T24 Extensibility for Java

TEMENOS
Learning Community

Lesson Overview

I am going to describe

T24 Extensibility in java

Customising T24 using Java hook routines

TEMENOS
Learning Community

Your Course » Objective and Learning Outcomes

This course will introduce you to:

- T24 Extensibility in Java

In particular you will be able to:

- Understand T24 extensibility and customisation using Java
- Setup and Use Design Studio to Create and execute VERSION, ENQUIRY, SERVICE hooks
- Create and execute hooks attached to a local application
- Debug the Java code in Design Studio

Your Course » Timetable

Day 1

- Introduction to T24 Extensibility in Java
- Setup Design Studio
- VERSION hooks
- DEBUG the Java code in Design Studio

Day 2

- ENQUIRY hooks
- SERVICE hooks
- Local application using EB.TABLE.DEFINITION

TEMENOS Learning Community

Lesson 1. Introduction

T24 Extensibility for Java

TEMENOS
Learning Community

Agenda - T24 Extensibility in Java

- Why do we need T24 Extensibility in Java ?
- APIs and Hooks
- T-Types
- DataAccess
- Complex classes
- Setup Design Studio

TEMENOS
Learning Community

Why do we need T24 Extensibility in Java ?

- Scalability and ownership
  - There are more java developers than jBC developers
  - Banks can use their own developers
- Simplify the T24 APIs
  - The developer should not need to know about STORE.END.ERROR, R.NEW or the different ways of storing an amount field in T24
- Governance
  - We can use the framework to protect t24. for e.g. prevent a select on a large table that can lead to performance issue for all users

Usage Scenarios

Java extensibility can be used for

- Updating User Defined Tables.
- Validating transaction IDs entered.
- Auto populating fields.
- Cross-validating records and fields.
- Altering and/or defaulting other field values in the record based on a field value.
- Updating local reference fields.
- Raising errors / overrides.
- Defining services / COB jobs.
- Combining data from different applications for a report

Prerequisites

- JD product must be installed

SPF SYSTEM

Products.129	JD	Java Development

SPF SYSTEM

Last Cache Reset Period.1	
Enquiry Sample Size	
Extensible Customisation	Yes No
Ext Security Frame	
Deployment Mode	

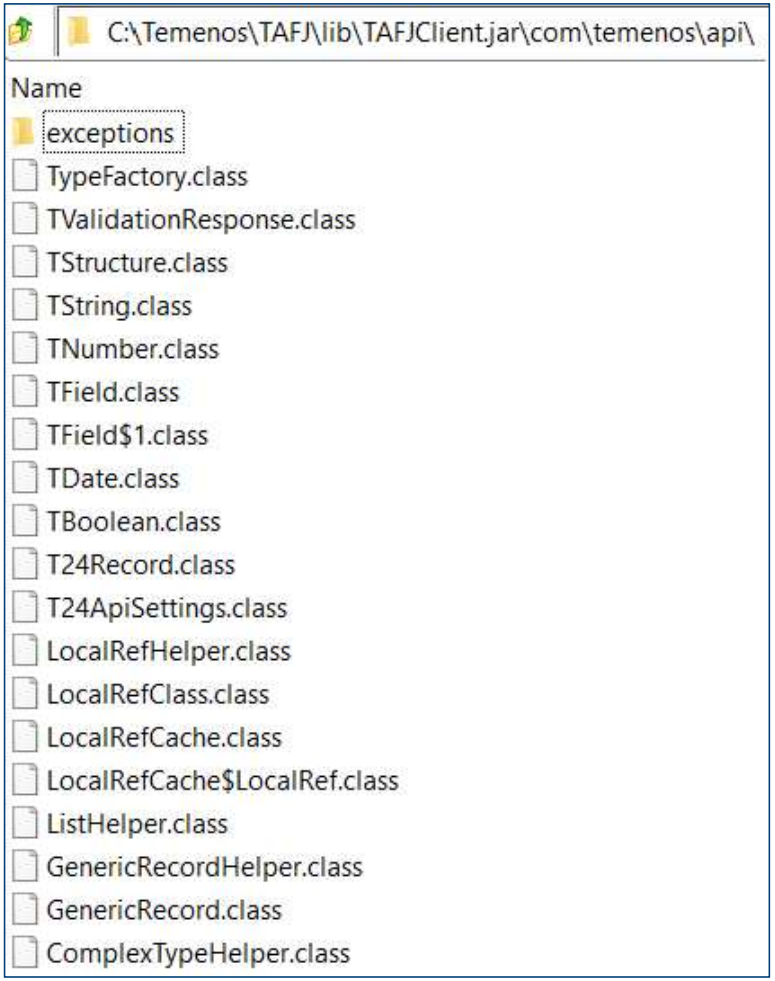
TEMENOS Learning Community

APIs and Hooks

- APIs are code Temenos has written that L3 developers can call
- Hooks are code L3 developers have written that T24 can call
- A developer can attach own logic to T24 to be called from core application exits using HOOKS
- Each hook has been provided with methods to carry out the task
- For e..g the Java developer will use the validateField method defined in com.temenos.t24.api.hook.system.RecordLifeCycle class for field validation exits

‘T’ types in T24

- Classes introduced by Temenos for Java developers in TAFJClient.jar
- TStructure
- TField
- TValidationResponse
- Many more like TString, TNumber, TBoolean, TDate …



TEMENOS
Learning Community

‘T’ types in T24 - TStructure

- The “**TStructure**” is a generic type containing a record object
- TStructure must be ‘cast’ to the correct record type
- Maps a jBC dynamic array to java object
- To access a record, the java developer must construct an instance of a record or complex type from the received parameter

```
public TValidationResponse validateRecord(String application, String recordId, TStructure record,
        TStructure lastLiveRecord) {
    FundsTransferRecord fr = new FundsTransferRecord(record);
    TField f1 = fr.getDebitCurrency();
    TField f2 = fr.getCreditCurrency();

    if (!f1.getValue().equals(f2.getValue()))
    {
        f2.setError("CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY");
    }
     return fr.getValidationResponse();
}
```

‘T’ types in T24 - TField

TField offers getters and setters for Enrichment, Error, field values.

Every field in a record is an internal type **TField** (not String by default)

FTVersionDemo.java

```
package com.newbank;

import java.util.List;

public class FTVersionDemo extends RecordLifecycle {

    @Override
    public TValidationResponse validateRecord(String application, String recordId, TStructure record,
            TStructure lastLiveRecord) {
        FundsTransferRecord fr = new FundsTransferRecord(record);
        TField f1 = fr.getDebitCurrency();
        TField f2 = fr.getCreditCurrency();

        if (!f1.getValue().equals(f2.getValue()))
        {
            f2.setError("CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY");
        }
         return fr.getValidationResponse();
    }

}
```

TEMENOS Learning Community

‘T’ types in T24 - TValidationResponse

Record validation is a common mechanism in T24 to allow custom validation on records.

FTVersionDemo.java

```
package com.newbank;

import java.util.List;

public class FTVersionDemo extends RecordLifecycle {

    @Override
    public TValidationResponse validateRecord(String application, String recordId, TStructure record,
            TStructure lastLiveRecord) {
        FundsTransferRecord fr = new FundsTransferRecord(record);
        TField f1 = fr.getDebitCurrency();
        TField f2 = fr.getCreditCurrency();

        if (!f1.getValue().equals(f2.getValue()))
        {
            f2.setError("CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY");
        }
         return fr.getValidationResponse();
    }

}
```

DataAccess

- The system.DataAccess class is a utility class to read, select and access data in T24.
- The api.records package is used to hold records from T24 applications

```
import com.temenos.t24.api.hook.system.Enquiry;
import com.temenos.t24.api.records.
import com.temenos.t24.api.system.Da
```

```
com.temenos.t24.api.records.aaaccountdetails
com.temenos.t24.api.records.aaaccountdetailshist
com.temenos.t24.api.records.aaaccountmovement
com.temenos.t24.api.records.aaaccrualfrequency
com.temenos.t24.api.records.aaaction
com.temenos.t24.api.records.aaactivity
com.temenos.t24.api.records.aaactivitybalances
com.temenos.t24.api.records.aaactivityclass
com.temenos.t24.api.records.aaactivityhistory
com.temenos.t24.api.records.aaactivityhistoryhist
com.temenos.t24.api.records.aaactivityprocessingclass
com.temenos.t24.api.records.aaadvicetype
Press 'Ctrl+Space' to show Template Proposals
```

```
import com.temenos.t24.api.hook.system.Enquiry;
import com.temenos.t24.api.records.account.
import com.temenos.t24.api.system.DataAccess
```

```
AccountRecord - com.temenos.t24.api.records.account
AccrChgCategClass - com.temenos.t24.api.records.accou
AccrCr2CategClass - com.temenos.t24.api.records.accour
AccrCrCategClass - com.temenos.t24.api.records.account
AccrDr2CategClass - com.temenos.t24.api.records.accour
AccrDrCategClass - com.temenos.t24.api.records.account
AltAcctTypeClass - com.temenos.t24.api.records.account
AvailableDateClass - com.temenos.t24.api.records.accoun
ClosureNotesClass - com.temenos.t24.api.records.accoun
ErValueDateClass - com.temenos.t24.api.records.account
EventClass - com.temenos.t24.api.records.account
Press 'Ctrl+Space' to show Template Proposals
```

```
DataAccess da = new DataAccess(this);
AccountRecord AccRec = new AccountRecord(da.getHistoryRecord("ACCOUNT", currentId));
TField title = AccRec.getAccountTitle1(0);
return title.toString();
```

TEMENOS Learning Community

‘T’ types in T24 – T24Context

- T24Context is the way to establish a connection between Java application and T24
- TAFJClient.jar is required in JAVA_PROJECT to establish the connection.(available in TAFJ_HOME/lib).
- T24Context is used to set the credentials that pass through T24 Validation.
- Hook routines do not need a T24Context to connect to T24.

‘T’ types in T24 – T24Context

```java
package com.newbank;

import com.temenos.t24.api.complex.st.customerapi.PersonalInfo;
import com.temenos.t24.api.party.Customer;
import com.temenos.tafj.api.client.impl.T24Context;

public class GetT24Context {

    public static void main(String[] args) {
        System.setProperty("tafj.home",
                "C:/Temenos/TAFJ");
        T24Context ctx = new T24Context("tafj");
        ctx.setPassword("123456");
        ctx.setUser("INPUTT");

        Customer customer = null;
        customer = new Customer(ctx);
        customer.setCustomerId("100352");

        PersonalInfo personalInfo = customer.getPersonalInfo();
        System.out.println("Nationality " + personalInfo.getNationality());
        System.out.println("Residence " + personalInfo.getResidence());
        System.out.println("Date of Birth " + personalInfo.getDateOfBirth());

    }
}
```

TAFJ properties file

OUTPUT

```
(OFS.INITIALISE.SOURCE) : JAVA.FRAMEWORK
Nationality AU
Residence AU
Date of Birth 19711225
```

TEMENOS Learning Community

Complex class in T24

- No need to set/get related information in separate parameters.
- For e.g. Customer PersonalInfo is made up of DOB, nationality and residence

```
package com.newbank;

import com.temenos.t24.api.complex.st.customerapi.PersonalInfo;
import com.temenos.t24.api.party.Customer;
import com.temenos.tafj.api.client.impl.T24Context;

public class GetT24Context {

    public static void main(String[] args) {
        System.setProperty("tafj.home",
                "C:/Temenos/TAFJ");
        T24Context ctx = new T24Context("tafj");
        ctx.setPassword("123456");
        ctx.setUser("INPUTT");

        Customer customer = null;
        customer = new Customer(ctx);
        customer.setCustomerId("100352");

        PersonalInfo personalInfo = customer.getPersonalInfo();
        System.out.println("Nationality " + personalInfo.getNationality());
        System.out.println("Residence " + personalInfo.getResidence());
        System.out.println("Date of Birth " + personalInfo.getDateOfBirth());

    }
}
```

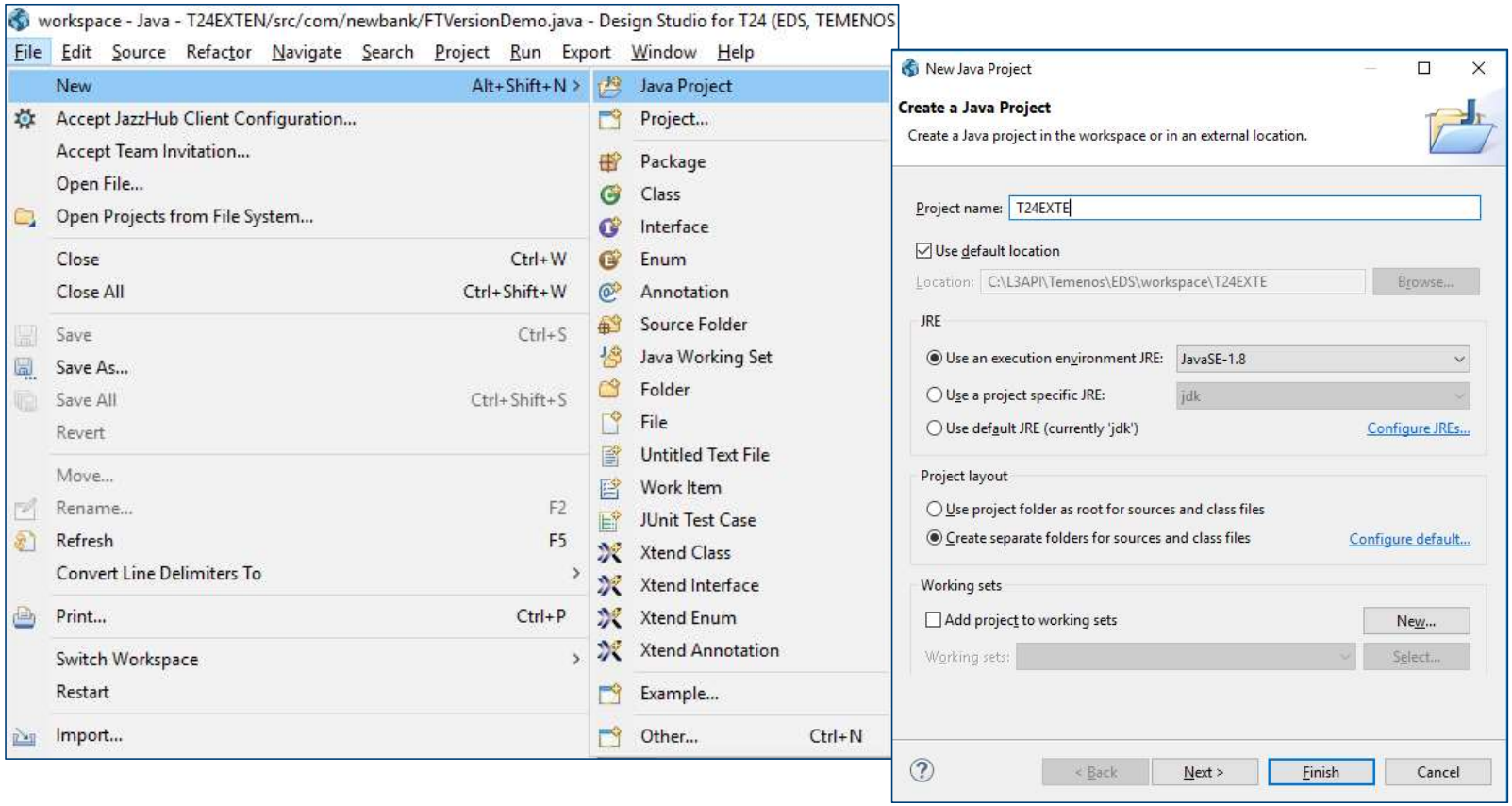
```
PersonalInfo
{
  dateOfBirth : date
  nationality : string
  residence : string
}
```

```
(OFS.INITIALISE.SOURCE) : JAVA.FRAMEWORK
Nationality AU
Residence AU
Date of Birth 19711225
```

TEMENOS Learning Community

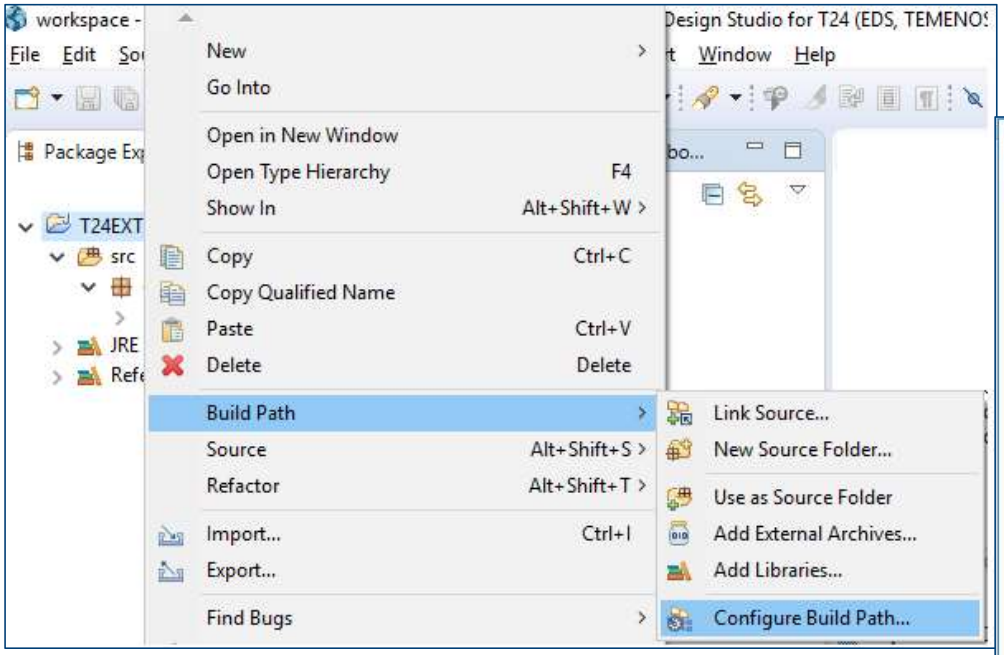
Practice 1.1 - Creating java project in Design Studio

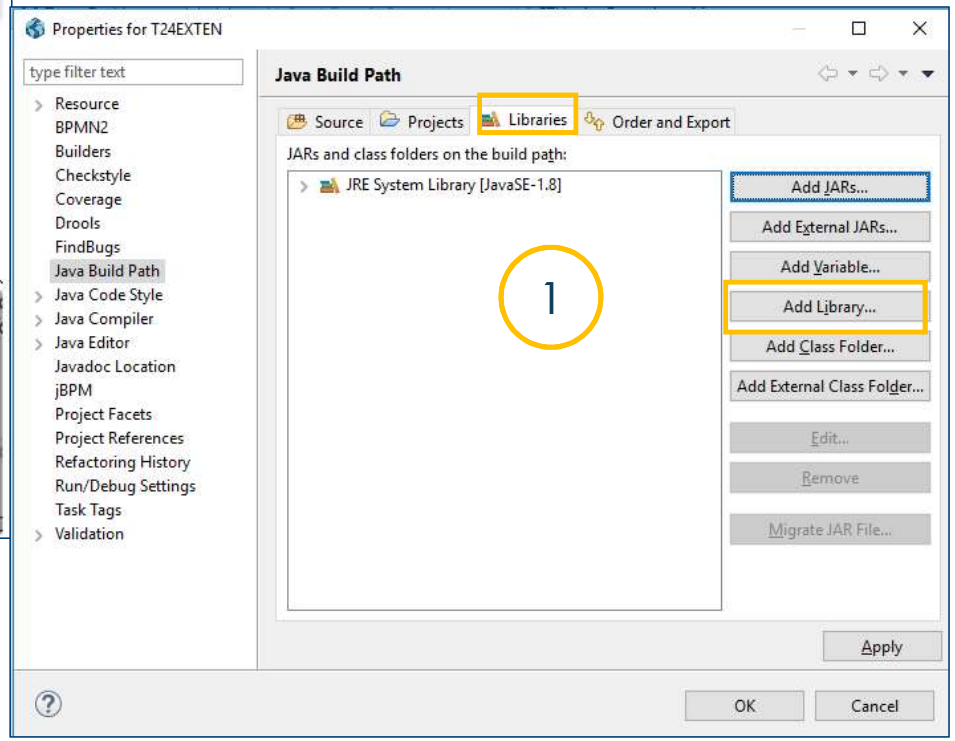
- File →New→Java Project (ProjectName)



TEMENOS Learning Community

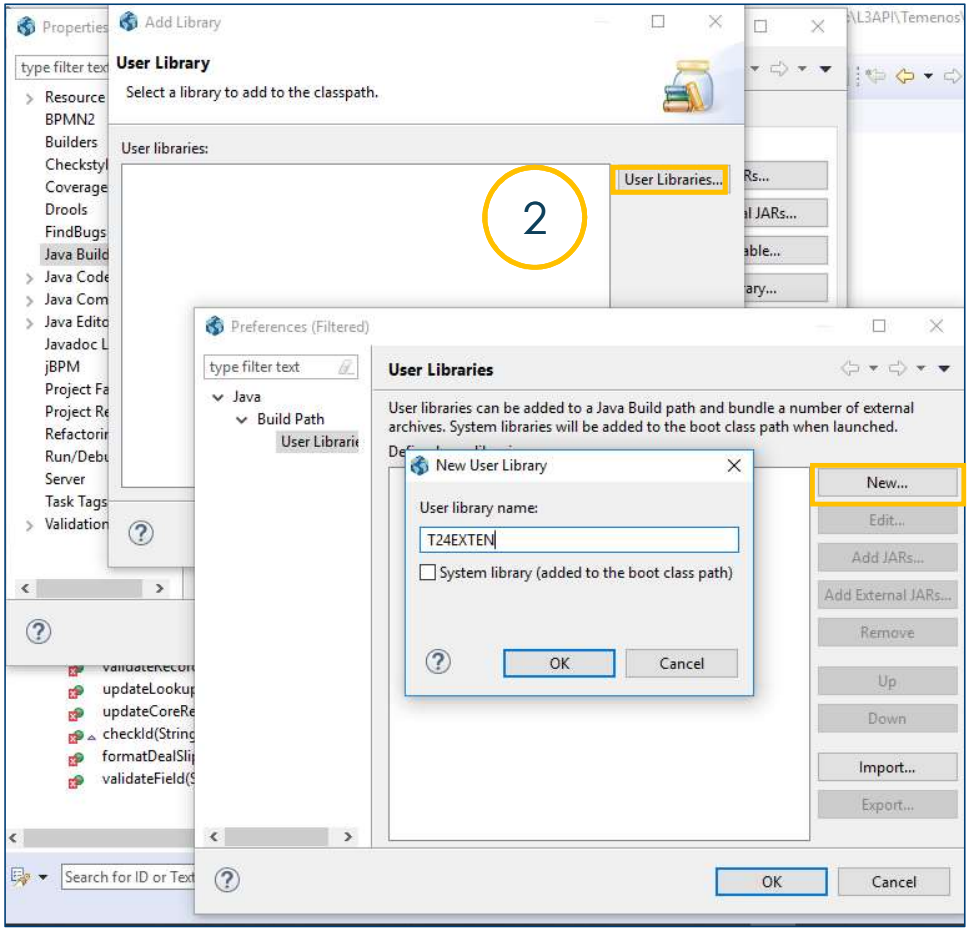
Setup - Configure Build Path – T24 Precompile

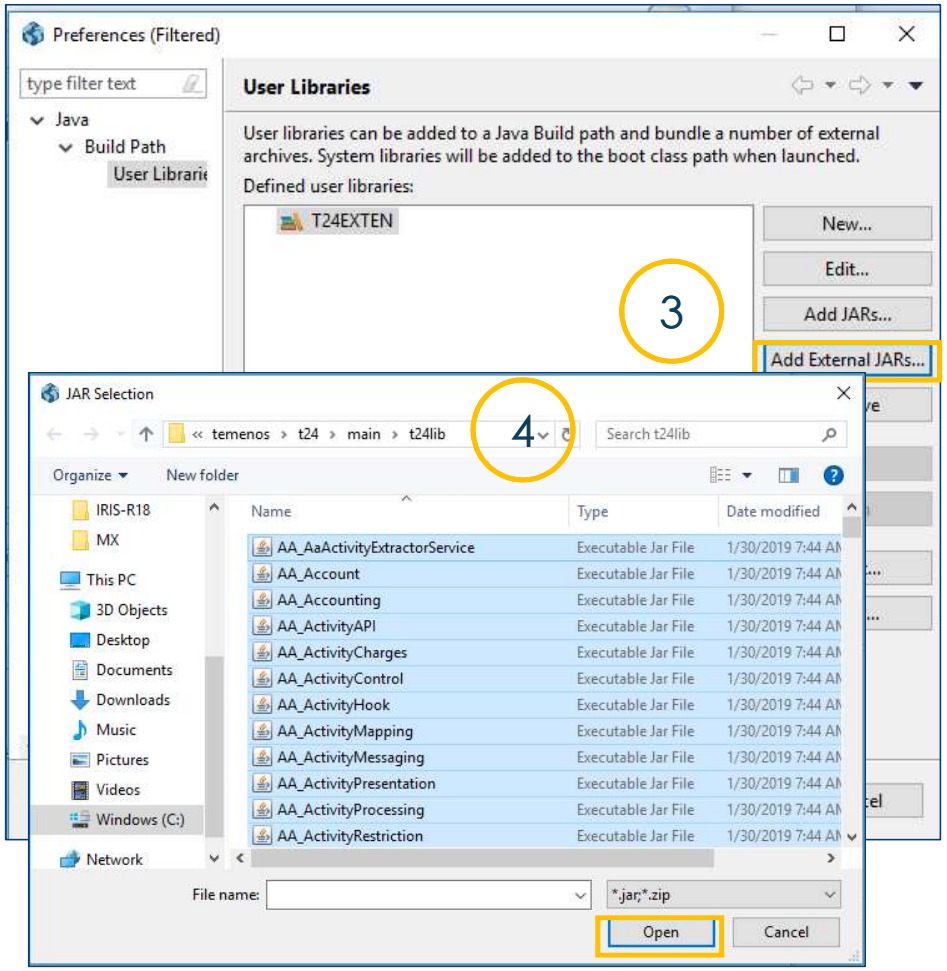




TEMENOS
Learning Community

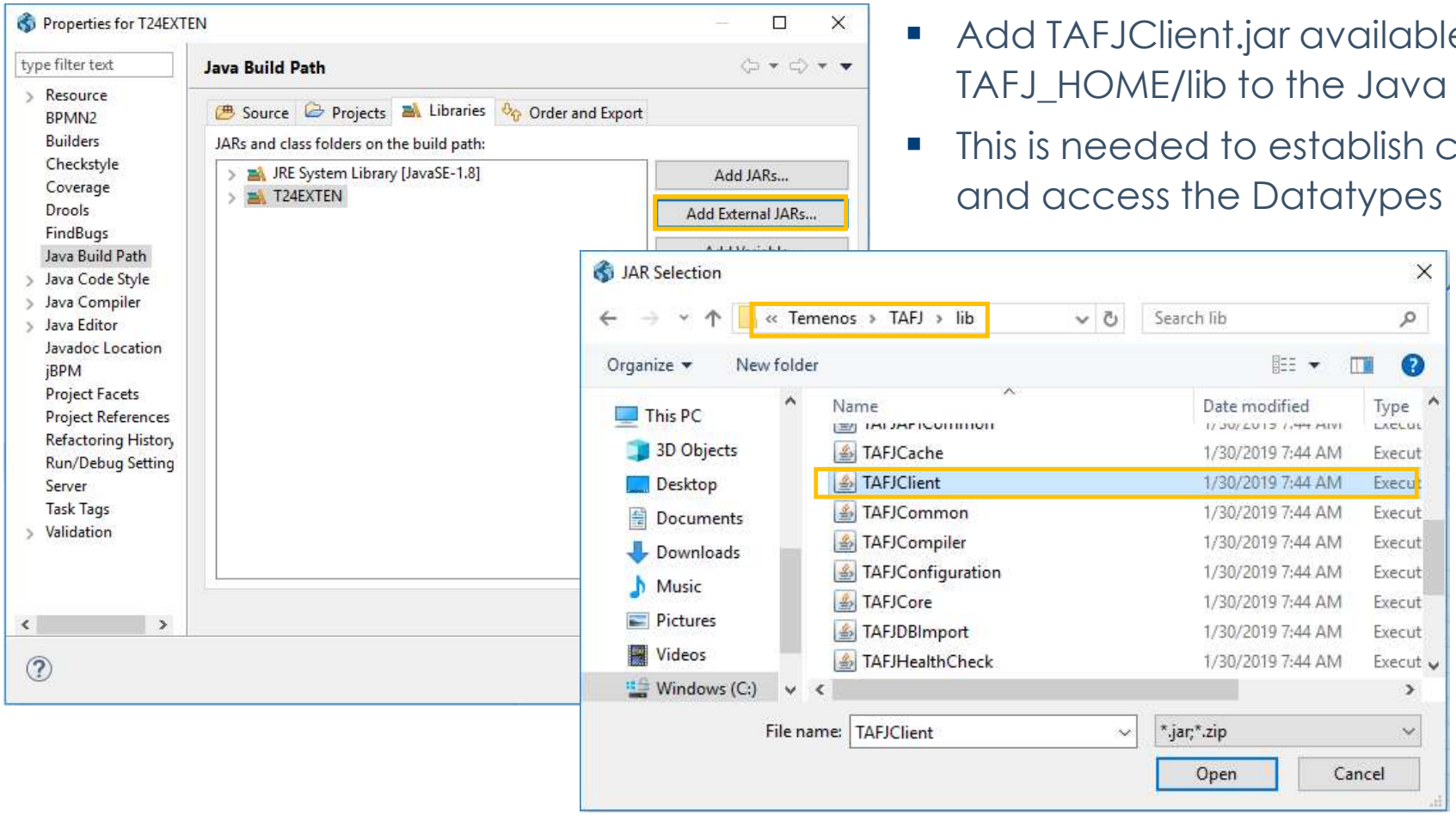
Configure Build Path – T24 Precompile





TEMENOS
Learning Community

Configure Build Path - TAFJClient

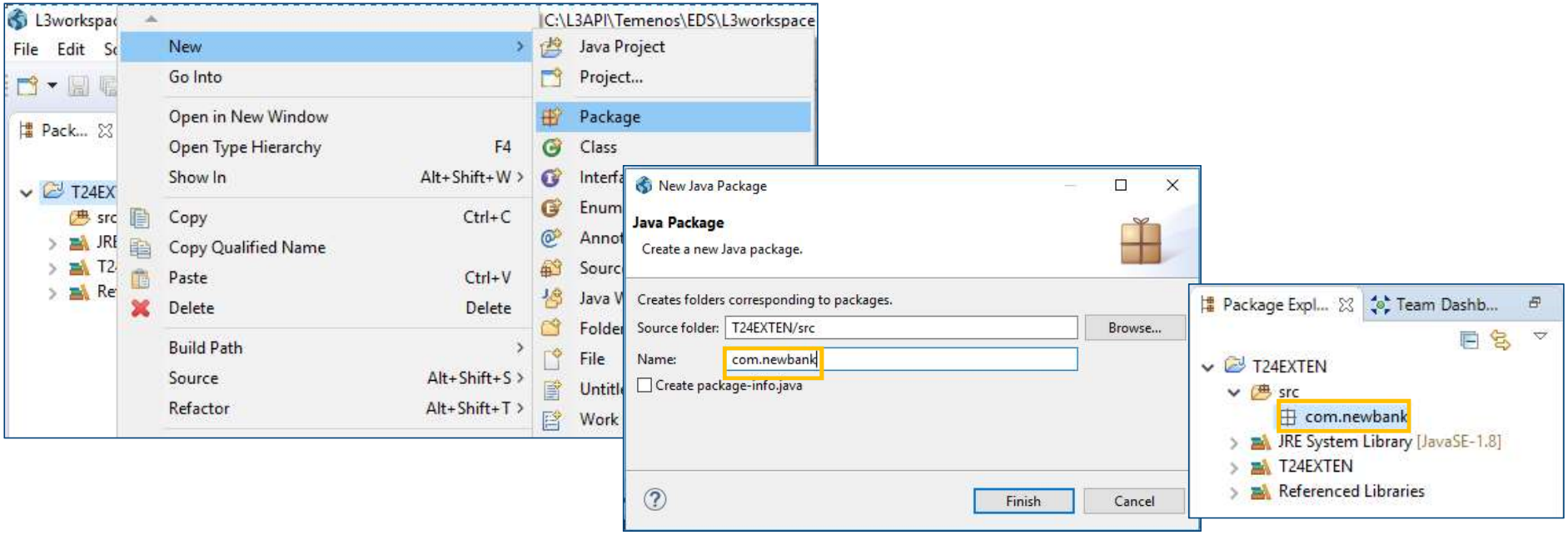


- Add TAFJClient.jar available in TAFJ_HOME/lib to the Java Build Path
- This is needed to establish connection and access the Datatypes

TEMENOS
Learning Community

Create package

- Java package is a mechanism for organizing Java classes into namespaces similar to the T24 modules.
- All classes go into a package
- Right Click on your project →New →Package→PackageName.



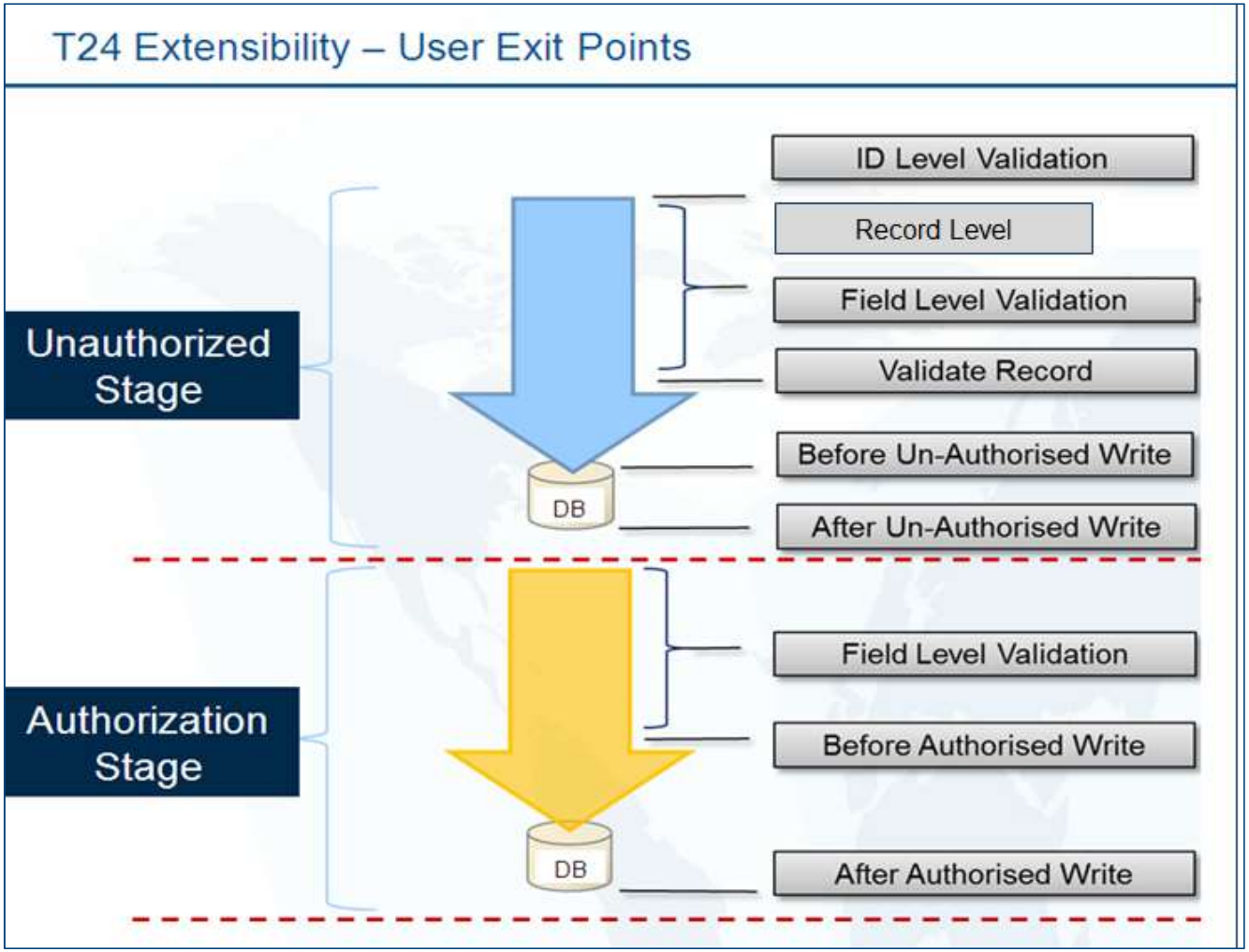
TEMENOS
Learning Community

Lesson 2. VERSION Hooks in Java

T24 Extensibility in Java

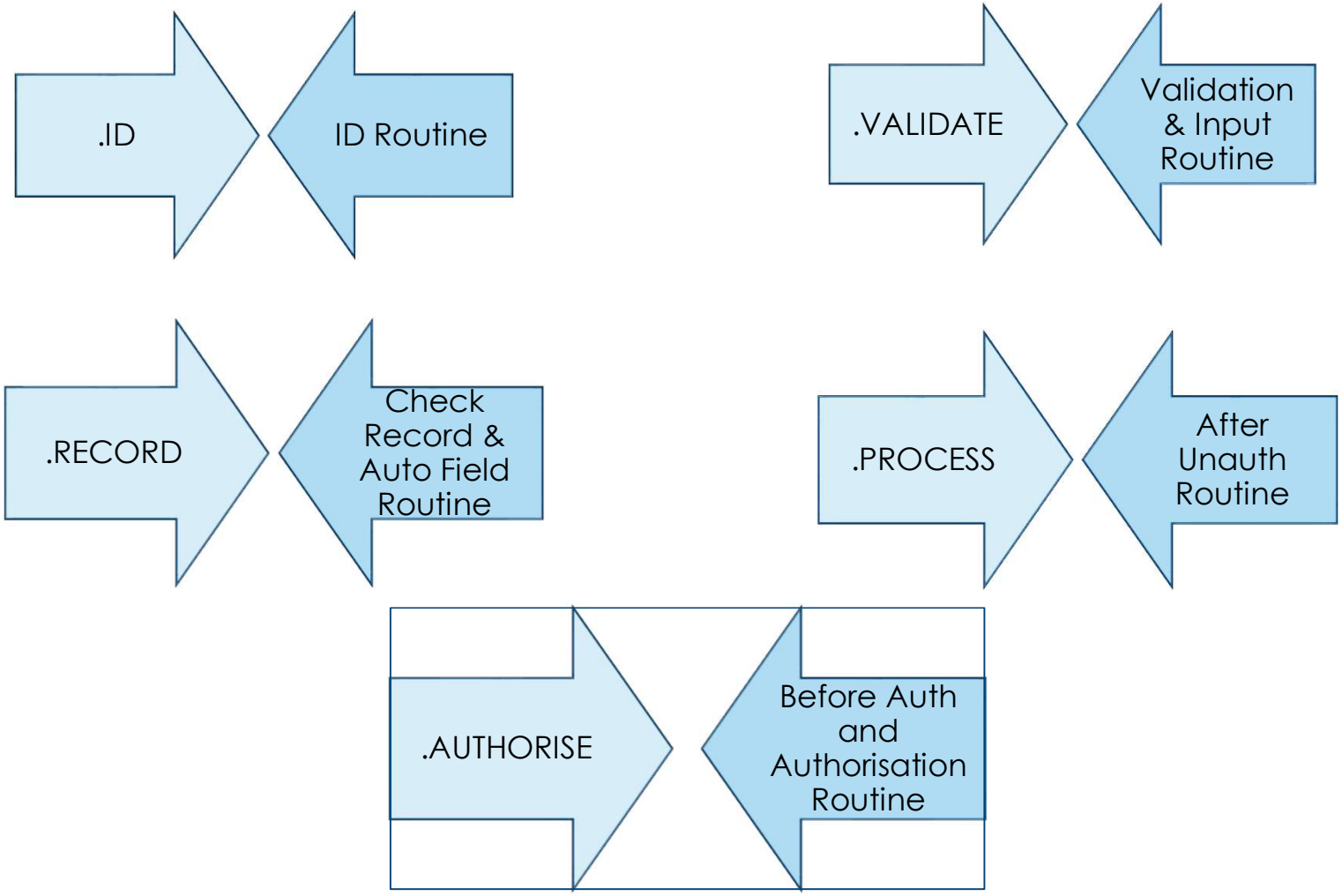
TEMENOS
Learning Community

T24 Extensibility



TEMENOS Learning Community

Routine Invocation Stages in the Template Life Cycle



Fields in VERSION to which hooks can be attached

- ID.RTN – hook to validate ID
- CHECK.REC.RTN – record level hook to check values or default values in the record before the record is displayed to the user
- AUT.NEW.CONTENT – field level hook to default/modify values in field before the record is displayed to the user
- VALIDATION.RTN – field level hook to validate data in field
- INPUT.RTN – record level hook to perform additional validations
- AFTER.UNAU.RTN – user exit that is called after unauthorised-record-write.
- BEFORE.AUTH.RTN - Invoked just prior to the final update of files at the authorised stage of a transaction
- AUTH.ROUTINE – invoked after the final update of files at the authorised stage of a transaction

VERSION HOOKS in T24

- Version hooks are attached to a VERSION
- **com.temenos.t24.api.hook.system.RecordLifecycle** has the following methods
  - checkId
  - defaultFieldValues
  - defaultFieldValuesOnHotField
  - formatDealSlip
  - updateCoreRecord
  - updateLookupTable
  - validateRecord

TEMENOS
Learning Community

Practice 2.1 – VERSION HOOK

Raise and error message when the CREDIT.CURRENCY and DEBIT.CURRENCY are not same in a FUNDS.TRANSFER transaction

Solution

1. Create a class that extends **RecordLifecycle** and override the validateRecord()
2. Create JAR, add in module.xml and restart jBoss
3. Make an entry in EB.API
4. Create a VERSION and attach the EB.API record to the Input routine to validate if DEBIT.CURRENCY and CREDIT.CURRENCY are the same

Workflow

EXIT Point field
- Identify the EXIT point. Read Helptext to get the EB.API record ID
- T24 Browser

EB.API record
- Get the hook method name to implement
- T24 Browser

Javadoc
- Find hook method signature and abstract class name
- Javadoc

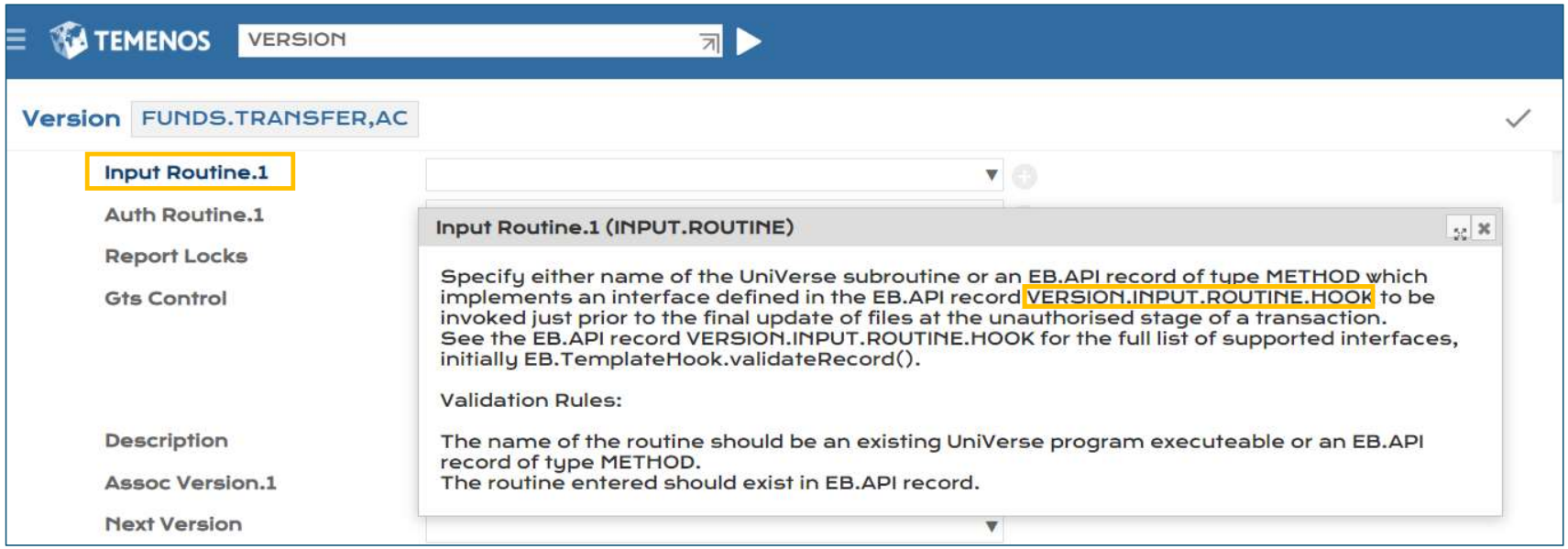
Code in Java
- Implement the hook
- Design Studio

EB.API
- Create EB.API record with SourceType=Method. Give hook method name, class and package
- T24 Browser

Link Method
- Link hook method to EXIT.POINT using EB.API
- T24 Browser

Determining the Superclass and Method – exit point

- Identify the exit points in the version to attach hook routines. In this case, INPUT.RTN
- The helptext for the exit point field tells you whether or not a Java routine can be attached.



Determining the Superclass and Method – Hook method

- View the EB.API record indicated in the helptext. This record gives the name of the hook method to override in the Java implementation of the exit point routine.
- The hook component name indicates the Javadoc jar with documentation on the method

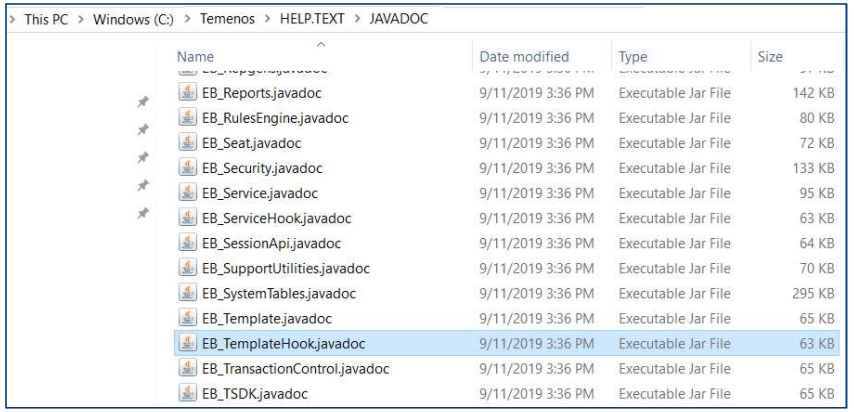
EB.API VERSION.INPUT.ROUTINE.HOOK

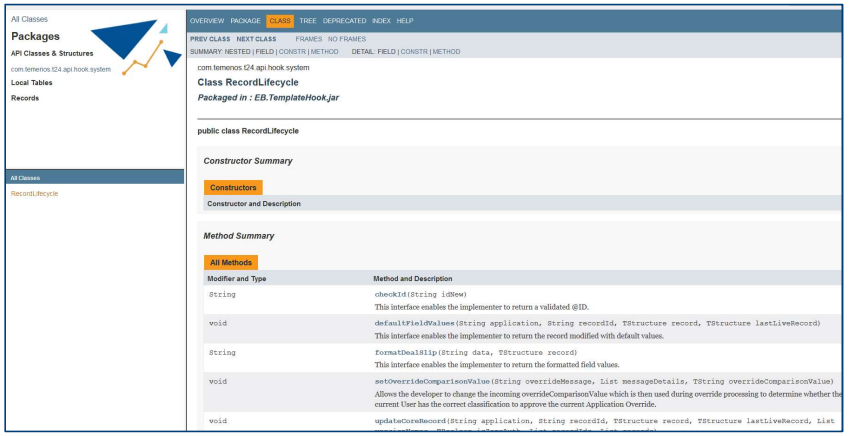
Description	EN Hooks for Version
Protection Level	Full
Source Type	Hook
Hook Description.1	Input Routine Hook
Hook Component.1	EB.TemplateHook
Hook Method.1	validateRecord
Invoke Component.1	EB.TemplateHook
Invoke Method.1	validateRecordInvoker

TEMENOS Learning Community

Determining the Superclass and Method – Javadoc

- Javadocs are packaged as component wise jars in the HELP.TEXT folder.
- View the hook methods' documentation by double clicking the component Javadoc jar EB_TemplateHook.javadoc.jar.
- Select the package com.temenos.t24.api.hook.system and the class RecordLifecycle. The documentation provides a detailed description of all the methods in the class and its parameters.





TEMENOS Learning Community

Writing the Java Implementation

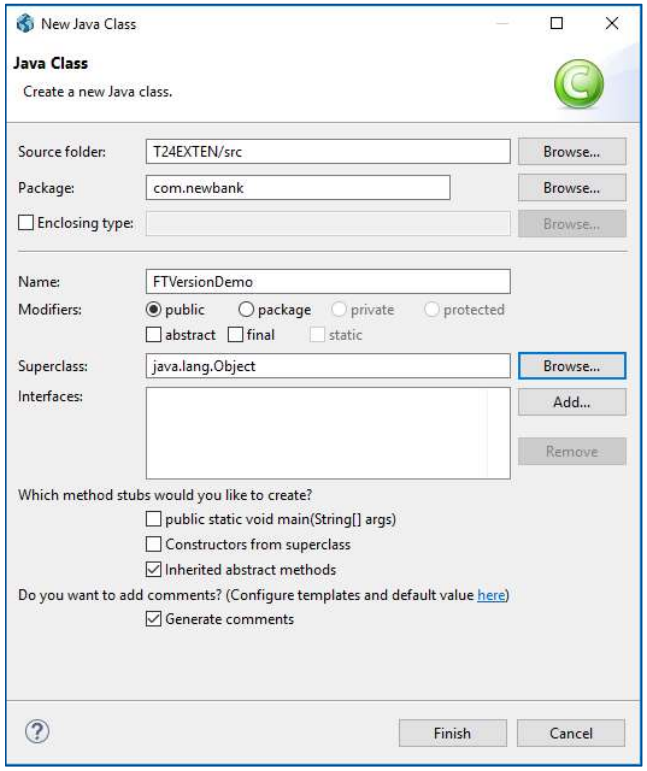
Raise and error message when the CREDIT.CURRENCY and DEBIT.CURRENCY are not same in a FUNDS.TRANSFER transaction

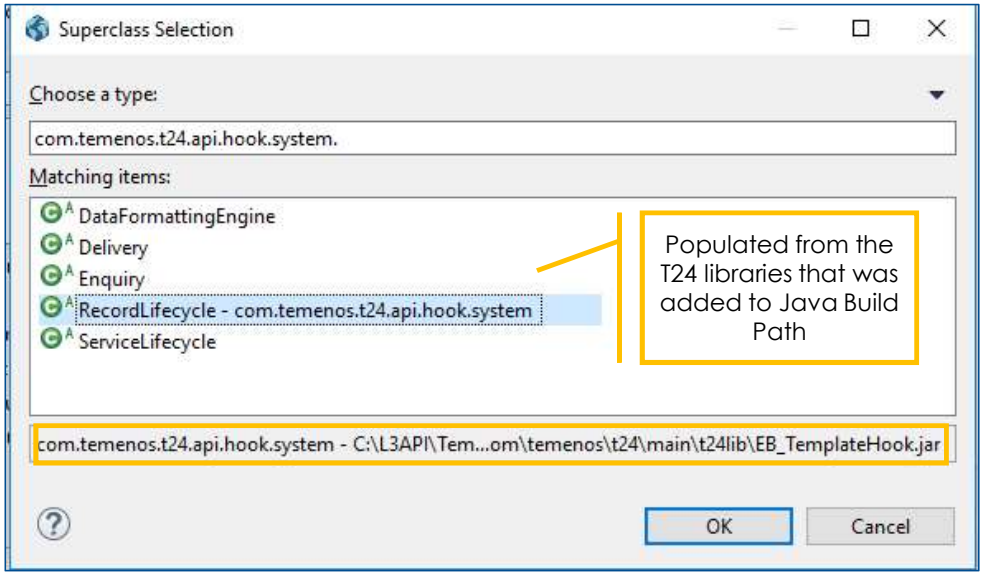
Steps:

1. Create a class that extends **RecordLifecycle** and override the validateRecord()
2. Create JAR, add in module.xml and restart jBoss
3. Make an entry in EB.API
4. Create a VERSION and attach the EB.API record to the Input routine to validate if DEBIT.CURRENCY and CREDIT.CURRENCY are the same

Step 1 – Create class

- Right Click on your Package →New →Class

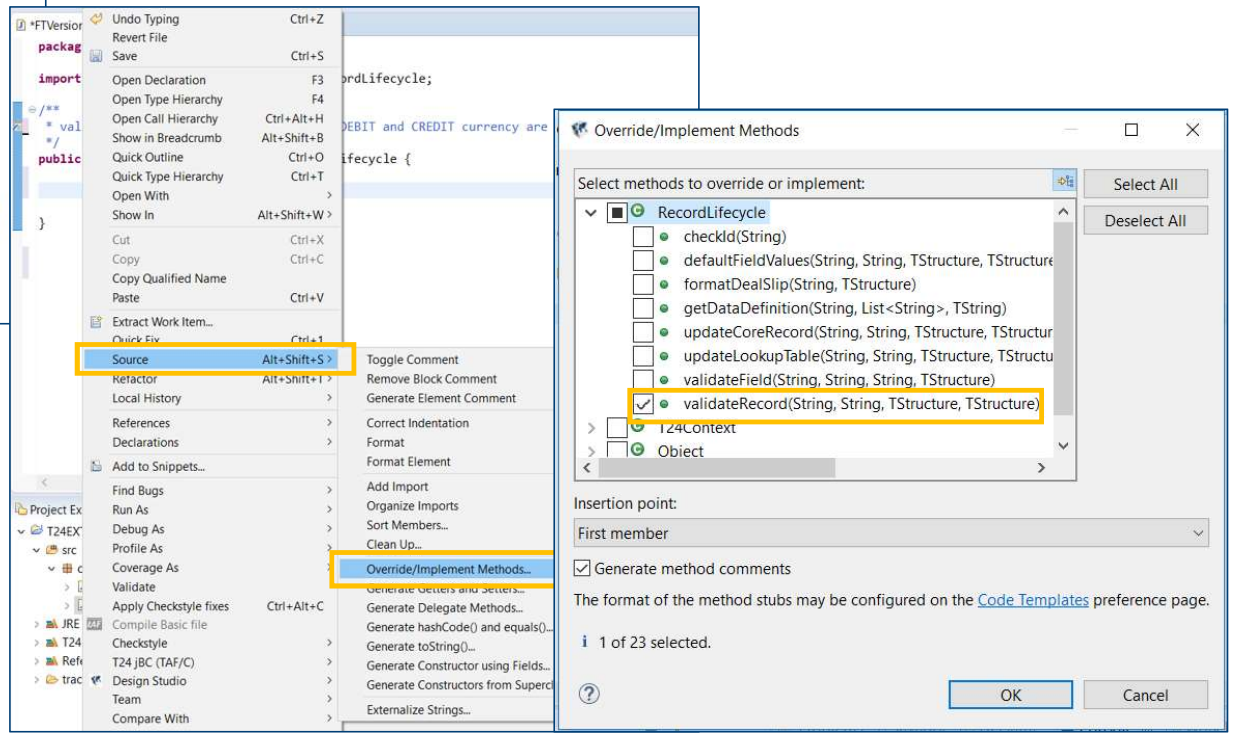




TEMENOS Learning Community

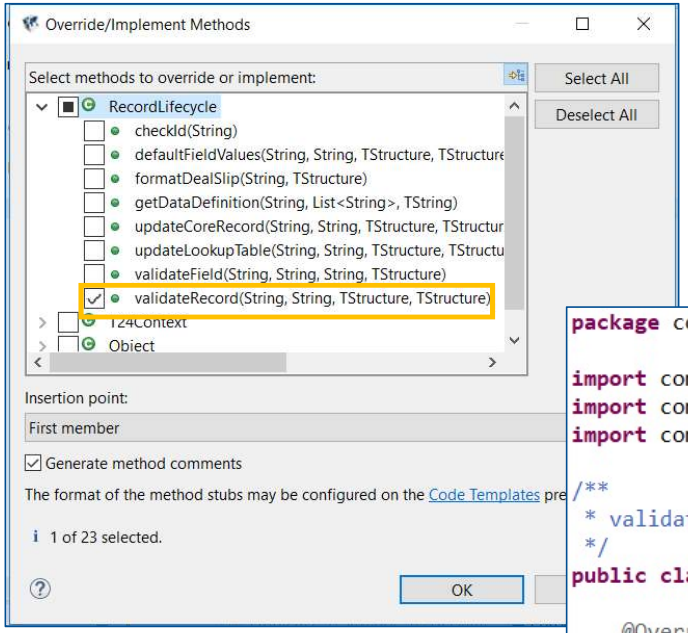
Step 2 – implement validateRecord

```
package com.newbank;

import com.temenos.t24.api.hook.system.RecordLifecycle;

/**
 * TODO: Document me!
 *
 * @author sabithakr
 *
 */
public class FTVersionDemo extends RecordLifecycle {

}
```



TEMENOS Learning Community

Step 3 – implement validateRecord




```java
package com.newbank;

import com.temenos.api.TStructure;
import com.temenos.api.TValidationResponse;
import com.temenos.t24.api.hook.system.RecordLifecycle;

/**
 * validateRecord is used to check if the DEBIT and CREDIT currency are the same
 */
public class FTVersionDemo extends RecordLifecycle {

    @Override
    public TValidationResponse validateRecord(String application, String recordId, TStructure record,
            TStructure lastLiveRecord) {
        // TODO Auto-generated method stub
        return super.validateRecord(application, recordId, record, lastLiveRecord);
    }
}
```

TEMENOS
Learning Community

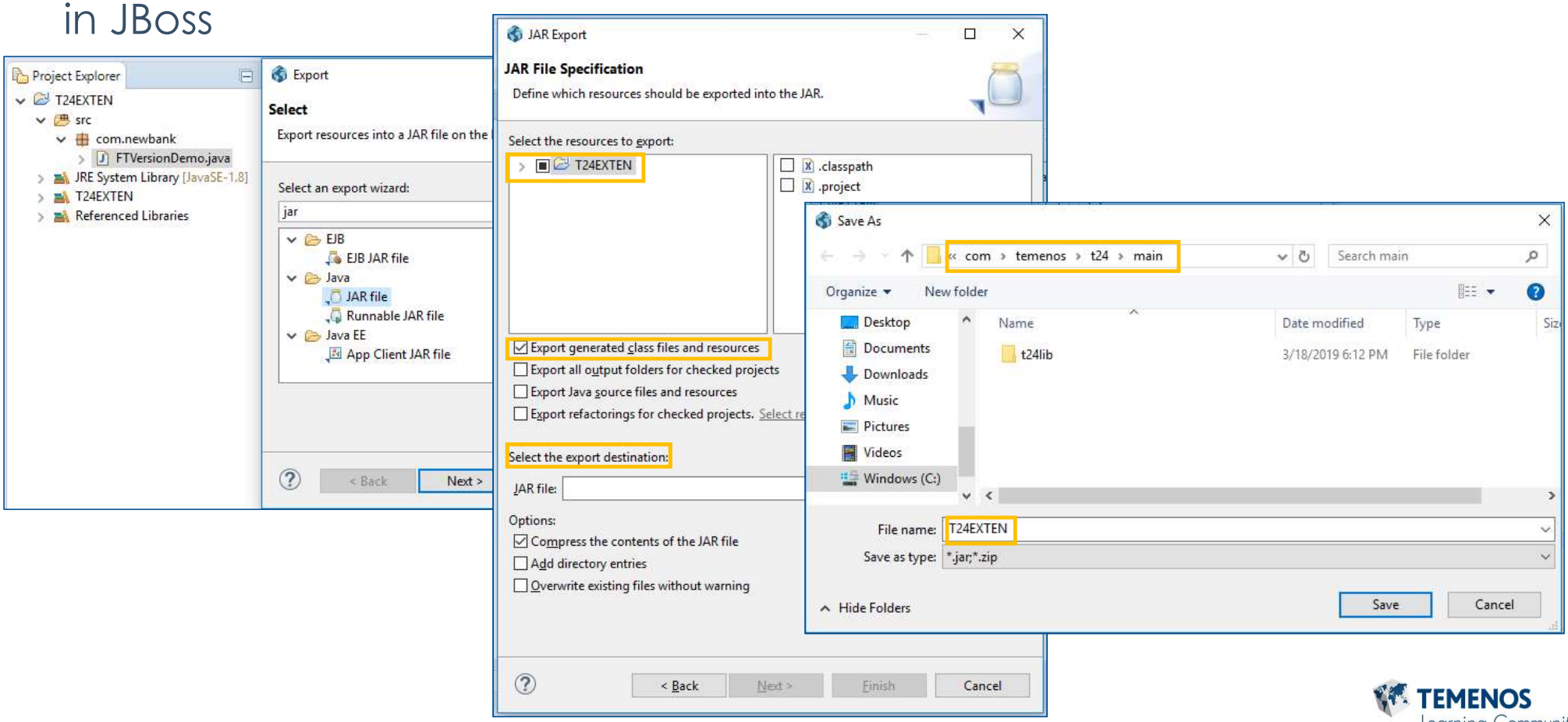
Step 3 – implement validateRecord

```java
package com.newbank;

import com.temenos.api.TField;
import com.temenos.api.TStructure;
import com.temenos.api.TValidationResponse;
import com.temenos.t24.api.hook.system.RecordLifecycle;
import com.temenos.t24.api.records.fundstransfer.FundsTransferRecord;

/**
 * validateRecord is used to check if the DEBIT and CREDIT currency are the same
 */
public class FTVersionDemo extends RecordLifecycle {

    @Override
    public TValidationResponse validateRecord(String application, String recordId, TStructure record,
            TStructure lastLiveRecord) {
        FundsTransferRecord fr = new FundsTransferRecord(record);
        TField f1 = fr.getDebitCurrency();
        TField f2 = fr.getCreditCurrency();


        if (!f1.getValue().equals(f2.getValue()))
        {
            f2.setError("CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY");
        }
        return fr.getValidationResponse();
    }
}
```

TEMENOS
Learning Community

Step 4 – Create the JAR and add to module.xml

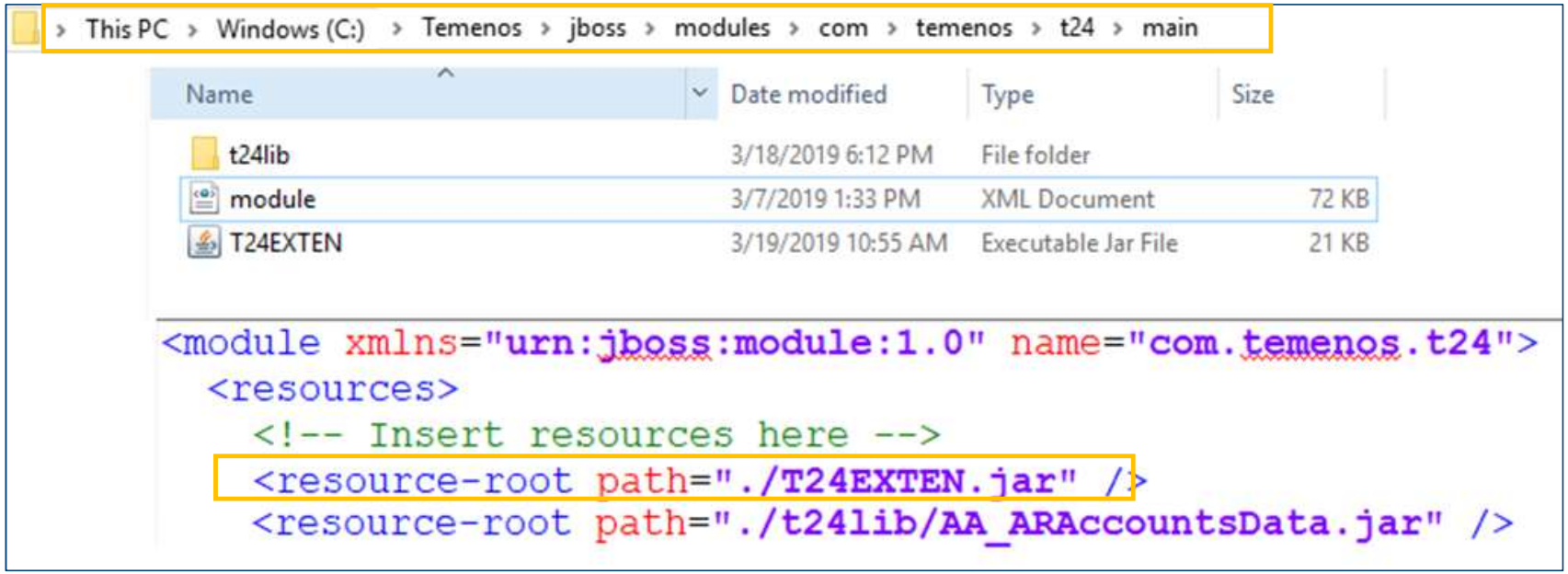
- The .class file must be exported as a JAR and added to the module.xml in JBoss



TEMENOS
Learning Community

Step 5 – Create the JAR and add to module.xml

- Add the JAR to the jBoss module.xml and restart jBoss



TEMENOS
Learning Community

Step 6 – EB.API

- Hooks are invoked when a transaction is validated/committed
- Input Hooks are attached to the field INPUT.ROUTINE in the VERSION application
- All hooks must have an entry in EB.API

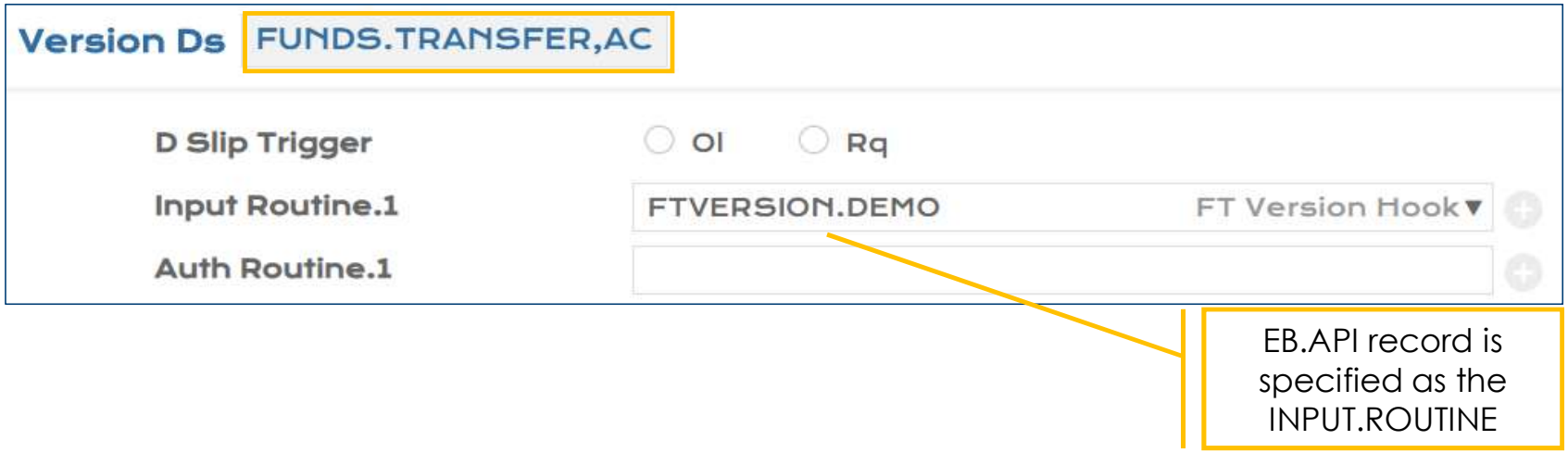
EB.API FTVERSION.DEMO

Description	EN FT Version Hook
Protection Level *	● Full ○ Partial ○ None
Source Type *	○ Basic ○ Java ○ Hook ● Method
Java Method	validateRecord
Java Class	FTVersionDemo
Java Package	com.newbank

TEMENOS Learning Community

Step 7 – Create/Modify version

- Modify the version FUNDS.TRANSFER,AC



TEMENOS
Learning Community

Launching the VERSION

- Error is raised when DEBIT.CURRENCY is not equal to CREDIT.CURRENCY

Transfer Between Accounts FT18107K2MR8

- Please Select GO

Transfer Between Accounts | Audit

Debit Account * 19461
WILLIAM HEWLETT

Debit Currency USD US Dollar

Debit Amount 123.00

Debit Value Date 17 APR 2018
17 APR 2018

Debit Narrative

Cheque Type

Cheque Number

Ordered By.1

Credit Account * 11223 NIKE

Credit Currency GBP Pound Ster
CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY

Credit Amount

=
Please resolve the errors below to proceed:

Credit Currency: CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY

Launching the VERSION

- Error is not raised when DEBIT.CURRENCY equal to CREDIT.CURRENCY

Transfer Between Accounts FT18107K2MR8

- Please Select

Information:
Field has been validated

Transfer Between Accounts | Audit

Debit Account *	19461 WILLIAM HEWLETT		
Debit Currency	USD US Dollar	Debit Amount	123.00
Debit Value Date	17 APR 2018 17 APR 2018	Debit Narrative	
Cheque Type		Cheque Number	
Ordered By.1			
Credit Account *	21261 ZURICH EQUITY F		
Credit Currency	USD US Dollar	Credit Amount	
Credit Value Date	17 APR 2018 17 APR 2018	Credit Narrative	

TEMENOS Learning Community

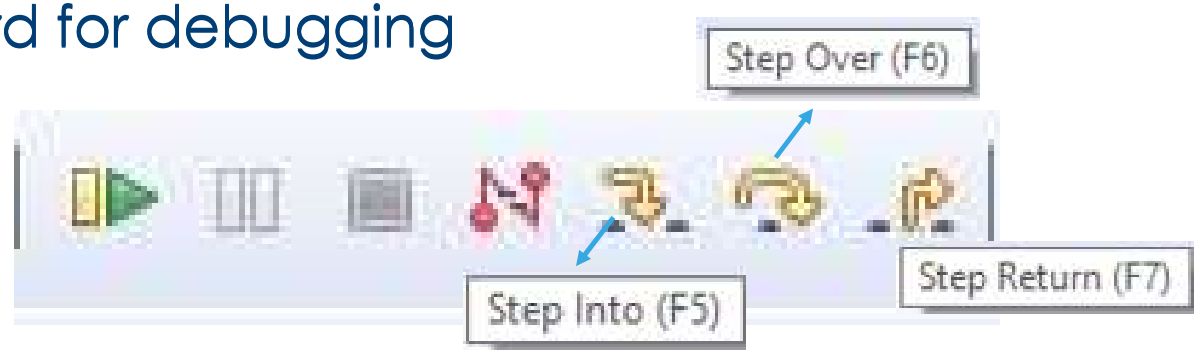
Lesson 3. DEBUG Java code in Design Studio

T24 Extensibility in Java

TEMENOS
Learning Community

DEBUG the Java code

- The Design Studio Java IDE provides many debugging tools and views grouped in the Debug Perspective
- To debug the program, define breakpoints. By adding breakpoints in the source code we can specify where the execution of the program should pause
- To set breakpoints in the source code double click on the small left margin in the source code editor.
- DS uses eclipse standard for debugging
  - Step In : F5
  - Step Over : F6
  - Step Out : F7
  - Continue : F8



TEMENOS
Learning Community

DEBUG the Java code

- Remote Debug jBoss from Design Studio
- Set JAVA_OPTS

For Windows

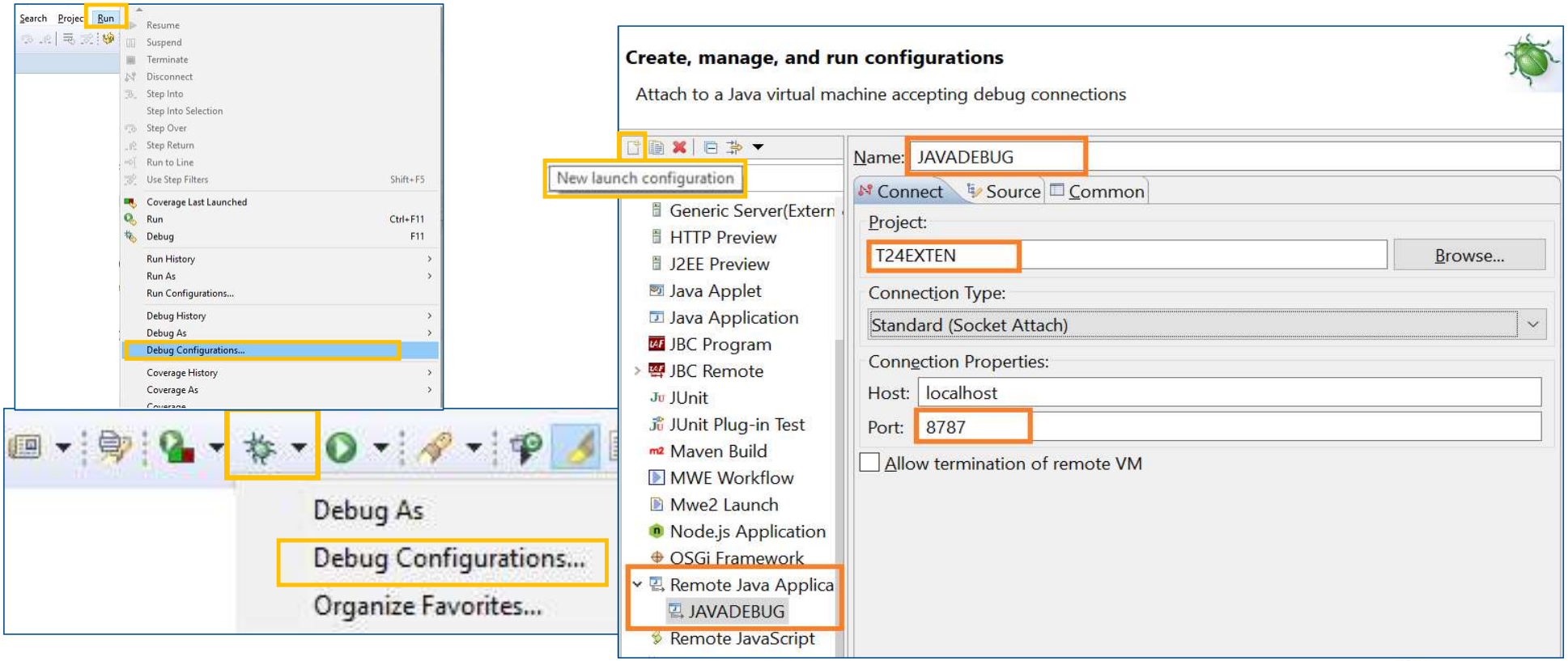
- SET JAVA_OPTS=-Xdebug -Xnoagent -Xrunjdwp:transport=dt_socket, address=8787,server=y,suspend=n%JAVA_OPTS%

For Linux

- JAVA_OPTS="-Xdebug -Xnoagent -Xrunjdwp:transport=dt_socket, address=8787,server=y,suspend=n$JAVA_OPTS"

DEBUG the Java code

- Create the DEBUG configuration



TEMENOS
Learning Community

DEBUG the java code

Transfer Between Accounts FT18107370J1

Transfer Between Accounts | Audit

Debit Account * 19461
WILLIAM HEWLETT
Debit Currency USD US Dollar
Debit Value Date DD MMM YYYY
Cheque Type
Ordered By.1
Credit Account * 11223

Debug | Servers

Thread [default task-15] (Suspended (breakpoint at line 20 in FTVersionDemo))
- FTVersionDemo.validateRecord(String, String, TStructure, TStructure) line: 20
- NativeMethodAccessorImpl.invoke0(Method, Object, Object[]) line: not available [native method]
- NativeMethodAccessorImpl.invoke(Object, Object[]) line: 62
- DelegatingMethodAccessorImpl.invoke(Object, Object[]) line: 43
- Method.invoke(Object, Object...) line: 498
- component_EB_TemplateHook_19_cl.validateRecord(Object, Object, Object, Object, Object) line: not available
- EB_TEMPLATE_HOOK_VALIDATE_RECORD_INVOKER_cl.main() line: not available
- EB_TEMPLATE_HOOK_VALIDATE_RECORD_INVOKER_cl.invoke(Object, Object, Object) line: not available
- EB_TEMPLATE_HOOK_VALIDATE_RECORD_INVOKER_cl.invoke(Object...) line: not available
- component_EB_TemplateHook_19_cl.validateRecordInvoker(Object, Object, Object) line: not available

Debugger view

Variables | Breakpoints

Name	Value
recordId	"FT18107370J1" (id=514)
record	TStructure (id=515)
lastLiveRecord	TStructure (id=520)
fr	FundsTransferRecord (id=528)
f1	TField (id=531)

USD

Variables and expressions view

FTVersionDemo.java | GetT24Context.java | FUNDS.TRANSFER,AC.version | NativeMethodAccessorImpl.class

```
@Override
public TValidationResponse validateRecord(String application, String recordId, TStructure record,
        TStructure lastLiveRecord) {
    FundsTransferRecord fr = new FundsTransferRecord(record);
    TField f1 = fr.getDebitCurrency();
    TField f2 = fr.getCreditCurrency();


    if (!f1.getValue().equals(f2.getValue()))
    {
        f2.setError("CREDIT CURRENCY IS NOT EQUAL TO DEBIT CURRENCY");
    }
     return fr.getValidationResponse();
}
```

The current instruction pointer

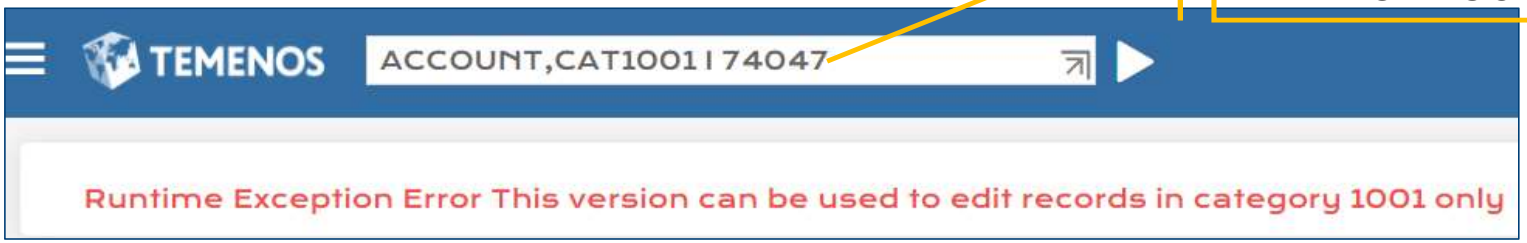
Outline
- com.newbank
- FTVersionDemo
  - validateRecord(String, String, TStructure, TStructure) : TValic

TEMENOS Learning Community

Practice 3.1 – Version Hook CheckRec

- Create a VERSION for ACCOUNT application that allows user to edit records belonging to CATEGORY 1001 only
- Expected Output

Error must be raised for ACCOUNT 74047 as it belongs to CATEGORY 6001



ACCOUNT in CATEGORY 1001 is editable

TEMENOS Learning Community

Practice 3.2 – Version Hook for ID

- Create a VERSION for the CUSTOMER that prefixes "99" to the ID given by the user
- Expected Output

TEMENOS
Learning Community