How to Setup a Grouper Development Environment for Grouper v2.5

This how-to describes how to set up a Grouper development environment so that you can code, test, and debug Grouper.

This page is specific to Grouper v2.5.

Instructions for Grouper v2.4 are found at How to Setup a Grouper Development Environment for Grouper 2.4.

This is a "no build" dev env where

- the developer does not run ant or maven on an ongoing basis
- the IDE compiles java classes and copies files around so that Java classes can be run (e.g. unit tests, grouper shell, grouper client, any main())
- the UI and WS webapp is ready be run by Tomee without any ant or maven, and just uses the compiled classes and files from the IDE
- no grouper dependencies need to be compiled or built or copied it is all done by the IDE automatically

Developers should understand how it works since it is a little involved

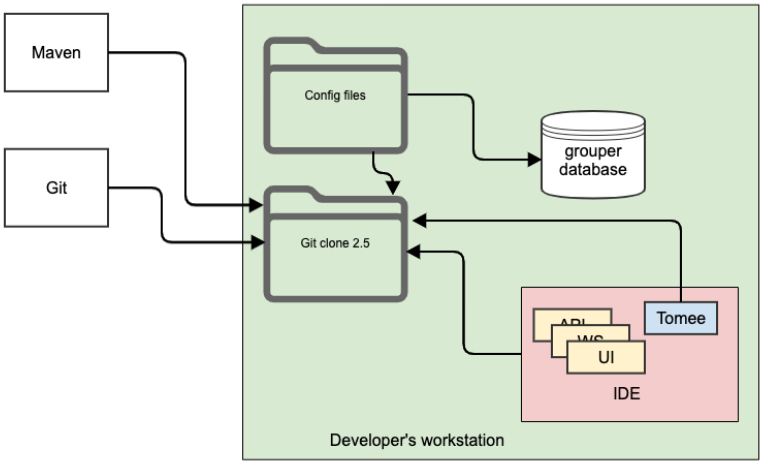
Note, if using Java 17, pass this argument to tests and tomcat

```
--add-opens java.base/java.lang=ALL-UNNAMED --add-opens java.base/java.util=ALL-UNNAMED --add-opens java.sql
/java.sql=ALL-UNNAMED
```

Troubleshooting

Issue	Resolution
3rd party class not found Something not compiling Config not found	Grouper-parent clean and install Run the maven copy dependencies (if webapp) Refresh project Project clean (in eclipse), select applicable project
Class not found in webapp for command line program	Java Build Path Libraries Add external class folder: webapp/WEB-INF/classes
Still having issues	Look where things compile in file explorer in your OS, and see that all classes and configs are there Make sure all linked source configs are right
Linked source already exists	The name conflicts with existing or previous folder. Cancel out and delete the folder in the project which used to link. Or give it a different name
Crashing	Add more memory? Should have at least 3 gigs for eclipse

Grouper dev env high level diagram

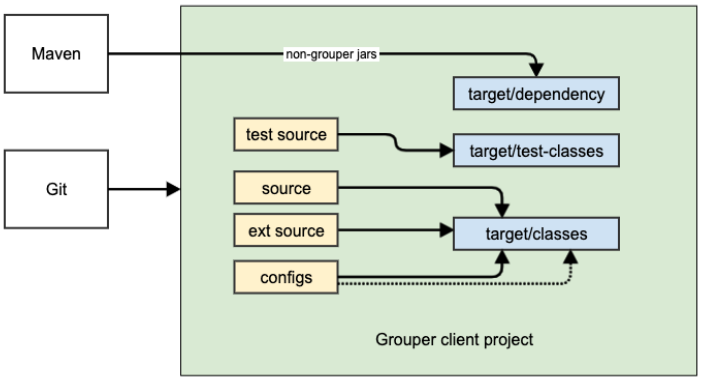


In general the dev env uses:

- Java 8
- IDE (e.g. eclipse)
- Git clone
- Tomee (with tomcat 8.5)

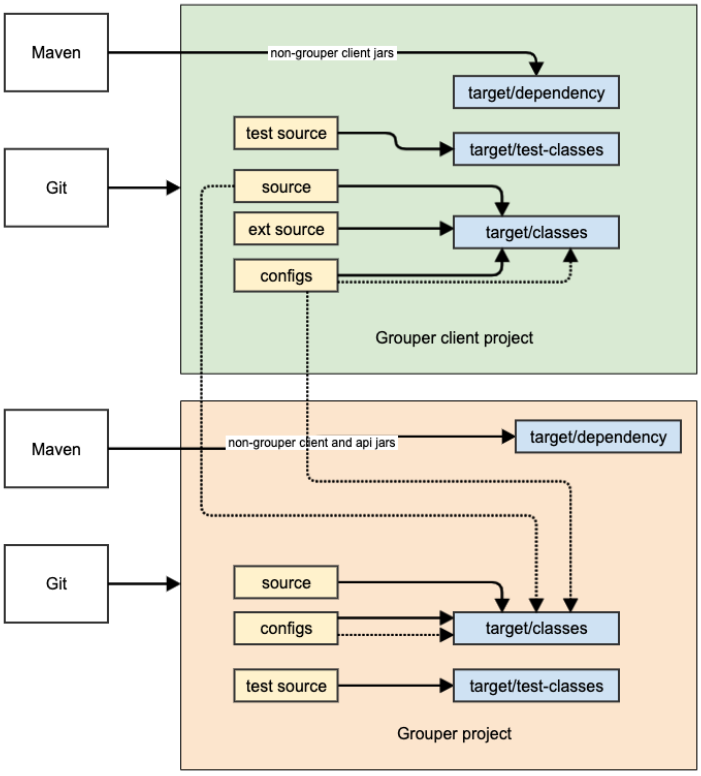
Here are diagrams for the client (base dependency of grouper), api (next dependency), and UI (example of a third and final level dependency)

Grouper client dev env diagram



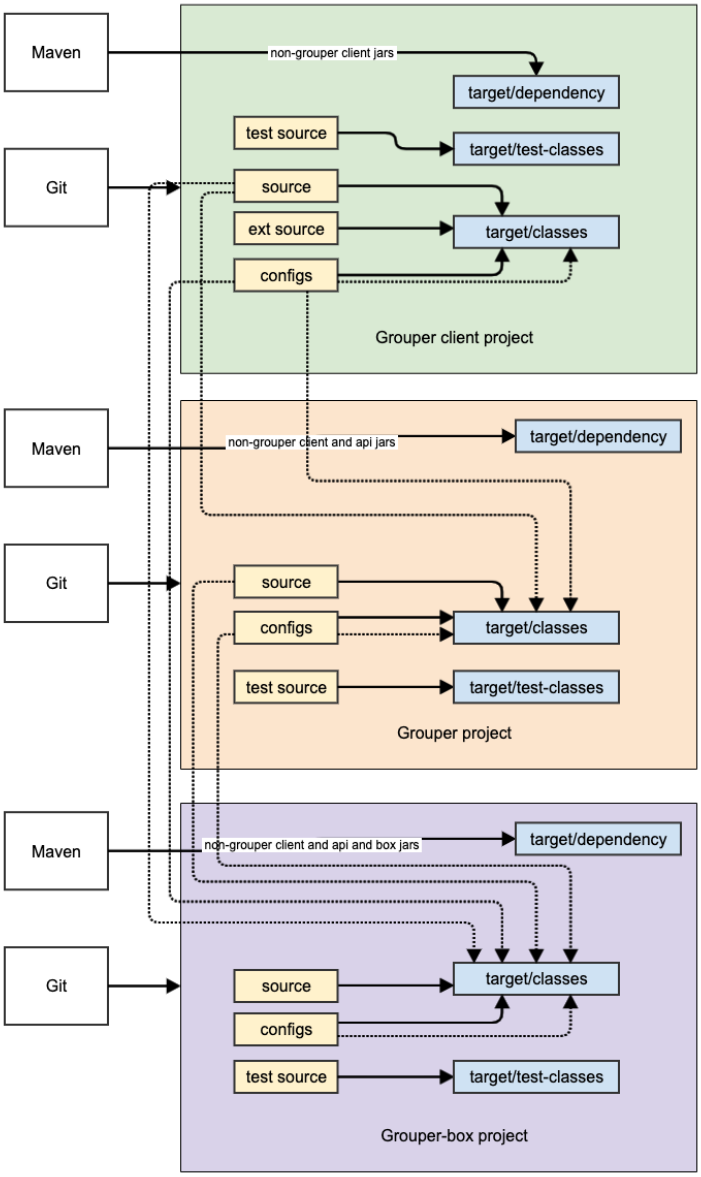
Grouper API dev env diagram

The API needs the client source and configs as linked source folders (e.g. for eclipse). This makes sure the current version of the configs and classes are in use when running tests or programs or compiling API source.



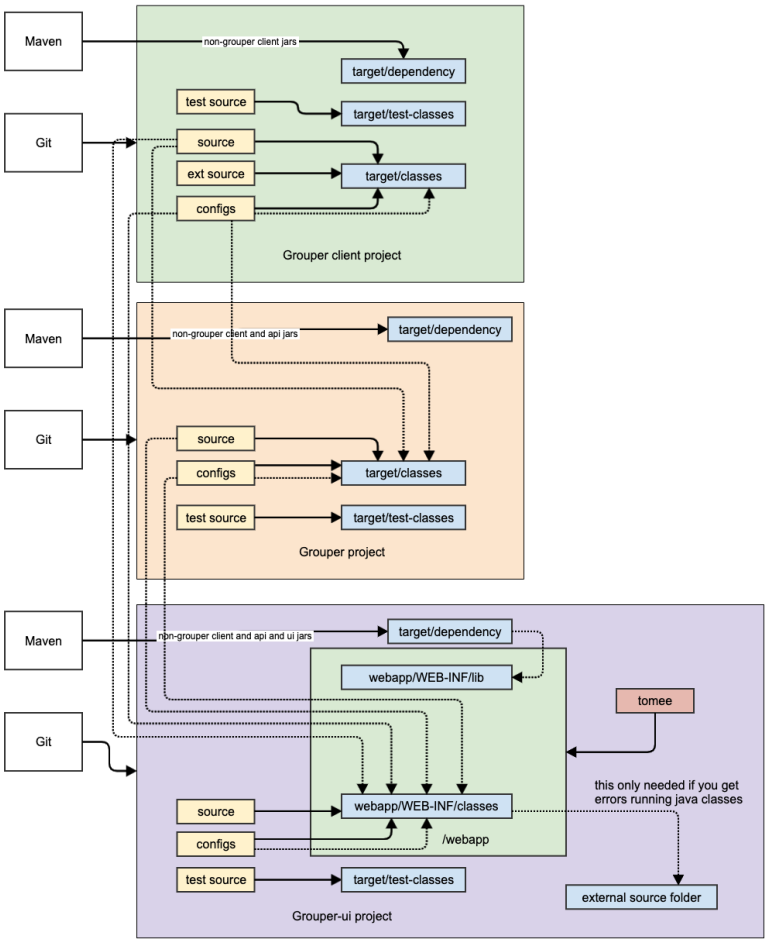
Non web-app dev env diagram

This an example of a non-web plugin for Grouper. Pretty much all modules for Grouper look like this (except web modules like UI/WS/scim). It generally depends on the API (and transitively client).



Webapp dev env diagram

This is an example of the UI dev env. Note other webapps (WS/scim) will look similar. There is no build script to run Tomee against the webapp, it will just work. When you edit Java classes you might need to restart Tomee (due to hotswap)



The example commands and screenshots are from Windows or MacOS and Eclipse, and may vary slightly for different environments. However, the overall process should be similar on any modern operating system and development tool chain. Developers can use whatever tools that let them work most efficiently.

Prerequisites

Git for source code version control

1. Install Git
    a. (Mac) Command line installs
        i. $ brew install git
        ii. https://github.com/fabriziocucci/git-bash-for-mac
        iii. Or Install from package https://git-scm.com/downloads
    b. Eclipse IDE plugin
        i. https://www.eclipse.org/egit/
    c. Github Desktop is also handy

Java - Grouper runs on Java

1. Install OpenJDK 8 exact version (not above or below). Grouper runs on Java.
    a. https://aws.amazon.com/corretto/
2. Note: do not use Java language features above Java 8 for most of Grouper - grouper, grouper-ui, grouper-ws, etc. The grouperClient code must be compliant with Java 6.

Apache Tomee - Grouper runs in Tomee

1. Download and unpack Tomee webprofile 7.0.7. Note: Grouper runs in tomcat 8.5, which is what tomee 7.07 has
    a. https://www.apache.org/dyn/closer.cgi/tomee/tomee-7.0.7/apache-tomee-7.0.7-webprofile.tar.gz

Database

- One option: Docker Desktop - to run our development database

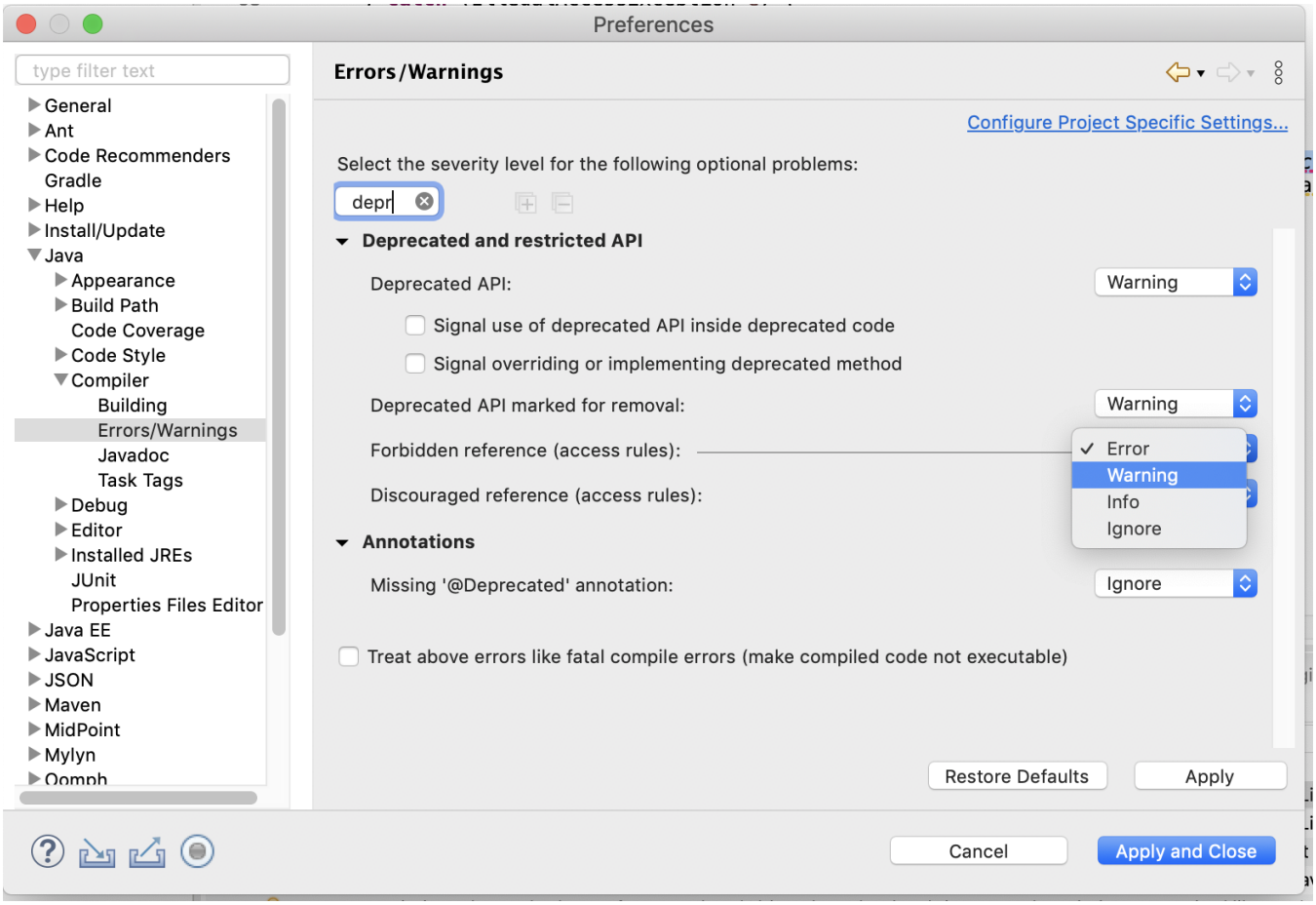
1. Install Docker Desktop. We'll use this to run our development database.
    a. https://www.docker.com/products/docker-desktop

- or You could just use hsql
- or Install mysql or postgres or use external database (external will be slow)

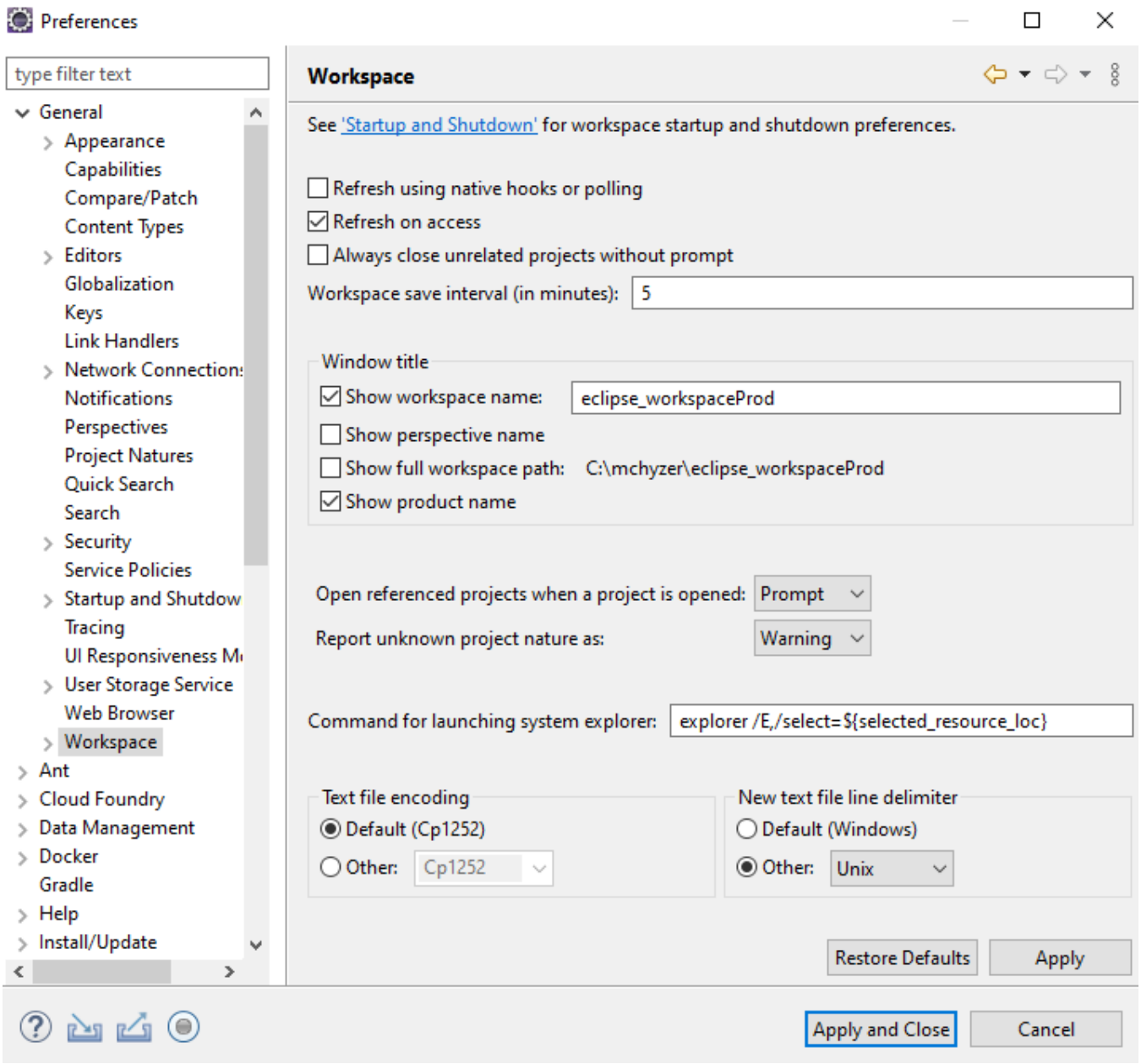
Eclipse - Grouper development happens in Eclipse (or your favorite IDE)

1. Install Eclipse IDE for Enterprise Java Developers or similar IDE
    a. https://www.eclipse.org/downloads/packages/
2. Make sure you have the latest eclipse (2020) or upgrade your current eclipse
3. Make sure the eclipse ini has at least 3 gig memory

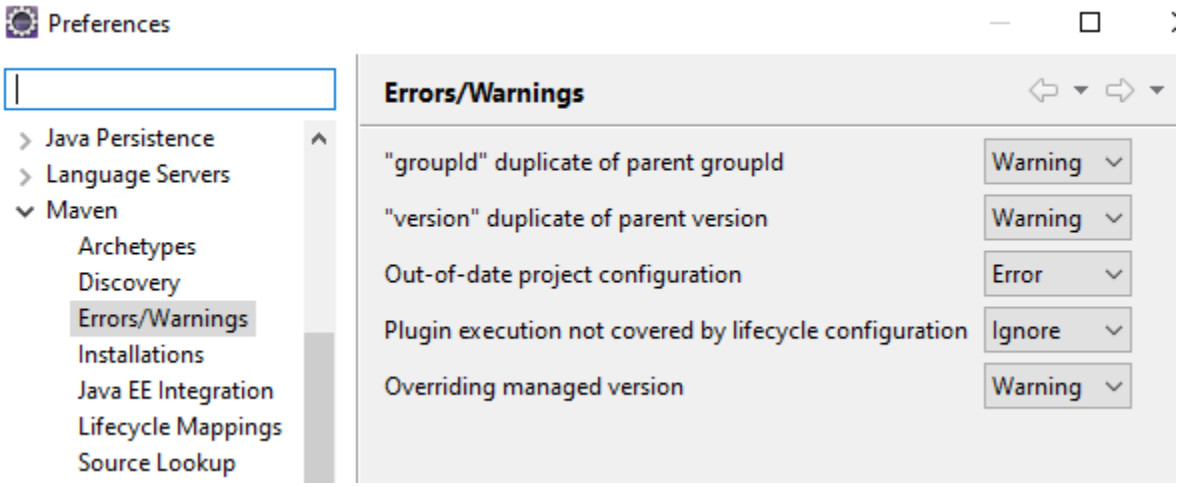
If you get errors on the client about deprecated libraries, you might need to adjust your compiler errors/warnings



Line endings should be unix


Preferences
type filter text
General
Appearance
Capabilities
Compare/Patch
Content Types
Editors
Globalization
Keys
Link Handlers
Network Connection:
Notifications
Perspectives
Project Natures
Quick Search
Search
Security
Service Policies
Startup and Shutdow
Tracing
UI Responsiveness M
User Storage Service
Web Browser
Workspace
Ant
Cloud Foundry
Data Management
Docker
Gradle
Help
Install/Update
Workspace
See 'Startup and Shutdown' for workspace startup and shutdown preferences.
Refresh using native hooks or polling
Refresh on access
Always close unrelated projects without prompt
Workspace save interval (in minutes): 5
Window title
Show workspace name: eclipse_workspaceProd
Show perspective name
Show full workspace path: C:\mchyzer\eclipse_workspaceProd
Show product name
Open referenced projects when a project is opened: Prompt
Report unknown project nature as: Warning
Command for launching system explorer: explorer /E,/select=${selected_resource_loc}
Text file encoding
Default (Cp1252)
Other: Cp1252
New text file line delimiter
Default (Windows)
Other: Unix
Restore Defaults
Apply
Apply and Close
Cancel

If you get errors on maven lifecycle set this:



Clone the Grouper Source Repository

The Grouper source code repository is managed in GitHub at https://github.com/Internet2/grouper.

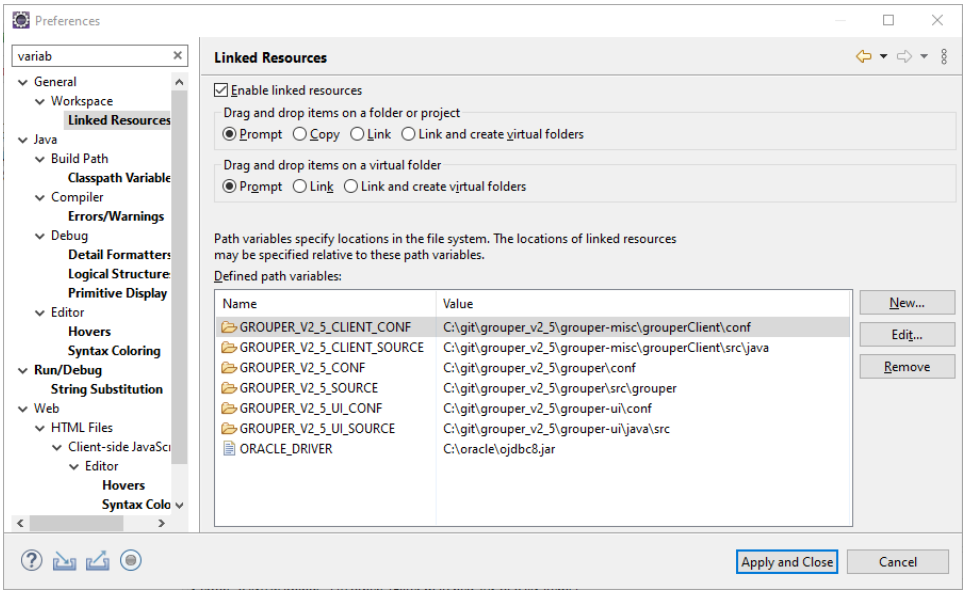
1. D:\mchyzer\git>git clone https://github.com/Internet2/grouper.git

Note: The build strategy changed for the 2.5 release. Switching between 2.4 and 2.5 branches in the same directory and workspace when developing is not recommended. Instead, keep 2.4 and 2.5 work in separate local directories each with their git repository and their own eclipse workspace. For 2.4 development see How to Setup a Grouper Development Environment for Grouper 2.4.

Import Projects into Eclipse

Start a new eclipse workspace and import grouper modules as individual projects. Project will import as Maven projects and automatically download the required Maven dependencies. The example commands below assume the git repository was cloned to the local directory 'D:\mchyzer\git\grouper_v2_5'.

1. Make a variable for the root of the git repo called GIT_ROOT: e.g. for D:\mchyzer\git\grouper, and other variables



2. Import grouper-parent
    a. Eclipse -> File -> Import Maven Existing Maven Project
    b. Import source: ${GIT_ROOT}/grouper-parent
3. That should automatically import all grouper projects

Maven clean on grouper-parent

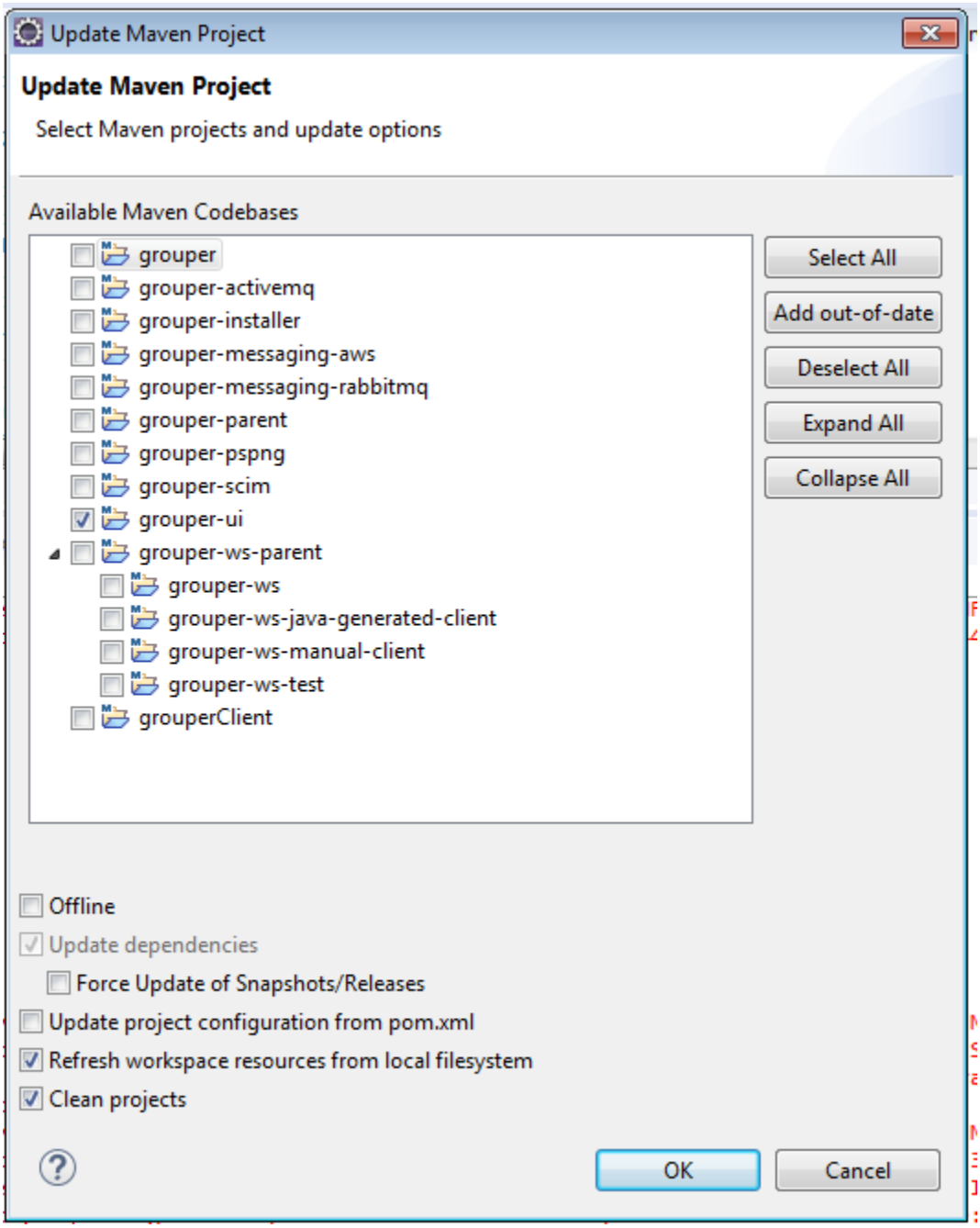
Run the maven grouper-parent clean and install (you can right click in eclipse on the pom and run as: maven clean, then install). you might need to delete .m2/repository/* if it is corrupt.

You might need to bump up memory to 512MB to get maven to build

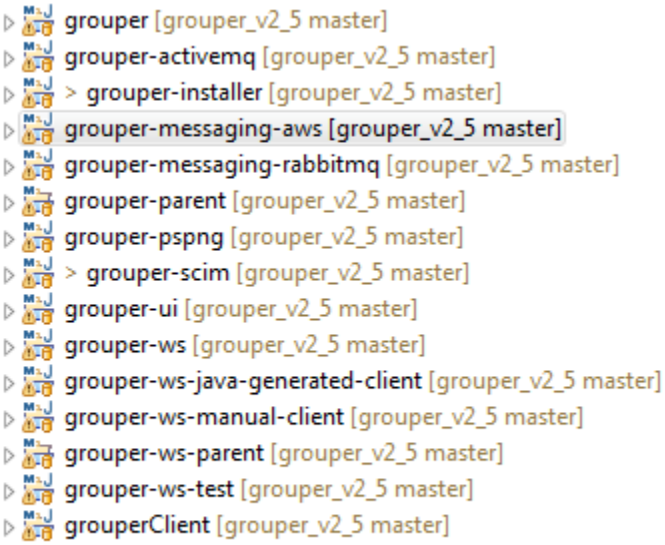
```
[INFO] Reactor Summary for Grouper 2.5.0-SNAPSHOT:
[INFO]
[INFO] Grouper ........................................... SUCCESS [  1.632 s]
[INFO] Grouper Client .................................... SUCCESS [  3.659 s]
[INFO] Grouper API ....................................... SUCCESS [  9.454 s]
[INFO] Grouper SCIM ...................................... SUCCESS [  0.256 s]
[INFO] Grouper UI ........................................ SUCCESS [  2.295 s]
[INFO] Grouper WS Parent ................................. SUCCESS [  0.063 s]
[INFO] Grouper WS ........................................ SUCCESS [  3.129 s]
[INFO] Grouper WS Generated Client ....................... SUCCESS [  5.320 s]
[INFO] Grouper WS Manual Client .......................... SUCCESS [  0.825 s]
[INFO] Grouper WS Test ................................... SUCCESS [  0.098 s]
[INFO] Grouper Installer ................................. SUCCESS [  8.198 s]
[INFO] Grouper AMQ ....................................... SUCCESS [  0.853 s]
[INFO] Grouper Rabbitmq .................................. SUCCESS [  6.950 s]
[INFO] Grouper AWS Messaging ............................. SUCCESS [  9.099 s]
[INFO] Grouper PSP-NG .................................... SUCCESS [  0.389 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

If there are problems in a project, you might need to right click, and do Maven Update project

When you do that, the **first time, check the box to update project configuration**. In **all subsequent times, do NOT have the box checked** to update project configuration from pom, or your settings will get undone

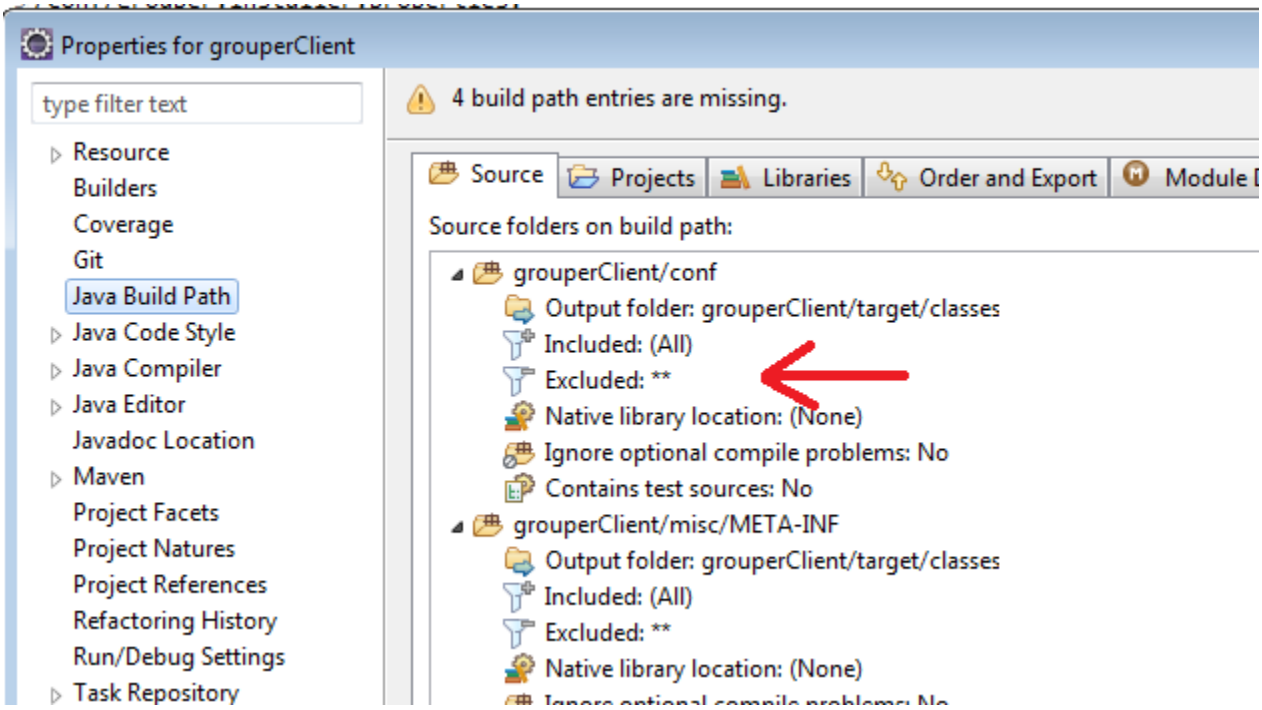


All the projects should now be open and compiled.



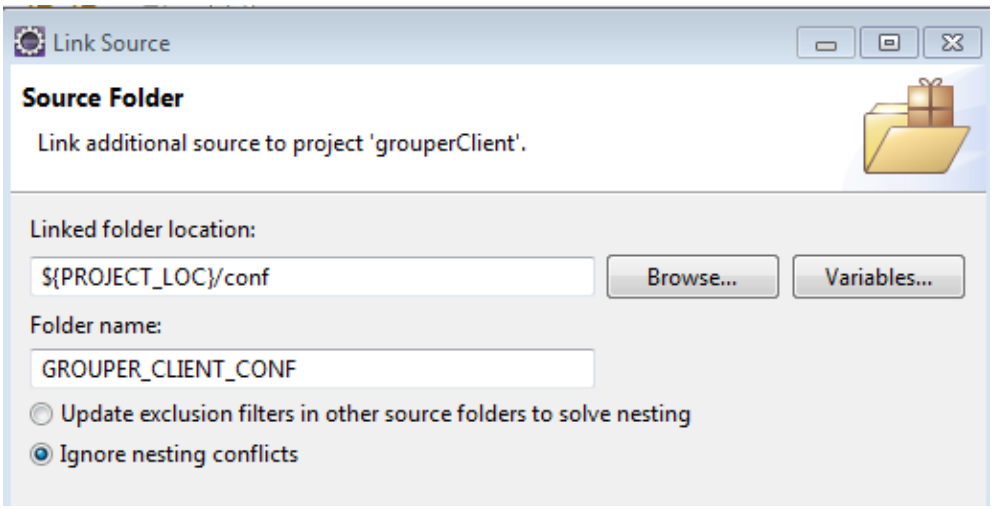
Link conf to grouper client project

This is one of the main tricks. In my eclipse, the "conf" dir is excluded due to the pom.xml in maven



Link the conf dir (even though its already a source folder) in java build path

- ${PROJECT_LOC}/conf GROUPER_CLIENT_CONF



Link source and conf to grouper project

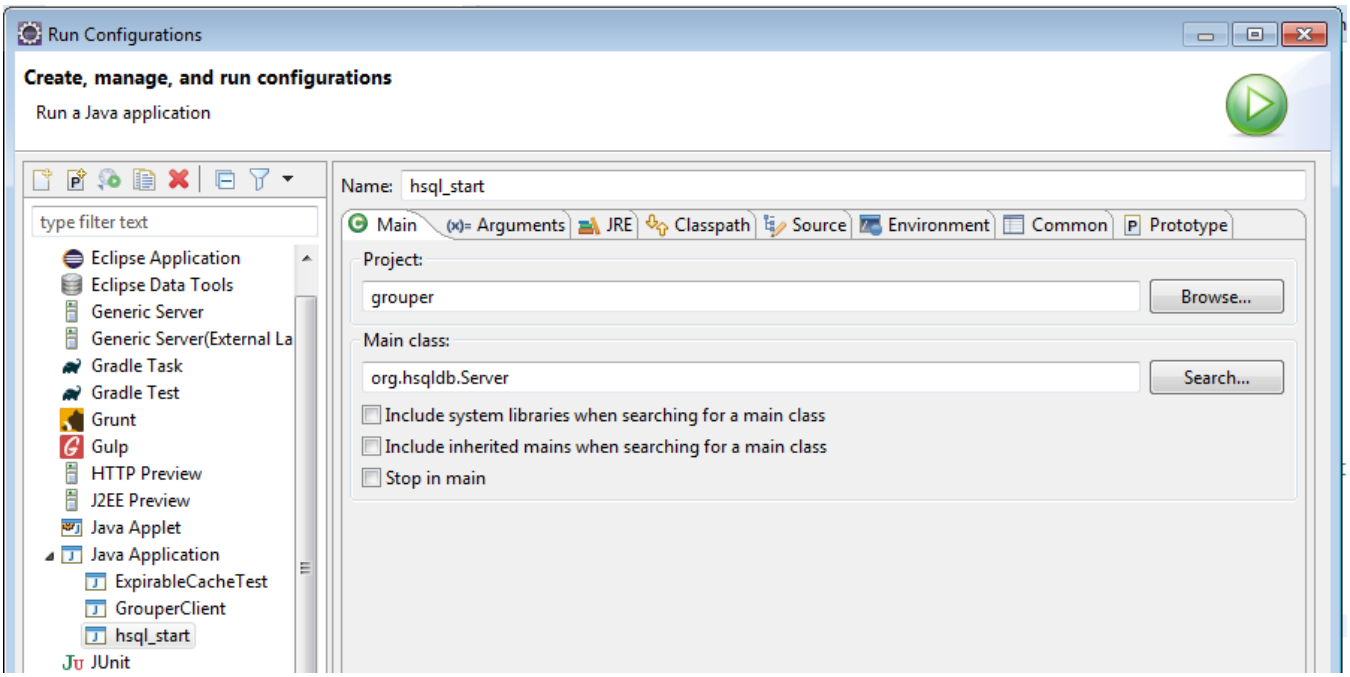
We want to be able to run and debug the Grouper so that it picks up client source changes as you develop. Also needs the grouper conf per above

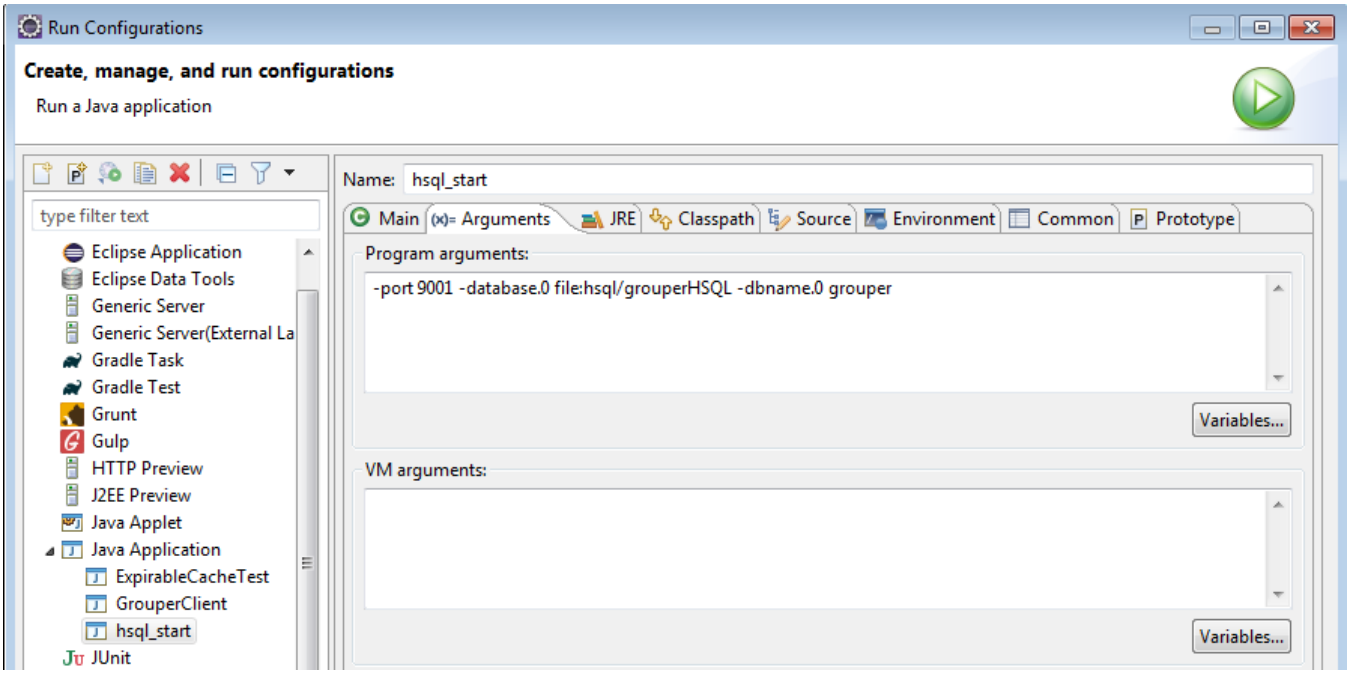
1. Link source: ${GIT_ROOT}/grouper-misc/grouperClient/src/java GROUPER_CLIENT_SOURCE
2. Link source: ${GIT_ROOT}/grouper-misc/grouperClient/conf GROUPER_CLIENT_CONF
3. Link source: ${PROJECT_LOC}/conf GROUPER_CONF
4. Look at src/test and make sure its not filtering only *.java. All files need to be on classpath for tests to work

Start HSQL (or whatever database)

Make a java application:

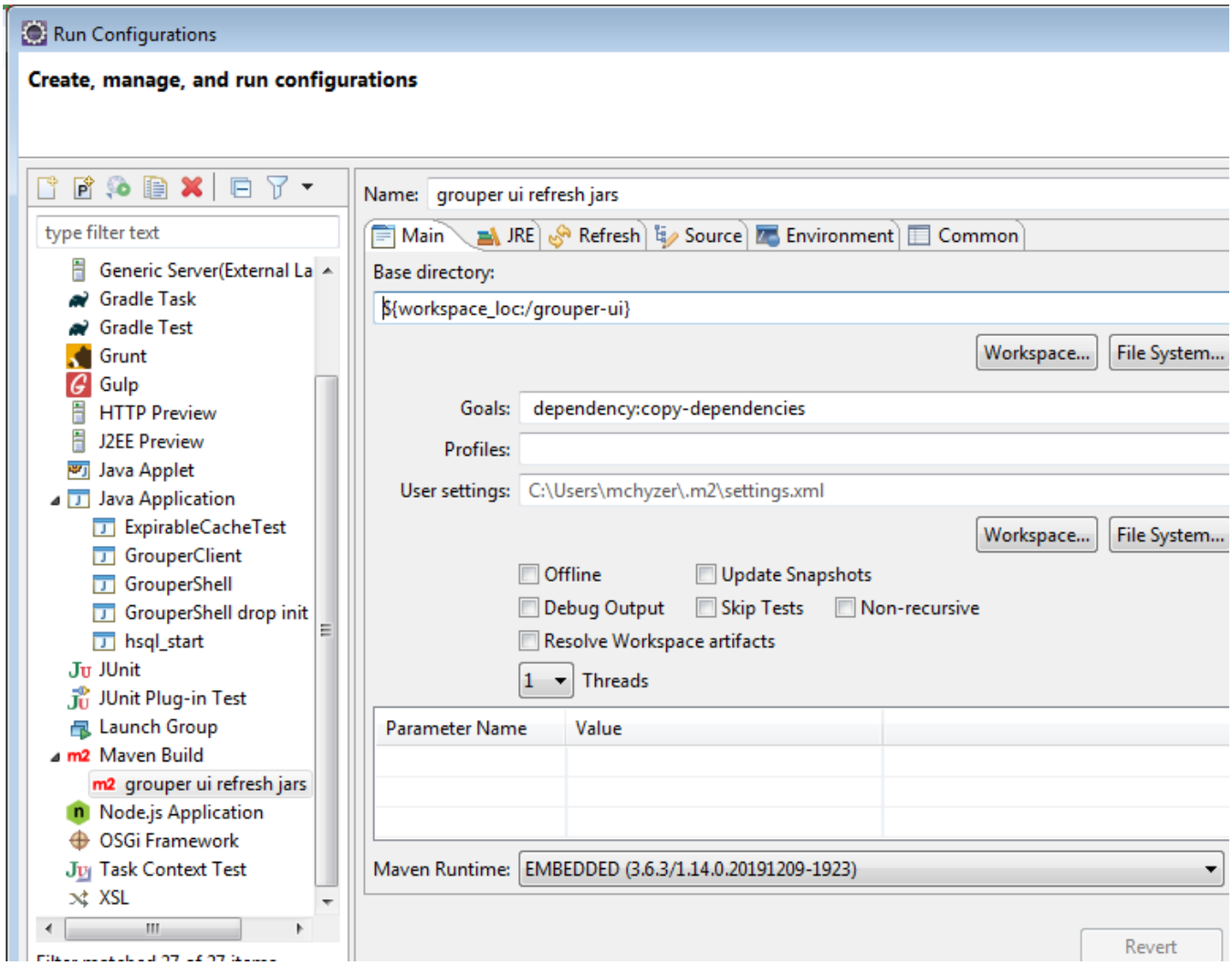
- Project: grouper
- Class:
- Arguments: -port 9001 -database.0 file:hsql/grouperHSQL -dbname.0 grouper



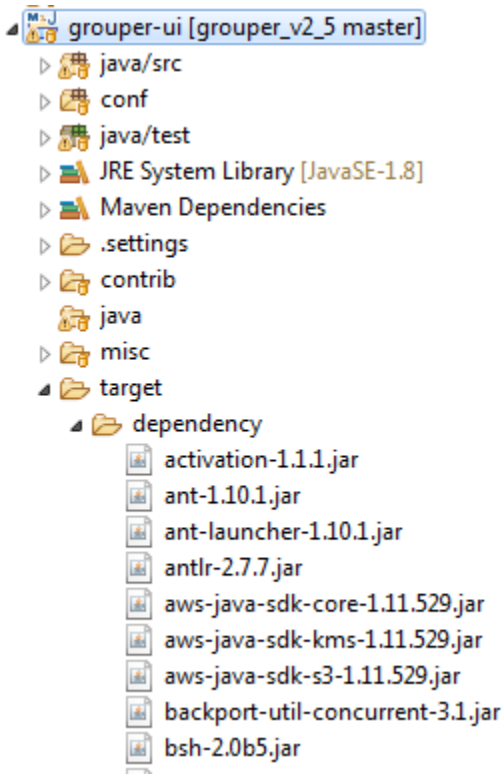


Get grouper UI Jars

First off, we need all the jars. You need to do this in the future if you have any class not found exceptions or weird errors, goals: dependency:copy-dependencies



All jars are not in target/dependency

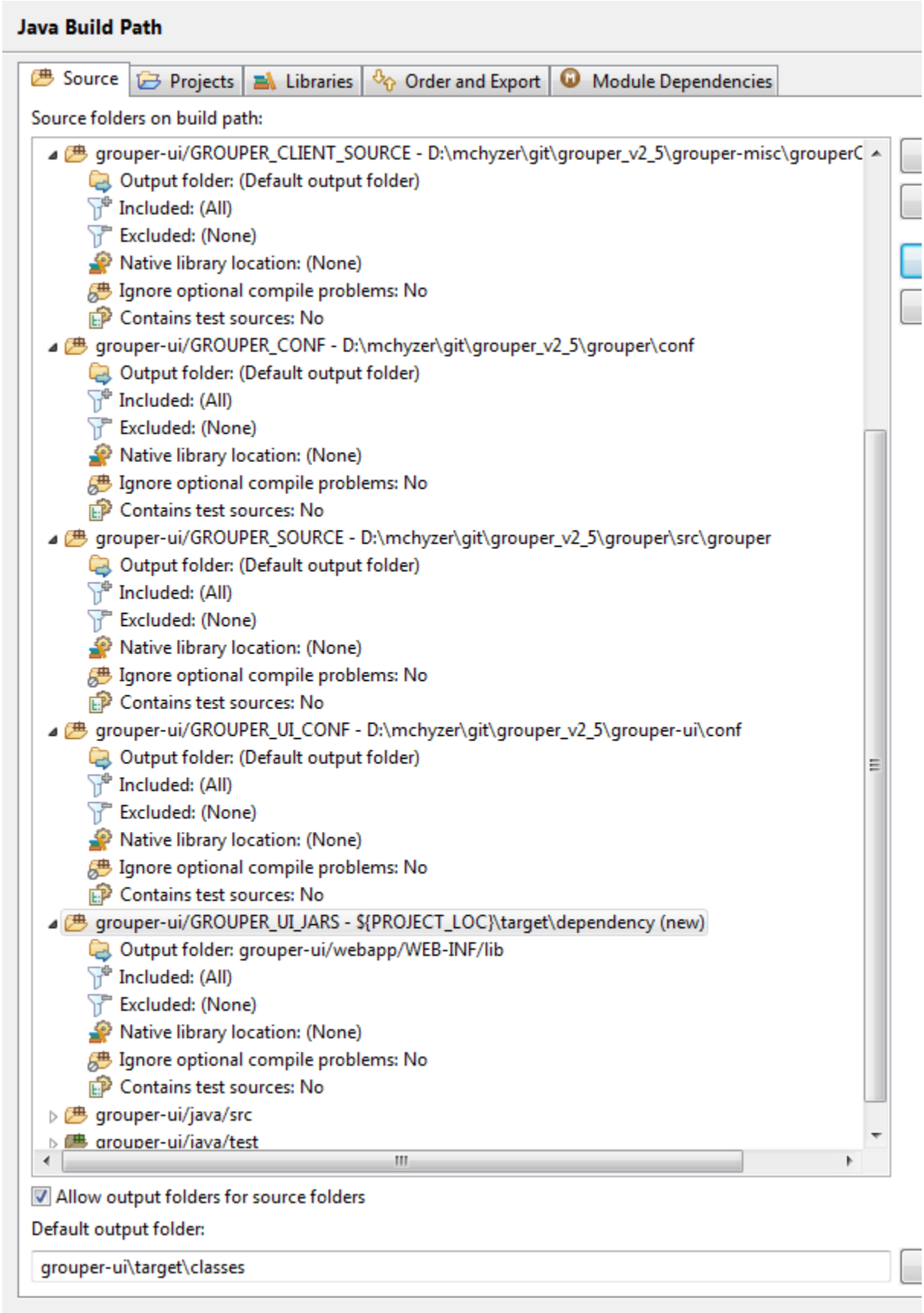


Link source and conf to grouper-ui project

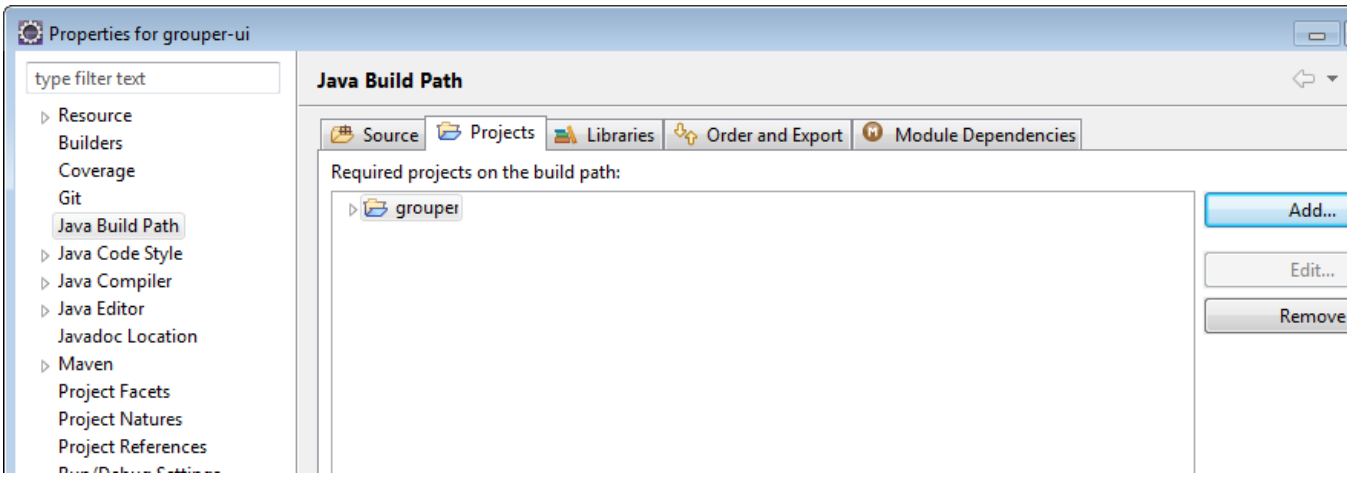
We want to be able to run and debug the Grouper UI from the grouper-ui/webapp folder, so that we can work on webapp artifacts (JSPs, etc), and at the same time update Java code in the grouper project and other code locations. To do this we will update the Java Build Path output folder so that compiled classes and other artifacts go to the right directories under grouper-ui/webapp. We will also add some dependent source and library folders to the grouper-ui Java Build Path.

1. grouper-ui -> File -> Properties -> Java Build Path -> Source tab
    a. Set 'Default output folder:' to: grouper-ui/webapp/WEB-INF/classes (you may have to create this directory)
2. Add dependent source and configuration folders to the grouper-ui Java Build Path
    a. Link source: ${GIT_ROOT}/grouper-misc/grouperClient/src/java GROUPER_CLIENT_SOURCE
    b. Link source: ${GIT_ROOT}/grouper-misc/grouperClient/conf GROUPER_CLIENT_CONF
    c. Link source: ${PROJECT_LOC}/conf GROUPER_UI_CONF
    d. Link source: ${GIT_ROOT}/grouper/conf GROUPER_CONF
    e. Link source: ${GIT_ROOT}/grouper/src/grouper GROUPER_SOURCE
    f. Link source: ${PROJECT_LOC}/target/dependency GROUPER_UI_JARS
        i. Output folder for jars is: webapp/WEB-INF/lib (create the folder)
3. Make sure all folders except 'grouper-ui/java/test' are using the default output folder
4. ~~Make sure grouper-ui/conf has 'Excluded:' set to (None)~~ (Note, this gets changed back for Chris, so ignore it)
5. Remove srcPoc and misc source folders (these are not needed)

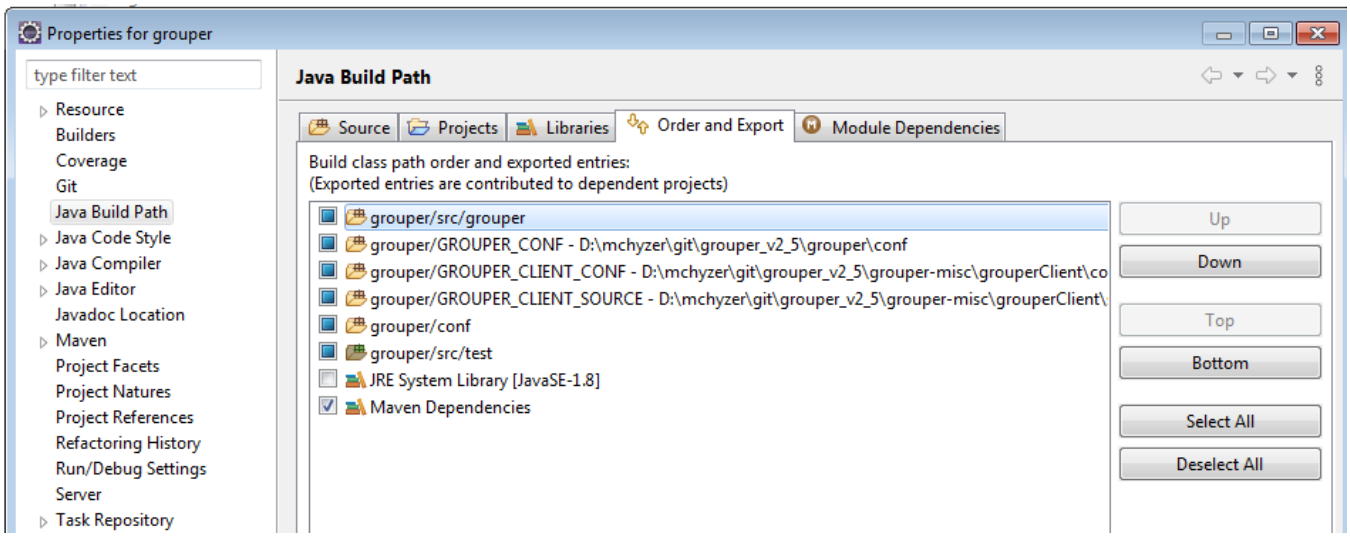
The grouper-ui Java Build Path should now look something like this:



Add Grouper project as build path project

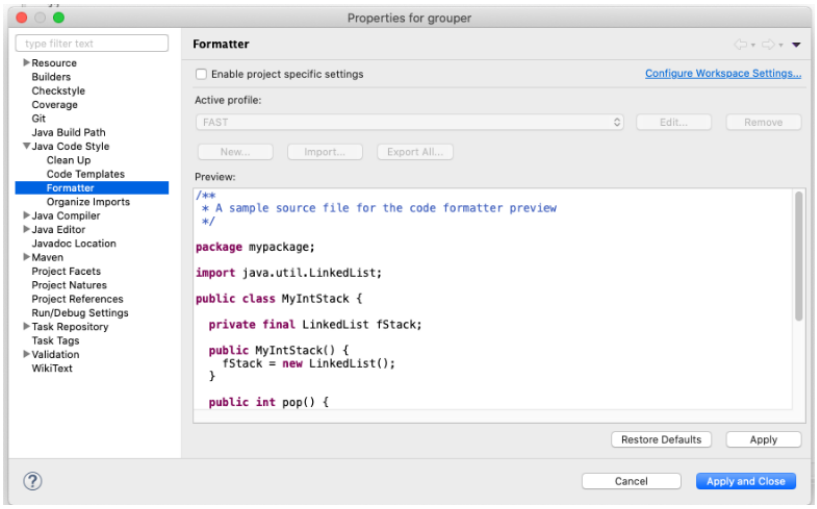


In Grouper project, export all the maven dependencies in build path



Configure Eclipse Code Formatter

1. Eclipse -> Preferences -> Java Code Style -> Formatter
2. Import… (navigate and select grouper/misc/eclipse/fastFormat.xml)
3. Apply and close



4. Look in Eclipse config and change all tabs to 2 spaces for indenting (search for "tab")

5. Disable folding

6. Disable spell check

7. Look in eclipse config and ignore whitespace changes

8. Settings General Workspace New text file line endings Unix

Development Database

Multiple databases are supported including Oracle, mySQL, and PostgreSQL. We'll use PostgreSQL for this how-to. The steps for other databases would be similar.

Start the development database

> **Running Postgres in Docker on a Windows Host**
>
> Postgres' docker image runs the database as an unprivileged user *postgres.* The container's startup script attempts to chown /var/lib/postgresql /data folder to this user. In Docker Desktop for Windows, this causes the script to throw an error and exit. Bind volumes cannot easily have their ownership changed from within the container running on a Windows host. The workaround for this is creating a named volume instead.

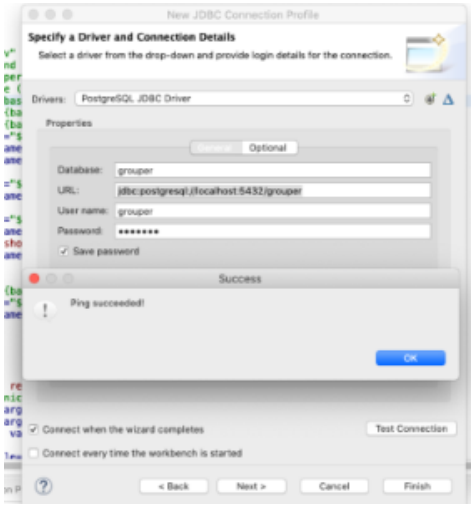
We will run postgres with a mounted external volume to preserve data between docker container restarts.

1. Create a named volume 'docker create volume grouper-postgres'
2. Run 'docker run --name grouperdb -e POSTGRES_PASSWORD=grouper -e POSTGRES_USER=grouper -d -p 5432:5432 -v grouper-postgres: /var/lib/postgresql/data postgres'

Connect to development database in Eclipse

Note, this is one way to do it. Using any other database UI browser works too. e.g. Squirrel SQL client works with all DBs

1. Window -> Perspective -> Open Perspective -> Other -> Database Development -> Open
2. Right click 'Database Connections' and create PostgreSQL connection profile
3. Add the postgess JDBC driver found in under ../grouper/grouper-ui/target/dependency
4. Enter the connection details and then click Test Connection



Configure minimum properties files for development

Create and configure the grouper-hibernate.properties for postgres (for example)

1. cd grouper/conf
2. cp .../misc/grouper.hibernate.example.properties grouper.hibernate.properties
3. edit grouper.hibernate.properties to look like the following:

```
28 ########################################
29 ## DB settings
30 ########################################
31
32 # e.g. mysql:           jdbc:mysql://localhost:3306/grouper
33 # e.g. p6spy (log sql): [use the URL that your DB requires]
34 # e.g. oracle:          jdbc:oracle:thin:@server.school.edu:1521:sid
35 # e.g. hsqldb (a):      jdbc:hsqldb:dist/run/grouper;create=true
36 # e.g. hsqldb (b):      jdbc:hsqldb:hsql://localhost:9001/grouper
37 # e.g. postgres:        jdbc:postgresql://localhost:5432/database
38 # e.g. mssql:           jdbc:sqlserver://localhost:3280;databaseName=grouper
39 hibernate.connection.url = jdbc:postgresql://localhost:5432/grouper
40
41 hibernate.connection.username         = grouper
42 # If you are using an empty password, depending upon your version of
43 # Java and Ant you may need to specify a password of "".
44 # Note: you can keep passwords external and encrypted: https://bugs.internet2.edu/jira/browse/GRP-122
45 hibernate.connection.password         = grouper
46
```

Example for HSQL database: make a file grouper/conf/grouper.hibernate.properties:

```
########################################
## DB settings
########################################

# e.g. mysql:           jdbc:mysql://localhost:3306/grouper?useSSL=false
# e.g. p6spy (log sql): [use the URL that your DB requires]
# e.g. oracle:          jdbc:oracle:thin:@server.school.edu:1521:sid
# e.g. hsqldb (a):      jdbc:hsqldb:dist/run/grouper;create=true
# e.g. hsqldb (b):      jdbc:hsqldb:hsql://localhost:9001/grouper
# e.g. postgres:        jdbc:postgresql://localhost:5432/database
# e.g. mssql:           jdbc:sqlserver://localhost:3280;databaseName=grouper
hibernate.connection.url = jdbc:hsqldb:hsql://localhost:9001/grouper

hibernate.connection.username         = sa
# If you are using an empty password, depending upon your version of
# Java and Ant you may need to specify a password of "".
# Note: you can keep passwords external and encrypted: https://bugs.internet2.edu/jira/browse/GRP-122
hibernate.connection.password         =
```

Create and configure morphString.properties

1. cd grouper/conf
2. cp .../misc/morphString.example.properties morphString.properties
3. Edit .../grouper/grouper/conf/morphString.properties
4. encrypt.key = not_a_random_key

- Make a file grouper/conf/morphString.properties (make up an encrypt.key or generate alphanumeric from password generator)

```
########################################
## Encryption configuration
########################################

# Put a random alphanumeric string (Case sensitive) for the password encryption.  e.g. fh43IRJ4Nf5jn4Qp9k2
# or put a filename where the random alphanumeric string is.  e.g. c:/whatever/key.txt
# use encrypt.key.elConfig instead if the config has an expression language scriptlet
encrypt.key = abcndme45jg32fj32JNdQ23
```

- Copy grouper/conf/log4j.example.properties to log4j.properties
- Make a grouper/conf/subject.properties

```
########################################

# enter the location of the sources.xml.  Must start with classpath: or file:
# blank means dont use sources.xml, use subject.properties
# default is: classpath:sources.xml
# e.g. file:/dir1/dir2/sources.xml
subject.sources.xml.location =



########################################
## Configuration for source id: jdbc
## Source configName: jdbc
########################################
subjectApi.source.jdbc.id = jdbc

# this is a friendly name for the source
subjectApi.source.jdbc.name = Example JDBC Source Adapter

# type is not used all that much.  Can have multiple types, comma separate.  Can be person, group,
application
subjectApi.source.jdbc.types = person

# the adapter class implements the interface: edu.internet2.middleware.subject.Source
# adapter class must extend: edu.internet2.middleware.subject.provider.BaseSourceAdapter
```

```
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter2  :  if doing JDBC this should be used
if possible.  All subject data in one table/view.
# edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter   :  oldest JDBC source.  Put freeform
queries in here
# edu.internet2.middleware.grouper.subj.GrouperJndiSourceAdapter   :  used for LDAP
subjectApi.source.jdbc.adapterClass = edu.internet2.middleware.grouper.subj.GrouperJdbcSourceAdapter

subjectApi.source.jdbc.param.jdbcConnectionProvider.value = edu.internet2.middleware.grouper.subj.
GrouperJdbcConnectionProvider

subjectApi.source.jdbc.param.emailAttributeName.value = email

# maximum number of results from a search, generally no need to get more than 1000
subjectApi.source.jdbc.param.maxResults.value = 1000

subjectApi.source.jdbc.param.maxPageSize.value = 100

# ldap attribute which is the subject id.  e.g. exampleEduRegID   Each subject has one and only one
subject id.  Generally it is opaque and permanent.
subjectApi.source.jdbc.param.SubjectID_AttributeType.value = id

# attribute which is the subject name
subjectApi.source.jdbc.param.Name_AttributeType.value = name

# attribute which is the subject description
subjectApi.source.jdbc.param.Description_AttributeType.value = description

# This virtual attribute index 0 is accessible via: subject.getAttributeValue("searchAttribute0");
subjectApi.source.jdbc.param.subjectVirtualAttribute_0_searchAttribute0.value = ${subject.
name},${subjectUtils.defaultIfBlank(subject.getAttributeValue('LFNAME'), "")},${subjectUtils.
defaultIfBlank(subject.getAttributeValue('LOGINID'), "")},${subjectUtils.defaultIfBlank(subject.
description, "")},${subjectUtils.defaultIfBlank(subject.getAttributeValue('EMAIL'), "")}

# the 1st sort attribute for lists on screen that are derived from member table (e.g. search for member
in group)
# you can have up to 5 sort attributes
subjectApi.source.jdbc.param.sortAttribute0.value = LFNAME

# the 2nd sort attribute for lists on screen that are derived from member table (e.g. search for member
in group)
# you can have up to 5 sort attributes
subjectApi.source.jdbc.param.sortAttribute1.value = LOGINID

# the 1st search attribute for lists on screen that are derived from member table (e.g. search for
member in group)
# you can have up to 5 search attributes
subjectApi.source.jdbc.param.searchAttribute0.value = searchAttribute0

subjectApi.source.jdbc.param.useInClauseForIdAndIdentifier.value = true

subjectApi.source.jdbc.param.identifierAttributes.value = LOGINID

# subject identifier to store in grouper's member table.  this is used to increase speed of loader and
perhaps for provisioning
# you can have up to max 1 subject identifier
subjectApi.source.jdbc.param.subjectIdentifierAttribute0.value = LOGINID

#searchSubject: find a subject by ID.  ID is generally an opaque and permanent identifier, e.g. 12345678.
#  Each subject has one and only on ID.  Returns one result when searching for one ID.

# sql is the sql to search for the subject by id should use an {inclause}
subjectApi.source.jdbc.search.searchSubject.param.sql.value = select    s.subjectid as id, s.name as
name,    (select sa2.value from subjectattribute sa2 where name='name' and sa2.SUBJECTID = s.subjectid)
as lfname,    (select sa3.value from subjectattribute sa3 where name='loginid' and sa3.SUBJECTID = s.
subjectid) as loginid,    (select sa4.value from subjectattribute sa4 where name='description' and sa4.
SUBJECTID = s.subjectid) as description,    (select sa5.value from subjectattribute sa5 where
name='email' and sa5.SUBJECTID = s.subjectid) as email from    subject s where    {inclause}

# inclause allows searching by subject for multiple ids or identifiers in one query, must have
{inclause} in the sql query,
#    this will be subsituted to in clause with the following.  Should use a question mark ? for bind
```

```
variable
subjectApi.source.jdbc.search.searchSubject.param.inclause.value = s.subjectid = ?

#searchSubjectByIdentifier: find a subject by identifier.  Identifier is anything that uniquely
#  identifies the user, e.g. jsmith or jsmith@institution.edu.
#  Subjects can have multiple identifiers.  Note: it is nice to have if identifiers are unique
#  even across sources.  Returns one result when searching for one identifier.

# sql is the sql to search for the subject by identifier should use an {inclause}
subjectApi.source.jdbc.search.searchSubjectByIdentifier.param.sql.value = select    s.subjectid as id, s.
name as name,    (select sa2.value from subjectattribute sa2 where name='name' and sa2.SUBJECTID = s.
subjectid) as lfname,    (select sa3.value from subjectattribute sa3 where name='loginid' and sa3.
SUBJECTID = s.subjectid) as loginid,    (select sa4.value from subjectattribute sa4 where
name='description' and sa4.SUBJECTID = s.subjectid) as description,    (select sa5.value from
subjectattribute sa5 where name='email' and sa5.SUBJECTID = s.subjectid) as email from    subject s,
subjectattribute a where    a.name='loginid' and s.subjectid = a.subjectid and {inclause}

# inclause allows searching by subject for multiple ids or identifiers in one query, must have
{inclause} in the sql query,
#    this will be subsituted to in clause with the following.  Should use a question mark ? for bind
variable
subjectApi.source.jdbc.search.searchSubjectByIdentifier.param.inclause.value = a.value = ?

#   search: find subjects by free form search.  Returns multiple results.

# sql is the sql to search for the subject free-form search.  user question marks for bind variables
subjectApi.source.jdbc.search.search.param.sql.value = select    s.subjectid as id, s.name as name,
(select sa2.value from subjectattribute sa2 where name='name' and sa2.SUBJECTID = s.subjectid) as
lfname,    (select sa3.value from subjectattribute sa3 where name='loginid' and sa3.SUBJECTID = s.
subjectid) as loginid,    (select sa4.value from subjectattribute sa4 where name='description' and sa4.
SUBJECTID = s.subjectid) as description,    (select sa5.value from subjectattribute sa5 where
name='email' and sa5.SUBJECTID = s.subjectid) as email from    subject s where    s.subjectid in (
select subjectid from subject where lower(name) like concat('%',concat(?,'%')) union       select
subjectid from subjectattribute where searchvalue like concat('%',concat(?,'%'))    )

# internal attributes are used by grouper only not exposed to code that uses subjects.  comma separated
subjectApi.source.jdbc.internalAttributes = searchAttribute0
```

- Make a grouper/conf/grouper.properties and add

```
grouper.dev.env.allowMissingServlets = true
```

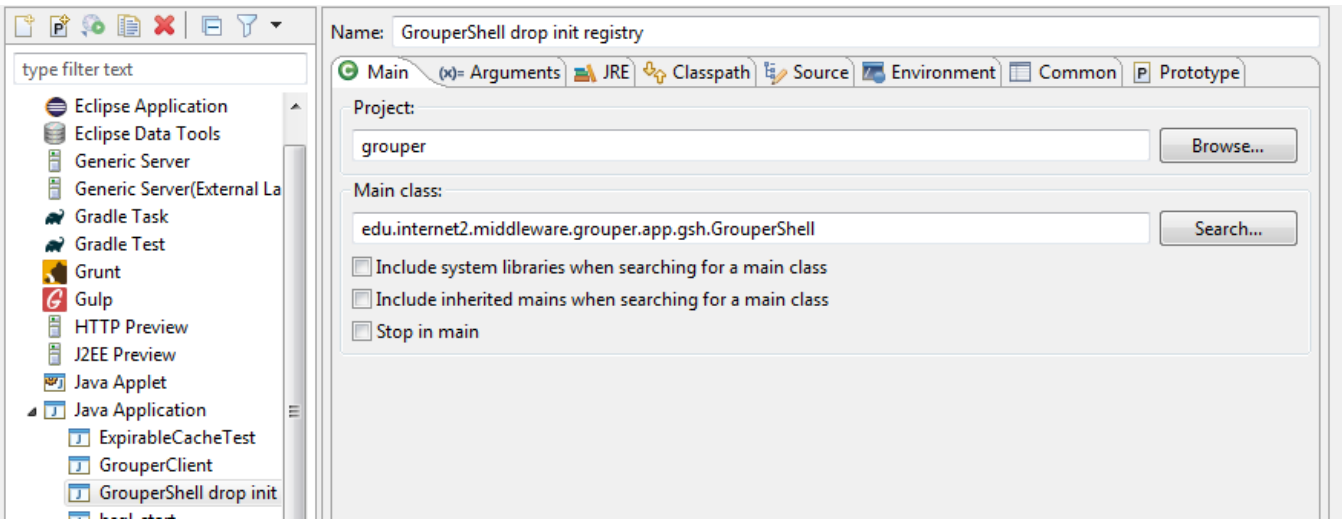
- if you are editing a lot of text, you can do this without waiting for cache to refresh

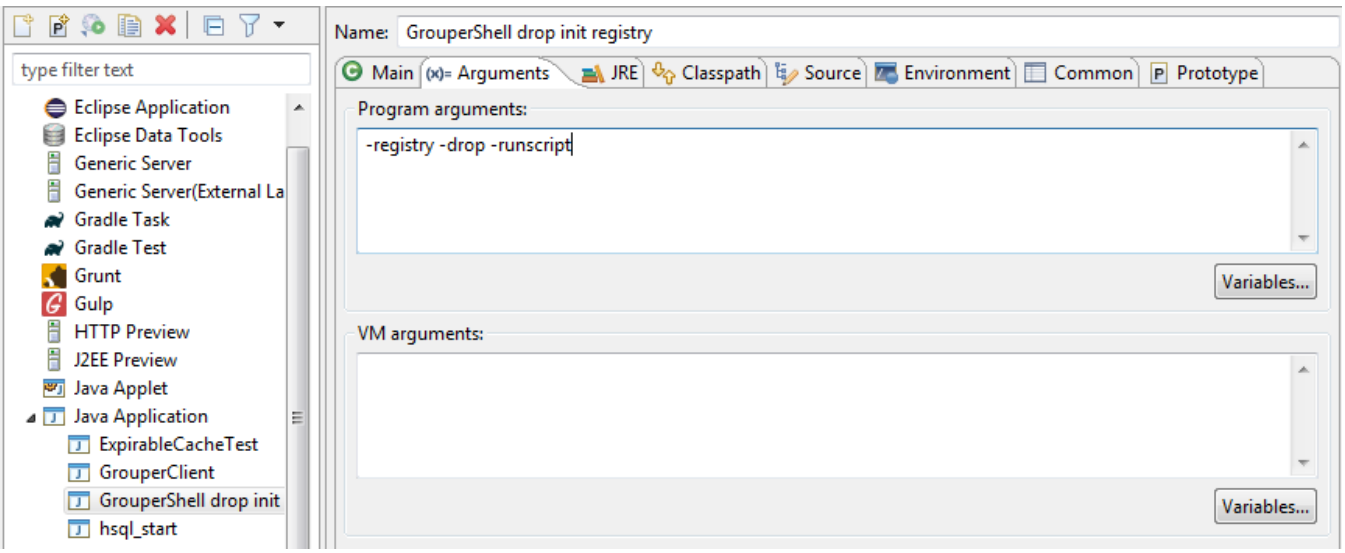
```
cd grouper/conf
Edit grouper-ui.properties (add if not there) (default is false, dont edit grouper-ui.base.properties)

##############################################
## Developer settings
##############################################

# if you're developing, it will refresh configs and text etc on every UI request (no need to compile,
build, restart, etc)
# note this affects caches if you are working on something that relies on caching
# you should only set this to true while working on externalized text etc, then set back to false for
final testing
# {valueType: "boolean", required: true}
grouperUi.refreshCaches.onEveryRequest = true
```

Run GrouperShell, and init db





FYI: how to run Java classes on the command line

1. Run maven goal (right click and type in goal like above with UI): dependency:copy-dependencies
2. Call class command line

```
grouper $ java -cp target/classes:target/dependency/* edu.internet2.middleware.grouper.app.gsh.
GrouperShell
```

Bootstrap the Grouper Database

For development purposes, we'll bootstrap the Grouper database, add sample subjects, and reset the database using a few Java classes.

Run GrouperShell from Eclipse to initialize the Grouper database:

1. Right click GrouperShell in the Grouper project explorer
2. Select 'Run as' and then 'Run Configurations…'
3. Name: GrouperShell -registry -runscript -noprompt
4. (x) = Arguments tab

    a. Program arguments: -registry -runscript -noprompt
5. Classpath tab
    a. Highlight User Entries and click ‘Advanced…’
    a. Add the ‘conf’ folder to the classpath for the properties files
6. Click ‘Apply’
7. Click ‘Run’

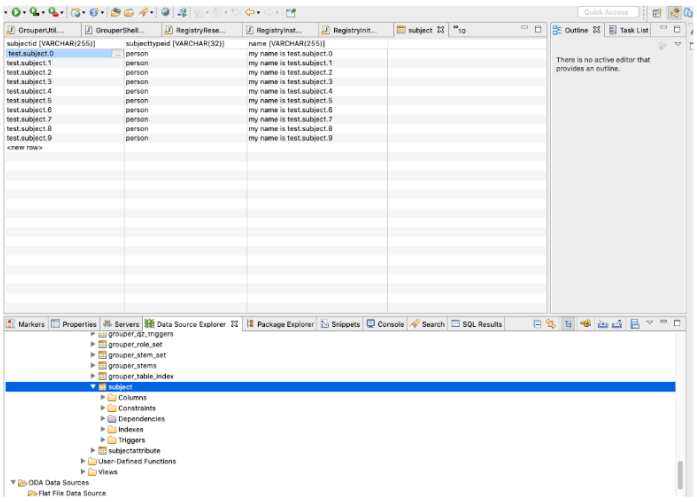
Run GrouperShell from Eclipse to check the Grouper database:

1. Right click GrouperShell in grouper project explorer
2. Select ‘Run as’ and then ‘Run Configurations…’
    a. Name: GrouperShell -registry -check -noprompt
3. (x) = Arguments tab
    a. -registry -check -noprompt
    a. Program arguments:
4. Classpath tab
    a. Highlight User Entries and click ‘Advanced…’
    b. Add the ‘conf’ folder to the classpath
5. Click ‘Apply’
6. Click ‘Run’

Run RegistryReset with ‘addSubjects’ as an argument to add sample subjects:

1. Right click RegistryReset in grouper project explorer
2. Select ‘Run as’ and then ‘Run Configurations…’
    a. Name: RegistryReset addSubjects
3. (x) = Arguments tab
    a. Program arguments:
        i. addSubjects
4. Classpath tab
    a. Highlight User Entries and click ‘Advanced…’
    b. Add the ‘conf’ folder to the classpath
    c. Click ‘Apply’
5. Click ‘Run’

Query the subjects table from the Eclipse Data Source Explorer to see the added subjects:



Add a GrouperSystem basic auth password

Preferred method in dev/test only, in a Grouper v2.5 build (maybe 2.5.25?) there is a config file override for UI/WS local basic password. in grouper-hibernate.properties add something like this

```
# override a grouper authn password here for testing and development
# grouperPasswordConfigOverride_<APP>_subjectId_pass = pass (hopefully encrypted but doesnt need to)
# e.g. grouperPasswordConfigOverride_UI_GrouperSystem_pass = abnf234
grouperPasswordConfigOverride_UI_GrouperSystem_pass = someGoodPass
grouperPasswordConfigOverride_WS_test.subject.0_pass = someGoodPass
```

Run this in GrouperShell (run again with no args) though this is not ideal in dev env since unit tests delete it, use the previous method

```
GrouperSession grouperSession = GrouperSession.startRootSession();
GrouperPasswordSave grouperPasswordSave = new GrouperPasswordSave();
grouperPasswordSave.assignUsername("GrouperSystem").assignPassword("password").assignEntityType("username");
grouperPasswordSave.assignApplication(GrouperPassword.Application.UI);
new Authentication().assignUserPassword(grouperPasswordSave);
```

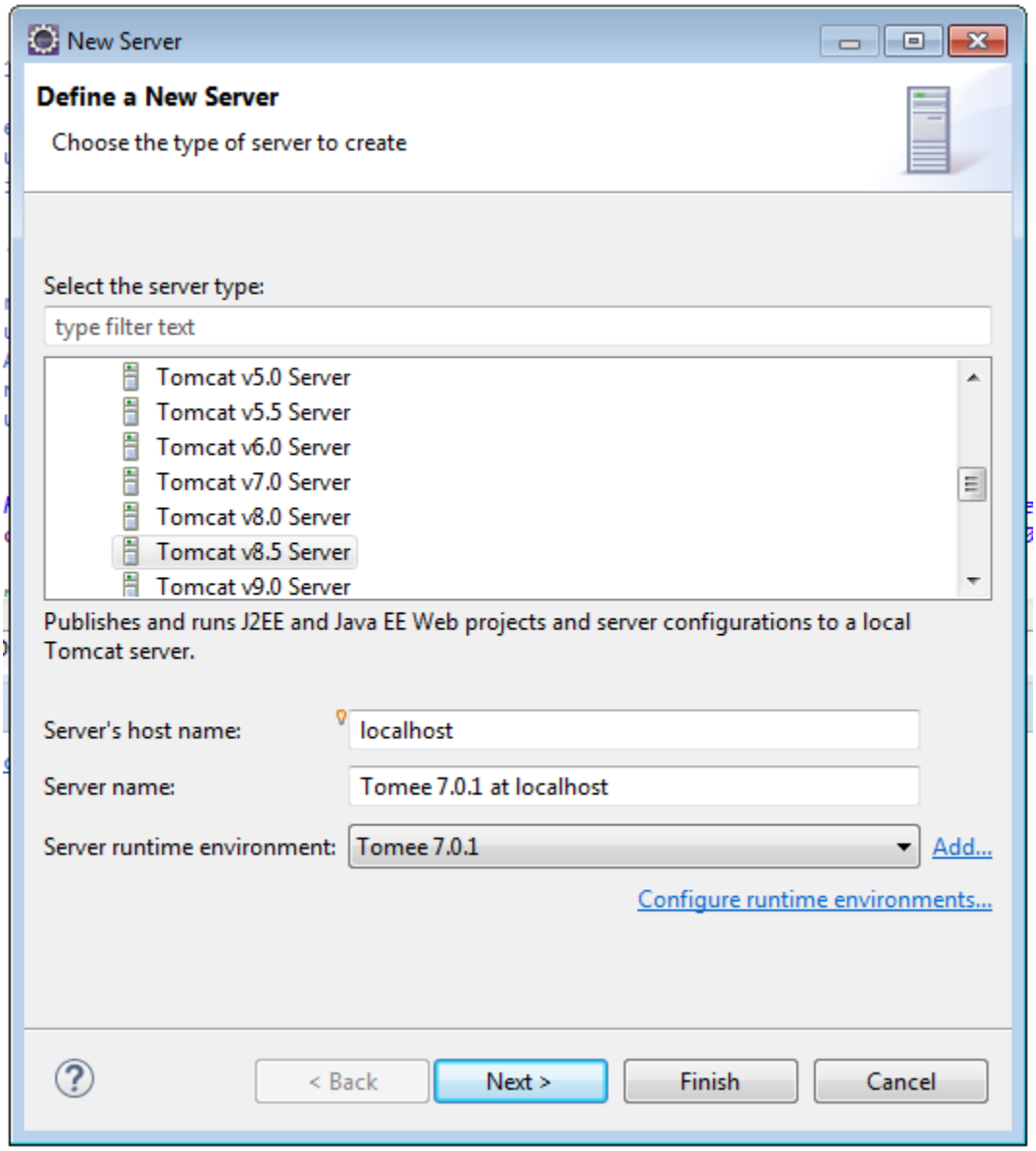
Run grouper-ui in Eclipse with Tomcat

Now that we have a Grouper database and some test subjects, the next step is to add the grouper-ui/webapp directory to the Eclipse Tomcat launcher so we can run and debug the grouper-ui.

Replace the tomee/lib/hsql jar with the one from grouper-ui/target/dependency

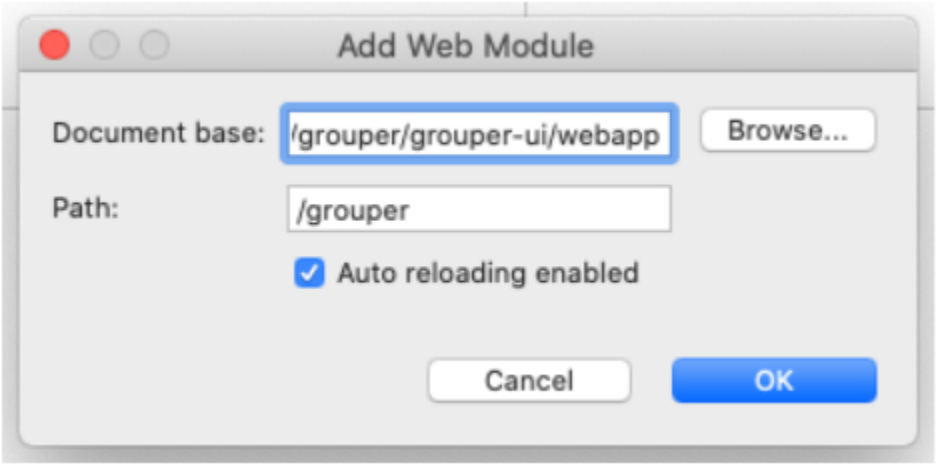
Add Tomcat server to Eclipse:

1. Eclipse -> J2EE Perspective -> Servers Tab
2. Click “No servers are available. Click this link to create a new server…”
    a. Select 'Tomcat v8.5'
    b. Select your tomee installation directory
    c. Select your JRE



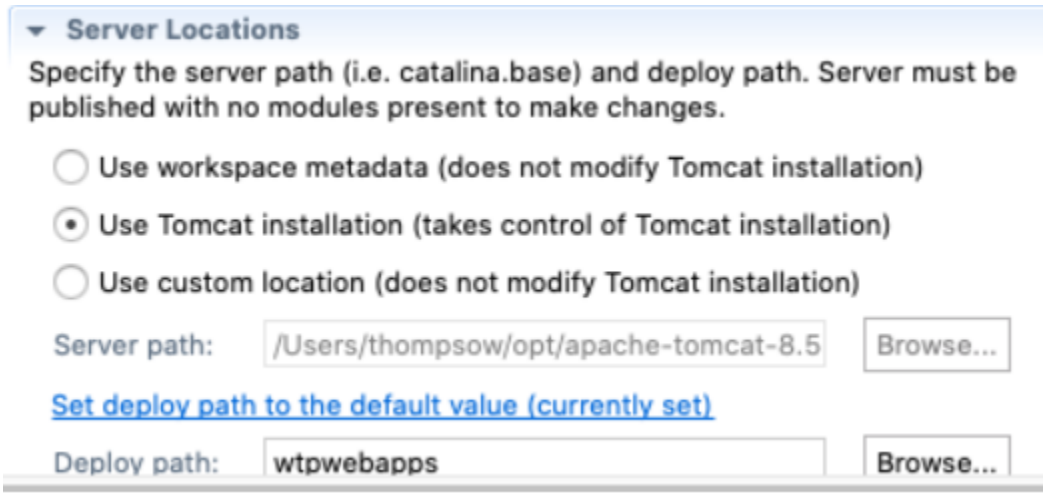
Add grouper-ui web module to Tomcat Server

1. Double click on Tomcat v8.5 at localhost [Stopped, Republish] to access configuration panel
2. Click on 'Modules' tab, click 'Add External Web Module…"



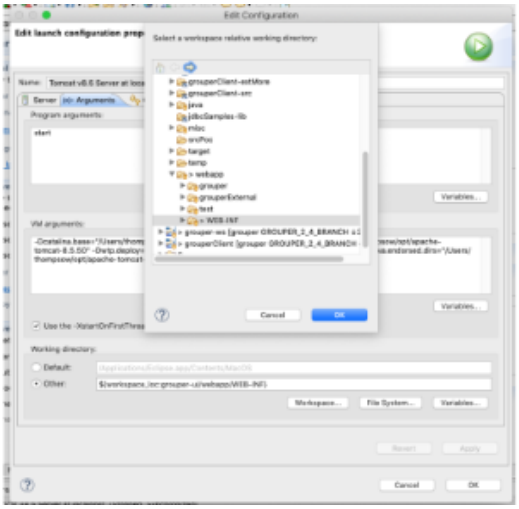
Configure Server Location

1. Select 'Use Tomcat installation (takes control Tomcat installation)



Configure Tomcat Server Working Directory to direct Grouper logs

1. Servers tab -> Overview -> click on "Open launch configuration"
2. (x)= Arguments tab
3. Working directory:
    a. Select grouper-ui/webapp/WEB-INF
    a. Choose Other
    b. Click Workspace…



Grouper logs will now show up under ../grouper-ui/webapp/WEB-INF/logs

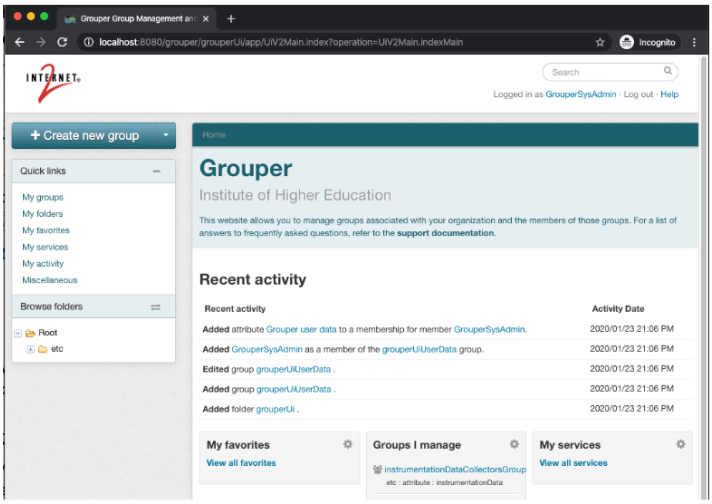
Update conf/grouper.hibernate.properties

```
grouper.is.ui = true
grouper.is.ui.basicAuthn = true
```

Run Grouper from Eclipse

Start Tomcat from Eclipse by selecting the server under the Servers tab and clicking the green 'Run' button.

Grouper UI should be available at http://localhost:8080/grouper. You should be able to log in with GrouperSystem or any of the test subjects and no password.



Debug Grouper from Eclipse

1. Set a breakpoint in UiV2Main.java line #114
2. Start the server in debug mode
3. Login and try to search in the Grouper UI
    a. This should hit the breakpoint in Eclipse and allow you step through the code.

Congrats! You now have a working Grouper development environment. Now go check out Grouper developers coding standards and then pick up some JIRAs!