JVM Configuration Options

Some of the default settings for IntelliJ's JVM are not very performant. You can increase the speed and performance of your build by doing the following:

- Specify a path to JNotify in the JVM arguments - this improves the file watching capabilities and allows Scala to only rebuild when files have changed
- Increase the Maximum Memory (Xmx) that can be used by your build

Developer Environment Setup

1. Install JetBrains Toolbox
2. Install IntelliJ
3. Import Clowder SBT project into IntelliJ - you should be automatically prompted to install SBT / Scala support
4. Create a "Play 2.0 App" Run Configuration
5. Download JNotify and edit your Run Configuration to add it to your path - your JVM arguments should include **-Djava.library.path=path/to/jnotify**

JNotify Setup

To compile Clowder you will need to have at least version 1.7 of JAVA. Version 1.8 will work as well.

On Windows and Mac, the JNotify library is required and not included with Java 1.8 64bit - this will result in degraded (e.g. extremely sluggish navigation) performance when debugging your developer instance.

The following message will display in the development console in IntelliJ if playframework is unable to locate JNotify in this case:

```
Cannot load the JNotify native library (no jnotify_64bit in java.library.path)
Play will check file changes for each request, so expect degraded reloading performace.
```

JNotify Documentation can be found here: http://jnotify.sourceforge.net/

Download JNotify here: https://sourceforge.net/projects/jnotify/files/

Windows

Simply paste **jnotify_64bit.dll** to the path\to\jdk1.8\bin folder.

Mac OS X

Create a new folder called "jnotify" in the "lib" directory: e.g. /Library/Java/JavaVirtualMachines/jdk1.8.0_77.jdk/Contents/Home/lib/jnotify

Next, paste **jnotify-0.94.jar**, **libjnotify.so**, and **libjnotify.jnilib** to the path/to/jdk1.8/lib/jnotify folder.

Finally, add the following argument (adjusted with your correct path) to the JVM arguments in your Run Configuration for Clowder in IntelliJ:

```
-Djava.library.path=/Library/Java/JavaVirtualMachines/jdk1.8.0_77.jdk/Contents/Home/lib/jnotify
```

Builds too slow? Increase -Xmx

In IntelliJ, often the default memory constraints are too low for Scala projects.

From the top menu, choose Help Edit Custom VM Options and find the `-Xmx` flag.

This is your JVM's maximum memory heap size, which defaulted to 750MB for me.

Upping this value will decrease the amount of garbage collection necessary during build, and can therefore increase your overall build speed.

Required Dependencies

There are still a few things missing before Clowder will run properly

1. MongoDB: docker run -it -d --name=mongo -v $(pwd)/mongo:/data/db -p 27017:27017 mongo:3.4

Optional dependencies

1. For RabbitMQ plugin (extractors): docker run -itd --name=rabbit -p 5672:5672 -p 15672:15672 rabbitmq:management-alpine
2. For Elasticsearch plugin (search indexing): docker run -itd --name=elasticsearch -p 9200:9200 -p 9300:9300 elasticsearch:2
3. For ToolManager plugin (analysis environments): docker run -itd --name=toolserver -p 8082:8082 clowder/toolserver
4. For pyCSW plugin (?): docker build -t clowder:pycsw . && docker run -it -d --name pycsw -p 80:8000 clowder/pycsw
5. For Geoserver plugin (QGIS analysis): ??

Running Extractors

```
docker run -it --rm --net=host -e RABBITMQ_URI="amqp://guest:guest@localhost/%2f" -e REGISTRATION_ENDPOINTS="http://141.142.60.193:9000/api/extractors?key=r1ek3rs" -v $(pwd):/home/clowder clowder/extractors-image-metadata
```