The FixLtxHyph package
A small fix in order to hyphenate emphasised words after a vocalic elision

Claudio Beccari

v.0.5 2024-12-28

Contents

Abstract

This file fixes a small feature of the hyphenation algorithm used by the TEX system typesetting engines that manifests itself only with those languages that use the apostrophe for marking a vocalic elision. This small package was set up to fix this little undesirable feature in Italian, but it was extended to Catalan, French, the fourth official Swiss language Rumantsch Grischun (Romansh in English) and the Regional Language Friulan, spoken and written in North Eastern Italy. This fix operates correctly with pdflatex, lualate, and xelatex.

1 What is the feature to be fixed

The five languages Catalan, French, Italian, Romansh, and Friulian use the apostrophe for marking the vocalic elision of the ending vowel at the end of prepositions, articles, articulated prepositions, definite adjectives, and other words playing similar rôles when they just precede nouns, adjectives, verbs, numerals, that start with a vowel. Probably there are other languages that use the apostrophe in a similar way. I can easily upgrade this small package if LATEX users of other languages let me know about such languages.

This feature is common to most Romance languages (from West to East) from Catalan and Valencian, to French, Langue d'oc, Occitan, Provençal, Vivaroalpin, Italian, Piedmontese, Lombard, Romansh, Ladin, Friulian; up to now only Catalan, French, Italian, Romansh, Friulian, Piedmontese, and Occitan are handled by the TEX-system programs; at the same time most of these languages are minority ones and are being protected by local legislation or are supported by specific cultural or linguistic institutions; Romansh has got a national/federal legal status in Switzerland and is being used in legal and official documents in the whole Swiss Confederation, not only in its area of everyday use, the Kanton Graubunden or Canton Grigioni or Chantun Grischun (where seven Romansh varieties are

being spoken, besides Swiss German, Italian, and other languages). The Friulian language has an official regional status in the North-eastern Italian Region Friuli-Venezia Giulia.

This spelling rule is very rigorous in French; I suppose it is also a rigorous rule in Catalan, Romansh, and Friulian but I am not so familiar with these languages even if I can understand them while reading texts written in these languages. In Italian it used to be a rigorous rule many years ago, but nowadays it is less frequently used when plurals are involved. Nevertheless apostrophes are practically the only analphabetic sign you see in an Italian texts besides punctuation and quotation marks.

In order to hyphenate correctly these word combinations all such languages have to declare the apostrophe, that has a category code of 12, as a glyph with non zero lower case code. In facts all such languages declare:

```
\lccode‘\’=‘\’
```

or something equivalent. With this little trick, the typesetting engine considers the apostrophe as a valid word character and treats the whole string as a single word; the hyphenation patterns of these languages, of course, take into consideration also the apostrophe so that the resulting correct line breaks are easily found:

Catalan	`d’aquesta`	d’a-ques-ta
French	`l’électricité`	l’élec-tri-ci-té
Friulian	`l’arbul`	l’ar-bul
Italian	`dell’eleganza`	del-l’e-le-gan-za
Romansh	`l’identitad`	l’i-den-ti-tad

So where is the problem? It emerges when the second part of the string is emphasised, because in this case no hyphenation takes place:

Catalan	`d’\emph{aquesta}`	d’*aquesta*
French	`l’\emph{électricité}`	l’*électricité*
Friulian	`l’\emph{arbul}`	l’*arbul*
Italian	`dell’\emph{eleganza}`	dell’*eleganza*
Romansh	`l’\emph{identitad}`	l’*identitad*

This behaviour is easily explained, so that it is not to be considered a bug, but a feature; a feature that is annoying only when using the above named languages.

The point is that all TEX system typesetting engines consider a word to be that character string starting after a character invalid in a word and finishing with the first token invalid in a word. Notice that when the hyphenating algorithm comes to work the command `\emph` has already been expanded and it ends up with the qualifications of the selected font; therefore a string such as ␣`d’aquesta`␣ (starting after a space and ending before the following space) is made up of valid characters; but ␣`d’\emph{aquesta}`␣ is a “word” starting after a space and ending before a space, but containing a font change. And this makes the word invalid for hyphenation.

The TEXbook is clear on this respect:

> If a suitable letter is found [as a starting character], let it be in font f. [...] TEX continues to scan forward until coming to something that's not one of the following three “admissible items”: (1) a character in font f whose `\lccode` is not zero; (2) a ligature formed entirely from

> characters of type (1); (3) an implicit kern. [...] Notice that all these items are in font f.

This was a specific programming choice decided by Donald E. Knuth together with Frank Liang, his PhD student who developed the hyphenation algorithm implemented in the typesetting engines of the TEX system[1]. All such decisions are a compromise between accuracy and speed. Remember also that at the beginning `tex` the program was used essentially with English, a language that does not use accented letters and uses elision in a much different way as the ones we are speaking here. The problem did non exist and, I suppose, it will never exist in English.

2 The solutions

As a compromise I decided to solve the problem in an automatic way only when the second part of the "word" to be hyphenated is *emphasised*. I assume it is the most frequent situation, although no one can avoid thinking to other situations; for example: the second part of such "word" after the apostrophe is bolded, is coloured, is written in another font selected on purpose or is in another alphabet, is in italics (with no automatic inclination switching); in such cases the solution is manual and remains manual, because there are too many possibilities and it is cumbersome to deal with all of them.

But manual or automatic, how should we proceed? Simply we must convince the typesetting program that the starting letter must not be the start of the part preceding the apostrophe, but what follows it.

This is simple: it suffices to put after the apostrophe an unbreakable, zero width glob of glue so that TEX starts looking for a potential starting letter after the glue. Therefore the manual solution consists in defining a short macro such as the following one:

```
\newcommand\hz{\nobreak\hskip\z@skip}
```

or, if you want to avoid setting this short command into a personal `.sty` file, simply change `\z@skip` with `0pt`. You will then have to modify the font changing phrase into something such as:

```
... d'\hz\textbf{aquesta} ...
```

The `\hz`, whose name reminds the phrase "Horizontal skip of an unbreakable Zero width glob of glue", finishes the preceding word and sets the grounds for starting the search of a new starting letter of another word; it will be found after the font selection code introduced in the horizontal list by the selected font identification.

The automatic solution, on the opposite, implies a small but substantial modification of the `\emph` command. In facts the text command uses the text declaration `\em`; on turn `\em` is a robust command, that is defined as `\protect\em␣`: it would be very unwise to modify a protected command, so it is necessary to modify the `protect`ed one, and this operation is not trivial because of the space in this macro name.

In any case if we find out how, we must add `\hz` to the definition of `\em␣` before its substitution text, so that the TEX search of the first character of a real word starts at the end of the substitution code.

[1] I have been told that LuaTEX developed a different algorithm that eliminates this feature.

This small package does exactly this only with the `\emph` command. Its functionalities actually are in force for any language, non only for the above named languages.

The `\hz` command is globally available to the user, so that when this package is loaded, the manual solution remains valid in any situation, as, for example, for the first line of a list item for the text that follows the `\item` command and its argument. It is necessary, especially within a *description* environment, because sometimes the item mandatory argument entry might be pretty long and the first line might require hyphenation at its end.

It has been tested with the above named five languages with both pdflatex, xelatex, and also with lualatex; and apparently it works as expected; it has been throughly tested in all situations with Italian; it should work properly also in French, in Romansh, and in Friulian; certainly it works with `utf8` text encoding. The adopted solution does not fiddle with active characters and therefore it does not interfere with the internal workings and settings of Catalan and other languages.

3 Installation

With modern TEX distributions these instructions are superfluous; should you need to manually install this package, download from CTAN in a scratch directory (possibly create one, and after finishing, delete the whole directory with its contents) run this file `fixltxhyph.dtx` through pdflatex; you get two files and move them as follows:

- Move all the files in the following directories on your disk; if you don't already have those directories, create them.
- These directories should be created in your personal `texmf` tree; if you don't have one, create it; how to do this and where to root it depends on your operating system; before doing any change to your hard disk, please read carefully the TeX Live or the MiKTeX documentations in order to find out what a personal tree is.
- Move `fixltxhyph.dtx` to `.../texmf/source/latex/FixLtxHyph/`;
- Move `fixltxhyph.pdf` to `.../texmf/doc/latex/FixLtxHyph/`;
- Move `fixltxhyph.sty` to `.../texmf/tex/latex/FixLtxHyph/`;
- if your distribution requires it, refresh the file name database.

You are now ready to use the package by simply invoking it in the preamble of your documents:

```
\usepackage{fixltxhyph}
```

4 Aknowledgements

I wish to thank Lorenzo Pantieri who tested the preliminary and the actual versions of this package and directly or indirectly helped debugging the code, especially in

the preliminary version that used active characters and was particularly buggy. Another big thank you to Enrico Gregorio who spotted the protection problem of the `\em` command.

5 The documented code

We start by identifying the package and the necessary format file:

```
\ProvidesPackage{fixltxhyph}[2024/12/01 v.0.5
  Small fix for hyphenating emphasised words]
\NeedsTeXFormat{LaTeX2e}[2022/01/01]
```

We need the package `etoolbox` in order to perform any action on control sequences that contain spaces in their names. We keep the old `\@ifpackageloaded` command, not because we love vintage commands, because since 2024 it is available the `\IfPackageLoadedF` command, that is more easily maintainable and does not require the empty argument, but because users sometimes work with vintage TEX-system installations.

First we define a very short command `\hz` in order to have available a handy command for inserting an unbreakable zero-width glob of glue in case we needed to manually do some sort of patching.

```
\newcommand\hz{\nobreak\hskip\z@skip}
```

Next we patch the `\em␣` command. To do so in an efficient way we need the `etooolbox` package.

```
\IfPackageLoadedF{etoolbox}{\RequirePackage{etoolbox}}
```

In a previous version we tested if one of the certainly (vulnerable languages that use the vocalic elision replaced by an apostrophe) was the current language; in this new package version 0.5 we omit such test because the patch is harmless even if the apostrophe does not imply any vocalic elision.

The next bit of code defines an alias in order to keep the original meaning of the declaration `\em`; in order to patch an alias to be used only in the redefinition of the same named new macro.

```
\letcs{\FLH@originalem}{em }
\RenewDocumentCommand{\em }{m}{\FLH@originalem\hz}
```

Notice that the `\em␣` defined by means of the LaTeX3 `\RenewDocumentCommand` is robust as all commands defined by means of this kind of LaTeX3 commands.

Eventually let us conclude with a comment: compared with the previous version 0.4 of this package the number of control sequences contained in this new version is drastically diminished from 84 to 16. This is one of the advantages gained by using the LaTeX3 language besides the `etoolbox` facilities.

This documented file is now finished and its final commands are issued:

```
\endinput
```

together with the `docstrip` command `\Finale` that allows to control if the final extracted code is complete.