The testhyphens package

Version number v.1.1, last revised on 2026/06/29.

Claudio Beccari
email: claudio.beccari(at)gmail.com

Abstract

This small file implements some code that was already published on TUGboat, but it is adapted to LaTeX engines, and it is usable as a specific environment. It helps those TeXers who create hyphenation patterns, as well normal LaTeX users, in order to test the correctness of the hyphenation points of words lists. It works with `pdflatex`, `xelatex`, and `lualatex`.

Contents

1 Introduction

This small package introduces the environment `checkhyphens` and the command `\testhyphens`. This environment may set the hyphenation parameters to any specific values, even different from those setup for the language in use; it processes a list of one or more space delimited words, and typesets them one per line with hyphens between each syllable. If typesetting this list takes place within a *multicolulmn* environment, it renders much easier to read long lists.

The hyphenation patterns of a specific language to be tested, may be those of the default language before entering the environment, or can be those of another language specified with `\selectlanguage` within the environment.

The environment *checkhyphens* accepts also an optional argument to set the hyphenation parameters, and a body consisting of a list of one or more space separated words; if the user just pastes a text copied from another document, and this text includes punctuation marks, such signs are ignored, but are not taken off from the printed list.

The main code was published long ago by Victor Eijkout, in "The bag of tricks", *TUGboat* 14.4 (1993), p. 424. He himself states that the code was created by Jonathan Kew; Victor Eijkout just added an empty `\discretionary{}{}{}`. The idea came to him from the code created by Oliver Schoett that is used for the

hyphenation exception list of TUGboat. Credit for the code goes completely to the authors mentioned above; I just added some sugar to use that code in LaTeX documents for the benefit of hyphenation pattern creators, and of regular users who want to know how a certain word or the words of a certain sentence would be hyphenated by TeX and friends.

In 2018 a small extension package `xltxtra` by Will Robertson was added to the TeX system distribution that contains some fixes and facilities to be used when typesetting with XeLaTeX; it contains the definition of the macro `\showhyphens` macro; this macro was provided by Enrico Gregorio; it is supposed to show the hyphenation of a small word list when using XeLaTeX; unfortunately in these years, up to 2024, LaTeX has undergone to many changes that this macro does not work any more, except perhaps with few selected languages.

Of course this package does not tell the whole truth: in facts TeX and friends hyphenate words only when certain conditions are met; these conditions are stated in Appendix H of the TeXbook, which is and remains the primary reference for understanding the hyphenation process used by TeX and friends.

Sometimes users complain that certain words are not hyphenated; users might use this package and find out how their words are hyphenated; then what's happening if they are not hyphenated the right way? Simply those conditions are not met. Therefore the full understanding of the above mentioned Appendix H is fundamental before complaining about TeX not working as expected with hyphenation and possibly before raising a bug notice.

2 Incompatibility with the previous version 0.5a

In 2015 I had to change some definitions within the body of this package in order to avoid clashes mainly with the `french` language option of babel; in facts that option declares most punctuation marks as active characters. Unfortunately those characters receive global definitions, therefore they remain active even when they should not perform any action.

It is necessary to specify that the previous version used the colon as a parameter separator; while, in order to avoid conflicts with `french`, now the parameter separator is a simple hyphen.

Defining active characters when using `polyglossia` may produce other incompatibilities that are easily circumvented by specifying languages in a different order. In facts, if you specify languages as such

```
\setmainlanguage[babelshorthands]{italian}
\setotherlanguages{english,french}
```

an error appears when `french` sets up its parameters for the UNICODE apostrophe: the cause of this error derives from the use of the `italian` option `babelshorthands`; of course if this option is not specified, no problems arise. But if you interchange the declarations as in

```
\setotherlanguages{english,french}
\setmainlanguage[babelshorthands]{italian}
```

no problems arise.

3 Changes from version 0.7

When version 0.7 was published, command `\testhyphens` worked regularly also with XeLaTeX; this new version maintains the environment ***checkhyphens***, but lets down `\texthyphens` because this command raised errors that I was not able to repair. Something was changed in the XeLaTeX format file that I could not spot.

4 Changes from version 0.8

Version 1.0 recovers the full functionality of both the ***checkhyphens*** environment and the `\testhyphens` command.

The described failure for version 0.7 was spotted also by Tristan Miller who got the right response from `stachexchange` with the correcting patch. I have to thank David Carlisle who specified the patch and Tristan Miller who was so kind to transmit it to me. The patch was necessary to complete a hook code; it was not necessary before the recent LaTeX kernel upgrades, but it has become essential in order to have the code working correctly with the several hooks that were added to the kernel code.

5 Usage

This package has been tested with `pdflatex`, `xelatex`, and `lualatex`; for `lualatex` a package `showhyphens` exists that marks the hyphen points of an entire document with thin red vertical strokes at the possible break points. Users may use the package they prefer by taking into consideration their needs with respect with the features and the functionalities of package testhyphens compared to those of showhyphens.

Notice that even normal Plain TeX and LaTeX have available the command `\showhyphens` but, differently from this package, such command works only when the typesetting engine is `tex` or `pdftex`; moreover the result appears on the console window and in the `.log` file and it is not written down in the output document as a record that remains available to the user. This package testhyphens, on the opposite, prints the result in the output document and works with the typesetting engines `pdftex`, `xetex`, and `luatex` as expected.

This simple package is loaded just with the usual statement

```
\usepackage{testhyphens}
```

The package new version defines for the end user both the ***checkhyphens*** environment and the `\testhyphens` command: their syntax is the following:

```
\begin{checkhyphens}[⟨hyphenation parameters⟩]
⟨space separated word list⟩
\end{checkhyphens}

\testhyphens{⟨space separated word list⟩}
```

where ⟨*hyphenation parameters*⟩ is a *hyphen separated list* of two digits that in order will be locally assigned by the environment respectively to `\lefthyphenmin` and `\righthyphenmin`; the ⟨*space separated word list*⟩ is self explanatory. Nevertheless it is good to remember that the environment isolates single words by using a space as a word separator; this requires that the list is preceded and followed by spaces or end of lines, Actually the initial and final spaces are necessary when the opening statement is on the same source line of the first word and the closing statement is in the same source line of the last word. The environment is more performant than the command because it allows hyphenation testing even with hyphenation parameters different from the default ones set up by the current language settings.

6 Using hyphenation parameters

The hyphenation parameters are just `\lefthyphenmin` and `\righthyphenmin`; they represent the minimum number of characters that the first and, respectively, the last word fragment must have before or until hyphenation takes place; for example in English the default values are 2 and 3; in Italian the default values are 2 and 2; in Greek the default values are 1 and 1.

Let us make an example: the word *idea* is spelt the same in English and in Italian, although it is pronounced differently; but the word is *not hyphenated at all* in both languages. According to the Italian rules its grammatical word division is *i-de-a*; in American English the grammatical word division is *i-dea* (according to Wikipedia; some other sources state that the word is indivisible); in any case the initial and/or the terminal syllables are too short to comply with the above mentioned TEX hyphenation parameters, and TEX refrains from hyphenating this word. On the opposite, in Greek ιδέα is hyphenated by TEX as ι-δέ-α.

If you use ***checkhyphens*** with the optional hyphenation parameters specified to 1 and 1 in Italian or in English, as in

```
\begin{checkhyphens}[1-1]
idea
\end{checkhyphens}
```

you get *i-dea* in English, and *i-de-a* in Italian, just as in Greek.

This happens because the Italian patterns were created by hand and tested with parameters 1 and 1, even if for typesetting purposes the parameters are set to 2 and 2. In English the patterns were created by means of the `patgen` program where parameters 2 and 3 were pre-set.

Typographically speaking the values 1 and 1 for Italian are almost unnecessary, but in certain difficult narrow-measure texts the user has the possibility to locally set the first and/or the second parameter to 1, and solve that particular instance of problematic narrow-measure typesetting.

7 Examples

Here we show some actual examples by using the ***checkhyphens*** environment.

While typesetting in English we might verify the hyphenation of some words. We set the source file with the following lines

```
\begin{{chechhyphens}
```

```
he analyses the samples and submits the analyses to the federal office
\end{checkhyphens}
```

The result produced by this package, typeset within a four-column *multicols* environment, is the following:

he	sam-ples	the	the
anal-y-ses	and	anal-y-ses	fed-eral
the	sub-mits	to	of-fice

Notice the two instances of the word `analyses`; in English TEX hyphenates them the same way, but one of the two words is hyphenated the wrong way; unfortunately there is no way for TEX to distinguish homographic words that are not homophonic ones. In plain terms the verb "analyses" is stressed on the 'y', while the noun "analyses" is stressed on the second 'a'; the standard phonetic hyphenation rules for English hyphenate the verb as shown, but TEX works only on the spelling and not on the sound; furthermore it has no means to find out if a word plays the rôle of a noun or of a verb.

In this and similar situations (for example: "the record" and "to record") the user should check a good reliable dictionary and possibly insert an explicit discretionary hyphen `\-`. A good although oldish such dictionary would be the *Webster's New Word Speller divider* — The 33 000 most used words spelled and syllabified. World & Collins World, (1971).

In Italian any grammar book states the rules that are valid for all Latin derived words and are applicable also to technical scientific words, although a etymologic hyphenation is allowed and is more adequate to the context; words derived from Greek (very common in medicine) and neologisms with foreign roots, do not find specific rules either in grammars and in most dictionaries; or better, if a word ends with one or more consonants, this consonantal group must not be separated from the word last vowel.

While typesetting in Italian it is possible to insert into the source file a set of lines such as these:

```
\begin{checkhyphens}
ebbe la bell'idea di viaggiare in scooter
\end{checkhyphens}
```

that produces the following result:

eb-be	bel-l'i-dea	viag-gia-re	scoo-ter
la	di	in	

Notice that "idea" cannot be hyphenated by TEX when it is an isolated word; but in Italian the apostrophe is used (also) to mark a vocalic elision, therefore for hyphenation purposes, even if it is not a letter, it is assigned a lowercase code, and it becomes part of a word, in our case the string "bell'idea" becomes a single word. Therefore the patterns that involve the apostrophe produce the result shown above, where the first grammatical hyphen point shows up again. This is so with `xelatex`; in `lualatex` the normal ACII apostrophe is not assigned a lowercase code, therefore with `lualatex` "`bell'idea`" is not hyphenated while "`bell'idea`" is correctly hyphenated; the Unicode single closed upper quotation

mark is used instead of the ASCII character available with any keyboard. These characters look the same (at least with some OpenType fonts) but have different Unicode code points.

While typesetting in Greek we might insert in the source file the following code:

```
\begin{checkhyphens}[1-1]
επεξεργάζεται αρχεία τα οποία περιέχουν τους
χαρακτήρες που υπάρχουν στο πληκτρολόγιο του
υπολογιστή σας
\end{checkhyphens}
```

and we would get:

ε-πε-ξερ-γά-ζε-ται	πε-ριέ-χουν	υ-πάρ-χουν	υ-πο-λο-γι-στή
αρ-χεί-α	τους	στο	σας
τα	χα-ρα-κτή-ρες	πλη-κτρο-λό-γιο	
ο-ποί-α	που	του	

while, had we set the hyphenation parameters different from the default values 1 and 1, such as in:

```
\begin{checkhyphens}[2-2]
επεξεργάζεται αρχεία τα οποία περιέχουν τους
χαρακτήρες που υπάρχουν στο πληκτρολόγιο του
υπολογιστή σας.
\end{checkhyphens}
```

the result would be:

επε-ξερ-γά-ζε-ται	πε-ριέ-χουν	υπάρ-χουν	υπο-λο-γι-στή
αρ-χεία	τους	στο	σας.
τα	χα-ρα-κτή-ρες	πλη-κτρο-λό-γιο	
οποία	που	του	

The pattern files for monotonic Greek used in these examples were rebuilt with Greek letters – avoiding the Latin transliteration originally used with the Latin Modern conformant Greek fonts encoded with the `LGR` encoding; that was done for the benefit of non-Greek Hellenists who would use their Latin national keyboards. The above mentioned upgrading was done by Dimitrios Filippou in 2011; since then no modifications were added, except for the patterns to be used with `xelatex` and `lualatex`.

For OpenType fonts Antonis Tsolomitis produced a new font set that assembles the Latin Modern Fonts together with the CBfonts Greek ones and the Cyrillic ones plus several other fonts for other scripts; this remarkable pretty complete OpenType font is named "New Computer Modern" (documentation with `texdoc NewCM`) that uses the `fontsetup` package for selecting its varieties; this package is described by the same documentation. This guide was typeset with `lualatex`, using this new font, and the above Greek language examples are correct.

The first Greek example shows very clearly the effect of `\lefthyphenmin=1` compared to the second example where it is `\lefthyphenmin=2`. On the other hand, the fact that some hyphen points may be missing is certainly a feature that precludes full hyphenation, but it does not produce hyphenation errors.

The use of the first version of this package already allowed to highlight some features/bugs in the Greek hyphenation patterns and to correct them.

8 The code

The following code has very few modifications compared to the original code published by Victor Eijkout. For my own benefit I changed the declaration name from `\printhyphens` to `\t@sthyphens`; with an at-sign internal name it is less likely that this command is redefined by some user. I also moved the command `\offinterlineskip` from its penultimate position in the original code to the first command to be executed within a `\vbox`; in this way I avoided certain cases in which the lack of interline skip would remain active after closing the environment.

The code is based on treating the inter word space as an active character that plays different rôles according to its position; it lets the environment isolate the words to be hyphenated and open suitable boxes, that are closed at the moment of processing the following word; the `\everypar` token-list distinguishes the various actions to be performed at each step.

Another advantage is that the environment processes one word at a time, therefore the word list may be arbitrarily long; while creating the patterns for the Albanian language I initially worked with a half dozen pages taken from a book, about 7000 words. Initially I deleted all duplicate words then, while fine tuning the pattern set, I deleted a certain number of the test words; eventually a good set of patterns was created under the supervision or a learned Albanian friend, MS in Albanian literature.

But the trick of this software is to set each word within a vertical box where the measure (the text width `\hsize`) is zero; in this way each word is hyphenated at the first hyphen point, but the word fragment that remains in the following line is still too long and gets hyphenated again; the process is repeated again and again until the initial string is exhausted. The contents of the `\vbox`, that now contains the syllables one per line, is reassembled to form one line, and this line is output within an `\hbox` and is therefore the box typeset in vertical mode; this is why the command must be issued in vertical mode, so as to start processing the word list with the right foot; this is also why the word list must start and end with spaces; the first space or end of line before the first word opens the first box, and the last space or end of line after the last word closes the box and outputs its contents in horizontal mode..

```
\def\t@sthyphens{%
  \everypar{\setbox0\lastbox
  \setbox1\hbox{\strut}\vbox\bgroup
  \offinterlineskip
  \everypar{\setbox0\lastbox \nobreak\hskip\z@}\dimen0=\hsize
  \hsize=\z@ \hfuzz\maxdimen
  \def\par{\endgraf \hsize=\dimen0\getlastline\endgraf
  \egroup\endgraf}}\breakafterword
}

\def\breakafterword{%
\catcode`\^^M=13 \catcode`\ =13
}

```

```
{\breakafterword\gdef^^M{\par}\global\let ^^M}

\def\getlastline{\setbox0\lastbox
\ifvoid0
  \let\next\nomorelines
\else
  \unskip\unpenalty
  \setbox1\hbox{\unhbox0\strut\discretionary{}{}{}%
  \unhbox1}\let\next\getlastline
\fi
\next}

\def\nomorelines{\unhbox1}
```

The above code is substantially the original one; where the internal declaration control sequence `\t@sthyphens` is defined. Now we define the ***checkhyphens*** environment where we make sure that the whole process of the word list contained in its body takes place in vertical mode. In oder to process the optional argument for setting the hyphenation parameters, we define a delimited argument macro `\s@thyphenpars` that uses the *hyphen* as the argument delimiter.

```
\newenvironment{checkhyphens}[1][\lefthyphenmin-\righthyphenmin]{%
\@tempcnta=\lefthyphenmin
\@tempcntb=\righthyphenmin
\s@thyphenpars[#1]\par
\bgroup\t@sthyphens
}{%
\egroup\par
}

\let\testhyphens\t@sthyphens
\def\s@thyphenpars[#1-#2]{%
\@tempcnta=#1\relax
\@tempcntb=#2\relax
\unless\ifnum\@tempcnta=\lefthyphenmin \lefthyphenmin=\@tempcnta\fi
\unless\ifnum\@tempcntb=\righthyphenmin \righthyphenmin=\@tempcntb\fi
}
```

That's all. Happy TEXing.