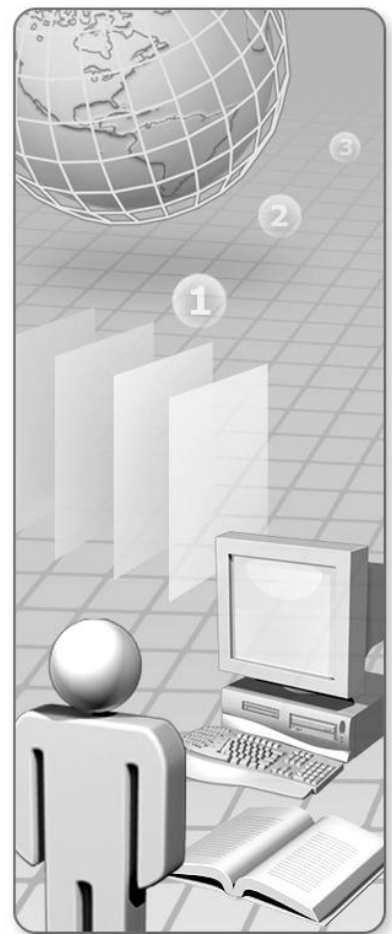


.NET Framework Fundamentals Assessment



Contents

Business Requirement.....	3
Sample File Format:	4
Pre-requisites:	5
API Technical Specifications:	6
How to upload the completed solution file:	10

Business Requirement

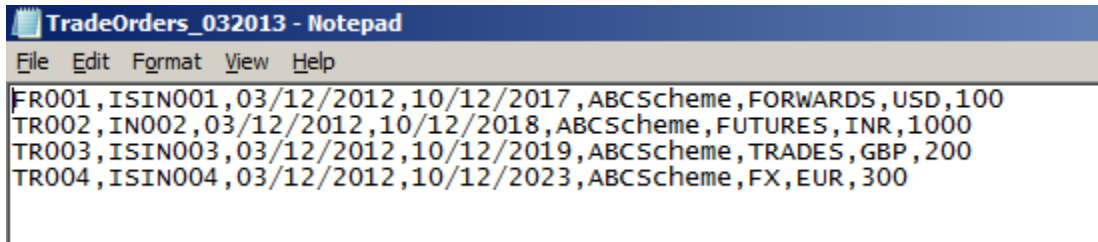
ABC bank is involved in investing and purchasing shares in share market. They have different trade activities for each month. The trade activities are identified based on the Trade Type.

The trade types are as follows.

- FUTURES
- FORWARDS
- TRADES
- FX

The production facility receives the consolidated trade details which are uploaded in a server in the form of a text file. The naming convention which they follow for their text file is **TradeOrder_mmyyyy.txt** which is a comma separated file.

Sample File Format:



```
TradeOrders_032013 - Notepad
File Edit Format View Help
FR001,ISIN001,03/12/2012,10/12/2017,ABCScheme,FORWARDS,USD,100
TR002,IN002,03/12/2012,10/12/2018,ABCScheme,FUTURES,INR,1000
TR003,ISIN003,03/12/2012,10/12/2019,ABCScheme,TRADES,GBP,200
TR004,ISIN004,03/12/2012,10/12/2023,ABCScheme,FX,EUR,300
```

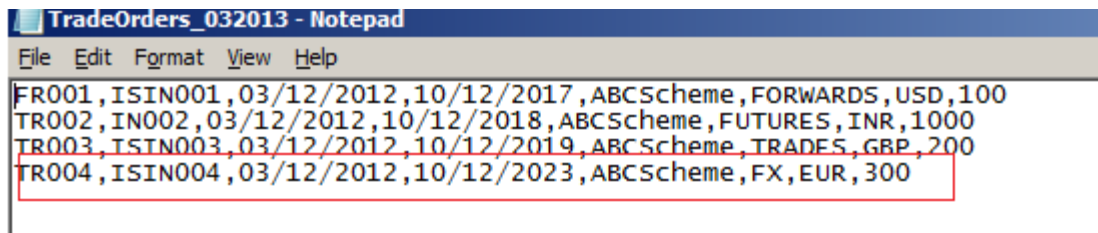
The records in the file are in the following order.

<TradeID>,<ISIN>,<TradeDate>,<MaturityDate>,<SchemeName>,<TradeType>,<Currency>,<Amount>

You, as a .NET expert have been requested to write a small console application to read all the records from the text file available in the Directory and update in the database only with valid records.

The validations are specified in the method [pickValidTradeDetails\(\)](#) below.

Also, you need to calculate Forex Rate only for Trade Type “FX”.



```
TradeOrders_032013 - Notepad
File Edit Format View Help
FR001,ISIN001,03/12/2012,10/12/2017,ABCScheme,FORWARDS,USD,100
TR002,IN002,03/12/2012,10/12/2018,ABCScheme,FUTURES,INR,1000
TR003,ISIN003,03/12/2012,10/12/2019,ABCScheme,TRADES,GBP,200
TR004,ISIN004,03/12/2012,10/12/2023,ABCScheme,FX,EUR,300
```

Note: The calculation part are clearly explained in the method [CalculateFxRate\(\)](#) below.

.

Pre-requisites:

Scripts will be provided to create the database and the tables. Below two tables should be there in the Database “**DBFXCalculation**” with the below columns and data types.

SBA.Trade_Details Table:

Field Name	Data Type
TradeID	varchar(5)
ISIN	varchar(7)
TradeDate	datetime
MaturityDate	datetime
SchemeName	varchar(100)
TradeType	varchar(50)
Currency	varchar(3)
Amount	numeric(18, 0)

SBA.FX_Rate Table:

Field Name	Data Type
TradeID	varchar(5)
Currency	varchar(3)
Amount	numeric(18, 0)
AppliedFXRate	float
CalculatedFXRate	float

- Do not add any new tables or other entities.
- Use only the given tables.
- Do not use stored procedures.
- Use the schema SBA in front of the table names while accessing the tables. Please do not delete the schema

API Technical Specifications:

ClassName	Method Name	Sample Input	Sample Output
FXCalculator	<pre>public void ProcessData(string sourceFolder, string fileName, string errorLogFilePath, string errorLogFileName, SqlConnection connectionObject, string archiveFilePath, string archiveFileName)</pre>	<p>"D:\Dotnetcaassessment\FXC alculator\Input file", "TradeOrders_032013.txt"</p> <p>"@"Data Source=PCName\SQLEXPRESS;Initial Catalog= DBNAVCalculation;Integrated Security=True"</p> <p>"D:\Dotnetcaassessment\FXC alculator\Input file\Archive", TradeOrders_032013_Proces sed.txt"</p>	
	<pre>public List<Trade> ReadAllDataFromInputFile(str ing sourceFolder, string fileName)</pre>	<p>"D:\Dotnetcaassessment\FXC alculator\Input file", "TradeOrders_032013.txt"</p>	<p>Trade< TR002,ISIN002,03/12 /2012,10/12/2018,AB CScheme,FUTURES, INR,1000></p> <p>*Note: Return only the date part. Don't return date with time stamp</p>
	<pre>public List<Trade> pickValidTradeDetails(List<T rade> trades,string errorLogFilePath, string errorLogFileName)</pre>	<p>Sample for Valid Trade Details</p> <p>Trade<TR002,ISIN002,03/12/ 2012,10/12/2018,ABCScheme ,FUTURES,INR,1000>, "D:\Dotnetcaassessment\FXC alculator\ErrorLog", "InvalidRecords_032013.txt"</p> <p>Sample for Invalid Trade Details</p> <p>Trade <FR001,ISIN001,03/12/2012, 10/12/2017,ABCScheme,FOR WARDS,USD,100>,"D:\Dotnet caassessment\FXCalculator\E rrorLog", "InvalidRecords_032013.txt"</p>	<p>Trade <TR002,ISIN002,03/1 2/2012,10/12/2018,A BCScheme, FUTURES,INR,1000></p> <p>*Note: Return only the date part. Don't return date with time stamp</p>
	<pre>public bool SaveValidRecordsToDB(List<Tr ade> validTrades, SqlConnection sqlConnectionObject)</pre>	<p>Trade<TR002,ISIN002,03/12/ 2012,10/12/2018,ABCScheme ,FUTURES,INR,1000>, @"Data Source=PCName\SQLEXPRESS;Initial Catalog=</p>	<p>true</p>

		DBNAVCalculation;Integrated Security=True"	
	<pre>public bool SaveInvalidRecordsToLogFile(List<Trade> invalidTrades, string errorLogFilePath, string errorLogFileName)</pre>	<pre>Trade<TR002,ISIN002,03/12/ 2012,10/12/2018,ABCScheme ,FUTURES,INR,1000>, "D:\Dotnetcaassessment\FXC alculator\ErrorLog\"," "InvalidRecords_032013.txt"</pre>	true
	<pre>public List<FXRate> CalculateFXRate(SqlConnectio n sqlConnectionObject)</pre>	<pre>Trade<TR002,ISIN002,03/12/ 2012,10/12/2018,ABCScheme ,FX,USD,1000>, "@Data Source=PCName\SQLEXPRES S;Initial Catalog= DBNAVCalculation;Integrated Security=True"</pre>	FXRate< TR004,INR,1000,0.5, 500>
	<pre>public bool SaveFXRate(List<FXRate> fxRates, SqlConnection sqlConnectionObject)</pre>	FXRate< TR004,INR,1000,0.5,500>	true
	<pre>public bool CopyToArchive(string sourcePathWithFileName, string targetPathWithFileName)</pre>	<pre>"D:\Dotnetcaassessment\FXC alculator\Input file\TradeOrders_032013.txt, "D:\Dotnetcaassessment\FXC alculator\Input file\Archive\TradeOrders_032 013_Processed.txt"</pre>	true

Method Description: ProcessData()

1. Call all the methods given in specification under this ProcessData method.
2. If any error occurs such as "File not exists" or "Source folder does not exists", an user defined exception called "**FXCalculationException**" has to be thrown from this method.

* The Exception class is already provided in the template.

Method Description: ReadAllDataFromInputFile()

1. Read the records from the input file. Sample input file is available in the template path
2. Return it in List collection.
3. ***Note: Return only the date part. Don't return date with time stamp**
4. If any exception occurs throw in "**FXCalculationException**".

Method Description: **pickValidTradeDetails()**

1. Do the below validations for the records in the input file and Return the List collection **only with valid records**.

DATA TYPE	VALIDATIONS
TradeID	Should start with "TR". Should not be null eg. TR003
ISIN	Should start with "ISIN" followed by 3 digit numeric literals. e.g. ISIN003
TradeDate	Should be a valid date field. Should be in "mm/dd/yyyy" format.
MaturityDate	<ul style="list-style-type: none">• Should be a valid date field. Should be in "mm/dd/yyyy" format.• Should be greater than 5 years from TradeDate
TradeType	Should not be null
Currency	<ul style="list-style-type: none">• Should be 3 letters.• should not be null.
Amount	<ul style="list-style-type: none">• Should be numeric.• Should not be null.

2. Save the invalid records in the error log file which is a text file. (Details about Invalid records and Error Log File are given in SaveInvalidTradesToLogFile method).
3. Call the SaveInvalidTradesToLogFile() method.
4. Return the List collection only with valid records.

Method Description: **SaveInvalidTradesToLogFile()**

1. Save the invalid records in the error log file which is a text file.
2. The error log file name should be "InvalidRecords_mmyyyy.txt"
3. Save the error log file in the "ErrorLog" folder which is available in the template path.
4. Return true if the file is saved successfully in the given file path.

Method Description: **SaveValidTradesToDB()**

1. Save the valid records in the database under the table "Trade_Details"
2. Return true if the values are saved successfully to the database.

Method Description: CalculateFxRate()

1. Calculate FX Rate for the Trade type "FX". It should be calculated as per the below formula,

If Currency is USD then FX Rate = Amount * 0.5

If Currency is GBP then FX Rate = Amount * 0.6

If Currency is EUR then FX Rate = Amount * 0.7

If Currency is INR then FX Rate = Amount*1.

2. Return the calculated FX rates in a List.
3. 0.5, 0.6, 0.7, etc., are the Applied FX Rates

Method Description: SaveFXRate()

1. Save the calculated FX Rate in the table "FX_Rate" along with the fields <TradeID>,<Currency>,<Amount>,<AppliedFXRate>, <CalculatedFXRate>
2. Return true if the values are saved successfully to the database.

Method Description: CopyToArchive()

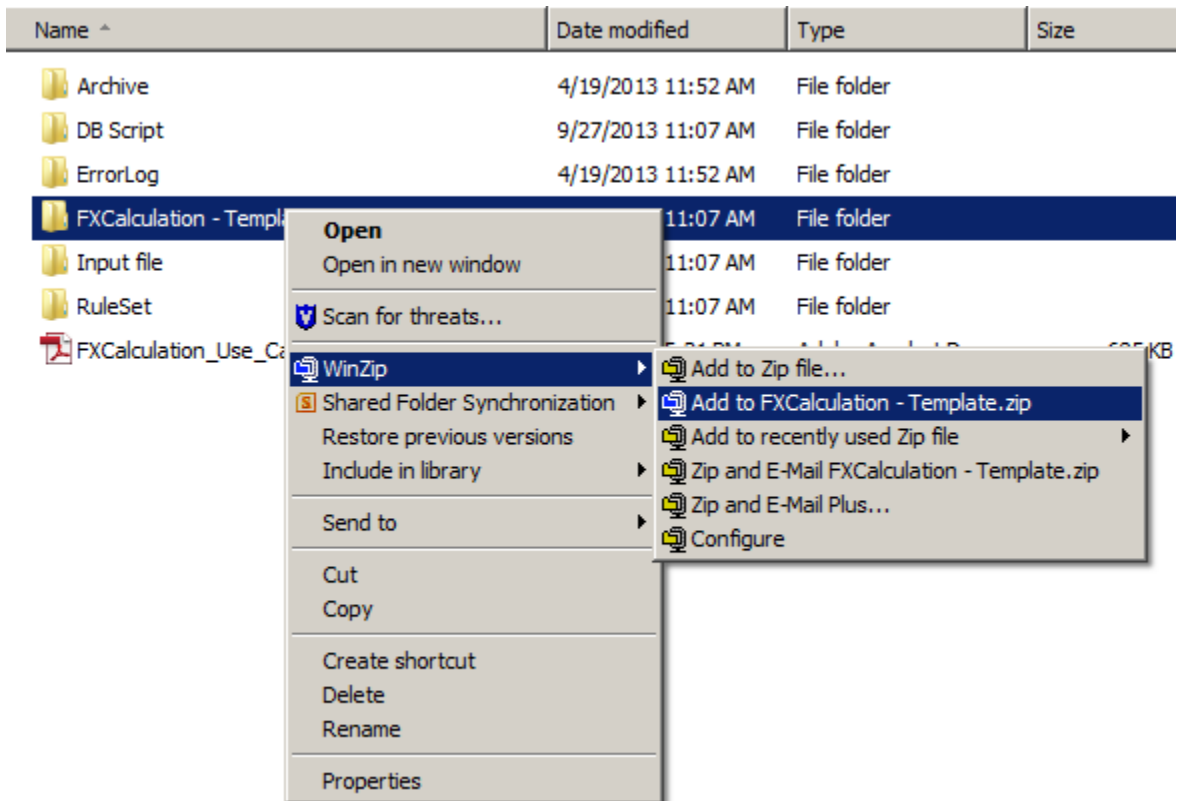
1. Create a copy of the input file and place it inside the archive folder.
2. The Archive folder is available in the zip file.
3. Rename the file to "TradeDetails_032013_Processed.txt"
4. If a file named "TradeDetails_032013_Processed " is already exists then delete the existing one and copy the new one.

Note:-

- DO NOT ALTER the METHOD SIGNATURES in the code template.
- Do not use stored procedures, use inline queries to connect to database.
- Do not hardcode the connection string or file names anywhere in the code. Use the existing variable names.

How to upload the completed solution file:

1. Save the completed solution and close it.
2. Right Click the “FXCalculation - Template” folder
3. Click on WinZip followed by “Add to FXCalculation - Template.zip” as shown below,



4. Rename the “FXCalculation - Template.zip” file to “YourAssociateId_ FXCalculation”.zip as shown below

Name ^	Date modified
Archive	4/19/2013 11:52 AM
DB Script	9/27/2013 11:07 AM
ErrorLog	4/19/2013 11:52 AM
FXCalculation - Template	9/27/2013 11:07 AM
Input file	9/27/2013 11:07 AM
RuleSet	9/27/2013 11:07 AM
123456_FXCalculation	9/27/2013 12:29 PM
FXCalculation_Use_Case Document	6/14/2013 5:21 PM

5. Upload the renamed zip file to the assessment server.