TFORMer Runtime

Output-Engine

Version 5.1

Developer Manual

21 January 2008

TEC-IT Datenverarbeitung GmbH
Wagnerstrasse 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
Wagnerstr. 6

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

3 Introduction

3.1 TFORMer

The **TFORMer** product family represents a lean, powerful output and reporting solution for arbitrary documents.

Professional design features are combined with versatile output capabilities. The integrated *bar-code generator*, full-featured *UNICODE* support, direct *PDF generation* and the possibility for client as well as server side use qualifies **TFORMer** as *multi-purpose, multi-platform report generator*.

3.2 Areas of Application for TFORMer Runtime

TFORMer Runtime provides the core output functionality of the **TFORMer** product family. It is available for all major operating systems and can be embedded easily in your own applications. Due to the broad support of output formats **TFORMer Runtime** is used in the following areas:

- **Reporting Engine**
  TFORMer Runtime and **TFORMer Designer** provide software developers a generic reporting solution which enables end-customers to create or edit form layouts directly.
- **PDF Library**
  TFORMer Runtime creates high-quality PDF documents based on graphical form 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 Runtime** 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.

3.3 This Document

The target audiences of this document are software developers and system administrators. The information in this document provides a high-level introduction regarding the supported operating systems and the available API's (Application programming Interfaces) of **TFORMer Runtime**.

For more detailed information regarding programming **TFORMer Runtime** we recommend to check out these additional documents:

- **TFORMer Runtime** – DLL Programming Reference
  This document describes the DLL API (Windows) as wells as the shared library API (Linux).
- **TFORMer Runtime** – COM Programming Reference
  This document describes the COM programming interface.
- **TFORMer Runtime** – .NET Programming Reference
  This document describes the .NET programming interface.
- **TFORMer Designer** User Manual
  This document provides an in-depth documentation of the graphical layout editor.

4 Highlights of TFORMer Runtime

4.1 Unified Output Solution

TFORMer Runtime 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 form layouts directly on all printers supported by Microsoft® Windows®.
- **ZEBRA®**
  ZPL-II compatible output for ZEBRA printers is available on all platforms, no special printer drivers are needed.
- **PDF**
  PDF export with full-featured bar-code support.
- **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, PNG and TIF formats.
- **ASCII**
  Pure ASCII output without any graphics for special purpose requirements.

4.2 UNICODE Support

The integrated UNICODE support prints all scripts in common use.

4.3 Barcode Support

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

4.4 Reusable, Smart Document Layouts

TFORMer separates form layouts from the underlying data, does not bind form layouts to certain printer models and offers complete database independence. Forms offer built in scripting features: the output may be controlled by means of conditional printing, tray control and scripting.

4.5 Cross Platform

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

4.6 Small Footprint, High Performance

TFORMer provides high performance PDF and printer output in combination with very low system requirements (only 20 MB hard-disc space is required for the binaries on Microsoft Windows).

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

5 Installation on Microsoft® Windows®

5.1 Introduction

This chapter describes the installation of **TFORMer** on Microsoft® Windows®. The setup application comes as MSI file (Microsoft Windows Installer). MSI setup applications can be used without installing any additional tools on Windows Vista®, the Windows Server 2003 family, Windows XP, and Windows 2000. MSI 2.0 or higher is required.

The core functionality of **TFORMer** is available without .NET 2.0 but **TFORMer** ships with some .NET based components like a .NET assembly, sample applications and **TFORMer QuickPrint**. If you are interested in these features install the .NET 2.0 runtime before installing **TFORMer**.

5.2 General

The installation of **TFORMer** on Microsoft Windows is straight-forward. Just execute the setup application to install **TFORMer** executables, sample applications and documentation. **TFORMer Runtime** is installed as part of **TFORMer Designer**.

The default installation path is:

```
C:\Program Files\TEC-IT\TFORMer5
```

On Windows Vista all sample applications and form layout templates are installed in the following directory (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\5.1
```

All other versions of Microsoft Windows use the following directory for sample applications and templates:

```
C:\Documents and Settings\All Users\Application Data\TEC-IT\TFORMer\5.1
```

5.3 Supported APIs

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

- **COM Component**
  This API is available via the file *TFORMerCOM51.dll*. The COM component is registered automatically by the setup application.
- **.NET Component**
  The file *TECIT.TFORMer.dll* provides the .NET functionality. .NET 2.0 or higher is required. The .NET assembly is automatically installed in the GAC.
- **DLL**
  TFormer51.dll is a 32-bit Windows DLL. The corresponding library and include file are named *TFormer51.lib* and *TFormer5.h*. The DLL is available on Linux and UNIX as well. For details please refer to chapter 6.
- **Command Line Application**
  The command-line based executable is named *tfprint.exe.* The command line application is available on Linux and UNIX as well. For details please refer to chapter 6.

6 Installation on Linux® or UNIX®

6.1 Introduction

This chapter describes the installation of **TFORMer Runtime** on Linux® and UNIX® based operating systems. Depending on the operating system **TFORMer Runtime** is available as *rpm* or *tarball*. For details, please refer to section 6.4.

> ► Please note that the graphical design tool **TFORMer Designer** is only available for Microsoft Windows.

6.2 General

The default installation path of **TFORMer Runtime** is

```
/usr/local/
```

The executable *tfprint* (**TFORMer Runtime** Command Line Application) is installed into

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

All configuration files as well as the license file **tfprint.ini** are located in the following directory:

```
/usr/local/share/TFORMer/
```

This directory contains the following subdirectories:

- *Templates* holds ready-to-use form layout templates designed with **TFORMer Designer**.
- *Demo Repository* contains a sample repository. The forms referenced by this repository are used by the sample applications.
- *Demos* contains sample shell-scripts which demonstrate the use of *tfprint* (see chapter 11).
- *SampleCode* contains a sample application written in C (TFORMerSimpleX.c). It generates barcode labels as a PDF-Document.
- *SampleCodeCGI* contains a PHP sample. This web-based sample creates PDF files from user supplied data using *tfprint*.

The include files needed for software development are stored in:

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

The developer documentation is available by opening *index.html* with your browser in:

```
/usr/local/share/TFORMer/APIDocs
```

Additional documentation is available through a man page.

$ man tfprint

6.3 Supported APIs

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

- **Shared Library**
  named *libTFORMer5.so* or *libTFORMer5.a*. The interface of the shared library is almost identical to the DLL API on Microsoft Windows.
- **Command Line Application**
  The command-line based executable is named *tfprint*

6.4 Platform specific Installation Hints

6.4.1 Linux® (deb- or rpm-based)

The listed Linux distributions are supported with ready-to-run binaries. Whenever possible, other platforms are provided on request.

- SuSE Linux 10.2 on i586
- SuSE Linux Enterprise 9 on i586
- RedHat Enterprise 5 on i586
- RedHat Fedora Core 6 on i586
- Ubuntu 7.04 on i586

6.4.1.1 Dependencies

Before installing **TFORMer Runtime** 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)
```

6.4.1.2 Rpm Installation

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

$ su
$ sh TFORMer-5.1.0-1.i586.rpm.bin

After confirming the license agreement with *yes* this script installs the rpm. After installation the possibility to generate a demo-report is offered.

1.1.1.1 Rpm De-Installation

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

$ su
$ rpm -e TFORMer

6.4.1.3 Deb (Debian) Installation

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

$ sudo su
$ sh TFORMer-5.1.0-1-i386.deb.bin

After confirming the license agreement with *yes* this script installs the rpm. After installation the possibility to generate a demo-report is offered.

1.1.1.2 Deb (Debian) De-Installation

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

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

6.4.2 HP-UX® (tarball)

The listed platforms are supported with ready-to-run binaries. Whenever possible, binaries for other platforms are provided on request.

- HP-UX 11i v2 (B.11.23) on IA64 (Itanium-2 Architecture)
- HP-UX 11 (11.00) on PA-RISC 2.0

6.4.2.1 Dependencies

Before installing **TFORMer Runtime** 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
```

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

6.4.2.2 Installation

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

```
$ gunzip SetupTFORMer-5.1.0-HPUX11.23-IA64.tar.gz
$ tar xf SetupTFORMer-5.1.0-HPUX11.23-IA64.tar
$ cd SetupTFORMer
$ sh install.sh
```

After confirming the license agreement with *yes* this script installs **TFORMer**. After installation the possibility to generate a demo-report is offered.

6.4.3 AIX® (tarball)

The listed platforms are supported with ready-to-run binaries. Whenever possible, binaries for other platforms are provided on request.

- AIX 4.3 PPC

6.4.3.1 Dependencies

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

► Installation of the RPM Package Manager is a must!

```
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
```

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

6.4.3.2 Installation

The installation is identical to HP-UX (see section 6.4.2.2).

6.4.4 Solaris® (tarball)

The listed platforms are supported with ready-to-run binaries. Whenever possible, binaries for other platforms are provided on request.

- Solaris 10 x86

6.4.4.1 Dependencies

Before installing **TFORMer Runtime** 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
```

6.4.4.2 Installation

To install TFORMer 5.0 (5.1) for Solaris 10 perform the following steps:

```
gunzip TFORMer-5.0.0-solaris10.x86.tar.gz
cp TFORMer-5.0.0-solaris10.x86.tar /
cd /
tar xvf TFORMer-5.0.0-solaris10.x86.tar
```

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

7 COM Component

7.1 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 2005, Borland C++ Builder, Borland Delphi, …
- the Microsoft Office product suite (in conjunction with Visual Basic for Applications – VBA)
- HTML pages on client- or server-side (ASP, ASP . NET, VBScript, …)
- command-line based scripting environments (Visual Basic Scripting Host)

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

7.2 Outline

► The name of the type library is *TFORMer 5.1 Runtime Type Library*.
► The GUID is *E53A469C-3353-4c2b-8DCE-13EA6E7C8921*.

The general steps for using **TFORMer Runtime** via the COM based API are:

1. Embed **TFORMer Runtime** into your application or document. Usually this is done by establishing a reference to *TFORMer 5.1 Runtime Type Library*.
2. Create an instance of **TFORMer Runtime** (*TFORMer51Lib.TFORMer*).
3. Create a job-instance.
4. Create a data-source instance and connect the job with the data-source.
5. Set the properties of the **TFORMer Runtime** to 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 data-source.
7. Finally call the *Print* method.

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

```
' This code snippet demonstrates the basic steps for using TFORMer Runtime from within
' VBA. Take care to establish a reference to TFORMer 5.1 Runtime Type Library beforehand!

Private Sub TFORMer_Output()

   ' Declare the variables

   Dim TFormer
   Dim PrintJob
   Dim DataSource As TFORMer51Lib.IDataSourceRecordSet

   ' create a TFORMer Runtime instance

   Set TFormer = CreateObject("TFORMer51Lib.TFormer")

   ' create a job object with TFORMer

   Set PrintJob = TFormer.CreateJob

   ' create (and connect) a data source for providing data-fields values

   Set DataSource = printjob.NewDataSourceRecordSet
```

```
    ' 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

    DataSource.AddNewRecord
    DataSource.SetDataField "NAME_OF_YOUR_DATAFIELD", "This is the value of the data-field"

    ' Finally print the form layout

    printjob.PrintForm
End Sub
```

7.3 Using the COM object with Development Environments

7.3.1 Microsoft Visual Basic 6

1. Open or create your Visual Basic Project.
2. Choose *Project* ► *References* from the menu.
3. Choose *TFORMer 5.1 Runtime Type Library* from the list, select the entry and click *OK*.
4. Now you can use **TFORMer Runtime** as a COM object in your program.

7.3.2 Microsoft Visual C/C++ 6

1. Open and create your MFC application (e.g. by using the MFC App Wizard). Make sure that *Enable Automation* is adjusted in your project.
2. Choose menu *View* ► *ClassWizard* to enter the Class Wizard.
3. Choose the button *Add Class* ► *From a type library* and select *TFORMerCOM51.dll* from the installation path of **TFORMer** (*Bin* subdirectory).
4. Now the type library is read from this DLL and the implemented classes (*IJob, IDataSourceRecordSet…*) are displayed in the *Confirm Classes* dialog.
5. Proceed with *OK*.
6. After this step you can see the new TFORMer COM classes in the ClassView.
7. Include the generated header file *tformercom51.h* into your cpp file.
8. **TFORMer Runtime** may now be used in your MFC application.

7.3.3 Other development Environments

Please refer to the documentation of your development environment.

7.4 More Information

Check out the following documents or sample applications for more details:

- COM Programming Reference
  Start Menu ► *All Programs* ► *TEC-IT TFORMer 5.1* ►*Documentation* ► *COM Programming Reference.*
- Sample Microsoft Excel Spreadsheet using **TFORMer Runtime**
  Start Menu ► *All Programs* ► *TEC-IT TFORMer 5.1* ► *Examples* ► *Microsoft Office.*

8 . NET Component

8.1 Introduction

The **TFORMer Runtime** .NET component is perfect suited for application development with any .NET programming language (or Mono on Linux/UNIX).

- On Microsoft Windows the setup application installs the assembly automatically in the GAC (Global Assembly Cache) and in the *Bin* subdirectory of the default installation path of **TFORMer**. The .NET component can be used immediately with development environments like Microsoft Visual Studio.
- On Linux the setup scripts are installing the suitable Mono-bindings automatically.

> ► On Microsoft Windows the .NET API of **TFORMer** requires.NET 2.0 or higher.
> ► On Linux/UNIX **TFORMer** works with Mono 1.2.3 or higher (Mono is the open source equivalent for .NET).

8.2 Outline

> ► When establishing a reference the name of the .NET component is *TFORMer 5.1 Runtime* (file *TECIT.TFORMer.dll*).

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

1. To use **TFORMer Runtime** as part of your application establish a reference to *TFORMer 5.1 Runtime*.
2. Depending on the programming language include the *TECIT.TFORMer* namespace.
3. Create a job-instance (*TECIT.TFORMer.Job*).
4. Create a data-source instance (*TECIT.TFORMer.DataSource* or derived classes).
5. Connect the job with the data-source.
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 data-source.
8. Finally generate the required output with the *Print* method.

The general Visual Basic .NET code for using **TFORMer Runtime** looks as follows:

```
' This code snippet demonstrates the basic steps for using TFORMer Runtime from within
' VB .NET. Take care to establish a reference to TFORMer 5.1 Runtime from within your
' project!

Private Sub TFORMer_Output ()

  ' Declare the variables

  Dim job As TECIT.TFORMer.Job
  Dim datasource As TECIT.TFORMer.DataSourceRecordSet

  ' create objects for a job and a data-source

  job = New TECIT.TFORMer.Job
  datasource = New TECIT.TFORMer.DataSourceRecordSet

  ' connect the data-source with the job

  job.DataSource = datasource
```

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

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

    ' provide data for the data-fields used in the form layout

    Dim record As TECIT.TFORMer.Record
    record = New TECIT.TFORMer.Record
    record.Data.Add "NAME_OF_YOUR_DATAFIELD", "This is the value of the data-field")
    datasource.Records.Add(record)

    ' finally print it

    job.Print()

End Sub
```

8.3 More Information

Check out the following documents or sample applications for more details:

- NET Programming Reference
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ►Documentation ► NET Programming Reference
- Sample Code
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ► Examples ► Programming

9 DLL (Dynamic Link Library)

9.1 Introduction

Besides the COM and the .NET API **TFORMer Runtime** is also available as 32 bit Microsoft Windows DLL. 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 10).

9.2 Compiling your Own Applications

The files *TFormer51.lib* (library) and *TFormer5.h* (header file) are required to compile and link your own applications with the DLL interface of **TFORMer Runtime**. These files are usually stored in the following folder:

```
C:\Program Files\TEC-IT\TFORMer5\Bin
```

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

9.3 Outline

The general steps for using the DLL interface of **TFORMer Runtime** are:

1. Include *TFormer5.h*. Make sure to define *TECIT_DLLIMPORT* before including the header file.
2. Initialize TFORMer Runtime 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 data-fields (if any) in the form layout.
6. Generate or print the output *(TFormer_Print).*
7. De-initialize TFORMer Runtime *(TFormer_Exit).*

The general C-code for using **TFORMer Runtime** 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 TFormer5.h!

#define  TECIT_DLLIMPORT
#include "PATH_TO_TFORMER/TFormer5.h"
#undef   TECIT_DLLIMPORT

// Handle for TFORMer Runtime

HTFORM   hTForm  = NULL;

// initialize TFORMer Runtime

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 Data-Field value */

TFormer_SetVarValue (hTForm, "NAME_OF_YOUR_DATAFIELD", "Value of the data-field");

/* Start printing */

TFormer_Print (hTForm);

/* Exit TFORMer */

TFormer_Exit (hTForm);
```

9.4 More Information

Check out the following documents or sample applications for more details:

- DLL Programming Reference
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ►Documentation ► DLL Programming Reference.
- Sample applications with source code
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ► Examples ► Programming.
- Appendix A describes how data is provided to **TFORMer Runtime.**
- Appendix B describes how to adjust general output options.

10 Shared Library

10.1 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 9).

10.2 Compiling Your Own Applications

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

- *TFormer5.h,* this is the include file with the **TFORMer Runtime** API
- *TECITStd.h,* an additional include file for Linux/UNIX
- *libTFORMer5.so,* this is the shared library
- *libTFORMer5.a,* this is the static library containing **TFORMer Runtime**

After **TFORMer Runtime** was installed the include files *TFormer5.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 -ITFORMer5 -L/usr/local/lib -I/usr/local/include

10.3 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 *TFormer5.h.*
4. Initialize TFORMer Runtime 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 data-fields (if any) in the form layout.
8. Generate or print the output *(TFormer_Print).*
9. De-initialize TFORMer Runtime *(TFormer_Exit).*

The general C-code for using **TFORMer Runtime** 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 TFormer5.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 TFORMer5.h */

#include <TFormer5.h>

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

#undef TECIT DLLIMPORT
#undef _TEC_UNIX
#undef TEC_UNIX_BUILD

// Handle for TFORMer Runtime

HTFORM   hTForm  = NULL;

// initialize TFORMer Runtime

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 Data-Field value */

TFormer_SetVarValue (hTForm, "NAME_OF_YOUR_DATAFIELD", "Value of the data-field");

/* Start printing */

TFormer_Print (hTForm);

/* Exit TFORMer */

TFormer_Exit (hTForm);
```

10.4 Sample Application

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

```
/usr/local/share/TFORMer/SampleCode
```

This C-code generates barcode labels as PDF-file. The sample application uses the same document layout as the script DemoBarcodeLabels.sh mentioned in section 11.3.2.2. Data-field values are passed programmatically.

Compile it by executing:

$ gcc TFORMerSimpleX.c -o TFORMerSimpleX -ldl -lTFORMer5 -L/usr/local/lib -I/usr/local/include

10.5 More Information

Check out the following documents or sample applications for more details:

- DLL Programming Reference
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ►Documentation ► DLL Programming Reference.
- Sample Applications with source code
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ► Examples ► Programming.
- Appendix A describes how data is provided to **TFORMer Runtime.**
- Appendix B describes how to adjust general output options.

11 Command Line Application

11.1 Introduction

The command line based version of **TFORMer Runtime** is available for all supported operating systems (Microsoft Windows, Linux and UNIX). It generates output based on arbitrary document 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

11.2 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 data-sources.
3. Embed *tfprint* into your batch-job, shell-script or applications.

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

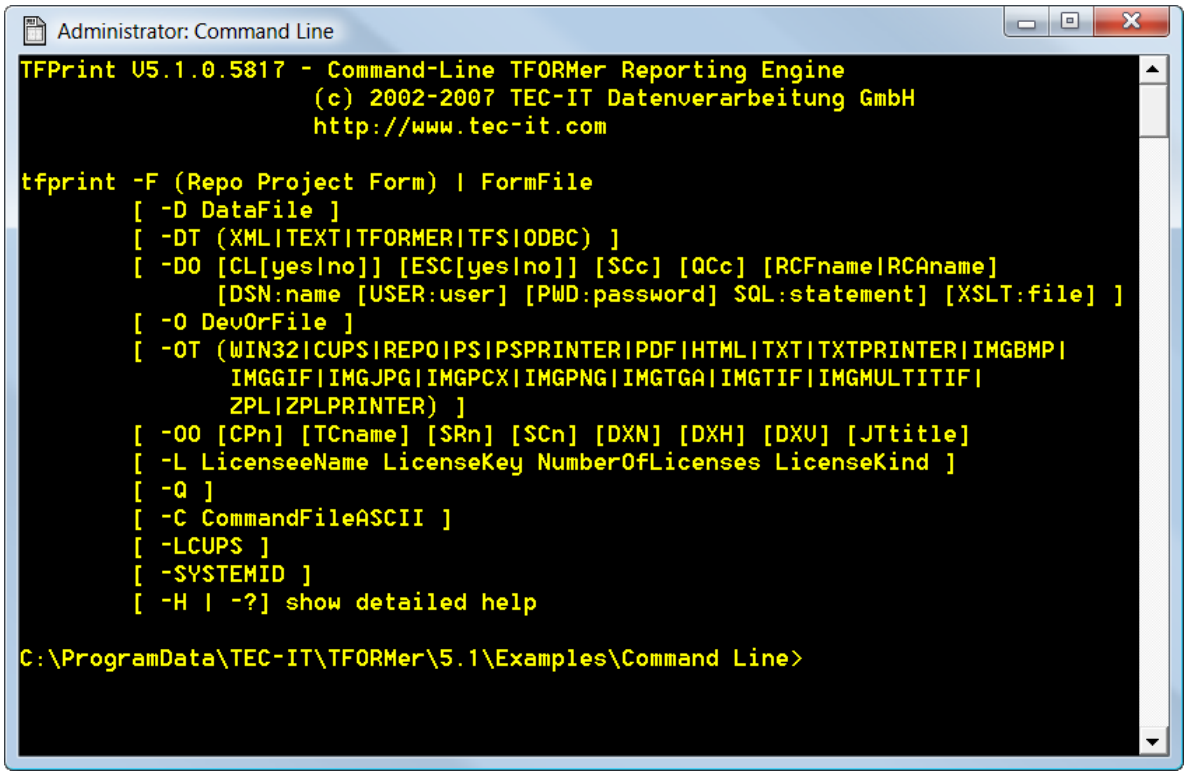

Figure 1: tfprint command line parameters

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

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

11.3 Samples

11.3.1 For Microsoft Windows

In the example below *tfprint* is used to generate a form layout (MyLayout.tff) using the data-source MyValues.txt (a data-source contains values for data-fields) 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 ► All Programs ► TEC-IT TFORMer 5.1 ► Examples ► Command Line Printing

11.3.2 For LINUX and UNIX

11.3.2.1 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/TFORMer/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!

11.3.2.2 Generate Barcode-Labels (PDF-Output)

This script generates barcode labels as PDF-file:

$ sh /usr/local/share/TFORMer/Demos/DemoBarcodeLabels.sh

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

11.4 More Information

Check out the following documents for more details:

- Check out the **TFPrint** User Manual.
  Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ►Documentation ► TFORMer V51 Command Line.
- 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 ► All Programs ► TEC-IT TFORMer 5.1 ►Documentation ► TFORMer V51 Designer.
- Appendix A describes how data is provided to **TFORMer Runtime.**
- Appendix B describes how to adjust general output options.

12 TFORMer Runtime with Microsoft Office

12.1 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 Runtime** 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 seamless. For example the **TFORMer Runtime** 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 Runtime** is also accessible via VBA.

12.2 Hints for different Office-Versions

This section concentrates on the first steps for integrating **TFORMer Runtime** 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.

12.2.1 Microsoft Word/Excel 2007

1. Open or create your document.
2. Enable the developer tab in the ribbon (*Office Button* ► *Word/Excel Options* ► *Show Developer tab in the Ribbon*).
3. Activate the Developer tab.
4. Click *Visual Basic,* Microsoft Visual Basic opens.
5. Click *Tools* ► *References* and check *TFORMer 5.1 Runtime Type Library* to insert a reference to **TFORMer Runtime.**
6. Click *OK.*
7. **TFORMer Runtime** may now be used with VBA.

12.2.2 Microsoft Access 2007

1. Open or create your database.
2. Edit or create a new form (or report) by clicking *Create* ►*Form Design* (or *Report Design).*
3. Click *View Code,* Microsoft Visual Basic opens.
4. Click *Tools* ► *References* and check *TFORMer 5.1 Runtime Type Library* to insert a reference to **TFORMer Runtime.**
5. Click *OK.*
6. **TFORMer Runtime** may now be used with VBA.

12.2.3 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. The Visual Basic editor opens.
4. Click *Tools* ► *References* and check *TFORMer 5.1 Runtime Type Library* to insert a reference to **TFORMer Runtime.**
5. Click *OK.*
6. **TFORMer Runtime** may now be used with VBA.

12.2.4 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. The Visual Basic editor opens.
7. Click *Tools* ► *References* and check *TFORMer 5.1 Runtime Type Library* to insert a reference to **TFORMer Runtime.**
8. Click *OK.*
9. **TFORMer Runtime** may now be used with VBA.

12.2.5 Other Microsoft Office versions

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

12.3 More Information

Check out the following documents or sample applications for more details:

- COM Programming Reference
  Start Menu ► *All Programs* ► *TEC-IT TFORMer 5.1* ►*Documentation* ► *COM Programming Reference*
- Sample Microsoft Excel Spreadsheet using **TFORMer Runtime**
  Start Menu ► *All Programs* ► *TEC-IT TFORMer 5.1* ► *Examples* ► *Microsoft Office*

13 TFORMer in Web Applications

13.1 TFORMer Runtime with Windows based Web-Applications

13.1.1 Client-side use of TFORMer Runtime

Printing from within a web-browser is usually only possible after the user confirms the operation by selecting a target printer (this is due to browser implementation details). You can avoid this limitation by using the COM based **TFORMer Runtime**. It enables your web-based application to print to arbitrary printers without any user interaction.

These are the steps to use **TFORMer Runtime** within web-pages on client-side:

1. **TFORMer Runtime** has to be installed on each client. Take care that only Microsoft Internet Explorer is supported. A CAB file for automatic installation is available from TEC-IT.
2. Use VBScript® or JavaScript in a web-page which instantiates **TFORMer Runtime** (a sample using VBScript is installed by the setup, check out the following menu entry:
   Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ► Examples ► 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 13.1.3).
4. The web-page provides data to **TFORMer Runtime** programmatically.
5. You web-application prints to local printers without any extra user confirmation.

13.1.2 TFORMer Runtime on a Windows-based Web-Server

TFORMer Runtime 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 Runtime** with server-based web-applications. Check out the following menu entry:

Start Menu ► All Programs ► TEC-IT TFORMer 5.1 ► Examples ► Programming

13.1.3 Web-Based Form Layouts

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

13.2 TFORMer Runtime on Linux/UNIX based Servers

TFORMer Runtime 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).

TFORMer Runtime 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/TFORMer/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.

14 FAQ

14.1 Linux/UNIX

14.1.1 TFORMer does not work

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

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

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

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

14.1.2 TFORMer Runtime does not prints text (32 bit TFORMer on 64-bit systems)

The output generated by **TFORMer Runtime** 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.

14.1.3 TFORMer Runtime does not prints Umlauts (e.g. ÄÖÜäöüß)

If you are using a text file for data import (e.g. with *tfprint*) and this file was generated on Windows, then **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 Runtime** uses UTF-8 internally on Linux/Unix platforms):

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

14.1.4 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.

14.1.4.1 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 users 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 **TFORMer Runtime**.

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

14.1.5 The Library libTFORMer5 cannot be found

Make sure the shared library *libTFORMer5.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*).

14.1.6 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*).

14.1.7 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.
```

This warning message results in uncompressed PDF files.

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

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

This is a known bug of **TFORMer Runtime** on AIX. Currently there is no solution available.

14.1.9 Mono cannot find TFORMerNet assembly

Mono exits with the following error:

```
** (Program.exe:3813): WARNING **: The following assembly referenced from Program.exe
could not be loaded:
     Assembly:   TECIT.TFORMer.TFORMerNet    (assemblyref_index=1)
     Version:    5.0.0.0
     Public Key: 1b5f4306b234b83d
The assembly was not found in the Global Assembly Cache, a path listed in the MONO PATH
environment variable, or in the location of the executing assembly.
```

Solution: Export MONO_PATH to include the directory containing the TFORMerNet bindings:

$ export MONO_PATH=/usr/local/lib

15 Contact and Support Information

15.1 Free Support

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

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

15.2 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: http://www.tec-it.com/order/
Fax: +43 / (0)7252 / 72 72 0 – 77

15.3 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.

15.4 Company Contact Information

TEC-IT Datenverarbeitung GmbH

Address: Wagnerstr. 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: http://www.tec-it.com

Appendix A: Passing Data to TFORMer Runtime

A.1 General

During runtime your application supplies data to **TFORMer Runtime** in order to provide current values for the data-fields used in a form layout.

For this purpose three general methods are supported:

- Imported data-file
  Pass data to **TFORMer Runtime** via a flat text-file (e.g. CSV) or via XML files. The software developer takes care to create these files with current data before printing the required form layouts.
- ODBC
  Pass data to **TFORMer Runtime** via an ODBC database.
- API (Application Programming Interface)
  Pass data to **TFORMer Runtime** via method or function calls. This method is not available for the command line interface (tfprint).

A.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.

A.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>
  <ImportedVariables/>
  </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 data-fields 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 data-field name (attribute “Name”) and the actual data-field value.
DataFieldName	Name of the data-field (must be unique within a record).
Value	The actual value of the data-field.
ImportedVariables	Used internally by TFORMer.

A.2.2 Generic XML-Format

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

A.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 data-field. The element must not necessarily be named “Record”. You can choose a name freely.
V	Data-Element. This element consists of the data-field name (attribute “Name”) and the actual data-field value.
DataFieldName	Name of the data-field (must be unique within a record).
Value	The actual value of the data-field.

► The elements can be named arbitrarily. This means the name “Root“, “Records“ and “Record“ can be selected at will. Only the order of the nodes is decisive. **TFORMer** interprets the lowest level of the XML-format as variable definitions and/or data-fields. 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>
```

A.2.2.2 Variant 2

Same as variant 1, but instead of using the syntax “<V Name="DataFieldName">Value</V>” you can specify data-field 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>
```

A.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>
```

A.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 a text string).

A.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 data-fields).
- The following lines contain the data-field 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 c*olumn separators, line separators* and *text qualifiers:*

A.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

A.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).

A.3.1.3 Valid Text Qualifiers

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

All characters which are enclosed with a text qualifier are imported as one single string. 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.

A.4 Important Notes

A.4.1.1 Empty or Missing Data-Field values

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

A.4.1.2 Too much Data-Fields are given in an Import File

TFORMer Runtime ignores data-fields not defined in the form design.

A.4.1.3 Escape Sequences like \n (newline) are used in the Data-Field values

Starting with **TFORMer** Version 5 such 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 B: Configuration file TFORMer.xml

B.1 General

TFORMer Runtime 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.

B.2 Location of TFORMer.xml

When using a Windows operating system *TFORMer.xml* resides in the same directory as the **TFORMer Runtime** files. Usually this directory is:

```
C:/Program Files/TEC-IT/TFormer5/Bin
```

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

```
/usr/local/share/TFORMer
```

B.3 Configuration File TFORMer.xml

The following settings can be configured:

- Common options
  - Error handling for barcodes, images and text-elements with expressions
- 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)
- Paper format names (for trays)

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

```
<TFORMer major="1" minor="0">

  <!-- (c) 1998-2007 TEC-IT Datenverarbeitung GmbH                              -->
  <!-- htpp://www.tec-it.com                                                    -->
  <!-- support@tec-it.com                                                       -->

  <!-- This TFORMer configuration file contains standard settings for           -->
  <!-- PDF/PostScript and HTML Output                                           -->
  <!-- Edit this file if you want to change PDF/PS/HTML specific output options -->
  <!-- or if you need to use custom form sizes                                  -->

  <!-- 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"/>
</COMMON>

<!-- PDF OPTIONS - Configure settings for PDF output                             -->
<!-- form               Default page size, must be listed in <Forms> below      -->
<!-- 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-type3-fonts  Embed all fonts as type3 fonts which are sometimes     -->
<!--                    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 form="A4" orientation="portrait" compression="1" margin-top="0" margin-left="0"
     margin-right="0" margin-bottom="0" embed-fonts="1"
     embed-type3-fonts="0" 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>
</PDF>

<!-- POSTSCRIPT OPTIONS: Configure settings for PostScript output               -->
<!-- form               Default page size, must be listed in <Forms> below      -->
<!-- orientation        Default orientation "landscape|portrait"                -->
<!-- colormode          color|grayscale|blackwhite                              -->
<!-- Header             Path to the PostScript header file                      -->
<!-- Footer             Path to the PostScript footer file                      -->
<!-- Encoding           Only relevant for Unix systems:                         -->
<!--                    The default encoding of your system                     -->
<!--                    UTF-8 is a good choice for Linux                        -->
<!--                    Latin1 is a good idea for AIX                           -->

<POSTSCRIPT form="A4" orientation="portrait" colormode="color" >
    <Header filename="header.ps" />
    <Footer filename="footer.ps" />
    <Encoding system="UTF-8" />
</POSTSCRIPT>

<!-- HTML OPTIONS:      Configure settings for HTML output                      -->
<!-- form               Default page size, must be listed in <Forms> below      -->
<!-- orientation        Default orientation "landscape|portrait"                -->
<!-- 96dpi - this should be used as default resolution for Microsoft Windows    -->
<!-- 108dpi - seems to work on Linux machines quite good (depends on X settings)-->
<!-- 72dpi - 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 form="A4" orientation="portrait" resolution="96" barcode-opt-resolution="0"
  always-print-barcodes="1" />

<!-- IMAGE OPTIONS:       Configure settings for image output                  -->
<!-- form                 Default page size, must be listed in <Forms> below   -->
<!-- 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,     -->
<!--                      jpeg6, 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 form="A4" orientation="portrait" resolution="96" anti-aliasing="1"
  jpeg-quality="80" tiff-compression="nocomp" colormode="color" dither-mode="0" />

<!-- ZEBRA OPTIONS:       Configure options for Zebra/ZPL output               -->
<!-- form                 Default page size, must be listed in <Forms> below   -->
<!-- orientation          Default orientation "landscape|portrait"             -->
<!-- compression          Image compression for graphics embedded in ZPL       -->
<!--                      'hex' = no compression, data is only hex encoded     -->
<!--                      'rle' = data is run-length-encoded                   -->
<!--                      'png' = data is png and base64 encoded               -->
<!-- don't 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                     -->
<!-- Header               custom ZPL code that is printed before each label    -->

<ZEBRA form="A4" orientation="portrait" compression="rle" scaling="0"
  dont-dither-text="1" dither-mode="0" resolution="203">
  <Header>^MMT</Header>
</ZEBRA>

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

 <Forms>
  <Form name="A0"           width="841000"  height="1189000" />
  <Form name="A1"           width="594000"  height="841000"  />
  <Form name="A2"           width="420000"  height="594000"  />
  <Form name="A3"           width="297000"  height="420000"  />
  <Form name="A4"           width="210000"  height="297000"  />
  <Form name="A5"           width="148000"  height="210000"  />
  <Form name="A6"           width="105000"  height="148000"  />
  <Form name="A7"           width="74000"   height="105000"  />
  <Form name="A8"           width="52000"   height="74000"   />
  <Form name="A9"           width="37000"   height="52000"   />
  <Form name="A10"          width="26000"   height="37000"   />
  <Form name="B0"           width="1000000" height="1414000" />
  <Form name="B1"           width="707000"  height="1000000" />
  <Form name="B2"           width="500000"  height="707000"  />
```

```
    <Form name="B3"           width="353000"  height="500000"  />
    <Form name="B4"           width="250000"  height="353000"  />
    <Form name="B5"           width="176000"  height="250000"  />
    <Form name="B6"           width="125000"  height="176000"  />
    <Form name="B7"           width="88000"   height="125000"  />
    <Form name="B8"           width="62000"   height="88000"   />
    <Form name="B9"           width="44000"   height="62000"   />
    <Form name="B10"          width="31000"   height="44000"   />
    <Form name="C0"           width="917000"  height="1297000" />
    <Form name="C1"           width="648000"  height="917000"  />
    <Form name="C2"           width="458000"  height="648000"  />
    <Form name="C3"           width="324000"  height="458000"  />
    <Form name="C4"           width="229000"  height="324000"  />
    <Form name="C5"           width="162000"  height="229000"  />
    <Form name="C6"           width="114000"  height="162000"  />
    <Form name="C7"           width="81000"   height="114000"  />
    <Form name="C8"           width="57000"   height="81000"   />
    <Form name="C9"           width="40000"   height="57000"   />
    <Form name="C10"          width="28000"   height="40000"   />
    <Form name="D0"           width="771000"  height="1091000" />
    <Form name="D1"           width="545000"  height="771000"  />
    <Form name="D2"           width="385000"  height="545000"  />
    <Form name="D3"           width="272000"  height="385000"  />
    <Form name="D4"           width="192000"  height="272000"  />
    <Form name="D5"           width="136000"  height="192000"  />
    <Form name="D6"           width="96000"   height="136000"  />
    <Form name="D7"           width="68000"   height="96000"   />
    <Form name="E0"           width="800000"  height="1120000" />
    <Form name="E1"           width="560000"  height="800000"  />
    <Form name="E2"           width="400000"  height="560000"  />
    <Form name="E3"           width="280000"  height="400000"  />
    <Form name="E4"           width="200000"  height="280000"  />
    <Form name="E5"           width="140000"  height="200000"  />
    <Form name="E6"           width="100000"  height="140000"  />
    <Form name="E7"           width="70000"   height="100000"  />
    <Form name="B0JIS"        width="1030000" height="1456000" />
    <Form name="B1JIS"        width="728000"  height="1030000" />
    <Form name="B2JIS"        width="515000"  height="728000"  />
    <Form name="B3JIS"        width="364000"  height="515000"  />
    <Form name="B4JIS"        width="257000"  height="364000"  />
    <Form name="B5JIS"        width="128000"  height="257000"  />
    <Form name="B6JIS"        width="128000"  height="182000"  />
    <Form name="B7JIS"        width="91000"   height="128000"  />
    <Form name="B8JIS"        width="64000"   height="91000"   />
    <Form name="B9JIS"        width="45000"   height="64000"   />
    <Form name="B10JIS"       width="32000"   height="45000"   />
    <Form name="Invoice"      width="140000"  height="216000"  />
    <Form name="Executive"    width="191000"  height="254000"  />
    <Form name="Legal"        width="216000"  height="356000"  />
    <Form name="Letter"       width="216000"  height="279000"  />
    <Form name="Ledger"       width="432000"  height="279000"  />
    <Form name="Tabloid"      width="279000"  height="432000"  />
    <Form name="Broadsheet"   width="432000"  height="559000"  />
    <Form name="Screen"       width="297000"  height="210000"  />
    <Form name="Custom"       width="210000"  height="297000"  />
    <Form name="Comm10"       width="105000"  height="241000"  />
    <Form name="DL"           width="110000"  height="220000"  />
    <Form name="Folio"        width="210000"  height="330000"  />
    <Form name="P1"           width="560000"  height="860000"  />
    <Form name="P2"           width="430000"  height="560000"  />
    <Form name="P3"           width="280000"  height="430000"  />
    <Form name="P4"           width="215000"  height="280000"  />
    <Form name="P5"           width="140000"  height="215000"  />
    <Form name="P6"           width="107000"  height="140000"  />

    <!-- add custom forms below this point -->

  </Forms>
</TFORMer>
```

Appendix C: Terms and Definitions

C.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 data-source and is used in the document layout via data-fields.

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

C.2 Stand-Alone Form

A stand-alone form is a document, which is stored as single file without being referred by a repository. **TFORMer Designer** normally creates stand-alone forms, but it can be instructed to add the form to a repository. Stand-alone forms are using the file extension *.tff.

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

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

C.3 Repository

A repository is a central location in the file-system and holds references to form layouts. A repository also holds the definitions of so-called projects and data-fields.

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 data-fields. 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 document layout.

C.4 Project

A repository is divided into so-called projects. Each project holds definitions of data-fields and one or more form layouts. All form layouts using the same data-fields should be grouped in a project.

C.5 Data-Field

A form layout may use data-fields as place-holders for dynamic data. The content of this place-holder is provided during runtime (= print-time) by external data-sources or by the user. At runtime each occurrence of a data-field is replaced with the current data. Data-fields may be used as part of expressions.

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

In addition to such project-specific data-fields **TFORMer** supports global data-fields: these data-fields are available for all document layouts in all projects.

C.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:

```
' Windows 2000 and Windows XP

C:\Documents and Settings\All Users\Application Data\TEC-IT\TFORMer\5.1\Templates

' Windows VISTA

C:\Program Data\TEC-IT\TFORMer\5.1\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!

C.7 Data-Source

Dynamic data is provided by external data-sources like in-memory data-sources, ODBC databases, text-files, XML-files or user input. Whenever printing a form, **TFORMer Designer** imports the data from the data-source 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 data-source. In this case data is not stored by **TFORMer**).

Appendix D: Document History

Date (yy-mm-dd)	User	Description
06-06-06	TEC-IT	Release of TFORMer 4.5.1
07-06-06	gkriegl	Updated Installation hints for AIX 4.3
15-05-07	gkriegl	Updated to V5
15-11-07	gkriegl	Updated to V5.1