Abstract

When you write a packege for ConTEXt there is a point when you need documentation of it to let users know how to use it. There are different ways how to write it, one of them is to include the documentation in the module itself.

To help the package author with this part ConTEXt gives you the option to pretty print the module code and format the code explanation and examples for their usage with various markup commands.

In the following chapters you will learn how to produce a pdf from the source and which commands are available besides ConTEXt's normal mechanisms.

Documenting TEX and METAPOST files

When you write a new package you do normally start first with the TEX code and add afterwards comments and other information to it. Before we continue how to add text to the code let's start with a short example.[1]

```
\unprotect

\def\????fb{@@@@fb} % Namespace: FancyBreak

\def\fancybreakparameter#1%
  {\csname\????fb#1\endcsname}

\def\setupfancybreak
  {\dodoubleargument\getparameters[\????fb]}

\def\fancybreak
  {\dowithnextboxcontent
     {\setupalign[\@@@@fbalign]}
     {\blank[\@@@@fbspacebefore]%
      \flushnextbox
      \blank[\@@@@fbspaceafter]}
     \vbox}

\setupfancybreak
  [\c!spacebefore=,
   \c!spaceafter=\v!nowhite,
   \c!align=\v!middle]

\protect \endinput
```

[1] This is a simplified version of the `\fancybreak` macro which lacks many features but for demonstrations this version is better suited.

As you can see from the code above there is only a short comment to explain the meaning of the namespace. To start with the documentation we will first add meta information at the begin of the document which has always the following form:

```
%D   [     file=t-fancybreak,
%D      version=20xx.xx.xx,
%D        title=\CONTEXT\ User Module,
%D     subtitle=Fancybreak,
%D       author=Ben Lee User,
%D         date=\currentdate,
%D    copyright=Ben Lee User,
%D        email=ben.lee.user@tex.org,
%D      license=Public Domain]
```

The block contains information about the name of the file (without the file extension) and other information like the title and subtitle of the module and the name of the author. Not all information here are mendatory but you should always fill the following fields.

Field	**Type**	**Default value**
file	text	`\jobname`
title	text	—
subtitle	text	—
author	text	Hans Hagen
date	date	`\currentdate`

All of them are required because ConTEXt use them when you produce a pdf of your document on the titlepage of the file. As text for the field `title` you can normaly write "`\CONTEXT\ User Module`". Other fields are possible and although ConTEXt will ignore them it's good style to include information about the license and contact information, so let's extend our list with the following ones.

Field	**Type**
version	date
copyright	text
license	text
email	text

Documenting Lua files

Interface files

A interesting feature of ConTEXt is to show a overview of a commands syntax with the possible parameters and keywords which can be either optional or mandatory.

The input for these diagrams use xml and is stored in a external file which has for a package the form `<package>.xml`.

```
<?xml version="1.0" encoding="UTF-8"?>

<cd:interface
    xmlns:cd="http://www.pragma-ade.com/commands"
    name="context" language="en" version="20xx.xx.xx">

</cd:interface>

<cd:command name="donothing" file="syst-aux.mkiv">
    <cd:sequence>
        <cd:string value="donothing"/>
    </cd:sequence>
</cd:command>

\showsetup{donothing}
```

```
\donothing
```

```
<cd:command name="text" type="environment" file="core-job.mkiv">
    <cd:sequence>
        <cd:string value="text"/>
    </cd:sequence>
</cd:command>

\showsetup{starttext}
```

```
\starttext ... \stoptext
```

```
<cd:command name="placelistoffloats" generated="yes" file="strc-flt.mkiv">
    <cd:sequence>
        <cd:string value="placelistof"/>
        <cd:variable value="floats"/>
    </cd:sequence>
</cd:command>

\showsetup{placelistoffloats*}
```

```
\placelistofFLOATS
```

```
<cd:command name="high" file="core-fnt.mkiv">
    <cd:sequence>
        <cd:string value="high"/>
    </cd:sequence>
    <cd:arguments>
        <cd:content n="1"/>
    </cd:arguments>
</cd:command>

\showsetup{high}
```

```
\high {.*.}

*   CONTENT
```

```
cd:assignments     [.=.]           [..,.=.,..]
cd:keywords        [...]           [...,...]
cd:displaymath     $$...$$         $$...$$
cd:index           {...}           {..+...+..}
cd:math            $...$           $...$
cd:nothing         ...             -
cd:file            ~...~           -
cd:position        (...)           (...,...)
cd:reference       [...]           [...,...]
cd:csname          \...            -
cd:destination     [{..[ref]}]     [..,{..[ref]},..]
cd:triplet         [x:y:z=]        [x:y:z=,..]
cd:word            {...}           {... ... ...}
cd:content         {...}           {... ... ...}

cd:command            COMMAND
cd:dimension          DIMENSION
cd:file               FILE
cd:name               IDENTIFIER
cd:character          -
cd:mark               MARK
cd:number             NUMBER
cd:reference          REFERENCE
cd:plural             PLURAL NAME
cd:singular           SINGULAR NAME
cd:text               TEXT
cd:formula            FORMULA
cd:matrix             N*M
cd:list               LIST
cd:section            SECTION
cd:noargument         \...
```

```
cd:oneargument        \...#1
cd:twoarguments       \...#1#2
cd:threearguments     \...#1#2#3
```