

A Babel language definition file for French

frenchb.dtx v3.5e, 2019/03/30

Daniel Flipo
daniel.flipo@free.fr

Contents

1 The French language	2
1.1 Basic interface	2
1.2 Customisation	5
1.2.1 <code>\frenchsetup</code>	5
1.2.2 Caption names	9
1.2.3 Figure and table captions	10
1.3 Hyphenation checks	10
1.4 Changes	11
2 The code	14
2.1 Initial setup	14
2.2 Punctuation	17
2.2.1 Punctuation with LuaTeX	20
2.2.2 Punctuation with XeTeX	30
2.2.3 Punctuation with standard (pdf)TeX	33
2.2.4 Punctuation switches common to all engines	35
2.3 Commands for French quotation marks	36
2.4 Date in French	40
2.5 Extra utilities	41
2.6 Formatting numbers	45
2.7 Caption names	47
2.8 Figure and table captions	49
2.9 Dots...	52
2.10 More checks about packages' loading order	53
2.11 Setup options: keyval stuff	53
2.12 French lists	67
2.13 French indentation of sections	72
2.14 Formatting footnotes	73
2.15 Clean up and exit	77
2.16 Files <code>frenchb.ldr</code> , <code>francais.ldr</code> , <code>canadien.ldr</code> and <code>acadian.ldr</code>	77
3 Change History	80

1 The French language

The file `frenchb.dtx`¹, defines all the language definition macros for the French language.

Customisation for the French language is achieved following the book “Lexique des règles typographiques en usage à l’Imprimerie Nationale” troisième édition (1994), ISBN-2-11-081075-0.

First version released: 1.1 (May 1996) as part of babel-3.6beta. Version 2.0a was released in February 2007 and version 3.0a in February 2014.

babel-french has been improved using helpful suggestions from many people, mainly from Jacques André, Michel Bovani, Thierry Bouche, Vincent Jalby, Denis Bitouzé and Ulrike Fisher. Thanks to all of them!

LaTeX-2.09 is no longer supported. This new version (3.x) has been designed to be used only with LaTeX2e and Plain formats based on TeX, pdfTeX, LuaTeX or XeTeX engines.

Changes between version 3.0 and v3.5e are listed in subsection 1.4 p. 11.

An extensive documentation in French (file `frenchb-doc.pdf`) is now included in babel-french.

1.1 Basic interface

In a multilingual document, some typographic rules are language dependent, i.e. spaces before ‘high punctuation’ (: ; ! ?) in French, others modify the general layout (i.e. layout of lists, footnotes, indentation of first paragraphs of sections) and should apply to the whole document.

The French language can be loaded with babel by a command like:

```
\usepackage[german,spanish,french,british]{babel}
```

²

babel-french takes account of babel’s *main language* defined as the *last* option at babel’s loading. When French is not babel’s main language, babel-french does not alter the general layout of the document (even in parts where French is the current language): the layout of lists, footnotes, indentation of first paragraphs of sections are not customised by babel-french.

When French is loaded as the last option of babel, babel-french makes the following changes to the global layout, *both in French and in all other languages*³:

1. the first paragraph of each section is indented (LaTeX only);
2. the default items in itemize environment are set to ‘—’ instead of ‘•’, and all vertical spacing and glue is deleted; it is possible to change ‘—’ to something else (‘-’ for instance) using `\frenchsetup{}` (see section 1.2 p. 5);
3. vertical spacing in general LaTeX lists is shortened;
4. footnotes are displayed “à la française”.

¹The file described in this section has version number v3.5e and was last revised on 2019/03/30.

²Always use `french` as option name for the French language, former aliases `frenchb` or `français` are *deprecated*; expect them to be removed sooner or later!

³For each item, hooks are provided to reset standard LaTeX settings or to emulate the behavior of former versions of babel-french (see command `\frenchsetup{}`, section 1.2 p. 5).

5. the separator following the table or figure number in captions is printed as ‘ – ’ instead of ‘ : ’; for changing this see [1.2.3 p. 10](#).

Regarding local typography, the command `\selectlanguage{french}` switches to the French language⁴, with the following effects:

1. French hyphenation patterns are made active;
2. ‘high punctuation’ characters (: ; ! ?) automatically add correct spacing⁵ in French; this is achieved using callbacks in Lua(La)TeX or ‘XeTeXinterchar’ mechanism in Xe(La)TeX; with TeX’82 and pdf(La)TeX these four characters are made active in the whole document;
3. `\today` prints the date in French;
4. the caption names are translated into French (LaTeX only). For customisation of caption names see section [1.2.2 p. 9](#).
5. the space after `\dots` is removed in French.

Some commands are provided by `babel-french` to make typesetting easier:

1. French quotation marks can be entered using the commands `\og` and `\fg` which work in LaTeX2e and PlainTeX, their appearance depending on what is available to draw them; even if you use LaTeX2e *and* T1-encoding, you should refrain from entering them as `<<~French quotation~>>`: `\og` and `\fg` provide better horizontal spacing (controlled by `\FBguillspace`). If French quote characters are available on your keyboard, you can use them, to get proper spacing in LaTeX2e see option `og=«, fg=»` p. [8](#).

`\og` and `\fg` can be used outside French, they typeset then English quotes “ and ”.

A new command `\frquote{}` has been added in version 3.1 to enter French quotations. `\frquote{texte}` is equivalent to `\og texte \fg{}` for short quotations. For quotations spreading over more than one paragraph, `\frquote` will add at the beginning of every paragraph of the quotation either an opening French guillemet («), or a closing one (») or nothing depending on option `EveryParGuill=open` or `=close` or `=none`, see p. [8](#). Command `\NoEveryParQuote` is provided to locally suppress unwanted guillemets (typically when lists are embedded in `\frquote{}`), it is meant to be used inside an environment or a group.

`\frquote` is recommended to enter embedded quotations “à la française”, several variants are provided through options.

- with all engines: the inner quotation is surrounded by double quotes (“*texte*”) unless option `InnerGuillSingle=true`, then a) the inner quotation is printed as `< texte >` and b) if the inner quotation spreads over more than one paragraph, every paragraph included in the inner quotation starts with a `<` or a `>` or nothing, depending on option `EveryParGuill=open` (default) or `=close` or `=none`.

⁴`\selectlanguage{français}` and `\selectlanguage{frenchb}` are no longer supported.

⁵Well, the automatic insertion may add unwanted spaces in some cases, for correction see `AutoSpacePunctuation` option and `\NoAutoSpacing` command p. [7](#).

- with LuaTeX based engines, it is possible to add a French opening or closing guillemet (« or ») at the beginning of every line of the inner quotation using option `EveryLineGuill=open` or `=close`; note that with any of these options, the inner quotation is surrounded by French guillemets (« and ») regardless option `InnerGuillSingle`; the default is `EveryLineGuill=none` so that `\frquote{}` behaves as with non-LuaTeX engines.

A starred variant `\frquote*` is meant for inner quotations which end together with the outer one: using `\frquote*` for the inner quotation will print only one closing quote character (the outer one) as recommended by the French ‘Imprimerie Nationale’.

2. `\frenchdate{<year>}{<month>}{<day>}` helps typesetting dates in French: `\frenchdate{2001}{01}{01}` will print 1^{er} janvier 2001 in a box without any linebreak.
3. A command `\up` is provided to typeset superscripts like `M\up{me}` (abbreviation for “Madame”), `1\up{er}` (for “premier”). Other commands are also provided for ordinals: `\ier`, `\iere`, `\iers`, `\ieres`, `\ieme`, `\iemes` (`3\iemes` prints 3^{es}). All these commands take advantage of real superscript letters when they are available in the current font.
4. Family names should be typeset in small capitals and never be hyphenated, the macro `\bsc` (boxed small caps) does this, e.g., `L.\bsc{Lamport}` will print the same as `L.\mbox{\textsc{Lamport}}`. Note that composed names (such as Dupont-Durant) may now be hyphenated on explicit hyphens, this differs from `babel-french v. 1.x`.
5. Commands `\primo`, `\secundo`, `\tertio` and `\quarto` print 1^o, 2^o, 3^o, 4^o. `\FrenchEnumerate{6}` prints 6^o.
6. Abbreviations for “Numéro(s)” and “numéro(s)” (`No` `Nos` `no` and `nos`) are obtained via the commands `\No`, `\Nos`, `\no`, `\nos`.
7. Two commands are provided to typeset the symbol for “degré”: `\degre` prints the raw character and `\degres` should be used to typeset temperatures (e.g., “20-`\degres` C” with a non-breaking space), or for alcohols” strengths (e.g., “45`\degres`” with *no* space in French).
8. In math mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the `TEXbook` p. 134). The command `\DecimalMathComma` makes the comma behave as an ordinary character *when the current language is French* (no space added); as a counterpart, if `\DecimalMathComma` is active, an explicit space has to be added in lists and intervals: `$(0,\ 1)$`, `$(x,\ y)$`. `\StandardMathComma` switches back to the standard behaviour of the comma in French.
The `icomma` package is an alternative workaround.
9. A command `\nombre` was provided in 1.x versions to easily format numbers in slices of three digits separated either by a comma in English or with a

space in French; `\nombre` is now mapped to `\numprint` from `numprint.sty`, see `numprint.pdf` for more information.

10. `babel-french` has been designed to take advantage of the `xspace` package if present: adding `\usepackage{xspace}` in the preamble will force macros like `\fg`, `\ier`, `\ieme`, `\dots`, ... to respect the spaces you type after them, for instance typing `'1\ier juin'` will print '1^{er} juin' (no need for a forced space after `1\ier`).

1.2 Customisation

Customisation of `babel-french` relies on command `\frenchsetup{}` (formerly called `\frenchbsetup{}`, the latter name will be kept for ever to ensure backwards compatibility), options are entered using the `keyval` syntax. The command `\frenchsetup{}` is to appear in the preamble only (after loading `babel`).

1.2.1 `\frenchsetup{options}`

`\frenchsetup{}` and `\frenchbsetup{}` are synonymous; the latter should be preferred as the language name for French in `babel` is no longer `frenchb` but `french`.

`\frenchsetup{ShowOptions}` prints all available options to the `.log` file, it is just meant as a remainder of the list of offered options. As usual with `keyval` syntax, boolean options (as `ShowOptions`) can be entered as `ShowOptions=true` or just `ShowOptions`, the `=true` part can be omitted.

The other options are listed below. Their default value is shown between braces, sometimes followed by a `*`. The `*` means that the default shown applies when `babel-french` is loaded as the *last* option of `babel` —*babel's main language*—, and is toggled otherwise.

`StandardLayout=true (false*)` forces `babel-french` not to interfere with the layout: no action on any kind of lists, first paragraphs of sections are not indented (as in English), no action on footnotes; it is useless unless French is the main language. This option can be used to avoid conflicts with classes or packages which customise lists or footnotes.

`GlobalLayoutFrench=false (true*)` can only be used when French is the main language; setting it to `false` will emulate what prior versions of `babel-french` (pre-2.2) did: lists, and first paragraphs of sections will be displayed the standard way in other languages than French, and “à la française” in French (changing the layout inside a document is a bad practice imho). Note that the layout of footnotes is language independent anyway (see below `FrenchFootnotes` and `AutoSpaceFootnotes`).

`IndentFirst=false (true*)` ; set this option to `false` if you do not want `babel-french` to force indentation of the first paragraph of sections. When French is the main language, this option applies to all languages.

`PartNameFull=false (true)` ; when true, `babel-french` numbers the title of `\part{}` commands as “Première partie”, “Deuxième partie” and so on.

With some classes which change the `\part{}` command (AMS classes do so), you could get “Première partie 1”, “Deuxième partie 2” in the toc; when this occurs, this option should be set to `false`, part titles will then be printed as “Partie I”, “Partie II”.

`ListItemsAsPar=true (false)` setting this option to `true` is recommended: list items will be displayed as paragraphs with indented labels (in the “Imprimerie Nationale” way) instead of having labels hanging into the left margin. How these two layouts differ is shown below:

<p>Text starting at ‘parindent’ \leq Leftmargin — first item running on two lines or more. . . — first second level item on two lines. . . — next one. . . — second item. . .</p>

Default French layout

<p>Text starting at ‘parindent’ \leq Leftmargin — first item running on two lines or more. . . — first second level item on two lines. . . — next one. . . — second item. . .</p>

With `ListItemsAsPar=true`

`StandardListSpacing=true (false*)` ⁶; `babel-french` customises the vertical spaces in the `list` environment, this affects all lists, including `itemize`, `enumerate`, `description`, but also `abstract`, `quote`, `quotation`, `verse`, etc. which are based on `list`. Setting this option to `true` reverts to the standard settings of the `list` environment as defined by the document class.

`StandardItemizeEnv=true (false*)` ; `babel-french` redefines the `itemize` environment to suppress any vertical space between items of `itemize` lists in French and customises left margins. Setting this option to `true` reverts to the standard definition of `itemize`.

`StandardEnumerateEnv=true (false*)` ; starting with version 2.6 `babel-french` redefines the `enumerate` and `description` environments to make left margins match those of the French version of `itemize` lists. Setting this option to `true` reverts to the standard definition of `enumerate` and `description`.

`StandardItemLabels=true (false*)` when set to `true` this option prevents `babel-french` from changing the labels in `itemize` lists in French.

`ItemLabels=\textbullet, \textendash, \ding{43},...(\textemdash*)` ; when `StandardItemLabels=false` (the default), this option enables to choose the label used in French `itemize` lists for all levels. The next four options do the same but each one for a specific level only. Note that the example `\ding{43}` requires `\usepackage{pifont}`.

`ItemLabeli=\textbullet, \textendash, \ding{43},...(\textemdash*)`

`ItemLabelii=\textbullet, \textendash, \ding{43},...(\textemdash*)`

⁶This option should be used instead of former option `ReduceListSpacing` (kept for backward compatibility) which could be misleading: with some classes (`smfart`, `smfbook` f.i.) you had to set `ReduceListSpacing=false` to revert to the class settings which actually reduce list’s spacings even more than `babel-french`! `StandardListSpacing=true` replaces `ReduceListSpacing=false`.

`ItemLabeliii=\textbullet, \textendash, \ding{43},..(\textemdash*)`

`ItemLabeliv=\textbullet, \textendash, \ding{43},...(\textemdash*)`

`StandardLists=true (false*)` forbids babel-french to customise any kind of list. Try the option `StandardLists` in case of conflicts with classes or packages that customise lists too. This option is just a shorthand setting all four options `StandardListSpacing=true`, `StandardItemizeEnv=true`, `StandardEnumerateEnv=true` and `StandardItemLabels=true`.

`ListOldLayout=true (false)` ; starting with version 2.6a, the layout of lists has changed regarding leftmargins' sizes and default itemize label ('—' instead of '-' up to 2.5k). This option, provided for backward compatibility, displays lists as they were up to version 2.5k.

`FrenchFootnotes=false (true*)` reverts to the standard layout of footnotes. By default babel-french typesets leading numbers as '1. ' instead of '1', but has no effect on footnotes numbered with symbols (as in the `\thanks` command). Two commands `\StandardFootnotes` and `\FrenchFootnotes` are available to change the layout of footnotes locally; `\StandardFootnotes` can help when some footnotes are numbered with letters (inside minipages for instance).

`AutoSpaceFootnotes=false (true*)` ; by default babel-french adds a thin space in the running text before the number or symbol calling the footnote. Making this option `false` reverts to the standard setting (no space added).

`AutoSpacePunctuation=false (true)` ; in French, the user *should* input a space before the four characters ' ; ! ? ' but as many people forget about it (even among native French writers!), the default behaviour of babel-french is to automatically typeset non-breaking spaces the width of which is either `\FBthinspace` (defaults to a thin space) before ' ; ' ! ' ? ' or `\FBcolonspace` (defaults to `\space`) before ' : ' ; the defaults follow the French 'Imprimerie Nationale's recommendations. This is convenient in most cases but can lead to addition of spurious spaces in URLs, in MS-DOS paths or in timetables (10:55) —this no longer occurs with LuaTeX—, except if they are typed in `\texttt` or `verbatim` mode. When the current font is a monospaced (typewriter) font, no spurious space is added in that case⁷, so the default behaviour of of babel-french in that area should be fine in most circumstances.

Choosing `AutoSpacePunctuation=false` will ensure that a proper space is added before ' : ; ! ? ' *if and only if* a (normal) space has been typed in. This option gives full control on space insertion before ' : ; ! ? '. Those who are unsure about their typing in this area should stick to the default option and use the provided `\NoAutoSpacing` command inside a group in case an unwanted space is added by babel-french (i.e. `{\NoAutoSpacing http://mysite}`⁸ or `{\NoAutoSpacing ???}` (needed for pdfTeX only).

⁷Unless option `OriginalTypewriter` is set, `\ttfamily` is redefined in French to switch off space tuning, see below.

⁸Actually, this is needed only with the XeTeX and pdfTeX engines. LuaTeX no longer inserts any space in strings like `http://mysite`, `C:\Foo`, `10:55`...

`ThinColonSpace=true (false)` changes the inter-word non-breaking space added before the colon ‘:’ to a thin space, so that the same amount of space is added before any of the four ‘high punctuation’ characters. The default setting is supported by the French ‘Imprimerie Nationale’.

`OriginalTypewriter=true (false)` prevents any customisation of `\ttfamily` and `\texttt{}` in French. This option should only be used to ensure backward compatibility. The current default behaviour is to switch off any addition of space before high punctuation with typewriter fonts (e.g. `verbatim`).

`UnicodeNoBreakSpaces=true (false)` ; (experimental) this option should be set to `true` *only while converting LuaLaTeX files* to HTML. It ensures that non-breaking spaces added by `babel-french` are inserted in the PDF file as U+A0 or U+202F (thin) instead of penalties and glues. Note that `lwarp` (v. 0.37 and up) is fully compatible with `babel-french` for translating PDFLaTeX or XeLaTeX files to HTML.

`og=«, fg=»` ; when guillemets characters are available on the keyboard (through a compose key for instance), it is nice to use them instead of typing `\og` and `\fg`. This option tells `babel-french` which characters are opening and closing French guillemets (they depend on the input encoding), then you can type either `« guillemets »` or `«guillemets»` (with or without spaces) to get properly typeset French quotes. This option works with LuaLaTeX and XeLaTeX; with pdfLaTeX it requires `inputenc` to be loaded with a proper encoding: 8-bits encoding (`latin1`, `latin9`, `ansinew`, `applemac`,...) or multi-byte encoding (`utf8`, `utf8x`).

`INGuillSpace=true (false)` resets the dimensions of spaces after opening French quotes and before closing French quotes to the French ‘Imprimerie Nationale’ standards (inter-word space). `babel-french`’s default setting produces slightly narrower spaces with less stretchability.

`EveryParGuill=open, close, none (open)` ; sets whether an opening quote (`«`) or a closing one (`»`) or nothing should be printed by `\frquote{}` at the beginning of every paragraph included in a level 1 (outer) quotation. This option is also considered for level 2 (inner) quotations to decide between `<` and `>` when `InnerGuillSingle=true` (see below).

`EveryLineGuill=open, close, none (none)` ; with LuaTeX based engines *only*, it is possible to set this option to `open` [resp. `close`]; this ensures that a ‘`«`’ [resp. ‘`»`’] followed by a proper space will be inserted at the beginning of every line of embedded (inner) quotations spreading over more than one line (provided that both outer and inner quotations are entered with `\frquote{}`). When `EveryLineGuill=open` or `=close` the inner quotation is always surrounded by `«` and `»`, the next option is ineffective.

`InnerGuillSingle=true (false)` ; if `InnerGuillSingle=false` (default), inner quotations entered with `\frquote{}` start with “ and end with ”. If `InnerGuillSingle=true`, `<` and `>` are used instead of British double quotes; moreover if option `EveryParGuill=open` (or `close`) is set, a `<` (or `>`) is added at the beginning of every paragraph included in the inner quotation.

`ThinSpaceInFrenchNumbers=true (false)` ; if `numprint` has been loaded with the `autoLanguage` option, while typesetting numbers with the `\numprint{}` command, `\npthousandsep` is defined as a non-breaking space (~)⁹ in French; when set to `true`, this option redefines `\npthousandsep` as a thin space (`\,`).

`SmallCapsFigTabCaptions=false (true*)` ; when set to `false`, `\figurename` and `\tablename` will be printed in French captions as “Figure” and “Table” instead of being printed in small caps (the default).

`CustomiseFigTabCaptions=false (true*)` ; when `false` the default separator (colon) is used instead of `\CaptionSeparator`. Anyway, `babel-french` tries hard to insert a proper space before it and warns if it fails to do so.

`OldFigTabCaptions=true (false)` is to be used when figures’ and tables’ captions must be typeset as with pre 3.0 versions of `babel-french` (with `\CaptionSeparator` in French and colon otherwise). Intended for standard LaTeX classes only.

`FrenchSuperscripts=false (true)` ; then `\up=\textsuperscript`. (option added in version 2.1). Should only be made `false` to recompile documents written before 2008 without changes: by default `\up` now relies on `\fup` designed to produce better looking superscripts.

`LowercaseSuperscripts=false (true)` ; by default `babel-french` inhibits the uppercasing of superscripts (for instance when they are moved to page headers). Making this option `false` will disable this behaviour (not recommended).

`SuppressWarning=true (false)` ; can be turned to `true` if you are bored with `babel-french`’s warnings; use this option as *first* option of `\frenchsetup{}` to cancel warnings launched by other options.

Options’ order – Please remember that options are read in the order they appear in the `\frenchsetup{}` command. Someone wishing that `babel-french` leaves the layout of lists and footnotes untouched but caring for indentation of first paragraph of sections should choose `\frenchsetup{StandardLayout, IndentFirst}` to get the expected layout. The reverse order `\frenchsetup{IndentFirst, StandardLayout}` would lead to option `IndentFirst` being overwritten by `StandardLayout`.

1.2.2 Caption names

All caption names can easily be customised in French using the simplified syntax introduced by `babel` 3.9, for instance `\def\frenchproofname{Preuve}` or `\def\acadianproofname{Preuve}` for the acadian dialect. The older syntax `\addto\captionsfrench{\def\proofname{Preuve}}` still works. Keep in mind that *only* `french` can be used to redefine captions, even if `babel`’s option was entered as `frenchb` or `français`.

⁹Actually without stretch nor shrink.

1.2.3 Figure and table captions

In French, captions in figures and tables should never be printed as ‘Figure 1: ’ which is the default in standard LaTeX2e classes (a space should *always* precede a colon in French), anyway ‘Figure 1 – ’ is preferred.

When French is the main language, the default behaviour of babel-french is to change the separator (colon) used in figures’ and tables’ captions *for all languages* to `\CaptionSeparator` which defaults to ‘ – ’ and can be redefined in the preamble with `\renewcommand*{\CaptionSeparator}{...}`. This works for the standard LaTeX2e classes, for the memoir and koma-script classes. In case this procedure fails a warning is issued.

When French is not the main language, the colon is preserved for all languages including French but babel-french tries hard to insert a proper space before it and warns if it fails to do so.

Three options are provided to customise figure and table captions:

- if `CustomiseFigTabCaptions` is set to `false` the colon will be used as separator in all languages, with a proper space before the colon in French (if possible);
- the second option, `OldFigTabCaptions`, can be set to `true` to print figures’ and tables’ captions as they were with versions pre 3.0 of babel-french (using `\CaptionSeparator` in French and colon in other languages); this option only makes sense with the standard LaTeX classes `article`, `report` and `book`;
- the last option, `SmallCapsFigTabCaptions`, can be set to `false` to typeset `\figurename` and `\tablename` in French as “Figure” and “Table” rather than in small caps (the default).

1.3 Hyphenation checks

Once you have built your format, a good precaution would be to perform some basic tests about hyphenation in French. For LaTeX2e I suggest this:

- run pdfLaTeX on the following file:

```
%%% Test file for French hyphenation.
\documentclass[french]{article}
\usepackage[utf8]{inputenc} % utf8, what else?
\usepackage[T1]{fontenc} % mandatory for French
\usepackage{lmodern} % or erewhon, palatino...
\usepackage{babel}
\begin{document}
\showhyphens{signal container \’ev\’enement alg\’ebre}
\showhyphens{signal container événement algèbre}
\end{document}
```

- check the hyphenations proposed by T_EX in your log-file; in French you should get with both 7-bit and 8-bit encodings
si-gnal contai-ner évé-ne-ment al-gèbre.

Do not care about how accented characters are displayed in the log-file, what matters is the position of the ‘-’ hyphen signs *only*.

If they are all correct, your installation (probably) works fine, if one (or more) is (are) wrong, ask a local wizard to see what’s going wrong and perform the test again (or e-mail me about what happens).

Frequent mismatches:

- you get sig-nal con-tainer, this probably means that the hyphenation patterns you are using are for US-English, not for French;
- you get no hyphen at all in évé-ne-ment, this probably means that you are using CM fonts and the macro `\accent` to produce accented characters. Using 8-bits fonts with built-in accented characters avoids this kind of mismatch.

1.4 Changes

What’s new in version 3.5?

Version 3.5a offers a new option `ListItemsAsPar`. The default layout of lists is unchanged (for backward compatibility), but users should try this new option which ensures a layout of lists closer to French typographic standards: see f.i. how lists are typeset in the book “Lexique des règles typographiques en usage à l’Imprimerie Nationale”.

Version 3.5b fixes a bug due to wrong `\everypar`’s management in `\frquote{}`; it showed up when `\frquote{}` immediately followed a sectioning command. Starting with version 3.5d, a new option `StandardListSpacing` has been added to supersede `ReduceListSpacing`.

A new command `\NoEveryParQuote` has been added in version 3.5e: it is meant to be used inside a group or environment to suppress unwanted guillemets (typically when lists are embedded in `\frquote{}`).

What’s new in version 3.4?

Version 3.4a adds a new command `\frenchdate` (see p. 4) and slightly changes number formatting: `\FBthousandsep` is now a *kern* instead of a rubber length. `\renewcommand*{\FBthousandsep}{~}` will switch back to the former (wrong) behaviour.

Both options `french` and `acadian` can now be used simultaneously in a document; currently `french` and `acadian` are identical, it is up to the user to customise `acadian` in terms of hyphenation patterns, captionnames, date format or high punctuation and quotes spacing if he/she needs a variant for French.

A new command `\FBsetspaces` has been added for easy customising of spacing before high punctuation and inside quotes independently for `french` and `acadian`, see p. 18.

Version 3.4 requires eTeX and LuaTeX 1.0.4 or newer.

What's new in version 3.3?

In version 3.3d the automatic insertion of non-breaking spaces before the colon character has been improved *with engine LuaTeX only*: a spurious space is no longer inserted in strings like `http://mysite`, `C:\Program Files` or `10:55`. Unfortunately, my attempts to do the same with XeTeX or pdfTeX were unsuccessful.

A few internal changes have been made in version 3.3c to improve the conversion into HTML of non-breaking spaces added by `babel-french`. Usage of `\lwrap` (v.0.37 and up) is recommended for HTML output, it works fine on files compiled with XeLaTeX or pdfLaTeX formats. A new experimental option `UnicodeNoBreakSpaces` has been added for LuaLaTeX in version 3.3c, see p. 8.

According to current `babel`'s standards, every dialect should have its own `.ldf` file; starting with version 3.3b, the main support for French is in `french.ldf`, portmanteau files `frenchb.ldf`, `francais.ldf`, `acadian.ldf` and `canadien.ldf` have been added. Recommended options are `french` or `acadian`, all other are deprecated. BTW, options `french` and `acadian` are currently strictly identical.

Release 3.3a is compatible with LuaTeX v. 0.95 (TL2016) and up. Former skips `\FBcolonskip`, `\FBthinskip` and `\FBguillskip` controlling punctuation spacings in LuaTeX have been removed; all three engines now rely on the same commands `\FBcolonspace`, `\FBthinspace` and `\FBguillspace`.

An alias `\frenchsetup{}` for `\frenchbsetup{}` has been added in version 3.3a, it might appear more relevant in the future as the language name `frenchb` should vanish.

Further customisation of the `\part{}` command is provided via three new commands `\frenchpartfirst`, `\frenchpartsecond` and `\frenchpartnameord`.

What's new in version 3.2?

Version 3.2g changes the default behaviour of `\frquote{}` with LuaTeX based engines, the output is now the same with all engines; to recover the former behaviour, add option `EveryLineGuill=open`.

The handling of footnotes has been redesigned for the `beamer`, `memoir` and `koma-script` classes. The layout of footnotes "à la française" should be unchanged but footnotes' customisations offered by these classes (i.e. font or color changes) are now available even when option `FrenchFootnotes` is `true`.

A long standing bug regarding the `xspace` package has been fixed: `\xspace` has been moved up from the internal command `\FB@fg` to `\fg`; `\frquote{}` now works properly when the `xspace` package is loaded.

Version 3.2b is the first one designed to work with LuaTeX v. 0.95 as included in TeXLive 2016 (LuaTeX's new glue node structure is not compatible with previous versions).

Warning to Lua(La)TeX users: starting with version 3.2b the lua code included in `frenchb.lua` will *not work* on older installations (TL2015 f.i.), so `babel-french` reverts to active characters while handling high punctuation with LuaTeX engines older than 0.95! The best way to go is to upgrade to TL2016 or equivalent asap. Xe(La)TeX and pdf(La)TeX users can safely use `babel-french` v. 3.2b and later on older installations too.

The internals of commands `\NoAutoSpacing`, `\ttfamilyFB`, `\rmfamilyFB` and `\sffamilyFB` have been completely redesigned in version 3.2c, they behave now consistently with all engines.

What's new in version 3.1?

New command `\frquote{}` meant to enter French quotations, especially long ones (spreading over several paragraphs) and/or embedded ones. see p. 3 for details.

What's new in version 3.0?

Many deep changes lead me to step babel-french's version number to 3.0a:

- babel 3.9 is required now to process `frenchb.ldf`, this change allows for cleaner definitions of dates and captions for the Unicode engines LuaTeX and XeTeX and also provides a simpler syntax for end-users, see section 1.2.2 p.9.
- `\frenchsetup{}` options management has been completely reworked; two new options added.
- Canadian French didn't work as a normal babel's dialect, it should now; btw. the French language should now be loaded as `french`, *not as* `frenchb` or `français` and preferably as a *global* option of `\documentclass`. Some tolerance still exists in v3.0, but do not rely on it.
- babel-french no longer loads `frenchb.cfg`: customisation should definitely be done using `\frenchsetup{}` options.
- Description lists labels are now indented; try setting `\descindentFB=0pt` (or `\listindentFB=0pt` for all lists) in the preamble if you don't like it.
- The last but not least change affects the (recent) LuaTeX-based engines, (this means version 0.76 as included in TL2013 and up): active characters are no longer used in French for 'high punctuation' ¹⁰. Functionalities and user interface are unchanged.

Many thanks to Paul Isambert who provided the basis for the lua code (see his presentation at GUT'2010) and kindly reviewed my first drafts suggesting significant improvements.

Please note that this code, still experimental, is likely to change until LuaTeX itself has reached version 1.0.

Starting with version 3.0c, babel-french no longer customises lists with the beamer class and offers a new option (`INGuillSpace`) to follow French 'Imprimerie Nationale' recommendations regarding quotes' spacing.

¹⁰The current babel-french version requires LuaTeX v. 1.0.4 as included in TL2017, see above.

2 The code

2.1 Initial setup

The macro `\LdfInit` takes care of preventing that this file is loaded more than once (even if both options `french` and `acadian` are used in the same document), checking the category code of the `@` sign, etc.

```
1 <*french>
2 \LdfInit\CurrentOption{FBclean@on@exit}
```

Let's provide a substitute for `\PackageError`, `\PackageWarning` and `\PackageInfo` not defined in Plain:

```
3 \def\fb@error#1#2{%
4   \begingroup
5     \newlinechar='\^^J
6     \def\{\^^J(french.ldf) }%
7     \errhelp{#2}\errmessage{\#\1^^J}%
8   \endgroup}
9 \def\fb@warning#1{%
10  \begingroup
11    \newlinechar='\^^J
12    \def\{\^^J(french.ldf) }%
13    \message{\#\1^^J}%
14  \endgroup}
15 \def\fb@info#1{%
16  \begingroup
17    \newlinechar='\^^J
18    \def\{\^^J}%
19    \wlog{#1}%
20  \endgroup}
```

Quit if eTeX is not available.

```
21 \let\bbl@tempa\relax
22 \begingroup\expandafter\expandafter\expandafter\endgroup
23 \expandafter\ifx\csname eTeXversion\endcsname\relax
24   \let\bbl@tempa\endinput
25   \fb@error{babel-french requires eTeX.\\
26     Aborting here}
27     {Original PlainTeX is not supported,\\
28     please use LuaTeX or XeTeX engines.}
29 \fi
30 \bbl@tempa
```

Quit if babel's version is less than 3.9i.

```
31 \let\bbl@tempa\relax
32 \ifdefined\babeltags
33 \else
34   \let\bbl@tempa\endinput
35   \ifdefined\PackageError
36     \PackageError{french.ldf}
37     {babel-french requires babel v.3.16.\MessageBreak
```

```

38     Aborting here}
39     {Please upgrade Babel!}
40 \else
41   \fb@error{babel-french requires babel v.3.16.\\
42     Aborting here}
43     {Please upgrade Babel!}
44 \fi
45 \fi
46 \bbl@tempa

```

Make sure that `\l@french` is defined (fallbacks are `\l@nohyphenation` if available or 0). `babel.def` (3.9i and up) defines `\l@<language>` also for eTeX, LuaTeX and XeTeX formats which set `\lang@<language>`.

```

47 \def\FB@nopatterns{%
48   \ifdefined\l@nohyphenation
49     \addialect\l@french\l@nohyphenation
50     \edef\bbl@nulllanguage{\string\language=nohyphenation}%
51   \else
52     \edef\bbl@nulllanguage{\string\language=0}%
53     \addialect\l@french0
54   \fi
55   \@nopatterns{French}}
56 \ifdefined\l@french \else \FB@nopatterns \fi

```

Babel's French language can be loaded with option `acadian` which stands for Canadian French. If no specific hyphenation patterns are available, Canadian French will use the French ones.

```

57 \ifdefined\l@acadian \else \addialect\l@acadian\l@french \fi

```

French uses the standard values of `\leftthyphenmin` (2) and `\rightthyphenmin` (3); let's provide their values though, as required by babel.

```

58 \providehyphenmins{french}{\tw@\thr@@}
59 \providehyphenmins{acadian}{\tw@\thr@@}

```

`\ifLaTeXe` No support is provided for late LaTeX-2.09: issue a warning and exit if LaTeX-2.09 is in use. Plain is still supported.

```

60 \newif\ifLaTeXe
61 \let\bbl@tempa\relax
62 \ifdefined\magnification
63 \else
64   \ifdefined\@compatibilitytrue
65     \LaTeXtrue
66   \else
67     \PackageError{french.ldf}
68       {LaTeX-2.09 format is no longer supported.\MessageBreak
69       Aborting here}
70       {Please upgrade to LaTeX2e!}
71     \let\bbl@tempa\endinput
72   \fi
73 \fi
74 \bbl@tempa

```

`\ifFBunicode` French hyphenation patterns are now coded in Unicode, see file `hyph-fr.tex`. XeTeX and LuaTeX engines require some extra code to deal with the French “apostrophe”.
`\ifBFLuaTeX` Let’s define three new ‘if’: `\ifBFLuaTeX`, `\ifFBXeTeX` and `\ifFBunicode` which will be true for XeTeX and LuaTeX engines and false for 8-bits engines.

```

75 \newif\ifFBunicode
76 \newif\ifBFLuaTeX
77 \newif\ifFBXeTeX
78 \begingroup\expandafter\expandafter\expandafter\endgroup
79 \expandafter\ifx\csname luatexversion\endcsname\relax
80 \else
81   \FBunicodetrue \BFLuaTeXtrue
82 \fi
83 \begingroup\expandafter\expandafter\expandafter\endgroup
84 \expandafter\ifx\csname XeTeXrevision\endcsname\relax
85 \else
86   \FBunicodetrue \FBXeTeXtrue
87 \fi

```

`\ifBfrench` True when the current language is French or any of its dialects; will be set to true by `\extrasfrench` and to false by `\noextrasfrench`. Used in `\DecimalMathComma` and `frenchsetup{og=«, fg=»}`.

```
88 \newif\ifBfrench
```

`\extrasfrench` The macro `\extrasfrench` will perform all the extra definitions needed for the French language. The macro `\noextrasfrench` is used to cancel the actions of `\extrasfrench`.

In French, character “apostrophe” (U+27 or U+2019) is a letter in expressions like `l’ambulance` (French hyphenation patterns provide entries for this kind of words). This means that the `\lccode` of “apostrophe” has to be non null in French for proper hyphenation of those expressions, and has to be reset to null when exiting French. The following code ensures correct hyphenation of words like `d’aventure`, `l’utopie`, with all TeX engines (XeTeX, LuaTeX, pdfTeX) using `hyph-fr.tex` patterns.

```

89 \def\extrasfrench{%
90   \FBfrenchtrue
91   \babel@savevariable{\lccode"27}%
92   \lccode"27="27
93   \ifFBunicode
94     \babel@savevariable{\lccode"2019}%
95     \lccode"2019="2019
96   \fi
97 }
98 \def\noextrasfrench{\FBfrenchfalse}

```

One more thing `\extrasfrench` needs to do is to make sure that “Frenchspacing” is in effect. `\noextrasfrench` will switch “Frenchspacing” off again if necessary.

```

99 \addto\extrasfrench{\bbl@frenchspacing}
100 \addto\noextrasfrench{\bbl@nonfrenchspacing}

```


2.2 Punctuation

As long as no better solution is available, the ‘high punctuation’ characters (; ! ? and :) have to be made `\active` for an automatic control of the amount of space to be inserted before them. Both XeTeX and LuaTeX provide an alternative to active characters (‘XeTeXinterchar’ mechanism and LuaTeX’s callbacks).

`\ifFB@active@punct` Three internal flags are needed for the three different techniques used for ‘high punctuation’ management.

```
101 \newif\ifFB@active@punct \FB@active@puncttrue
```

`\ifFB@luatex@punct` With LuaTeX, starting with version 1.0.4, callbacks are used to get rid of active punctuation. With previous versions, ‘high punctuation’ characters remain active (see below).

```
102 \newif\ifFB@luatex@punct
103 \ifB LuaTeX
104 \ifnum\luatexversion<100
105 \ifx\PackageWarning\@undefined
106 \fb@warning{Please upgrade LuaTeX to version 1.0.4 or above!\\%
107 babel-french will make high punctuation characters (;!?)\\%
108 active with LuaTeX < 1.0.4.}%
109 \else
110 \PackageWarning{french.ldf}{Please upgrade LuaTeX
111 to version 1.0.4 or above!\MessageBreak
112 babel-french will make high punctuation characters%
113 \MessageBreak (;!?) active with LuaTeX < 1.0.4;%
114 \MessageBreak reported}%
115 \fi
116 \else
117 \FB@luatex@puncttrue\FB@active@punctfalse
118 \fi
119 \fi
```

`\ifFB@xetex@punct` For XeTeX, the availability of `\XeTeXinterchartokenstate` decides whether the ‘high punctuation’ characters (; ! ? and :) have to be made `\active` or not. The number of available character classes has been increased from 256 to 4096 in XeTeX v. 0.99994, the class for non-characters is now 4095 instead of 255.

```
120 \newcount\FB@nonchar
121 \newif\ifFB@xetex@punct
122 \ifdefined\XeTeXinterchartokenstate
123 \FB@xetex@puncttrue\FB@active@punctfalse
124 \ifdim\the\XeTeXversion\XeTeXrevision pt<0.99994pt
125 \FB@nonchar=255 \relax
126 \else
127 \FB@nonchar=4095 \relax
128 \fi
129 \fi
```

`\FBguillspace` These three commands are meant for basic French. Other French dialects can use
`\FBcolonspace` different settings, see below. According to the I.N. specifications, the ‘:’ requires
`\FBthinspace`

an inter-word space before it, the other three require just a thin space. We define `\FBcolonspace` as `\space` (inter-word space) and `\FBthinspace` as an half inter-word space with no shrink nor stretch. `\FBguillspace` is defined btw. as spacing for French quotes is handled together with high punctuation for LuaTeX and XeTeX. `\FBguillspace` has been fine tuned by Thierry Bouche to 80% of an inter-word space with reduced stretchability. All three are user customisable in the preamble, best using the `\FBsetspaces` command described below. A penalty will be added before these spaces to prevent line breaking.

```

130 \newcommand*{\FBguillspace}{\hskip .8\fontdimen2\font
131                               plus .3\fontdimen3\font
132                               minus .8\fontdimen4\font \relax}
133 \newcommand*{\FBcolonspace}{\space}
134 \newcommand*{\FBthinspace}{\hskip .5\fontdimen2\font \relax}

```

\FBsetspaces This command makes it easy to fine tune `\FBguillspace`, `\FBcolonspace` and `\FBthinspace` in French (default) or independently in a French dialect using the optional argument. They are meant for LaTeX2e *only* and can only be used in the preamble. Four mandatory arguments are expected besides the optional one: the first one is a *string* either "guill", "colon", or "thin", the last four are decimal numbers specifying *width*, *stretch* and *shrink* relative to *fontdimens*. For instance `\FBsetspaces[acadian]{colon}{0.5}{0}{0}` defines `\acadianFBcolonspace` as a thinspace which will be used for the Acadian dialect only. When used without optional argument or with argument 'french', the same command would tune the basic `\FBcolonspace` command.

```

135 \ifLaTeXe
136   \newcommand*{\FBsetspaces}[5][french]{%
137     \def\bb1@tempa{french}\def\bb1@tempb{#1}%
138     \ifx\bb1@tempa\bb1@tempb \def\bb1@tempb{}\fi
139     \@namedef{\bb1@tempb FB#2space}{\hskip #3\fontdimen2\font
140                                   plus #4\fontdimen3\font
141                                   minus #5\fontdimen4\font \relax}%

```

With option "acadian", fill the corresponding LuaTeX table. All unset values in the "acadian" subtables will be filled 'AtBeginDocument' by `\set@glue@table` with the value available for "french".

```

142   \ifFB@luatex@punct
143     \ifx\bb1@tempb\FB@acadian
144       \directlua{
145         FBsp.#2.gl.ac[1] = #3
146         FBsp.#2.gl.ac[2] = #4
147         FBsp.#2.gl.ac[3] = #5
148         if #3 > 0.6 then
149           FBsp.#2.ch.ac = 0xA0
150         elseif #3 > 0.2 then
151           FBsp.#2.ch.ac = 0x202F
152         else
153           FBsp.#2.ch.ac = 0x200B
154         end
155       }%

```

```

156     \fi
157     \fi
158   }
159   \@onlypreamble\FBsetspace
160 \fi

```

Remember that the *same* `\extrasfrench` command is executed when switching to French or to a French dialect (Acadian). Acadian and French may share the same patterns (or not), and may use different spacing for high punctuation and/or quotes. Basically, for pdfLaTeX and XeLaTeX, the spacing is set for French, then potentially tuned differently for Acadian. LuaTeX relies on an attribute `\FB@dialect` to decide what spacing is needed for French or Acadian (see LuaTeX table `FBsp`). As a rough test on `\languagename` would be unreliable to set the value of `\FB@dialect` (see `babel.pdf`), we use a trick based on `\detokenize`; another option would be to use the `\IfLanguageName` command from Oberdiek's package `iflang`.

```

161 \ifLaTeXe
162   \addto\extrasfrench{%
163     \ifFB@luatex@punct
164       \edef\bbl@tempa{\detokenize\expandafter{\languagename}}%
165       \edef\bbl@tempb{\detokenize{french}}%
166       \ifx\bbl@tempa\bbl@tempb \FB@dialect=0 \relax
167       \else \FB@dialect=1 \relax
168     \fi

```

The first time we enter French, we have to set the LuaTeX tables for French (`\FB@dialect=0`) *before* any dialect redefines any `\FB...space` command. Doing this 'AtBeginDocument' would be too late: if French or a French dialect is the main language, `\extrasfrench` has been executed before!

```

169     \ifdefined\FB@once\else
170       \set@glue@table{colon}%
171       \set@glue@table{thin}%
172       \set@glue@table{guill}%
173       \def\FB@once{}%
174     \fi
175   \fi

```

Any dialect dependent customisation done using `\FBsetspace[dialect]` command or alike is now taken into account: the value of `\FBthinspace` (meant for French, i.e. `\FB@dialect=0`) is first saved then changed (for Acadian).

```

176   \ifcsname\languagename FBthinspace\endcsname
177     \babel@save\FBthinspace
178     \renewcommand*{\FBthinspace}{%
179       \csname\languagename FBthinspace\endcsname}%
180   \fi

```

Same for `\FBcolonspace`:

```

181   \ifcsname\languagename FBcolonspace\endcsname
182     \babel@save\FBcolonspace
183     \renewcommand*{\FBcolonspace}{%
184       \csname\languagename FBcolonspace\endcsname}%
185   \fi

```

And for `\FBguillspace`:

```
186 \ifcsname\language\language FBguillspace\endcsname
187 \babel@save\FBguillspace
188 \renewcommand*\FBguillspace{%
189 \csname\language\language FBguillspace\endcsname}%
190 \fi
191 }
192 \fi
```

The conditional `\ifFB@spacing` will be used by pdfTeX and XeTeX engines to switch on or off space tuning before high punctuation and inside French quotes. A matching attribute will be defined later for LuaTeX.

```
193 \newif\ifFB@spacing \FB@spacingtrue
```

`\FB@spacing@off` Two internal commands to switch on and off all space tuning for all six characters
`\FB@spacing@on` ‘;:!?«»’. They will be triggered by user command `\NoAutoSpacing` and by font family switching commands `\ttfamilyFB` `\rmfamilyFB` and `\sffamilyFB`. These four commands will now behave the same with any engine (up to version 3.2b, results were engine dependent).

```
194 \newcommand*\FB@spacing@on{%
195 \ifFB@luatex@punct
196 \FB@spacing=1 \relax
197 \else
198 \FB@spacingtrue
199 \fi}
200 \newcommand*\FB@spacing@off{%
201 \ifFB@luatex@punct
202 \FB@spacing=0 \relax
203 \else
204 \FB@spacingfalse
205 \fi}
```

2.2.1 Punctuation with LuaTeX

The following part holds specific code for punctuation with modern LuaTeX engines, i.e. version 1.0.4 (included in TL2017) or newer.

```
206 \ifFB@luatex@punct
207 \ifdefined\newluafunction\else
```

This code is for Plain: load `ltxlua.tex` if it hasn't been loaded before babel.

```
208 \input ltxlua.tex
209 \fi
```

We define five LuaTeX attributes to control spacing in French and/or Acadian for ‘high punctuation’ and quotes, making sure that `\newattribute` is defined.

`\FB@spacing=0` switches off any space tuning both before high punctuation characters and inside French quotes (i.e. function `french_punctuation` doesn't alter the node list at all).

`\FB@addDPspace=0` switches off automatic insertion of spaces before high punctuation characters (but typed spaces are still turned into non-breaking thin- or word-spaces).

\FB@addGUILspace will be set to 1 by option `og=«, fg=»`, thus enabling automatic insertion of proper spaces after ‘«’ and before ‘»’.

\FB@ucsNBSP triggers the replacement of glues by characters, it is controlled by option `UnicodeNoBreakSpaces`.

\FB@dialect is 0 for French and 1 for Acadian; its value controls which parts of the glue table (`.fr` or `.ac`) are taken into account.

```
210 \newattribute\FB@spacing      \FB@spacing=1 \relax
211 \newattribute\FB@addDPspace  \FB@addDPspace=1 \relax
212 \newattribute\FB@addGUILspace \FB@addGUILspace=0 \relax
213 \newattribute\FB@ucsNBSP     \FB@ucsNBSP=0 \relax
214 \newattribute\FB@dialect     \FB@dialect=0 \relax
215 \ifLaTeXe
216   \PackageInfo{french.ldf}{No need for active punctuation
217     characters\MessageBreak with this version
218     of LuaTeX!\MessageBreak reported}
219 \else
220   \fb@info{No need for active punctuation characters\
221     with this version of LuaTeX!}
222 \fi
```

The next command will be used in the first call of `\extrasfrench` to convert `\FBcolonspace`, `\FBthinspace` and `\FBguillspace` into a table usable by LuaTeX. This way, any customisation done in the preamble (by `\frenchsetup{}`, redefinitions or `\FBsetspaces` commands) are taken into account. Values not explicitly set for Acadian by `\FBsetspaces[acadian]` commands are copied from the French ones. In case parsing by the Lua function `FBget_glue` (defined in file `frenchb.lua`) fails due to unexpected syntax in `\FB...space` the table remains unchanged and a warning is issued. The matching space characters for option `UnicodeNoBreakSpaces` are set as word space, thin space or null space according to the *width* parameter.

```
223 \newcommand*\set@glue@table}[1]{%
224   \directlua {
225     local s = token.get_meaning("FB#1space")
226     local t = FBget_glue(s)
227     if t then
228       FBsp.#1.gl.fr = t
229       if not FBsp.#1.gl.ac[1] then
230         FBsp.#1.gl.ac = t
231       end
232       if FBsp.#1.gl.fr[1] > 0.6 then
233         FBsp.#1.ch.fr = 0xA0
234       elseif FBsp.#1.gl.fr[1] > 0.2 then
235         FBsp.#1.ch.fr = 0x202F
236       else
237         FBsp.#1.ch.fr = 0x200B
238       end
239       if not FBsp.#1.ch.ac then
240         FBsp.#1.ch.ac = FBsp.#1.ch.fr
241       end
242     else
243       texio.write_nl('term and log', '')
```

```

244 texio.write_nl('term and log',
245 '*** french.ldf warning: Unexpected syntax in FB#1space,')
246 texio.write_nl('term and log',
247 '*** french.ldf warning: LuaTeX table FBsp unchanged.')
```

```

248 texio.write_nl('term and log',
249 '*** french.ldf warning: Consider using FBsetspace to ')
250 texio.write('term and log', 'customise FB#1space.')
```

```

251 texio.write_nl('term and log', '')
252 end
253 }%
254 }
255 \fi
256 </french>
```

frenchb.lua This is frenchb.lua. It holds Lua code to deal with ‘high punctuation’ and quotes. This code is based on suggestions from Paul Isambert. First we define two flags to control spacing before French ‘high punctuation’ (thin space or inter-word space).

```

257 <*lua>
258 local FB_punct_thin =
259   {[string.byte("!")] = true,
260    [string.byte("?")] = true,
261    [string.byte(";")] = true}
262 local FB_punct_thick =
263   {[string.byte(":")] = true}
```

Managing spacing after ‘<<’ (U+00AB) and before ‘>>’ (U+00BB) can be done by the way; we define two flags, FB_punct_left for characters requiring some space before them and FB_punct_right for ‘<<’ which must be followed by some space. In case LuaTeX is used to output T1-encoded fonts instead of OpenType fonts, codes 0x13 and 0x14 have to be added for ‘<<’ and ‘>>’.

```

264 local FB_punct_left =
265   {[string.byte("!")] = true,
266    [string.byte("?")] = true,
267    [string.byte(";")] = true,
268    [string.byte(":")] = true,
269    [0x14] = true,
270    [0xBB] = true}
271 local FB_punct_right =
272   {[0x13] = true,
273    [0xAB] = true}
```

Two more flags will be needed to avoid spurious spaces in strings like !! ?? or (?)

```

274 local FB_punct_null =
275   {[string.byte("!")] = true,
276    [string.byte("?")] = true,
277    [string.byte("[")] = true,
278    [string.byte("(")] = true,
```

or if the user has typed a non-breaking space U+00A0 or U+202F (thin) before a ‘high punctuation’ character: no space should be added by babel-french. Same is true inside French quotes.

```

279   [0xA0]           = true,
280   [0x202F]        = true}
281 local FB_guil_null =
282   {[0xA0]          = true,
283    [0x202F]        = true}

```

Local definitions for nodes:

```

284 local new_node     = node.new
285 local copy_node    = node.copy
286 local node_id      = node.id
287 local HLIST        = node_id("hlist")
288 local TEMP          = node_id("temp")
289 local KERN          = node_id("kern")
290 local GLUE          = node_id("glue")
291 local GLYPH        = node_id("glyph")
292 local PENALTY      = node_id("penalty")
293 local nobreak       = new_node(PENALTY)
294 nobreak.penalty    = 10000
295 local insert_node_before = node.insert_before
296 local insert_node_after  = node.insert_after
297 local remove_node       = node.remove

```

Commands `\FBthinspace`, `\FBcolonspace` and `\FBguillspace` are converted 'At-BeginDocument' by the next function `FBget_glue` into tables of three values which are fractions of `\fontdimen2`, `\fontdimen3` and `\fontdimen4`. If parsing fails due to unexpected syntax, the function returns *nil* instead of a table.

```

298 function FBget_glue(toks)
299   local t = nil
300   local f = string.match(toks,
301     "[^%w]hskip%s*([%d%.]*)%s*[^%w]fontdimen 2")
302   if f == "" then f = 1 end
303   if tonumber(f) then
304     t = {tonumber(f), 0, 0}
305     f = string.match(toks, "plus%s*([%d%.]*)%s*[^%w]fontdimen 3")
306     if f == "" then f = 1 end
307     if tonumber(f) then
308       t[2] = tonumber(f)
309       f = string.match(toks, "minus%s*([%d%.]*)%s*[^%w]fontdimen 4")
310       if f == "" then f = 1 end
311       if tonumber(f) then
312         t[3] = tonumber(f)
313       end
314     end
315   elseif string.match(toks, "[^%w]F?B?thinspace") then
316     t = {0.5, 0, 0}
317   elseif string.match(toks, "[^%w]space") then
318     t = {1, 1, 1}
319   end
320   return t
321 end

```

Let's initialize the global LuaTeX table `FBsp`: it holds the characteristics of the glues

used in French and Acadian for high punctuation and quotes and the corresponding no-breaking space characters for option [UnicodeNoBreakSpaces](#).

```

322 FBsp = {}
323 FBsp.thin = {}
324 FBsp.thin.gl = {}
325 FBsp.thin.gl.fr = {.5, 0, 0} ; FBsp.thin.gl.ac = {}
326 FBsp.thin.ch = {}
327 FBsp.thin.ch.fr = 0x202F ; FBsp.thin.ch.ac = nil
328 FBsp.colon = {}
329 FBsp.colon.gl = {}
330 FBsp.colon.gl.fr = { 1, 1, 1} ; FBsp.colon.gl.ac = {}
331 FBsp.colon.ch = {}
332 FBsp.colon.ch.fr = 0xA0 ; FBsp.colon.ch.ac = nil
333 FBsp.guill = {}
334 FBsp.guill.gl = {}
335 FBsp.guill.gl.fr = {.8, .3, .8} ; FBsp.guill.gl.ac = {}
336 FBsp.guill.ch = {}
337 FBsp.guill.ch.fr = 0xA0 ; FBsp.guill.ch.ac = nil

```

The next function converts the glue table returned by function `FBget_glue` into `sp` for the current font; beware of null values for `fid`, see `\nullfont` in TikZ, and of special fonts like `lcircle1.pfb` for which `font.getfont(fid)` does not return a proper font table, in such cases the function returns `nil`.

```

338 local font_table = {}
339 local function new_glue_scaled (fid,table)
340   if fid > 0 and table[1] then
341     local fp = font_table[fid]
342     if not fp then
343       local ft = font.getfont(fid)
344       if ft then
345         font_table[fid] = ft.parameters
346         fp = font_table[fid]
347       end
348     end
349     local gl = new_node(GLUE,0)
350     if fp then
351       node.setglue(gl, table[1]*fp.space,
352                    table[2]*fp.space_stretch,
353                    table[3]*fp.space_shrink)
354       return gl
355     else
356       return nil
357     end
358   else
359     return nil
360   end
361 end

```

Let's catch LuaTeX attributes `\FB@spacing`, `\FB@addDPspace` and `\FB@addGUILspace`.

```

362 local FBspacing = luatexbase.attributes['FB@spacing']
363 local addDPspace = luatexbase.attributes['FB@addDPspace']

```



```

364 local addGUILSpace = luatexbase.attributes['FB@addGUILSpace']
365 local FBucsNBSP    = luatexbase.attributes['FB@ucsNBSP']
366 local FBdialect    = luatexbase.attributes['FB@dialect']
367 local has_attribute = node.has_attribute

```

The following function will be added to kerning callback. It catches all nodes of type GLYPH in the list starting at head and checks the language attributes of the current glyph: nothing is done if the current language is not French and only specific punctuation characters (those for which FB_punct_left or FB_punct_right is true) need a special treatment. In French, local variables are defined to hold the properties of the current glyph (item) and of the previous one (prev) or the next one (next). Constants FR_fr (french) and FR_ca (acadian) are defined by command \activate@luatexpunct.

```

368 local function french_punctuation (head)
369   for item in node.traverse_id(GLYPH, head) do
370     local lang = item.lang
371     local char = item.char
372     local fid  = item.font
373     local FRspacing = has_attribute(item, FBspacing)
374     FRspacing = FRspacing and FRspacing > 0
375     local FRucsNBSP = has_attribute(item, FBucsNBSP)
376     FRucsNBSP = FRucsNBSP and FRucsNBSP > 0
377     local FRdialect = has_attribute(item, FBdialect)
378     FRdialect = FRdialect and FRdialect > 0
379     local SIG = has_attribute(item, addGUILSpace)
380     SIG = SIG and SIG > 0
381     if lang ~= FR_fr and lang ~= FR_ca then
382       FRspacing = nil
383     end
384     local nbspace = new_node("glyph")
385     if FRspacing and FB_punct_left[char] and fid > 0 then
386       local prev = item.prev
387       local prev_id, prev_subtype, prev_char
388       if prev then
389         prev_id = prev.id
390         prev_subtype = prev.subtype
391         if prev_id == GLYPH then
392           prev_char = prev.char
393         end
394       end

```

If the previous node is a glue, check its natural width, only positive glues (actually glues > 1 sp, for tabular 'l' columns) are to be replaced by a non-breaking space.

```

395       local is_glue = prev_id == GLUE
396       local glue_wd
397       if is_glue then
398         glue_wd = prev.width
399       end
400       local realglue = is_glue and glue_wd > 1

```

For characters for which FB_punct_thin or FB_punct_thick is *true*, the amount of spacing to be typeset before them is controlled by commands \FBthinspace

and `\FBcolonspace` respectively. Two options: if a space has been typed in before (turned into *glue* in the node list), we remove the *glue* and add a nobreak penalty and the required *glue*. Otherwise (auto option), the penalty and the required *glue* are inserted if attribute `\FB@addDPspace` is set, unless any of these four conditions is met: a) node is `'.'` and the next one is of type GLYPH (avoids spurious spaces in `http://mysite, C:\` or `10:35`); b) the previous character is part of type `FB_punct_null` (avoids spurious spaces in strings like `(!)` or `??`); c) a null glue (actually glues ≤ 1 sp for tabulars) precedes the punctuation character (for tabulars and listings); d) the punctuation character starts a paragraph or an `\hbox{}`.

When option `UnicodeNoBreakSpaces` is set to `true`, a Unicode character U+00A0 or U+202F is inserted instead of penalty and glue.

```

401     if FB_punct_thin[char] or FB_punct_thick[char] then
402         local SBDP = has_attribute(item, addDPspace)
403         local auto = SBDP and SBDP > 0
404         if FB_punct_thick[char] and auto then
405             local next = item.next
406             local next_id
407             if next then
408                 next_id = next.id
409             end
410             if next_id and next_id == GLYPH then
411                 auto = false
412             end
413         end
414         if auto then
415             if (prev_char and FB_punct_null[prev_char]) or
416                (is_glue and glue_wd <= 1) or
417                (prev_id == HLIST and prev_subtype == 3) or
418                (prev_id == TEMP) then
419                 auto = false
420             end
421         end
422         local fbglue
423         local t
424         if FB_punct_thick[char] then
425             if FRdialect then
426                 t = FBsp.colon.gl.ac
427                 nspace.char = FBsp.colon.ch.ac
428             else
429                 t = FBsp.colon.gl.fr
430                 nspace.char = FBsp.colon.ch.fr
431             end
432         else
433             if FRdialect then
434                 t = FBsp.thin.gl.ac
435                 nspace.char = FBsp.thin.ch.ac
436             else
437                 t = FBsp.thin.gl.fr
438                 nspace.char = FBsp.thin.ch.fr
439             end

```

```

440         end
441         fbglue = new_glue_scaled(fid, t)

```

In case `new_glue_scaled` fails (returns nil) the node list remains unchanged.

```

442         if (realglue or auto) and fbglue then
443             if realglue then
444                 head = remove_node(head,prev,true)
445             end
446             if (FRucsNBSP) then
447                 nbspace.font = fid
448                 insert_node_before(head, item, copy_node(nbspace))
449             else
450                 insert_node_before(head, item, copy_node(nobreak))
451                 insert_node_before(head, item, copy_node(fbglue))
452             end
453         end

```

Let's consider '»' now (the only remaining glyph of `FB_punct_left` class): we just have to remove any *glue* possibly preceding '»', then to insert the nobreak penalty and the proper *glue* (controlled by `\FBguillspace`). This is done only if French quotes have been 'activated' by options `og=«`, `fg=»` in `\frenchsetup{}` and can be denied locally with `\NoAutoSpacing` (this is controlled by the SIG flag). If either a) the preceding glyph is member of `FB_guill_null`, or b) '»' is the first glyph of an `\hbox{}` or a paragraph, nothing is done, this is controlled by the `addgl` flag.

```

454         elseif SIG then
455             local addgl = (prev_char and not FB_guill_null[prev_char]) or
456                 (not prev_char and
457                 prev_id ~= TEMP and
458                 not (prev_id == HLIST and prev_subtype == 3)
459                 )

```

Correction for tabular 'c' (glue 0 plus 1 fil) and 'l' (glue 1sp) columns:

```

460         if is_glue and glue_wd <= 1 then
461             addgl = false
462         end
463         local t = FBsp.guill.gl.fr
464         nbspace.char = FBsp.guill.ch.fr
465         if FRdialect then
466             t = FBsp.guill.gl.ac
467             nbspace.char = FBsp.guill.ch.ac
468         end
469         local fbglue = new_glue_scaled(fid, t)
470         if addgl and fbglue then
471             if is_glue then
472                 head = remove_node(head,prev,true)
473             end
474             if (FRucsNBSP) then
475                 nbspace.font = fid
476                 insert_node_before(head, item, copy_node(nbspace))
477             else
478                 insert_node_before(head, item, copy_node(nobreak))

```

```

479             insert_node_before(head, item, copy_node(fbg_lue))
480         end
481     end
482 end
483 end

```

Similarly, for '«' (unique member of the FB_punct_right class): unless either a) the next glyph is member of FB_guil_null, or b) '«' is the last glyph of an \hbox{} or a paragraph (then the addgl flag is false, nothing is done), we remove any *glue* possibly following it and insert first the proper *glue* then a nobreak penalty so that finally the penalty precedes the *glue*.

```

484     if FRspacing and FB_punct_right[char]
485         and fid > 0 and SIG then
486         local next = item.next
487         local next_id, next_subtype, next_char, nextnext, kern_wd
488         if next then
489             next_id = next.id
490             next_subtype = next.subtype
491             if next_id == GLYPH then
492                 next_char = next.char

```

A kern0 might hide a glue, so look ahead if next is a kern (this occurs with « \texttt{a} »):

```

493         elseif next_id == KERN then
494             kern_wd = next.kern
495             if kern_wd == 0 then
496                 nextnext = next.next
497                 if nextnext then
498                     next = nextnext
499                     next_id = nextnext.id
500                     next_subtype = nextnext.subtype
501                     if next_id == GLYPH then
502                         next_char = nextnext.char
503                     end
504                 end
505             end
506         end
507     end
508     local is_glue = next_id == GLUE
509     if is_glue then
510         glue_wd = next.width
511     end
512     local addgl = (next_char and not FB_guil_null[next_char]) or
513                 (next and not next_char)

```

Correction for tabular 'c' columns. For 'r' columns, a final '«' character needs to be coded as \mbox{«} for proper spacing (\NoAutoSpacing is another option).

```

514     if is_glue and glue_wd == 0 then
515         addgl = false
516     end
517     local fid = item.font

```

```

518     local t = FBsp.guill.gl.fr
519     nbspace.char = FBsp.guill.ch.fr
520     if FRdialect then
521         t = FBsp.guill.gl.ac
522         nbspace.char = FBsp.guill.ch.ac
523     end
524     local fbg glue = new_glue_scaled(fid, t)
525     if addgl and fbg glue then
526         if is_glue then
527             head = remove_node(head,next,true)
528         end
529         if (FRucsNBSP) then
530             nbspace.font = fid
531             insert_node_after(head, item, copy_node(nbspace))
532         else
533             insert_node_after(head, item, copy_node(fbg glue))
534             insert_node_after(head, item, copy_node(nobreak))
535         end
536     end
537 end
538 end
539 return head
540 end
541 return french_punctuation
542 </lua>

```

`\FB@luatex@punct@french` As a language tag is part of glyph nodes in LuaTeX, no more switching has to be done in `\extrasfrench`, setting the dialect attribute has already been done (see above, p. 19). We will just redefine `\shorthandoff` and `\shorthandon` in French to issue a warning reminding the user that active characters are no longer used in French with recent LuaTeX engines.

```

543 <*french>
544 \ifFB@luatex@punct
545   \newcommand*\FB@luatex@punct@french{%
546     \babel@save\shorthandon
547     \babel@save\shorthandoff
548     \def\shorthandoff##1{%
549       \ifx\PackageWarning\@undefined
550         \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
551           LuaTeX,\ use \noexpand\NoAutoSpacing
552           *inside a group* instead.}%
553       \else
554         \PackageWarning{french.lda}{\protect\shorthandoff{;:!?} is
555           helpless with LuaTeX,\MessageBreak use \protect\NoAutoSpacing
556           \space *inside a group* instead;\MessageBreak reported}%
557       \fi}%
558     \def\shorthandon##1{%
559   }
560   \addto\extrasfrench{\FB@luatex@punct@french}

```

The next definition will be used to activate Lua punctuation: it loads `frenchb.lua` and adds function `french_punctuation` at the end of the kerning callback (no priority).

```

561 \def\activate@luatexpunct{%
562   \directlua{%
563     FR_fr = \the\l@french ; FR_ca = \the\l@acadian ;
564     local path = kpse.find_file("frenchb.lua", "lua")
565     if path then
566       local f = dofile(path)
567       luatexbase.add_to_callback("kerning",
568         f, "frenchb.french_punctuation")
569     else
570       texio.write_nl('')
571       texio.write_nl('*****')
572       texio.write_nl('Error: frenchb.lua not found.')
573       texio.write_nl('*****')
574       texio.write_nl('')
575     end
576   }%
577 }
578 \fi

```

End of specific code for punctuation with LuaTeX engines.

2.2.2 Punctuation with XeTeX

If `\XeTeXinterchartokenstate` is available, we use the “inter char” mechanism to provide correct spacing in French before the four characters `;` `!` `?` and `:`. The basis of the following code was borrowed from the `polyglossia` package, see `gloss-french.ldf`. We use the same mechanism for French quotes (`«` and `»`), when automatic spacing for quotes is required by options `og=«` and `fg=»` in `\frenchsetup{}` (see section 2.11).

The default value for `\XeTeXcharclass` is 0 for characters tokens and `\FB@nonchar` for all other tokens (glues, kerns, math and box boundaries, etc.). These defaults should not be changed otherwise the spacing before the ‘high punctuation’ characters and inside quotes might not be correct.

We switch `\XeTeXinterchartokenstate` to 1 and change the `\XeTeXcharclass` values of `;` `!` `?` `:` `(` `]` `«` and `»` when entering French. Special care is taken to restore them to their initial values when leaving French.

The following part holds specific code for punctuation with XeTeX engines.

```

579 \ifFB@xetex@punct
580   \ifLaTeXe
581     \PackageInfo{french.ldf}{No need for active punctuation characters%
582       \MessageBreak with this version of XeTeX!%
583       \MessageBreak reported}
584   \else
585     \fb@info{No need for active punctuation characters\
586       with this version of XeTeX!}
587   \fi

```

Six new character classes are defined for `babel-french`.

```

588 \newXeTeXintercharclass\FB@punctthick
589 \newXeTeXintercharclass\FB@punctthin
590 \newXeTeXintercharclass\FB@punctnul
591 \newXeTeXintercharclass\FB@guilo
592 \newXeTeXintercharclass\FB@guilf
593 \newXeTeXintercharclass\FB@guilnul

```

As `\babel@savevariable` doesn't work inside a `\bbl@for` loop, we define a variant to save the `\XeTeXcharclass` values which will be modified in French.

```

594 \def\FBsavevariable@loop#1#2{\begingroup
595   \toks@\expandafter{\originalTeX #1}%
596   \edef\x{\endgroup
597     \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
598   \x}

```

`\FB@charlist` holds the all list of characters which have their `\XeTeXcharclass` value modified in French: the first set includes high punctuation, French quotes, opening delimiters and no-break spaces

"21	"3A	"3B	"3F	"AB	"BB	"28	"5B	"A0	"202F
!	:	;	?	«	»	([

the second one holds those which need resetting in French when `xeCJK.sty` is in use

"29	"5D	"7B	"7D	"2C	"2D	"2E	"22	"25	"27	"60	"2019
)]	{	}	,	-	.	"	%	'	'	'

```

599 \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F,%
600                   "29,"5D,"7B,"7D,"2C,"2D,"2E,"22,"25,"27,"60,"2019}

```

`\FB@xetex@punct@french` The following command will be executed when entering French, it first saves the values to be modified, then fits them to our needs. It also redefines `\shorthandoff` and `\shorthandon` (locally) to avoid error messages with XeTeX-based engines.

```

601 \newcommand*\FB@xetex@punct@french{%
602   \babel@savevariable{\XeTeXinterchartokenstate}%
603   \babel@save{\shorthandon}%
604   \babel@save{\shorthandoff}%
605   \bbl@for\FB@char\FB@charlist
606     {\FBsavevariable@loop{\XeTeXcharclass}{\FB@char}}%
607   \def\shorthandoff##1{%
608     \ifx\PackageWarning\@undefined
609       \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
610         XeTeX,\ use \noexpand\NoAutoSpacing
611         *inside a group* instead.}%
612     \else
613       \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?} is
614         helpless with XeTeX,\MessageBreak use \protect\NoAutoSpacing
615         \space *inside a group* instead;\MessageBreak reported}%
616     \fi}%
617   \def\shorthandon##1{%

```

Let's now set the classes and interactions between classes. When false, the flag `\ifFB@spacing` switches off any interaction between classes (this flag is controlled by

user-level command `\NoAutoSpacing`; this flag is also set to false when the current font is a typewriter font).

```

618 \XeTeXinterchartokenstate=1
619 \XeTeXcharclass '\: = \FB@punctthick
620 \XeTeXinterchartoks \z@ \FB@punctthick = {%
621 \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
622 \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
623 \ifFB@spacing\FDP@colonspace\fi}%

```

Small glues such as “glue 1sp” in tabular ‘l’ columns or “glue 0 plus 1 fil” in tabular ‘c’ columns or `lstlisting` environment should not trigger any extra space; they will still do when `AutoSpacePunctuation` is true: unfortunately `\XeTeXcharclass=\FB@nonchar` isn’t specific to glue tokens (this class includes box and math boundaries f.i.), so the `\else` part cannot be omitted.

```

624 \XeTeXinterchartoks \FB@nonchar \FB@punctthick = {%
625 \ifFB@spacing
626 \ifhmode
627 \ifdim\lastskip>1sp
628 \unskip\penalty\@M\FBcolonspace
629 \else
630 \FDP@colonspace
631 \fi
632 \fi
633 \fi}%
634 \bbl@for\FB@char
635 {\;,\;!,\?}%
636 {\XeTeXcharclass\FB@char=\FB@punctthin}%
637 \XeTeXinterchartoks \z@ \FB@punctthin = {%
638 \ifFB@spacing\ifhmode\FDP@thinspace\fi\fi}%
639 \XeTeXinterchartoks \FB@guilf \FB@punctthin = {%
640 \ifFB@spacing\FDP@thinspace\fi}%
641 \XeTeXinterchartoks \FB@nonchar \FB@punctthin = {%
642 \ifFB@spacing
643 \ifhmode
644 \ifdim\lastskip>1sp
645 \unskip\penalty\@M\FBthinspace
646 \else
647 \FDP@thinspace
648 \fi
649 \fi
650 \fi}%
651 \XeTeXinterchartoks \FB@guilo \z@ = {%
652 \ifFB@spacing\FB@guillspace\fi}%
653 \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
654 \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
655 \XeTeXinterchartoks \z@ \FB@guilf = {%
656 \ifFB@spacing\FB@guillspace\fi}%
657 \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
658 \ifFB@spacing\FB@guillspace\fi}%
659 \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
660 \ifFB@spacing\unskip\FB@guillspace\fi}%

```


This will avoid spurious spaces in (!), [?] and with Unicode non-breaking spaces (U+00A0, U+202F):

```
661 \bbl@for\FB@char
662     {'\[, '\(, "A0, "202F}%
663     {\XeTeXcharclass\FB@char=\FB@punctnul}%
```

These characters have their class changed by `xeCJK.sty`, let's reset them to 0 in French.

```
664 \bbl@for\FB@char
665     {'\{, '\., '\., '\-, '\), '\], '\}, '\%, "22, "27, "60, "2019}%
666     {\XeTeXcharclass\FB@char=\z@}%
667 }
668 \addto\extrasfrench{\FB@xetex@punct@french}
```

End of specific code for punctuation with modern XeTeX engines.

```
669 \fi
```

2.2.3 Punctuation with standard (pdf)TeX

In standard (pdf)TeX we need to make the four characters ; ! ? and : 'active' and provide their definitions.

```
670 \ifFB@active@punct
671 \initiate@active@char{:}%
672 \initiate@active@char{;}%
673 \initiate@active@char{!}%
674 \initiate@active@char{?}%
```

We first tune the amount of space before ; ! ? and :. This should only happen in horizontal mode, hence the test `\ifhmode`.

In horizontal mode, if a space has been typed before ';' we remove it and put a non-breaking `\FBthinspace` instead. If no space has been typed, we add `\FDP@thinspace` which will be defined, up to the user's wishes, as a non-breaking `\FBthinspace` or as `\@empty`.

```
675 \declare@shorthand{french}{;}{;%
676 \ifFB@spacing
677 \ifhmode
678 \ifdim\lastskip>1sp
679 \unskip\penalty\M\FBthinspace
680 \else
681 \FDP@thinspace
682 \fi
683 \fi
684 \fi
```

Now we can insert a ; character.

```
685 \string;}
```

The next three definitions are very similar.

```
686 \declare@shorthand{french}{!}{;%
687 \ifFB@spacing
688 \ifhmode
```

```

689     \ifdim\lastskip>1sp
690     \unskip\penalty\@M\FBthinspace
691     \else
692     \FDP@thinspace
693     \fi
694 \fi
695 \fi
696 \string!}
697 \declare@shorthand{french}{?}{%
698 \ifFB@spacing
699 \ifhmode
700 \ifdim\lastskip>1sp
701 \unskip\penalty\@M\FBthinspace
702 \else
703 \FDP@thinspace
704 \fi
705 \fi
706 \fi
707 \string?}
708 \declare@shorthand{french}{:}{%
709 \ifFB@spacing
710 \ifhmode
711 \ifdim\lastskip>1sp
712 \unskip\penalty\@M\FBcolonspace
713 \else
714 \FDP@colonspace
715 \fi
716 \fi
717 \fi
718 \string:}

```

When the active characters appear in an environment where their French behaviour is not wanted they should give an ‘expected’ result. Therefore we define shorthands at system level as well.

```

719 \declare@shorthand{system}{:}{\string:}
720 \declare@shorthand{system}{!}{\string!}
721 \declare@shorthand{system}{?}{\string?}
722 \declare@shorthand{system}{;}{\string;}
723 %}

```

We specify that the French group of shorthands should be used when switching to French.

```

724 \addto\extrasfrench{\languageshorthands{french}%

```

These characters are ‘turned on’ once, later their definition may vary. Don’t misunderstand the following code: they keep being active all along the document, even when leaving French.

```

725 \bbl@activate{:}\bbl@activate{;}%
726 \bbl@activate{!}\bbl@activate{?}%
727 }
728 \addto\noextrasfrench{%
729 \bbl@deactivate{:}\bbl@deactivate{;}%

```

```

730 \bbl@deactivate{!}\bbl@deactivate{?}%
731 }
732 \fi

```

2.2.4 Punctuation switches common to all engines

A new ‘if’ `\ifFBAutoSpacePunctuation` needs to be defined now to control the two possible ways of dealing with ‘high punctuation’. its default value is true, but it can be set to false by `\frenchsetup{AutoSpacePunctuation=false}` for finer control.

```
733 \newif\ifFBAutoSpacePunctuation \FBAutoSpacePunctuationtrue
```

`\AutoSpaceBeforeFDP` `\autospace@beforeFDP` and `\noautospace@beforeFDP` are internal commands. `\NoAutoSpaceBeforeFDP` `\autospace@beforeFDP` defines `\FDP@thinspace` and `\FDP@colonspace` as non-breaking spaces and sets LuaTeX attribute `\FB@addDPspace` to 1 (true), while `\noautospace@beforeFDP` lets these spaces empty and sets flag `\FB@addDPspace` to 0 (false). User commands `\AutoSpaceBeforeFDP` and `\NoAutoSpaceBeforeFDP` do the same and take care of the flag `\ifFBAutoSpacePunctuation` in \LaTeX . Set the default now for Plain (done later for LaTeX).

```

734 \def\autospace@beforeFDP{%
735   \ifFB@luatex@punct\FB@addDPspace=1 \fi
736   \def\FDP@thinspace{\penalty\@M\FBthinspace}%
737   \def\FDP@colonspace{\penalty\@M\FBcolonspace}}
738 \def\noautospace@beforeFDP{%
739   \ifFB@luatex@punct\FB@addDPspace=0 \fi
740   \let\FDP@thinspace\@empty
741   \let\FDP@colonspace\@empty}
742 \ifLaTeXe
743   \def\AutoSpaceBeforeFDP{\autospace@beforeFDP
744     \FBAutoSpacePunctuationtrue}
745   \def\NoAutoSpaceBeforeFDP{\noautospace@beforeFDP
746     \FBAutoSpacePunctuationfalse}
747   \AtEndOfPackage{\AutoSpaceBeforeFDP}
748 \else
749   \let\AutoSpaceBeforeFDP\autospace@beforeFDP
750   \let\NoAutoSpaceBeforeFDP\noautospace@beforeFDP
751   \AutoSpaceBeforeFDP
752 \fi

```

`\rmfamilyFB` In LaTeX2e `\ttfamily` (and hence `\texttt`) will be redefined ‘AtBeginDocument’ `\sffamilyFB` as `\ttfamilyFB` so that no space is added before the four ; : ! ? characters, `\ttfamilyFB` even if `AutoSpacePunctuation` is true. When `AutoSpacePunctuation` is false, the eventually typed spaces are left unchanged (not turned into thin spaces, no penalty added). `\rmfamily` and `\sffamily` need to be redefined also (`\ttfamily` is not always used inside a group, its effect can be cancelled by `\rmfamily` or `\sffamily`). These redefinitions can be canceled if necessary, for instance to recompile older documents, see option `OriginalTypewriter` below.

To be consistent with what is done for the ; : ! ? characters, `\ttfamilyFB` also switches off insertion of spaces inside French guillemets *when they are typed in as*

characters with the ‘og’/‘fg’ options in `\frenchsetup{}`. This is also a workaround for the weird behaviour of these characters in verbatim mode.

```
753 \ifLaTeXe
754 \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
755 \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on \rmfamilyORI}
756 \DeclareRobustCommand\sffamilyFB{\FB@spacing@on \sffamilyORI}
757 \fi
```

`\NoAutoSpacing` The following command disables automatic spacing for high punctuation and French quote characters; it also switches off active punctuation characters (if any). It is engine independent (works for TeX, LuaTeX and XeTeX based engines) and is meant to be used inside a group.

```
758 \DeclareRobustCommand*\NoAutoSpacing}{%
759 \FB@spacing@off
760 \ifFB@active@punct\shorthandoff{;:!?}\fi
761 }
```

2.3 Commands for French quotation marks

`\guillemotleft` pdfLaTeX users are supposed to use 8-bit output encodings (T1, LY1, ...) to typeset French, those who still stick to OT1 should load `aeguill` or a similar package. In both cases the commands `\guillemotleft` and `\guillemotright` will print the French opening and closing quote characters from the output font. For XeLaTeX and LuaLaTeX, `\guillemotleft` and `\guillemotright` are defined by package `fontspec` (v. 2.5d and up).

We provide the following definitions for non-LaTeX users only as fall-back, they are welcome to change them for anything better.

```
762 \ifLaTeXe
763 \else
764 \ifFBunicode
765 \def\guillemotleft{{\char"00AB}}
766 \def\guillemotright{{\char"00BB}}
767 \def\textquotedblleft{{\char"201C}}
768 \def\textquotedblright{{\char"201D}}
769 \else
770 \def\guillemotleft{\leavevmode\raise0.25ex
771 \hbox{\scriptscriptstyle\ll$}}
772 \def\guillemotright{\raise0.25ex
773 \hbox{\scriptscriptstyle\gg$}}
774 \def\textquotedblleft{' '}
775 \def\textquotedblright{' '}
776 \fi
777 \let\xspace\relax
778 \fi
```

`\FBgspchar` The next step is to provide correct spacing after ‘<<’ and before ‘>>’; no line break is allowed neither *after* the opening one, nor *before* the closing one. French quotes `\FB@og` `\FB@fg`

(including spacing) are printed by `\FB@og` and `\FB@fg`, the expansion of the top level commands `\og` and `\og` is different in and outside French.

The definitions of `\FB@og` and `\FB@fg` need some engine-dependent tuning: for LuaTeX, `\FB@spacing` is set to 0 locally to prevent the quotes characters from adding space when option `og=«, fg=»` is set.

```
779 \newcommand*{\FB@guillspace}{\penalty\@M\FBguillspace}
780 \newcommand*{\FBgspchar}{\char"A0\relax}
781 \newif\ifFBucsNBSP
782 \ifFB@luatex@punct
783   \DeclareRobustCommand*{\FB@og}{\leavevmode
784     \bgroup\FB@spacing=0 \guillemotleft\egroup
785     \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi}
786 \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>\z@\unskip\fi
787   \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi
788   \bgroup\FB@spacing=0 \guillemotright\egroup}
789 \fi
```

With XeTeX, `\ifFB@spacing` is set to false locally for the same reason.

```
790 \ifFB@xetex@punct
791   \DeclareRobustCommand*{\FB@og}{\leavevmode
792     \bgroup\FB@spacingfalse\guillemotleft\egroup
793     \FB@guillspace}
794 \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>\z@\unskip\fi
795   \FB@guillspace
796   \bgroup\FB@spacingfalse\guillemotright\egroup}
797 \fi
798 \ifFB@active@punct
799   \DeclareRobustCommand*{\FB@og}{\leavevmode
800     \guillemotleft
801     \FB@guillspace}
802 \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>\z@\unskip\fi
803   \FB@guillspace
804   \guillemotright}
805 \fi
```

`\og` The user level macros for quotation marks are named `\og` (“ouvrez guillemets”) and `\fg` (“fermez guillemets”). Another option for typesetting quotes in French is to use the command `\frquote` (see below). Dummy definition of `\og` and `\fg` just to ensure that this commands are not yet defined.

```
806 \newcommand*{\og}{\@empty}
807 \newcommand*{\fg}{\@empty}
```

The definitions of `\og` and `\fg` for quotation marks are switched on and off through the `\extrasfrench \noextrasfrench` mechanism. Outside French, `\og` and `\fg` will typeset standard English opening and closing double quotes. We’ll try to be smart to users of David Carlisle’s `xspace` package: if this package is loaded there will be no need for `}` or `\` to get a space after `\fg`, otherwise `\xspace` will be defined as `\relax` (done at the end of this file).

```
808 \ifLaTeXe
809   \def\bbl@frenchguillemets{%
```

```

810     \renewcommand*{\og}{\FB@og}%
811     \renewcommand*{\fg}{\FB@fg\xspace}}
812 \renewcommand*{\og}{\textquotedblleft}
813 \renewcommand*{\fg}{\ifdim\lastskip>\z@\unskip\fi
814     \textquotedblright\xspace}
815 \else
816 \def\bbf@frenchguillemets{\let\og\FB@og
817     \let\fg\FB@fg}
818 \def\og{\textquotedblleft}
819 \def\fg{\ifdim\lastskip>\z@\unskip\fi\textquotedblright}
820 \fi

821 \addto\extrasfrench{\babel@save\og \babel@save\fg \bbf@frenchguillemets}

```

`\frquote` Another way of entering French quotes relies on `\frquote{}` with supports up to two levels of quotes. Let's define the default quote characters to be used for level one or two of quotes. . .

```

822 \newcommand*{\ogi}{\FB@og}
823 \newcommand*{\fgi}{\FB@fg}
824 \newcommand*{\@ogi}{\ifmmode\hbox{\ogi}\else\ogi\fi}
825 \newcommand*{\@fgi}{\ifmmode\hbox{\fgi}\else\fgi\fi}
826 \newcommand*{\ogii}{\textquotedblleft}
827 \newcommand*{\fgii}{\textquotedblright}
828 \newcommand*{\@ogii}{\ifmmode\hbox{\ogii}\else\ogii\fi}
829 \newcommand*{\@fgii}{\ifmmode\hbox{\fgii}\else\fgii\fi}

```

and the needed technical stuff to handle options:

```

830 \newcount\FBguill@level
831 \newtoks\FBbold@everypar

```

`\FB@addquote@everypar` was borrowed from `csquotes.sty`.

```

832 \def\FB@addquote@everypar{%
833 \let\FBnew@everypar\everypar
834 \FBbold@everypar=\expandafter{\the\everypar}%
835 \FBnew@everypar={\the\FBbold@everypar\FBeverypar@quote}%
836 \let\everypar\FBbold@everypar
837 \let\FB@addquote@everypar\relax
838 }
839 \newif\ifFBcloseguill \FBcloseguilltrue
840 \newif\ifFBinnerGuillSingle
841 \def\FBguilllopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
842 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
843 \let\FBguillnone\empty
844 \let\FBeveryparguill\FBguilllopen
845 \let\FBverylineguill\FBguillnone
846 \let\FBeverypar@quote\relax
847 \let\FBveryline@quote\empty

```

The main command `\frquote` accepts (in LaTeX2e only) a starred version which suppresses the closing quote; it is meant to be used for inner quotations which end

together with the outer one, then only one closing guillemet (the outer one) should be printed.

```

848 \ifLaTeXe
849 \DeclareRobustCommand\frquote{%
850     \@ifstar{\FBcloseguillfalse\fr@quote}%
851             {\FBcloseguilltrue\fr@quote}}
852 \else
853 \newcommand\frquote[1]{\fr@quote{#1}}
854 \fi

```

The internal command `\fr@quote` takes one (long) argument: the quotation text.

```

855 \newcommand{\fr@quote}[1]{%
856 \leavevmode
857 \advance\FBguill@level by \@ne
858 \ifcase\FBguill@level
859 \or

```

This for level 1 (outer) quotations: set `\FBeverypar@quote` for level 1 quotations and add it to `\everypar` using `\FB@addquote@everypar`, then print the quotation:

```

860 \ifx\FBeveryparguill\FBguillnone
861 \else
862 \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
863 \FB@addquote@everypar
864 \fi
865 \@ogi #1\@fgi
866 \or

```

This for level 2 (inner) quotations: Omega's command `\lcalleftbox` included in LuaTeX, is convenient for repeating guillemets at the beginning of every line.

```

867 \ifx\FBeverylineguill\FBguillopen
868 \def\FBeveryline@quote{\FB@addGUllspace=0 \guillemotleft
869 \FB@guillspace}%
870 \lcalleftbox{\FBeveryline@quote}%
871 \let\FBeverypar@quote\relax
872 \@ogi #1\ifFBcloseguill\@fgi\fi
873 \else
874 \ifx\FBeverylineguill\FBguillclose
875 \def\FBeveryline@quote{\FB@addGUllspace=0 \guillemotright
876 \FB@guillspace}%
877 \lcalleftbox{\FBeveryline@quote}%
878 \let\FBeverypar@quote\relax
879 \@ogi #1\ifFBcloseguill\@fgi\fi
880 \else

```

otherwise we need to redefine `\FBeverypar@quote` (and eventually `\logii`, `\fgii`) for level 2 quotations:

```

881 \let\FBeverypar@quote\relax
882 \ifFBInnerGuillSingle
883 \def\logii{\leavevmode
884 \guillsinglleft\FB@guillspace}%
885 \def\fgii{\ifdim\lastskip>\z@\unskip\fi
886 \FB@guillspace\guillsinglright}%

```

```

887     \ifx\FBeveryparguill\FBguillopen
888     \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
889     \fi
890     \ifx\FBeveryparguill\FBguillclose
891     \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
892     \fi
893     \fi
894     \@ogii #1\ifFBcloseguill \@fgii \fi
895     \fi
896     \fi
897     \else

```

Warn if \FBguill@level > 2:

```

898     \ifx\PackageWarning\@undefined
899     \fb@warning{\noexpand\frquote\space handles up to
900         two levels.\\ Quotation not printed.}%
901     \else
902     \PackageWarning{french.ldf}{%
903         \protect\frquote\space handles up to two levels.
904         \MessageBreak Quotation not printed. Reported}
905     \fi
906     \fi

```

Closing: step down \FBguill@level and clean on exit. Changes made global in case \frquote{} ends inside an environment.

```

907     \global\advance\FBguill@level by \m@ne
908     \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
909     \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
910     \global\let\FBeverylin@quote\empty
911     \ifx\FBeverylin@quote\FBguillnone\else\localleftbox{}\fi
912     \fi
913 }

```

The next command is intended to be used in list environments to suppress quotes which might be added by \FBeverypar@quote after items for instance.

```

914 \newcommand*{\NoEveryParQuote}{\let\FBeveryparguill\FBguillnone}

```

2.4 Date in French

`\frenchtoday` The following code creates a macro `\datefrench` which in turn defines command `\frenchdate` (`\today` is defined as `\frenchtoday` in French). The corresponding commands for the French dialect, `\dateacadian` and `\acadiantoday` are also created btw. This new implementation relies on commands `\SetString` and `\SetStringLoop`, therefore requires babel 3.10 or newer.

Explicitly defining `\BabelLanguages` as the list of all French dialects defines *both* `\datefrench` and `\dateacadian`; this is required as `french.ldf` is read only once even if both language options `french` and `acadian` are supplied to `babel`. Note that coding `\StartBabelCommands*{french,acadian}` would *only* define `\csname date\CurrentOption\endcsname`, leaving the second language undefined in `babel`'s sens.

```

915 \def\BabelLanguages{french,acadian}

```



```

916 \StartBabelCommands*\BabelLanguages}{date}
917     [unicode, fontenc=TU EU1 EU2, charset=utf8]
918     \SetString\monthiiname{février}
919     \SetString\monthviiiname{août}
920     \SetString\monthxiiname{décembre}
921 \StartBabelCommands*\BabelLanguages}{date}
922     \SetStringLoop{month#1name}{%
923         janvier,f\`evrier,mars,avril,mai,juin,juillet,%
924         ao\^ut,septembre,octobre,novembre,d\`ecembre}
925     \SetString\today{\FB@date{\year}{\month}{\day}}
926 \EndBabelCommands

```

`\frenchdate` (which produces an unbreakable string) and `\frenchtoday` (breakable) both rely on `\FB@date`, the inner group is needed for `\hbox`.

```

927 \newcommand*\FB@date}[3]{%
928     {\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
929     \csname month\romannumeral#2name\endcsname
930     \ifx#1\@empty\else\FBdatespace\number#1\fi}}
931 \newcommand*\FBdatebox{\hbox}
932 \newcommand*\FBdatespace{\space}
933 \newcommand*\frenchdate{\FBdatebox\FB@date}
934 \newcommand*\acadiandate{\FBdatebox\FB@date}

```

2.5 Extra utilities

Let's provide the French user with some extra utilities.

`\up` `\up` eases the typesetting of superscripts like '1^{er}'. Up to version 2.0 of `babel-french` `\up` was just a shortcut for `\textsuperscript` in LaTeX2e, but several users complained that `\textsuperscript` typesets superscripts too high and too big, so we now define `\fup` as an attempt to produce better looking superscripts. `\up` is defined as `\fup` but `\frenchsetup{FrenchSuperscripts=false}` redefines `\up` as `\textsuperscript` for compatibility with previous versions.

When a font has built-in superscripts, the best thing to do is to just use them, otherwise `\fup` has to simulate superscripts by scaling and raising ordinary letters. Scaling is done using package `scalegnt` which will be loaded at the end of `babel`'s loading (`babel-french` being an option of `babel`, it cannot load a package while being read).

```

935 \newif\ifFB@poorman
936 \newdimen\FB@Mht
937 \ifLaTeXe
938     \AtEndOfPackage{\RequirePackage{scalegnt}}

```

`\FB@up@fake` holds the definition of fake superscripts. The scaling ratio is 0.65, raising is computed to put the top of lower case letters (like 'm') just under the top of upper case letters (like 'M'), precisely 12% down. The chosen settings look correct for most fonts, but can be tuned by the end-user if necessary by changing `\FBsupR` and `\FBsupS` commands.

`\FB@lc` is defined as `\MakeLowercase` to inhibit the uppercasing of superscripts (this may happen in page headers with the standard classes but is wrong);

`\FB@lc` can be redefined to do nothing by option `LowercaseSuperscripts=false` of `\frenchsetup{}`.

```

939 \newcommand*\FBsupR}{-0.12}
940 \newcommand*\FBsupS}{0.65}
941 \newcommand*\FB@lc}[1]{\MakeLowercase{#1}}
942 \DeclareRobustCommand*\FB@up@fake}[1]{%
943   \settoheight{\FB@Mht}{M}%
944   \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
945   \addtolength{\FB@Mht}{-\FBsupS ex}%
946   \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
947 }

```

The only packages I currently know to take advantage of real superscripts are a) `realscripts` used in conjunction with XeLaTeX or LuaLaTeX and OpenType fonts having the font feature ‘VerticalPosition=Superior’ and b) `fourier` (from version 1.6) when Expert Utopia fonts are available.

`\FB@up` checks whether the current font is a Type1 ‘Expert’ (or ‘Pro’) font with real superscripts or not (the code works currently only with `fourier-1.6` but could work with any Expert Type1 font with built-in superscripts, see below), and decides to use real or fake superscripts. It works as follows: the content of `\f@family` (family name of the current font) is split by `\FB@split` into two pieces, the first three characters (‘fut’ for Fourier, ‘ppl’ for Adobe’s Palatino, ...) stored in `\FB@firstthree` and the rest stored in `\FB@suffix` which is expected to be ‘x’ or ‘j’ for expert fonts.

```

948 \def\FB@split#1#2#3#4@nil{\def\FB@firstthree{#1#2#3}%
949   \def\FB@suffix{#4}}
950 \def\FB@x{x}
951 \def\FB@j{j}
952 \DeclareRobustCommand*\FB@up}[1]{%
953   \bgroup \FB@poormantrue
954   \expandafter\FB@split\f@family\nil

```

Then `\FB@up` looks for a .fd file named `t1fut-sup.fd` (Fourier) or `t1ppl-sup.fd` (Palatino), etc. supposed to define the subfamily (fut-sup or ppl-sup, etc.) giving access to the built-in superscripts. If the .fd file is not found by `\IfFileExists`, `\FB@up` falls back on fake superscripts, otherwise `\FB@suffix` is checked to decide whether to use fake or real superscripts.

```

955   \edef\reserved@a{\lowercase{%
956     \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
957   \reserved@a
958   {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
959     \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
960     \ifFB@poorman \FB@up@fake{#1}%
961     \else \FB@up@real{#1}%
962     \fi}%
963   {\FB@up@fake{#1}}%
964   \egroup}

```

`\FB@up@real` just picks up the superscripts from the subfamily (and forces lowercase).

```

965 \newcommand*\FB@up@real}[1]{\bgroup
966   \fontfamily{\FB@firstthree -sup}\selectfont \FB@lc{#1}\egroup}

```

`\fup` is defined as `\FB@up` unless `\realsuperscript` is defined by `realscripts.sty`.

```
967 \DeclareRobustCommand*\fup}[1]{%
968   \ifx\realsuperscript\@undefined
969     \FB@up{#1}%
970   \else
971     \bgroup\let\fakesuperscript\FB@up@fake
972     \realsuperscript{\FB@lc{#1}}\egroup
973   \fi}
```

Let's provide a temporary definition for `\up` (redefined 'AtBeginDocument' as `\fup` or `\textsuperscript` according to `\frenchsetup{}` options).

```
974 \providecommand*\up}{\relax}
```

Poor man's definition of `\up` for Plain.

```
975 \else
976 \providecommand*\up}[1]{\leavevmode\raise1ex\hbox{\sevenrm #1}}
977 \fi
```

`\ieme` Some handy macros for those who don't know how to abbreviate ordinals:

```
\ier 978 \def\ieme{\up{e}\xspace}
\iere 979 \def\iemes{\up{es}\xspace}
\iemes 980 \def\ier{\up{er}\xspace}
\iers 981 \def\iers{\up{ers}\xspace}
\ieres 982 \def\iere{\up{re}\xspace}
983 \def\ieres{\up{res}\xspace}
```

`\FBmedkern`
`\FBthickkern`

```
984 \newcommand*\FBmedkern{\kern+.2em}
985 \newcommand*\FBthickkern{\kern+.3em}
```

`\No` And some more macros relying on `\up` for numbering, first two support macros.

```
\no 986 \newcommand*\FrenchEnumerate}[1]{#1\up{o}\FBthickkern}
\nos 987 \newcommand*\FrenchPopularEnumerate}[1]{#1\up{o})\FBthickkern}
\nos Typing \primo should result in '°',
\primo 988 \def\primo{\FrenchEnumerate1}
\fprimo) 989 \def\secundo{\FrenchEnumerate2}
990 \def\tertio{\FrenchEnumerate3}
991 \def\quarto{\FrenchEnumerate4}
```

while typing `\fprimo)` gives '°'.

```
992 \def\fprimo){\FrenchPopularEnumerate1}
993 \def\fsecundo){\FrenchPopularEnumerate2}
994 \def\ftertio){\FrenchPopularEnumerate3}
995 \def\fquarto){\FrenchPopularEnumerate4}
```

Let's provide four macros for the common abbreviations of "Numéro".

```
996 \DeclareRobustCommand*\No}{N\up{o}\FBmedkern}
997 \DeclareRobustCommand*\no}{n\up{o}\FBmedkern}
998 \DeclareRobustCommand*\Nos}{N\up{os}\FBmedkern}
999 \DeclareRobustCommand*\nos}{n\up{os}\FBmedkern}
```

`\bsc` As family names should be written in small capitals and never be hyphenated, we provide a command (its name comes from Boxed Small Caps) to input them easily. Note that this command has changed with version 2 of babel-french: a `\kern0pt` is used instead of `\hbox` because `\hbox` would break microtype's font expansion; as a (positive?) side effect, composed names (such as Dupont-Durand) can now be hyphenated on explicit hyphens. Usage: Jean~\bsc{Duchemin}.

```
1000 \DeclareRobustCommand*\bsc}[1]{\leavevmode\begingroup\kern0pt
1001                                     \scshape #1\endgroup}
1002 \ifLaTeXe\else\let\scshape\relax\fi
```

Some definitions for special characters. We won't define `\tilde` as a Text Symbol not to conflict with the macro `\tilde` for math mode and use the name `\tild` instead. Note that `\boi` may *not* be used in math mode, its name in math mode is `\backslash`. `\degre` can be accessed by the command `\r{}` for ring accent.

```
1003 \ifBUnicode
1004   \newcommand*\at{{\char"0040}}
1005   \newcommand*\circonflexe{{\char"005E}}
1006   \newcommand*\tild{{\char"007E}}
1007   \newcommand*\boi{{\char"005C}}
1008   \newcommand*\degre{{\char"00B0}}
1009 \else
1010   \ifLaTeXe
1011     \DeclareTextSymbol{\at}{T1}{64}
1012     \DeclareTextSymbol{\circonflexe}{T1}{94}
1013     \DeclareTextSymbol{\tild}{T1}{126}
1014     \DeclareTextSymbolDefault{\at}{T1}
1015     \DeclareTextSymbolDefault{\circonflexe}{T1}
1016     \DeclareTextSymbolDefault{\tild}{T1}
1017     \DeclareRobustCommand*\boi{\textbackslash}
1018     \DeclareRobustCommand*\degre{\r{}}
1019   \else
1020     \def\T@one{T1}
1021     \ifx\fontencoding\T@one
1022       \newcommand*\degre{{\char6}}
1023     \else
1024       \newcommand*\degre{{\char23}}
1025     \fi
1026     \newcommand*\at{{\char64}}
1027     \newcommand*\circonflexe{{\char94}}
1028     \newcommand*\tild{{\char126}}
1029     \newcommand*\boi{{\backslash}}
1030   \fi
1031 \fi
```

`\degres` We now define a macro `\degres` for typesetting the abbreviation for 'degrees' (as in 'degrees Celsius'). As the bounding box of the character 'degree' has very different widths in CM/EC and PostScript fonts, we fix the width of the bounding box of `\degres` to 0.3 em, this lets the symbol 'degree' stick to the preceding (e.g., 45\degres) or following character (e.g., 20~\degres C).

If T_EX Companion fonts are available (textcomp.sty), we pick up \textdegree from them instead of emulating ‘degrees’ from the \r{} accent. Otherwise we advise the user (once only) to use TS1-encoding.

```

1032 \ifLaTeXe
1033   \newcommand*{\degres}{\degre}
1034   \ifFBunicode
1035     \DeclareRobustCommand*{\degres}{\degre}
1036   \else
1037     \def\Warning@degree@TSone{\FBWarning
1038       {Degrees would look better in TS1-encoding:%
1039         \MessageBreak add \protect
1040         \usepackage{textcomp} to the preamble.%
1041         \MessageBreak Degrees used}}
1042     \AtBeginDocument{\ifx\DeclareEncodingSubset\@undefined
1043       \DeclareRobustCommand*{\degres}{%
1044         \leavevmode\hbox to 0.3em{\hss\degre\hss}%
1045         \Warning@degree@TSone
1046         \global\let\Warning@degree@TSone\relax}%
1047       \else
1048         \DeclareRobustCommand*{\degres}{%
1049           \hbox{\UseTextSymbol{TS1}{\textdegree}}}%
1050       \fi
1051     }
1052   \fi
1053 \else
1054   \newcommand*{\degres}{%
1055     \leavevmode\hbox to 0.3em{\hss\degre\hss}}
1056 \fi

```

2.6 Formatting numbers

`\StandardMathComma` As mentioned in the T_EXbook p. 134, the comma is of type `\mathpunct` in math mode: `\DecimalMathComma` it is automatically followed by a thin space. This is convenient in lists and intervals but unpleasant when the comma is used as a decimal separator in French: it has to be entered as `{,}`. `\DecimalMathComma` makes the comma be an ordinary character (of type `\mathord`) in French *only* (no space added); `\StandardMathComma` switches back to the standard behaviour of the comma.

Unfortunately, `\newcount` inside `\if` breaks Plain formats.

```

1057 \newif\ifFB@icomma
1058 \newcount\mc@charclass
1059 \newcount\mc@charfam
1060 \newcount\mc@charslot
1061 \newcount\std@mcc
1062 \newcount\dec@mcc
1063 \ifBLuaTeX
1064   \mc@charclass=\Umathcharclass'\,
1065   \newcommand*{\dec@math@comma}{%
1066     \mc@charfam=\Umathcharfam'\,
1067     \mc@charslot=\Umathcharslot'\,

```

```

1068 \Umathcode'\,= 0 \mc@charfam \mc@charslot
1069 }
1070 \newcommand*\std@math@comma{%
1071 \mc@charfam=\Umathcharfam'\,
1072 \mc@charslot=\Umathcharslot'\,
1073 \Umathcode'\,= \mc@charclass \mc@charfam \mc@charslot
1074 }
1075 \else
1076 \std@mcc=\mathcode'\,
1077 \dec@mcc=\std@mcc
1078 \@tempcnta=\std@mcc
1079 \divide\@tempcnta by "1000
1080 \multiply\@tempcnta by "1000
1081 \advance\dec@mcc by -\@tempcnta
1082 \newcommand*\dec@math@comma{\mathcode'\,=\dec@mcc}
1083 \newcommand*\std@math@comma{\mathcode'\,=\std@mcc}
1084 \fi
1085 \newcommand*\DecimalMathComma{%
1086 \ifFBfrench\dec@math@comma\fi
1087 \ifFB@icomma\else\addto\extrasfrench{\dec@math@comma}\fi
1088 }
1089 \newcommand*\StandardMathComma{%
1090 \std@math@comma
1091 \ifFB@icomma\else\addto\extrasfrench{\std@math@comma}\fi
1092 }
1093 \ifLaTeXe
1094 \AtBeginDocument{\@ifpackageloaded{icomma}%
1095 {\FB@icommatrue}%
1096 {\addto\noextrasfrench{\std@math@comma}}%
1097 }
1098 \else
1099 \addto\noextrasfrench{\std@math@comma}
1100 \fi

```

\nombre The command `\nombre` is now borrowed from `numprint.sty` for LaTeX2e. There is no point to maintain the former tricky code when a package is dedicated to do the same job and more. For Plain based formats, `\nombre` no longer formats numbers, it prints them as is and issues a warning about the change. Fake command `\nombre` for Plain based formats, warning users of babel-french v. 1.x. about the change:

```

1101 \newcommand*\nombre}[1]{\fb@warning{*** \noexpand\nombre
1102 no longer formats numbers\string! ***}}

```

Let's activate LuaTeX punctuation if necessary (LaTeX or Plain) so that `\FBsetspace` commands can be used in the preamble, then cleanup and exit without loading any `.cfg` file in case of Plain formats.

```

1103 \ifFB@luatex@punct
1104 \activate@luatexpunct
1105 \fi
1106 \let\FBstop@here\relax

```

```

1107 \def\FBclean@on@exit{%
1108   \let\ifLaTeXe\undefined
1109   \let\LaTeXtrue\undefined
1110   \let\LaTeXfalse\undefined
1111   \let\FB@llc\loadlocalcfg
1112   \let\loadlocalcfg@gobble}
1113 \ifx\magnification\@undefined
1114 \else
1115   \def\FBstop@here{%
1116     \FBclean@on@exit
1117     \ldf@finish\CurrentOption
1118     \let\loadlocalcfg\FB@llc
1119     \endinput}
1120 \fi
1121 \FBstop@here

```

What follows is for LaTeX2e *only*. We redefine `\nombre` for LaTeX2e. A warning is issued at the first call of `\nombre` if `\numprint` is not defined, suggesting what to do. The package `numprint` is *not* loaded automatically by `babel-french` because of possible options conflict.

```

1122 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1123 \newcommand*{\Warning@nombre}[1]{%
1124   \ifdefined\numprint
1125     \numprint{#1}%
1126   \else
1127     \PackageWarning{french.ldf}{%
1128       \protect\nombre\space now relies on package numprint.sty,%
1129       \MessageBreak add \protect
1130       \usepackage[autolanguage]{numprint},\MessageBreak
1131       see file numprint.pdf for more options.\MessageBreak
1132       \protect\nombre\space called}%
1133     \global\let\Warning@nombre\relax
1134     {#1}%
1135   \fi
1136 }

1137 \newcommand*{\FBthousandsep}{\kern \fontdimen2\font \relax}

```

2.7 Caption names

The next step consists in defining the French equivalents for the LaTeX caption names.

`\captionsfrench` Let's first define `\captionsfrench` which sets all strings used in the four standard document classes provided with LaTeX.

Let's give a chance to a class or a package read before `babel-french` to define `\FBfigtabshape` as `\relax`, otherwise `\FBfigtabshape` will be defined as `\scshape` (can be changed with `\frenchsetup{SmallCapsFigTabCaptions=false}`).

```

1138 \providecommand*{\FBfigtabshape}{\scshape}

```

New implementation for caption names(requires babel's 3.10 or newer).

```
1139 \StartBabelCommands*\BabelLanguages}{captions}
1140     [unicode, fontenc=TU EU1 EU2, charset=utf8]
1141     \SetString{\refname}{Références}
1142     \SetString{\abstractname}{Résumé}
1143     \SetString{\prefacename}{Préface}
1144     \SetString{\contentsname}{Table des matières}
1145     \SetString{\ccname}{Copie à }
1146     \SetString{\proofname}{Démonstration}
1147     \SetString{\partfirst}{Première}
1148     \SetString{\partsecond}{Deuxième}
1149     \SetStringLoop{ordinal#1}{%
1150         \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%
1151         Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1152         Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1153         Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1154 \StartBabelCommands*\BabelLanguages}{captions}
1155     \SetString{\refname}{R'ef'erences}
1156     \SetString{\abstractname}{R'esum'e}
1157     \SetString{\bibname}{Bibliographie}
1158     \SetString{\prefacename}{Pr'eface}
1159     \SetString{\chaptername}{Chapitre}
1160     \SetString{\appendixname}{Annexe}
1161     \SetString{\contentsname}{Table des mati\`eres}
1162     \SetString{\listfigurename}{Table des figures}
1163     \SetString{\listtablename}{Liste des tableaux}
1164     \SetString{\indexname}{Index}
1165     \SetString{\figurename}{\FBfigtabshape Figure}}
1166     \SetString{\tablename}{\FBfigtabshape Table}}
1167     \SetString{\pagename}{page}
1168     \SetString{\seename}{voir}
1169     \SetString{\alsoname}{voir aussi}
1170     \SetString{\enclname}{P.~J. }
1171     \SetString{\ccname}{Copie \`a }
1172     \SetString{\headtoname}{}
1173     \SetString{\proofname}{D\`emonstration}
1174     \SetString{\glossaryname}{Glossaire}
```

When `PartNameFull=true` (default), `\part{}` is printed in French as “Première partie” instead of “Partie I”. As logic is prohibited inside `\SetString`, let's hide the test about `PartNameFull` in `\FB@partname`.

```
1175     \SetString{\partfirst}{Premi\`ere}
1176     \SetString{\partsecond}{Deuxi\`eme}
1177     \SetString{\partnameord}{partie}
1178     \SetStringLoop{ordinal#1}{%
1179         \partfirst,\partsecond,Troisi\`eme,Quatri\`eme,%
1180         Cinqi\`eme,Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
1181         Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1182         Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1183         Vingti\`eme}
1184 \AfterBabelCommands{%
```



```

1185 \DeclareRobustCommand*\FB@emptypart\def\thepart{}}%
1186 \DeclareRobustCommand*\FB@partname}{%
1187   \ifFBPartNameFull
1188     \csname ordinal\romannumeral\value{part}\endcsname\space
1189     \partnameord\FB@emptypart
1190   \else
1191     Partie%
1192   \fi}%
1193 }
1194 \SetString{\partname}\FB@partname}
1195 \EndBabelCommands

```

2.8 Figure and table captions

\FBWarning `\FBWarning` is an alias of `\PackageWarning{french.ldb}` which can be made silent by option `SuppressWarning`.

```
1196 \newcommand*\FBWarning[1]{\PackageWarning{french.ldb}{#1}}
```

\CaptionSeparator Let's consider now captions in figures and tables. In French, captions in figures and tables should never be printed as 'Figure 1:' which is the default in standard LaTeX2e classes (a space should precede the colon in French). This flaw may occur with pdfLaTeX as ':' is made active too late. With LuaLaTeX and XeLaTeX, this glitch doesn't occur, you get 'Figure 1 :' which is correct in French. With pdfLaTeX babel-french provides the following workaround.

The standard definition of `\@makecaption` (e.g., the one provided in `article.cls`, `report.cls`, `book.cls` which is frozen for LaTeX2e according to Frank Mittelbach), is saved in `\STD@makecaption`. 'AtBeginDocument' we compare it to its current definition (some classes like `memoir`, `koma-script` classes, `AMS` classes, `ua-thesis.cls`... change it). If they are identical, `babel-french` just adds a hook called `\FBCaption@Separator` to `\@makecaption`; `\FBCaption@Separator` defaults to ':' as in the standard `\@makecaption` and will be changed to ':' in French 'AtBeginDocument'; it can be also set to `\CaptionSeparator` ('-') using `CustomiseFigTabCaptions`.

While saving the standard definition of `\@makecaption` we have to make sure that characters ':' and '>' have `\catcode 12` (`babel-french` makes ':' active and `spanish.ldb` makes '>' active).

```

1197 \bgroup
1198 \catcode':=12 \catcode'>=12 \relax
1199 \long\gdef\STD@makecaption#1#2{%
1200   \vskip\abovecaptionskip
1201   \sbox\@tempboxa{#1: #2}%
1202   \ifdim \wd\@tempboxa >\hsize
1203     #1: #2\par
1204   \else
1205     \global \@minipagefalse
1206     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1207   \fi
1208   \vskip\belowcaptionskip}
1209 \egroup

```

No warning is issued for SMF, AMS and ACM classes as their layout of captions is compatible with French typographic standards.

With memoir and koma-script classes, babel-french customises `\captiondelim` or `\captionformat` in French (unless option `CustomiseFigTabCaptions` is set to `false`) and issues no warning.

When `\@makecaption` has been changed by another class or package, a warning is printed in the .log file.

Enable the standard warning only if high punctuation is active.

```
1210 \newif\if@FBwarning@capsep
1211 \ifFB@active@punct\@FBwarning@capseptrue\fi
1212 \newcommand*\CaptionSeparator{\space\textendash\space}
1213 \def\FBCaption@Separator{: }
1214 \long\def\FB@makecaption#1#2{%
1215   \vskip\abovecaptionskip
1216   \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1217   \ifdim \wd\@tempboxa >\hsize
1218     #1\FBCaption@Separator #2\par
1219   \else
1220     \global \@minipagefalse
1221     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1222   \fi
1223   \vskip\belowcaptionskip}
```

Disable the standard warning with ACM, AMS and SMF classes.

```
1224 \@ifclassloaded{acmart}{\@FBwarning@capsepfalse}{}
1225 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1226 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1227 \@ifclassloaded{amsdtx}{\@FBwarning@capsepfalse}{}
1228 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1229 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1230 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1231 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}

```

No warning with memoir or koma-script classes: they change `\@makecaption` but we will manage to customise them in French later on (see below after executing `\FBprocess@options`).

```
1232 \newif\ifFB@koma
1233 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1234 \@ifclassloaded{scrartcl}{\@FBwarning@capsepfalse\FB@komatrue}{}
1235 \@ifclassloaded{scrbook}{\@FBwarning@capsepfalse\FB@komatrue}{}
1236 \@ifclassloaded{scrreprt}{\@FBwarning@capsepfalse\FB@komatrue}{}

```

No warning with the beamer class which defines `\beamer@makecaption` (customised below) instead of `\@makecaption`. No warning either if `\@makecaption` is undefined (i.e. letter).

```
1237 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1238 \ifdefined\@makecaption\else\FB@warning@capsepfalse\fi

```

The caption, subcaption and floatrow packages are compatible with babel-french if they are loaded after babel.

Check if packages `caption3` `subcaption` or `floatrow` are loaded now (before `babel-french`) and step counter `FBcaption@count` accordingly; it's value will be checked `\AtBeginDocument`. N.B.: `caption` loads `caption3`, `subcaption` loads `caption3` and `floatrow` loads `caption3`.

```
1239 \newcounter{FBcaption@count}
1240 \@ifpackageloaded{caption3}{\addtocounter{FBcaption@count}{4}}{}
1241 \@ifpackageloaded{subcaption}{\addtocounter{FBcaption@count}{2}}{}
1242 \@ifpackageloaded{floatrow}{\stepcounter{FBcaption@count}}{}
```

First check the definition of `\@makecaption`, change it or issue a warning in case it has been changed by a class or package not (yet) compatible with `babel-french`; then change the definition of `\FBCaption@Separator`, taking care that the colon is typeset correctly in French (*not* 'Figure 1: légende').

```
1243 \AtBeginDocument{%
1244   \ifx\@makecaption\STD@makecaption
1245     \global\let\@makecaption\FB@makecaption
```

If `OldFigTabCaptions=true`, do not overwrite `\FBCaption@Separator` (already saved as `'` for other languages and set to `\CaptionSeparator` by `\extrasfrench` when French is the main language); otherwise add a space before the `'` in French in order to avoid problems when `AutoSpacePunctuation=false`.

```
1246   \ifFBOldFigTabCaptions
1247   \else
1248     \def\FBCaption@Separator{\ifFBfrench\space\fi : }%
1249   \fi
1250   \ifFBCustomiseFigTabCaptions
1251     \ifFB@mainlanguage@FR
1252       \def\FBCaption@Separator{\CaptionSeparator}%
1253     \fi
1254   \fi
1255   \@FBwarning@capsepfalse
1256 \fi
```

Cancel the warning if `caption3.sty` has been loaded *after* `babel`.

```
1257 \@ifpackageloaded{caption3}{%
1258   \ifnum\value{FBcaption@count}=0 \@FBwarning@capsepfalse\fi
1259 }{}%
1260 \if@FBwarning@capsep
1261   \ifnum\value{FBcaption@count}>0
```

`caption3.sty` has been loaded *before* `babel`, maybe by the class...

```
1262   \FBWarning
1263     {Figures' and tables' captions might look like\MessageBreak
1264     'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1265     If you have loaded any of the packages caption,\MessageBreak
1266     subcaption or floatrow BEFORE babel/french,\MessageBreak
1267     please move them AFTER babel/french.\MessageBreak
1268     If one of them is loaded by your class,\MessageBreak
1269     you can still add AFTER babel/french\MessageBreak
1270     \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1271     \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
```

```

1272     ... live with it; reported}%
1273     \else
caption3.sty hasn't been loaded at all.
1274     \FBWarning
1275     {Figures' and tables' captions might look like\MessageBreak
1276     'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1277     If it happens, see your class documentation to\MessageBreak
1278     fix this issue or add AFTER babel/french\MessageBreak
1279     \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1280     \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1281     or ... live with it; reported}%
1282     \fi
1283     \fi
1284     \let\FB@makecaption\relax
1285     \let\STD@makecaption\relax
1286 }

```

2.9 Dots...

`\FBtextellipsis` LaTeX's standard definition of `\dots` in text-mode is `\textellipsis` which includes a `\kern` at the end; this space is not wanted in some cases (before a closing brace for instance) and `\kern` breaks hyphenation of the next word. We define `\FBtextellipsis` for French (in LaTeX only).

The `\if` construction in the LaTeX definition of `\dots` doesn't allow the use of `xspace` (`xspace` is always followed by a `\fi`), so we use the AMS-LaTeX construction of `\dots`; this has to be done 'AtBeginDocument' not to be overwritten when `amsmath.sty` is loaded after `babel`.

LY1 has a ready made character for `\textellipsis`, it should be used in French too. The same is true for Unicode fonts in use with XeTeX and LuaTeX.

```

1287 \iffBunicode
1288 \let\FBtextellipsis\textellipsis
1289 \else
1290 \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1291 \DeclareTextCommandDefault{\FBtextellipsis}{%
1292     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}
1293 \fi

```

`\Mdots@` and `\Tdots@` hold the definitions of `\dots` in Math and Text mode. They default to those of `amsmath-2.0`, and will revert to standard LaTeX definitions 'AtBeginDocument', if `amsmath` has not been loaded. `\Mdots@` doesn't change when switching from/to French, while `\Tdots@` is redefined as `\FBtextellipsis` in French.

```

1294 \newcommand*{\Tdots@}{\@xp\textellipsis}
1295 \newcommand*{\Mdots@}{\@xp\mdots@}
1296 \AtBeginDocument{\DeclareRobustCommand*{\dots}{\relax
1297     \csname\ifmmode M\else T\fi dots@\endcsname}%
1298     \ifdefined\@xp\else\let\@xp\relax\fi
1299     \ifdefined\mdots@\else\let\Mdots@\mathellipsis\fi
1300 }
1301 \def\bbl@frenchdots{\babel@save\Tdots@ \let\Tdots@\FBtextellipsis}

```

```
1302 \addto\extrasfrench{\bbl@frenchdots}
```

2.10 More checks about packages' loading order

Like packages `captions` and `floatrow` (see section 2.8), package listings should be loaded after `babel-french` due to active characters issues (pdfLaTeX only).

```
1303 \ifFB@active@punct
1304   \@ifpackageloaded{listings}
1305     {\AtBeginDocument{%
1306       \FBWarning{Please load the "listings" package\MessageBreak
1307         AFTER babel/french; reported}}%
1308     }{}
1309 \fi
```

Package `natbib` should be loaded before `babel-french` due to active characters issues (pdfLaTeX only).

```
1310 \newif\if@FBwarning@natbib
1311 \ifFB@active@punct
1312   \@ifpackageloaded{natbib}{}{\@FBwarning@natbibtrue}
1313 \fi
1314 \AtBeginDocument{%
1315   \if@FBwarning@natbib
1316     \@ifpackageloaded{natbib}{}{\@FBwarning@natbibfalse}%
1317   \fi
1318   \if@FBwarning@natbib
1319     \FBWarning{Please load the "natbib" package\MessageBreak
1320       BEFORE babel/french; reported}%
1321   \fi
1322 }
```

Package `beamerarticle` should be loaded before `babel-french` to avoid list's conflicts, see p. 55.

```
1323 \newif\if@FBwarning@beamerarticle
1324 \@ifpackageloaded{beamerarticle}{}{\@FBwarning@beamerarticlettrue}
1325 \AtBeginDocument{%
1326   \if@FBwarning@beamerarticle
1327     \@ifpackageloaded{beamerarticle}{}%
1328     {\@FBwarning@beamerarticletfalse}%
1329   \fi
1330   \if@FBwarning@beamerarticle
1331     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1332       BEFORE babel/french; reported}%
1333   \fi
1334 }
```

2.11 Setup options: keyval stuff

All setup options are handled by command `\frenchsetup{}` using the `keyval` syntax. A list of flags is defined and set to a default value which will possibly be changed

'AtEndOfPackage' if French is the main language. After this, `\frenchsetup{}` eventually modifies the preset values of these flags.

Option processing can occur either in `\frenchsetup{}`, but *only for options explicitly set* by `\frenchsetup{}`, or 'AtBeginDocument'; any option affecting `\extrafrench{}` *must* be processed by `\frenchsetup{}`: when French is the main language, `\extrafrench{}` is executed by babel when it switches the main language and this occurs *before* reading the stuff postponed by babel-french 'AtBeginDocument'. Reexecuting `\extrafrench{}` is an option which was used up to v2.6h, it has been dropped in v3.0a because of its side-effects (f.i. `\babel@save` and `\babel@savevariable` did not work for French).

`\frenchsetup` Let's now define this command which reads and sets the options to be processed either immediately (i.e. just after setting the key) or later (at `\begin{document}`) by `\FBprocess@options`. `\frenchsetup{}` can only be called in the preamble.

```
1335 \newcommand*{\frenchsetup}[1]{%
1336   \setkeys{FB}{#1}%
1337 }%
1338 \@onlypreamble\frenchsetup
```

Keep the former name `\frenchbsetup` working for compatibility.

```
1339 \let\frenchbsetup\frenchsetup
1340 \@onlypreamble\frenchbsetup
```

We define a collection of conditionals with their defaults (true or false).

```
1341 \newif\ifFBShowOptions
1342 \newif\ifFBStandardLayout           \FBStandardLayouttrue
1343 \newif\ifFBGlobalLayoutFrench      \FBGlobalLayoutFrenchtrue
1344 \newif\ifFBReduceListSpacing
1345 \newif\ifFBStandardListSpacing    \FBStandardListSpacingtrue
1346 \newif\ifFBListOldLayout
1347 \newif\ifFBListItemsAsPar
1348 \newif\ifFBCompactItemize
1349 \newif\ifFBStandardItemizeEnv      \FBStandardItemizeEnvtrue
1350 \newif\ifFBStandardEnumerateEnv    \FBStandardEnumerateEnvtrue
1351 \newif\ifFBStandardItemLabels      \FBStandardItemLabelstrue
1352 \newif\ifFBStandardLists          \FBStandardListstrue
1353 \newif\ifFBIndentFirst
1354 \newif\ifFBFrenchFootnotes
1355 \newif\ifFBAutoSpaceFootnotes
1356 \newif\ifFBOriginalTypewriter
1357 \newif\ifFBThinColonSpace
1358 \newif\ifFBThinSpaceInFrenchNumbers
1359 \newif\ifFBFrenchSuperscripts      \FBFrenchSuperscriptstrue
1360 \newif\ifFBLowercaseSuperscripts   \FBLowercaseSuperscriptstrue
1361 \newif\ifFBPartNameFull            \FBPartNameFulltrue
1362 \newif\ifFBCustomiseFigTabCaptions
1363 \newif\ifFBOldFigTabCaptions
1364 \newif\ifFBSmallCapsFigTabCaptions \FBSmallCapsFigTabCaptionstrue
1365 \newif\ifFBSuppressWarning
1366 \newif\ifFBINGuillSpace
```

The default values of these flags have been chosen so that babel-french does not change anything regarding the global layout. `\bbl@main@language`, set by the last option of babel, controls the global layout of the document. ‘AtEndOfPackage’ we check the main language in `\bbl@main@language`; if it is French (or a French dialect) the values of some flags have to be changed to ensure a French looking layout for the whole document (even in parts written in languages other than French); the end-user will then be able to customise the values of all these flags with `\frenchsetup{}`. The following patch is for koma-script classes: the `\partformat` command, defined as `\partname~\thepart\autodot`, is incompatible with our redefinition of `\partname`.

```

1367 \ifFB@koma
1368   \ifdefined\partformat
1369     \def\FB@partformat@fix{%
1370       \ifFBPartNameFull
1371         \babel@save\partformat
1372         \renewcommand*{\partformat}{\partname}%
1373       \fi
1374     \addto\extrasfrench{\FB@partformat@fix}%
1375   \fi
1376 \fi

```

Our list customisation conflicts with the beamer class and with the beamerarticle package. The patch provided in `beamerbasecompatibility` solves the conflict except in case of language changes, so we provide our own patch. When the beamer is loaded, lists are not customised at all to ensure compatibility. The beamerarticle package needs to be loaded *before* babel, a warning is issued otherwise, see section 2.10; a light customisation is compatible with the beamerarticle package.

```

1377 \def\FB@french{french}
1378 \def\FB@acadian{acadian}
1379 \newif\ifFB@mainlanguage@FR
1380 \AtEndOfPackage{%
1381   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1382   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1383   \fi
1384   \ifFB@mainlanguage@FR
1385     \FBGlobalLayoutFrenchtrue
1386     \@ifclassloaded{beamer}%
1387       {\PackageInfo{french.ldf}{%
1388         No list customisation for the beamer class,%
1389         \MessageBreak reported}}%
1390     {\@ifpackageloaded{beamerarticle}%
1391       {\FBStandardItemLabelsfalse
1392        \FBStandardListSpacingfalse
1393        \PackageInfo{french.ldf}{%
1394          Minimal list customisation for the beamerarticle%
1395          \MessageBreak package; reported}}%

```

Otherwise customise lists “à la française”:

```

1396     {\FBStandardListSpacingfalse
1397     \FBStandardItemizeEnvfalse
1398     \FBStandardEnumerateEnvfalse

```

```

1399         \FBStandardItemLabelsfalse}%
1400     }
1401     \FBIndentFirsttrue
1402     \FBFrenchFootnotestru
1403     \FBAutoSpaceFootnotestru
1404     \FBCustomiseFigTabCaptionstru
1405 \else
1406     \FBGlobalLayoutFrenchfalse
1407 \fi

```

babel-french being an option of babel, it cannot load a package (keyval) while french. ldf is read, so we defer the loading of keyval and the options setup at the end of babel's loading.

```

1408 \RequirePackage{keyval}%
1409 \define@key{FB}{ShowOptions}[true]%
1410     {\csname FBShowOptions#1\endcsname}%

```

The next two keys can only be toggled when French is the main language.

```

1411 \define@key{FB}{StandardLayout}[true]%
1412     {\ifFB@mainlanguage@FR
1413         \csname FBStandardLayout#1\endcsname
1414     \else
1415         \PackageWarning{french. ldf}%
1416             {Option 'StandardLayout' skipped:\MessageBreak
1417                 French is *not* babel's last option.\MessageBreak
1418                 Reported}%
1419     \fi
1420 \ifFBStandardLayout
1421     \FBStandardListSpacingtrue
1422     \FBStandardItemizeEnvtrue
1423     \FBStandardItemLabelstru
1424     \FBStandardEnumerateEnvtrue
1425     \FBIndentFirstfalse
1426     \FBFrenchFootnotesfalse
1427     \FBAutoSpaceFootnotesfalse
1428     \FBGlobalLayoutFrenchfalse
1429 \else
1430     \FBStandardListSpacingfalse
1431     \FBStandardItemizeEnvfalse
1432     \FBStandardItemLabelsfalse
1433     \FBStandardEnumerateEnvfalse
1434     \FBIndentFirsttrue
1435     \FBFrenchFootnotestru
1436     \FBAutoSpaceFootnotestru
1437 \fi}%
1438 \define@key{FB}{GlobalLayoutFrench}[true]%
1439     {\ifFB@mainlanguage@FR
1440         \csname FBGlobalLayoutFrench#1\endcsname
1441     \else
1442         \PackageWarning{french. ldf}%
1443             {Option 'GlobalLayoutFrench' skipped:\MessageBreak
1444                 French is *not* babel's last option.\MessageBreak

```



```

1445         Reported}%
1446     \fi}%

```

If this key is set to `true` when French is the main language, nothing to do: all flags keep their default value. If this key is set to `false`, nothing to do either: `\babel@save` will do the job.

```

1447 \define@key{FB}{ReduceListSpacing}[true]%
1448     {\csname FBReduceListSpacing#1\endcsname
1449     \ifFBReduceListSpacing \FBStandardListSpacingfalse
1450     \else \FBStandardListSpacingtrue\fi
1451     }%
1452 \define@key{FB}{StandardListSpacing}[true]%
1453     {\csname FBStandardListSpacing#1\endcsname}%
1454 \define@key{FB}{ListOldLayout}[true]%
1455     {\csname FBListOldLayout#1\endcsname
1456     \ifFBListOldLayout
1457         \FBStandardEnumerateEnvtrue
1458         \renewcommand*\FrenchLabelItem{\textendash}%
1459     \fi}%
1460 \define@key{FB}{CompactItemize}[true]%
1461     {\csname FBCompactItemize#1\endcsname
1462     \ifFBCompactItemize
1463         \FBStandardItemizeEnvfalse
1464         \FBStandardEnumerateEnvfalse
1465     \else
1466         \FBStandardItemizeEnvtrue
1467         \FBStandardEnumerateEnvtrue
1468     \fi}%
1469 \define@key{FB}{StandardItemizeEnv}[true]%
1470     {\csname FBStandardItemizeEnv#1\endcsname}%
1471 \define@key{FB}{StandardEnumerateEnv}[true]%
1472     {\csname FBStandardEnumerateEnv#1\endcsname}%
1473 \define@key{FB}{StandardItemLabels}[true]%
1474     {\csname FBStandardItemLabels#1\endcsname}%
1475 \define@key{FB}{ItemLabels}%
1476     {\renewcommand*\FrenchLabelItem{#1}}%
1477 \define@key{FB}{ItemLabeli}%
1478     {\renewcommand*\Frlabelitemi{#1}}%
1479 \define@key{FB}{ItemLabelii}%
1480     {\renewcommand*\Frlabelitemii{#1}}%
1481 \define@key{FB}{ItemLabeliii}%
1482     {\renewcommand*\Frlabelitemiii{#1}}%
1483 \define@key{FB}{ItemLabeliv}%
1484     {\renewcommand*\Frlabelitemiv{#1}}%
1485 \define@key{FB}{StandardLists}[true]%
1486     {\csname FBStandardLists#1\endcsname
1487     \ifFBStandardLists
1488         \FBStandardListSpacingtrue
1489         \FBStandardItemizeEnvtrue
1490         \FBStandardEnumerateEnvtrue
1491         \FBStandardItemLabelstrue

```

```

1492         \else
1493             \FBStandardListSpacingfalse
1494             \FBStandardItemizeEnvfalse
1495             \FBStandardEnumerateEnvfalse
1496             \FBStandardItemLabelsfalse
1497         \fi}%
1498 \define@key{FB}{ListItemsAsPar}[true]%
1499     {\csname FBListItemsAsPar#1\endcsname}
1500 \define@key{FB}{IndentFirst}[true]%
1501     {\csname FBIndentFirst#1\endcsname}%
1502 \define@key{FB}{FrenchFootnotes}[true]%
1503     {\csname FBFrenchFootnotes#1\endcsname}%
1504 \define@key{FB}{AutoSpaceFootnotes}[true]%
1505     {\csname FBAutoSpaceFootnotes#1\endcsname}%
1506 \define@key{FB}{AutoSpacePunctuation}[true]%
1507     {\csname FBAutoSpacePunctuation#1\endcsname}%
1508 \define@key{FB}{OriginalTypewriter}[true]%
1509     {\csname FBOriginalTypewriter#1\endcsname}%
1510 \define@key{FB}{ThinColonSpace}[true]%
1511     {\csname FBThinColonSpace#1\endcsname
1512     \ifFBThinColonSpace
1513         \renewcommand*{\FBcolonspace}{\FBthinspace}%
1514     \fi}%
1515 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1516     {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1517 \define@key{FB}{FrenchSuperscripts}[true]%
1518     {\csname FBFrenchSuperscripts#1\endcsname}
1519 \define@key{FB}{LowercaseSuperscripts}[true]%
1520     {\csname FBLowercaseSuperscripts#1\endcsname}
1521 \define@key{FB}{PartNameFull}[true]%
1522     {\csname FBPartNameFull#1\endcsname}%
1523 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1524     {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1525 \define@key{FB}{OldFigTabCaptions}[true]%
1526     {\csname FBOldFigTabCaptions#1\endcsname
1527     \ifFBOldFigTabCaptions
1528         \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1529             \def\FBCaption@Separator{\CaptionSeparator}}%
1530         \addto\extrasfrench{\FB@capsep@fix}%
1531         \ifdefined\extrasacadian
1532             \addto\extrasacadian{\FB@capsep@fix}%
1533         \fi
1534     \fi}%
1535 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1536     {\csname FBSmallCapsFigTabCaptions#1\endcsname
1537     \ifFBSmallCapsFigTabCaptions
1538         \let\FBfigtabshape\scshape
1539     \else
1540         \let\FBfigtabshape\relax
1541     \fi}%
1542 \define@key{FB}{SuppressWarning}[true]%

```

```

1543     {\csname FBSuppressWarning#1\endcsname
1544     \ifFBSuppressWarning
1545       \renewcommand{\FBWarning}[1]{}%
1546     \fi}%

```

Here are the options controlling French guillemets spacing and the output of `\frquote{}`.

```

1547 \define@key{FB}{INGuillSpace}[true]%
1548   {\csname FBINGuillSpace#1\endcsname
1549   \ifFBINGuillSpace
1550     \renewcommand*{\FBguillspace}{\space}%
1551   \fi}%
1552 \define@key{FB}{InnerGuillSingle}[true]%
1553   {\csname FBInnerGuillSingle#1\endcsname}%
1554 \define@key{FB}{EveryParGuill}[open]%
1555   {\expandafter\let\expandafter
1556     \FBeveryparguill\csname FBguill#1\endcsname
1557   \ifx\FBeveryparguill\FBguillopen
1558   \else\ifx\FBeveryparguill\FBguillclose
1559     \else\ifx\FBeveryparguill\FBguillnone
1560     \else
1561       \let\FBeveryparguill\FBguillopen
1562       \FBWarning{Wrong value for 'EveryParGuill':
1563         try 'open',\MessageBreak
1564         'close' or 'none'. Reported}%
1565     \fi
1566   \fi
1567 \fi}%
1568 \define@key{FB}{EveryLineGuill}[open]%
1569   {\ifFB@luatex@punct
1570   \expandafter\let\expandafter
1571     \FBeverylineguill\csname FBguill#1\endcsname
1572   \ifx\FBeverylineguill\FBguillopen
1573   \else\ifx\FBeverylineguill\FBguillclose
1574     \else\ifx\FBeverylineguill\FBguillnone
1575     \else
1576       \let\FBeverylineguill\FBguillnone
1577       \FBWarning{Wrong value for 'EveryLineGuill':
1578         try 'open',\MessageBreak
1579         'close' or 'none'. Reported}%
1580     \fi
1581   \fi
1582   \fi
1583   \else
1584     \FBWarning{Option 'EveryLineGuill' skipped:%
1585       \MessageBreak this option is for
1586       LuaTeX *only*.\MessageBreak Reported}%
1587   \fi}%

```

Option [UnicodeNoBreakSpaces](#) (LuaLaTeX only) is meant for HTML translators: when true, all non-breaking spaces added by babel-french are coded in the PDF file as

Unicode characters, namely U+A0 or U+202F, instead of penalties and glues.

```
1588 \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1589     {\ifFB@luatex@punct
1590         \csname FBucsNBSP#1\endcsname
1591         \ifFBucsNBSP \FB@ucsNBSP=1 \fi
1592     \else
1593         \FBWarning{Option 'UnicodeNoBreakSpaces' skipped:%
1594             \MessageBreak this option is for
1595             LuaTeX *only*.\MessageBreak Reported}%
1596     \fi
1597 }%
```

Inputting French quotes as *single characters* when they are available on the keyboard (through a compose key for instance) is more comfortable than typing `\og` and `\fg`. Life is simple here with modern LuaTeX or XeTeX engines: we just have to activate the `\FB@addGUILspace` attribute for LuaTeX or set `\XeTeXcharclass` of quotes to the proper value for XeTeX.

With pdfTeX (or old LuaTeX and XeTeX engines), quote characters are made active and expand to `\og\ignorespaces` and `{\fg}` respectively if the current language is French, and to `\guillemotleft` and `\guillemotright` otherwise (think of German quotes), this is done by `\FB@@og` and `\FB@@fg`; thus correct non-breaking spaces will be added automatically to French quotes. The quote characters typed in depend on the input encoding, it can be single-byte (latin1, latin9, applemac, ...) or multi-bytes (utf-8, utf8x); the next command is meant for checking whether a character is single-byte (`\FB@second` is empty) or not.

```
1598 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
1599 \define@key{FB}{og}%
1600     {\ifFBunicode
```

LuaTeX or XeTeX in use, first try modern LuaTeX: we just need to set LuaTeX's attribute `\FB@addGUILspace` to 1,

```
1601         \ifFB@luatex@punct
1602             \FB@addGUILspace=1 \relax
1603         \fi
```

then with XeTeX it is a bit more tricky:

```
1604         \ifFB@xetex@punct
```

`\XeTeXinterchartokenstate` is defined, we just need to set `\XeTeXcharclass` to `\FB@guilo` for the French opening quote in T1 and Unicode encoding (see subsection 2.2).

```
1605             \XeTeXcharclass"13 = \FB@guilo
1606             \XeTeXcharclass"AB = \FB@guilo
1607             \XeTeXcharclass"A0 = \FB@guilnul
1608             \XeTeXcharclass"202F = \FB@guilnul
1609         \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1610         \ifFB@active@punct
1611             \FBWarning{Option og=< not supported with this version
1612                 of\MessageBreak LuaTeX/XeTeX; reported}%
```

```

1613         \fi
1614     \else

```

This is for conventional TeX engines:

```

1615         \newcommand*{\FB@og}{%
1616             \iffBfrench
1617             \ifFB@spacing\FB@og\ignorespaces
1618             \else\guillemotleft
1619             \fi
1620             \else\guillemotleft\fi}%
1621     \AtBeginDocument{%
1622         \ifdefined\uc@dclc

```

Package inputenc with utf8x (ucs) encoding loaded, use \uc@dclc:

```

1623         \uc@dclc{171}{default}{\FB@og}%
1624     \else

```

if encoding is not utf8x, check if the argument of og is a single-byte character:

```

1625         \FB@parse#1\endparse
1626         \ifx\FB@second\@empty

```

This means 8-bit character encoding. Package MULEenc (from CJK) defines \mule@def to map characters to control sequences.

```

1627         \ifdefined\mule@def
1628         \mule@def{11}{\FB@og}%
1629     \else
1630         \ifdefined\DeclareInputText
1631         \@tempcnta'#1\relax
1632         \DeclareInputText{\the\@tempcnta}{\FB@og}%
1633     \else

```

Package inputenc not loaded, no way...

```

1634         \FBWarning{Option 'og' requires package
1635             inputenc;\MessageBreak reported}%
1636     \fi
1637     \fi
1638     \else

```

This means multi-byte character encoding, we assume UTF-8

```

1639         \DeclareUnicodeCharacter{00AB}{\FB@og}%
1640     \fi
1641     \fi}%
1642     \fi
1643     }%

```

Same code for the closing quote.

```

1644     \define@key{FB}{fg}{%
1645         {\iffBunicode
1646             \ifFB@luatex@punct
1647             \FB@addGUILspace=1 \relax
1648             \fi
1649             \ifFB@xetex@punct
1650             \XeTeXcharclass"14 = \FB@guilf
1651             \XeTeXcharclass"BB = \FB@guilf

```

```

1652         \XeTeXcharclass"A0 = \FB@guilnul
1653         \XeTeXcharclass"202F = \FB@guilnul
1654     \fi
1655     \ifFB@active@punct
1656         \FBWarning{Option fg=> not supported with this version
1657             of\MessageBreak LuaTeX/XeTeX; reported}%
1658     \fi
1659 \else
1660     \newcommand*\FB@fg}{%
1661         \ifFBfrench
1662             \ifFB@spacing\FB@fg
1663             \else\guillemotright
1664             \fi
1665         \else\guillemotright\fi}%
1666 \AtBeginDocument{%
1667     \ifdefined\uc@dclc
1668         \uc@dclc{187}{default}{\FB@fg}%
1669     \else
1670         \FB@parse#1\endparse
1671         \ifx\FB@second\@empty
1672             \ifdefined\mule@def
1673                 \mule@def{27}{\FB@fg}%
1674             \else
1675                 \ifdefined\DeclareInputText
1676                     \@tempcnta'#1\relax
1677                     \DeclareInputText{\the\@tempcnta}{\FB@fg}%
1678                 \else
1679                     \FBWarning{Option 'fg' requires package
1680                         inputenc;\MessageBreak reported}%
1681                 \fi
1682             \fi
1683         \else
1684             \DeclareUnicodeCharacter{00BB}{\FB@fg}%
1685         \fi
1686     \fi}%
1687 \fi
1688 }%
1689 }

```

\FBprocess@options \FBprocess@options will be executed at \begin{document}: it first checks about packages loaded in the preamble (possibly after babel) which customise lists: currently enumitem, paralist and enumerate; then it processes the options as set by \frenchsetup{} or forced for compatibility with packages loaded in the preamble. When French is the main language, \extrasfrench and \captionsfrench *have already been processed* by babel at \begin{document} *before* \FBprocess@options.

```

1690 \newcommand*\FBprocess@options}{%

```

Update flags if a package customising lists has been loaded, currently: enumitem, paralist, enumerate.

```

1691 \@ifpackageloaded{enumitem}{%
1692     \ifFBStandardItemizeEnv

```

```

1693 \else
1694 \FBStandardItemizeEnvtrue
1695 \PackageInfo{french.ldb}%
1696 {Setting StandardItemizeEnv=true for\MessageBreak
1697 compatibility with enumitem package,\MessageBreak
1698 reported}%
1699 \fi
1700 \ifFBStandardEnumerateEnv
1701 \else
1702 \FBStandardEnumerateEnvtrue
1703 \PackageInfo{french.ldb}%
1704 {Setting StandardEnumerateEnv=true for\MessageBreak
1705 compatibility with enumitem package,\MessageBreak
1706 reported}%
1707 \fi}{}%
1708 \@ifpackageloaded{paralist}{%
1709 \ifFBStandardItemizeEnv
1710 \else
1711 \FBStandardItemizeEnvtrue
1712 \PackageInfo{french.ldb}%
1713 {Setting StandardItemizeEnv=true for\MessageBreak
1714 compatibility with paralist package,\MessageBreak
1715 reported}%
1716 \fi
1717 \ifFBStandardEnumerateEnv
1718 \else
1719 \FBStandardEnumerateEnvtrue
1720 \PackageInfo{french.ldb}%
1721 {Setting StandardEnumerateEnv=true for\MessageBreak
1722 compatibility with paralist package,\MessageBreak
1723 reported}%
1724 \fi}{}%
1725 \@ifpackageloaded{enumerate}{%
1726 \ifFBStandardEnumerateEnv
1727 \else
1728 \FBStandardEnumerateEnvtrue
1729 \PackageInfo{french.ldb}%
1730 {Setting StandardEnumerateEnv=true for\MessageBreak
1731 compatibility with enumerate package,\MessageBreak
1732 reported}%
1733 \fi}{}%

```

Reset \FB@ufl's normal meaning and update lists' settings now in case French is the main language:

```

1734 \def\FB@ufl{\update@frenchlists}
1735 \ifFB@mainlanguage@FR
1736 \update@frenchlists
1737 \fi

```

The layout of footnotes is handled at the \begin{document} depending on the values of flags [FrenchFootnotes](#) and [AutoSpaceFootnotes](#) (see section 2.14), nothing has to be done here for footnotes.

`AutoSpacePunctuation` adds a non-breaking space (in French only) before the four active characters (:;!?) even if none has been typed before them.

```
1738 \ifFBAutoSpacePunctuation
1739   \autospace@beforeFDP
1740 \else
1741   \noautospace@beforeFDP
1742 \fi
```

When `OriginalTypewriter` is set to `false` (the default), `\ttfamily`, `\rmfamily` and `\sffamily` are redefined as `\ttfamilyFB`, `\rmfamilyFB` and `\sffamilyFB` respectively to prevent addition of automatic spaces before the four active characters in computer code.

```
1743 \ifFBOriginalTypewriter
1744 \else
1745   \let\ttfamilyORI\ttfamily
1746   \let\rmfamilyORI\rmfamily
1747   \let\sffamilyORI\sffamily
1748   \let\ttfamily\ttfamilyFB
1749   \let\rmfamily\rmfamilyFB
1750   \let\sffamily\sffamilyFB
1751 \fi
```

When package `numprint` is loaded with option `autolanguage`, `numprint`'s command `\npstylefrench` has to be redefined differently according to the value of flag `ThinSpaceInFrenchNumbers`. As `\npstylefrench` was undefined in old versions of `numprint`, we provide this command.

```
1752 \@ifpackageloaded{numprint}%
1753   {\ifnprt@autolanguage
1754     \providecommand*\npstylefrench{}%
1755     \ifFBThinSpaceInFrenchNumbers
1756       \renewcommand*\FBthousandsep{\,}%
1757     \fi
1758     \g@addto@macro\npstylefrench{\npthousandsep\FBthousandsep}%
1759   \fi
1760 }{}%
```

`FrenchSuperscripts`: if `true` `\up=\fup`, else `\up=\textsuperscript`. Anyway `\up*=\FB@up@fake`. The star-form `\up*{}` is provided for fonts that lack some superior letters: Adobe Jenson Pro and Utopia Expert have no “g superior” for instance.

```
1761 \ifFBFrenchSuperscripts
1762   \DeclareRobustCommand*\up{\@ifstar\FB@up@fake}\fup}%
1763 \else
1764   \DeclareRobustCommand*\up{\@ifstar\FB@up@fake%
1765     \textsuperscript}%
1766 \fi
```

`LowercaseSuperscripts`: if `false` `\FB@lc` is redefined to do nothing.

```
1767 \ifFBLowercaseSuperscripts
1768 \else
1769   \renewcommand*\FB@lc[1]{##1}%
1770 \fi
```


Unless `CustomiseFigTabCaptions` has been set to `false`, use `\CaptionSeparator` for koma-script, memoir and beamer classes.

```

1771 \ifFBCustomiseFigTabCaptions
1772   \ifFB@koma
1773     \renewcommand*{\captionformat}{\CaptionSeparator}%
1774   \fi
1775   \@ifclassloaded{memoir}%
1776     {\captiondelim{\CaptionSeparator}}{%
1777   \@ifclassloaded{beamer}%
1778     {\defbeamertemplate{caption label separator}{FBcustom}{%
1779       \CaptionSeparator}%
1780     \setbeamertemplate{caption label separator}[FBcustom]}{%
1781   \else

```

When `CustomiseFigTabCaptions` is `false`, have the colon behave properly in French: locally force `\autospace@beforeFDP` in case of `AutoSpacePunctuation=false`.

```

1782   \ifFB@koma
1783     \renewcommand*{\captionformat}{\autospace@beforeFDP : }%
1784   \fi
1785   \@ifclassloaded{memoir}%
1786     {\captiondelim{\autospace@beforeFDP : }%
1787     }{%
1788   \@ifclassloaded{beamer}%
1789     {\defbeamertemplate{caption label separator}{FBcolon}{%
1790       \autospace@beforeFDP : }%
1791     \setbeamertemplate{caption label separator}[FBcolon]%
1792     }{%
1793   \fi

```

`ShowOptions`: if `true`, print the list of all options to the `.log` file.

```

1794 \ifFBShowOptions
1795   \GenericWarning{* }{%
1796     ***** List of possible options for babel-french *****\MessageBreak
1797     [Default values between brackets when french is loaded *LAST*]%
1798     \MessageBreak
1799     ShowOptions=true [false]\MessageBreak
1800     StandardLayout=true [false]\MessageBreak
1801     GlobalLayoutFrench=false [true]\MessageBreak
1802     PartNameFull=false [true]\MessageBreak
1803     IndentFirst=false [true]\MessageBreak
1804     ListItemsAsPar=true [false]\MessageBreak
1805     StandardListSpacing=true [false]\MessageBreak
1806     StandardItemizeEnv=true [false]\MessageBreak
1807     StandardEnumerateEnv=true [false]\MessageBreak
1808     StandardItemLabels=true [false]\MessageBreak
1809     ItemLabels=\textendash, \textbullet,
1810     \protect\ding{43},... [\textendash]\MessageBreak
1811     ItemLabeli=\textendash, \textbullet,
1812     \protect\ding{43},... [\textendash]\MessageBreak
1813     ItemLabelii=\textendash, \textbullet,
1814     \protect\ding{43},... [\textendash]\MessageBreak

```

```

1815 ItemLabeliii=\textendash, \textbullet,
1816 \protect\ding{43},... [\textendash]\MessageBreak
1817 ItemLabeliv=\textendash, \textbullet,
1818 \protect\ding{43},... [\textendash]\MessageBreak
1819 StandardLists=true [false]\MessageBreak
1820 ListOldLayout=true [false]\MessageBreak
1821 FrenchFootnotes=false [true]\MessageBreak
1822 AutoSpaceFootnotes=false [true]\MessageBreak
1823 AutoSpacePunctuation=false [true]\MessageBreak
1824 ThinColonSpace=true [false]\MessageBreak
1825 OriginalTypewriter=true [false]\MessageBreak
1826 UnicodeNoBreakSpaces=true [false]\MessageBreak
1827 og= <left quote character>, fg= <right quote character>%
1828 INGuillSpace=true [false]\MessageBreak
1829 EveryParGuill=open, close, none [open]\MessageBreak
1830 EveryLineGuill=open, close, none
1831 [open in LuaTeX, none otherwise]\MessageBreak
1832 InnerGuillSingle=true [false]\MessageBreak
1833 ThinSpaceInFrenchNumbers=true [false]\MessageBreak
1834 SmallCapsFigTabCaptions=false [true]\MessageBreak
1835 CustomiseFigTabCaptions=false [true]\MessageBreak
1836 OldFigTabCaptions=true [false]\MessageBreak
1837 FrenchSuperscripts=false [true]\MessageBreak
1838 LowercaseSuperscripts=false [true]\MessageBreak
1839 SuppressWarning=true [false]\MessageBreak
1840 \MessageBreak
1841 *****
1842 \MessageBreak\protect\frenchsetup{ShowOptions}}
1843 \fi
1844 }

```

At `\begin{document}`, we have to provide an `\xspace` command in case the `xspace` package is not loaded, do some setup for `hyperref`'s bookmarks, execute `\FBprocess@options`, switch LuaTeX punctuation on and issue some warnings if necessary.

```

1845 \AtBeginDocument{%
1846 \providecommand*\xspace{\relax}%

```

Let's redefine some commands in `hyperref`'s bookmarks.

```

1847 \ifdefined\pdfstringdefDisableCommands
1848 \pdfstringdefDisableCommands{%
1849 \let\up\relax
1850 \let\fup\relax
1851 \let\degre\textdegree
1852 \let\degres\textdegree
1853 \def\ieme{e\xspace}%
1854 \def\iemes{es\xspace}%
1855 \def\ier{er\xspace}%
1856 \def\iers{ers\xspace}%
1857 \def\iere{re\xspace}%
1858 \def\ieres{res\xspace}%

```

```

1859     \def\FrenchEnumerate#1{#1\degre\space}%
1860     \def\FrenchPopularEnumerate#1{#1\degre)\space}%
1861     \def\No{N\degre\space}%
1862     \def\no{n\degre\space}%
1863     \def\Nos{N\degre\space}%
1864     \def\nos{n\degre\space}%
1865     \def\FB@og{\guillemotleft\space}%
1866     \def\FB@fg{\space\guillemotright}%
1867     \def\frquote#1{\FB@og #1\FB@fg}%
1868     \def\at{@}%
1869     \def\circonflexe{\string^}%
1870     \def\tild{\string~}%
1871     \def\boi{\textbackslash}%
1872     \let\bsc\textsc
1873     }%
1874 \fi

```

Let's now process the remaining options, either not explicitly set by `\frenchsetup{}` or possibly modified by packages loaded after `babel-french`.

```
1875 \FBprocess@options
```

When option `UnicodeNoBreakSpaces` is `true` (LuaLaTeX only) we need to redefine `\FBmedkern`, `\FBthickkern` and `\FBthousandsep` as Unicode characters.

```

1876 \ifFBucsNBSP
1877   \renewcommand*\FBmedkern{\char"202F\relax}%
1878   \renewcommand*\FBthickkern{\char"A0\relax}%
1879   \ifFBThinSpaceInFrenchNumbers
1880     \renewcommand*\FBthousandsep{\char"202F\relax}%
1881   \else
1882     \renewcommand*\FBthousandsep{\char"A0\relax}%
1883   \fi
1884 \fi

```

Finally, a warning is issued with pdfLaTeX when OT1 encoding is in use at the `\begin{document}`; mind that `\encodingdefault` is defined as 'long', the test would fail if `\FBOTone` was defined with `\newcommand*`!

```

1885 \begingroup
1886   \newcommand{\FBOTone}{OT1}%
1887   \ifx\encodingdefault\FBOTone
1888     \FBWarning{OT1 encoding should not be used for French.%
1889               \MessageBreak
1890               Add \protect\usepackage[T1]{fontenc} to the
1891               preamble\MessageBreak of your document; reported}%
1892   \fi
1893 \endgroup
1894 }

```

2.12 French lists

`\listFB` Vertical spacing in lists should be shorter in French texts than the defaults provided by `\listORI` LaTeX. Note that the easy way, just changing values of vertical spacing parameters `\FB@listVsettings`

when entering French and restoring them to their defaults on exit would not work; so we define the command `\FB@listVsettings` to hold the settings to be used by the French variant `\listFB` of `\list`. Note that switching to `\listFB` reduces vertical spacing in *all* environments built on `\list`: `itemize`, `enumerate`, `description`, but also `abstract`, `quotation`, `quote` and `verse`...

The amount of vertical space before and after a list is given by `\topsep` + `\parskip` (+ `\partopsep` if the list starts a new paragraph). IMHO, `\parskip` should be added *only* when the list starts a new paragraph, so I subtract `\parskip` from `\topsep` and add it back to `\partopsep`; this will normally make no difference because `\parskip`'s default value is `0pt`, but will be noticeable when `\parskip` is *not* null.

```
1895 \let\listORI\list
1896 \let\endlistORI\endlist
1897 \def\FB@listVsettings{%
1898     \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1899     \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
1900     \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1901     \setlength{\partopsep}{0.4ex plus 0.2ex minus 0.2ex}%
```

`\parskip` is of type 'skip', its mean value only (*not the glue*) should be subtracted from `\topsep` and added to `\partopsep`, so convert `\parskip` to a 'dimen' using `@tempdima`.

```
1902     \@tempdima=\parskip
1903     \addtolength{\topsep}{-\@tempdima}%
1904     \addtolength{\partopsep}{\@tempdima}%
1905 }
1906 \def\listFB#1#2{\listORI{#1}\FB@listVsettings #2}}
1907 \let\endlistFB\endlist
```

Let's now consider French itemize-lists. They differ from those provided by the standard LaTeX classes:

- The '•' is never used in French itemize-lists, an emdash '—' or an en-dash '–' is preferred for all levels. The item label to be used in French is stored in `\FrenchLabelItem`, it defaults to '—' and can be changed using `\frenchsetup{}` (see section 2.11).
- Vertical spacing between items, before and after the list, should be *null* with *no glue* added;
- In French the labels of itemize-lists are vertically aligned as shown p. 6.

`\FrenchLabelItem` Default labels for French itemize-lists (same label for all levels):

```
\Frlabelitemi 1908 \newcommand*\FrenchLabelItem{\textemdash}
\Frlabelitemii 1909 \newcommand*\Frlabelitemii{\FrenchLabelItem}
\Frlabelitemiii 1910 \newcommand*\Frlabelitemiii{\FrenchLabelItem}
\Frlabelitemiv 1911 \newcommand*\Frlabelitemiv{\FrenchLabelItem}
1912 \newcommand*\Frlabelitemiv{\FrenchLabelItem}
```

`\listindentFB` Let's define four dimens `\listindentFB`, `\descindentFB`, `\labelindentFB` and `\descindentFB` `\labelwidthFB` to customise lists' horizontal indentations. They are given silly `\labelindentFB` `\labelwidthFB`

negative values here in order to eventually enable their customisation in the preamble. They will get reasonable defaults later when entering French (see `\setlabelitemsFB` and `\setlistindentFB`) unless they have been customised.

```

1913 \newdimen\listindentFB
1914 \setlength{\listindentFB}{-1pt}
1915 \newdimen\descindentFB
1916 \setlength{\descindentFB}{-1pt}
1917 \newdimen\labelindentFB
1918 \setlength{\labelindentFB}{-1pt}
1919 \newdimen\labelwidthFB
1920 \setlength{\labelwidthFB}{-1pt}

```

`\leftmarginFB` `\FB@listHsettings` holds the new horizontal settings chosen for French lists `\FB@listHsettings` `itemize`, `enumerate` and `description` (two possible layouts).

```

1921 \newdimen\leftmarginFB
1922 \def\FB@listHsettings{%
1923   \ifFBListItemsAsPar

```

Optional layout: lists' items are typeset as paragraphs with indented labels.

```

1924   \itemindent=\labelindentFB
1925   \advance\itemindent by \labelwidthFB
1926   \advance\itemindent by \labelsep
1927   \leftmargini\z@
1928   \bbl@for\FB@dp {2, 3, 4, 5, 6}%
1929   {\csname leftmargin\romannumeral\FB@dp\endcsname=\labelindentFB}%
1930   \else

```

Default layout: labels hanging into the left margin.

```

1931   \leftmarginFB=\labelwidthFB
1932   \advance\leftmarginFB by \labelsep
1933   \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1934   {\csname leftmargin\romannumeral\FB@dp\endcsname=\leftmarginFB}%
1935   \advance\leftmargini by \listindentFB
1936   \fi
1937   \leftmargin=\csname leftmargin\ifnum\@listdepth=\@ne i\else
1938   ii\fi\endcsname
1939 }

```

`\itemizeFB` New environment for French `itemize`-lists.

`\FB@itemizesettings` `\FB@itemizesettings` does two things: first suppress all vertical spaces including glue unless option `StandardListSpacing` is set, then set horizontal indentations according to `\FB@listHsettings` unless option `ListOldLayout` is `true` (compatibility with lists up to v. 2.5k).

```

1940 \def\FB@itemizesettings{%
1941   \ifFBStandardListSpacing
1942   \else
1943     \setlength{\itemsep}{\z@}%
1944     \setlength{\parsep}{\z@}%
1945     \setlength{\topsep}{\z@}%
1946     \setlength{\partopsep}{\z@}%

```

```

1947     \@tempdima=\parskip
1948     \addtolength{\topsep}{-\@tempdima}%
1949     \addtolength{\partopsep}{\@tempdima}%
1950   \fi
1951   \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
1952   \ifFBListOldLayout
1953     \setlength{\leftmargin}{\labelwidth}%
1954     \addtolength{\leftmargin}{\labelsep}%
1955     \addtolength{\leftmargin}{\parindent}%
1956   \else
1957     \FB@listHsettings
1958   \fi
1959 }

```

The definition of `\itemizeFB` follows the one of `\itemize` in standard LaTeX classes (see `ltxlists.dtx`), spaces are customised by `\FB@itemizesettings`.

```

1960 \def\itemizeFB{%
1961   \ifnum \@itemdepth >\thr@@\@toodeep\else
1962     \advance\@itemdepth by \@ne
1963     \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
1964     \expandafter
1965     \listORI
1966     \csname\@itemitem\endcsname
1967     \FB@itemizesettings
1968   \fi
1969 }
1970 \let\enditemizeFB\endlistORI

1971 \def\setlabelitemsFB{%
1972   \let\labelitemi\Frlabelitemi
1973   \let\labelitemii\Frlabelitemii
1974   \let\labelitemiii\Frlabelitemiii
1975   \let\labelitemiv\Frlabelitemiv
1976   \ifdim\labelwidthFB<\z@
1977     \settowidth{\labelwidthFB}{\FrenchLabelItem}%
1978   \fi
1979 }
1980 \def\setlistindentFB{%
1981   \ifdim\labelindentFB<\z@
1982     \ifdim\parindent=\z@
1983       \setlength{\labelindentFB}{1.5em}%
1984     \else
1985       \setlength{\labelindentFB}{\parindent}%
1986     \fi
1987   \fi
1988   \ifdim\listindentFB<\z@
1989     \ifdim\parindent=\z@
1990       \setlength{\listindentFB}{1.5em}%
1991     \else
1992       \setlength{\listindentFB}{\parindent}%
1993     \fi
1994   \fi

```

```

1995 \ifdim\descindentFB<\z@
1996   \ifFBListItemsAsPar
1997     \setlength{\descindentFB}{\labelindentFB}%
1998   \else
1999     \setlength{\descindentFB}{\listindentFB}%
2000   \fi
2001 \fi
2002 }

```

\enumerateFB The definition of `\enumerateFB`, new to version 2.6a, follows the one of `\enumerate` in standard LaTeX classes (see `ltxlists.dtx`), vertical spaces are customised (or not) via `\list` ($=\code{\listFB}$ or $\code{\listORI}$) and horizontal spaces (left margins) are borrowed from `itemize` lists via `\FB@listHsettings`.

```

2003 \def\enumerateFB{%
2004   \ifnum \@enumdepth >\thr@@\toodeep\else
2005     \advance\@enumdepth by \@ne
2006     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2007     \expandafter
2008     \list
2009       \csname label\@enumctr\endcsname
2010       {\FB@listHsettings
2011         \usecounter\@enumctr\def\makeLabel##1{\hss\llap{##1}}}%
2012   \fi
2013 }
2014 \let\endenumerateFB\endlistORI

```

\descriptionFB Same tuning for the `description` environment (see `classes.dtx` for the original definition). Customisable `\descindentFB`, which defaults to `\listindentFB`, is added to `\itemindent` (first level only). When `\descindentFB=0pt` (1st level labels start at the left margin), `\leftmargini` is reduced to `\listindentFB` instead of `\listindentFB + \leftmarginFB`.

When option `ListItemsAsPar` is turned to `true`, the `description` items are also displayed as paragraphs; `\descindentFB=0pt` can be used to push labels to the left margin.

```

2015 \def\descriptionFB{%
2016   \list{}{\FB@listHsettings
2017     \labelwidth=\z@
2018     \ifFBListItemsAsPar
2019       \itemindent=\descindentFB
2020     \else
2021       \itemindent=-\leftmargin
2022     \ifnum\@listdepth=1
2023       \ifdim\descindentFB=\z@
2024         \ifdim\listindentFB>\z@
2025           \leftmargini=\listindentFB
2026         \leftmargin=\leftmargini
2027         \itemindent=-\leftmargin
2028       \fi
2029     \else
2030       \advance\itemindent by \descindentFB

```

```

2031             \fi
2032             \fi
2033             \fi
2034             \let\makelabel\descriptionlabel}%
2035 }
2036 \let\enddescriptionFB\endlistORI

```

`\update@frenchlists` `\update@frenchlists` will set up lists according to the final options (default or part of `\frenchsetup{}` eventually overruled in `\FBprocess@options`).

```

2037 \def\update@frenchlists{%
2038   \setlistindentFB
2039   \ifFBStandardListSpacing
2040   \else \let\list\listFB \fi
2041   \ifFBStandardItemizeEnv
2042   \else \let\itemize\itemizeFB \fi
2043   \ifFBStandardItemLabels
2044   \else \setlabelitemsFB \fi
2045   \ifFBStandardEnumerateEnv
2046   \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2047 }

```

If `GlobalLayoutFrench=true`, nothing has to be done at language's switches regarding lists. Otherwise, `\extrasfrench` saves the standard settings for lists and then executes `\update@frenchlists`. In both cases, there is nothing to do for lists in `\noextrasfrench`.

In order to ensure compatibility with packages customising lists, the command `\update@frenchlists` should not be included in the first call to `\extrasfrench` which occurs *before* the relevant flags are finally set, so we define `\FB@ufl` as `\relax`, it will be redefined later 'AtBeginDocument' by `\FBprocess@options` as `\update@frenchlists`, see p. 63.

```

2048 \def\FB@ufl{\relax}
2049 \def\bbl@frenchlistlayout{%
2050   \ifFBGlobalLayoutFrench
2051   \else
2052     \babel@save\list           \babel@save\itemize
2053     \babel@save\enumerate     \babel@save\description
2054     \babel@save\labelitemi    \babel@save\labelitemii
2055     \babel@save\labelitemiii  \babel@save\labelitemiv
2056     \FB@ufl
2057   \fi
2058 }
2059 \addto\extrasfrench{\bbl@frenchlistlayout}

```

2.13 French indentation of sections

`\bbl@frenchindent` `\bbl@frenchindent` In French the first paragraph of each section should be indented, this is another difference with US-English. This is controlled by the flag `\if@afterindent`. We will need to save the value of the flag `\if@afterindent` 'AtBeginDocument' before eventually changing its value.


```

2060 \def\bbl@frenchindent{%
2061   \ifFBGlobalLayoutFrench
2062   \else
2063     \babel@save\@afterindentfalse
2064   \fi
2065   \ifFBIndentFirst
2066     \let\@afterindentfalse\@afterindenttrue
2067     \@afterindenttrue
2068   \fi}
2069 \def\bbl@nonfrenchindent{%
2070   \ifFBGlobalLayoutFrench
2071     \ifFBIndentFirst
2072       \@afterindenttrue
2073     \fi
2074   \fi}
2075 \addto\extrasfrench{\bbl@frenchindent}
2076 \addto\noextrasfrench{\bbl@nonfrenchindent}

```

2.14 Formatting footnotes

The `bigfoot` package deeply changes the way footnotes are handled. When `bigfoot` is loaded, we just warn the user that `babel-french` will drop the customisation of footnotes.

The layout of footnotes is controlled by two flags `\ifFBAutoSpaceFootnotes` and `\ifFBFrenchFootnotes` which are set by options of `\frenchsetup{}` (see section 2.11). The layout of footnotes *does not depend* on the current language (just think of two footnotes on the same page looking different because one was called in a French part, the other one in English!).

We save the original definition of `\@footnotemark` at the `\begin{document}` in order to include any customisation that packages might have done; we define a variant `\@footnotemarkFB` which just adds a thin space before the number or symbol calling a footnote (any space typed in is removed first). The choice between the two definitions (valid for the whole document) is controlled by flag `\ifFBAutoSpaceFootnotes`.

```

2077 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2078   {\PackageInfo{french.ldb}%
2079     {bigfoot package in use.\MessageBreak
2080     babel-french will NOT customise footnotes;%
2081     \MessageBreak reported}}%
2082   {\let\@footnotemarkORI\@footnotemark
2083     \def\@footnotemarkFB{\leavevmode\unskip\unkern
2084       \,\@footnotemarkORI}%
2085     \ifFBAutoSpaceFootnotes
2086       \let\@footnotemark\@footnotemarkFB
2087     \fi}%
2088   }

```

`\@makefntextFB` We then define `\@makefntextFB`, a variant of `\@makefntext` which is responsible for the layout of footnotes, to match the specifications of the French ‘Imprimerie Nationale’: footnotes will be indented by `\parindentFFN`, numbers (if any) typeset on

the baseline (instead of superscripts), right aligned on `\parindentFFN` and followed by a dot and an half quad kern. Whenever symbols are used to number footnotes (as in `\thanks` for instance), we switch back to the standard layout (the French layout of footnotes is meant for footnotes numbered by arabic or roman digits).

The value of `\parindentFFN` will be redefined at the `\begin{document}`, as the maximum of `\parindent` and `1.5em` *unless* it has been set in the preamble (the weird value `10in` is just for testing whether `\parindentFFN` has been set or not).

```
2089 \newdimen\parindentFFN
2090 \parindentFFN=10in
```

`\FBfnindent` will be set ‘AtBeginDocument’ to the width of the box holding the footnote mark, `\dotFFN` and `\kernFFN` (flushed right). It is used by memoir and koma-script classes.

```
2091 \newcommand*{\dotFFN}{.}
2092 \newcommand*{\kernFFN}{\kern .5em}
2093 \newdimen\FBfnindent
```

`\@makefntextFB`’s definition is now tuned according to the document’s class for better compatibility.

Koma-script classes provide `\deffootnote`, a handy command to customise the footnotes’ layout (see English manual `scrguien.pdf`); it redefines `\@makefntext` and `\@@makefnmark`. First, save the original definitions.

```
2094 \ifFB@koma
2095   \let\@makefntextORI\@makefntext
2096   \let\@@makefnmarkORI\@@makefnmark
```

`\@makefntextFB` and `\@@makefnmarkFB` will be used when option **FrenchFootnotes** is **true**.

```
2097 \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
2098           {\thefootnotemark\dotFFN\kernFFN}
2099 \let\@makefntextFB\@makefntext
2100 \let\@@makefnmarkFB\@@makefnmark
```

`\@makefntextTH` and `\@@makefnmarkTH` are meant for the `\thanks` command used by `\maketitle` when **FrenchFootnotes** is **true**.

```
2101 \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2102           {\textsuperscript{\thefootnotemark}}
2103 \let\@makefntextTH\@makefntext
2104 \let\@@makefnmarkTH\@@makefnmark
```

Restore the original definitions.

```
2105 \let\@makefntext\@makefntextORI
2106 \let\@@makefnmark\@@makefnmarkORI
2107 \fi
```

Definitions for the memoir class:

```
2108 \@ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2109   {\newcommand{\@makefntextFB}[1]{%
2110     \def\footscript##1{##1\dotFFN\kernFFN}%
2111     \setlength{\footmarkwidth}{\FBfnindent}%
```

```

2112     \setlength{\footmarksep}{-\footmarkwidth}%
2113     \setlength{\footparindent}{\parindentFFN}%
2114     \makefootmark #1}%
2115   }{}

```

Definitions for the beamer class:

```
2116 \@ifclassloaded{beamer}
```

(see original definition in beamerbaseframecomponents.sty), note that for the beamer class footnotes are LR-boxes, not paragraphs, so \parindentFFN is irrelevant. class.

```

2117   {\def\@makefntextFB#1{%
2118     \def\insertfootnotetext{#1}%
2119     \def\insertfootnotemark{\insertfootnotemarkFB}%
2120     \usebeamertemplate***{footnote}}%
2121   \def\insertfootnotemarkFB{%
2122     \usebeamercolor[fg]{footnote mark}%
2123     \usebeamerfont*{footnote mark}%
2124     \llap{\@thefnmark}\dotFFN\kernFFN}%
2125   }{}

```

Now the default definition of \@makefntextFB for standard LaTeX and AMS classes. The next command prints the footnote mark according to the specifications of the French 'Imprimerie Nationale'. Keep in mind that \@thefnmark might be empty (i.e. in AMS classes' titles)!

```

2126 \providecommand*\insertfootnotemarkFB}{%
2127   \parindent=\parindentFFN
2128   \rule\z@\footnotesep
2129   \setbox\@tempboxa\hbox{\@thefnmark}%
2130   \ifdim\wd\@tempboxa>\z@
2131     \llap{\@thefnmark}\dotFFN\kernFFN
2132   \fi}
2133 \providecommand\@makefntextFB[1]{\insertfootnotemarkFB #1}

```

The rest of \@makefntext's customisation is done at the \begin{document}. We save the original definition of \@makefntext, and then redefine \@makefntext according to the value of flag \ifFBFrenchFootnotes (true or false). Koma-script classes require a special treatment.

The LuaTeX command \localleftbox and \FBeverypar@quote used by \frquote{} have to be reset inside footnotes; done for LaTeX based formats only.

```

2134 \providecommand\localleftbox[1]{}
2135 \AtBeginDocument{%
2136   \@ifpackageloaded{bigfoot}{}%
2137   {\ifdim\parindentFFN<10in
2138     \else
2139       \parindentFFN=\parindent
2140       \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2141     \fi
2142     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2143     \addtolength{\FBfnindent}{\parindentFFN}%
2144     \let\@makefntextORI\@makefntext

```

2145 \ifFB@koma

Definition of \@makefntext for koma-script classes: running makefntextORI inside a group to reset \localleftbox{} and \FBeverypar@quote would mess up the layout of footnotes whenever the first mandatory argument of \deffootnote{} (used as \leftskip) is non-nil (default is 1em, 0pt in French).

```
2146           \let\@makefntextORI\@makefntextORI
2147           \long\def\@makefntext#1{%
2148           \localleftbox{}%
2149           \let\FBEverypar@save\FBEverypar@quote
2150           \let\FBEverypar@quote\relax
2151           \ifFBFrenchFootnotes
2152           \ifx\footnote\thanks
2153           \let\@makefntextTH\@makefntextTH
2154           \@makefntextTH{#1}
2155           \else
2156           \let\@makefntextFB\@makefntextFB
2157           \@makefntextFB{#1}
2158           \fi
2159           \else
2160           \let\@makefntextORI\@makefntextORI
2161           \@makefntextORI{#1}%
2162           \fi
2163           \let\FBEverypar@quote\FBEverypar@save
2164           \localleftbox{\FBEveryline@quote}}%
2165           \else
```

Special add-on for the memoir class: \maketitle redefines \@makefntext as \makethanksmark which is customised as follows to match the other notes' vertical alignment.

```
2166           \@ifclassloaded{memoir}%
2167           {\ifFBFrenchFootnotes
2168           \setlength{\thanksmarkwidth}{\parindentFFN}%
2169           \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2170           \fi
2171           }}%
```

Special add-on for the beamer class: issue a warning in case \parindentFFN has been changed.

```
2172           \@ifclassloaded{beamer}%
2173           {\ifFBFrenchFootnotes
2174           \ifdim\parindentFFN=1.5em\else
2175           \FBWarning{%
2176           \protect\parindentFFN\space is ineffective%
2177           \MessageBreak within the beamer class.%
2178           \MessageBreak Reported}%
2179           \fi
2180           \fi
2181           }}%
```

Definition of \@makefntext for all other classes:

```
2182           \long\def\@makefntext#1{%
```

```

2183         \lcalleftbox{}}%
2184         \let\FBeverypar@save\FBeverypar@quote
2185         \let\FBeverypar@quote\relax
2186         \ifFBFrenchFootnotes
2187             \@makefntextFB{#1}%
2188         \else
2189             \@makefntextORI{#1}%
2190         \fi
2191         \let\FBeverypar@quote\FBeverypar@save
2192         \lcalleftbox{\FBeveryline@quote}}}%
2193     \fi
2194 }%
2195 }

```

For compatibility reasons, we provide definitions for the commands dealing with the layout of footnotes in babel-french version 1.6. `\frenchsetup{}` (see in section 2.11) should be preferred for setting these options. `\StandardFootnotes` may still be used locally (in minipages for instance), that's why the test `\ifFBFrenchFootnotes` is done inside `\@makefntext`.

```

2196 \newcommand*\AddThinSpaceBeforeFootnotes{\FBAutoSpaceFootnotestru}
2197 \newcommand*\FrenchFootnotes{\FBFrenchFootnotestru}
2198 \newcommand*\StandardFootnotes{\FBFrenchFootnotesfalse}

```

2.15 Clean up and exit

Final cleaning. The macro `\ldf@finish` takes care for setting the main language to be switched on at `\begin{document}` and resetting the category code of `@` to its original value. `\loadlocalcfg` is redefined locally in order not to load any `.cfg` file for French.

```

2199 \FBclean@on@exit
2200 \ldf@finish\CurrentOption
2201 \let\loadlocalcfg\FB@llc
2202 </french>

```

2.16 Files frenchb.ldf, francais.ldf, canadien.ldf and acadian.ldf

Babel now expects a `<lang>.ldf` file for each `<lang>`. So we create portmanteau `.ldf` files for options `canadien`, `francais`, `frenchb` and `acadian`. These files themselves only load `french.ldf` which does the real work. Warn users about options `canadien`, `frenchb` and `francais` being deprecated and force recommended options `acadian` or `french`.

```

2203 <*acadian>
2204 \PackageInfo{acadian.ldf}%
2205 {'acadian' dialect is currently\MessageBreak
2206  *absolutely identical* to the\MessageBreak
2207  'french' language; reported}
2208 </acadian>
2209 <*canadien>

```

```

2210 \PackageWarning{canadien.ldf}%
2211 {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2212 it might be removed sooner or later. Please\MessageBreak
2213 use 'acadian' instead; reported}%
2214 \let\l@canadien\l@acadian
2215 \def\CurrentOption{acadian}
2216 </canadien>
2217 <*français>
2218 \PackageWarning{français.ldf}%
2219 {Option 'français' for Babel is *deprecated*,\MessageBreak
2220 it might be removed sooner or later. Please\MessageBreak
2221 use 'french' instead; reported}%
2222 \let\l@français\l@french
2223 \def\CurrentOption{french}
2224 </français>

```

Compatibility code for babel pre-3.13: frenchb.ldf could be loaded with options acadian, canadien, frenchb or français.

```

2225 <*frenchb>
2226 \def\bbl@tempa{frenchb}
2227 \ifx\CurrentOption\bbl@tempa
2228 \let\l@frenchb\l@french
2229 \def\CurrentOption{french}
2230 \PackageWarning{babel-french}%
2231 {Option 'frenchb' for Babel is *deprecated*,\MessageBreak
2232 it might be removed sooner or later. Please\MessageBreak
2233 use 'french' instead; reported}
2234 \else
2235 \def\bbl@tempa{français}
2236 \ifx\CurrentOption\bbl@tempa
2237 \let\l@français\l@french
2238 \def\CurrentOption{french}

```

Plain formats: no warning when français.sty loads frenchb.ldf (babel pre-3.13).

```

2239 \ifx\magnification@undefined
2240 \PackageWarning{babel-french}%
2241 {Option 'français' for Babel is *deprecated*,\MessageBreak
2242 it might be removed sooner or later. Please\MessageBreak
2243 use 'french' instead; reported}%
2244 \fi
2245 \else
2246 \def\bbl@tempa{canadien}
2247 \ifx\CurrentOption\bbl@tempa
2248 \let\l@canadien\l@acadian
2249 \def\CurrentOption{acadian}
2250 \PackageWarning{babel-french}%
2251 {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2252 it might be removed sooner or later. Please\MessageBreak
2253 use 'acadian' instead; reported}
2254 \fi
2255 \fi
2256 \fi

```

```
2257 </frenchb>
2258 <acadian|canadien|frenchb|français>\input french.lda\relax
2259 <acadian|canadien>\let\extrasacadian\extrasfrench
2260 <acadian|canadien>\let\noextrasacadian\noextrasfrench
```

3 Change History

Changes are listed in reverse order (latest first) and limited to babel-french v3.

v3.5e		items “as paragraphs”.	54
	<code>\frenchsetup: StandardLayout</code> and <code>GlobalLayoutFrench</code> options can no longer be toggled when French is not the main language.		55
	<code>\frquote: Make resettings global on exit.</code>		40
	new command <code>\NoEveryParQuote</code> . reset <code>\FB@addGUIlSpace</code> attribute inside <code>\localleftbox</code> (LuaTeX).		40
			39
v3.5d			
	General: Add <code>\frquote</code> to redefinitions for bookmarks.		66
	<code>\frenchsetup: ReduceListSpacing</code> option deprecated: see <code>StandardListSpacing</code>		54
v3.5c			
	General: Remove grouping inside <code>\@makefntext</code> , <code>\localleftbox</code> and <code>\FBeverypar@quote</code> saved and restored instead.		75
	<code>\frquote: \FBeverypar@quote</code> ’s value now properly reset across level changes.		39
	<code>\noextrasfrench: \lccode</code> of quote 0x27 changed from 0x2019 to 0x27 for Unicode engines.		16
v3.5b			
	General: Reset <code>\FBeverypar@quote</code> locally inside <code>\@makefntext</code> . Needed by <code>\frquote</code>		75
	<code>\frquote: New command <code>\FB@addquote@everypar</code> to manage <code>\everypar: \frquote</code> failed when used immediately after a sectioning command.</code>		38
v3.5a			
	General: New optional layout for lists: lists’ items can be typeset as paragraphs with indented labels while the default leaves the labels hanging into the left margin.		69
	<code>\descriptionFB: ListltemsAsPar</code> option taken into account for description lists.		71
	<code>\frenchsetup: New option <code>ListltemsAsPar</code> for displaying lists’</code>		
v3.4d			
	<code>\frenchsetup: New test for deciding about utf8 encoding for keys og and fg</code> (the former one fails with LaTeX 2018 release).		60
v3.4c			
	<code>\ifFBXeTeX: Reverting to former test, beware of <code>\XeTeXrevision</code> left as <code>\relax</code> by careless testing.</code>		16
v3.4b			
	<code>\datefrench: Do not redefine <code>\date</code> as <code>\frenchdate</code> in French.</code>		40
v3.4a			
	General: <code>\LdfInit</code> checks <code>\FBclean@on@exit</code> instead of <code>\captionfrench</code> (undefined in PLain). Prevents loading <code>french.ldf</code> again with <code>acadian</code> option.		14
	babel-french now requires eTeX.		14
	Lua function <code>token.get_meaning</code> requires LuaTeX 1.0.		21
	New <code>\FBgspchar</code> to customise the space character to be used for <code>\og</code> and <code>\fg</code> with the <code>UnicodeNoBreakSpaces</code> option.		36
	New attribute <code>\FB@dialect</code> for the French dialect <code>acadian</code>		20
	New command <code>\FBsetspaces</code> to fine tune spacing independently in French and in French dialects.		18
	<code>Shrink/stretch</code> removed in <code>\FBthousandsep</code>		47
	Toks <code>\FBcolonsp</code> , <code>\FBthinsp</code> and <code>\FBguillsp</code> removed.		18
	<code>frenchb.lua: Global ‘FBsp’ table added; local function ‘get_glue’ changed into global ‘FBget_glue’.</code>		23
	<code>\datefrench: Specific code for Plain finally removed</code> (babel bug reported).		40
	<code>\extrasfrench: Change <code>\(no)extras\CurrentOption</code> to <code>\(no)extrasfrench</code>. <code>\(no)extrasacadian</code> will be defined as <code>\(no)extrasfrench</code> in file <code>acadian.ldf</code>.</code>		16

<ul style="list-style-type: none"> <ul style="list-style-type: none"> \frenchsetup: Patch for koma-script classes moved here, after \ifFBPartNameFull is defined, so that it applies to \extrasacadian too: \AtEndOfPackage is too late. 55 	<ul style="list-style-type: none"> <ul style="list-style-type: none"> \frenchpartsecond and \frenchpartnameord added. 47
<ul style="list-style-type: none"> v3.3d <ul style="list-style-type: none"> frenchb.lua: In default mode, for ‘:’ only, check if next node is a glyph or not. If it is, turn the ‘auto’ flag to false (avoids spurious spaces in URLs, MSDOS paths or 10:35). 25 	<ul style="list-style-type: none"> <ul style="list-style-type: none"> \FBthinspace: Skips \FBcolonskip and \FBthinspace replaced by toks \FBcolonsp and \FBthinsp. 17 <ul style="list-style-type: none"> \frenchsetup: \frenchbsetup is now an alias for \frenchsetup. 54 Options INGullSpace, ThinColonSpace no longer delayed AtBeginDocument. 54 <ul style="list-style-type: none"> \frquote: \FB@quotespace (kern), changed into \FB@guillspace. 39
<ul style="list-style-type: none"> v3.3c <ul style="list-style-type: none"> General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, fontspec is no longer required. 67 New command \FBthousandsep to customise numprint. 47 New configurable kerns \FBmedkern, and \FBthickkern suitable for HTML translation. 43 Reorganise warnings when the caption, subcaption or floatrow packages are loaded before babel/french. 50 Reset \localleftbox locally inside \@makefntext. Needed by \frquote with LuaTeX. 75 frenchb.lua: Function ‘get_glue’ robustified. ‘french_punctuation’ can insert Unicode characters instead of glues. 22 \frenchsetup: New option ‘UnicodeNoBreakSpaces’ for html translators (LuaLaTeX only). 59 	<ul style="list-style-type: none"> v3.2h <ul style="list-style-type: none"> \@makefntextFB: With beamer.cls, add \llap to \@thefnmark for notes numbered over 99. 75 \bbl@frenchlistlayout: Execute \update@frenchlists only if GlobalLayoutFrench is false. Delete stuff for lists in \noextrasfrench. 72 \frenchsetup: Option GlobalLayoutFrench skipped when French is not the main language. 55
<ul style="list-style-type: none"> v3.3b <ul style="list-style-type: none"> General: Generate portmanteau files acadian.ldf, canadien.ldf, frenchb.ldf, and francais.ldf and warn about deprecated options. 77 New ‘if’ \ifFBfrench to replace \iflanguage test which is based on patterns. 16 	<ul style="list-style-type: none"> v3.2g <ul style="list-style-type: none"> General: Add \boi to redefinitions for bookmarks. 66 Changed Unicode definition of \boi. 44 fontspec defines TU encoding now and no longer loads xunicode.sty. Test changed. 67 Issue a warning if beamerarticle.sty is loaded after babel. 53 \frenchsetup: Minimal list customisation when beamerarticle.sty is loaded. 55 Warn when wrong values are provided to options EveryParGuill or EveryLineGuill. 59 \frquote: Default options of \frquote are no longer engine-dependent. 38
<ul style="list-style-type: none"> v3.3a <ul style="list-style-type: none"> General: Compatibility code for pre 2015/10/01 LaTeX release removed, see ltnews23.tex. 20 Skip \FBguillskip for LuaTeX replaced by toks \FBguillsp. 18 \captionsfrench: Commands \frenchpartfirst, 	<ul style="list-style-type: none"> v3.2f <ul style="list-style-type: none"> \DecimalMathComma: Fixed conflict with the icomma package. 45 v3.2e <ul style="list-style-type: none"> General: Add missing redefinitions for \leftmarginiv, \leftmarginvi. Suggested by J.F. Burnol. 69

\DecimalMathComma:		characters: it's value will be 4095	
\DecimalMathComma didn't work		for new engines and 255 for older	
with LuaTeX. Fixed now.	45	ones.	17
v3.2d		\NoAutoSpacing: \NoAutoSpacing	
\descriptionFB: Changed		made robust.	36
\listindentFB to		v3.2a	
\descindentFB which defaults to		\@makefnstextFB: beamer.cls requires	
\listindentFB. \leftmargini		a specific definition of	
reduced when \descindentFB is		\@makefnstextFB (pointed out by	
null.	71	DB). The same is true for memoir	
v3.2c		and koma-script classes (done). .	74
General: New LuaTeX attribute		\fg: \xspace moved from \FB@fg to	
\FB@spacing.	20	\fg: \xspace messes up	
Newif \ifFB@spacing and new		\frquote, pointed out by Sonia	
commands \FB@spacingon,		Labetoulle. As a side effect	
\FB@spacingoff to control space		\xspace is now active in \fg in	
tuning in French.	20	and outside French.	37
Switch \ifFB@spacing added to		v3.1m	
the four French shorthands.	33	frenchb.lua: new_glue_scaled	
\FB@xetex@punct@french: Switch		returns nil in case of invalid font	
\ifFB@spacing added to all		table (i.e. lcircle1.pfb). In such	
\XeTeXinterchartoks		cases babel-french leaves the	
commands.	31	node list unchanged.	24
\FBthinspace: Change .16667em to		v3.1l	
.5\fontdimen2\font to get in		General: Add a variant of	
XeTeX and pdfTeX the same		\babel@savevariable to save	
spacing as in LuaTeX.	17	\XeTeXcharclass(es) in a loop. .	31
\frenchsetup: Add a warning about		frenchb.lua: font.getfont(fid)	
options og/fg for old XeTeX or		possibly returns nil even for a	
LuaTeX engines requiring active		positive fid (i.e. AMS lcircle1.pfb).	
characters.	60	Reported by François Legendre. .	24
\NoAutoSpacing: New definition		\FB@luatex@punct@french: Use	
based on \FB@spacing@off		\babel@save to save and restore	
common to all engines.	36	\shorthandon and	
\ttfamilyFB: New definitions of		\shorthandoff.	29
\ttfamilyFB and co, common to		\FB@xetex@punct@french: Save and	
all engines, based on		restore	
\FB@spacing@off		\XeTeXinterchartokenstate,	
and\FB@spacing@on.	35	\shorthandon, \shorthandoff	
v3.2b		using \babel@savevariable and	
General: Load ltuatex.tex for plain		\babel@save,	
LuaTeX to ensure \newattribute		\XeTeXcharclass(es) using	
is defined.	20	\FB@savevariable@loop.	31
Warning added when the		v3.1k	
subcaption package is loaded		General: (pdfTeX shorthands) test on	
before babel/french.	50	\lastskip changed from 0pt to	
frenchb.lua: glue_spec removed;		1sp for active punctuation for	
starting with LuaTeX 0.95, glue		consistency with XeTeX and	
specifications fit in glue.	24	LuaTeX.	33
\ifFB@xetex@punct: New counter		\FB@xetex@punct@french: Thin	
\FB@nonchar needed for non		glues (less than 1sp) should not	

trigger space insertion before high punctuation. Add a check on <code>\lastkip</code>	31	<code>\hboxes</code>	23
v3.1j		<code>\captionsfrench</code> : <code>\partname</code> 's definition depends now on flag <code>PartNameFull</code> . No need to redefine it in <code>\frenchbsetup</code>	47
General: Loading <code>luatexbase.sty</code> is no longer needed with LaTeX release 2015/10/01 or later.	20	<code>\frenchsetup</code> : Bug fix for koma-scripts classes: a spurious dot was added by the <code>\partformat</code> command.	55
<code>\frquote</code> : <code>\fr@quote</code> completely rewritten: <code>\leavevmode</code> added and explicitly <code>save/retore</code> <code>\everypar</code> and <code>\localleftbox</code> instead of using a group in order to ensure compatibility with package <code>wrapfig</code>	39	<code>PartNameFull</code> now just sets the flag, nothing to add to <code>\captionsfrench</code> when false.	54
<code>\PackageWarning</code> is undefined in Plain, use <code>\fb@warning</code> instead.	39	v3.1f	
v3.1i		General: <code>\FBCaption@Separator</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.	51
General: <code>\nombre</code> command changed when <code>numprint.sty</code> is not loaded: only one warning, no error.	47	<code>\FBprocess@options</code> : Bug fix for the beamer class: figure and table captions are now consistent with babel-french's documentation. Pointed out by Denis Bitouzé.	65
Remove restriction about loading <code>numprint.sty</code> after <code>babel</code>	53	Definition of <code>\captionformat</code> and <code>\captiondelim</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.	65
<code>\frquote</code> : <code>\luatexlocalleftbox</code> changed to <code>\localleftbox</code> by new LaTeX release 2015/10/01.	39	<code>\FBthinspace</code> : <code>\FBthinspace</code> is no longer a kern but a skip (babel-french adds a nobreak penalty before it).	17
v3.1h		v3.1e	
General: <code>french.cfg</code> from e-french conflicts with <code>babel-french</code> . Do NOT load it (no need for <code>.cfg</code> files with <code>babel-french</code> anyway).	77	<code>\frenchsetup</code> : Corrected typo: <code>SmallCapsFigTabcaptions</code> instead of <code>SmallCapsFigTabCaptions</code> . Pointed out by Céline Chevalier.	54
v3.1g		v3.1d	
General: Lua function <code>french_punctuation</code> is now inserted at the end of the 'kerning' callback (no priority) instead of 'hpack_filter' and 'pre_linebreak_filter'.	29	General: New section: issue warnings if packages <code>listings</code> , <code>numprint</code> and <code>natbib</code> are loaded too early or too late vs <code>babel</code>	53
Use Babel defined loops <code>\bb\@for</code> instead of <code>\@for</code> borrowed from file <code>ltntr.dtx</code> (<code>\@for</code> is undefined in Plain).	30	v3.1c	
<code>frenchb.lua</code> : Flag <code>addgl</code> set to false for '«' at the end of an <code>\hbox</code> or a paragraph or when followed by a null glue (i.e. springs).	28	<code>frenchb.lua</code> : Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André.	25
flag <code>addgl</code> set to false for '»' at the beginning of an <code>\hbox</code> or a paragraph or a tabular 'l' and 'c' columns.	27	v3.1b	
Node <code>HLIST</code> added; node <code>TEMP</code> added for the first node of		<code>frenchb.lua</code> : Add a check for null fid in <code>french_punctuation</code> (Tikz <code>\nullfont</code>). Bug pointed out by Paul Gaborit.	25

<code>\captionsfrench</code> : Change <code>\scshape</code> to customisable <code>\FBfigtabshape</code> for <code>\figurename</code> and <code>\tablename</code>	47	environment of the listings package.	25
<code>\fprimo</code>): Removed <code>\lowercase</code> from definitions of <code>\FrenchEnumerate</code> , . . . <code>\No</code> and <code>co</code> : <code>\up</code> already does the conversion.	43	<code>\frenchsetup</code> : New option <code>INGuillSpace</code>	54
<code>\frenchsetup</code> : New option <code>SmallCapsFigTabCaptions</code>	54	No list customisation when beamer class is loaded.	55
<code>\ieres</code> : Removed <code>\lowercase</code> from definitions of <code>\ieme</code> and <code>co</code> : <code>\up</code> already does the conversion.	43	v3.0b	
v3.1a		General: <code>frenchb.lua</code> was not found by Lua function <code>dofile</code> (not <code>kpathsea</code> aware). Call function <code>kpse.find_file</code> first, as suggested by Paul Gaborit.	29
General: <code>fontspec</code> is not required for T1 fonts used with the <code>luainputenc.sty</code> package.	67	Require <code>luatexbase</code> with <code>LaTeX2e</code> in case <code>fontspec</code> has not been loaded before <code>babel</code>	20
Misplaced <code>\fi</code> for plain formats.	20	v3.0a	
New command <code>\frquote</code> for imbedded or long French quotations.	38	General:	
<code>frenchb.lua</code> : Added flag <code>addgl</code> which must also be true when <code>prev</code> or <code>next</code> is not a char (i.e. <code>\kern0</code> in <code>«\texttt{a}»</code>).	27	<code>\bbl@nonfrenchguillemets</code> deleted, use <code>\babel@save</code> instead.	38
Codes <code>0x13</code> and <code>0x14</code> added for French quotes in T1-encoding.	22	<code>\LdfInit</code> checks	
Look ahead when next is a kern (i.e. in <code>«\texttt{a}»</code>).	28	<code>\captionsfrench</code> instead of <code>\datefrench</code> to avoid a conflict with <code>papertex.cls</code> which loads <code>datetime.sty</code>	14
<code>\frenchsetup</code> : Codes <code>0x13</code> and <code>0x14</code> added for French quotes in T1-encoding. Support for older versions of <code>LuaTeX</code> and <code>XeTeX</code> dropped.	60	<code>french.cfg</code> will be loaded (if found) instead of <code>frenchb.cfg</code> . NO NEED for <code>.cfg</code> files in French anyway.	77
New options <code>InnerGuillSingle</code> , <code>EveryParGuill</code> and <code>EveryLineGuill</code> to control <code>\frquote</code>	54	In Plain, provide a substitute for <code>\PackageWarning</code> and <code>\PackageInfo</code>	14
v3.0c		Merging of <code>\captionsfrenchb</code> , <code>\captionsfrançais</code> with <code>\captionsfrench</code> deleted in favor of new <code>babel 3.9</code> syntax.	49
General: <code>babel-french</code> requires <code>babel-3.9i</code>	14	More informative, less TeXnical warning about <code>\@makecaption</code>	51
Just load <code>luatexbase.sty</code> instead of <code>luaotfload.sty</code> with plain formats.	20	New flag <code>\ifFB@luatex@punct</code> for ‘high punctuation’ management with <code>LuaTeX</code> engines.	17
No need to define <code>\l@french</code> as <code>\lang@french</code> , <code>babel.def (3.9j)</code> takes care for this.	15	New handling of ‘high punctuation’ through callbacks with <code>LuaTeX</code> engines.	20
<code>frenchb.lua</code> : Null glues should not trigger space insertion before high punctuation. Bug pointed out by Benoit Rivet for the ‘ <code>Istlisting</code> ’		No warning about <code>\@makecaption</code> for SMF classes.	50
		Options processing completely reorganised, now <code>\babel@save</code> and <code>\babel@savevariable</code> are usable for French.	53
		Support for options <code>frenchb</code> , <code>français</code> , <code>canadien</code> , <code>acadian</code>	

changed.	14	Suggested by Denis Bitouzé. ...	71
Test <code>\ifXeTeX</code> changed to <code>\ifFBunicode</code> and <code>'xltextra'</code> changed to <code>'fontspec'</code>	67	<code>\extrasfrench</code> : Take advantage of babel's <code>\babel@savevariable</code> to handle apostrophe's <code>\lccode</code> . ..	16
<code>\CaptionSeparator</code> : Remove <code>\FBCaption@SeparatorORI</code> , use <code>\babel@save</code> instead.	49	<code>\FB@fg</code> : Definitions of <code>\FB@og</code> and <code>\FB@fg</code> now depend on punctuation handling (LuaTeX / XeTeX / active).	37
<code>\captionsfrench</code> : Take advantage of babel's <code>\SetString</code> commands for <code>captionnames</code>	47	<code>\FBprocess@options</code> : With koma-script and memoir class, customise <code>\captionformat</code> and <code>\captiondelim</code>	65
<code>\datefrench</code> : Take advantage of babel's <code>\SetString</code> commands for <code>\datefrench</code> . Doesn't work with Plain (yet?).	40	<code>\frenchsetup</code> : New options <code>OldFigTabCaptions</code> and <code>CustomiseFigTabCaptions</code>	54
<code>\descriptionFB</code> : Added <code>\listindentFB</code> to <code>\itemindent</code> .			