



BeeCon 2016

GroovyRunner

Scripting on steroids

Joost Horward
Open-T



About me

- Founder of Open-T
- Alfresco experience since 2007
- Lead dev for
 - Workflow4people
 - Alice
 - GroovyRunner



BeeCon 2016



What is GroovyRunner?

- Run Groovy script in repository
- **Full access to the entire Java API**
 - Auto-inject all Spring beans
 - Some helpers to make things easier
- Full transaction control
- → It's like scripting, but with the Java API





What can you do with it?

- Repairs
- Imports/exports
- Configuration
- Test
- Analysis
- Troubleshooting





WARNING

- With great power comes great responsibility
- You can break literally anything
- alfresco-global-properties:
 - `groovyrunner.requireAdmin=true`
 - This is the default for obvious reasons





Installation

- Download from:
`www.open-t.nl/projects/groovyrunner`
- Copy `alfresco-groovy-runner.jar` to
`tomcat/webapps/alfresco/WEB-INF/lib`
- Restart alfresco





Running

- `http://[hostname]:
[port]/alfresco/service/open_t/groovyrunner`
- Or commandline:

```
runscript.sh script.groovy
```



GroovyRunner in Action

GroovyRunner

Run!

```
14 // The NodeRef has been extended using a Groovy MetaClass so it has extra
15 // All properties from the cm: namespace map to their short names:
16 println nodeRef.name
17 println nodeRef.title
18
19 // Properties from other namespaces are mapped to prefix_name (as a colon :
20 println nodeRef.sys_locale
21
22 // Let's find Guest home
23 guestHome=findNode("app:guest_home")
24
25 def testNode
26 // We can wrap code in a transaction by using withTransaction { }
27 withTransaction {
28     testNode=guestHome.createNode("testNode")
29     testNode.addAspect("cm:versionable")
30     testNode.title="My test node"
31 }
32
33 println testNode.properties
```

```
1 org.alfresco.service.cmr.repository.NodeRef
2 [{http://www.alfresco.org/model/content/1.0}created:Fri Apr 03 15:57:45 CEST
3 invite-email.html.ftl
4 invite-email.html.ftl
5 invite-email.html.ftl
6 en_US
7 [cm:autoVersionOnUpdateProps:true, cm:created:Sun Jan 31 21:05:54 CET 2016,
8
```

Parameters

[Open-T GroovyRunner 1.0.0 for Alfresco](#). For more information, see the [GroovyRunner project page](#)



Groovy in 60 seconds

- Java-like syntax
- Compiles to normal .class
- “def” - untyped (like Object)
- Closures
- Simple List and Map: Syntax [] and [:]
- Iterators (.each {})
- GString





Let's run something really simple

- Instantiate a Java Date object:

```
Date d = new Date()  
println d
```

Example #1



BeeCon 2016



Access to the Java API

- Spring beans are auto-injected by name
- e.g. nodeService

```
println nodeService.getStores()
```



Access to the Java API

```
nodeService.getStores().each { store ->
  println "Store: ${store}"
  nodeService.getAllRootNodes(store).each { node
->
    println node
    nodeService.getProperties(node).each { name,
value ->
    println "  ${name}=${value}"
  }
}
}
```



Work with Spring Beans

- Get bean names:

```
applicationContext.getBeanDefinitionNames().each  
{ name -> println name }
```

- Get bean by name:

```
def gp=applicationContext.getBean("global-  
properties")  
gp.each { println it }
```



Finding a node

- Simple helpers for finding and traversing:
 - `findNode(path)`
 - Path can be node names or association names
 - `nodeRef.list`



More Groovy in >60 seconds

- MetaClass
 - Add methods to existing classes
- Dynamic properties
 - Add properties to existing classes
 - Implement getProperty and/or setProperty on the MetaClass





Dynamic properties

- Classic way to get the name of a node:
 - `NodeService.getProperty(nodeRef, ContentModel.CM_NAME)`
- Dynamic property:
 - `nodeRef.name`
 - `NodeRef.cm_name`

```
println swsdp.name  
println swsdp.creator  
println swsdp.modifier
```

Example #7



BeeCon 2016



Dynamic properties

- Easy way to get type and aspects:

```
println swsdp.type  
println swsdp.aspects
```



Transaction control

- Wrap in withTransaction { } :

```
withTransaction {  
    doc.title="Test"  
    doc.addAspect("cm:versionable")  
}
```



Run as other user

- Wrap in runAs {} :

```
runAs "mjackson" {  
  withTransaction {  
    doc.title="Test"  
    doc.addAspect("cm:versionable")  
  }  
}
```





Creating nodes

- `nodeRef.createNode(name, type, properties, aspects)`
- `createPath(path)`
- `createSite(name, title, description)`
- `createPerson(properties, password)`
- `createGroup(name)`





Upload content

- `nodeRef.upload(path [,mimetype])`
- Expects 'path' on the server filesystem





Permissions

- `nodeRef.inherit`
- `nodeRef.inheritPermissions(flag)`
- `nodeRef.permissions`
- `nodeRef.setPermission("GROUP_EVERYONE", "Consumer")`
- `nodeRef.deletePermissions()`





Test helper

- Run closure with title and catch any exceptions:

```
test.run "Check versionable aspect is not present", {  
  assert !node.aspects.contains("cm:versionable")  
}
```

- Use “log” for output in tests



How it's made

- Groovy-backed webscript
 - GroovyRunner
- Script delegate
 - GroovyRunnerDelegate
- Service bean
 - GroovyRunnerService





How it's made

- NodeRefMeta
 - Extra properties and methods on NodeRef
- HttpHelper
- TestHelper
- GroovyRunnerResources
 - provides Javascript and CSS for front





There's more to explore

- http helper
- search
- putText
- createPerson
- createGroup etc.
- transform

Full docs at:

<http://www.open-t.nl/projects/groovyrunner/documentation/reference-guide>

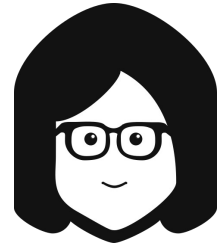


BeeCon 2016



Questions?

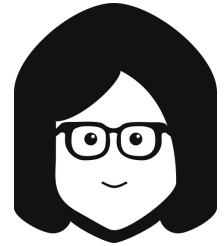
- Anyone?
- Really?
- Thank you!
- You can always find me at joost@open-t.nl and often at orderofthebee IRC (xoost)





Alice

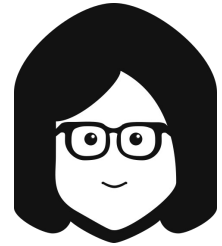
- Engine/platform
- UI
- Power Tools
- Activiti
- Prometheus





Alice platform

- Grails-like-in-Alfresco:
 - Bootstrap 3
 - Groovy Services
 - Groovy Controllers (reloadable)
 - Taglib
 - Message source
 - Freemarker views
- All these are reloadable in development





Alice Power Tools

- Node browser
- GroovyRunner
- Shell
- SQL console
- ActiveMQ
- Log levels
- Threads
- Solr status
- Global properties
- System properties
- Scheduled job overview
- Prometheus instrumentation
- System environment
- Subsystems
- AMP modules
- Memory statistics
- Scheduled jobs
- Content model
 - Browse
 - upload
- Activiti
 - task list
 - process list
 - history
 - upload



Alice Power Tools

<http://www.open-t.nl/products/alice-power-tools>

