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

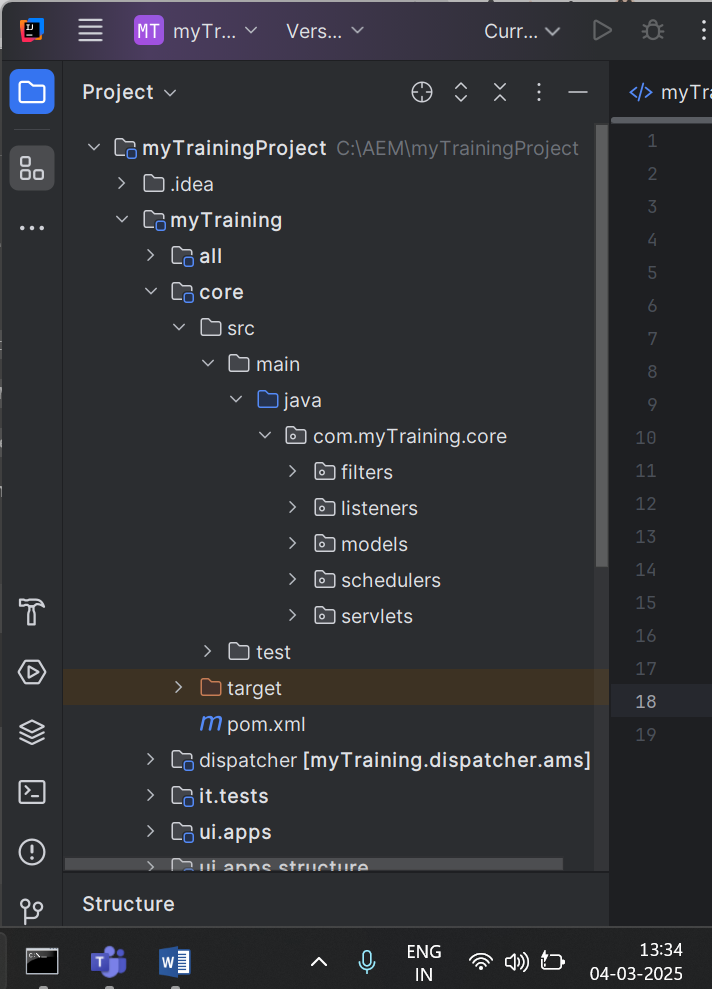
The core module is the backend of an AEM project. It contains all the business logic, services, and code that run behind the scenes. It helps in:

* Processing data from the repository (JCR).
* Creating and managing reusable backend logic.
* Defining custom components, servlets, and models.

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

Inside the core folder , will find:

* **OSGi Components** – Reusable services using @Component.
* **Sling Models** – Java classes that map to AEM content (@Model).
* **Servlets** – Handle HTTP requests (@SlingServletResourceTypes).
* **Event Listeners & Schedulers** – Automate tasks based on AEM events.



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

The ui.apps folder contains the frontend code (UI-related files), such as:

* Components (HTML + JavaScript + CSS).
* Templates for pages.
* Client Libraries (clientlibs) for styling and scripts.

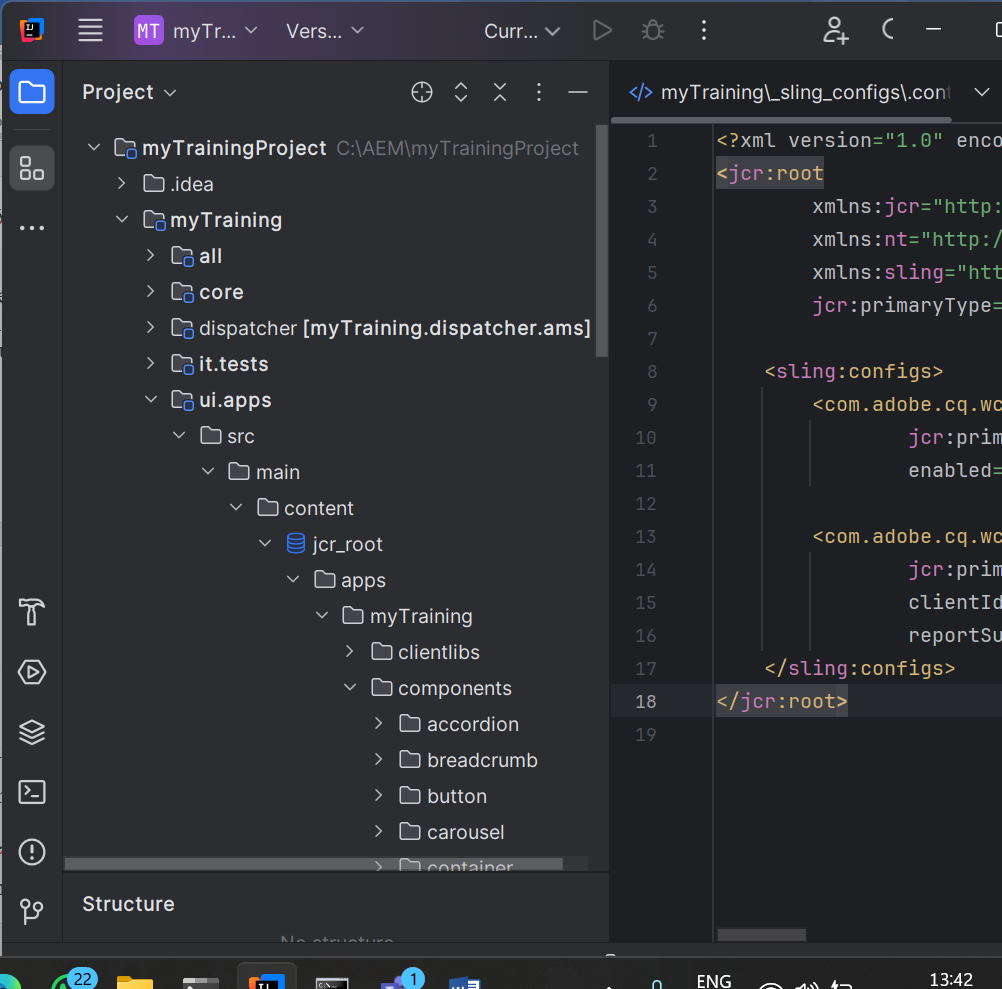
1. **How are components structured in the ui.apps folder?**

Components are stored in:

🡪ui.apps/src/main/content/jcr\_root/apps/myproject/components/

Each component has:

* **HTL File (.html)** – Defines what the component displays.
* **JavaScript / CSS** – Adds functionality and styling.
* **Dialog (\_cq\_dialog.xml)** – Defines user-editable fields.



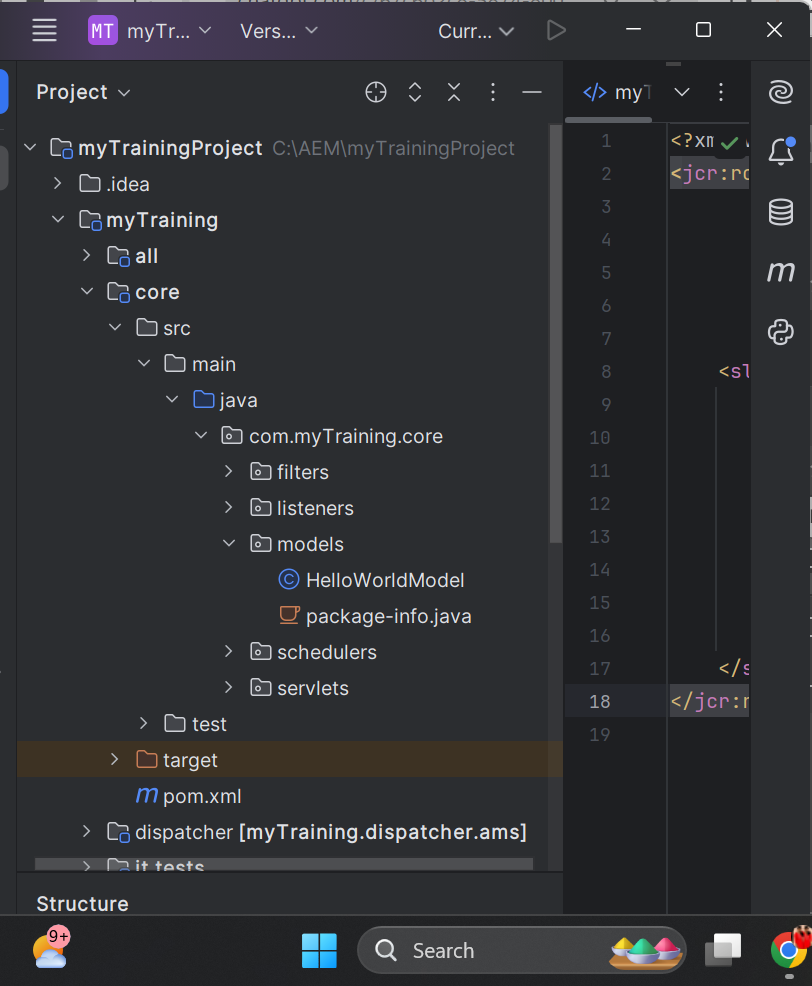
1. **Hello World Component:**

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

The **Hello World component** is present in both the **backend (core)** and **frontend (ui.apps)**:

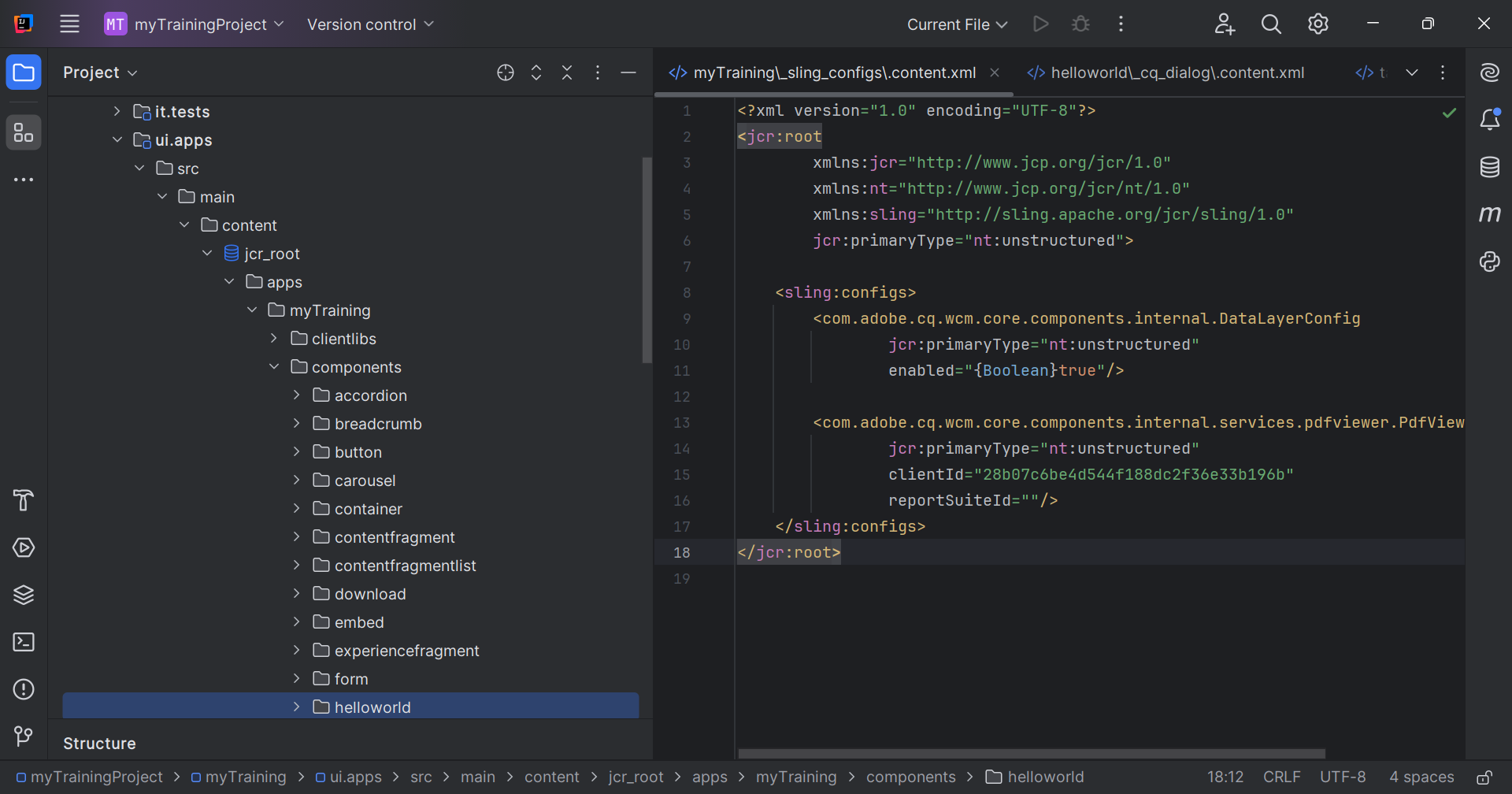
* **Backend (Java logic) – in core**:

🡪core/src/main/java/com/myproject/core/models/HelloWorldModel.java



* **Frontend (HTL + dialogs) – in ui.apps**:

🡪ui.apps/src/main/content/jcr\_root/apps/myproject/components/helloWorld/



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

This is a **Sling Model**, which allows AEM to fetch data from the JCR and pass it to the frontend.

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

HTL (HTML Template Language) is used to display the backend data.

* How are properties and dialogs defined for this component?

To allow users to enter a message in AEM's **Touch UI**, we define a **dialog**.

🡪ui.apps/src/main/content/jcr\_root/apps/myproject/components/helloWorld/\_cq\_dialog.xml

#### **How it Works:**

1. Creates a **dialog box** in the AEM editor.
2. Defines a **textfield** where authors can enter a message.
3. Stores the message in the JCR under ./message.
4. **What are the different types of AEM modules (core, ui.apps, ui.content, etc.)?**

* **core** – Backend logic (Java services, models, servlets).
* **ui.apps** – Frontend components, templates, and client libraries.
* **ui.content** – Stores site content (pages, assets).
* **ui.config** – Stores configuration settings.

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

* The **parent POM** defines the structure.
* Each module has its own **child POM** with dependencies.
* Maven compiles each module and bundles everything into a single package.

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

* **Clean** – Deletes previous builds.
* **Compile** – Compiles Java code.
* **Package** – Bundles code into JARs and ZIPs.
* **Install** – Saves artifacts in a local repository.
* **Deploy** – Uploads the package to AEM.

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

Dependencies are added like this:

Xml:

<dependency>

<groupId>com.adobe.aem</groupId>

<artifactId>uber-jar</artifactId>

<version>6.5.0</version>

<scope>provided</scope>

</dependency>

* Maven automatically downloads the required JAR files.

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

* **Dependency Management** – Automatically downloads libraries.
* **Multi-Module Support** – Handles core, ui.apps, etc., together.
* **Easy Deployment** – Uploads packages to AEM directly.

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

* **Manages Dependencies Automatically** – No need to download AEM libraries manually.
* **Handles Multiple Modules** – Helps organize backend (core), frontend (ui.apps), and content (ui.content).
* **Easy Build and Deployment** – Use simple commands like mvn clean install to build and deploy to AEM.
* **Uses Plugins for AEM** – Special plugins like content-package-maven-plugin help create and install AEM packages.
* **Supports Automation (CI/CD)** – Works with Jenkins and GitHub Actions for auto-deployment.
* **Follows Best Practices** – Keeps AEM projects well-structured and organized.

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

* **Dependencies** – Manage libraries like AEM APIs.
* **Plugins** – Automate tasks like installing code into AEM.

Example:

xml

<plugin>

<groupId>com.adobe.granite</groupId>

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

</plugin>

* This plugin builds AEM packages.

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

* Cleans previous builds (clean).
* Builds the project (install).
* Generates and installs a deployable package.

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

🡪mvn clean install -PautoInstallPackage  
--This command will directly add packages to AEM.

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

* **autoInstallPackage** – Installs the full package into AEM.
* **autoInstallBundle** – Installs only the backend (core).

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

🡪It helps **debug client libraries**.

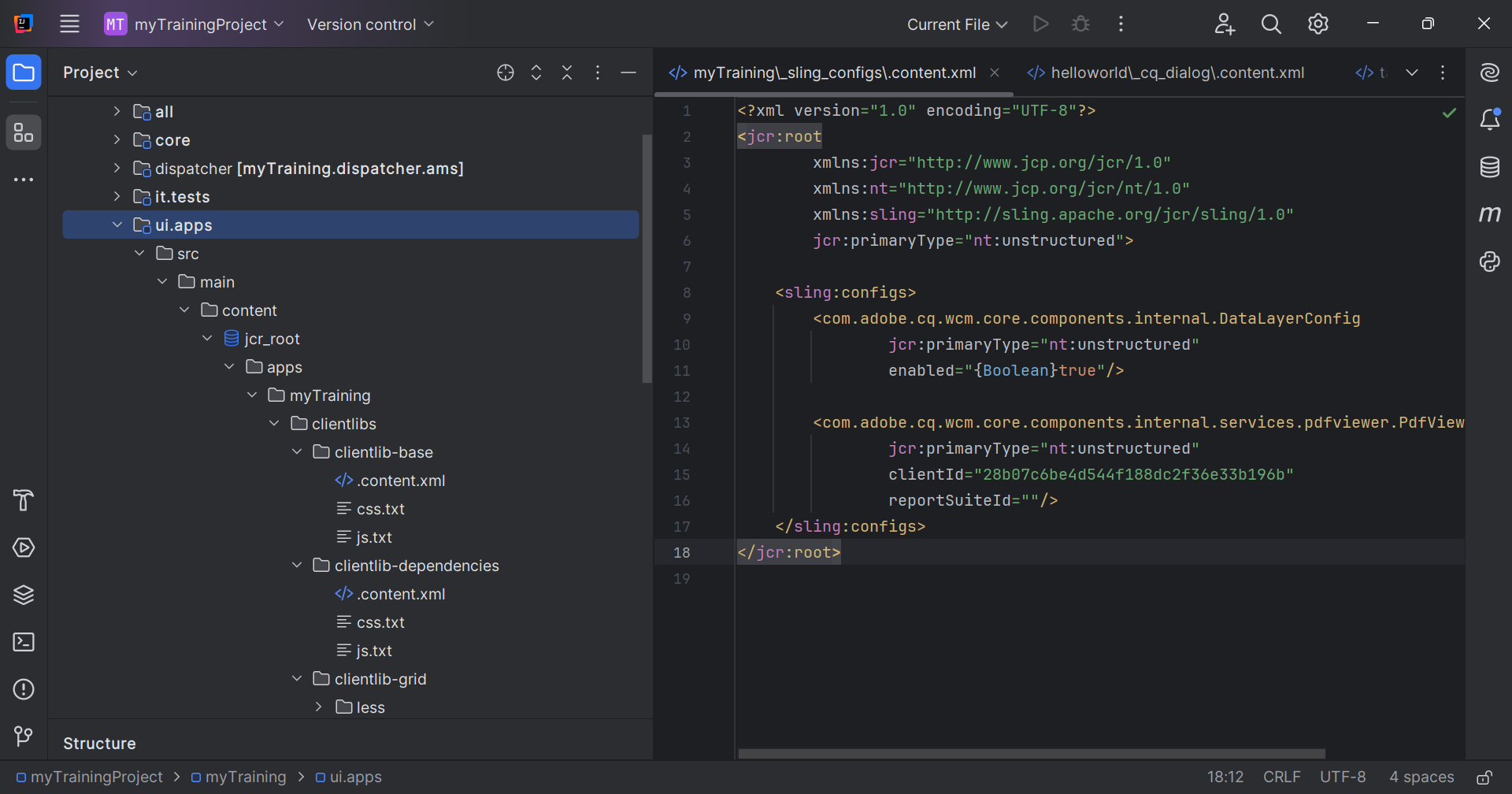
How can you view client libraries using dumplibs?

 Open dumplibs in a browser.

 Search for your library.

 Check if CSS/JS files are loading correctly.

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

****