AEM Groovy Console

Problems?

Modify/fix content (including on live environment)

Change content structure (news/feb/event -> news/2015-2016/feb/event)

Analyze content (MSM, editors use assets from another website)

Site migration (CQ5.4 -> AEM6)

Solutions

Servlets/JSP/Scriptlets

AEM Util packages

External tools (applications/services/Unit Tests(!))

Groovy Console

Advantages of Groovy Console

- do not mess project code
- short and clear scripts
- predefined services/methods
- ability for remote run

Let’s start

- https://github.com/Citytechinc/cq-groovy-console

The AEM Groovy Console provides an interface for running Groovy scripts in the AEM (Adobe CQ) container. Scripts can be created to manipulate content in the JCR, call OSGi services, or execute arbitrary code using the AEM, Sling, or JCR APIs. After installing the package in AEM (instructions below), see the console page for documentation on the available bindings and methods. Sample scripts are included in the package for reference.

- Requirements

AEM 6.1 running on localhost:4502
Versions 7.x.x are compatible with AEM 6.0
Versions 5.x.x and 6.x.x are compatible with CQ 5.6
Versions 3.x.x are compatible with CQ 5.4 and 5.5
Maven 3.x

Let’s start

http://localhost:4502/etc/groovyconsole.html

The binding variables listed below are available for use in all scripts.

session - javax.jcr.Session

pageManager - com.day.cq.wcm.api.PageManager

resourceResolver - org.apache.sling.api.resource.ResourceResolver

slingRequest - org.apache.sling.api.SlingHttpServletRequest

queryBuilder - com.day.cq.search.QueryBuilder

bundleContext - org.osgi.framework.BundleContext

log - org.slf4j.Logger

Let’s start

The methods listed below are available for use in all scripts.

getPage(String path) - Get the Page for the given path, or null if it does not exist.

getNode(String path) - Get the Node for the given path. Throws javax.jcr.RepositoryException if it does not exist.

getResource(String path) - Get the Resource for the given path, or null if it does not exist.

getService(Class<ServiceType> serviceType) - Get the OSGi service instance for the given type, e.g. com.day.cq.workflow.WorkflowService.

copy "sourceAbsolutePath" to "destinationAbsolutePath" - Groovy DSL syntax for copying a node, equivalent to calling session.workspace.copy(sourceAbsolutePath, destinationAbsolutePath).

activate(String path) - Activate the node at the given path.

deactivate(String path) - Deactivate the node at the given path.

doWhileDisabled(String componentClassName, Closure closure) - Execute the provided closure while the specified OSGi component is disabled.

createQuery(Map predicates) - Create a Query instance from the QueryBuilder for the current JCR session.

Sample 1

```
def titleTrimmer =
getService("me.victar.aemexam.components.tool.TitleTrimmer");

getPage("/content/geometrixx").recurse{ page ->

    println titleTrimmer.getTrimmedTitle(page.adaptTo(Resource.class), 5)

}
```

Sample 2

```
def query = createSQL2Query("/content/geometrixx", "geometrixx")

def result = query.execute()

result.rows.each{row -> println row.path }

def createSQL2Query(path, term) {

    def queryManager = session.workspace.queryManager

    def statement = "SELECT * FROM [nt:base] AS s WHERE ISDESCENDANTNODE([${path}]) and CONTAINS(s.*, '${term}')"

    def query = queryManager.createQuery(statement, "JCR-SQL2")

    query

}
```

Sample 3

```
import com.day.cq.commons.jcr.JcrConstants
def stringSearch =  "Banking"
def stringReplace = "Baking"
def query = createSQL2Query("/content/geometrixx",  stringSearch )
def result = query.execute()
result.nodes.each{node ->
    def title = node.get(JcrConstants.JCR_TITLE)
    node.set(JcrConstants.JCR_TITLE, title.replaceAll(stringSearch , stringReplace))
    println node.path
}
save()

def createSQL2Query(path, term) {
    def queryManager = session.workspace.queryManager
    def statement = "SELECT * FROM [cq:PageContent] AS s WHERE ISDESCENDANTNODE([${path}]) and s.[jcr:title] like
'%${term}%'"
     println statement
    def query = queryManager.createQuery(statement, "JCR-SQL2")
    query
}
```