PDTool Developer's Startup Guide

An Open Source Asset for use with TIBCO® Data Virtualization

Project Name	AS Assets PDTool (Promotion and Deployment Tool)
Document Location	This document is only valid on the day it was printed. The source of the document will be found in the PDTool and PDToolRelease folder (https://github.com/TIBCOSoftware)
Purpose	Developer's Guide

www.tibco.com

Global Headquarters
3303 Hillview Avenue
Palo Alto, CA 94304

Tel: +1 650-846-1000
+1 800-420-8450
Fax: +1 650-846-1005

Revision History

Version	Date	Author	Comments
1.0	12/06/2017	Mike Tinius	Initial revision with Tibco
1.1	10/20/2020	Mike Tinius	Updated documentation

Related Documents

Name	Version

Supported Versions

Name	Version
TIBCO® Data Virtualization	7.0.4 or later

Table of Contents

1 Introduction

Purpose

The purpose of this document is to provide guidance on how develop the AS Assets PDTool (Promotion and Development Tool).

Audience

This document is intended to provide guidance for the following users:

- Developers

References

Product references are shown below. Any references to CIS or DV refer to the current TIBCO® Data Virtualization.

- TIBCO® Data Virtualization was formerly known as
  - Cisco Data Virtualization (DV)
  - Composite Information Server (CIS)

2 Github Repository Structure

Recommended Development Tools

CIS itself can (and should) be used for developing the SQL Script procedures. Any robust Java IDE can be used to develop the CJPs, however up to this point, the CJP's have been developed using Eclipse. The CJP artifacts in the open source repository (GitHub) are therefore geared towards an Eclipse project.

Git is used as the version control system for the ASAssets open source project. For those new to Git, a visit to https://help.github.com/articles/set-up-git will provide an overview of Git and instructions on downloading and setting up the basic Git tools.

How Can I Download a Copy of the PDTool and PDToolRelease GitHub Repository?

The GitHub master repository is located at https://github.com/TIBCOSoftware/PDTool. The PDTool Release master repository is located at https://github.com/TIBCOSoftware/PDToolRelease.

Repository Folder Structure

The folder structure of the GitHub repository contains Java source, documentation source, and distribution resources.

3 Configure the Eclipse Development Environment

Downloading Eclipse

1. Download the latest Eclipse from here:

http://www.eclipse.org/platform

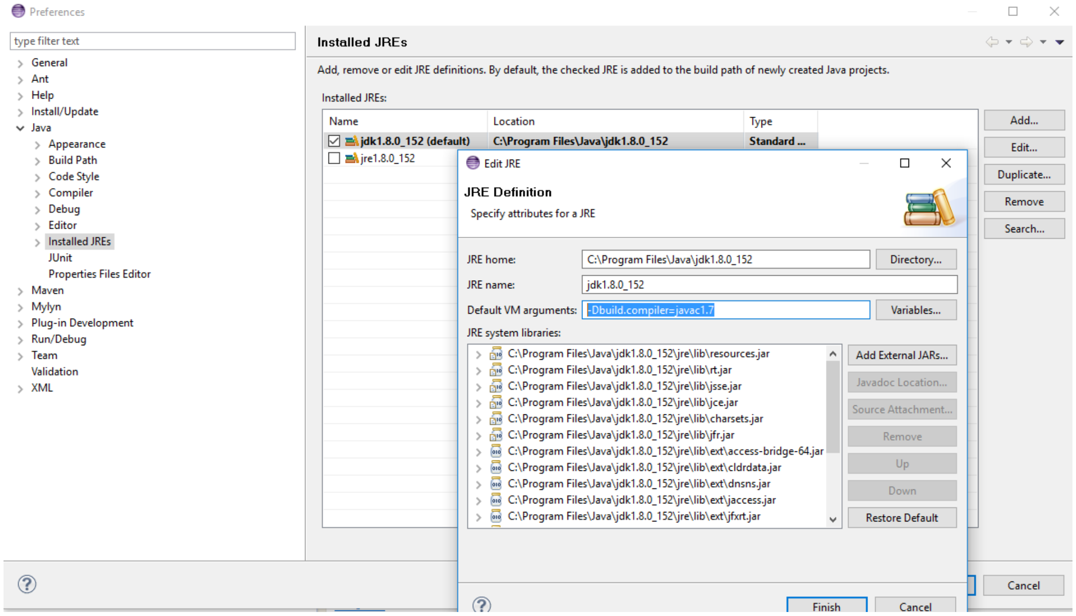
Eclipse IDE for Java Developers (includes Incubating components)
Version: 2020-06 (4.16.0)
Build id: 20200615-1200

(c) Copyright Eclipse contributors and others 2000, 2020. All rights reserved. Eclipse and the Eclipse logo are trademarks of the Eclipse Foundation, Inc., https://www.eclipse.org/.

The Eclipse logo cannot be altered without Eclipses permission. Eclipse logos are provided for use under the Eclipse logo and trademark guidelines, https://www.eclipse.org/logotm/. Oracle and Java are trademarks or registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

This product includes software developed by other open source projects including the Apache Software Foundation, https://www.apache.org/.

Set the Installed JREs Default VM arguments
PDTool 7.0.0 and PDTool 8.0.0 use JDK 1.8
PDTool 8.3.0 use JDK 11
From eclipse, open Windows → Preferences → Java → Click on Installed JREs → Edit jdk 1.8.0 and set the VM arguments.
-Dbuild.compiler=javac1.8



Installed JREs

Add, remove or edit JRE definitions. By default, the checked JRE is added to the build path of newly created Java projects.

Installed JREs:

Name	Location	Type
☑ jdk1.8.0_152 (default)	C:\Program Files\Java\jdk1.8.0_152	Standard ...
□ jre1.8.0_152		

JRE Definition

Specify attributes for a JRE

JRE home: C:\Program Files\Java\jdk1.8.0_152

JRE name: jdk1.8.0_152

Default VM arguments: -Dbuild.compiler=javac1.7

JRE system libraries:

```
C:\Program Files\Java\jdk1.8.0_152\jre\lib\resources.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\rt.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\jsse.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\jce.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\charsets.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\jfr.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\ext\access-bridge-64.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\ext\cldrdata.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\ext\dnsns.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\ext\jaccess.jar
C:\Program Files\Java\jdk1.8.0_152\jre\lib\ext\jfxrt.jar
```

2. **Open Eclipse, Install New Software, enter this URL:**

   http://eclipse.org/egit/download/

Configure Eclipse Variables

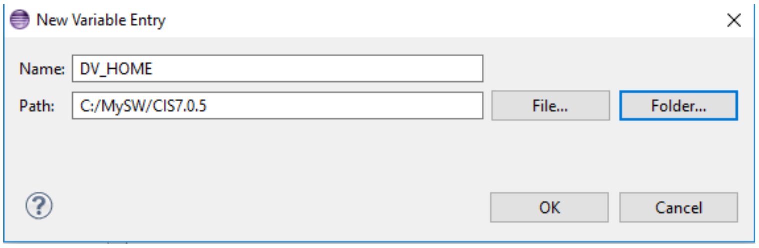
There are two variables that need to be created. One is for Eclipse and one is for Ant.

Create DV_HOME for Eclipse.

In Eclipse, select Window > Preferences > Java > Build Path > Classpath Variables > New

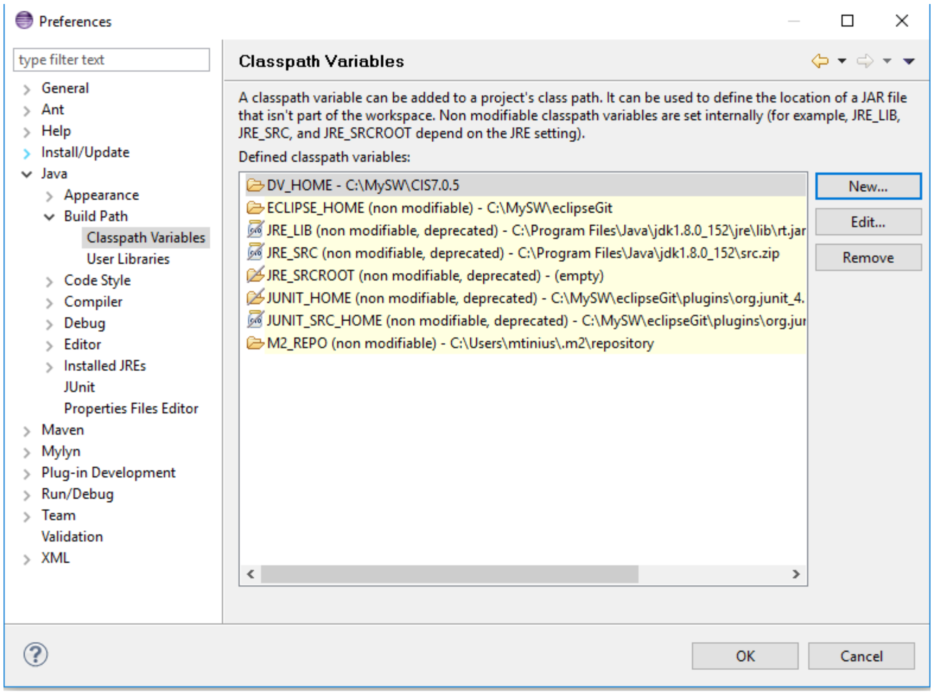
Enter Name: DV_HOME

Path: Browse to your data virtualization home folder


New Variable Entry

Name: DV_HOME

Path: C:/MySW/CIS7.0.5


Preferences

Classpath Variables

A classpath variable can be added to a project's class path. It can be used to define the location of a JAR file that isn't part of the workspace. Non modifiable classpath variables are set internally (for example, JRE_LIB, JRE_SRC, and JRE_SRCROOT depend on the JRE setting).

Defined classpath variables:

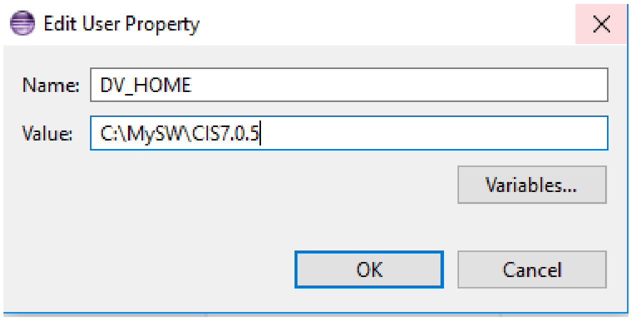
- DV_HOME - C:\MySW\CIS7.0.5
- ECLIPSE_HOME (non modifiable) - C:\MySW\eclipseGit
- JRE_LIB (non modifiable, deprecated) - C:\Program Files\Java\jdk1.8.0_152\jre\lib\rt.jar
- JRE_SRC (non modifiable, deprecated) - C:\Program Files\Java\jdk1.8.0_152\src.zip
- JRE_SRCROOT (non modifiable, deprecated) - (empty)
- JUNIT_HOME (non modifiable, deprecated) - C:\MySW\eclipseGit\plugins\org.junit_4.
- JUNIT_SRC_HOME (non modifiable, deprecated) - C:\MySW\eclipseGit\plugins\org.jur
- M2_REPO (non modifiable) - C:\Users\mtinius\.m2\repository

Create DV_HOME for Ant.

In Eclipse, select Window > Preferences > Ant > Runtime > Properties > Add Property

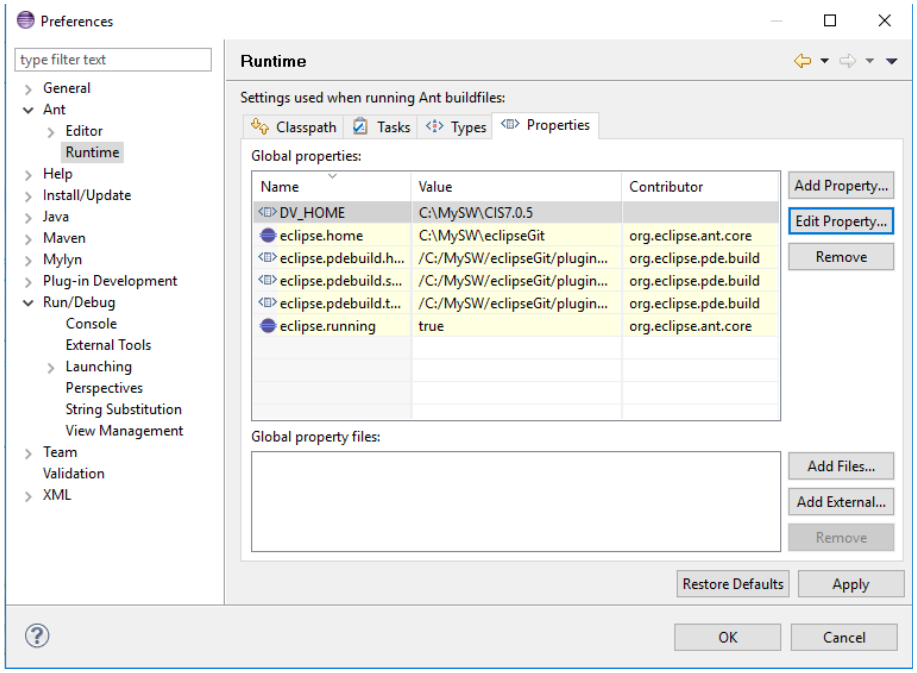
Enter Name: DV_HOME

Path: Locate your data virtualization home folder and type it or copy and paste it.



Name: DV_HOME

Value: C:\MySW\CIS7.0.5



Settings used when running Ant buildfiles:

Global properties:

Name	Value	Contributor
DV_HOME	C:\MySW\CIS7.0.5	
eclipse.home	C:\MySW\eclipseGit	org.eclipse.ant.core
eclipse.pdebuild.h...	/C:/MySW/eclipseGit/plugin...	org.eclipse.pde.build
eclipse.pdebuild.s...	/C:/MySW/eclipseGit/plugin...	org.eclipse.pde.build
eclipse.pdebuild.t...	/C:/MySW/eclipseGit/plugin...	org.eclipse.pde.build
eclipse.running	true	org.eclipse.ant.core

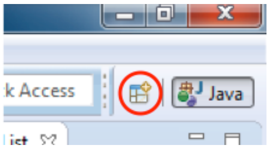
Checking Out the PDTool Project

PDTool is currently developed in Eclipse. These instructions are based on the Luna distribution of Eclipse (if not using this release then the following steps may need to be modified. For instance, Luna includes the Git client, whereas earlier releases of Eclipse do not.) If installing Eclipse for the first time, the "Eclipse Standard" (or "Eclipse IDE for Java Developers", if space is tight) should be used.

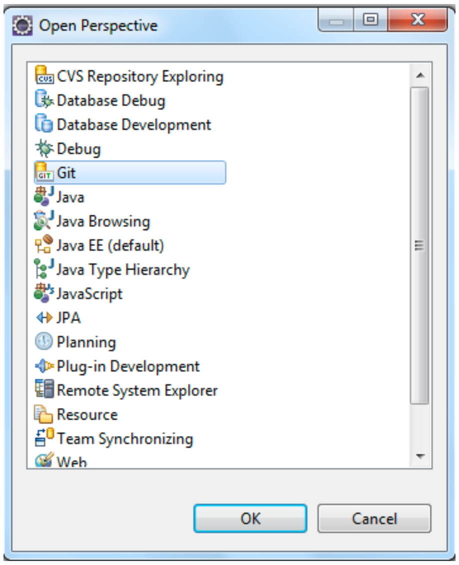
Clone the PDTool Git repository to your local machine

For the next steps, start Eclipse and make sure to apply any updates (Help -> Check for Updates).

1. Open the Git perspective using the "Open Perspective" panel in the upper right of the Eclipse screen:

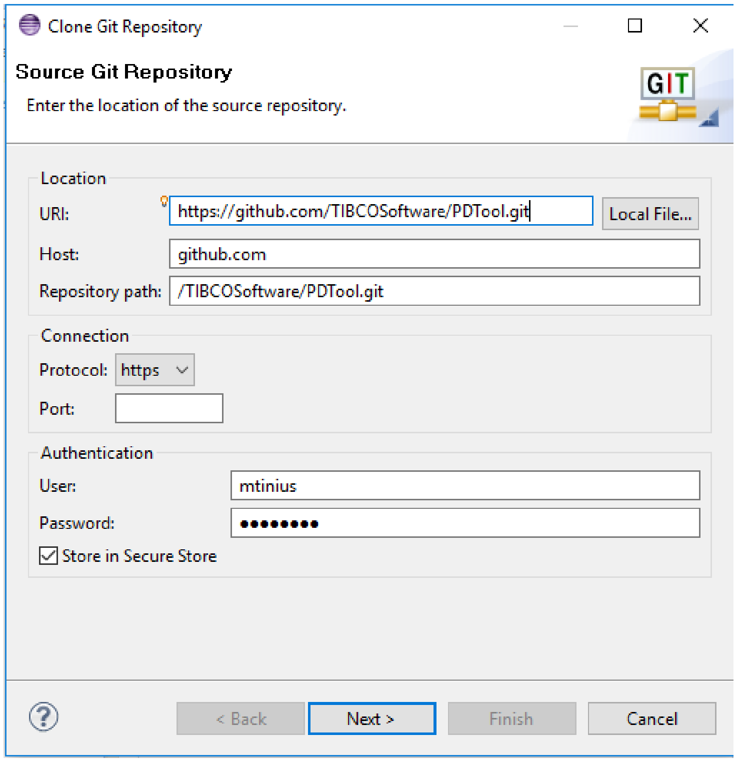


2. Choose the Git perspective:



3. In the "Git Repositories" panel, click the "Clone a Git repository" link. Alternatively, go to the "File" menu and select "New"->"Other…" Choose "Git"->"Git Repository" from the resulting dialog.

4. In the resulting dialog, paste the Git repository URL ***https://github.com/TIBCOSoftware/PDTool.git*** into the "URI" field. The "Host" and "Repository Path" fields should auto-populate. Enter your Git user name and password and click "Next >".



Clone Git Repository

Source Git Repository

Enter the location of the source repository.

Location

URI: https://github.com/TIBCOSoftware/PDTool.git

Host: github.com

Repository path: /TIBCOSoftware/PDTool.git

Connection

Protocol: https

Port:

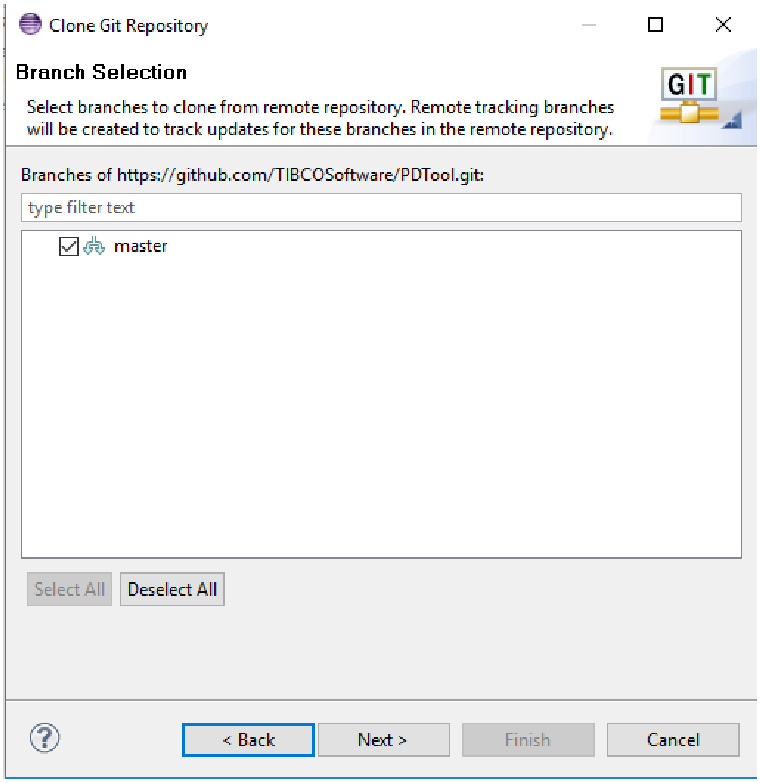
Authentication

User: mtinius

Password: ●●●●●●●●●

☑ Store in Secure Store

5. The "Branch Selection" screen should show one branch called "master". This should already be selected so click "Next >".



Clone Git Repository

Branch Selection

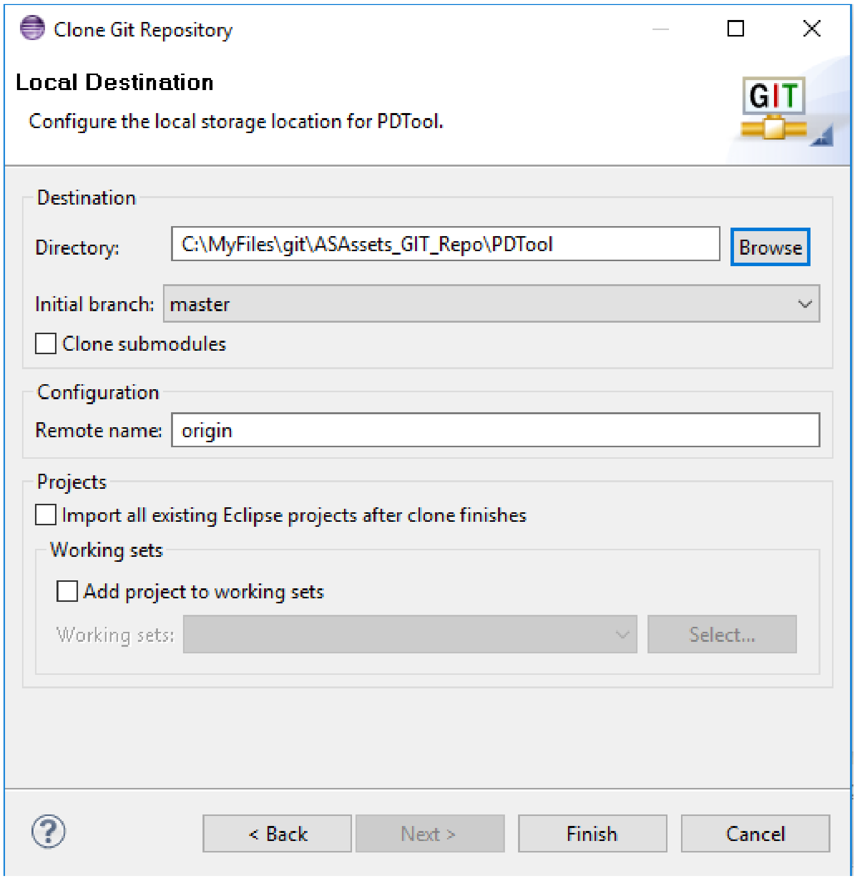
Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.

Branches of https://github.com/TIBCOSoftware/PDTool.git:

type filter text

☑ master

6. Choose the location where the downloaded source code should be stored. Click "Finish".



Clone Git Repository

Local Destination

Configure the local storage location for PDTool.

Destination

Directory: C:\MyFiles\git\ASAssets_GIT_Repo\PDTool

Initial branch: master

□ Clone submodules

Configuration

Remote name: origin

Projects

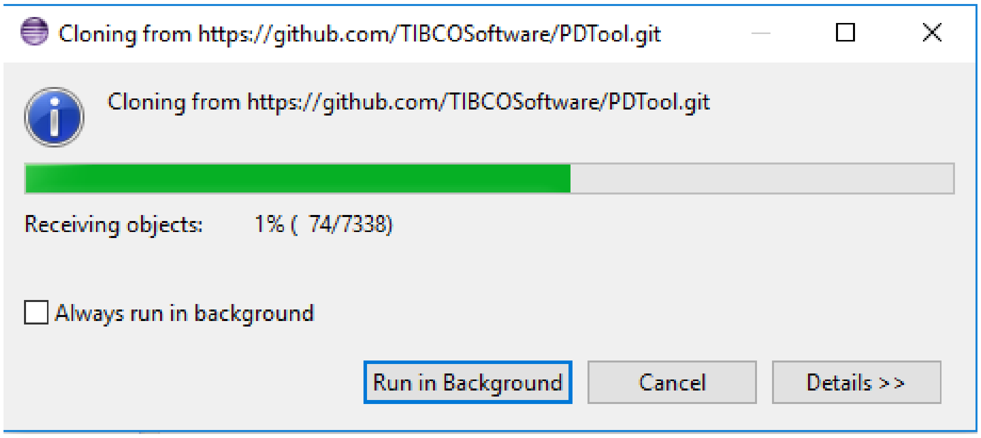
□ Import all existing Eclipse projects after clone finishes

Working sets

□ Add project to working sets

Working sets:

7. A progress dialog will appear indicating how far along the clone process is. Once done, the PDTool clone should then appear in your Git Repositories list.



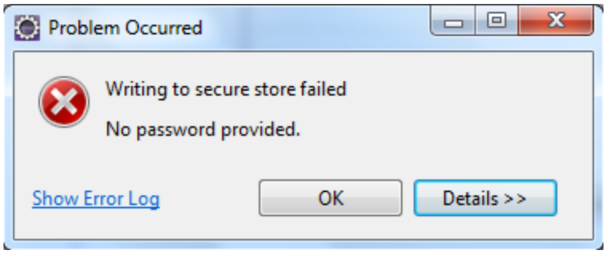
Cloning from https://github.com/TIBCOSoftware/PDTool.git

Cloning from https://github.com/TIBCOSoftware/PDTool.git

Receiving objects: 1% (74/7338)

□ Always run in background

8. You may get an error stating that your secure store password has not been set. This can be safely ignored but it's probably a good idea to go into the Eclipse preferences and set a password for your secure storage.



Writing to secure store failed

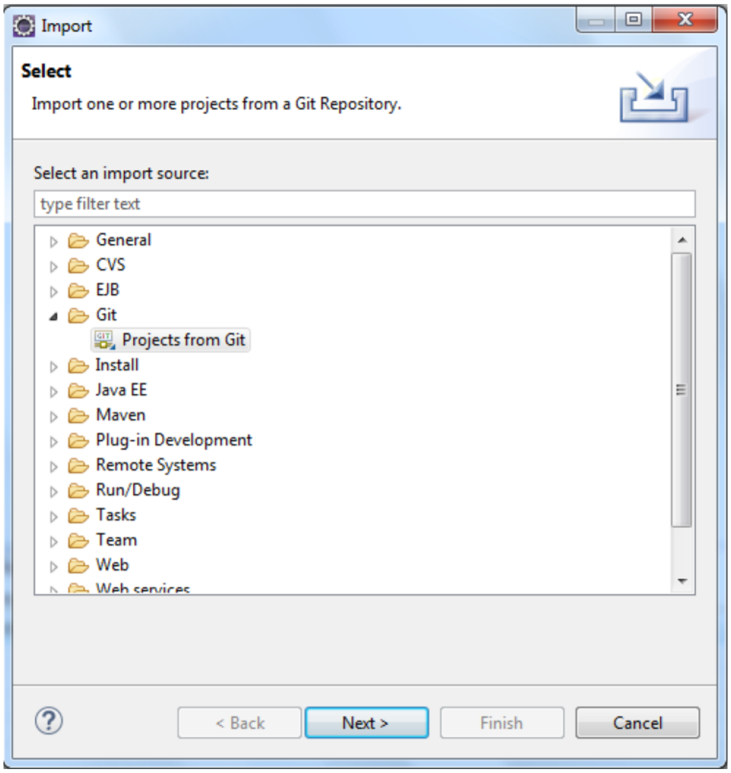
No password provided.

Create a General project from the Git repository

1. Open the Java perspective clicking the "Java" perspective button in the upper right of the Eclipse screen. Since the entire PDTool project is a java project, it comes into Eclipse as a Java project and is ready to start using as is.

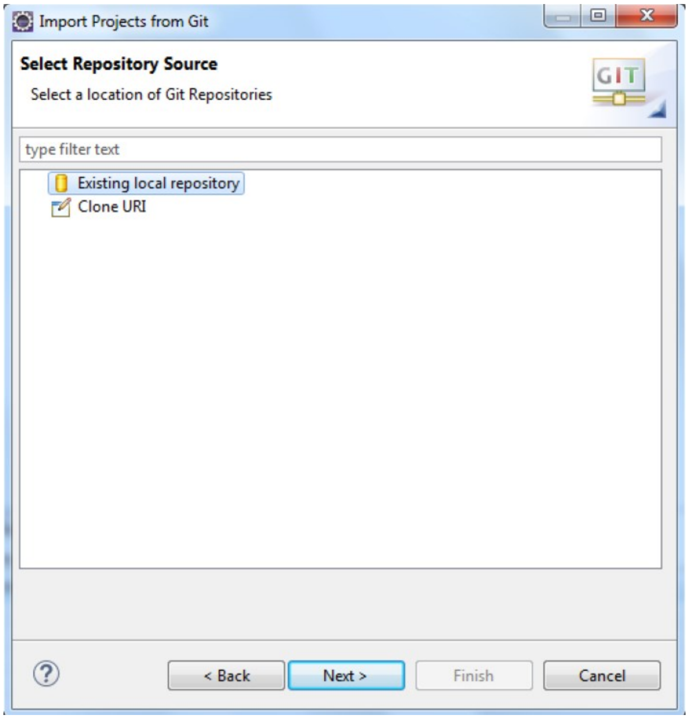
2. Right-click in the "Package Explorer" panel on the left side of Eclipse. Select "Import …"

3. Drill into "Git" and select "Projects from Git". Click "Next >".



Import one or more projects from a Git Repository.

4. In the next panel choose "Existing local repository". Click "Next >".



Import Projects from Git

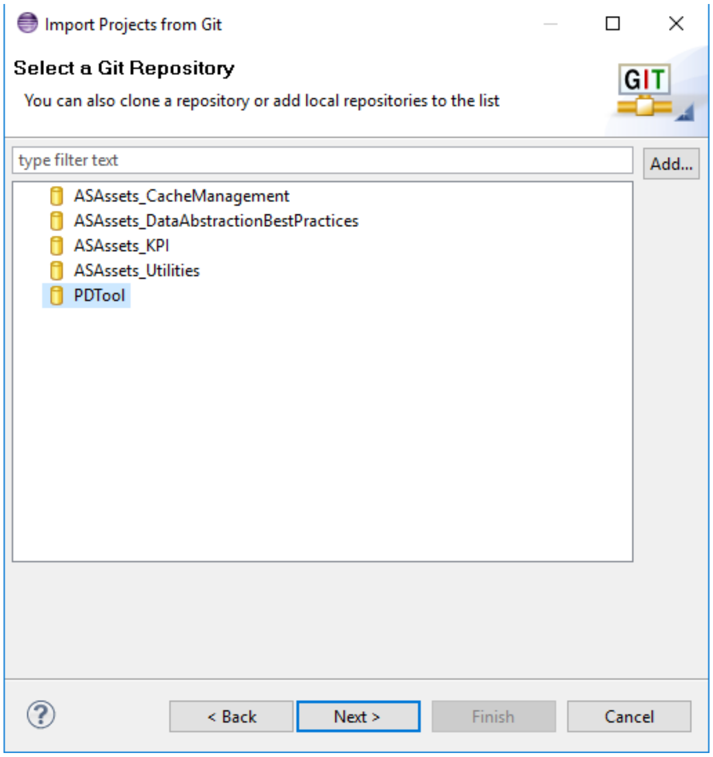
Select Repository Source

Select a location of Git Repositories

type filter text

- Existing local repository
- Clone URI

5. Choose the "PDTool" repository. Click "Next >".



Import Projects from Git

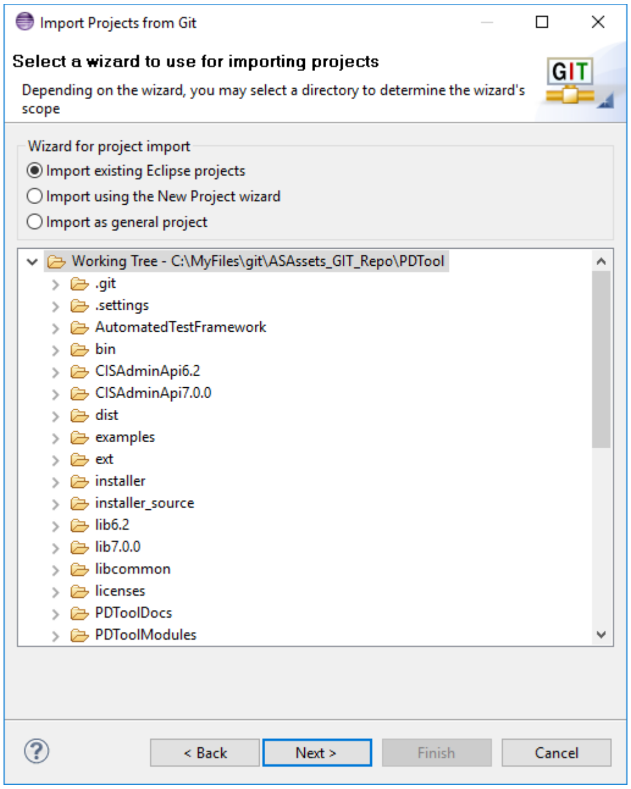
Select a Git Repository

You can also clone a repository or add local repositories to the list

type filter text

- ASAssets_CacheManagement
- ASAssets_DataAbstractionBestPractices
- ASAssets_KPI
- ASAssets_Utilities
- PDTool

6. Select "**Import existing Eclipse projects**" as the root folder to import. Click "Next >".



Import Projects from Git

Select a wizard to use for importing projects

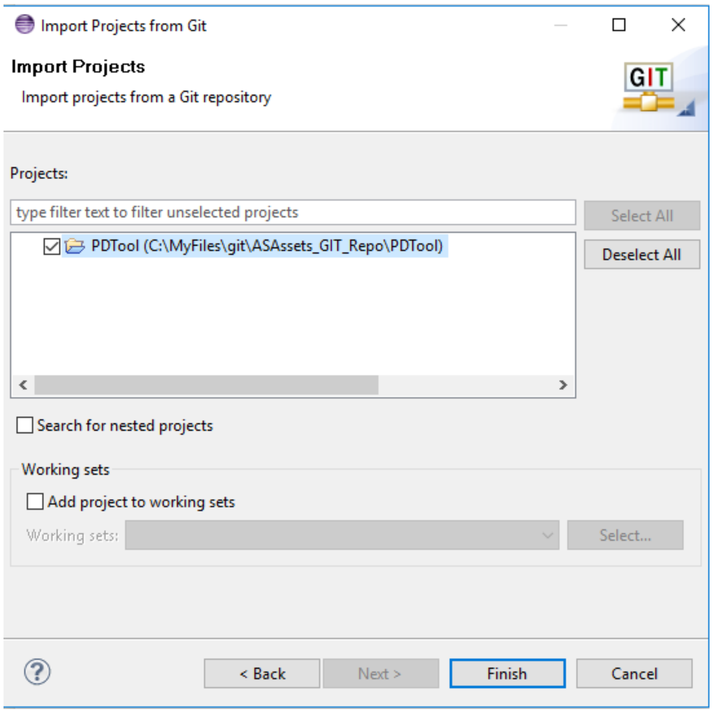
Depending on the wizard, you may select a directory to determine the wizard's scope

Wizard for project import

● Import existing Eclipse projects
○ Import using the New Project wizard
○ Import as general project

```
v Working Tree - C:\MyFiles\git\ASAssets_GIT_Repo\PDTool
  > .git
  > .settings
  > AutomatedTestFramework
  > bin
  > CISAdminApi6.2
  > CISAdminApi7.0.0
  > dist
  > examples
  > ext
  > installer
  > installer_source
  > lib6.2
  > lib7.0.0
  > libcommon
  > licenses
  > PDToolDocs
  > PDToolModules
```

7. Select only the "PDTool" project and click "Finish".



Import Projects from Git

Import Projects

Import projects from a Git repository

Projects:

type filter text to filter unselected projects

☑ PDTool (C:\MyFiles\git\ASAssets_GIT_Repo\PDTool)

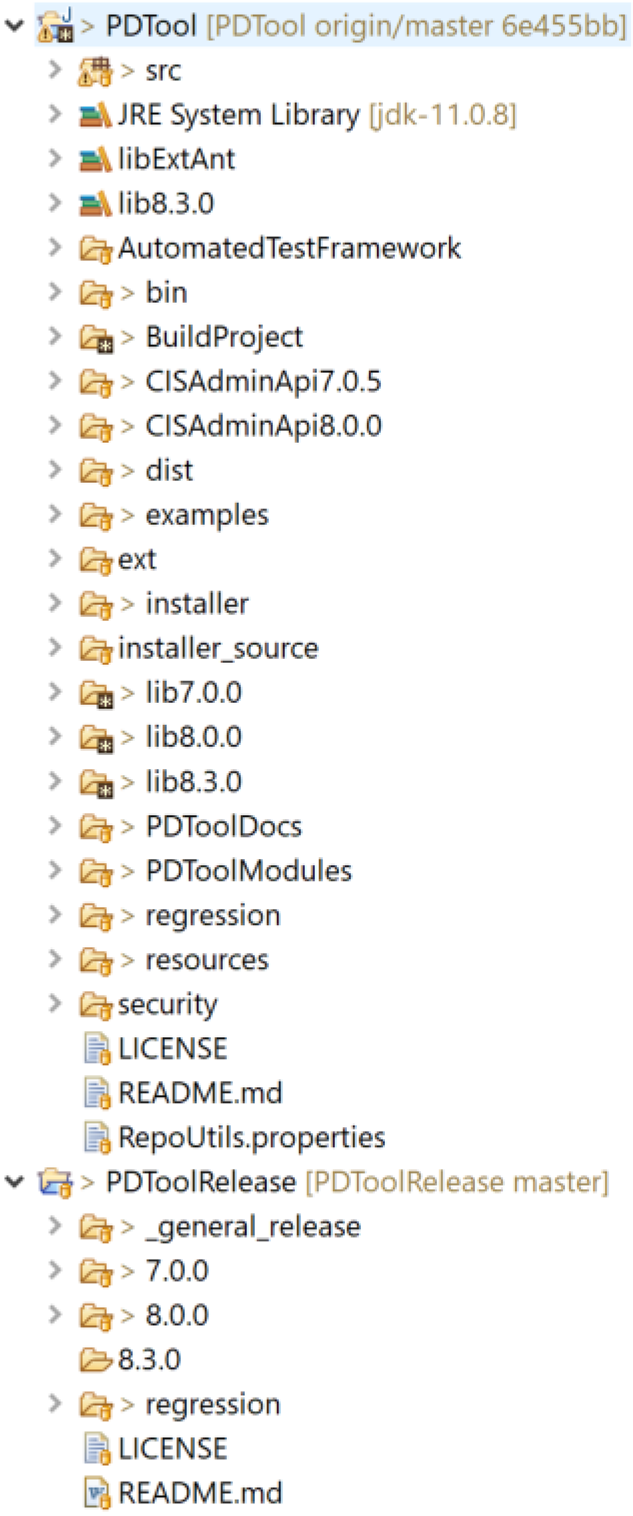
□ Search for nested projects

Working sets

□ Add project to working sets

Working sets:

8. The imported project:



```
PDTool [PDTool origin/master 6e455bb]
    src
    JRE System Library [jdk-11.0.8]
    libExtAnt
    lib8.3.0
    AutomatedTestFramework
    bin
    BuildProject
    CISAdminApi7.0.5
    CISAdminApi8.0.0
    dist
    examples
    ext
    installer
    installer_source
    lib7.0.0
    lib8.0.0
    lib8.3.0
    PDToolDocs
    PDToolModules
    regression
    resources
    security
    LICENSE
    README.md
    RepoUtils.properties
PDToolRelease [PDToolRelease master]
    _general_release
    7.0.0
    8.0.0
    8.3.0
    regression
    LICENSE
    README.md
```

Build the PDTool jar files

Build just the jar distribution file:

Build PDTool 7.0.0

1. Change the Java Build Path Library to lib7.0.0
    a. Select the current lib8.0.0 or lib8.3.0 and click "Remove Libary"
    b. "Add library" "User Libraries" Select lib7.0.0 "Finish" "Apply and Close"
2. Eclipse Menu "Project" "Clean" "PDTool" Click "Clean"
3. To build /dist/PDTool7.0.0.jar only, right-click on build7.0.0.xml select "Run As" "Ant Build"
4. To build the PDToolRelease,
    a. Refer to instructions in PDToolRelease/_general_release/README.txt
    b. Right-click on build_release7.0.0.xml to build the project

Build PDTool 8.0.0

1. Change the Java Build Path Library to lib8.0.0
    a. Select the current lib7.0.0 or lib8.3.0 and click "Remove Libary"
    b. "Add library" "User Libraries" Select lib8.0.0 "Finish" "Apply and Close"
2. Eclipse Menu "Project" "Clean" "PDTool" Click "Clean"
3. To build /dist/PDTool8.0.0.jar only, right-click on build8.0.0.xml select "Run As" "Ant Build"
4. To build the PDToolRelease,
    a. Refer to instructions in PDToolRelease/_general_release/README.txt
    b. Right-click on build_release8.0.0.xml to build the project

Build PDTool 8.3.0

1. Change the Java Build Path Library to lib8.3.0
    a. Select the current lib7.0.0 or lib8.0.0 and click "Remove Libary"
    b. "Add library" "User Libraries" Select lib8.3.0 "Finish" "Apply and Close"
2. Eclipse Menu "Project" "Clean" "PDTool" Click "Clean"
3. To build /dist/PDTool8.3.0.jar only, right-click on build8.3.0.xml select "Run As" "Ant Build"
4. To build the PDToolRelease,
    a. Refer to instructions in PDToolRelease/_general_release/README.txt
    b. Right-click on build_release8.3.0.xml to build the project

Build Regressions

1. Eclipse Menu "Project" "Clean" "PDTool" Click "Clean"
2. To build the PDToolRelease regression,
    a. Right-click on build_regression.xml to build the project. Select “Run As” “Ant Build”
    b. The zip files are created in the PDToolRelease project

Debugging PDTool (Java) Source Code

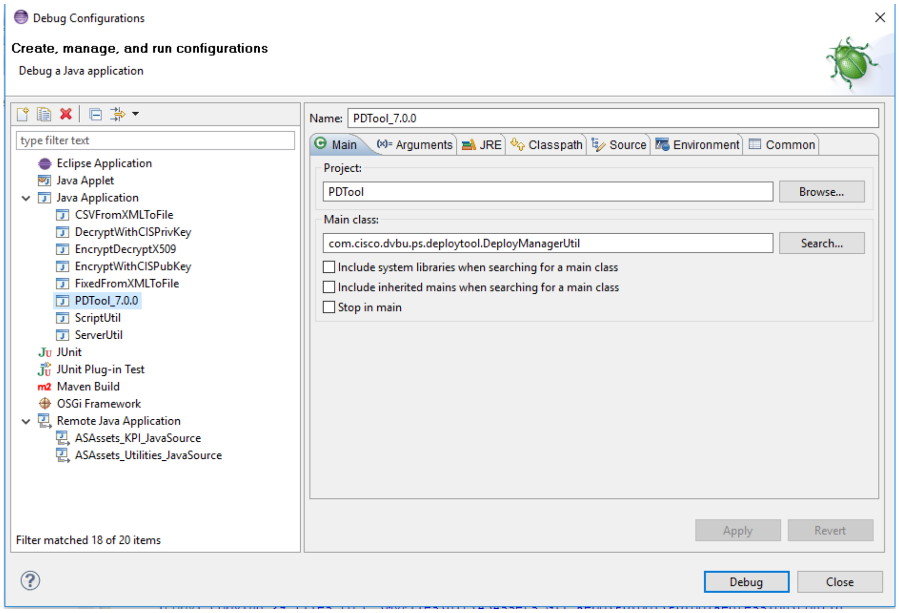
These steps discuss how you can debug the PDTool Java code.

1. Open your Eclipse workspace **PDTool** java project**.**

    a. Open the Debug Configurations and create a “Java Application” configuration for the PDTool Java Project

    b. Set the parameters as shown below for Host and Port.

    c. Use this when debugging and setting breakpoints.



Name: PDTool_7.0.0

Project: PDTool

Main class: com.cisco.dvbu.ps.deploytool.DeployManagerUtil

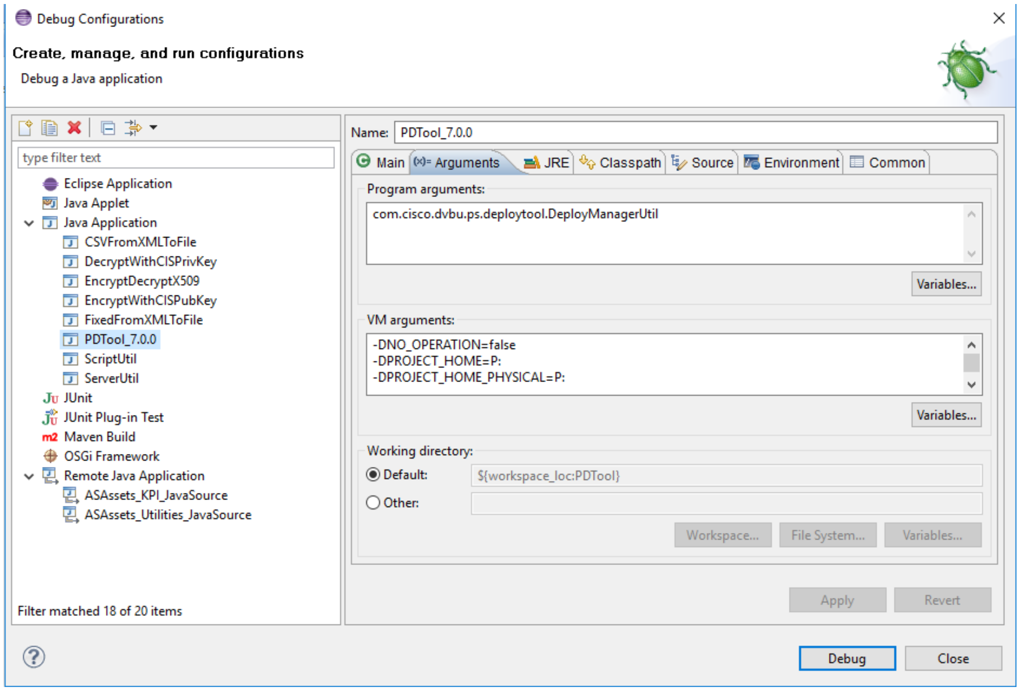
☐ Include system libraries when searching for a main class

☐ Include inherited mains when searching for a main class

☐ Stop in main

d. Set the arguments as follows:

--add-exports=java.base/sun.security.provider=ALL-UNNAMED --add-exports=java.security.jgss/sun.security.krb5=ALL-UNNAMED --add-opens=java.base/sun.nio.ch=ALL-UNNAMED --add-opens=java.base/jdk.internal.misc=ALL-UNNAMED --add-opens=java.base/java.nio=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=jdk.management/com.sun.management.internal=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.ldap=ALL-UNNAMED --add-exports=java.base/sun.nio.ch=ALL-UNNAMED --add-exports=java.base/jdk.internal.ref=ALL-UNNAMED --add-exports=java.base/sun.net.www.protocol.file=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED
-DNO_OPERATION=false
-DPROJECT_HOME=C:\PDTool\PDTool8.3.0_20201020\PDTool
-DPROJECT_HOME_PHYSICAL=C:\PDTool\PDTool8.3.0_20201020\PDTool
-DCONFIG_PROPERTY_FILE=deploy_GIT_9810.properties
-Dlog4j.configuration="file:resources\config\log4j.properties"
-Dcom.tibco.ps.configroot=resources\config
-Djavax.net.ssl.trustStore="C:\PDTool\PDTool8.3.0_20201020\PDTool\security\cis_studio_truststore.jks"
-Djavax.net.ssl.trustStorePassword=changeit
-DSTUDENTID=01
NOTE: For 7.0 use this:
-Djava.endorsed.dirs=lib7.0.0\td\endorsed
Note: For 8.0 use this
-Djava.endorsed.dirs=lib8.0.0\td\endorsed
Note: For 8.3 do not include endorsed dirs



Name: PDTool_7.0.0

Program arguments:

```
com.cisco.dvbu.ps.deploytool.DeployManagerUtil
```

VM arguments:

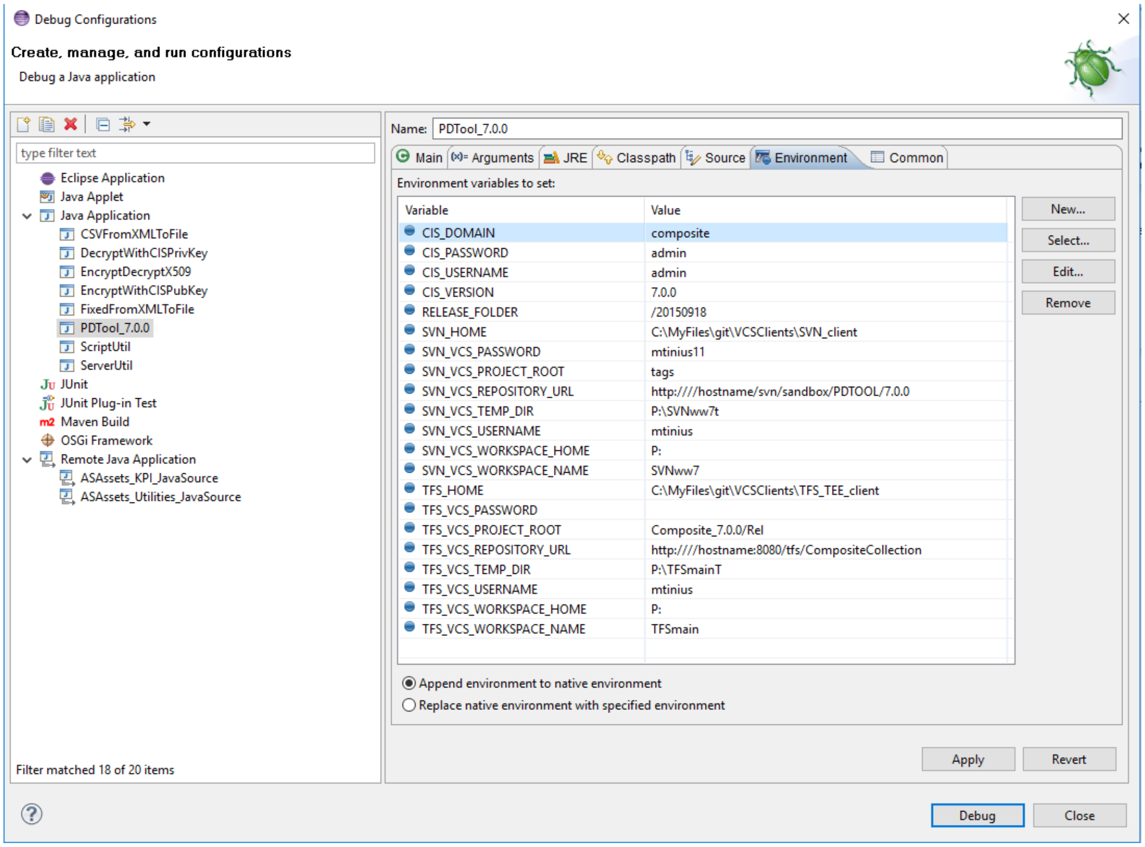
```
-DNO_OPERATION=false
-DPROJECT_HOME=P:
-DPROJECT_HOME_PHYSICAL=P:
```

Working directory:

● Default: ${workspace_loc:PDTool}

○ Other:

e. Set the Environment variables as follows:



Name: PDTool_7.0.0

Environment variables to set:

Variable	Value
CIS_DOMAIN	composite
CIS_PASSWORD	admin
CIS_USERNAME	admin
CIS_VERSION	7.0.0
RELEASE_FOLDER	/20150918
SVN_HOME	C:\MyFiles\git\VCSClients\SVN_client
SVN_VCS_PASSWORD	mtinius11
SVN_VCS_PROJECT_ROOT	tags
SVN_VCS_REPOSITORY_URL	http:////hostname/svn/sandbox/PDTOOL/7.0.0
SVN_VCS_TEMP_DIR	P:\SVNww7t
SVN_VCS_USERNAME	mtinius
SVN_VCS_WORKSPACE_HOME	P:
SVN_VCS_WORKSPACE_NAME	SVNww7
TFS_HOME	C:\MyFiles\git\VCSClients\TFS_TEE_client
TFS_VCS_PASSWORD	
TFS_VCS_PROJECT_ROOT	Composite_7.0.0/Rel
TFS_VCS_REPOSITORY_URL	http:////hostname:8080/tfs/CompositeCollection
TFS_VCS_TEMP_DIR	P:\TFSmainT
TFS_VCS_USERNAME	mtinius
TFS_VCS_WORKSPACE_HOME	P:
TFS_VCS_WORKSPACE_NAME	TFSmain

● Append environment to native environment

○ Replace native environment with specified environment

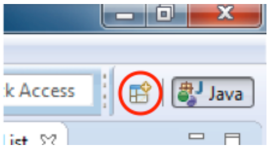
Checking Out the PDToolRelease Project

PDToolRelease is used for the synchronization between the PDTool Java project and the PDToolRelease project and Github master repository. This is where PDTool build releases are stored and then checked in to Github.

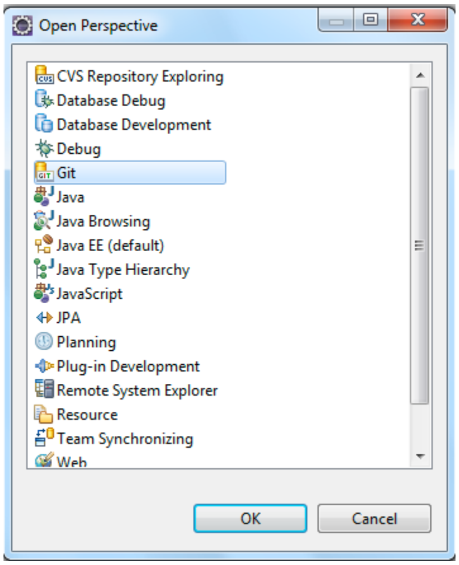
Clone the PDToolRelease Git repository to your local machine

For the next steps, start Eclipse and make sure to apply any updates (Help -> Check for Updates).

9. Open the Git perspective using the "Open Perspective" panel in the upper right of the Eclipse screen:

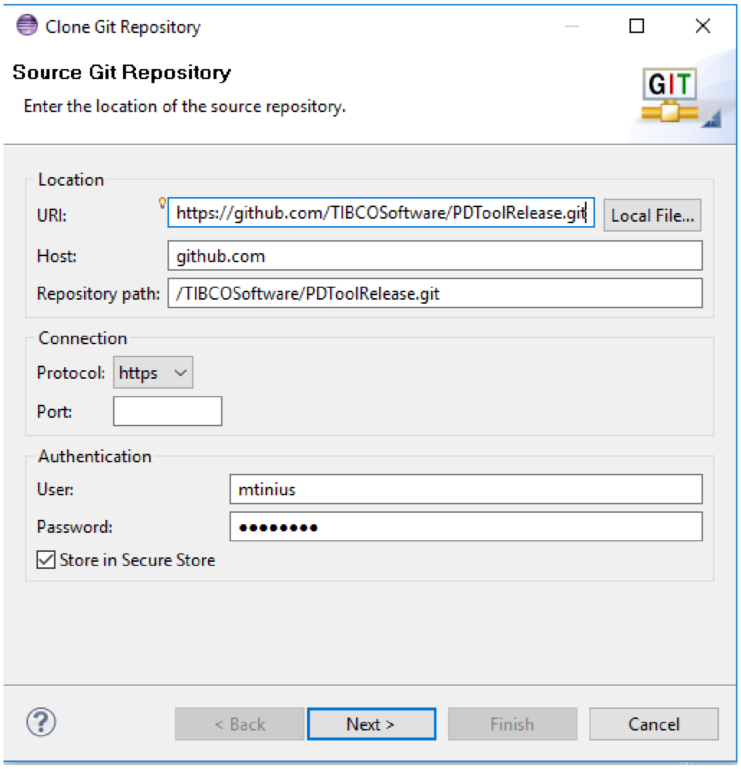


10. Choose the Git perspective:



11. In the "Git Repositories" panel, click the "Clone a Git repository" link. Alternatively, go to the "File" menu and select "New"->"Other…" Choose "Git"->"Git Repository" from the resulting dialog.

12. In the resulting dialog, paste the Git repository URL ***https://github.com/TIBCOSoftware/PDToolRelease.git*** into the "URI" field. The "Host" and "Repository Path" fields should auto-populate. Enter your Git user name and password and click "Next >".



Clone Git Repository

Source Git Repository

Enter the location of the source repository.

Location

URI: https://github.com/TIBCOSoftware/PDToolRelease.git

Host: github.com

Repository path: /TIBCOSoftware/PDToolRelease.git

Connection

Protocol: https

Port:

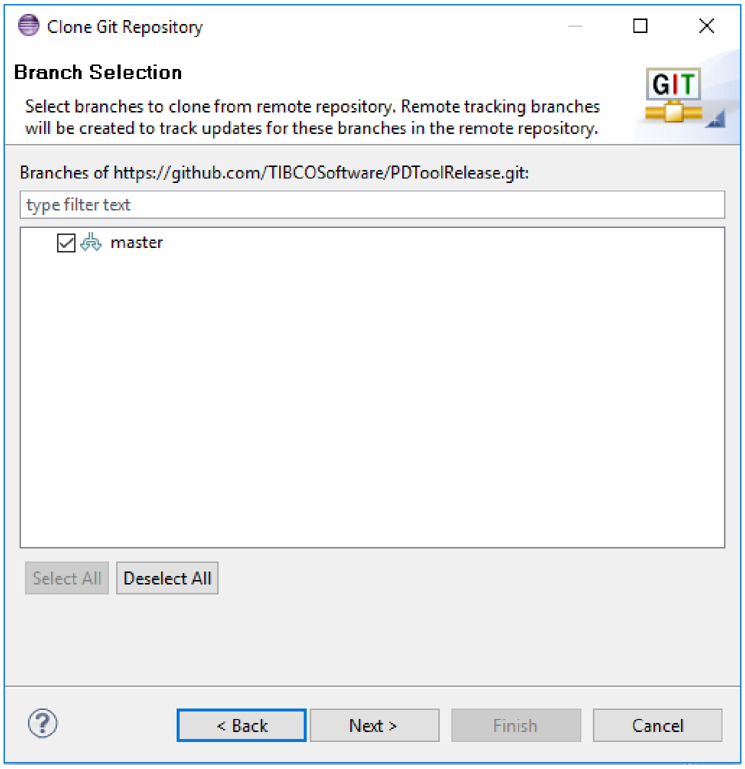
Authentication

User: mtinius

Password: ●●●●●●●●●

☑ Store in Secure Store

13. The "Branch Selection" screen should show one branch called "master". This should already be selected so click "Next >".



Clone Git Repository

Branch Selection

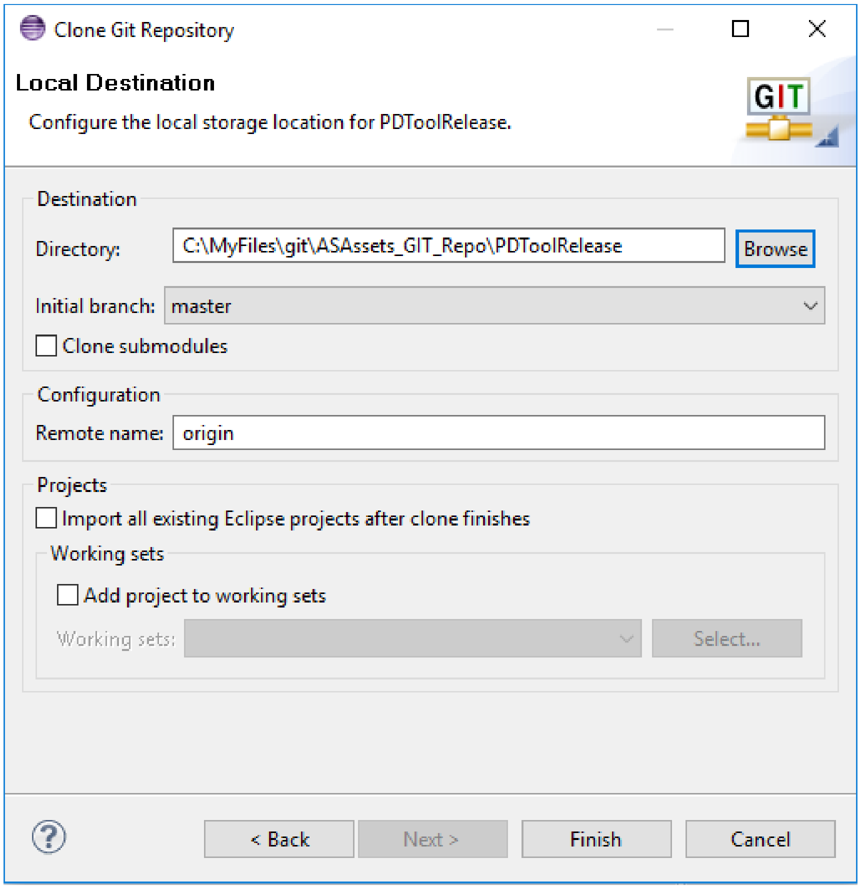
Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.

Branches of https://github.com/TIBCOSoftware/PDToolRelease.git:

type filter text

☑ master

14. Choose the location where the downloaded source code should be stored. Click "Finish".



Clone Git Repository

Local Destination

Configure the local storage location for PDToolRelease.

Destination

Directory: C:\MyFiles\git\ASAssets_GIT_Repo\PDToolRelease

Initial branch: master

□ Clone submodules

Configuration

Remote name: origin

Projects

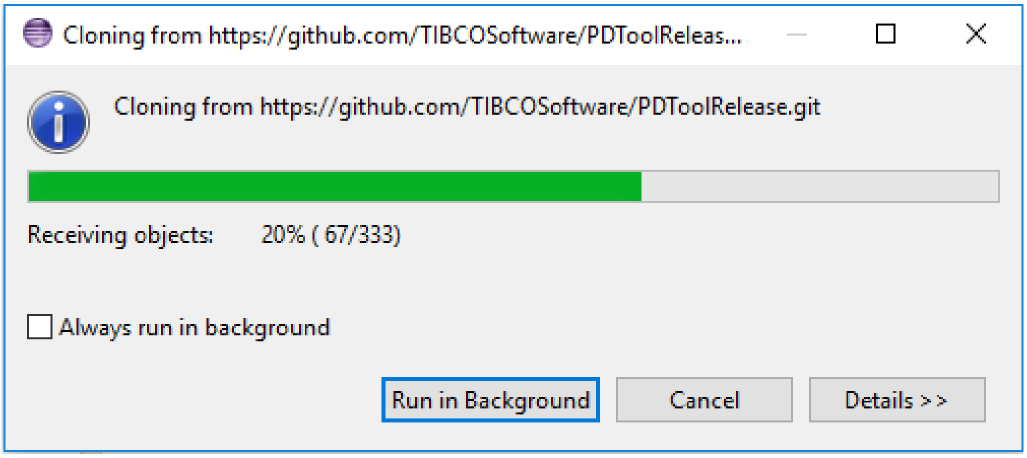
□ Import all existing Eclipse projects after clone finishes

Working sets

□ Add project to working sets

Working sets:

15. A progress dialog will appear indicating how far along the clone process is. Once done, the PDToolRelease clone should then appear in your Git Repositories list.



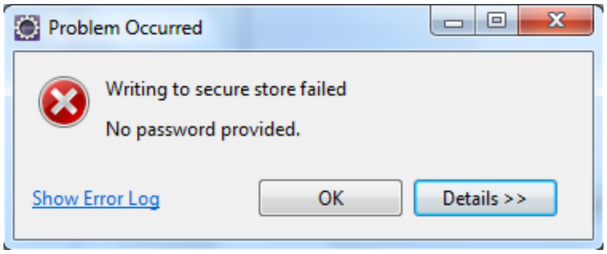
Cloning from https://github.com/TIBCOSoftware/PDToolReleas...

Cloning from https://github.com/TIBCOSoftware/PDToolRelease.git

Receiving objects: 20% (67/333)

□ Always run in background

16. You may get an error stating that your secure store password has not been set. This can be safely ignored but it's probably a good idea to go into the Eclipse preferences and set a password for your secure storage.


Problem Occurred

Writing to secure store failed

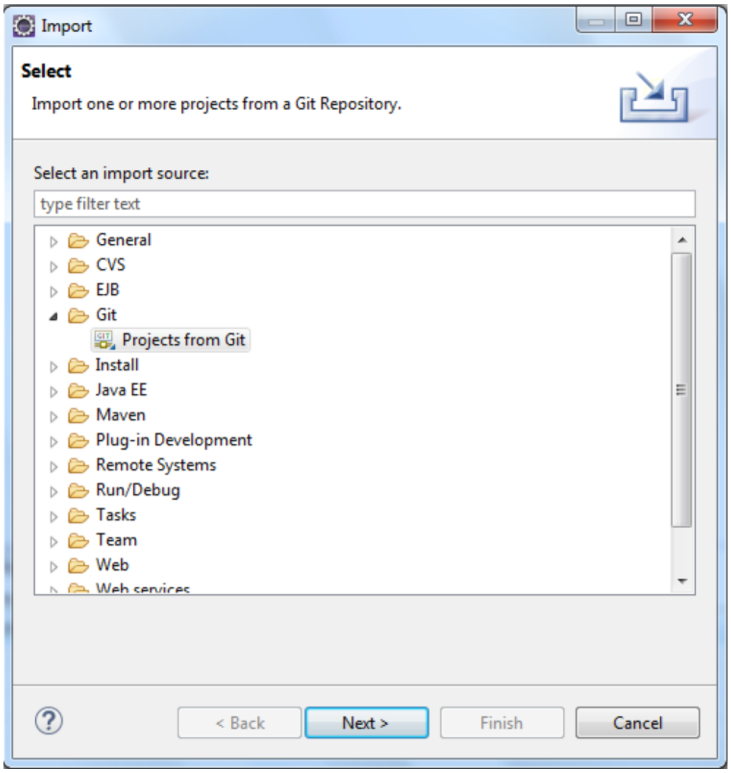
No password provided.

Create a General project from the Git repository

9. Open the Java perspective clicking the "Java" perspective button in the upper right of the Eclipse screen.

10. Right-click in the "Package Explorer" panel on the left side of Eclipse. Select "Import …"

11. Drill into "Git" and select "Projects from Git". Click "Next >".


Import

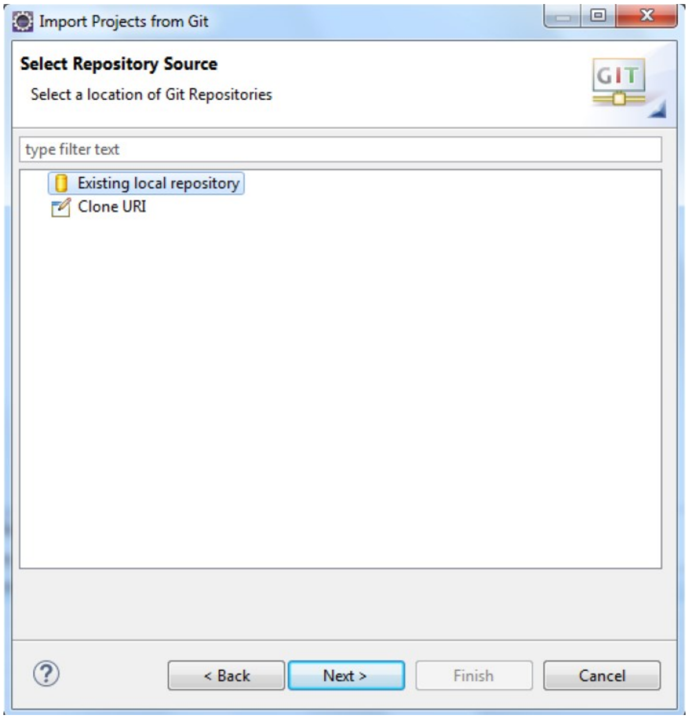
Select

Import one or more projects from a Git Repository.

Select an import source:

type filter text

12. In the next panel choose "Existing local repository". Click "Next >".



Import Projects from Git

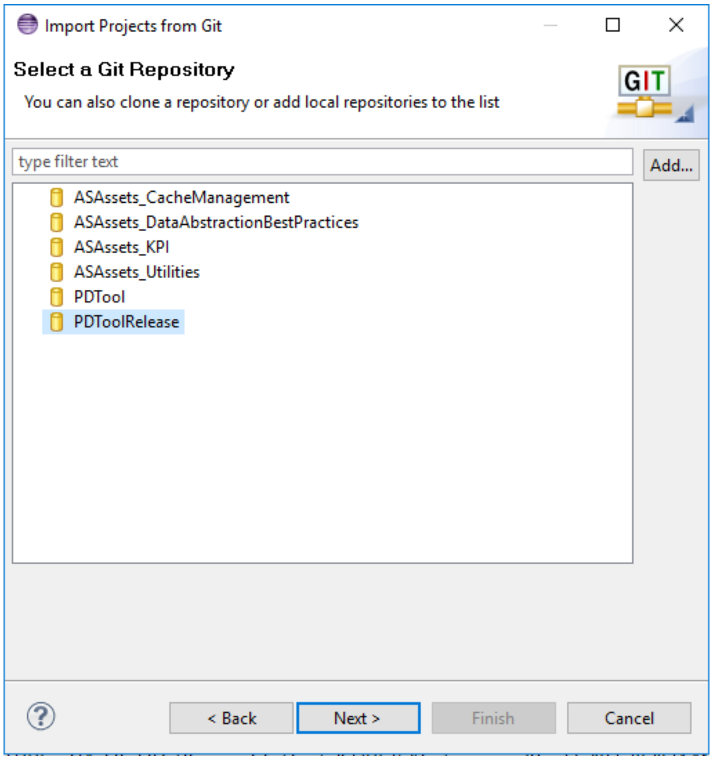
Select Repository Source

Select a location of Git Repositories

type filter text

- Existing local repository
- Clone URI

13. Choose the "PDToolRelease" repository. Click "Next >".



Import Projects from Git

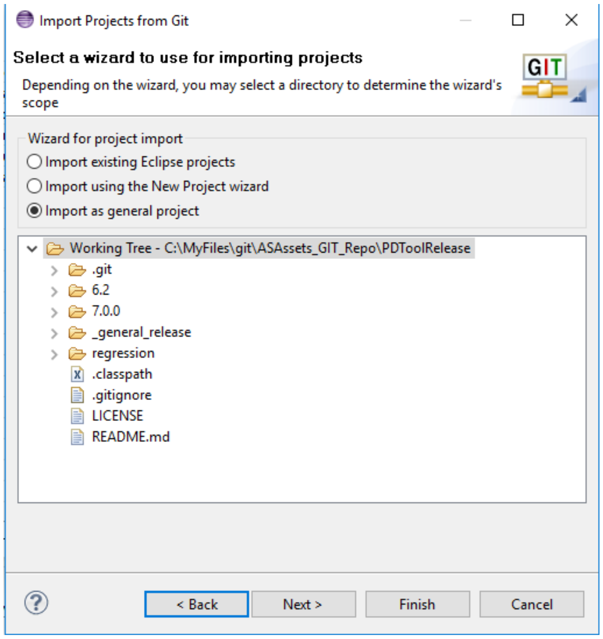
Select a Git Repository

You can also clone a repository or add local repositories to the list

type filter text

- ASAssets_CacheManagement
- ASAssets_DataAbstractionBestPractices
- ASAssets_KPI
- ASAssets_Utilities
- PDTool
- PDToolRelease

14. Select "**Import as general project**" as the root folder to import. Click "Next >".



Import Projects from Git

Select a wizard to use for importing projects

Depending on the wizard, you may select a directory to determine the wizard's scope

Wizard for project import

○ Import existing Eclipse projects

○ Import using the New Project wizard

● Import as general project

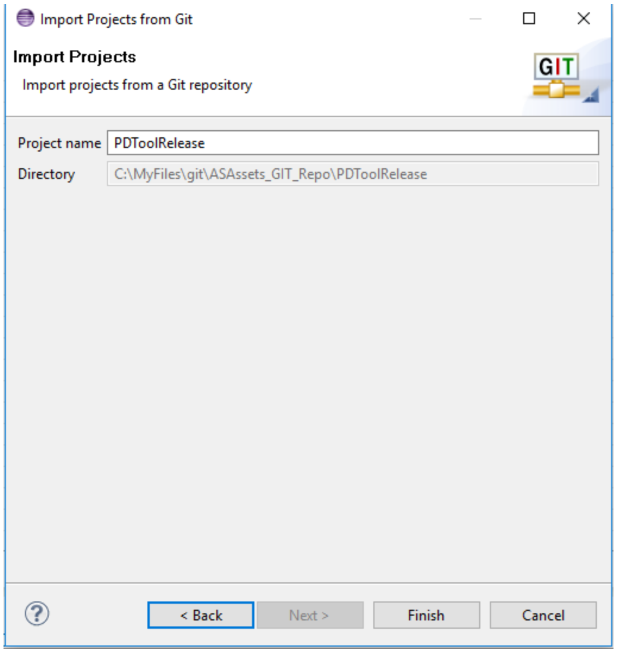
Working Tree - C:\MyFiles\git\ASAssets_GIT_Repo\PDToolRelease

15. Handling errors.

    a. If an error message is shown that the ".project" already exists, then remove the .project file from the "Git" perspective and the PDToolRelease directory.

    b. Try again.

16. Select only the "PDToolRelease" project and click "Finish".



Import Projects from Git

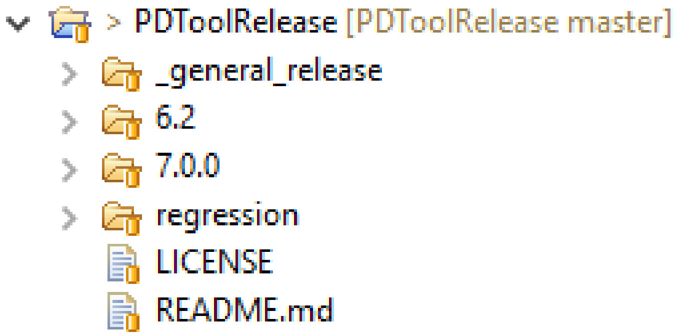
Import Projects

Import projects from a Git repository

Project name: PDToolRelease

Directory: C:\MyFiles\git\ASAssets_GIT_Repo\PDToolRelease

17. The imported project:



```
> PDToolRelease [PDToolRelease master]
    _general_release
    6.2
    7.0.0
    regression
    LICENSE
    README.md
```