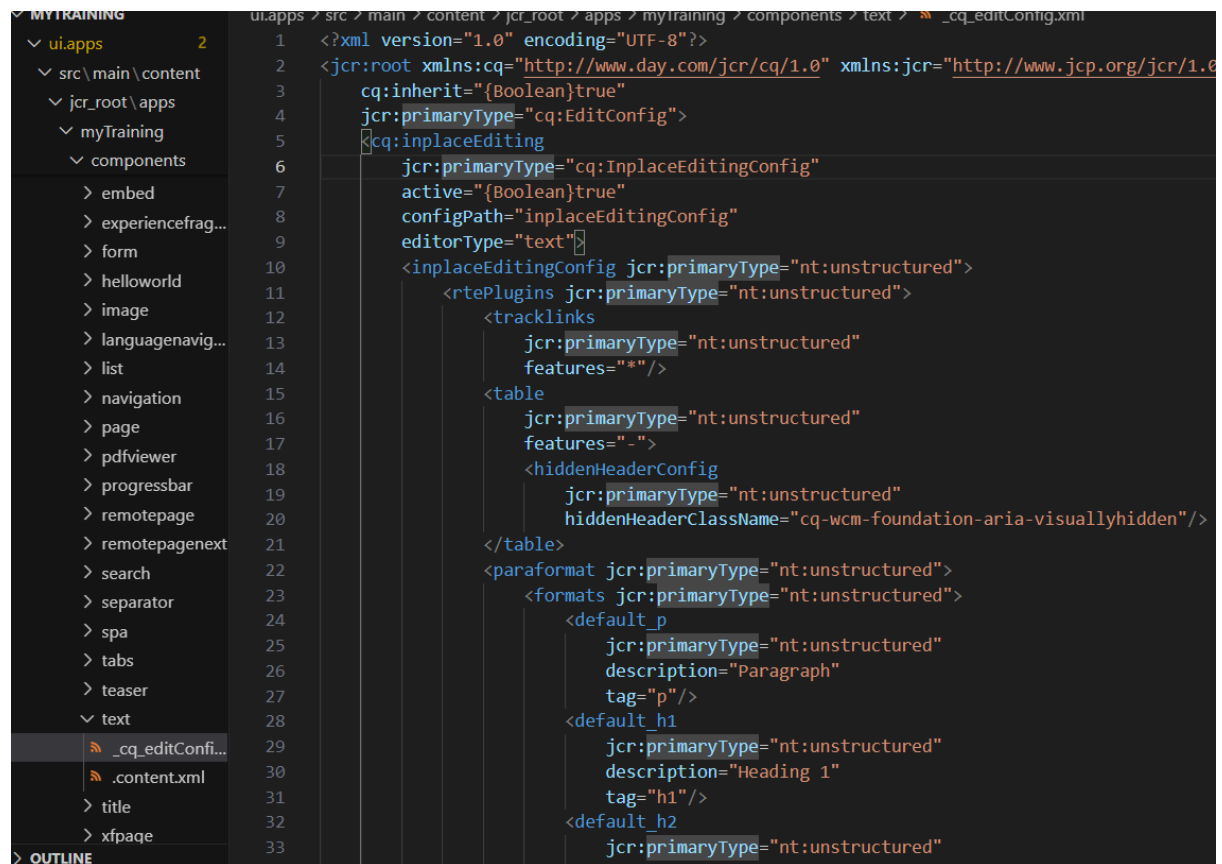
Explain the role of ui.apps in AEM projects.

Defines Front-End Components: Stores all AEM components (cq:Component) used in pages.

Handles Client-Side Resources: Manages CSS, JavaScript, and other assets via client libraries (clientlibs).

How are components structured in the ui.apps folder?

In an AEM project, components are stored under the ui.apps module inside the /apps/<project-name>/components/ folder.



```
<?xml version="1.0" encoding="UTF-8"?>
<jcr:root xmlns:cq="http://www.day.com/jcr/cq/1.0" xmlns:jcr="http://www.jcp.org/jcr/1.0"
  cq:inherit="{Boolean}true"
  jcr:primaryType="cq:EditConfig">
  <cq:inplaceEditing
    jcr:primaryType="cq:InplaceEditingConfig"
    active="{Boolean}true"
    configPath="inplaceEditingConfig"
    editorType="text">
    <inplaceEditingConfig jcr:primaryType="nt:unstructured">
      <rtePlugins jcr:primaryType="nt:unstructured">
        <tracklinks
          jcr:primaryType="nt:unstructured"
          features="*" />
        <table
          jcr:primaryType="nt:unstructured"
          features="-" />
          <hiddenHeaderConfig
            jcr:primaryType="nt:unstructured"
            hiddenHeaderClassName="cq-wcm-foundation-aria-visuallyhidden" />
        </table>
        <paraformat jcr:primaryType="nt:unstructured">
          <formats jcr:primaryType="nt:unstructured">
            <default_p
              jcr:primaryType="nt:unstructured"
              description="Paragraph"
              tag="p" />
            <default_h1
              jcr:primaryType="nt:unstructured"
              description="Heading 1"
              tag="h1" />
            <default_h2
              jcr:primaryType="nt:unstructured"
              description="Heading 2"
              tag="h2" />
          </formats>
        </paraformat>
      </rtePlugins>
    </inplaceEditingConfig>
  </cq:inplaceEditing>
</jcr:root>
```

Hello World Component:

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

- In the core module (Backend - Java Logic)
C:\myTraining\core\src\main\java\com\myTraining\core\models\HelloWorldModel.java
- In the ui.apps module (Frontend - Component Definition)
C:\myTraining\ui.apps\src\main\content\jcr_root\apps\myTraining\components\helloworld

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

- The Java class for the Hello World component in the core module is typically a Sling Model, which provides the business logic for the component

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

- In the ui.apps module, the Hello World component's HTL (Sightly) script is responsible for rendering the output based on the Sling Model (HelloWorldModel.java) from the core module.
- Injecting the Sling Model into HTL
- Rendering Dynamic Content

- How are properties and dialogs defined for this component?

The ui.apps module in AEM defines the components and dialogs for a Hello World component. Content authors can use the Touch UI Dialog to create values (such as a custom message) that are subsequently saved in the JCR (AEM repository) and retrieved by the backend logic.

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

1. Core Module

Typical Contents:

Sling Models (com.example.core.models.HelloWorldModel.java)

OSGi Services & Components (@Component, @Service)

Custom Servlets (@SlingServlet)

Workflow Process Steps

2. UI Apps Module (ui.apps)

Typical Contents:

Components

Client Libraries

Component Dialogs (_cq_dialog.xml)

3. UI Content Module (ui.content)

Typical Contents:

Site Structure (/content/example/...)

Initial Content (.content.xml)

How does Maven build these modules?

AEM projects use Maven to structure, compile, and deploy various modules. The build process follows a parent-child module structure, where different modules (core, ui.apps, ui.content, etc.) contribute to the final AEM package.

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

validate-checks the project is correctly configured

compile-compiles java code in the core module

test-runs unit test

package-Creates the required output: .jar (for core) or .zip (for ui.apps, ui.content).

verify-verify the package

install-for reuse

deploy-deploy the package to AEM

How are dependencies managed in pom.xml?

Maven manages dependencies using the <dependencies> section in pom.xml. In an AEM project, dependencies include:

1. AEM APIs (e.g., Sling, JCR, OSGi).
2. Custom Libraries (e.g., internal Java utilities).
3. Third-party Dependencies (e.g., Apache Commons, Gson).

Why is Maven used instead of other build tools?

Maven is used for its dependency management, multi-module support, and structured build lifecycle

What advantages does Maven offer for AEM development?

- Dependency Management
- Multi-Module Project Support
- Automated Build and Deployment
- Integration with AEM Plugin

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

Maven plays a crucial role in handling dependencies and plugins in Adobe Experience Manager (AEM) projects by simplifying package management, ensuring version consistency, and automating builds.

What does mvn clean install do in an AEM project?

The command mvn clean install is used in AEM development to build, package, and deploy the project while ensuring a clean environment. It executes two key lifecycle phases:

clean → Deletes the target/ folder to remove previous builds

install → Compiles code, runs tests, packages the project, and installs the output

How to deploy packages directly to AEM using Maven commands?

Basic Deployment Command- mvn clean install -PautoInstallPackage

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

- autoInstallPackage- Deploys the entire content package to an AEM instance.

command- mvn clean install -PautoInstallPackage

- autoInstallBundle- Deploys only the OSGi bundle (.jar) to an AEM instance.

command- mvn clean install -PautoInstallBundle

What is the purpose of dumplibs in AEM?

- If a CSS or JavaScript file isn't loading correctly, dumplibs helps identify missing dependencies
- Ensures that JavaScript and CSS files are loaded in the correct sequence.

How can you view client libraries using dumplibs?

- Access the dumplibs URL
- Use the "Filter" field to search for client libraries by name, category, or path.
- The table will show details such as:Category,Path,Dependencies,CSS/JS Files Included.
- <http://localhost:4502/content/my-site/home.html?debugClientLibs=true>.use this command to verify client library
- Open Developer Tools in your browser and look for requests to /etc.clientlibs/... to ensure the client libraries are loading correctly.

Explain how client libraries are structured in AEM.

- Stored under /apps or /etc.clientlibs/
- Uses cq:ClientLibraryFolder node type
- Organized into css, js, and other assets
- Included in pages using categories