NanoTraderApp Getting Started

Getting Started

Source control system

Our source control system is git, hosted at github.com. If you don't already have SpringSource access into github, request it from Brian Dussault, who in turn will request access from Trevor Marshall. nanoTrader is a private project under SpringSource account in github.

- Add your ssh public key under your github account ("Edit your profile"/"SSH Keys")
- *git config --global user.name "Your Name"*
- *git config --global user.email "youremailaddress@vmware.com"*
- *git config --global core.autocrlf <val>* (<val> = true for Windows, input for Mac/Linux; see http://help.github.com/line-endings/)
- For read-only access to Nanotrader:
    - *git clone git@github.com:SpringSource/nanotrader.git*
- For contributors to Nanotrader, we recommend an intermediate step of using a fork
    - Create a fork of the SpringSource/nanotrader repository, e.g. myname/nanotrader (Go to SpringSource/nanotrader, then click Fork button.)
        - This will allow you to save your revisions to github, without immediately pushing them out to everyone else on the team.
        - from your personal fork, changes can be pushed into the shared master repository (SpringSource/nanotrader)
        - Details are here
    - *git clone git@github.com:myname/nanotrader.git*
    - *cd nanotrader*
    - *git remote add upstream git@github.com:/SpringSource/nanotrader.git*
    - Note: if you already added the upstream using "git://github.com/SpringSource/nanotrader.git" (which won't work), then use the following command to fix it:
      git remote set-url upstream git@github.com:/SpringSource/nanotrader.git
- Optional: To enable git auto completion, follow these steps: http://www.codethatmatters.com/2010/01/git-autocomplete-in-mac-os-x/

Software installations

Required dependencies

- Postgres 9.x or later
- groovy 1.8.1 or later
- tc Server 2.6.3 or 2.7
- rabbitMQ 2.7 or later
- Erlang (required by rabbitMQ server)
- SQLfire 1.02 Professional Edition

Recommended dependencies

- rabbitMQ management plugin (to view channel/queue stats)
- pgAdmin
- DBVisualizer (for SQLFire)

Setup on a MAC

- It's recommended that you install all required dependencies using installers provided by software owners
- Alternatively use homebrew. Most of us use this on our mac without any issues but it doesn't work well if you already have macports.
- Homebrew install instructions are available [here | https://github.com/mxcl/homebrew/wiki/installation]

```
* brew install groovy
* brew install postgres
* brew install rabbitmq
```

- If you already have macports (http://guide.macports.org/), then install like this:

```
* sudo port install groovy
* sudo port install rabbitmq-server
```

- It is NOT recommended to install postgres via macports; it may install, but it's difficult to get running. Instead, use a DMG installer: http://www.postgresql.org/download/macosx/
- install tcServer (Developer version is available at: https://www.vmware.com/tryvmware/p/survey.php?p=tcserver-dev&lp=1 alternatively, download it from S3 from: /dist.springsource.com/release/TCS/)
- Install SQLFire (Developer version is available at https://my.vmware.com/web/vmware/info/slug/application_platform/vmware_vfabric_sqlfire/1_0, alternatively ,%20alternatively it can be downloaded from S3 Distribution)
    - Installation instructions are located at SQLfire Install Documentation

Build nanotrader application files

- `cd nanotrader`
- `./gradlew clean build`
    - If you get a "Java heap space" error, then set an environment variable for GRADLE_OPTS:

```
export GRADLE_OPTS='-Xmx1024m -Xms256m -XX:MaxPermSize=512m -XX:+CMSClassUnloadingEnabled
-XX:+HeapDumpOnOutOfMemoryError'
```

    - If successful this should generate following 3 war files

```
./spring-nanotrader-asynch-services/build/libs/spring-nanotrader-asynch-services-0.0.1.BUILD
```

Prepare a database: SQLFire (default) or Postgres

common database configuration

- add a definition of "nanodbserver" to your /etc/hosts

```
127.0.0.1 localhost nanodbserver
```

    - WARNING: on a MAC (seen on Lion and Snow Leopard), the /etc/hosts file can be reset by VPN connection software. See http://superuser.com/questions/354902/etc-hosts-getting-reset-in-lion. If you're using VPN "https://sslvpn-vmc.engx.vmware.com/", this fix will have no effect, so use "https://sslvpnb.vmware.com/" instead.
- Prepare database tools within nanotrader/tools
    - cd nanotrader/tools
- ./gradlew build
- cd build/libs
- unzip DataGenerator.zip

Postgres

- if not already done, install postgres. We've been using version 9.1.2 (confirm with *postgres --version*)
- cd nanotrader/tools/build/libs
- Update nanotrader.properties if your postgres database admin username/password is not postgres/postgres or it is not listening on default port (5432)

```
dbAdminUser = postgres
dbAdminPasswd = postgres
dbPort = 5432
```

- ./createSchema (this will create nanotrader user and database)

SQLFire

- if not already done, install SQLFire.

- **Start SQLFire Server**
  Run following command to start sqlfire with a single server and locator service

```
cd <vFabric SQLFire directory> (e.g. cd /Users/kparikh/vFabric_SQLFire_102)
 mkdir locator1 server1
 sqlf locator start -peer-discovery-address=127.0.0.1 -peer-discovery-port=3241 -dir=locator1
-client-port=1527 -client-bind-address=127.0.0.1
 sqlf server start -dir=server1 -client-bind-address=127.0.0.1 -client-port=1528
-locators=127.0.0.1[3241]
```

- cd /nanotrader/tools/build/libs
- update nanotrader.sqlf.properties if using non-default values
- run the createSqlfSchema script
  - *or* manually execute the following commands at the sqlf interactive shell:

```
self
CONNECT CLIENT 'nanodbserver:1527;user=nanotrader;password=nanotrader';
run 'nanotrader-sqlf-tables.ddl';
run 'initdb-sqlf.sql';
exit;
```

Starting nanotrader

- configure tcServer instance
  - Quick way is to use a tcServer template
    - `cd <tcserver-home>`
    - `copy the folder nanotrader/templates/nanotrader to ./templates`
    - `./tcruntime-instance.sh create nano -t nanotrader`
  - *OR* configure tcServer instance the long/slow way using manual updates
    - `cd <tcserver-home>`
    - `./tcruntime-instance.sh create nano`
    - `cd nano`
    - Update permgen memory for tc Server
      - Add `-XX:MaxPermSize=192m` to nano/bin/setenv.sh under JVM_OPTS

```
JVM_OPTS="-Xmx712M -Xss192K -XX:MaxPermSize=192m"
```

    - Configure the nanotrader TC JNDI Dataource. Add the following configuration to the <tc-instance>/conf/server.xml file:

```
<GlobalNamingResources>
...
<Resource auth="Container" driverClassName="com.vmware.sqlfire.jdbc.ClientDriver"
initialSize="10" jmxEnabled="true" logAbandoned="true" maxActive="20" maxIdle="10"
maxWait="-1" minEvictableIdleTimeMillis="30000" name="jdbc/nanodb"
password="nanotrader" removeAbandoned="true" removeAbandonedTimeout="60"
testOnBorrow="true" testOnReturn="false" testWhileIdle="true"
timeBetweenEvictionRunsMillis="5000" type="javax.sql.DataSource"
url="jdbc:sqlfire://nanodbserver:1527/" username="nanotrader"
validationInterval="30000" validationQuery="select 1 from
nanotrader.hibernate_sequences where sequence_name='ACCOUNT'"/>
...
 </GlobalNamingResources>
```

    - Link the global resource to the application. Add the following configuration to the <tc-instance>/conf/context.xml file:

```
<ResourceLink type="javax.sql.DataSource"
               name="jdbc/nanodb"
               global="jdbc/nanodb"/>
```

    - Copy the <sqlfire-root>/lib/sqlfireclient.jar (jdbc driver) to the tcServer instance lib directory <tcserver-home>/nano/lib.
    - Add `spring.profiles.active=production,jndi` property to nano/conf/catalina.properties.
  - copy following war files into the tcServer webapp directory. While copying the files remove "-0.0.1.BUILD-SNAPSHOT" from their names.

```
cp.
./spring-nanotrader-asynch-services/build/libs/spring-nanotrader-asynch-services-0.0.1.BUILD
<tcserver-home>/nano/webapps/spring-nanotrader-asynch-services.war
./spring-nanotrader-services/build/libs/spring-nanotrader-services-0.0.1.BUILD-SNAPSHOT.war
<tcserver-home>/nano/webapps/spring-nanotrader-services.war
./spring-nanotrader-web/build/libs/spring-nanotrader-web-0.0.1.BUILD-SNAPSHOT.war
<tcserver-home>/nano/webapps/spring-nanotrader-web.war
```

STS Configuration using tc server template
Note: To expedite the configuration of the STS server. you can copy the `git/nanotrader/templates/nanotrader-insight` directory to following location: `C:\software\sts-2-8\vfabric-tc-server-developer-2.6.1.RELEASE\templates` (Windows) or `/Users/username/springsource/vfabric-tc-server-developer-2.6.4.RELEASE\templates` (MAC).
Watch STS Server Configuration Step by Step Video

- start rabbitMQ server
  - `rabbitmq-server &`
- or start RabbitMQ Server detached:
  - `sudo rabbitmq-server -detached`
- start nanotrader instance
  - `<tcserver-home>/nano/bin/tcruntime-ctl.sh start`
- navigate to the following URL to view login page:
  - http://localhost:8080/spring-nanotrader-web (for the Android emulator, use: http://<VPN ip address>:8080/spring-nanotrader-web)
  - Note that the following may not work:
    - Versions of Safari prior to 5.1.5 (I think I had 3.X on OSX 10.6); you'll have to upgrade OSX in order to upgrade Safari
    - Versions of the iOS Simulator prior to 4.3; once again, you'll have to upgrade OSX in order to upgrade xCode
- generate sample data in the database
  - cd <your-nanotrader-root>/tools/build/libs
  - ./generateData
- login using a sample username/password such as nanouser1/nano