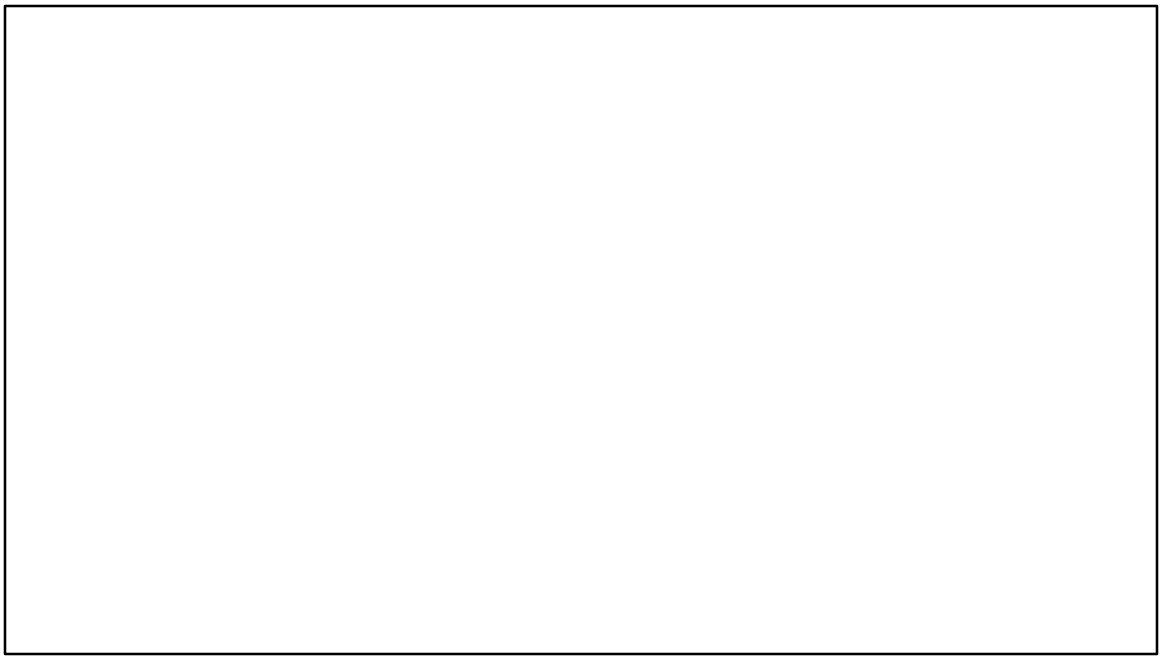


Setup Design Studio

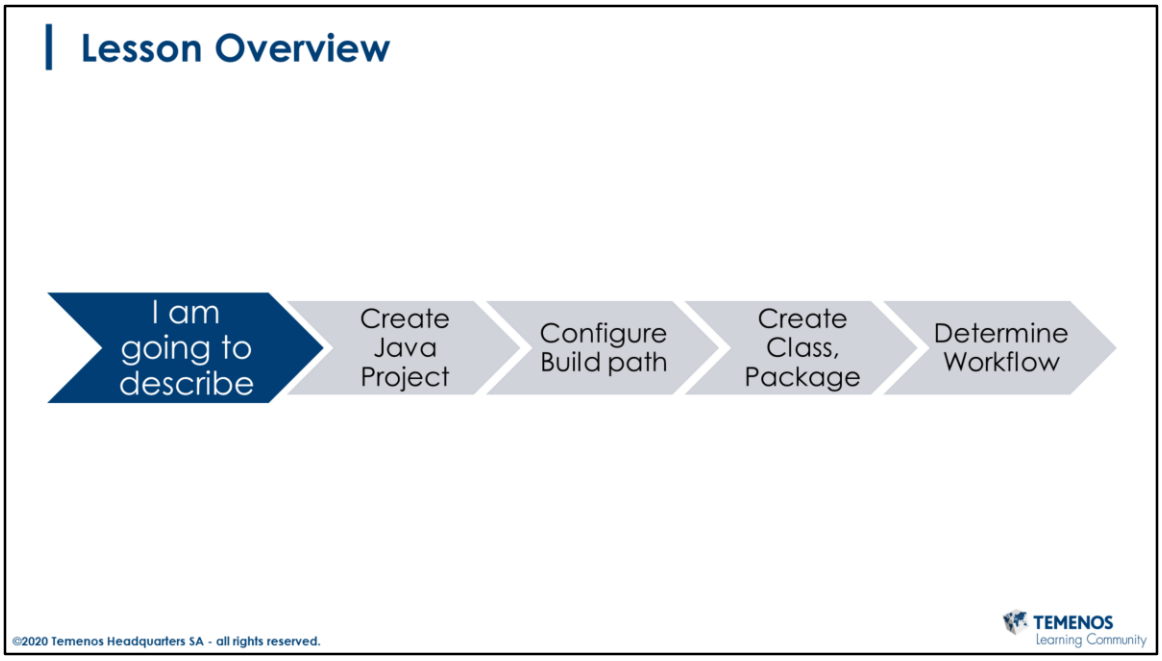
Transact Extensibility for Java (English)

2020 Q1

Copyright Notice

©2020 Temenos Headquarters SA - all rights reserved.

TEMENOS Learning Community

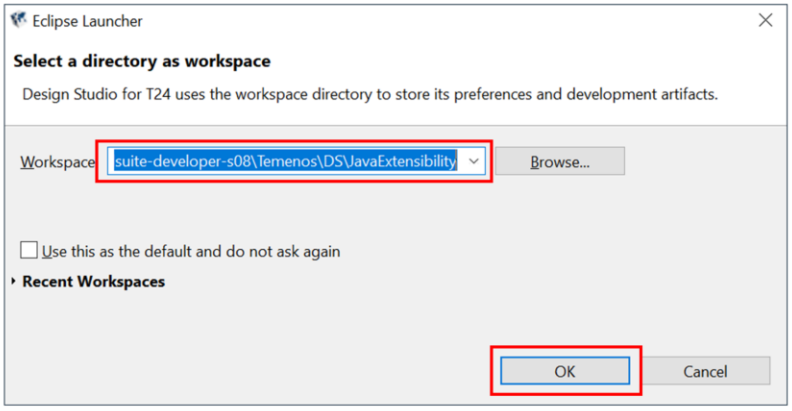


I am going to describe

- Create Java Project
- Configure Build path
- Create Class, Package
- Determine Workflow

Setup Design Studio

- Launch Design Studio from the UTP package
- Create a new workspace

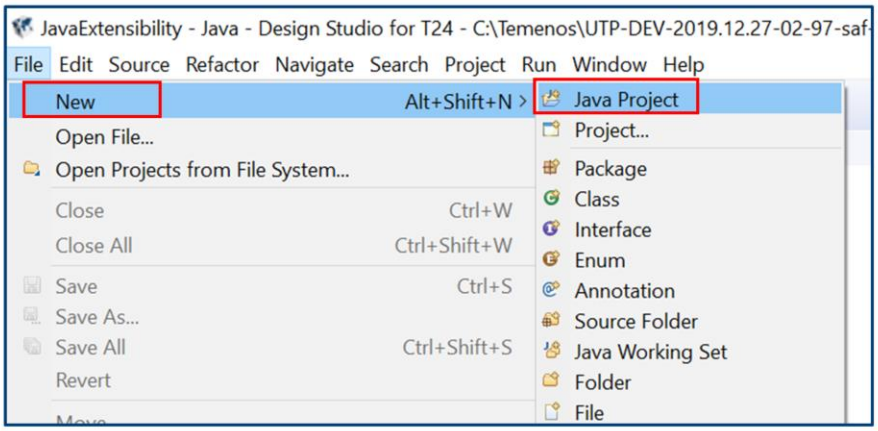


TEMENOS Learning Community

Launch Design Studio from the UTP package. Design Studio can be found under the UTP Package DS folder.
Create a workspace with a valid name.
Click ok to open the workspace

Create Java Project

- From File Menu -> New -> Java Project



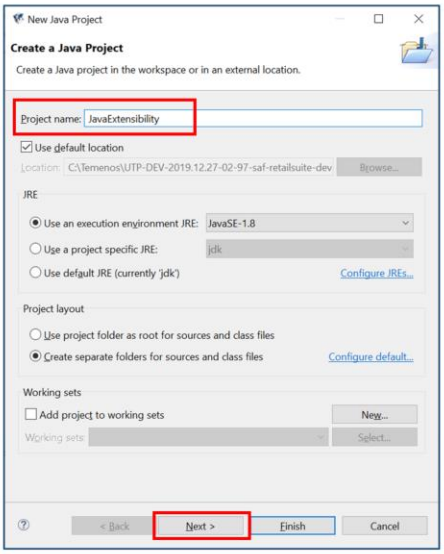
TEMENOS Learning Community

After opening the workspace. Lets create the Java Project
Click on File Menu -> New -> Java Project

This will open the new project wizard

Create Java Project

- Name the Project
- Select the JRE Runtime
- Click Next



TEMENOS Learning Community

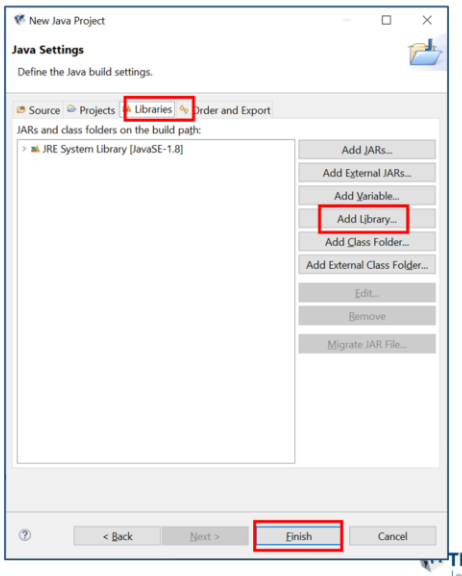
In the New Project Wizard,

Enter the name of the project
Choose the JRE for the project
And Click Next

Configure Build path

- It is required to add the Transact Library and TAFJ Library in Build Path to make you of core APIs

- Select Libraries Tab
- Click on Add Library



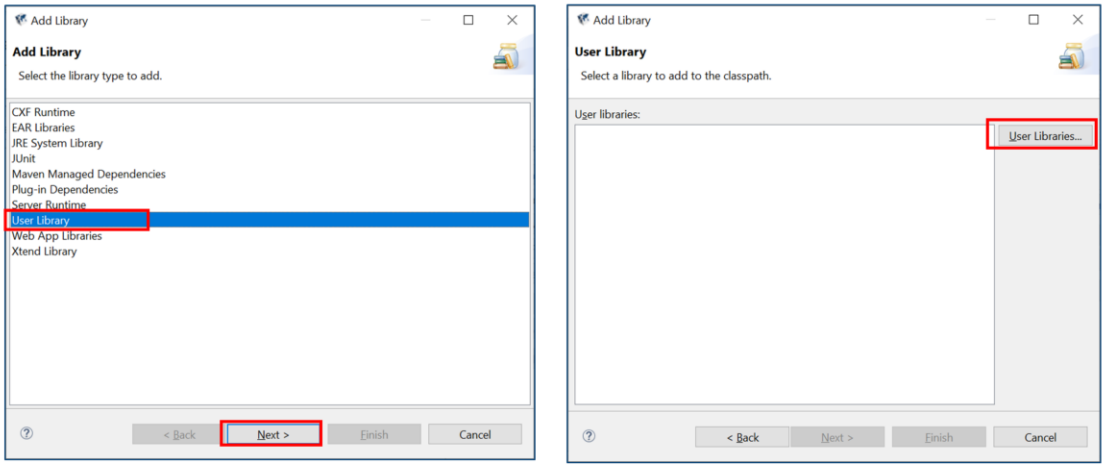
TEMENOS Learning Community

It is required to add the T24 Library and TAFJ Library in Build Path to make you of core APIs

Under the Libraries tab,

Click on “Add Library”

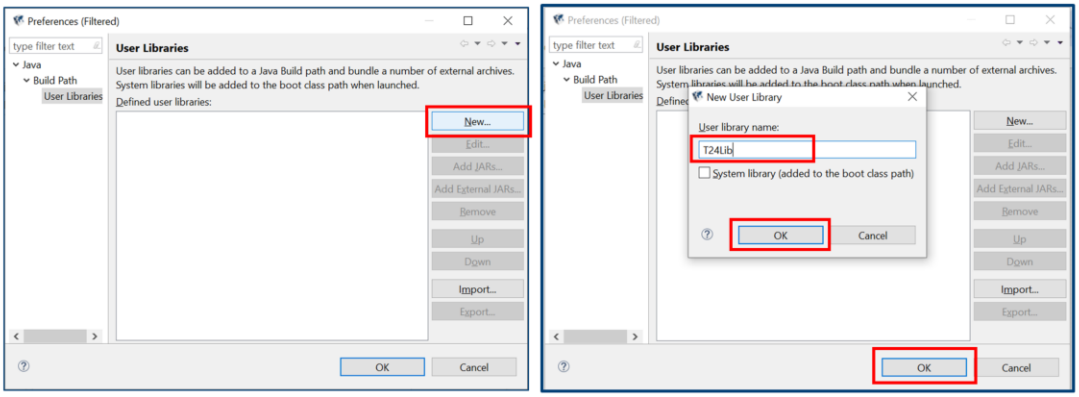
Add User Library



TEMENOS Learning Community

Select User Library. This will open “Add Library” Wizard.
From the “Add Library” Wizard, Select User Libraries

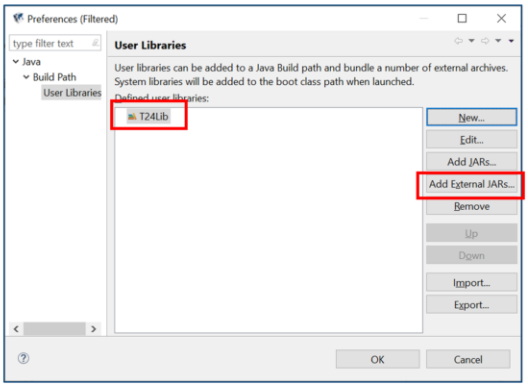
Add Transact Lib as User Library

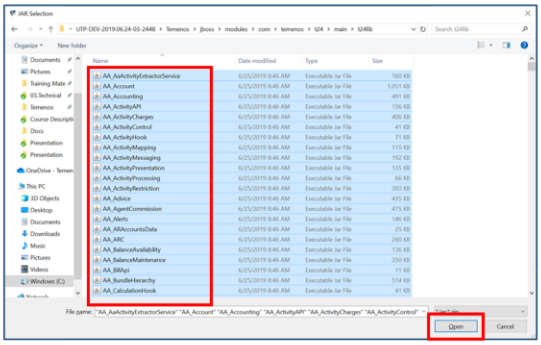


TEMENOS Learning Community

From the Preferences Wizard, Select New
Name the Library. In Our case we are going to add the T24 Lib as a reference.
Click ok to create the library

Add T24 Lib as User Library



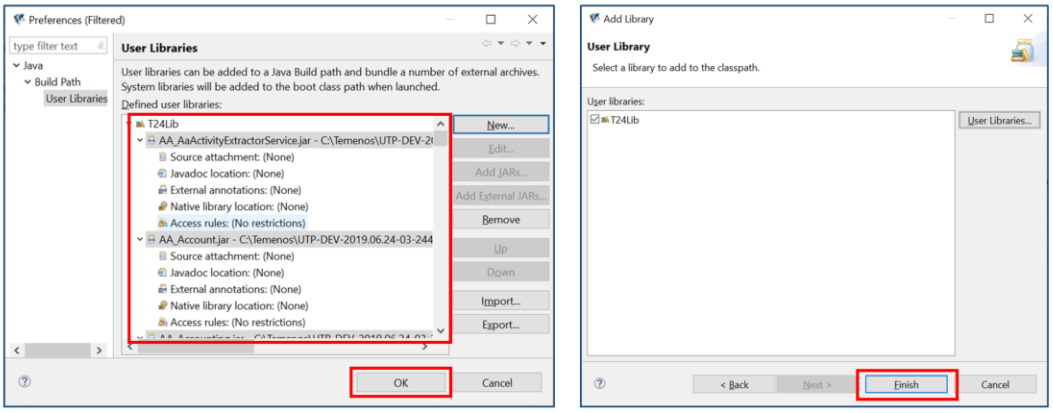


TEMENOS Learning Community

Select the User Library created just now
Select Add External JARs
From the file explorer, navigate to C:\Temenos\UTP-DEV-2020.03.01-02-240-saf-retailsuite-developer-s08\Temenos\jboss\modules\com\temenos\t24\main\t24lib

Select all the JARs
Select Open

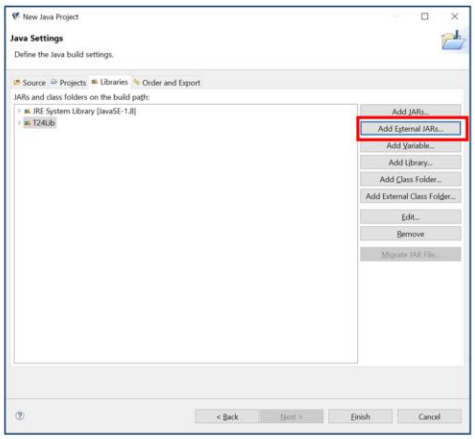
Added T24 libraries

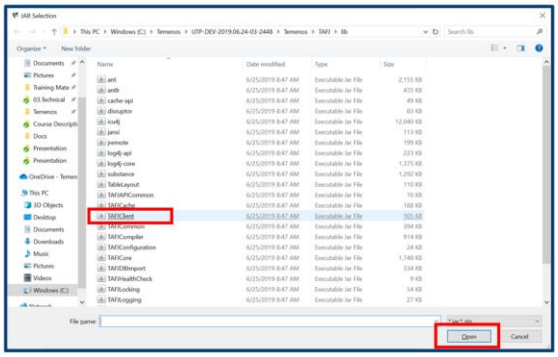


TEMENOS Learning Community

All the JARs are added to the library
Click ok to close the wizard

Add TAFJ Lib as External JARs

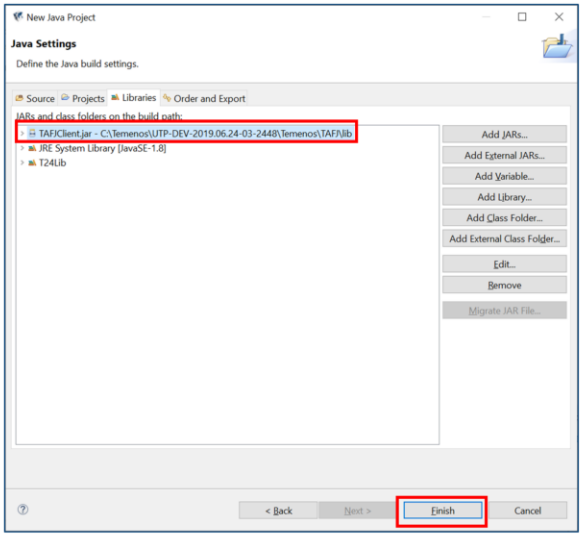


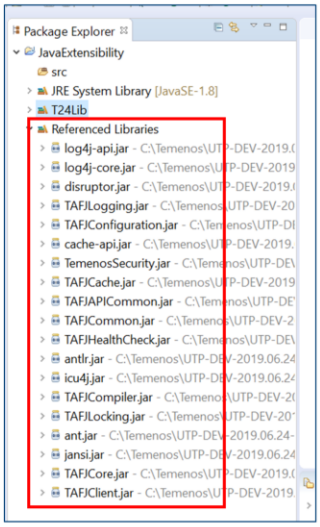


TEMENOS Learning Community

From the New Project Wizard,
Select Add Extenal Jars
From the file explorer, Navigate to C:\Temenos\UTP-DEV-2020.03.01-02-240-saf-retailsuite-developer-s08\Temenos\TAFJ\lib
Select TAFJClient.jar
Click ok Open

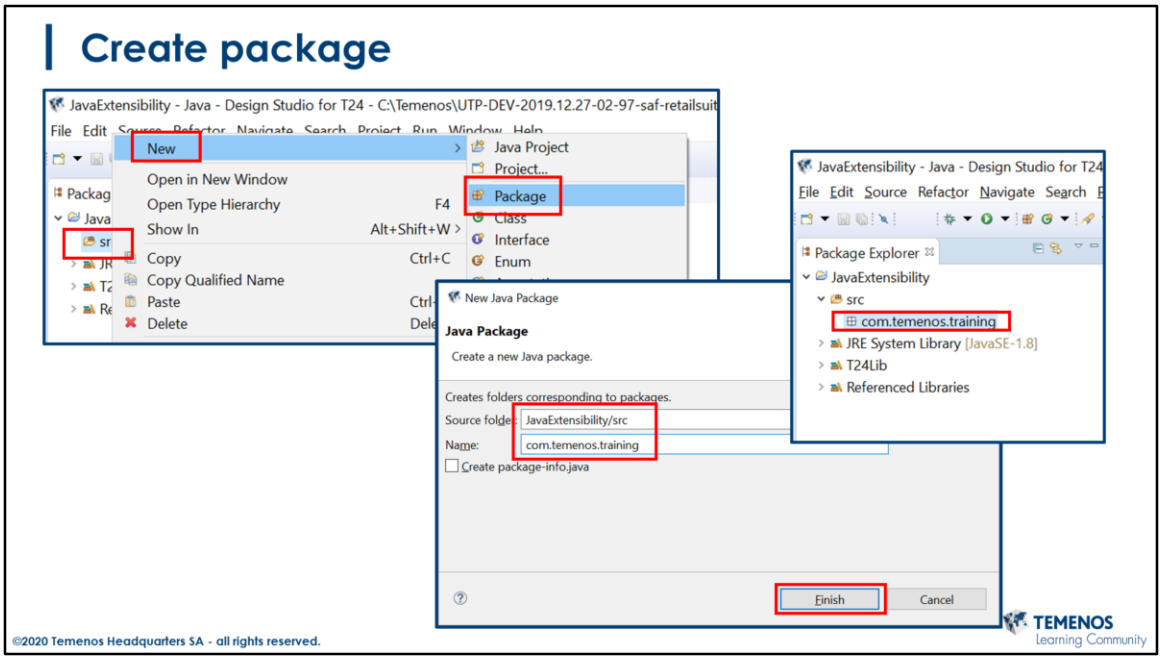
Added TAFJ Lib



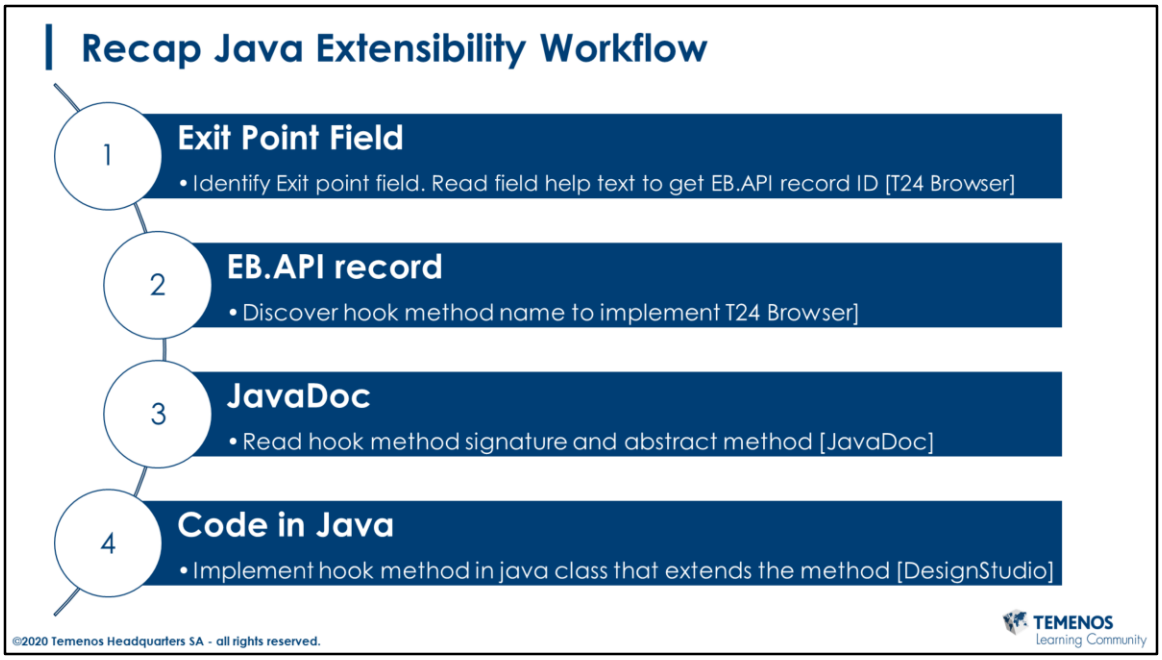


TEMENOS Learning Community

Click Finish to add the reference library



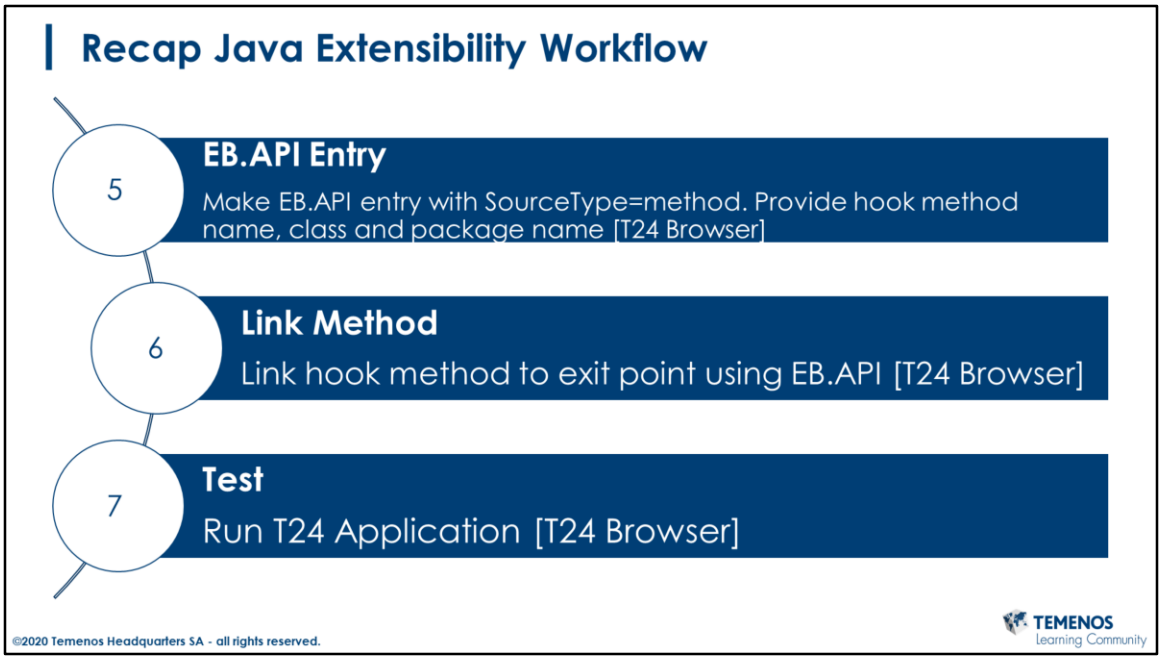
Add a new Java Package by selecting src folder
Right click on src folder
Select New -> Package
Name the package with java naming convension
Click finish
New package created.



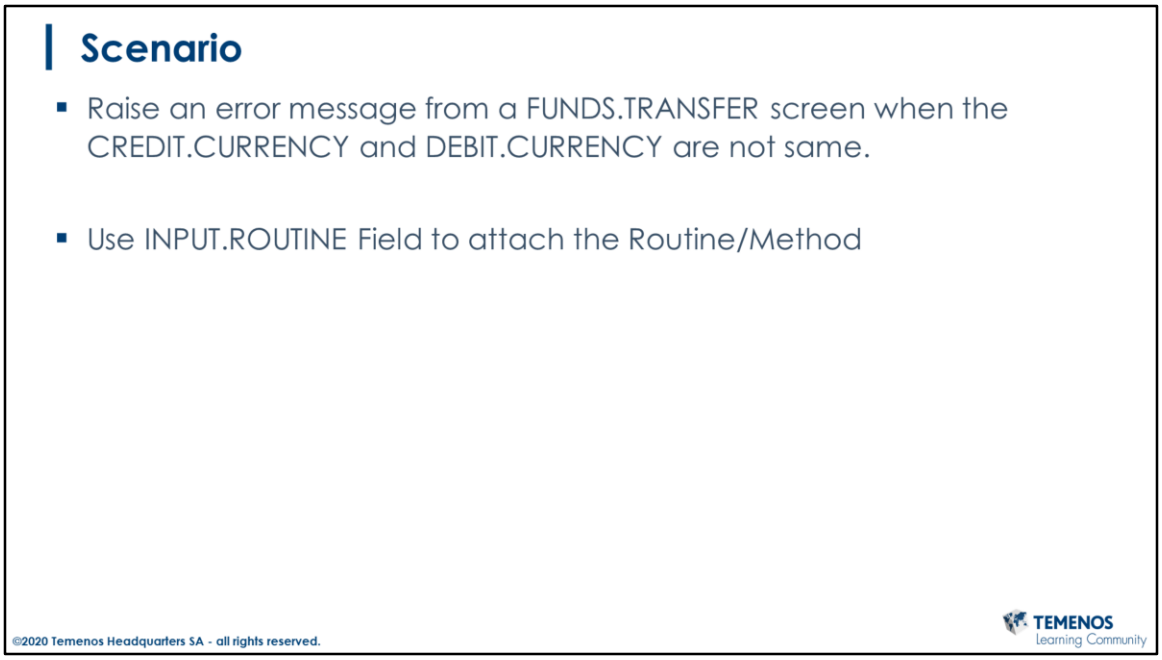
Recap of Java Extensibility Workflow

Java Extensibility workflow,

1. Identify the Exit Point field that where do you want to attach your logic. Read the help text to get EB.API record ID of the core invoker
2. Discover the hoot method name to implement. Read the EB.API
3. Read hook method signature and abstract method from the Java document.
4. Implement hook method in java class that extends the method using Design Studio/Eclipse



5. Make an entry in EB.API with the source type as method. Provide hook method name, class and package name. This can be done using the T24 Browser
6. Link Hook method to exit point field using EB.API. Example: Attach the EB.API id to the INPUT.ROUTINE Field in Version application
7. Run the T24 Application



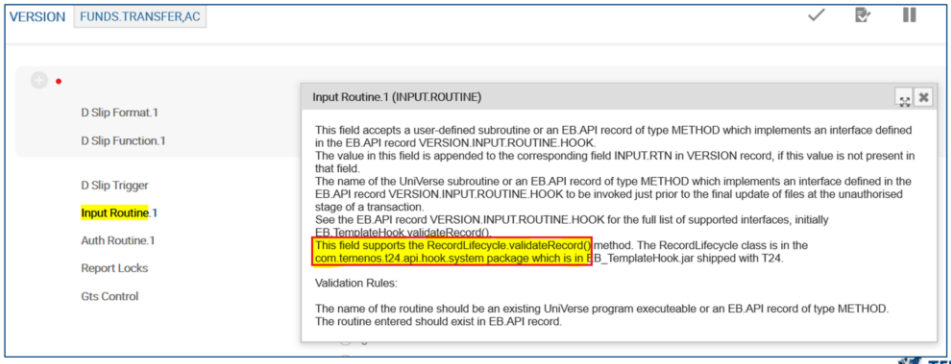
Example,
Raise an error message from a FUNDS.TRANSFER Screen when the CREDIT.CURRENCY and DEBIT.CURRENCY are not same.

Use INPUT.ROUTINE Field to attach the Routine/Method

Determine the Exit Point, Super class and Method

- Identify the exit points in the screen to attach hook routines.

  In our scenario, lets assume that INPUT.ROUTINE
- The help text for the exit point field tells you what is the package name, Class name and Method can be overridden



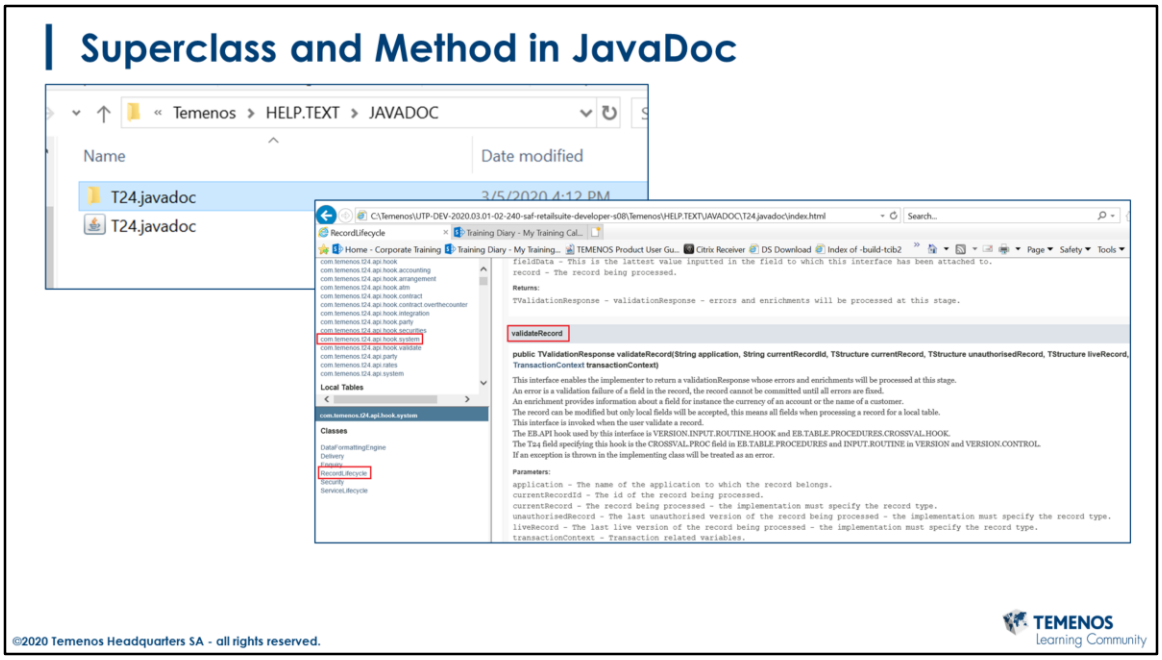
TEMENOS Learning Community

Determine the Exit Point, Super class and Method,

Identify the exit points in the screen to attach hook routines.

In our scenario, lets assume that INPUT.ROUTINE

The help text for the exit point field tells you what is the package name, Class name and Method can be overridden



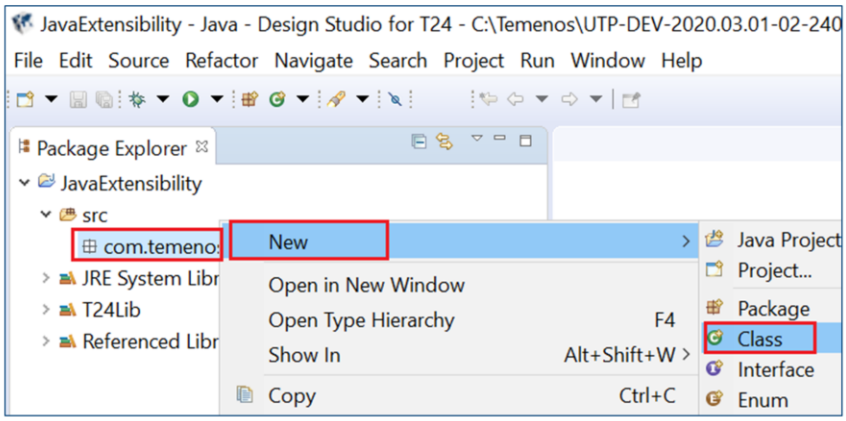
Identify Superclass and Method in JavaDoc,

Java Doc can be found under, C:\Temenos\UTP-DEV-2020.03.01-02-240-saf-retailsuite-developer-s08\Temenos\HELP.TEXT\JAVADOC

Select the package from the packages window
Once you select the package, available classes are listed below in the “All Classes” window
Select the Class from All classes window.

On the right hand side in the main window we can see the Summary and Detail of the class

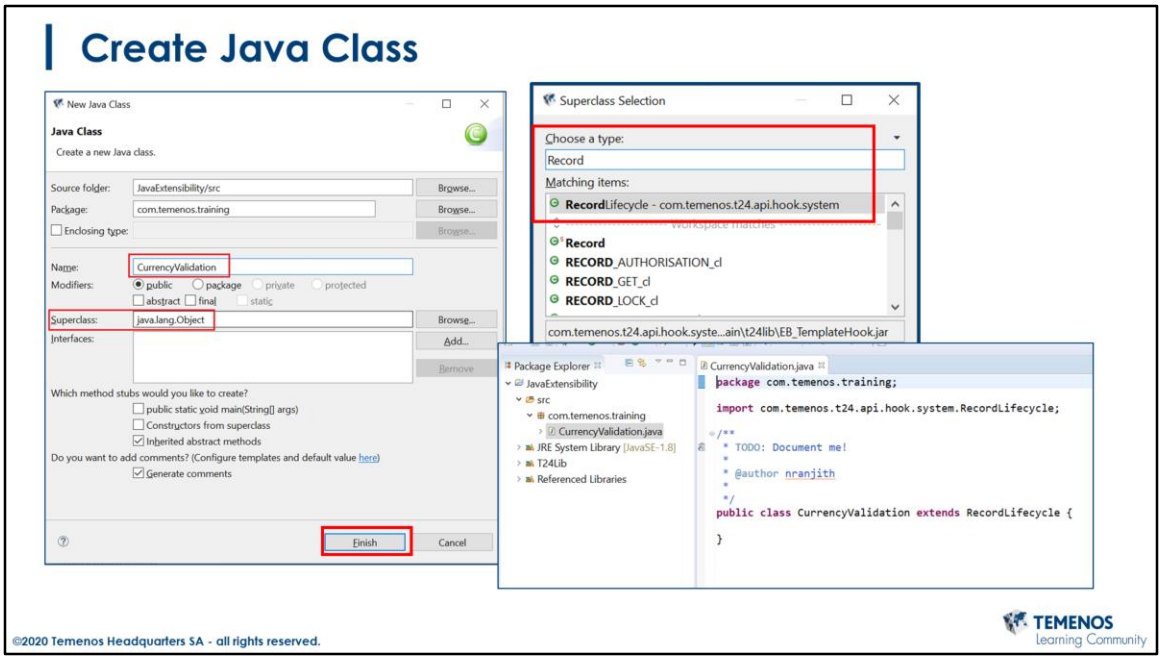
Create Java Class



TEMENOS Learning Community

Create Java Class,

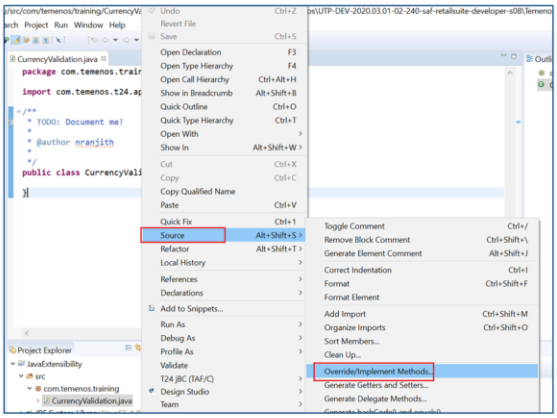
Select the package from the package explorer
Right click on the package -> New -> Class

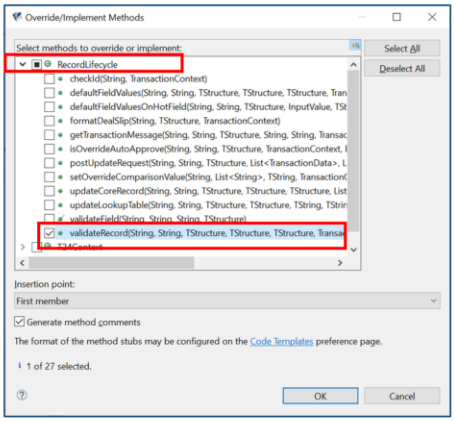


New Java Class wizard will appear,

Enter the valid name to the class
Select the appropriate super class (identified in the previous slide), by selecting browse
Click finish to create the class

Override or Implement Methods





TEMENOS Learning Community

Override or Implement Methods,

By right clicking on the class ->select source -> Override / Implement class we can see the available methods to implement.

Select the methods you want to implement (Methods identified in the previous stage) and say Ok.

Override or Implement Methods

*CurrencyValidation.java

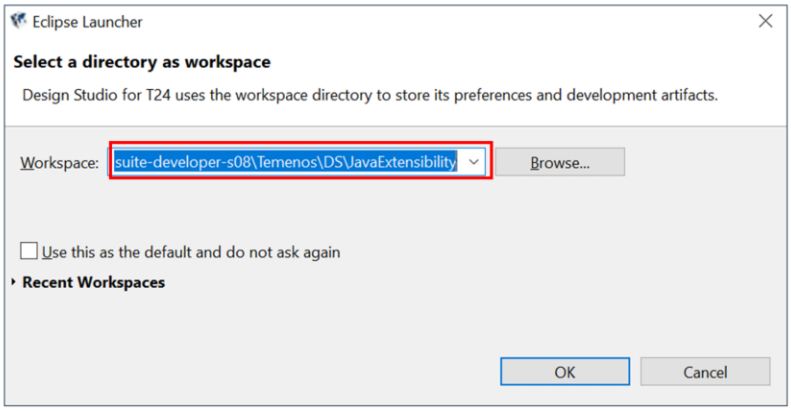
```
package com.temenos.training;

import com.temenos.api.TStructure;
import com.temenos.api.TValidationResponse;
import com.temenos.t24.api.complex.eb.templatehook.TransactionContext;
import com.temenos.t24.api.hook.system.RecordLifecycle;

/**
 * TODO: Document me!
 *
 * @author nranjith
 *
 */
public class CurrencyValidation extends RecordLifecycle {

    @Override
    public TValidationResponse validateRecord(String application, String currentRecordId, TStructure currentRecord,
            TStructure unauthorisedRecord, TStructure liveRecord, TransactionContext transactionContext) {
        // TODO Auto-generated method stub
        return super.validateRecord(application, currentRecordId, currentRecord, unauthorisedRecord, liveRecord,
                transactionContext);
    }

}
```


TEMENOS Learning Community

Method validateRecord is overwritten in the above example

Practice 3.1

- Launch Design Studio and Create a workspace for Java Programming

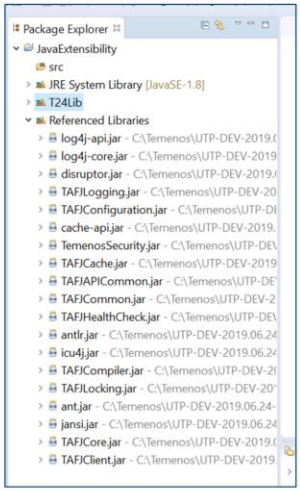


TEMENOS Learning Community

Launch Design Studio and Create a workspace for Java Programming

Practice 3.2

- Create a Java Project and add T24 Lib and TAFJClient.jar as libraries

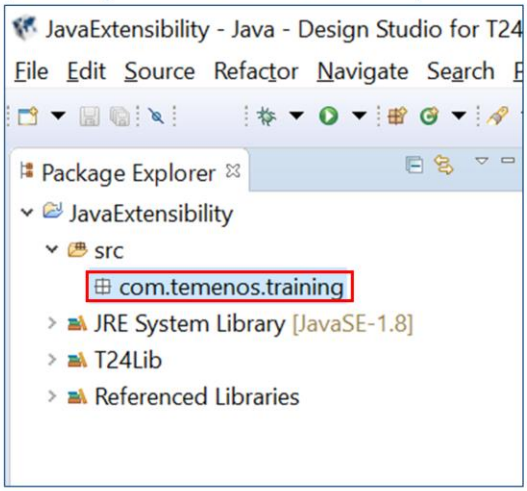


TEMENOS Learning Community

Create a Java Project and add T24 Lib and TAFJClient.jar as libraries

Practice 3.3

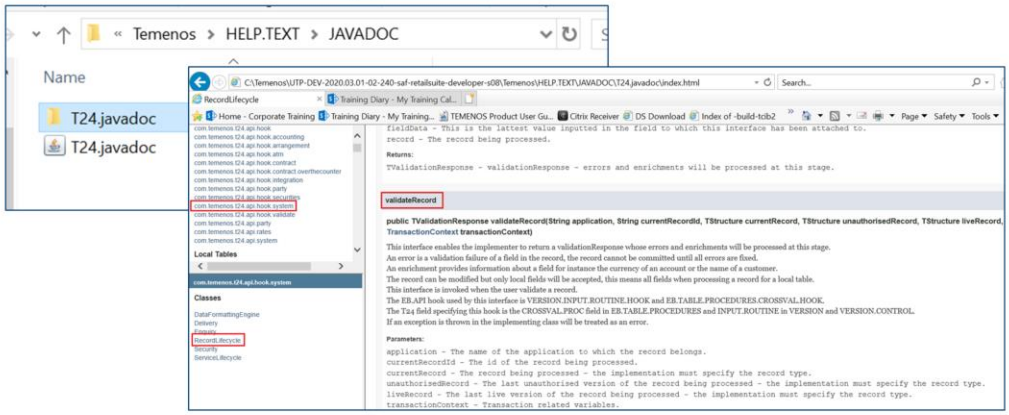
- Create a java package under the Java Project



TEMENOS Learning Community

Practice 3.4

- Identify the Super Class and Abstract method for the field INPUT.ROUTINE in Version Application

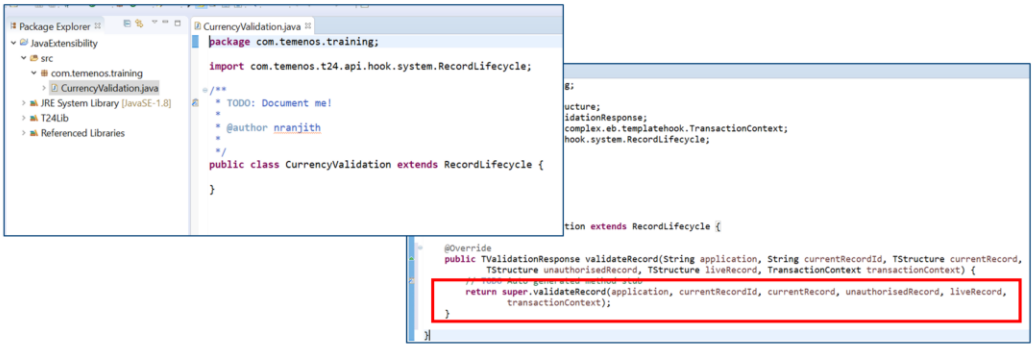


TEMENOS Learning Community

Identify the Super Class and Abstract method for the field INPUT.ROUTINE in Version Application

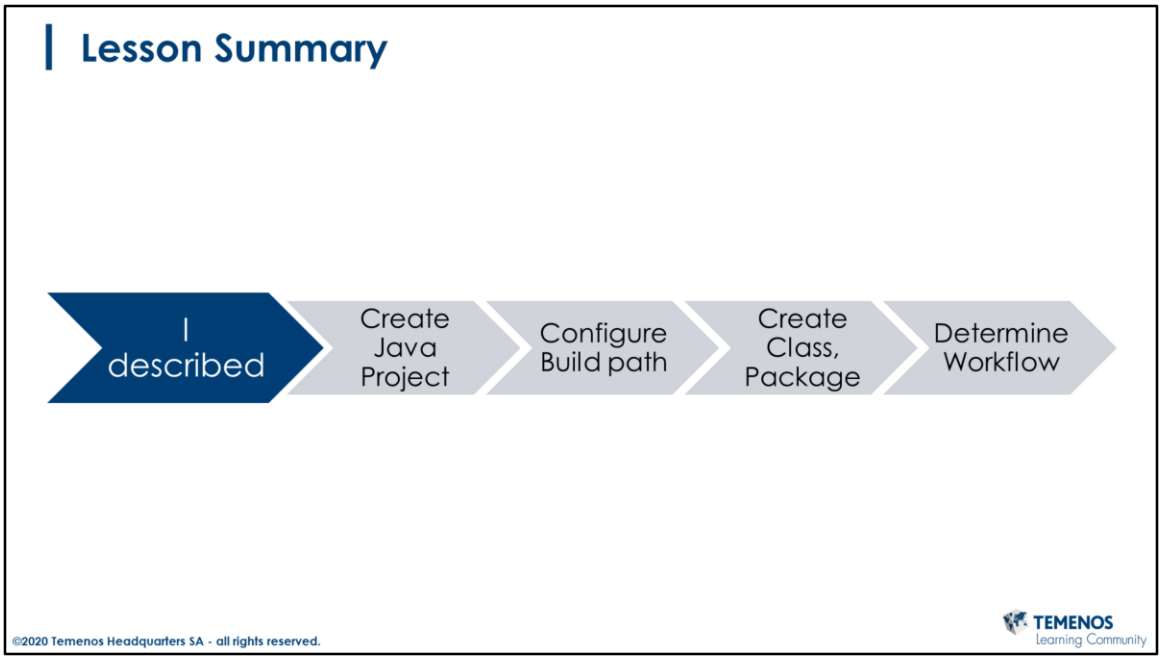
Practice 3.5

- Create a Java class under the package created and extend the Super class identified in the previous exercise
- Override the method identified in the previous exercise



TEMENOS Learning Community

Create a Java class under the package created and extend the Super class identified in the previous exercise
Override the method identified in the previous exercise



I described

- Create Java Project
- Configure Build path
- Create Class, Package
- Determine Workflow

TEMENOS

Learning Community

thank.you

tlc.temenos.com