Typesetting dropped capitals with LaTeX

Daniel Flipo
`daniel.flipo@free.fr`

1 Introduction

The file `lettrine.dtx`[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 David Carlisle’s `keyval.sty` 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 v2.30 and was last revised on 2022-09-25.

- `ante=`*<text>* can be used to typeset *<text>* *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, 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.
  N.B.: Redefining `\LettrineFont` as `\LettrineFontEPS` is no longer supported and `\LettrineFontEPS` has been renamed as `\LettrineImage`.

- `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=0.2pt). 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`[2] is meant for fancy initials with irregular heights (i.e. taken in fonts like Yinit (OpenType), `cfr-initials`,…). `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.

---

[2] Unlike `grid` or `realheight`, `refstring` is not a flag (*do not* add `=true`!); it is possible to set `refstring=`*<string>* to override `\LettrineTestString` locally.

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

Coloured initials are possible 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, `\LettrineWidth`, `\LettrineHeight` and `\LettrineDepth`, store the final size of the initial.

Have a look at files `demo-fr.tex` and `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 be specified as options passed to the lettrine *package*. These options are the same as those of the `\lettrine` command [3]: 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 be customized in a config file `lettrine.cfg` for a global usage. 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}{0.2pt}`,
- `\LettrineImagefalse`,
- `\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 (or anywhere in the document) will force each call to `\lettrine` to read the file <*filename*>. Alternatively, `\usepackage[optionsfile=<`*filename*`>]{lettrine}` produces the same effect. See examples of such config files in the subdirectory `contrib`.

---

[3] With one exception: `refstring` is available only for the `\lettrine` command.

The dimensional parameters `findent`, `nindent` and `slope`, can be set relative to `\LettrineWidth` if needed, *in <filename> only*. The settings read from this file will be overridden by the optional arguments eventually given to `\lettrine`.

`\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`.

`\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, p.187–192), 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.

Important notice: the sizing works fine with *fully scalable* fonts (like the standard PostScript 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 now 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}`[4] 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!

---

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

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 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[5]. Three rollback versions are provided in order to produce exactly the same output as with former versions: you can request `\usepackage{lettrine}[=v1.6]`[6] 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.

2 TeXnical details

The lettrine package now uses the rollback mechanism recently introduced by the LaTeX Team to provide easier backward compatibility. This requires a recent LaTeX kernel (at least 2018-04-01), roolback is ignored with older kernels.

```
\ifdefined\DeclareRelease
  \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}
  \DeclareCurrentRelease{}{2020-03-14}
\else
  \PackageWarning{lettrine}{Your LaTeX kernel is too old to provide
    access\MessageBreak to former versions of the lettrine package.%
    \MessageBreak If you need rollback, please upgrade%
    \MessageBreak your LaTeX kernel (2018-04-01 or newer),%
    \MessageBreak otherwise you are fine; reported}
\fi
```

---

[5] A LaTeX kernel dated 2018-04-01 or newer is required.

[6] Don't forget the = sign!

This package only runs with LaTeX and requires files `xkeyval.sty` and `minifp.sty`.

```
\RequirePackage{xkeyval,minifp}
```

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}{0.2\p@}
\newif\ifLettrineImage
\newif\ifLettrineOnGrid
\newif\ifLettrineRealHeight
```

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.

```
\DeclareOptionX{lines}[\value{DefaultLines}]{\setcounter{DefaultLines}{#1}}
\DeclareOptionX{depth}[\value{DefaultDepth}]{\setcounter{DefaultDepth}{#1}}
\DeclareOptionX{lhang}[]{\ifx#1\@empty\else
                                \renewcommand*{\DefaultLhang}{#1}\fi}
\DeclareOptionX{loversize}[]{\ifx#1\@empty\else
                                \renewcommand*{\DefaultLoversize}{#1}\fi}
\DeclareOptionX{lraise}[]{\ifx#1\@empty\else
                                \renewcommand*{\DefaultLraise}{#1}}
\DeclareOptionX{findent}[\the\DefaultFindent]{%
                                \setlength{\DefaultFindent}{#1}}
\DeclareOptionX{nindent}[\the\DefaultNindent]{%
                                \setlength{\DefaultNindent}{#1}}
\DeclareOptionX{slope}[\the\DefaultSlope]{\setlength{\DefaultSlope}{#1}}
\DeclareOptionX{image}[true]{\csname LettrineImage#1\endcsname}
\DeclareOptionX{grid}[true]{\csname LettrineOnGrid#1\endcsname}
\DeclareOptionX{novskip}[\the\DiscardVskip]{\setlength{\DiscardVskip}{#1}}
\DeclareOptionX{realheight}[true]{\csname LettrineRealHeight#1\endcsname}
\DeclareOptionX{optionsfile}[]{\ifx#1\@empty\else
                               \renewcommand*{\DefaultOptionsFile}{#1}\fi}
```

```
\ProcessOptionsX
```

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

```
\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@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
\newif\ifL@selfref
```

Provide commands for the fonts used to typeset the two mandatory arguments of `\lettrine`.

`\LettrineTextFont` In French, small caps usually follow the dropped capital.

```
\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`.

`\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).

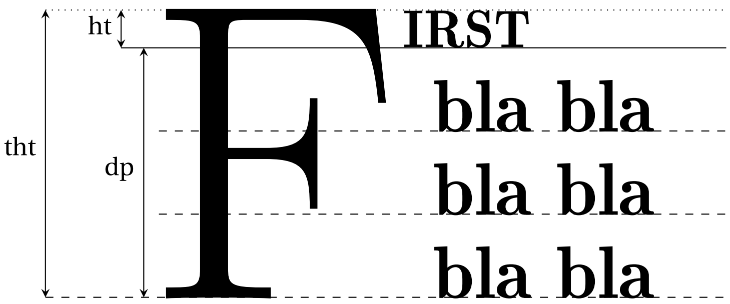

Figure 1: Initial's targeted dimensions (`\lines` = 4)

`\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}%
}
```

`\LettrineTestString` 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). 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}
```

`\compute@fontsize` 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`. Up to v2.01, possible values for `\L@factor` were either 1 or any value ≥ 1.1 which was fine for almost every font but not all (i.e. `cfr-initials`). Starting with v2.1, `\L@factor` is computed more accurately by `\MFPdiv` from `minifp.sty`.

The conditional `\ifL@selfref` is set to `true` by `\lettrine` option `refstring`, then 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{%
  \ifL@selfref
    \def\Lettrine@RefString{\l@refstring}%
  \else
    \def\Lettrine@RefString{\LettrineTestString}%
  \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
    \MFPdiv{\the\@tempcntb}{\the\@tempcnta}\L@factor
  \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
}
```

`\LettrineImage` The following definition is for use with dropped capitals defined as images: EPS, PDF, JPG, PNG files (see examples in demo-fr.tex). Its requires the `graphicx` package to be loaded in the preamble with `\usepackage{graphicx}`. The image’s height is computed just as in the standard case and passed to `\includegraphics`.

```
\newcommand*{\LettrineImage}{%
   \LettrineFontHook\includegraphics[height=\L@height]%
}
```

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

Set up keyval initializations for the `\lettrine` command.

```
\define@key{L}{lines}{\setcounter{L@lines}{#1}}
\define@key{L}{depth}{\setcounter{L@depth}{#1}}
\define@key{L}{lhang}{\renewcommand*{\L@hang}{#1}}
\define@key{L}{loversize}{\renewcommand*{\L@oversize}{#1}}
\define@key{L}{lraise}{\renewcommand*{\L@raise}{#1}}
\define@key{L}{ante}{\renewcommand*{\L@ante}{#1}}
\define@key{L}{findent}{\setlength{\L@Findent}{#1}}
\define@key{L}{nindent}{\setlength{\L@Nindent}{#1}}
\define@key{L}{slope}{\setlength{\L@slope}{#1}}
\define@key{L}{image}[true]{\csname L@image#1\endcsname}
\define@key{L}{grid}[true]{\csname L@grid#1\endcsname}
\define@key{L}{realheight}[true]{\csname L@realh#1\endcsname}
\define@key{L}{novskip}{\setlength{\L@novskip}{#1}}
\define@key{L}{refstring}[\l@initial]{\L@selfreftrue
                                      \def\l@refstring{#1}}
```

`\LettrineOptionsFor` This command is to be used in an optional config file (the name of which is found in `\DefaultOptionsFile`) to set the values of parameters on a per character basis, for instance:
`\LettrineOptionsFor{A}{slope=0.6em, findent=-1em, nindent=0.6em}`
creates an internal command (`\l@A-keys` in this example), which expands to the options given as second argument of `\LettrineOptionsFor` for letter 'A'.

```
\newcommand*{\LettrineOptionsFor}[2]{\@namedef{l@#1-keys}{#2}}
```

`\lettrine` Now let's define `\lettrine`.

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

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
```

The final initial size depends on the height of `\L@tbox`; the content of `\L@tbox` depends on option `realheight`, so we have to read `\lettrine`'s optional argument and initialise the `\L@tbox` content now[7].

---

[7] Now means before eventually reading the config file.

```
\setkeys{L}{#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
```

Then try to read an optional file (its name is given by `\DefaultOptionsFile`), do this inside a group, and define a global command `\l@LOKeys` which will expand to the list of options given by `\LettrineOptionsFor` for the current initial (defined by `#2`)…

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

Gobble the colour commands, just keep the letter argument.

```
  \def\color##1##{\l@color{##1}}%
  \let\l@color\@gobbletwo
  \def\textcolor##1##{\l@textcolor{##1}}%
  \def\l@textcolor##1##2##3{##3}%
```

Save the list of options relevant to the letter in `#2` in `\l@LOKeys` (list is empty eventually).

```
  \expandafter\ifx\csname l@#2-keys\endcsname\relax
                \gdef\l@LOKeys{}%
              \else
                \xdef\l@LOKeys{\csname l@#2-keys\endcsname}%
              \fi
  \endgroup
```

Now apply these options (the following code executes `\setkeys{L}{\l@LOKeys}}` where `\l@LOKeys` is expanded, see `keyval.sty`).

```
  \def\KV@prefix{KV@L@}%
  \let\@tempc\relax
  \expandafter\KV@do\l@LOKeys,\relax,
```

As 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 to evaluate `\LettrineWidth`.

```
  \setupL@lbox
  \setlength{\LettrineWidth}{\wd\L@lbox}%
  \def\KV@prefix{KV@L@}%
  \let\@tempc\relax
  \expandafter\KV@do\l@LOKeys,\relax,
```

As local options prevail on those held in the config file, we have to read local options again:

```
  \setkeys{L}{#1}%
\fi
```

Options and optional config file have be taken into account, we can now finally save the first mandatory argument of `\lettrine` properly scaled into `\L@lbox`.

```
\setupL@lbox
```

Store the initial's final dimensions,

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

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@\else\space loversize=\L@oversize\fi;%
   \MessageBreak reported}%
\endgroup
```

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

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

Start a new paragraph, skipping the necessary amount of space if the dropped capital sticks out of the top of paragraph. We use `\L@first` to compute the amount of space to be skipped.

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

```
\setlength{\L@first}{\LettrineHeight}%
\setlength{\L@lraise}{\L@raise\L@target@tht}%
\addtolength{\L@first}{\L@lraise}%
\ifnum\value{L@lines}=1
  \addtolength{\L@first}{-\ht\L@tbox}%
\else
  \addtolength{\L@first}{-\L@target@tht}%
  \addtolength{\L@lraise}{-\L@target@dp}%
\fi
\par
```

`\L@first` now holds the height of the needed `\vskip`; if too small it will be discarded.

```
\ifdim\L@first>\L@novskip
```

When the `grid` option is `true`, let's round up `\L@first` to the next integer number of `\baselineskip`.

```
  \ifL@grid
    \@tempdima=\baselineskip
    \@tempdimb=\@tempdima
    \advance\@tempdimb by \L@novskip
    \@tempcnta=1
    \loop\ifdim\L@first>\@tempdimb
       \advance\@tempcnta by 1
       \advance\L@first by -\@tempdima
    \repeat
    \L@first=\@tempcnta\baselineskip
  \fi
  \vskip\L@first
\fi
```

Again, 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`).

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

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

```
  \smash{\llap{\mbox{\L@ante}\raisebox{\L@lraise}{\usebox{\L@lbox}}%
         \hskip \the\L@Findent}}%
  \usebox{\L@tbox}%
```

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
                 }%
}
```

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}{0.2pt}
%%
%% Theses are *flags* (value=true/false):
%%\LettrineImagefalse
%%\LettrineOnGridfalse
%%\LettrineRealHeightfalse
%%
%% 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

v2.30

- General: Added global options to package lettrine. 6

v2.23

- General: Documentation cleanup. . . 1

v2.22

- `\compute@fontsize`: Warn if `\fontsize` computation fails due to division by 0. 8
- `\lettrine`: `\@totalleftmargin` is the correct indentation for quote, quotation and abstract environments. 13
  - `\parshape` reset added in lists. . . 14

v2.21

- General: `\LettrineFontEPS` renamed as `\LettrineImage`. 9
  - Code clean up, new commands `\computeL@height`, `\compute@fontsize`, `\setupL@lbox`. 7
- `\lettrine`: `findent` computation relative to `\LettrineWidth` in `.cfl` files fixed; thanks to Frank Mittelbach for raising the issue. . 11

v2.2

- General: Rollback mechanism used for recovering older versions. 5

v2.1

- General: New option ‘refstring’. 6
  - Newif `\ifLettrineVone` and new option ‘Vone’ (removed in v2.2, rollback prefered). 6
- `\compute@fontsize`: Computation of `\L@factor` for `\fontsize` done by the minifp package. 8
- `\computeL@height`: Height computations moved out of `\LettrineFont`: `\global` settings no longer required. 8

v2.0

- `\computeL@height`: Store targeted dimensions of the dropped capital (ht, dp, tht) for further use. 8
- `\lettrine`: Add informations about targeted and actual height of the initial to the .log file. 12
- `\LettrineTestString`: changed from ‘ABCDEFGHIJKLMNOPQRSTUVWXYZ’ to ‘EFTZ’ as some capitals like C, G, O, Q or X might be slightly taller (possible optical correction). 8

v1.9

- General: New customisable string `\LettrineSecondString` to tune the initial’s height. 3
  - New keyval option: ‘realheight’ (true/false) and new global flag `\ifLettrineRealHeight`. 2
- `\computeL@height`: `\theL@lines` changed to `\value\{L@lines\}`. Needed for babel-hebrew which redefines `\@arabic`. 8
- `\lettrine`: `\theDefaultLines` changed to `\value\{DefaultLines\}`, same with `\theDefaultDepth`. Needed for babel-hebrew which redefines `\@arabic`. Thanks to Ulrike Fischer for providing the fix. . . . 10
  - `\theL@depth` changed to `\value\{L@depth\}`. 13
  - Use the second mandatory argument of `\lettrine` or `\LettrineSecondString` (which defaults to ‘x’) to compute `\L@height`. This is controlled by the ‘realheight’ flag. 10

v1.8

- General: Added newif `\ifLettrineOnGrid` and new dimen `\DiscardVskip`, default (0.2pt) set for compatibility with previous releases. 6
  - Added two keyval options: ‘grid’ (true/false) and ‘novskip’ to override `\DiscardVskip`. 2
- `\lettrine`: The 0.2pt limit for discarded vskips is now customisable through `\DiscardVskip` and option ‘novskip’. 12

v1.7

- General: New counter to add lines for dropped capitals with positive depth, like Q. 1

v1.65

- `\lettrine`: Measure and store the initial's final dimensions. 12

v1.64

- `\lettrine`: Remove $ around `\smash` and add `\relax`. Bug pointed out by David Monniaux. Correction by Enrico Gregorio. 13

v1.63

- `\LettrineTestString`: (new) it defaults to 'ABCDEFGHIJKLMNO-QPRSTUVWXYZ'. In previous versions height computations were based on letter 'X' which might not exist in some (rare) fonts. Pointed out by Raphaël Pinson. 8

v1.6

- General: Add a flag to switch to images in eps or pdf format. Suggested by Bill Jetzer. 2
  - Added newif `\ifL@grid`. 7
  - Added newif `\ifL@image`. 7
  - Added newif `\ifLettrineImage`. . 6
- `\lettrine`: Add braces around #3 to allow commands taking an argument (such as `\MakeLowercase`) in `\LettrineTextFont`. Suggested by Philipp Lehman. 12
- `\LettrineImage`: Added `\LettrineFontHook` to `\LettrineFontEPS`. This is needed for color options. 9

v1.5

- General: `\LettrineOptionsFor` and `\LettrineWidth` added. 10
- `\lettrine`: Added reading of an optional config file `\DefaultOptionsFile`. 11

v1.4

- `\lettrine`: `\lettrine` still didn't work properly in quote, quotation, abstract environments, pointed out by Matthias C. Schmidt. `\rightmargin` was added too early to `\L@Nindent`, thus making `\\L @next` too short by `\rightmargin`. 13

v1.3

- General: Correct the documentation to mention the cm-super fonts and the type1ec package by Vladimir Volovich. 4

v1.2

- General: `\newlength` changed to `\newdimen`, to correct a bug with seminar.cls (pointed out by Peter Münster). 6
- `\computeL@height`: `\baselineskip` may be a rubber length, we convert it to a dimen. 8
- `\lettrine`: `\baselineskip` may be a rubber length, we convert it to a dimen. 12

v1.1

- `\lettrine`: Add `\rightmargin` to `\L@Pindent` for `\Lettrine` to work properly in quote, quotation, abstract environments… but do not change `\linewidth` which is set by these environments. 13