HTMLFromTeXBooks Documentation

March 15, 2022

Contents

1 Module `Lib` : Library used for the project

Author(s): Charlotte Thomas

Type declaration and aliases useful

```
type chapter =
  | Nil
  | Chap of string * string * string list * int
```

Chapter type : A chapter is Nil Or A tuple of a string (name), a string (the chapter text) a string list (the lines of chapter text) and an integer (the number of the chapter)

```
type glossary =
  | Nul
  | Entry of string * string * string
```

Glossary type : It can be a Nil or an "Entry" : a tuple of a string (the key), a string (the name) and a string (the definition)

```
type book = chapter list
```

Alias : a book is a list of chapters

Utility

```
val string_to_list : String.t -> char list
```

Transform a string to a list of character

```
val glossaries : (string, string * string) Hashtbl.t
```

The Hashtbl with the (name,defition) of glossaries by their key

```
val print_chapter : chapter -> unit
```

A little function to print a chapter

```
val read_file : ?joining:string -> string -> string
```

A function reading the inputed file and outputs a string of the lines separated by a line break \n

`val glossary_exists : string -> bool`

Alias for Sys.file_exists

`val glossary_provided : unit -> bool`

Returns true if the length of the glossary is greater than 0 and false otherwise

`val write_to_file : string -> string -> unit`

Write a string in a file

Book functions

`val parse_book : ?book:bool -> string -> string list`

Parse a string into a list of string representing chapters (book) or section (article)

```
val transform_list_of_chars_to_string_without_the_newline :
  char list -> string
```

Returns a string from a char list, ignoring the lines break

`val extract_chapter : string -> int -> chapter`

Creating a chapter object from a string and the number of the chapter

`val extract_chapters : string -> chapter list`

Reads a file, Parses the book and returns a list of chapters with the extract_chapter function

`val recognize_gls : char list -> (string * string) * string * char list`

takes the char list and returns the (name,description) of the given glossary entry and the remaining of the list

`val replace_generalities : string -> string`

Replaces the common found structure in a TeX file

`val parse_more_than_generalities : String.t -> string`

Parse more complex command : \textit, \textbf and \gls

`val transform_chapter : chapter -> chapter`

Transform a chapter with the functions replace_generalities and replace_more_than_generalities

`val chapter_to_string : chapter -> string`

prints a chapter

`val not_is_empty : string -> bool`

returns false if a string is empty or a newline

Glossary functions

`val parse_glossary_entry : String.t -> unit`

Reads an entry, extract the description, name and definition, and adds it to the Hashtbl

`val parse_glossaries : string -> string list`

Takes a glossary file and parse it in a list of string

`val total_glossaries : string -> unit`

Combines parse_glosaries and parse_glossary_entry

`val prints_glossary : unit -> string`

Transform the glossary to an HTML file

Writing book function

`val print_table_of_content : chapter list -> string`

Generate the table of content HTML

`val prepare_body : ?name:string -> chapter list -> string -> string`

Prepare the body for the content

`val write_core_chapter_to_file : chapter -> string -> unit`

Write single chapter in an HML file

`val write_book : ?name:string -> string -> chapter list -> unit`

Write the book in an HTML file