

A Babel language definition file for French

frenchb.dtx v3.5c, 2018/09/14

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	52
2.11 Setup options: keyval stuff	53
2.12 French lists	67
2.13 French indentation of sections	72
2.14 Formatting footnotes	72
2.15 Clean up and exit	77
2.16 Files frenchb.ldf, francais.ldf, canadien.ldf and acadian.ldf	77
3 Change History	79

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 v3.5c are listed in subsection 1.4 p. 11.

An extensive documentation is available in French here:

<http://daniel.flipo.free.fr/frenchb>

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 v3.5c and was last revised on 2018/09/14.

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.

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

⁵ Well, 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)” (N^o N^{os} n^o and n^{os}) 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 T_EXbook 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. This option can be used to avoid conflicts with classes or packages which customise lists or footnotes.

When French is not the main language, `StandardLayout=false` can be misused to ensure French typography (in French only). This is a *bad practice*: the document layout should not be altered by language switches.

`GlobalLayoutFrench=false (true*)` should no longer be used; it was intended to emulate, when French is the main language, what prior versions of `babel-french` (pre-2.2) did: lists, and first paragraphs of sections would be displayed the standard way in other languages than French, and “à la française” in French. 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:

<div>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...</div>	<div>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...</div>
Default French layout	With <code>ListItemsAsPar=true</code>

`ReduceListSpacing=false (true*)` ; babel-french reduces the values of the vertical spaces used in the *all* list environments in French (this includes `itemize`, `enumerate`, `description`, but also `abstract`, `quote`, `quotation` and `verse` and possibly others). Setting this option to `false` reverts to the standard settings of the `list` environment.

`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*)`

`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 `ReduceListSpacing=false`, `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.

`CompactItemize=false (true*)` ; is kept only for backward compatibility, it is replaced by `StandardItemizeEnv` and `StandardEnumerateEnv`.

`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 case ⁶, 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.

⁶Unless option `OriginalTypewriter` is set, `\ttfamily` is redefined in French to switch off space tuning, see below.

`{\NoAutoSpacing http://mysite}`⁷ or `{\NoAutoSpacing ???}` (needed for pdfTeX only).

`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.

⁷Actually, 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...`

`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 paragrah 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 (~)⁸ 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

⁸Actually without stretch nor shrink.

`\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 francais.

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, with the encoding suitable for your machine (*my-encoding* will be latin1 for Unix machines, ansinew for PCs running Windows, applemac or latin1 for Macintoshes, or utf8...

```
%%% Test file for French hyphenation.
\documentclass[french]{article}
\usepackage[my-encoding]{inputenc}
\usepackage[T1]{fontenc} % Use LM fonts
\usepackage{lmodern}      % for French
\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 sectionning command.

What’s new in version 3.4?

Version 3.4a adds a new command `\frenchdate` (see p. 40) 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 eT_EX and LuaT_EX 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 `lwarp` (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.lda`, 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 `francais` 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'⁹. 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.

⁹The 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.lda) }%
7     \errhelp{#2}\errmessage{\#\1^^J}%
8   \endgroup}
9 \def\fb@warning#1{%
10  \begingroup
11    \newlinechar='\^^J
12    \def\{\^^J(french.lda) }%
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.lda}
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 `\lefthyphenmin` (2) and `\righthyphenmin` (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 and **\ifFBXeTeX** 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

```

\ifFBfrench 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\ifFBfrench

```

\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 \ifBLaTeX
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\bbl@tempa{french}\def\bbl@tempb{#1}%
138     \ifx\bbl@tempa\bbl@tempb \def\bbl@tempb{}\fi
139     \@namedef{\bbl@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\bbl@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 `\language` 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{\language}}%
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\language FBthinspace\endcsname
177     \babel@save\FBthinspace
178     \renewcommand*{\FBthinspace}{%
179       \csname\language FBthinspace\endcsname}%
180   \fi

```

Same for `\FBcolonspace`:

```

181   \ifcsname\language FBcolonspace\endcsname
182     \babel@save\FBcolonspace
183     \renewcommand*{\FBcolonspace}{%
184       \csname\language 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 `ltxluatex.tex` if it hasn’t been loaded before babel.

```
208 \input ltxluatex.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 `\FBsetspace` commands) are taken into account. Values not explicitly set for Acadian by `\FBsetspace[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#lspace,')
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#lspace.')
```

```

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) preceeds 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                 nbspace.char = FBsp.colon.ch.ac
428             else
429                 t = FBsp.colon.gl.fr
430                 nbspace.char = FBsp.colon.ch.fr
431             end
432         else
433             if FRdialect then
434                 t = FBsp.thin.gl.ac
435                 nbspace.char = FBsp.thin.ch.ac
436             else
437                 t = FBsp.thin.gl.fr
438                 nbspace.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 preceeding '»', 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 glue))
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 preceeds 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 fbglue = new_glue_scaled(fid, t)
525     if addgl and fbglue 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(fbglue))
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\FB@colonspace
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\FB@thinspace
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 $\text{\LaTeX}2\epsilon$ `\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 `\fg` 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*{\ogii}{\textquotedblleft}
825 \newcommand*{\fgii}{\textquotedblright}

```

and the needed technical stuff to handle options:

```

826 \newcount\FBguill@level
827 \newtoks\FBbold@everypar

```

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

```

828 \def\FB@addquote@everypar{%
829 \let\FBnew@everypar\everypar
830 \FBold@everypar=\expandafter{\the\everypar}%
831 \FBnew@everypar={\the\FBold@everypar\FBeverypar@quote}%
832 \let\everypar\FBold@everypar
833 \let\FB@addquote@everypar\relax
834 }
835 \newif\ifFBcloseguill \FBcloseguilltrue
836 \newif\ifFBinnerguillsingle
837 \def\FBguillopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
838 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
839 \let\FBguillnone\empty
840 \let\FBeveryparguill\FBguillopen
841 \let\FBeverylanguill\FBguillnone
842 \let\FBeverypar@quote\relax
843 \let\FBeverylanguill@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.

```

844 \ifLaTeXe
845 \DeclareRobustCommand\frquote{%
846 \ifstar{\FBcloseguillfalse\fr@quote}%

```

```

847             {\FBcloseguilltrue\fr@quote}}
848 \else
849   \newcommand\fr@quote[1]{\fr@quote{#1}}
850 \fi

```

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

```

851 \newcommand{\fr@quote}[1]{%
852   \leavevmode
853   \advance\FBguill@level by \@ne
854   \ifcase\FBguill@level
855   \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:

```

856   \ifx\FBeveryparguill\FBguillnone
857   \else
858     \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
859     \FB@addquote@everypar
860   \fi
861   \ogi #1\fgi
862 \or

```

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

```

863   \ifx\FBeverylineguill\FBguilllopen
864     \def\FBeveryline@quote{\guillemotleft\FB@guillspace}%
865     \localleftbox{\FBeveryline@quote}%
866     \let\FBeverypar@quote\relax
867     \ogi #1\ifFBcloseguill\fgi\fi
868   \else
869     \ifx\FBeverylineguill\FBguillclose
870       \def\FBeveryline@quote{\guillemotright\FB@guillspace}%
871       \localleftbox{\FBeveryline@quote}%
872       \let\FBeverypar@quote\relax
873       \ogi #1\ifFBcloseguill\fgi\fi
874     \else

```

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

```

875     \let\FBeverypar@quote\relax
876     \ifFBInnerGuillSingle
877       \def\ogii{\leavevmode
878         \guilsinglleft\FB@guillspace}%
879       \def\fgii{\ifdim\lastskip>\z@\unskip\fi
880         \FB@guillspace\guilsinglright}%
881       \ifx\FBeveryparguill\FBguilllopen
882         \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
883       \fi
884       \ifx\FBeveryparguill\FBguillclose
885         \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
886       \fi
887     \fi

```

```

888      \ogii #1\ifFBcloseguill \fgii \fi
889      \fi
890      \fi
891      \else
Warn if \FBguill@level > 2:
892      \ifx\PackageWarning\@undefined
893      \fb@warning{\noexpand\frquote\space handles up to
894      two levels.\\ Quotation not printed.}%
895      \else
896      \PackageWarning{french.ldf}{%
897      \protect\frquote\space handles up to two levels.
898      \MessageBreak Quotation not printed. Reported}
899      \fi
900      \fi
Closing: step down \FBguill@level and clean on exit.
901      \advance\FBguill@level by \m@ne
902      \ifcase\FBguill@level \let\FBeverypar@quote\relax
903      \or \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
904      \let\FBeverylines@quote\empty
905      \ifx\FBeverylinesguill\FBguillnone\else\localleftbox{}\fi
906      \fi
907 }

```

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.

```

908 \def\BabelLanguages{french,acadian}
909 \StartBabelCommands*{\BabelLanguages}{date}
910     [unicode, fontenc=TU EU1 EU2, charset=utf8]
911     \SetString\monthiiname{février}
912     \SetString\monthviiiname{août}
913     \SetString\monthxiiname{décembre}
914 \StartBabelCommands*{\BabelLanguages}{date}
915     \SetStringLoop{month#1name}{%
916         janvier,f\'evrier,mars,avril,mai,juin,juillet,%
917         ao\^ut,septembre,octobre,novembre,d\'ecembre}
918     \SetString\today{\FB@date{\year}{\month}{\day}}
919 \EndBabelCommands

```


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

```

920 \newcommand*{\FB@date}[3]{%
921   {\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
922   \csname month\romannumeral#2name\endcsname
923   \ifx#1\@empty\else\FBdatespace\number#1\fi}}
924 \newcommand*{\FBdatebox}{\hbox}
925 \newcommand*{\FBdatespace}{\space}
926 \newcommand*{\frenchdate}{\FBdatebox\FB@date}
927 \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).

```

928 \newif\ifFB@poorman
929 \newdimen\FB@Mht
930 \ifLaTeXe
931   \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{}`.

```

932 \newcommand*{\FBsupR}{-0.12}
933 \newcommand*{\FBsupS}{0.65}
934 \newcommand*{\FB@lc}[1]{\MakeLowercase{#1}}
935 \DeclareRobustCommand*{\FB@up@fake}[1]{%
936   \settoheight{\FB@Mht}{M}%
937   \addtolength{\FB@Mht}{\FBsupR\FB@Mht}%
938   \addtolength{\FB@Mht}{-\FBsupS ex}%
939   \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
940 }

```

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.

```

941 \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
942                               \def\FB@suffix{#4}}
943 \def\FB@x{x}
944 \def\FB@j{j}
945 \DeclareRobustCommand*\FB@up{[1]{%
946   \bgroup \FB@poormantrue
947   \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.

```

948   \edef\reserved@a{\lowercase{%
949     \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
950   \reserved@a
951   {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
952    \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
953    \if\FB@poorman \FB@up@fake{#1}%
954    \else          \FB@up@real{#1}%
955    \fi}%
956   {\FB@up@fake{#1}}%
957 \egroup}

```

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

```

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

```

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

```

960 \DeclareRobustCommand*\fup{[1]{%
961   \ifx\realsuperscript\@undefined
962     \FB@up{#1}%
963   \else
964     \bgroup\let\fakesuperscript\FB@up@fake
965     \realsuperscript{\FB@lc{#1}}\egroup
966   \fi}

```

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

```

967 \providecommand*\up{\relax}
Poor man's definition of \up for Plain.
968 \else
969 \providecommand*\up[1]{\leavevmode\raiselex\hbox{\sevenrm #1}}
970 \fi

\ieme Some handy macros for those who don't know how to abbreviate ordinals:
\ier 971 \def\ieme{\up{e}\xspace}
\iere 972 \def\iemes{\up{es}\xspace}
\iemes 973 \def\ier{\up{er}\xspace}
\iers 974 \def\iers{\up{ers}\xspace}
\ieres 975 \def\iere{\up{re}\xspace}
976 \def\ieres{\up{res}\xspace}

\FBmedkern
\FBthickkern 977 \newcommand*\FBmedkern{\kern+.2em}
978 \newcommand*\FBthickkern{\kern+.3em}

\No And some more macros relying on \up for numbering, first two support macros.
\no 979 \newcommand*\FrenchEnumerate[1]{#1\up{o}\FBthickkern}
\Nos 980 \newcommand*\FrenchPopularEnumerate[1]{#1\up{o})\FBthickkern}
\nos Typing \primo should result in '°',
\primo 981 \def\primo{\FrenchEnumerate1}
\fprimo 982 \def\secundo{\FrenchEnumerate2}
983 \def\tertio{\FrenchEnumerate3}
984 \def\quarto{\FrenchEnumerate4}
while typing \fprimo) gives '°)'.
985 \def\fprimo{\FrenchPopularEnumerate1}
986 \def\fsecundo{\FrenchPopularEnumerate2}
987 \def\ftertio{\FrenchPopularEnumerate3}
988 \def\fquarto{\FrenchPopularEnumerate4}

Let's provide four macros for the common abbreviations of "Numéro".
989 \DeclareRobustCommand*\No{N\up{o}\FBmedkern}
990 \DeclareRobustCommand*\no{n\up{o}\FBmedkern}
991 \DeclareRobustCommand*\Nos{N\up{os}\FBmedkern}
992 \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}.
993 \DeclareRobustCommand*\bsc[1]{\leavevmode\begingroup\kern0pt
994 \scshape #1\endgroup}
995 \ifLaTeX\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`. `\degree` can be accessed by the command `\r{}` for ring accent.

```

996 \iffBUnicode
997   \newcommand*{\at}{\char"0040}
998   \newcommand*{\circonflexe}{\char"005E}
999   \newcommand*{\tild}{\char"007E}
1000  \newcommand*{\boi}{\char"005C}
1001  \newcommand*{\degree}{\char"00B0}
1002 \else
1003   \ifLaTeXe
1004     \DeclareTextSymbol{\at}{T1}{64}
1005     \DeclareTextSymbol{\circonflexe}{T1}{94}
1006     \DeclareTextSymbol{\tild}{T1}{126}
1007     \DeclareTextSymbolDefault{\at}{T1}
1008     \DeclareTextSymbolDefault{\circonflexe}{T1}
1009     \DeclareTextSymbolDefault{\tild}{T1}
1010     \DeclareRobustCommand*{\boi}{\textbackslash}
1011     \DeclareRobustCommand*{\degree}{\r{}}
1012   \else
1013     \def\T@one{T1}
1014     \ifx\f@encoding\T@one
1015       \newcommand*{\degree}{\char6}
1016     \else
1017       \newcommand*{\degree}{\char23}
1018     \fi
1019     \newcommand*{\at}{\char64}
1020     \newcommand*{\circonflexe}{\char94}
1021     \newcommand*{\tild}{\char126}
1022     \newcommand*{\boi}{\backslash}
1023   \fi
1024 \fi

```

\degrees We now define a macro `\degrees` 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 `\degrees` to 0.3 em, this lets the symbol ‘degree’ stick to the preceding (e.g., 45\degrees) or following character (e.g., 20~\degrees 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.

```

1025 \ifLaTeXe
1026   \newcommand*{\degrees}{\degree}
1027 \iffBUnicode
1028   \DeclareRobustCommand*{\degrees}{\degree}
1029 \else
1030   \def\Warning@degree@TSone{\FBWarning
1031     {Degrees would look better in TS1-encoding:%
1032     \MessageBreak add \protect

```

```

1033         \usepackage{textcomp} to the preamble.%
1034         \MessageBreak Degrees used}}
1035     \AtBeginDocument{\ifx\DeclareEncodingSubset\undefined
1036         \DeclareRobustCommand*\degrees{%
1037             \leavevmode\hbox to 0.3em{\hss\degre\hss}%
1038             \Warning@degree@TSone
1039             \global\let\Warning@degree@TSone\relax}%
1040         \else
1041             \DeclareRobustCommand*\degrees{%
1042                 \hbox{\UseTextSymbol{TS1}{\textdegree}}}%
1043         \fi
1044     }
1045 \fi
1046 \else
1047     \newcommand*\degrees{%
1048         \leavevmode\hbox to 0.3em{\hss\degre\hss}}
1049 \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.

```

1050 \newif\ifFB@icomma
1051 \newcount\mc@charclass
1052 \newcount\mc@charfam
1053 \newcount\mc@charslot
1054 \newcount\std@mcc
1055 \newcount\dec@mcc
1056 \ifBLaTeX
1057     \mc@charclass=\Umathcharclass'\,
1058     \newcommand*\dec@math@comma{%
1059         \mc@charfam=\Umathcharfam'\,
1060         \mc@charslot=\Umathcharslot'\,
1061         \Umathcode'\,= 0 \mc@charfam \mc@charslot
1062     }
1063     \newcommand*\std@math@comma{%
1064         \mc@charfam=\Umathcharfam'\,
1065         \mc@charslot=\Umathcharslot'\,
1066         \Umathcode'\,= \mc@charclass \mc@charfam \mc@charslot
1067     }
1068 \else
1069     \std@mcc=\mathcode'\,
1070     \dec@mcc=\std@mcc
1071     \@tempcnta=\std@mcc
1072     \divide\@tempcnta by "1000

```

```

1073 \multiply\@tempcnta by "1000
1074 \advance\dec@mcc by -\@tempcnta
1075 \newcommand*\{dec@math@comma}\{\mathcode'\,\,=\dec@mcc}
1076 \newcommand*\{std@math@comma}\{\mathcode'\,\,=\std@mcc}
1077 \fi
1078 \newcommand*\{DecimalMathComma}\{%
1079 \ifFBfrench\dec@math@comma\fi
1080 \ifFB@icomma\else\addto\extrasfrench{\dec@math@comma}\fi
1081 }
1082 \newcommand*\{StandardMathComma}\{%
1083 \std@math@comma
1084 \ifFB@icomma\else\addto\extrasfrench{\std@math@comma}\fi
1085 }
1086 \ifLaTeXe
1087 \AtBeginDocument{\@ifpackageloaded{icomma}%
1088 \{\FB@icommatrue}%
1089 \{\addto\noextrasfrench{\std@math@comma}}}%
1090 }
1091 \else
1092 \addto\noextrasfrench{\std@math@comma}
1093 \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:

```

1094 \newcommand*\{nombre}[1]{\{#1}\fb@warning{*** \noexpand\nombre
1095 \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.

```

1096 \ifFB@luatexpunct
1097 \activate@luatexpunct
1098 \fi
1099 \let\FBstop@here\relax
1100 \def\FBclean@on@exit{%
1101 \let\ifLaTeXe\undefined
1102 \let\LaTeXetrue\undefined
1103 \let\LaTeXefalse\undefined
1104 \let\FB@llc\loadlocalcfg
1105 \let\loadlocalcfg@gobble}
1106 \ifx\magnification\@undefined
1107 \else
1108 \def\FBstop@here{%
1109 \FBclean@on@exit
1110 \ldf@finish\CurrentOption
1111 \let\loadlocalcfg\FB@llc

```

```

1112 \endinput}
1113 \fi
1114 \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.

```

1115 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1116 \newcommand*{\Warning@nombre}[1]{%
1117   \ifdefined\numprint
1118     \numprint{#1}%
1119   \else
1120     \PackageWarning{french.ldf}{%
1121       \protect\nombre\space now relies on package numprint.sty,%
1122       \MessageBreak add \protect
1123       \usepackage[autolanguage]{numprint},\MessageBreak
1124       see file numprint.pdf for more options.\MessageBreak
1125       \protect\nombre\space called}%
1126     \global\let\Warning@nombre\relax
1127     {#1}%
1128   \fi
1129 }

1130 \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}`).

```

1131 \providecommand*{\FBfigtabshape}{\scshape}

```

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

```

1132 \StartBabelCommands*{\BabelLanguages}{captions}
1133   [unicode, fontenc=TU EU1 EU2, charset=utf8]
1134   \SetString{\refname}{Références}
1135   \SetString{\abstractname}{Résumé}
1136   \SetString{\prefacename}{Préface}
1137   \SetString{\contentsname}{Table des matières}
1138   \SetString{\ccname}{Copie à }
1139   \SetString{\proofname}{Démonstration}
1140   \SetString{\partfirst}{Première}
1141   \SetString{\partsecond}{Deuxième}
1142   \SetStringLoop{ordinal#1}{%
1143     \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%

```

```

1144     Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1145     Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1146     Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1147 \StartBabelCommands*{\BabelLanguages}{captions}
1148     \SetString{\refname}{R\’ef\’erences}
1149     \SetString{\abstractname}{R\’esum\’e}
1150     \SetString{\bibname}{Bibliographie}
1151     \SetString{\prefacename}{Pr\’eface}
1152     \SetString{\chaptername}{Chapitre}
1153     \SetString{\appendixname}{Annexe}
1154     \SetString{\contentsname}{Table des mati\’eres}
1155     \SetString{\listfigurename}{Table des figures}
1156     \SetString{\listtablename}{Liste des tableaux}
1157     \SetString{\indexname}{Index}
1158     \SetString{\figurename}{{\FBfigtabshape Figure}}
1159     \SetString{\tablename}{{\FBfigtabshape Table}}
1160     \SetString{\pagename}{page}
1161     \SetString{\seename}{voir}
1162     \SetString{\alsoname}{voir aussi}
1163     \SetString{\enclname}{P.-J. }
1164     \SetString{\ccname}{Copie \’a }
1165     \SetString{\headtoname}{}
1166     \SetString{\proofname}{D\’emonstration}
1167     \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`.

```

1168     \SetString{\partfirst}{Premi\’ere}
1169     \SetString{\partsecond}{Deuxi\’eme}
1170     \SetString{\partnameord}{partie}
1171     \SetStringLoop{ordinal#1}{%
1172         \partfirst,\partsecond,Troisi\’eme,Quatri\’eme,%
1173         Cinqui\’eme,Sixi\’eme,Septi\’eme,Huiti\’eme,Neuvi\’eme,Dixi\’eme,%
1174         Onzi\’eme,Douzi\’eme,Treizi\’eme,Quatorzi\’eme,Quinzi\’eme,%
1175         Seizi\’eme,Dix-septi\’eme,Dix-huiti\’eme,Dix-neuvi\’eme,%
1176         Vingti\’eme}
1177 \AfterBabelCommands{%
1178     \DeclareRobustCommand*{\FB@emptypart}{\def\thepart{}}%
1179     \DeclareRobustCommand*{\FB@partname}{%
1180         \ifFBPartNameFull
1181             \csname ordinal\romannumeral\value{part}\endcsname\space
1182             \partnameord\FB@emptypart
1183         \else
1184             Partie%
1185         \fi}%
1186     }
1187     \SetString{\partname}{\FB@partname}
1188 \EndBabelCommands

```


2.8 Figure and table captions

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

```
1189 \newcommand{\FBWarning}[1]{\PackageWarning{french.ldf}{#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.ldf makes '>' active).

```
1190 \bgroup
1191 \catcode'::=12 \catcode'>:=12 \relax
1192 \long\gdef\STD@makecaption#1#2{%
1193   \vskip\abovcaptionskip
1194   \sbox\@tempboxa{#1: #2}%
1195   \ifdim \wd\@tempboxa >\hsize
1196     #1: #2\par
1197   \else
1198     \global \@minipagefalse
1199     \hbext\@hsize{\hfil\box\@tempboxa\hfil}%
1200   \fi
1201   \vskip\belowcaptionskip}
1202 \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.

```
1203 \newif\if@FBwarning@capsep
1204 \ifFB@active@punct\@FBwarning@capseptrue\fi
1205 \newcommand*\CaptionSeparator{\space\textendash\space}
1206 \def\FBCaption@Separator{: }
```

```

1207 \long\def\FB@makecaption#1#2{%
1208   \vskip\abovecaptionskip
1209   \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1210   \ifdim \wd\@tempboxa >\hsize
1211     #1\FBCaption@Separator #2\par
1212   \else
1213     \global \@minipagefalse
1214     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1215   \fi
1216   \vskip\belowcaptionskip}

```

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

```

1217 \@ifclassloaded{acmart}{\@FBwarning@capsepfalse}{}
1218 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1219 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1220 \@ifclassloaded{amstex}{\@FBwarning@capsepfalse}{}
1221 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1222 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1223 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1224 \@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) .

```

1225 \newif\ifFB@koma
1226 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1227 \@ifclassloaded{scrartcl}{\@FBwarning@capsepfalse\FB@komatrue}{}
1228 \@ifclassloaded{scrbook}{\@FBwarning@capsepfalse\FB@komatrue}{}
1229 \@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).

```

1230 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1231 \ifdefined\@makecaption\else\@FBwarning@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.

```

1232 \newcounter{FBCaption@count}
1233 \@ifpackageloaded{caption3}{\addtocounter{FBCaption@count}{4}}{}
1234 \@ifpackageloaded{subcaption}{\addtocounter{FBCaption@count}{2}}{}
1235 \@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').

```

1236 \AtBeginDocument{%
1237   \ifx\@makecaption\STD@makecaption
1238     \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`.

```

1239   \ifFBOldFigTabCaptions
1240   \else
1241     \def\FBCaption@Separator{\ifFBfrench\space\fi : }%
1242   \fi
1243   \ifFBCustomiseFigTabCaptions
1244     \ifFB@mainlanguage@FR
1245       \def\FBCaption@Separator{\CaptionSeparator}%
1246     \fi
1247   \fi
1248   \@FBwarning@capsepfalse
1249 \fi

```

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

```

1250 \@ifpackageloaded{caption3}{%
1251   \ifnum\value{FBcaption@count}=0 \@FBwarning@capsepfalse\fi
1252 }{}%
1253 \if@FBwarning@capsep
1254   \ifnum\value{FBcaption@count}>0

```

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

```

1255   \FBWarning
1256     {Figures' and tables' captions might look like\MessageBreak
1257      'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1258      If you have loaded any of the packages caption,\MessageBreak
1259      subcaption or floatrow BEFORE babel/french,\MessageBreak
1260      please move them AFTER babel/french.\MessageBreak
1261      If one of them is loaded by your class,\MessageBreak
1262      you can still add AFTER babel/french\MessageBreak
1263      \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1264      \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1265      ... live with it; reported}%
1266   \else

```

`caption3.sty` hasn't been loaded at all.

```

1267   \FBWarning
1268     {Figures' and tables' captions might look like\MessageBreak
1269      'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1270      If it happens, see your class documentation to\MessageBreak
1271      fix this issue or add AFTER babel/french\MessageBreak
1272      \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1273      \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1274      or ... live with it; reported}%
1275   \fi
1276 \fi

```

```

1277 \let\FB@makecaption\relax
1278 \let\STD@makecaption\relax
1279 }

```

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.

```

1280 \ifFBunicode
1281 \let\FBtextellipsis\textellipsis
1282 \else
1283 \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1284 \DeclareTextCommandDefault{\FBtextellipsis}{%
1285   .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}
1286 \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.

```

1287 \newcommand*{\Tdots@}{\@xp\textellipsis}
1288 \newcommand*{\Mdots@}{\@xp\mdots@}
1289 \AtBeginDocument{\DeclareRobustCommand*{\dots}{\relax
1290   \csname\ifmmode M\else T\fi dots@endcsname}%
1291   \ifdefined\@xp\else\let\@xp\relax\fi
1292   \ifdefined\mdots@\else\let\Mdots@\mathellipsis\fi
1293 }
1294 \def\bbl@frenchdots{\babel@save\Tdots@ \let\Tdots@\FBtextellipsis}
1295 \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).

```

1296 \ifFB@active@punct
1297 \ifpackageloaded{listings}
1298 {\AtBeginDocument{%
1299   \FBWarning{Please load the "listings" package\MessageBreak
1300     AFTER babel/french; reported}}%
1301 }{}
1302 \fi

```

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

```

1303 \newif\if@FBwarning@natbib
1304 \ifFB@active@punct
1305   \@ifpackageloaded{natbib}{}{\@FBwarning@natbibtrue}
1306 \fi
1307 \AtBeginDocument{%
1308   \if@FBwarning@natbib
1309     \@ifpackageloaded{natbib}{}{\@FBwarning@natbibfalse}%
1310   \fi
1311   \if@FBwarning@natbib
1312     \FBWarning{Please load the "natbib" package\MessageBreak
1313               BEFORE babel/french; reported}%
1314   \fi
1315 }

```

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

```

1316 \newif\if@FBwarning@beamerarticle
1317 \@ifpackageloaded{beamerarticle}{}{\@FBwarning@beamerarticlettrue}
1318 \AtBeginDocument{%
1319   \if@FBwarning@beamerarticle
1320     \@ifpackageloaded{beamerarticle}{}%
1321                                     {\@FBwarning@beamerarticlefalse}%
1322   \fi
1323   \if@FBwarning@beamerarticle
1324     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1325               BEFORE babel/french; reported}%
1326   \fi
1327 }

```

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 `\extrasfrench{}` *must* be processed by `\frenchsetup{}`: when French is the main language, `\extrasfrench{}` is executed by babel when it switches the main language and this occurs *before* reading the stuff postponed by babel-french 'AtBeginDocument'. Reexecuting `\extrasfrench{}` 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.

```

1328 \newcommand*{\frenchsetup}[1]{%
1329   \setkeys{FB}{#1}%
1330 }%
1331 \@onlypreamble\frenchsetup

```

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

```

1332 \let\frenchbsetup\frenchsetup
1333 \@onlypreamble\frenchbsetup

```

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

```

1334 \newif\ifFBShowOptions
1335 \newif\ifFBStandardLayout           \FBStandardLayouttrue
1336 \newif\ifFBGlobalLayoutFrench       \FBGlobalLayoutFrenchtrue
1337 \newif\ifFBReduceListSpacing
1338 \newif\ifFBListOldLayout
1339 \newif\ifFBListItemsAsPar
1340 \newif\ifFBCompactItemize
1341 \newif\ifFBStandardItemizeEnv        \FBStandardItemizeEnvtrue
1342 \newif\ifFBStandardEnumerateEnv     \FBStandardEnumerateEnvtrue
1343 \newif\ifFBStandardItemLabels       \FBStandardItemLabelstrue
1344 \newif\ifFBStandardLists            \FBStandardListstrue
1345 \newif\ifFBIndentFirst
1346 \newif\ifFBFrenchFootnotes
1347 \newif\ifFBAutoSpaceFootnotes
1348 \newif\ifFBOriginalTypewriter
1349 \newif\ifFBThinColonSpace
1350 \newif\ifFBThinSpaceInFrenchNumbers
1351 \newif\ifFBFrenchSuperscripts        \FBFrenchSuperscriptstrue
1352 \newif\ifFBLowercaseSuperscripts    \FBLowercaseSuperscriptstrue
1353 \newif\ifFBPartNameFull              \FBPartNameFulltrue
1354 \newif\ifFBCustomiseFigTabCaptions
1355 \newif\ifFBOldFigTabCaptions
1356 \newif\ifFBSmallCapsFigTabCaptions \FBSmallCapsFigTabCaptionstrue
1357 \newif\ifFBSuppressWarning
1358 \newif\ifFBINGuillSpace

```

The defaults 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`.

```

1359 \ifFB@koma
1360   \ifdefined\partformat
1361     \def\FB@partformat@fix{%
1362       \ifFBPartNameFull
1363         \babel@save\partformat
1364         \renewcommand*{\partformat}{\partname}%

```

```

1365         \fi}
1366     \addto\extrasfrench{\FB@partformat@fix}%
1367 \fi
1368 \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.

```

1369 \def\FB@french{french}
1370 \def\FB@acadian{acadian}
1371 \newif\ifFB@mainlanguage@FR
1372 \AtEndOfPackage{%
1373     \ifx\bbbl@main@language\FB@french \FB@mainlanguage@FRtrue
1374     \else \ifx\bbbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1375     \fi
1376     \ifFB@mainlanguage@FR
1377         \FBGlobalLayoutFrenchtrue
1378         \@ifclassloaded{beamer}%
1379             {\PackageInfo{french.ldf}{%
1380                 No list customisation for the beamer class,%
1381                 \MessageBreak reported}}%
1382             {\@ifpackageloaded{beamerarticle}%
1383                 {\FBStandardItemLabelsfalse
1384                  \FBReduceListSpacingtrue
1385                  \PackageInfo{french.ldf}{%
1386                      Minimal list customisation for the beamerarticle%
1387                      \MessageBreak package; reported}}%

```

Otherwise customise lists “à la française”:

```

1388         {\FBReduceListSpacingtrue
1389         \FBStandardItemizeEnvfalse
1390         \FBStandardEnumerateEnvfalse
1391         \FBStandardItemLabelsfalse}%
1392     }
1393     \FBIndentFirsttrue
1394     \FBFrenchFootnotesttrue
1395     \FBAutoSpaceFootnotesttrue
1396     \FBCustomiseFigTabCaptionstrue
1397 \else
1398     \FBGlobalLayoutFrenchfalse
1399 \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.

```

1400 \RequirePackage{keyval}%
1401 \define@key{FB}{ShowOptions}[true]%
1402     {\csname FBShowOptions#1\endcsname}%

```

```

1403 \define@key{FB}{StandardLayout}[true]%
1404     {\csname FBStandardLayout#1\endcsname
1405     \ifFBStandardLayout
1406         \FBReduceListSpacingfalse
1407         \FBStandardItemizeEnvtrue
1408         \FBStandardItemLabelstrue
1409         \FBStandardEnumerateEnvtrue
1410         \FBIndentFirstfalse
1411         \FBFrenchFootnotesfalse
1412         \FBAutoSpaceFootnotesfalse
1413         \FBGlobalLayoutFrenchfalse
1414     \else
1415         \FBReduceListSpacingtrue
1416         \FBStandardItemizeEnvfalse
1417         \FBStandardItemLabelsfalse
1418         \FBStandardEnumerateEnvfalse
1419         \FBIndentFirsttrue
1420         \FBFrenchFootnotesttrue
1421         \FBAutoSpaceFootnotesttrue
1422     \fi}%
1423 \define@key{FB}{GlobalLayoutFrench}[true]%
1424     {\csname FBGlobalLayoutFrench#1\endcsname

```

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. Warn and reset in case this key is set to true while the main language is *not* French.

```

1425     \ifFBGlobalLayoutFrench
1426         \ifFB@mainlanguage@FR
1427         \else
1428             \FBGlobalLayoutFrenchfalse
1429             \PackageWarning{french.ldf}%
1430                 {Option ‘GlobalLayoutFrench’ skipped:\MessageBreak
1431                 French is *not* babel’s last option.\MessageBreak
1432                 Reported}%
1433         \fi
1434     \fi}%
1435 \define@key{FB}{ReduceListSpacing}[true]%
1436     {\csname FBReduceListSpacing#1\endcsname}%
1437 \define@key{FB}{ListOldLayout}[true]%
1438     {\csname FBListOldLayout#1\endcsname
1439     \ifFBListOldLayout
1440         \FBStandardEnumerateEnvtrue
1441         \renewcommand*{\FrenchLabelItem}{\textendash}%
1442     \fi}%
1443 \define@key{FB}{CompactItemize}[true]%
1444     {\csname FBCompactItemize#1\endcsname
1445     \ifFBCompactItemize
1446         \FBStandardItemizeEnvfalse
1447         \FBStandardEnumerateEnvfalse
1448     \else

```



```

1449         \FBStandardItemizeEnvtrue
1450         \FBStandardEnumerateEnvtrue
1451     \fi}%
1452 \define@key{FB}{StandardItemizeEnv}[true]%
1453     {\csname FBStandardItemizeEnv#1\endcsname}%
1454 \define@key{FB}{StandardEnumerateEnv}[true]%
1455     {\csname FBStandardEnumerateEnv#1\endcsname}%
1456 \define@key{FB}{StandardItemLabels}[true]%
1457     {\csname FBStandardItemLabels#1\endcsname}%
1458 \define@key{FB}{ItemLabels}%
1459     {\renewcommand*{\FrenchLabelItem}{#1}}%
1460 \define@key{FB}{ItemLabeli}%
1461     {\renewcommand*{\Frlabelitemi}{#1}}%
1462 \define@key{FB}{ItemLabelii}%
1463     {\renewcommand*{\Frlabelitemii}{#1}}%
1464 \define@key{FB}{ItemLabeliii}%
1465     {\renewcommand*{\Frlabelitemiii}{#1}}%
1466 \define@key{FB}{ItemLabeliv}%
1467     {\renewcommand*{\Frlabelitemiv}{#1}}%
1468 \define@key{FB}{StandardLists}[true]%
1469     {\csname FBStandardLists#1\endcsname
1470     \ifFBStandardLists
1471         \FBReduceListSpacingfalse
1472         \FBCompactItemizefalse
1473         \FBStandardItemizeEnvtrue
1474         \FBStandardEnumerateEnvtrue
1475         \FBStandardItemLabelstrue
1476     \else
1477         \FBReduceListSpacingtrue
1478         \FBCompactItemizetrue
1479         \FBStandardItemizeEnvfalse
1480         \FBStandardEnumerateEnvfalse
1481         \FBStandardItemLabelsfalse
1482     \fi}%
1483 \define@key{FB}{ListItemsAsPar}[true]%
1484     {\csname FBListItemsAsPar#1\endcsname}%
1485 \define@key{FB}{IndentFirst}[true]%
1486     {\csname FBIndentFirst#1\endcsname}%
1487 \define@key{FB}{FrenchFootnotes}[true]%
1488     {\csname FBFrenchFootnotes#1\endcsname}%
1489 \define@key{FB}{AutoSpaceFootnotes}[true]%
1490     {\csname FBAutoSpaceFootnotes#1\endcsname}%
1491 \define@key{FB}{AutoSpacePunctuation}[true]%
1492     {\csname FBAutoSpacePunctuation#1\endcsname}%
1493 \define@key{FB}{OriginalTypewriter}[true]%
1494     {\csname FBOriginalTypewriter#1\endcsname}%
1495 \define@key{FB}{ThinColonSpace}[true]%
1496     {\csname FBThinColonSpace#1\endcsname
1497     \ifFBThinColonSpace
1498         \renewcommand*{\FBcolonspace}{\FBthinspace}%
1499     \fi}%

```

```

1500 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1501     {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1502 \define@key{FB}{FrenchSuperscripts}[true]%
1503     {\csname FBFrenchSuperscripts#1\endcsname}%
1504 \define@key{FB}{LowercaseSuperscripts}[true]%
1505     {\csname FBLowercaseSuperscripts#1\endcsname}%
1506 \define@key{FB}{PartNameFull}[true]%
1507     {\csname FBPartNameFull#1\endcsname}%
1508 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1509     {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1510 \define@key{FB}{OldFigTabCaptions}[true]%
1511     {\csname FBOldFigTabCaptions#1\endcsname}
1512     \ifFBOldFigTabCaptions
1513         \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1514             \def\FBCaption@Separator{\CaptionSeparator}}%
1515         \addto\extrasfrench{\FB@capsep@fix}%
1516         \ifdefined\extrasacadian
1517             \addto\extrasacadian{\FB@capsep@fix}%
1518         \fi
1519     \fi}%
1520 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1521     {\csname FBSmallCapsFigTabCaptions#1\endcsname}
1522     \ifFBSmallCapsFigTabCaptions
1523         \let\FBfigtabshape\scshape
1524     \else
1525         \let\FBfigtabshape\relax
1526     \fi}%
1527 \define@key{FB}{SuppressWarning}[true]%
1528     {\csname FBSuppressWarning#1\endcsname}
1529     \ifFBSuppressWarning
1530         \renewcommand{\FBWarning}[1]{}%
1531     \fi}%

```

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

```

1532 \define@key{FB}{INGuillSpace}[true]%
1533     {\csname FBINGuillSpace#1\endcsname}
1534     \ifFBINGuillSpace
1535         \renewcommand*{\FBguillspace}{\space}%
1536     \fi}%
1537 \define@key{FB}{InnerGuillSingle}[true]%
1538     {\csname FBInnerGuillSingle#1\endcsname}%
1539 \define@key{FB}{EveryParGuill}[open]%
1540     {\expandafter\let\expandafter
1541         \FBeveryparguill\csname FBguill#1\endcsname
1542         \ifx\FBeveryparguill\FBguillopen
1543             \else\ifx\FBeveryparguill\FBguillclose
1544                 \else\ifx\FBeveryparguill\FBguillnone
1545                     \else
1546                         \let\FBeveryparguill\FBguillopen
1547                         \FBWarning{Wrong value for 'EveryParGuill':

```

```

1548                                     try 'open',\MessageBreak
1549                                     'close' or 'none'. Reported}%
1550                                 \fi
1551                            \fi
1552                    \fi}%
1553 \define@key{FB}{EveryLineGuill}[open]%
1554     {\ifFB@luatex@punct
1555         \expandafter\let\expandafter
1556             \FBeverylineguill\csname FBguill#1\endcsname
1557         \ifx\FBeverylineguill\FBguillopen
1558         \else\ifx\FBeverylineguill\FBguillclose
1559         \else\ifx\FBeverylineguill\FBguillnone
1560         \else
1561             \let\FBeverylineguill\FBguillnone
1562             \FBWarning{Wrong value for 'EveryLineGuill':
1563                 try 'open',\MessageBreak
1564                 'close' or 'none'. Reported}%
1565         \fi
1566     \fi
1567 \fi
1568 \else
1569     \FBWarning{Option 'EveryLineGuill' skipped:%
1570         \MessageBreak this option is for
1571         LuaTeX *only*.\MessageBreak Reported}%
1572 \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.

```

1573 \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1574     {\ifFB@luatex@punct
1575         \csname FBucsNBSP#1\endcsname
1576         \ifFBucsNBSP \FB@ucsNBSP=1 \fi
1577     \else
1578         \FBWarning{Option 'UnicodeNoBreakSpaces' skipped:%
1579             \MessageBreak this option is for
1580             LuaTeX *only*.\MessageBreak Reported}%
1581     \fi
1582 }%

```

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@addGUILLspace 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.

```
1583 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
```

```
1584 \define@key{FB}{og}%
```

```
1585 {\ifFBunicode
```

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

```
1586 \ifFB@luatex@punct
```

```
1587 \FB@addGUIspace=1 \relax
```

```
1588 \fi
```

then with XeTeX it is a bit more tricky:

```
1589 \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).

```
1590 \XeTeXcharclass"13 = \FB@guilo
```

```
1591 \XeTeXcharclass"AB = \FB@guilo
```

```
1592 \XeTeXcharclass"A0 = \FB@guilnul
```

```
1593 \XeTeXcharclass"202F = \FB@guilnul
```

```
1594 \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1595 \ifFB@active@punct
```

```
1596 \FBWarning{Option og=« not supported with this version
1597 of\MessageBreak LuaTeX/XeTeX; reported}%
```

```
1598 \fi
```

```
1599 \else
```

This is for conventional TeX engines:

```
1600 \newcommand*{\FB@@og}{%
```

```
1601 \ifFBfrench
```

```
1602 \ifFB@spacing\FB@og\ignorespaces
```

```
1603 \else\guillemotleft
```

```
1604 \fi
```

```
1605 \else\guillemotleft\fi}%
```

```
1606 \AtBeginDocument{%
```

```
1607 \ifdefined\uc@dclc
```

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

```
1608 \uc@dclc{171}{default}{\FB@@og}%
```

```
1609 \else
```

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

```
1610 \FB@parse#1\endparse
```

```
1611 \ifx\FB@second\@empty
```

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

```
1612 \ifdefined\mule@def
```

```
1613 \mule@def{11}{\FB@@og}%
```

```

1614             \else
1615             \ifdefined\DeclareInputText
1616             \@tempcnta'#1\relax
1617             \DeclareInputText{\the\@tempcnta}{\FB@og}%
1618             \else
Package inputenc not loaded, no way...
1619             \FBWarning{Option 'og' requires package
1620             inputenc;\MessageBreak reported}%
1621             \fi
1622             \fi
1623             \else
This means multi-byte character encoding, we assume UTF-8
1624             \DeclareUnicodeCharacter{00AB}{\FB@og}%
1625             \fi
1626             \fi}%
1627         \fi
1628     }%
Same code for the closing quote.
1629 \define@key{FB}{fg}{%
1630     {\ifFBunicode
1631         \ifFB@luatex@punct
1632             \FB@addGUILspace=1 \relax
1633         \fi
1634         \ifFB@xetex@punct
1635             \XeTeXcharclass"14 = \FB@guilf
1636             \XeTeXcharclass"BB = \FB@guilf
1637             \XeTeXcharclass"A0 = \FB@guilnul
1638             \XeTeXcharclass"202F = \FB@guilnul
1639         \fi
1640         \ifFB@active@punct
1641             \FBWarning{Option fg=> not supported with this version
1642             of\MessageBreak LuaTeX/XeTeX; reported}%
1643         \fi
1644         \else
1645             \newcommand*{\FB@fg}{%
1646                 \ifFBfrench
1647                     \ifFB@spacing\FB@fg
1648                     \else\guillemotright
1649                     \fi
1650                 \else\guillemotright\fi}%
1651             \AtBeginDocument{%
1652                 \ifdefined\uc@dcl
1653                     \uc@dcl{187}{default}{\FB@fg}%
1654                 \else
1655                     \FB@parse#1\endparse
1656                     \ifx\FB@second\@empty
1657                         \ifdefined\mule@def
1658                             \mule@def{27}{\FB@fg}%
1659                         \else

```

```

1660             \ifdefined\DeclareInputText
1661             \@tempcnta'#1\relax
1662             \DeclareInputText{\the\@tempcnta}{\FB@@fg}%
1663         \else
1664             \FBWarning{Option 'fg' requires package
1665             inputenc;\MessageBreak reported}%
1666         \fi
1667     \fi
1668     \else
1669         \DeclareUnicodeCharacter{00BB}{\FB@@fg}%
1670     \fi
1671 \fi}%
1672 \fi
1673 }%
1674 }

```

\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.

```

1675 \newcommand*{\FBprocess@options}{%

```

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

```

1676 \@ifpackageloaded{enumitem}{%
1677     \ifFBStandardItemizeEnv
1678     \else
1679         \FBStandardItemizeEnvtrue
1680         \PackageInfo{french.ldb}{%
1681             {Setting StandardItemizeEnv=true for\MessageBreak
1682             compatibility with enumitem package,\MessageBreak
1683             reported}%
1684         \fi
1685     \ifFBStandardEnumerateEnv
1686     \else
1687         \FBStandardEnumerateEnvtrue
1688         \PackageInfo{french.ldb}{%
1689             {Setting StandardEnumerateEnv=true for\MessageBreak
1690             compatibility with enumitem package,\MessageBreak
1691             reported}%
1692         \fi}{}%
1693 \@ifpackageloaded{paralist}{%
1694     \ifFBStandardItemizeEnv
1695     \else
1696         \FBStandardItemizeEnvtrue
1697         \PackageInfo{french.ldb}{%
1698             {Setting StandardItemizeEnv=true for\MessageBreak
1699             compatibility with paralist package,\MessageBreak
1700             reported}%

```

```

1701 \fi
1702 \ifFBStandardEnumerateEnv
1703 \else
1704   \FBStandardEnumerateEnvtrue
1705   \PackageInfo{french.ldf}%
1706     {Setting StandardEnumerateEnv=true for\MessageBreak
1707       compatibility with paralist package,\MessageBreak
1708       reported}%
1709 \fi}{}%
1710 \@ifpackageloaded{enumerate}{%
1711   \ifFBStandardEnumerateEnv
1712   \else
1713     \FBStandardEnumerateEnvtrue
1714     \PackageInfo{french.ldf}%
1715       {Setting StandardEnumerateEnv=true for\MessageBreak
1716         compatibility with enumerate package,\MessageBreak
1717         reported}%
1718 \fi}{}%

```

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

```

1719 \def\FB@ufl{\update@frenchlists}
1720 \ifFB@mainlanguage@FR
1721   \update@frenchlists
1722 \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.

```

1723 \ifFBAutoSpacePunctuation
1724   \autospace@beforeFDP
1725 \else
1726   \noautospace@beforeFDP
1727 \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.

```

1728 \ifFBOriginalTypewriter
1729 \else
1730   \let\ttfamilyORI\ttfamily
1731   \let\rmfamilyORI\rmfamily
1732   \let\sffamilyORI\sffamily
1733   \let\ttfamily\ttfamilyFB
1734   \let\rmfamily\rmfamilyFB
1735   \let\sffamily\sffamilyFB
1736 \fi

```

When package `numprint` is loaded with option `autolanguage`, `numprint`'s com-

mand `\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.

```

1737 \ifpackageloaded{numprint}%
1738   {\ifnprt@autolanguage
1739     \providecommand*\npstylefrench{}{}%
1740     \ifFBThinSpaceInFrenchNumbers
1741       \renewcommand*\FBthousandsep{\,}%
1742     \fi
1743     \g@addto@macro\npstylefrench{\npthousandsep\FBthousandsep}%
1744   \fi
1745   }{}%
```

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.

```

1746 \ifFBFrenchSuperscripts
1747   \DeclareRobustCommand*\up*{\@ifstar\FB@up@fake}\fup}%
1748 \else
1749   \DeclareRobustCommand*\up*{\@ifstar\FB@up@fake%
1750                               \textsuperscript}%
1751 \fi
```

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

```

1752 \ifFBLowercaseSuperscripts
1753 \else
1754   \renewcommand*\FB@lc[1]{##1}%
1755 \fi
```

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

```

1756 \ifFBCustomiseFigTabCaptions
1757   \ifFB@koma
1758     \renewcommand*\captionformat{\CaptionSeparator}%
1759   \fi
1760   \@ifclassloaded{memoir}%
1761     {\captiondelim{\CaptionSeparator}}{}%
1762   \@ifclassloaded{beamer}%
1763     {\defbeamertemplate{caption label separator}{FBcustom}{%
1764       \CaptionSeparator}%
1765     \setbeamertemplate{caption label separator}[FBcustom]}{}%
1766 \else
```

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

```

1767   \ifFB@koma
1768     \renewcommand*\captionformat{\autospace@beforeFDP : }%
1769   \fi
1770   \@ifclassloaded{memoir}%
1771     {\captiondelim{\autospace@beforeFDP : }}%
1772   }{}%
1773   \@ifclassloaded{beamer}%
```



```

1774     {\defbeamertemplate{caption label separator}{FBcolon}{%
1775       {\autospace@beforeFDP : }}%
1776       \setbeamertemplate{caption label separator}[FBcolon]%
1777     }{}%
1778 \fi

```

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

```

1779 \ifFBSHOWOptions
1780   \GenericWarning{* }{%
1781     ***** List of possible options for babel-french *****\MessageBreak
1782     [Default values between brackets when french is loaded *LAST*]%
1783     \MessageBreak
1784     ShowOptions=true [false]\MessageBreak
1785     StandardLayout=true [false]\MessageBreak
1786     GlobalLayoutFrench=false [true]\MessageBreak
1787     PartNameFull=false [true]\MessageBreak
1788     IndentFirst=false [true]\MessageBreak
1789     ListItemsAsPar=true [false]\MessageBreak
1790     ReduceListSpacing=false [true]\MessageBreak
1791     StandardItemizeEnv=true [false]\MessageBreak
1792     StandardEnumerateEnv=true [false]\MessageBreak
1793     StandardItemLabels=true [false]\MessageBreak
1794     ItemLabels=\textendash, \textbullet,
1795       \protect\ding{43},... [\textendash]\MessageBreak
1796     ItemLabeli=\textendash, \textbullet,
1797       \protect\ding{43},... [\textendash]\MessageBreak
1798     ItemLabelii=\textendash, \textbullet,
1799       \protect\ding{43},... [\textendash]\MessageBreak
1800     ItemLabeliii=\textendash, \textbullet,
1801       \protect\ding{43},... [\textendash]\MessageBreak
1802     ItemLabeliv=\textendash, \textbullet,
1803       \protect\ding{43},... [\textendash]\MessageBreak
1804     StandardLists=true [false]\MessageBreak
1805     ListOldLayout=true [false]\MessageBreak
1806     CompactItemize=false [true]\MessageBreak
1807     FrenchFootnotes=false [true]\MessageBreak
1808     AutoSpaceFootnotes=false [true]\MessageBreak
1809     AutoSpacePunctuation=false [true]\MessageBreak
1810     ThinColonSpace=true [false]\MessageBreak
1811     OriginalTypewriter=true [false]\MessageBreak
1812     UnicodeNoBreakSpaces=true [false]\MessageBreak
1813     og= <left quote character>, fg= <right quote character>%
1814     INGuillSpace=true [false]\MessageBreak
1815     EveryParGuill=open, close, none [open]\MessageBreak
1816     EveryLineGuill=open, close, none
1817       [open in LuaTeX, none otherwise]\MessageBreak
1818     InnerGuillSingle=true [false]\MessageBreak
1819     ThinSpaceInFrenchNumbers=true [false]\MessageBreak
1820     SmallCapsFigTabCaptions=false [true]\MessageBreak
1821     CustomiseFigTabCaptions=false [true]\MessageBreak
1822     OldFigTabCaptions=true [false]\MessageBreak

```

```

1823     FrenchSuperscripts=false [true]\MessageBreak
1824     LowercaseSuperscripts=false [true]\MessageBreak
1825     SuppressWarning=true [false]\MessageBreak
1826     \MessageBreak
1827     %*****%
1828     \MessageBreak\protect\frenchsetup{ShowOptions}}
1829 \fi
1830 }

```

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.

```

1831 \AtBeginDocument{%
1832   \providecommand*\xspace{\relax}%

```

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

```

1833   \ifdefined\pdfstringdefDisableCommands
1834     \pdfstringdefDisableCommands{%
1835       \let\up\relax
1836       \let\up\relax
1837       \let\degre\textdegree
1838       \let\degres\textdegree
1839       \def\ieme{e\xspace}%
1840       \def\iemes{es\xspace}%
1841       \def\ier{er\xspace}%
1842       \def\iers{ers\xspace}%
1843       \def\iere{re\xspace}%
1844       \def\ieres{res\xspace}%
1845       \def\FrenchEnumerate#1{#1\degre\space}%
1846       \def\FrenchPopularEnumerate#1{#1\degre)\space}%
1847       \def\No{N\degre\space}%
1848       \def\no{n\degre\space}%
1849       \def\Nos{N\degre\space}%
1850       \def\nos{n\degre\space}%
1851       \def\FB@og{\guillemotleft\space}%
1852       \def\FB@fg{\space\guillemotright}%
1853       \def\at{@}%
1854       \def\circonflexe{\string^}%
1855       \def\tild{\string~}%
1856       \def\boi{\textbackslash}%
1857       \let\bsc\textsc
1858     }%
1859   \fi

```

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

```

1860   \FBprocess@options

```

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

```

1861   \ifFBucsNBSP

```

```

1862 \renewcommand*{\FBmedkern}{\char"202F\relax}%
1863 \renewcommand*{\FBthickkern}{\char"A0\relax}%
1864 \ifFBThinSpaceInFrenchNumbers
1865 \renewcommand*{\FBthousandsep}{\char"202F\relax}%
1866 \else
1867 \renewcommand*{\FBthousandsep}{\char"A0\relax}%
1868 \fi
1869 \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*`!

```

1870 \begingroup
1871 \newcommand{\FBOTone}{OT1}%
1872 \ifx\encodingdefault\FBOTone
1873 \FBWarning{OT1 encoding should not be used for French.%
1874 \MessageBreak
1875 Add \protect\usepackage[T1]{fontenc} to the
1876 preamble\MessageBreak of your document; reported}%
1877 \fi
1878 \endgroup
1879 }

```

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 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.

```

1880 \let\listORI\list
1881 \let\endlistORI\endlist
1882 \def\FB@listVsettings{%
1883 \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1884 \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
1885 \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1886 \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`.

```

1887 \@tempdima=\parskip
1888 \addtolength{\topsep}{-\@tempdima}%

```

```

1889      \addtolength{\partopsep}{\@tempdima}%
1890 }
1891 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1892 \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):

```

\FrenchLabelItem
\Frlabelitemi 1893 \newcommand*{\FrenchLabelItem}{\textemdash}
\Frlabelitemii 1894 \newcommand*{\Frlabelitemi}{\FrenchLabelItem}
\Frlabelitemiii 1895 \newcommand*{\Frlabelitemii}{\FrenchLabelItem}
\Frlabelitemiv 1896 \newcommand*{\Frlabelitemiii}{\FrenchLabelItem}
1897 \newcommand*{\Frlabelitemiv}{\FrenchLabelItem}

```

`\listindentFB` Let's define four dimens `\listindentFB`, `\descindentFB`, `\labelindentFB` and `\descindentFB` `\labelwidthFB` to customise lists' horizontal indentations. They are given silly negative values here in order to eventually enable their customisation in the `\labelindentFB` preamble. They will get reasonable defaults later when entering French (see `\setlabelitemsFB` and `\setlistindentFB`) unless they have been customised.

```

1898 \newdimen\listindentFB
1899 \setlength{\listindentFB}{-1pt}
1900 \newdimen\descindentFB
1901 \setlength{\descindentFB}{-1pt}
1902 \newdimen\labelindentFB
1903 \setlength{\labelindentFB}{-1pt}
1904 \newdimen\labelwidthFB
1905 \setlength{\labelwidthFB}{-1pt}

```

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

```

1906 \newdimen\leftmarginFB
1907 \def\FB@listHsettings{%
1908   \ifFBListItemsAsPar

```

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

```

1909   \itemindent=\labelindentFB
1910   \advance\itemindent by \labelwidthFB
1911   \advance\itemindent by \labelsep
1912   \leftmargini\z@

```

```

1913 \bbl@for\FB@dp {2, 3, 4, 5, 6}%
1914 {\csname leftmargin\romannumeral\FB@dp\endcsname=\labelindentFB}%
1915 \else

```

Default layout: labels hanging into the left margin.

```

1916 \leftmarginFB=\labelwidthFB
1917 \advance\leftmarginFB by \labelsep
1918 \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1919 {\csname leftmargin\romannumeral\FB@dp\endcsname=\leftmarginFB}%
1920 \advance\leftmarginFB by \listindentFB
1921 \fi
1922 \leftmargin=\csname leftmargin\ifnum\@listdepth=\@ne i\else
1923 ii\fi\endcsname
1924 }

```

\itemizeFB New environment for French itemize-lists.

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

```

1925 \def\FB@itemizesettings{%
1926 \ifFBReduceListSpacing
1927 \setlength{\itemsep}{\z@}%
1928 \setlength{\parsep}{\z@}%
1929 \setlength{\topsep}{\z@}%
1930 \setlength{\partopsep}{\z@}%
1931 \@tempdima=\parskip
1932 \addtolength{\topsep}{-\@tempdima}%
1933 \addtolength{\partopsep}{-\@tempdima}%
1934 \fi
1935 \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
1936 \ifFBListOldLayout
1937 \setlength{\leftmargin}{\labelwidth}%
1938 \addtolength{\leftmargin}{\labelsep}%
1939 \addtolength{\leftmargin}{\parindent}%
1940 \else
1941 \FB@listHsettings
1942 \fi
1943 }

```

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

```

1944 \def\itemizeFB{%
1945 \ifnum \@itemdepth >\thr@@\toodeep\else
1946 \advance\@itemdepth by \@ne
1947 \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
1948 \expandafter
1949 \listORI
1950 \csname\@itemitem\endcsname
1951 \FB@itemizesettings
1952 \fi

```

```

1953 }
1954 \let\enditemizeFB\endlistORI

1955 \def\setlabelitemsFB{%
1956   \let\labelitemi\Frlabelitemi
1957   \let\labelitemii\Frlabelitemii
1958   \let\labelitemiii\Frlabelitemiii
1959   \let\labelitemiv\Frlabelitemiv
1960   \ifdim\labelwidthFB<\z@
1961     \settowidth{\labelwidthFB}{\FrenchLabelItem}%
1962   \fi
1963 }
1964 \def\setlistindentFB{%
1965   \ifdim\labelindentFB<\z@
1966     \ifdim\parindent=\z@
1967       \setlength{\labelindentFB}{1.5em}%
1968     \else
1969       \setlength{\labelindentFB}{\parindent}%
1970     \fi
1971   \fi
1972   \ifdim\listindentFB<\z@
1973     \ifdim\parindent=\z@
1974       \setlength{\listindentFB}{1.5em}%
1975     \else
1976       \setlength{\listindentFB}{\parindent}%
1977     \fi
1978   \fi
1979   \ifdim\descindentFB<\z@
1980     \ifFBListItemsAsPar
1981       \setlength{\descindentFB}{\labelindentFB}%
1982     \else
1983       \setlength{\descindentFB}{\listindentFB}%
1984     \fi
1985   \fi
1986 }

```

\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` ($=\texttt{\backslash listFB}$ or $\texttt{\backslash listORI}$) and horizontal spaces (leftmargins) are borrowed from `itemize` lists via `\FB@listHsettings`.

```

1987 \def\enumerateFB{%
1988   \ifnum \@enumdepth >\thr@@\toodeep\else
1989     \advance\@enumdepth by \@ne
1990     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
1991     \expandafter
1992     \list
1993       \csname label\@enumctr\endcsname
1994       {\FB@listHsettings
1995         \usecounter{\@enumctr}\def\makelabel##1{\hss\llap{##1}}}%
1996   \fi
1997 }

```

```
1998 \let\endenumerateFB\endlistORI
```

\descriptionFB Same tuning for the description environment (see `classes.dtx` for the original definition). Customisable dimen `\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.

```
1999 \def\descriptionFB{%
2000     \list{}\FB@listHsettings
2001         \labelwidth=\z@
2002         \ifFBListItemsAsPar
2003             \itemindent=\descindentFB
2004         \else
2005             \itemindent=-\leftmargin
2006             \ifnum\@listdepth=1
2007                 \ifdim\descindentFB=\z@
2008                     \ifdim\listindentFB>\z@
2009                         \leftmargini=\listindentFB
2010                         \leftmargin=\leftmargini
2011                         \itemindent=-\leftmargin
2012                     \fi
2013                 \else
2014                     \advance\itemindent by \descindentFB
2015                 \fi
2016             \fi
2017         \fi
2018         \let\makelabel\descriptionlabel}%
2019 }
2020 \let\enddescriptionFB\endlistORI
```

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

```
2021 \def\update@frenchlists{%
2022     \setlistindentFB
2023     \ifFBReduceListSpacing \let\list\listFB \fi
2024     \ifFBStandardItemizeEnv
2025     \else \let\itemize\itemizeFB \fi
2026     \ifFBStandardItemLabels
2027     \else \setlabelitemsFB \fi
2028     \ifFBStandardEnumerateEnv
2029     \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2030 }
```

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.

```

2031 \def\FB@ufl{\relax}
2032 \def\bbl@frenchlistlayout{%
2033   \ifFBGlobalLayoutFrench
2034   \else
2035     \babel@save\list           \babel@save\itemize
2036     \babel@save\enumerate     \babel@save\description
2037     \babel@save\labelitemi    \babel@save\labelitemii
2038     \babel@save\labelitemiii  \babel@save\labelitemiv
2039     \FB@ufl
2040   \fi
2041 }
2042 \addto\extrasfrench{\bbl@frenchlistlayout}

```

2.13 French indentation of sections

`\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.

```

2043 \def\bbl@frenchindent{%
2044   \ifFBGlobalLayoutFrench
2045   \else
2046     \babel@save\@afterindentfalse
2047   \fi
2048   \ifFBIndentFirst
2049     \let\@afterindentfalse\@afterindenttrue
2050     \@afterindenttrue
2051   \fi}
2052 \def\bbl@nonfrenchindent{%
2053   \ifFBGlobalLayoutFrench
2054     \ifFBIndentFirst
2055       \@afterindenttrue
2056     \fi
2057   \fi}
2058 \addto\extrasfrench{\bbl@frenchindent}
2059 \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 sec-

tion 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`.

```

2060 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2061                 {\PackageInfo{french.ldf}%
2062                 {bigfoot package in use.\MessageBreak
2063                 babel-french will NOT customise footnotes;%
2064                 \MessageBreak reported}}%
2065                 {\let\@footnotemarkORI\@footnotemark
2066                 \def\@footnotemarkFB{\leavevmode\unskip\unkern
2067                                     \,\@footnotemarkORI}%
2068                 \ifFBAutoSpaceFootnotes
2069                 \let\@footnotemark\@footnotemarkFB
2070                 \fi}%
2071                 }

```

`\@makefnmarkFB` We then define `\@makefnmarkFB`, a variant of `\@makefnmark` 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).

```

2072 \newdimen\parindentFFN
2073 \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.

```

2074 \newcommand*{\dotFFN}{.}
2075 \newcommand*{\kernFFN}{\kern .5em}
2076 \newdimen\FBfnindent

```

`\@makefnmarkFB`’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 `\@makefnmark` and `\@makefnmark`. First, save the original definitions.

```

2077 \ifFB@koma
2078   \let\@makefnmarkORI\@makefnmark
2079   \let\@makefnmarkORI\@makefnmark

```

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

```
2080 \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
2081         {\thefootnotemark\dotFFN\kernFFN}
2082 \let\@makefntextFB\@makefntext
2083 \let\@@makefnmarkFB\@@makefnmark
```

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

```
2084 \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2085         {\textsuperscript{\thefootnotemark}}
2086 \let\@makefntextTH\@makefntext
2087 \let\@@makefnmarkTH\@@makefnmark
```

Restore the original definitions.

```
2088 \let\@makefntext\@makefntextORI
2089 \let\@@makefnmark\@@makefnmarkORI
2090 \fi
```

Definitions for the memoir class:

```
2091 \@ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2092 {\newcommand{\@makefntextFB}[1]{%
2093     \def\footscript##1{##1\dotFFN\kernFFN}%
2094     \setlength{\footmarkwidth}{\FBfnindent}%
2095     \setlength{\footmarksep}{-\footmarkwidth}%
2096     \setlength{\footparindent}{\parindentFFN}%
2097     \makefootmark #1}%
2098 }
```

Definitions for the beamer class:

```
2099 \@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.

```
2100 {\def\@makefntextFB#1{%
2101     \def\insertfootnotetext{#1}%
2102     \def\insertfootnotemark{\insertfootnotemarkFB}%
2103     \usebeamertemplate***{footnote}}%
2104 \def\insertfootnotemarkFB{%
2105     \usebeamercolor[fg]{footnote mark}%
2106     \usebeamerfont*{footnote mark}%
2107     \llap{\@thefnmark}\dotFFN\kernFFN}%
2108 }
```

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)!

```
2109 \providecommand*{\insertfootnotemarkFB}{%
2110     \parindent=\parindentFFN
```

```

2111 \rule\z@\footnotesep
2112 \setbox\@tempboxa\hbox{\@thefnmark}%
2113 \ifdim\wd\@tempboxa>\z@
2114     \llap{\@thefnmark}\dotFFN\kernFFN
2115 \fi}
2116 \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.

```

2117 \providecommand\localleftbox[1]{}
2118 \AtBeginDocument{%
2119     \ifpackageloaded{bigfoot}{}%
2120     {\ifdim\parindentFFN<10in
2121         \else
2122             \parindentFFN=\parindent
2123             \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2124         \fi
2125         \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2126         \addtolength{\FBfnindent}{\parindentFFN}%
2127         \let\@makefntextORI\@makefntext
2128         \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).

```

2129         \let\@makefnmarkORI\@makefnmark
2130         \long\def\@makefntext#1{%
2131             \localleftbox{}%
2132             \let\FBeverypar@save\FBeverypar@quote
2133             \let\FBeverypar@quote\relax
2134             \ifFBFrenchFootnotes
2135                 \ifx\footnote\thanks
2136                     \let\@makefnmark\@makefnmarkTH
2137                     \@makefntextTH{#1}
2138                 \else
2139                     \let\@makefnmark\@makefnmarkFB
2140                     \@makefntextFB{#1}
2141                 \fi
2142             \else
2143                 \let\@makefnmark\@makefnmarkORI
2144                 \@makefntextORI{#1}%
2145             \fi
2146             \let\FBeverypar@quote\FBeverypar@save
2147             \localleftbox{\FBeverypar@quote}}%
2148         \else

```

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

```

2149      \@ifclassloaded{memoir}%
2150      {\ifFBFrenchFootnotes
2151        \setlength{\thanksmarkwidth}{\parindentFFN}%
2152        \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2153        \fi
2154      }{}%

```

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

```

2155      \@ifclassloaded{beamer}%
2156      {\ifFBFrenchFootnotes
2157        \ifdim\parindentFFN=1.5em\else
2158          \FBWarning{%
2159            \protect\parindentFFN\space is ineffective%
2160            \MessageBreak within the beamer class.%
2161            \MessageBreak Reported}%
2162          \fi
2163        \fi
2164      }{}%

```

Definition of `\@makefnstext` for all other classes:

```

2165      \long\def\@makefnstext#1{%
2166        \localleftbox{}%
2167        \let\FBeverypar@save\FBeverypar@quote
2168        \let\FBeverypar@quote\relax
2169        \ifFBFrenchFootnotes
2170          \@makefnstextFB{#1}%
2171        \else
2172          \@makefnstextORI{#1}%
2173        \fi
2174        \let\FBeverypar@quote\FBeverypar@save
2175        \localleftbox{\FBeveryline@quote}}%
2176      \fi
2177    }%
2178 }

```

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 `\@makefnstext`.

```

2179 \newcommand*{\AddThinSpaceBeforeFootnotes}{\FBAutoSpaceFootnotesttrue}
2180 \newcommand*{\FrenchFootnotes}{\FBFrenchFootnotesttrue}
2181 \newcommand*{\StandardFootnotes}{\FBFrenchFootnotestfalse}

```

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.

```
2182 \FBclean@on@exit
2183 \ldf@finish\CurrentOption
2184 \let\loadlocalcfg\FB@llc
2185 </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`.

```
2186 <*acadian>
2187 \PackageInfo{acadian.ldf}%
2188 {'acadian' dialect is currently\MessageBreak
2189  *absolutely identical* to the\MessageBreak
2190  'french' language; reported}
2191 </acadian>
2192 <*canadien>
2193 \PackageWarning{canadien.ldf}%
2194 {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2195  it might be removed sooner or later. Please\MessageBreak
2196  use 'acadian' instead; reported}%
2197 \let\l@canadien\l@acadian
2198 \def\CurrentOption{acadian}
2199 </canadien>
2200 <*francais>
2201 \PackageWarning{francais.ldf}%
2202 {Option 'francais' for Babel is *deprecated*,\MessageBreak
2203  it might be removed sooner or later. Please\MessageBreak
2204  use 'french' instead; reported}%
2205 \let\l@francais\l@french
2206 \def\CurrentOption{french}
2207 </francais>
```

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

```
2208 <*frenchb>
2209 \def\bbl@tempa{frenchb}
2210 \ifx\CurrentOption\bbl@tempa
2211  \let\l@frenchb\l@french
2212  \def\CurrentOption{french}
2213  \PackageWarning{babel-french}%
```

```

2214 {Option 'frenchb' for Babel is *deprecated*,\MessageBreak
2215 it might be removed sooner or later. Please\MessageBreak
2216 use 'french' instead; reported}
2217 \else
2218 \def\bbl@tempa{français}
2219 \ifx\CurrentOption\bbl@tempa
2220 \let\l@français\l@french
2221 \def\CurrentOption{french}

```

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

```

2222 \ifx\magnification\@undefined
2223 \PackageWarning{babel-french}%
2224 {Option 'français' for Babel is *deprecated*,\MessageBreak
2225 it might be removed sooner or later. Please\MessageBreak
2226 use 'french' instead; reported}%
2227 \fi
2228 \else
2229 \def\bbl@tempa{canadien}
2230 \ifx\CurrentOption\bbl@tempa
2231 \let\l@canadien\l@acadian
2232 \def\CurrentOption{acadian}
2233 \PackageWarning{babel-french}%
2234 {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2235 it might be removed sooner or later. Please\MessageBreak
2236 use 'acadian' instead; reported}
2237 \fi
2238 \fi
2239 \fi
2240 </frenchb>
2241 <acadian|canadien|frenchb|français>\input french.ldf\relax
2242 <acadian|canadien>\let\extrasacadian\extrasfrench
2243 <acadian|canadien>\let\noextrasacadian\noextrasfrench

```

3 Change History

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

v3.5c	General: Remove grouping inside <code>\@makefntext</code> , <code>\localleftbox</code> and <code>\FBeverypar@quote</code> reset instead. 75	<code>\captionsfrench</code> (undefined in PLain). Prevents loading <code>french.ldf</code> again with <code>acadian</code> option. 14
	<code>\frquote</code> : <code>\FBeverypar@quote</code> 's value now properly reset across level changes. 39	babel-french now requires eTeX. .. 14
	<code>\noextrasfrench</code> : <code>\lccode</code> of quote 0x27 changed from 0x2019 to 0x27 for Unicode engines. 16	Lua function <code>token.get_meaning</code> requires LuaTeX 1.0. 21
v3.5b	General: Reset <code>\FBeverypar@quote</code> locally inside <code>\@makefntext</code> . Needed by <code>\frquote</code> 75	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
	<code>\frquote</code> : New command <code>\FB@addquote@everypar</code> to manage <code>\everypar</code> : <code>\frquote</code> failed when used immediately after a sectioning command. .. 38	New attribute <code>\FB@dialect</code> for the French dialect <code>acadian</code> 20
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. ... 68	New command <code>\FBsetspaces</code> to fine tune spacing independently in French and in French dialects. .. 18
	<code>\descriptionFB</code> : <code>ListItemsAsPar</code> option taken into account for description lists. 71	Shrink/stretch removed in <code>\FBthousandsep</code> 47
	<code>\frenchsetup</code> : New option <code>ListItemsAsPar</code> for displaying lists' items "as paragraphs". 53	Toks <code>\FBcolonsp</code> , <code>\FBthinsp</code> and <code>\FBguillsp</code> removed. 18
v3.4d	<code>\frenchsetup</code> : New test for deciding about utf8 encoding for keys <code>og</code> and <code>fg</code> (the former one fails with LaTeX 2018 release). 59	<code>frenchb.lua</code> : Global 'FBsp' table added; local function 'get_glue' changed into global 'FBget_glue'. 23
v3.4c	<code>\ifFBXeTeX</code> : Reverting to former test, beware of <code>\XeTeXrevision</code> left as <code>\relax</code> by careless testing. 16	<code>\datefrench</code> : Specific code for Plain finally removed (babel bug reported). 40
v3.4b	<code>\datefrench</code> : Do not redefine <code>\date</code> as <code>\frenchdate</code> in French. 40	<code>\extrasfrench</code> : 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> 16
v3.4a	General: <code>\LdfInit</code> checks <code>\FBClean@on@exit</code> instead of	<code>\frenchsetup</code> : Patch for koma-script classes moved here, after <code>\ifFBPartNameFull</code> is defined, so that it applies to <code>\extrasacadian</code> too: <code>\AtEndOfPackage</code> is too late. 54
		v3.3d
		<code>frenchb.lua</code> : 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
		v3.3c
		General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, <code>fontspec</code> is no longer required. .. 67

New command <code>\FBthousandsep</code> to customise numprint.	47	notes numbered over 99.	74
New configurable kerns <code>\FBmedkern</code> , and <code>\FBthickkern</code> suitable for HTML translation. ..	43	<code>\bbl@frenchlistlayout</code> : Execute <code>\update@frenchlists</code> only if <code>GlobalLayoutFrench</code> is false. Delete stuff for lists in <code>\noextrasfrench</code>	71
Reorganise warnings when the caption, subcaption or floatrow packages are loaded before <code>babel/french</code>	50	<code>\frenchsetup</code> : Option <code>GlobalLayoutFrench</code> skipped when French is not the main language.	55
Reset <code>\localleftbox</code> locally inside <code>\@makefntext</code> . Needed by <code>\frquote</code> with LuaTeX.	75	v3.2g	
<code>frenchb.lua</code> : Function ‘ <code>get_glue</code> ’ robustified. ‘ <code>french_punctuation</code> ’ can insert Unicode characters instead of glues.	22	General: Add <code>\boi</code> to redefinitions for bookmarks.	66
<code>\frenchsetup</code> : New option ‘ <code>UnicodeNoBreakSpaces</code> ’ for html translators (LuaLaTeX only).	59	Changed Unicode definition of <code>\boi</code>	44
v3.3b		<code>fontspec</code> defines TU encoding now and no longer loads <code>xunicode.sty</code> . Test changed.	67
General: Generate portmanteau files <code>acadian.ldf</code> , <code>canadien.ldf</code> , <code>frenchb.ldf</code> , and <code>francais.ldf</code> and warn about deprecated options.	77	Issue a warning if <code>beamerarticle.sty</code> is loaded after <code>babel</code>	53
New ‘ <code>if</code> ’ <code>\ifFBfrench</code> to replace <code>\iflanguage</code> test which is based on patterns.	16	<code>\frenchsetup</code> : Minimal list customisation when <code>beamerarticle.sty</code> is loaded.	55
v3.3a		Warn when wrong values are provided to options <code>EveryParGuill</code> or <code>EveryLineGuill</code>	58
General: Compatibility code for pre 2015/10/01 LaTeX release removed, see <code>ltnews23.tex</code>	20	<code>\frquote</code> : Default options of <code>\frquote</code> are no longer engine-dependent.	38
Skip <code>\FBguillskip</code> for LuaTeX replaced by toks <code>\FBguillsp</code> . ..	18	v3.2f	
<code>\captionsfrench</code> : Commands <code>\frenchpartfirst</code> , <code>\frenchpartsecond</code> and <code>\frenchpartnameord</code> added. ...	47	<code>\DecimalMathComma</code> : Fixed conflict with the <code>icomma</code> package.	45
<code>\FBthinspace</code> : Skips <code>\FBcolonskip</code> and <code>\FBthinskip</code> replaced by toks <code>\FBcolonsp</code> and <code>\FBthinsp</code>	17	v3.2e	
<code>\frenchsetup</code> : <code>\frenchbsetup</code> is now an alias for <code>\frenchsetup</code> . .	53	General: Add missing redefinitions for <code>\leftmarginiv</code> , <code>\leftmarginvi</code> . Suggested by J.F. Burnol.	68
Options <code>INGuillSpace</code> , <code>ThinColonSpace</code> no longer delayed <code>AtBeginDocument</code>	53	<code>\DecimalMathComma</code> : <code>\DecimalMathComma</code> didn't work with LuaTeX. Fixed now.	45
<code>\frquote</code> : <code>\FB@quotespace</code> (kern), changed into <code>\FB@guillspace</code> . .	39	v3.2d	
v3.2h		<code>\descriptionFB</code> : Changed <code>\listindentFB</code> to <code>\descindentFB</code> which defaults to <code>\listindentFB</code> . <code>\leftmargini</code> reduced when <code>\descindentFB</code> is null.	71
<code>\@makefntextFB</code> : With <code>beamer.cls</code> , add <code>\llap</code> to <code>\@thefnmark</code> for		v3.2c	
		General: New LuaTeX attribute <code>\FB@spacing</code>	20
		Newif <code>\ifFB@spacing</code> and new commands <code>\FB@spacingon</code> ,	

\FB@spacingoff to control space tuning in French.	20	\xspace is now active in \fg in and outside French.	37
Switch \ifFB@spacing added to the four French shorthands.	33	v3.1m	
\FB@xetex@punct@french: Switch \ifFB@spacing added to all \XeTeXinterchartoks commands.	31	frenchb.lua: new_glue_scaled returns nil in case of invalid font table (i.e. lcircle1.pfb). In such cases babel-french leaves the node list unchanged.	24
\FBthinspace: Change .16667em to .5\fontdimen2\font to get in XeTeX and pdfTeX the same spacing as in LuaTeX.	17	v3.1l	
\frenchsetup: Add a warning about options og/fg for old XeTeX or LuaTeX engines requiring active characters.	59	General: Add a variant of \babel@savevariable to save \XeTeXcharclass(es) in a loop. .	31
\NoAutoSpacing: New definition based on \FB@spacing@off common to all engines.	36	frenchb.lua: font.getfont(fid) possibly returns nil even for a positive fid (i.e. AMS lcircle1.pfb). Reported by François Legendre. .	24
\ttfamilyFB: New definitions of \ttfamilyFB and co, common to all engines, based on \FB@spacing@off and \FB@spacing@on.	35	\FB@luatex@punct@french: Use \babel@save to save and restore \shorthandon and \shorthandoff.	29
v3.2b		\FB@xetex@punct@french: Save and restore \XeTeXinterchartokenstate, \shorthandon, \shorthandoff using \babel@savevariable and \babel@save, \XeTeXcharclass(es) using \FB@savevariable@loop.	31
General: Load ltuatex.tex for plain LuaTeX to ensure \newattribute is defined.	20	v3.1k	
Warning added when the subcaption package is loaded before babel/french.	50	General: (pdfTeX shorthands) test on \lastskip changed from 0pt to 1sp for active punctuation for consistency with XeTeX and LuaTeX.	33
frenchb.lua: glue_spec removed; starting with LuaTeX 0.95, glue specifications fit in glue.	24	\FB@xetex@punct@french: Thin glues (less than 1sp) should not trigger space insertion before high punctuation. Add a check on \lastkip.	31
\ifFB@xetex@punct: New counter \FB@nonchar needed for non characters: it's value will be 4095 for new engines and 255 for older ones.	17	v3.1j	
\NoAutoSpacing: \NoAutoSpacing made robust.	36	General: Loading luatexbase.sty is no longer needed with LaTeX release 2015/10/01 or later.	20
v3.2a		\frquote: \fr@quote completely rewritten: \leavevmode added and explicitly save/retore \everypar and \localleftbox instead of using a group in order to ensure compatibility with package wrapfig.	39
\@makefntextFB: beamer.cls requires a specific definition of \@makefntextFB (pointed out by DB). The same is true for memoir and koma-script classes (done). .	73		
\fg: \xspace moved from \FB@fg to \fg: \xspace messes up \frquote, pointed out by Sonia Labetoulle. As a side effect			

\PackageWarning is undefined in Plain, use \fb@warning instead.	39	changed when option CustomiseFigTabCaptions is set to false.	50
v3.1i		\FBprocess@options: Bug fix for the beamer class: figure and table captions are now consistent with babel-french's documentation. Pointed out by Denis Bitouzé. . . .	64
General: \nombre command changed when numprint.sty is not loaded: only one warning, no error.	47	Definition of \captionformat and \captiondelim changed when option CustomiseFigTabCaptions is set to false.	64
Remove restriction about loading numprint.sty after babel.	52	\FBthinspace: \FBthinspace is no longer a kern but a skip (babel-french adds a nobreak penalty before it).	17
\frquote: \luatexlocalleftbox changed to \localleftbox by new LaTeX release 2015/10/01. . . .	39		
v3.1h		v3.1e	
General: french.cfg from e-french conflicts with babel-french. Do NOT load it (no need for .cfg files with babel-french anyway).	77	\frenchsetup: Corrected typo: SmallCapsFigTabcaptions instead of SmallCapsFigTabCaptions. Pointed out by Céline Chevalier. . .	53
v3.1g		v3.1d	
General: Lua function french_punctuation 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 listings, numprint and natbib are loaded too early or too late vs babel.	52
Use Babel defined loops \bbl@for instead of \@for borrowed from file ltcntrl.dtx (\@for is undefined in Plain).	30	v3.1c	
frenchb.lua: Flag addgl set to false for '«' at the end of an \hbox or a paragraph or when followed by a null glue (i.e. springs).	28	frenchb.lua: Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André. . . .	25
flag addgl set to false for '»' at the beginning of an \hbox or a paragraph or a tabular 'l' and 'c' columns.	27	v3.1b	
Node HLIST added; node TEMP added for the first node of \hboxes.	23	frenchb.lua: Add a check for null fid in french_punctuation (Tikz \nullfont). Bug pointed out by Paul Gaborit.	25
\captionsfrench: \partname's definition depends now on flag PartNameFull. No need to redefine it in \frenchbsetup.	47	\captionsfrench: Change \scshape to customisable \FBfigtabshape for \figurename and \tablename.	47
\frenchsetup: Bug fix for koma-scripts classes: a spurious dot was added by the \partformat command.	54	\fprimo): Removed \lowercase from definitions of \FrenchEnumerate, ... \No and co: \up already does the conversion.	43
PartNameFull now just sets the flag, nothing to add to \captionsfrench when false. . . .	53	\frenchsetup: New option SmallCapsFigTabCaptions.	53
v3.1f		\ieres: Removed \lowercase from definitions of \ieme and co: \up already does the conversion. . . .	43
General: \FBCaption@Separator			

v3.1a

General: fontspec is not required for T1 fonts used with the luainputenc.sty package.	67
Misplaced \fi for plain formats. .	20
New command \frquote for imbedded or long French quotations.	38
frenchb.lua: Added flag addgl which must also be true when prev or next is not a char (i.e. \kern0 in «\texttt{a}»).	27
Codes 0x13 and 0x14 added for French quotes in T1-encoding. .	22
Look ahead when next is a kern (i.e. in «\texttt{a} »).	28
\frenchsetup: Codes 0x13 and 0x14 added for French quotes in T1-encoding. Support for older versions of LuaTeX and XeTeX dropped.	59
New options InnerGuillSingle, EveryParGuill and EveryLineGuill to control \frquote.	53

v3.0c

General: babel-french requires babel-3.9i.	14
Just load luatexbase.sty instead of luaotfload.sty with plain formats.	20
No need to define \l@french as \lang@french, babel.def (3.9j) takes care for this.	15
frenchb.lua: Null glues should not trigger space insertion before high punctuation. Bug pointed out by Benoit Rivet for the 'lstlisting' environment of the listings package.	25
\frenchsetup: New option INGuillSpace.	53
No list customisation when beamer class is loaded.	55

v3.0b

General: frenchb.lua was not found by Lua function dofile (not kpathsea aware). Call function kpse.find_file first, as suggested by Paul Gaborit.	29
Require luatexbase with LaTeX2e in case fontspec has not been loaded before babel.	20

v3.0a

General:	
\bbl@nonfrenchguillemets deleted, use \babel@save instead.	38
\LdfInit checks \captionsfrench instead of \datefrench to avoid a conflict with papertex.cls which loads datetime.sty.	14
french.cfg will be loaded (if found) instead of frenchb.cfg. NO NEED for .cfg files in French anyway. .	77
In Plain, provide a substitute for \PackageWarning and \PackageInfo.	14
Merging of \captionsfrenchb, \captionsfrançais with \captionsfrench deleted in favor of new babel 3.9 syntax.	48
More informative, less TeXnical warning about \makecaption. .	50
New flag \ifFB@luatex@punct for 'high punctuation' management with LuaTeX engines.	17
New handling of 'high punctuation' through callbacks with LuaTeX engines.	20
No warning about \makecaption for SMF classes.	50
Options processing completely reorganised, now \babel@save and \babel@savevariable are usable for French.	53
Support for options frenchb, français, canadien, acadien changed.	14
Test \ifXeTeX changed to \ifFBunicode and 'xltextra' changed to 'fontspec'.	67
\CaptionSeparator: Remove \FBCaption@SeparatorORI, use \babel@save instead.	49
\captionsfrench: Take advantage of babel's \SetString commands for captionnames.	47
\datefrench: Take advantage of babel's \SetString commands for \datefrench. Doesn't work with Plain (yet?).	40

\descriptionFB: Added	XeTeX / active).	37
\listindentFB to \itemindent. Suggested by Denis Bitouzé. . . .		71
\extrasfrench: Take advantage of babel's \babel@savevariable to handle apostrophe's \lccode. . .		16
\FB@fg: Definitions of \FB@og and \FB@fg now depend on punctuation handling (LuaTeX /		
	\FBprocess@options: With koma-script and memoir class, customise \captionformat and \captiondelim.	64
	\frenchsetup: New options OldFigTabCaptions and CustomiseFigTabCaptions.	53