textengine

COLLABORATORS			
	TITLE : textengine		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

textengine

1.1 main

```
                (beware: this doc is far from perfect)

Textengine 1.0 (should rather be 0.1)

Converts Textengine-style to either PageStream3-style or HTML-2, or anything
else someone's written a converter for.


                Copyright

                Overview

                Usage

                Programming

                Files
                If strange things happen when using Textengine (quite possible)  ↩
                   contact me.

toenges@fact.rhein-ruhr.de
```

1.2 copyr

```
Never care about any rights.
```

1.3 ov

```
Converts Textengine-style to any other style (currently PGS3 and HTML), incl.
all headers etc. If possible, the text will get directly transfered to the
target application.

Textengine Commands are in "<" and ">" brackets, usually their scope is limited
```

by the corresponding "</command>". Case sensitiv (maybe, didn't care about that). Every command will be transformed to a more or less complex different text-format, like PageStream3 or HTML. Commands which can't be transformed won't make trouble, they'll produce simple-text output.

The Textengine-style is designed to be easily readable (at least for human beings), even before it get's transformed to the final version. That means, it's something like TeX-light, but the source is a bit easier to read. Main advantage is, that you can use ONE file for any supported target. This is similar to texinfo.

TextEngine is context-sensitive, see example.

Important features:

- WORKING Drop Caps in PGS
- Lists (numbered, bulleted, custom)
- Random emphasize function in PGS (<E> automatically selects one out of big, bold, subscript, superscript...)
- It's easy for you to programm your own functions

Useful: GOLDED 3+ (won't work without)
PageStream 3
a HTML-viewer
match (for HTML)
raw2ent (for HTML)
rep (for HTML)

1.4 use

There are two kinds of "commands": The context-commands, which Textengine does automatically, and the "real" commands, which must be in the text.

Textengine is designed to do as much as possible of the formating based on the context.

Context

Commands

Place it in your GoldED's menu and run it this way: textengine.ged [TARGET]

If you give a TARGET (currently only PGS3 and HTML), the text will be fully converted.

If you don't, Textengine does just the pre-run. You won't need this, because it's completely senseless.

Note: there's about no error-tolerance. If the output is really weird, even the smallest mistake could be the cause.

1.5 cont

Context-interpretation depends partly on the selected style (style must be namend in the first line).

Article: If the second line is smaller than 26 bytes, then it becomes a heading.

If there is a third line immediately following, it becomes a sub-heading.

If the last line is only two bytes long, or contains several two-byte words seperated by ", ", then it's the author's name.

In PGS, the first real-bodytext-char is a drop cap, two lines high.

There could be other things i forgot to mention.

Box: If the second line is smaller than 45 bytes, it becomes a heading.

There could be other things i forgot to mention.

Letter: Much like box.

There is a special context-interpretation in lists. See there (sorry, no link).

1.6 comm

The Style must be given in the FIRST line, in the form of: <STYLE>

```
Example:--------------
  <ARTICLE>
  Blahtitle
  blahsubtitle

  text etc.
----------------------
```

Textengine Commands are in "<" and ">" brackets, usually their scope is limited by the corresponding "</command>". Case sensitiv (maybe, didn't care about that). Every command will be transformed to a more or less complex different text-format, like PageStream3 or HTML. Commands which can't be transformed won't make trouble, they'll produce simple-text output. Most commands must end in the same line they start.

The Textengine-STYLE is designed to be easily readable, even before it get's transformed to the final version. That means, it's something like TeX-light, but the source is a bit easier to read. Main advantage is, that you can use ONE file for any supported target. This is similar to texinfo.

If you want to get additional targets, or any new feature, contact me.

There are a lot of unexplained behaviors, better you find them out by practising. There might even be bugs. If you can't find a workaround yourself, contact me.

```
It’s possible that i forgot to mention some commands, for a complete feature
list, check the sources (if you understand them).

Supported targets:
  PGS3                      PageStream3 text formating codes, incl. transfer
            (PGS must be running, a textbox for the text
            selected).

  HTML                      HTML-2 text formating codes

Supported STYLEs:
  ARTICLE                   In PGS3-mode, paragraph-STYLEs get the prefix
            "article". First line is checked, if it’s longer
            than 25 bytes, the first and, if it actually
            contains any text, the immediately follwing lines
            are taken as headlines. Automatic drop caps. If
            the last line is exactly 2 chars (or several
            ", " seperated two-byte-strings) long, it’s
            formated as the name of the author. Needs an extra
            column for the headline. See Example.

  BOX                       In PGS3-mode, paragraph-STYLEs get the prefix
            "box", first line is necessarily the headline. See
            Example.

  LETTER                    Much like Box. See Example. Don’t use this for
            HTML-output.

Supported Commands:
  "’                        Right german quotations

  "`                        Left german quotations

  --                        En-dash

  ---                       Em-dash

  <%>                       Nop, can be used to make a line longer then a
            specific limit or whatever else.

  <%START>                  Start-folding-mark. No output.

  <%END>                    End-folding-mark. No output.

  ·                         Bullet

  <B>, </B>                 Bold

  <I>, </I>                 Italic

  <FIX>, </FIX>             Fixed-width-font

  <G>, </G>                 Big font (doesn’t support HTML)

  <E>, </E>                 Random marker. Either bold, italic, subscript,
            superscript or a bigger font. In HTML always bold.
  <LIST [arg]>, </LIST> Begin of a list, items separated by blank lines.
            Can have additional commands as [arg].
            Knows about numbered lists, bulleted lists and
            definition-lists. See example.

  <LB>                  Linebreak in a list.

  <TAB>                 Real TAB (ASCII 9)

  <BREAK>               Divides a text in the appropriate way (column
            break, horziontal rules etc.). Normaly only for
            internal use.

  <NOI>                 NoIndent, normaly only for internal use.

  <HEAD>                Headline, internal

  <HEAD2>               dito

  <T>                   Title. Actually only a subtitle, but see example.

  <ST>                  Subtitle

  <M>                   Smallest title. See example.

  <NAME [arg]>          [arg] can be the author's name.

  <PIC name>            Includes picture "name". Gif only (technical
            reasons).
```

1.7 prog

You'll probably want to edit the files, or add new styles or targets. If you're a programmer, take the given files and base yours on them. If you're not a programmer, ask me; if it's not to complicated, i'll see what i can do about it.

It could be a good idea to place new styles or targets on the Aminet, so everyone can use it.

1.8 files

Put TextEngine.ged and TEM whereeever you like. Textengine needs the env-variable Textengine, which specifies the path of the TEM-directory (for the styles end targets). New modules are simply installed by putting them in the appropriate directory.

You will probably have to edit the HTML-style, because it wants to insert a footer-file (currently it loads: HOME:store/html-footer.txt).

textengine.styles contains most of the PGS3 styles you'll need. But there are some missing, which you might have to add following the given examples. Edit it to fit your needs. PageStream3: is a good directory for this file.

The examples aren’t necessary for anything but showing you the way textengine works.