Diagram

Description automatically generated

AEM is web content management solution from Adobe. It was previously known as CQ.  
  
AEM runs on Java Virtual Machine ([JVM](https://docs.oracle.com/javase/specs/jvms/se7/html/)). The main container in AEM is Felix OSGi container which contains a lot of components in the form of OSGi bundles. Sling, Jetty Application Server and other custom java code is contained in the OSGi container in the form of bundles.  
  
Jetty application server is used as one of the bundles which provides the ability to listen to a configured port. Let's learn the AEM technology stack by studying the lifecycle of the request which comes into AEM.

1. Request comes to the configured port hence Jetty can serve it.
2. App server sends the request to the Sling which acts as a controller (in [MVC](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller)pattern). Here's where the resource resolution happens. The URL is broken into pieces and sling's resource resolution rules are applied and a renderer (or a page in this example which can be understood as a pseudo-renderer for our convenience) is identified which can serve this request.
3. Sling knows where to go and what to do.
4. The page is read thru the Jackrabbit Oak API's thru TarMK.  Jackrabbit Oak is the latest implementation of JSR-170 and 283 specifications.
5. The TarMk acts as Kernel between JCR API and the disk.
6. The page generally has a slingResourceType: <component name>. This means the "component" is an actual renderer which can serve this request.
7. The resolution of slingResourceType: <component name> happens again in the Sling. Here the actual renderer is identified again by applying the resource resolution rules.
8. After the renderer is identified, the HTML (View in MVC pattern) code of the renderer is needed. Hence, again thru the Jackrabbit Oak API's, the sightly HTML is read into the memory and compiled into a Java class using sightly (this part is not shown in the diagram for the purpose of simplicity). This component can optionally contain images, videos, client-side code (JavaScript)
9. which will also be included in the view.
10. An AEM component, business logic also can be included as part of JSP, Sightly classes. These form the "Model" (of the MVC pattern) of the application.
11. After processing the business logic and modifying the "View" accordingly, we finally have a response that can be sent back to the client. Since a page can have multiple components and content, the steps 6,7,8,9 may repeat multiple times until all components are rendered.
12. Final output is assembled into an HTML and sent back to Sling
13. Sling processes the response and sends back to Jetty.
14. Jetty sends the processed response back to the user.

**Granite:** Granite is a framework used in AEM which consists of Segment Mk, Tar MK, WCM Java API's, out of the box AEM components, JavaScript API's and some configurations related to these.

**CRX:** Content Repository Extreme is the Adobe custom implementation which takes the basic sling implementation and adds few features like CRXDE (CRX - dev environment), custom Tar MK.

Basically, everything is content or resource in sling. The separation of content (Pages and Assets) and code (HTML, JSP, JS) is just for our understanding.