Document Preparation with Mark IV ConTEXt

by wiki.contextgarden.net
version 1
December 2007

Preface

This document describes the basics of the ConTEXt document preparation system. ConTEXT is a system for creating high quality print documents. This document merely introduces techniques to start creating basic documents. For a more into depth description, the reader is referred to the ConTEXT manuals at www.pragma-ade.com. The first part of this document until chapter 9 describes basic text document preparation, the second part introduces floats and the final part describes ConTEXt modules.

This document is for the ConTEXt document preparation system.

Keywords: ConTEXt, TEX, MetaFun, METAPOST.

Published by wiki.contextgarden.net
http://wiki.contextgarden.net
ftp://wiki.contextgarden.net
mailto:info@wiki.contextgarden.net
sip:100@wiki.contextgarden.net

This document was typeset using the ConTEXt macro package for TEX using the pdfTEX engine. ConTEXt was written by Hans Hagen et al. and is distributed by Pragma Advanced Document Engineering (Pragma ADE), TEX by Donald E. Knuth, and pdfTEX by Han The Thanh. Additional macros were provided by Taco Hoekwater (the bibTEX bibliography module for ConTEXt). Figures were rendered using METAPOST and MetaFun, the graphic extensions to the typeface rendered METAFONT by Donald E. Knuth, written by John. D. Hobby and Hans Hagen, respectively. This document has been typeset in the Computer Modern Roman (CMR) typeface family, also due to Donald E. Knuth.

Table of Contents

Chapter 1

Introduction to ConTEXt

1.1 Introduction

ConTEXt is a document preparation system written in a TEX macro package with Perl and Ruby. ConTEXt is written by Hans Hagen and Ton Otten of Pragma Advanced Document Engineering (Pragma ADE) in the Netherlands. Unlike LaTEX, ConTEXt is monolithic, i.e. it is designed, implemented and distributed as a whole. ConTEXt has a native XML parser, and MathML support. It also has integrated graphics with METAPOST, called MetaFun. Normally, one creates a ConTEXt document in a text editor and then uses the ConTEXt software to convert it to device independent (DVI), portable document format (PDF) or postscript (pos) format. ConTEXt was developed for typesetting and production of documents, ranging from simple books to very complex and advanced technical manuals and study books, either electronic or printed. ConTEXt was developed for non-technical users in the WYSIWYG area. Primary goals are a user friendly interface and easy commands.

1.2 Mark IV ConTEXt

Mark IV is the name of the LuaTeX-aware part of ConTEXt, when you use ConTEXt over the LuaTeX engine in stead of PDFTeX or XeTeX.

1.3 Literature

These official Pragma ADE ConTEXt publications can be found on their website.

- Hans Hagen and Ton Otten, *ConTEXt an Excursion*, Hasselt, Pragma ADE, 1999.
- Hans Hagen, *ConTEXt the Manual*, Hasselt, Pragma ADE, 2001.

Various publications have described ConTEXt as well.

- Berend de Boer, *LaTEX in proper ConTEXt*, 2000.
- Karel Wesseling, *A do-it-yourself thebibliography in ConTEXt*, MAPS, 2002.

1.4 Websites

The following websites provide valuable information about ConTEXt.

- http://www.pragma-ade.com
- http://wiki.contextgarden.net
- http://www.ntg.nl/context

Chapter 2

Installing ConTEXt

2.1 Installation on MS Windows

The easiest way to install ConTEXt is first to install ProTEXt and then ConTEXt. ProTEXt includes MikTeX, TEXnicCenter, GhostScript and GSview. Alternatively, install the ConTEXt editing environment and ConTEXt, both can be downloaded from the www.pragma-ade.com website.

2.2 ConTEXt Utilities

2.2.1 The SciTE Text Editor

The Scintilla text editor (SciTE) is the source code editor that is used for standalone ConTEXt. Pragma ADE has added to SciTE much support for ConTEXt, for example the command completion feature, which will show alternative commands when entering the first couple of letters of a command.

2.2.2 The Texexec Utility

The texexec utility is a Perl script that is the front end for ConTEXt.

2.2.3 The Texutil Utility

Texutil

2.2.4 The Texfont Utility

Texutil

2.2.5 The Texhow Utility

Texutil

2.3 Installing the ProTEXt Suite with ConTEXt

ProTEXt is a suite of MikTEX,TEXnicCenter, GhostScript, and GSview.

2.4 Installing the Standalone ConTEXt Distribution on MS Windows

2.4.1 Installing Adobe Acrobat Reader

First, you need to install Adobe Acrobat Reader Standard version 8.1.1, which is the latest version.

1. Log on to the Internet, open Internet Explorer and go to the website 'http://www.abobe.com/products/acrobat/readstep2.html'.
2. Unselect the 'Google Toolbar' checkbox and select the 'download now' button. If the download does not start, select 'click here to download'.
3. A 'Save as' window will popup, save the program on 'Desktop'.
4. When the program has been downloaded, double click on the program on the Desktop to install it.
5. The Adobe download manager will be started, when all files have been downloaded, select the 'Install Now' button.
6. Choose the destination location and select 'Next'.
7. To start installation, select 'Install'.
8. An icon will appear on the Desktop entitled 'Acrobat Reader 8.1', click on the icon to launch the program.

2.4.2 Installing Perl

Next, you need to install the Perl scripting language.

1. Log on to the Internet, open Internet Explorer and go to the website 'http://www.activestate.com/downloads' and download the current ActivePerl Windows MSI package on your Desktop.
2. Double click on the ActivePerl MSI installation file and follow any instruction that it shows. Use any default settings it may ask you. If it asks you to reboot the computer, click 'Yes'.
3. If you have an older version of MS Windows, you may get an error warning when you try to open the MSI file saying that a newer version of the 'Windows Installer' is required. If so, download and install 'Windows Installer 2.0+', using one of the following packages.
   - Windows 9x/ME: 'instmsia.exe'
   - Windows NT/2K: 'instmsiw.exe'
   - Windows XP comes with the Windows Installer.
4. ActivePerl is now installed and you should be able to run Perl script on the PC.
5. If you want to test and see if Perl is working, select [START]>[RUN]>[CMD] and enter the following in the window with the command line interface

   ```
   perl -v
   ```

9. Press [ENTER] and you should see the text 'This is perl,…'. This message tells you that Perl is installed properly.

2.4.3 Installing Ruby

Next, you need to install the Ruby scripting language.

1. Log on to the Internet, open Internet Explorer and go to the website 'http://www.ruby-lang.org/en/downloads/', go to the 'Ruby on Windows' section, select the 'one-click installer', then the 'one-click self-contained Windows Installer' and download the current package on your Desktop.
2. Double click on the Ruby installation file and follow any instruction that it shows.

2.4.4 Installing ConTEXt

1. Create on your PC the 'c:\context' directory. Any location will work to store ConTEXt, but do not use a space in the file path, such as for example 'c:\program files\context'.
2. Download the file 'mswincontext.zip' from the Pragma ADE website 'www.pragma-ade/context/install/mswincontext.zip' and put it in the 'c:\context' directory.
3. Open the 'zip' file, there should be a folder 'usr' and a few files in it.
4. Unzip the file and put the contents of the 'isoimage' folder in the 'c:\contect' directory, so you have the folder 'c:\contect\usr'.
5. Open the folder 'c:\context' and double-click on the file 'cdemo.bat'.
6. The SciTE editor will open a few files. Press <F7> to compile one of the files.
7. Create a shortcut on the desktop by clicking the right mouse button on the desktop and selecting [NEW]>[SHORTCUT] and then select the file 'c:\context\cstart.bat'. Do not move the 'cstart.bat' file out of the ConTEXt directory.
8. When a desktop shortcut to ConTEXt has been created, start ConTEXT with the shortcut.
9. Now, the formats need to be build, select [START]>[RUN]>[CMD] and enter the following in the window with the command line interface

   ```
   cd c:\context
   cdemo.bat
   ```

10. Close the SciTE window but stay in the command line interface.
11. Enter the following in one line in the command line interface

    ```
    c:\context\usr\local\context\tex\setuptext.bat c:\context \usr\local\context\tex
    ```

12. And enter the following in the command line interface

    ```
    texexec --make --all
    ```

13. Close the command line interface and double click the desktop shortcut to the 'cstart.bat' file to start ConTEXt.

14. Alternatively, you can enter the texexec command directly in the SciTE console.

    ```
    texexec --make --all
    ```

2.5 Installing True Type Fonts

ConTEXt font installation is completely independent from the operating system fonts. Though, in most cases it is possible to install the operating system fonts for ConTEXt to use. First you need to acquire the true type font, which has the file extension 'ttf'. The easiest way is to download the Windows font files from the foundry.

Chapter 3

Command Syntax

3.1 ConTEXt Commands and Command Options

The syntax of ConTEXt commands differs significantly from dialects such as plain TEX or LaTEX. ConTEXt commands will often specify parameters to the command as command arguments in a square bracket pair `[]`. In case a command requires an argument that will produce typeset text, it should be specified in a curly brace pair `{}`. A general command syntax could look as following.

```
\command[..,..]{...}
```

In formal command descriptions, the required arguments shall be shown in a regular square bracket pair `[]`, and an optional argument shall be shown in a slanted square bracket pair *`[]`*. Parameters in a square bracket pair are mostly optional to use whereas parameters in curly braces are always required. When specifying options in ConTEXt, be aware that there are three different kinds of command options available to put in a square bracket pair.

1. Keyword options: these are predefined words that have a special meaning to ConTEXt. If you enter an unknown keyword, ConTEXt shall simply ignore it without warning. An example is the keyword 'packed' that can be used in itemizations.
2. Assignment options: these always have the following format: 'name=value'. In this case, name is another keyword, and value could also be a keyword but not necessarily. An example is 'align=left' (keyword=keyword), where both name and value are keywords. Another example is 'width=2.3cm' (keyword=value), where 'name' is a keyword, but 'value' is not.
3. Implicit options: in many cases ConTEXt is intelligent enough to understand the meaning of an option without explicit specifications. For example the header command accepts an option in brackets that defines a label that you can refer to.

In many cases multiple keywords and/or multiple assignments can be specified. Lists of keywords or assignments are always separated by commas. The order in which such list items are entered is not significant, for example.

```
\startitemize[a,packed][width=10mm,textstyle=bold]
```

The `\startitemize` command has the following general syntax.

```
\startitemize[..,..][..,..=..,..] ... \stopitemize
```

Table 3.1 shows some of the `\startitemize` command options.

Arg	Option	Description
1	n	Item separator is 1, 2, 3, 4...
	r	Item separator is i, ii, iii, iv, iv...
	standard	
	number	
	broad	Additional whitespace shall be put between line items
	packed	No whitespace shall be put between line items

Arg	Option	Value	Description
2	width	mm	The width in millimeter
		pt	The width in points
	textstyle	bold	A bold textstyle
		slanted	A slanted textstyle

TABLE 3.1 – Some \startitemize Command Options.

The 'arg' column indicates for which argument these keyword options are available and the 'option' and 'value' columns give some available command options. Since both square bracket pairs are slanted, for this command, both command options are optional and can be left out. Default options and assignments are underlined, as is the case with the keyword 'standard'. Slanted values you can set the value for yourself, as is the case with the keyword 'number'. Below are a few imaginary commands to clarify this.

```
\command[option1,option2]
\anothercommand[required]
\yetanothercommand[required][option1=x,option2=y]
\nextcommand[option]{required}
```

The command `\acommand` can be used without arguments, whereas the command `\anothercommand` requires a parameter in a square bracket pair. The command `\yetanothercommand` requires a parameter in square brackets and it optionally accepts another parameter, also in square brackets. The command `\nextcommand` requires a parameter in curly braces, optionally preceded by a parameter in square brackets.

3.2 ConTEXt Setup Commands

ConTEXt setup commands are mostly placed in the setup area of the document, which is the area before the `\starttext` command. These commands have a global effect, but you also use them mid-document. The setup commands all have the same structure, we look at the `\setuphead` command as an example.

```
\setuphead[chapter][style=slanted]
```

The `\setuphead` command has the following general syntax

```
\setuphead[...][..,..=..,..]
```

Table 3.2 shows some of the `\setuphead` command options.

Arg	Option	Description
1	*name*	A header name such as chapter, section, etc.
	number	An integer
	each	

Arg	Option	Value	Description
2	style	<u>normal</u>	
		bold	
		slanted	
		boldslanted	
		type	
		cap	
		small	
		command	
	align	left	
		right	
		middle	
		<u>width</u>	
	tolerance	veryrigged	
		rigged	
		<u>tolerant</u>	
		verytolerent	
	Line	on	
		<u>off</u>	
	width	*dimension*	A size with a unit in cm, mm, in, pt, sp, em or ex
	height	*dimension*	A size with a unit in cm, mm, in, pt, sp, em or ex
	distance	*dimension*	A size with a unit in cm, mm, in, pt, sp, em or ex
	before	*command*	
	after	*command*	
	inner	*command*	
	command		

TABLE 3.2 – Some \setuphead Command Options.

A typical setup command consists of a more or less logical name and a number of square bracket pairs.

```
\setupcommand[implicit option][keyword options][assignment options]
```

The default keyword options and assignment options are <u>underlined</u>. Furthermore, you will notice that some values are typeset in italics: *section, name, dimension, number, command* and *text.* This indicates that you can set the value yourself.

Option	**Description**
section	A header name such as ‘chapter’, ‘section’, or ‘paragraph’
name	An identifier (logical name)
dimension	A size with a unit in cm, mm, in, pt, sp, em or ex
number	An integer
command	A command
text	Text

TABLE 3.3 – Some Setup Command Options.

3.3 ConTEXt Input Files

A ConTEXt document has a core and an optional setup area before the core. The setup part is similar to the preamble in LaTEX. ConTEXt markup is placed with ASCII flat text format in an input file with file extension `.tex`, for example `myfile.tex`. All ConTEXt commands start with a backslash (`\`). This backslash and the command combination, for example `\starttext` is an instruction for ConTEXt. The command itself is not shown in the compiled text. The content of any ConTEXt document is placed between the `\starttext` and `\stoptext` command pair and this textual input is subject to marco expansion during code compiling. The area before the `\starttext` command is the setup area. The following code fragment is a ConTEXt input file example named `myfile.tex` with only a core.

```
\starttext
Hello, World!
\stoptext
```

When compiled, after macro expansion it will show the text `Hello, World!` in the document. If the output file needs to be a portable document format (PDF) after input file compilation, we add instructions in the setup area. The setup area and core structure of the input file `myfile.tex` could for example look as following.

```
output=pdftex

\starttext
Hello, World!
\stoptext
```

Except for texexec utility settings, the setup area can also contain regular ConTEXt commands as the following example illustrates.

```
output=pdftex
\setupbodyfont[12pt]

\starttext
Hello, World!
\stoptext
```

Even though this is a simple example, all ConTEXt input files have a similar structure. Any comments in the input file which are commented with the percent symbol (%) will be ignored and these lines shall not be processed during the compiling process and are not subject to macro expansion.

```
%This document has no setup area
\starttext
Hello, World!
\stoptext
%This is the end of the ConTEXt document
```

In case a command requires clarification, it is possible to put the percent symbol after the command.

```
\starttext %start of document
Hello, World! %the text ‘Hello, World!’ shall be rendered in the document
\stoptext %end of document
```

Just like with plain TEX, there is no need to keep everything in one file. Using the `\input` command, the document can be split into separate files.

```
\input myfile
```

This way you can keep certain parts or chapters in separate files. If you only specify a file name without file extension, the file extension `.tex` will be assumed. In case the file has not the `.tex` file extension, you need to specify it.

```
\input myfile.txd
```

In case you want to specify a full path, you should use forward slashes (/) and not backslashes (\), for example.

```
\input /tex/styles/myfile.ctx
```

Forward slashes are also used in MS Windows.

3.2 ConTEXt Command Line Compilation

The ConTEXt input file `myfile.tex` is compiled as following with the texexec utility.

```
texexec myfile.tex
```

The texexec command compiles the table of contents, indexes, references and sorted lists. The texexec utility recompiles automatically to update the references. This is called a two-pass, three-pass or multi-pass job. ConTEXt supports various language specific interfaces such as English, German, and Dutch. This enables users to work with ConTEXT in their own language. We shall use the English version. To run ConTEXt with the English interface, enter

```
texexec –interface=en myfile.tex
```

It is not required to enter the `.tex` file extension when using texexec.

```
texexec –interface=en myfile
```

3.1.3 ConTEXt Graphical User Interface Compilation

Text

3.1.4 ConTEXt Output Files

Normally, the ConTEXt output format is a `.dvi` file. To change the ConTEXt output format to PDF format, enter

```
texexec –output=pdftex myfile.tex
```

Or shorter

```
texexec –-pdf myfile.tex
```

Alternatively, you can put these parameters in the setup area of the input file, so you do not need to enter them upon compiling.

```
interface=en
output=pdftex

\starttext
Hello, World!
\stoptext
```

ConTEXt can also create various other document types, such as for example PostScript (ps).

Chapter 4

Document Structure

4.1 ConTEXT Documents with Headers

Most documents require some type of header, either numbered or not. ConTEXt has various commands available to provide for headers and sub-headers.

4.2 A ConTEXt Document with Chapters and Sections

ConTEXt creates chapters with the `\chapter` command. The text that is put inside the curly braces `{}` is the chapter title that will appear.

```
\starttext
\chapter{First Chapter Title}
This is the text in the first chapter.
\chapter{Second Chapter Title}
This is the text in the second chapter.
\stoptext
```

The created document will automatically get sequential chapter numbers and each chapter will start on a new page. Similar to chapters, sections and subsections can be created with other header commands as shown in table 4.1.

Numbered Header	Numberless Header
\chapter	\title
\section	\subject
\subsection	\subsubject
\subsubsection	\subsubsubject
…	…

TABLE 4.1 – ConTEXt Header Commands.

These commands will produce a header with or without a number in a predefined font size and font type with an amount of vertical spacing before and after the header.

```
\chapter{This is the First Chapter Title}
```

These header commands have the following general syntax.

```
\chapter[...,...]{...}
```

The command has two arguments. The first optional argument is a square bracket pair and the second argument an argument in a curly braces pair. The curly braces contain the text that will be rendered in the document as the chapter title. The square bracket pair is optional and can be used for internal references elsewhere in the document.

```
\chapter[first-chapter]{This is the First Chapter Title}
```

If somewhere else in the document you want to refer to this header, you type for example

```
\on{page}[first-chapter]
```

Or, alternatively

```
\at{page}[first-chapter]
```

An input file with numbered chapters and sections would look like this.

```
\starttext
\chapter{First Chapter Title}
\section{First Chapter First Section Title}
This is the text in the first section of the first chapter.
\section{First Chapter Second Section Title}
This is the text in the second section of the first chapter.
\chapter{Second Chapter Title}
\section{Second Chapter First Section Title}
This is the text in the first section of the second chapter.
\section{Second Chapter Second Section Title}
This is the text in the second section of the second chapter.
\stoptext
```

4.3 A ConTEXt Document with a Table of Contents

The command `\placecontent` creates a table of contents on the same page.

```
\starttext
\placecontent
\chapter{First Chapter Title}
This is the text in the first chapter.
\chapter{Second Chapter Title}
This is the text in the second chapter.
\stoptext
```

The command `\completecontent` creates a table of contents on a new page.

```
\starttext
\completecontent
\chapter{First Chapter Title}
This is the text in the first chapter.
\chapter{Second Chapter Title}
This is the text in the second chapter.
\stoptext
```

The chapter numbers assigned to each chapter in the document will automatically match the corresponding chapter numbers in the table of contents that ConTEXt creates with either the `\placecontent` or the `\completecontent` command.

4.4 Changing the Header Style

Look at the following `\setuphead` command example.

```
\setuphead[chapter][style=bold]
```

This command will provide for chapter headers to have a bold font. The following command will arrange for section headers to be slanted.

```
\setuphead[section][style=slanted]
```

The `\definehead` command together with the `\setuphead` commands can modify the headers as the following example illustrates.

```
\definehead[myheader][section]
\setuphead[myheader]
[numberstyle=bold,
textstyle=cap,
before=\hairline\blank,
after=\nowhitespace\hairline]
\myheader[myhead]{First Section Header Title}
```

The `\setupheads` command is used to set up numbering of the numbered headings.

```
\setupheads[alternative=inmargin,separator=--]
```

With this command, all numbers will appear in the margin, and ‘Section 1.1’ will look like ‘Section 1-1’. The `\setupheads` command should be put in the setup area of the document.

4.5 Conventional Columns

The `\startcolumns` and `\stopcolumns` command pair enclose text that will be put in columns, as the following example illustrates.

```
\startcolumns[n=2,tolerance=verytolerant]
The text that is put inside both the column commands will be rendered in the
document in columns.

The properties of the text can be modified with assignment options.
\stopcolumns
```

The general syntax of the `\startcolumns` and `\stopcolumns` command pair is.

```
\startcolumns[..,..=..,..] . . . \stopcolumns
```

The `\startcolumns` command has a single argument in a square bracket pair that can contain various keys and key values. Table 4.2 shows some of the `\startcolumns` command optional keys and their available key values.

Option	Value	Description
n	1, 2, 3, etc.	the number of columns
rule	on off	
tolerance	verystrict strict tolerant verytolerent stretch	

TABLE 4.2 – Some \startcolumns Command Options.

If necessary, a new column can be forced with the following command.

```
\column
```

Columns can be setup with the following command.

```
\setupcolumns[...]
```

This is a global parameter and is usually put in the setup area of the input file, so all columns in the document will get its properties. It has the same assignment options as the `\startcolumns` command.

4.6 New Style Columns

There are some limitations with the method described above. The column set routine is used for complex layouts with graphics and text spanning columns and even spreads.

4.7 End of Chapter Review

4.7.1 ConTEXt Commands

Command	Description
`\starttext` `\stoptext`	any text between this command pair shall be subject to macro expansion during compiling

`\chapter[..,..]{...}`
`\section[..,..]{...}`
`\subsection[..,..]{...}`
`\subsubsection[..,..]{...}`
	numbered chapter header
	numbered section header
	numbered subsection header
	numbered subsubsection header

`\title[..,..]{...}`
`\subject[..,..]{...}`
`\subsubject[..,..]{...}`
`\subsubsubject[..,..]{...}`
	unnumbered title header
	unnumbered subject header
	unnumbered subsubject header
	unumbered subsubsubject header

`\on{page}[..]`
`\at{page}[..]`
	refer to this header elsewhere in the document
	refer to this header elsewhere in the document

`\placecontent` `\completecontent`	creates a table of contents on the same page creates a table of contents on a new page

4.7.2 ConTEXt Code Example

The following code is an example of a ConTEXt input file in which all commands from this chapter are used.

```
\starttext
\completecontent
\chapter{First Chapter Title}
```

```
\section{First Chapter First Section Title}
This is the text in the first section of the first chapter.
\section{First Chapter Second Section Title}
This is the text in the second section of the first chapter.
\chapter{Second Chapter Title}
\section{Second Chapter First Section Title}
This is the text in the first section of the second chapter.
\section{Second Chapter Second Section Title}
This is the text in the second section of the second chapter.
\subsection{Second Chapter Second Section First Subsection Title}
This is the text in the first subsection of the second section of the first
chapter.
\subsection{Second Chapter Second Section Second Subsection Title}
This is the text in the second subsection of the second section of the first
chapter.
\stoptext
```

Chapter 5

Document Style

5.1 Selecting the Paper Size

The command `\setuppapersize` is used to define both the page size for screen typesetting and paper size for document printing. The following example illustrates an A4 page size and an A3 paper size at landscape orientation.

```
\setuppapersize[A4,landscape][A3,landscape]
```

The general syntax of the `\setuplayout` command is.

```
\setuppapersize[..,..,..][..,..,..]
```

The command has two arguments, both in a square bracket pair, the second command option is optional. Each square bracket pair can contain multiple options. Table 5.1 shows some of the `\setuppapersize` command options.

Option	Description
A3	ISO A3 standard
A4	ISO A4 standard
A5	ISO A5 standard
A6	ISO A6 standard
S6	
letter	U.S. letter standard
legal	U.S. legal standard
portrait	rendered in portrait format
landscape	rendered in landscape format
oversized	adds 1.5 cm on each side
doublesized	double the height
doubleoversized	
mirrored	
rotated	
90	
180	
270	

TABLE 5.1 – Some \setuppapersize Command Options.

The `\definepapersize` command together with the `\setuppapersize` command can modify the paper size as the following example illustrates.

```
\definepapersize[mypapersize][width=8.5in,height=11in]
\setuppapersize[mypapersize][mypapersize]
```

As indicated, after defining the 'mypapersize' option, it can be used just as well as the regular A3, A4, etc. paper sizes for both screen size and paper size.

5.2 Page Header and Footer Texts

The `\setupheadertexts` and `\setupfootertexts` commands are used for creating a header and footer. Both have the following general syntax.

```
\setupheadertexts[...][...][...]
```

```
\setupfootertexts[...][...][...]
```

The commands have three arguments. The first square bracket pair is used to set the location of the header or footer. The header and footer itself are placed within the second square bracket pairs. In a double sided document, the second and third square bracket pairs are used for the header and the footer on the left and right side pages, respectively. Table 5.2 shows some of the command options that can be used at the first square bracket pair.

Arg	Option	Description
1	*text* margin edge	
2	*text* *section* date *mark* pagenumber	
3	*text* *section* date *mark* pagenumber	

TABLE 5.2 – Some \setupheadertexts and \setupfootertexts Command Options.

In many cases you can omit parameters, as in the next example.

```
\setupheadertexts[chapter]
```

In this case the parameter ‘chapter’ is interpreted as an implicit option. This keyword will yield the title of the current chapter on the middle of the page header. This header will change dynamically at every new chapter. The following example illustrates the addition of the book title as well.

```
\setupheadertexts[my Book Title][chapter]
```

In this case the text ‘my Book Title’ will appear on the left of the page header and the title of the current chapter on the right of the page header. You can setup the page header and footer with.

```
\setupheader[...]
```

and

```
\setupheader[...]
```

Table 5.3 shows some of the command options that can be used at the square bracket pair.

Option	Value	Description
style	normal bold slanted	

TABLE 5.3 – Some \setupheader and \setupfooter Command Options.

The following example illustrates its use.

```
\setupheadertexts[style=bold]
```

or

```
\setupheadertexts[style=\ss]
```

If you want to leave out the page header and footer on one particular page, for example the title page, you can use

```
\noheaderandfooterlines
```

5.3 Setting Global Page Numbers

The `\setuppagenumbering` command is used for setting page numbers. The command has the following general syntax.

```
\setuppagenumbering[..,..=..,..]
```

The command has a single argument in a square bracket pair that can contain multiple assignment options and their key values. Table 5.4 shows some of the `\setuppagenumbering` command assignment options.

Option	Value	Description
alternative	singlesided	
	doublesided	
location	header	
	footer	
	left	
	right	
	middle	
	margin	
style	normal	
	bold	
	slanted	
left	*text*	
right	*text*	
text	*text*	
conversion	numbers	
	characters	
	romannumerals	

TABLE 5.4 – Some \setuppagenumbering Command Options.

In order to remove page numbering entirely, set location to blank.

```
\setuppagenumbering[location=]
```

5.4 Changing Local Page Numbering

Page numbering takes place automatically but you can enforce a page number with

```
\page[25]
```

Sometimes it is better to state a relative page number like

```
\page[+2]
```

or

```
\page[-2]
```

5.5 Page Breaking

A page can be enforced or blocked by the page command

```
\page[...]
```

The `\page` command has a single argument in a square bracket pair that can contain a single keyword option. Table 5.5 shows some of the available keyword options.

Option	Description
yes	enforce a page
makeup	enforce a page without filling
no	no page
preference	prefer a new page here
bigpreference	great preference for a new page here
left	next page is a left handside page
right	next page is a right handside page
disable	following commands have no effect
reset	following commands do have effect
empty	insert an empty page
last	add pages until an even number is reached
quadriple	add pages until foursome is reached

TABLE 5.5 – Some \page Command Options.

5.6 The Page Background

The \setupbackgrounds command is used for aaaa. The command has the following general syntax.

```
\setupbackgrounds[...][..,..][..,..=..,..]
```

The command has a single argument in a square bracket pair that can contain multiple keys and key values. Table 5.6 shows some of the \setuppagenumbering command options.

Option	Value	Description

TABLE 5.6 – Some \setupbackgrounds Command Options.

In order to remove page numbering entirely, set location to blank.

5.7 End of Chapter Review

5.7.1 Commands

Command	Description

5.7.2 Code Example

The following code is an example of a ConTEXt input file in which all commands from this chapter are used.

```
\starttext
\stoptext
```

Chapter 6

Page Layout

6.1 Showing the Page Layout

In ConTEXt, a page is divided in a number of areas such as the text, margin, header and footer. The size of each area can be specified. The different areas in the page layout are shown in figure 6.1.

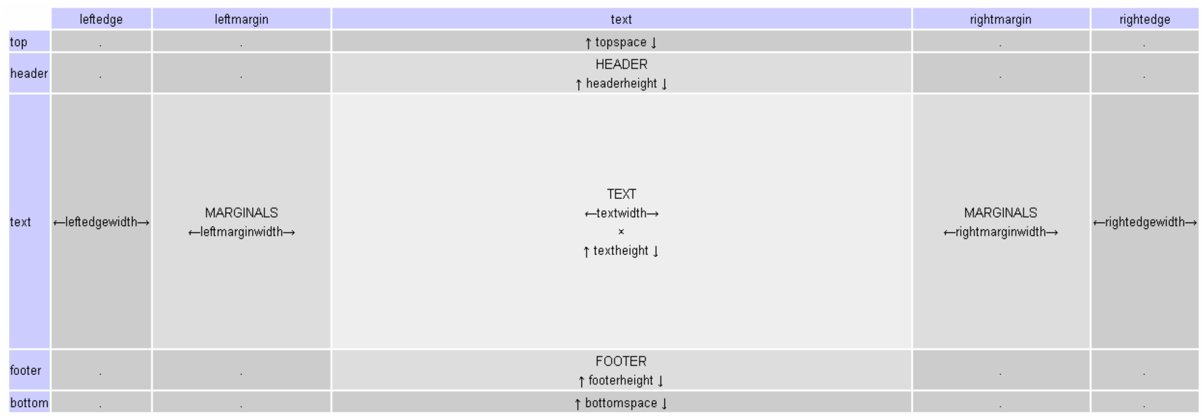


FIGURE 6.1 – The Page Areas of a ConTEXt Document.

If you want ConTEXt to show these visual markers that indicate the page layout, use the `\showframe` command

```
\showframe
```

and process one page or the whole file. The areas are shown in a number of frames. The `\showlayout` command

```
\showlayout
```

show the numerical values of the page layout parameters on the first couple of pages when processed.

6.2 Setting the Global Page Layout

When you want to center the page and add crop marks, add the `\setuplayout` command as illustrated.

```
\setuplayout[location=middle,marking=on]
```

Another example of the `\setuplayout` command is

```
\setuplayout[header=20pt,headerdistance=1cm]
```

The general syntax of the `\setuplayout` command is

```
\setuplayout[..,..=..,..]
```

The `\setuplayout` command should appear in the setup area of the document and has an argument in a square bracket pair that can contain multiple assignment options. Table 6.1 shows some of the `\setuplayout` command assignment options.

Option	Value	Description

width	*dimension* fit middle	
height	*dimension* fit middle	
topspace backspace bottomspace cutspace header headerdistance footer footerdistance margin leftmargindistance rightmargindistance	*dimension* *dimension* *dimension* *dimension* *dimension* *dimension* *dimension* *dimension* *dimension* *dimension* *dimension*	
marking	on of 0pt	add crop marks
location	middle	center the page
Style	bold slanted cap small	
grid	yes no	

TABLE 6.1 – Some \setuplayout Command Options.

6.3 Changing the Local Page Layout

In case you want to make only slight changes to the global page layout, you should use the `\adaptlayout` command as the following example illustrates.

```
\adaptlayout[21,38][height=+.5cm]
```

In this case pages 21 and 38 would have a height of the default height + 0.5 cm. The `\adaptlayout` command has the following general syntax.

```
\adaptlayout[...][...]
```

For local changes in the page layout you can use the `\startlocal` and `\stoplocal` commands.

```
\startlocal
\setuplayout[height=+0.5cm]
This is some text
\stoplocal
```

6.4 Relative Layout Parameters

The values of the relative layout parameters are available as commands. This enables you to work more accurately when defining measures of columns, figures and tables. If you want to define the width of a column or the height of a figure, this can be done relatively with the `\makeupwidth` and `\makeupheight` command options.

	Description
\topdistance	
\topheight	
\headerdistance	
headerheight	

TABLE 6.2 – Text

Changes in page width or page height will alter columns and figures proportionally.

Chapter 7

Paragraph Layout

7.1 Paragraph Separation

In ConTEXt, you can start a new paragraph with either en empty line or the `\par` command. It is advised to use empty lines as paragraph separators. This will lead to clearly structured and well organized input files and will prevent mistakes. Only in situations where a command has to be closed explicitly, the `\par` command should be used.

7.2 Paragraph Formatting

Sometime you want to apply a special format to paragraphs. You can define a special format with the `\defineparagraphs` command, after which you can specify the details of this format with the `\setupparagraphs` command, which have the following general syntax.

```
\defineparagraphs[..,..][..,..=..,..]
\setupparagraphs[..,..][..,..][..,..=..,..]
```

A few `\defineparagraphs` command options are listed in table 7.1.

Arg	Option	Value	Description
1	*name*		
Arg	**Option**	**Value**	**Description**
2	n	number	
	rule	on off	
	height	fit dimension	
	before	*command*	
	after	*command*	
	inner	*command*	
	distance	*command*	
	tolerance	verystrict strict tolerant verytolerant stretch	
	align	left right middle	

TABLE 7.1 – Some /defineparagraphs Command Options.

A few `\setupparagraphs` command options are listed in table 7.2.

Arg	Option	Description
1	*name*	
Arg	**Option**	**Description**
2	*number* *each*	

Arg	Option	Value	Description
3	style	normal bold slanted	
	width	*command*	
	height	*command*	
	before	*command*	
	after	*command*	
	tolerance	verystrict strict <u>tolerant</u> verytolerant	
	align	left right middle <u>width</u>	

TABLE 7.2 – Some /setupparagraphs Command Options.

After defining a paragraph with `\defineparagraphs` you can format the paragraph with `\setupparagraphs`. Next, you can start your paragraph with the `\start ...` and the `\stop ...` command combination. A new paragraph starts with the name of your paragraph, in this case 'mypar'.

```
\defineparagraphs[mypar]
[n=3,
before={\blank},
after={\blank}]
\setupparagraphs[mypar]
[1]
[width=.08\textwidth,
style=bold]
\setupparagraphs[mypar]
[2]
[width=.3\textwidth]
\startmypar
1360
\mypar
First oil painting in western Europe
\mypar
With the introduction of oil painting into western Europe, the earliest
naturalistic painting is created. Its subject is the French king, John the Good.
After this, naturalistic portraitures become prominent in European art.
\stopmypar
```

Below is another example of paragraph formatting.

```
\defineparagraphs[paintpar]
[n=3,
before=,
after=,distance=1em]
\setupparagraphs[paintpar]
[1]
[width=.1\textwidth]
\setupparagraphs[paintpar]
[2]
[width=.3\textwidth]
\startpaintpar
\it 1360
\paintpar
First oil painting in western Europe
\paintpar
With the introduction of oil painting into western Europe, the earliest
naturalistic painting is created. Its subject is the French king, John the Good.
After this, naturalistic portraitures become prominent in European art.
\stoppaintpar
```

This can also be coded in a more cryptic and efficient way.

```
\paintpar
```

```
\it 1360
\\
First oil painting in western Europe
\\
With the introduction of oil painting into western Europe, the earliest
naturalistic painting is created. Its subject is the French king, John the Good.
After this, naturalistic portraitures become prominent in European art.
\\
```

The \\ columns are used as column separators and they are essential. In order to condense the ConTEXt code, this cane be coded as following.

```
\paintpar \it 1360
\\ First oil painting in western Europe
\\ With the introduction of oil painting into western Europe, the earliest
naturalistic painting is created. Its subject is the French king, John the Good.
After this, naturalistic portraitures become prominent in European art. \\
```

7.3 Paragraph Indentation

If you want the paragraph to start with an indentation, you can type in the setup area of the input file the `\indenting` command, which has the following general syntax.

```
\indenting[..,..]
```

You can specify what kind of indentation you want as table 7.3 indicates.

Option	**Description**
never	
not	
no	
yes	
always	
first	
next	

TABLE 7.3 – Some /indenting Command Options.

By default indenting is set to 'never'. If you choose to use indentations you may sometimes have to switch it off explicitly, which can be done as following.

```
\noindenting
```

You can setup the value of indentation with.

```
\setupindenting[..,..]
```

Table 7.4 shows the available command options for the `\setupindenting` command.

Option	**Description**
none	
small	
medium	
big	
next	
first	
normal	
odd	
even	
dimension	

TABLE 7.4 – Some /setupindenting Command Options.

ConTEXt automatically suppresses indentation at certain points, for example after a chapter heading.

7.4 Paragraph Vertical Spacing

Vertical spacing between paragraphs can be specified by.

```
\setupwhitespace[..,..]
```

Table 7.5 shows the available command options for the `\setupwhitespace` command.

Option	Description
<u>none</u>	
small	
medium	
big	
dimension	

TABLE 7.5 – Some /setupwhitespace Command Options.

When inter-paragraph spacing is specified there are two commands available that are seldom used.

```
\nowhitespace
```

and

```
\whitespace
```

If a paragraph contains horizontal lines, the line spacing requires extra care. Consider the following example where the `\framed{some text}` command is used to put lines around the word enclosed inside the curly braces pair.

```
This is a paragraph that contains some words that are
\framed{enclosed with lines}
in order to show the required correction.
```

When processed, we see an alignment problem that must be corrected. This can be done as following with the `\startlinecorrection ... \stoplinecorrection` command pair.

```
This is a paragraph that contains some words that are
\startlinecorrection
\framed{enclosed with lines}
\stoplinecorrection
in order to show the required correction.
```

The vertical spacing of the output will be better.

7.5 Implicit Vertical Spacing

Furthermore, the `\blank` command provides for vertical white spacing, too. The general syntax for the \bank command is.

```
\blank[..,..]
```

Table 7.6 shows the available command options for the `\blank` command.

Option	Description
none	
small	
medium	
big	
nowhite	
back	
white	
dimension	

TABLE 7.6 – Some /blank Command Options.

You can specify the amount of blank space using keyword options such as small, middle, and big, which are related to the current font size as the following example illustrates.

```
This is the first line of text
\blank
This is the second line of text
\blank[big]
This is the third line of text
\blank[2*big]
This is the fourth line of text
```

If you do not specify a parameter, the `\blank` command will yield the default space. Default spacing can be setup with

```
\setupblank[..,..]
```

This is a global parameter and is usually put in the setup area of the input file, so all vertical blank spaces in the document will get its properties. It has the same keyword options as the `\blank` command. Alternatively, a 'dimension' such as for example '3pt' can be specified. If you want to suppress vertical spacing you can use the following command combination.

```
\startpacked[..,..] . . . \stoppacked
```

If you want to increase vertical spacing you can use the following command pair combination.

```
\startunpacked[..,..] . . . \stopunpacked
```

The following code illustrates its use.

```
\defineparagraphs[year][n=2,before=,after=]
\year 500 \\ beginning of the middle ages \\
\year 1500 \\ end of the middle ages \\
\startpacked
\year 500 \\ beginning of the middle ages \\
\year 1500 \\ end of the middle ages \\
\stoppacked
\startunpacked
\year 500 \\ beginning of the middle ages \\
\year 1500 \\ end of the middle ages \\
\stopunpacked
```

The first two lines have regular vertical spacing, the next two lines have reduced vertical spacing and the last two lines have increased vertical spacing.

7.6 Explicit Vertical Spacing

You can force vertical space as following.

```
\godown[...]
```

The command has the same keyword options as the `\blank` command. Alternatively, a 'distance' such as for example '10in' can be specified.

7.7 Margin Text

The \inmargin command is used to put text in the margin. The command has the following general syntax.

```
\inmargin{...}
```

Where the text that needs to be put in the margin is put inside the curly braces pair as the following example illustrates.

```
\inmargin{this text will be put in the margin}
```

7.8 Regular Outlined Text

The `\framed` command is used to outline a text.

```
\framed[height=1cm,width=fit]{This text is outlined}
```

The `\framed` command has the following general syntax.

```
\framed[..,..=..,..]{...}
```

The command has two arguments. The first argument is a square bracket pair that optional and contains the properties of the outlining. The second argument is a curly braces pair that encloses the text that will be outlined. Table 7.7 shows some of the `\framed` command keys and their available key values.

Option	Value	Description
width	fit	
height	cm	
	screen	
background		

TABLE 7.7 – Some \framed Command Options.

7.9 Inline Outlined Text

The `\inframed` command is used to outline a text inline and is a better command to use to outline small pieces of text within a paragraph because it automatically takes caer of interline spacing.

```
This line has \inframed[width=fit]{some of its words} outlined.
```

The `\inframed` command uses the same general syntax and command options as the `\framed` command.

7.10 Paragraph Outlined Text

If you want to outline complete paragraphs you can use the following command pair combination.

```
\startframedtext[..,..] ... \stopframedtext
```

The following code illustrates its use.

```
\startframedtext[width=.7\makeupwidth]
This text is a small paragraph and it shall be outlined. This is done with the
ConTEXt startframed and stopframed command pair.
\blank
Various command options are available to modify the settings of the outlining.
\stopframedtext
```

Table 7.8 shows some of the `\startframedtext` command options.

Option	Description
left	
right	
middle	
none	

TABLE 7.8 – Some \startframedtext Command Options.

If you do not specify a parameter at the `\startframed` command, it will use the default values, which can be setup with

```
\setupframed[..,..=..,..]
```

This is a global parameter and is usually put in the setup area of the input file, so all outlined paragraphs in the document will get its properties. Table 7.9 shows some of the `\setupframed` command options.

Option	Value	Description
height	fit broad *dimension*	
width	fit broad *dimension*	
align	<u>yes</u> no	
corner	round <u>right</u>	

TABLE 7.9 – Some \setupframed Command Options.

7.11 Alignment

Single lines of text can be aligned with the commands `\leftaligned, \midaligned and \rightalinged`, as demonstrated in the example below.

```
\leftaligned{\inframed[width=fit]{This text is aligned left}}
\midaligned{\inframed[height=1.5cm,frame=off]{This text is aligned in the middle}}
\rightaligned{\inframed[height=10mm]{This text is aligned right}}
```

Alignment of paragraphs is done with the following commands.

```
\startalignment[..,..] . . . \stopalignment
```

You can optionally specify the stretch tolerance and the vertical and horizontal direction.

```
\setuptolerance[..,..]
```

Table 7.10 shows some of the `\setuptolerance` command options.

Value	Description
horizontal	
vertical	
stretch	
space	
verystrict	
strict	
tolerant	
verytolerant	

TABLE 7.10 – Some \setuptolerance Command Options.

In columns you could specify the tolerance as following.

```
\setuptolerance[horizontal,verytolerant]
```

Horizontal and vertical alignment can be setup with the `\setupalign` command.

```
\setupalign[..,..]
```

Table 7.11 shows some of the `\setupalign` command options.

Value	Description
width	
left	
right	
middle	
broad	
line	

TABLE 7.11 – Some \setupalign Command Options.

Chapter 8

Fonts and Font Switches

8.1 Introduction

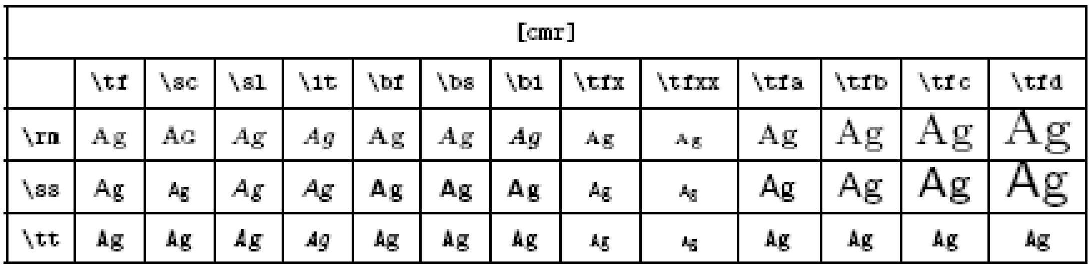
The default font type in ConTEXt is Computer Modern Roman (CMR). You can also use the Lucinda Bright font family. Many of the font types of the American Mathematical Society (AMS), but also PostScript (pos) such as Times fonts can be used. Fonts can be installed using the ‘texfont.pl’ Perl script which can generate the font metrics needed. In ConTEXt there are four ways to switch fonts.

1. A complete font change (\setupbodyfont, \switchtobodyfont).
2. Font size (\tfa, tfb, etc.)
3. Font style (\rm, \ss, etc.)
4. Alternative font style (\bold, \sans, etc.)

If you want an overview of the available font family, you can type

```
\showbodyfont[cmr]
```

This command will show a table with all font variations of the CMR font family.



[cmr]													
	\tf	\sc	\sl	\it	\bf	\bs	\bi	\tfx	\tfxx	\tfa	\tfb	\tfc	\tfd
\rm	Ag	AG	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag
\ss	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag
\tt	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag	Ag

8.2 Defining the Body Font

8.2.1 Defining the Body Font at the Beginning of the Document

The `\setupbodyfont` command is used in the setup area of the input file to select the font family, style and size for a document. The following example would set at the setup area the font type for the document at sans serif at a size of 9 points.

```
\setupbodyfont[sansserif,9pt]
\starttext
This is a text in sans-serif.
\stoptext
```

The `\setupbodyfont` command has the following general syntax.

```
\setupbodyfont[..,..,..]
```

The command has a single argument that can contain various keyword options. Table 8.1 shows some of the `\setupbodyfont` command options and their description.

Option	Description	Example
	Font Types	
sansserif	Sans-Serif font	
palatino	Palatino font	
ber	Karl Berry font names	

phv	Helvetica font definitions	
ss	Sans-Serif font	
Font Sizes		
pt	point	10pt

TABLE 8.1 – Some \setupbodyfont Command Options.

The following example specifies that you want to use the Karl Berry fontnames.

```
\setupbodyfont[ber,phv,ss]
\starttext
This is a text in sans-serif.
\stoptext
```

With `phv` you specify that you want to load the Helvetica font definitions, and with `ss` you specify that you want to use a sans-serif font as the body font. To use the default postscript font Times, Helvetica and Courier, use

```
\setupbodyfont[ber,pos]
\starttext
This is a text in Times.
\stoptext
```

8.2.2 Defining the Body Font in the Middle of the Document

The `\switchtobodyfont` command is used for changes mid-document and at section level. The command has the following general syntax.

```
\switchtobodyfont[..,..,..]
```

The command has a single argument that can contain several keyword options. The following example starts with sans serif font type at 9 points and in the middle of the document, it changes to Palatino font type at 8 points.

```
\setupbodyfont[sansserif,9pt]
\starttext
This text will be rendered in the document 9 point sans serif.
\switchtobodyfont[palatino,8pt]
This text will be rendered in the document 8 point palatino.
\stoptext
```

Font sizes are available from 4pt to 12pt.

8.3 Style and Size Switches in Commands

In a number of commands you can optionally specify style switches to obtain the desired typeface, for example

```
\setuphead[chapter][style=\boldslanted]
```

You can also use size switches to change the typestyle inside a group, for example

```
\setuphead[chapter][style=\tfd]
```

Table 8.2 shows some of the available typeface switch options, typestyle options shall be explained later in this chapter.

Option	**Example**
\normal	
\bold	
\boldslanted	
\bolditalic	
\small	
\smallbold	
\smallboldslanted	
\smallbolditalic	
\sans	
\sansbold	
\slanted	
\slantedbold	
\italicbold	
\smallnormal	
\smallslanted	

\smallslantedbold
\smallitalicbold
\sansserif
\smallcaps

TABLE 8.2 – Some Typeface Command Options.

You can either use those as style or size switches inside a group, or as a font changing command that takes an argument, for example

```
This is {\bold some bold text} but \bold{this text is bold} too.
```

The result on the document is the same.

8.4 Local Font Settings

The font switch is used to change the local font setting, for example

```
There is some {\em emphasized text} in this line.
```

The switch influences everything in its group. Notice how curly braces are used to keep font switching local.

8.4.1 Changing Local Typeface

In running local text, the inline typeface can be changed into some other typeface. The following example shows how text is changed to slanted inside curly braces with the `\sl` command.

```
Some of the {\sl text in this line} will be rendered in slanted typeface.
```

Table 8.3 shows the available inline typefaces.

Typeface	Font
\sl	slanted
\it	italic
\bf	bold

TABLE 8.3 – Inline Typefaces.

8.4.2 Changing Local Typestyle

In running local text, the inline typestyle can be changed into some other typestyle. The following example shows how some typestyle is changed inside curly braces with the `\tfc` command.

```
Some of the {\tfc text in this line} will be rendered in another typestyle.
```

Table 8.4 shows the available inline typestyles.

Typestyle	Font
\tfxx	somewhat smaller then \tfx
\tfx	somewhat smaller then \ft
\tf	actual style
\tfa	somewhat greater then \tf
\tfb	somewhat greater then \tfa
\tfc	somewhat greater then \tfb
\tfd	somewhat greater then \tfc

TABLE 8.4 – Inline Typestyles.

The actual typestyle is indicated with `\tf`. If you want to switch to a somewhat greater size, you can type `\tfa, \tfb, \tfc` or `\tfd`.

8.4.3 Changing Local Fontstyle

In running local text, the inline fontstyle can be changed locally into some other fontstyle. The following example shows how some text is changed to teletype fontstyle inside curly braces with the `\tt` command.

```
Some of the {\tt text in this line} will be rendered in teletype fontstyle.
```

Table 8.5 shows the available inline fontstyles.

Fontstyle	Font
\rm	roman
\ss	sans serif

\tt	teletype

TABLE 8.5 – Inline Fontstyles.

8.4.4 Changing and Combining Local Typeface and Typestyle

It is also possible to combine a typeface and a typestyle. An addition of `a, b, c` or `d` to `\sl, \it` or `\bf` is also allowed. like at the following example where the commands `\sl` and `\tfb` are combined to the `\slb` command that renders the inline text both slanted and in a somewhat greater size. The following example will demonstrate the effect.

```
Some of the {\slb text in this line} will be rendered in another typeface and
typestyle.
```

Similarly, the combined commands `\slc` and `\sld` can be used.

8.4.5 Redefining the Body Font

For special purposes you can define your own font size using the `\definebodyfont` command, which has the following general syntax.

```
\definebodyfont[size][style][options]
```

In the example below we will define the body font to be Roman at 9 points.

```
\definebodyfont[9pt][rm][tfe=cmr12 at 24pt]{\tfe Some Text}
```

Additionally we redefine the `\tfe` command as a Computer Modern Roman font at 24 points.

8.5 Emphasized Text

The command `\em` is used to emphasize text consistently throughout the document as the following example illustrates.

```
Some words in this {\em text have been} emphasized, and {\bf \em some words} have
been both boldfaced and emphasized.
```

Emphasized words appear in either slanted or italic style. An emphasize within an already emphasized environment will result in normal print again, so some kind of contrast will be ensured.

8.6 Small Caps

Many acronyms are printed in small capitals. In ConTEXt there are two types of small capitcals, real small caps and pseudo small caps. Real small caps are somewhat smaller than the capital of the actual bodyfont and are created with the `{\sc...}` command. Pseudo small caps are more intelligent since they adapt to the surrounding text and are created with the `{\cap...}` command.

```
PDF document is written in regular capitals, {\sc PDF document} in real small caps
and {\cap PDF document} is a pseudo small caps.
```

The reason for using pseudo small caps instead of real small caps is not just a matter of taste.

8.7 Quotations

8.7.1 Block Quotations

The `/startquotation` and `/stopquotation` command pair creates quotations.

```
/startquotation
This is a quotation.
/stopquotation
```

The `/startquotation` command creates the quotation and the `/stopquotation` command closes off the quotation.

8.7.2 Inline Quotes

ConTEXt also has inline quotes that can be created with either the `/quote` or `/quotation` commands.

```
This is a /quote{single quote here} and this is a /quotation{double quotation
here}.
```

The `/quote` command surrounds the quote with single quote characters. The `/quotation` command surrounds the quote with double quote characters.

8.8 Teletype Verbatim Text

8.8.1 Block Verbatim Text

Text that is not subject to macro expansion is created with the `/starttyping` command and is closed off wit the `/stoptyping` command. The text inside this command pair will appear exactly as typed, meaning it will display typed text and the output will reflect the line breaks as the appear in the input.

```
/starttyping
This is verbatim text in ConTEXt that is not subject to macro expansion.
/stoptyping
```

The text will be displayed with a monospace font such as for example Courier or Courier New.

8.8.2 Inline Verbatim Text

The `\type{...}` command is used for inline verbatim. The curly braces enclose the text that you want to type verbatim.

```
Some of the text in this sentence \type{is typed} inline verbatim. The command
\type{\type} starts with a backslash \type{\}.
```

Be careful with the `\type` command since it will temporarily disable th eline breaking mechanism.

8.8.3 Setting up Varbatim Text Layout

You can setup the layout of verbatim text with the commands

```
\setuptype[...]
```

and

```
\setuptyping[...]
```

Table 8.6 shows some optional `\setuptype` and `\setuptyping` command options.

Key	Value

TABLE 8.6 – Some \setuptype and \setuptyping Command Options.

These commands allow you to specify how verbatim text should be typeset. Various mechanisms are build-in as the following example illustrates.

```
\setuptype[option=slanted]
\type{This is a <<special>> case.}

\setuptypeing[margin=1cm]
\starttyping
What is that?
\stoptyping

\setuptyping[option=color]
\starttyping
An \extrordinary{surprise}
   \for \all!
\stoptyping
```

In this example, the strings `\extraordinary`, `\for` and `\all` are typeset in green and the braces are typeset in red. All text, including these strings is typeset verbatim.

8.9 Text Colors

8.9.1 Block Text Colors

The `\setupcolor` command activates the use of red, green and blue colored text and the `\startcolor` and `\stopcolor` command pair specifiy the colored text section.

```
\setupcolor[state=start]

\startcolor[red]
This text will be rendered red in the document.
\stopcolor
```

8.9.2 Inline Text Colors

The `\color` command is used for coloring text inline.

```
\setupcolor[state=start]
```

```
This \color[green]{word} is green.
```

8.9.3 Combined Block and Inline Text Colors

The `\startcolor`, `\stopcolor` and `\color` commands specifiy the colored text section with an inline differently colored text.

```
\setupcolor[state=start]
\startcolor[red]
This text will be rendered red on the document with a single green
\color[green]{word}.
\stopcolor
```

On a black and white printer you will see only grey shades.

8.10 Background Text

In order to emphasize a section, you can use backgrounds with the `\startbackground` and `\stopbackground` command pair as the following example illustrates.

```
\setupbackgrounds[[background=screen,corner=round]
\startbackground
This text has a background with round corners.
\stopbackground
```

The `\setupbackgrounds` command is used to set some values for the background. The background command has the following general syntax.

```
\setupbackgrounds[..,..=..,..]
```

Table 8.7 shows some optional `\setupbackgrounds` command options.

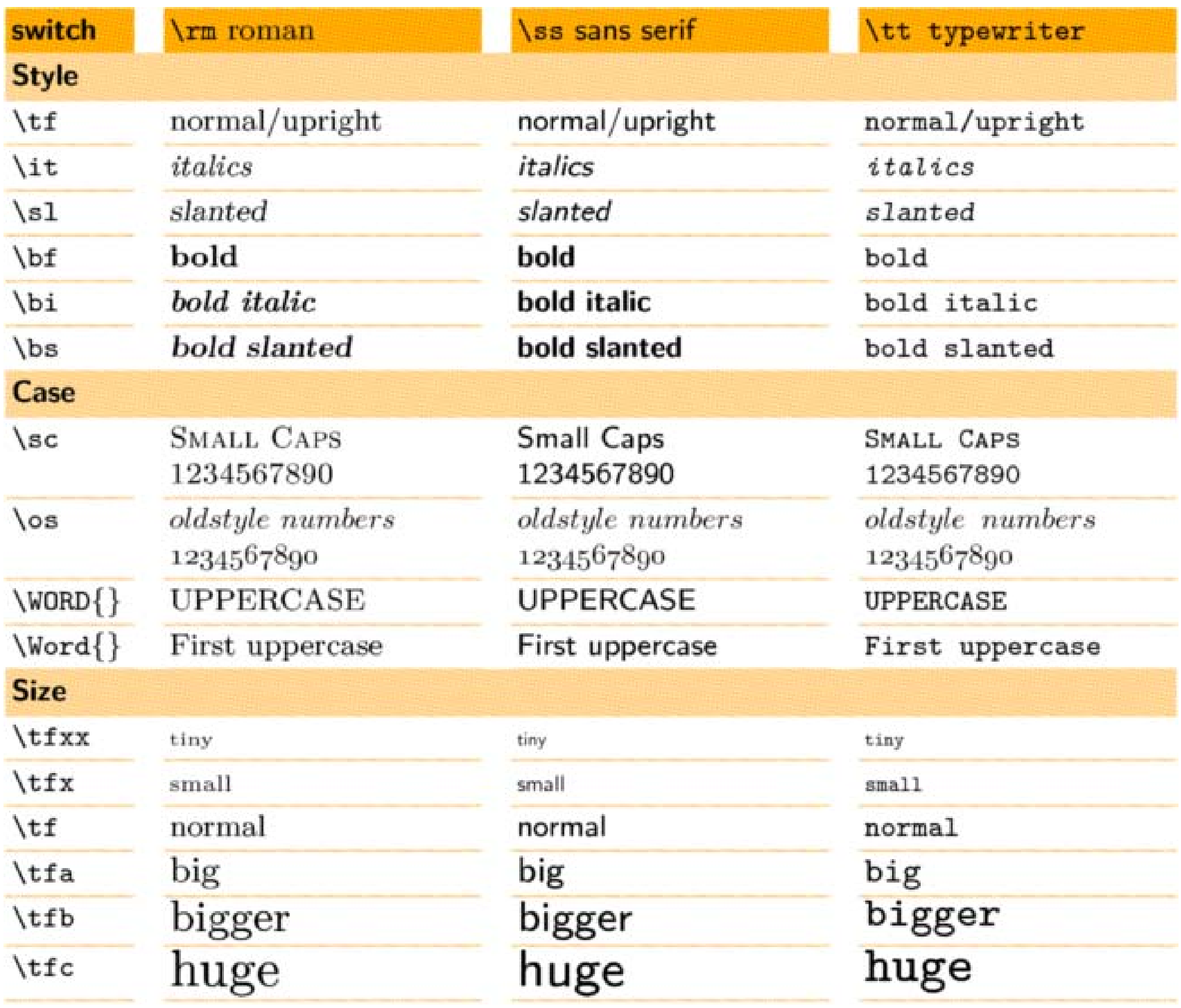
Key	**Value**
background	screen
	color
corner	round
page	NO VALUE
text	NO VALUE
backgroundcolor	grey

TABLE 8.7 – Some \setupbackgrounds Command Options.

8.11 End of Chapter Review

8.11.1 Commands

Table 8.8 shows some optional `\setupbackgrounds` command keys and their available key values.



switch	\rm roman	\ss sans serif	\tt typewriter
Style			
\tf	normal/upright	normal/upright	normal/upright
\it	*italics*	*italics*	*italics*
\sl	*slanted*	*slanted*	*slanted*
\bf	**bold**	**bold**	**bold**
\bi	***bold italic***	***bold italic***	***bold italic***
\bs	***bold slanted***	***bold slanted***	***bold slanted***
Case			
\sc	SMALL CAPS 1234567890	Small Caps 1234567890	SMALL CAPS 1234567890
\os	*oldstyle numbers* 1234567890	*oldstyle numbers* 1234567890	*oldstyle numbers* 1234567890
\WORD{}	UPPERCASE	UPPERCASE	UPPERCASE
\Word{}	First uppercase	First uppercase	First uppercase
Size			
\tfxx	tiny	tiny	tiny
\tfx	small	small	small
\tf	normal	normal	normal
\tfa	big	big	big
\tfb	bigger	bigger	bigger
\tfc	huge	huge	huge

TABLE 8.8 – fff

8.11.2 Code Example

Chapter 9

Structural Elements

9.1 Itemization

9.1.1 Unnumbered Lists

The `/startitemize` command creates an unnumbered list.

```
/startitemize
/item This is the first item.
/item This is the second item.
/item This is the third item.
/stopitemize
```

With the `/startitemize` command, each line item will get a bullet.

9.1.2 Numbered Lists

The `/startitemize[n]` command with the n option creates a numbered list.

```
/startitemize[n]
/item This is the first item.
/item This is the second item.
/item This is the third item.
/stopitemize
```

With the `/startitemize[n]` command, each line item will get a number.

9.1.3 Narrower

The following command renders the text narrower on the screen

```
\startnarrower … \stopnarrower
```

The following command combination ensures that line breaks entered in the input file are obeyed.

```
\startlines . . . \stoplines
```

As

9.3 Activating Hyperlinks

The `\setupinteraction` command is used to activate hyperlinks in the document.

```
\setupinteraction[state=start,color=blue,style=normal]
```

As indicated, you can use this command to setup colors associated to hyperlinks, as well as enable some special features. The \setupinteraction command has the following general syntax.

```
\setupinteraction[..,..=..,..]
```

9.4 The LaTEX ‘abstract’ Environment

ConTEXt does not have environments like LaTEX. The LaTEX ‘abstract’ environment can be simulated in ConTEXt as following.

```
\starttext
\startnarrower
\switchtobodyfont[small]
```

```
\midaligned{\bf Abstract}\par
Enter your abstract text here.
\stopnarrower
\stoptext
```

The command `\startnarrower` creates a paragraph, left and right indented by some white space. The command `\switchtobodyfont[small]` makes the font size some smaller than the current font size, and the command `\midaligned` creates a centered line. The `\bf` command will have the text shown as bold font.

9.5 The LaTEX ‘description’ Environment

The LaTEX ‘description’ environment can be simulated in ConTEXt as following.

```
\starttext
\stoptext
```

Text

9.6 End of Chapter Review

9.6.1 Commands

9.6.2 Code Example

Chapter 10

Figures in ConTEXt

11.1 Floats

Figures and certain types of tables are so called floats. They are floats, just like pieces of text that do not follow the main flow, but can go on the same page or elsewhere. ConTEXt will look whether there is enough space for the float on the page. If not, the float will be placed at another location and the text carries on, while the figure f3loats in your document until the optimal location is found.

11.2 Placing a Figure

Photographs and pictures can be inserted in your document with the `\placefigure` command.

```
\placefigure
[left]
[fig:test]
{Test Picture.}
{\externalfigure[test]}
```

The command `\placefigure` handles numbering and vertical spacing before and after your figure and has the following general syntax.

```
\placefigure[...][...,...]{...}{...}
```

The command has four arguments that will be described below. Furthermore, this command initializes a float mechanism. The first square bracket pair influences the float mechanism and the options are shown in table 8.1.

Option	Description
here	put figure at this location if possible
force	ignore float mechanism and place figure
page	put figure at top of the next page
top	put the caption above figure
bottom	put caption under figure
left	place figure at the left margin
right	place figure at the right margin

TABLE 8.1 – Float Options of the \placefigure Command.

The second square bracket pair is for cross-referencing elsewhere in the document. You can refer to this particular figure by typing.

```
\in{figure}[fig:test]
```

The first curly brace pair is used for the caption. You can type any text you want. If you want no caption and no number, you can type `{none}`. The second curly brace pair is used for defining the picture and addressing the file names of external figures. ConTEXt in combination with TEXUTIL supports EPS, TIF, JPG, MPS, PDF and PNG type files, although inclusion depends on the DVI drivers used. The file needs to be in the same directory as the ConTEXt input file. The curly brace pair encloses the command `\externalfigure` command, which has the following general syntax.

```
\externalfigure[...][..,..=..,..]
```

This command gives you freedom to do anything you want with a figure. The command has two square bracket pairs. The first one is used for the exact file name without extension, and the second for file formats and dimensions.

```
\inmarge
{\externalfigure
[test1]
[width=\marginwidth]}
```

Text

11.3 Combining a Figure and Text

If you want a more integrated layout of the picture and the text, you can use the `\startfiguretext` command.

```
\startfiguretext
[left]
[fig:test]
{none}
{\externalfigure[test1]
[width=.5\makeupwidth]}
The text that you put here will be placed at the picture.
\stopfiguretext
```

11.4 Float Setup

The layout of figures can be setup with the `\setupfloats` command, which has the following general syntax.

```
\setupfloats[..,..=..,..]
```

The command has a single argument that can contain multiple keys to set the properties of the floats. The numbering and labels of a figure can be setup with the `\setupcaptions` command, which has the following general syntax.

```
\setupcaptions[..,..=..,..]
```

The command has a single argument that can contain multiple keys to set the properties of the captions. These commands are typed in the setup area of the input file and have a global effect on all floating blocks as the following example illustrates.

```
\setupfloats
[location=right]
\setupcaptions
[location=top,
height=.4\makeupheight,
style=boldslanted]

\starttext
This is the text in the core of the document.
\placefigure
{just any picture.}
{\externalfigure
[test1]
[width=4cm]}
\stoptext
```

Chapter 12

Tables in ConTEXt

12.1 Introduction

Two commands exist in ConTEXt to create tables. If the table is started with the \starttable and closed off with the \stoptable commands, it can split a table across various pages as a float. If the table is started with the \starttabulate command and closed off with the \stoptabulate command, it does not split across pages, it does not support vertical lines, it aligns data vertically, and the header specifies the format of the columns. The width of the columns will be equal to the row with the largest contents but you can also explicitly specify the width. Every row starts with \NC (next column) and ends with \NC and \NR (next row). \HL (horizontal line) creates a horizontal line.

10.2 Creating a Floating Table

The following example creates a simple table.

```
\starttable[¦c¦c¦]
\HL
\NC Year \NC Quantity \NC\SR
\HL
\NC 2000 \NC 21 \NC\FR
\NC 2001 \NC 20 \NC\MR
\NC 2002 \NC 24 \NC\MR
\NC 2003 \NC 22 \NC\FR
\stoptable
```

In case the command \placetable is given as an argument to the table, the table is placed in the list of tables with the \placelistoftables command.

```
\starttext
\placecontent
\placelistoftables
\chapter{First Chapter Title}
This is the text in the first chapter.
\chapter{Second Chapter Title}
This is the text in the second chapter.
\placetable
\starttable[¦c¦c¦]
\HL
\NC Year \NC Quantity \NC\SR
\HL
\NC 2000 \NC 21 \NC\FR
\NC 2001 \NC 20 \NC\MR
\NC 2002 \NC 24 \NC\MR
\NC 2003 \NC 22 \NC\FR
\stoptable
\stoptext
```

The command \completelistoftables creates a list of tables on a new page, whereas the \placelistoftables command places the list on the same page. You can make the heading of the table bold with the \bf command.

```
\starttext
\placecontent
\placelistoftables
\chapter{First Chapter}
This is the text in the first chapter.
\chapter{Second Chapter}
This is the text in the second chapter.
\placetable
\starttable[¦c¦c¦]
\HL
\NC \bf Year \NC \bf Quantity \NC\SR
\HL
\NC 2000 \NC 21 \NC\FR
\NC 2001 \NC 20 \NC\MR
\NC 2002 \NC 24 \NC\MR
\NC 2003 \NC 22 \NC\FR
\stoptable
\stoptext
```

10.3 Creating a Non-Floating Table

Similarly, the `\starttabulate` and `\stoptablate` commands create a table as following.

```
\starttext
\starttabulate[¦l¦p¦]
\NC Armadillo: \NC not edible \NC\NR
\NC Gnat: \NC not edible \NC\NR
\stoptabulate
\stoptext
```

Chapter 11

Bibliography with the bibTEX Module

11.1 Introduction

In ConTEXt, the bibliography is created with the bibTEX bibliographic module. Bibliography references have to be in a separate file, preferably with the same file name as the input file but with file extension `.bbl`, for example `myfile.bbl`. Bibliography files also contain flat ASCCI text. Bibliography entries in the `.bbl` file are started with the `\startpublication` command and closed off with the `\stoppublication` command. Within those two commands, the commands `\artauthor`, `\arttitle`, `\journal`, and `\pubyear` provide all details of the publication. In order to place these publications in the document, the `.tex` input file needs to have the command `\usemodule[bib]` in the setup area and the command `\completepublications` where the bibliography needs to be placed.

11.2 Usage of the BIB Bibliography Module

The following example of the `myfile.bbl` file defines the entry of a book.

```
\startpublication[k=Brodie84,
t=article,
a=L.~Brodie,
y=1984,
s=LB84]
\artauthor[]{Leo}[L.]{}{Brodie}
\arttitle{Thinking Forth, a language and philosophy for solving problems}
\journal{Prentice Hall}
\pubyear{1984}
\stoppublication
```

The `\startpublication` command has the following general syntax.

```
\startpublication[..,..=..,..]
```

The command has a single argument that can contain multiple keys and key values to enter input for the bibliography of the entry. The available key options for the `\startpublication` command are shown in table 12.1.

Key	Meaning	Example
k	key	Brodie84
t	type	article
a	author	L.~Brodie
y	year	1984
s	short	LB84
n	number	1

TABLE 12.1 – Keys of the \startpublication Command.

The following example of the input file `myfile.tex` places the bibliography in the ConTEXt document.

```
\usemodule[bib]

\starttext
\section{My Life}
I no longer think Forth \cite[Brodie84].
\placepublications
\stoptext
```

The command `\placepublications` does not add something to the table of contents. This command can be replaced by the command `\completepublications` to provide a list of publications.

```
\usemodule[bib]

\starttext
\completecontent
\section{My Life}
I no longer think Forth \cite[Brodie84].
\completepublications
\stoptext
```

Alternatively, the optional `\setuppublications` command can be added to the setup area in order to define entries to include, how to sort them or to include every entry or only the referenced ones.

```
\usemodele[bib]
\setuppulications
[numbering=yes,
sort=author]

\starttext
\completecontent
\section{My Life}
I no longer think Forth \cite[Brodie84].
\completepublications
\stoptext
```

As indicated in this code fragment, bibliography entries are referred to with the `\cite` command elsewhere in the document, which has the following general syntax.

```
\cite[key]
```

The command has a single argument, where [key] is the key or identification name you attached to the reference in the bibliography, and produces output on the spot where the command is issued.

Chapter 12

Math in ConTEXt

12.1 Introduction

Formulas in ConTEXt are very close to how most TEX Math commands work.

12.2 Display Style Formulas

Display style formulas are made with the `\startformula` command.

```
\startformula E = mc^2
```

Alternatively, display style formulas can be produced by surrounding the formula by two $ characters.

```
\$$ E = mc^2 $$
```

Numbered formulas are made by adding the prefix command `/placeformula` to the display formula.

```
\placeformula $$ E = mc^2 $$
```

12.3 Multiline Formulas

Multiline formulas are produced with the TEX `\displaylines` command. A new line is started after the `\cr` (carriage return) command.

```
$$\displaylines{ZeroOrOne = ((Amount\ \cr
Mod\ Currency_RoundingFactor) +\ \cr
Currency_Boundary) dev\ \cr
Currency_RoundingFactor)$$
```

Chapter 13

Chemical Structures in ConTEXt with PPCTEX

13.1 Introduction

The chemical module used in ConTEXt is based on PPCHTEX.

```
\chemical{CaCO_3,~,GIVES,~,CaO,~,+,~,CO_2}
```

Text

```
\chemical{CaCO_3,~,GIVES,~,%
CaO,~,+,~,CO_2}
```

Text

```
\startchemical[scale=small,width=fit,%
top=3000,bottom=3000]
\chemical[SIX,SB22357,DB14,Z2346,SR3,RZ3,%
-SR6,+SR6,-RZ6,+RZ6]
[C,N,C,C,H,H,H]
\chemical[PB:Z1,ONE,ZO,DIR8,ZO,SB24,DB7,%
Z27,PE][C,C,CH_3,0]
\chemical[PB:Z5,ONE,Z0,DIR6,Z0,SB24,DB7,%
Z47,PE][C,C,H_3C,0]
\chemical[SR24,RZ24][CH_3,H_3C]
\bottext{Compound A}
\stopchemical
```

Text

Chapter 14

Using GNUPlot with ConTEXt

14.1 Introduction

Formulas in ConTEXt are very close to how most TEX Math commands work.

Chapter 15

Using MetaFun in ConTEXt

15.1 Introduction

Formulas in ConTEXt are very close to how most TEX Math commands work.