



PUBLIC

SAP BusinessObjects Business Intelligence platform

Document Version: 4.3 Support Package 4 – 2023-12-07

SAP Business Intelligence Platform Translation Management Tool SDK Developer Guide

Content

1	Document History.	4
2	About this Guide.	5
2.1	Who Should Use this Guide.	5
2.2	What this Guide Contains.	5
3	Technical Requirements.	6
4	Introduction TMT Developer Guide.	7
4.1	System Diagram.	7
4.2	Translation Process.	8
5	TMT SDK Development Workflow.	9
6	Developing TMT SDK.	10
6.1	Setting up the TMT SDK Environment.	10
6.2	Creating TMT SDK Workflows.	15
	Implementation Code for Importing InfoObjects from CMS.	15
	Implementation Code for Importing InfoObjects from Local Folder.	15
	Implementation Code for Extracting Translations From the InfoObject.	16
	Implementation Code for Getting the Translation Manager Engine.	17
	Implementation Code for Getting and Setting Available locale, Visible and Fallback Locale.	17
	Implementation Code for Getting and Setting the Property Values and Property Status of Entities.	19
	Implementation Code for Saving Translation Manager Document (.tmgr file) to the Local Folder.	21
	Implementation Code for Exporting Translation Manager Document (.tmgr file) to the CMS	21
	Implementation Code for Exporting Translation Manager Document (.tmgr file) to XLIFF file	22
	Implementation Code for Importing Translation Manager Document from XLIFF file.	22
	Implementation Code for Exporting Translation Manager Document to XLS file.	22
	Implementation Code for Importing Translation Manager Document from XLS file.	23
	Sample Implementation Code.	23
7	Testing TMT SDK Workflows.	25
8	Using TMT SDK.	26
9	Limitations.	27

10	Troubleshooting Tips.	28
11	Frequently Asked Questions.	29

1 Document History

The following table provides an overview of the most important document changes.

Version	Date	Description
SAP BusinessObjects Business Intelligence platform 4.3	June, 2020	Initial release

2 About this Guide

This guide gives you information on how to develop an application using Translation Management Tool (TMT) SDK. Earlier, translations that were saved in the Translation Management Tool, were not visible or editable through semantic layer SDK. Now, we are sharing functionalities that help you build your own application, which helps you to reintegrate modifications without reentering the same in Translation Management Tool. We also provide a new feature where in, you can analyze universe and extract a list of translations into an Excel format. So, using TMT SDK you can process multiple InfoObjects and perform translations.

2.1 Who Should Use this Guide

This guide is intended for developers, who can create their own application to perform translations. You can use both semantic layer SDK and translation manager SDK to programmatically update the translations of labels.

For Example: If customers or partners wants to automate their translation process, they can automate either by having translation APIs, or creating APIs that automate import and export of `XLIFF` files.

2.2 What this Guide Contains

This guide contains the following information:

- Introduction to Translation Management Tool (TMT) SDK
- System diagrams and development workflows in TMT SDK
- Creating, testing, and deploying TMT SDK
- How to read InfoObjects (.blx, .dfx, .unv) from local folders
- How to read InfoObjects from Central Management Server (CMS) repository
- Modifying InfoObjects Property Values and Property Status
- How to translate imported InfoObjects from local folders or CMS repository
- Saving changes in local folders
- Committing changes to the CMS repository
- Testing TMT SDK
- Exporting Translation Manager Document (.tmgr) to XLIFF file and reading .tmgr from XLIFF file
- Exporting Translation Manager Document (.tmgr) to XLS file and reading .tmgr from XLS file
- Limitations
- Troubleshooting Tips
- Frequently Asked Questions

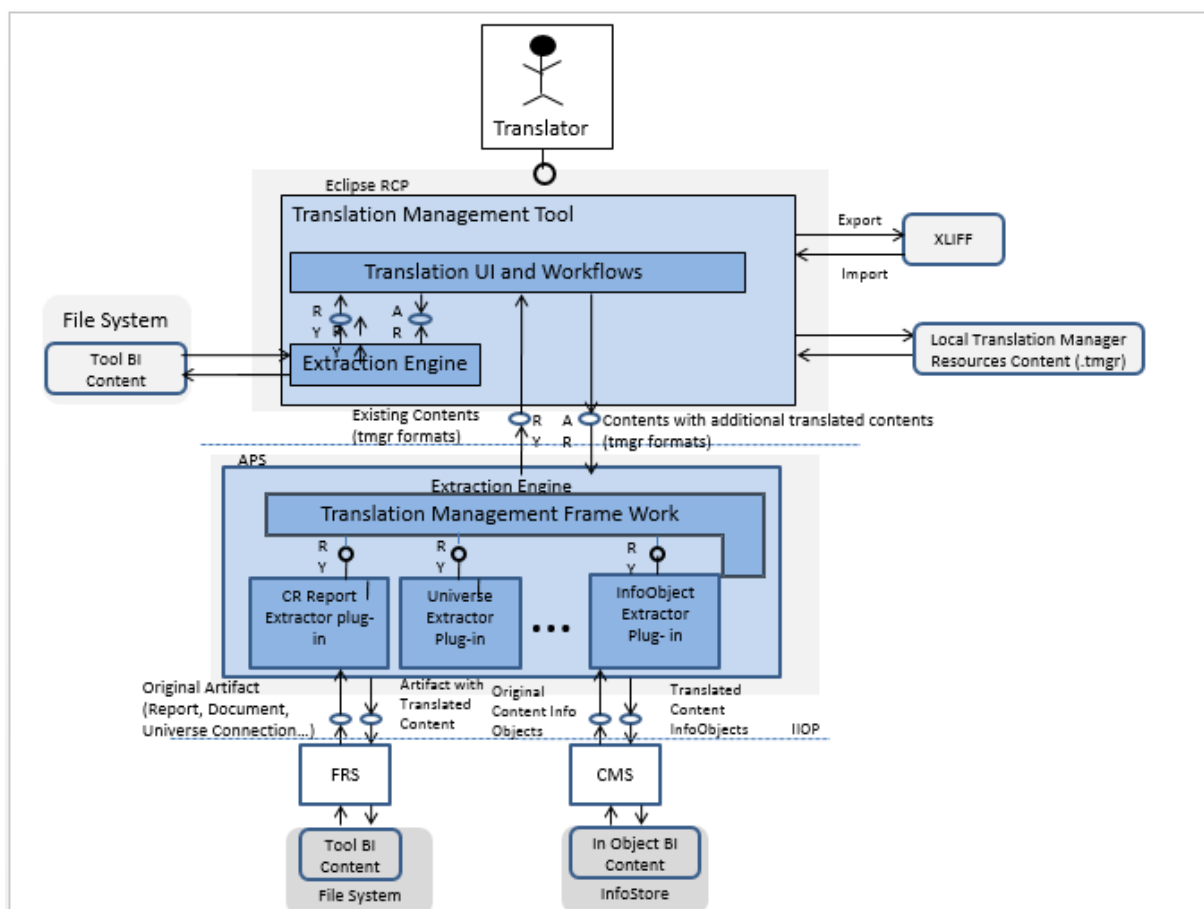
3 Technical Requirements

You must be familiar with the Translation Management Tool and the following basic workflows:

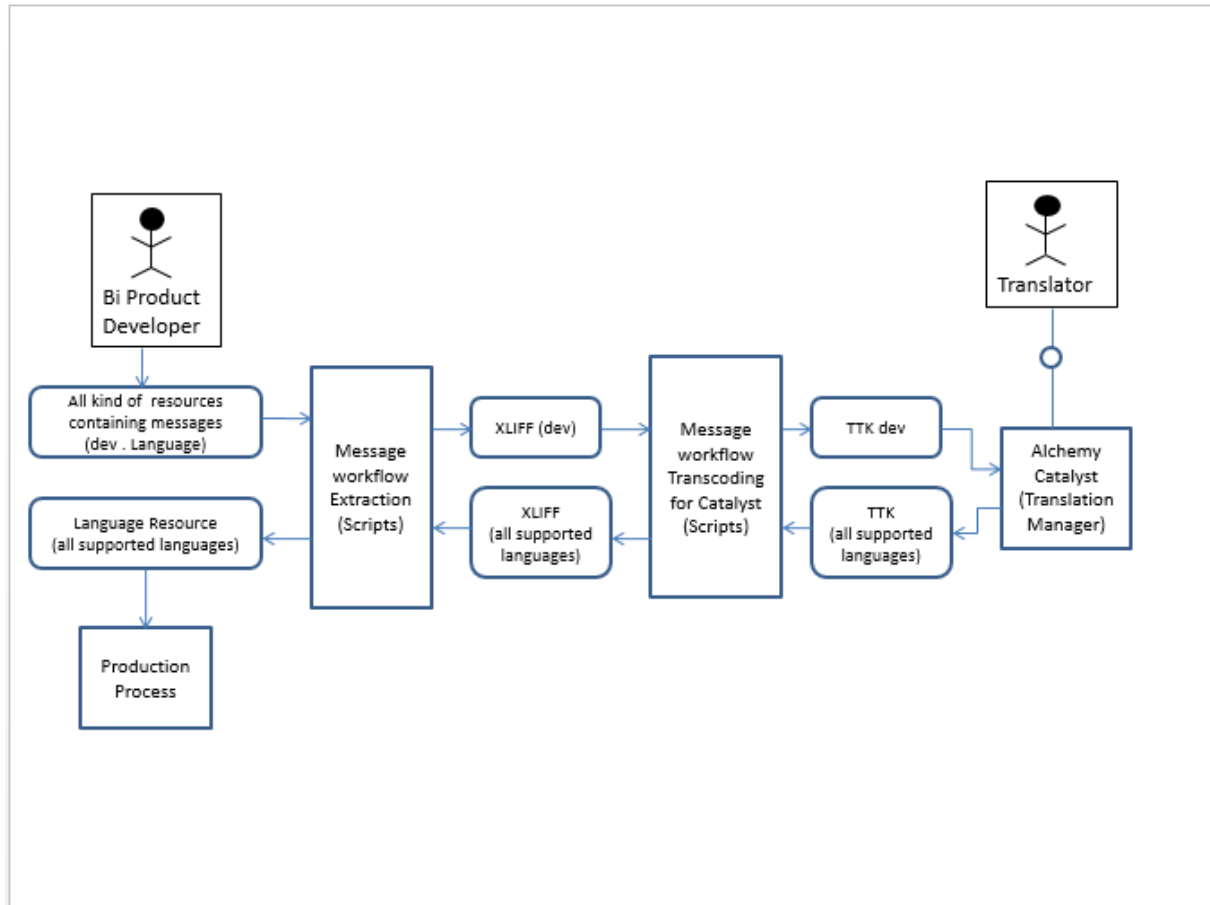
- Importing InfoObjects
- Translating InfoObjects
- Getting the Translation Manager Engine
- Saving InfoObjects as Translation Manager document (.tmgr) to local folders
- Committing Translation Manager Document (.tmgr) to the CMS
- Exporting Translation Manager Document (.tmgr) to XLIFF files

4 Introduction TMT Developer Guide

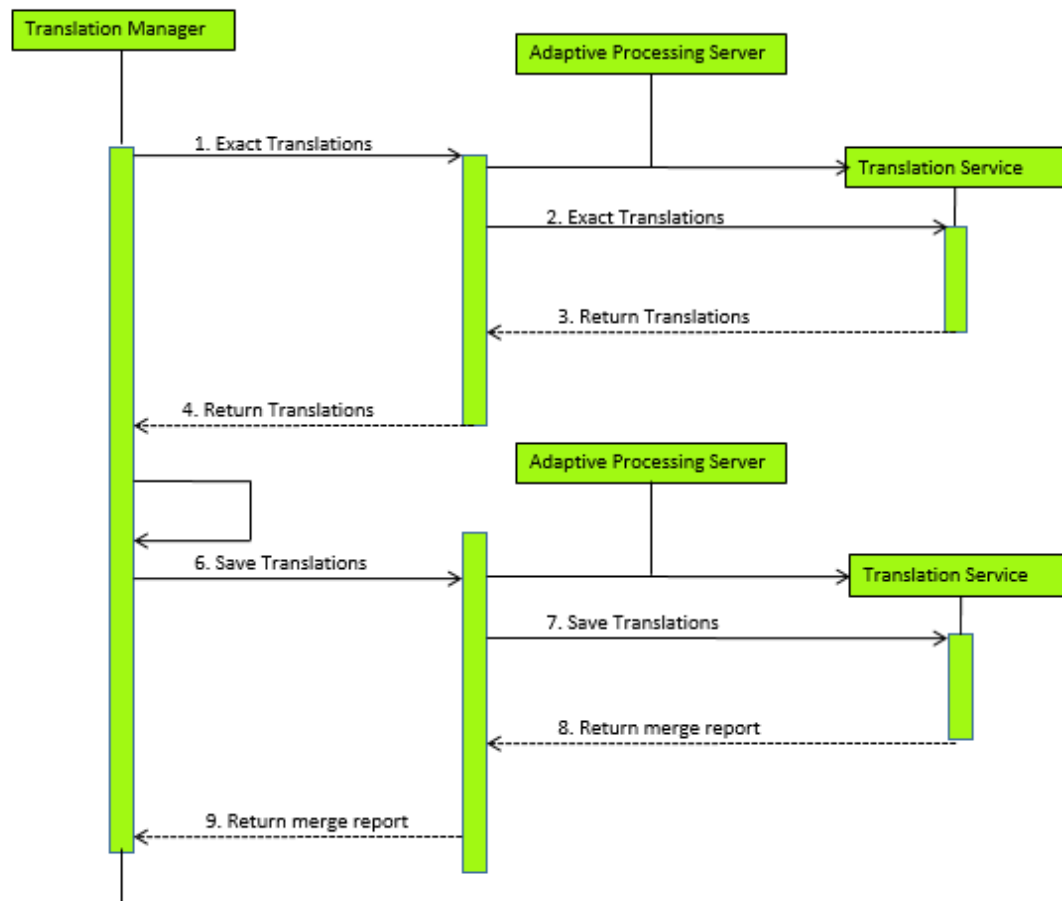
4.1 System Diagram



4.2 Translation Process



5 TMT SDK Development Workflow



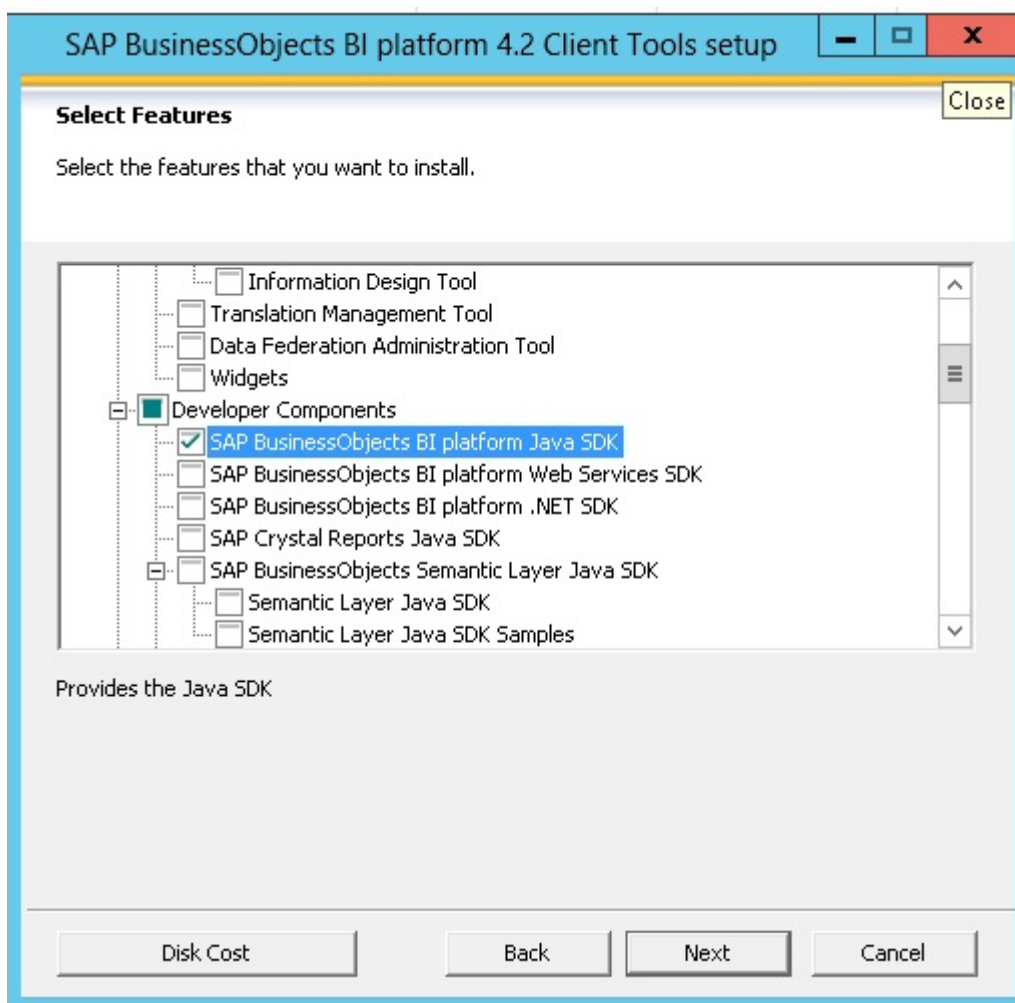
6 Developing TMT SDK

6.1 Setting up the TMT SDK Environment

There are two ways to set up the TMT SDK environment:

Method1:

1. While installing the BOE client, select SAP BusinessObjects BI Platform Java SDK. See the below screenshot for your reference.



2. After installing the BOE client, you can see the Translation Manager SDK JARS in the location:
`%BOINSTALLDIR%\java\lib\TranslationManagerSDK`

The following are the list of JAR files that must be present in %BOINSTALLDIR%\java\lib\TranslationManagerSDK location:

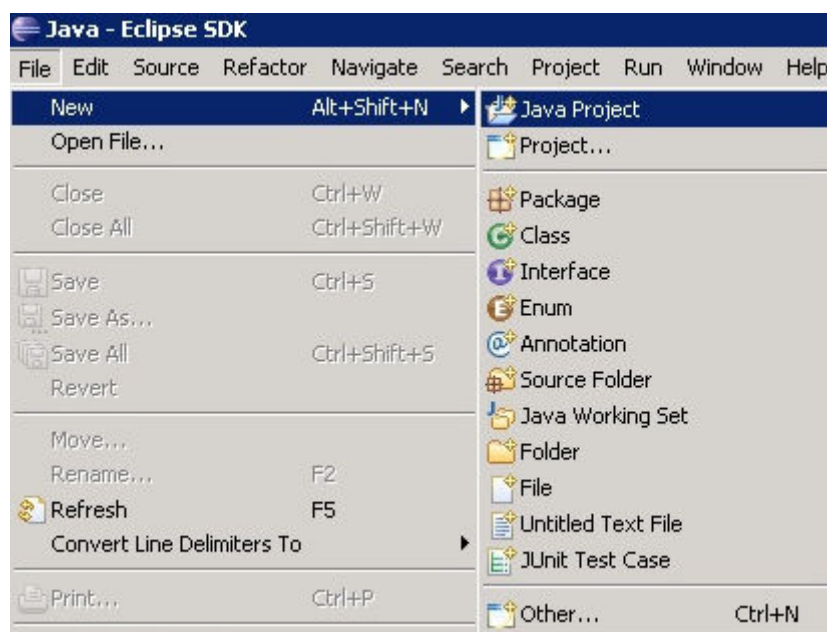
- aspectjrt.jar
- axiom-api-1.2.13.jar
- axiom-impl-1.2.13.jar
- axis2-adb-1.6.2.jar
- axis2-kernel-1.6.2.jar
- bcm.jar
- biplugins.jar
- ceaspect.jar
- cecore.jar
- celib.jar
- cesession.jar
- com.businessobjects.dsl.common.jar
- com.businessobjects.mds.connection.jar
- com.businessobjects.mds.datafoundation.jar
- com.businessobjects.mds.datafoundationsecurityaccess.jar
- com.businessobjects.mds.entity.jar
- com.businessobjects.mds.parameter.jar
- com.businessobjects.mds.repository.jar
- com.businessobjects.mds.resource.jar
- com.businessobjects.mds.toolkit.jar
- com.businessobjects.mds.universe.jar
- com.businessobjects.transmgr.unv.cms.jar
- com.businessobjects.transmgr.unv.jar
- com.businessobjects.transmgr.webi.jar
- com.businessobjects.xi.sdk.jar
- com.sap.translation.cms.client.core.jar
- com.sap.translation.core.jar
- com.sap.translation.sdk.jar
- com.sap.translation.unx.cms.jar
- com.sap.translation.unx.jar
- com.sap.translation.unx.ui.jar
- commons-logging.jar
- corbaidl.jar
- cryptojFIPS.jar
- ebus405.jar
- il8n4j.jar
- logging.jar
- neethi-3.0.2.jar
- org.eclipse.emf.common_2.4.0.v200902171115.jar
- org.eclipse.emf.ecore_2.4.2.v200902171115.jar

- org.xmlpull_1.1.3.4.jar
- poi-3.9-20121203.jar
- SL_plugins.jar
- TraceLog.jar
- translation-service-client.jar
- wicdzcorelib.jar
- wsdl4j-1.6.2.jar
- XmlSchema-1.4.7.jar
- com.sap.translation.sdk.jar

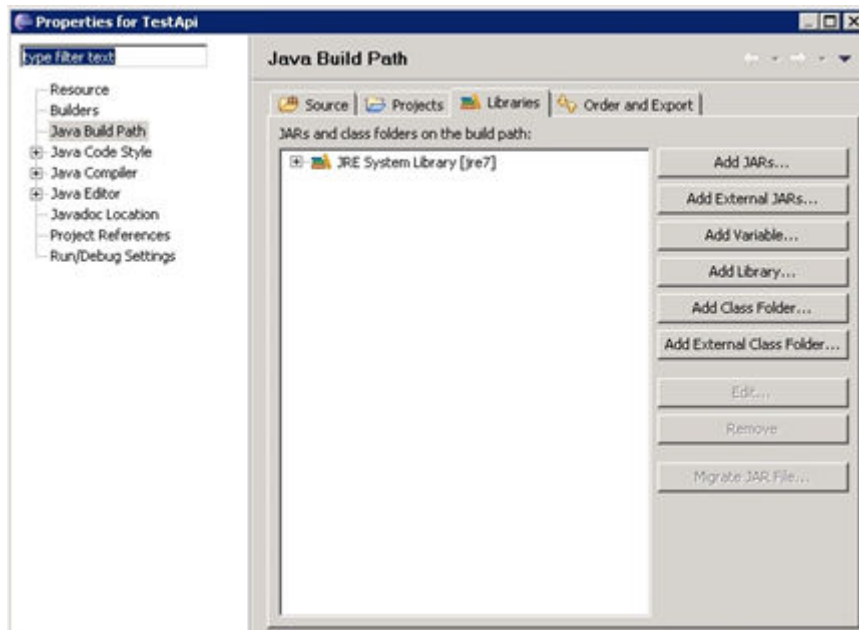
Note

- The default directory for %BOINSTALLDIR% is C:\Program Files (x86)\SAP BusinessObjects.

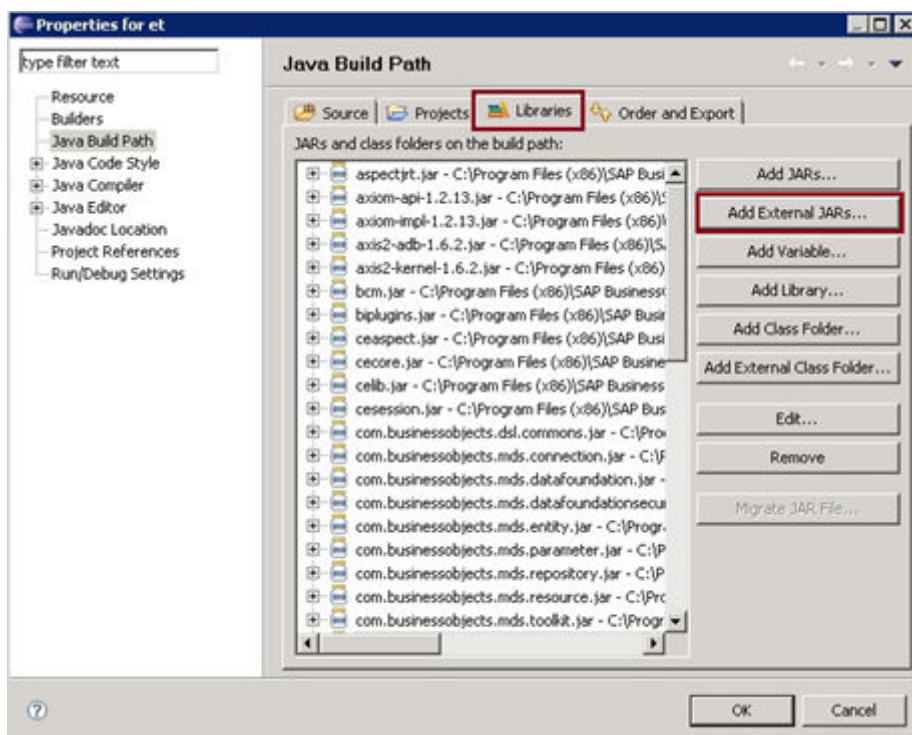
3. Launch Eclipse (The SDK is tested using Eclipse SDK 3.4.2 version).
4. Choose **File** > > **New** > > **Java Project** to create a new Java project.



5. Add the required JAR files in the Java Build Path (shown in the figure).



6. To add the JAR files, choose **Libraries**, then choose **Add External JARs...**, and select the required JARS from %BOINSTALLDIR%\java\lib\TranslationManagerSDK (C:\Program Files (x86)\SAP BusinessObjects\java\lib\TranslationManagerSDK) path. Following screen shot helps you to add libraries in the build path.



7. Click **OK**.
8. You have set the environment successfully. You can now use the available APIs to build your own application. For more information on TMT SDK, see *SAP Business Intelligent Platform Translation Manager SDK Javadoc* for your reference.

Method 2:

1. Step 1:

Create a new batch file.

2. Step 2:

Add the following details and save file as a batch file (.bat):

Note

This is available only for windows.

- set LIBHOME=%BOINSTALLDIR%\java\lib\TranslationManagerSDK

Note

By default "C:\Program Files (x86)\SAP BusinessObjects" is the "%BOINSTALLDIR%".

- set CLASSPATH=%LIBHOME%\aspectjrt.jar;%LIBHOME%\axiom-api-1.2.13.jar;%LIBHOME%\axiom-impl-1.2.13.jar;%LIBHOME%\axis2-adb-1.6.2.jar;%LIBHOME%\axis2-kernel-1.6.2.jar;bcm.jar;%LIBHOME%\biplugins.jar;%LIBHOME%\ceaspect.jar;%LIBHOME%\cecore.jar;%LIBHOME%\celib.jar;%LIBHOME%\cesession.jar;%LIBHOME%\com.businessobjects.dsl.common.jar;%LIBHOME%\com.businessobjects.mds.connection.jar;%LIBHOME%\com.businessobjects.mds.datafoundation.jar;%LIBHOME%\com.businessobjects.mds.datafoundationsecurityaccess.jar;%LIBHOME%\com.businessobjects.mds.entity.jar;%LIBHOME%\com.businessobjects.mds.parameter.jar;%LIBHOME%\com.businessobjects.mds.repository.jar;%LIBHOME%\com.businessobjects.mds.resource.jar;%LIBHOME%\com.businessobjects.mds.toolkit.jar;%LIBHOME%\com.businessobjects.mds.universe.jar;%LIBHOME%\com.businessobjects.transmgr.unv.cms.jar;%LIBHOME%\com.businessobjects.transmgr.unv.jar;%LIBHOME%\com.businessobjects.transmgr.webi.jar;%LIBHOME%\com.businessobjects.xi.sdk.jar;%LIBHOME%\com.sap.translation.cms.client.core.jar;%LIBHOME%\com.sap.translation.core.jar;%LIBHOME%\com.sap.translation.sdk.jar;%LIBHOME%\com.sap.translation.unx.cms.jar;%LIBHOME%\com.sap.translation.unx.jar;%LIBHOME%\com.sap.translation.unx.ui.jar;%LIBHOME%\commons-logging.jar;%LIBHOME%\corbaidl.jar;%LIBHOME%\cryptojFIPS.jar;%LIBHOME%\ebus405.jar;%LIBHOME%\i18n4j.jar;%LIBHOME%\logging.jar;%LIBHOME%\neethi-3.0.2.jar;%LIBHOME%\org.eclipse.emf.common_2.4.0.v200902171115.jar;%LIBHOME%\org.eclipse.emf.ecore_2.4.2.v200902171115.jar;%LIBHOME%\org.xmlpull_1.1.3.4.jar;%LIBHOME%\poi-3.9-20121203.jar;%LIBHOME%\SL_plugins.jar;%LIBHOME%\TraceLog.jar;%LIBHOME%\translation-service-client.jar;%LIBHOME%\wiczcorelib.jar;%LIBHOME%\wsdl4j-1.6.2.jar;%LIBHOME%\XmlSchema-1.4.7.jar;%LIBHOME%\com.sap.translation.sdk.jar
set PATH=C:\Windows\System32;%BOINSTALLDIR%;%PATH%
javac TApi.java
java -cp C:\TestApiTests -Djava.library.path="C:\Windows\System32;%BOINSTALLDIR%" TApi

Note

- The "TApi.java" is a sample java file.

6.2 Creating TMT SDK Workflows

6.2.1 Implementation Code for Importing InfoObjects from CMS

Using the Translation Manager SDK, you can import multiple InfoObjects or single InfoObject.

Code Snippet for importing InfoObject from CMS is given below:

Sample Code

```
String strQuery=null;
ISessionMgr sessionManager = CrystalEnterprise.getSessionMgr();
ceEnterpriseSession =
sessionManager.logon("Administrator","Password1","10.160.209.188:6400","secEnt
erprise");
ceEnterpriseSession.setLocale(Locale.ENGLISH);
IInfostore infostore = (IInfoStore)
ceEnterpriseSession.getService("InfoStore");
strQuery = "select * from ci_infoobjects where si_name='" + docname + "'" +
"and
        SI_PARENT_FOLDER='" + parentfolderid + "'";

IInfoObjects ceInfoObjects = (IInfoObjects) infostore.query(strQuery);
if (ceInfoObjects.size() != 0)
infoObj = (IInfoObject) ceInfoObjects.get(0);
else
    //May be an application object
    {
    strQuery="select * from ci_appobjects where si_name='" + docname + "'" + "and
    SI_PARENT_FOLDER='" + parentfolderid + "'";
    ceInfoObjects = (IInfoObjects) infostore.query(strQuery);
    if (ceInfoObjects.size() != 0)
    infoObj = (IInfoObject) ceInfoObjects.get(0);
    }
System.out.println("Importing " + infoObj.getTitle() + " " +
infoObj.getKind() + " from CMS");
catch(Exception e)
{
return null;
}
```

6.2.2 Implementation Code for Importing InfoObjects from Local Folder

Using the translation manager SDK, you can import data foundation(.dfx), business layer(.blx) and universe documents(.unv) from local folder, and translation manager Document (.tmgr) to perform the translations.

Code snippets for importing .unv, .dfx, .blx, .tmgr:

Sample Code

```
//Importing the .unv file from local folder
File filepath = new File("C:\\Universes\\Univers2.unv");
TranslationSDKManager manager = new TranslationSDKManager();
ITMgrDocument tmgrdoc =
manager.loadArtifact(TranslatableEntity.UNIVERSE,filepath);
ITMgrEngine engine = ITMgrEngine.Factory.createInstance(tmgrdoc);
```

Sample Code

```
//Importing the .dfx file from local folder
File filepath = new File("C:\\Universes\\NConnection.dfx");
TranslationSDKManager manager = new TranslationSDKManager();
ITMgrDocument tmgrdoc =
manager.loadArtifact(TranslatableEntity.DATAFOUNDATION,filepath);
ITMgrEngine engine = ITMgrEngine.Factory.createInstance(tmgrdoc);
```

Sample Code

```
// Importing .blx from the local folder
File filepath = new File("C:\\Universes\\NConnection.blx");
TranslationSDKManager manager = new TranslationSDKManager();
ITMgrDocument tmgrdoc =
manager.loadArtifact(TranslatableEntity.UNIVERSE,filepath);
ITMgrEngine engine = ITMgrEngine.Factory.createInstance(tmgrdoc);
```

Sample Code

```
//Importing .tmgr from the local folder
File filepath = new File("C:\\Universes\\NConnection.tmgr");
TranslationSDKManager manager = new TranslationSDKManager();
ITMgrDocument tmgrdoc =
manager.loadArtifact(TranslatableEntity.TRANSMGR,filepath);
ITMgrEngine engine = ITMgrEngine.Factory.createInstance(tmgrdoc);
```

6.2.3 Implementation Code for Extracting Translations From the InfoObject

Use the following code snippet to extract the Meta data translations from the InfoObject.

Sample Code

```
TranslationSDKManager manager = new
TranslationSDKManager(ceEnterpriseSession);
InputStream translations = manager.extractTranslations(infoObj.getCUID());
return translations;
```


6.2.4 Implementation Code for Getting the Translation Manager Engine

After getting the metadata translations from InfoObject, you first need to get the translation manager document and then the translation manager engine from the document. Use the following code snippet to perform the operation:

Sample Code

```
Public static ITMgrDocument GetTmgrDoc (InputStream inputstream, IInfoObject
infoobj) throws TranslationException
{
    try
    {
        ITMgrDocument tmgrdoc = null;
        System.out.println("Saving as " + infoobj.getTitle() + ".tmgr file under C:\
\sdk\ folder\n" );
        FileOutputStream os = new FileOutputStream("C:\\sdk\\" + infoobj.getTitle() +
".tmgr");

        byte[] buffer = new byte[1024];
        int bytesRead;
        //read from is to buffer
        while((bytesRead = inputstream.read(buffer)) !=-1){
            os.write(buffer, 0, bytesRead);
        }
        //flush OutputStream to write any buffered data to file
        os.flush();
        os.close();
        //Gets the Translation Manager Document
        return manager.loadArtifact(TranslatableEntity.TRANSMGR,new File("C:\\sdk\
\\"+infoobj.getTitle() + ".tmgr"));
    }
    catch(FileNotFoundException
    {
        e.printStackTrace();
        return null;
    }
    catch(IOException e1)
    {
        e1.printStackTrace();
        return null;
    }
}
//Gets the Translation Manager Engine
ITMgrEngine
engine=TMgrEngine.Factory.createInstance(GetTmgrDoc(translations,infoObj));
```

6.2.5 Implementation Code for Getting and Setting Available locale, Visible and Fallback Locale

6.2.5.1 Setting and Getting Available Locale

You can add single or multiple locales from the available locales using the TMT SDK.

Example:

Sample Code

```
//Adds the Available Locale
engine.addAvailableLocale(Locale.US);
engine.getAvailableLocales(); //Getting the available locales
```

6.2.5.2 Setting the Translation Engine with the Translated Locale Value

You can set the Translation Engine with the value of translated locale.

Example:

Sample Code

```
//Sets the Translation Engine with translated locale value
engine.setModifiedLocale(Locale.US);
```

6.2.5.3 Remove Available Locale

Using TMT SDK, you can remove single or multiple locales.

Example:

Sample Code

```
//Removes the Available Locale
engine.removeAvailableLocale(Locale.US);
```

6.2.5.4 Setting and Getting Visible locale

Using TMT SDK, you can set any locale as visible from available locales list for a given document.

Example:

Sample Code

```
//Sets the locale as visible
engine.setLocaleVisible(Locale.US,true);

//Gets the visible locales
engine.getVisibleLocales();
```

6.2.5.5 Setting and Getting Fallback locale

Using TMT SDK, you can set any locale as fallback from available locales list for a given document.

Note

If you have added single locale, by default the fallback locale is the original content language.

Example:

Sample Code

```
//Sets the locale as fallback
engine.setFallbackLocale(Locale.US,true); //Setting the fallback locale
//Gets the fallback locale
engine.getFallbackLocale();
```

6.2.6 Implementation Code for Getting and Setting the Property Values and Property Status of Entities

Using the Translation Manager SDK, you can get and set the property values and property status of entities in Translation Manager document (.tmgr).

Using the following code snippet, you can get and set Property Values of root entity from Translation Manager document (.tmgr) file:

Sample Code

```
System.out.println("--MODIFYING THE VALUES--WITH--"+name+"---
AND--"+description);
Locale locales[] = FormLocales();
Object rootEntity = engine.getRoot();
IPropertyInfo[] propertyinforoot = engine.getLocalizedProperties(rootEntity);
if(propertyinforoot.length == 0)
{
Object[] obj = engine.getChilden(rootEntity);
if(obj.length > 0) //this means it is only rootentity
{
for(int index=0;index < obj.length;index++)
{
```

```

IPropertyInfo[] propertyInfo = engine.getLocalizedProperties(obj[index]);
for(int index1=0; index1
    < locales.length; index1++)
{
    System.out.println("Id is -->" + propertyInfo[0].getId());
    System.out.println("Description is --->" + propertyInfo[1].getId());
    System.out.println("Locales index is --->" +
        locales[index1].getDisplayLanguage());
    propertyInfo[0].setValue(obj[index],locales[index1],name+locales[index1].getCo
        untry());
    propertyInfo[1].setValue(obj[index],locales[index1],description+locales[index1
        ].getCountry());
}
}
}
else
{

    IPropertyInfo[] propertyInfo = engine.getLocalizedProperties(rootEntity);
    for(int index1=0; index1 < locales.length; index1++)
    {

        propertyInfo[0].setValue(rootEntity,locales[index1],name+locales[index1].getCo
            untry());

        propertyInfo[1].setValue(rootEntity,locales[index1],description+locales[index1
            ].getCountry());
    }
}
}

```

Using the following code snippet, you can get and set Property status of root entity from Translation Manager document (.tmgr):

Sample Code

```

System.out.println("--MODIFYING THE STATUS--WITH--"+status.name());
Locale locales[] = {Locale.US,Locale.FRANCE};
Object rootEntity = engine.getRoot();
IPropertyInfo[] propertyInfo = engine.getLocalizedProperties(rootEntity);
if(propertyInfo.length==0)
{
    Object[] obj = engine.getChildren(rootEntity);
    if(obj.length > 0) //this means it is only root entity
    {
        for(int index=0;index < obj.length;index++)
        {
            IPropertyInfo[] propertyInfo = engine.getLocalizedProperties(obj[index]);
            for(int index1=0; index1 < locales.length; index1++)
            {
                propertyInfo[0].setStatus(obj[index],locales[index1],status);
                propertyInfo[1].setStatus(obj[index],locales[index1],status);
            }
        }
    }
}
else
{

    IPropertyInfo[] propertyInfo = engine.getLocalizedProperties(rootEntity);
    for(int index1=0; index1 < locales.length; index1++)
    {
        propertyInfo[0].setStatus(rootEntity,locales[index1],status);
        propertyInfo[1].setStatus(rootEntity,locales[index1],status);
    }
}
}

```

```
}
```

6.2.7 Implementation Code for Saving Translation Manager Document (.tmgr file) to the Local Folder

Using the following code snippet, you can save the translation manager document (.tmgr) file locally, open the file, and perform translation.

Sample Code

```
//Saves the translation manager document(.tmgr).  
engine.save();
```

6.2.8 Implementation Code for Exporting Translation Manager Document (. tmgr file) to the CMS

Using the following code snippet, you can export the InfoObjects to the system repository after performing translations.

Sample Code

```
try  
{  
    engine.save();  
    ByteArrayOutputStream tmgrStream = new ByteArrayOutputStream();  
    FileDocumentFactory.exportToTMgr(engine,tmgrStream);  
    InputStream saveResult =  
        manager.saveTranslations(infoObj.getCUID(),tmgrStream,true);  
    System.out.println("InfoObject cuid is " + infoObj.getCUID());  
    TMgrMergeReport mergeResult = new TMgrMergeReport(saveResult);  
    if(mergeResult.hasConflict())  
    {  
        System.out.println("Conflict has occurred");  
    }  
    else  
    {  
        System.out.println("Translations are saved");  
    }  
} catch(Exception e)  
{  
    e.printStackTrace();  
}
```

6.2.9 Implementation Code for Exporting Translation Manager Document (.tmgr file) to XLIFF file

Using the following code snippet, you can export the translation manager document to XLIFF file.

Sample Code

```
try
{
    IEntityInfo info = engine.getEntityInfo(engine.getRoot());
    IXliffExporter exportxliff = manager.createXliffExporter();
    exportxliff.setSourceLocale(Locale.US,true);
    exportxliff.setTargetLocale(Locale.ITALY,true);
    File path = new File("C:\\sdk\\sundar.xliff");
    exportxliff.write(info,path);
}
catch(TranslationException ex)
{
    ex.getMessage();
}
```

6.2.10 Implementation Code for Importing Translation Manager Document from XLIFF file

Using the following code snippet, you can import the translation manager document from XLIFF file:

Sample Code

```
IXliffImporter importxliff = manager.createXliffImporter();
File path3 = new File("C:\\sdk\\sundar.xliff");
File path2 = new File("C:\\sdk\\POC4.tmgr");

ITMgrDocument
tmgrdocim=manager.loadArtifact(TranslatableEntity.TRANSMGR,path2);
TMgrEngine engine2 = new TMgrEngine(tmgrdocim);
engine2.save();
importxliff.load(engine2,path3);
```

6.2.11 Implementation Code for Exporting Translation Manager Document to XLS file

Using the following code snippet, you can export the translation manager document to XLS file.

Note

The Excel (.xlsx) format is not supported.

Sample Code

```
IExcelExporter exportexcel = manager.createExcelExporter();
path = new File("C:\\sdk\\sundar.xls");
Locale loc[] = { Locale.JAPAN,Locale.ITALY};
exportexcel.setSourceLocale(Locale.US);
exportexcel.setTargetLocale(loc);
exportexcel.write(engine, path);
```

6.2.12 Implementation Code for Importing Translation Manager Document from XLS file

Using the following code snippet, you can import the translation manager document from Excel (.xls) file:

Sample Code

```
IExcelImporter importexcel = manager.createExcelImporter();
importexcel.load(engine,path,new File("C:\\sdk\\excel.properties"));
engine.saveAs(new File("C:\\sdk\\POCXLS.tmgr"));
```

6.2.13 Sample Implementation Code

Sample implementation code to extract the InfoObject from CMS, get the meta data translations, set available ,fallback and visible locale, setting the property status, saving the engine and exporting it to CMS

Sample Code

```
String strQuery=null;
ISessionMgr sessionManager = CrystalEnterprise.getSessionMgr();
ceEnterpriseSession =
sessionManager.logon("Administrator","Password123","localhost:6400","secEnterp
rise");
ceEnterpriseSession.setLocale(Locale.ENGLISH);
IInfoStore infostore = (IInfoStore)
ceEnterpriseSession.getService("InfoStore");
strQuery = "select * from ci_infoobjects where si_name='" + docname + "'" +
"and SI_PARENT_FOLDER='" + parentfolderid + "'";

IInfoObjects ceInfoObjects = (IInfoObjects) infostore.query(strQuery);
if (ceInfoObjects.size() != 0)
infoObj = (IInfoObject) ceInfoObjects.get(0);
else //May be an application object
{
strQuery="select * from ci_appobjects where si_name='" + docname + "'" + "and
SI_PARENT_FOLDER='" + parentfolderid + "'";
ceInfoObjects = (IInfoObjects) infostore.query(strQuery);
if (ceInfoObjects.size() != 0)
infoObj = (IInfoObject) ceInfoObjects.get(0);
}
System.out.println("Importing " + infoObj.getTitle() + " " +
infoObj.getKind() + " from CMS");
```

```

manager = new TranslationSDKManager(ceEnterpriseSession);
    InputStream translations =
manager.extractTranslations(infoObj.getCUID());
ITMgrEngine engine =
ITMgrEngine.Factory.createInstance(GetTmgrDoc(translations,infoObj));
engine.removeAvailableLocale(Locale.US);
engine.addAvailableLocale(Locale.US);
engine.setFallbackLocale(Locale.US);
engine.setVisibleLocale(Locale.US,true);
IPropertyInfo[] propertyInfo = engine.getLocalizedProperties(rootEntity);
propertyInfo[0].setStatus(rootEntity,Locale.US,status);
engine.setModifiedLocale(Locale.US);
engine.save();
ByteArrayOutputStream tmgrStream = new ByteArrayOutputStream();

FileDocumentFactory.exportToTMgr(engine,tmgrStream);

InputStream saveResult =
manager.saveTranslations(infoObj.getCUID(),tmgrStream,true);

System.out.println("InfoObject cuid is " + infoObj.getCUID());

TMgrMergeReport mergeResult = new TMgrMergeReport(saveResult);

if(mergeResult.hasConflict())
{
    System.out.println("Conflict has occurred");
}
else
{
    System.out.println("Translations are saved");
}

```


7 Testing TMT SDK Workflows

Before testing, you need to check for the following requirements:

1. SAP Business Intelligence Server must be installed (local or remote), and CMS must be running.
2. Adaptive Processing Server must have translation service running.
3. If translation service JAR is present under %BOINSTALLDIR%\ SAP BusinessObjects Enterprise XI 4.0\java\pjs\services\TranslationService\lib path.

ⓘ Note

- Client and server installation must be available (Server can be installed locally or remotely)
- Ensure that Adaptive Processing Server (APS) must be hosting the translation service in the server machine and also corresponding plugins needed for translation is present in the server.

8 Using TMT SDK

Now that you have developed and tested your application, you can use this application to perform the following workflows:

- Importing the InfoObject from CMS: [Implementation Code for Importing InfoObjects from CMS \[page 15\]](#)
- Importing the InfoObject from local folder: [Implementation Code for Importing InfoObjects from Local Folder \[page 15\]](#)
- Extracting meta data translations: [Implementation Code for Extracting Translations From the InfoObject \[page 16\]](#)
- Getting the Translation Manager Engine: [Implementation Code for Getting the Translation Manager Engine \[page 17\]](#)
- Setting and Getting the available locale: [Setting and Getting Available Locale \[page 18\]](#)
- Setting the Translation Engine with the Translated Locale Value: [Setting the Translation Engine with the Translated Locale Value \[page 18\]](#)
- Setting and Getting the visible locale: [Setting and Getting Visible locale \[page 18\]](#)
- Setting and Getting the fallback locale: [Setting and Getting Fallback locale \[page 19\]](#)
- Setting and Getting the property value and property status: [Implementation Code for Getting and Setting the Property Values and Property Status of Entities \[page 19\]](#)
- Saving the Translation Manager Document (.tmgr) file: [Implementation Code for Saving Translation Manager Document \(.tmgr file\) to the Local Folder \[page 21\]](#)
- Exporting to XLIFF file: [Implementation Code for Exporting Translation Manager Document \(.tmgr file\) to XLIFF file \[page 22\]](#)
- Importing from XLIFF file: [Implementation Code for Importing Translation Manager Document from XLIFF file \[page 22\]](#)
- Exporting to XLS file: [Implementation Code for Exporting Translation Manager Document to XLS file \[page 22\]](#)
- Importing from XLS file: [Implementation Code for Importing Translation Manager Document from XLS file \[page 23\]](#)
- Exporting translations back to the CMS: [Implementation Code for Exporting Translation Manager Document \(.tmgr file\) to the CMS \[page 21\]](#)

9 Limitations

The following are the limitations for TMT SDK:

- Using Translation Manager SDK, you cannot open a Web Intelligence file (.wid) from a local folder and do the translations.
- When exporting Translation Manager Document (.tmgr) document to XLS file, the property values are saved in XLS file, but property status is not saved in XLS file.
- Once the locale is added using SDK, if you attempt to add the same locale, no exception is thrown for now. However, you can customize the SDK code to throw an error in this particular situation.
- Using SDK, if you try to remove a Locale, which is not yet added, no exception is thrown. However, you can use the below SDK code to avoid this situation.

Sample Code

```
Locale loc[] = engine.getAvailableLocales();
Locale toremove = Locale.US;
for(int index = 0; index < loc.length; index++)
{
    If(toremove.getDisplayName().compareTo(loc[index].getDisplayName() )
    == 0)
        Engine.removeLocale(toremove);
    else
        Throw TranslationException(e);
}
```

10 Troubleshooting Tips

- **Problem:**
While writing the Translation Manager SDK code, `ClassNotFoundException` is thrown.
Action:
Make sure that you have added all the required JAR files under `%BOINSTALLDIR%\java\lib\TranslationManagerSDK` is set in the `<CLASSPATH>`.
- **Problem:**
Exception is thrown while importing InfoObject from Central Management Server (CMS).
Action:
Make sure that the CMS is up and running and also, the translation service is available for Adaptive Processing Server (APS).
- **Problem:**
When `engine.save ()` is used, not able to locate where the Translation Manager Document (.tmgr file) is saved.
Action:
By default, the Translation Manager Document gets saved under `C:\Users\Administrator` path.
- **Problem:**
The Translation Manager Document (.tmgr file) is corrupted; opening a Translation Manager Document (.tmgr file) through SDK fails.
Action:
Remove all the available locales, add them again, and then perform the translations.
- **Problem:**
Encountered an error while executing a workflow.
Action:
Try executing the same workflow using Translation Management Tool (TMT) to verify whether it works fine on TMT.
- **Problem:**
Encountering `NullPointerException` Exception.
Action:
Make sure that the InfoObject from CMS is imported successfully, before proceeding to get the meta data from the InfoObject.

11 Frequently Asked Questions

1. *How to make sure all JAR Files are available for Translation Manager SDK to work?*

Once you install Business Objects Client, click [PlatformJavaSDK](#), and make sure all the relevant JAR files are present under %BOINSTALLDIR%\java\lib\TranslationManagerSDK. The JAR file list is present in the *Setting up the TMT SDK Environment* section of this guide.

2. *How to make sure Adaptive Processing Server has translation service running?*

Once you install Business Objects Server locally or remotely perform the following to check whether the translation service is running.:

1. Launch Central Management Console (CMC) portal.
2. Select [Servers](#).
3. Under servers list, choose [Core Services](#) then choose [Adaptive Processing Server](#).
4. Right click on [Adaptive Processing Server](#) under [Services label](#), choose [Metrics](#) to see the list of services hosted.

3. *How to do the basic workflows using Translation Manager SDK?*



You must first get the `ITMgrEngine` using the Translation Manager APIs. Once you have this, using the `ITMgrEngine` you can perform all the operations listed under **Creating TMT SDK Workflows** section.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon  : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
 - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
 - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon  : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.