# VDI Setup

1. Please check if JDK is installed on your box. If JDK is not installed please install JDK jdk1.7.0\_17.
2. Set JAVA\_HOME system variable to the JDK path.
3. Add %JAVA\_HOME%\bin to the end of the system variable PATH.
4. Install apache maven on the hard drive.
5. Set M2\_HOME system variable to apache maven directory.
6. Add %M2\_HOME%\bin the system variable PATH.
7. Log into cloud hub. Click on the support. Go to Downloads. Download latest AnyPointStudio.
8. Once AnyPointStudio is launched go to Windows Preferences. In the select Network connections. Make sure you select the mode to Manual. Edit HTTP and HTTPS to provide your userid/password for the proxy server. Select SOCKS hit clear button to clear all entries in the SOCKS. This is very important. Select OK to set the Window network Preferences.
9. Open Window Preferences in AnyPointStudio set up JDK installation, if it is not picked up automatically. Installed JREs should point to JDK installed on your machine.
10. Install Maven Eclipse plugin installation step by step
11. Click Help -> Install New Software...
12. Click Add button at top right corner
13. At pop up: fill up Name as "M2Eclipse" and Location as "<http://download.eclipse.org/technology/m2e/releases>"
14. Now click OK
15. After installed M2 plugin restart the AnyPoint Studio.
16. Create settings.xml in the directory of C:\users\<KOID>\.m2 from the settings.xml that is added in the APPENDIX section of this document.
17. Go to Window Preferences again. In the Maven subsection Add your maven installation on the hard drive to the installations. Go to user settings select settings.xml from hard drive if it not already there.
18. Install TORTOISE svn. Once tortoise svn is installed go to settings and under the section of network, provide your proxy,user and password details.
19. Check out the code using TORTOISE SVN.
20. Go to AnyPointStudio select import and choose pom file to import the project into AnyPointStudio.
21. Once you are imported, maven will download all the dependencies and update the maven dependencies in the build path.
22. If Maven task fails, it could be necessary to install the jar file manually.
23. Use mvn install:install-file command to install the jar files manually.

# Appendix settings.xml

<?xml version="1.0" encoding="UTF-8"?>

<!--

Licensed to the Apache Software Foundation (ASF) under one

or more contributor license agreements. See the NOTICE file

distributed with this work for additional information

regarding copyright ownership. The ASF licenses this file

to you under the Apache License, Version 2.0 (the

"License"); you may not use this file except in compliance

with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing,

software distributed under the License is distributed on an

"AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY

KIND, either express or implied. See the License for the

specific language governing permissions and limitations

under the License.

-->

<!--

| This is the configuration file for Maven. It can be specified at two levels:

|

| 1. User Level. This settings.xml file provides configuration for a single user,

| and is normally provided in ${user.home}/.m2/settings.xml.

|

| NOTE: This location can be overridden with the CLI option:

|

| -s /path/to/user/settings.xml

|

| 2. Global Level. This settings.xml file provides configuration for all Maven

| users on a machine (assuming they're all using the same Maven

| installation). It's normally provided in

| ${maven.home}/conf/settings.xml.

|

| NOTE: This location can be overridden with the CLI option:

|

| -gs /path/to/global/settings.xml

|

| The sections in this sample file are intended to give you a running start at

| getting the most out of your Maven installation. Where appropriate, the default

| values (values used when the setting is not specified) are provided.

|

|-->

<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">

<!-- localRepository

| The path to the local repository maven will use to store artifacts.

|

| Default: ${user.home}/.m2/repository

<localRepository>/path/to/local/repo</localRepository>

-->

<!-- interactiveMode

| This will determine whether maven prompts you when it needs input. If set to false,

| maven will use a sensible default value, perhaps based on some other setting, for

| the parameter in question.

|

| Default: true

<interactiveMode>true</interactiveMode>

-->

<!-- offline

| Determines whether maven should attempt to connect to the network when executing a build.

| This will have an effect on artifact downloads, artifact deployment, and others.

|

| Default: false

<offline>false</offline>

-->

<!-- pluginGroups

| This is a list of additional group identifiers that will be searched when resolving plugins by their prefix, i.e.

| when invoking a command line like "mvn prefix:goal". Maven will automatically add the group identifiers

| "org.apache.maven.plugins" and "org.codehaus.mojo" if these are not already contained in the list.

|-->

<pluginGroups>

<!-- pluginGroup

| Specifies a further group identifier to use for plugin lookup.

<pluginGroup>com.your.plugins</pluginGroup>

-->

<!-- Mule Plugin Group -->

<pluginGroup>org.mule.tools</pluginGroup>

</pluginGroups>

<!-- proxies

| This is a list of proxies which can be used on this machine to connect to the network.

| Unless otherwise specified (by system property or command-line switch), the first proxy

| specification in this list marked as active will be used.

|-->

<proxies>

<!-- proxy

| Specification for one proxy, to be used in connecting to the network.

|

<proxy>

<id>optional</id>

<active>true</active>

<protocol>http</protocol>

<username>proxyuser</username>

<password>proxypass</password>

<host>proxy.host.net</host>

<port>80</port>

<nonProxyHosts>local.net|some.host.com</nonProxyHosts>

</proxy>

-->

<proxy>

<id>proxy</id>

<active>true</active>

<protocol>http</protocol>

<username>your ko id</username>

<password>Yourpassword</password>

<host>webproxy.na.ko.com</host>

<port>880</port>

<nonProxyHosts>maven</nonProxyHosts>

</proxy>

<proxy>

<id>https-proxy</id>

<active>true</active>

<protocol>https</protocol>

<username>your ko id</username>

<password>Your password</password>

<host>webproxy.na.ko.com</host>

<port>880</port>

<nonProxyHosts>internal.hostname</nonProxyHosts>

</proxy>

</proxies>

<!-- servers

| This is a list of authentication profiles, keyed by the server-id used within the system.

| Authentication profiles can be used whenever maven must make a connection to a remote server.

|-->

<servers>

<!-- server

| Specifies the authentication information to use when connecting to a particular server, identified by

| a unique name within the system (referred to by the 'id' attribute below).

|

| NOTE: You should either specify username/password OR privateKey/passphrase, since these pairings are

| used together.

|

<server>

<id>deploymentRepo</id>

<username>repouser</username>

<password>repopwd</password>

</server>

-->

<!-- Another sample, using keys to authenticate.

<server>

<id>siteServer</id>

<privateKey>/path/to/private/key</privateKey>

<passphrase>optional; leave empty if not used.</passphrase>

</server>

-->

<!-- Mule enterprise maven repository -->

<server>

<id>MuleRepository</id>

<username>coca.cola.nexus</username>

<password>7grtdfUEmsdd4</password>

</server>

</servers>

<!-- mirrors

| This is a list of mirrors to be used in downloading artifacts from remote repositories.

|

| It works like this: a POM may declare a repository to use in resolving certain artifacts.

| However, this repository may have problems with heavy traffic at times, so people have mirrored

| it to several places.

|

| That repository definition will have a unique id, so we can create a mirror reference for that

| repository, to be used as an alternate download site. The mirror site will be the preferred

| server for that repository.

|-->

<mirrors>

<!-- mirror

| Specifies a repository mirror site to use instead of a given repository. The repository that

| this mirror serves has an ID that matches the mirrorOf element of this mirror. IDs are used

| for inheritance and direct lookup purposes, and must be unique across the set of mirrors.

|

<mirror>

<id>mirrorId</id>

<mirrorOf>repositoryId</mirrorOf>

<name>Human Readable Name for this Mirror.</name>

<url>http://my.repository.com/repo/path</url>

</mirror>

-->

</mirrors>

<!-- profiles

| This is a list of profiles which can be activated in a variety of ways, and which can modify

| the build process. Profiles provided in the settings.xml are intended to provide local machine-

| specific paths and repository locations which allow the build to work in the local environment.

|

| For example, if you have an integration testing plugin - like cactus - that needs to know where

| your Tomcat instance is installed, you can provide a variable here such that the variable is

| dereferenced during the build process to configure the cactus plugin.

|

| As noted above, profiles can be activated in a variety of ways. One way - the activeProfiles

| section of this document (settings.xml) - will be discussed later. Another way essentially

| relies on the detection of a system property, either matching a particular value for the property,

| or merely testing its existence. Profiles can also be activated by JDK version prefix, where a

| value of '1.4' might activate a profile when the build is executed on a JDK version of '1.4.2\_07'.

| Finally, the list of active profiles can be specified directly from the command line.

|

| NOTE: For profiles defined in the settings.xml, you are restricted to specifying only artifact

| repositories, plugin repositories, and free-form properties to be used as configuration

| variables for plugins in the POM.

|

|-->

<profiles>

<!-- profile

| Specifies a set of introductions to the build process, to be activated using one or more of the

| mechanisms described above. For inheritance purposes, and to activate profiles via <activatedProfiles/>

| or the command line, profiles have to have an ID that is unique.

|

| An encouraged best practice for profile identification is to use a consistent naming convention

| for profiles, such as 'env-dev', 'env-test', 'env-production', 'user-jdcasey', 'user-brett', etc.

| This will make it more intuitive to understand what the set of introduced profiles is attempting

| to accomplish, particularly when you only have a list of profile id's for debug.

|

| This profile example uses the JDK version to trigger activation, and provides a JDK-specific repo.

<profile>

<id>jdk-1.4</id>

<activation>

<jdk>1.4</jdk>

</activation>

<repositories>

<repository>

<id>jdk14</id>

<name>Repository for JDK 1.4 builds</name>

<url>http://www.myhost.com/maven/jdk14</url>

<layout>default</layout>

<snapshotPolicy>always</snapshotPolicy>

</repository>

</repositories>

</profile>

-->

<!--

| Here is another profile, activated by the system property 'target-env' with a value of 'dev',

| which provides a specific path to the Tomcat instance. To use this, your plugin configuration

| might hypothetically look like:

|

| ...

| <plugin>

| <groupId>org.myco.myplugins</groupId>

| <artifactId>myplugin</artifactId>

|

| <configuration>

| <tomcatLocation>${tomcatPath}</tomcatLocation>

| </configuration>

| </plugin>

| ...

|

| NOTE: If you just wanted to inject this configuration whenever someone set 'target-env' to

| anything, you could just leave off the <value/> inside the activation-property.

|

<profile>

<id>env-dev</id>

<activation>

<property>

<name>target-env</name>

<value>dev</value>

</property>

</activation>

<properties>

<tomcatPath>/path/to/tomcat/instance</tomcatPath>

</properties>

</profile>

-->

<!-- Mule enterprise maven repository profile -->

<profile>

<id>Mule</id>

<activation>

<activeByDefault>true</activeByDefault>

</activation>

<repositories>

<repository>

<id>Central</id>

<name>Central</name>

<url>http://repo1.maven.org/maven2/</url>

<layout>default</layout>

</repository>

<repository>

<id>MuleRepository</id>

<name>MuleRepository</name>

<url>https://repository.mulesoft.org/nexus-ee/content/repositories/releases-ee/</url>

<layout>default</layout>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

</repository>

<repository>

<id>codehaus-mule-repo</id>

<name>codehaus-mule-repo</name>

<url>https://repository-master.mulesoft.org/nexus/content/groups/public/</url>

<layout>default</layout>

</repository>

<repository>

<id>splunk-artifactory</id>

<name>Splunk Releases</name>

<url>http://splunk.artifactoryonline.com/splunk/ext-releases-local</url>

</repository>

<!--

<repository>

<id>mulesoft-releases</id>

<name>MuleSoft Releases Repository</name>

<url>http://repository.mulesoft.org/releases/</url>

<layout>default</layout>

</repository>

<repository>

<id>mulesoft-snapshots</id>

<name>MuleSoft Snapshots Repository</name>

<url>http://repository.mulesoft.org/snapshots/</url>

<layout>default</layout>

</repository>

-->

</repositories>

<pluginRepositories>

<pluginRepository>

<id>mule-public</id>

<url> https://repository.mulesoft.org/nexus/content/repositories/public </url>

</pluginRepository>

</pluginRepositories>

</profile>

</profiles>

<!-- activeProfiles

| List of profiles that are active for all builds.

|

-->

<activeProfiles>

<activeProfile>Mule</activeProfile>

</activeProfiles>

</settings>