Typesetting dropped capitals with LaTeX

Daniel Flipo
daniel.flipo@free.fr

1 Introduction

The file `lettrine.sty`[1], provides a command `\lettrine` which requires two mandatory arguments, and an optional one.

Adding `\usepackage{lettrine}` in the preamble of a document defines the command `\lettrine`, the simplest use of which is `\lettrine{<letter>}{<text>}`. It produces a dropped capital *<letter>* (2 lines high), followed by *<text>* typeset in small caps, and the rest of the paragraph is wrapped around the dropped capital.

The `\lettrine[<options>]{<letter>}{<text>}` command accepts various optional arguments to control the size and layout of the dropped capital and match the requirements described in the books

- "Lexique des règles typographiques en usage à l'Imprimerie nationale" troisième édition (1994), ISBN-2-11-081075-0,
- "Mise en page et impression" Yves Perrousseaux, ISBN-2-911220-01-3.

The parameters can be set using the `key=value` syntax:

- `lines=`*<integer>* sets how many lines the dropped capital will occupy (default=2);
- `depth=`*<integer>* sets the number of lines to be reserved under the baseline, this is meant for dropped capital with positive depth, like Q (default=0);
- `lhang=`*<decimal>* ($0 \leq$ `lhang` ≤ 1) sets how much of the dropped capital's width should hang into the margin (default=0);
- `loversize=`*<decimal>* ($-1 <$ `loversize` ≤ 1) enlarges the dropped capital's height: with `loversize=0.1` its height is enlarged by 10% so that it raises above the top paragraph's line (default=0);
- `lraise=`*<decimal>* does not affect the dropped capital's height, but moves it up (if positive), down (if negative); useful with capitals like `J` or `Q` which have a positive depth, (default=0);
- `findent=`*<dimen>* (positive or negative) controls the horizontal gap between the dropped capital and the indented block of text (default=0pt);
- `nindent=`*<dimen>* shifts all indented lines, starting from the second one, horizontally by *<dimen>* (this shift is relative to the first line, default=0.5em);
- `slope=`*<dimen>* can be used with dropped capitals like `A` or `V` to add *<dimen>* (positive or negative) to the indentation of each line starting from the third one (no effect if `lines=2`, default=0pt);

[1] The file described in this section has version number v2.70 and was last revised on 2024-11-14.

- `ante=<`*string*`>` can be used to typeset <*string*> *before* the dropped capital (typical use is for French guillemets starting the paragraph).

- `image[=true]` will force `\lettrine` to replace the letter normally used as dropped capital by an image in eps format (latex) or in pdf, jpg, png, etc. format (pdflatex, xelatex, lualatex); this requires the `graphicx` package to be loaded in the preamble. `\lettrine[image=true]{A}{n exemple}` or just `\lettrine[image]{A}{n exemple}` will load `A.eps`, `A.jpg`, `A.png` or `A.pdf` instead of letter A.

- `viewport=<`*llx lly urx ury*`>` is passed to `\includegraphics` (same four dimen parameters); when present, `\lettrine` only considers the contents of the rectangle defined by its lower left and upper right corners to compute the scaling ratio (which will apply to the whole image). It's up to the user to ensure that the rest of the image will not overwrite the surrounding text, f.i. providing a `\vspace*{...}` in case a significant part sticks out on top of the defined rectangle. This option may be useful in case the letter covers only a limited part of the image, see `https://tex.stackexchange.com/questions/693270/` for an example, or when the image's bounding box is inaccurate.

- `grid[=true]` will force the vertical skip added above the paragraph starting with `\lettrine` to be rounded up to an integer number of `\baselineskip`. This option is meant for grid typesetting.

- `novskip=<`*dimen*`>` overrides `\DiscardVskip` (default=1pt). In some cases (see options `lraise`, `loversize` or accentuated dropped capitals,…) the top of the dropped capital will raise above the top of following text (usually in small caps), this will trigger a corresponding vertical skip above the paragraph starting with `\lettrine`, *only if* this skip exceeds `\DiscardVskip`. Consider enlarging `novskip` (or `\DiscardVskip`) to prevent small vertical skips from being rounded up to `\baselineskip` when using the 'grid' option.

- `realheight[=true]` will compute the default height of the initial so that the top of it is exactly aligned with the top of the text entered as second mandatory argument of `\lettrine` taking possible accents into account. Otherwise, the default height is computed using a customisable string `\LettrineSecondString` instead of the real argument. For backward compatibility, option `realheight` defaults to `false` and `\LettrineSecondString` to 'x'.

  You probably don't need this option if you choose to typeset the second mandatory argument of `\lettrine` in small caps (the default). If you change `\LettrineTextFont` to `\relax` or `\upshape`, consider these two examples:

  `\lettrine{H}{ello}` you might like the top of the 'H' to be aligned with the top of the 'll' rather than with the top of the 'e', adding option `realheight` does the trick: `\lettrine[realheight]{H}{ello}`.

  Global variants : `\LettrineRealHeighttrue` or (without the `realheight` option) `\renewcommand*{\LettrineSecondString}{l}`.

  `\lettrine{L}{a misère}` option `realheight=true` would align with the top of the 'L' with the top of the grave accent, the default is probably better (top of the 'L' aligned with the top of the non accented letters).

- `refstring` with no value, is meant for fancy initials with irregular heights (i.e. taken in fonts like Yinit (OpenType), `cfr-initials`,...). Option `refstring` forces the `\fontsize` computations to be run on the initial given as `\lettrine`'s first mandatory argument instead of the reference string `\LettrineTestString`. In most cases, this option should *not* be used: think of accentuated initials or capitals with optical correction.

  `refstring=`<*string*> can be used to override `\LettrineTestString`, the default reference string (option *seldom useful*).

Example: `\lettrine[lines=4, lraise=0.1, nindent=0em, slope=-.5em]%`
`{V}{oici} un exemple` ...

Coloured initials are available in conjonction with package `color`, examples:
`\lettrine{\textcolor{red}{A}}{n} example` or
`\lettrine{\textcolor[gray]{0.5}{A}}{nother} one`
see package `color` for the syntax of colour commands. Another possibility to colour initials globally is described below, see `\LettrineFontHook`.

Three dimensions, `\LettrineHeight`, `\LettrineDepth` and `\LettrineWidth` hold the final size of the initial (*height* and *depth* being measured from the paragraph's n-th baseline if `lines=`n.

Have a look at files `lettrine-demo-fr.tex` and `lettrine-demo-lua.tex` and at the resulting PDFs in the `doc` folder to see the possible usage of these parameters.

Starting with version 2.30, the default settings can easily be specified as options passed to the lettrine *package*. These options are the same as those of the `\lettrine` command previously described [2]: f.i. `\usepackage[lines=3]{lettrine}` will set the default to three lines of text. Options passed to the lettrine package override the defaults set in the `lettrine.cfg` file (see below) and will be overriden by options passed to the `\lettrine` command.

The default settings can also be customized in a config file `lettrine.cfg` (backward compatibility). The following list shows the syntax to set them and their default values:

- `\setcounter{DefaultLines}{2}`,
- `\setcounter{DefaultDepth}{0}`,
- `\renewcommand*{\DefaultLoversize}{0}`,
- `\renewcommand*{\DefaultLraise}{0}`,
- `\renewcommand*{\DefaultLhang}{0}`,
- `\setlength{\DefaultFindent}{0pt}`,
- `\setlength{\DefaultNindent}{0.5em}`,
- `\setlength{\DefaultSlope}{0pt}`.
- `\setlength{\DiscardVskip}{1pt}`,
- `\LettrineImagefalse`,

[2] With the exception of `ante` and `viewport` which do not make sense for a global usage.

- `\LettrineOnGridfalse`,
- `\LettrineRealHeightfalse`.

Instead of giving optional parameters to the `\lettrine` command, it is possible to set them on a per character basis in a second config file (suggested by Pascal Kockaert): `\renewcommand{\DefaultOptionsFile}{<`*filename*`>}` in the preamble will force this file to be read 'AtBeginDocument'. Alternatively, `\usepackage[optionsfile=<`*filename*`>]{lettrine}` produces the same effect. See examples of such config files in the subdirectory `contrib`.

The idea is to provide settings for specific fonts, or to define suitable parameters for some initials like A or V, for instance
`\LettrineOptionsFor{A}{slope=0.5em, findent=-1.5em, nindent=.7em}`
Options passed this way are meant to fine tune how the text will be wrapped around the initial; for convenience *inside* `\LettrineOptionsFor` *only*, `\LettrineWidth` can be used instead of `em` as a unit length.

Since version 2.70, these commands are also allowed in the document's preamble. Mixing `\LettrineOptionsFor` commands in the preamble and in an external config file is not recommended: remember that if an options file is declared, the settings it holds will be taken into account 'AtBeginDocument' thus possibly overriding `\LettrineOptionsFor` commands in the preamble.

Anyway, the settings read from this file will be overridden by the optional arguments eventually given to the `\lettrine` command.

More customisation possibilities are offered by the next four commands:

- `\renewcommand*{\LettrineFontHook}{}`,
- `\renewcommand*{\LettrineTextFont}{\scshape}`,
- `\renewcommand*{\LettrineTestString}{EFTZ}`,
- `\renewcommand*{\LettrineSecondString}{x}`,

`\LettrineTextFont` sets the font used for the second argument of `\lettrine`, its default definition is `\newcommand{\LettrineTextFont}{\scshape}` (second argument in small caps), this can be changed using `\renewcommand`.

`\LettrineTestString` and `\LettrineSecondString` provide reasonable defaults for Latin scripts (`EFTZ` and `x`). For other scripts they *should be changed* respectively to uppercase and lowercase letters of the given script, as the dropped cap's height computation is based on these strings; another possibility is to use the previously described `refstring` and `realheight` options.

`\LettrineFont` *is not customisable* see `\LettrineFontHook` below, it sets the font used for the dropped capital, usually the current font in a (large) size, computed automatically from the number of lines it will fill: the font size is computed so that, a *standard* dropped capital (say Z, not À) when sitting on its baseline, gets its top aligned with the top of the following text (provided `loversize` = 0 and `lines` ≥ 2). When `lines=1`, size is computed as if `lines` was 2.

A hook `\LettrineFontHook` is provided to change the font used for the dropped capital, syntax follows LaTeX's low-level font interface (see LaTeX Companion III, section I-9.3),

the \selectfont command is issued by \LettrineFont:
```
\renewcommand{\LettrineFontHook}{\fontfamily{ppl}\fontseries{bx}%
                                 \fontshape{sl}}
```
selects Palatino bold expanded slanted for the dropped capital.

With LuaLaTeX or XeLaTeX changing the lettrine's font is even easier, simply use the \fontspec command:
```
\renewcommand{\LettrineFontHook}{\fontspec{LinLibertine_I.otf}}
```
will switch to Linux Libertine Initials.

\LettrineFontHook can also be used to change the colour of all initials in a (part of) document: `\renewcommand{\LettrineFontHook}{\color[gray]{0.5}}` will colour the initials following this command in grey. A \color command can be added in \LettrineTextHook if the text following the dropped cap requires the same or another colour.

Important notice: the sizing works fine with *fully scalable* fonts (like the standard PostScript or OpenType fonts), but might not work well with CM/EC fonts which have two limitations: only a limited number of sizes is available by default (precise adjustments are impossible), and the largest size (25pt or 35pt) is often too small. The CM fonts are available in PostScript type1 format for free (courtesy of BlueSky/Y&Y), to make them fully scalable, it is mandatory to add \usepackage{type1cm} in the preamble of your document. The EC fonts are also available in type1 format for free (thanks to Vladimir Volovich, they are called cm-super), and adding \usepackage{type1ec}[3] in the preamble will make them fully scalable too. So, if you want lettrine.sty to work properly with CM or EC fonts, you will need *PostScript versions* of these fonts *and* one of the packages type1cm.sty or type1ec.sty.

The LM fonts are a good replacement for both CM and EC fonts they are fully scalable, so you should use them instead of CM or EC fonts. \usepackage{lmodern} is the command to switch them on (add \usepackage[T1]{fontenc} when composing in one of the western languages other than English in order to get proper hyphenation).

You can also consider using one of the standard PostScript fonts (Palatino, Times, Utopia…), or any OpenType font, they are fully scalable too!

Breaking change in version 2.50: in order to improve the alignment of side by side parboxes starting with a dropped cap, the internals of the \lettrine{} command have changed. Formerly, the initial was completely smashed (*height*=0, *depth*=0) and a \vskip was added in case the initial sticked out too much above the paragraph's first baseline. From version 2.50 on, the initial gets a null *depth* and its proper *height* (measured from the paragraph's first baseline, please note that it is different from \LettrineHeight). This change has the following side effect: in multicolumn typesetting, when a dropped cap starts a column and sticks out significantly above the baseline, it might be necessary to 'smash' the dropped cap and eventually to add a \vspace{} before the multicolumn environment. See lettrine-demo-fr.pdf p. 4 for an example. Using the rollback mechanism to switch back to version 2.40 is another option: \usepackage{lettrine}[=v2.4].

New in version 2.60: the \lettrine command is now compatible with right to left typesetting, with LuaLaTeX and XeLaTeX (+babel or polyglossia). With XeLaTeX, file

[3] This package, available on CTAN, was first released on 2002/07/30.

lettrine-xetex-bidi.def (v0.8 [2022/11/06]) is automatically loaded by the bidi package; it redefines the \@lettrine command based on version 2.30 of lettrine. This code is *incompatible* with versions 2.50 and newer, you can safely remove lettrine-xetex-bidi.def when installing v2.60 of lettrine.

Known issues:

- Nothing is done to prevent page-breaking in a paragraph starting with a dropped capital; when it happens to hang into the footer, page-breaking has to be done manually.
- \lettrine works within 'quote' 'quotation', 'abstract' environments but does not work within 'center' environments except with option [lines=1].
- \lettrine does not work within lists.
- If \lettrine is used inside any environment, it is *mandatory* to end the paragraph starting with the dropped capital *inside* the environment; adding a \par command before the end of environment usually fixes placement issues.
- The LaTeX \raggedright command doesn't work well with the TeX \parshape command used internally by \lettrine. Please use the \RaggedRight command from the ragged2e package together with appropriate settings, f.i. \setlength{\RaggedRightRightskip}{0pt plus .1\hsize}.
  See https://tex.stackexchange.com/questions/97808/ for details.
- If a *list* has to be included in a paragraph starting with \lettrine, it is necessary to add the command \parshape=0 just after the end of the list (starting a new paragraph just before or just after the list works too). Remember that 'quote', 'quotation', 'abstract' environments are implemented as *lists* in LaTeX.
- If you are facing some slight height inaccuracy for a dropped capital, you can try option refstring; this option is meant for fancy (unaccented) initials. Informations about targeted and effective initial's height are available in the .log file. Using LuaTeX or XeTeX engines with OpenType fonts may be an option (some TFM files for Type1 fonts are slightly inaccurate).
- \LettrineTestString's value has changed over the time; these changes may result in slight size differences for the initial. Starting with version 2.2, the lettrine package takes advantage of the rollback facilities recently introduced by the LaTeX Team[4]. Three rollback versions are provided in order to produce exactly the same output as with former versions: you can request \usepackage{lettrine}[=v1.6][5] for 1999-2012 documents or \usepackage{lettrine}[=v1.9] (documents from 2012 to July 2018) or \usepackage{lettrine}[=v2.0] (August 2018). Using any date in ISO format works too: \usepackage{lettrine}[=2014-03-15] will load v1.9.

[4] A LaTeX kernel dated 2018-04-01 or newer is required.
[5] Don't forget the = sign!

2 TEXnical details

The lettrine package uses the rollback mechanism introduced by the LaTeX Team to provide easier backward compatibility. The current release requires a LaTeX version not older than 2022-06-01 (kernel packages ltkeys and xfp are required).

```
\DeclareRelease{v1.6}{1999-03-03}{lettrine-2006-03-17.sty}
\DeclareRelease{v1.9}{2012-07-20}{lettrine-2015-08-31.sty}
\DeclareRelease{v2.0}{2018-07-21}{lettrine-2018-08-18.sty}
\DeclareRelease{v2.3}{2022-09-25}{lettrine-2022-09-25.sty}
\DeclareRelease{v2.4}{2023-04-18}{lettrine-2023-04-18.sty}
\DeclareCurrentRelease{}{2023-01-20}
\RequirePackage{xfp}
```

Default initializations: define the necessary counters, lengths, and commands to hold the default settings and set these default settings. They can be overwritten in file `lettrine.cfg`.

```
\newcounter{DefaultLines}
\setcounter{DefaultLines}{2}
\newcounter{DefaultDepth}
\newcommand*{\DefaultOptionsFile}{\relax}
\newcommand*{\DefaultLoversize}{0}
\newcommand*{\DefaultLraise}{0}
\newcommand*{\DefaultLhang}{0}
\newdimen\DefaultFindent
\setlength{\DefaultFindent}{\z@}
\newdimen\DefaultNindent
\setlength{\DefaultNindent}{0.5em}
\newdimen\DefaultSlope
\setlength{\DefaultSlope}{\z@}
\newdimen\DiscardVskip
\setlength{\DiscardVskip}{1\p@}
\newif\ifLettrineImage
\newif\ifLettrineOnGrid
\newif\ifLettrineRealHeight
```

Then let's define the necessary internal counters, lengths, and commands.

```
\newsavebox{\L@lettrine}
\newsavebox{\L@lbox}
\newsavebox{\L@tbox}
\newcounter{L@lines}
\newcounter{L@depth}
\newdimen\L@Pindent
\newdimen\L@Findent
\newdimen\L@Nindent
\newdimen\L@lraise
\newdimen\L@first
\newdimen\L@next
\newdimen\L@slope
\newdimen\L@height
\newdimen\L@novskip
\newdimen\L@ruleht
\newdimen\L@target@ht
```

```
\newdimen\L@target@dp
\newdimen\L@target@tht
\newdimen\LettrineWidth
\newdimen\LettrineHeight
\newdimen\LettrineDepth
\newcommand*{\L@file}{}
\newcommand*{\L@hang}{}
\newcommand*{\L@oversize}{}
\newcommand*{\L@raise}{}
\newcommand*{\L@ante}{}
\newif\ifL@image
\newif\ifL@grid
\newif\ifL@realh
\let\ifL@RTL\iffalse
\newcommand*{\L@viewport}{}
```

`\LettrineTestString` As some font designers apply optical correction to capitals C, G, O, or Q (they are slightly taller than 'T' or 'Z'), they are better left out of `\LettrineTestString`. `EFTZ` should be a good default for most fonts.

```
\newcommand*{\LettrineTestString}{EFTZ}
```

Load a local config file if present in LaTeX's search path.

```
\InputIfFileExists{lettrine.cfg}
  {\typeout{Loading lettrine.cfg}}
  {\typeout{lettrine.cfg not found, using default values}}
```

Global package options enable to override the default values given above to generic parameters. These may be overriden again by options passed to the `\lettrine[]{}{}` command. This code is based on ltkeys.dtx.

```
\DeclareKeys[LettrineGlobal]
 {
   lines.code        = \setcounter{DefaultLines}{#1}   ,
   depth.code        = \setcounter{DefaultDepth}{#1}   ,
   lhang.store       = \DefaultLhang                   ,
   loversize.store   = \DefaultLoversize               ,
   lraise.store      = \DefaultLraise                  ,
   findent.code      = \setlength{\DefaultFindent}{#1} ,
   nindent.code      = \setlength{\DefaultNindent}{#1} ,
   slope.code        = \setlength{\DefaultSlope}{#1}   ,
   novskip.code      = \setlength{\DiscardVskip}{#1}   ,
   image.if          = LettrineImage                   ,
   grid.if           = LettrineOnGrid                  ,
   realheight.if     = LettrineRealHeight              ,
   refstring.default:n = \L@initial                    ,
   refstring.store   = \LettrineTestString             ,
   optionsfile.store = \DefaultOptionsFile
 }
\ProcessKeyOptions[LettrineGlobal]
```

Ditto for the `\lettrine` command's options.

```
\DeclareKeys[LettrineLocal]
```

```
{
  lines.code        = \setcounter{L@lines}{#1}        ,
  depth.code        = \setcounter{L@depth}{#1}        ,
  lhang.store       = \L@hang                         ,
  loversize.store   = \L@oversize                     ,
  lraise.store      = \L@raise                        ,
  ante.store        = \L@ante                         ,
  findent.code      = \setlength{\L@Findent}{#1}      ,
  nindent.code      = \setlength{\L@Nindent}{#1}      ,
  slope.code        = \setlength{\L@slope}{#1}        ,
  novskip.code      = \setlength{\L@novskip}{#1}      ,
  image.if          = L@image                         ,
  viewport.store    = \L@viewport                     ,
  grid.if           = L@grid                          ,
  realheight.if     = L@realh                         ,
  refstring.default:n = \L@initial                    ,
  refstring.store   = \L@refstring
}
```

Read the per letter options file 'AtBeginDocument'.

```
\AtBeginDocument{%
  \if\DefaultOptionsFile\relax
  \else
    \InputIfFileExists{\DefaultOptionsFile}%
      {}%
      {\PackageWarning{lettrine.sty}%
         {File \DefaultOptionsFile\space not found}%
      }%
  \fi}
```

`\LettrineOptionsFor` This command sets the values of parameters on a per character basis, for instance: `\LettrineOptionsFor{A}{slope=0.6em, findent=-1em, nindent=0.6em}`

```
\newtoks\Lettrine@tweaks
\newcommand*{\LettrineOptionsFor}[2]{%
  \Lettrine@tweaks=\expandafter{\the\Lettrine@tweaks
                                \@lettrine@optionsfor{#1}{#2,}}}
```

The internal `\@lettrine@optionsfor` checks if its first argument matches the current initial, if so it passes the options (second argument) to `\SetKeys`.

```
\newcommand*{\@lettrine@optionsfor}[2]{%
  \edef\L@tmpa{#1}%
```

Gobble potential color commands for the initial.

```
  \begingroup
    \def\color##1##{\L@color{##1}}%
    \let\L@color\@gobbletwo
    \def\textcolor##1##{\L@textcolor{##1}}%
    \def\L@textcolor##1##2##3{##3}%
    \xdef\L@tmpb{\L@initial}%
  \endgroup
  \ifx\L@tmpa\L@tmpb \SetKeys[LettrineLocal]{#2}\fi
  }
```

\LettrineTextFont In French, small caps usually follow the initial.

```
\newcommand*{\LettrineTextFont}{\scshape}
\newcommand*{\LettrineSecondString}{x}
```

\LettrineFontHook \LettrineFontHook enables to select another font for the dropped capital. Its default definition is empty (the current text font is used).

```
\newcommand*{\LettrineFontHook}{}
```

\computeL@height The default size for the dropped capital is computed so that the top of it is exactly aligned with the top of the following text; an extra height (positive or negative) may be added globally by redefining \Defaultloversize or locally using optional argument loversize=. If lines=1, the default size for the dropped capital is computed as if lines=2.

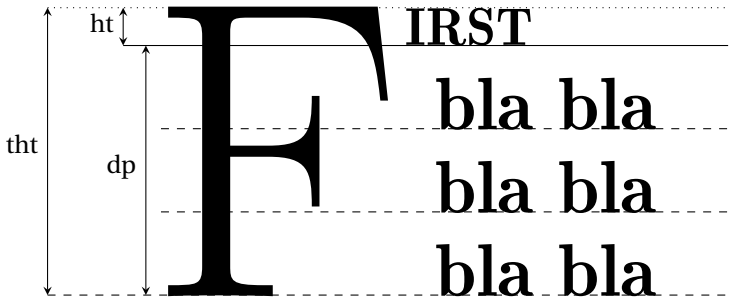

Figure 1: Initial's targeted dimensions (\lines = 4)

\computeL@height first computes the targeted height for the dropped capital and stores it into \L@target@tht. This height only depends on L@lines and on the height of \L@tbox (see fig. 1). So options *must* be read and \L@tbox must be properly initialised *before* executing \computeL@height (see below in \@lettrine code).

\L@height is set to \L@target@tht raised by the \L@oversize factor.

```
\def\computeL@height{%
  \setlength{\L@target@ht}{\ht\L@tbox}%
```

As \baselineskip might be a rubber length, let's convert it into a 'dimen' using \@tempdima.

```
  \@tempdima=\baselineskip
  \setlength{\L@target@dp}{\value{L@lines}\@tempdima}%
  \ifnum\value{L@lines}>1
    \addtolength{\L@target@dp}{-\@tempdima}%
  \else
    \addtolength{\L@target@ht}{\L@target@dp}%
    \setlength{\L@target@dp}{0pt}%
  \fi
  \setlength{\L@target@tht}{\L@target@ht}%
  \addtolength{\L@target@tht}{\L@target@dp}%
  \setlength{\L@height}{\L@target@tht}%
  \addtolength{\L@height}{\L@oversize\L@target@tht}%
}
```

\compute@fontsize After executing \computeL@height, \L@height holds the exact height required for the dropped capital, nothing more is needed if the initial is a picture, otherwise we need to compute the matching \fontsize's value. This is done by measuring the height of a "reference" capital (i.e. either listed in \LettrineTestString or the initial itself). This command compares the height of a "reference" capital scaled by \fontsize with argument \L@height, to \L@height (the required height for the initial); both are converted into integers (in sp), to compute a ratio \L@factor (decimal number). Then, the initial will be scaled by \L@factor\L@height. Starting with v2.40, \L@factor is computed by \fpeval from xfp.sty.

If the option refstring is set in the \lettrine command the initial itself is taken as reference to compute \fontsize, this can be handy when working with fancy fonts (i.e. cfr-initials, Yinit). In most cases, the default is a better choice .

```
\def\compute@fontsize{%
  \ifx\L@refstring\@empty
    \def\Lettrine@RefString{\LettrineTestString}%
  \else
    \def\Lettrine@RefString{\L@refstring}%
  \fi
  \sbox{\@tempboxa}{\LettrineFontHook
                    \fontsize{\L@height}{\L@height}\selectfont
                    \Lettrine@RefString}%
  \@tempcnta=\ht\@tempboxa
  \ifnum\@tempcnta=0
    \PackageWarning{lettrine}{Unable to compute \protect\fontsize!%
    \MessageBreak \protect\LettrineTestString\space empty? reported}
    \def\L@factor{1}%
  \else
    \@tempcntb=\L@height
    \def\L@factor{\fpeval{\the\@tempcntb/\the\@tempcnta}}%
  \fi
}
```

\LettrineFont \fontsize's argument providing the requested \L@height is \L@factor\L@height.

```
\newcommand*{\LettrineFont}{%
   \LettrineFontHook
   \fontsize{\L@factor\L@height}{\L@factor\L@height}%
   \selectfont
}
```

\setupL@lbox The next (internal) command computes the requested size for the initial (letter or image) and prepares a box \L@lbox holding it.

```
\def\setupL@lbox{%
  \computeL@height
  \ifL@image
    \ifx\L@viewport\@empty
      \sbox{\L@lbox}{\includegraphics[height=\L@height]{\L@initial}}%
    \else
      \sbox{\L@lbox}{%
        \expanded{\noexpand\includegraphics%
                  [viewport=\L@viewport, height=\L@height]{\L@initial}%
                 }}%
```

```
    \fi
  \else
    \compute@fontsize
    \sbox{\L@lbox}{\LettrineFont \L@initial}%
  \fi
}
```

\lettrine Now let's define the \lettrine command.

```
\def\lettrine{\@ifnextchar[\@lettrine{\@lettrine[]}}
\def\@lettrine[#1]#2#3{%
  \def\L@initial{#2}\def\L@refstring{}\def\L@viewport{}%
```

First reset the parameters to their default values:

```
  \setcounter{L@lines}{\value{DefaultLines}}%
  \setcounter{L@depth}{\value{DefaultDepth}}%
  \renewcommand*{\L@hang}{\DefaultLhang}%
  \renewcommand*{\L@oversize}{\DefaultLoversize}%
  \renewcommand*{\L@raise}{\DefaultLraise}%
  \renewcommand*{\L@ante}{}%
  \setlength{\L@Findent}{\DefaultFindent}%
  \setlength{\L@Nindent}{\DefaultNindent}%
  \setlength{\L@slope}{\DefaultSlope}%
  \setlength{\L@novskip}{\DiscardVskip}%
  \ifLettrineImage\L@imagetrue\else\L@imagefalse\fi
  \ifLettrineOnGrid\L@gridtrue\else\L@gridfalse\fi
  \ifLettrineRealHeight\L@realhtrue\else\L@realhfalse\fi
```

Then take the local options passed to \lettrine into account. The content of \L@tbox depends on option realheight, so we have to initialise the \L@tbox content now [6].

```
  \SetKeys[LettrineLocal]{#1}%
  \sbox{\L@tbox}{\LettrineTextFont{\LettrineSecondString}}%
  \ifL@realh
    \def\@tempa{#3}%
    \ifx\@tempa\@empty
      \PackageWarning{lettrine.sty}%
        {Empty second argument,\MessageBreak
         ignoring option `realheight';}%
    \else
      \sbox{\L@tbox}{{\LettrineTextFont{#3}}}%
    \fi
  \fi
```

Take the per character options into account if any. For these options some parameters' values findent, nindent and slope —which do not influence the initial's size— may be given relative to \LettrineWidth, the \L@lbox has to be set up first to evaluate \LettrineWidth.

```
  \setupL@lbox
  \setlength{\LettrineWidth}{\wd\L@lbox}%
  \the\Lettrine@tweaks
```

[6] Now means before eventually reading the per character config file.

As local options always prevail, read again the optionnal argument of `\lettrine`.

```
\SetKeys[LettrineLocal]{#1}%
```

Store the initial's final dimensions,

```
\setupL@lbox
\setlength{\LettrineWidth}{\wd\L@lbox}%
\setlength{\LettrineHeight}{\ht\L@lbox}%
\setlength{\LettrineDepth}{\dp\L@lbox}%
```

and reset `\L@tbox`'s content (mandatory in case `realheight=false`):

```
\sbox{\L@tbox}{{\LettrineTextFont{#3}}}%
```

Start a new paragraph and compute in `\L@ruleht` the height of the top part of the dropped capital which raises above the paragraph's first baseline.

The basis for `\L@raise` (and `\L@oversize`, see `\LettrineFont`) is `\L@target@tht`.

```
\par
\setlength{\L@ruleht}{\LettrineHeight}%
\setlength{\L@lraise}{\L@raise\L@target@tht}%
\addtolength{\L@ruleht}{\L@lraise}%
\ifnum\value{L@lines}>\@ne
  \@tempcnta=\value{L@lines}%
  \advance\@tempcnta \m@ne
  \addtolength{\L@ruleht}{-\@tempcnta\baselineskip}%
  \addtolength{\L@lraise}{-\L@target@dp}%
\fi
```

When `\L@ruleht` is larger than `\baselineskip - \L@novskip` and the `grid` option is `true`, let's skip an integer number of `\baselineskip` (and smash the dropped cap, see below).

```
\@tempdima=\L@ruleht
\advance\@tempdima \L@novskip
\@tempdimb=\baselineskip
\ifdim\@tempdima>\@tempdimb
  \ifL@grid
    \@tempcnta=\z@
    \loop\ifdim\@tempdima>\@tempdimb
       \advance\@tempcnta \@ne
       \advance\@tempdima -\@tempdimb
    \repeat
    \vskip\@tempcnta\baselineskip
  \fi
\fi
```

Print some informations about accuracy to the log file,

```
\begingroup
\def\IeC##1{##1}%
\@tempdima=\L@oversize pt\relax
\PackageInfo{lettrine.sty}%
  {Targeted height = \the\L@target@tht\MessageBreak
   (for loversize=0, accent excluded),\MessageBreak
   Lettrine height = \the\LettrineHeight\space (#2)%
```

```
    \ifdim\@tempdima>\z@\space loversize=\L@oversize\fi;%
    \MessageBreak reported}%
\endgroup
```

We (mis)use the length `\L@first` to compute the width of the text eventually coming before the dropped capital. It is reset later on to hold the first line's length.

```
\setlength{\L@Pindent}{\wd\L@lbox}%
\addtolength{\L@Pindent}{-\L@hang\wd\L@lbox}%
\settowidth{\L@first}{\L@ante}%
\addtolength{\L@Pindent}{\L@first}%
\addtolength{\L@Pindent}{\L@Findent}%
\setlength{\L@first}{\linewidth}%
\addtolength{\L@first}{-\L@Pindent}%
```

Now let's compute `\L@Nindent` and `\L@next` for the next lines.

```
\addtolength{\L@Nindent}{\L@Pindent}%
\setlength{\L@next}{\linewidth}%
\addtolength{\L@next}{-\L@Nindent}%
```

This is for quotation, quote, abstract… environments: `\linewidth` is set by these environments, all we have to do is to shift our text left by `\@totalleftmargin`.

```
\addtolength{\L@Pindent}{\@totalleftmargin}%
\addtolength{\L@Nindent}{\@totalleftmargin}%
```

Now, set up the shape of the new paragraph (designed by `\parshape`). It obiously depends on the text direction, the code previously available in `lettrine-xetex-bidi.def` for right to left scripts is integrated here now.

```
\addtocounter{L@lines}{1}%
\addtocounter{L@lines}{\value{L@depth}}%
\ifL@RTL
  \def\L@parshape{\c@L@lines \z@ \the\L@first}%
\else
  \def\L@parshape{\c@L@lines \the\L@Pindent \the\L@first}%
\fi
\@tempcnta=\tw@
\@whilenum \@tempcnta<\c@L@lines\do{%
   \ifL@RTL
     \edef\L@parshape{\L@parshape \z@ \the\L@next}%
   \else
     \edef\L@parshape{\L@parshape \the\L@Nindent \the\L@next}%
   \fi
   \addtolength{\L@Nindent}{\L@slope}%
   \addtolength{\L@next}{-\L@slope}%
   \advance\@tempcnta\@ne}%
\ifL@RTL
  \edef\L@parshape{\L@parshape \z@ \the\linewidth}%
\else
  \edef\L@parshape{\L@parshape \@totalleftmargin \the\linewidth}%
\fi
\noindent
\parshape=\L@parshape\relax
```

Write the dropped capital into the left margin, and wrap the rest of paragraph around it.

```
  \llap{\smash{\mbox{\L@ante}\raisebox{\L@lraise}{\usebox{\L@lbox}}}%
        \ifL@grid\else\rule{0pt}{\L@ruleht}\fi
        \hskip \the\L@Findent}%
  \unhcopy\L@tbox\relax
```

A `\parshape` reset is required in abstract, quote and quotation environments beginning with `\lettrine` and spreading over several paragraphs. When the list ends, `\parshape` returns to `0`.

```
  \ifnum\@listdepth>0 \Lreset@listparshape \fi
}
```

This ends the definition of `\lettrine`; `\Lreset@listparshape` adds the parshape reset to the first occurence of `\everypar` following the `\lettrine` command.

```
\newtoks\Llist@everypar
\def\Lreset@listparshape{%
  \let\Lnew@everypar\everypar
  \Llist@everypar=\expandafter{\the\everypar}%
  \Lnew@everypar={\the\Llist@everypar
                  \parshape=\@ne \@totalleftmargin \linewidth \relax
                  \let\everypar\Llist@everypar
                 }%
}
```

Compatibility with the bidi package (loaded *after* lettrine).

```
\AtBeginDocument{%
  \IfPackageLoadedTF{bidi}{\eqnewif{\ifL@RTL}{\if@RTL}}{}%
}
```

3 Configuration file

```
%% lettrine.cfg: configuration file for lettrine.sty
%%
%% If you want to customize lettrine, please *do not* hack into the
%% code, copy this file to your working directory and customize the
%% copy as you like.
%%
%% Uncomment any of these lines and change the parameters' values
%% to fit your needs (see lettrine.dtx).
%%
%%\setcounter{DefaultLines}{2}
%%\setcounter{DefaultDepth}{0}
%%
%% These are *decimal* numbers:
%%\renewcommand*{\DefaultLoversize}{0}
%%\renewcommand*{\DefaultLraise}{0}
%%\renewcommand*{\DefaultLhang}{0}
%%
%% These are *lengths* (don't forget the unit):
%%\setlength{\DefaultFindent}{0pt}
%%\setlength{\DefaultNindent}{0.5em}
%%\setlength{\DefaultSlope}{0mm}
%%\setlength{\DiscardVskip}{1pt}
%%
%% Theses are *flags* (value=true/false):
%%\LettrineImagefalse
%%\LettrineOnGridfalse
%%\LettrineRealHeightfalse
%%
%% This is a *command*, define it as \relax if you dont want the second
%% mandatory argument of \lettrine[]{}{} to be typset in small caps.
%%\renewcommand*{\LettrineTextFont}{\scshape}|
%%
%% Theses are *commands* (value=string, only height matters):
%%\renewcommand*{\LettrineTestString}{EFTZ}
%%\renewcommand*{\LettrineSecondString}{x}
%%
%% In case you want to set parameters for some letters
%% in file `optfile.cfl'
%%\renewcommand{\DefaultOptionsFile}{optfile.cfl}
```

4 Change History

Changes are listed in reverse order (latest first) from version 1.0