

Extracting Information from (L^A)T_EX Source Files

Jean-Michel Hufflen

TUG 2022

24 July 2022

Contents

Introduction

Description

Discussion

Presently

Conclusion

Introduction

Description

Discussion

Presently

Conclusion

Contents

Introduction

Description

Discussion

Presently

Conclusion

TEX...

is a wonderful tool for typesetting texts \Leftarrow well known;

Contents

Introduction

Description

Discussion

Presently

Conclusion

is a wonderful tool for typesetting texts \Leftarrow well known;
knows only its own formats \Leftarrow well known, too.

[Contents](#)[Introduction](#)[Description](#)[Discussion](#)[Presently](#)[Conclusion](#)

is a wonderful tool for typesetting texts \Leftarrow well known;
knows only its own formats \Leftarrow well known, too.
Some information belonging to (L^A)T_EX source files may be
usable for other purposes than typesetting, e.g., generating
metadata for Web search engines.

[Contents](#)[Introduction](#)[Description](#)[Discussion](#)[Presently](#)[Conclusion](#)

is a wonderful tool for typesetting texts \Leftarrow well known;
knows only its own formats \Leftarrow well known, too.

Some information belonging to (L^A)T_EX source files may be
usable for other purposes than typesetting, e.g., generating
metadata for Web search engines.

(L^A)T_EX's commands can do such jobs, but this is *misuse* and
complicates the writing of classes. T_EX has not been
designed for that, it is preferable to use modern programming
languages, with more suitable structures.

[Contents](#)[Introduction](#)[Description](#)[Discussion](#)[Presently](#)[Conclusion](#)

Functional programming

Functions are first-class objects, as other data.

Functional programming

Functions are first-class objects, as other data.
Functions can be results of a computation.

Functional programming

Functions are first-class objects, as other data.
Functions can be results of a computation.
So we can easily write *generators* of functions.

Functional programming

Functions are first-class objects, as other data.

Functions can be results of a computation.

So we can easily write *generators* of functions.

Scheme \Leftarrow elegant, data and programs have the same format, as in any Lisp dialect.

Building a *parsing function*

(g-mk-tex-parsing-f *directive* ...)

All the *directives* are grouped, 'compiled' into a function ready to parse a source file.

Directives

```
(g-retain-command command-name arg-nb optional-arg?  
                 top-level? recursive? preamble?  
                 occ-nb-info function)
```

where:

command-name is the name of the command to be caught;

arg-nb is the argument number of this command;

optional-arg? is true if the first argument is optional,
surrounded by square brackets, false otherwise;

top-level? is true if this command is to be searched only at
the top level, false otherwise;

recursive? is used when `\input` commands are encountered:
if it is true, corresponding files are searched
recursively, otherwise such an `\input` command is
just skipped;

preamble? stops searching after a preamble if it is bound to
true; otherwise, goes on.

occ-nb-info may be bound to:

- ▶ 0 or the false value (`#f`): the command should not appear within the file, this is checked;
- ▶ a positive integer n : the first n occurrences of this command are processed, and following ones are ignored;
- ▶ the true value (`#t`): all the occurrences of this command are processed;

function the Scheme function to call, it *must* have the same number of arguments than `\command-name`.

Contents

Introduction

Description

Discussion

Presently

Conclusion

Directives (*con'd*)

```
(g-retain-match command-name s top-level?  
                recursive? preamble occ-nb-info  
                function)
```

is used when `\command-name`'s arguments are expressed by means of a pattern, e.g., `"#1\endcsname"` for the `\csname` command. `s` is a string bound to such a pattern, the other arguments have the same meaning than `g-retain-command`'s.

Contents

Introduction

Description

Discussion

Presently

Conclusion

Directives (*con'd*)

```
(g-retain-match command-name s top-level?  
               recursive? preamble occ-nb-info  
               function)
```

is used when `\command-name`'s arguments are expressed by means of a pattern, e.g., `"#1\endcsname"` for the `\csname` command. `s` is a string bound to such a pattern, the other arguments have the same meaning than `g-retain-command`'s.

The arguments of corresponding functions in Scheme are strings in both cases.

[Contents](#)

[Introduction](#)

[Description](#)

[Discussion](#)

[Presently](#)

[Conclusion](#)

Result's result

`g-mk-tex-parsing-f` builds a function that applies to a filename and returns:

- `false` if something went wrong, or a forbidden command is included into the file;
- `true` in all other cases.

Result's result

`g-mk-tex-parsing-f` builds a function that applies to a filename and returns:

`false` if something went wrong, or a forbidden command is included into the file;
`true` in all other cases.

You have to update your own structures when a file is parsed. If an error occurs, they may be in an inconsistent state.

Destructuring

These functions can be used to destructure an optional argument— s, s_0 are strings—:

Destructuring

These functions can be used to destructure an optional argument— s, s_0 are strings—:

`(g-parse-to-list s)` returns the elements of a comma-separated list;

`(g-parse-to-alist $s s_0$)` returns the successive associations of a comma-separated list whose elements are `key=value` pairs; if a key is given without a value, this missing value is replaced by s_0 .

In both cases, the original order is preserved.

Destructuring

These functions can be used to destructure an optional argument— s, s_0 are strings—:

`(g-parse-to-list s)` returns the elements of a comma-separated list;

`(g-parse-to-alist $s s_0$)` returns the successive associations of a comma-separated list whose elements are *key=value* pairs; if a key is given without a value, this missing value is replaced by s_0 .

In both cases, the original order is preserved.

Remark Note that `g-parse-to-list`, `g-parse-to-alist`, `g-retain-command` and `g-retain-match` are functions, whereas `g-mk-tex-parsing-f` is a *macro*.

Example

Considering a source text for L^AT_EX, give:

- ▶ the used options of the babel package,
- ▶ the title,
- ▶ the number of occurrences of the `\emph` command.

(Show.)

'History'

When I realised MIBIB_TE_X, I needed to know which options of the babel package were used. Only the preamble was to be searched.

'History'

When I realised MIBIB \TeX , I needed to know which options of the babel package were used. Only the preamble was to be searched.

The same to determine which encoding was used.

'History'

When I realised MIBIB $\text{T}_{\text{E}}\text{X}$, I needed to know which options of the babel package were used. Only the preamble was to be searched.

The same to determine which encoding was used.

One year and a half ago, I became the new editor of the journal of the French-speaking $\text{T}_{\text{E}}\text{X}$ user group.

'History'

When I realised MIBIB $\text{T}_{\text{E}}\text{X}$, I needed to know which options of the babel package were used. Only the preamble was to be searched.

The same to determine which encoding was used.

One year and a half ago, I became the new editor of the journal of the French-speaking $\text{T}_{\text{E}}\text{X}$ user group.

I decided to revise the class used for this journal and discovered that the previous class was used to build other files, such as metadata for Web search engines.

'History'

When I realised MIBIB \TeX , I needed to know which options of the babel package were used. Only the preamble was to be searched.

The same to determine which encoding was used.

One year and a half ago, I became the new editor of the journal of the French-speaking \TeX user group.

I decided to revise the class used for this journal and discovered that the previous class was used to build other files, such as metadata for Web search engines.

From my point of view, \TeX is intended for typesetting, and other tasks should be delegated to more modern programming languages.

'History'

When I realised MIBIB \TeX , I needed to know which options of the babel package were used. Only the preamble was to be searched.

The same to determine which encoding was used.

One year and a half ago, I became the new editor of the journal of the French-speaking \TeX user group.

I decided to revise the class used for this journal and discovered that the previous class was used to build other files, such as metadata for Web search engines.

From my point of view, \TeX is intended for typesetting, and other tasks should be delegated to more modern programming languages.

I could have used Lua. . .

'History'

When I realised MIBIB \TeX , I needed to know which options of the babel package were used. Only the preamble was to be searched.

The same to determine which encoding was used.

One year and a half ago, I became the new editor of the journal of the French-speaking \TeX user group.

I decided to revise the class used for this journal and discovered that the previous class was used to build other files, such as metadata for Web search engines.

From my point of view, \TeX is intended for typesetting, and other tasks should be delegated to more modern programming languages.

I could have used Lua... but Lua \TeX was unable to process some texts designed for pdf \TeX or X \LaTeX .

Simple extractions—e.g., a title—must be simple.

Simple extractions—e.g., a title—must be simple.
If addressing occurrences of a T_EX command inside a source text, it should be possible to catch it.

Simple extractions—e.g., a title—must be simple.
If addressing occurrences of a T_EX command inside a source
text, it should be possible to catch it.
But more experience will be needed.

Current state

Some bugs to fix for the pattern-matching of TEX command's arguments.

Current state

Some bugs to fix for the pattern-matching of \TeX command's arguments.

The other points seems to be OK.

Current state

Some bugs to fix for the pattern-matching of T_EX
command's arguments.

The other points seems to be OK.

Available as a Scheme library.

Ending

For many years, we have seen that in addition to T_EX's works, many other tasks are more related to 'classical' programming. In particular, that is why LuaT_EX emerged.

Ending

For many years, we have seen that in addition to T_EX's works, many other tasks are more related to 'classical' programming. In particular, that is why LuaT_EX emerged. We relate our work to avoiding *information redundancy*.