# TAFJ-Distribution

## R14/R15

10/3/2015
Temenos

# TAFJ-Distribution

**Amendment History:**

| Revision | Date Amended | Name | Description |
|---|---|---|---|
| 1 | 1$^{st}$ April 2011 | TAFJ team | Initial version |
| 2 | 7$^{st}$ February 2012 | H. Aubert | R12GA review |
| 3 | 16$^{th}$ January 2013 | Manoj Kumar | R13GA review |
| 4 | 20$^{th}$ February 2013 | R. Vincent | R14GA Review |
| 5 | 15th April 2014 | H. Aubert | R14GA review |
| 6 | 6th March 2015 | H. Aubert | R15 AMR review |

## Copyright

## Errata and Comments

If you have any comments regarding this manual or wish to report any errors in the documentation, please document them and send them to the address below:

Technology Department

Temenos Headquarters SA

2 Rue de l'Ecole-de-Chimie,

CH - 1205 Geneva,

Switzerland

Tel SB: +41 (0) 22 708 1150

Fax: +41 (0) 22 708 1160

Please include your name, company, address, and telephone and fax numbers, and email address if applicable. TAFJdev@temenos.com

# TAFJ-Distribution

# Table of Contents

# TAFJ-Distribution

## tPackager:

### Introduction

tPackager is mainly used for Bytecode encryption.  It encrypts the .class files so that nobody can decompile our .class files into java source files.

### Syntax

Execute tPackager.bat or tPackager from bin.

```
tPackager —encrypt path

tPackager —jar jarfilename path

tPackager —encrypt path —jar jarfilename
```

In the above, the 1st option will encrypt .class files from the specified directory.  The 2nd option will create a jar file from the specified path.  The 3rd option will encrypt and create a jar file from the specified path.

Note: If you want to package the jar from 'com' directory as root (for example com.temenos.t24), then give the path like c:\plugintest\com, so the jar file will contain all files inside 'com' directory.

**!! WARNING  !!**

If you decide to have T24.jar encrypted, ALL precompiled jars loaded by TAFJ have to be encrypted used and define with the key: **"temn.tafj.directory.precompile".**

**This function doesn't work with the classloader of the Application Server**
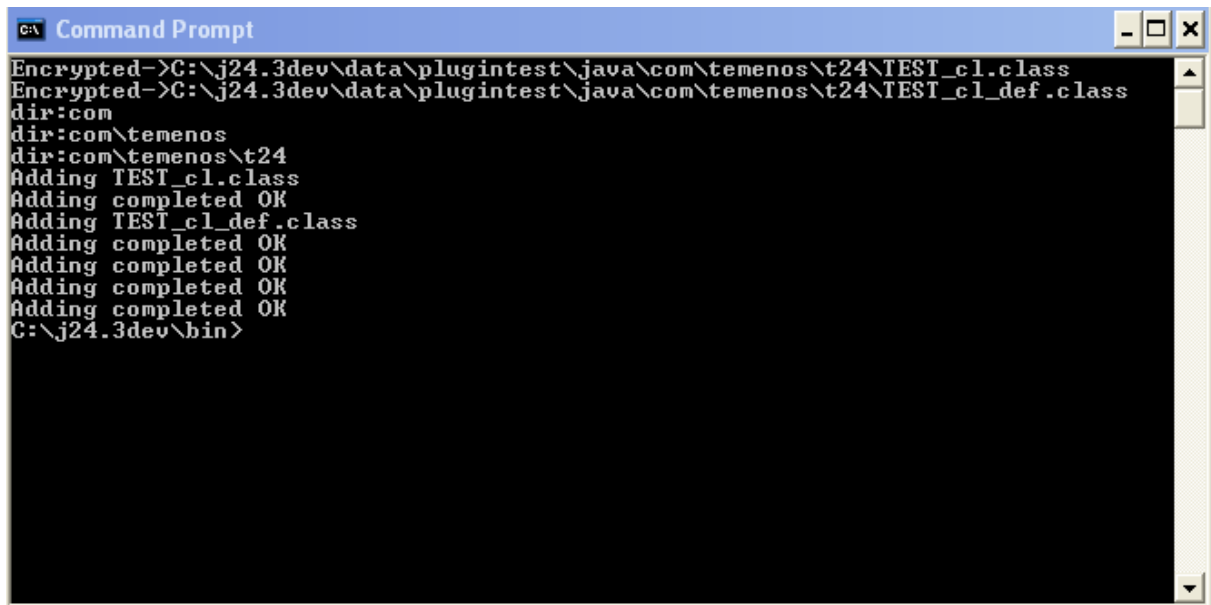
**Example**

After executing the batch file by following example:

```
        tPackager —encrypt c:\plugintest\ —jar enc.jar
```

tPackager —encrypt c:\plugintest\ —jar enc.jar.
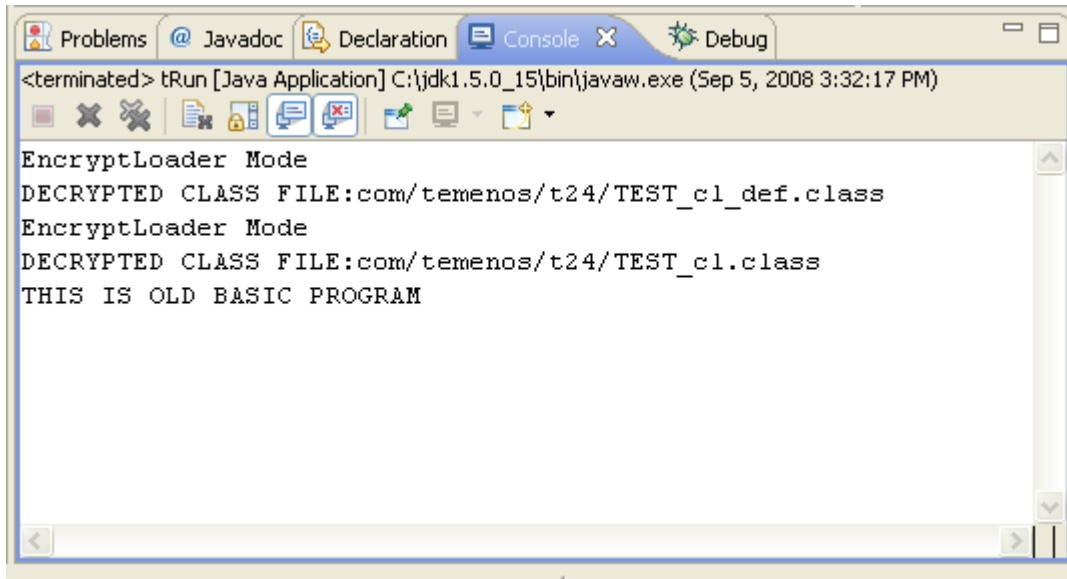
The resulting output looks like this:



**Steps to use Encrypt Classloader:**

TAFJ has two classloaders, one is general classloader and another is Encryptclassloader.  If your .class files are encrypted then you must use Encryptclassloader. This classloader will decrypt the class files at runtime.  To use this classloader take the properties file and **add** the field '**temn.tafj.runtime.custom.classloader=jEncryptLoader'**.  If you are using the default classloader then give 'jClassLoader' to this field.

i.e

```
# Enable the custome classloader. Desabling it could be necessary when running in an
# application server. If this flag is set to false, then you must explicitly
# specify the T24 classes in the classpath. Default is 'jClassLoader'
#
temn.tafj.runtime.custom.classloader          = jEncryptLoader
```

While executing encrypted class files the output will be like below.

```
EncryptLoader Mode
DECRYPTED CLASS FILE:com/temenos/t24/TEST_cl_def.class
EncryptLoader Mode
DECRYPTED CLASS FILE:com/temenos/t24/TEST_cl.class
THIS IS OLD BASIC PROGRAM
```

## tComponentSplitter

### Introduction

The ComponentSplitter is used for componentization of class files under the directory which is specified in **'temn.tafj.directory.classes'** variable in the property file. The ComponentSplitter will iterate all .class files under this directory and find appropriate components for each class file. After getting the component name of each class file then jar files are created in the name of component and all .class files are added to appropriate jar according to its component.

The ComponentSplitter knows the component name for each class file by reading the **F.PGM.DATA.CONTROL** file.

If a class file doesn't have a component then that class file will be stored under a default jar file. So we can specify the default jar file name under 'temn.tafj.default.jar.name' variable in the properties file.

Note: Specify the jar file name alone without ".jar" extension in 'temn.tafj.default.jar.name' variable. If this variable hasn't been specified, then the jar file will be called "GENERAL" will be create as default jar.

**Syntax**

Execute tComponentSplitter.bat or tComponentSplitter from bin.

```
tComponentSplitter [—cf properties]
```

After creating jar files all class files will be deleted.  Jar files are kept under <TAFJ_HOME>/data folder. After creating all jar files then we can specify this directory in 'temn.tafj.directory.precompile' variable without specifying all jar files.

## tIntegrate:

Introduction

tIntegrate will iterate through all basic folders which is mentioned in 'temn.tafj.directory.basic' property from the configuration file and compile all basic files and creates two jar files.  One jar will contains all class files which it compiled from the basic files and another jar will contains all Insert files.

**Syntax**

Execute tIntegrate.bat or tIntegrate from bin.

```
tIntegrate [—cf properties] version
```

Version is a mandatory argument in which the jar will be named. The first jar file (ie which contains class files) will be named as version.jar and the second jar (ie which contains Insert files) will be named as I_Version.jar.

The resulting jar files will be placed in the directory which is specified in the property **'temn.tafj.eb.temp.release'** from the configuration file.

# TAFJ-Distribution

## tMerge

### Introduction

tMerge is part of the TAFJ tools. It allows merging your T24 precompiled jar files with a patch jar. It is recommended to have:

- Destination: precompiled T24 jar files in a T24_HOME\lib folder
- Source: precompiled T24 Patch jar files in a T24_HOME\patch folder

The tool will generate a patched T24 doing the following step:

1. Backing up your lib folder or the parent folder of the destination in a bkup folder in a zip file. The zip file name is composed: T24BKUP[Destination jar name].zip
2. Moving the lib folder to tmp folder
3. Merging patch classes in the correspondent destination jar.

### Syntax

The tMerge syntax is the following:

```
tMerge source destination
```

### Example

To patch a T24 Release

```
tMerge c:\Temenos\T24\patch\Patch.jar c:\Temenos\T24\lib\T24.jar
```

The resulting output looks like this:

```
―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――

tMerge will merge:
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\patch\Patch.jar with
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\lib\T24.jar

―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――

Backuping H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\lib\T24.jar

―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――

Merging H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\patch\Patch.jar in
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\lib\T24.jar

―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――

tMerge finish.


Log file :
[INFO] 2009-12-18 13:41:51,285 [main] RUNTIME  - Backuping
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\
[INFO] 2009-12-18 13:41:51,285 [main] RUNTIME  -
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\lib will be backuped with name
T24BKUP200912.004.jar.tar
[INFO] 2009-12-18 13:42:12,310 [main] RUNTIME  - Backup done.
[INFO] 2009-12-18 13:42:20,848 [main] RUNTIME  - Merging jar source :
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\patch\Patch.jar destination :
H:\dev\rtc\tafj\TAFJCore\test\ressources\t24\lib\200912.004.jar
[INFO] 2009-12-18 13:53:34,755 [main] RUNTIME  - Merge done
```