Grouper Client

These topics are discussed in the "Grouper Client" training series.

Grouper Client

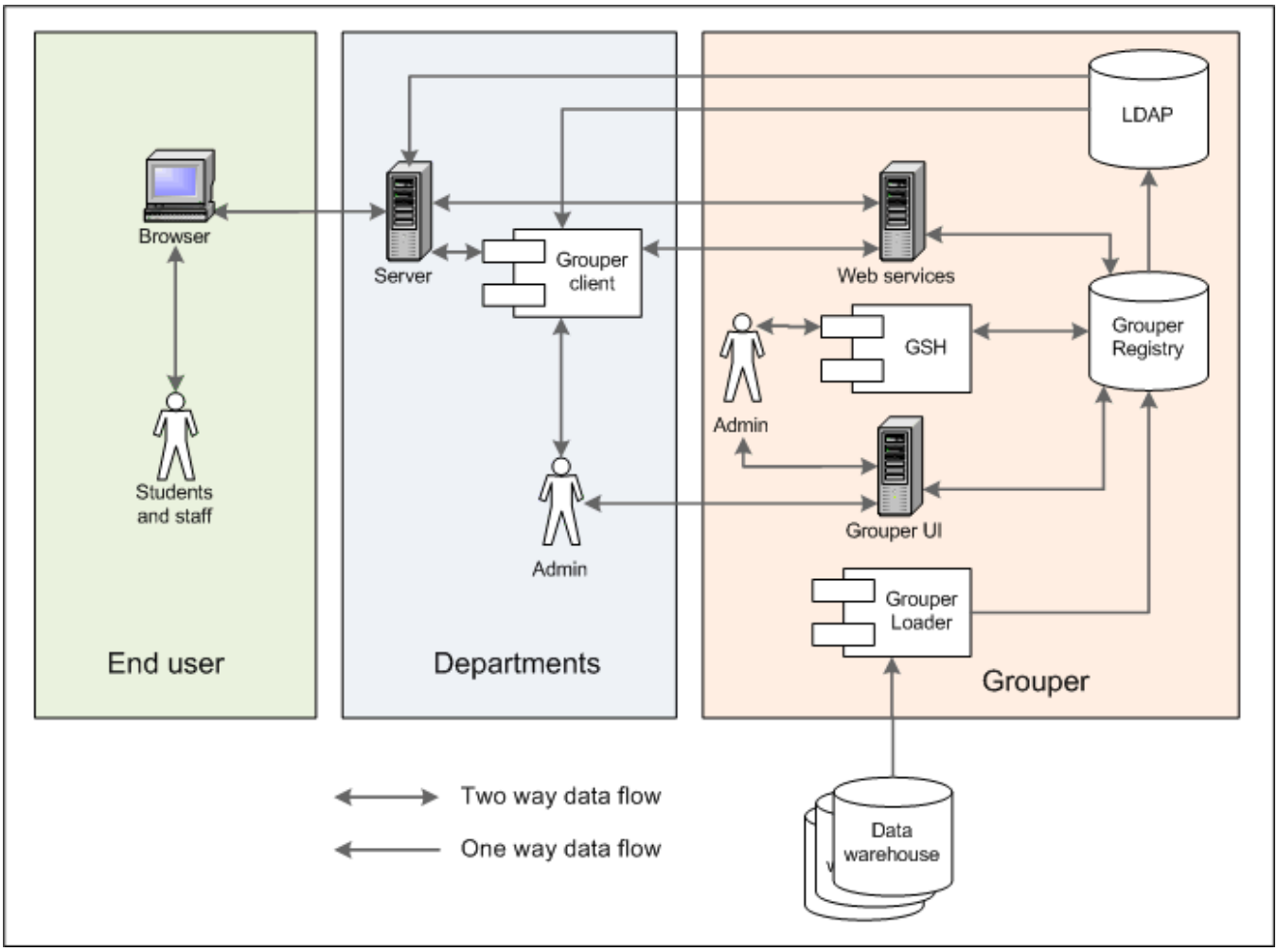
- Build new grouper client
- Grouper Client JDBC API
- Grouper client script from Excel CSV
- Grouper discovery client

Grouper Client is a client for Grouper LDAP and Web Services. See architectural diagram.

- It is a command line client and Java API
- It is self contained in one jar which will not conflict with any other jars you are using
- It is an example of how to use Grouper, a way to quickly try out proof of concepts, and can be used in production
- Grouper client is aimed at the department user, who probably did not install Grouper, but who is using an installation at their institution
- If LDAP and web services are too low level an API for you at some point in your development cycle, Grouper Client is for you
- Another advantage is the web services client (REST) is forwards compatible, so if Grouper Web Services passes back new XML elements going forward, it will still work with this client.
- You can use this to easily see the HTTP/XML requests and responses to grouper-ws. Note, the headers are not detected, they are assumed to be correct.

As of Grouper v. 2.1, there is also a Failover Client and a Discovery Client.

Here is a detailed diagram showing the Grouper Client:



FAQ

- **How can I return pennnames/pennkeys from the web service? (note: pennname/pennkey is the subjectIdentifier at Penn, this is a text based login, not the numeric subjectId)**

First your web service's sources.xml needs to return the subject attribute. I had to make a column in my jdbc source of PENNNAME (Oracle jdbc metadata makes this always UPPER)

You can specify to return pennnames as the subject attributes, and you can use them in your output template:

```
C:\temp>java -jar grouperClient.jar --operation=getMembersWs --groupNames=test:testGroup --
subjectAttributeNames=PENNNAME --outputTemplate=${wsSubject.attributeValues[0]}$newline$
bwh
mchyzer
```

BTW: you can show this on the UI in the custom/media.properties entry: subject.attributes.order.pennperson=name,description,subjectType,id,PENNNAME

- **How can I query based on pennkey from the web service? (again pennkey is Penn's subject identifier)**

First you need to understand that in the grouper.client.properties file you can mask "subjectId" and "subjectIdentifier" with terms used at your institution. So you have custom commands. For Penn we can do:

You can use the built in pennkey support in Penn's grouper client (needs custom configuration over the generic Grouper download):

```
C:\temp>java -jar grouperClient.jar --operation=hasMemberWs --groupName=test:testGroup --pennKeys=mchyzer,bwh
Index 0: success: T: code: IS_MEMBER: 10099999: true
Index 1: success: T: code: IS_MEMBER: 10099998: true
```

This is due to our grouper.client.properties settings:

```
#note: you will see documentation in the grouper.client.example.properties
grouperClient.alias.subjectIds = pennIds
grouperClient.alias.subjectIdentifiers = pennKeys
grouperClient.alias.subjectId = pennId
grouperClient.alias.subjectIdentifier = pennKey
grouperClient.alias.SubjectId = PennId
grouperClient.alias.SubjectIdentifier = PennKey
```

If you didn't have this customization, you can simply look by subject identifier:

```
C:\temp>java -jar grouperClient.jar --operation=hasMemberWs --groupName=test:testGroup --
subjectIdentifiers=mchyzer,bwh
Index 0: success: T: code: IS_MEMBER: 10099999: true
Index 1: success: T: code: IS_MEMBER: 10099998: true
```

* **How can I make a group which has a manual membership list and requires users to be faculty student or staff?**

First off, you need permission to view the facultyStudentStaff group, if it is not public. Note, the composite arguments shouldnt be necessary, but until it is fixed, use them and it will work. This makes a group, a system of record group (where the manual entries go), and the overall group is a composite intersection of the manual group and the facultyStudentStaff group. Note you need to enable "requireGroups" in your grouper.properties

```
C:\temp>java -jar grouperClient.jar --operation=groupSaveWs --name=test:isc:astt:chris:myGroup --
includeGroupDetail=true --description="test group with requiring active facultyStudentStaff" --
displayExtension="My test group" --attributeName0=requireAlsoInGroups --attributeValue0=penn:somewhere:
facultyStudentStaff --typeNames=requireInGroups --compositeType=INTERSECTION --leftGroupName=test:isc:astt:
chris:myGroup_systemOfRecord --rightGroupName=penn:somewhere:facultyStudentStaff
Success: T: code: SUCCESS_INSERTED: test:isc:astt:chris:myGroup
```

Using (from end-user's perspective)

To use grouper client, you need java 1.5+, the grouperClient.jar, and the grouper.client.properties file (either in your classpath, or in the same directory as grouperClient.jar

To use command line, just type this to see usage:

```
java -jar grouperClient.jar
```

The usage will be specific to your institution... but here is a sample usage:

```
Grouper Client USAGE:

This program runs queries against grouper ldap and web services
The system exit code will be 0 for success, and not 0 for failure
Output data is printed to stdout, error messages are printed to stderr or logs (configured in grouper.client.
properties)
Grouper client webpage: https://wiki.internet2.edu/confluence/display/GrouperWG/Grouper+Client

Arguments are in the format: --argName=argValue
Example argument: --operation=encryptPassword
Example argument(OS dependent): --operation="value with whitespace"

Optional arguments below are in [brackets]

##################################################
## Misc operations

Encrypt passwords for storing passwords in external encrypted files:
  java -jar grouperClient.jar --operation=encryptPassword [--dontMask=true|false]

Usage (this message):
  java -jar grouperClient.jar

Send file to web service (readOnly is a designation for the always available client):
  java -jar grouperClient.jar --operation=sendFile --urlSuffix=groups/aStem:aGroup/members
[fileName=theFileName] [fileContents=theFileContents] [--contentType=text/xml] [--labelForLog=addMember] [--
```

```
indentOutput=false] [--saveResultsToFile=fileName] [--readOnly=true] [--debug=true] [--
clientVersion=someVersion]
  e.g. java -jar grouperClient.jar --operation=sendFile --fileName="C:/addMember.xml" --urlSuffix=groups/aStem:
aGroup/members

##################################################
## LDAP operations

NOTE: CHANGE THIS OR REMOVE IN grouper.client.usage.txt FOR YOUR SCHOOOL'S LDAP QUERIES
pennname to pennid usage:
  java -jar grouperClient.jar --operation=pennnameToPennid --pennnameToDecode=pennname [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--debug=true]
  e.g.: java -jar grouperClient.jar --operation=pennnameToPennid --pennnameToDecode=jsmith
  output: pennid: 12341234

NOTE: CHANGE THIS OR REMOVE IN grouper.client.usage.txt FOR YOUR SCHOOOL'S LDAP QUERIES
pennid to pennname usage:
  java -jar grouperClient.jar --operation=pennidToPennkey --pennidToDecode=pennid [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--debug=true]
  e.g.: java -jar grouperClient.jar --operation=pennidToPennkey --pennidToDecode=12341234
  output: pennname: jsmith

NOTE: CHANGE THIS OR REMOVE IN grouper.client.usage.txt FOR YOUR SCHOOOL'S LDAP QUERIES
hasMember ldap usage:
  java -jar grouperClient.jar --operation=hasMemberLdap --groupName=a:b:c --pennnameToCheck=pennkey [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--debug=true]
  e.g.: java -jar grouperClient.jar --operation=hasMemberLdap --groupName=penn:myfolder:mygroup --
pennnameToCheck=jsmith
  output: hasMemberLdap: true

NOTE: CHANGE THIS OR REMOVE IN grouper.client.usage.txt FOR YOUR SCHOOOL'S LDAP QUERIES
getMembers ldap usage:
  java -jar grouperClient.jar --operation=getMembersLdap --groupName=a:b:c [--saveResultsToFile=fileName] [--
outputTemplate=somePattern] [--debug=true]
  e.g.: java -jar grouperClient.jar --operation=getMembersLdap --groupName=penn:myfolder:mygroup
  output: groupList: jsmith, tsmith, msmith
  note: extremely large group lists might not display fully (e.g. over 1000 members)

##################################################
## Web Service operations

addMemberWs web service usage (note: you can replace all members of a group also):
  java -jar grouperClient.jar --operation=addMemberWs [--groupName=a:b:c] [--groupUuid=123abc] [--
subjectIds=subjId0,subjId1] [--subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1] [--
subjectIdsFile=fileName] [--subjectIdentifiersFile=fileName] [--subjectSourcesFile=fileName] [--
defaultSubjectSource=subjectSourceId] [--fieldName=fieldNameToAdd] [--txType=NONE|READ_WRITE_NEW] [--
includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--
replaceAllExisting=true|false] [--disabledTime=yyyy/mm/dd hh:mi:ss] [--enabledTime=yyyy/mm/dd hh:mi:ss] [--
addExternalSubjectIfNotFound=true|false] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=addMemberWs --groupName=aStem:aGroup --subjectIds=12345,23456
  output line: Index 0: success: T: code: SUCCESS: 12345

getMembersWs web service usage:
  java -jar grouperClient.jar --operation=getMembersWs [--groupNames=a:b:c,a:b:d] [--groupUuids=1234,abcd] [--
fieldName=fieldNameToAdd] [--memberFilter=All|Immediate|NonImmediate|Effective|Composite] [--
sourceIds=sourceId1,sourceId2] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion] [--pointInTimeFrom=yyyy/mm/dd hh:mi:ss] [--pointInTimeTo=yyyy/mm/dd hh:mi:ss]
  e.g.: java -jar grouperClient.jar --operation=getMembersWs --groupNames=aStem:aGroup,aStem:aGroup2
  output line: GroupIndex 0: success: T: code: SUCCESS: group: aStem:aGroup: subjectIndex: 0: 12345

deleteMemberWs web service usage:
  java -jar grouperClient.jar --operation=deleteMemberWs [--groupName=a:b:c] [--groupUuid=abc123] [--
subjectIds=subjId0,subjId1] [--subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1] [--
subjectIdsFile=fileName] [--subjectIdentifiersFile=fileName] [--subjectSourcesFile=fileName] [--
defaultSubjectSource=subjectSourceId] [--fieldName=fieldNameToAdd] [--txType=NONE|READ_WRITE_NEW] [--
```

```
includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=deleteMemberWs --groupName=aStem:aGroup --subjectIds=12345,23456
  output line: Index 0: success: T: code: SUCCESS: 12345

hasMemberWs web service usage:
  java -jar grouperClient.jar --operation=hasMemberWs [--groupName=a:b:c] [groupUuid=123abc] [--
subjectIds=subjId0,subjId1] [--subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1] [--
subjectIdsFile=fileName] [--subjectIdentifiersFile=fileName] [--subjectSourcesFile=fileName] [--
defaultSubjectSource=subjectSourceId] [--fieldName=fieldNameToAdd] [--memberFilter=GcMemberFilter] [--
includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion] [--
pointInTimeFrom=yyyy/mm/dd hh:mi:ss] [--pointInTimeTo=yyyy/mm/dd hh:mi:ss]
  e.g.: java -jar grouperClient.jar --operation=hasMemberWs --groupName=aStem:aGroup --subjectIds=12345,23456
  output line: Index 0: success: T: code: IS_MEMBER: 12345: true

getGroupsWs web service usage:
  java -jar grouperClient.jar --operation=getGroupsWs [--subjectIds=subjId0,subjId1] [--
subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1] [--subjectIdsFile=fileName] [--
subjectIdentifiersFile=fileName] [--subjectSourcesFile=fileName] [--defaultSubjectSource=subjectSourceId] [--
memberFilter=GcMemberFilter] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion] [--scope=some:folder:] [--stemName=stemNameToSearchIn] [--
stemUuid=stemUuidToSearchIn] [--stemScope=ONE_LEVEL|ALL_IN_SUBTREE] [--enabled=A|T|F] [--pageSize=100] [--
pageNumber=1] [--sortString=displayName] [--ascending=true|false] [--fieldName=members] [--pointInTimeFrom=yyyy
/mm/dd hh:mi:ss] [--pointInTimeTo=yyyy/mm/dd hh:mi:ss]
  e.g.: java -jar grouperClient.jar --operation=getGroupsWs --subjectIds=12345,23456
  output line: SubjectIndex 0: success: T: code: SUCCESS: subject: 12345: groupIndex: 0: aStem:aGroup2

groupSaveWs web service usage:
  java -jar grouperClient.jar --operation=groupSaveWs --name=a:b:c [--includeGroupDetail=true] [--
txType=NONE|READ_WRITE_NEW] [--saveMode=INSERT_OR_UPDATE|INSERT|UPDATE] [--groupLookupName=a:b:c] [--
groupLookupUuid=sd87f-dsf87-sdf89-df78f] [--description=theDescription] [--
displayExtension=theDisplayExtension] [--createParentStemsIfNotExist=true|false] [--
typeOfGroup=group|role|entity] [--attributeName0=someName] [--attributeValue0=someValue] [--
attributeNameX=xthName] [--attributeValueX=xthValue] [--compositeType=COMPLEMENT|INTERSECTION|UNION] [--
leftGroupName=compositeLeft] [--rightGroupName=compositeRight] [--groupDetailParamName0=paramName] [--
groupDetailParamValue0=paramValue] [--groupDetailParamNameX=xthName] [--groupDetailParamNameX=xthValue] [--
typeNames=namesOfGroupTypes]  [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=groupSaveWs --name=aStem:aGroup
  output: Success: T: code: SUCCESS_INSERTED: aStem:aGroup

stemSaveWs web service usage:
  java -jar grouperClient.jar --operation=stemSaveWs --name=groupName [--txType=NONE|READ_WRITE_NEW] [--
saveMode=INSERT_OR_UPDATE|INSERT|UPDATE] [--stemLookupName=theName] [--stemLookupUuid=theUuid] [--
description=theDescription] [--displayExtension=theDisplayExtension] [--createParentStemsIfNotExist=true|false]
[--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=stemSaveWs --name=aStem:someStem
  output: Success: T: code: SUCCESS_INSERTED: aStem:someStem

groupDeleteWs web service usage:
  java -jar grouperClient.jar --operation=groupDeleteWs --groupNames=groupName0,groupName1 [--
txType=NONE|READ_WRITE_NEW] [--includeGroupDetail=true|false] [--actAsSubjectId=subjId] [--
actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--saveResultsToFile=fileName] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=groupDeleteWs --groupNames=aStem:aGroup0,aStem:aGroup1
  output line: Index 0: success: T: code: SUCCESS: aStem:aGroup0

stemDeleteWs web service usage:
```

```
  java -jar grouperClient.jar --operation=stemDeleteWs --stemNames=a:b,a:c [--txType=NONE|READ_WRITE_NEW] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=stemDeleteWs --stemNames=aStem:aStem0,aStem:aStem1
  output line: Index 0: success: T: code: SUCCESS: aStem:aStem0

getGrouperPrivilegesLiteWs web service usage
  java -jar grouperClient.jar --operation=getGrouperPrivilegesLiteWs [--groupName=a:b:c] [--stemName=a:b] [--
privilegeName=admin|view|read|optin|optout|update|stem|create|etc] [--privilegeType=access|naming|etc] [--
subjectId=subjId0] [--subjectIdentifier=subjIdent0] [--subjectSource=source0] [--includeGroupDetail=true|false]
[--includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--
actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--saveResultsToFile=fileName] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=getGrouperPrivilegesLiteWs --groupName=aStem:aGroup --
subjectId=test.subject.0
  output line: Index 0: success: T: code: SUCCESS: group: aStem:aGroup: subject: test.subject.0: access: admin

assignGrouperPrivilegesWs web service usage
  java -jar grouperClient.jar --operation=assignGrouperPrivilegesWs --
privilegeNames=admin|view|read|optin|optout|update|stem|create|etc (comma separated) --allowed=true|false [--
groupName=a:b:c] [--stemName=a:b] [--privilegeType=access|naming|etc] [--subjectIds=subjId0,subjId1] [--
subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1] [--txType=NONE|READ_WRITE_NEW] [--
replaceAllExisting=true|false] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=assignGrouperPrivilegesWs --groupName=aStem:aGroup --
subjectIds=test.subject.0,test.subject.1 --privilegeNames=admin,update --allowed=true
  output: Index: 0, success: T, code: SUCCESS_ALLOWED, group: aStem:aGroup, subject: test.subject.0, access:
admin

assignGrouperPrivilegesLiteWs web service usage
  java -jar grouperClient.jar --operation=assignGrouperPrivilegesLiteWs --
privilegeName=admin|view|read|optin|optout|update|stem|create|etc --allowed=true|false [--groupName=a:b:c] [--
stemName=a:b] [--privilegeType=access|naming|etc] [--subjectId=subjId0] [--subjectIdentifier=subjIdent0] [--
subjectSource=source0] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=assignGrouperPrivilegesLiteWs --groupName=aStem:aGroup --
subjectId=test.subject.0 --privilegeName=admin --allowed=true
  output: Success: T: code: SUCCESS_ALLOWED: group: aStem:aGroup: subject: test.subject.0: access: admin

findGroupsWs web service usage
  java -jar grouperClient.jar --operation=findGroupsWs --
queryFilterType=AND|MINUS|OR|FIND_BY_APPROXIMATE_ATTRIBUTE|FIND_BY_EXACT_ATTRIBUTE|FIND_BY_GROUP_NAME_APPROXIMAT
E|FIND_BY_GROUP_NAME_EXACT|FIND_BY_GROUP_UUID|FIND_BY_STEM_NAME|FIND_BY_TYPE|etc [--groupName=a:b:c] [--
groupUuid=12as-1234gjth] [--groupNames=a:b,b:c] [--groupUuids=12ab,23cd] [--stemName=aStem:someStem] [--
stemUuid=sfds-sds234] [--stemNameScope=ONE_LEVEL|ALL_IN_SUBTREE] [--groupTypeName=someName] [--
groupAttributeName=someName] [--groupAttributeValue=someValue] [--includeGroupDetail=true|false] [--
sortString=T|F] [--ascending=T|F] [--pageNumber=2] [--pageSize=50] [--typeOfGroups=group,role,entity] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=findGroupsWs --queryFilterType=FIND_BY_GROUP_NAME_APPROXIMATE --
groupName=aStem:aGroup
  output: Index 0: name: aStem:aGroup, displayName: A stem:A Group

  Note: to specify group math, use queryFilterType of AND|OR|MINUS, and then specify attribute for the left
group with a 0 after attribute name, and 1 for the right group.
  e.g.: java -jar grouperClient.jar --operation=findGroupsWs --queryFilterType=OR --queryFilterType0=OR --
queryFilterType00=FIND_BY_GROUP_NAME_APPROXIMATE --groupName00=aStem:aGroup --
queryFilterType01=FIND_BY_GROUP_NAME_APPROXIMATE --groupName01=aStem:aGroup --
queryFilterType1=FIND_BY_GROUP_NAME_APPROXIMATE --groupName1=aStem:aGroup

  Note: it is not clear which attributes go with which filter types, the rules are in the Java class:
WsQueryFilterType
```

```
  or use trial and error

findStemsWs web service usage
  java -jar grouperClient.jar --operation=findStemsWs --
stemQueryFilterType=AND|MINUS|OR|FIND_BY_APPROXIMATE_ATTRIBUTE|FIND_BY_PARENT_STEM_NAME|FIND_BY_STEM_NAME|FIND_B
Y_STEM_NAME_APPROXIMATE|FIND_BY_STEM_UUID|etc [--stemName=a:b:c] [--stemUuid=12as-1234gjth] [--stemNames=a:b,b:
c] [--stemUuids=12ab,23cd] [--parentStemName=aStem:someStem] [--parentStemNameScope=ONE_LEVEL|ALL_IN_SUBTREE]
[--stemAttributeName=someName] [--stemAttributeValue=someValue] [--sortString=T|F] [--ascending=T|F] [--
pageNumber=2] [--pageSize=50] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=findGroupsWs --
stemQueryFilterType=FIND_BY_STEM_NAME_APPROXIMATE --stemName=aStem:aGroup
  output: Index 0: name: aStem:aStem0, displayName: A stem:A Stem 0

  Note: to specify group math, use stemQueryFilterType of AND|OR|MINUS, and then specify attribute for the left
stem with a 0 after attribute name, and 1 for the right stem.
  e.g.: java -jar grouperClient.jar --operation=findStemsWs --stemQueryFilterType=OR --stemQueryFilterType0=OR
--stemQueryFilterType00=FIND_BY_STEM_NAME --stemName00=aStem --stemQueryFilterType01=FIND_BY_STEM_NAME --
stemName01=aStem --stemQueryFilterType1=FIND_BY_STEM_NAME --stemName1=aStem

  Note: it is not clear which attributes go with which filter types, the rules are in the Java class:
WsStemQueryFilterType
  or use trial and error

memberChangeSubjectWs web service usage (note: you need to be in the sysAdminGroup or actAs someone who is)
  java -jar grouperClient.jar --operation=memberChangeSubjectWs [--oldSubjectId=oldId] [--
oldSubjectIdentifier=oldIdent] [--oldSubjectSource=oldSourceId] [--newSubjectId=newId] [--
newSubjectIdentifier=newIdent] [--newSubjectSource=newSourceId] [--deleteOldMember=false] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=memberChangeSubjectWs --oldSubjectId=test.subject.0 --
newSubjectId=test.subject.1 --actAsSubjectId=GrouperSystem
  output: Success: T: code: SUCCESS: oldSubject: test.subject.0, newSubject: test.subject.1

getMembershipsWs web service usage:
  java -jar grouperClient.jar --operation=getMembershipsWs [--groupNames=a:b:c,a:b:d] [--groupUuids=1234,abcd]
[--subjectIds=subjId0,subjId1] [--subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1]
[--fieldName=fieldNameToAdd] [--memberFilter=All|Immediate|NonImmediate|Effective|Composite] [--
sourceIds=sourceId1,sourceId2] [--membershipIds=abc,bcd] [--scope=school:folder:somewhere] [--stemName=a:b:c]
[--stemUuid=abc] [--stemScope=ONE_LEVEL|ALL_IN_SUBTREE] [--enabled=A|T|F] [--includeGroupDetail=true|false] [--
includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--
actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--saveResultsToFile=fileName] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=getMembershipsWs --groupNames=aStem:aGroup,aStem:aGroup2
  output line: Index 0: group: aStem:aGroup, subject: 12345, list: members, type: Immediate, enabled: T
  Note: subjectSources are the sources for the subjects specified.  sourceIds are if you arent specifying
subjectIds, and you just want all person memberships for example.

getSubjectsWs web service usage:
  java -jar grouperClient.jar --operation=getSubjectsWs [--searchString=someone] [--subjectIds=subjId0,subjId1]
[--subjectIdentifiers=subjIdent0,subjIdent1] [--subjectSources=source0,source1] [--fieldName=fieldNameToAdd] [--
memberFilter=All|Immediate|NonImmediate|Effective|Composite] [--sourceIds=sourceId1,sourceId2] [--
includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--groupName=a:b:c]
[--groupUuid=1234] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0] [--
paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=getSubjectsWs --subjectIds=subjId0,subjId1
  output line: Index 0: success: T, code: SUCCESS, subject: 12345
  Note: subjectSources are the sources for the subjects specified.  sourceIds are if you arent specifying
subjectIds, and you want to filter the searchString

getAttributeAssignmentsWs web service usage:
  java -jar grouperClient.jar --operation=getAttributeAssignmentsWs --
attributeAssignType=group|member|stem|any_mem|imm_mem|attr_def|any_mem_asgn|attr_def_asgn|group_asgn|imm_mem_asg
n|mem_asgn|stem_asgn [--includeAssignmentsOnAssignments=true|false] [--attributeDefNames=a:b,b:c] [--
attributeDefUuids=1a,2b] [--attributeDefNameNames=a:b,b:c] [--attributeDefNameUuids=1a,2b] [--
ownerAttributeDefNames=a:b,b:c] [--ownerAttributeDefUuids=1a,2b] [--ownerGroupNames=a:b:c,a:b:d] [--
```

```
ownerGroupUuids=1234,abcd] [--owner0SubjectId=subjId0] [--owner0SubjectIdentifier=subjIdent0] [--
owner0SubjectSource=source0] [--ownerMembershipUuids=abc,bcd] [--ownerStemNames=a:b,b:c] [--ownerStemUuids=1a,
2b] [--ownerMembershipAny0SubjectId=12] [--ownerMembershipAny0SubjectIdentifier=ab] [--
ownerMembershipAny0SourceId=xyz] [--ownerMembershipAny0GroupName=3c] [--ownerMembershipAny0GroupUuid=1a] [--
attributeAssignUuids=a1,b2] [--attributeDefValueType=floating|integer|memberId|string|timestamp] [--
theValue=123] [--includeAssignmentsFromAssignments=T|F] [--attributeDefType==attr|domain|type|limit|perm] [--
assignAssignOwnerAttributeAssignUuids=a1,b2] [--assignAssignOwnerNamesOfAttributeDefs=a:b,b:c] [--
assignAssignOwnerUuidsOfAttributeDefs=1a,2b] [--assignAssignOwnerNamesOfAttributeDefNames=a:b,b:c] [--
assignAssignOwnerUuidsOfAttributeDefNames=1a,2b] [--assignAssignOwnerActions=read] [--enabled=A|T|F] [--
actions=read,write] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=getAttributeAssignmentsWs --attributeAssignType=group --
attributeDefNames=test:testAttributeAssignDefNameDef
  output line: Index: 0: attributeAssignType: group, owner: test:groupTestAttrAssign, attributeDefNameName:
test:testAttributeAssignDefName, action: assign, values: 15,5,5, enabled: T, id:
a9c83eeb78c04ae5befcea36272d318c

assignAttributesWs web service usage:
  java -jar grouperClient.jar --operation=assignAttributesWs --
attributeAssignType=group|member|stem|any_mem|imm_mem|attr_def|group_asgn|mem_asgn|stem_asgn|any_mem_asgn|imm_me
m_asgn|attr_def_asgn --attributeAssignOperation=assign_attr|add_attr|remove_attr|replace_attrs [--
attributeDefNameNames=a:b,b:c] [--attributeDefNameUuids=1a,2b] [--ownerAttributeDefNames=a:b,b:c] [--
ownerAttributeDefUuids=1a,2b] [--ownerGroupNames=a:b:c,a:b:d] [--ownerGroupUuids=1234,abcd] [--
owner0SubjectId=subjId0] [--owner0SubjectIdentifier=subjIdent0] [--owner0SubjectSource=source0] [--
ownerMembershipUuids=abc,bcd] [--ownerStemNames=a:b,b:c] [--ownerStemUuids=1a,2b] [--
ownerMembershipAny0SubjectId=12] [--ownerMembershipAny0SubjectIdentifier=ab] [--
ownerMembershipAny0SourceId=xyz] [--ownerMembershipAny0GroupName=3c] [--ownerMembershipAny0GroupUuid=1a] [--
ownerAttributeAssignUuids=a1,b2] [--
attributeAssignValueOperation=assign_value|add_value|remove_value|replace_values] [--values0Id=a1] [--
values0Formatted=hey] [--values0System=there] [--attributeAssignUuids=a:b,b:c] [--actions=read,write] [--
assignmentDisabledTime=2010/03/05_17:05:13.123] [--assignmentEnabledTime=2010/03/05_17:05:13.123] [--
assignmentNotes=someNotes] [--delegatable=TRUE|FALSE|GRANT] [--attributeDefNamesToReplace=a:b,b:c] [--
attributeDefUuidsToReplace=1a,2b] [--actionsToReplace=read,write] [--attributeDefTypesToReplace=attr,perm,limit,
domain,type] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=assignAttributesWs --attributeAssignType=group --
attributeAssignOperation=assign_attr --attributeDefNameNames=test:testAttributeAssignDefNameDef --
ownerGroupNames=a:b:c
  output line: Index: 0: attributeAssignType: group, owner: test:groupTestAttrAssign, attributeDefNameName:
test:testAttributeAssignDefName, action: assign, values: 15,5,5, enabled: T, id:
a9c83eeb78c04ae5befcea36272d318c, changed: T, deleted: F, valuesChanged: F

assignAttributesBatchWs web service usage (X is the assignment entry starting from and incrementing from 0):
  java -jar grouperClient.jar --operation=assignAttributesBatchWs --
entry_X_attributeAssignType=group|member|stem|any_mem|imm_mem|attr_def|group_asgn|mem_asgn|stem_asgn|any_mem_asg
n|imm_mem_asgn|attr_def_asgn --entry_X_attributeAssignOperation=assign_attr|add_attr|remove_attr [--
entry_X_nameOfAttributeDefName=a:b] [--entry_X_uuidOfAttributeDefName=1a] [--entry_X_ownerNameOfAttributeDef=a:
b] [--entry_X_ownerUuidOfAttributeDef=1a] [--entry_X_ownerGroupName=a:b:c] [--entry_X_ownerGroupUuid=1234] [--
entry_X_ownerSubjectId=subjId0] [--entry_X_ownerSubjectIdentifier=subjIdent0] [--
entry_X_ownerSubjectSource=source0] [--entry_X_ownerMembershipUuid=abc] [--entry_X_ownerStemName=a:b] [--
entry_X_ownerStemUuid=1a] [--entry_X_ownerMembershipAnySubjectId=12] [--
entry_X_ownerMembershipAnySubjectIdentifier=ab] [--entry_X_ownerMembershipAnySourceId=xyz] [--
entry_X_ownerMembershipAnyGroupName=a:b:c] [--entry_X_ownerMembershipAnyGroupUuid=1a] [--
entry_X_ownerAttributeAssignUuid=a1] [--entry_X_ownerAttributeAssignBatchIndex=0] [--
entry_X_attributeAssignValueOperation=assign_value|add_value|remove_value|replace_values] [--
entry_X_values0Id=a1] [--entry_X_values0Formatted=hey] [--entry_X_values0System=there] [--
entry_X_attributeAssignUuid=a:b] [--entry_X_action=read] [--entry_X_assignmentDisabledTime=2010/03/05_17:05:
13.123] [--entry_X_assignmentEnabledTime=2010/03/05_17:05:13.123] [--entry_X_assignmentNotes=someNotes] [--
entry_X_delegatable=TRUE|FALSE|GRANT] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--
subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0]
[--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--
clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=assignAttributesBatchWs --entry_0_attributeAssignType=group --
entry_0_attributeAssignOperation=assign_attr --entry_0_nameOfAttributeDefName=test:
```

```
testAttributeAssignDefNameDef --entry_0_ownerGroupName=a:b:c --entry_1_attributeAssignType=group --
entry_1_attributeAssignOperation=assign_attr --entry_1_nameOfAttributeDefName=test:
testAttributeAssignDefNameDef2 --entry_1_ownerGroupName=a:b:c
  output line: Index: 0, result: 0: attributeAssignType: group, owner: test:groupTestAttrAssign,
attributeDefNameName: test:testAttributeAssignDefName, action: assign, values: 15,5,5, enabled: T, id:
a9c83eeb78c04ae5befcea36272d318c, changed: T, deleted: F, valuesChanged: F

getPermissionAssignmentsWs web service usage:
  java -jar grouperClient.jar --operation=getPermissionAssignmentsWs [--includeAttributeAssignments=true|false]
[--includeAssignmentsOnAssignments=true|false] [--includeAttributeDefNames=true|false] [--
includePermissionAssignDetail=true|false] [--attributeDefNames=a:b,b:c] [--attributeDefUuids=1a,2b] [--
attributeDefNameNames=a:b,b:c] [--attributeDefNameUuids=1a,2b] [--roleNames=a:b:c,a:b:d] [--roleUuids=1234,
abcd] [--subject0SubjectId=subjId0] [--subject0SubjectIdentifier=subjIdent0] [--subject0SubjectSource=source0]
[--enabled=A|T|F] [--actions=read,write] [--includeGroupDetail=true|false] [--includeSubjectDetail=true|false]
[--subjectAttributeNames=name0,name1] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--pointInTimeFrom=yyyy/mm/dd hh:mi:ss] [--pointInTimeTo=yyyy/mm/dd hh:mi:ss] [--
immediateOnly=T|F] [--permissionType=role_subject|role] [--
permissionProcessor=FILTER_REDUNDANT_PERMISSIONS|FILTER_REDUNDANT_PERMISSIONS_AND_PROCESS_LIMITS|FILTER_REDUNDAN
T_PERMISSIONS_AND_ROLES|FILTER_REDUNDANT_PERMISSIONS_AND_ROLES_AND_PROCESS_LIMITS|PROCESS_LIMITS] [--
limitEnvVarName0=name0] [--limitEnvVarValue0=value0] [--
limitEnvVarType0=integer|decimal|date|timestamp|text|boolean|null|emptyString] [--limitEnvVarNameX=xthName] [--
limitEnvVarValueX=xthValue] [--limitEnvVarTypeX=xthType] [--includeLimits=T|F] [--saveResultsToFile=fileName]
[--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=getPermissionAssignmentsWs --permissionType=role_subject --
attributeDefNames=test:testAttributeAssignDefNameDef
  output line: Index: 0: permissionType: role_subject, role: test:someRole, subject: 123456,
attributeDefNameName: test:testPermission, action: assign, allowedOverall: T, enabled: T

assignPermissionsWs web service usage:
  java -jar grouperClient.jar --operation=assignPermissionsWs --permissionType=role|role_subject --
permissionAssignOperation=assign_permission|remove_permission|replace_permissions [--permissionDefNameNames=a:b,
b:c] [-permissionDefNameUuids=1a,2b] [--roleNames=a:b:c,a:b:d] [--roleUuids=1234,abcd] [--
subjectRole0SubjectId=12] [--subjectRole0SubjectIdentifier=ab] [--subjectRole0SourceId=xyz] [--
subjectRole0RoleName=3c] [--subjectRole0RoleUuid=1a] [--attributeAssignUuids=a:b,b:c] [--actions=read,write] [--
disallowed=true|false] [--assignmentDisabledTime=2010/03/05_17:05:13.123] [--assignmentEnabledTime=2010/03
/05_17:05:13.123] [--assignmentNotes=someNotes] [--delegatable=TRUE|FALSE|GRANT] [--
includeGroupDetail=true|false] [--includeSubjectDetail=true|false] [--subjectAttributeNames=name0,name1] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--attributeDefNamesToReplace=a:b,b:c] [--
attributeDefUuidsToReplace=1a,2b] [--actionsToReplace=read,write] [--paramName0=name0] [--paramValue0=value1]
[--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=assignPermissionsWs --permissionType=role --
permissionAssignOperation=assign_permission --permissionDefNameNames=test:testAttributeAssignDefNameDef --
roleNames=a:b:c
  output line: Index: 0: permissionType: role, owner: a:b:c, permissionDefNameName: test:
testAttributeAssignDefName, action: assign, disallowed: T, enabled: T, attributeAssignId:
a9c83eeb78c04ae5befcea36272d318c, changed: T, deleted: F

attributeDefNameSaveWs web service usage:
  java -jar grouperClient.jar --operation=attributeDefNameSaveWs --name=a:b:c [--
saveMode=INSERT_OR_UPDATE|INSERT|UPDATE] [--attributeDefNameLookupName=a:b:c] [--
attributeDefNameLookupUuid=sd87f-dsf87-sdf89-df78f] [--description=theDescription] [--
displayExtension=theDisplayExtension] [--createParentStemsIfNotExist=true|false] [--uuidOfAttributeDef=sd87f-
dsf87-sdf89-df78f] [--nameOfAttributeDef=a:b:c] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent]
[--actAsSubjectSource=source] [--saveResultsToFile=fileName] [--outputTemplate=somePattern] [--
paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--paramValueX=xthParamValue] [--
debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=attributeDefNameSave --name=aStem:aGroup
  output: Success: T: code: SUCCESS_INSERTED: aStem:aGroup

attributeDefNameDeleteWs web service usage:
  java -jar grouperClient.jar --operation=attributeDefNameDeleteWs --
attributeDefNameNames=attributeDefNameName0,attributeDefNameName1 [--txType=NONE|READ_WRITE_NEW] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
saveResultsToFile=fileName] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=attributeDefNameDelete --attributeDefNameNames=aStem:
anAttributeDefName0,aStem:anAttributeDefName1
  output line: Index 0: success: T: code: SUCCESS: aStem:anAttributeDefName0
```

```
assignAttributeDefNameInheritanceWs web service usage:
  java -jar grouperClient.jar --operation=assignAttributeDefNameInheritanceWs --
attributeDefNameName=attributeDefNameName0 --relatedAttributeDefNameNames=relatedName0,relatedName1 --
assign=T|F [--replaceAllExisting=T|F] [--txType=NONE|READ_WRITE_NEW] [--actAsSubjectId=subjId] [--
actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--saveResultsToFile=fileName] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=assignAttributeDefNameInheritanceWs --
attributeDefNameName=aStem:anAttributeDefName0 --relatedAttributeDefNameNames=aStem:anAttributeDefName1 --
assign=T
  output line: code: SUCCESS, message: Had 1 successful adds, 0 adds which already existed, 0 successful
removes, and 0 removes which didnt exist.

findAttributeDefNamesWs web service usage
  java -jar grouperClient.jar --operation=findAttributeDefNamesWs [--scope=some:scopeOrTerms] [--
splitScope=T|F] [--attributeDefNameNames=a:b,b:c] [--attributeDefNameUuids=12ab,23cd] [--nameOfAttributeDef=a:b:
c] [--uuidOfAttributeDef=12fg-34fg] [--
attributeAssignType=any_mem|any_mem_asgn|attr_def|attr_def_asgn|group|group_asgn|imm_mem|imm_mem_asgn|mem_asgn|m
ember|stem|stem_asgn] [--attributeDefType=attr|domain|limit|perm|type] [--
inheritanceSetRelation=IMPLIED_BY_THIS|IMPLIED_BY_THIS_IMMEDIATE|THAT_IMPLY_THIS|THAT_IMPLY_THIS_IMMEDIATE] [--
sortString=name|displayName|extension|displayExtension] [--ascending=T|F] [--pageNumber=2] [--pageSize=50] [--
actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--actAsSubjectSource=source] [--
outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--paramNameX=xthParamName] [--
paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion]
  e.g.: java -jar grouperClient.jar --operation=findAttributeDefNamesWs --scope=aStem:
  output: Index 0: name: aStem:anAttributeDefName, displayName: A stem:An AttributeDefName

sendMessageWs web service usage
  java -jar grouperClient.jar --operation=sendMessageWs --queueType=queue|topic --
queueOrTopicName=queue_or_topic_name --messageBody0=test-message-body [--messageBodyX=message body x] [--
messagingSystemName=someMessagingSystemName] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion] [--
routingKey=routing-key] [--autocreateObjects=T|F]
  e.g.: java -jar grouperClient.jar --operation=sendMessageWs --queueType=topic --queueOrTopicName=test-topic3
--messageBody0=test-message-body --messagingSystemName=rabbitMqMessaging --routingKey=test-key --
auotcreateObjects=T --paramName0=exchangeType --paramValue0=TOPIC
  output line: Success: T, queueOrTopicName=test-topic3, numberOfMessages=1


receiveMessageWs web service usage
  java -jar grouperClient.jar --operation=receiveMessageWs --queueOrTopicName=queue_name [--
messagingSystemName=someMessagingSystemName] [--actAsSubjectId=subjId] [--actAsSubjectIdentifier=subjIdent] [--
actAsSubjectSource=source] [--outputTemplate=somePattern] [--paramName0=name0] [--paramValue0=value1] [--
paramNameX=xthParamName] [--paramValueX=xthParamValue] [--debug=true] [--clientVersion=someVersion] [--
routingKey=routing-key] [--autocreateObjects=T|F]
  e.g.: java -jar grouperClient.jar --operation=receiveMessageWs --queueOrTopicName=test-queue --
messagingSystemName=grouperBuiltinMessaging --auotcreateObjects=T --paramName0=queueType --paramValue0=queue
  output line: Index 0: success: T, queueOrTopicName: test-queue, messageBody: test-message-body

##############################################
Common options:
  --outputTemplate=${index}: ${wsGroup.name}
    the output template allow the caller to customize what is displayed in the output from the XML
    anything in ${} will be evaluated, and there are different variables available for various operations.
    if you pass in --debug=true, it will tell you the xml and the variables you can use.  You can drill down
    in the variables, e.g. ${wsGroupDeleteResult.wsGroup.name}, you can do operations, e.g. ${index+1},
    you can do simple string utilities from GrouperClientUtils or GrouperClientCommonUtils, e.g.
    ${grouperClientUtils.trimToEmpty(wsGroup.name)}

  --debug=true
    this will display debug information including the request and response to stderr

  --saveResultsToFile=/tmp/somefile.txt
    you can save the stdout to a file if you like

  --actAsSubjectId=subjId --actAsSubjectIdentifier=subjIdent --actAsSubjectSource=source
    if you want to run the operation as a different user than the user who is authenticating
    to the web service, then specify the actAsSubjectId or actAsSubjectIdentifier (and optionally
    the actAsSubjectSource).  You would do this e.g. to run a command as admin, or as a user who
    is using the end layer application.  Note you need permissions to do this in grouper.
```

```
  --paramName0=name0 --paramValue0=value1 --paramNameX=xthParamName --paramValueX=xthParamValue
    you can specify params in name/value pairs if the operation supports it (see grouper
    web service documentation for details)

  --clientVersion=someVersion
    generally this does not need to be changed.  This is the version label sent to the web service
    which might affect the output from the web service.  Not it does not affect the request to the
    web service (besides the label), it only affect the response from the web service.

  --txType=GcTransactionType
    affects how batched operations are executed on the server (e.g. adding multiple subjects to a group)
    generally the only values which make sense are to use a large transaction or not: READ_WRITE_NEW, NONE

  --includeGroupDetail=true
    if applicable, this option will return not only the group's name, but more information such as the
    attribuites, types, composite members, etc.

  --subjectAttributeNames=a,b,c
    if applicable, subjects will be returned from the server with these attributes in a string array
```

To use grouperClient as a Java API, just add the grouperClient.jar to your classpath (e.g. in your WEB-INF/lib directory for a web app), add grouper.client.properties to your classpath, then use the classes generally in the edu.internet2.middleware.grouperClient.api package. e.g.

```
      WsAddMemberResults wsAddMemberResults = new GcAddMember().assignGroupName("aStem:aGroup").addSubjectId
("12345").execute();
```

Here is an example of finding a stem

```
/*
 * @author mchyzer
 * $Id$
 */
package edu.internet2.middleware.grouperClient.poc;

import edu.internet2.middleware.grouperClient.api.GcFindStems;
import edu.internet2.middleware.grouperClient.ws.beans.WsFindStemsResults;
import edu.internet2.middleware.grouperClient.ws.beans.WsResultMeta;
import edu.internet2.middleware.grouperClient.ws.beans.WsStem;
import edu.internet2.middleware.grouperClient.ws.beans.WsStemQueryFilter;


/**
 *
 */
public class FindStem {

  /**
   * @param args
   */
  public static void main(String[] args) {

    GcFindStems gcFindStems = new GcFindStems();

    WsStemQueryFilter wsStemQueryFilter = new WsStemQueryFilter();
    wsStemQueryFilter.setStemName("penn");
    wsStemQueryFilter.setStemQueryFilterType("FIND_BY_STEM_NAME_APPROXIMATE");

    gcFindStems.assignStemQueryFilter(wsStemQueryFilter);

    WsFindStemsResults wsFindStemsResults = gcFindStems.execute();

    WsResultMeta resultMetadata = wsFindStemsResults.getResultMetadata();

    if (!"T".equals(resultMetadata.getSuccess())) {
      throw new RuntimeException("Error finding stems: " + resultMetadata.getSuccess()
          + ", " + resultMetadata.getResultCode()
          + ", " + resultMetadata.getResultMessage());
    }

    WsStem[] wsStems = wsFindStemsResults.getStemResults();

    if (wsStems != null) {
      for (WsStem wsStem : wsStems) {
        System.out.println(wsStem.getName());
      }
    }
  }
}
```

Note: you can use method chaining for compact usage, or put each parameter in its own statement.

Use cases currently implemented

1. Connect to LDAP SSL from windows freely and easily
2. Run LDAP lookups that might not even be grouper related (e.g. we want to convert a pennid to a pennkey, in our grouper ldap)
3. Run LDAP queries for hasMember, getMembers
4. Get the members of a group from LDAP, and store in a file, with each subjectId on a line
5. Get the members of a group from web services, and store a certain subject attribute in a file, with each one on its own line
6. Take a file of subjectIdentifiers, and replace the members of a group with it
7. All of the web service calls (non-Lite where applicable)
8. Send an XML file to web services, receive an XML file
9. Encrypt passwords

Using (from Grouper deployer's perspective)

Get the binary release:

1. Unzip, configure the grouper.client.properties and grouper.client.usage.txt, and use

Get the source release, unzip, cd to the dir

```
ant
##### NOW CUSTOMIZE THE conf/grouper.client.properties, conf/grouper.client.usage.txt, misc/README.txt
ant
##### OUTPUT is in dist dir: grouperClient.jar, or grouperClient.institution-1.4.0.tar.gz
```

Checkout grouper client:

```
cvs -d:pserver:anoncvs@anoncvs.internet2.edu:/home/cvs/i2mi login
cvs -d:pserver:anoncvs@anoncvs.internet2.edu:/home/cvs/i2mi co grouper-misc/grouperClient
cd grouper-misc\grouperClient
ant
##### NOW CUSTOMIZE THE conf/grouper.client.properties, conf/grouper.client.usage.txt, misc/README.txt
ant
##### OUTPUT is in dist dir: grouperClient.jar, or grouperClient.institution-1.4.0.tar.gz
```

Now you can zip up the grouperClient.jar, grouper.client.properties, and a README.txt and post on a website for your department users to download. Of course no credentials should be in the zip, the users can fill these in when they are authorized.

Customizing the grouper.client.properties

Here is the example grouper.client.properties

```
#
# Grouper client configuration
# $Id: grouper.client.example.properties,v 1.3 2008/12/01 07:40:28 mchyzer Exp $
#

########################################
## LDAP connection settings
########################################

# url of directory, including the base DN (distinguished name)
# e.g. ldap://server.school.edu/dc=school,dc=edu
# e.g. ldaps://server.school.edu/dc=school,dc=edu
grouperClient.ldap.url =

# kerberos principal used to connect to ldap
grouperClient.ldap.kerberosPrincipal =

# password for shared secret authentication to ldap
# or you can put a filename with an encrypted password
grouperClient.ldap.password =
```

The above section is generally for the user, though the url can be filled in when distributing to users

---

```
########################################
## Web service Connection settings
########################################

# url of web service, should include everything up to the first resource to access
# e.g. http://groups.school.edu:8090/grouperWs/servicesRest
# e.g. https://groups.school.edu/grouperWs/servicesRest
grouperClient.webService.url =

# kerberos principal used to connect to web service
grouperClient.webService.kerberosPrincipal =

# password for shared secret authentication to web service
# or you can put a filename with an encrypted password
grouperClient.webService.password =
```

The above section is generally for users, though the url can be filled in before distributing to users

---

```
########################################
## Encrypted password settings
########################################

# Put a random alphanumeric string (Case sensitive) for the password encryption.  e.g. fh43IRJ4Nf5
# or put a filename where the random alphanumeric string is.
# e.g. c:/whatever/key.txt
# e.g. sdfklj24lkj34lk34
encrypt.key =

# set this to true if you have slashes in your passwords and dont want to look in external files or unencrypt
encrypt.disableExternalFileLookup = false
```

grouperClient contains a version of i2mi morphString to keep passwords encrypted in external files from the config file (e.g. so the config file can be more safely distributed, or stored in version control)

```
#########################################
## Logging
#########################################

# For java.util.logging, only for the grouperClient package (not below)
# from java java.util.logging.Level class: ALL, CONFIG, FINE, FINER, FINEST, OFF, SEVERE, WARNING
grouperClient.logging.grouperClientOnly.logLevel = WARNING

# If you are not using log4j (will use java.util.logging, you can turn logging on which will go to stderr
# (if no file specified below).  This is default log level
# from java java.util.logging.Level class: ALL, CONFIG, FINE, FINER, FINEST, OFF, SEVERE, WARNING
grouperClient.logging.logLevel = WARNING

# If you dont want the logging to go to stderr, then put a lot file location here: e.g. f:/temp/grouperClient.
log
grouperClient.logging.logFile =

# if you want ws requests and responses being logged to files, put the directory here.
# The grouper client will create subdirs
grouperClient.logging.webService.documentDir =

# try to indent the xml.  If this fails for some reason, or you want the raw xml,
# set to false
grouperClient.logging.webService.indent = true
```

grouperClient contains a version of commons logging. So if used by itself, very basic logging will be used, either to stderr, or to a log file is specified above. There is a lot of debug logging, so if you are having issues, set the grouperClient.logging.grouperClientOnly.logLevel to ALL. You can segregate the grouperClient log level, from everything else. Also, you can store all web service files (e.g. for debugging purposes). Also, these can be indented for easy reading. Note: you can pass in --debug=true to see debug logs and web service request/responses in stderr.

```
##########################################################################################
##########################################################################################
#### Institutional and advanced settings
##########################################################################################
##########################################################################################

########################################
## output templates
########################################

webService.addMember.output = Index ${index}: success: ${resultMetadata.success}: code: ${resultMetadata.
resultCode}: ${wsSubject.id}$newline$
webService.getMembers.output = GroupIndex ${groupIndex}: success: ${resultMetadata.success}: code:
${resultMetadata.resultCode}: group: ${wsGroup.name}: subjectIndex: ${subjectIndex}: ${wsSubject.id}$newline$
webService.deleteMember.output = Index ${index}: success: ${resultMetadata.success}: code: ${resultMetadata.
resultCode}: ${wsSubject.id}$newline$
webService.hasMember.output = Index ${index}: success: ${resultMetadata.success}: code: ${resultMetadata.
resultCode}: ${wsSubject.id}: ${hasMember}$newline$
webService.getGroups.output = SubjectIndex ${subjectIndex}: success: ${resultMetadata.success}: code:
${resultMetadata.resultCode}: subject: ${wsSubject.id}: groupIndex: ${groupIndex}: ${wsGroup.name}$newline$
webService.groupSave.output = Success: ${resultMetadata.success}: code: ${resultMetadata.resultCode}: ${wsGroup.
name}$newline$
webService.stemSave.output = Success: ${resultMetadata.success}: code: ${resultMetadata.resultCode}: ${wsStem.
name}$newline$
webService.groupDelete.output = Index ${index}: success: ${resultMetadata.success}: code: ${resultMetadata.
resultCode}: ${wsGroup.name}$newline$
webService.stemDelete.output = Index ${index}: success: ${resultMetadata.success}: code: ${resultMetadata.
resultCode}: ${wsStem.name}$newline$
webService.getGrouperPrivilegesLite.output = Index ${index}: success: ${resultMetadata.success}: code:
${resultMetadata.resultCode}: ${objectType}: ${objectName}: subject: ${wsSubject.id}:
${wsGrouperPrivilegeResult.privilegeType}: ${wsGrouperPrivilegeResult.privilegeName}$newline$
webService.assignGrouperPrivilegesLite.output = Success: ${resultMetadata.success}: code: ${resultMetadata.
resultCode}: ${objectType}: ${objectName}: subject: ${wsSubject.id}: ${wsAssignGrouperPrivilegesLiteResult.
privilegeType}: ${wsAssignGrouperPrivilegesLiteResult.privilegeName}$newline$
webService.findGroups.output = Index ${index}: name: ${wsGroup.name}, displayName: ${wsGroup.
displayName}$newline$
webService.findStems.output = Index ${index}: name: ${wsStem.name}, displayName: ${wsStem.displayName}$newline$
webService.memberChangeSubject.output = Success: ${resultMetadata.success}: code: ${resultMetadata.resultCode}:
oldSubject: ${wsSubjectOld.id}, newSubject: ${wsSubjectNew.id}$newline$
```

Note that the settings the end user is likely to need to to change are up top in the config file. Output templates are central to grouper client, so that the command line output can be parsed easily by clients, or used in other programs. The syntax is Java EL, and uses a version of the jakarta library jexl. In different circumstances different objects are in scope, this needs more documentation and examples, but the point is that you can customize the output to suit your needs, and make sure it will not change with upgrades.

```
#######################################
## ldap queries
#######################################

# operation name
ldapSearchAttribute.operationName.0 = pennnameToPennid
ldapSearchAttribute.ldapName.0 = ou=pennnames
ldapSearchAttribute.matchingAttributes.0 = pennname
ldapSearchAttribute.matchingAttributeLabels.0 = pennnameToDecode
ldapSearchAttribute.returningAttributes.0 = pennid
ldapSearchAttribute.outputTemplate.0 = pennid: ${pennid}
ldapSearchAttribute.resultType.0 = STRING

ldapSearchAttribute.operationName.1 = pennidToPennname
ldapSearchAttribute.ldapName.1 = ou=pennnames
ldapSearchAttribute.matchingAttributes.1 = pennid
ldapSearchAttribute.matchingAttributeLabels.1 = pennidToDecode
ldapSearchAttribute.returningAttributes.1 = pennname
ldapSearchAttribute.outputTemplate.1 = pennname: ${pennname}
ldapSearchAttribute.resultType.1 = STRING

ldapSearchAttribute.operationName.2 = hasMemberLdap
ldapSearchAttribute.ldapName.2 = ou=groups
ldapSearchAttribute.matchingAttributes.2 = cn, hasMember
ldapSearchAttribute.matchingAttributeLabels.2 = groupName, pennnameToCheck
ldapSearchAttribute.returningAttributes.2 = cn
ldapSearchAttribute.outputTemplate.2 = isInGroup: ${resultBoolean}
ldapSearchAttribute.resultType.2 = BOOLEAN

ldapSearchAttribute.operationName.3 = getMembersLdap
ldapSearchAttribute.ldapName.3 = ou=groups
ldapSearchAttribute.matchingAttributes.3 = cn
ldapSearchAttribute.matchingAttributeLabels.3 = groupName
ldapSearchAttribute.returningAttributes.3 = hasMember
ldapSearchAttribute.outputTemplate.3 = ${resultString}$newline$
ldapSearchAttribute.resultType.3 = STRING_LIST
```

The LDAP API is very generic. Right now simple attribute lookups are supported, checking to see if there is an attribute match, or listing a multi-valued attribute. More documentation is needed here, and perhaps more options, let us know what you need for ldap access.

```
#########################################
## Authentication settings
#########################################

# user prefix
grouperClient.ldap.user.prefix = uid=

# user suffix
grouperClient.ldap.user.suffix = ,ou=entities,dc=upenn,dc=edu

# config name for the ldap user name between prefix and suffix
grouperClient.ldap.user.label = kerberosPrincipal

# config name for the webService user name between prefix and suffix
grouperClient.webService.user.label = kerberosPrincipal

#version of the output, as we upgrade the client, we will maintain previous output versions
grouperClient.output.version = 1.4.0
```

To authenticate to LDAP the username might not need to be exposed to the end user, so you can put a prefix and suffix here. Also, the label used in the config file for the login id can be cusotmized to make it easier to use.

```
#########################################
## Web service settings
#########################################

# web service client version
grouperClient.webService.client.version = v1_4_000

# socket timeout
grouperClient.webService.httpSocketTimeoutMillis = 90000

# connection manager timeout
grouperClient.webService.httpConnectionManagerTimeoutMillis = 90000

# ignore extraneous xml fields from server (e.g. on server upgrade, when the client isnt upgraded)
# if you dont ignore, and there is an extraneous field which is not omitted (below), then an exception
# will be thrown
grouperClient.webService.ignoreExtraneousXmlFields = true

# register fields to be ignored with xstream.  this is useful if you are not
# ignoring extraneous fields (above), but know that there are a few to be ignored
# place them here with fully qualified classname dont property name, comma separated
# e.g. edu.internet2.middleware.grouperClient.ws.beans.WsResponseMeta.millis, edu.internet2.middleware.
grouperClient.ws.beans.WsResponseMeta.millis2
grouper.webService.omitXmlProperties =
```

The timeouts and client version are stored here. If you want to ignore some XML that clients send that is not valid (e.g. if the service has changed, and there are old clients), then you can specify here. Also you can call out any specific properties in objects to ignore (inbound or outbound)

```
#########################################
## Misc
#########################################

# if there are extra command line args, should we fail or just log?
grouperClient.failOnExtraCommandLineArgs = true

# you can have aliases for subjectId and subjectIdentifer in command line args
# (though subjectId will still be allowed, but you cant pass both)
# if this value is pennIds, then e.g. for addMemberWs, you can use --pennIds=123,234
# instead of --subjectIds=123,345
grouperClient.alias.subjectIds =

# if this value is pennKeys, then e.g. for addMemberWs, you can use --pennKeys=abc,bcd
# instead of --subjectIdentifiers=abc,bcd
grouperClient.alias.subjectIdentifiers =

# if this value is pennId, then e.g. for getGrouperPrivilegesLite, you can use --pennId=123
# instead of --subjectId=123
grouperClient.alias.subjectId =

# if this value is pennKey, then e.g. for getGrouperPrivilegesLite, you can use --pennKey=abc
# instead of --subjectIdentifiers=abc
grouperClient.alias.subjectIdentifier =

# if this value is PennId, then e.g. for addMemberWs, you can use --actAsPennId=123
# instead of --actAsSubjectId=abc,bcd
grouperClient.alias.SubjectId =

# if this value is PennKey, then e.g. for addMemberWs, you can use --actAsPennKey=abc
# instead of --actAsSubjectIdentifier=abc
grouperClient.alias.SubjectIdentifier =

# this should probably be changed to UTF-8 for international charsets... for US it can be: ISO-8859-1
grouperClient.default.fileEncoding = ISO-8859-1
```

If an invalid option is passed in, should it throw an error?

Also, you can put aliases on arguments that are for subjectId and subjectIdentifier. This means that this alternate argument name can be used instead of subjectId and subjectIdentifier. You can use either the alias or the original name, but not both at the same time. The examples above show what we are doing at Penn, where subjectId is pennId, and subjectIdentifier is pennKey.

Output templates

Per the usage readme:

```
--outputTemplate=${index}: ${wsGroup.name}
  the output template allow the caller to customize what is displayed in the output from the XML
  anything in ${} will be evaluated, and there are different variables available for various operations.
  if you pass in --debug=true, it will tell you the xml and the variables you can use.  You can drill down
  in the variables, e.g. ${wsGroupDeleteResult.wsGroup.name}, you can do operations, e.g. ${index+1},
  you can do simple string utilities from GrouperClientUtils or GrouperClientCommonUtils, e.g.
  ${grouperClientUtils.trimToEmpty(wsGroup.name)}
```

This uses the jakarta library JEXL.

The easiest way to use this, is first to do a request with debug mode:

```
C:\dev_inst\eclipse\workspace_v33\grouperClient\dist>java -jar grouperClient.jar --operation=addMemberWs --
groupName=aStem:aGroup --subjectIds=test.subject.0 --debug=true

...
################ RESPONSE START (indented) ###############

HTTP/1.1 201 Created
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=xxxxxxxxxxxx; Path=/grouperWs
X-Grouper-resultCode: SUCCESS
X-Grouper-success: T
X-Grouper-resultCode2: NONE
Content-Type: text/xml
Date: Mon, 08 Dec 2008 05:43:36 GMT
Connection: close

<WsAddMemberResults>
  <results>
    <WsAddMemberResult>
      <wsSubject>
        <resultCode>SUCCESS</resultCode>
        <success>T</success>
        <id>test.subject.0</id>
        <sourceId>jdbc</sourceId>
      </wsSubject>
      <resultMetadata>
        <resultCode>SUCCESS_ALREADY_EXISTED</resultCode>
        <success>T</success>
      </resultMetadata>
    </WsAddMemberResult>
  </results>
  <wsGroupAssigned>
    <extension>aGroup</extension>
    <displayExtension>aGroup</displayExtension>
    <displayName>aStem:aGroup</displayName>
    <name>aStem:aGroup</name>
    <uuid>01a1d70c-df7c-4ffa-b6ed-f90fa7c37f6b</uuid>
  </wsGroupAssigned>
  <resultMetadata>
    <resultCode>SUCCESS</resultCode>
    <resultMessage>Success for: clientVersion: v1_4_000, wsGroupLookup: WsGroupLookup[groupName=aSte
m:aGroup], subjectLookups: Array size: 1: [0]: WsSubjectLookup[subjectId=test.subject.0]

, replaceAllExisting: false, actAsSubject: null, fieldName: null, txType: NONE, includeGroupDetail:
false, includeSubjectDetail: false, subjectAttributeNames: null
, params: null</resultMessage>
    <success>T</success>
  </resultMetadata>
  <responseMetadata>
    <millis>453</millis>
    <serverVersion>v1_4_000</serverVersion>
  </responseMetadata>
</WsAddMemberResults>
################ RESPONSE END ###############
...
DEBUG: Output template: Index ${index}: success: ${resultMetadata.success}: code: ${resultMetadata.resultCode}:
${wsSubject.id}
, available variables: wsAddMemberResults, grouperClientUtils, wsGroupAssigned, index, wsAddMemberResult,
wsSubject, resultMetadata
```

You can see what the available variables are, and you can see the XML response.

So, if you want to print out some variables, you can use the objects available, and drill down by looking at what is available in the xml:

```
C:\grouper>java -jar grouperClient.jar --operation=addMemberWs --groupName=aStem:aGroup --subjectIds=test.
subject.0 --outputTemplate="sourceId: ${wsAddMemberResult.wsSubject.sourceId}, uuid: ${wsGroupAssigned.uuid}"
sourceId: jdbc, uuid: 01a1d70c-df7c-4ffa-b6ed-f90fa7c37f6b
C:\grouper>
```

If you evaluate an expression which returns null (which is what is returned if a variable doesnt exist), a warning will be displayed to stderr, but the stdout will still be intact. If this is intended, use grouperClientUtils.defaultString() and it will not return null.

Custom operations

If you want the grouper client to execute custom Java operations, then follow these instructions. For example, at Penn we will have a couple of operations that decode Cosign single-signon tokens.

- Put your code in grouperClientHome/src/custom (this is for the code which depends on ext or extCustom code, but nothing else)
- If you have external code, put that in grouperClientHome/src/extCustom (this is for typically 3rd party code which might depend on other jars at compile time, but not runtime)
- If you have jars for extCustom, but them in grouperClientHome/lib/custom
- The class which responds to an operation should implement the interface: edu.internet2.middleware.grouperClient.ClientOperation
- This has one method: public String operate(OperationParams operationParams);
- Register this in the grouper.client.properties:

```
########################################
## Custom operations
## Implement the interface ClientOperation, put it in the jar
## Increment the int index for multiples (must be in order)
########################################

customOperation.name.0 = cosignDecode
customOperation.class.0 = edu.upenn.isc.grouperClient.CosignDecodeOperation
```

* Implement the interface with the logic, and get params from the command line:

```
  /**
   * @see edu.internet2.middleware.grouperClient.ClientOperation#operate(edu.internet2.middleware.grouperClient.
OperationParams)
   */
  public String operate(OperationParams operationParams) {

    Map<String, String> argMap = operationParams.getArgMap();
    Map<String, String> argMapNotUsed = operationParams.getArgMapNotUsed();

    //get params from command line
    String serviceName = GrouperClientUtils.argMapString(argMap, argMapNotUsed, "serviceName", true);
    String cookie = GrouperClientUtils.argMapString(argMap, argMapNotUsed, "cosignCookie", true);

    //get params from grouper.client.properties
    String keyStorePath = GrouperClientUtils.propertiesValue("cosign.keyStorePath", true);

    ... etc, execute the logic, and return the result which should be printed to screen or written to file
```

* Build with: ant

- Call the operation from the command line:

  ```
  C:\grouperClient\dist\institution\grouperClient.institution-1.4.0>java -jar grouperClient.jar --
  operation=cosignDecode --serviceName=cosign-isc-whatever-0 --cosignCookie=0mmN5ZwyJukNxxxxxxxxx
  203-PennNet ID  mchyzer
  203-8-digit PennID      123456
  203-Timestamp   1111111111
  203 IP Address  1.2.3.4
  C:\grouperClient\dist\institution\grouperClient.institution-1.4.0>
  ```

  sdf

Misc

If you don't want to validate the SSL (e.g. self signed certificate) follow these instructions in grouperClient.properties

```
# to not require valid SSL, use: edu.internet2.middleware.grouperClient.ssl.EasySslSocketFactory
grouperClient.https.customSocketFactory =

# to not require valid SSL, use: edu.internet2.middleware.grouperClient.ssl.BlindSslSocketFactory
grouperClient.ldaps.customSocketFactory =
```

Todo

- Add uuids to the client operations

See Also

Grouper Always Available Web Services and Client

Failover Client

Grouper Discovery Client