T24 Extensibility in Java

TEMENOS
Learning Community

Lesson Overview

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

Day 2

- DEBUG the Java code in Design Studio
- ENQUIRY hooks
- SERVICE hooks
- Local application using EB.TABLE.DEFINITION

TEMENOS Learning Community

Lesson 1. Introduction

T24 Extensibility in Java

TEMENOS
Learning Community

Agenda - T24 Extensibility in Java

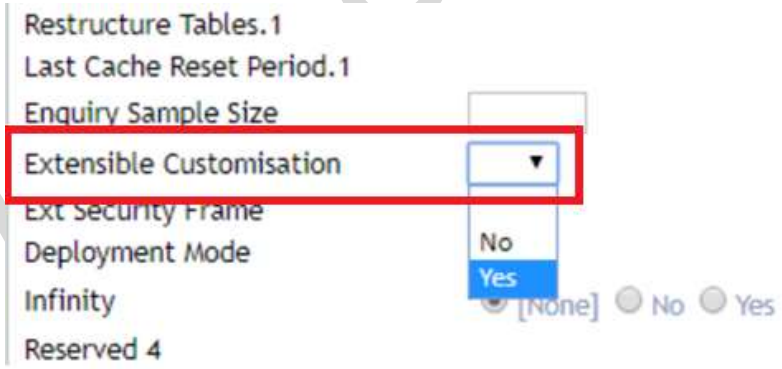
- Why do we need T24 Extensibility in Java ?
- APIs and Hooks
- T-Types
- DataAccess
- Complex classes
- Setup Design Studio

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

Prerequisites

- JD product must be installed



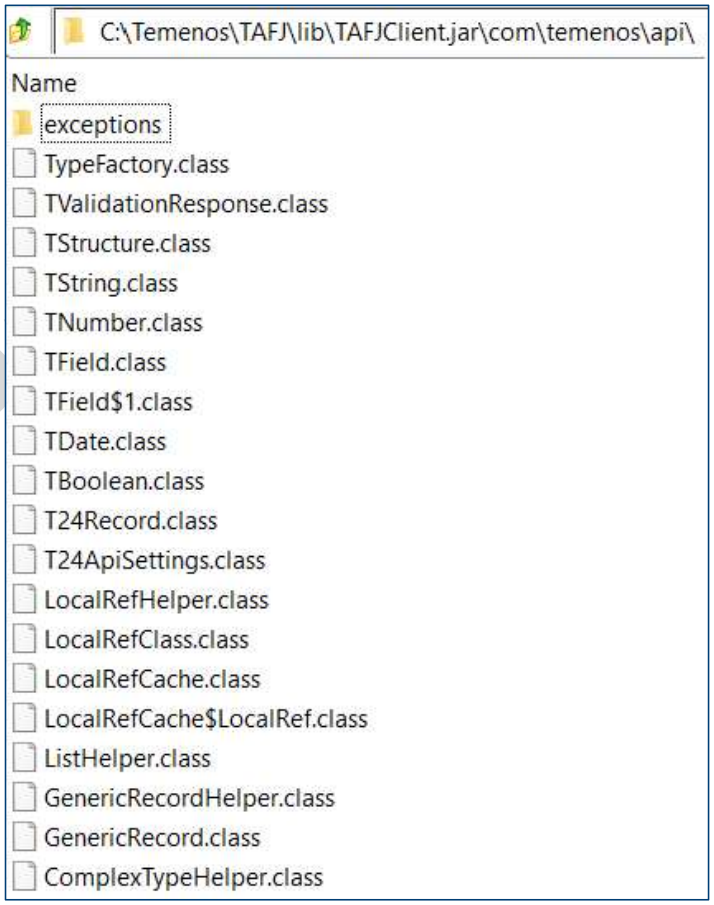
TEMENOS Learning Community

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 …



‘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();
    }

}
```

‘T’ types in T24 - TValidationResponse

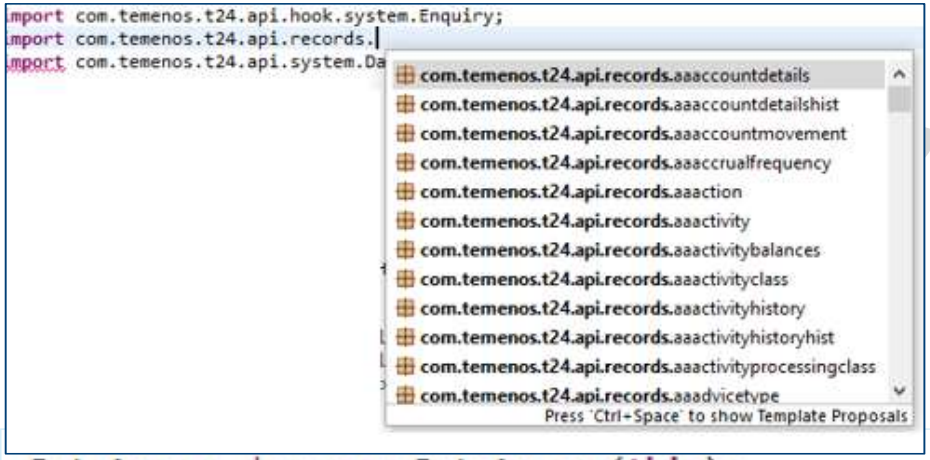
Record validation is a common mechanism in T24 to allow custom validation on records.

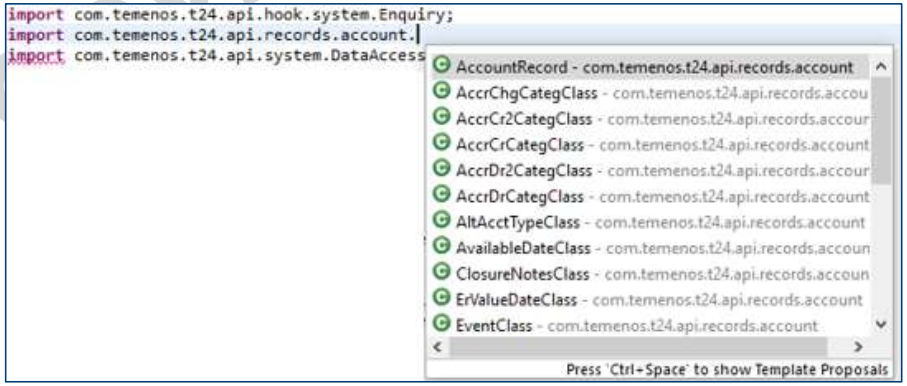
FTVersionDemo.java

```
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






```
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
```

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

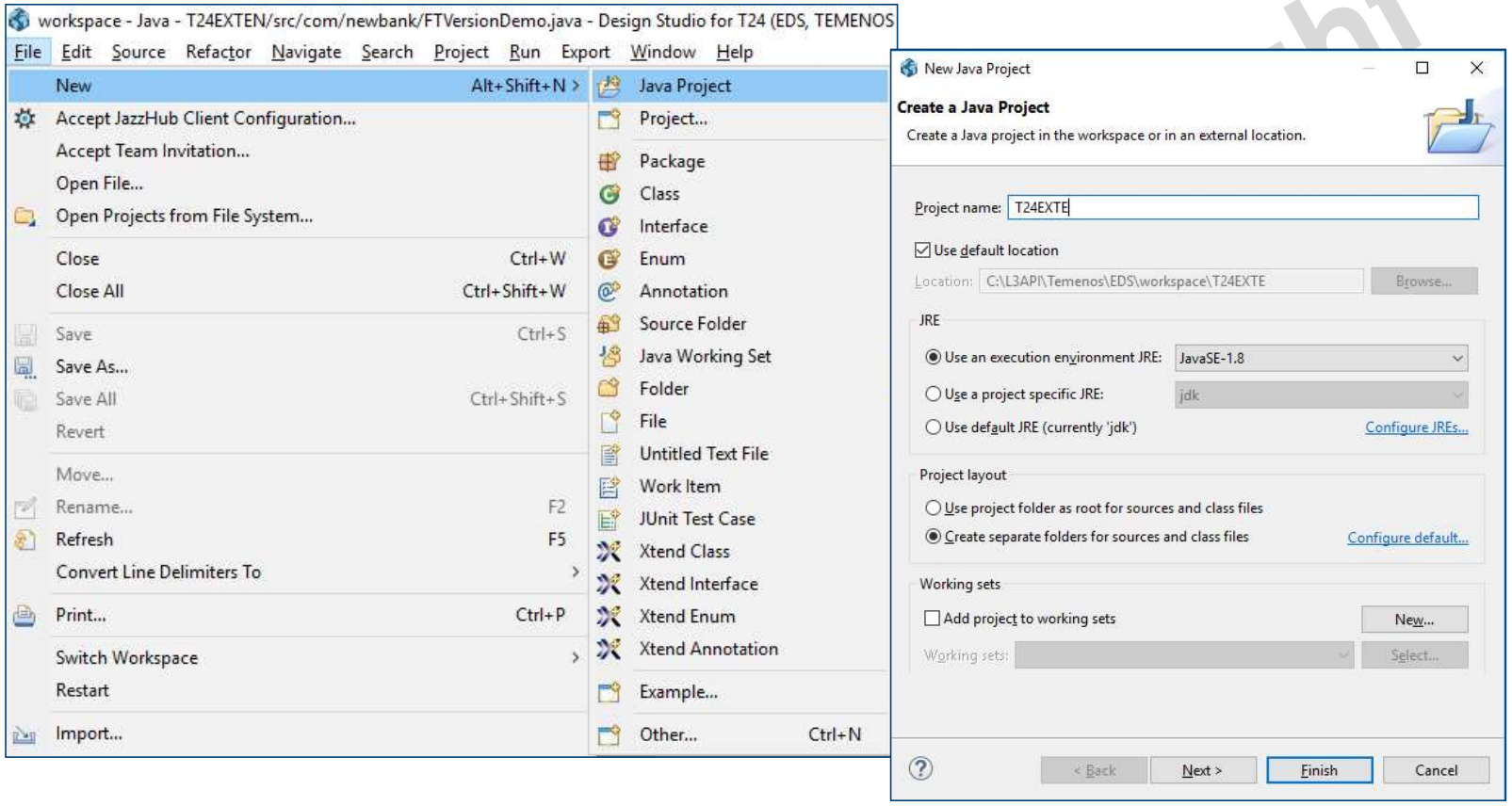
```
package com.newbank;

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
```

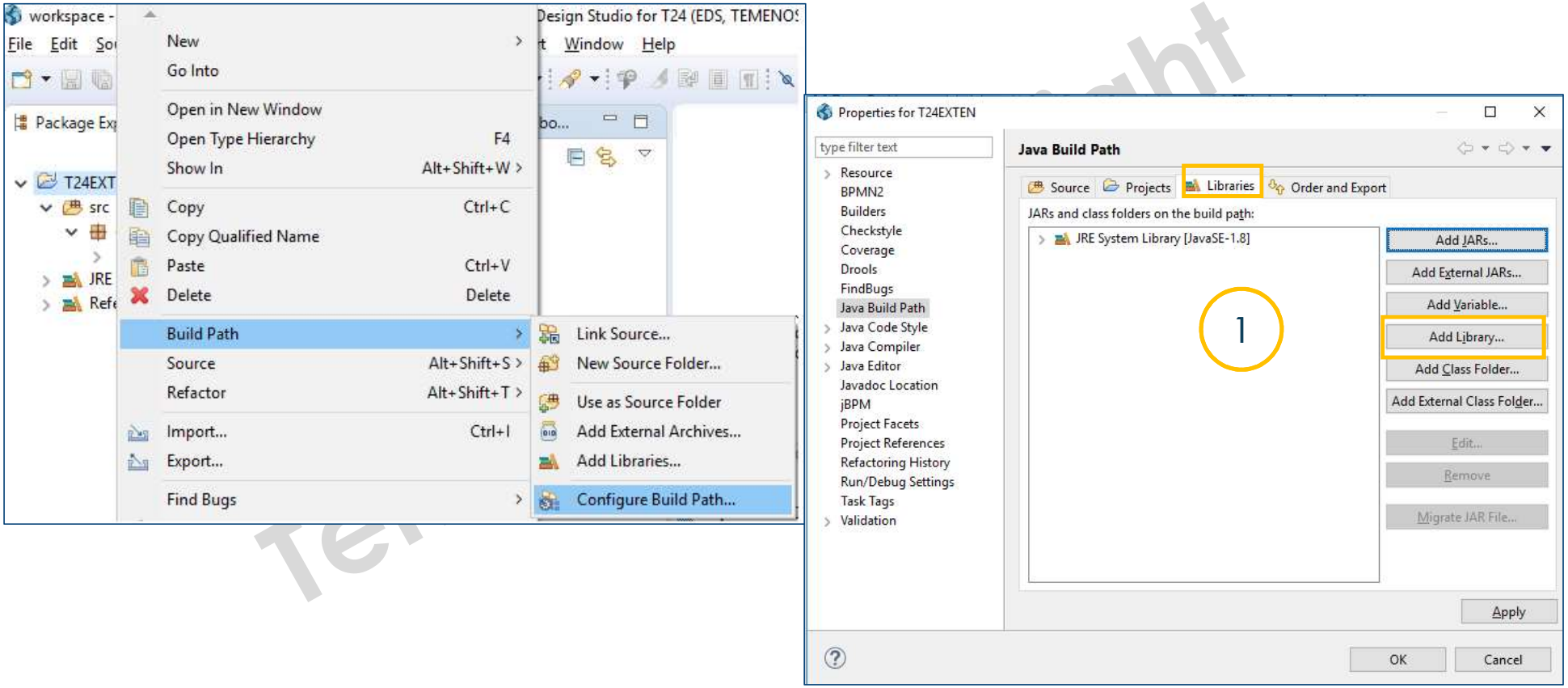
Setup - Creating java project in Design Studio

- File →New→Java Project (ProjectName)



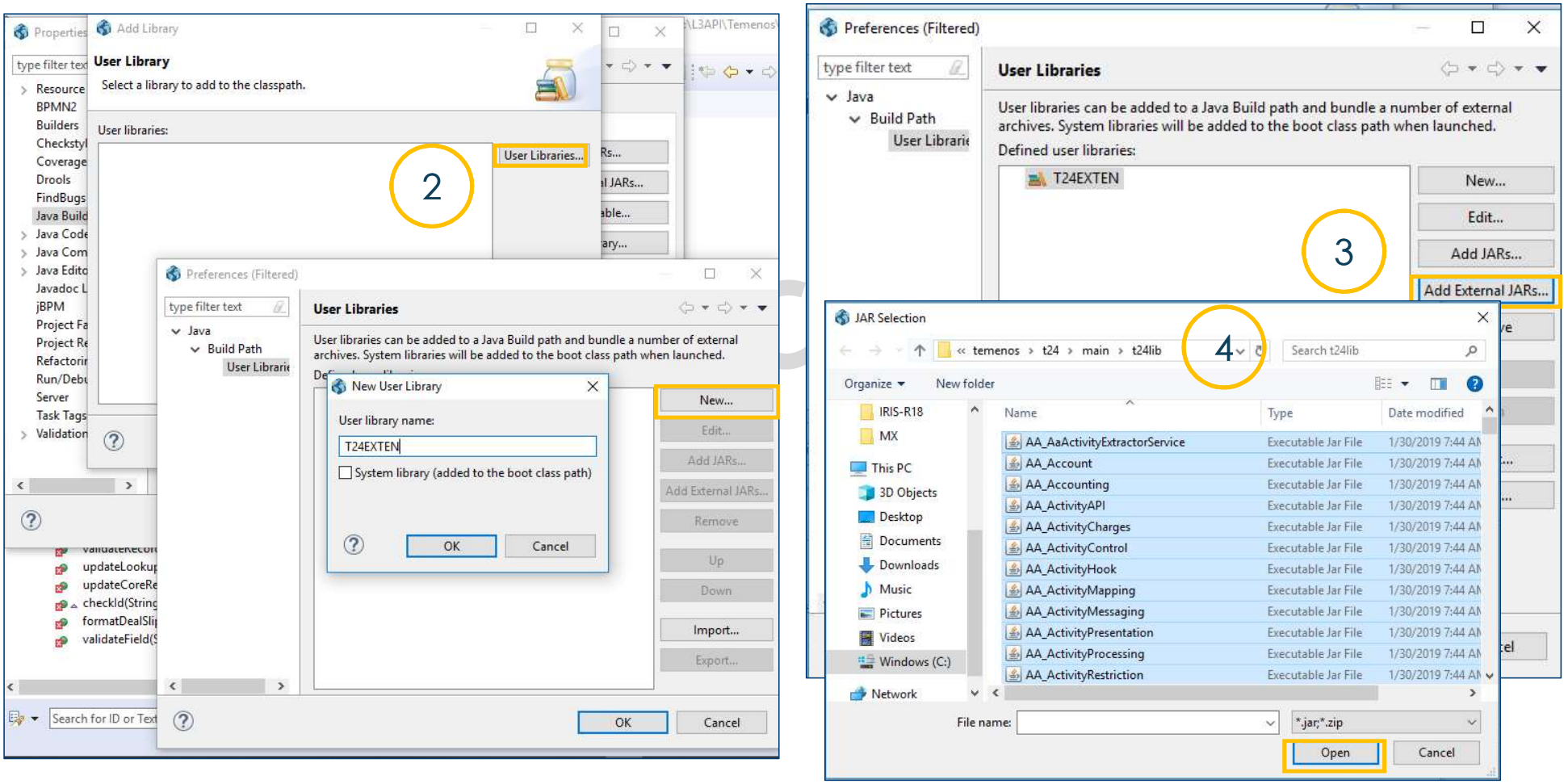
TEMENOS Learning Community

Setup - Configure Build Path – T24 Precompile



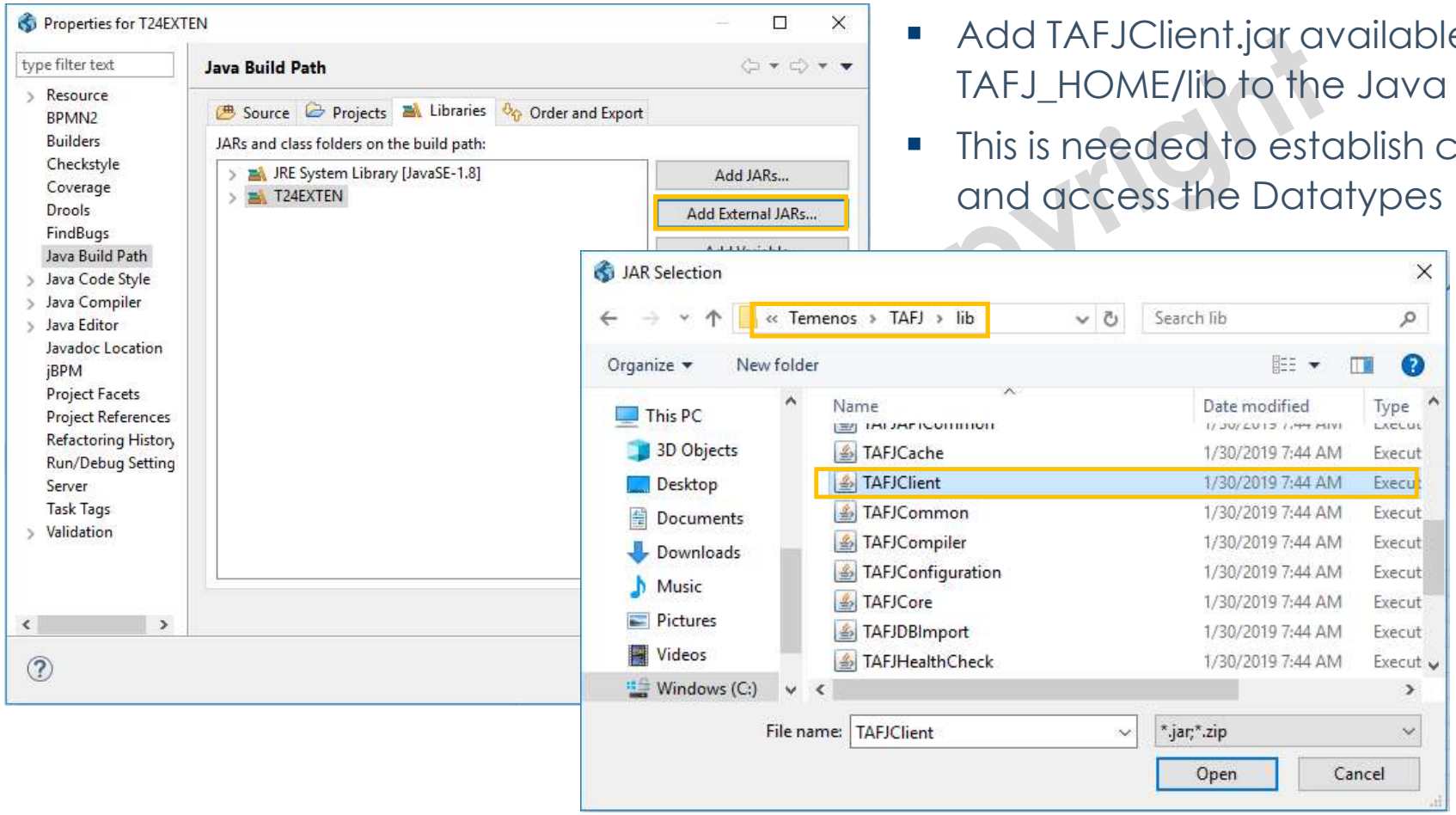
TEMENOS Learning Community

Configure Build Path – T24 Precompile



TEMENOS Learning Community

Configure Build Path - TAFJClient

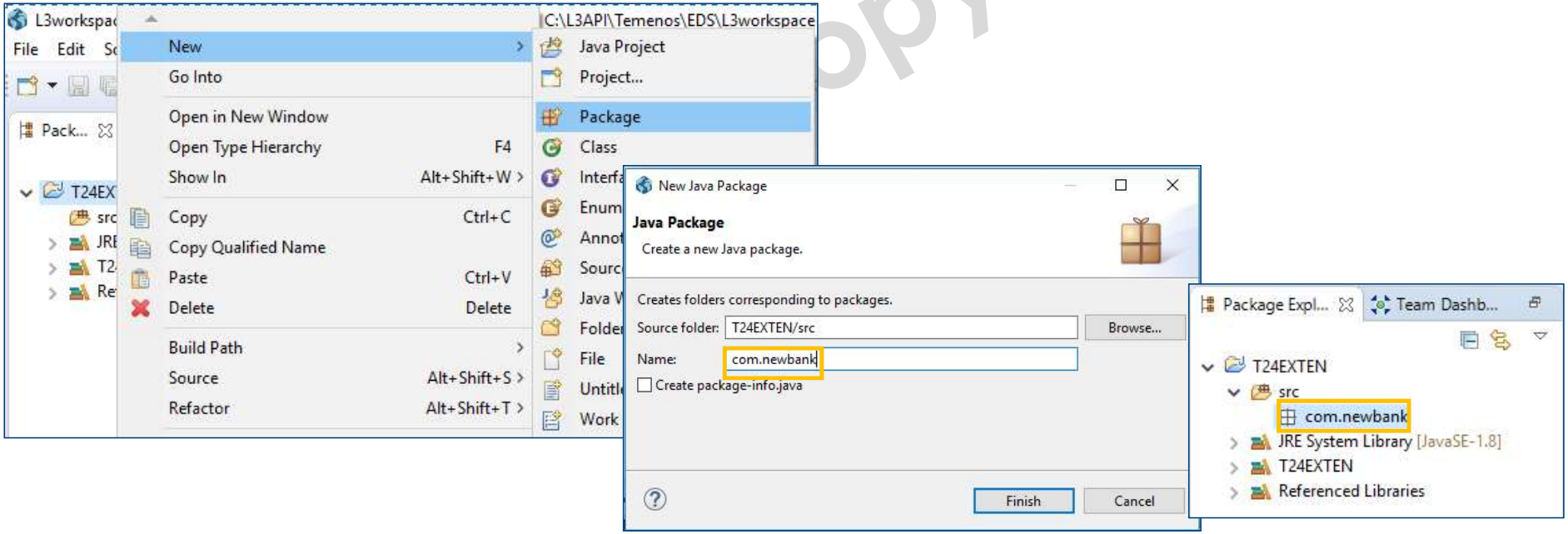


- Add TAFJClient.jar available in TAFJ_HOME/lib to the Java Build Path
- This is needed to establish connection and access the Datatypes

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.



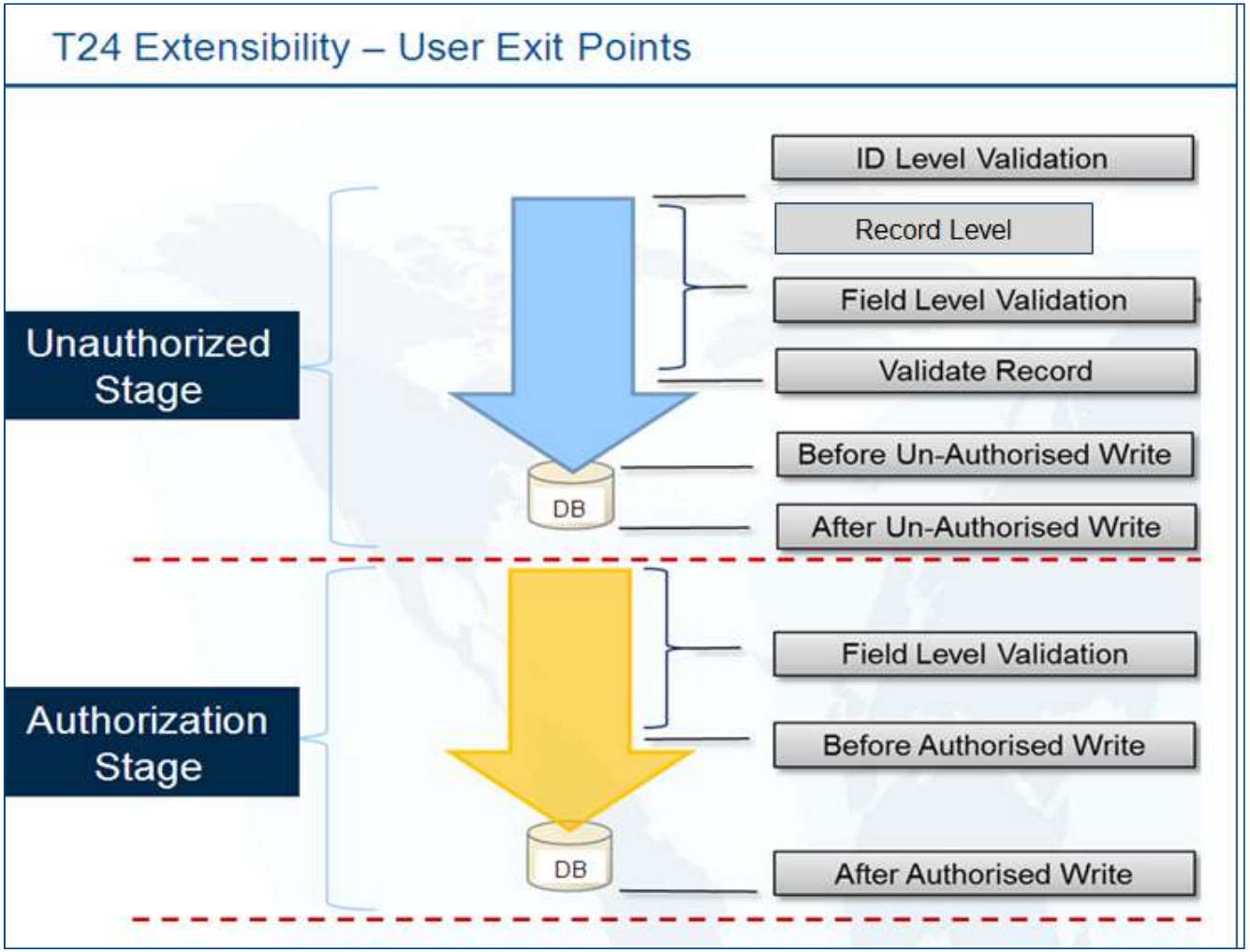
TEMENOS Learning Community

Lesson 2. VERSION Hooks in Java

T24 Extensibility in Java

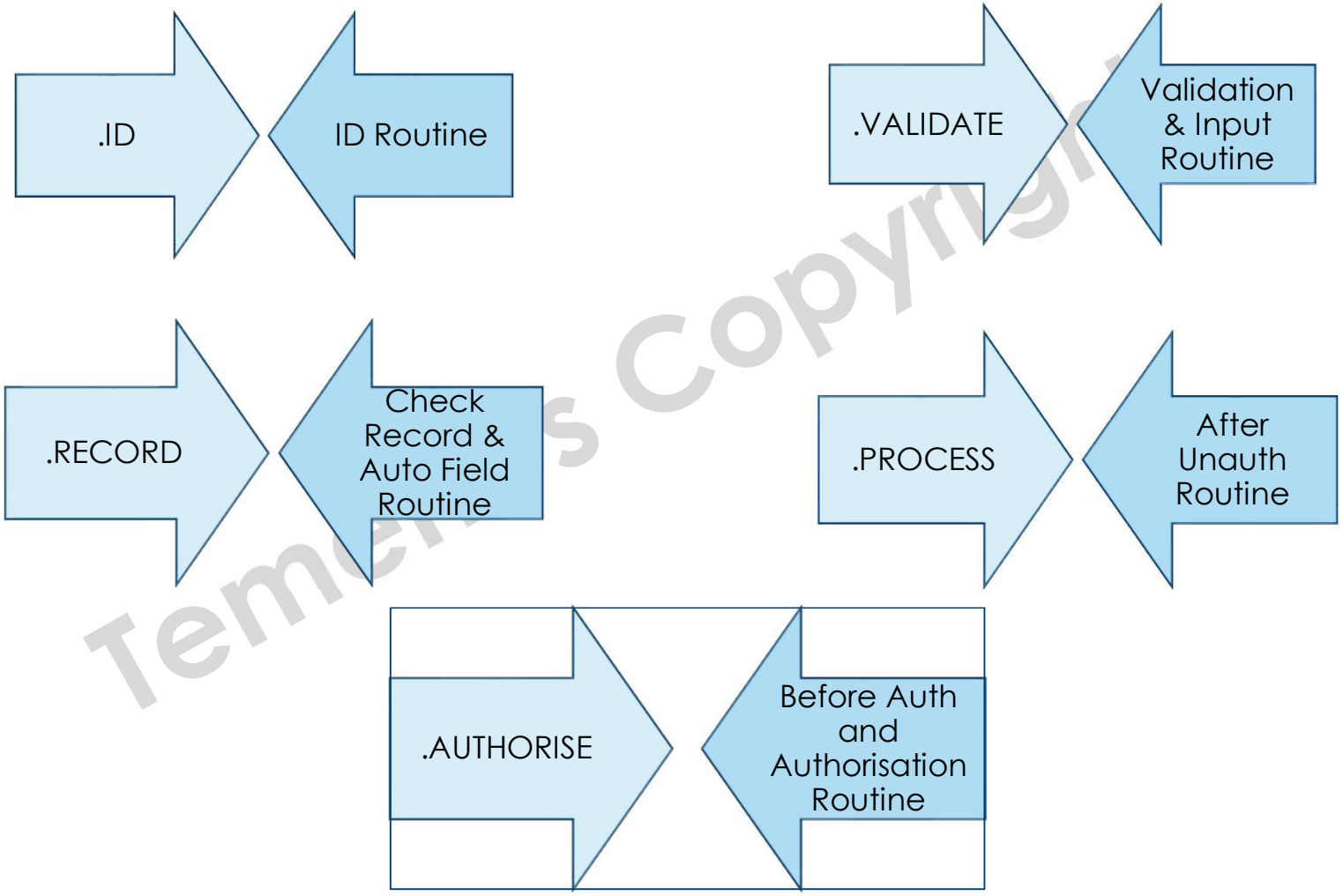
TEMENOS
Learning Community

T24 Extensibility



TEMENOS Learning Community

Routine Invocation Stages in the Template Life Cycle



VERSION HOOKS in T24

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

DEMO – VERSION HOOK

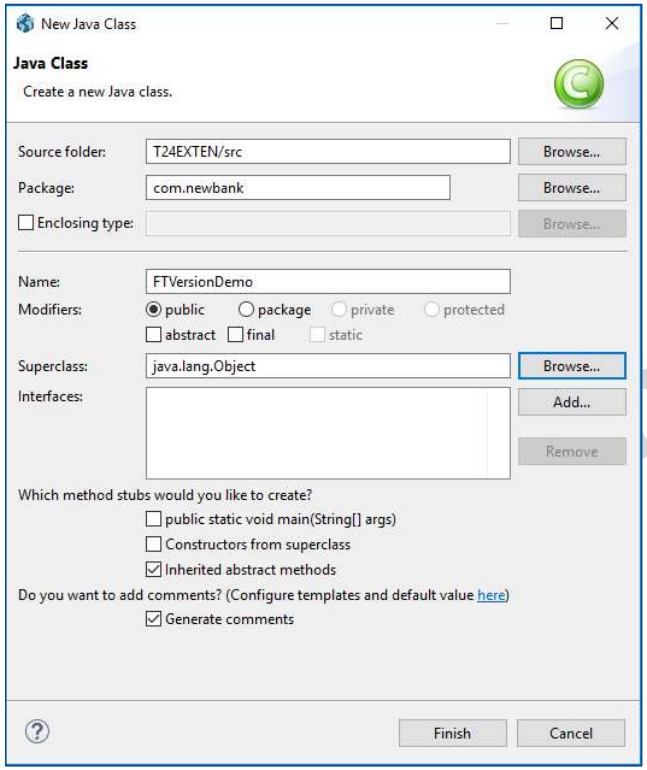
For a FUNDS.TRANSFER transaction, error must be raised when the DEBIT.CURRENCY and CREDIT.CURRENCY are not equal

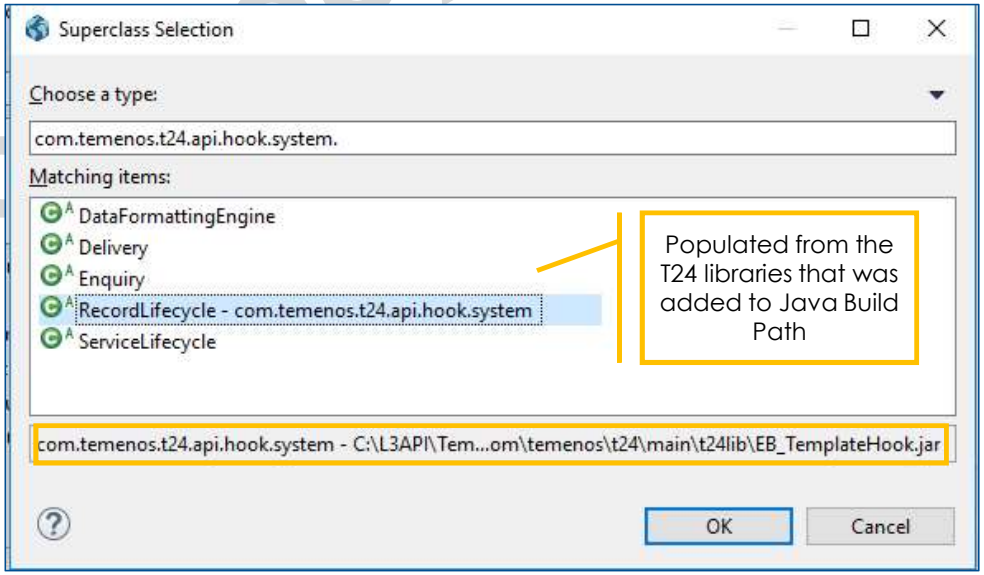
Solution

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

Step 1 – Create class

- Right Click on your Package →New →Class

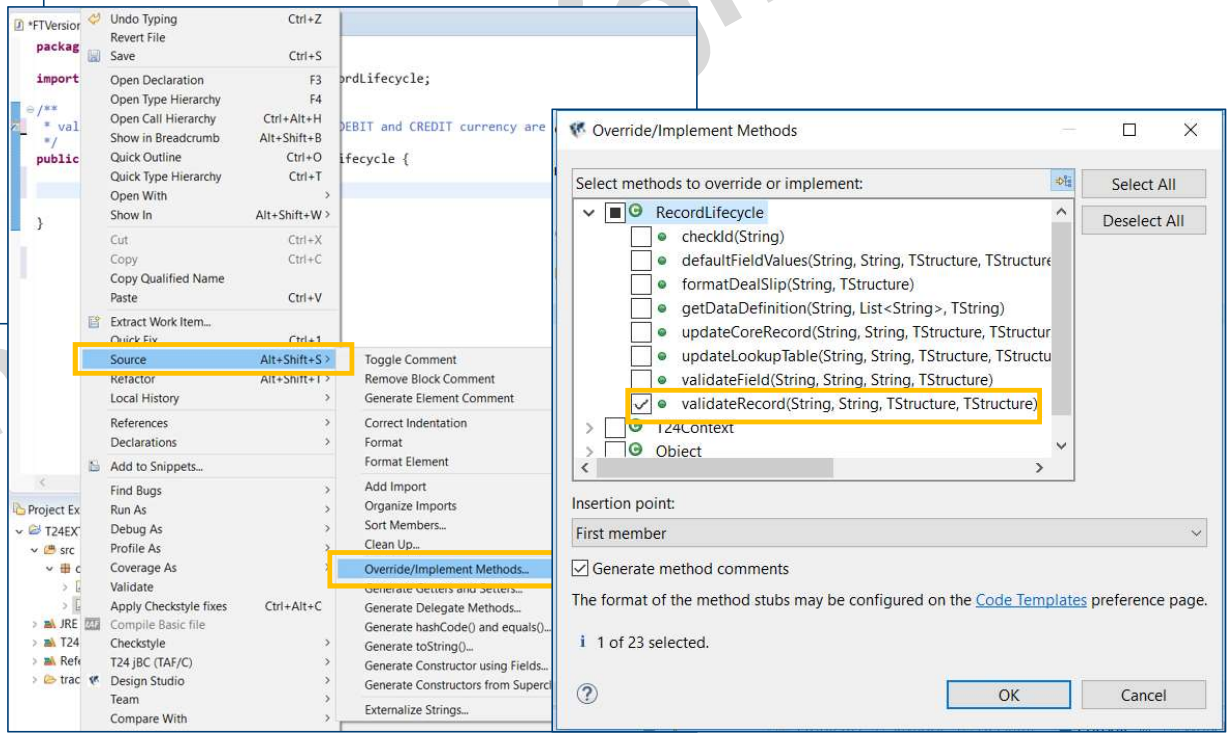




TEMENOS Learning Community

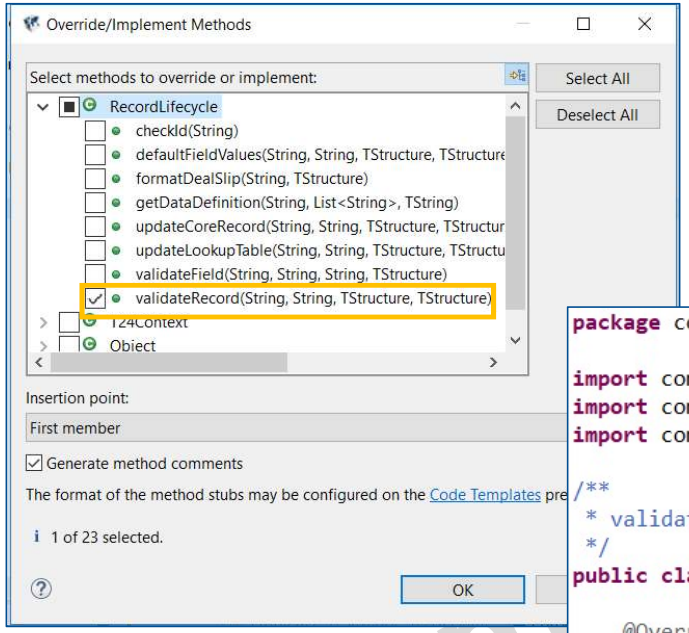
Step 1 – implement validateRecord

```
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 1 – 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

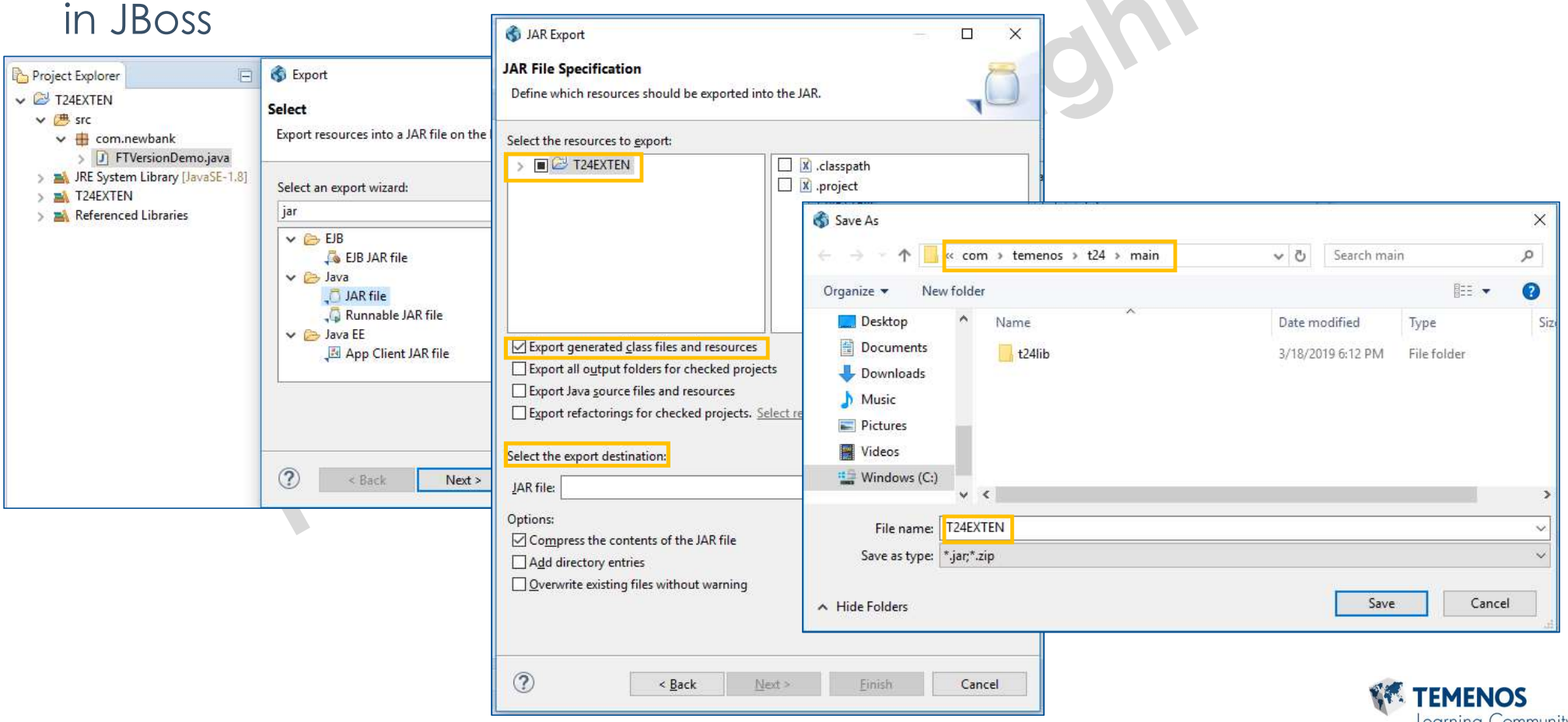
Step 1 – implement validateRecord

```
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 2 – Create the JAR and add to module.xml

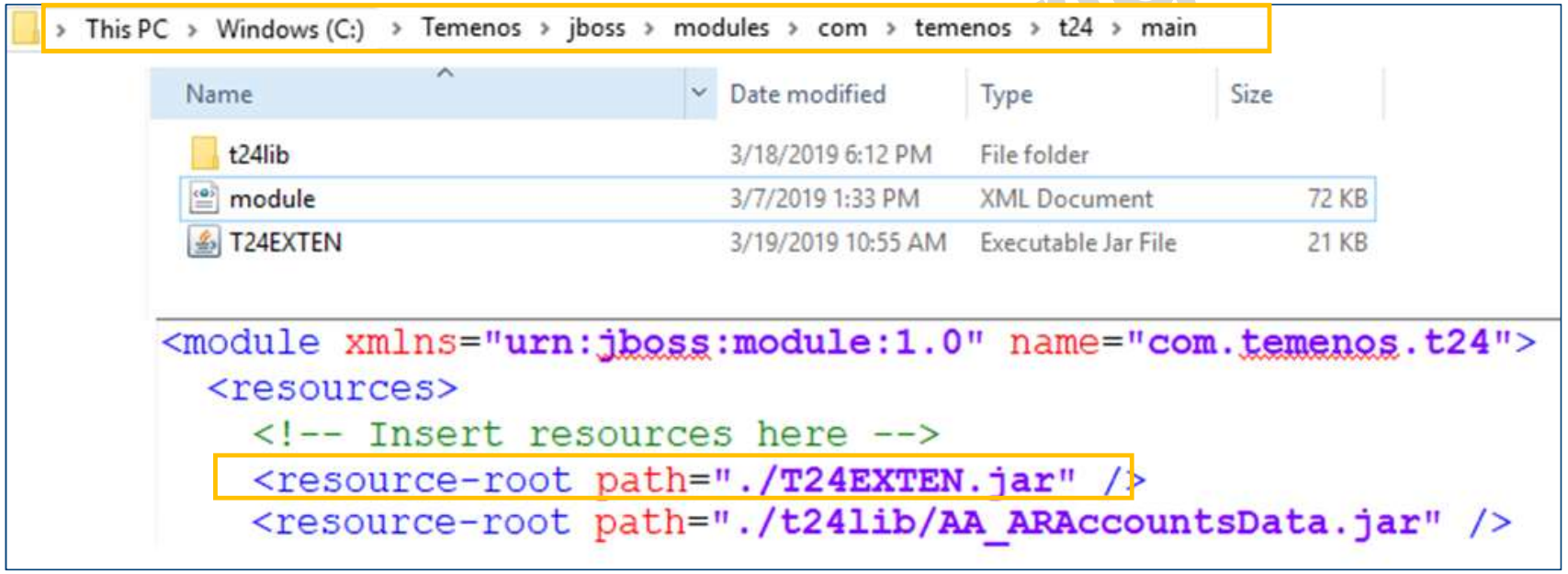
- The .class file must be exported as a JAR and added to the module.xml in JBoss



TEMENOS Learning Community

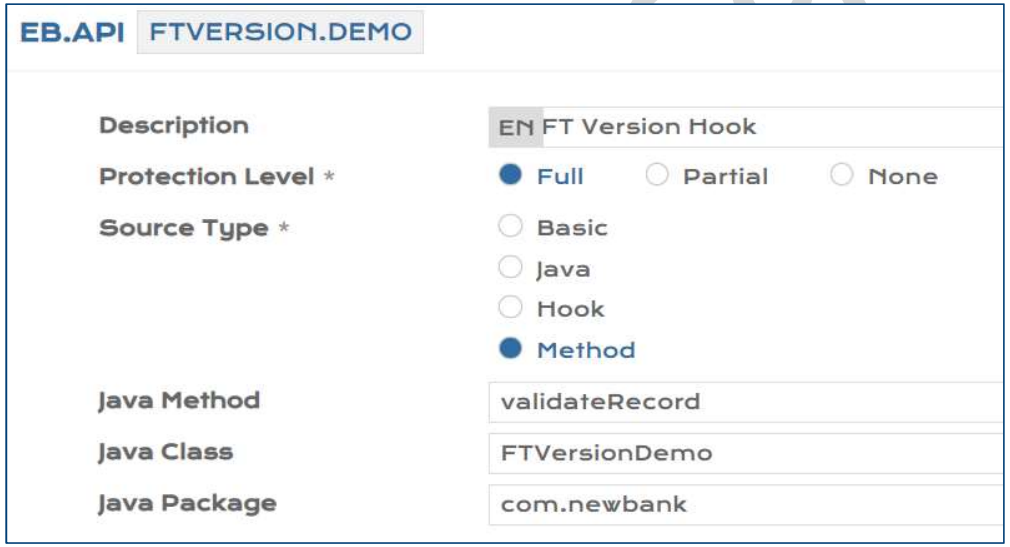
Step 2 – Create the JAR and add to module.xml

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



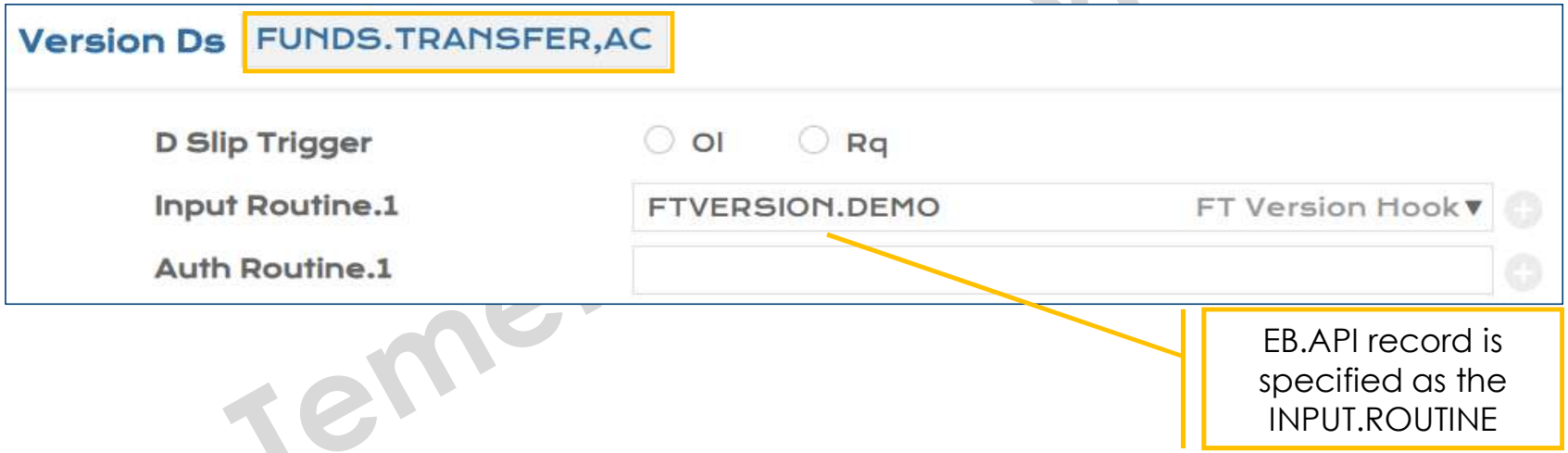
Step 3 – EB.API

- 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



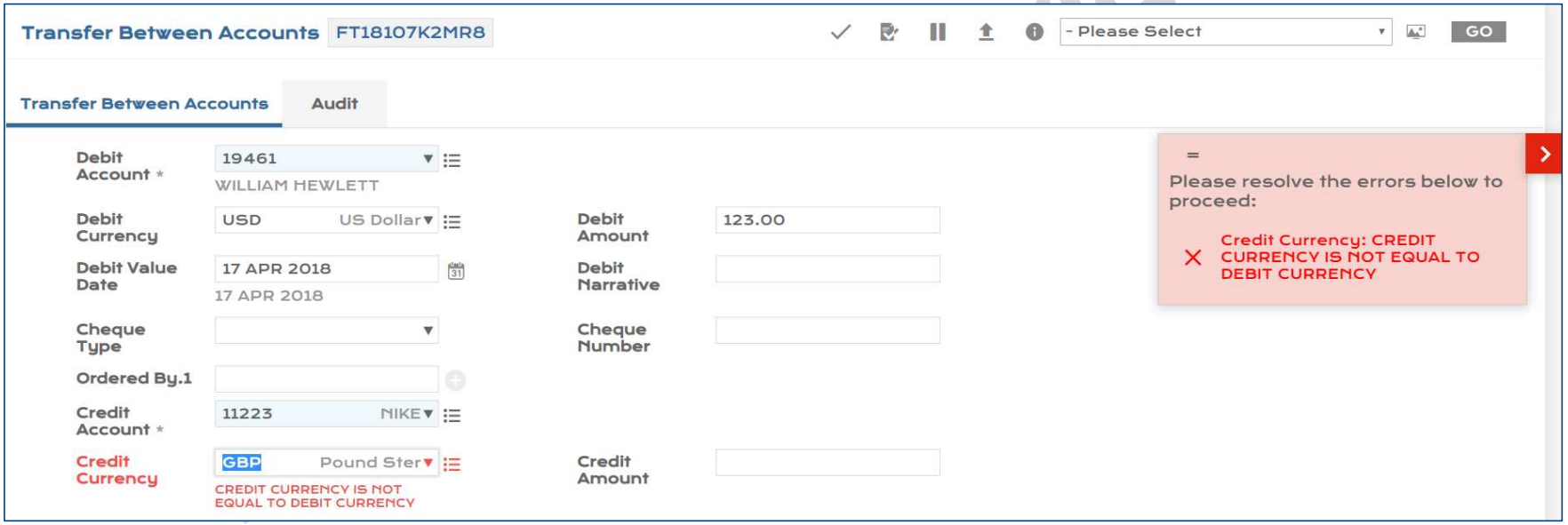
Step 4 – Create/Modify version

- Modify the version FUNDS.TRANSFER,AC



Launching the VERSION

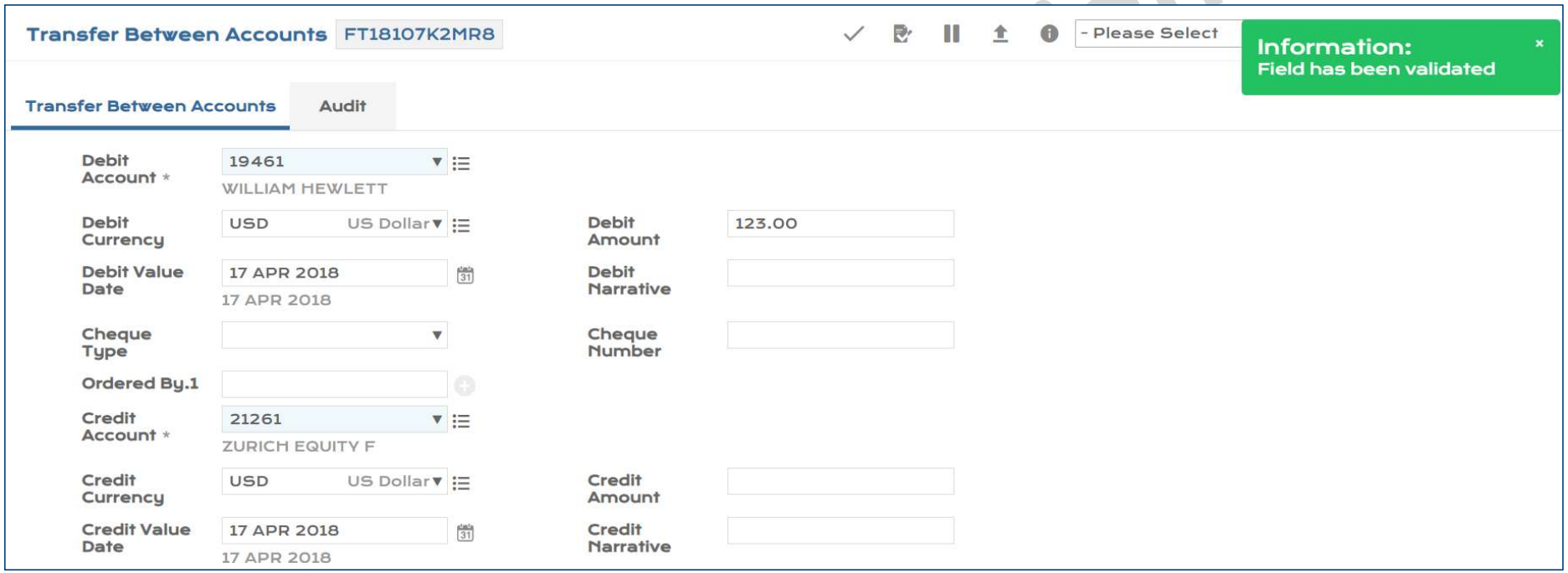
- Error is raised when DEBIT.CURRENCY is not equal to CREDIT.CURRENCY



TEMENOS Learning Community

Launching the VERSION

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



TEMENOS Learning Community

Lesson 3. DEBUG Java code in Design Studio

T24 Extensibility in Java

TEMENOS
Learning Community

DEMO – 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

DEBUG the Java code

- Remote Debug jBoss from Design Studio
- Set JAVA_OPTS

For Windows

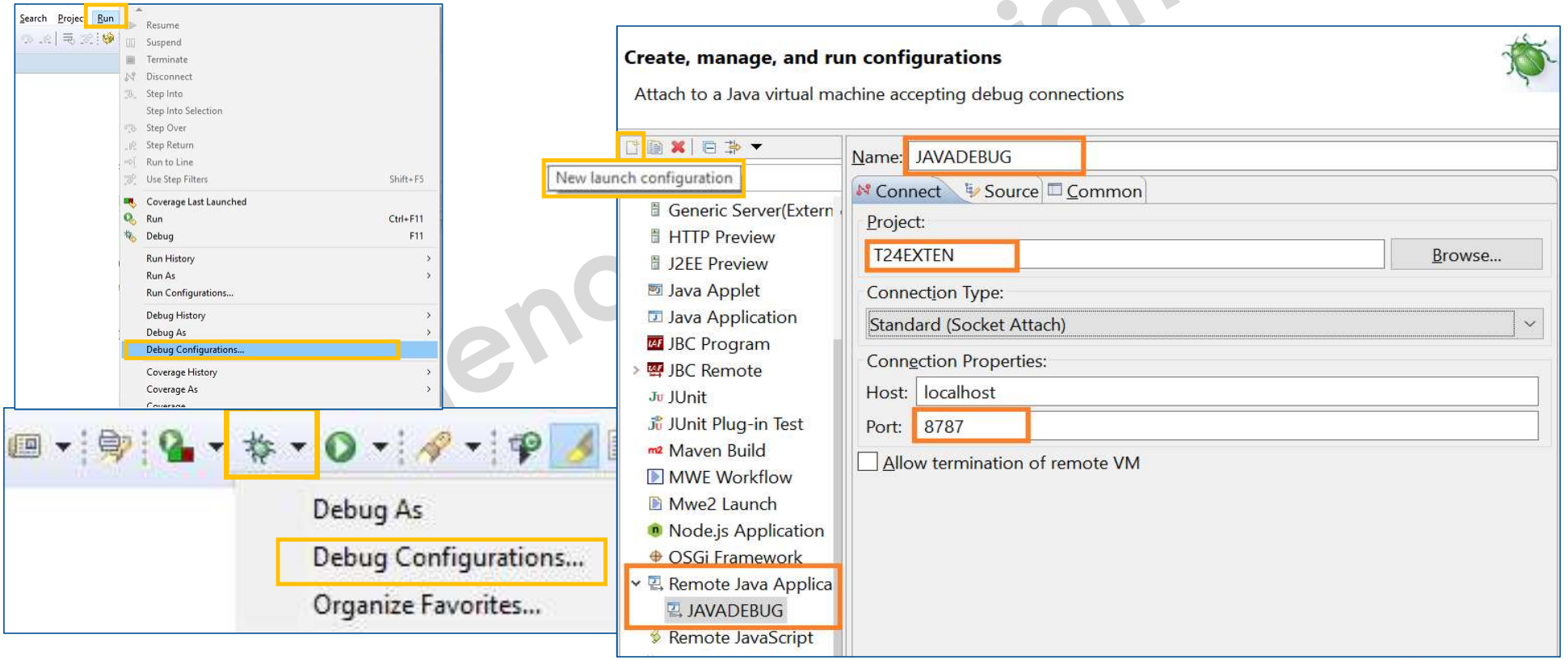
- SET JAVA_OPTS=-Xdebug -Xnoagent -Xrunjdwp:transport=dt_socket, address=8787,server=y,suspend=n%JAVA_OPTS%

For Linux

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

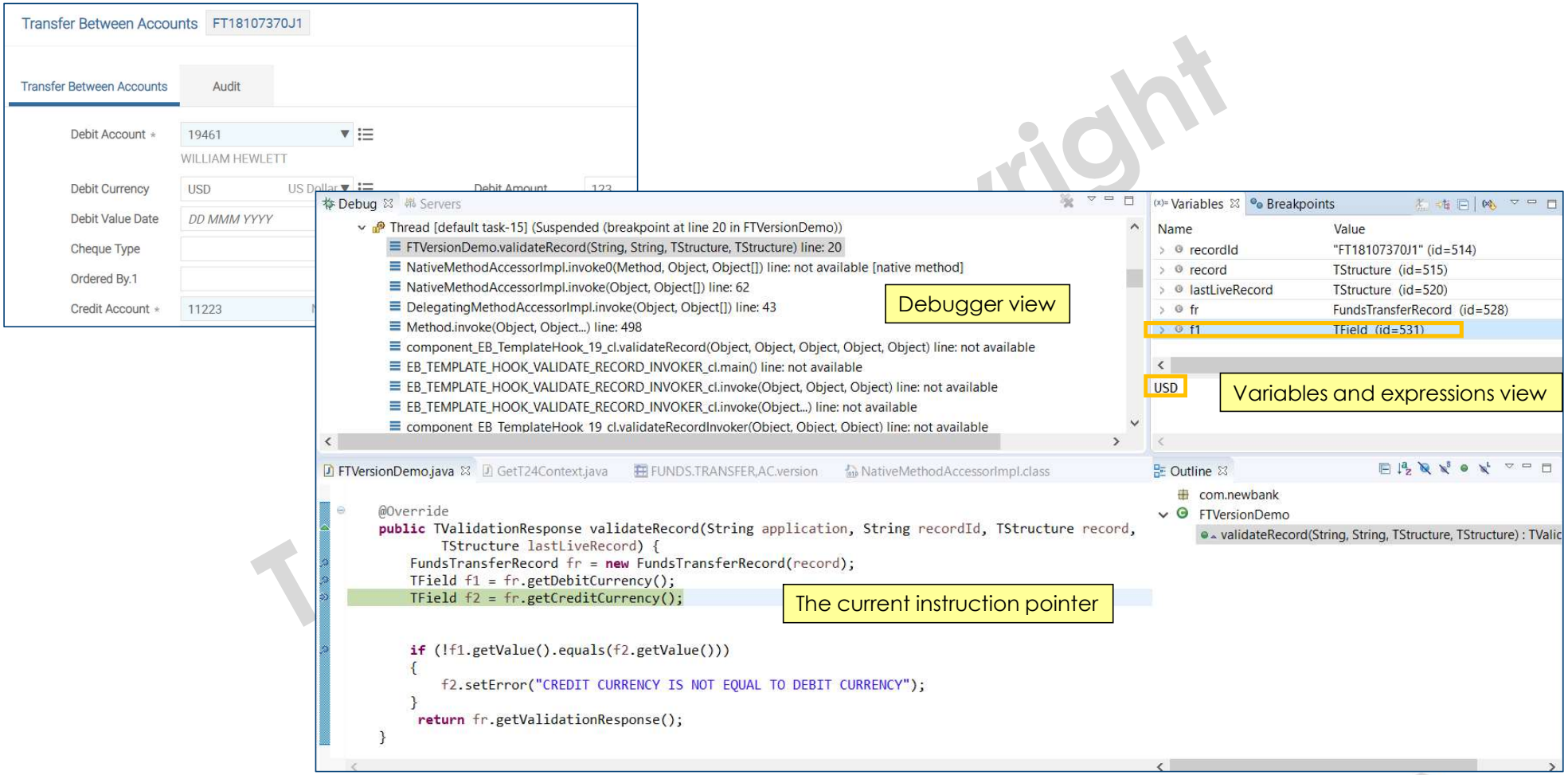
DEBUG the Java code

- Create the DEBUG configuration



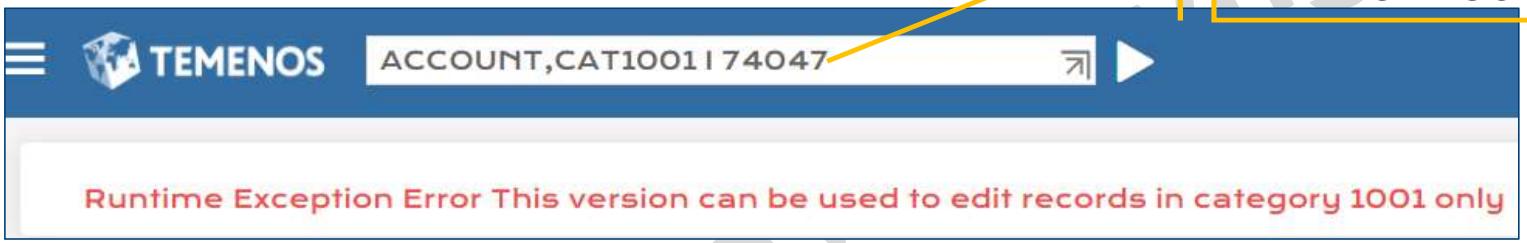
TEMENOS Learning Community

DEBUG the java code



Practice – 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 – Version Hook for ID

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

Lesson 3. ENQUIRY Hooks in Java

T24 Extensibility in Java

TEMENOS
Learning Community

Enquiry hooks

We can customize ENQUIRY by

- modifying the dynamic selection box – BUILD hook
- modifying the data before display – CONVERSION hook

Extend the **com.temenos.t24.api.hook.system.Enquiry** class

- setRecord()
- setFilterCriteria()
- setValue()
- setIds()

DEMO – Enquiry Build hook

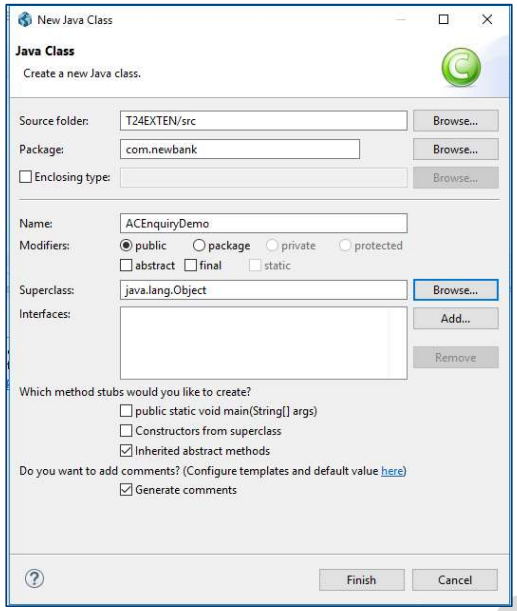
- The Build hook in ENQUIRY, is used to modify the selection criteria
- We can set the Field Name, Operator and the value and add these to the filter criteria

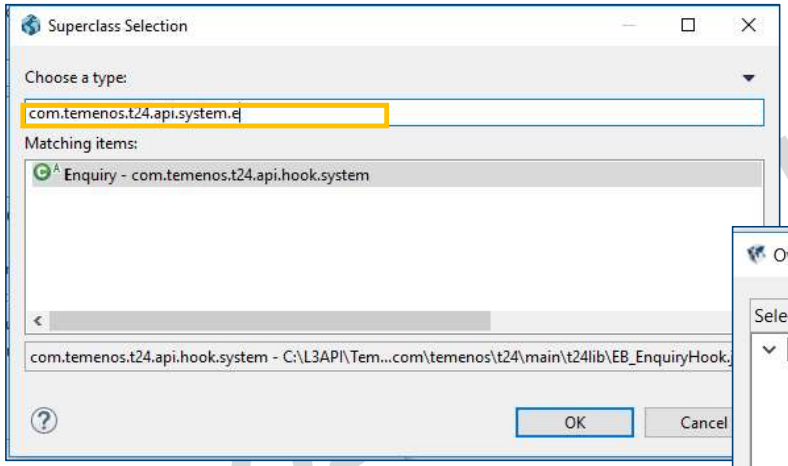
Create an enquiry to display Customer Id, Account Id and the respective working balance after accepting category from the user.

Based on category, additional filtering has to be done

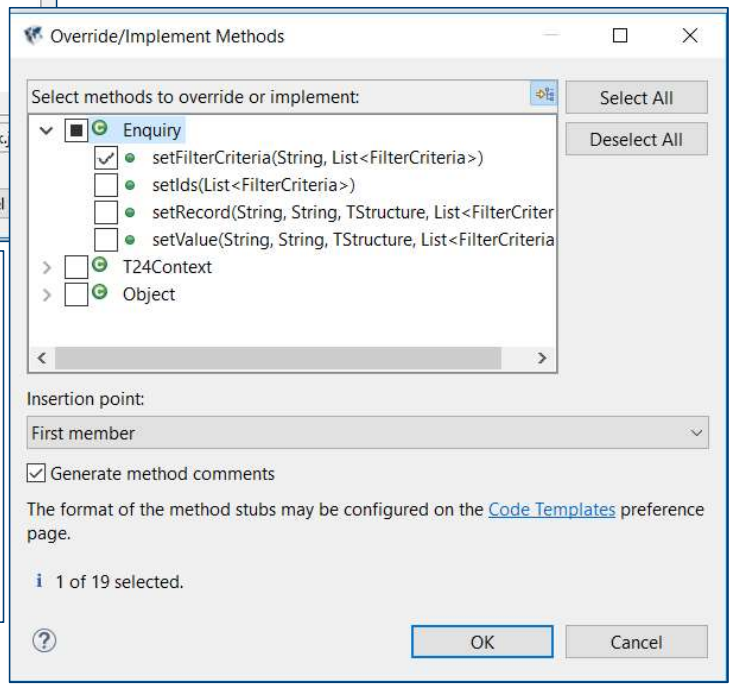
- For category 1001 - Balance should be in the range 0 and 49999
- For category 6001 - Balance should be in the range 50000 and 100000
- For category greater than 7000 - Balance should be in the range 100000 and 500000

Step 1 – Create the class






```
package com.newbank;

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

/**
demo for Build routine
 */
public class ACEnquiryDemo extends Enquiry {


}
```

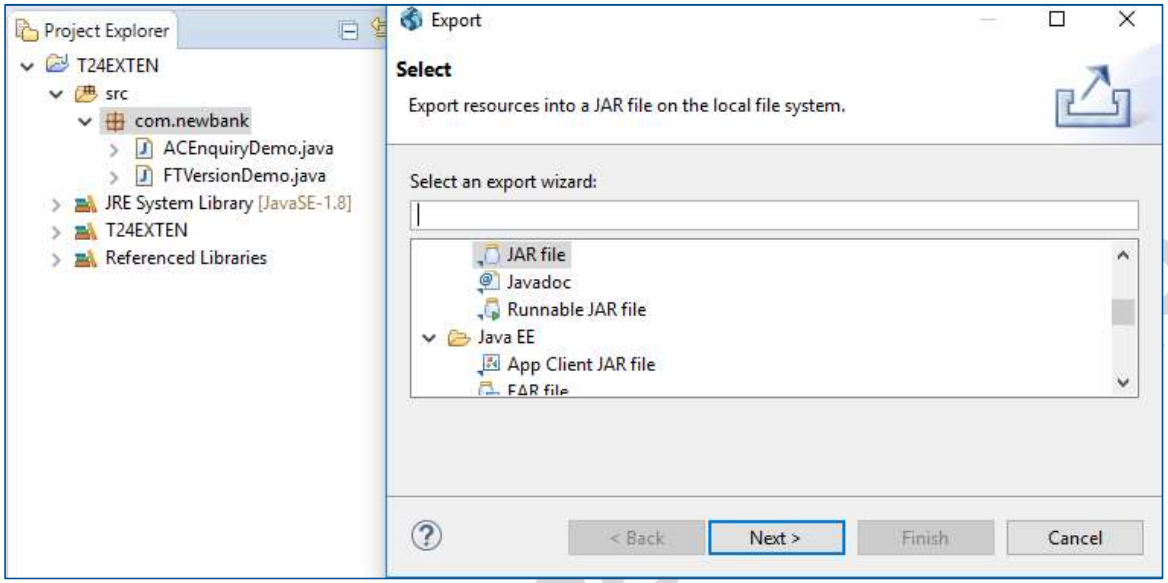


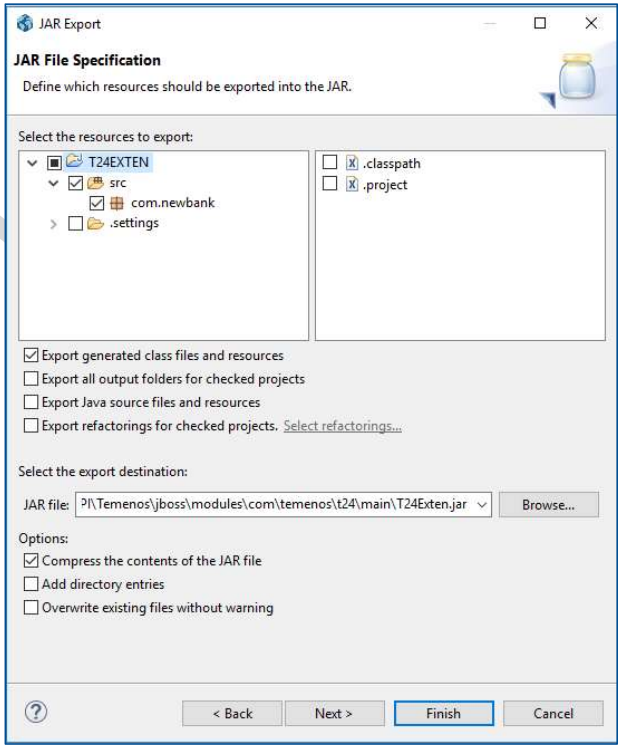
TEMENOS Learning Community

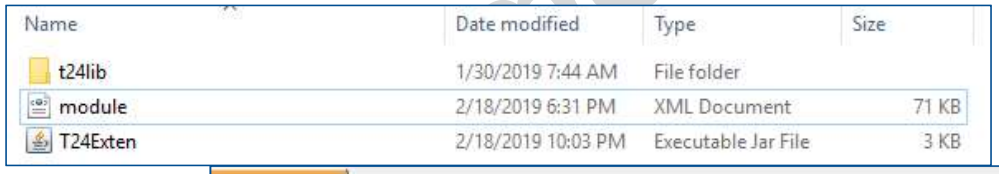
Step 1 – Create the class

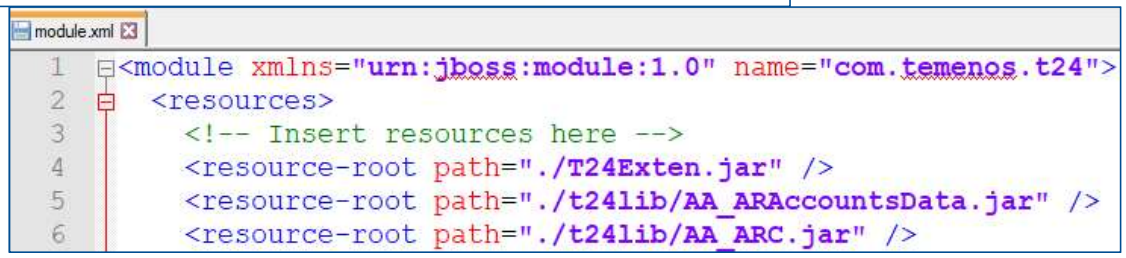
```
package com.newbank;

import java.util.List;

import com.temenos.t24.api.complex.eb.enquiryhook.FilterCriteria;
import com.temenos.t24.api.hook.system.Enquiry;

/**
 demo for ENQUIRY Build hook
 */
public class ACEnquiryDemo extends Enquiry {

    @Override
    public List<FilterCriteria> setFilterCriteria(String enquiryName, List<FilterCriteria> filterCriteria) {
        FilterCriteria fc = new FilterCriteria();
        String s1 = filterCriteria.get(0).getFieldname();
        String s3 = filterCriteria.get(0).getValue();

        fc.setFieldname("WORKING.BALANCE");
        fc.setOperand("RG");
        if (s1.equals("CATEGORY")) {
            if (Integer.parseInt(s3) > 7000) {
                fc.setValue("50000 100000");
            } else {
                switch (s3) {
                case "1001":
                    fc.setValue("0 10000");
                    break;
                case "6001":
                    fc.setValue("10000 50000");
                    break;
                default:
                }
            }

        }
        filterCriteria.add(fc);
        return filterCriteria;
    }
}
```

Step 2 – Create JAR and update module.xml





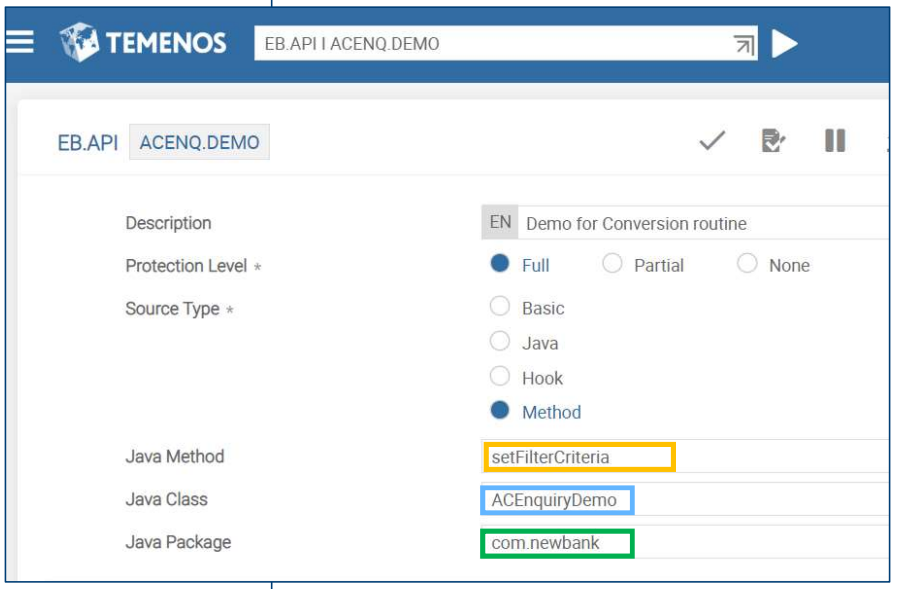




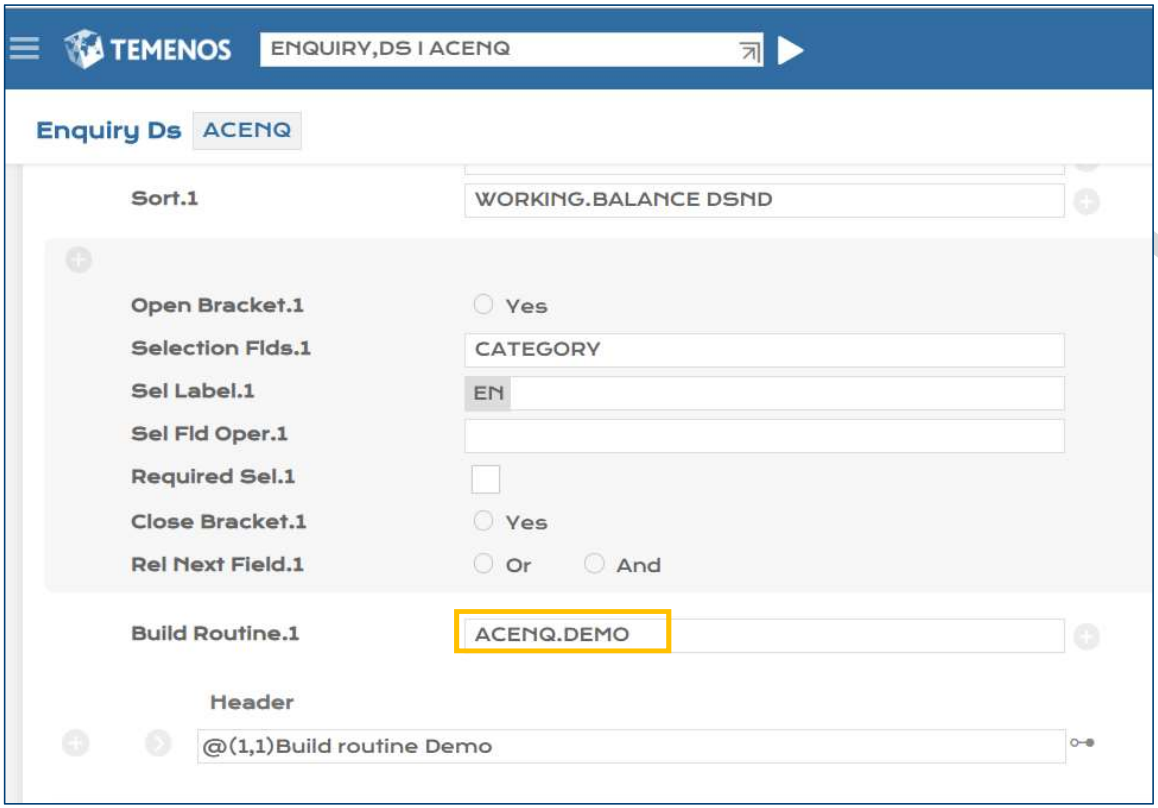
TEMENOS
Learning Community

Step 3 – EB.API

```java
package com.newbank;

import java.util.List;

import com.temenos.t24.api.complex.eb.enquiryhook.FilterCriteria;
import com.temenos.t24.api.hook.system.Enquiry;

/**
 demo for ENQUIRY Build hook
 */
public class ACEnquiryDemo extends Enquiry {

    @Override
    public List<FilterCriteria> setFilterCriteria(String enquiryName, List<FilterCriteria> filterCriteria) {
        FilterCriteria fc = new FilterCriteria();
        String s1 = filterCriteria.get(0).getFieldname();
        String s3 = filterCriteria.get(0).getValue();

        fc.setFieldname("WORKING.BALANCE");
        fc.setOperand("RG");
        if (s1.equals("CATEGORY")) {
            if (Integer.parseInt(s3) > 7000) {
                fc.setValue("50000 100000");
            } else {
                switch (s3) {
                case "1001":
                    fc.setValue("0 10000");
                    break;
                case "6001":
                    fc.setValue("10000 50000");
                    break;
                default:
                }
            }

        }
        filterCriteria.add(fc);
        return filterCriteria;
    }
}
```

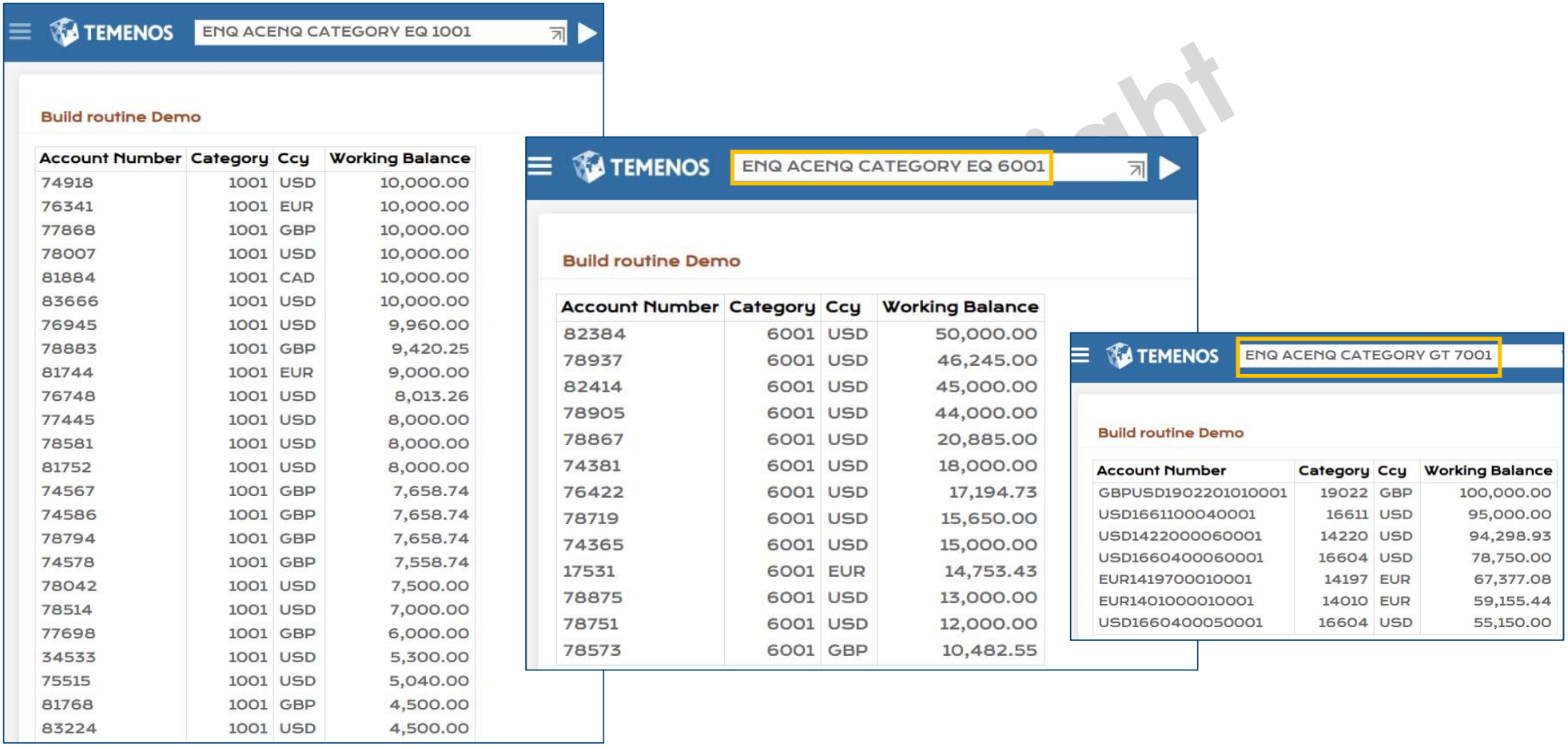


Step 4 – Create ENQUIRY and attach hook



TEMENOS Learning Community

Launch the ENQUIRY



Account Number	Category	Ccy	Working Balance
74918	1001	USD	10,000.00
76341	1001	EUR	10,000.00
77868	1001	GBP	10,000.00
78007	1001	USD	10,000.00
81884	1001	CAD	10,000.00
83666	1001	USD	10,000.00
76945	1001	USD	9,960.00
78883	1001	GBP	9,420.25
81744	1001	EUR	9,000.00
76748	1001	USD	8,013.26
77445	1001	USD	8,000.00
78581	1001	USD	8,000.00
81752	1001	USD	8,000.00
74567	1001	GBP	7,658.74
74586	1001	GBP	7,658.74
78794	1001	GBP	7,658.74
74578	1001	GBP	7,558.74
78042	1001	USD	7,500.00
78514	1001	USD	7,000.00
77698	1001	GBP	6,000.00
34533	1001	USD	5,300.00
75515	1001	USD	5,040.00
81768	1001	GBP	4,500.00
83224	1001	USD	4,500.00

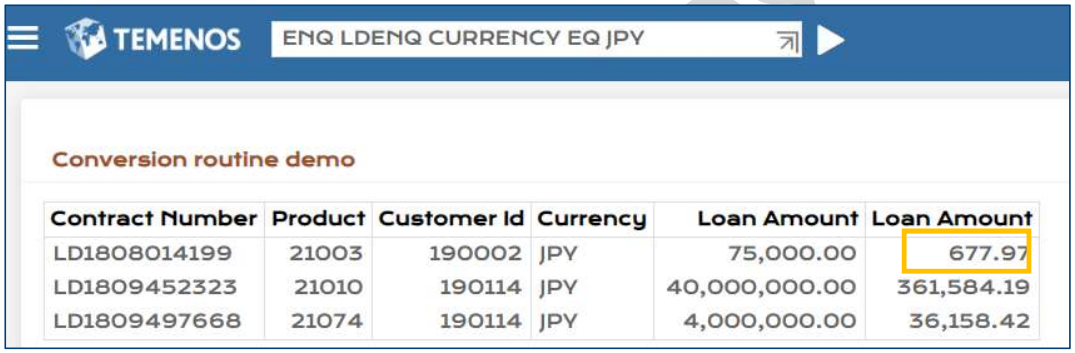
Account Number	Category	Ccy	Working Balance
82384	6001	USD	50,000.00
78937	6001	USD	46,245.00
82414	6001	USD	45,000.00
78905	6001	USD	44,000.00
78867	6001	USD	20,885.00
74381	6001	USD	18,000.00
76422	6001	USD	17,194.73
78719	6001	USD	15,650.00
74365	6001	USD	15,000.00
17531	6001	EUR	14,753.43
78875	6001	USD	13,000.00
78751	6001	USD	12,000.00
78573	6001	GBP	10,482.55

Account Number	Category	Ccy	Working Balance
GBPUSD1902201010001	19022	GBP	100,000.00
USD1661100040001	16611	USD	95,000.00
USD1422000060001	14220	USD	94,298.93
USD1660400060001	16604	USD	78,750.00
EUR1419700010001	14197	EUR	67,377.08
EUR1401000010001	14010	EUR	59,155.44
USD1660400050001	16604	USD	55,150.00

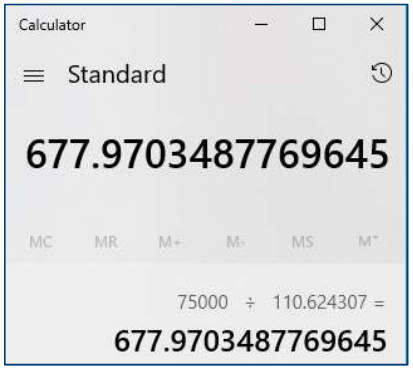
TEMENOS Learning Community

Practice – Enquiry Conversion hook

- Create an ENQUIRY TO LIST the LD records that are in foreign currency. List the ID, CUSTOMER, AMOUNT and Equivalent amount in LCCY.
- Use conversion hook to convert the amount in foreign currency to LCCY
- Expected output



Contract Number	Product	Customer Id	Currency	Loan Amount	Loan Amount
LD1808014199	21003	190002	JPY	75,000.00	677.97
LD1809452323	21010	190114	JPY	40,000,000.00	361,584.19
LD1809497668	21074	190114	JPY	4,000,000.00	36,158.42



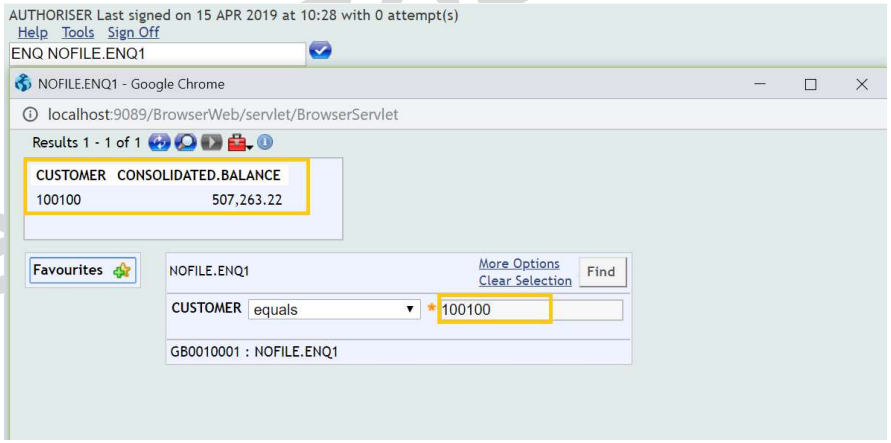
TEMENOS Learning Community

NOFILE Enquiry

- What if there are multiple files involved in creation of an enquiry?
- What if the record ID of another application is not part of the base application?
- This can be achieved using NOFILE enquiries
- Since more than one file might be required to extract data, it is a NOFILE enquiry. We can extract data from any number of T24 applications
- Subroutine is written to extract all required data and form the record
- **ENQUIRY** application is used to display extracted data

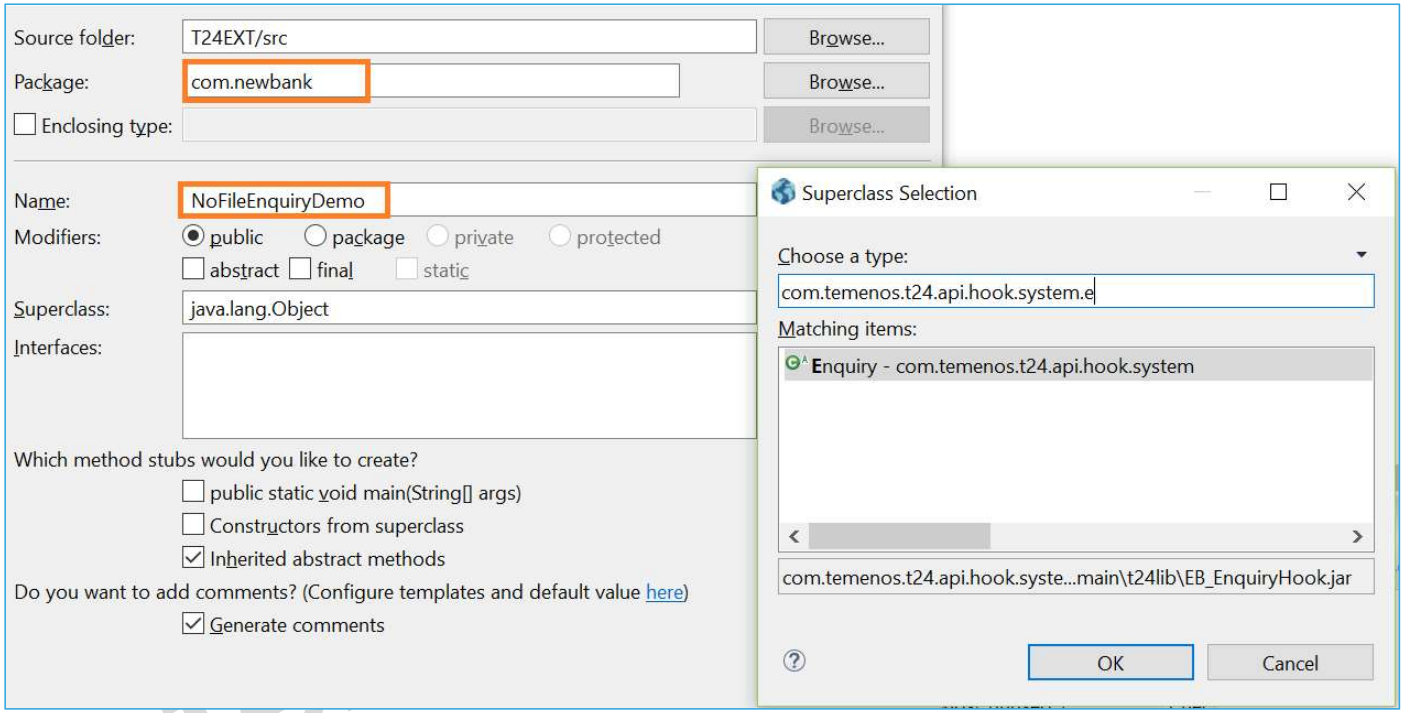
DEMO – NOFILE enquiry

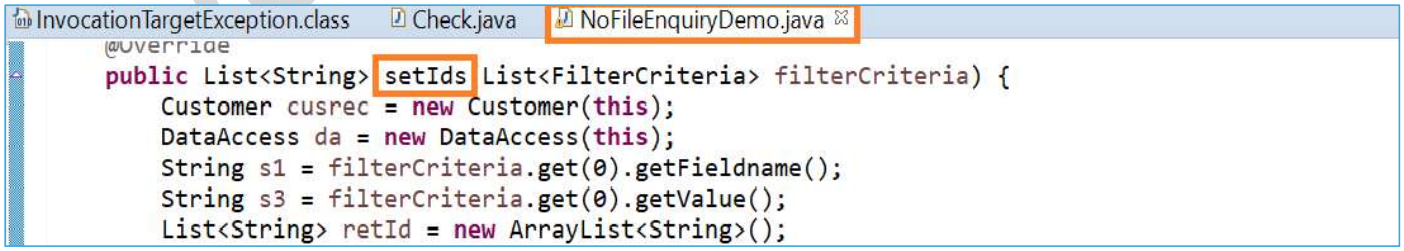
- Create an ENQUIRY to display the consolidated balance for a CUSTOMER.
- Hint : Accumulate all the balance from all the accounts available in CUSTOMER.ACCOUNT.
- Expected Output



TEMENOS Learning Community

Step 1 – Create the class





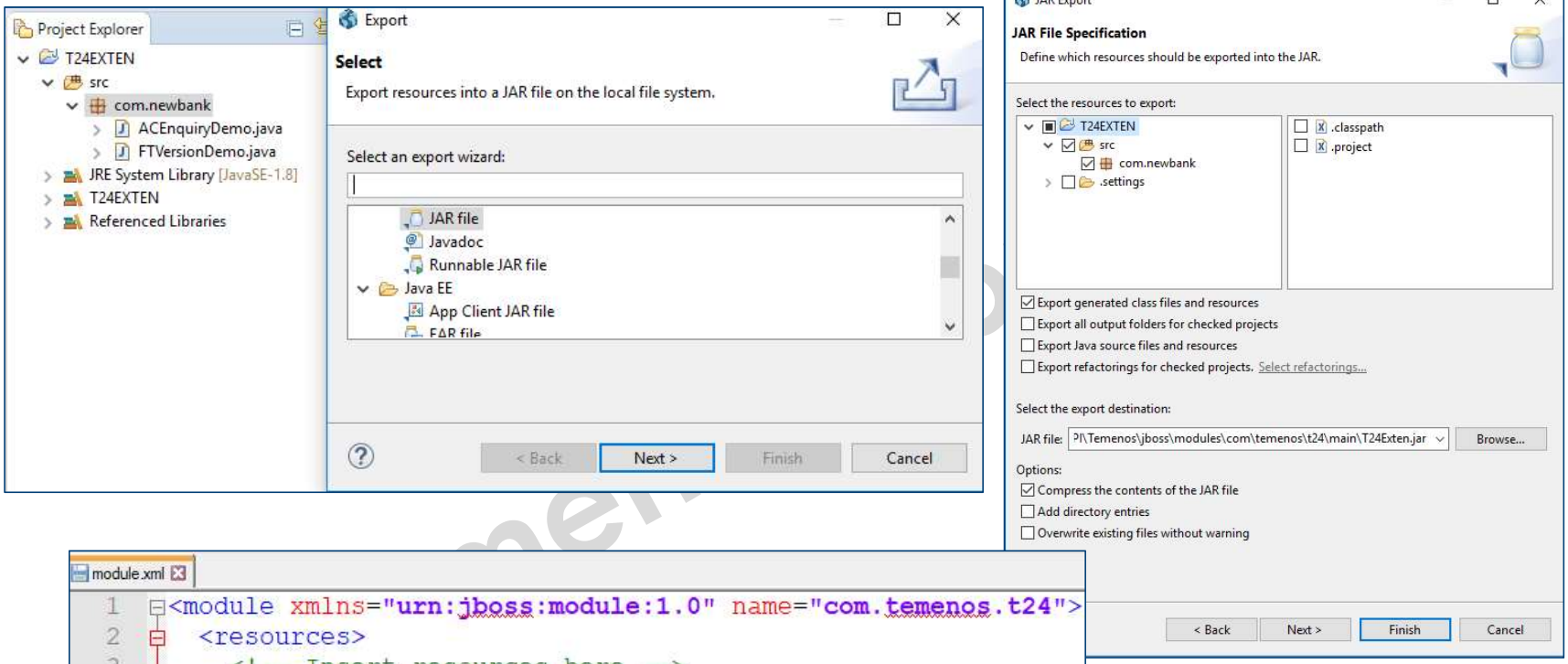
TEMENOS Learning Community

Step 1 – Create the class

NoFileEnquiryDemo.java

```
package com.newbank;

import java.util.ArrayList;

public class NoFileEnquiryDemo extends Enquiry {

    @Override
    public void setRecord(String value, String currentId, TStructure currentRecord,
            List<FilterCriteria> filterCriteria) {
        }

    @Override
    public List<FilterCriteria> setFilterCriteria(String enquiryName, List<FilterCriteria> filterCriteria) {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public String setValue(String value, String currentId, TStructure currentRecord,
            List<FilterCriteria> filterCriteria) {
        // TODO Auto-generated method stub
        return null;
    }
    public List<String> setIds(List<FilterCriteria> filterCriteria) {

        Customer cusrec = new Customer(this);
        DataAccess da = new DataAccess(this);
        String s1 = filterCriteria.get(0).getFieldname();
        String s3 = filterCriteria.get(0).getValue();
        List<String> retId = new ArrayList<String>();
        Double consAmt = 0.0;
       if (s1.equals("CUSTOMER")) {
            if (!s3.equals("")) {
                cusrec.setCustomerId(s3);
                List<String> accnum = null;
                try {
                    accnum = cusrec.getAccountNumbers();
                    for (String acnum : accnum) {
                        AccountRecord ar = new AccountRecord(da.getRecord("ACCOUNT", acnum));
                        String amt = ar.getWorkingBalance().getValue();
                         if(amt.isEmpty() || amt.equals("")) {
                            continue;
                        } else {
                            double amt1 = Double.valueOf(amt);
                            consAmt += amt1;
                        }
                    }

                } catch (Exception e) {
                }
                retId.add(Double.toString(consAmt));
            }
        }
        return retId;
    }
}
```

Step 2 – Create JAR and update module.xml

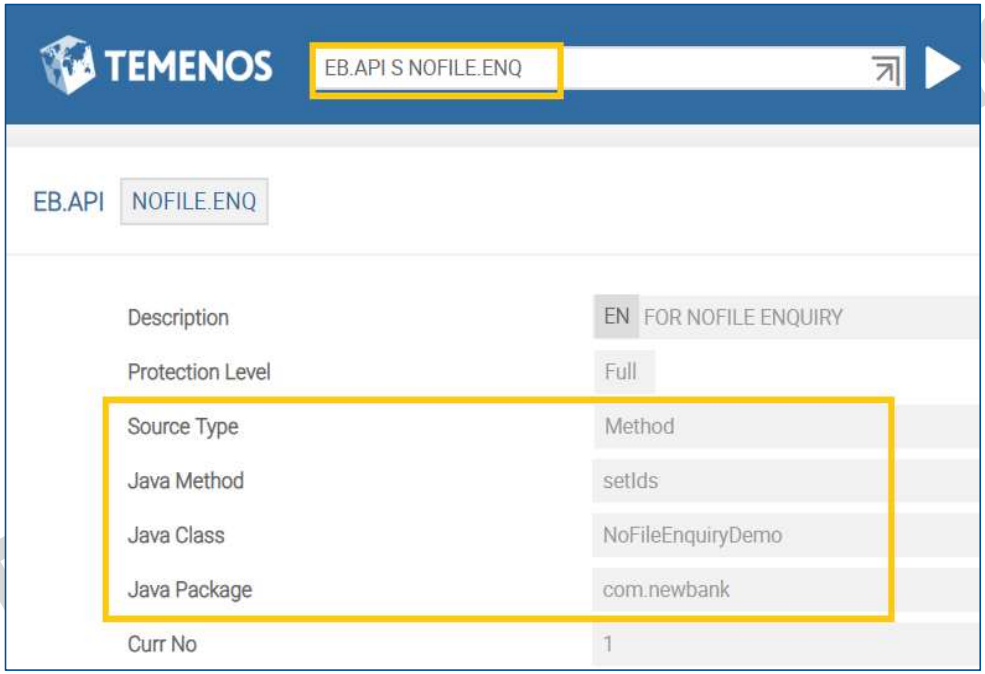


module.xml

```
1  <module xmlns="urn:jboss:module:1.0" name="com.temenos.t24">
2    <resources>
3      <!-- Insert resources here -->
4      <resource-root path="./T24Exten.jar" />
5      <resource-root path="./t24lib/AA_ARAccountsData.jar" />
6      <resource-root path="./t24lib/AA_ARC.jar" />
```

TEMENOS Learning Community

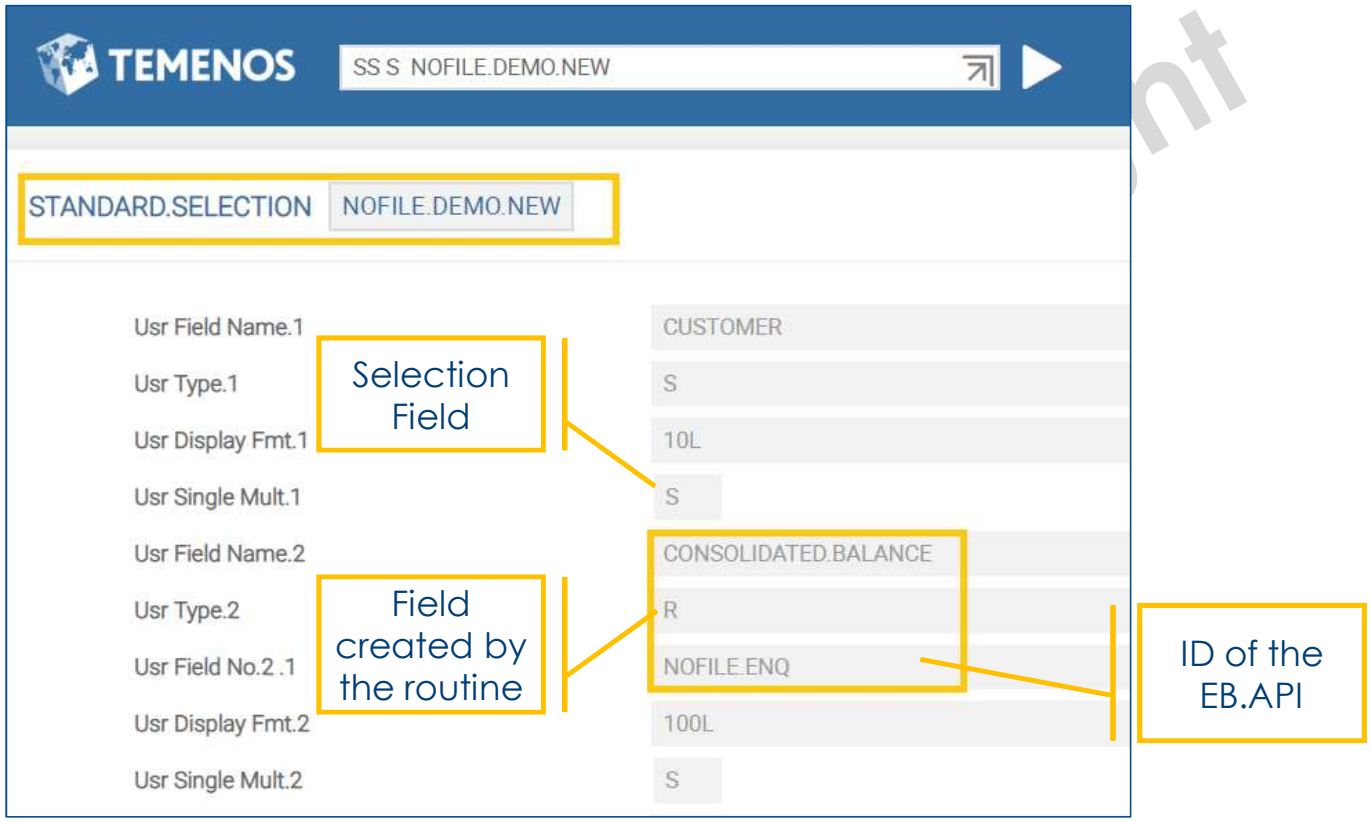
Step 3 – EB.API



Creating the ENQUIRY Record

- A valid application name should be given in the field FILE.NAME which is checked against **STANDARD.SELECTION**
- A NOFILE enquiry does not have a single associated file
- Create a record in **STANDARD.SELECTION** with **NOFILE** prefix for the ID e.g. NOFILE.CUSBALANCE
- This ID should be given in FILE.NAME in the **ENQUIRY** application

NOFILE Enquiry - STANDARD.SELECTION



TEMENOS Learning Community

NoFile Enquiry - ENQUIRY

ENQUIRY NOFILE.DEMO

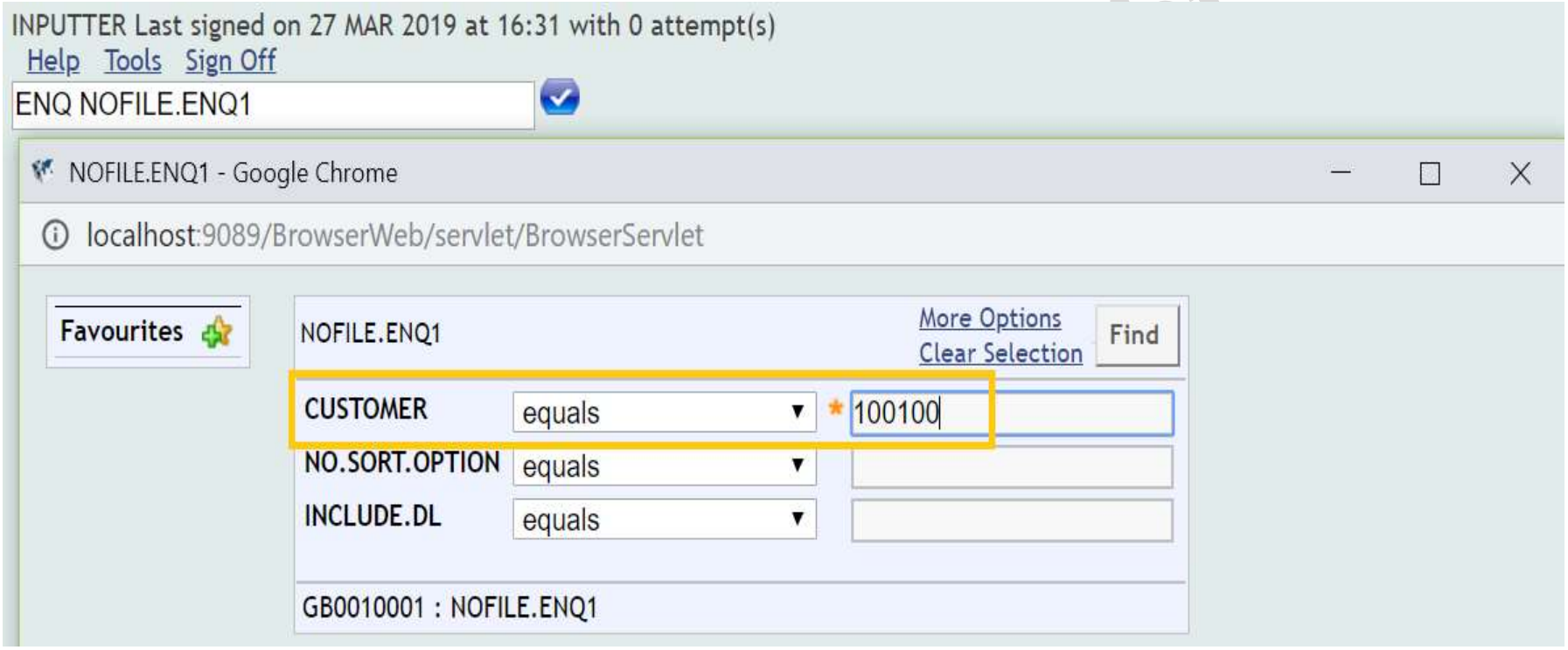
Page Size	4,20
File Name	NOFILE.DEMO
Fixed Selection.1	CONSOLIDATED.BAL NE "
Selection Flds.1	CUSTOMER
Required Sel.1	Y
Field Name.1	CUSTOMER
Operation.1 .1	SELECTION CUSTOMER
Column.1	1
Length Mask.1	10L
Single Multi.1	S
Field Name.2	CONSOLIDATED.BAL
Operation.2 .1	0

ENQUIRY NOFILE.DEMO

Field Name.2	CONSOLIDATED.BAL
Operation.2 .1	0
Column.2	2
Conversion.2 .1	F *,2,1
Page Fields	2
Multi Fields	0
Break Fields	0
Process Breaks	0
Total Fields	0
Curr No	13
Inputter.1	202_INPUTTER__OFS_BROWSERTC
Date time.1	07 AUG 19 13:41
Authoriser	202_INPUTTER_OFS_BROWSERTC

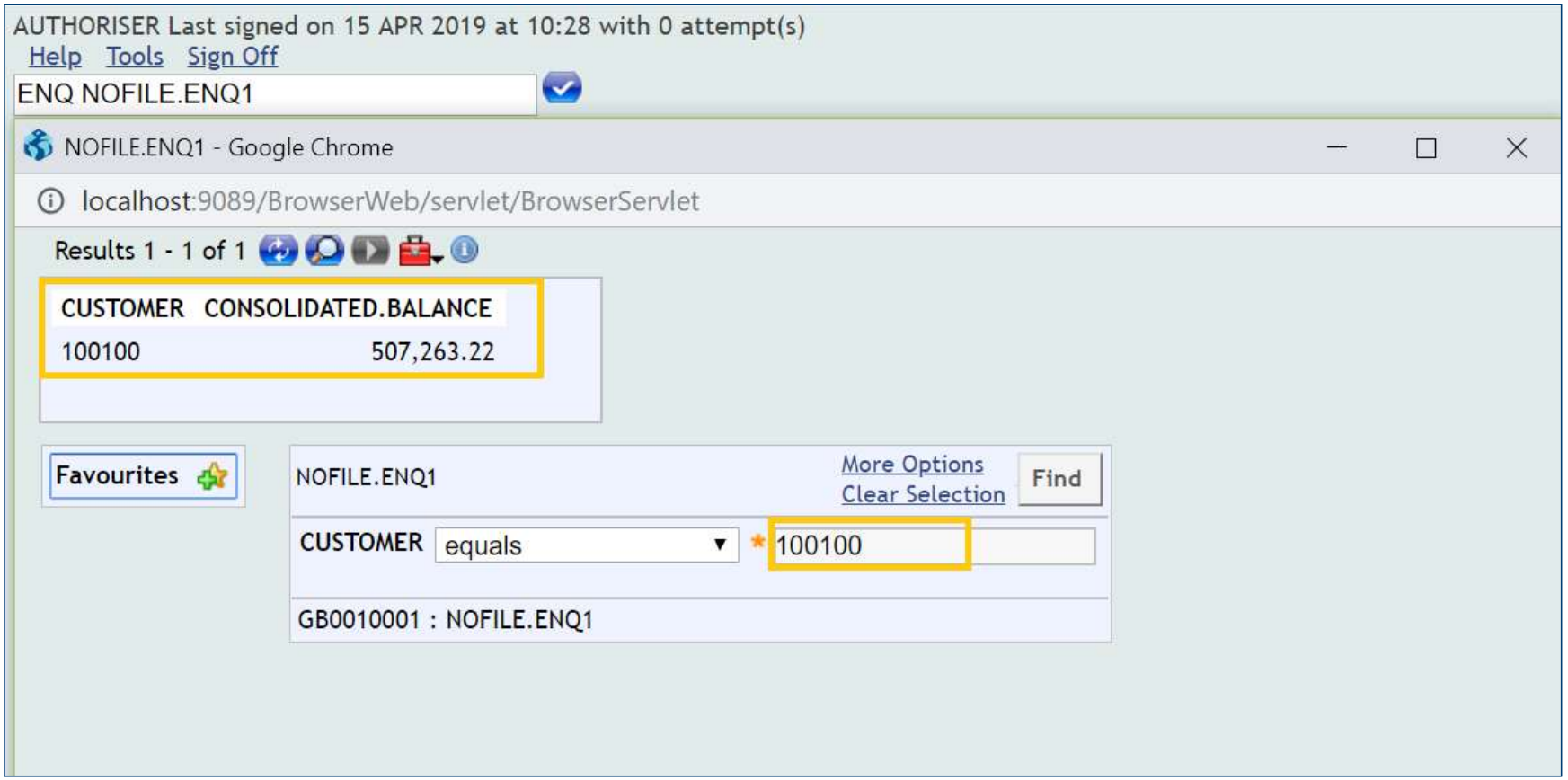
TEMENOS Learning Community

NoFile Enquiry – Execution



TEMENOS Learning Community

NoFile Enquiry – OUTPUT



TEMENOS Learning Community

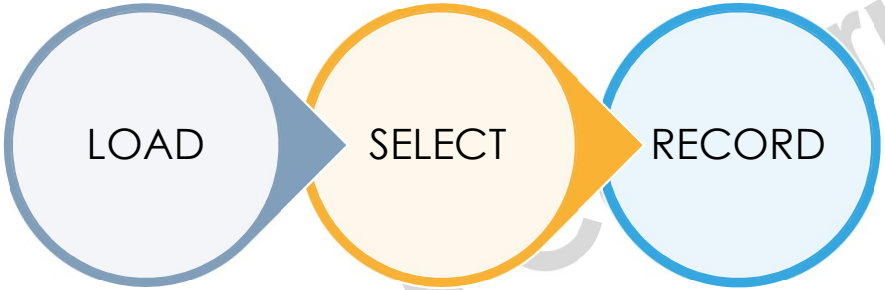
Lesson 4. SERVICE Hooks in Java

T24 Extensibility in Java

TEMENOS
Learning Community

Services in T24

- Multithreaded processes
- Can be run independently or as part of COB



- Must implement **com.temenos.t24.api.hook.system.ServiceLifecycle**
- Equivalent of

  LOAD – initialize()

  SELECT – getIds() or getTableName()

  RECORD – inputRecord() or process()

DEMO- Service

- Simple service that updates the SHORT.NAME of CUSTOMERS in SECTOR 1001 as TEST + existing MNEMONIC

- EXPECTED OUTPUT

1. Create the class

```java
package com.newbank;

import java.util.List;

import com.temenos.api.TBoolean;
import com.temenos.api.TStructure;
import com.temenos.t24.api.complex.eb.servicehook.ServiceData;
import com.temenos.t24.api.hook.system.ServiceLifecycle;
import com.temenos.t24.api.records.customer.CustomerRecord;
import com.temenos.t24.api.system.DataAccess;

/* Service DEMO */

public class MyServiceDemo extends ServiceLifecycle {
    @Override
    public void inputRecord(String id, ServiceData serviceData, String controlItem, TBoolean setZeroAuth,
            List<String> versionNames, List<String> recordIds, List<TStructure> records) {

        DataAccess da= new DataAccess(this);
        versionNames.add("CUSTOMER");
        recordIds.add(id);
        CustomerRecord cusRec = new CustomerRecord(da.getRecord("CUSTOMER", id));
        cusRec.setShortName("TEST "+cusRec.getMnemonic().toString(), 0);
        records.add(cusRec.toStructure());
        setZeroAuth.set(true);

    }
    @Override
    public List<String> getIds(ServiceData serviceData, List<String> controlList) {

        List<String> recIds = null;
        DataAccess da = new DataAccess(this);
        recIds=da.selectRecords("BNK", "CUSTOMER", "", "WITH MNEMONIC EQ KATRIN KMART KOS ");
          return recIds;

    }
```

Export JAR and restart jBoss

2. Create the EB.API records

```
public class MyServiceDemo extends ServiceLifecycle {
    @Override
    public void inputRecord(String id, ServiceData serviceData,
            List<String> versionNames, List<String> recordIds,

        DataAccess da= new DataAccess(this);
        versionNames.add("CUSTOMER");
        recordIds.add(id);
        CustomerRecord cusRec = new CustomerRecord(da.getRecord
        cusRec.setShortName("TEST "+cusRec.getMnemonic().toStri
        records.add(cusRec.toStructure());
        setZeroAuth.set(true);

    }
    @Override
    public List<String> getIds(ServiceData serviceData, List<St

        List<String> recIds = null;
        DataAccess da = new DataAccess(this);
        recIds=da.selectRecords("BNK", "CUSTOMER", "", "WITH MN
          return recIds;

    }

}
```

EB.API SERVICE.DEMO

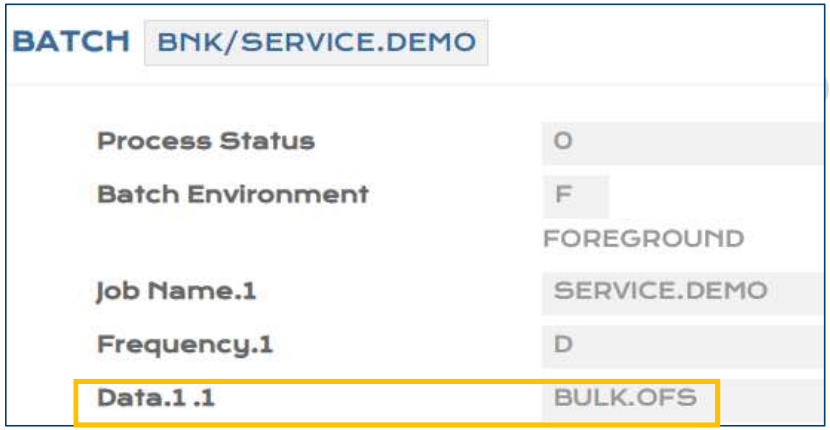
Description	EN SERVICE DEMO
Protection Level	Full
Source Type	Method
Java Method	inputRecord
Java Class	MyServiceDemo
Java Package	com.newbank

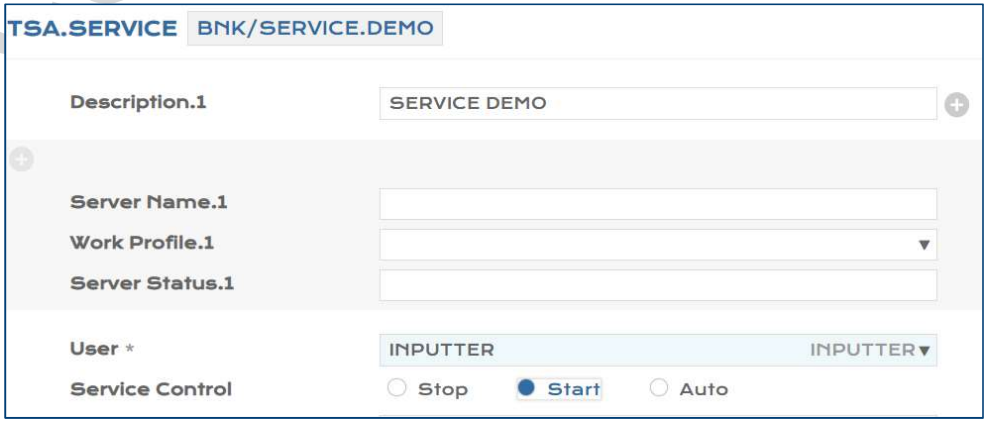
EB.API SERVICE.DEMO.SELECT

Description	EN SERVICE DEMO SELECT
Protection Level	Full
Source Type	Method
Java Method	getIds
Java Class	MyServiceDemo
Java Package	com.newbank

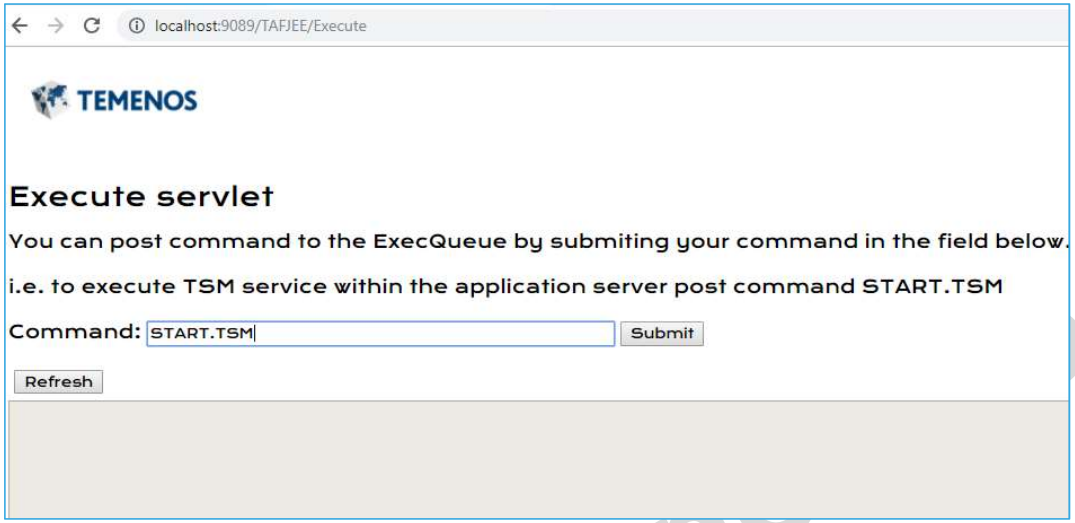
TEMENOS Learning Community

3. Create the PGM.FILE, BATCH & TSA.SERVICE records





4. Run the TSM and TSA



- When the service is run, we can see the SHORT.NAME is updated

TEMENOS Learning Community

Lesson 5. Local APPLICATION

T24 Extensibility in Java

TEMENOS
Learning Community

DEMO Creating Local Application

- We can create a new application using EB.TABLE.DEFINITION. OVERRIDE must be raised if the phone / email is used by existing customer

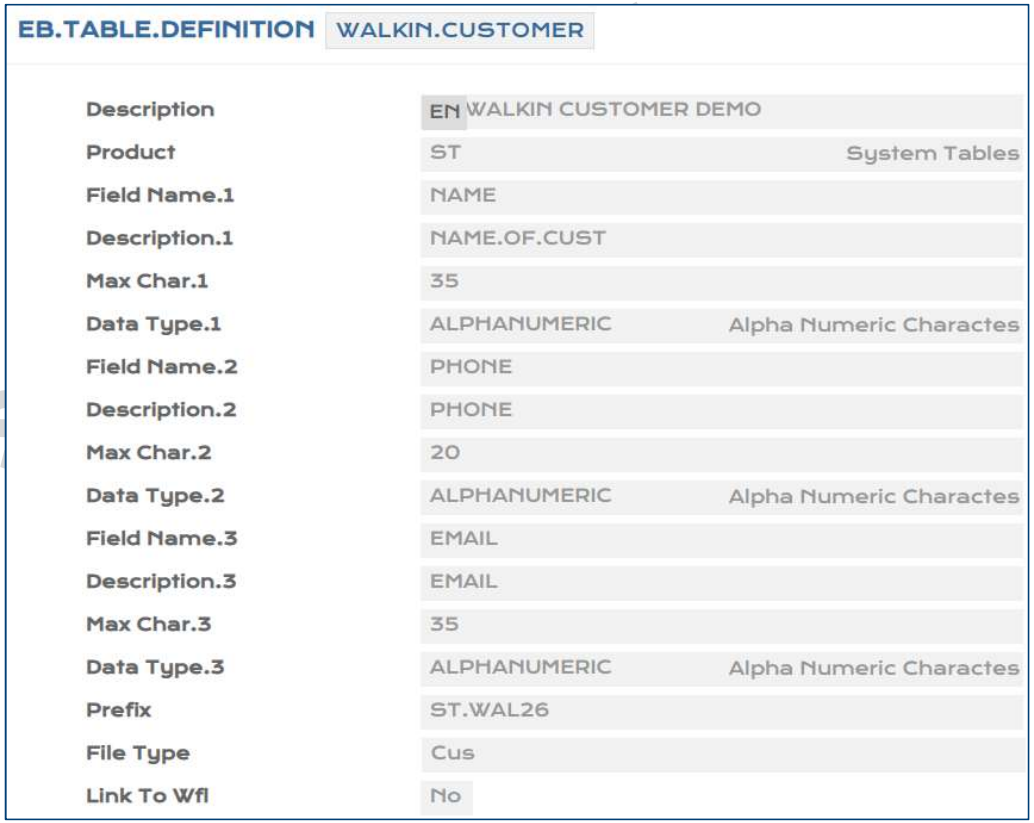
TABLE.NAME	WALKIN.CUSTOMER
PRODUCT	ST
FILE.TYPE	H
CLASSIFICATION	CUS

@ID, OVERRIDE, AUDIT fields are added automatically

Field Name	Data Type	Remarks
NAME	A	
PHONE		Raise override if this PHONE is available in CUSTOMER
EMAIL		Raise override if this EMAIL is available in CUSTOMER

1. Creating the TABLE – EB.TABLE.DEFINITION

- Create the table using EB.TABLE.DEFINITION
- ETD creates the required artefacts – PGM.FILE, FILE.CONTROL, STANDARD.SELECTION, DATA FILES

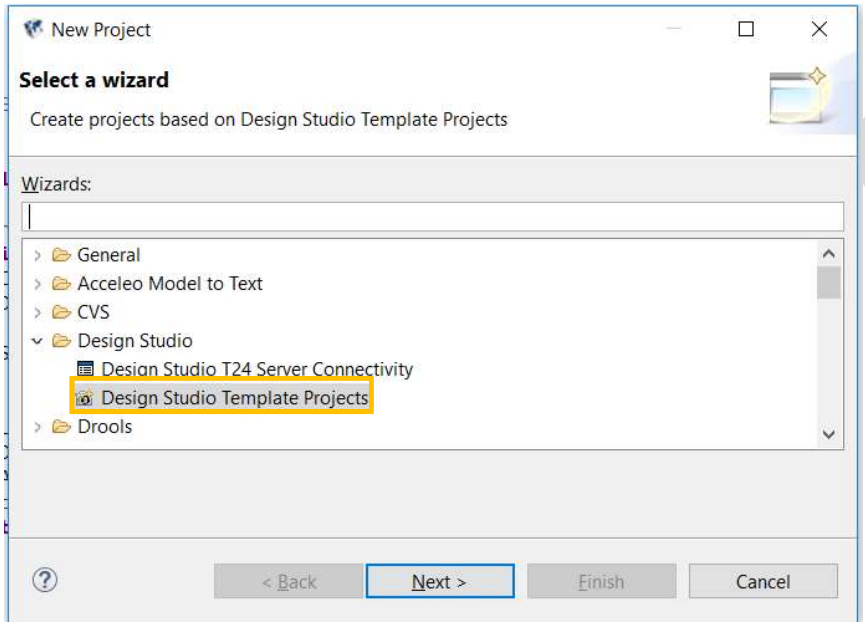


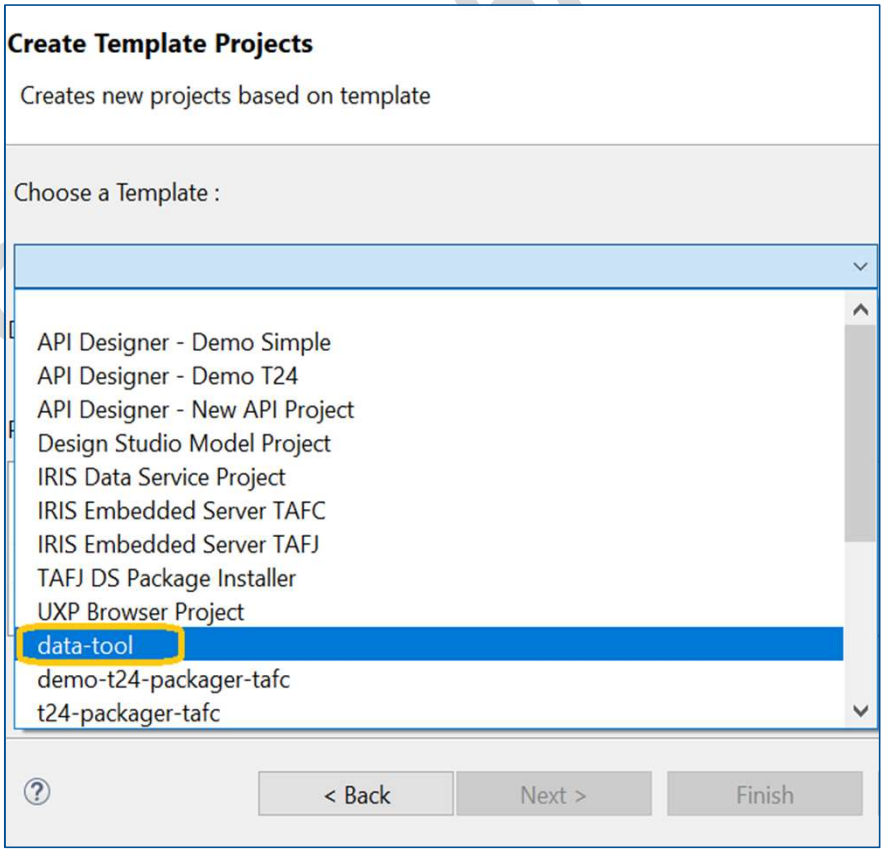
2. Generate the API for the new Table

- Using Design Studio, we can generate the API for the APPLICATION created using EB.TABLE.DEFINITION
- we need a server project – to establish connection with T24
- We need a models project – to import the application from T24

2a. Create the –DataTool project

- Creating the server project
- File→New→Project

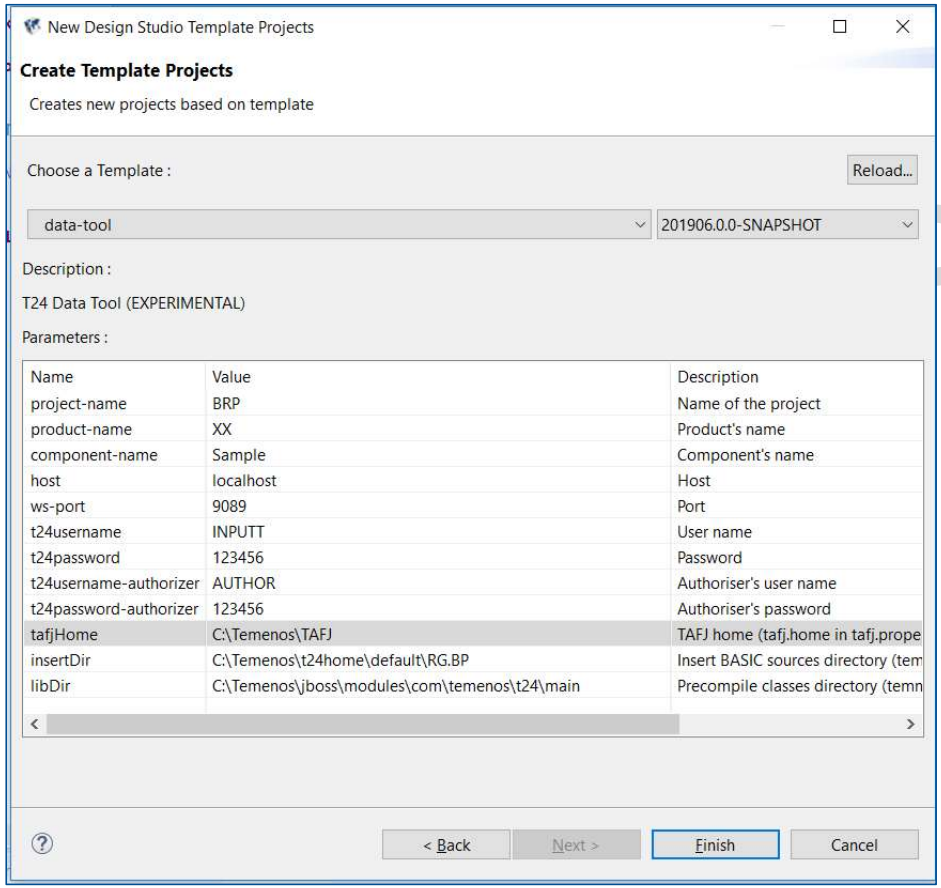


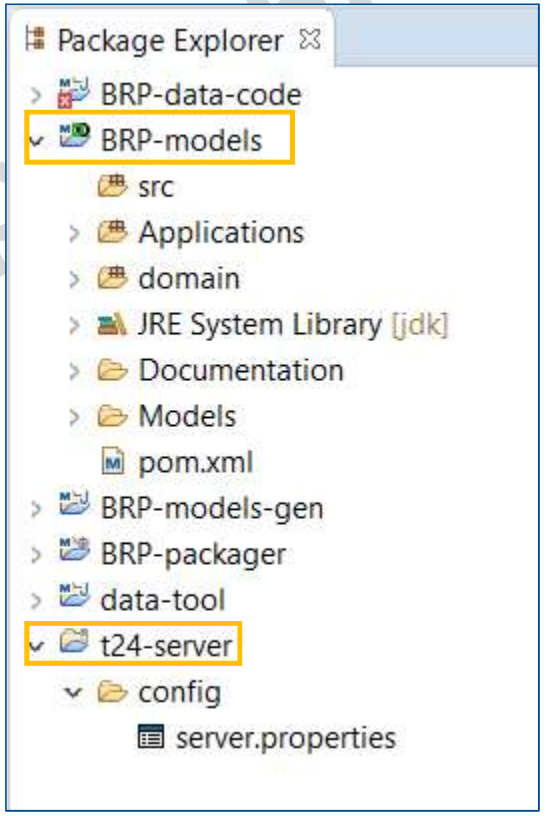


TEMENOS Learning Community

2b. Create the DS-DataTool Project

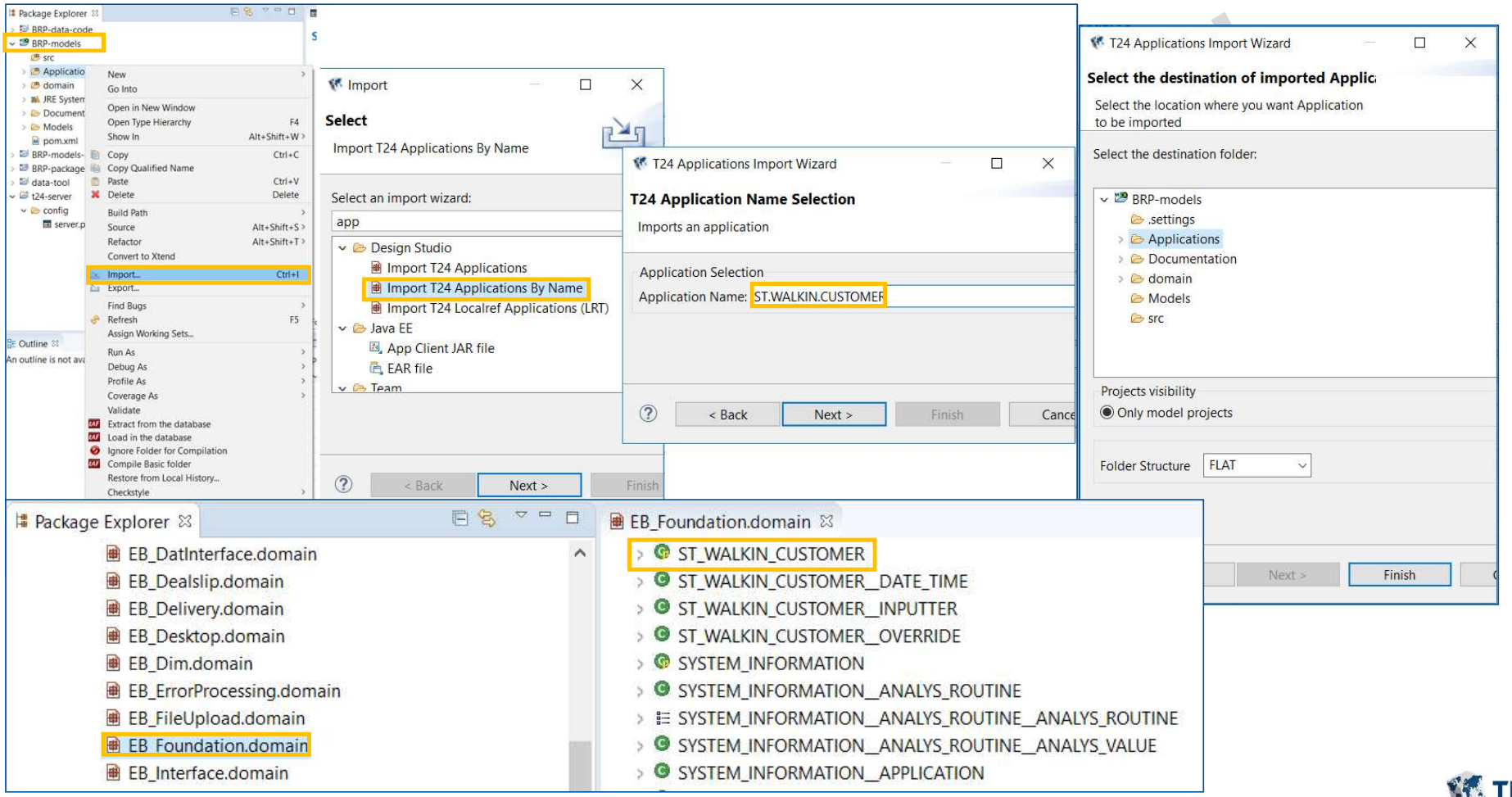
- Create the DS-DataTool project





TEMENOS Learning Community

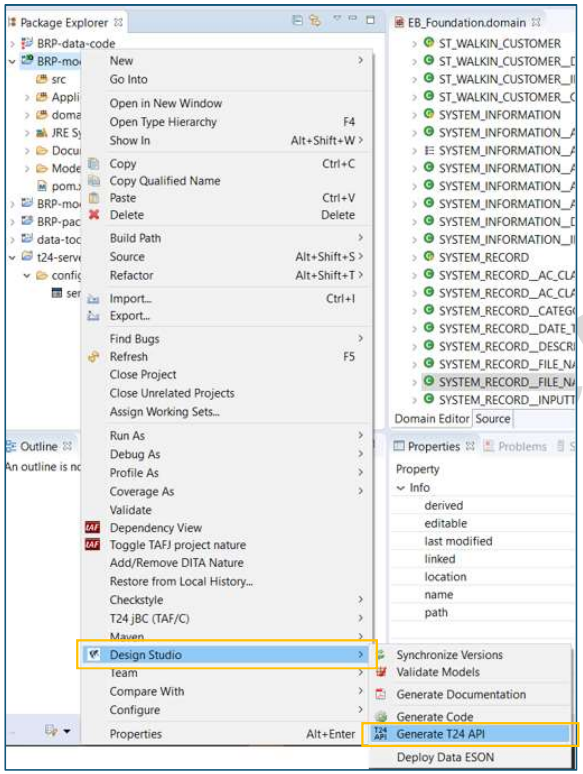
2c. Import the Local Application

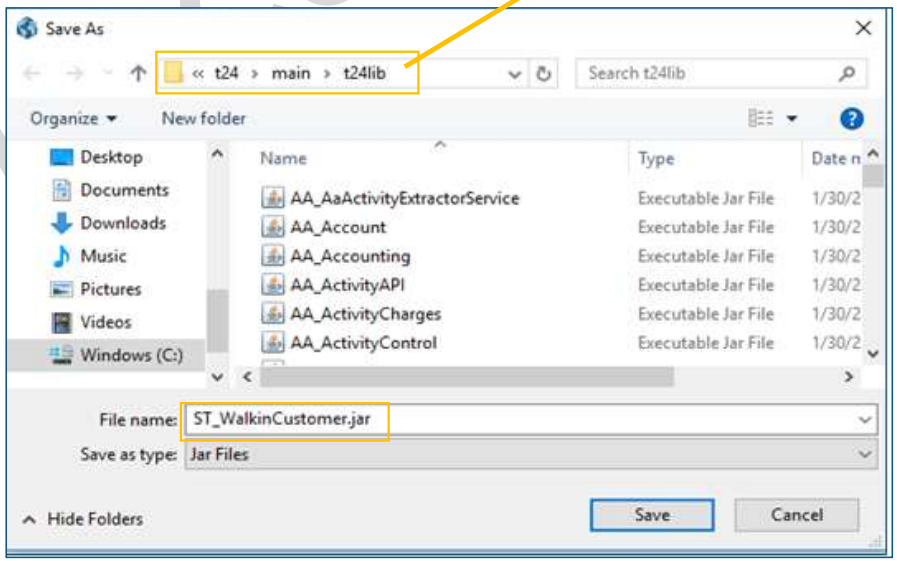


TEMENOS Learning Community

2d. Generate the API for the new Table

- Toggle the DS project as a TAFJ project
- Right click on project→Design Studio→Generate T24 API

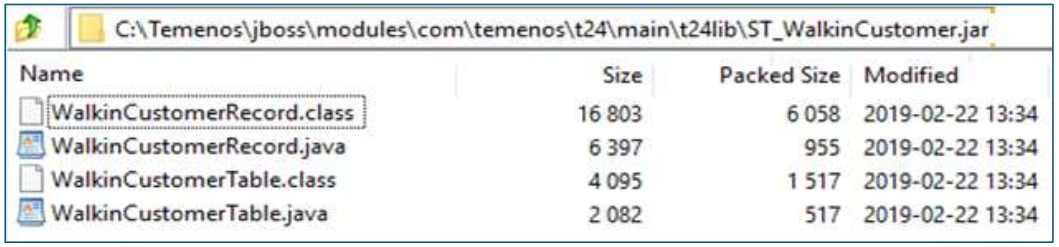




Give the location and name of the jar

TEMENOS Learning Community

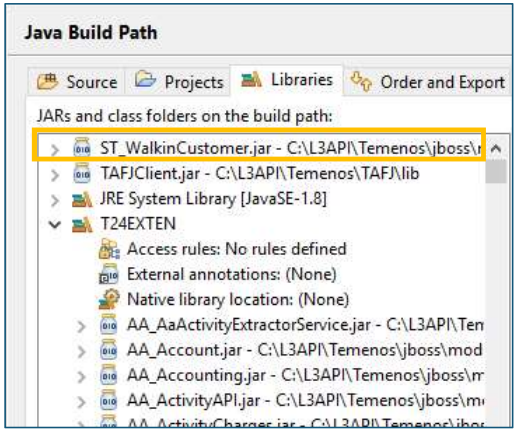
2e. Update module.xml and Build path



- Update module.xml and restart jBoss

```
<module xmlns="urn:jboss:module:1.0" name="com.temenos.t24">
  <resources>
    <!-- Insert resources here -->
    <resource-root path="./T24Exten.jar" />
    <resource-root path="./t24lib/ST_WalkinCustomer.jar" />
    <resource-root path="./t24lib/AA_ARAccountsData.jar" />
    <resource-root path="./t24lib/AA_ARC.jar" />
```

- Add jar to the java Build path in DS



TEMENOS Learning Community

3. Create the class

*WalkInCustomerOverride.java

```
package com.newbank;
import java.util.List;
import com.temenos.api.TBoolean;
import com.temenos.api.TString;
import com.temenos.api.TStructure;
import com.temenos.api.TValidationResponse;
import com.temenos.t24.api.hook.system.RecordLifecycle;
import com.temenos.t24.api.system.DataAccess;
import com.temenos.t24.api.tables.walkincustomer.WalkinCustomerRecord;
public TValidationResponse validateRecord(String application, String recordId, TStructure record,
        TStructure lastLiveRecord) {
    // validate phone and raise override
    DataAccess da = new DataAccess(this);
    String cusIds = "";
    List<String> recIds;
    int len;
    WalkinCustomerRecord WCR = new WalkinCustomerRecord(record);
    String phone = WCR.getPhone().toString();
    if (!phone.isEmpty()) {
        recIds = da.selectRecords("BNK", "CUSTOMER", "", "WITH PHONE.1 EQ '" + WCR.getPhone().toString() + "'");
        len = recIds.size();
        if (len > 0) {
            for (int i = 0; i < len; i++)
                cusIds = cusIds + " " + recIds.get(i);
            WCR.getPhone().setOverride("This phone number is used by CUSTOMER " + cusIds);
        }
    }
    // VALIDATE email and raise override
    String email = WCR.getEmail().toString();
    if (!email.isEmpty()) {
        recIds = da.selectRecords("BNK", "CUSTOMER", "", "WITH EMAIL.1 EQ '" + email + "'");
        len = recIds.size();
        cusIds = "";

        if (len > 0) {
            for (int i = 0; i < len; i++)
                cusIds = cusIds + " " + recIds.get(i);
            WCR.getEmail().setOverride("This email is used by CUSTOMER " + cusIds);

        }
    }
    return WCR.getValidationResponse();
}
```

4. Create the EB.API, VERSION

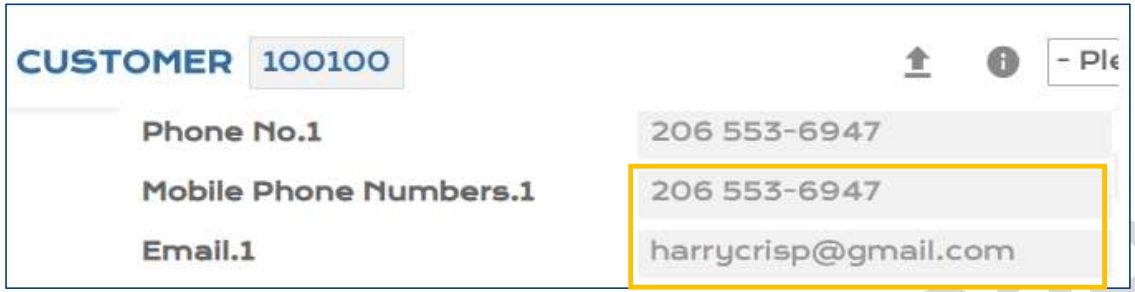
EB.API WALKIN.CUSTOMER.OVERRIDE

Description	EN WALKIN CUST
Protection Level	Full
Source Type	Method
Java Method	validateRecord
Java Class	WalkinCustomerOverride
Java Package	com.newbank

Version ST.WALKIN.CUSTOMER,DEMO

Records Per Page	1
Fields Per Line	Multi
Language Code.1	1
Field No.1	NAME
Text Char Max.1	35
Pre Syndication ID.1 .1	Name
Enrichm Char.1	35
Table Line.1	0
Field No.2	EMAIL
Text Char Max.2	35
Pre Syndication ID.2 .1	Email
Enrichm Char.2	35
Table Line.2	1
Field No.3	PHONE
Text Char Max.3	10
Pre Syndication ID.3 .1	Phone
Enrichm Char.3	10
Table Line.3	2
Input Routine.1	WALKIN.CUSTOMER.OVERRIDE

5. Launch the VERSION



TEMENOS Learning Community

Using EB.TABLE.PROCEDURES

EB.API * WALKI

Key	Description	Protection Level	Source Type	Java Method
WALKIN.CUSTOMER.DEFAULT	WALKIN CUST	FULL	METHOD	defaultFieldValues
WALKIN.CUSTOMER.OVERRIDE	WALKIN CUST	FULL	METHOD	validateRecord

EB.TABLE.PROCEDURES ST.WALKIN.CUSTOMER

Description *	EN WALKIN CUSTOMER
Initialise Proc.1	
Check Id Proc.1	
Check Rec Proc.1	WALKIN.CUSTOMER.DEFAULT WALKIN CUST
Prs Fld Proc.1	
Check Fld Proc.1	
Crossval Proc.1	WALKIN.CUSTOMER.OVERRIDE WALKIN CUST

PGM.FILE WALKIN.CUSTOMER.DEFAULT

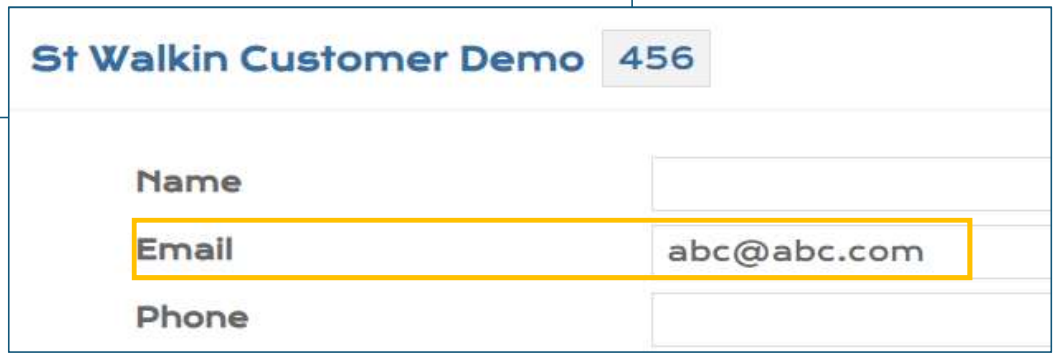
Type	S
Product	EB

PGM.FILE WALKIN.CUSTOMER.OVERRIDES

Type	S
Product	EB

Using EB.TABLE.PROCEDURES

```
public class WalkInCustomerOverride extends RecordLifecycle {

    public String checkId(String arg0) {

    @Override
    public void defaultFieldValues(String application, String recordId, TStruct
        WalkinCustomerRecord WCR = new WalkinCustomerRecord(record);
        if (WCR.getEmail().toString().isEmpty()) {
            WCR.setEmail("abc@abc.com");
            record.set(WCR.toStructure());
        }

    }
```



TEMENOS Learning Community

Local Table

Lesson - 6

TEMENOS
Learning Community

DEMO Creating Local Table

- Create a local field called CUS.HOBBIES
- Attach field to CUSTOMER
- The local field gets its values from a LOCAL APPLICATION
- Use version to input hobbies for authorized CUSTOMER
- Minimum of 3 hobbies must be input

1. Creating the Local Application

- Create local application CUS.HOBBIES and input few records

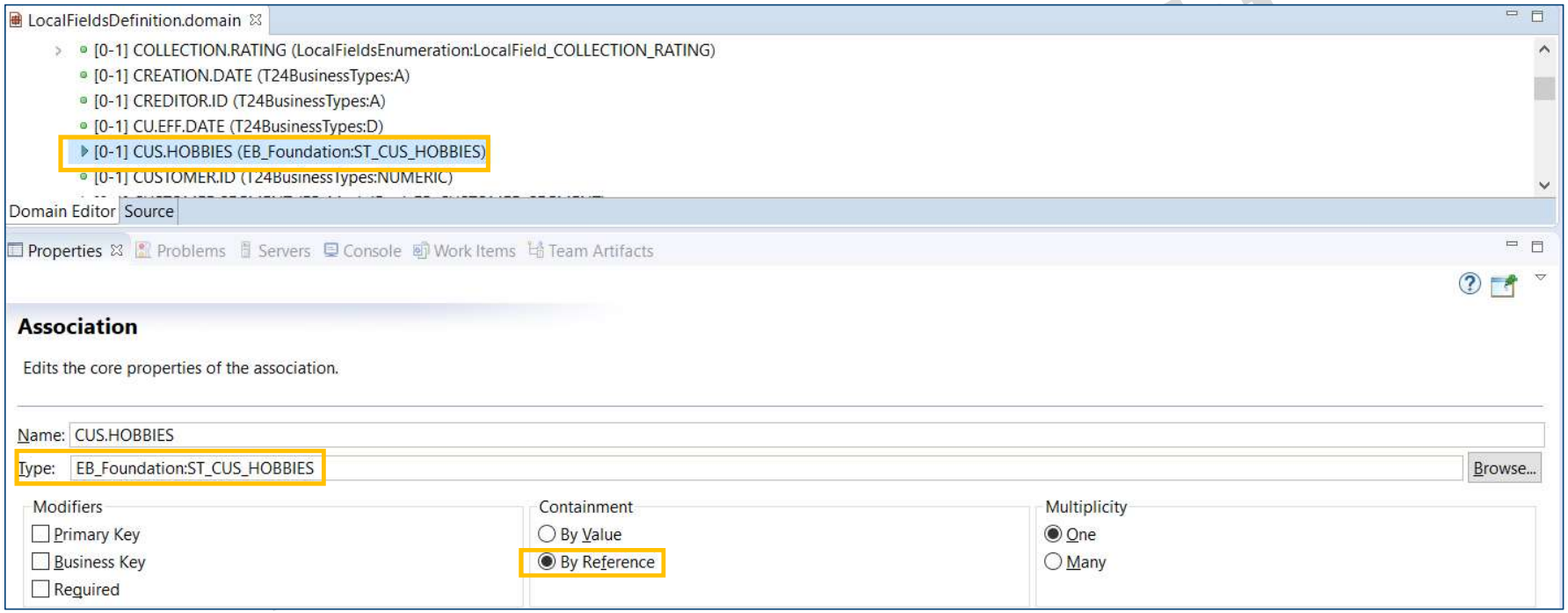
EB.TABLE.DEFINITION CUS.HOBBIES

Description	EN CUSTOMER HOBBIES
Product	ST
Field Name.1	DESCRIPTION
Description.1	HOBBY
Max Char.1	30
Data Type.1	ALPHANUMERIC
Prefix	ST.CUS86
File Type	Cus

List of Hobbies

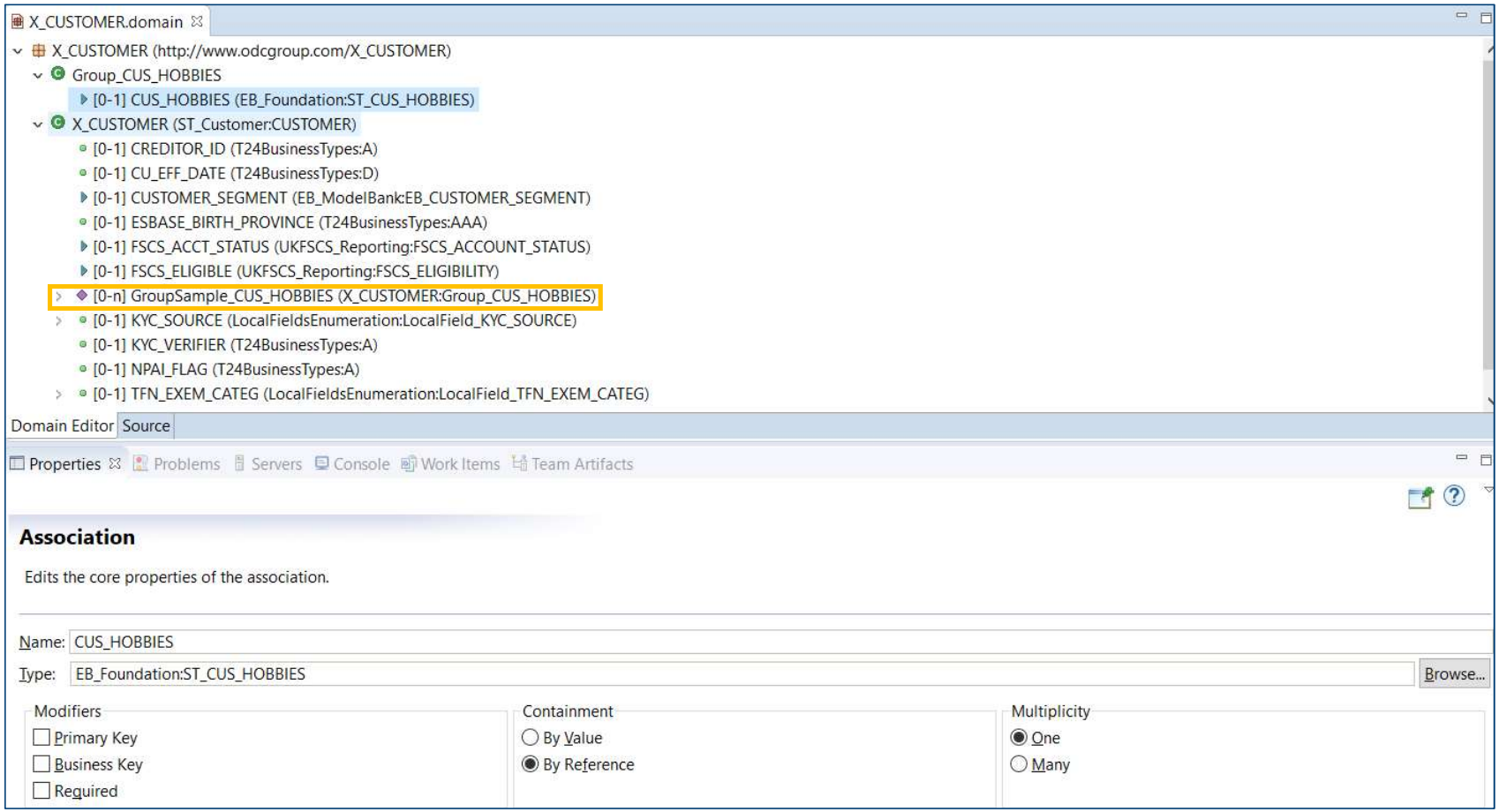
Deal Number	Description
DANCING	Dancing
GARDENING	Gardening
MOVIES	Watching Movies
ORIGAMI	Origami
READING	Reading Books

2. Create LOCAL.TABLE called CUS.HOBBIES



TEMENOS Learning Community

3. Add the LOCAL.TABLE to CUSTOMER



TEMENOS Learning Community

4. Create the class

```
package com.newbank;

import java.util.List;
import com.temenos.api.LocalRefClass;
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.customer.CustomerRecord;
import com.temenos.t24.api.system.DataAccess;

/**  VERSION CUSTOMER,HOBBY is used to add a minimum of 3 hobbies for authorised customers  */
public class CustomerHobbies extends RecordLifecycle {

    @Override
    public TValidationResponse validateRecord(String application, String recordId, TStructure record,
            TStructure lastLiveRecord) {
        CustomerRecord cusRec = new CustomerRecord(record);
        LocalRefClass lrf = cusRec.getLocalRef("CUS.HOBBIES");
        List<TField> hobbies = lrf.get();
     TField dummy = new TField("");
        if (hobbies.size() <= 2) {
            lrf.set(dummy, 0); // to overcome array index out of bounds when no hobby is entered
            lrf.get(0).setError("Please select minimum of 3 hobbies");
        }
        return cusRec.getValidationResponse();
    }
    @Override
    public String checkId(String idNew) {
        DataAccess da = new DataAccess(this);
        CustomerRecord cusRec = null;
        try {
            cusRec = new CustomerRecord(da.getRecord("CUSTOMER", idNew));
        } catch (Exception e) {

                throw new Error("This VERSION can be used to edit Authorised customers only or Invalid Customer ID");
            }
        return idNew;
    }

}
```

Create class , Export JAR and restart jBoss

TEMENOS Learning Community

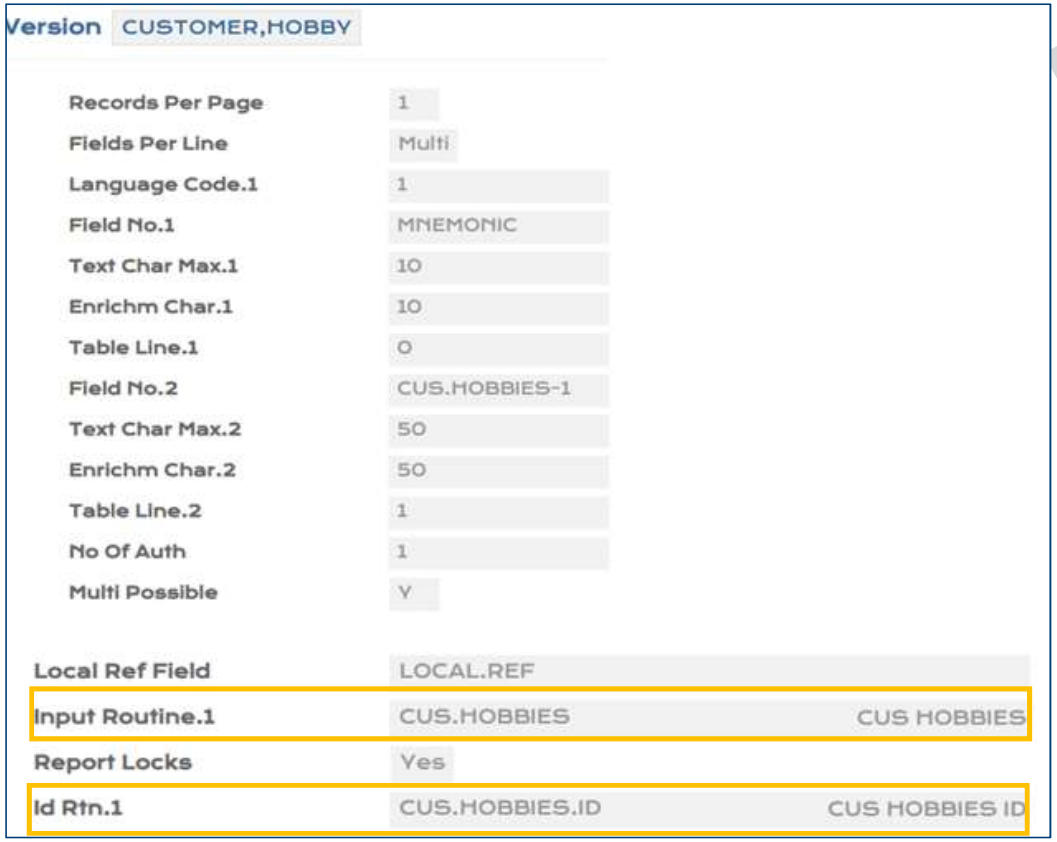
5. Create the EB.API records

EB.API * CUS.H

Key	Description	Protection Level	Source Type	Java Method
CUS.HOBBIES	CUS HOBBIES	FULL	METHOD	validateRecord
CUS.HOBBIES.ID	CUS HOBBIES ID	FULL	METHOD	checkId

6. Create/Modify version

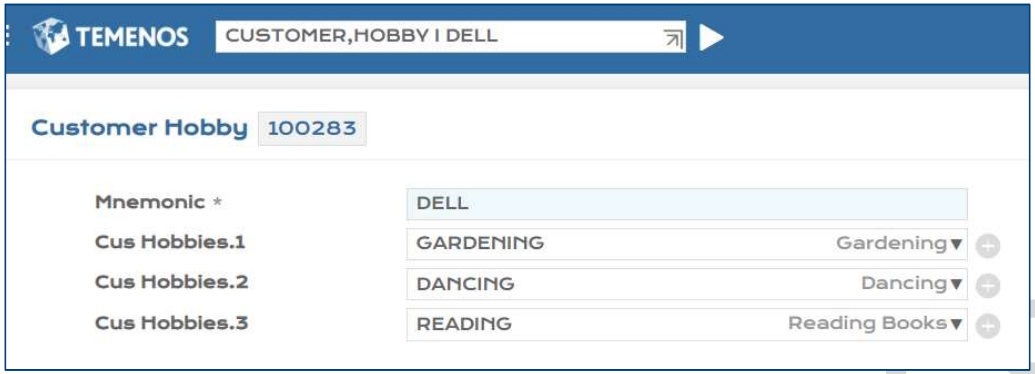
- Create the version CUSTOMER,HOBBY in DS and generate code. Modify the VERSION and add the hooks

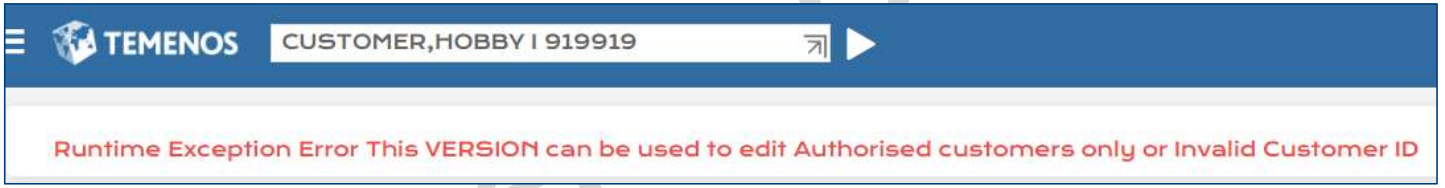


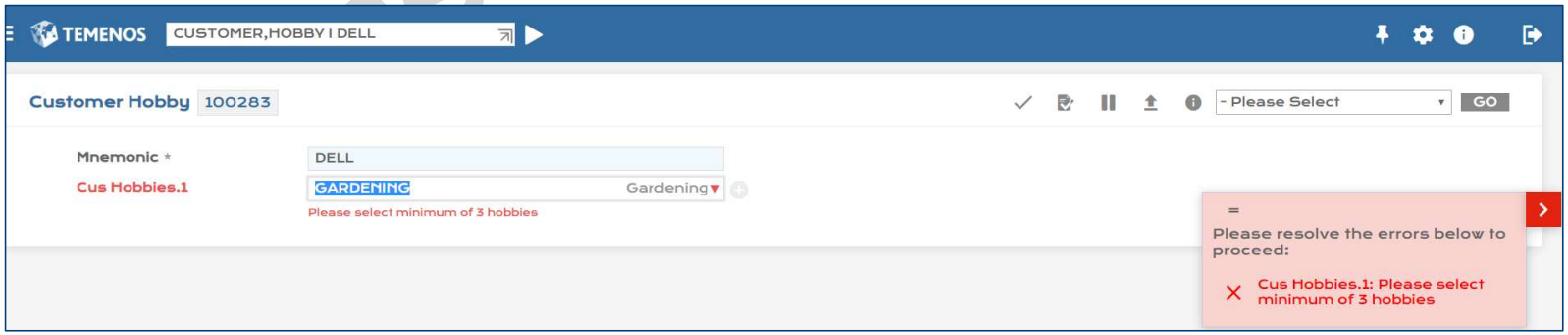
Version CUSTOMER,HOBBY

Records Per Page	1	
Fields Per Line	Multi	
Language Code.1	1	
Field No.1	MNEMONIC	
Text Char Max.1	10	
Enrichm Char.1	10	
Table Line.1	0	
Field No.2	CUS.HOBBIES-1	
Text Char Max.2	50	
Enrichm Char.2	50	
Table Line.2	1	
No Of Auth	1	
Multi Possible	Y	
Local Ref Field	LOCAL.REF	
Input Routine.1	CUS.HOBBIES	CUS HOBBIES
Report Locks	Yes	
Id Rtn.1	CUS.HOBBIES.ID	CUS HOBBIES ID

Launching the VERSION







TEMENOS Learning Community

Course Summary

- We have seen
  - Need for T24 Extensibility
  - T types
  - How to attach VERSION hooks
  - Attaching Enquiry hooks
  - Creating and executing a multi threaded SERVICE
  - Creating a local table using EB.TABLE.DEFINITION and customizing the application
  - How to debug the Java code

TEMENOS

Learning Community

thank.you

tlc.temenos.com