TFORMer SDK

Reporting and Labeling SDK

Version 9.0

Developer Manual

13 January 2026

TEC-IT Datenverarbeitung GmbH
Hans-Wagner-Strasse 6
A-4400 Steyr, Austria

t ++43 (0)7252 72720
f ++43 (0)7252 72720 77
office@tec-it.com
www.tec-it.com

1 Content

1.1 Table of Figures

2 Disclaimer

The actual version of this product (document) is available as is. TEC-IT declines all warranties which go beyond applicable rights. The licensee (or reader) bears all risks that might take place during the use of the system (the documentation). TEC-IT and its contractual partners cannot be penalized for direct and indirect damages or losses (this includes non-restrictive, damages through loss of revenues, constriction in the exercise of business, loss of business information or any kind of commercial loss), which is caused by use or inability to use the product (documentation), although the possibility of such damage was pointed out by TEC-IT.

TEC-IT Datenverarbeitung GmbH
Hans-Wagner-Str. 6

A-4400 Austria
t.: +43 (0)7252 72720
f.: +43 (0)7252 72720 77
www.tec-it.com

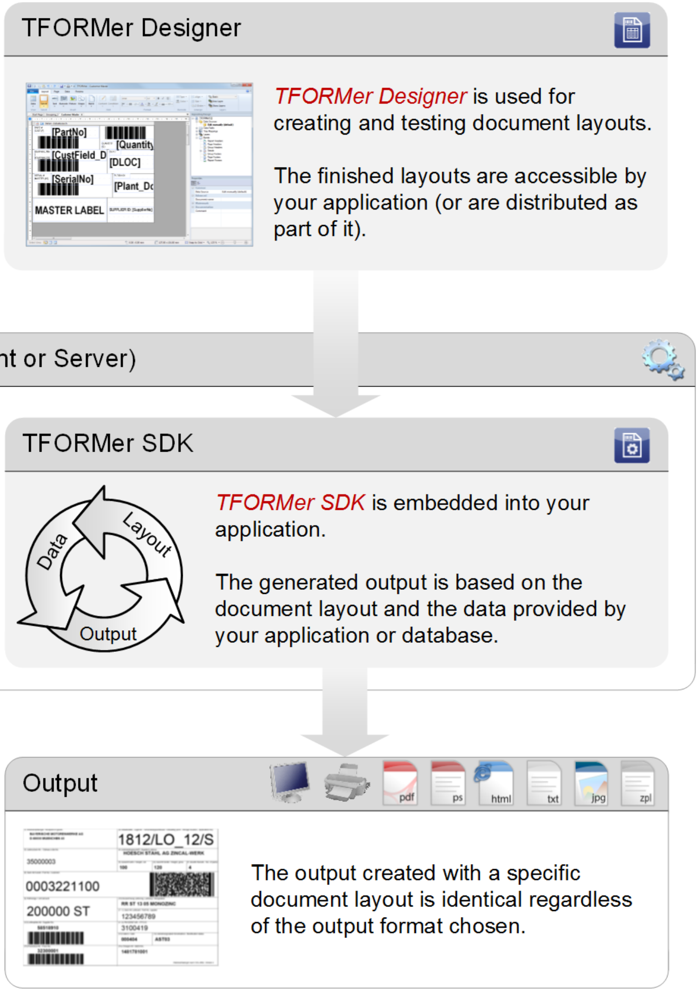
3 Introduction

3.1 TFORMer 9

The TFORMer product family represents a complete, lean and powerful solution for generating arbitrary documents. It combines the features of barcode labeling tools with the characteristics of report generators into a unified printing-solution. It provides *professional layout and output* capabilities, an integrated *barcode generator,* full-featured *UNICODE* support, output *streaming* and direct *PDF generation.*

3.2 Using the TFORMer SDK

TFORMer SDK provides the core output functionality of the TFORMer product family. It can be used on client and on server side and it is available for all major operating systems. It can be embedded in your own applications easily.



TFORMer Designer is used for creating and testing document layouts.

The finished layouts are accessible by your application (or are distributed as part of it).

TFORMer SDK is available for all platforms.

Available APIs: JAVA, .NET, COM, DLL or EXE.

TFORMer SDK is embedded into your application.

The generated output is based on the document layout and the data provided by your application or database.

The output created with a specific document layout is identical regardless of the output format chosen.

Figure 1: Using the TFORMer SDK

3.3 Areas of Application

Due to the support of multiple output formats TFORMer SDK has a wide field of applications:

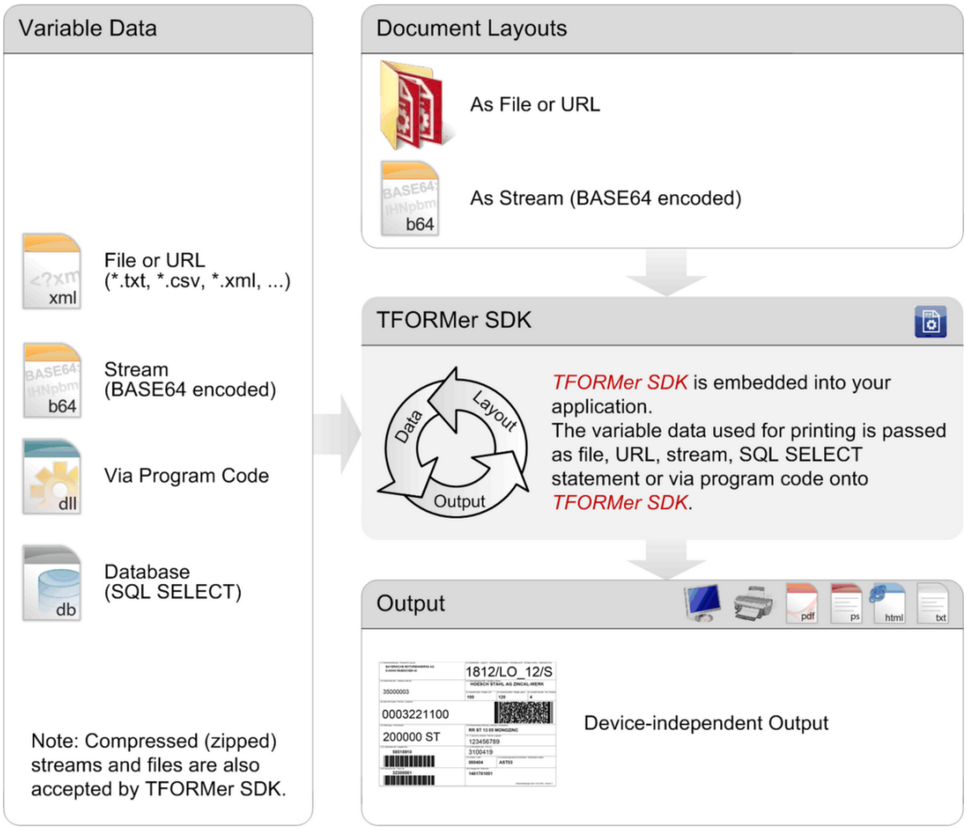
- **Reporting Engine**
  For software developers the combination of TFORMer SDK and TFORMer Designer provides a generic reporting solution which enables end-customers to create, edit and print forms and labels.
- **PDF Library**
  TFORMer SDK creates high-quality PDF documents based on graphical layouts.
- **Industrial Printing**
  The built-in barcode support satisfies almost all industry labeling requirements.
- **Client- or Server based Reporting**
  When it comes to web applications TFORMer SDK reduces round-trips and bandwidth requirements. Direct printing or PDF generation from within Microsoft® Internet Explorer is supported.
- **Pre-Press and Print-Shop Applications**
  Mass-mailings and serial letters are created and printed within minutes.

3.4 Generating Output

For generating output, the following data must be specified:

- The document layout
- Dynamic data for printing
- Output device or output file

For passing the layout and data to TFORMer SDK you can choose between various methods:



TFORMer SDK is embedded into your application.
The variable data used for printing is passed as file, URL, stream, SQL SELECT statement or via program code onto *TFORMer SDK*.

Note: Compressed (zipped) streams and files are also accepted by TFORMer SDK.

Figure 2: Generating output with the TFORMer SDK

A typical workflow is as follows:

- In TFORMer Designer:
  - Create a Template and save it
- In TFORMer SDK:
  - Select the created TFF file
  - Define a data source
  - Define an output type
  - Print

3.5 This Document

This document provides a high-level introduction to the TFORMer SDK, with regard to the supported operating systems and the available APIs (Application Programming Interfaces). The target audience are software developers and system administrators.

For more information about programming TFORMer SDK, please refer to chapter 5.

4 Highlights of TFORMer SDK

4.1 Unified Output Solution

TFORMer SDK unifies output tasks. A specific form layout produces identical output across printer models, operating systems and output formats. Supported output formats are:

- **Direct Printing**
  Print layouts directly on all printers supported by Microsoft® Windows.
- **PDF**
  PDF export with full-featured barcode support, Unicode and font embedding.
- **PostScript®**
  Used for printing under Linux/UNIX and for pre-press applications.
- **HTML**[1]
  The built-in HTML output is ideal for previewing and for web-based applications.
- **Image Formats**
  The built-in image output supports BMP, GIF, JPG, PCX, TGA, PNG and TIF formats (including multipage TIF).
- **ZEBRA®**
  Print to ZEBRA® printers without any additional driver. ZPL-II output is generated directly.
- **ASCII**
  Pure ASCII output without any graphics for special purpose requirements.

4.2 UNICODE Support

The integrated UNICODE support allows you to use TFORMer SDK with almost any established language.

4.3 Barcode Support

TFORMer SDK offers integrated support for all linear, 2D and composite barcodes in common use.

4.4 Streaming

Most of the supported output formats can be created as pure in-memory stream without temporary files on the file-system.

4.5 Reusable, Smart Form Layouts

TFORMer SDK separates form layouts from the data. It does not bind form layouts to certain printer models or databases. The output is dynamically controlled by means of layout properties, conditional printing, tray control and individual computations.

[1] Due to the nature of HTML some output features may be limited or not available.

4.6 Cross Platform

TFORMer SDK was designed to cover all operating systems in common use. It is available for Microsoft® Windows and for major Mac® OS/Linux®/UNIX® variants.

4.7 Small Footprint, High Performance

TFORMer SDK provides high performance PDF and printer output in combination with very low system requirements.

5 Documentation, API Reference, Samples

5.1 Overview

For an overview over installed APIs, available Documentation and Samples open the *Windows Start-Menu* and select *TEC-IT TFORMer 9 SDK* ► *Documentation*.

These are the most relevant documents and samples:

5.2 API References and Manuals

- **TFORMer SDK – DLL Programming Reference**
  This document describes the DLL API (Windows) as wells as the shared library API (Linux).
  See *Start Menu* ► *TEC-IT TFORMer 9 SDK* ► *DLL Programming Reference*
  Online: https://www.tec-it.com/Documentation/TFORMer9_Library_Reference/index.html

- **TFORMer SDK – COM Programming Reference**
  This document describes the COM programming interface.
  See *Start Menu* ► *TEC-IT TFORMer 9 SDK* ► *COM Programming Reference*

- **TFORMer SDK – .NET Programming Reference**
  This document describes the .NET programming interface.
  See *Start Menu* ► *TEC-IT TFORMer 9 SDK* ► *.NET Programming Reference*
  Online: https://www.tec-it.com/Documentation/TFORMer9_Assembly_Reference/index.html

- **TFORMer SDK – JAVA Programming Reference**
  This document describes the JAVA programming interface.
  See *Start Menu* ► *TEC-IT TFORMer 9 SDK* ► *JAVA Programming Reference*
  Online: https://www.tec-it.com/Documentation/TFORMer9_Java_Reference/index.html

- **TFORMer SDK – TFPrint Command Line Printing Tool**
  This document describes the TFPrint command line printing tool.
  See *Start Menu* ► *TEC-IT TFORMer 9 SDK* ► *Documentation* ► *TFORMer TFPrint User Manual*

- **TFORMer Designer User Manual**
  This document provides an in-depth documentation of the graphical layout editor.
  See TFORMer Designer: *Menu* ► *Help* (or press F1).

5.3 Samples

- **TFORMer SDK – Programming Samples**
  See *Start Menu* ► *TEC-IT TFORMer 9 SDK* ► *Programming*

5.4 Online FAQ

- Frequently asked questions (FAQs) and answers

6 Installation on Microsoft Windows

6.1 Introduction

This chapter describes the installation of TFORMer SDK on Microsoft® Windows.

TFORMer SDK is available for 32-bit and 64-bit Windows. The installer contains both versions and installs the correct version automatically. The 64-bit version additionally includes all 32-bit components (like the 32-bit TFORMer COM API, etc.).

TFORMer SDK ships with some .NET based components like a .NET assembly and .NET sample applications. If you are interested in these features, install the .NET Framework 4.5 (or higher) before installing TFORMer SDK 8.1+.

Please note:

- Even though .NET is supported, the TFORMer SDK core components do not rely on .NET.

6.2 General

The installation of TFORMer SDK is straight-forward, the setup application will install the TFORMer SDK components, sample applications and documentation.

The default installation path is one of the following:

```
C:\Program Files\TEC-IT\TFORMer9 SDK
C:\Program Files (x86)\TEC-IT\TFORMer9 SDK
```

Please note:

- On 32-bit Windows, only the 32-bit versions of the executables and DLLs are installed.
- On 64-bit Windows, the 64-bit and the 32-bit versions of the executables and DLLs are installed, both in the same directory.

6.3 Supported APIs

After installation the functionality of TFORMer SDK is available via the following APIs:

- **COM Component**
  This API is available via the file *TFORMerCOM9.dll*. The COM component is registered automatically by the setup application.
- **.NET Component**
  The file *TECIT.TFORMer.dll* provides the .NET functionality (.NET 4.5 or higher is required). The .NET assembly is automatically installed in the GAC as well as in the *Bin* directory.
- **DLL**
  TFormer9.dll is a 32-bit Windows DLL, *TFormer9x64.dll* is the 64 bit variant of it (this naming scheme applies to all DLLs as well). The corresponding library and include files are named *TFormer9.lib* and *TFormer9.h*. For Linux and UNIX please refer to chapter 7.
- **Command Line Application**
  The command-line based executable is named *tfprint.exe.* The command line application is also available for Linux and UNIX. For details, please refer to chapter 7.

- **JAVA**
  The JAVA API can be integrated in J2EE architectures (web applications or web services) and GUI applications (e.g. Swing, AWT). JAVA version 1.5 or higher is required.

6.4 Sample Applications

Depending on your Windows version the sample applications and layout templates are installed in one of the following directories (please note that Windows Explorer does not display the folder *ProgramData* by default, you need to enable *Show hidden files and folders* in the *Folder and Search options*):

```
C:\ProgramData\TEC-IT\TFORMer\9
```

or:

```
C:\Documents and Settings\All Users\Application Data\TEC-IT\TFORMer\9
```

7 Installation on Linux® or UNIX®

7.1 Introduction

This chapter describes the installation of TFORMer SDK on Linux® and UNIX® based operating systems. Depending on the operating system TFORMer SDK is available as *rpm* or *tarball*. For details, please refer to section 7.5.

For a list of Linux or UNIX distributions which are supported with ready-to-run binaries please check out www.tec-it.com. If you need binaries for your specific platform, please let us know. Whenever possible, we provide them on request. In some cases, we perform remote builds after signing a NDA.

► Please note that the report and label designer TFORMer Designer is only available for Microsoft Windows.

7.2 General

The default installation path of TFORMer SDK is

```
/usr/local/
```

The TFORMer SDK Command Line Application *tfprint* is installed into

```
/usr/local/bin/tfprint
```

All configuration files, the license file **TFORMer.ini** and sample applications are located in the following directory:

```
/usr/local/share/TFORMer9/
```

This directory contains the following subdirectories:

- *APIDocs*: API references
- *Copyright:* this folder holds license information for used 3rd party software
- *Demos:* contains sample shell scripts which demonstrate the use of *tfprint* (see chapter 13)
- *Examples:* contains folders with sample source code (see section 7.3)
- *Templates:* contains ready-to-use layout templates designed with TFORMer Designer

The directory also contains the Java library:

- *JTFORMer9.jar*

The include files needed for software development are stored in:

```
/usr/local/include/TFORMer9/
```

The developer documentation for C++, Java and.NET is located in the following directory. Navigate to the according sub-directory and open the file *index.html* with your browser.

```
/usr/local/share/TFORMer9/APIDocs
```

Additional documentation is available through a man page.

$ man tfprint

7.3 Supported APIs

After installation the functionality of TFORMer SDK is available via the following APIs:

- **Shared Library**
  named *libTFORMer9.so* or *libTFORMer9.a*. The interface of the shared library is 100% compatible with the DLL API on Microsoft® Windows.
- **JAVA API**
  the JAVA interface is available in the file *JTFORMer9.jar*.
- **.NET API**
  the .NET interface is available in the file *TECIT.TFORMer.dll and TECIT.TFORMer.Std.dll.*
- **Command Line Application**
  The command-line based executable is named *tfprint*

7.4 Sample Applications

The directory

```
/usr/local/share/TFORMer9/Examples
```

contains a sample repository in the directory

- *Demo Repository*
  The forms contained in this repository are used by sample applications.

Sample applications with source code are also included:

- *C Command Line*
  This sample is written in C (TFORMerSimpleX.c). It generates barcode labels as a PDF document.
- *CSharp .NET Command Line*
  This sample demonstrates the use of the C# DLL with .NET for printing business cards from the sample repository. Please note: This sample requires the *TECIT.TFORMer.dll.*
- *JAVA Command Line*
  This sample demonstrates the use of the JAVA API for printing business cards from the sample repository.
- *JAVA (J2EE)*
  This sample demonstrates the use of the Java API in a J2EE servlet. It generates barcode labels as a PDF document.

7.5 Platform Specific Installation Hints

7.5.1 Linux® (deb- or rpm-based)

7.5.1.1 Dependencies

Before installing TFORMer SDK make sure to check if the following packages are available:

```
fontconfig-2.2.9
freetype2-2.1.7
libxml2-2.6.7
libxslt-1.1.14
cups-1.1.20 (or later)
unixODBC-2.2.11 (optional; only required if SQL is used)
libart_lgpl_2-2.3.19 (optional; only required for image, HTML and ZPL-II output)
freeimage-3.15.4
```

7.5.1.2 Rpm Installation

Install the executables for TFORMer SDK (which also include some sample applications) with the following command (as root-user) within the shell:

```
$ su
$ sh TFORMer-9.0.0-1.i586.rpm.bin
```

After confirming the license agreement with *yes* this script installs the rpm. After the installation you will be asked if you want to generate a demo-report.

7.5.1.3 Rpm De-Installation

TFORMer SDK can be de-installed using the commands below:

```
$ su
$ rpm -e TFORMer
```

7.5.1.4 Deb (Debian) Installation

Install the executables for TFORMer SDK including some sample applications with the following command (as root-user) within the shell:

```
$ sudo su
$ sh TFORMer-9.0.0-1-i386.deb.bin
```

After confirming the license agreement with *yes* this script installs the rpm. After the installation you will be asked if you want to generate a demo-report.

7.5.1.5 Deb (Debian) De-Installation

TFORMer SDK can be de-installed using the commands below:

```
$ sudo su
$ dpkg -r TFORMer
```

7.5.2 HP-UX® (tarball)

7.5.2.1 Dependencies

Before installing TFORMer SDK make sure to check if the following packages are available:

```
freetype-2.1.10-ia64-11.23.depot
libxml2-2.6.23-ia64-11.23.depot
gettext-0.14.5-ia64-11.23.depot
libxslt-1.1.15-ia64-11.23.depot
expat-1.95.8-ia64-11.23.depot
lcms-1.15-ia64-11.23.depot.gz
zlib-1.2.3-ia64-11.23.depot
fontconfig-2.3.2-ia64-11.23.depot
libiconv-1.10-ia64-11.23.depot
libart_lgpl-2.3.21-ia64-11.31.depot (only required for image, HTML and ZPL-II output)
```

A good place for downloading precompiled packages is: http://hpux.connect.org.uk/

7.5.2.2 Installation

TFORMer SDK installation is straight forward. TFORMer is distributed as tarball which includes an install script.

```
$ gunzip SetupTFORMer-9.0.0-HPUX11.23-IA64.tar.gz
$ tar xf SetupTFORMer-9.0.0-HPUX11.23-IA64.tar
```

$ cd SetupTFORMer
$ sh install.sh

After confirming the license agreement with *yes* this script installs TFORMer. After the installation you will be asked if you want to generate a demo-report.

7.5.3 AIX® (tarball or rpm/bin based)

7.5.3.1 Dependencies

Before installing TFORMer SDK make sure to check if the following packages are available:

► Installation of the RPM package manager is a must!

AIX 4.3

```
rpm.rte.3.0.5.30
fontconfig-2.2.0-1.aix4.3.ppc.rpm
freetype2-2.1.5-1.aix4.3.ppc.rpm
libxml2-2.6.20-1.aix4.3.ppc.rpm
zlib-1.1.4-3.aix4.3.ppc.rpm
```

AIX 5.x, AIX 6.1

```
rpm.rte.3.0.5.30
fontconfig-2.2.2-3.aix5.1.ppc.rpm
freetype2-2.1.7-2.aix5.1.ppc.rpm
libxml2-2.6.21-4.aix5.2.ppc.rpm
zlib-1.2.2-4.aix5.1.ppc.rpm
libart_lgpl-2.3.17-4.aix5.1.ppc.rpm
libxslt-1.1.5-2.aix5.1.ppc.rpm
```

AIX 7.1

```
rpm.rte.3.0.5.30
libstdcplusplus-4.8.5-1.aix7.1.ppc.rpm
libgcc-4.8.5-1.aix7.1.ppc.rpm
libiconv-1.15-1.aix5.1.ppc.rpm
fontconfig-2.2.2-3.aix5.1.ppc.rpm
freetype2-2.1.7-2.aix5.1.ppc.rpm
zlib-1.2.2-4.aix5.1.ppc.rpm
libart_lgpl-2.3.17-4.aix5.1.ppc.rpm

# Please do not use newer versions of libxml2 since they are buggy
libxml2-2.7.8-1.aix6.1.ppc.rpm
libxslt-1.1.5-2.aix5.1.ppc.rpm
```

It is necessary to extract some libraries from their archives and create symbolic links:

```
cd /opt/freeware/lib
# extract shared library and create a link
ar -x libfreetype.a
ln -sf libfreetype.so.6 libfreetype.so

# extract shared library and create a link
ar -x libart_lgpl_2.a
ln -sf libart_lgpl_2.so.2 libart_lgpl_2.so
```

Good places for downloading precompiled packages are:

- http://www-03.ibm.com/servers/aix/products/aixos/linux/download.html
- ftp://ftp.software.ibm.com/aix/freeSoftware/aixtoolbox/RPMS/ppc
- http://www.oss4aix.org/download/RPMS/

7.5.3.2 Installation from Tarball

The installation is identical to HP-UX (see section 7.5.2.2).

7.5.3.3 Installation from Rpm/bin

Install the TFORMer SDK by executing the downloaded bin file as script. Run the following command (as root-user) within the shell:

$ su
$ sh tformer-9.0.0-AIX-7.1-power7.bin

After confirming the license agreement with *yes* this script installs the rpm. After the installation you will be asked if you want to generate a demo-report.

7.5.3.4 Rpm De-Installation

TFORMer SDK can be de-installed using the commands below:

$ su
$ rpm -e TFORMer

7.5.4 Solaris® (tarball)

7.5.4.1 Dependencies

Before installing TFORMer SDK make sure to check if the following packages are available:

```
freetype-2.3.1-sol10-x86-local.gz
fontconfig-2.4.2-sol10-x86-local.gz
```

7.5.4.2 Installation

To install TFORMer SDK for Solaris 10 perform the following steps:

```
gunzip TFORMer-9.0.0-solaris10.x86.tar.gz
cp TFORMer-9.0.0-solaris10.x86.tar /
cd /
tar xvf TFORMer-9.0.0-solaris10.x86.tar
```

TFORMer is now installed in */usr/local/bin/*

- The configuration file and documentation can be found in */usr/local/share/TFORMer9*
- You should now be able to print a demo PDF by executing:
  /usr/local/share/TFORMer9/Demos/DemoBarcodeLabels.sh

8 COM Component

8.1 Introduction

COM components (Component Object Model) are software components which can be integrated seamlessly into other software products or used with development environments like:

- Visual® Basic®, Visual C++, Visual Studio .NET, Visual Studio 2019 or higher, Borland C++ Builder, Borland Delphi, …
- Microsoft Office product suite (in conjunction with Visual Basic for Applications – VBA)
- HTML pages on client- or server-side (ASP, ASP .NET, VBScript, Internet Explorer, …)
- Command-line based scripting environments (Visual Basic Scripting Host)

► Please note: COM technology is only available on Microsoft Windows.

8.2 Outline

► The name of the type library is *TFORMer 9 Runtime Type Library*.

► The GUID is *7709C4E4-4641-4B08-8BA6-8649E3672C32.*

► The Class ID of the COM Control is *AEB782F4-D12D-485F-BF49-1D602BD74BF8*.

The general steps for using the COM API of TFORMer SDK are:

1. Embed the TFORMer COM component into your application or document. Usually this is done by establishing a reference to the "*TFORMer 9 Runtime Type Library*".
2. Create an instance of the "*TFORMer9Lib.TFORMer*" *COM class.*
3. Create a job-instance.
4. Create a jobdata instance and connect the job with it.
5. Set the properties of the job object (e.g. name of the form layout, output type and printer name).
6. Provide or import the values for data fields via the jobdata object.
7. Finally call the *Print* method.

The general VBA-code for using TFORMer SDK looks as follows:

```
' This code snippet demonstrates the basic steps for using the TFORMer SDK – COM component
' from within VBA. Take care to establish a reference to TFORMer 9 Runtime Type Library
' beforehand!

Private Sub TFORMer_Output()

   ' Declare the variables

   Dim TFormer
   Dim PrintJob
   Dim JobData As TFORMer9Lib.IJobDataRecordSet

   ' Create a TFORMer SDK instance

   Set TFormer = CreateObject("TFORMer9Lib.TFormer")

   ' Create a job object with TFORMer

   Set PrintJob = TFormer.CreateJob
   ' Create (and connect) the job data instance for providing data field values

   Set JobData = printjob.NewJobDataRecordSet

   ' Select the form layout to be printed/generated, select type and name of the output

   PrintJob.RepositoryName = "FILENAME_OF_YOUR_FORMLAYOUT.tff"
   PrintJob.OutputName = "C:\temp\output.pdf"
   PrintJob.PrinterType = TECIT.TFORMer.PrinterType.PdfFile

   ' Provide data for the data fields used in the form layout

   JobData.AddNewRecord
   JobData.SetDataField "NAME_OF_YOUR_DATAFIELD", "This is the value of the datafield"

   ' Start printing

   printjob.PrintForm
End Sub
```

8.3 Using the COM Object in Development Environments

8.3.1 Microsoft Visual Basic for Applications (VBA)

Please refer to chapter 14, TFORMer SDK with Microsoft Office.

8.3.2 Other Development Environments

Please refer to the documentation of your development environment.

8.4 More Information

For more information, please check out the following documents and sample applications (see also chapter 5):

- COM Programming Reference
  Start Menu ► TEC-IT TFORMer 9 SDK ► COM Programming Reference
- Sample Microsoft® Excel Spreadsheet using TFORMer SDK
  Start Menu TEC-IT TFORMer 9 SDK ► Microsoft Office

9 .NET Component

9.1 Introduction

The TFORMer SDK .NET component is perfectly suited for application development with any .NET programming language.

- On Microsoft Windows the setup application installs the TFORMer assembly automatically in the GAC (Global Assembly Cache) and in the *Bin* subdirectory of the TFORMer SDK default installation path. The .NET classes can be used immediately with development environments like Microsoft Visual Studio.
- Alternatively, the TFORMer assembly is also available as NuGet package: https://www.nuget.org/packages/TECIT.TFORMer

> ► On Microsoft® Windows the TFORMer SDK .NET component requires .NET 4.5 or higher.
>
> ► On Linux/UNIX TFORMer SDK for .NET works with .NET Core / .NET 5+.

9.2 Outline

> ► When establishing a reference, the name of the .NET component is *TFORMer 9 SDK* (file *TECIT.TFORMer.dll*).

The general steps for using the TFORMer SDK .NET assembly are:

1. Establish a reference to *TFORMer 9 SDK*.
2. Depending on the programming language include the *TECIT.TFORMer* namespace.
3. Create a job-instance (*TECIT.TFORMer.Job*).
4. Create a jobdata instance (*TECIT.TFORMer.JobData* or derived classes).
5. Connect the job with the jobdata.
6. Set the properties of the job to the required values (e.g. name of the form layout, output type and printer name).
7. Provide or import the values for data fields via the jobdata.
8. Finally generate the required output with the *Print* method.

The general code for using TFORMer SDK in C# .NET looks as follows:

```
// This code snippet demonstrates the basic steps for using the TFORMer SDK from within
// C# .NET. Take care to establish a reference to TFORMer 9 SDK from within your project!

using TECIT.TFORMer;

private void PrintWithTFORMer()
{

  // Create a new Job instance

  Job printJob = new Job();

  // Select the form layout (stand-alone or Repository-based)
  // If you are using a Repository you need to specify the Project (Job.ProjectName)
  // and the FormLayout (Job.FormName) as well.

  printJob.RepositoryName = "FILENAME_OF_YOUR_FORMLAYOUT.tff";
  // Provide the data for the data fields used in the form layout.
  // Here we use an in-memory data source for manually providing the data.

  JobDataRecordSet jobData = new JobDataRecordSet();


  Record record = new Record();
  record.Data.Add("NAME_OF_YOUR_DATAFIELD", "This is the value of the datafield");
  jobData.Records.Add(record);

  record = new Record();
  record.Data.Add("NAME_OF_YOUR_DATAFIELD", "This is a second datafield value");
  jobData.Records.Add(record);

  // Assign the JobData to the Job

  printJob.JobData = jobData;

  // Select the Windows or Linux (CUPS) printer name, use null for the default printer

  printJob.PrinterName = null;

  // Start printing

  printJob.Print();
}
```

9.3 Additional Samples

9.3.1 Print to PDF

```
  // Create a new Job instance

  Job printJob = new Job();

  // Select the form layout

  printJob.RepositoryName = "FILENAME_OF_YOUR_FORMLAYOUT.tff";

  // Provide the data for the data fields used in the form layout.
  // Here we import data from an ODBC connection using the specified SQL SELECT statement.

  JobDataOdbc jobData = new JobDataOdbc (
    "DSN=TFORMer_Sample",
    "", // Username
    "", // Password
    "SELECT * FROM tbl_Example"
  );

  // Assign the JobData to the Job

  printJob.JobData = jobData;

  // Select type and name of the output

  printJob.OutputName = "C\temp\output.pdf";
  printJob.PrinterType = PrinterType.PdfFile;

  // Print it

  printJob.Print ();
```

9.3.2 Save a PNG Thumbnail

```
  // Create a new Job instance

  Job printJob = new Job();
```

```
// Select the form layout

printJob.RepositoryName = "FILENAME_OF_YOUR_FORMLAYOUT.tff";

// Provide the data for the data fields used in the form layout.
// Here we import a CSV file and specify separator and qualifier characters.

JobDataCsv jobData = new JobDataCsv("/temp/Import.txt", ',', '"');

// Assign the JobData to the Job

printJob.JobData = jobData;

// Select type and name of the output

printJob.OutputName = "C:/temp/output.png";
printJob.PrinterType = PrinterType.ImagePng;

// Set the output resolution to 96 dpi (= screen resolution)

printJob.Options = "resolution=96";

// Print it

printJob.Print ();
```

9.4 More Information

For more information, please check out the following documents and sample applications (see also chapter 5):

- .NET Programming Reference
  Start Menu ► TEC-IT TFORMer 9 SDK ► .NET Programming Reference
- Sample Code
  Start Menu ► TEC-IT TFORMer 9 SDK ► Programming

10 DLL (Dynamic Link Library)

10.1 Introduction

Besides the COM and .NET Interface, the TFORMer SDK contains also a Microsoft® Windows DLL (32 bit/64 bit). A DLL is a library, which offers its functionality via a documented interface. DLLs can be used in

- Most programming languages and development environments (e.g.: Visual C/C++, .NET, all versions of Visual Studio, Visual Basic, Borland C++ Builder, Borland Delphi, …)
- Web-based scripting languages like PHP

► Please note: The DLL is only available on Microsoft® Windows. If you need to use a DLL-like interface on Linux/UNIX please use the shared library (see chapter 11).

10.2 Compiling Your Applications

The files *TFormer9.lib* (library) and *TFormer9.h* (header file) are required to compile and link your own applications with the TFORMer DLL. These files are usually stored in one of the following folders:

```
C:\Program Files\TEC-IT\TFORMer9\Bin
C:\Program Files (x86)\TEC-IT\TFORMer9\Bin
```

Make sure the compiler and the linker are able to locate these files.

10.3 Outline

The general steps for using the TFORMer DLL are:

1. Include *TFormer9.h*. Make sure to define *TECIT_DLLIMPORT* before including the header file.
2. Initialize the TFORMer DLL and save the handle *(TFormer_Init).*
3. Select the form layout to be printed *(TFormer_SetRepositoryName).*
4. Select the type of the output and the target file or device *(TFormer_SetPrinterName).*
5. Provide values for datafields (if any) in the form layout.
6. Generate or print the output *(TFormer_Print).*
7. De-initialize TFORMer SDK *(TFormer_Exit).*

The general C-code for using TFORMer DLL looks as follows:

```
// This is not a complete application. It is just a sample without any error handling
// to demonstrate the basic steps for using the DLL interface
//
// define TECIT_DLLIMPORT before including TFormer9.h!

#define  TECIT_DLLIMPORT
#include "PATH_TO_TFORMER/TFormer9.h"
#undef   TECIT_DLLIMPORT

// Handle for TFORMer SDK

HTFORM   hTForm  = NULL;
```

```
// Initialize TFORMer SDK

hTForm = TFormer_Init (NULL);

/* Select the form to be printed

TFormer_SetRepositoryName (hTForm, "FILENAME_OF_YOUR_FORMLAYOUT.tff");

/* Use the default printer of the system */

TFormer_SetPrinterName (hTForm, NULL, NULL);

/* Add a Data record */

TFormer_NewRecord (hTForm);

/* Set a Datafield value */

TFormer_SetVarValue (hTForm, "NAME_OF_YOUR_DATAFIELD", "Value of the datafield");

/* Start printing */

TFormer_Print (hTForm);

/* Exit TFORMer */

TFormer_Exit (hTForm);
```

10.4 More Information

For more information, please check out the following documents and sample applications (see also chapter 5):

- DLL Programming Reference
  Start Menu ► TEC-IT TFORMer 9 SDK ► DLL Programming Reference
- Sample applications with source code
  Start Menu ► TEC-IT TFORMer 9 SDK ► Programming
- Appendix B describes how data is provided to TFORMer SDK.
- Appendix C describes how to adjust general output options.

11 Shared Library

11.1 Introduction

A shared library on Linux or UNIX is the counterpart to a DLL on Microsoft® Windows. Such a library offers its functionality via a documented interface to applications. All programming languages on Linux/UNIX are able to use shared libraries.

► Please note: The shared library is only available on Linux/UNIX. If you need to use a similar interface on Microsoft® Windows, please use the DLL (see chapter 10).

11.2 Compiling Your Own Applications

The following files are required to build your own applications with the shared library:

- *TFormer9.h:* This is the include file for the TFORMer SDK shared library API.
- *TECITStd.h:* An additional include file for Linux/UNIX.
- *libTFORMer9.so:* This is the shared library.
- *libTFORMer9.a:* This is the static library version of TFORMer SDK.

After the shared library of TFORMer SDK has been installed the include files *TFormer9.h* and *TECITStd.h* are stored in:

```
/usr/local/include/
/usr/local/include/TECITSTd/
```

Make sure the compiler and the linker are able to locate these files. Compile and link your applications with a command similar to the following (see also the shell script *build_sample.sh):*

```
$ gcc TFORMerSimpleX.c -o TFORMerSimpleX -ldl –lTFORMer9 -L/usr/local/lib -I/usr/local/include
```

11.3 Outline

The general steps for using the shared or static library are:

1. Make sure to define *TECIT_DLLIMPORT* and *_TEC_UNIX* and *TEC_UNIX_BUILD.*
2. Depending on your platform define *TEC_LINUX, TEC_FREEBSD, TEC_AIX, TEC_HPUX, TEC_SOLARIS, TEC_OS400* or *TEC_SCO.* Only one platform is allowed!
3. Include *TECITStd.h* and *TFormer9.h.*
4. Initialize the TFORMer Shared Library and save the handle: *TFormer_Init ()*
5. Select the form layout to be printed: *TFormer_SetRepositoryName()*
6. Select the type of the output and the target file or device: *TFormer_SetPrinterName()*
7. Provide values for datafields (if any) in the form layout.
8. Generate or print the output: *TFormer_Print()*
9. De-initialize the TFORMer Library: *TFormer_Exit()*

The general C-code for using TFORMer SDK looks as follows:

```
// This is not a complete application. It is just a sample without any error handling
// to demonstrate the basic steps for using the DLL interface
//
// define TECIT_DLLIMPORT before including TFormer9.h!

#define TECIT_DLLIMPORT

/* Enable a TFORMer Unix build using the two defines below */

#define _TEC_UNIX
#define TEC_UNIX_BUILD

/* Set the define for the concrete OS you are using */

#define TEC_LINUX

/* #define TEC_FREEBSD */
/* #define TEC_AIX */
/* #define TEC_HPUX */
/* #define TEC_SOLARIS */
/* #define TEC_OS400 */
/* #define TEC_SCO */

/* include the TECITStd header for required defines */

#include <TECITStd/TECITStd.h>

/* include TFORMer9.h */

#include <TFormer9.h>

/* undefine - as they are only required for TFORMer Header files */

#undef TECIT_DLLIMPORT
#undef _TEC_UNIX
#undef TEC_UNIX_BUILD

// Handle for TFORMer SDK

HTFORM   hTForm   = NULL;

// Initialize TFORMer SDK

hTForm = TFormer_Init (NULL);

/* Select the form to be printed

TFormer_SetRepositoryName (hTForm, "FILENAME_OF_YOUR_FORMLAYOUT.tff");

/* Use the CUPS default printer of the system */

TFormer_SetPrinterName (hTForm, NULL, NULL);

/* Add a Data record */

TFormer_NewRecord (hTForm);

/* Set a Datafield value */

TFormer_SetVarValue (hTForm, "NAME_OF_YOUR_DATAFIELD", "Value of the datafield");

/* Start printing */

TFormer_Print (hTForm);

/* Exit TFORMer */

TFormer_Exit (hTForm);
```

11.4 C Sample Application

A sample application which uses the shared library is installed in:

```
/usr/local/share/TFORMer9/Examples/C Command Line
```

This C code generates a PDF file with barcode labels. The sample application uses the same document layout as the script DemoBarcodeLabels.sh mentioned in section 13.3.2.2. Datafield values are passed programmatically.

Compile it by executing:

$ gcc TFORMerSimpleX.c -o TFORMerSimpleX -ldl –lTFORMer9 -L/usr/local/lib -I/usr/local/include

11.5 More Information

For more information, please check out the following documents and sample applications:

- Programming References

  ```
  /usr/local/share/TFORMer9/APIDocs
  ```

- Sample Applications with source code (C, C#, Java)

  ```
  /usr/local/share/TFORMer9/Examples
  ```

- Appendix B describes how data is provided to TFORMer SDK.
- Appendix C describes how to adjust general output options.

12 Java Component

12.1 Introduction

The Java component of TFORMer SDK is a Java Native Interface (JNI) software layer between the TFORMer SDK (DLL/Shared Library) and the Java Virtual Machine. The architecture of this wrapper is composed by:

- The library *TFORMer9* (DLL or shared library), in order to execute the TFORMer SDK;
- The JNI library *TFORMer9JNI* (DLL or shared library), to translate the Java calls to the *TFORMer SDK* requests and responses;
- The Java library *TFormer9.jar,* to simplify the access to the TFORMer SDK.

In particular, the Java library is built following the .NET architecture in order to maintain a simply and uniform program interface. It's important that the paths of the TFORMer SDK – DLL (or shared library on UNIX platform) and the JNI library are included into the *java.library.path* of the Java Virtual Machine: so, you must ensure that your system respects one of the following cases: these libraries are

- defined into the environment variable *PATH* (Windows), *LD_LIBRARY_PATH* (Linux and Macintosh), *SHLIB* (HPUX) or *LIBPATH* (AIX);
- defined into the parameter *-Djava.library.path* of the java command;
- defined as system libraries.

As usual the Java library *(JTFormer9.jar)* must be included in the *classpath* of the *javac/java* command in order to compile and execute your projects.

12.2 Outline

The general steps for using the Java component of TFORMer SDK are:

1. Include the *com.tecit.TFORMer* and *com.tecit.TFORMer.Printing* namespaces.
2. Create a job-instance *(com.tecit.TFORMer.Printing.Job)*.
3. Create a jobdata instance (with the derived classes of *com.tecit.TFORMer.Printing.JobDataSource).*
4. Connect the job with the jobdata instance.
5. Use the setter methods of the job to define the required values (e.g. name of the form layout, output type and printer name).
6. Provide or import the values for data fields via the jobdata instance.
7. Generate the required output with the *print* method.
8. Finally free the resources allocated using the *dispose* method.

The general Java code for using TFORMer SDK looks as follows:

```
// This code snippet demonstrates the basic steps for using the TFORMer SDK from within
// Java. Take care to include the TFORMer Java library in your project and to include the
// TFORMer SDK and JNI DLL/shared library into the java.library.path

import com.tecit.TFORMer.*;
import com.tecit.TFORMer.Enumerations.*;
import com.tecit.TFORMer.Printing.JobDataRecordSet;
import com.tecit.TFORMer.Printing.Job;
import com.tecit.TFORMer.Printing.JobDataRecordSet.Record;

private void printWithJTFORMer()
throws TFormerException
{
  // Create objects for a job and a jobdata
  // NOTE: the constructor can generate an exception if the TFORMer SDK is not
  // accessible (java.library.path doesn't contain the TFORMer DLLs or shared libraries)

  Job job = new Job();
  JobDataRecordSet jobdata = new JobDataRecordSet();

  try {

    // Connect the jobdata with the job

    job.setJobData(jobdata);

    // Select the form layout to be printed/generated,
    // adjust the type and name of the output

    job.setRepositoryName("FILENAME_OF_YOUR_FORMLAYOUT.tff");
    job.setOutputName("C:\\temp\\output.pdf");
    job.setPrinterType(EPrinterType.PDFFile);

    // Provide data for the data fields used in the form layout

    Record record = new JobDataRecordSet.Record();
    record.setData("NAME_OF_YOUR_DATAFIELD", "This is the value of the datafield");
    jobdata.add(record);

    // Start printing

    job.print()
  }
  catch(TFormerException ex)
  {
    throw ex;
  }
  Finally
  {

    // ALWAYS call dispose in order to free the TFORMer SDK resources

    job.dispose();
    job = null;
  }
}
```

► Please note: In order to free the resources allocated by TFORMer SDK, it's important to call the method *dispose* of the JTFormer classes. A use of the TFORMer Java classes without the invocation of the method *dispose* can cause memory allocation problems and leaks.

12.3 More Information

For more information, please check out the following documents and sample applications (see also chapter 5):

- Java Programming Reference
  Start Menu ► TEC-IT TFORMer 9 SDK ► Java Programming Reference
- Sample Code
  Start Menu ► TEC-IT TFORMer 9 SDK ► Programming

13 Command Line Application

13.1 Introduction

The TFORMer SDK command line application is available for Microsoft® Windows, Linux® and UNIX®. It generates output based on arbitrary form layouts created with TFORMer Designer.

The executable is named *tfprint* (or *tfprint.exe* on Microsoft® Windows) and can be used

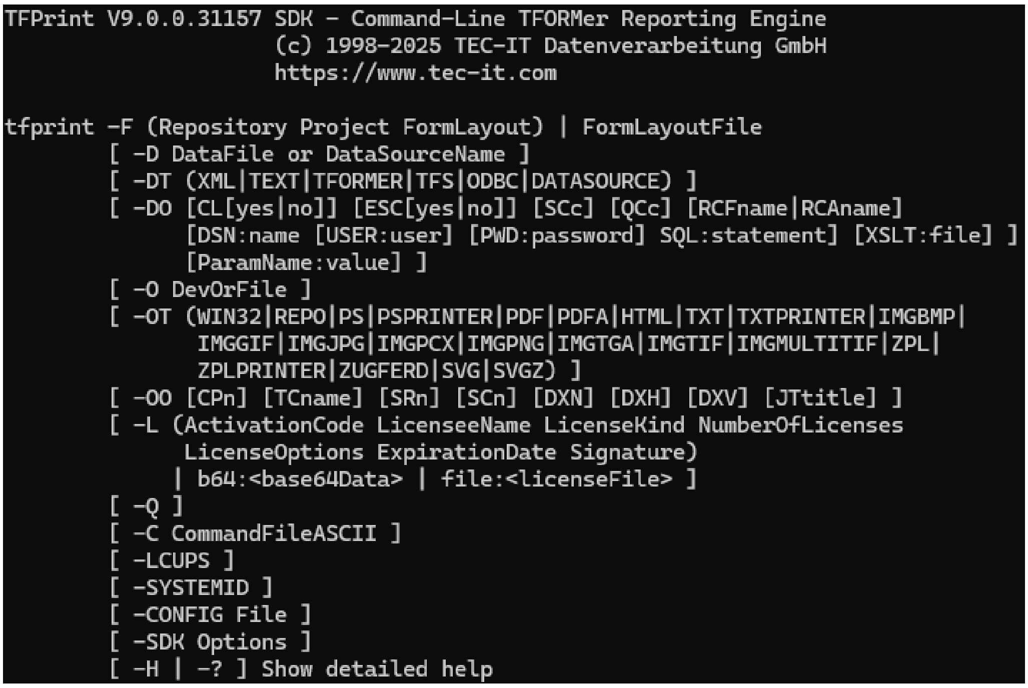
- on its own (executed manually)
- as part of batch jobs or shell scripts
- via “shell-execute” as “out-of-process” reporting engine
- as extension for printer filter applications on Linux/UNIX

13.2 Outline

The general steps for using *tfprint* are:

1. Create the required form layouts using TFORMer Designer.
2. Provide data for the form layouts via XML, CSV, TXT or ODBC datasources.
3. Embed *tfprint* into your batch-job, shell script or applications.

► On Microsoft® Windows the files for the form layouts or file-based datasources (XML, CSV or TXT) may be provided as a filename (as file stored in the file system) or as http-based URL.



```
TFPrint V9.0.0.31157 SDK - Command-Line TFORMer Reporting Engine
                     (c) 1998-2025 TEC-IT Datenverarbeitung GmbH
                     https://www.tec-it.com

tfprint -F (Repository Project FormLayout) | FormLayoutFile
        [ -D DataFile or DataSourceName ]
        [ -DT (XML|TEXT|TFORMER|TFS|ODBC|DATASOURCE) ]
        [ -DO [CL[yes|no]] [ESC[yes|no]] [SCc] [QCc] [RCFname|RCAname]
              [DSN:name [USER:user] [PWD:password] SQL:statement] [XSLT:file] ]
              [ParamName:value] ]
        [ -O DevOrFile ]
        [ -OT (WIN32|REPO|PS|PSPRINTER|PDF|PDFA|HTML|TXT|TXTPRINTER|IMGBMP|
               IMGGIF|IMGJPG|IMGPCX|IMGPNG|IMGTGA|IMGTIF|IMGMULTITIF|ZPL|
               ZPLPRINTER|ZUGFERD|SVG|SVGZ) ]
        [ -OO [CPn] [TCname] [SRn] [SCn] [DXN] [DXH] [DXV] [JTtitle] ]
        [ -L (ActivationCode LicenseeName LicenseKind NumberOfLicenses
              LicenseOptions ExpirationDate Signature)
             | b64:<base64Data> | file:<licenseFile> ]
        [ -Q ]
        [ -C CommandFileASCII ]
        [ -LCUPS ]
        [ -SYSTEMID ]
        [ -CONFIG File ]
        [ -SDK Options ]
        [ -H | -? ] Show detailed help
```

Figure 3: tfprint command line parameters

When using *tfprint* to generate output you need at least 3 parameter groups:

- the form layout (command line parameter *–F*)
- the datasource (command line parameters *–D, –DT, –DO*)
- the type and target of the generated output (command line parameters *–O, –OT, –OO*)

13.3 Samples

13.3.1 For Microsoft Windows

In the example below *tfprint* is used to generate a form layout (MyLayout.tff) using the datasource MyValues.txt (a datasource contains values for datafields) as a PDF document named Output.pdf:

```
tfprint -F MyLayout.tff  -D "MyValues.txt" -DT TEXT -DO SC, QC\" -O Output.pdf -OT PDF
```

Numerous sample applications are installed by the setup application – check out the following menu entry for details:

Start Menu ► TEC-IT TFORMer 9 SDK ► Command Line Printing TFPrint

13.3.2 For LINUX and UNIX

13.3.2.1 Generate all Templates as PDF

In order to generate all form layouts stored in the *Templates* directory as PDF-files execute the following script:

```
$ sh /usr/local/share/TFORMer9/Demos/DemoPDF.sh
```

The resulting PDF-files are created in your current working directory. The generated PDF files will need about 60 MB of disc space!

13.3.2.2 Generate Barcode-Labels (PDF-Output)

This script generates barcode labels as PDF-file:

```
$ sh /usr/local/share/TFORMer9/Demos/DemoBarcodeLabels.sh
```

The layout for the label is stored in the demo repository. The result is created in your current working directory.

13.4 More Information

For more information, please check out the following documents:

- Check out the TFPrint User Manual.
  Start Menu ► TEC-IT TFORMer 9 SDK ► Documentation ► TFORMer TFPrint User Manual
- Execute *tfprint –h* to display a detailed description of the command line parameters. On Linux/UNIX type *man tfprint* to display the man-pages.
- Check out the TFORMer Designer User Manual
  Start Menu ► TEC-IT TFORMer 9 ► TFORMer Designer Manual
- Appendix B describes how data is provided to TFORMer SDK.
- Appendix C describes how to adjust general output options.

14 TFORMer SDK with Microsoft Office

14.1 Introduction

Most applications of the Microsoft® Office Suite are supporting the Component Object Model (COM). That means that COM-compliant software components like the TFORMer SDK can be embedded and programmed directly as part of documents, spreadsheets, forms or databases.

The integration of COM components into the Microsoft® Office Suite works completely seamlessly. For example, the TFORMer SDK can be embedded into Microsoft® Access™ for printing barcode labels or generating PDF output.

Within the Microsoft® Office suite VBA (Visual Basic for Applications) is used as programming language. Thus, the functionality of the TFORMer SDK is also accessible via VBA.

14.2 Hints for Different Office-Versions

This section concentrates on the first steps for integrating the COM component of TFORMer SDK into a specific Microsoft® Office product.

- ► Depending on the application it may be necessary to switch to "design" or "edit" mode in order to create VBA program code.
- ► Macros must be activated/enabled (Word, Excel…).
- ► The security settings of the container application must be adjusted to allow active content or macros. Scripting has to be enabled.

14.2.1 Microsoft Word/Excel 2007-2024 and 365

1. Open or create your document.
2. Enable the developer tab in the ribbon:
   File ► *Options* ► *Customize Ribbon* ► enable *Developer* Tab
   (or *Office Button* ► *Word/Excel Options* ► *Show Developer tab in the Ribbon*)
3. Activate the Developer tab.
4. Click *Visual Basic* to open the Microsoft® Visual Basic editor.
   (Alternatively you can also use the shortcut *Alt+F11*).
5. Click *Tools* ► *References* and check *TFORMer 9 Runtime Type Library* to insert a reference to TFORMer SDK.
6. Click *OK.*
7. TFORMer SDK can now be used with VBA.

14.2.2 Microsoft Access 2007-2024 and 365

1. Open or create your database.
2. Edit or create a new form (or report) by clicking *Create* ►*Form Design* (or *Report Design).*
3. In the "Database Tools" (or in the "Report Design") tab click *View Code* to open the Microsoft® Visual Basic editor.
   (Alternatively you can also use the shortcut *Alt+F11*).
4. Click *Tools* ► *References* and check *TFORMer 9 Runtime Type Library* to insert a reference to TFORMer SDK.
5. Click *OK.*
6. TFORMer SDK can now be used with VBA.

14.2.3 Microsoft Word 2003 / Excel 2003

1. Open or create your document.
2. Activate the *Visual Basic-Editor*. This can be done by pressing the keyboard shortcut *Alt+F11* or by enabling the Visual Basic toolbar and clicking onto the Visual Basic-Editor button.
3. In the Visual Basic editor click *Tools* ► *References* and check *TFORMer 9 Runtime Type Library* to insert a reference to TFORMer SDK.
4. Click *OK.*
5. TFORMer SDK may now be used with VBA.

14.2.4 Microsoft Access 2003

1. Open or create your database.
2. Create a form or report by selecting *Insert* ► *Form* (*Report*) in the menu.
3. An empty form (or report) is opened.
4. Make sure the Design view is activated. This can be done by selecting *View* ► *Design* in the menu.
5. Open the Visual Basic-Editor by selecting *View* ► *Code* in the menu or by clicking the *View Code* symbol in the Form Design toolbar.
6. In the Visual Basic editor click *Tools* ► *References* and check *TFORMer 9 Runtime Type Library* to insert a reference to TFORMer SDK.
7. Click *OK.*
8. TFORMer SDK may now be used with VBA.

14.2.5 Other Microsoft Office versions

The required steps are similar to the steps outlined for Microsoft® Office 2003.

14.3 More Information

For more information, please check out the following documents and sample applications:

- COM Programming Reference
  Start Menu ► *TEC-IT TFORMer 9 SDK* ► *COM Programming Reference*
- Sample Microsoft Excel Spreadsheet using TFORMer SDK
  Start Menu ► *TEC-IT TFORMer 9 SDK* ► *Microsoft Office*

15 TFORMer in Web Applications

15.1 TFORMer SDK with Web-Applications on Windows

15.1.1 TFORMer SDK on a Windows-based Web-Server

TFORMer SDK at server-side can be used as part of web applications or web services. Server-side use does not bind your clients to specific browsers or operating systems.

Some .NET based sample applications demonstrate the use of TFORMer SDK with server-based web applications. Check out the following menu entry:

Start Menu ► TEC-IT TFORMer 9 SDK ► Programming

15.1.2 Web Based Form Layouts

TFORMer SDK is able to access form layouts, repositories or data files via http/https. When using *tfprint* this looks like *tfprint -F "http://something.com/OnlineReport_1.tff" …*

15.1.3 Client-side use of TFORMer SDK

Deprecated: Since MSIE is at its end of life and other browsers stopped support for NPAPI, this section is for reference only.

When printing from within a web browser the user usually has to confirm the print job by selecting a target printer (this is due to browser implementation details). You can avoid this limitation by using the COM component of TFORMer SDK. It enables your web-based application to print to arbitrary printers without any user interaction.

These are the steps to use the COM based API of TFORMer SDK within web pages on client-side:

1. The TFORMer SDK COM component has to be installed on each client. Take care that only Microsoft® Internet Explorer can embed the COM component directly. A CAB file for automatic installation is available from TEC-IT.
2. Use VBScript® or JavaScript™ in a web-page which instantiates a TFORMer class (a sample using VBScript is installed by the setup). Check out the following menu entry: *Start Menu ► TEC-IT TFORMer 9 SDK ► Programming*
3. Provide the form layouts either locally on the client, on a shared network folder, as BASE64 strings embedded directly into the HTML page or via http:// on an arbitrary server (see 15.1.2).
4. The web page provides data to the TFORMer SDK COM object programmatically.
5. Your web application prints to local printers without any extra user confirmation.

15.2 TFORMer SDK on Linux or UNIX Servers

The TFORMer SDK can be incorporated easily into server-based applications. Your server-based application (e.g. written in PHP, CGI-Perl, Java, C/C++, …) invokes *tfprint* as external process to generate the output data (e.g. a PDF file).

The TFORMer SDK for Linux/UNIX includes a ready-to-use example showing how *tfprint* can be integrated in a web application. The sample application is written in PHP and is installed in:

```
/usr/local/share/TFORMer9/SampleCodeCGI
```

This sample requires a configured apache web server which supports PHP5. To deploy the application, all files in the application directory have to be copied into the root directory of the web server.

The file feedback.php contains the application logic to supply user data using XML to *tfprint* and to create a PDF file from the user data and a form layout file.

15.3 TFORMer SDK on Java Web-Applications

Through the Java component of TFORMer SDK, it's possible to build J2EE printing applications and deploy them into the J2EE Application Server (i.e. Tomcat and JBoss).

This solution permits to print files on server-side, based on certain user configurations or data.

It's important to remember that the Application Server must read the TFORMer DLLs or shared libraries in order to access to the TFORMer SDK. For details see chapter 11, Java Component.

Check out the following menu entry to find simple examples of JSP/Servlet using the Java component of TFORMer SDK:

Start Menu ► TEC-IT TFORMer 9 SDK ►Programming

16 FAQ

16.1 Linux/UNIX

16.1.1 TFORMer does not work

If there are any problems with the installation of TFORMer SDK we recommend executing the following script:

$ /usr/local/share/TFORMer9/support.sh

This script is installed by the setup application and lists all relevant information like installed operating system, library versions and the TFORMer SDK version.

Contact the support of TEC-IT (support@tec-it.com) with the output of this script along with a detailed error description.

16.1.2 TFORMer SDK does not print texts (32 bit TFORMer on 64-bit systems)

The output generated by the TFORMer SDK only contains images, lines, rectangles or similar elements. Text elements are not shown.

Most likely this problem occurs if you are using a 32-bit build of TFORMer on a 64-bit system. Please make sure *iconv/gconv* (character set conversion) is installed correctly. (Re-)Install *glibc-32bit* using *yast* to solve the problem.

16.1.3 TFORMer SDK does not print Umlauts (e.g. ÄÖÜäöüß)

If you are using a text file for data import (e.g. with *tfprint*) and if this file was generated on Windows, TFORMer is not able to print special characters like Ä or Ü.

To avoid this problem, use XML import files (this is the recommended way). Another possibility is to convert the text file with *iconv* to UTF-8 (TFORMer SDK uses UTF-8 internally on Linux/Unix platforms):

$ iconv –f Windows-1252 –t UTF-8 –o output.csv input.csv

16.1.4 Fonts are looking strange / Errors in the generated layout

When generating output TFORMer tries to match fonts available on your system with fonts used in the form layout.

Most likely the form layout uses Windows-based fonts (like Arial) which are not available on your box. If an exact font matching is not possible the output may look strange. In order to avoid this behavior, install the required fonts on your Linux/UNIX machine.

Use the following commands which are part of the fontconfig package to install new fonts:

- *fc-list:* list all of the fonts currently available on your box.
- *fc-cache:* add additional fonts.

16.1.4.1 Installing True Type fonts

Simply copy the required fonts (*.ttf) to the directory */usr/share/fonts/local/* or to the *"/.fonts"* sub directory of the user's home (e.g. */home/user/.fonts*).

Then run *fc-cache*. The fonts are always installed for the user, who runs the fc-cache command. Make sure this is the same user, which uses the TFORMer SDK.

Check out the man pages of your system for an exact description of the fontconfig commands.

16.1.5 The Library libTFORMer9 cannot be found

Make sure the shared library *libTFORMer9.a* can be found by the linker. Usually this library is installed in

```
/usr/local/lib/
```

Some platforms require that you add this path to your linker options (e.g. *–L/usr/local/lib*).

16.1.6 TFORMer Include Files are not found

Make sure the TFORMer include files can be found by the compiler. Usually, these files are installed in

```
/usr/local/include/
```

Some platforms require that you add this path to your compiler options (e.g*. –I/usr/local/include*).

16.1.7 ZLib was not found on HP-UX

When creating PDF files, the following warning is displayed:

```
Warning: ZLib was not found. Compression support will be disabled.
```

When you see this warning message an uncompressed PDF file will be created.

On HP-UX make sure that the directory containing *libz.so* is in your *SHLIB_PATH* environment variable.

$ export SHLIB_PATH=/usr/local/lib/hpux32

16.1.8 The EURO sign (€) does not work on AIX

This is a known bug of the TFORMer SDK on AIX. Currently there is no solution available.

17 Contact and Support Information

17.1 Free Support

If you have questions, need help or simply want to tell us about your application, contact:

Email:	support@tec-it.com
Web:	www.tec-it.com/support

17.2 How to Unlock the Demo Version

You can unlock the demo version with a license key. License keys can be obtained from TEC-IT by email, online order form or fax.

Email:	sales@tec-it.com
Online:	www.tec-it.com/order
Fax:	+43 / (0)7252 / 72 72 0 – 77

17.3 Your Feedback is Welcome!

Don’t hesitate to contact us – let us hear your feedback! If the product does not fulfill your requirements, please tell us why. We are highly interested in meeting the requirements of our customers.

17.4 Company Contact Information

TEC-IT Datenverarbeitung GmbH

Address: Hans-Wagner-Str. 6
AT-4400 Steyr
Austria/Europe

Phone: +43 / (0)7252 / 72 72 0
Fax: +43 / (0)7252 / 72 72 0 – 77
Email: support@tec-it.com
Web: www.tec-it.com

Appendix A: Terms and Definitions

A.1 Form Layout

A form layout is a document created with TFORMer Designer. It is basically an XML-file, which stores all layout information with the corresponding parameters like position, font-size, color …

A form layout usually contains static and dynamic data. Dynamic data is provided by a datasource and used via datafields in the form layout.

A form layout is either stored in a repository (*.tfr) or stored as a stand-alone form layout (*.tff).

A.2 Stand-Alone Form

A stand-alone form is a document, which is stored as single file and which is not contained in a repository. TFORMer Designer normally creates stand-alone forms, but it can be instructed to add forms to a repository. Stand-alone forms are using the file extension *.tff.

All required information of a stand-alone form is contained in the layout – therefore it can be printed by its own.

Stand-alone form layouts are the most basic type of layout definitions and are very easy to use.

A.3 Repository

A repository is a central location in the file system for storing form layouts. Furthermore, the repository contains datasources and datafields, which may be defined once for multiple layouts.

The main file of a repository is using the file extension *.tfr. The subdirectories are named like the projects – and these subdirectories are holding the single form layouts. Therefore, moving or copying a repository must always include the subdirectories!

Using a repository simplifies maintenance of multiple form layouts which share the same datafields. In addition, a repository is a must with TFORMer Server.

When printing a repository-based form the user must specify the name and path of the repository, the project within the repository and the name of the form layout.

A.4 Project

A repository is divided into so-called projects. Each project holds definitions for datasources, datafields and one or more form layouts. All form layouts using the same datafields and/or datasources should be grouped in a project.

A.5 Datafield

A form layout may use datafields as place-holders for dynamic data. The content of these placeholders is provided during runtime (= print-time) by external datasources or by the user. At runtime each occurrence of a datafield is replaced with the current data. Datafields may be used as part of expressions.

In contrast to a stand-alone form which stores all datafield definitions itself, a repository-based form may use datafields defined in the project where the document layout is located. That means that all forms belonging to a project can use the same datafields.

In addition to such project-specific datafields TFORMer supports global datafields. These datafields are available for all document layouts in all projects.

A.6 Template

A template is a document layout, which has been stored in the template database of TFORMer Designer. If you select “New Form” in TFORMer Designer you can start your actual document layout based on such a template.

TFORMer Designer is shipped with various predefined templates like industry compliant labels. The user can also create his own templates by saving them to the template database.

Templates are stored per default in the following directory:

```
C:\ProgramData\TEC-IT\TFORMer\9\Templates
```

This directory can be changed by modifying the registry (see TFORMer Designer User Manual).

► Do not edit the files in the template path directly!

A.7 Datasource

Dynamic data is provided by external datasources like in-memory datasources, ODBC databases, text files, XML files or user input. Whenever printing a form, TFORMer Designer imports the data from the datasource and generates the output based on this data. The same data set may be used for the next output session without importing it again (the only exception is an in-memory datasource. In this case data is not stored by TFORMer).

Appendix B: Passing Data to TFORMer SDK

B.1 General

During runtime your application supplies data to TFORMer SDK in order to provide values for the datafields used in a form layout.

For this purpose, three general methods are supported:

- Imported data file
  Pass data to TFORMer SDK via a flat text file (e.g. CSV) or via XML files. Take care to create the files with current data before printing the required form layouts.
- User-defined datasource
  The creator of a form layout can also specify a datasource (XML file, TXT/CSV-file, ODBC queries) in the document layout itself. A unique name is assigned to each data source. When printing a form layout just specify the name of the datasource and TFORMer SDK queries the data for printing automatically.
- ODBC
  Pass data to TFORMer SDK via an ODBC database. DSN, user, password and the suitable SQL select statement can be specified as required.
- API (Application Programming Interface)
  Pass data to TFORMer SDK via method or function calls. This method is not available for the command line interface (tfprint).

B.2 XML Files

Three different XML formats can be imported:

- The internal XML-format.
- A generic XML-format.
- The XML-format used in current versions of Microsoft® Access™.

B.2.1 Internal TFORMer XML-Format

The XML format used internally by TFORMer must be available exactly as specified below:

```
<PSData>
  <Form>
    <Records>
      <Record Copies="1">
        <V Name="DataFieldName1">Value</V>
        <V Name="DataFieldName2">Value</V>
        <V Name="DataFieldName3">Value</V>
      </Record>
      <Record Copies="1">
        <V Name="DataFieldName1">Value</V>
        <V Name="DataFieldName2">Value</V>
        <V Name="DataFieldName3">Value</V>
      </Record>
    </Records>
  </Form>
</PSData>
```

Description:

Name	Description
PSData	This is the root element. It encloses all data within the xml file.
Form	This element encloses all data records for one Form. It must be specified only once in an XML-file

Records	This element encloses all data records to be printed with the enclosing Form. Must be specified only once in an XML-file.
Record	This element encloses all datafields for one record. For each record there is one (optional) attribute named “Copies”. The default value (if the attribute is missing) is 1 – meaning that this record is used once. If you want to use the same data for multiple records you can specify the number of record copies in this attribute.
V	Data-Element. This element consists of the datafield name (attribute “Name”) and the actual datafield value.
DataFieldName	Name of the datafield (must be unique within a record).
Value	The actual value of the datafield.

B.2.2 Generic XML-Format

The generic XML file allows two variants (see below).

B.2.2.1 Variant 1

```
<Root>
  <Records>
    <Record>
      <V Name="DataFieldName1">Value</V>
      <V Name="DataFieldName2">Value</V>
      <V Name="DataFieldName3">Value</V>
    </Record>
    <Record>
      <V Name="DataFieldName1">Value</V>
      <V Name="DataFieldName2">Value</V>
      <V Name="DataFieldName3">Value</V>
    </Record>
    :
    Further Data Records
    :
  </Records>
</Root>
```

Description:

Name	Description
Root	This is the root element. It encloses all data within the xml file. The element must not necessarily be named “Root”. You can choose a name freely.
Records	This element encloses all data records. The element must not necessarily be named “Records”. You can choose a name freely.
Record	This element describes one data record. Within a record one value can be specified for each datafield. The element must not necessarily be named “Record”. You can choose a name freely.
V	Data-Element. This element consists of the datafield name (attribute “Name”) and the actual datafield value.
DataFieldName	Name of the datafield (must be unique within a record).
Value	The actual value of the datafield.

► The nodes must not necessarily be named “Root“, “Records“ and “Record“. The names can be chosen arbitrarily. Only the order of the nodes is decisive. TFORMer interprets the lowest level of the XML-format as variable definitions and/or datafields. The superordinate nodes always correspond to data records.

Example:

```
<Root>
  <Records>
```

```
    <Record>
      <V Name="ArticleName">Chair</V>
      <V Name="ArticleNo">558963</V>
      <V Name="ArticlePrice">110</V>
    </Record>
    <Record>
      <V Name="ArticleName">Desk</V>
      <V Name="ArticleNo">778920</V>
      <V Name="ArticlePrice">150</V>
    </Record>
    :
    Further Data Records
    :
  </Records>
</Root>
```

B.2.2.2 Variant 2

Same as variant 1, but instead of using the syntax “<V Name="DataFieldName">Value</V>” you can specify datafield values using the following syntax:

```
<Root>
  <Records>
    <Record>
      <DataFieldName1>Value</DataFieldName1>
      <DataFieldName2>Value</DataFieldName2>
      <DataFieldName3>Value</DataFieldName3>
    </Record>
    <Record>
      <DataFieldName1>Value</DataFieldName1>
      <DataFieldName2>Value</DataFieldName2>
      <DataFieldName3>Value</DataFieldName3>
    </Record>
    :
    Further Data Records
    :
  </Records>
</Root>
```

B.2.3 Microsoft Access XML-Format

The XML files which are exported from Microsoft® Access™ looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<dataroot xmlns:od="urn:schemas-microsoft-com:officedata">
  <tblTableName>
    <DataFieldName1>Value</DataFieldName1>
    <DataFieldName2>Value</DataFieldName2>
    <DataFieldName3>Value</DataFieldName3>
  </tblTableName>
  <tblTableName>
    <DataFieldName1>Value</DataFieldName1>
    <DataFieldName2>Value</DataFieldName2>
    <DataFieldName3>Value</DataFieldName3>
  </tblTableName>
  :
  Further Data Records
  :
</dataroot>
```

B.3 Text Files

TFORMer can import text files in the CSV format (Comma Separated Values) or modifications of this format. When importing text files you can specify the separator (semicolon, comma, tab, space) and the text qualifier (= the symbol which encloses the actual values like text strings).

B.3.1 File Format

Text files must be structured as follows:

- The first line in the text file lists all column names (= names of the datafields).
- The following lines contain the datafield values (one record per line).

```
DataFieldName1;DataFieldName2;..;DataFieldNameN
Value;Value;..;Value
Value;Value;..;Value
:
Further Data Records
:
```

Example:

```
ArticleName;ArticleNo;ArticlePrice
Chair;558963;110
Desk;778920;150
Monitor;775116;236
Panel;544593;40
Coffee Machine;549896;30
Printer;458862;100
Fax;445866;115
Phone;458932;50
```

You have the possibility to use different *column separators, line separators* and *text qualifiers:*

B.3.1.1 Valid Column Separators

Char	Description	Example
TAB	Tabulator character (\0x09)	Value Value Value
;	Semicolon (\0x3B)	Value;Value;Value
,	Comma (\0x2C)	Value,Value,Value
Space	Space character (\0x20)	Value Value Value
Other	Any other single character.	Value#Value#Value

B.3.1.2 Valid Line Separators

Char	Description	Example
LF	Line Feed (\0x0A)	Text files created under UNIX or LINUX.
CRLF	Carriage Return + Line Feed (\0x0D\0x0A)	Text files created under Windows (or MS DOS).

B.3.1.3 Valid Text Qualifiers

Char	Description	Example
“	Double quote character (\0x22)	“Value”
‘	Single quote character (\0x27)	‘Value’
Other	Any other single character.	+ Value+
	None	Value

All characters which are enclosed with a text qualifier are imported as one single value. Thus, you can use the column separator character in strings, too!

If you want to use a line break within a string, please use the escape sequence “\n” (new line). When importing the data, the “\n” will be replaced with a new line character.

B.4 Important Notes

B.4.1.1 Empty or Missing Datafield Values

TFORMer SDK uses the default value of a datafield if no value is defined in the datasource. This behavior can be adjusted using with tfprint command line parameters (see command line switch *CL[yes|no]*).

B.4.1.2 Additional Datafields in an Import File

TFORMer SDK ignores datafields which are not defined in the form design.

B.4.1.3 Use of Escape Sequences (e.g. \n - Newline) in the Datafield Values

Since TFORMer Version 6 escape sequences are not translated automatically. This behavior can be adjusted via the tfprint command line (see *ESC[yes|no]*) or via the API.

Appendix C: Configuration File TFORMer.xml

C.1 General

The TFORMer SDK uses a configuration file which holds basic output settings suitable for most requirements. This configuration file is named *TFORMer.xml* and is installed automatically. After installation it can be adjusted to meet customized output needs.

C.2 Location of TFORMer.xml on Microsoft Windows

When using a Windows operating system, the default *TFORMer.xml* resides in the directory:

```
C:\ProgramData\TEC-IT\TFORMer\9
```

In addition, TFORMer supports user specific files which override the default. TFORMer Designer creates this file automatically in a user specific folder:

```
C:\Users\**USERNAME**\AppData\Local\TEC-IT\TFORMer\9
```

Via an API call the developer may also specify the filename of a specific configuration file which should be used.

C.3 Location of TFORMer.xml on LINUX or UNIX

When using Linux or UNIX this file is stored in the following directory:

```
/usr/local/share/TFORMer9
```

C.4 Configuration File TFORMer.xml

The following settings can be configured:

- Common options
  - Error handling for barcodes, images and text-elements with expressions
  - Settings for downloading pictures from an URL for printing
- PDF output options
  - page setup (size, orientation, compression, margins, embedding of fonts)
  - image compression method (and jpeg-quality)
  - maximum resolution for images (down-sampling)
- PostScript output options
  - page setup (size, orientation, color-mode)
  - external header/footer file
  - character encoding
- HTML output options
  - page setup (size, resolution)
  - barcode output options (optimize for readability, print unreadable barcodes)
- ZPL-II (ZEBRA printers) output options
- Barcode generator options
  - drawing method for windows printers
- Paper format names (for trays)

File dump of the default *TFORMer.xml* configuration file:

```
<?xml version="1.0" encoding="UTF-8"?>
<TFORMer major="1" minor="0">
  <!-- (c) 1998-2025 TEC-IT Datenverarbeitung GmbH                                -->
  <!-- htpp://www.tec-it.com                                                      -->
  <!-- support@tec-it.com                                                         -->

  <!-- This TFORMer configuration file contains standard settings for the         -->
  <!-- output methods. Edit this file if you want to change specific output       -->
  <!-- options or if you need to add custom media sizes.                          -->
  <!-- Note: In previous versions the term <Form> was used instead of <Media>     -->


  <!-- COMMON Options - Configure settings that apply to all printers             -->
  <COMMON>

  <!-- ErrorHandling                                                              -->
  <!-- Decides what TFORMer does if a barcode, image or expression error occurs   -->
  <!-- during printing. All 3 attributes accept the following arguments:          -->
  <!-- * abort          Abort printing if an error occurs                         -->
  <!-- * ignore         Ignore the error and continue printing                    -->
  <!-- * print          Print error information instead of the component that     -->
  <!--                  caused the error                                          -->
    <ErrorHandling barcode-error="print" image-error="print"
                   expression-error="print" unicode-font-error="print"
                   mediasource-error="abort" />

  <!-- URLPictures                                                                -->
  <!-- Specifies how TFORMer handles requests for downloading pictures via URL.   -->
  <!-- URLPictures are supported on Windows systems only.                         -->
  <!-- enabled             "0|1" enable/disable downloading of images             -->
  <!-- size-max            the maximum bytes per picture allowed. "0" = unlimited -->
  <!-- timeout             the download time in [ms] allowed per picture.         -->
  <!--                     "0" = no timeout                                       -->
    <URLPictures enabled="1" size-max="0" timeout="0" />

  <!-- Hyphenation                                                                -->
  <!-- soft-hyphens        characters which should be treated as soft hyphens     -->
  <!--                     (additional to &shy; and &#173;)                       -->
  <!--                     A soft hyphen marks a point at which a word may be     -->
  <!--                     divided. If the word is divided, then a hyphen is      -->
  <!--                     displayed at the end of the line, otherwise the hyphen -->
  <!--                     is not displayed.                                      -->
  <!--                     Sample: "%&#174;" to add 2 additional softhyphens      -->
  <!-- hyphen              value which should be used to render visible           -->
  <!--                     soft hyphens                                           -->
    <Hyphenation soft-hyphens="" hyphen="-" />

  <!-- TextFile                                                                   -->
  <!-- encoding            default encoding for CSV data sources using the        -->
  <!--                     standard text encoding and for importing text files    -->
  <!--                     via the SDK. Used when no file BOM is present.         -->
  <!--                     Supported values are:                                  -->
  <!--                     - "default"                                            -->
  <!--                     - "system", ansi(for Windows), utf-8(on other OS)      -->
  <!--                     - "ansi"                                               -->
  <!--                     - "utf-8"                                              -->
  <!--                     - "utf-16"                                             -->
  <!--                     - "utf-16be"                                           -->
    <TextFile encoding="default" />

  </COMMON>

  <!-- PDF OPTIONS - Configure settings for PDF output                            -->
  <!-- media               Default media size, must be listed in <DefinedMedia>   -->
  <!-- orientation         Default orientation "landscape|portrait"               -->
  <!-- compression         Enable/disable zip compression of the PDF "1|0"        -->
  <!-- margin-top          Additional top margin of the page in 1/1000mm          -->
  <!-- margin-left         Additional left margin of the page in 1/1000mm         -->
  <!-- margin-right        Additional right margin of the page in 1/1000mm        -->
  <!-- margin-bottom       Additional bottom margin of the page in 1/1000mm       -->
  <!-- embed-fonts         Embed all TrueType fonts in the document "0|1"         -->
  <!-- embed-subset-fonts  Embed all TrueType fonts as subset "0|1"               -->
  <!--                     This will usually result in smaller PDF files.         -->
  <!-- embed-type3-fonts   Embed all fonts as type3 fonts which are some times    -->
  <!--                     smaller, but might look bad on screen. "0|1"           -->
```

```
<!-- colormode           color|grayscale|blackwhite                             -->
<!-- simulate-font-styles if 1 TFORMer will simulate bold fonts for fonts       -->
<!--                      that are not available as bold on this system.        -->
<PDF media="A4" orientation="portrait" compression="1" margin-top="0" margin-left="0"
     margin-right="0" margin-bottom="0" embed-fonts="1" embed-subset-fonts="0"
     embed-type3-fonts="1" colormode="color" simulate-font-styles="1" >

<!-- compression-method  "jpeg|zip|auto":                                       -->
<!--                     - "jpeg"  all images will be jpeg compressed           -->
<!--                     - "zip"   all images are zip compressed                -->
<!--                     - "auto"  use jpeg compression for jpegs, else zip     -->
<!-- jpeg-quality        jpeg quality setting for jpeg compression (0-100)      -->
  <Images compression-method="auto" jpeg-quality="80">

<!-- Images can be downsampled if they are higher than the specified resolution -->
<!-- enabled             "0|1" enable/disable downsampling of images            -->
<!-- dpi                 all images are downsampled to this resolution          -->
    <Downsample enabled="1" dpi="300" />
  </Images>

<!-- user-password       password required for opening a document                  -->
<!-- owner-password      password required to change user permissions              -->
<!-- printing-permission "none|low-resolution|high-resolution":                    -->
<!--                     - "none"              disallow printing                   -->
<!--                     - "low-resolution"    allow only low-resolution printing  -->
<!--                     - "high-resolution"   allow full printing                 -->
<!-- modify-permission   "none|assembly|annotate|all":                             -->
<!--                     - "none"              allow no modifications              -->
<!--                     - "pages|assembly"    allow document assembly only        -->
<!--                     - "comments|annotate" assembly permissions plus commenting -->
<!--                     - "all"               allow full document modification    -->
<!-- allow-copy-content  "0|1"                 allow copying of any content        -->
<!-- allow-screen-reader-text-access  "0|1"                                        -->
<!-- encrypt-content     "all|exclude-meta-data":                                  -->
<!--                     - "all"                encrypt all content                -->
<!--                     - "exclude-meta-data"  prevent encryption of metadata     -->
  <Security user-password="" owner-password="" printing-permission="high-resolution"
            modify-permission="none" allow-copy-content="1"
            allow-screen-reader-text-access="1" encrypt-content="exclude-meta-data" />
</PDF>

<!-- POSTSCRIPT OPTIONS: Configure settings for PostScript output             -->
<!-- media               Default media size, must be listed in <DefinedMedia> -->
<!-- orientation         Default orientation "landscape|portrait"             -->
<!-- colormode           color|grayscale|blackwhite                           -->
<!-- duplex              Selects duplex or double-sided printing for printers -->
<!--                     capable of duplex printing.                          -->
<!--                     "default|simplex|vertical|horizontal":               -->
<!--                     - "default"    use printer settings                  -->
<!--                     - "simplex"    normal (nonduplex) printing           -->
<!--                     - "vertical"   double-sided printing using a         -->
<!--                                    vertical page turn                    -->
<!--                     - "horizontal" double-sided printing using a         -->
<!--                                    horizontal page turn                  -->
<!-- Header              Path to the PostScript header file                   -->
<!-- Footer              Path to the PostScript footer file                   -->
<POSTSCRIPT media="A4" orientation="portrait" colormode="color" duplex="default" >
    <Header filename="header.ps" />
    <Footer filename="footer.ps" />

<!-- compression-method  "nocomp|jpeg|flate|rle|auto":                          -->
<!--                     - "nocomp"no compression                               -->
<!--                     - "jpeg"  non monochrome images will be jpeg compressed-->
<!--                               monochrome images will use flate or rle      -->
<!--                               compression                                  -->
<!--                     - "flate" all images will be flate compressed          -->
<!--                     - "rle"   all images will be run length encoded        -->
<!--                     - "auto"  depends on jpeg-enabled and flate-enabled    -->
<!--                               uses jpeg compression for jpeg files         -->
<!--                               flate or rle for others                      -->
<!-- jpeg-quality        jpeg quality setting for jpeg compression (0-100)      -->
<!-- jpeg-enabled        enables jpeg compression for auto compression mode     -->
<!-- flate-enabled       enables flate compression for auto compression mode    -->
<!--                     requires Postscript Language Level 3 support           -->
```

```
    <Images compression-method="auto" jpeg-quality="75" jpeg-enabled="1" flate-enabled="0"
>
  <!-- Images can be downsampled if they are higher than the specified resolution -->
  <!-- enabled             "0|1" enable/disable downsampling of images            -->
  <!-- dpi                 all images are downsampled to this resolution          -->
      <Downsample enabled="1" dpi="300" />
    </Images>
  </POSTSCRIPT>

  <!-- HTML OPTIONS:       Configure settings for HTML output                     -->
  <!-- media               Default media size, must be listed in <DefinedMedia>   -->
  <!-- orientation         Default orientation "landscape|portrait"               -->
  <!--  96 dpi - this should be used as default resolution for browser apps       -->
  <!-- 108 dpi - seems to work better on Linux machines (depends on module width) -->
  <!--  72 dpi - might be a good choice if your target audience is on MAC OS      -->
  <!--                                                                            -->
  <!-- barcode-opt-resolution         print barcodes in optimal resolution        -->
  <!--                                This will create readable barcodes which    -->
  <!--                                might look different as in design mode.     -->
  <!-- always-print-barcodes          will print barcodes even if they will be    -->
  <!--                                not readable. This is ONLY useful for       -->
  <!--                                preview purposes.                           -->
  <HTML media="A4" orientation="portrait" resolution="96" barcode-opt-resolution="0"
    always-print-barcodes="1" />

  <!-- IMAGE OPTIONS:       Configure settings for image output                   -->
  <!-- media                Default media size, must be listed in <DefinedMedia>  -->
  <!-- orientation          Default orientation "landscape|portrait"              -->
  <!-- resolution           The output resolution in dpi.                         -->
  <!-- anti-aliasing        1 = enabled                                           -->
  <!--                      0 = disabled                                          -->
  <!--                      Anti-aliasing improves the quality of the output      -->
  <!--                      in lower resolutions or for on screen display.        -->
  <!-- jpeg-quality         jpeg quality setting for jpeg compression (0-100)     -->
  <!-- tiff-compression     Compression of generated tiff images.                 -->
  <!--                      A few compression schemes are only allowed for b&w    -->
  <!--                      or color output. Suggested values that should always  -->
  <!--                      work are: nocomp, lzw or jpeg                         -->
  <!--                      Valid values:                                         -->
  <!--                      nocomp, lzw, packbits, ccitt3, ccitt4, ccittfax, jpeg -->
  <!-- colormode            color|grayscale|blackwhite                            -->
  <!-- dither-mode          Specifies how images are converted to black&white     -->
  <!--                      Applies only if colormode="blackwhite"                -->
  <!--                      0 = Use scatter dithering (simulates grayscale)       -->
  <!--                      1 = Use ordered dithering (simulates grayscale)       -->
  <!--                      2 = Use threshold dithering (results in b&w only)     -->
  <IMAGE media="A4" orientation="portrait" resolution="200" anti-aliasing="1"
    jpeg-quality="80" tiff-compression="nocomp" colormode="color" dither-mode="0" />

  <!-- ZEBRA OPTIONS:       Configure options for Zebra/ZPL output                -->
  <!-- media                Default media size, must be listed in <DefinedMedia>  -->
  <!-- orientation          Default orientation "landscape|portrait"              -->
  <!-- compression          Image compression for graphics embedded in ZPL        -->
  <!--                      'none' = no compression, data is hex encoded          -->
  <!--                      'rle'  = data is run-length-encoded                   -->
  <!--                      'png'  = data is png and base64 encoded               -->
  <!--                      'zip'  = data is zip compressed and base64 encoded    -->
  <!-- dont-dither-text     Text will always be drawn black, no dithering occurs  -->
  <!--                      This leads to improved text quality                   -->
  <!-- dither-mode          Specifies how images are converted to black&white     -->
  <!--                      0 = Use scatter dithering (simulates grayscale)       -->
  <!--                      1 = Use ordered dithering (simulates grayscale)       -->
  <!--                      2 = Use threshold dithering (results in b&w only)     -->
  <!-- scaling              Resolution mode of the Zebra printer device           -->
  <!--                      0 = normal resolution (^JMA  Command)                 -->
  <!--                      1 = half resolution, doubled output size (^JMB Cmd)   -->
  <!-- resolution           resolution of the printer in dpi                      -->
  <!-- lt                   metric value for ^LT (label top) command              -->
  <!--                      Default unit is mm, if required you can specify other -->
  <!--                      units like in, cm, m, mil, etc. (Default: 0 mm)       -->
  <!--                      provide an empty value to omit the command            -->
  <!-- ls                   metric value for ^LS (label shift) command            -->
  <!--                      Default unit is mm, if required you can specify other -->
  <!--                      units like in, cm, m, mil, etc. (Default: 0 mm)       -->
  <!--                      provide an empty value to omit the command            -->
```

```
<!-- Header               custom ZPL code that is printed before each print-job -->
<!-- Footer               custom ZPL code that is printed after each print-job  -->
<ZEBRA media="A4" orientation="portrait" compression="rle" scaling="0"
  dont-dither-text="1" dither-mode="0" resolution="203" lt="0 mm" ls="0 mm" >
  <Header></Header>
  <Footer></Footer>
</ZEBRA>

<!-- Settings for Windows Printers                                              -->
<WINGDI>

  <!-- drawing-mode       Specifies the method for printing barcodes on GDI     -->
  <!--                    0 = Default method                                    -->
  <!--                    1 = Compatibility mode                                -->
  <!--                        Use the GDI rectangle functions to draw bars.     -->
  <!--                        May result in sub-optimal output quality,         -->
  <!--                        but is compatible to all printers.                -->
  <!--                    2 = Quality mode                                      -->
  <!--                        Best quality. Supported by most printers.         -->
  <!--                    3 = Dual                                              -->
  <!--                        A combination of 1 and 2                          -->
  <TBARCODE drawing-mode="1" />

  <!-- transparent-drawing-mode                                                 -->
  <!--                    Specifies the drawing method for transparent images   -->
  <!--                    on Win32 printers (trouble shoot printer issues)      -->
  <!--                    'default' = Default method                            -->
  <!--                    'auto'    = tries to determine best method            -->
  <!--                    'TransparentBlt' = use GDI TransparentBlt method      -->
  <!--                    'MaskBlt' = use GDI MaskBlt method                    -->
  <!--                    'BitBlt'  = use combination of BitBlts, needs access  -->
  <!--                                to device pixels                          -->
  <!--                    'Opaque'  = ignore transparency and print image opaque-->
  <Images transparent-drawing-mode="default" />
</WINGDI>

<!-- SVG Options:         Configure options for Scalable Vector Graphics output -->
<!-- media                Default media size, must be listed in <DefinedMedia>  -->
<!-- orientation          Default orientation "landscape|portrait"              -->
<!-- colormode            color|grayscale|blackwhite                            -->
<!-- minify               Specifies whether the SVG output is minified          -->
<!--                      1 = removes unnecessary line breaks and indentation,  -->
<!--                          metadata is omitted                               -->
<!--                      0 = formats output for readability, includes          -->
<!--                          line breaks, indentation and metadata             -->
<SVG media="A4" orientation="portrait" colormode="color" minify="0">

  <!-- weld                "0|1" enables or disables merging (welding) of adjacent-->
  <!--                           shapes into a single path                        -->
  <!--                           (default: 1)                                     -->
  <Barcodes weld="1" />

  <!-- compression-method  "auto|jpeg|png":                                       -->
  <!--                     - "auto"  jpeg images stay jpeg compressed             -->
  <!--                               all others, including monochrome or          -->
  <!--                               transparent images will use png compression  -->
  <!--                     - "jpeg"  images will be jpeg compressed               -->
  <!--                               monochrome or transparent images will use    -->
  <!--                               png compression                              -->
  <!--                     - "png"   all images will be png compressed            -->
  <!-- jpeg-quality        jpeg quality setting for jpeg compression (0-100)      -->
  <Images compression-method="auto" jpeg-quality="80">

    <!-- Images can be downsampled if they are higher than the specified resolution -->
    <!-- enabled             "0|1" enable/disable downsampling of images             -->
    <!-- dpi                 all images are downsampled to this resolution           -->
    <Downsample enabled="0" dpi="300"/>
  </Images>
</SVG>

<!-- DEFINED MEDIA SIZES                                                        -->
<!-- Add YOUR CUSTOM MEDIA-SIZES to the list and use them in TFORMer            -->
<!-- Use them by specifying the name of the media as output tray name           -->
<!-- (enter the custom name directly into the tray option in TFORMer Designer)  -->
<!-- All measurements are specified in 1/1000 millimeters                       -->
```

```
   <DefinedMedia>
    <Media name="A0"           width="841000"  height="1189000" />
    <Media name="A1"           width="594000"  height="841000"  />
    <Media name="A2"           width="420000"  height="594000"  />
    <Media name="A3"           width="297000"  height="420000"  />
    <Media name="A4"           width="210000"  height="297000"  />
    <Media name="A5"           width="148000"  height="210000"  />
    <Media name="A6"           width="105000"  height="148000"  />
    <Media name="A7"           width="74000"   height="105000"  />
    <Media name="A8"           width="52000"   height="74000"   />
    <Media name="A9"           width="37000"   height="52000"   />
    <Media name="A10"          width="26000"   height="37000"   />
    <Media name="B0"           width="1000000" height="1414000" />
    <Media name="B1"           width="707000"  height="1000000" />
    <Media name="B2"           width="500000"  height="707000"  />
    <Media name="B3"           width="353000"  height="500000"  />
    <Media name="B4"           width="250000"  height="353000"  />
    <Media name="B5"           width="176000"  height="250000"  />
    <Media name="B6"           width="125000"  height="176000"  />
    <Media name="B7"           width="88000"   height="125000"  />
    <Media name="B8"           width="62000"   height="88000"   />
    <Media name="B9"           width="44000"   height="62000"   />
    <Media name="B10"          width="31000"   height="44000"   />
    <Media name="C0"           width="917000"  height="1297000" />
    <Media name="C1"           width="648000"  height="917000"  />
    <Media name="C2"           width="458000"  height="648000"  />
    <Media name="C3"           width="324000"  height="458000"  />
    <Media name="C4"           width="229000"  height="324000"  />
    <Media name="C5"           width="162000"  height="229000"  />
    <Media name="C6"           width="114000"  height="162000"  />
    <Media name="C7"           width="81000"   height="114000"  />
    <Media name="C8"           width="57000"   height="81000"   />
    <Media name="C9"           width="40000"   height="57000"   />
    <Media name="C10"          width="28000"   height="40000"   />
    <Media name="D0"           width="771000"  height="1091000" />
    <Media name="D1"           width="545000"  height="771000"  />
    <Media name="D2"           width="385000"  height="545000"  />
    <Media name="D3"           width="272000"  height="385000"  />
    <Media name="D4"           width="192000"  height="272000"  />
    <Media name="D5"           width="136000"  height="192000"  />
    <Media name="D6"           width="96000"   height="136000"  />
    <Media name="D7"           width="68000"   height="96000"   />
    <Media name="E0"           width="800000"  height="1120000" />
    <Media name="E1"           width="560000"  height="800000"  />
    <Media name="E2"           width="400000"  height="560000"  />
    <Media name="E3"           width="280000"  height="400000"  />
    <Media name="E4"           width="200000"  height="280000"  />
    <Media name="E5"           width="140000"  height="200000"  />
    <Media name="E6"           width="100000"  height="140000"  />
    <Media name="E7"           width="70000"   height="100000"  />
    <Media name="B0JIS"        width="1030000" height="1456000" />
    <Media name="B1JIS"        width="728000"  height="1030000" />
    <Media name="B2JIS"        width="515000"  height="728000"  />
    <Media name="B3JIS"        width="364000"  height="515000"  />
    <Media name="B4JIS"        width="257000"  height="364000"  />
    <Media name="B5JIS"        width="128000"  height="257000"  />
    <Media name="B6JIS"        width="128000"  height="182000"  />
    <Media name="B7JIS"        width="91000"   height="128000"  />
    <Media name="B8JIS"        width="64000"   height="91000"   />
    <Media name="B9JIS"        width="45000"   height="64000"   />
    <Media name="B10JIS"       width="32000"   height="45000"   />
    <Media name="Invoice"      width="140000"  height="216000"  />
    <Media name="Executive"    width="191000"  height="254000"  />
    <Media name="Legal"        width="215900"  height="355600"  />
    <Media name="JuniorLegal"  width="203200"  height="127000"  />
    <Media name="Letter"       width="215900"  height="279400"  />
    <Media name="Ledger"       width="431800"  height="279400"  />
    <Media name="Tabloid"      width="279400"  height="431800"  />
    <Media name="Broadsheet"   width="432000"  height="559000"  />
    <Media name="Screen"       width="297000"  height="210000"  />
    <Media name="Custom"       width="210000"  height="297000"  />
    <Media name="Comm10"       width="105000"  height="241000"  />
    <Media name="DL"           width="110000"  height="220000"  />
    <Media name="Folio"        width="210000"  height="330200"  />
    <Media name="P1"           width="560000"  height="860000"  />
    <Media name="P2"           width="430000"  height="560000"  />
    <Media name="P3"           width="280000"  height="430000"  />
    <Media name="P4"           width="215000"  height="280000"  />
    <Media name="P5"           width="140000"  height="215000"  />
    <Media name="P6"           width="107000"  height="140000"  />
    <!-- add custom forms below this point -->
    <Media name="KLT"          width="210000"  height="74000"   />
    <Media name="KLT2"         width="210000"  height="42000"   />
    <Media name="4x6"          width="101600"  height="152400"  />
    <Media name="4x5"          width="101600"  height="127000"  />

  </DefinedMedia>
</TFORMer>
```

Appendix D: Distribution and Deployment for Microsoft Windows

D.1 Core Requirements

If you want to use the TFORMer SDK within your own application (or on the command line) you have to distribute the files listed below. These files must be shipped in order to enable basic TFORMer features and to offer complete printing support on Microsoft® Windows based printers. You need to deploy the x64 versions only if required (if you are using the 64-bit SDK).

- TFormer9.dll (the main DLL for TFORMer SDK required in each case)
- TFormerRep9.dll
- TECBaseU.dll
- TFPrint.exe (required for command-line printing only)
- FreeImageMT.dll
- freeimage.LICENSE
- freetype248MT.dll
- freetype.LICENSE
- iconvx32.dll
- iconv.LICENSE
- libart_lgpl.LICENSE
- libxml2x32.dll
- libxml2.LICENSE
- libxsltx32.dll
- zlib1.dll
- zlib.README
- TFORMer.xml (contains system-wide output options like page size, PDF compression mode, and more – see Appendix C)

Additionally, the **Microsoft Common Runtime DLLs** (Visual C++ 2015-2022 runtime components) are required on the target system.

- You can install the components with the *Microsoft Visual C++ 2015-2022 Redistributable Package (x86)* available at
  https://aka.ms/vs/17/release/vc_redist.x86.exe
  and (x64) available at
  https://aka.ms/vs/17/release/vc_redist.x64.exe
  The package installs all required runtime DLLs.

Please note:

TFORMer 9.0.0	Requires Microsoft® Visual C++ 2015-2022 Redistributable Package (x86).

Please contact TEC-IT Support if you need help.

D.2 Licensing

If you are redistributing the TFORMer SDK as part of your application, please use programmatic licensing. This allows you to securely embed or protect your license key.

If you are using the TFORMer SDK in a web or server application, you can also apply the license by deploying this additional license file:

- TFORMer.ini
  (should be located in the SDK deployment or installation path – when using the default TFORMer SDK installation, you could copy it to
  `"C:\Program Files\TEC-IT\TFORMer9 SDK"` or
  `"C:\Program Files (x86)\TEC-IT\TFORMer9 SDK"`)

For detailed information, please refer to the API Reference: *Related Pages* ► *Licensing TFORMer*.

D.3 API Specific Requirements

D.3.1 COM API

If you want to use the COM based interface of TFORMer SDK, this additional DLL is required:

- TFORMerCOM9.dll
  This DLL provides the COM interface; mark TFORMerCOM9.DLL as self-registering file (or use `"regsvr32.exe TFORMerCOM9.dll"`)

D.3.2 .NET API

If you want to use the .NET interface of TFORMer SDK, this additional DLL is required:

- TECIT.TFORMer.dll
  This DLL provides the .NET interface; it is installed by default in the GAC as well as in the Program Files folder.

> ► For executing .NET code, the Microsoft .NET Framework 4.5 or higher is required! A redistributable package of the framework is available at https://dotnet.microsoft.com/download/dotnet-framework

D.3.3 JAVA API

If you want to use the JAVA interface of TFORMer SDK, these additional files are required:

- JTFORMer9.jar
- TFORMer9JNI.dll

> ► Please note that the JAR file is compatible with version 1.5 and higher of the Java Runtime Edition (JRE)

D.4 Requirements for PostScript/HTML Output

In addition, ship the following files whenever native PostScript/HTML output is required. These files support font-handling on non-Windows based output devices.

- header.ps (Required only for native PostScript compatible output)
- footer.ps (Required only for native PostScript compatible output)
- template.html (Required only for native HTML compatible output)
- footer.html (Required only for native HTML compatible output)

D.5 Distribution of TFORMer Document Layouts

Ship your TFORMer Form Layouts as part of your application.

- **Repository** based Form Layouts
  Ship the Repository (*.tfr file) and all subdirectories (Forms/*). Make sure to include all files (*.xmd, *.xml, embedded images) and take care to preserve the directory structure below the *.tfr file!
- **Stand-alone** Form Layouts
  Ship all the *.tff file(s) and the corresponding *.xml data file(s) as well as embedded image files.
- **ZIP files** containing the files mentioned above
  TFORMer SDK is able to handle zip-files directly. When using ZIP-files take care to include all files as mentioned above and preserve the directory structure (relative to the *.tfr or *.tff file) within the ZIP-file.
- Instead of shipping files you can also pass Form Layouts in **BASE64 encoded strings.**
  TFORMer SDK is able to handle BASE64 strings which contain a Form Layout or even a complete Repository. The files for the Form Layout or Repository may also be zipped (including all required image files) before being BASE64 encoded. Pass the BASE64 encoded strings with the prefix `“BASE64:”` instead of filenames to the TFORMer SDK functions.

Appendix E: Distribution and Deployment for Linux® or UNIX®

E.1 Core Requirements

If you want to use the TFORMer SDK within your own application (or on the command line) you have to distribute the files listed below:

- libTFORMer9.so
  (must be accessible through the library search path)
- TFORMer.xml
  (should be located in `"/usr/local/share/TFORMer9/"` – otherwise the path to the file must be specified with TFORMer_SetConfigFile())

Furthermore, the following libraries are required on the target system:

- LibXml2
- LibXslt
- FreeImage
- FreeType2
- FontConfig
- ZLib
- LibArt_LGPL (optional – required for image output)
- CUPS (optional – required for direct printing under Linux)
- UnixODBC (optional – required for SQL support)

E.2 Licensing

If you are redistributing the TFORMer SDK as part of your application, please use programmatic licensing. This allows you to securely embed or protect your license key.

If you are using the TFORMer SDK in a web or server application, you can also apply the license by deploying this additional license file:

- TFORMer.ini
  (should be located in the TFORMer SDK deployment or installation path – when using the default TFORMer SDK installation, you could also copy it to `"/usr/local/share/TFORMer9"`)

For detailed information, please refer to the API Reference: *Related Pages* ► *Licensing TFORMer*.

E.3 API Specific Requirements

E.3.1 Shared Library / DLL

► For compiling C/C++ projects under Linux or UNIX you need to install the FontConfig and FreeType2 development packages!
Try the following command to install them:

```
apt-get install libfontconfig1-dev
apt-get install libfreetype6-dev
```

E.3.2 .NET API

If you want to use the .NET interface of TFORMer SDK, this additional DLL is required:

- TECIT.TFORMer.dll

E.3.3 JAVA API

If you want to use the JAVA interface of TFORMer SDK, these additional files are required:

- JTFORMer9.jar
- TFORMer9JNI.dll

► Please note that the JAR file is compatible with version 1.5 and higher of the Java Runtime Edition (JRE)

E.4 Requirements for PostScript/HTML Output

Ship the following files whenever native PostScript/HTML output is required.

- header.ps (Required for native PostScript compatible output)
- footer.ps (Required for native PostScript compatible output)
- template.html (Required for native HTML compatible output)
- footer.html (Required for native HTML compatible output)

E.5 Distribution of TFORMer Document Layouts

See section D.5!