*HPC Data MANAGEMENT*

DEVELOPMENT ENVIRONMENT SETUP

Version *1.8*

*06/11/2021*

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Implemented**  **By** | **Revision**  **Date** | **Description of Change** |
| 1.0 | Eran Rosenberg | *5/17/2015* | Initial Draft |
| 1.1 | Eran Rosenberg | 8/17/2015 | Create local Keystore |
| 1.2 | Eran Rosenberg | 1/1/2017 | Updates after lib/tools upgrades |
| 1.3 | Eran Rosenberg | 6/12/2017 | Configure servicemix w/ Keystore + Git repo |
| 1.4 | William Liu | 11/16/2017 | Added material in Build section about saving time on Maven builds |
| 1.5 | Eran Rosenberg | 12/17/2017 | Added section to configure Eclipse and Code Style formatter |
| 1.6 | Eran Rosenberg | 12/20/2017 | Changed Servicemix config for keystore |
| 1.7 | Sunita Menon | 01/31/2019 | Updated Git Code Repository, ServiceMix and Deploy API To ServiceMix section. Added sections HPCDME Web Client and HPCDME CLI Client. |
| 1.8 | Sunita Menon | 06/11/2021 | Updated software versions, updated DB section |

TABLE OF CONTENTS

[Git Code Repository 4](#_Toc505280670)

[Java Development Kit (JDK) 4](#_Toc505280671)

[MAVEN 4](#_Toc505280672)

[SERVICEMIX 4](#_Toc505280673)

[PostgreSQL DB 5](#_Toc505280674)

[IRODS 5](#_Toc505280675)

[PATH Environment Variable 5](#_Toc505280676)

[BUILD 5](#_Toc505280677)

[DEPLOY API to SERVICEMIX 6](#_Toc505280678)

[ENDPOINTS 6](#_Toc505280679)

[SOAP-UI 7](#_Toc505280680)

[HPCDME WEB Client 7](#_Toc505280681)

[HPCDME CLI Client 7](#_Toc505280682)

[Eclipse 8](#_Toc505280683)

# Git Code Repository

Check out the source code from Git:

Repository URL: <https://github.com/CBIIT/HPC_DME_APIs>

Branch name: master

Set HPC\_HOME environment variable to the <Path to HPC\_DME\_APIs>/src directory path in the source tree.

e.g. /Development/HPC\_DME\_APIs/src

# Java Development Kit (JDK)

Install JDK 1.8.x: https://www.oracle.com/java/technologies/javase-downloads.html

Set JAVA\_HOME environment variable accordingly.

# MAVEN

Install Maven 3.5.0: <http://maven.apache.org/download.cgi>

Set MAVEN\_HOME environment variable accordingly. For e.g.

export MAVEN\_HOME=/Users/menons2/Development/programs/apache-maven-3.5.0

# SERVICEMIX

Install Servicemix 7.0.1.: <http://servicemix.apache.org/downloads.html>

Set SERVICE\_MIX\_HOME environment variable accordingly. For e.g.

export SERVICE\_MIX\_HOME=/Users/menons2/Development/programs/apache-servicemix-7.0.1

To allow the server to communicate with iRODS and LDAP, we need to deploy/configure a keystore to Servicemix:

1. Add the following to the end of $SERVICE\_MIX\_HOME/etc/system.properties

# HPC-DM keystore

javax.net.ssl.keyStore=${karaf.home}/etc/hpc-server/cacerts.jks

javax.net.ssl.keyStorePassword=changeit

javax.net.ssl.trustStore=${karaf.home}/etc/hpc-server/cacerts.jks

javax.net.ssl.trustStorePassword=changeit

1. Update $HPC\_HOME/hpc-server/hpc-features/src/main/resources/hpc-server-local.properties as follows:
   1. Set the hpc.bus.aspect.systemAdministratorUserId to be your nih username
   2. Set the hpc.integration.ldap.password.
   3. Set hpc.dao.oracle.password.

**Important Note:** Ensure that during deployment, this file is copied to

$SERVICE\_MIX\_HOME/etc/gov.nih.nci.hpc.cfg

# 

# Oracle DB

Ensure that the property file indicated abovein step 2 **hpc-server-local.properties** is pointing to the already installed Dev. Environment Instance.

# IRODS

You will need an iRODS account in the DEV environment. Ask a team member to register you as an HPC-DM user in the DEV environment.. The registration in DEV will create your iRODS user.

# PATH Environment Variable

Set your $PATH environment variable

export PATH=$MAVEN\_HOME/bin:$SERVICE\_MIX\_HOME/bin:$PATH

# BUILD

cd $HPC\_HOME

mvn clean install

Saving time on Maven builds

The above instruction runs Maven build of **all** HPC DME software products. This could be overkill, and consequently you spend more time executing a build than necessary due to building products that you do not need.

To save time on Maven builds, invoke Maven using the -pl option to exclude modules from the build process. Two Maven command line invocations applying the -pl option are shown next as examples.

Windows command line invocation, notice double quotes for -pl switch argument

>mvn **-pl “!hpc-server/hpc-ws-rs-test,!hpc-cli”** clean install

Unix/Linux/Mac command line invocation, notice single quotes for -pl switch argument

>mvn **-pl** **‘!hpc-server/hpc-ws-rs-test,!hpc-cli’** clean install

With the -pl option, you can indicate which modules to skip by specifying a comma-separated list of module specifications each prefixed with !.  In the above examples, the **hpc-ws-rs-test** module of the **hpc-server** module and the **hpc-cli** module are declared to be skipped.

The preceding examples assume you want to build all modules except for specific exclusion(s). If instead you want to build specific module(s) and exclude the rest, you can use the -pl option to specify comma-separated list of modules you wish to build. In that case, each module is declared without the ! prefix.

# DEPLOY API to SERVICEMIX

Find the server version (from the top level pom file). You will need to use it in the command to start the server.

Run Servicemix console with the command:

servicemix clean debug

**Note:** The ‘clean’ option resets servicemix to a clean state, this will delete the data directory in the installation folder.

The ‘debug’ option enables attaching a debugger on port 5005.

Then Install hpc-server (in Servicemix console) using the commands:

feature:repo-add mvn:gov.nih.nci.hpc/hpc-features*/<server-version>/*xml/features

feature:install hpc-server-rest-services

Optionally (please see note below first), install the scheduler using the command:

feature:install hpc-server-scheduler

**Important Note**: In a local development environment, we typically skip the 3rd step above (server-scheduler install) since we share the same iRODS server with DEV and the scheduler is running there. Having 2 scheduler running (pointing to the same iRODS but having different HPC DB) will cause issues in the async upload of data objects.

# ENDPOINTS

HPC DM Services are deployed to [https://localhost:7338/hpc-server/<hpc-service](https://localhost:7338/hpc-server/%3chpc-service)>

This is the URL to be used for accessing the services from SOAP-UI or curl.

# HPCDME WEB Client

cd $HPC\_HOME/hpc-web

mvn clean install

To start the web client,

java –Xdebug –Xrunjdwp:server=y, transport=dt\_socket,suspend=n,address=5015 –jar target/<hpc-web-<version number>.war

For e.g. if the version number is 2.8.0, then run

java –Xdebug –Xrunjdwp:server=y, transport=dt\_socket,suspend=n,address=5015 –jar target/hpc-web-2.8.0.war

The URL of the local web client is

<https://localhost:9080>

# HPCDME CLU/CLI Client

1. Set HPC\_DM\_UTILS environment variable to the <Path to HPC\_DME\_APIs>/utils directory in the source tree e.g. /Development/HPC\_DME\_APIs/utils
2. Copy the file keystore.jks from $SERVICE\_MIX\_HOME/etc/hpc-server/ directory to $HPC\_DM\_UTILS/hpc-client/keystore/ directory.
3. Update/uncomment the server url and keystore properties in $HPC\_DM\_UTILS/hpcdme.properties:

hpc.server.url=https://localhost:7738/hpc-server

hpc.ssl.keystore.path=hpc-client/keystore/keystore.jks

hpc.ssl.keystore.password=<replace with keystore password>

To start the CLI Client,

cd $HPC\_DM\_UTILS/hpc-client

java –Xdebug –Xrunjdwp:server=y,transport=dt\_socket,suspend=n,address=5025 -Dhpc.client.properties=$HPC\_DM\_UTILS/hpcdme.properties –jar hpc-client/<hpc-cli-<version-number>.jar

For e.g. if the version number is 2.8.0 then run

java –Xdebug –Xrunjdwp:server=y, transport=dt\_socket,suspend=n,address=5025 -Dhpc.client.properties=$HPC\_DM\_UTILS/hpcdme.properties –jar hpc-client/hpc-cli-2.8.0.jar

**Important Note**: If CLUs (Command Line Utilities) are being used, then it is not required to start the CLI Client, this will be invoked internally from the CLUs as required.

# Eclipse

To create Eclipse projects:

cd $HPC\_HOME/hpc-server

mvn eclipse:eclipse

cd $HPC\_HOME/hpc-web

mvn eclipse:eclipse

You can now import the HPC-DME projects into Eclipse. Make sure it builds clean within Eclipse.

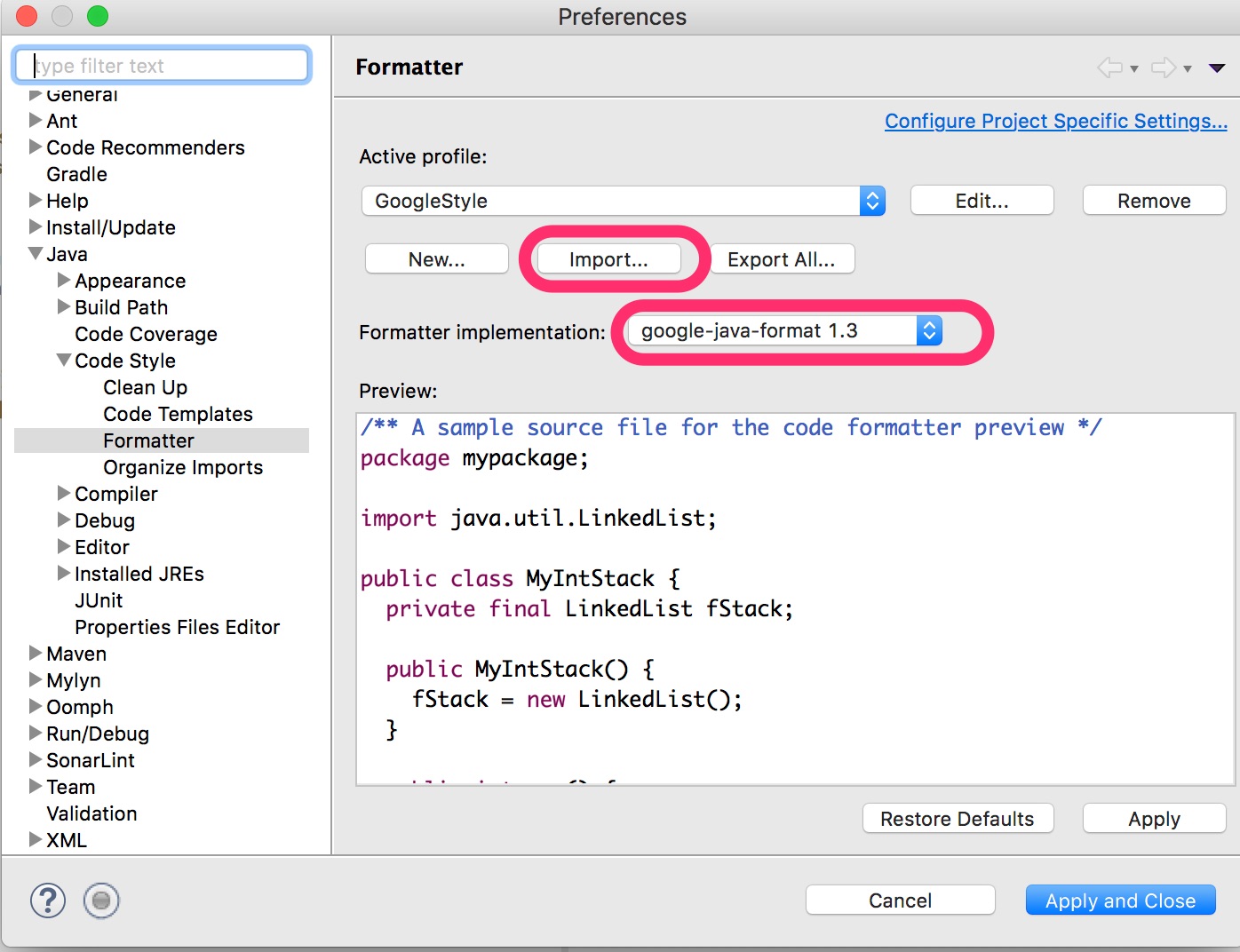
We use the Google Code Style for Java. To set up eclipse, we will use 2 files found under:

<https://github.com/CBIIT/HPC_DME_APIs/blob/master/doc/development/hpc-code-style>

First copy the jar file to your Eclipse ‘dropins’ folder. Then restart Eclipse.

Go to Preferences -> Java -> Code Style -> Formatter

Import the XML file as your profile, and select the google formatter.



# SOAP-UI

Soap UI workspace and project can be found under $HPC\_HOME/hpc-soap-ui folder.

Note – after pointing soap-ui to the workspace, you will need to correct the path to the project based on the directory you pulled the code into.