HCL Digital Experience

HDX-DEV-300
DX Setup Local Java Development Environment Lab

Edition: July 2024

Herbert Hilhorst – herbert.hilhorst@hcl-software.com

Thorsten Reinheimer - thorsten.reinheimer@hcl-software.com

Table of Contents

Author(s)

This document was created by the following Subject Matter Experts:

Herbert Hilhorst
Company: HCLSoftware

Bio

Herbert Hilhorst is an HCL Digital Experience (DX) Technical Advisor at HCLSoftware.

Contact: herbert.hilhorst@hcl-software.com

Thorsten Reinheimer
Company: HCLSoftware

Bio

Thorsten Reinheimer is an HCL Digital Experience (DX) Technical Lead at HCLSoftware.

Contact: thorsten.reinheimer@hcl-software.com

Introduction

This hands-on lab gets you started on Java development for the HCL Digital Experience (DX) platform. This is used for several more advanced topics, like plugins and Java portlets development.

You will learn how to set this up a free developer tools (IDE) Microsoft Visual Studio Code, and optionally with Eclipse and IBM Rational Application Developer. You will prepare the different development environments to work easily with DX Java artifacts for your DX server.

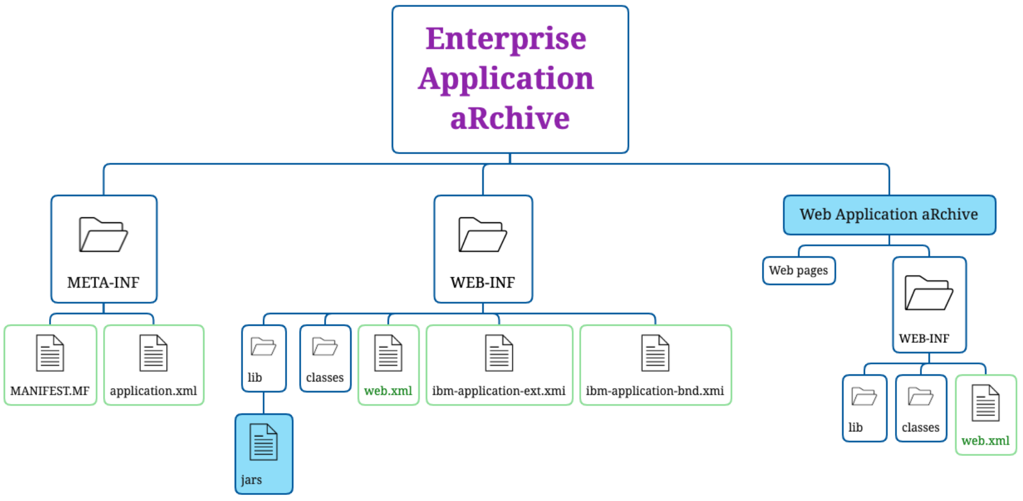
Prerequisites

1. Completion of the HDX-INTRO course as this gives you access to your own DX instance on HCL SoFy
2. Completion of HDX-DEV-100 as this helps you setting up the DXClient and deploy DX standalone as a Docker-Compose instance
3. Access to a DX server, remote or local (Docker-Compose or traditional installation). Here are instructions for a traditional installation of DX (not recommended, as hard to install and update): https://support.hcltechsw.com/csm?id=kb_article&sysparm_article=KB0079596
4. Download and extract the DX Portlet Development Utilities from https://github.com/HCL-TECH-SOFTWARE/dx-portlet-development-utilities

Lab Overview

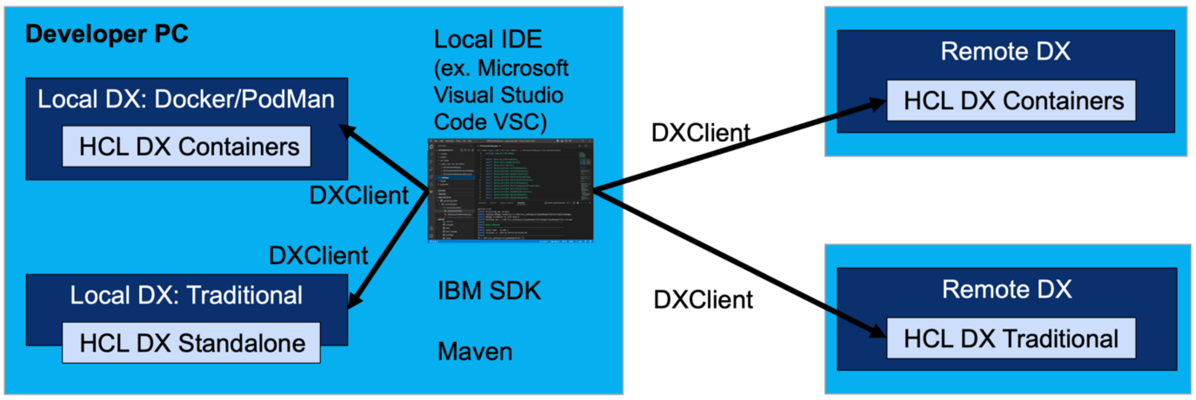
In this lab, you will explore how to set up your local development environment to build Java artifacts for HCL Digital Experience. These artifacts may be Java Portlets, plugins, themes, etc. They are typically deployed in three different ways: JAR, WAR and EAR. All of them are created using Zip/Jar compression. However, they are intended for different purposes:

- **JAR** (Extension called Java ARchive) – Supports Java and contains all your .class files and other resources like xml descriptors and other kind of files
- **WAR** (Extension called Web ARchive) – Supports servlet and JSP APIs and contains web-based resources, such as images, HTML, property files and compiled Java code
- **EAR** (Extension called Enterprise Application aRchive files) – Supports Java EE API and contains other Java EE archives, such as WAR, RAR, EJB-JAR and JAR files



There are many ways to develop locally with Java. In this lab, you will set up your local development environment using Maven to simplify creating different artifacts that later can be deployed on a DX server. Detailed steps will be discussed to connect to a DX server remotely or locally using different developer tools (IDEs).

Overview:



You will first learn how to set this up using Visual Studio Code, which is a free, well-known and popular developer tool (IDE). You may also optionally learn how to set it up using Eclipse or IBM Rational Application Developer (RAD). Eclipse is a powerful IDE with a lot of extensions and plugins. For details, please check: https://www.eclipse.org. Please note that IBM Rational Application Developer requires an official license to develop in a professional way. A trail version can be used for 30-60 days depending on the features that will be used.

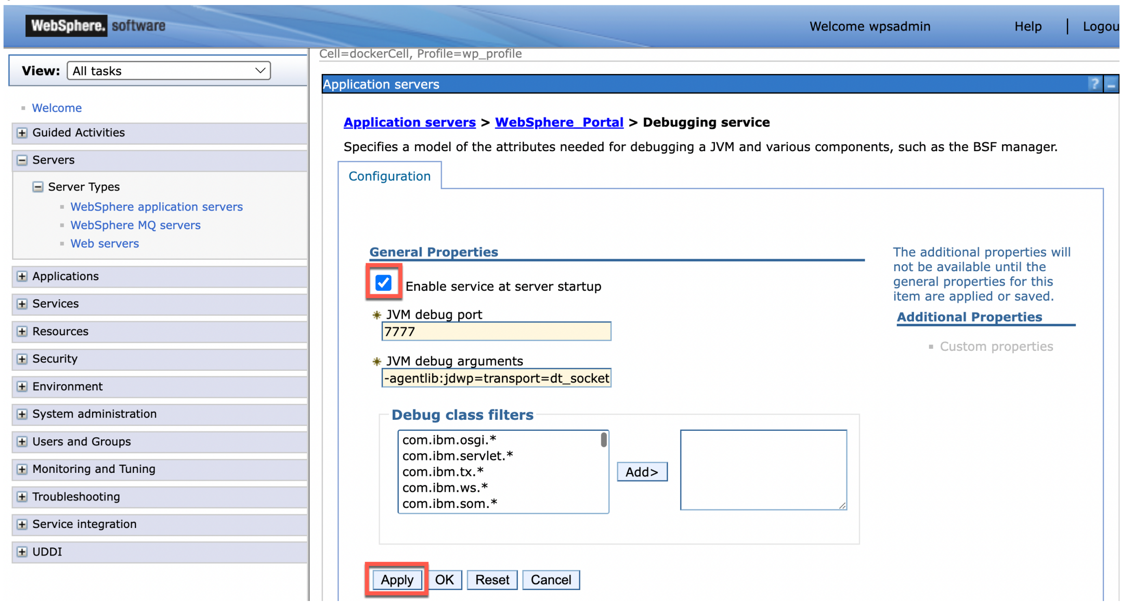
This lab covers three parts:

Part 1: DX Java Development with Visual Studio Code

In this part, you will set up your local development environment to use the right (IBM) Java SDK, Maven to simplify creating different artifacts, DXClient to deploy them to your DX server, and have it work with Visual Studio Code.

Part 2: Setup DX Server for Debugging

As you want to be able to debug your Java developments, you will enable debugging on your DX server.



Cell=dockerCell, Profile=wp_profile

Application servers > WebSphere_Portal > Debugging service

Specifies a model of the attributes needed for debugging a JVM and various components, such as the BSF manager.

General Properties

[✓] Enable service at server startup

JVM debug port: 7777

JVM debug arguments: -agentlib:jdwp=transport=dt_socket

Debug class filters:

com.ibm.osgi.*
com.ibm.servlet.*
com.ibm.tx.*
com.ibm.ws.*
com.ibm.som.*

The additional properties will not be available until the general properties for this item are applied or saved.

Optional Part 3: DX Java Development with Eclipse IDE for Java EE Developers

Optionally, you may set up and configure the Eclipse IDE to work with Java and combine it with Maven to make it simple to create and deploy the different artifacts for DX on a local or remote DX server.

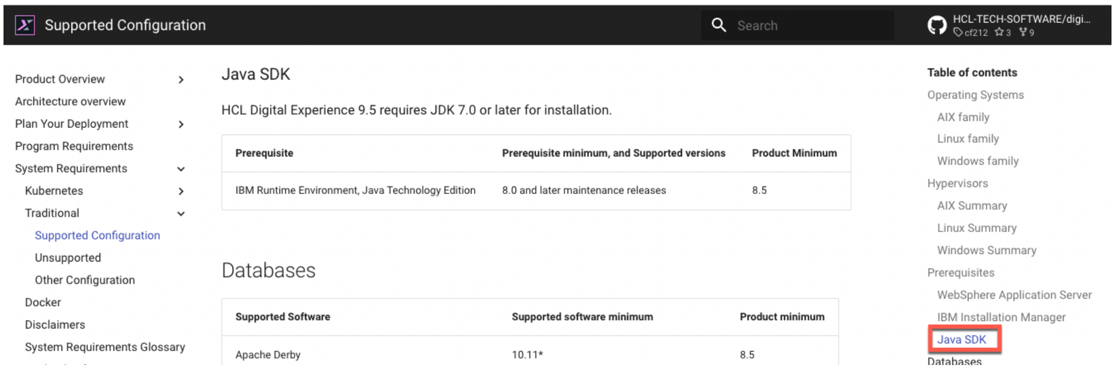
Optional Part 4: DX Java Development with IBM Rational Application Developer

And optionally, use IBM Rational Application Developer to develop with Java for DX.

Part 1: DX Java Development with Visual Code Studio and Maven

In this part, you will set up your local development environment to use the right (IBM) Java SDK, Maven to simplify creating different artifacts, DXClient to deploy them to your DX server, and have it work with Visual Studio Code.

1. First ensure to have IBM Java SDK installed. This allows you to compile and run your Java code and to build your java packages. Find the supported version that matches your DX server version runtime, as you may find here: https://opensource.hcltechsw.com/digital-experience/latest/get_started/system_requirements/traditional/supported_config/.



Java SDK

HCL Digital Experience 9.5 requires JDK 7.0 or later for installation.

Prerequisite	Prerequisite minimum, and Supported versions	Product Minimum
IBM Runtime Environment, Java Technology Edition	8.0 and later maintenance releases	8.5

Databases

Supported Software	Supported software minimum	Product minimum
Apache Derby	10.11*	8.5

2. You may get it from IBM's website, e.g. for V8.0: https://www.ibm.com/support/pages/java-sdk-downloads-version-80. Download the version that matches your OS install it locally using the instructions provided. Make sure that the system environment variables JAVA_HOME and CLASSPATH are set to ensure that by default the correct Java version will be used:

```
JAVA_HOME=<point to the directory of your Java location>
For example: "/opt/IBM/Java/8.0/jre" or "C:\IBM\JAVA\8.0\jre"
CLASSPATH=<point to the location of the rt.jar file>
For example: "/opt/IBM/Java/8.0/jre/lib/rt.jar" or "C:\IBM\JAVA\8.0\jre\rt.jar"
```

3. To check if it works well, open a shell or Command Line and test with command:

```
java-version
```

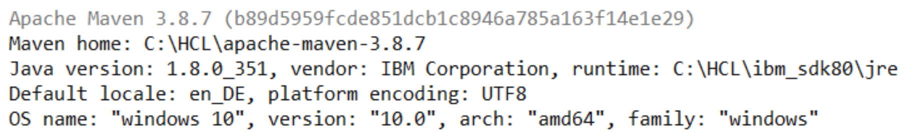
This should give something like:

```
java version "1.8.0_351"
Java(TM) SE Runtime Environment (build 8.0.7.20 - pwa6480sr7fp20-20221020_01(SR7 FP20))
IBM J9 VM (build 2.9, JRE 1.8.0 Windows 10 amd64-64-Bit Compressed References 20220929_37824 (JIT enabled, AOT enabled)
OpenJ9   - 02180fe
OMR      - 48fc32a
IBM      - bf759bf)
JCL - 20220922_01 based on Oracle jdk8u351-b10
```

4. Then install Apache Maven. Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information. You will use Maven archetypes that provide project templates to develop the Java code. You can find more details about the Maven archetypes here: https://maven.apache.org/guides/introduction/introduction-to-archetypes.html. You may find more information on https://maven.apache.org/. You will use Maven to manage your development more easily. Download Maven using: https://maven.apache.org/download.cgi and install it locally using the instructions provided (installation instructions are also provided in the README.txt in the downloaded zip file): https://maven.apache.org/install.html. Learn more on Maven using: https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html. Then confirm the installation with the command line:

```
mvn -v
```

Which should give you the version number and more details on your Maven installation, like:

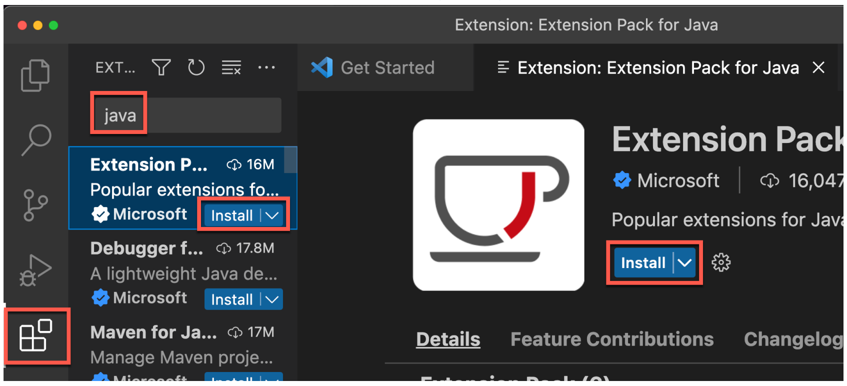


```
Apache Maven 3.8.7 (b89d5959fcde851dcb1c8946a785a163f14e1e29)
Maven home: C:\HCL\apache-maven-3.8.7
Java version: 1.8.0_351, vendor: IBM Corporation, runtime: C:\HCL\ibm_sdk80\jre
Default locale: en_DE, platform encoding: UTF8
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

5. Install Microsoft Visual Studio Code (VSC). You may use your favorite code editor to develop with Java. In this part, you will find instructions using VSC, a free open-source code editor. You may find details on this editor in this link: https://code.visualstudio.com/, download it using https://code.visualstudio.com/download and then set it up on your macOS, Linux or Windows using https://code.visualstudio.com/docs/setup/setup-overview.

6. As soon as Microsoft Visual Studio Code is installed, start the IDE. In the next steps install the following extensions that allows you to work with Java artifacts easily:
   - Language Support for Java - Java Linting, Intellisense, formatting, refactoring, Maven/Gradle support and more...
   - Debugger for Java - A lightweight Java debugger
   - Test Runner for Java - Run and debug JUnit or TestNG test cases.
   - Maven for Java - Manage Maven projects, execute goals, generate project from archetype, improve user experience for Java developers.
   - Project Manager for Java - Manage Java projects, referenced libraries, resource files, packages, classes, and class members
   - Visual Studio IntelliCode - AI-assisted Development and Completion list ranked by AI

   More details are available under: https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-java-pack. You can find the extensions by clicking in the Extensions icon in the Activity bar on the left side of the tool or in the View menu (**View** -> **Extensions**). For example search for **java** and add the **Extension Pack for Java,** using **Install**. Make sure that all mentioned plugins from the list above are installed! There is also a great Java tutorial available for VSC: https://code.visualstudio.com/docs/java/java-tutorial.



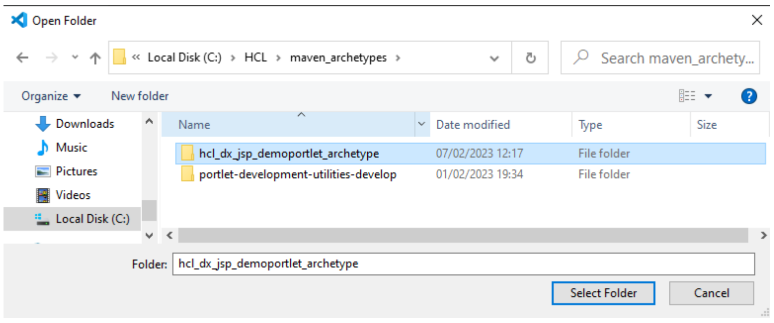
7. Then install the HCL DX Maven repository add-ons. The global Maven repository provides a lot of Maven archetypes, which are project templates for different kinds of products and functions. For details, please check: https://maven.apache.org/guides/introduction/introduction-to-archetypes.html. You will use HCL DX archetypes to speed up the development artifacts, like portlets, plugins, etc. for HCL DX. First download the git-repository: https://github.com/HCL-TECH-SOFTWARE/dx-portlet-development-utilities. You may install using a Git client with the command line:

```
git clone https://github.com/HCL-TECH-SOFTWARE/dx-portlet-
development-utilities
```

8. You may also download the repository as a ZIP-file. Click the green **<> Code** button.

9. And then click the **Download ZIP** option.

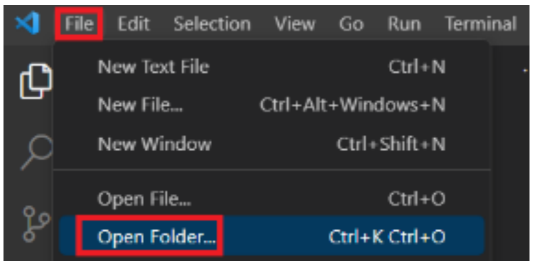
10. After downloading the repository, extract the file into a new directory. For example, on Mac or Linux in directory **/opt/HCL/maven-archetypes** or on Windows in **C:\HCL\maven-archetypes**.



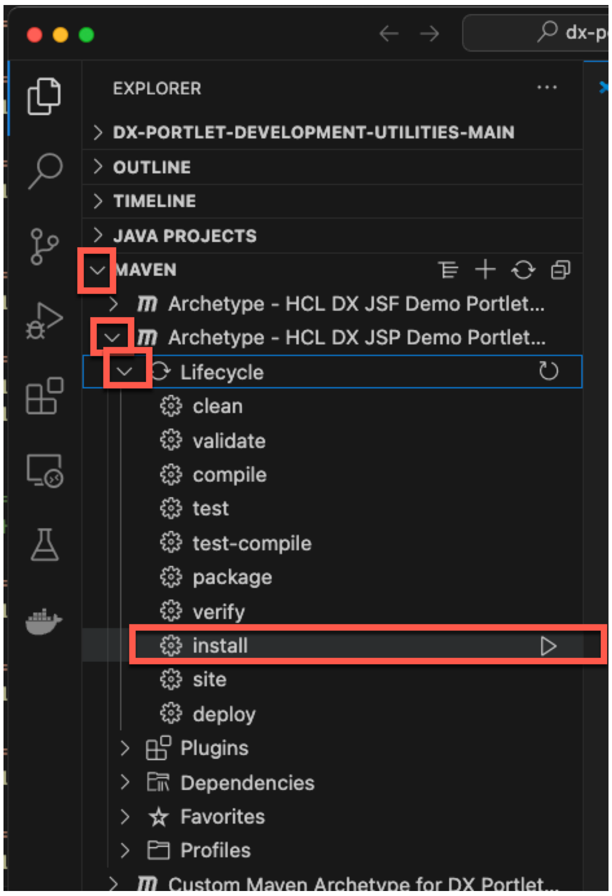
Name	Date modified	Type	Size
hcl_dx_jsp_demoportlet_archetype	07/02/2023 12:17	File folder	
portlet-development-utilities-develop	01/02/2023 19:34	File folder	

Folder: hcl_dx_jsp_demoportlet_archetype

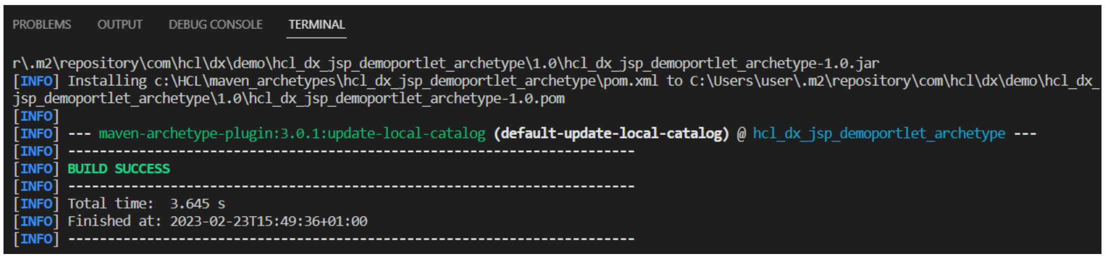
11. Then open this new folder in Visual Studio Code Client. Click **File** - **Open Folder**.



12. And open your expanded directory **dx-portlet-development-utilities-main**. Now try out if your Maven is set up correctly. Expand the **MAVEN** - **Archetype – HCL DX JSP Demo Portlet** - **Lifecycle** and click **install.**

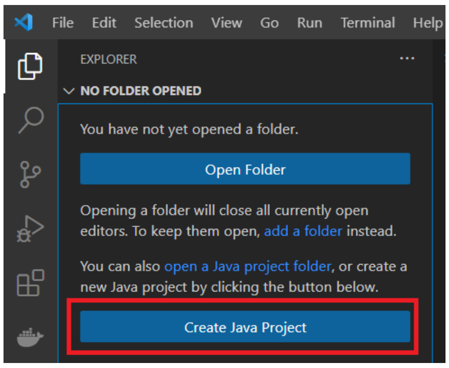

EXPLORER
DX-PORTLET-DEVELOPMENT-UTILITIES-MAIN
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
Archetype - HCL DX JSF Demo Portlet...
Archetype - HCL DX JSP Demo Portlet...
Lifecycle
clean
validate
compile
test
test-compile
package
verify
install
site
deploy
Plugins
Dependencies
Favorites
Profiles
Custom Maven Archetype for DX Portlet...

13. Under **TERMINAL**, you will see it being installed and it should complete with BUILD SUCCESS.



```
r\.m2\repository\com\hcl\dx\demo\hcl_dx_jsp_demoportlet_archetype\1.0\hcl_dx_jsp_demoportlet_archetype-1.0.jar
[INFO] Installing c:\HCL\maven_archetypes\hcl_dx_jsp_demoportlet_archetype\pom.xml to C:\Users\user\.m2\repository\com\hcl\dx\demo\hcl_dx_
jsp_demoportlet_archetype\1.0\hcl_dx_jsp_demoportlet_archetype-1.0.pom
[INFO]
[INFO] --- maven-archetype-plugin:3.0.1:update-local-catalog (default-update-local-catalog) @ hcl_dx_jsp_demoportlet_archetype ---
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.645 s
[INFO] Finished at: 2023-02-23T15:49:36+01:00
[INFO] ------------------------------------------------------------------------
```

14. Next, you will test if the new Maven archetype works. In VSC, click **File** - **Close Folder**, then the **Explorer** icon and **Create Java Project.**

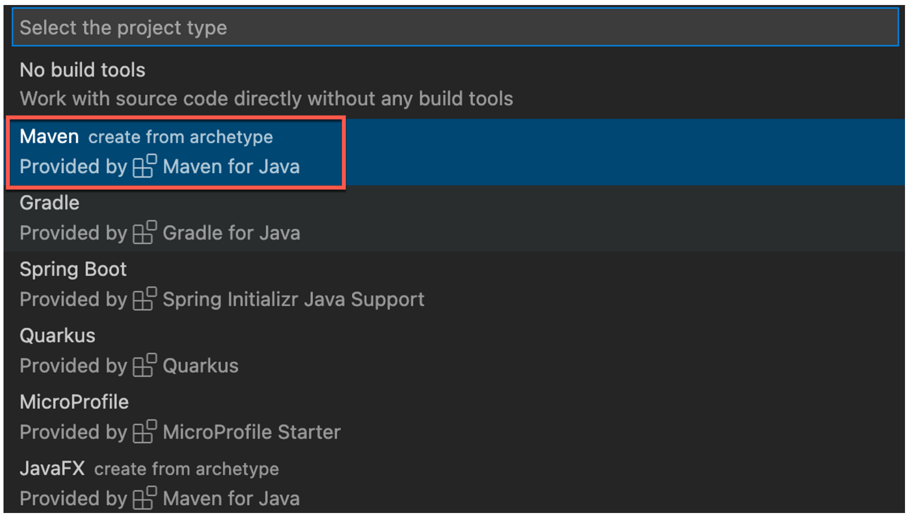


You have not yet opened a folder.

Opening a folder will close all currently open editors. To keep them open, add a folder instead.

You can also open a Java project folder, or create a new Java project by clicking the button below.

15. You should see different project types for the Java artifacts. Select **Maven.**



Select the project type

No build tools
Work with source code directly without any build tools

Maven create from archetype
Provided by Maven for Java

Gradle
Provided by Gradle for Java

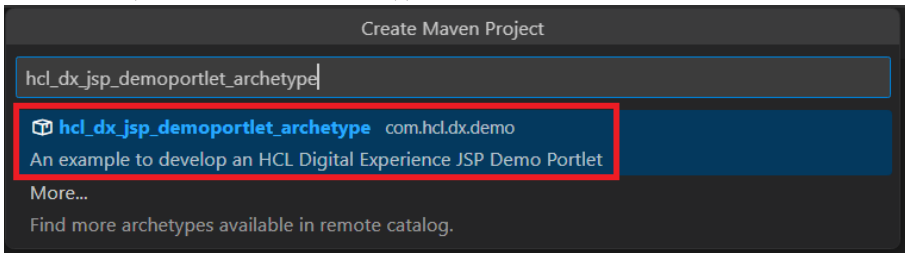
Spring Boot
Provided by Spring Initializr Java Support

Quarkus
Provided by Quarkus

MicroProfile
Provided by MicroProfile Starter

JavaFX create from archetype
Provided by Maven for Java

16. Search for the name **hcl_dx_jsp_demoportlet_archetype**. If you can find this, you have successfully installed that archetype.



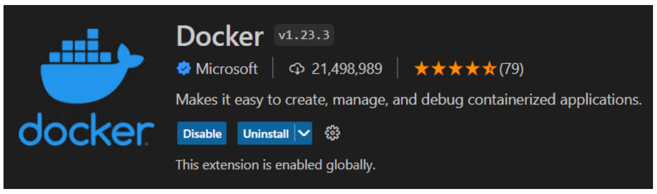
Create Maven Project

hcl_dx_jsp_demoportlet_archetype

hcl_dx_jsp_demoportlet_archetype com.hcl.dx.demo
An example to develop an HCL Digital Experience JSP Demo Portlet

More...
Find more archetypes available in remote catalog.

17. If you want to run HCL DX on Docker, you should install the additional extension **Docker for Visual Studio Code (Vendor Microsoft)** in Visual Studio Code. The plugin works very well when working with a local HCL DX server running on Docker.



Docker v1.23.3
Microsoft | 21,498,989 | ★★★★★ (79)
Makes it easy to create, manage, and debug containerized applications.
Disable Uninstall
This extension is enabled globally.

Use the terminal to run DXClient and to test if the remote connection to your HCL DX server is working. The tool will allow you to easily deploy DX Java artifacts. You may use this to remotely connect to your DX server, e.g. on HCL SoFy. With the DXClient, you also have options to restart your server remotely. See details on https://opensource.hcltechsw.com/digital-experience/latest/extend_dx/development_tools/dxclient/dxclient_artifact_types/dxcoreserver/. If you are using Docker Compose locally or have traditional installation, you need to ensure your server1 is started. Use the following command to start server1:

Docker:

docker exec dx-core sh -c "/opt/HCL/AppServer/profiles/cw_profile/bin/startServer.sh server1

Traditional: go to your <cw_profile_root> directory, e.g. cd /opt/HCL/AppServer/profiles/cw_profile/bin/ and then start server1, e.g. ./startServer.sh server1

18. Then check out the list of APIs and SPIs and their corresponding Java documentation for HCL DX. You can find this under https://opensource.hcltechsw.com/digital-experience/latest/extend_dx/apis/.

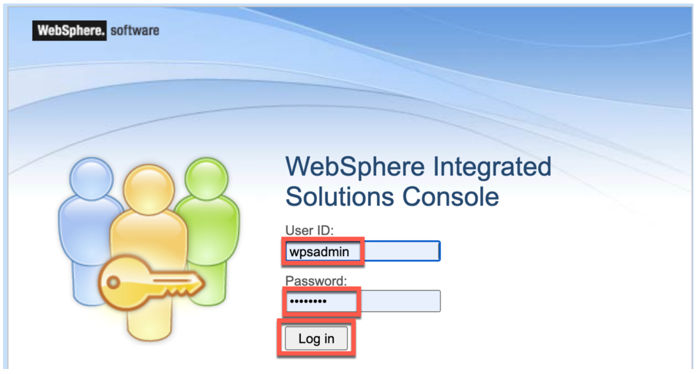
19. You also may need to use a few DX Jar files to be able to call these APIs. For example, you have the ones for Portal and WCM https://opensource.hcltechsw.com/digital-experience/latest/manage_content/wcm_development/wcm_dev_api and for Personalization https://opensource.hcltechsw.com/digital-experience/latest/manage_content/pzn/pzn_programming_ref/using_apis/pzn_jar_files_public_api/.

You now successfully have setup the IBM Java SDK, Visual Studio Code with the Java extension and Maven and you are ready to start developing your Java artifacts using Visual Studio Code.

Part 2: Setup DX Server for Debugging

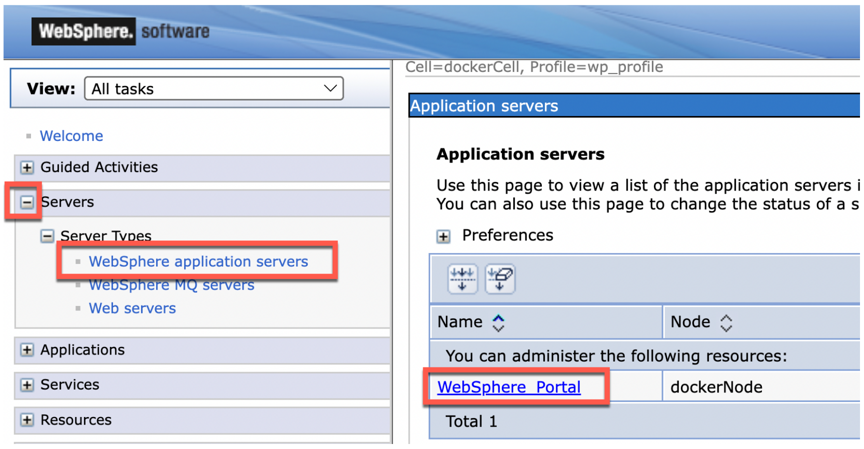
As you want to be able to debug your Java developments, you will enable debugging on your DX server. This is currently not possible on HCL SoFy, as the needed ports are not open. However, you may do this on your local deployment.

1. You do this using the Integrated Solution Console of your DX server. Open a new browser tab or window and open <host>/ibm/console, e.g. localhost/ibm/console. Then enter your admin credentials: e.g. wpsadmin/wpsadmin and click **Log in**.



User ID: wpsadmin
Password: ••••••••

2. Expand **Servers**, **Server Types**, click **WebSphere application servers** and **WebSphere_Portal**.



View: All tasks

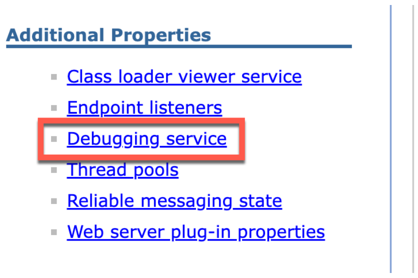
Cell=dockerCell, Profile=wp_profile

Application servers

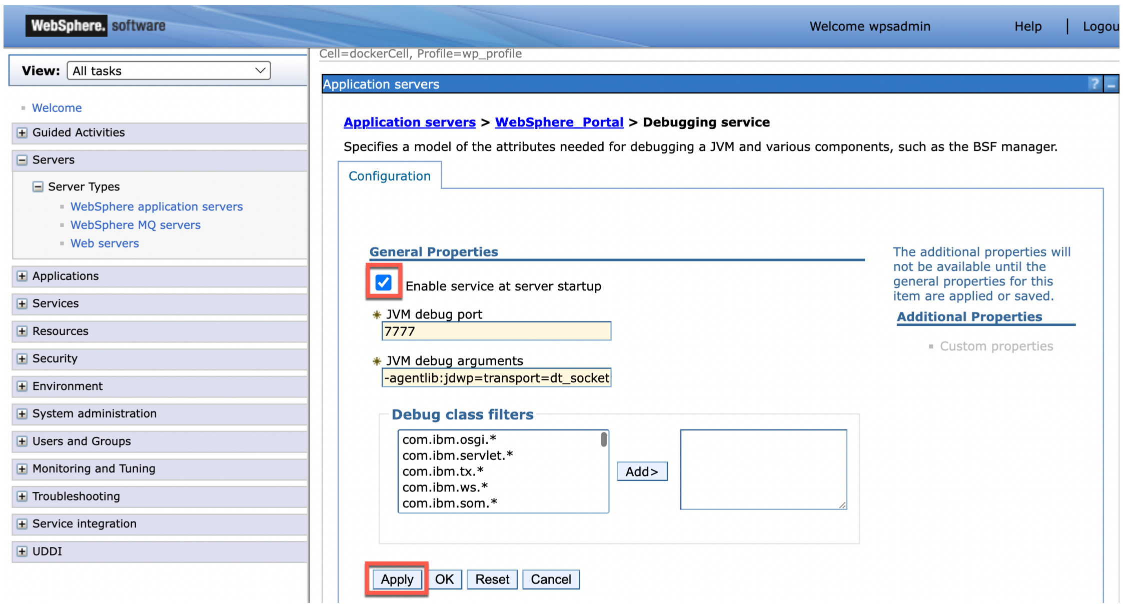
Use this page to view a list of the application servers i
You can also use this page to change the status of a s

Name	Node
You can administer the following resources:	
WebSphere_Portal	dockerNode
Total 1	

3. Scroll down and under **Additional Properties**, click **Debugging service**.



4. Then enable **Enable service at server startup**. This sets the JVM debug port to 7777 and JVM debug arguments to -agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=7777 automatically. Keep those. Click **Apply**.



Cell=dockerCell, Profile=wp_profile

Application servers

Application servers > **WebSphere_Portal** > **Debugging service**

Specifies a model of the attributes needed for debugging a JVM and various components, such as the BSF manager.

Configuration

General Properties

☑ Enable service at server startup

* JVM debug port
7777

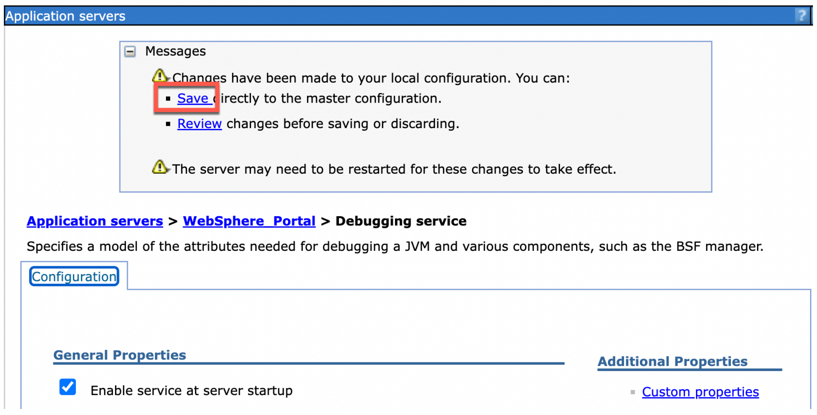
* JVM debug arguments
-agentlib:jdwp=transport=dt_socket

Debug class filters
com.ibm.osgi.*
com.ibm.servlet.*
com.ibm.tx.*
com.ibm.ws.*
com.ibm.som.*

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties
- Custom properties

5. And click **Save** to save the changes to the master configuration.



Application servers

Messages

Changes have been made to your local configuration. You can:
- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Application servers > **WebSphere_Portal** > **Debugging service**

Specifies a model of the attributes needed for debugging a JVM and various components, such as the BSF manager.

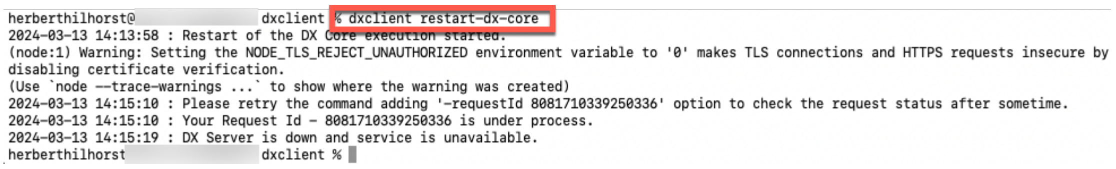
Configuration

General Properties

☑ Enable service at server startup

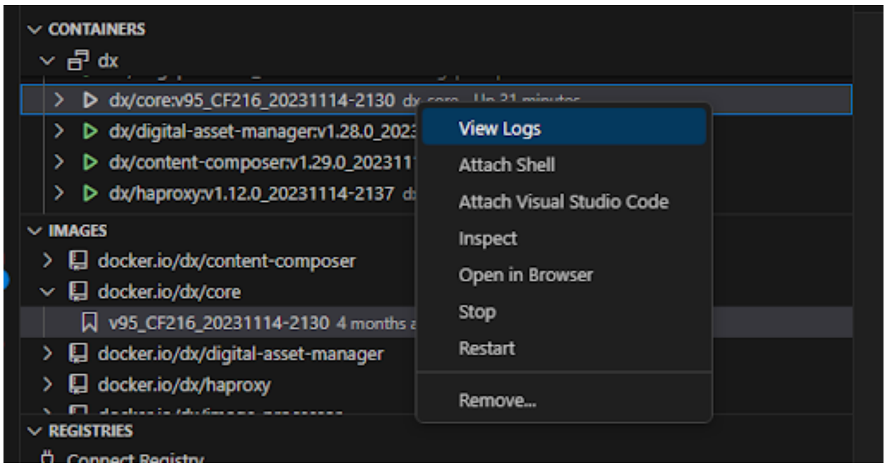
Additional Properties
- Custom properties

6. And restart the DX server. You may use your DXClient for this with the restart-dx-core for a local Docker-Compose or traditional installation and restart-code-pods with a Kubernetes deployment.



```
herberthilhorst@[redacted] dxclient % dxclient restart-dx-core
2024-03-13 14:13:58 : Restart of the DX Core execution started.
(node:1) Warning: Setting the NODE_TLS_REJECT_UNAUTHORIZED environment variable to '0' makes TLS connections and HTTPS requests insecure by
disabling certificate verification.
(Use `node --trace-warnings ...` to show where the warning was created)
2024-03-13 14:15:10 : Please retry the command adding '-requestId 8081710339250336' option to check the request status after sometime.
2024-03-13 14:15:10 : Your Request Id - 8081710339250336 is under process.
2024-03-13 14:15:19 : DX Server is down and service is unavailable.
herberthilhorst[redacted] dxclient %
```

7. In addition, you may want to simplify accessing the log files on your DX server. For VSC, you may use the Docker plugin which allows you to easily work with local containers. For example, easily access your Docker log files, stop and start them, and more. See https://marketplace.visualstudio.com/items?itemName=ms-azuretools.vscode-docker.


CONTAINERS
dx
dx/core:v95_CF216_20231114-2130
dx/digital-asset-manager:v1.28.0_202
dx/content-composer:v1.29.0_202311
dx/haproxy:v1.12.0_20231114-2137
IMAGES
docker.io/dx/content-composer
docker.io/dx/core
v95_CF216_20231114-2130 4 months
docker.io/dx/digital-asset-manager
docker.io/dx/haproxy
REGISTRIES
Connect Registry
View Logs
Attach Shell
Attach Visual Studio Code
Inspect
Open in Browser
Stop
Restart
Remove...

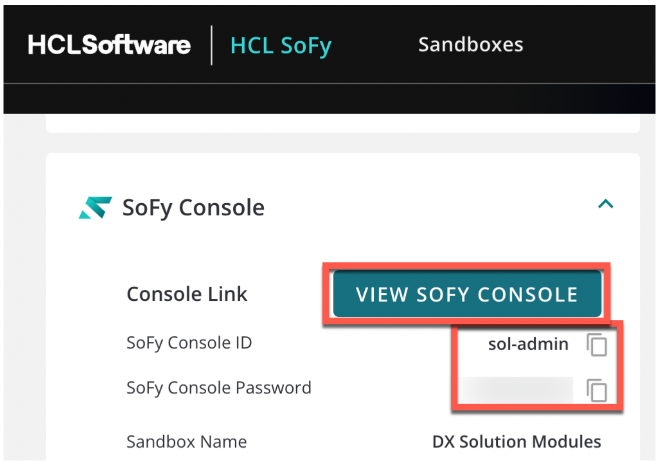
8. For other DX deployments, like local traditional or remote DX server, you may want to check what's the easiest way to access your log files. See details in https://opensource.hcltechsw.com/digital-experience/latest/deployment/manage/troubleshooting/logging_and_tracing/. For example, with Kubernetes you can use read the SystemOut.log using

```
kubectl logs -n dxns dx-deployment-core-0 system-out-log
```

9. For Docker-Compose you can use

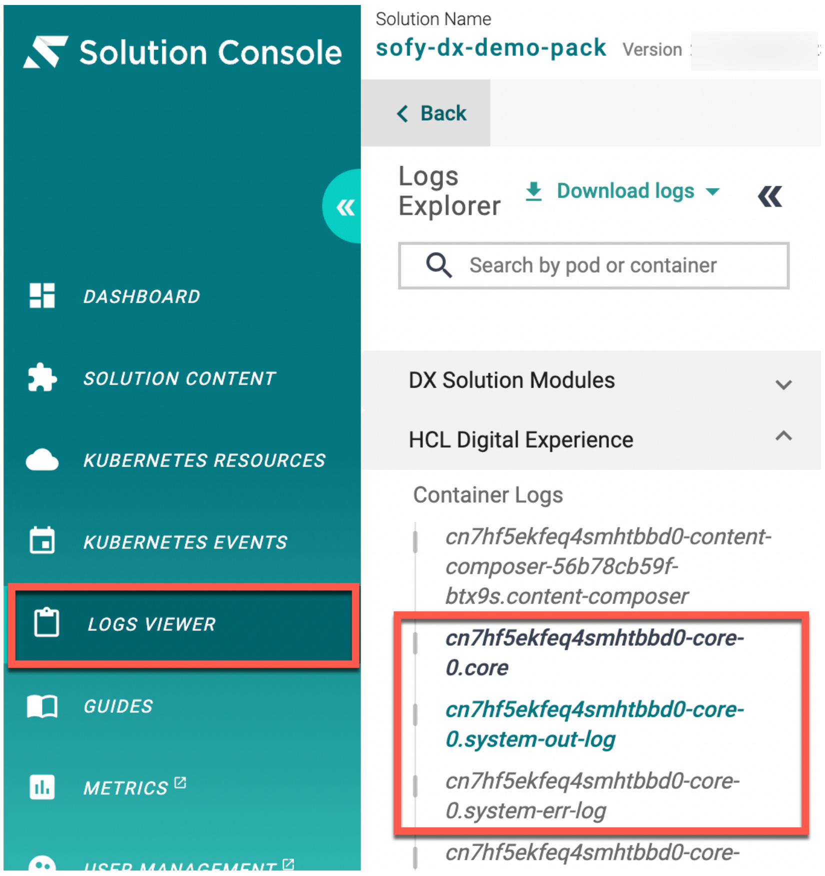
```
call docker logs --tail 1000 -f dx-core
```

10. And in SoFy, you can connect to the Solution Console and use the Pod log files. On your instance, open the SoFyConsole and use the console ID and Password to log in.



Console Link
SoFy Console ID sol-admin
SoFy Console Password
Sandbox Name DX Solution Modules

11. And go to the **LOGS VIEWER** where you you find the SystemOut and SystemErr log files.


Solution Console
Solution Name
sofy-dx-demo-pack
Version
Back
Logs Explorer
Download logs
Search by pod or container
DASHBOARD
SOLUTION CONTENT
KUBERNETES RESOURCES
KUBERNETES EVENTS
LOGS VIEWER
GUIDES
METRICS
USER MANAGEMENT
DX Solution Modules
HCL Digital Experience
Container Logs
cn7hf5ekfeq4smhtbbd0-content-composer-56b78cb59f-btx9s.content-composer
cn7hf5ekfeq4smhtbbd0-core-0.core
cn7hf5ekfeq4smhtbbd0-core-0.system-out-log
cn7hf5ekfeq4smhtbbd0-core-0.system-err-log
cn7hf5ekfeq4smhtbbd0-core-

You have successfully set up your DX server for debugging!

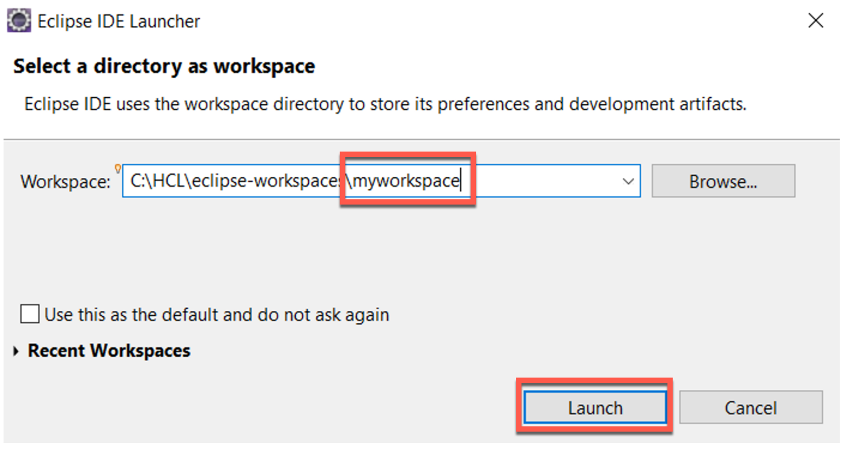
Optional Part 3: DX Java Development with Eclipse and Maven

You may set up and configure the Eclipse IDE to work with Java and combine it with Maven to make it simple to create and deploy the different artifacts for DX on Docker-Compose or remote DX server, or with a traditional installation.

1. First use the instructions of part 1 to install Java and Maven locally and download the additional DX Maven archetypes. Then install Eclipse. The Eclipse IDE supports plugins to connect to a containerized environment, like Docker-Compose or Podman. Unless you are using a local traditional installation of your DX server, download the latest Eclipse IDE for Enterprise Java and Web Developers that is available, as the "Eclipse Docker Tooling Plugin" is just compatible with the latest versions. In this lab, you will find instructions using version "2022-12", which is running with OpenJDK version 11. Downloaded the Eclipse version from URL: https://www.eclipse.org/downloads/packages/. If you have local traditional installation, there are different instructions and go to step 15.

2. Then extract the installation binaries into any folder. For example, on Microsoft Windows in directory C:\HCL\eclipse.

3. Start Eclipse and choose a workspace directory for your new projects. For example: **myworkspace**. Then click **Launch**.



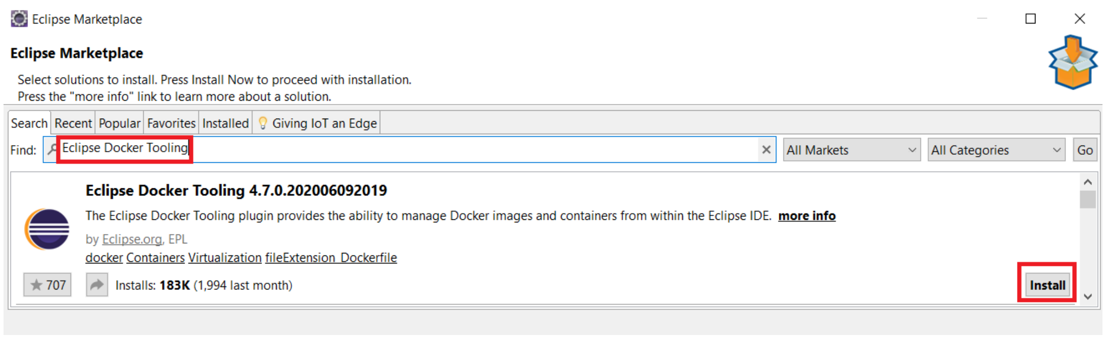
Eclipse IDE Launcher

Select a directory as workspace

Eclipse IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\HCL\eclipse-workspace\myworkspace

4. If you have a local DX server running on Docker-Compose, configure Eclipse to work with it. In the Eclipse menu click to **Help** - **Eclipse Marketplace**. Then search for the plugin "**Eclipse Docker Tooling**" and click to the **Install** button to install it.



Eclipse Marketplace

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Find: Eclipse Docker Tooling

Eclipse Docker Tooling 4.7.0.202006092019

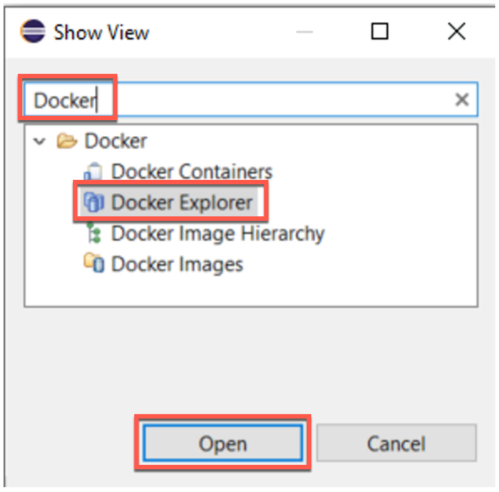
The Eclipse Docker Tooling plugin provides the ability to manage Docker images and containers from within the Eclipse IDE. **more info**

by Eclipse.org, EPL

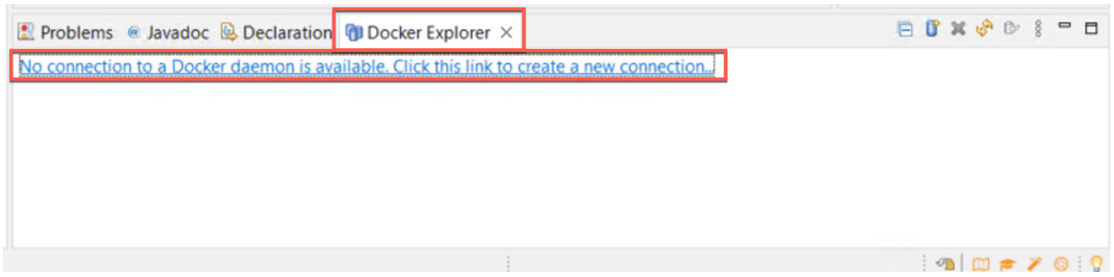
docker Containers Virtualization fileExtension_Dockerfile

707 Installs: **183K** (1,994 last month)

5. As soon as the plugin is installed, go in the Eclipse menu to **Window** - **Show View**
   Then search for **Docker**, select **Docker Explorer** and click **Open** button.



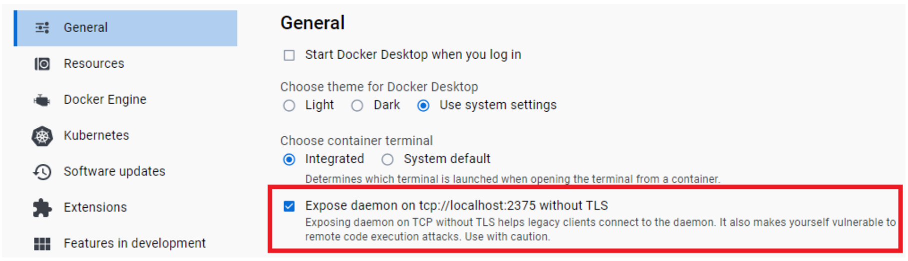
6. In the new Docker Explorer tab, click **No connection to a Docker daemon is available. Click this link to create a new connection….**



No connection to a Docker daemon is available. Click this link to create a new connection...

7. When running Docker Desktop, open the Administration GUI of Docker Desktop and click the settings icon: .

8. Under **General** make sure that **Expose daemon on tcp://localhost:2375 without TLS** is selected. If it is not selected, select it and restart Docker. If you are using a different platform, please check the documentation to find out in how to expose the daemon.



General

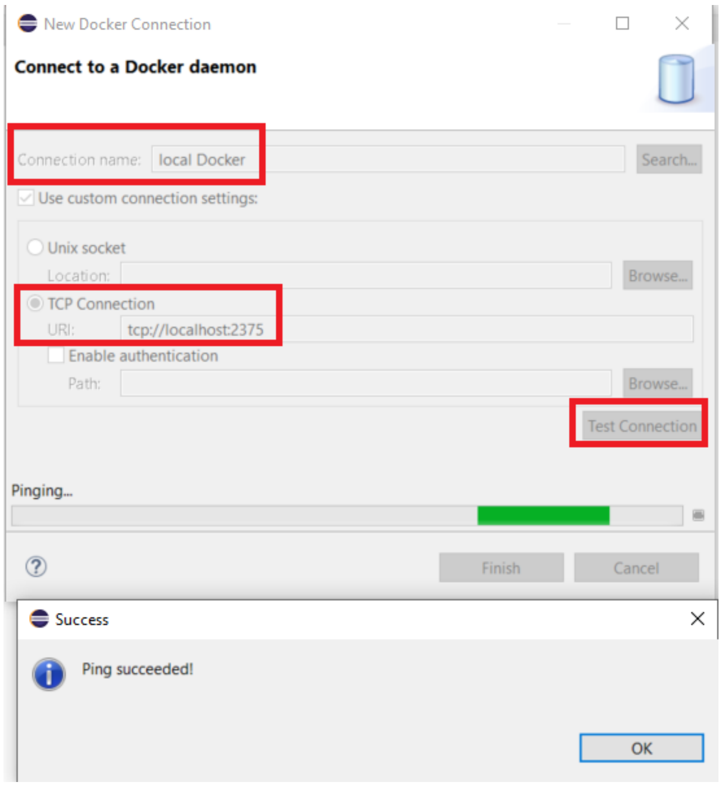
- [ ] Start Docker Desktop when you log in

Choose theme for Docker Desktop
Light   Dark   [✓ Use system settings]

Choose container terminal
[✓ Integrated]   System default
Determines which terminal is launched when opening the terminal from a container.

- [x] Expose daemon on tcp://localhost:2375 without TLS
  Exposing daemon on TCP without TLS helps legacy clients connect to the daemon. It also makes yourself vulnerable to remote code execution attacks. Use with caution.

9. Back in Eclipse, type in a Connection name. For example: **local Docker** and under **TCP Connection** enter the URL that is mentioned in the Docker user interface, for the above example: tcp://localhost:2375. Then click **Test Connection**. If the test connection is successful, click **OK** and **Finish** to complete.


New Docker Connection

Connect to a Docker daemon

Connection name: local Docker

Use custom connection settings:

Unix socket

Location:

TCP Connection

URI: tcp://localhost:2375

Enable authentication

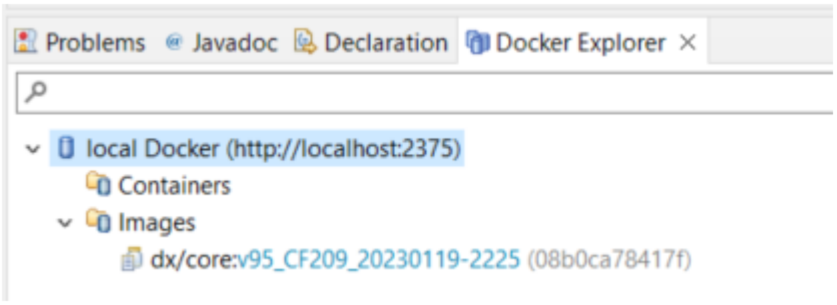
Path:

Pinging...

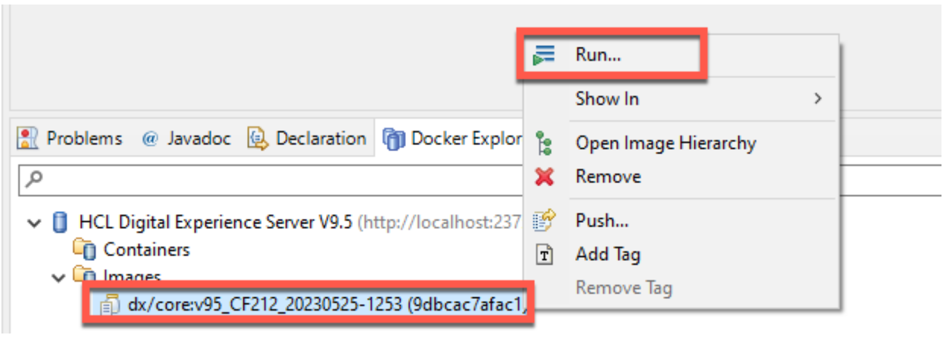
Success

Ping succeeded!

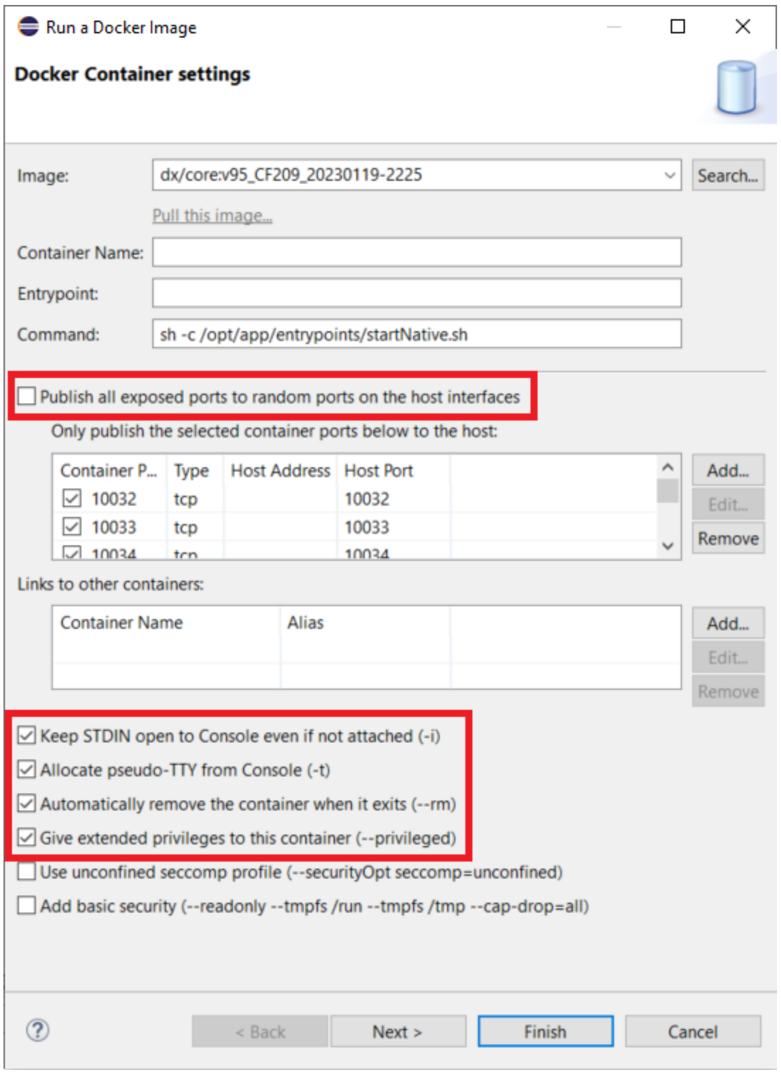
10. The Docker Explorer should now show up all available Containers and images of your Docker instance. For example:


Problems
Javadoc
Declaration
Docker Explorer
local Docker (http://localhost:2375)
Containers
Images
dx/core:v95_CF209_20230119-2225 (08b0ca78417f)

11. Use a right mouse-click on the image name to start a new container and click **Run...**



12. Uncheck the **Publish all exposed ports to random parts on the host interfaces** and check the options shown below, before clicking **Finish** to run a container:



Run a Docker Image

Docker Container settings

Image: dx/core:v95_CF209_20230119-2225

Container Name:

Entrypoint:

Command: sh -c /opt/app/entrypoints/startNative.sh

☐ Publish all exposed ports to random ports on the host interfaces

Only publish the selected container ports below to the host:

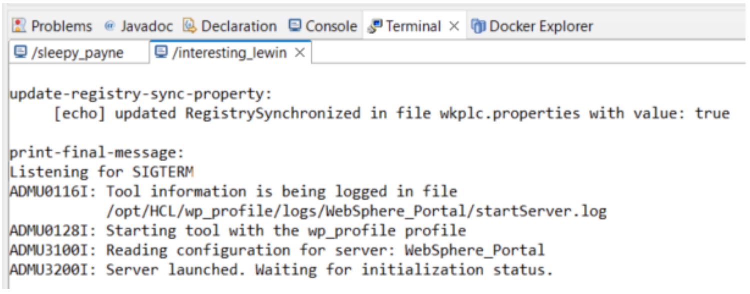
Container P...	Type	Host Address	Host Port
✓ 10032	tcp		10032
✓ 10033	tcp		10033
✓ 10034	tcp		10034

Links to other containers:

Container Name	Alias

☑ Keep STDIN open to Console even if not attached (-i)
☑ Allocate pseudo-TTY from Console (-t)
☑ Automatically remove the container when it exits (--rm)
☑ Give extended privileges to this container (--privileged)
☐ Use unconfined seccomp profile (--securityOpt seccomp=unconfined)
☐ Add basic security (--readonly --tmpfs /run --tmpfs /tmp --cap-drop=all)

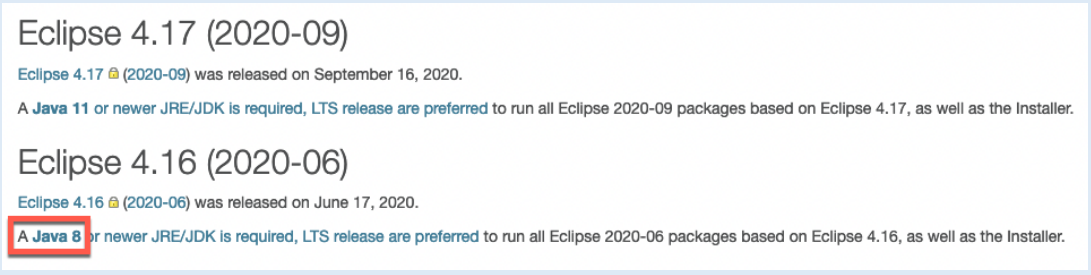
13. The Terminal window shows up with the container logs. For example:



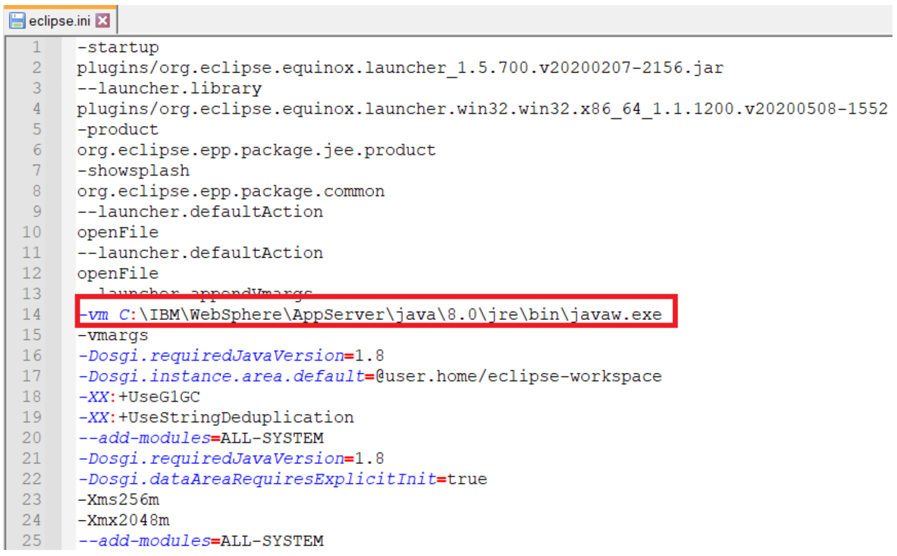
```
update-registry-sync-property:
     [echo] updated RegistrySynchronized in file wkplc.properties with value: true

print-final-message:
Listening for SIGTERM
ADMU0116I: Tool information is being logged in file
           /opt/HCL/wp_profile/logs/WebSphere_Portal/startServer.log
ADMU0128I: Starting tool with the wp_profile profile
ADMU3100I: Reading configuration for server: WebSphere_Portal
ADMU3200I: Server launched. Waiting for initialization status.
```

14. Your Eclipse is now set up to work with your local Docker-Compose instance. You are now ready to add the Maven archetypes needed to develop your Java portlets more easily. Go to step 26 to install it.

15. In the case, you have a local traditional installation, download an Eclipse version from URL that supports Java 8: https://www.eclipse.org/downloads/packages/.

> If you are using a local traditional installation of DX, you need to use an Eclipse version that supports Java 8, because the WebSphere Development Toolkit just supports Java 8 right now. (For example use Eclipse version 4.16)
>
> 
>
> Eclipse 4.17 (2020-09)
>
> Eclipse 4.17 (2020-09) was released on September 16, 2020.
>
> A **Java 11** or newer JRE/JDK is required, LTS release are preferred to run all Eclipse 2020-09 packages based on Eclipse 4.17, as well as the Installer.
>
> Eclipse 4.16 (2020-06)
>
> Eclipse 4.16 (2020-06) was released on June 17, 2020.
>
> A **Java 8** or newer JRE/JDK is required, LTS release are preferred to run all Eclipse 2020-06 packages based on Eclipse 4.16, as well as the Installer.

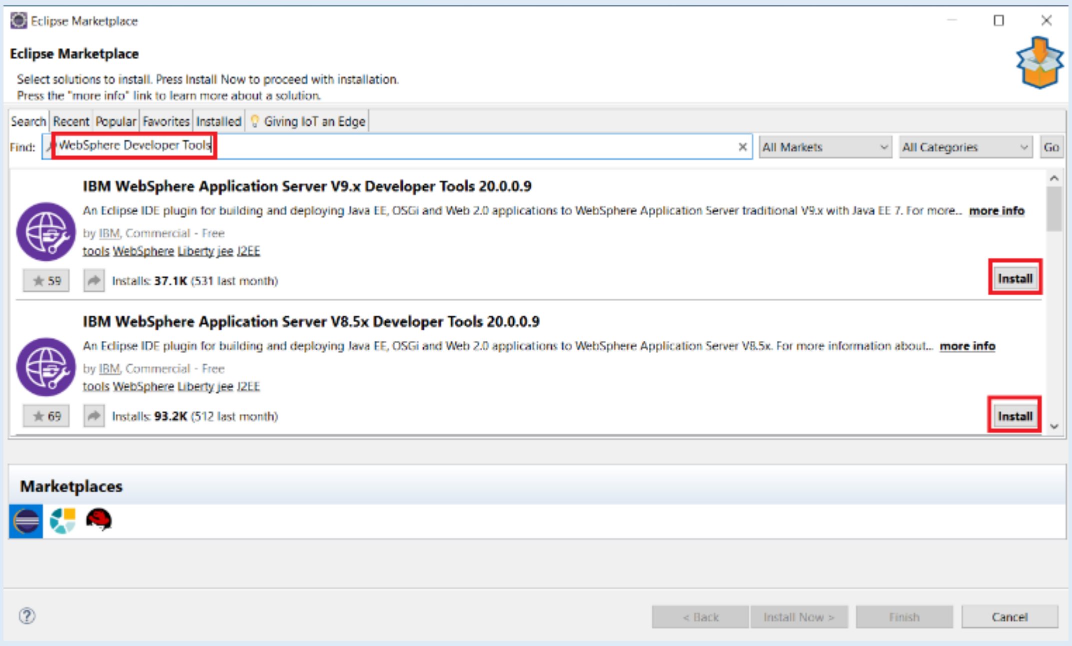
16. Then extract the installation binaries into any folder. For example, on Microsoft Windows in C:\HCL\eclipse. As soon as eclipse is installed, check that your eclipse.ini file points to the right Java location of your IBM Java binary. You may specify that with a -vm parameter entry, as shown in the next screenshot:



```
-startup
plugins/org.eclipse.equinox.launcher_1.5.700.v20200207-2156.jar
--launcher.library
plugins/org.eclipse.equinox.launcher.win32.win32.x86_64_1.1.1200.v20200508-1552
-product
org.eclipse.epp.package.jee.product
-showsplash
org.eclipse.epp.package.common
--launcher.defaultAction
openFile
--launcher.defaultAction
openFile
--launcher.appendVmargs
-vm C:\IBM\WebSphere\AppServer\java\8.0\jre\bin\javaw.exe
-vmargs
-Dosgi.requiredJavaVersion=1.8
-Dosgi.instance.area.default=@user.home/eclipse-workspace
-XX:+UseG1GC
-XX:+UseStringDeduplication
--add-modules=ALL-SYSTEM
-Dosgi.requiredJavaVersion=1.8
-Dosgi.dataAreaRequiresExplicitInit=true
-Xms256m
-Xmx2048m
--add-modules=ALL-SYSTEM
```

17. Start eclipse, choose a workspace directory to open your first project. In the Eclipse menu click to **Help → Install New Software…** Search for **WebSphere Developer Tools** and hit the install button.

Please note that there are two different WebSphere Developer Tools available.
1. IBM WebSphere Application Server V9.x Developer Tools 20.0.0.9
   -> Select this plugin, when you're running on HCL DX with IBM WebSphere Application Server version 9.x
2. IBM WebSphere Application Server v8.5.x Developer Tools 20.0.0.9
   -> Select this plugin when you run on HCL DX with IBM WebSphere Application Server version 8.5.x



Eclipse Marketplace

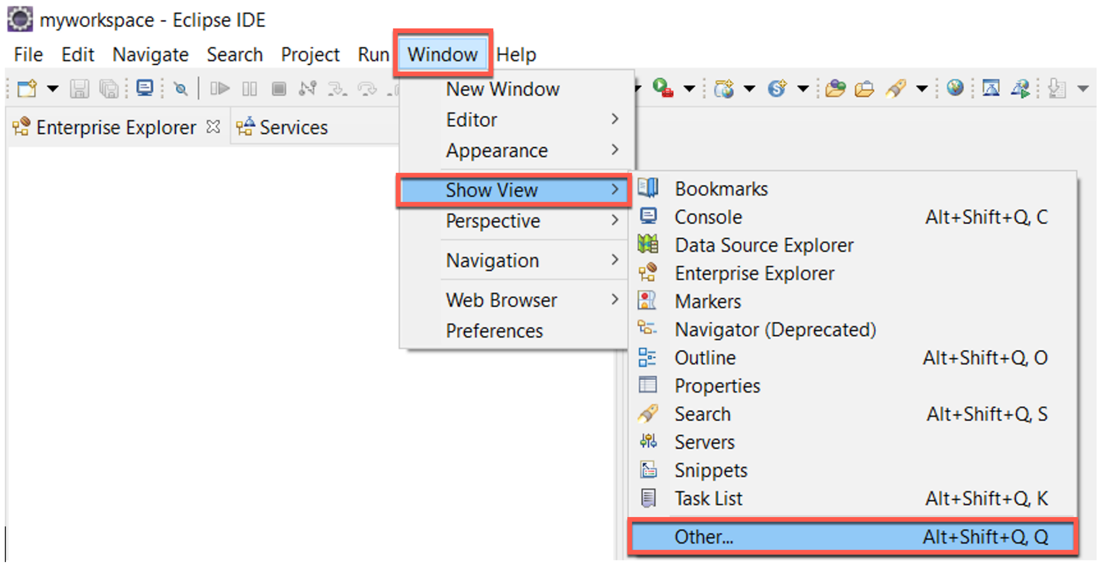
Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Find: WebSphere Developer Tools

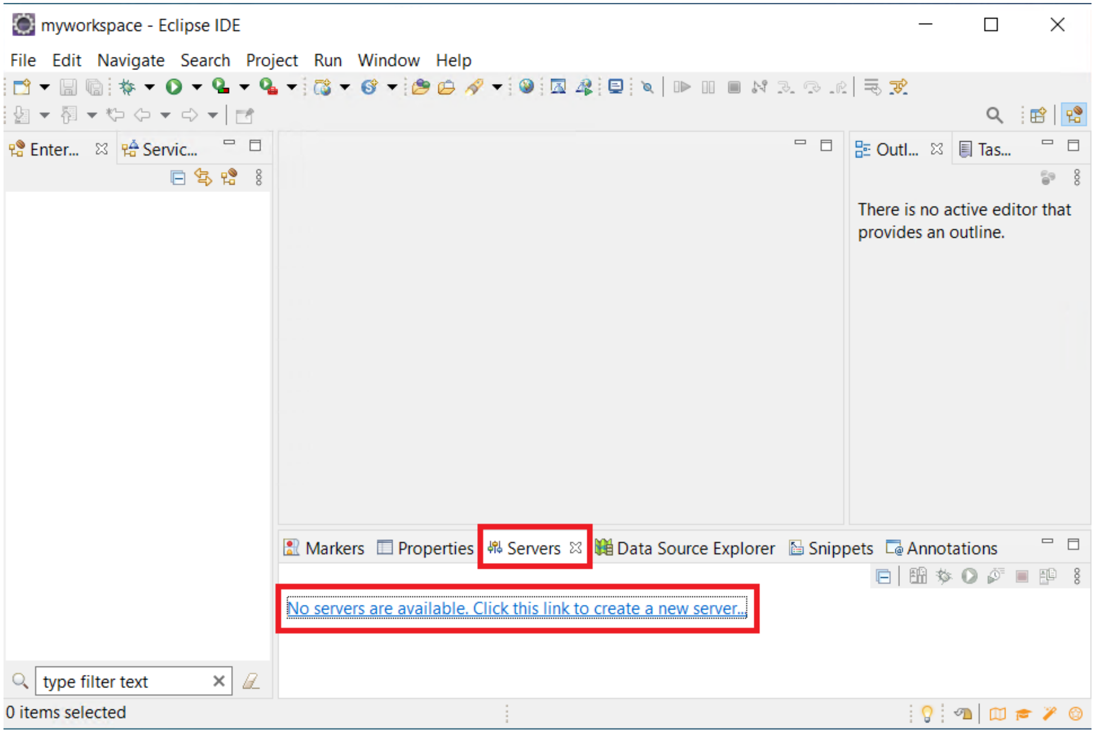
IBM WebSphere Application Server V9.x Developer Tools 20.0.0.9
An Eclipse IDE plugin for building and deploying Java EE, OSGi and Web 2.0 applications to WebSphere Application Server traditional V9.x with Java EE 7. For more... more info
by IBM, Commercial - Free
tools WebSphere Liberty jee J2EE
Installs: 37.1K (531 last month)

IBM WebSphere Application Server V8.5x Developer Tools 20.0.0.9
An Eclipse IDE plugin for building and deploying Java EE, OSGi and Web 2.0 applications to WebSphere Application Server V8.5x. For more information about... more info
by IBM, Commercial - Free
tools WebSphere Liberty jee J2EE
Installs: 93.2K (512 last month)

18. If a message pops up to ask for a restart, click **Restart Now**. When Eclipse is restarted click **Windows** - **Show View** - **Other…**



19. Click the **Servers** tab and **No servers are available. Click this link to create a new server…**

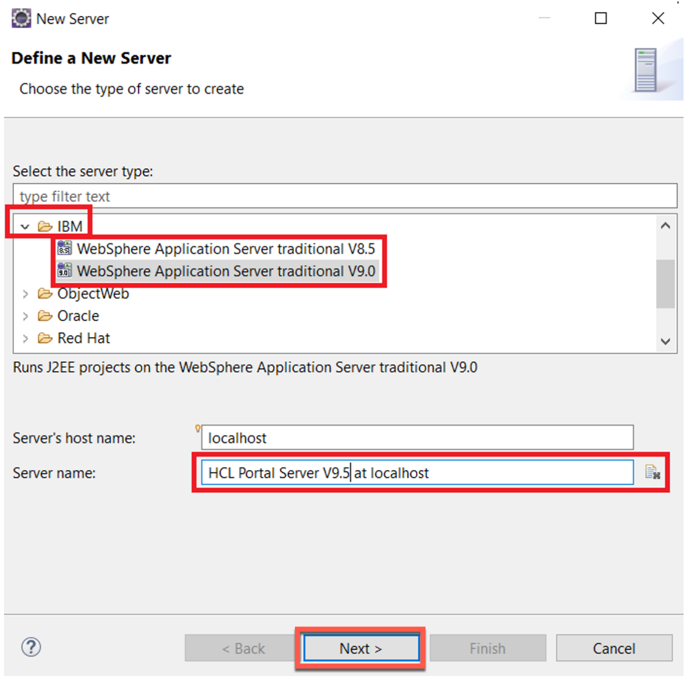


There is no active editor that provides an outline.

No servers are available. Click this link to create a new server...

0 items selected

20. In the **New Server** tab, expand **IBM** and select the WebSphere Application Server base version on which your DX Server is running. Choose V9.0 if you have DX v9.5 installed. Ensure the hostname is either localhost or the real hostname of your local development environment. Enter **Server name**, e.g. **HCL Portal Server V9.5 at localhost**. Then click **Next**.



Define a New Server

Choose the type of server to create

Select the server type:

type filter text

WebSphere Application Server traditional V8.5

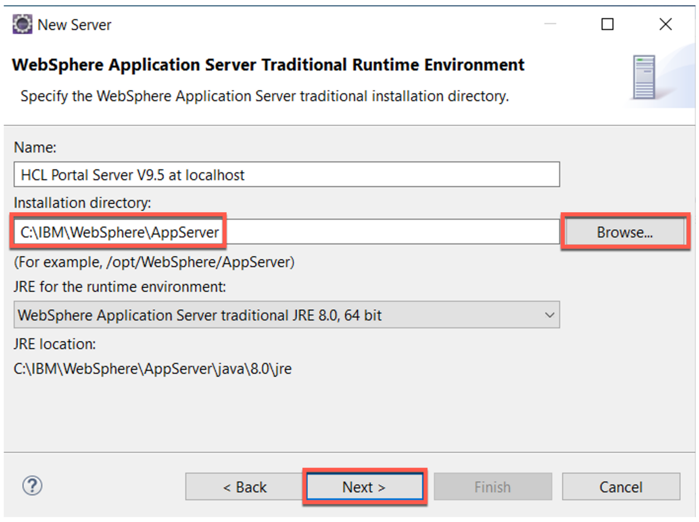
WebSphere Application Server traditional V9.0

Runs J2EE projects on the WebSphere Application Server traditional V9.0

Server's host name: localhost

Server name: HCL Portal Server V9.5 at localhost

21. Click **Browse…** and point to the IBM WebSphere Application Server Installation binary folder. Then click **Next**. For example:



New Server

WebSphere Application Server Traditional Runtime Environment

Specify the WebSphere Application Server traditional installation directory.

Name: HCL Portal Server V9.5 at localhost

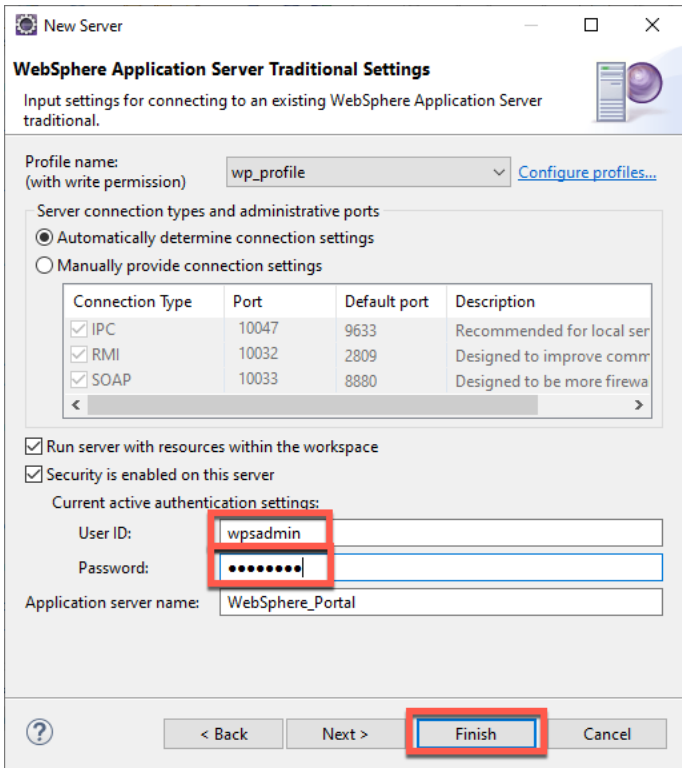
Installation directory: C:\IBM\WebSphere\AppServer

(For example, /opt/WebSphere/AppServer)

JRE for the runtime environment: WebSphere Application Server traditional JRE 8.0, 64 bit

JRE location: C:\IBM\WebSphere\AppServer\java\8.0\jre

22. Select the **Profile name** (default is **wp_profile**) and enter a **User ID** and **Password** of the WebSphere Application Server administration user (by default **wpsadmin/wpsadmin**). Click **Finish**. For example:



New Server

WebSphere Application Server Traditional Settings

Input settings for connecting to an existing WebSphere Application Server traditional.

Profile name: (with write permission) wp_profile

Server connection types and administrative ports

(● Automatically determine connection settings)

○ Manually provide connection settings

Connection Type	Port	Default port	Description
✓ IPC	10047	9633	Recommended for local ser
✓ RMI	10032	2809	Designed to improve comm
✓ SOAP	10033	8880	Designed to be more firewa

☑ Run server with resources within the workspace

☑ Security is enabled on this server

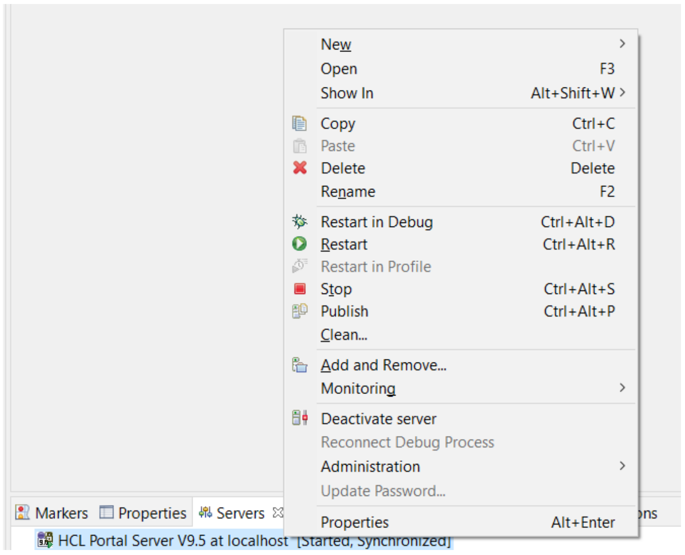
Current active authentication settings:

User ID: wpsadmin

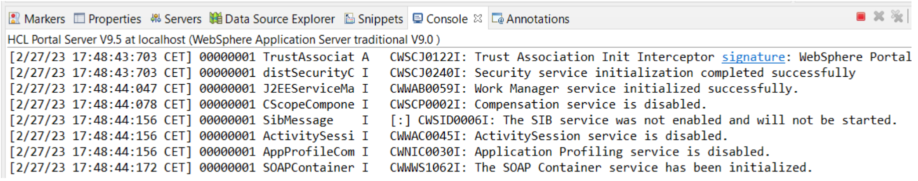
Password: ●●●●●●●●

Application server name: WebSphere_Portal

23. The new server's name should now be listed under the Servers tab in Eclipse. In addition to the name also the state of the servers should be shown right to the server's name. Doing a right click to the server's name, a menu will be opened that provide further actions that can be used on that server like Start, Stop, Restart, Add and Remove… and so on.



24. When the server is started a **Console** tab automatically opens to show the log entries. For example:

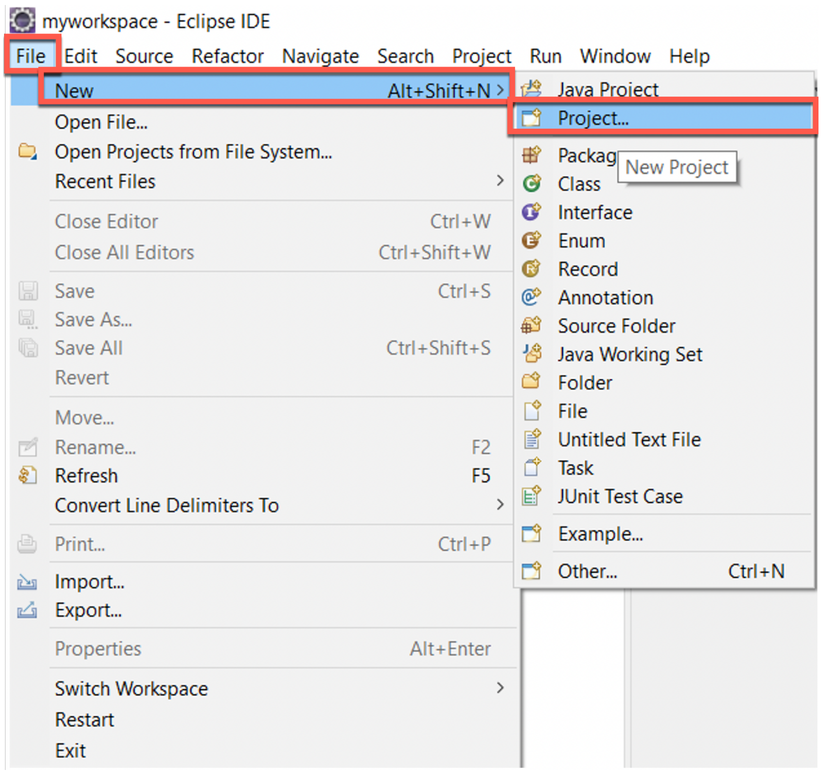


HCL Portal Server V9.5 at localhost (WebSphere Application Server traditional V9.0)

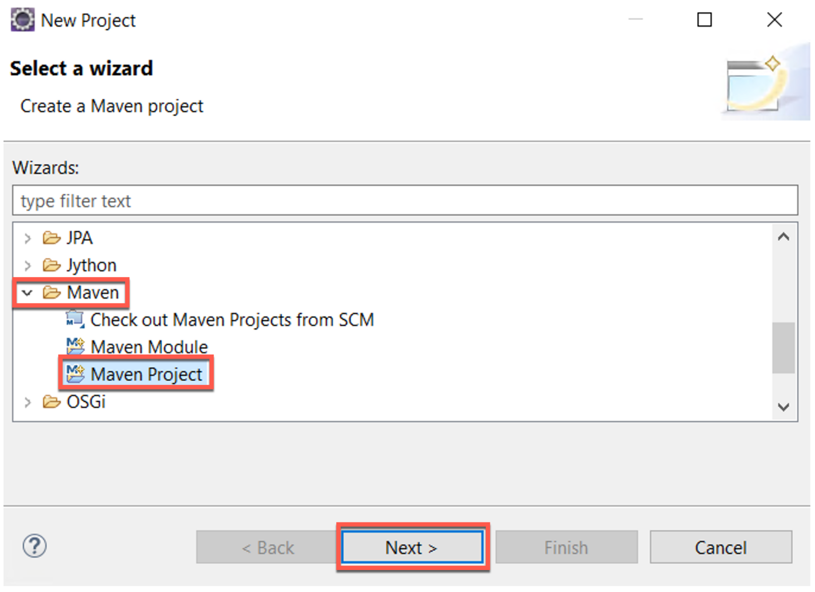
```
[2/27/23 17:48:43:703 CET] 00000001 TrustAssociat A   CWSCJ0122I: Trust Association Init Interceptor signature: WebSphere Portal
[2/27/23 17:48:43:703 CET] 00000001 distSecurityC I   CWSCJ0240I: Security service initialization completed successfully
[2/27/23 17:48:44:047 CET] 00000001 J2EEServiceMa I   CWWAB0059I: Work Manager service initialized successfully.
[2/27/23 17:48:44:078 CET] 00000001 CScopeCompone I   CWSCP0002I: Compensation service is disabled.
[2/27/23 17:48:44:156 CET] 00000001 SibMessage    I   [:] CWSID0006I: The SIB service was not enabled and will not be started.
[2/27/23 17:48:44:156 CET] 00000001 ActivitySessi I   CWWAC0045I: ActivitySession service is disabled.
[2/27/23 17:48:44:156 CET] 00000001 AppProfileCom I   CWNIC0030I: Application Profiling service is disabled.
[2/27/23 17:48:44:172 CET] 00000001 SOAPContainer I   CWWWS1062I: The SOAP Container service has been initialized.
```

25. Congratulations. You have successfully installed Eclipse to connect to a local HCL DX Server. Now you can set up Maven.

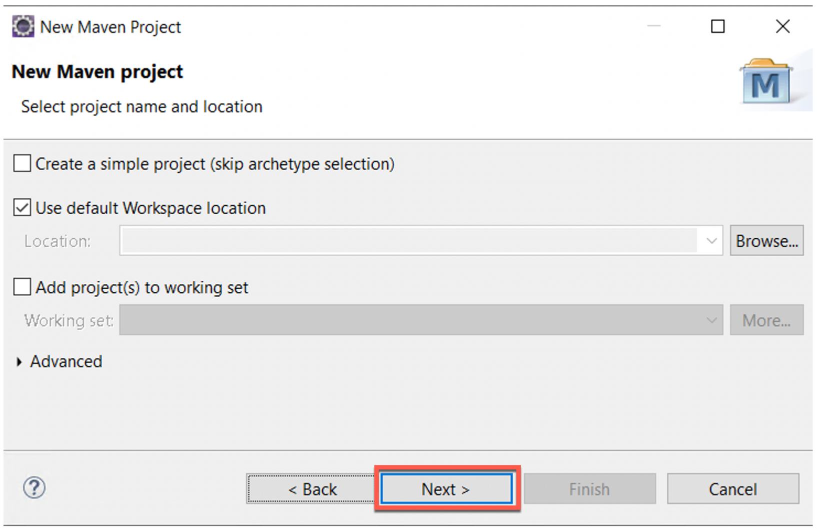
26. Now install Maven. You need to have downloaded and extracted the DX Portlet Development Utilities from https://github.com/HCL-TECH-SOFTWARE/dx-portlet-development-utilities. Start Eclipse and choose a new workspace directory to start a new project. Then in the Eclipse menu click to **File → New → Project…**



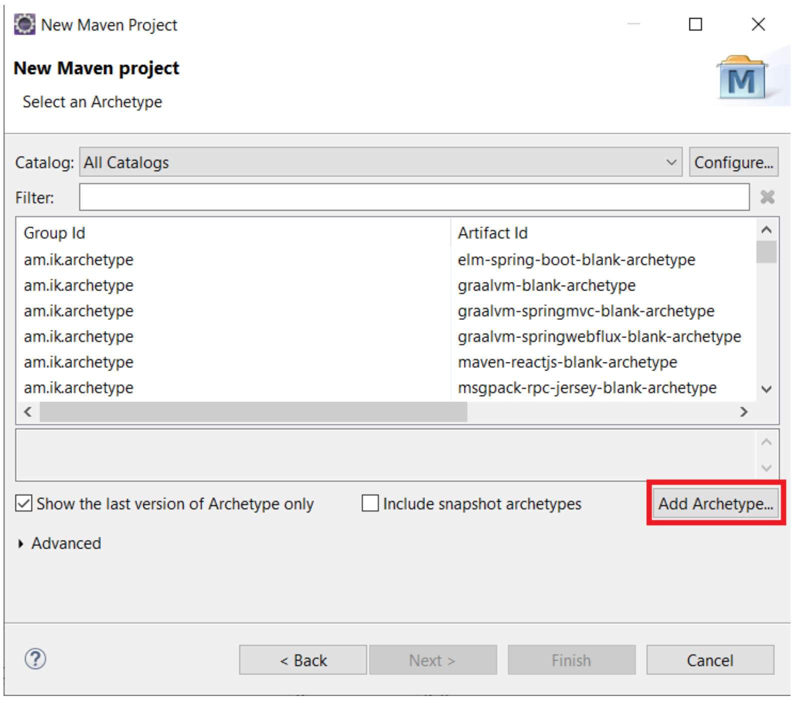
27. Scroll down, expand **Maven**, select **Maven Project** and click **Next >**.



28. Keep the default settings and click **Next >**.

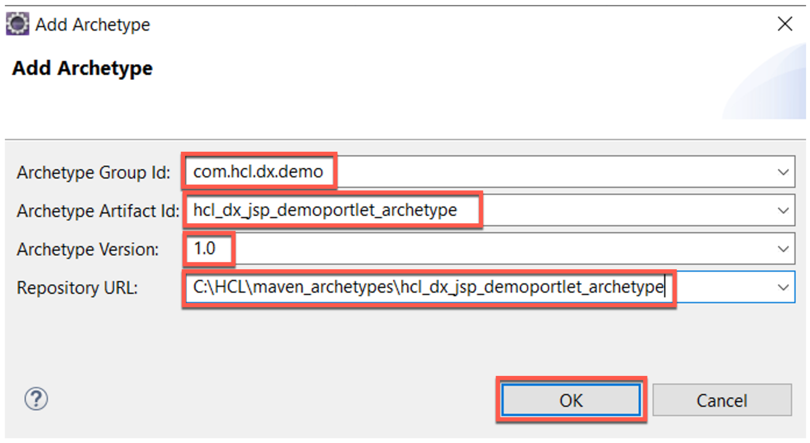

New Maven Project
New Maven project
Select project name and location
Create a simple project (skip archetype selection)
Use default Workspace location
Location:
Browse...
Add project(s) to working set
Working set:
More...
Advanced
< Back
Next >
Finish
Cancel

29. Then, click **Add Archetype…**



Group Id	Artifact Id
am.ik.archetype	elm-spring-boot-blank-archetype
am.ik.archetype	graalvm-blank-archetype
am.ik.archetype	graalvm-springmvc-blank-archetype
am.ik.archetype	graalvm-springwebflux-blank-archetype
am.ik.archetype	maven-reactjs-blank-archetype
am.ik.archetype	msgpack-rpc-jersey-blank-archetype

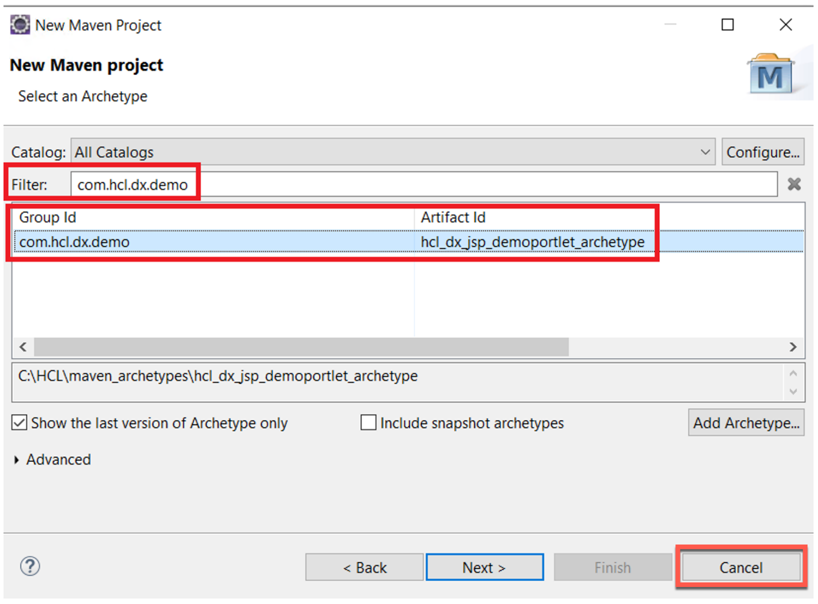
30. Add the following and click **OK**:
    Archetype Group Id: com.hcl.dx.demo
    Archetype Artifact Id: hcl_dx_jsp_demoportlet_archetype
    Archetype Version: 1.0
    Repository URL:
    <location of the downloaded GitHub repository>\hcl_dx_jsp_demoportlet_archetype


Add Archetype

Add Archetype

Archetype Group Id: com.hcl.dx.demo
Archetype Artifact Id: hcl_dx_jsp_demoportlet_archetype
Archetype Version: 1.0
Repository URL: C:\HCL\maven_archetypes\hcl_dx_jsp_demoportlet_archetype

31. Then, search for **com.hcl.dx.demo** group and check if the archetype can be found. If the plugin can be found, the Eclipse is configured correctly and can be used to develop HCL DX Portlets. Finally click to the **Cancel** button to close the project wizard.


New Maven Project

New Maven project
Select an Archetype

Catalog: All Catalogs
Filter: com.hcl.dx.demo

Group Id	Artifact Id
com.hcl.dx.demo	hcl_dx_jsp_demoportlet_archetype

C:\HCL\maven_archetypes\hcl_dx_jsp_demoportlet_archetype

☑ Show the last version of Archetype only
☐ Include snapshot archetypes

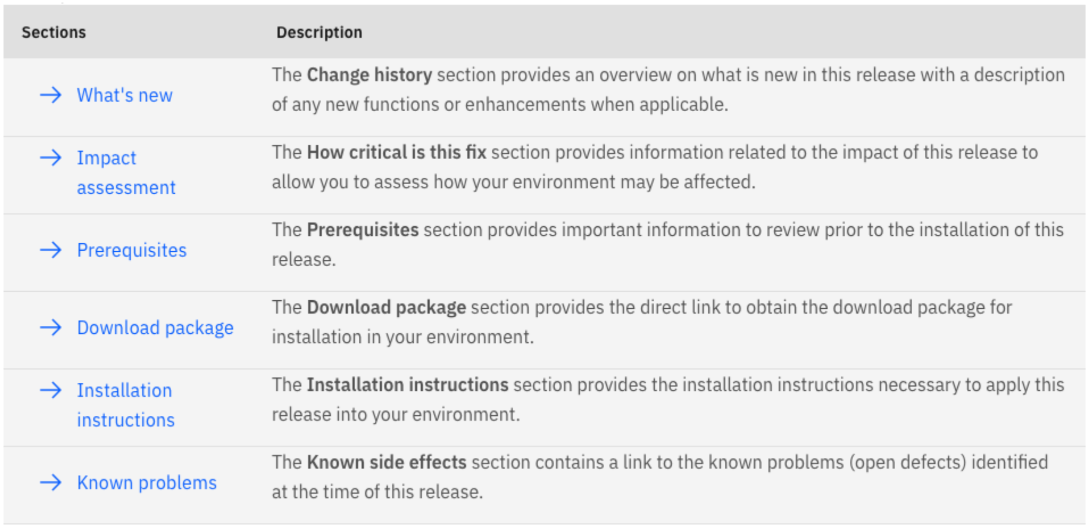
Congratulations! Your Eclipse is fully set up to work with Java for DX and prepared to use the specific DX Maven archetypes to simplify development.

Optional Part 4: DX Java Development with IBM Rational Application Developer

Optionally use IBM Rational Application Developer (RAD) to develop Java Portlets with DX.

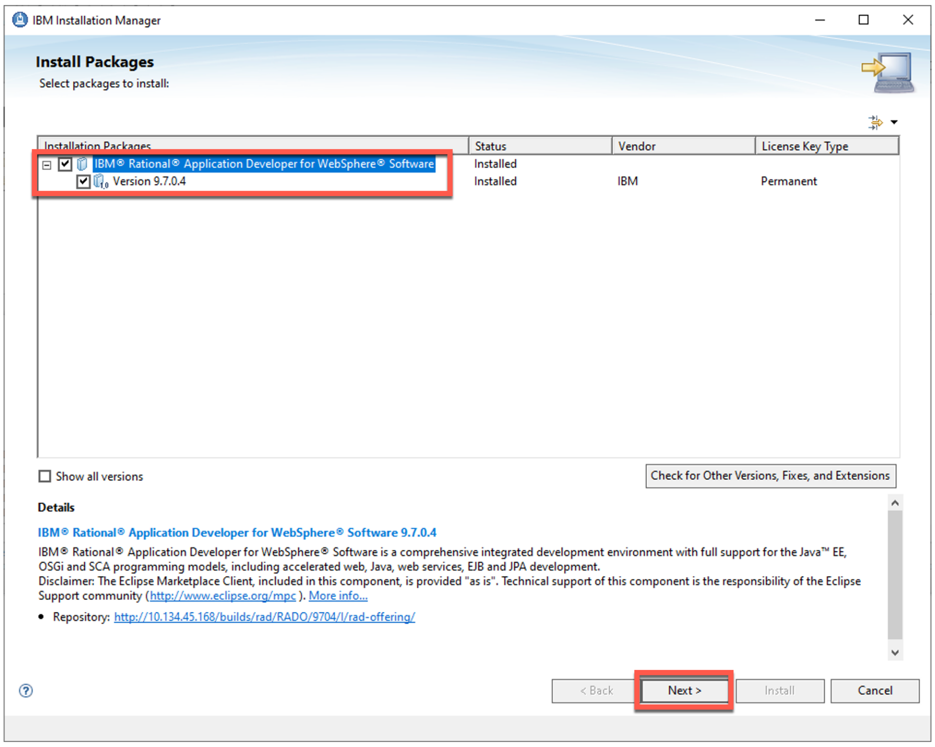
IBM RAD is a paid Eclipse based development toolkit with a lot of features. It can be used for developing different programming languages (Java, C++, Python and so on). It supports Java Portlet Development in an easy way (automatic project creation and deployment). Additional information can be found on this link: https://www.ibm.com/products/rad-for-websphere-software.

1. You may use a free trial version first before purchasing the right licenses. Download the latest IBM Rational Application Developer package and use the installation instructions from: https://www.ibm.com/support/pages/rational-application-developer-websphere-software-97. (This document is based on version 9.7.0.4. Later releases may work in the same way)



Sections	Description
→ What's new	The **Change history** section provides an overview on what is new in this release with a description of any new functions or enhancements when applicable.
→ Impact assessment	The **How critical is this fix** section provides information related to the impact of this release to allow you to assess how your environment may be affected.
→ Prerequisites	The **Prerequisites** section provides important information to review prior to the installation of this release.
→ Download package	The **Download package** section provides the direct link to obtain the download package for installation in your environment.
→ Installation instructions	The **Installation instructions** section provides the installation instructions necessary to apply this release into your environment.
→ Known problems	The **Known side effects** section contains a link to the known problems (open defects) identified at the time of this release.

2. Use the Installation Manager to install IBM Rational Application Developer. As soon as the RAD Installation Repository is included in the Installation Manager, click **Install** and select the product. An installation wizard starts showing the product version. Click **Next >**.



Installation Packages	Status	Vendor	License Key Type
IBM® Rational® Application Developer for WebSphere® Software	Installed		
Version 9.7.0.4	Installed	IBM	Permanent

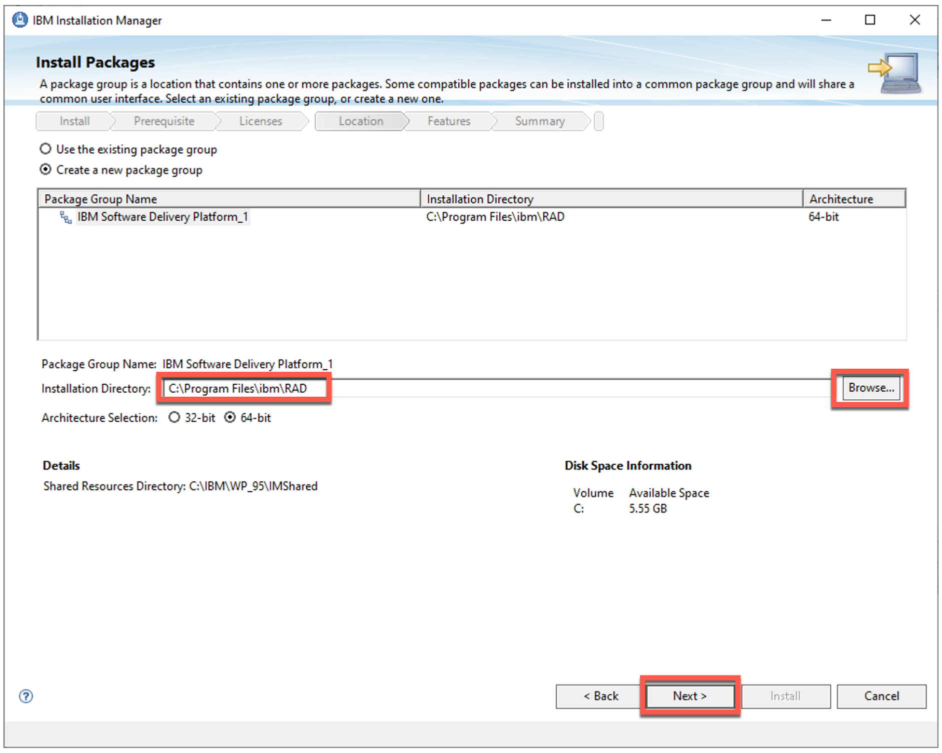
Details

IBM® Rational® Application Developer for WebSphere® Software 9.7.0.4

IBM® Rational® Application Developer for WebSphere® Software is a comprehensive integrated development environment with full support for the Java™ EE, OSGi and SCA programming models, including accelerated web, Java, web services, EJB and JPA development.
Disclaimer: The Eclipse Marketplace Client, included in this component, is provided "as is". Technical support of this component is the responsibility of the Eclipse Support community (http://www.eclipse.org/mpc). More info...

- Repository: http://10.134.45.168/builds/rad/RADO/9704/I/rad-offering/

3. Enter an **Installation Directory**, for example on Windows **C:\Program Files\ibm\RAD** and click **Next >**.



A package group is a location that contains one or more packages. Some compatible packages can be installed into a common package group and will share a common user interface. Select an existing package group, or create a new one.

Use the existing package group [✓ Create a new package group]

Package Group Name	Installation Directory	Architecture
IBM Software Delivery Platform_1	C:\Program Files\ibm\RAD	64-bit

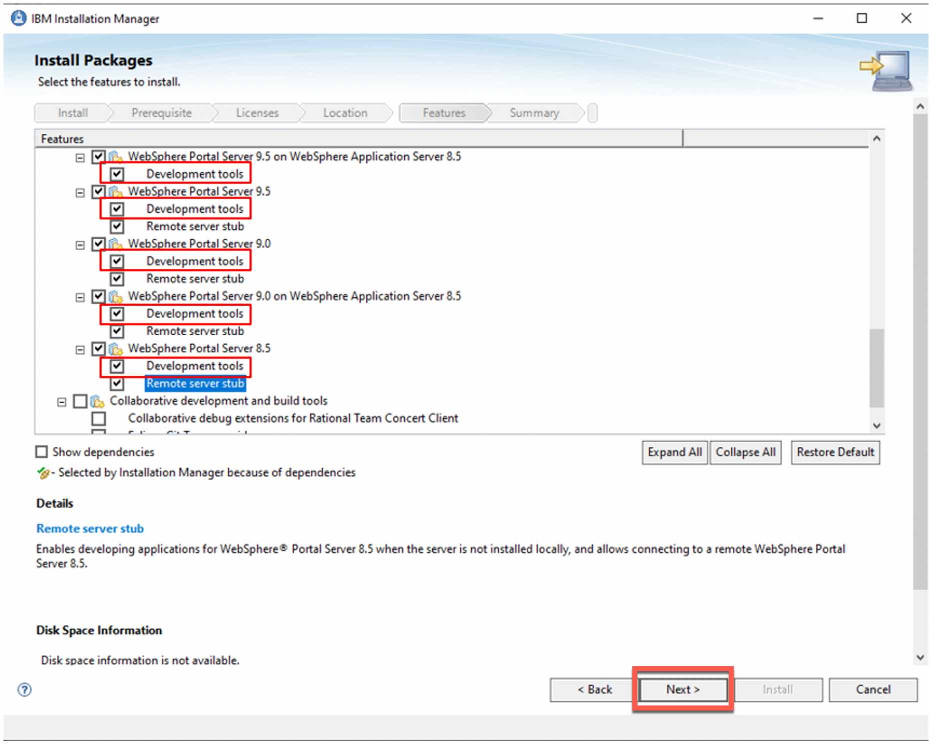
Package Group Name: IBM Software Delivery Platform_1
Installation Directory: C:\Program Files\ibm\RAD
Architecture Selection: 32-bit [✓ 64-bit]

Details
Shared Resources Directory: C:\IBM\WP_95\IMShared

Disk Space Information

Volume	Available Space
C:	5.55 GB

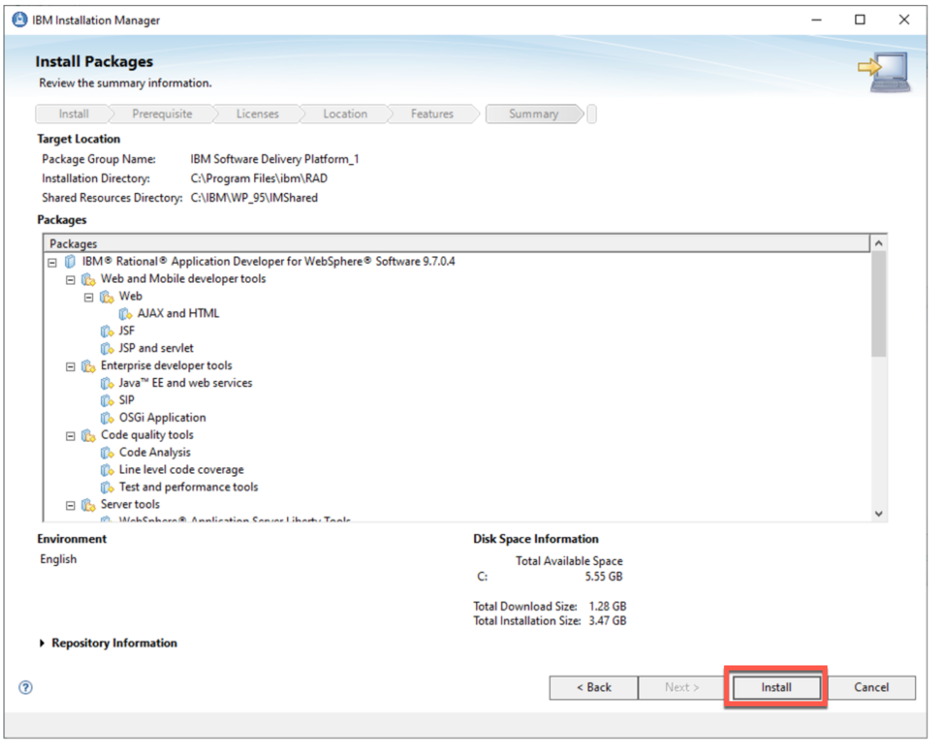
4. Then ensure all WebSphere Portal Server Features are selected to be installed, as shown. **This is very important to show up all features later while using IBM RAD to develop Java Portlets!** Click **Next >**.


IBM Installation Manager
Install Packages
Select the features to install.
Install
Prerequisite
Licenses
Location
Features
Summary
Features
☑ WebSphere Portal Server 9.5 on WebSphere Application Server 8.5
☑ Development tools
☑ WebSphere Portal Server 9.5
☑ Development tools
☑ Remote server stub
☑ WebSphere Portal Server 9.0
☑ Development tools
☑ Remote server stub
☑ WebSphere Portal Server 9.0 on WebSphere Application Server 8.5
☑ Development tools
☑ Remote server stub
☑ WebSphere Portal Server 8.5
☑ Development tools
☑ Remote server stub
☐ Collaborative development and build tools
☐ Collaborative debug extensions for Rational Team Concert Client
Show dependencies
Expand All
Collapse All
Restore Default
- Selected by Installation Manager because of dependencies
Details
Remote server stub
Disk Space Information
< Back
Next >
Install
Cancel

Enables developing applications for WebSphere® Portal Server 8.5 when the server is not installed locally, and allows connecting to a remote WebSphere Portal Server 8.5.

Disk space information is not available.

5. And then **Install**.



Target Location

Package Group Name:	IBM Software Delivery Platform_1
Installation Directory:	C:\Program Files\ibm\RAD
Shared Resources Directory:	C:\IBM\WP_95\IMShared

Environment

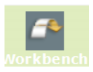
English

Disk Space Information

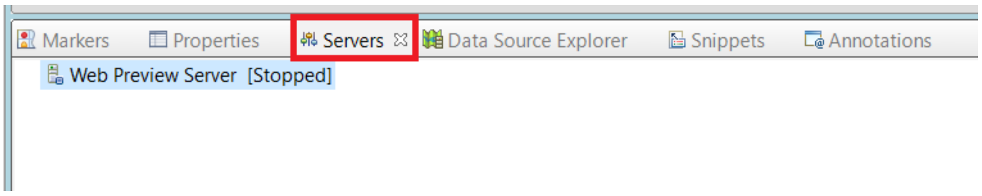
	Total Available Space
C:	5.55 GB

Total Download Size: 1.28 GB
Total Installation Size: 3.47 GB

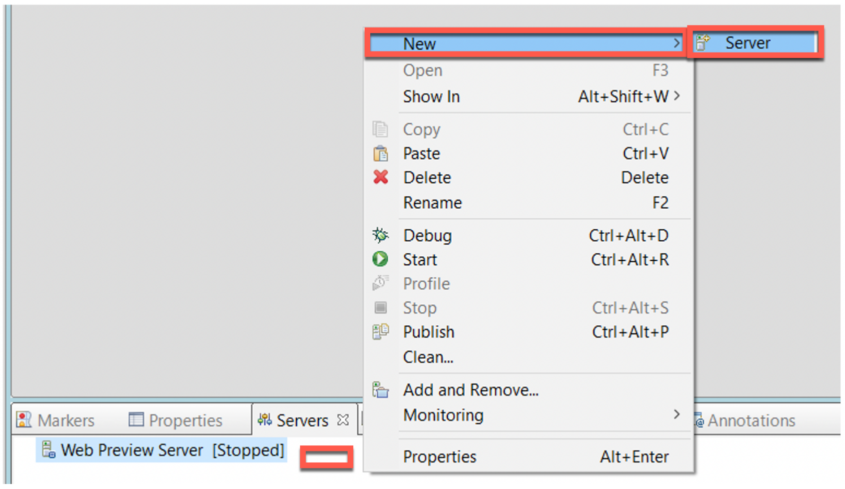
6. Then start the IBM Rational Application Developer and enter a Workspace Directory. On its start screen, click **Workbench**.



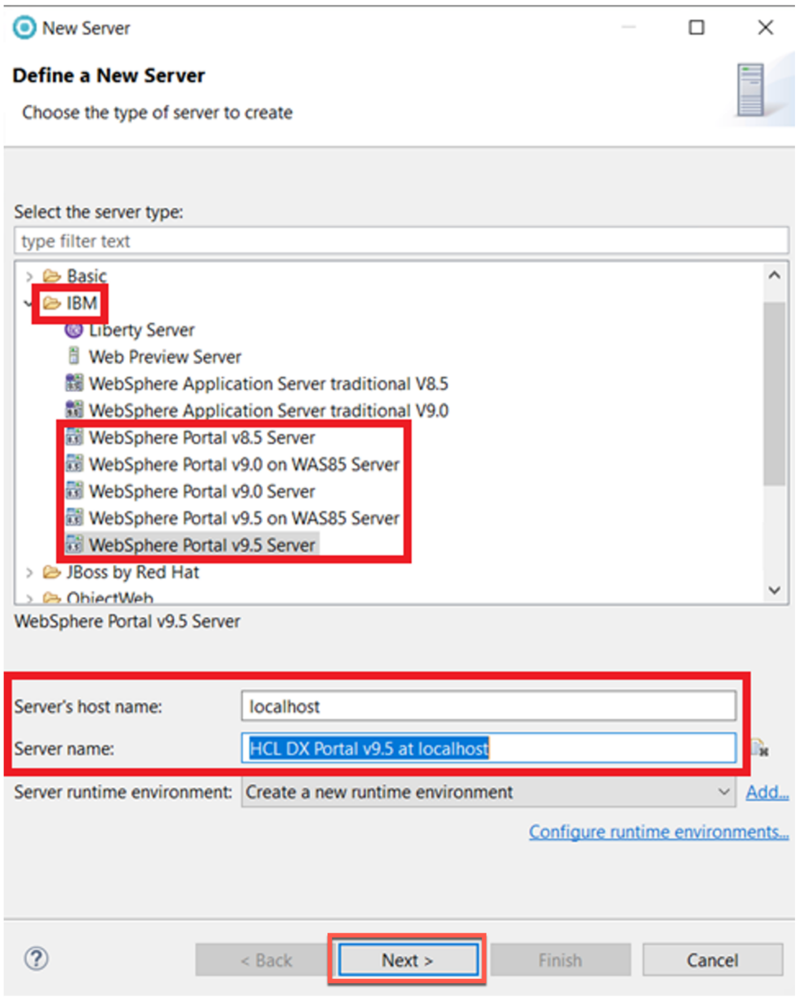
7. Then click the **Servers** tab in the lower screen area.



8. Do a right click inside this **Servers** and click **New** - **Server**.



9. Then expand the **IBM** folder. Choose your server type that matches the versions of your DX server and used WebSphere Application Server (WAS). Then enter a **Server's host name** and **Server name,** and click **Next >**. For example:



New Server

Define a New Server

Choose the type of server to create

Select the server type:

type filter text

- Basic
- IBM
  - Liberty Server
  - Web Preview Server
  - WebSphere Application Server traditional V8.5
  - WebSphere Application Server traditional V9.0
  - WebSphere Portal v8.5 Server
  - WebSphere Portal v9.0 on WAS85 Server
  - WebSphere Portal v9.0 Server
  - WebSphere Portal v9.5 on WAS85 Server
  - WebSphere Portal v9.5 Server
- JBoss by Red Hat
- ObjectWeb

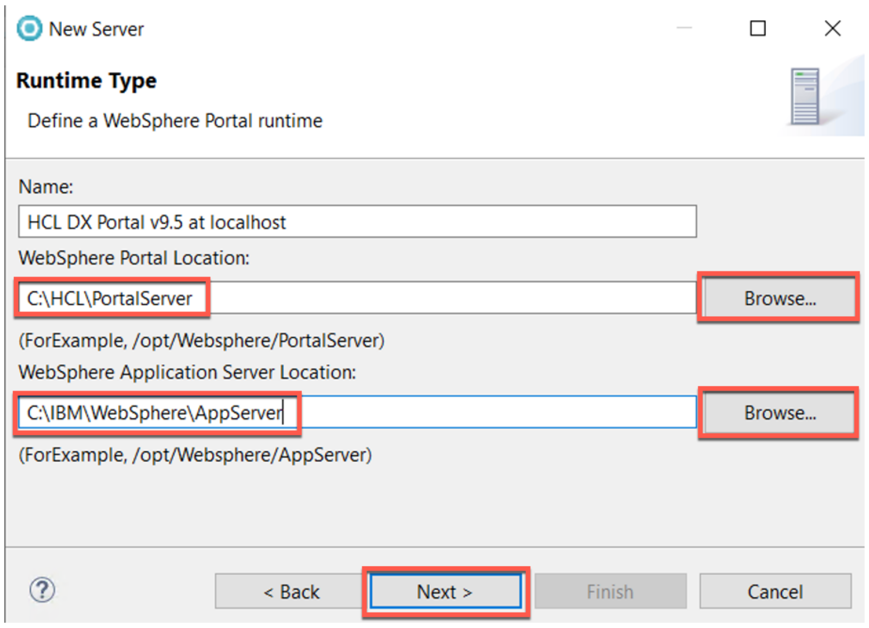
WebSphere Portal v9.5 Server

Server's host name: localhost

Server name: HCL DX Portal v9.5 at localhost

Server runtime environment: Create a new runtime environment

10. In the next window specify the installation directories of the local HCL DX and IBM WebSphere Application Server. Click **Next >**. For example:



New Server

Runtime Type

Define a WebSphere Portal runtime

Name: HCL DX Portal v9.5 at localhost

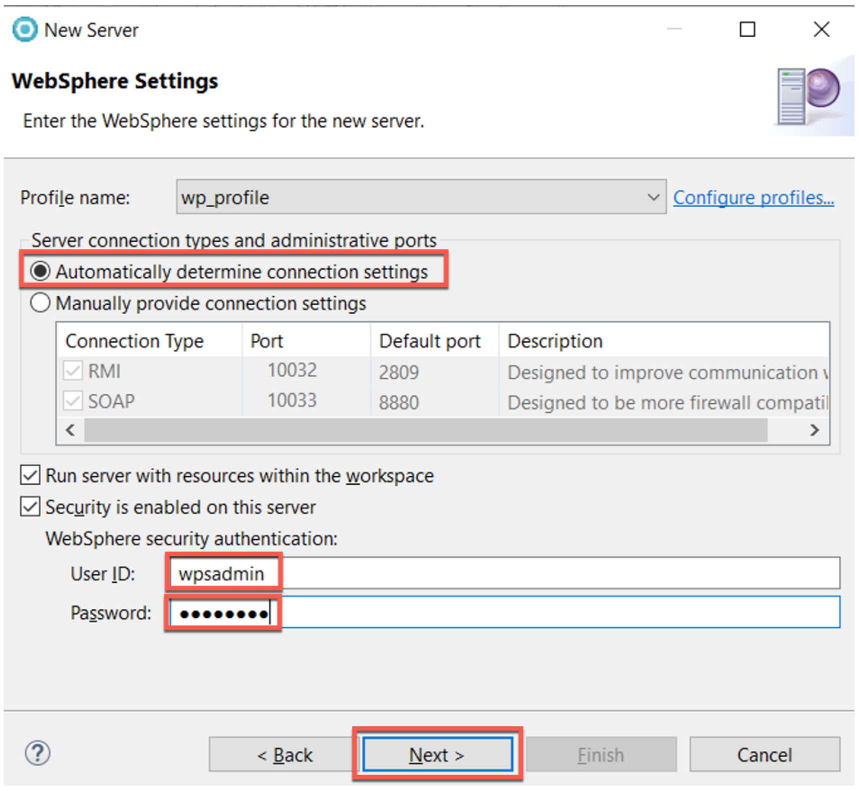
WebSphere Portal Location: C:\HCL\PortalServer

(ForExample, /opt/Websphere/PortalServer)

WebSphere Application Server Location: C:\IBM\WebSphere\AppServer

(ForExample, /opt/Websphere/AppServer)

11. In the profile configuration window make sure that the correct Portal server profile will be selected. (Default is: **wp_profile**). Keep the **Server connection type and administrative ports** set to **Automatically determine connection settings**. Then enter a **User ID** and **Password** for the WebSphere Application Server administration user (by default this is wpsadmin/wpsadmin) and click **Next >**.



New Server

WebSphere Settings

Enter the WebSphere settings for the new server.

Profile name: wp_profile

Server connection types and administrative ports

(●) Automatically determine connection settings

( ) Manually provide connection settings

Connection Type	Port	Default port	Description
✓ RMI	10032	2809	Designed to improve communication v
✓ SOAP	10033	8880	Designed to be more firewall compati

[✓] Run server with resources within the workspace

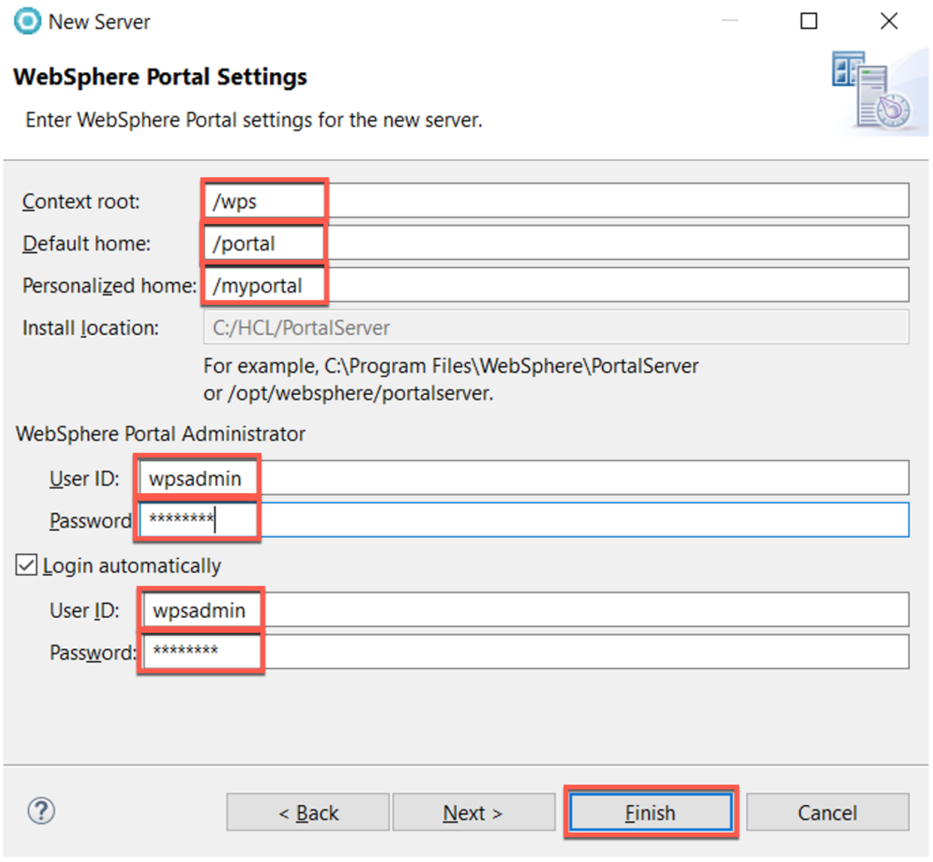
[✓] Security is enabled on this server

WebSphere security authentication:

User ID: wpsadmin

Password: ••••••••

12. In the next window make sure that the **Context root**, **Default home** and **Personalized home** are set correctly along with the correct user IDs and passwords (again default is wpsadmin/wpsadmin). The default settings for the context root, default home and personalized home should already match, if no customization is done yet on that profile. Click **Finish**.


New Server

WebSphere Portal Settings

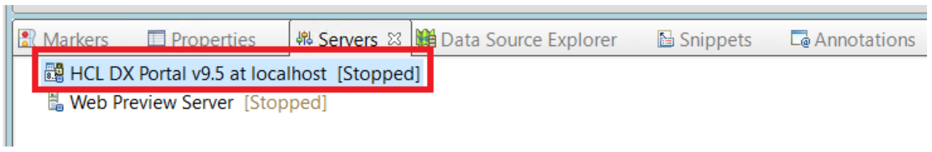
Enter WebSphere Portal settings for the new server.

Context root: /wps
Default home: /portal
Personalized home: /myportal
Install location: C:/HCL/PortalServer
For example, C:\Program Files\WebSphere\PortalServer or /opt/websphere/portalserver.

WebSphere Portal Administrator
User ID: wpsadmin
Password: ********

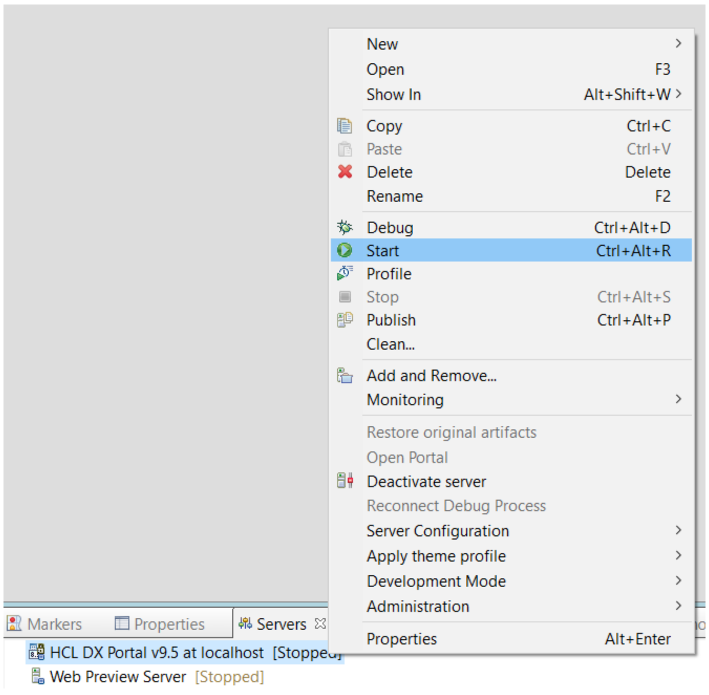
☑ Login automatically
User ID: wpsadmin
Password: ********

13. In the **Servers** tab of the IBM Rational Application Developer the server name **HCL DX Portal v9.5 at localhost** should now be listed.

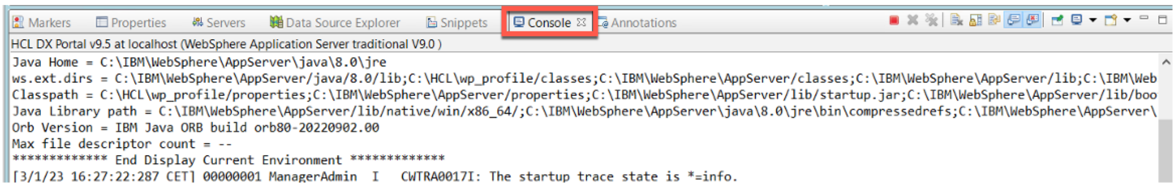


HCL DX Portal v9.5 at localhost [Stopped]
Web Preview Server [Stopped]

14. Use a right click on the Server name to show the server options, like start and stop the server.



15. As soon as the server is started, a **Console** tab is opened to show up the log entries.



HCL DX Portal v9.5 at localhost (WebSphere Application Server traditional V9.0)

```
Java Home = C:\IBM\WebSphere\AppServer\java\8.0\jre
ws.ext.dirs = C:\IBM\WebSphere\AppServer/java/8.0/lib;C:\HCL\wp_profile/classes;C:\IBM\WebSphere\AppServer/classes;C:\IBM\WebSphere\AppServer/lib;C:\IBM\Web
Classpath = C:\HCL\wp_profile/properties;C:\IBM\WebSphere\AppServer/properties;C:\IBM\WebSphere\AppServer/lib/startup.jar;C:\IBM\WebSphere\AppServer/lib/boo
Java Library path = C:\IBM\WebSphere\AppServer/lib/native/win/x86_64/;C:\IBM\WebSphere\AppServer\java\8.0\jre\bin\compressedrefs;C:\IBM\WebSphere\AppServer\
Orb Version = IBM Java ORB build orb80-20220902.00
Max file descriptor count = --
************* End Display Current Environment *************
[3/1/23 16:27:22:287 CET] 00000001 ManagerAdmin  I   CWTRA0017I: The startup trace state is *=info.
```

IBM Rational Application Developer already includes extensions to create Java Portlets in an easy way. There are no additional steps that needed to be prepared.

You have successfully set up IBM Rational Application Developer to work with HCL DX server.

Conclusion

Using this lab tutorial, you have learned how to set up Microsoft Visual Studio Code (and optionally Eclipse or IBM Rational Application Developer) to work with Java artifacts for HCL Digital Experience. This included installing the right Java JDK, Maven, and installation of Maven archetypes.

You are now ready to start creating any of these Java artifacts.

Resources

Refer to the following resources to learn more:
HCL Digital Experience Home - https://hclsw.co/dx

HCL Digital Experience on HCL SoFy - https://hclsofy.com/

HCL Software - https://hclsw.co/software

HCL Product Support - https://hclsw.co/product-support

HCL DX Product Documentation - https://hclsw.co/dx-product-documentation

HCL DX Support Q&A Forum - https://hclsw.co/dx-support-forum

HCL DX Video Playlist on YouTube - https://hclsw.co/dx-video-playlist

HCL DX Product Ideas - https://hclsw.co/dx-ideas

HCL DX Product Demos - https://hclsw.co/dx-product-demo

HCL DX Did You Know? Videos - https://hclsw.co/dx-dyk-videos

HCL DX GitHub - https://hclsw.co/dx-github

Legal statements

Disclaimers