Project Structure in Adobe AEM

Overview

In the traditional Adobe AEM application a project is broken between the Bundle (Java code) and Content (components, templates, js and cs) projects. The approach I recommend is breaking your project into five distinct modules: Bundle, Content, UI, Config and Dependency. The bundle module will contain the traditional java code and configuration. The Content module will contain the website content nodes so you can build the complete site from scratch. The UI module contains the templates, components, js, css and non-authorable assets. The Config module contains application config that will be managed in the Felix console. The dependency module contains 3rd party OSGi dependencies that can be deployed as a package.

Bundle Module

The bundle module follows the traditional maven project structure for a Java module.

Structure

/src/main/java
/src/main/resources
/src/test/java
/src/test/resource
/target

UI Module

The UI module follows the AEM maven base structure for structuring content. The structure below variates from the standard structure by including a foundation folder and spliting components into two subfolders. The foundation folder if for any foundation components that the application intends to override/extend. By moving those components to the foundation folder you protect yourself from changes when upgrading AEM. By separating the components folder into pages and content you are showing a clear divide between template components and content components.

Structure

/src/main/content/jcr_root/apps/foundation/components
/src/main/content/jcr_root/apps/<appname>/components/content
/src/main/content/jcr_root/apps/<appname>/components/pages
/src/main/content/jcr_root/apps/<appname>/templates
/src/main/content/jcr_root/apps/<appname>/install
/src/main/content/jcr_root/apps/sling

/src/main/content/jcr_root/etc/designs/apps/<appname>design/clinetlibs/css
/src/main/content/jcr_root/etc/designs/apps/<appname>design/clinetlibs/js
/src/main/content/jcr_root/etc/designs/apps/<appname>design/clinetlibs/font
/src/main/content/jcr_root/etc/designs/apps/<appname>design/clinetlibs/images

/src/main/content/META-INF
/target

Content Module

The content module contains the dam assets and website node structure for your website. The content module is a best practice for easing the development of websites. As developers build new pages and utilize new assets the rest of the development team can simple sync up with the source control, get all the new content then deploy the package so they have the latest version of the site locally. This will remove the need to continually pull down content from your AEM servers just to make your local environment work right.

Structure

/src/main/content/jcr_root/content/dam/<appname>
/src/main/content/jcr_root/content/<appname>/en
/target

Config Module

The config module contains properties that your application depends upon. These properties can then be maintained inside the AEM Web Console configuration. These properties can be loaded dynamically based on run modes. AEM uses run modes to define server setup (Author,Publish) and environment (Test, QA, Production).

Structure

/src/main/content/jcr_root/apps/<appname>/config
/src/main/content/jcr_root/apps/<appname>/config.<runmode>
/target

Dependency Module (Optional)

The dependency module is an optional module that can be utilized to deploy all OSGi dependencies as a single package. This can be very convenient if you are utilizing 3rd party java libraries like Spring, Jackson or Hibernate.

Structure

/src/assembly
/src/main/content/jcr_root/apps/dependencies/install
/target