Reference

Alejandro "alkeon" Castilla

Contents

1 Introduction

`Tristes armas`

` si no son las palabras`

`Miguel Hernández`

Tedi was created from the need of a faster development of my own website (more than five ago). I know that there were plenty of CMS that just work. Some years later I divided files into two parts (content and layout). First came Maquetador, as a first try to update faster the website. This made me realize that I could create a new markup language that compresses content files in a cleaner way.
With this idea the language started as a lightweight and easier HTML. As it got new tags and programs, I started using it also as an alternative to Latex. (I don't like Latex). This generated the last idea for the language, a language meant to be converted to others called Tedi.
Tedi has been in use more than five years, and it seems that won't be stopped that easily. There were some changes really interesting that I'll detail when they get available.
First change was erasing !head and !end tags. This allowed detect Tedi files. Also image tag was modified from (image.png) to ([Alternative text] image.png). Next changes were create new tag called metatag with line break. This allowed user to add line breaks or not to lines with metatags.

2 Tedi

Tedi is a markup language that can be used for note taking and create a scientific paper with the smallest set of tags possible.
Tags
Tags are a group of elements that allows an algorithm use an specific rule inside a region.

2.1 Basic tags

These tags were the first subgroup that were implemented. They are the smallest subset of functionality of a markup language without losing readability.

2.1.1 Titles

Titles help the writer to organize better the information in file.
Syntax:

```
#First
```

```
##Subtitle
```

```
###Subsubtitle
```

It will be converted only if it's the first tag of the line. Next line won't be converted as a title.

```
[(https://example.org/) ##Example)
```

Next line will be converted as a title.

```
##[(https://example.org) text)
```

2.1.2 Links

Links were meant for connect one part of the document with other part or a website. Tag is used like this:

```
[(Link) Text]
```

By default all links are document internal. As an example:

```
[(https://example.org) Example] is very useful.
```

Previous line is converted like this Example is very useful.

2.1.3 Images

Tag is

```
([Text describing image] image.png)
```

Preferred formats are PNG, JPG and SVG.

2.1.4 Containers

Containers are tags that divide file into pieces that can be used for new user specific rules. Similar to divs in HTML. It's written like this

```
{(class) text}
```

Class is an identifier for that specific part of the document.

2.1.5 Lists

Lists are represented with three tags.

- The first tag for a list is `__`. Without it lists aren't going to start properly.
- Every item must be preceded with `--`.
- The last tag is `,,`. Used it when want to end the list.

All lists by default are unordered and if prefer using an ordered list you must use a other language specific tag.

```
--

-- First item

-- Second item

,,
```

Results are:

- First item
- Second item

2.2 Control tags

Control tags are useful to control the conversion process of the text. This is the most difficult part of the language but for every user luck, they aren't that necessary except quotes and metatags. These tags should be used with care because some could trigger some incompatibilities with some conversions. Every tag in this type are only valid if are written at the start of the line.

2.2.1 Embed tag

This tag allows embed code from other languages. It avoids conversions and leave it unchanged. As an example

```
<> \LaTeX
```

This will get to
$\LaTeX$
This tag makes special limitations to conversions. It was designed to take advantage of Latex math formula.

2.2.2 Comments

Comments were added because copyright, add some specific details and to point out other languages tags used in embed tags.
Example:

```
<! Comments are social networking of programmers.
```

Missing because it's a comment read source code at git page.

2.2.3 Quotes

Quotes are partial control tags. Valid use is when line start with a quotation mark and ends with other quotation mark. Between this characters, containers, links and images tags aren't converted and are leave like they were written.

```
"4 + [(3 + 6) * 2]"
```

It will show like this:
4 + [(3 + 6) * 2]
If wanna add a line break to a quoted line just append a white space before last quotation mark.

2.2.4 Metatag

Metatag is a tag created at first public version of this document. This tag define a piece of text that tags won't be checked.
Syntax is

```
<Metatag is written with < and allows you to

write |, # and a group of tags without converting them
```

Conversion is like this:

```
Metatag is written with < and allows you to

write |, # and a group of tags without converting them
```

Metatag with line break is:

```
<+ This is a metatag '<+' with line break
```

As a result it shows this:

```
 This is a metatag '<+' with line break
```

2.3 Special tags

Special tags were the last tags added to Tedi. In spite of being the last one, some of them are really necessary to modern languages like tables and dividing parts of document into some pieces.

2.3.1 Tables

Simple and easy to use tables:

```
Tables are text enclosed between | characters.
```

They can be used like this:

```
| 1 | 2 |

| 3 | 4 |
```

That will be converted to:

1	2
3	4

2.3.2 Line break

Writer can force a line break leaving a white space as last character.
Example:

```
One line

Other
```

Result:
One line
Other

2.3.3 Insert

This tag is a way to combine multiples files into a single file. Only Tedi files are allowed.
Syntax:

```
 +document.te
```

Before filename must have a '+' symbol without white spaces and user must ensure file exist. Inserted file is preprocessed to avoid possible fails.

2.4 Recommended style

When writing a Tedi document, it's recommended follow a coding style for a better performance. All users should use Tedi as they want to but it's easy to work with some rules.

- Do not use other specific language line breaks, user Tedi line breaks every time you can.
- Limit titles sublevels to 4, it isn't recommended more sublevels.
- Prefer quotes over metatags.
- Limit your metatags line to 80 characters.
- Use .te extension to Tedi files.
- Use comments to point out which language is use in embedded tags.
- Never insert a Tedi file into a Tedi file.