

A Babel language definition file for French

frenchb.dtx v3.5I, 2020/10/10

Daniel Flipo
daniel.flipo@free.fr

Contents

1	The French language	2
1.1	Basic interface	2
1.2	Customisation	5
1.2.1	\frenchsetup	5
1.2.2	Caption names	9
1.2.3	Figure and table captions	9
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	34
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...	51
2.10	More checks about packages' loading order	52
2.11	Setup options: keyval stuff	53
2.12	French lists	66
2.13	French indentation of sections	70
2.14	Formatting footnotes	71
2.15	Clean up and exit	75
2.16	Files frenchb.ldf, francais.ldf, canadien.ldf and acadian.ldf	75
3	Change History	77

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, Ulrike Fisher and Marcel Krüger. 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 v3.5l are listed in subsection 1.4 p. 11.

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

1.1 Basic interface

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

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

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

²

A variant `acadian` of `french` is provided; it is originally identical to `french` but can be customised independently in terms of patterns, punctuation spacing, captions, etc. Both variants can be used together inside the same document.

`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 v3.5l and was last revised on 2020/10/10.

²Always use `french` as option name for the French language, former aliases `frenchb` or `français` are *depreciated*; 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. 9.

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

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

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

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

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

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

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

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

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

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

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

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

2. `\frenchdate{<year>}{<month>}{<day>}` helps typesetting dates in French: `\frenchdate{2001}{01}{01}` will print 1^{er} janvier 2001 in a box without any linebreak.
3. A command `\up` is provided to typeset superscripts like `M\up{me}` (abbreviation for “Madame”), `l\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) or for angles in math mode.
8. In math mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the `TEXbook` p. 134). The command `\DecimalMathComma` makes the comma behave as an ordinary character *when the current language is French* (no space added); as a counterpart, if `\DecimalMathComma` is active, an explicit space has to be added in lists and intervals: `[$[0,\ 1]$, $(x,\ y)$`. `\StandardMathComma` switches back to the standard behaviour of the comma in French.
The `icomma` package is an alternative workaround.
9. A command `\nombre` was provided in 1.x versions to easily format numbers in slices of three digits separated either by a comma in English or with a space in French; `\nombre` is now mapped to `\numprint` from `numprint.sty`, which should be loaded *after* `Babel`, 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 `'\ier juin'` will print `'1er juin'` (no need for a forced space after `\ier`).

1.2 Customisation

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

1.2.1 `\frenchsetup{options}`

`\frenchsetup{}` and `\frenchbsetup{}` are synonymous; the latter should be preferred as the language name for French in `Babel` is no longer `frenchb` but `french`. `\frenchsetup{ShowOptions}` prints all available options to the `.log` file, it is just meant as a remainder of the list of offered options. As usual with `keyval` syntax, boolean options (as `ShowOptions`) can be entered as `ShowOptions=true` or just `ShowOptions`, the `=true` part can be omitted.

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

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

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

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

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

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

<p>Text starting at ‘parindent’ \leq Leftmargin — first item running on two lines or more... — first second level item on two lines... — next one... — second item...</p>	<p>Text starting at ‘parindent’ \leq Leftmargin — first item running on two lines or more... — first second level item on two lines... — next one... — second item...</p>
Default French layout	With <code>ListItemsAsPar=true</code>

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

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

`StandardEnumerateEnv=true` (`false*`); `babel-french` redefines `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 `\ding{43}` requires loading the `pifont` package.

`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

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

packages that customise lists too. This option is just a shorthand setting all four options `StandardListSpacing=true`, `StandardItemizeEnv=true`, `StandardEnumerateEnv=true` and `StandardItemLabels=true`.

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

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

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

`AutoSpacePunctuation=false` (`true`); in French, the user *should* input a space before the four characters ' ; ! ? ' but as many people forget about it (even among native French writers!), the default behaviour of `babel-french` is to automatically typeset non-breaking spaces the width of which is either `\FBthinspace` (defaults to a thin space) before ' ; ' ! ' ? ' or `\FBcolonspace` (defaults to `\space`) before ' : ' ; the defaults follow the French 'Imprimerie Nationale's recommendations. This is convenient in most cases but can lead to addition of spurious spaces in URLs, in MS-DOS paths or in timetables (10:55)—this no longer occurs with LuaTeX—, except if they are typed in `\texttt` or verbatim mode. When the current font is a monospaced (typewriter) font, no spurious space is added in that case ⁷, so the default behaviour of `babel-french` in that area should be fine in most circumstances.

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

`ThinColonSpace=true` (`false`) changes the 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).

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

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

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

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

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

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

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

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

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

`SmallCapsFigTabCaptions=false (true*)`; when set to `false`, `\figurename` and `\tablename` will be printed in French captions as “Figure” and “Table” instead of being printed in small caps (the default). The same result can be achieved by defining `\FBfigtabshape` as `\relax` before loading `babel-french` (in a document class f.i.).

⁹Actually without stretch nor shrink.

`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 in French and warns if it fails to do so.

`OldFigTabCaptions=true (false)` is to be used *only* 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 up-casing of superscripts (for instance when they are moved to page headers). Making this option `false` will disable this behaviour (not recommended).

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

Options' order – Please remember that options are read in the order they appear in the `\frenchsetup{}` command. Someone wishing that `babel-french` leaves the layout of lists and footnotes untouched but caring for indentation of first paragraph of sections should choose

`\frenchsetup{StandardLayout,IndentFirst}` to get the expected layout. The reverse order `\frenchsetup{IndentFirst,StandardLayout}` would lead to option `IndentFirst` being overwritten by `StandardLayout`.

1.2.2 Caption names

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

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

1.3 Hyphenation checks

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

- run pdfLaTeX on the following file:

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

- check the hyphenations proposed by T_EX in your log-file; in French you should get with both 7-bit and 8-bit encodings
si-gnal contai-ner évé-ne-ment al-gèbre.
Do not care about how accented characters are displayed in the log-file, what matters is the position of the '-' hyphen signs *only*.

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

Frequent mismatches:

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

1.4 Changes

What's new in version 3.5?

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

Version 3.5b fixes a bug due to wrong `\everypar`’s management in `\frquote{}`; it showed up when `\frquote{}` immediately followed a sectioning command.

Starting with version 3.5d, a new option `StandardListSpacing` has been added to supersede `ReduceListSpacing`.

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

Version 3.5g fixes a long standing bug affecting LuaTeX: legacy kerning was disabled for Type1 fonts since v3.1g (2015).

Version 3.5j also fixes a long standing bug affecting koma-script, memoir et beamer classes: redefinitions of the caption separator (commands `\captionformat`, `\captiondelim`, etc.) are now taken into account properly.

Version 3.5k is a cleanup release:

- the translations in French of `\figurename` and `\tablename` no longer hold font changing commands (switch to small caps), the font switch has been moved to `\fnum@figure` and `\fnum@table` as suggested by Axel Sommerfeldt.
- Package `caption` can now be loaded whether before or after `babel`, indifferently.
- `\pdfstringdefDisableCommands` is no longer used: as suggested by the LaTeX3 team, all commands requiring special care in `hyperref`’s bookmarks are now defined using `\textorpdfstring{}`.

What's new in version 3.4?

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

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

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

Version 3.4 requires eTeX and LuaTeX 1.0.4 or newer.

What's new in version 3.3?

In version 3.3d the automatic insertion of non-breaking spaces before the colon character has been improved *with engine LuaTeX only*: a spurious space is no longer

inserted in strings like `http://mysite`, `C:\Program Files` or `10:55`. Unfortunately, my attempts to do the same with XeTeX or pdfTeX were unsuccessful.

A few internal changes have been made in version 3.3c to improve the conversion into HTML of non-breaking spaces added by `babel-french`. Usage of `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.ldf, this change allows for cleaner definitions of dates and captions for the Unicode engines LuaTeX and XeTeX and also provides a simpler syntax for end-users, see section 1.2.2 p.9.
- `\frenchsetup{}` options management has been completely reworked; two new options added.
- Canadian French didn't work as a normal Babel's dialect, it should now; btw. the French language should now be loaded as `french`, *not as* `frenchb` or `français` and preferably as a *global* option of `\documentclass`. Some tolerance still exists in v3.0, but do not rely on it.
- `babel-french` no longer loads `frenchb.cfg`: customisation should definitely be done using `\frenchsetup{}` options.
- Description lists labels are now indented; try setting `\descindentFB=0pt` (or `\listindentFB=0pt` for all lists) in the preamble if you don't like it.
- The last but not least change affects the (recent) LuaTeX-based engines, (this means version 0.76 as included in TL2013 and up): active characters are no longer used in French for 'high punctuation' ¹⁰. Functionalities and user interface are unchanged.

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

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

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

2 The code

2.1 Initial setup

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

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

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

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

Quit if eTeX is not available.

```
21 \let\bb1@tempa\relax
22 \begingroup\expandafter\expandafter\expandafter\endgroup
23 \expandafter\ifx\cename eTeXversion\endcename\relax
24   \let\bb1@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 \bb1@tempa
```

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

```
31 \let\bb1@tempa\relax
32 \ifdefined\babeltags
33 \else
34   \let\bb1@tempa\endinput
35   \ifdefined\PackageError
36     \PackageError{french.ldf}
37     {babel-french requires babel v.3.16.\MessageBreak
38     Aborting here}
39     {Please upgrade Babel!}
40   \else
```

```

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

```

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

```

47 \def\FB@nopatterns{%
48   \ifdefined\l@nohyphenation
49     \adddialect\l@french\l@nohyphenation
50     \edef\bbl@nulllanguage{\string\language=nohyphenation}%
51   \else
52     \edef\bbl@nulllanguage{\string\language=0}%
53     \adddialect\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
58   \adddialect\l@canadien\l@acadian
59 \else
60   \adddialect\l@acadian\l@french
61   \adddialect\l@canadien\l@french
62 \fi

```

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

```

63 \providehyphenmins{french}{\tw@\thr@@}
64 \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.

```

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

```

`\ifBunicode` 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”.
`\ifBLuaTeX`
`\ifB XeTeX` Let’s define three new ‘if’: `\ifBLuaTeX`, `\ifB XeTeX` and `\ifBunicode` which will be true for XeTeX and LuaTeX engines and false for 8-bits engines.

```

80 \newif\ifBunicode
81 \newif\ifBLuaTeX
82 \newif\ifB XeTeX
83 \begingroup\expandafter\expandafter\expandafter\endgroup
84 \expandafter\ifx\csname luatexversion\endcsname\relax
85 \else
86   \FBunicodetrue \BLuaTeXtrue
87 \fi
88 \begingroup\expandafter\expandafter\expandafter\endgroup
89 \expandafter\ifx\csname XeTeXrevision\endcsname\relax
90 \else
91   \FBunicodetrue \B XeTeXtrue
92 \fi

```

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

```

93 \newif\ifBFrench

```

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

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

```

94 \def\extrasfrench{%
95   \FBfrenchtrue
96   \babel@savevariable{\lccode"27}%
97   \lccode"27="27
98   \ifBunicode
99     \babel@savevariable{\lccode"2019}%
100    \lccode"2019="2019
101   \fi
102 }
103 \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.

```

104 \addto\extrasfrench{\bbl@frenchspacing}
105 \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.

```
106 \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).

```
107 \newif\ifFB@luatex@punct
108 \ifBLuaTeX
109   \ifnum\luatexversion<100
110     \ifx\PackageWarning@\undefined
111       \fb@warning{Please upgrade LuaTeX to version 1.0.4 or above!\\%
112         babel-french will make high punctuation characters (;!?)\\%
113         active with LuaTeX < 1.0.4.}%
114     \else
115       \PackageWarning{french.ldf}{Please upgrade LuaTeX
116         to version 1.0.4 or above!\MessageBreak
117         babel-french will make high punctuation characters%
118         \MessageBreak (;!?) active with LuaTeX < 1.0.4;%
119         \MessageBreak reported}%
120     \fi
121   \else
122     \FB@luatex@puncttrue\FB@active@punctfalse
123   \fi
124 \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.

```
125 \newcount\FB@nonchar
126 \newif\ifFB@xetex@punct
127 \ifdefined\XeTeXinterchartokenstate
128   \FB@xetex@puncttrue\FB@active@punctfalse
129   \ifdim\the\XeTeXversion\XeTeXrevision pt<0.99994pt
130     \FB@nonchar=255 \relax
131   \else
132     \FB@nonchar=4095 \relax
133   \fi
134 \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.

```

135 \newcommand*{\FBguillspace}{\hskip .8\fontdimen2\font
136           plus .3\fontdimen3\font
137           minus .8\fontdimen4\font \relax}
138 \newcommand*{\FBcolonspace}{\space}
139 \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.

```

140 \ifLaTeXe
141   \newcommand*{\FBsetspaces}[5][french]{%
142     \def\bb@tempa{french}\def\bb@tempb{#1}%
143     \ifx\bb@tempa\bb@tempb \def\bb@tempb{}\fi
144     \@namedef{\bb@tempb FB#2space}{\hskip #3\fontdimen2\font
145                                   plus #4\fontdimen3\font
146                                   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".

```

147   \ifFB@luatex@punct
148     \ifx\bb@tempb\FB@acadian
149       \directlua{
150         FBsp.#2.gl.ac[1] = #3
151         FBsp.#2.gl.ac[2] = #4
152         FBsp.#2.gl.ac[3] = #5
153         if #3 > 0.6 then
154           FBsp.#2.ch.ac = 0xA0
155         elseif #3 > 0.2 then
156           FBsp.#2.ch.ac = 0x202F
157         else
158           FBsp.#2.ch.ac = 0x200B
159         end
160       }%
161     \fi
162   \fi
163 }
164 \@onlypreamble\FBsetspaces
165 \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`.

```

166 \ifLaTeXe
167   \addto\extrasfrench{%
168     \ifFB@luatex@punct
169       \edef\bbl@tempa{\detokenize\expandafter{\language}}}%
170       \edef\bbl@tempb{\detokenize{french}}}%
171       \ifx\bbl@tempa\bbl@tempb \FB@dialect=0 \relax
172       \else
173         \FB@dialect=1 \relax
174       \fi

```

When first entering French, we must 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!

```

174   \ifdefined\FB@once\else
175     \set@glue@table{colon}%
176     \set@glue@table{thin}%
177     \set@glue@table{guill}%
178     \def\FB@once{}%
179   \fi
180 \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).

```

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

```

Same for `\FBcolonspace`:

```

186   \ifcsname\language FBcolonspace\endcsname
187     \babel@save\FBcolonspace
188     \renewcommand*{\FBcolonspace}{%
189       \csname\language FBcolonspace\endcsname}%
190   \fi

```

And for `\FBguillspace`:

```

191   \ifcsname\language FBguillspace\endcsname
192     \babel@save\FBguillspace
193     \renewcommand*{\FBguillspace}{%
194       \csname\language FBguillspace\endcsname}%
195   \fi
196 }
197 \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.

```
198 \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).

```
199 \newcommand*{\FB@spacing@on}{%
200   \ifFB@luatex@punct
201     \FB@spacing=1 \relax
202   \else
203     \FB@spacingtrue
204   \fi}
205 \newcommand*{\FB@spacing@off}{%
206   \ifFB@luatex@punct
207     \FB@spacing=0 \relax
208   \else
209     \FB@spacingfalse
210   \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.

```
211 \ifFB@luatex@punct
212   \ifdefined\newluafunction\else
```

This code is for Plain: load `ltxluatex.tex` if it hasn’t been loaded before Babel.

```
213   \input ltxluatex.tex
214   \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@addGUIILspace` 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.

```
215   \newattribute\FB@spacing      \FB@spacing=1 \relax
216   \newattribute\FB@addDPspace  \FB@addDPspace=1 \relax
217   \newattribute\FB@addGUIILspace \FB@addGUIILspace=0 \relax
218   \newattribute\FB@ucsNBSP     \FB@ucsNBSP=0 \relax
```

```

219 \newattribute\FB@dialect      \FB@dialect=0 \relax
220 \ifLaTeXe
221   \PackageInfo{french.ldf}{No need for active punctuation
222     characters\MessageBreak with this version
223     of LuaTeX!\MessageBreak reported}
224 \else
225   \fb@info{No need for active punctuation characters\
226     with this version of LuaTeX!}
227 \fi

```

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

```

228 \newcommand*{\set@glue@table}[1]{%
229   \directlua {
230     local s = token.get_meaning("FB#1space")
231     local t = FBget_glue(s)
232     if t then
233       FBsp.#1.gl.fr = t
234       if not FBsp.#1.gl.ac[1] then
235         FBsp.#1.gl.ac = t
236       end
237       if FBsp.#1.gl.fr[1] > 0.6 then
238         FBsp.#1.ch.fr = 0xA0
239       elseif FBsp.#1.gl.fr[1] > 0.2 then
240         FBsp.#1.ch.fr = 0x202F
241       else
242         FBsp.#1.ch.fr = 0x200B
243       end
244       if not FBsp.#1.ch.ac then
245         FBsp.#1.ch.ac = FBsp.#1.ch.fr
246       end
247     else
248       texio.write_nl('term and log', '')
249       texio.write_nl('term and log',
250         '*** french.ldf warning: Unexpected syntax in FB#1space,')
251       texio.write_nl('term and log',
252         '*** french.ldf warning: LuaTeX table FBsp unchanged.')
253       texio.write_nl('term and log',
254         '*** french.ldf warning: Consider using FBsetspaces to ')
255       texio.write('term and log', 'customise FB#1space.')
256       texio.write_nl('term and log', '')
257     end
258   }%
259 }
260 \fi
261 </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).

```

262 <*Lua>
263 local FB_punct_thin =
264   {[string.byte("!")] = true,
265    [string.byte("?")] = true,
266    [string.byte(";")] = true}
267 local FB_punct_thick =
268   {[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 ‘»’.

```

269 local FB_punct_left =
270   {[string.byte("!")] = true,
271    [string.byte("?")] = true,
272    [string.byte(";")] = true,
273    [string.byte(":")] = true,
274    [0x14]           = true,
275    [0xBB]           = true}
276 local FB_punct_right =
277   {[0x13]           = true,
278    [0xAB]           = true}

```

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

```

279 local FB_punct_null =
280   {[string.byte("!")] = true,
281    [string.byte("?")] = true,
282    [string.byte("[")] = true,
283    [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.

```

284   [0xA0]           = true,
285   [0x202F]         = true}
286 local FB_guil_null =
287   {[0xA0]           = true,
288    [0x202F]         = true}

```

Local definitions for nodes:

```

289 local new_node      = node.new
290 local copy_node     = node.copy
291 local node_id       = node.id
292 local HLIST         = node_id("hlist")
293 local TEMP          = node_id("temp")
294 local KERN          = node_id("kern")
295 local GLUE          = node_id("glue")
296 local GLYPH         = node_id("glyph")
297 local PENALTY       = node_id("penalty")

```

```

298 local nobreak      = new_node(PENALTY)
299 nobreak.penalty    = 10000
300 local nbspace      = new_node(GLYPH)
301 local insert_node_before = node.insert_before
302 local insert_node_after  = node.insert_after
303 local remove_node      = node.remove

```

Commands `\FBthinspace`, `\FBcolonspace` and `\FBguillspace` are converted ‘AtBeginDocument’ 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.

```

304 function FBget_glue(toks)
305   local t = nil
306   local f = string.match(toks,
307     "[^%w]hskip%s*([%d%.]*)%s*[^%w]fontdimen 2")
308   if f == "" then f = 1 end
309   if tonumber(f) then
310     t = {tonumber(f), 0, 0}
311     f = string.match(toks, "plus%s*([%d%.]*)%s*[^%w]fontdimen 3")
312     if f == "" then f = 1 end
313     if tonumber(f) then
314       t[2] = tonumber(f)
315       f = string.match(toks, "minus%s*([%d%.]*)%s*[^%w]fontdimen 4")
316       if f == "" then f = 1 end
317       if tonumber(f) then
318         t[3] = tonumber(f)
319       end
320     end
321   elseif string.match(toks, "[^%w]F?B?thinspace") then
322     t = {0.5, 0, 0}
323   elseif string.match(toks, "[^%w]space") then
324     t = {1, 1, 1}
325   end
326   return t
327 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`.

```

328 FBsp = {}
329 FBsp.thin = {}
330 FBsp.thin.gl = {}
331 FBsp.thin.gl.fr = {.5, 0, 0} ; FBsp.thin.gl.ac = {}
332 FBsp.thin.ch = {}
333 FBsp.thin.ch.fr = 0x202F ; FBsp.thin.ch.ac = nil
334 FBsp.colon = {}
335 FBsp.colon.gl = {}
336 FBsp.colon.gl.fr = { 1, 1, 1} ; FBsp.colon.gl.ac = {}
337 FBsp.colon.ch = {}
338 FBsp.colon.ch.fr = 0xA0 ; FBsp.colon.ch.ac = nil
339 FBsp.guill = {}
340 FBsp.guill.gl = {}
341 FBsp.guill.gl.fr = {.8, .3, .8} ; FBsp.guill.gl.ac = {}
342 FBsp.guill.ch = {}

```

```
343 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`.

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

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

```
368 local FBspacing      = luatexbase.attributes['FB@spacing']
369 local addDPspace     = luatexbase.attributes['FB@addDPspace']
370 local addGUILspace   = luatexbase.attributes['FB@addGUILspace']
371 local FBucsNBSP     = luatexbase.attributes['FB@ucsNBSP']
372 local FBdialect     = luatexbase.attributes['FB@dialect']
373 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`.

```
374 -- Main function (to be added to the kerning callback).
375 local function french_punctuation (head)
```

Restore the built-in kerning for 8-bits fonts.

```
376   node.kerning(head)
377   for item in node.traverse_id(GLYPH, head) do
378     local lang = item.lang
```



```

379     local char = item.char
Skip glyphs not concerned by French kernings.
380     if (lang == FR_fr or lang == FR_ca) and
381         (FB_punct_left[char] or FB_punct_right[char]) then
382         local fid = item.font
383         local attr = item.attr
384         local FRspacing = has_attribute(item, FBspacing)
385         FRspacing = FRspacing and FRspacing > 0
386         local FRucsNBSP = has_attribute(item, FBucsNBSP)
387         FRucsNBSP = FRucsNBSP and FRucsNBSP > 0
388         local FRdialect = has_attribute(item, FBdialect)
389         FRdialect = FRdialect and FRdialect > 0
390         local SIG = has_attribute(item, addGUILspace)
391         SIG = SIG and SIG > 0
392         if FRspacing and fid > 0 then
393             if FB_punct_left[char] then
394                 local prev = item.prev
395                 local prev_id, prev_subtype, prev_char
396                 if prev then
397                     prev_id = prev.id
398                     prev_subtype = prev.subtype
399                     if prev_id == GLYPH then
400                         prev_char = prev.char
401                     end
402                 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.

```

403         local is_glue = prev_id == GLUE
404         local glue_wd
405         if is_glue then
406             glue_wd = prev.width
407         end
408         local realglue = is_glue and glue_wd > 1

```

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

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

```

409         if FB_punct_thin[char] or FB_punct_thick[char] then
410             local SBDP = has_attribute(item, addDPspace)
411             local auto = SBDP and SBDP > 0
412             if FB_punct_thick[char] and auto then
413                 local next = item.next

```

```

414         local next_id
415         if next then
416             next_id = next.id
417         end
418         if next_id and next_id == GLYPH then
419             auto = false
420         end
421     end
422     if auto then
423         if (prev_char and FB_punct_null[prev_char]) or
424            (is_glue and glue_wd <= 1) or
425            (prev_id == HLIST and prev_subtype == 3) or
426            (prev_id == TEMP) then
427             auto = false
428         end
429     end
430     local fbglue
431     local t
432     if FB_punct_thick[char] then
433         if FRdialect then
434             t = FBsp.colon.gl.ac
435             nspace.char = FBsp.colon.ch.ac
436         else
437             t = FBsp.colon.gl.fr
438             nspace.char = FBsp.colon.ch.fr
439         end
440     else
441         if FRdialect then
442             t = FBsp.thin.gl.ac
443             nspace.char = FBsp.thin.ch.ac
444         else
445             t = FBsp.thin.gl.fr
446             nspace.char = FBsp.thin.ch.fr
447         end
448     end
449     fbglue = new_glue_scaled(fid, t)

```

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

```

450         if (realglue or auto) and fbglue then
451             if realglue then
452                 head = remove_node(head,prev,true)
453             end
454             if (FRucsNBSP) then
455                 nspace.font = fid
456                 nspace.attr = attr
457                 insert_node_before(head,item,copy_node(nspace))
458             else
459                 nobreak.attr = attr
460                 fbglue.attr = attr
461                 insert_node_before(head,item,copy_node(nobreak))
462                 insert_node_before(head,item,copy_node(fbglue))
463             end
464         end

```

Let's consider '»' now (the only remaining glyph of `FB_punct_left` class): we just have

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

```

465         elseif SIG then
466             local addgl = (prev_char and
467                 not FB_guil_null[prev_char])
468                 or
469                 (not prev_char and
470                 prev_id ~= TEMP and
471                 not (prev_id == HLIST and
472                     prev_subtype == 3)
473             )

```

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

```

474         if is_glue and glue_wd <= 1 then
475             addgl = false
476         end
477         local t = FBsp.guill.gl.fr
478         nspace.char = FBsp.guill.ch.fr
479         if FRdialect then
480             t = FBsp.guill.gl.ac
481             nspace.char = FBsp.guill.ch.ac
482         end
483         local fbg glue = new_glue_scaled(fid, t)
484         if addgl and fbg glue then
485             if is_glue then
486                 head = remove_node(head,prev,true)
487             end
488             if (FRucsNBSP) then
489                 nspace.font = fid
490                 nspace.attr = attr
491                 insert_node_before(head,item,copy_node(nspace))
492             else
493                 nobreak.attr = attr
494                 fbg glue.attr = attr
495                 insert_node_before(head,item,copy_node(nobreak))
496                 insert_node_before(head,item,copy_node(fbg glue))
497             end
498         end
499     end

```

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

```

500         elseif SIG then
501             local next = item.next
502             local next_id, next_subtype, next_char, nextnext, kern_wd
503             if next then

```

```

504         next_id = next.id
505         next_subtype = next.subtype
506         if next_id == GLYPH then
507             next_char = next.char

```

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

```

508         elseif next_id == KERN then
509             kern_wd = next.kern
510             if kern_wd == 0 then
511                 nextnext = next.next
512                 if nextnext then
513                     next = nextnext
514                     next_id = nextnext.id
515                     next_subtype = nextnext.subtype
516                     if next_id == GLYPH then
517                         next_char = nextnext.char
518                     end
519                 end
520             end
521         end
522     end
523     local is_glue = next_id == GLUE
524     if is_glue then
525         glue_wd = next.width
526     end
527     local addgl = (next_char and not FB_guil_null[next_char])
528                 or (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).

```

529         if is_glue and glue_wd == 0 then
530             addgl = false
531         end
532         local fid = item.font
533         local t = FBsp.guill.gl.fr
534         nbspace.char = FBsp.guill.ch.fr
535         if FRdialect then
536             t = FBsp.guill.gl.ac
537             nbspace.char = FBsp.guill.ch.ac
538         end
539         local fbglue = new_glue_scaled(fid, t)
540         if addgl and fbglue then
541             if is_glue then
542                 head = remove_node(head,next,true)
543             end
544             if (FRucsNBSP) then
545                 nbspace.font = fid
546                 nbspace.attr = attr
547                 insert_node_after(head, item, copy_node(nbspace))
548             else
549                 nobreak.attr = attr
550                 fbglue.attr = attr
551                 insert_node_after(head, item, copy_node(fbglue))

```

```

552             insert_node_after(head, item, copy_node(nobreak))
553         end
554     end
555 end
556     end
557 end
558 end
559 return head
560 end
561 return french_punctuation
562 </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.

```

563 <*french>
564 \ifFB@luatex@punct
565   \newcommand*\FB@luatex@punct@french{%
566     \babel@save\shorthandon
567     \babel@save\shorthandoff
568     \def\shorthandoff##1{%
569       \ifx\PackageWarning\@undefined
570         \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
571           LuaTeX, \ use \noexpand\NoAutoSpacing
572           *inside a group* instead.}%
573       \else
574         \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?}
575           is helpless with LuaTeX, \MessageBreak
576           use \protect\NoAutoSpacing \space *inside a group*
577           instead; \MessageBreak reported}%
578       \fi}%
579     \def\shorthandon##1{%
580   }
581   \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` to the kerning callback; "adding" anything actually disables the built-in kerning for Type1 fonts (which is now added to `french_punctuation`).

```

582 \def\activate@luatexpunct{%
583   \directlua{%
584     FR_fr = \the\l@french ; FR_ca = \the\l@acadian ;
585     local path = kpse.find_file("frenchb.lua", "lua")
586     if path then
587       local f = dofile(path)
588       luatexbase.add_to_callback("kerning",
589         f, "frenchb.french_punctuation")
590     else
591       texio.write_nl('')
592       texio.write_nl('*****')
593       texio.write_nl('Error: frenchb.lua not found.')

```

```

594         texio.write_nl('*****')
595         texio.write_nl('')
596     end
597 }%
598 }
599 \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.

```

600 \ifFB@xetex@punct
601   \ifLaTeXe
602     \PackageInfo{french.ldf}{No need for active punctuation
603                           characters\MessageBreak with this
604                           version of XeTeX!\MessageBreak reported}
605   \else
606     \fb@info{No need for active punctuation characters\
607             with this version of XeTeX!}
608   \fi

```

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

```

609 \newXeTeXintercharclass\FB@punctthick
610 \newXeTeXintercharclass\FB@punctthin
611 \newXeTeXintercharclass\FB@punctnul
612 \newXeTeXintercharclass\FB@guilo
613 \newXeTeXintercharclass\FB@guilf
614 \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.

```

615 \def\FBsavevariable@loop#1#2{\begingroup
616   \toks@\expandafter{\originalTeX #1}%
617   \edef\x{\endgroup
618     \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
619   \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
)]	{	}	,	-	.	"	%	'	'	'

```
620 \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F,%
621 "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.

```
622 \newcommand*\FB@xetex@punct@french{%
623 \babel@savevariable{\XeTeXinterchartokenstate}%
624 \babel@save{\shorthandon}%
625 \babel@save{\shorthandoff}%
626 \bbl@for\FB@char\FB@charlist
627 {\FBsavevariable@loop{\XeTeXcharclass}{\FB@char}}%
628 \def\shorthandoff##1{%
629 \ifx\PackageWarning\@undefined
630 \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
631 XeTeX,\ use \noexpand\NoAutoSpacing
632 *inside a group* instead.}%
633 \else
634 \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?}
635 is helpless with XeTeX,\MessageBreak
636 use \protect\NoAutoSpacing\space *inside a group*
637 instead;\MessageBreak reported}%
638 \fi}%
639 \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).

```
640 \XeTeXinterchartokenstate=1
641 \XeTeXcharclass \: = \FB@punctthick
642 \XeTeXinterchartoks \z@ \FB@punctthick = {%
643 \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
644 \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
645 \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: `\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.

```
646 \XeTeXinterchartoks \FB@nonchar \FB@punctthick = {%
647 \ifFB@spacing
648 \ifhmode
649 \ifdim\lastskip>1sp
```

```

650         \unskip\penalty\@M\FBcolonspace
651         \else
652         \FDP@colonspace
653         \fi
654     \fi
655 \fi}%
656 \bbl@for\FB@char
657     {\`;,\`!,\`?}%
658     {\XeTeXcharclass\FB@char=\FB@punctthin}%
659 \XeTeXinterchartoks \z@ \FB@punctthin = {%
660     \ifFB@spacing\ifhmode\FDP@thinspace\fi\fi}%
661 \XeTeXinterchartoks \FB@guilf \FB@punctthin = {%
662     \ifFB@spacing\FDP@thinspace\fi}%
663 \XeTeXinterchartoks \FB@nonchar \FB@punctthin = {%
664     \ifFB@spacing
665     \ifhmode
666     \ifdim\lastskip>lsp
667     \unskip\penalty\@M\FBthinspace
668     \else
669     \FDP@thinspace
670     \fi
671     \fi
672     \fi}%
673 \XeTeXinterchartoks \FB@guilo \z@ = {%
674     \ifFB@spacing\FB@guillspace\fi}%
675 \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
676     \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
677 \XeTeXinterchartoks \z@ \FB@guilf = {%
678     \ifFB@spacing\FB@guillspace\fi}%
679 \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
680     \ifFB@spacing\FB@guillspace\fi}%
681 \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
682     \ifFB@spacing\unskip\FB@guillspace\fi}%

```

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

```

683 \bbl@for\FB@char
684     {\`[,`\(`,"A0,"202F}%
685     {\XeTeXcharclass\FB@char=\FB@punctnul}%

```

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

```

686 \bbl@for\FB@char
687     {\`{,\`.,,\`.,,\`-,,\`),,\`},,\`%, "22,"27,"60,"2019}%
688     {\XeTeXcharclass\FB@char=\z@}%
689 }
690 \addto\extrasfrench{\FB@xetex@punct@french}

```

End of specific code for punctuation with modern XeTeX engines.

```
691 \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. Before doing so, we have to save some definitions involving :

```
692 \newif\ifFB@koma
693 \ifLaTeXe
694 \ifclassloaded{scrartcl}{\FB@komatruetrue}{}
695 \ifclassloaded{scrbook}{\FB@komatruetrue}{}
696 \ifclassloaded{scrreprt}{\FB@komatruetrue}{}
697 \ifFB@koma\def\FB@std@capsep{: \ } \fi
698 \ifclassloaded{beamer}{\def\FB@std@capsep{: \ }}{}
699 \ifclassloaded{memoir}{\def\FB@std@capsep{: }}{}
700 \fi

701 \ifFB@active@punct
702 \initiate@active@char{:}%
703 \initiate@active@char{;}%
704 \initiate@active@char{!}%
705 \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.

```
706 \declare@shorthand{french}{;}{%
707   \ifFB@spacing
708     \ifhmode
709       \ifdim\lastskip>lsp
710         \unskip\penalty\@M\FBthinspace
711       \else
712         \FDP@thinspace
713       \fi
714     \fi
715   \fi
```

Now we can insert a ; character.

```
716 \string;}
```

The next three definitions are very similar.

```
717 \declare@shorthand{french}{!}%
718 \ifFB@spacing
719   \ifhmode
720     \ifdim\lastskip>lsp
721       \unskip\penalty\@M\FBthinspace
722     \else
723       \FDP@thinspace
724     \fi
725   \fi
726 \string!}
727 \declare@shorthand{french}{?}%
```

```

729 \iffB@spacing
730 \ifhmode
731 \ifdim\lastskip>1sp
732 \unskip\penalty\@M\FBthinspace
733 \else
734 \FDP@thinspace
735 \fi
736 \fi
737 \fi
738 \string?}
739 \declare@shorthand{french}{:}{}%
740 \iffB@spacing
741 \ifhmode
742 \ifdim\lastskip>1sp
743 \unskip\penalty\@M\FBcolonspace
744 \else
745 \FDP@colonspace
746 \fi
747 \fi
748 \fi
749 \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.

```

750 \declare@shorthand{system}{:}{\string;}
751 \declare@shorthand{system}{!}{\string!}
752 \declare@shorthand{system}{?}{\string?}
753 \declare@shorthand{system}{;}{\string;}

```

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

```

754 \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.

```

755 \bbl@activate{:}\bbl@activate{;}%
756 \bbl@activate{!}\bbl@activate{?}%
757 }
758 \addto\noextrasfrench{%
759 \bbl@deactivate{:}\bbl@deactivate{;}%
760 \bbl@deactivate{!}\bbl@deactivate{?}%
761 }
762 \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.

```

763 \newif\iffBAutoSpacePunctuation \iffBAutoSpacePunctuationtrue

```

`\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 L^AT_EX. Set the default now for Plain (done later for LaTeX).

```

764 \def\autospace@beforeFDP{%
765   \iffB@luatex@punct\FB@addDPspace=1 \fi
766   \def\FDP@thinspace{\penalty\@M\FBthinspace}%
767   \def\FDP@colonspace{\penalty\@M\FBcolonspace}}
768 \def\noautospace@beforeFDP{%
769   \iffB@luatex@punct\FB@addDPspace=0 \fi
770   \let\FDP@thinspace\@empty
771   \let\FDP@colonspace\@empty}
772 \ifLaTeXe
773   \def\AutoSpaceBeforeFDP{\autospace@beforeFDP
774     \FBAutoSpacePunctuationtrue}
775   \def\NoAutoSpaceBeforeFDP{\noautospace@beforeFDP
776     \FBAutoSpacePunctuationfalse}
777   \AtEndOfPackage{\AutoSpaceBeforeFDP}
778 \else
779   \let\AutoSpaceBeforeFDP\autospace@beforeFDP
780   \let\NoAutoSpaceBeforeFDP\noautospace@beforeFDP
781   \AutoSpaceBeforeFDP
782 \fi

```

`\rmfamilyFB` In LaTeX2e `\ttfamily` (and hence `\texttt`) will be redefined ‘AtBeginDocument’ as `\sffamilyFB` `\ttfamilyFB` so that no space is added before the four ; : ! ? characters, even if `\ttfamilyFB` `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.

```

783 \ifLaTeXe
784   \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
785   \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on \rmfamilyORI}
786   \DeclareRobustCommand\sffamilyFB{\FB@spacing@on \sffamilyORI}
787 \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.

```

788 \DeclareRobustCommand*\NoAutoSpacing}{%
789   \FB@spacing@off
790   \ifFB@active@punct\shorthandoff{;:!?}\fi
791 }

```

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.

```

792 \ifLaTeXe
793 \else
794   \ifBUnicode
795     \def\guillemotleft{{\char"00AB}}
796     \def\guillemotright{{\char"00BB}}
797     \def\textquotedblleft{{\char"201C}}
798     \def\textquotedblright{{\char"201D}}
799   \else
800     \def\guillemotleft{\leavevmode\raise0.25ex
801                       \hbox{$\scriptscriptstyle\ll$}}
802     \def\guillemotright{\raise0.25ex
803                       \hbox{$\scriptscriptstyle\gg$}}
804     \def\textquotedblleft{``}
805     \def\textquotedblright{''}
806   \fi
807   \let\xspace\relax
808 \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` and `\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. `\FB@og` and `\FB@fg` are now designed to work in bookmarks.

```

809 \providecommand\textorpdfstring[2]{#1}
810 \newcommand*\FB@og{\textorpdfstring{\@FB@og}{\guillemotleft\space}}
811 \newcommand*\FB@fg{\textorpdfstring{\@FB@fg}{\space\guillemotright}}

```

The internal definitions `\@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.

```

812 \newcommand*\FB@guillspace{\penalty\@M\FBguillspace}
813 \newcommand*\FBgspchar{\char"A0\relax}
814 \newif\ifFBucsNBSP
815 \ifFB@luatex@punct
816   \DeclareRobustCommand*\@FB@og{\leavevmode
817     \bgroup\FB@spacing=0 \guillemotleft\egroup

```

```

818         \ifFBucsNBSP\FBbspchar\else\FB@guillspace\fi}
819 \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@unskip\fi
820         \ifFBucsNBSP\FBbspchar\else\FB@guillspace\fi
821         \bgroup\FB@spacing=0 \guillemotright\egroup}
822 \fi

```

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

```

823 \ifFB@xetex@punct
824 \DeclareRobustCommand*\@FB@og{\leavevmode
825         \bgroup\FB@spacingfalse\guillemotleft\egroup
826         \FB@guillspace}
827 \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@unskip\fi
828         \FB@guillspace
829         \bgroup\FB@spacingfalse\guillemotright\egroup}
830 \fi
831 \ifFB@active@punct
832 \DeclareRobustCommand*\@FB@og{\leavevmode
833         \guillemotleft
834         \FB@guillspace}
835 \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@unskip\fi
836         \FB@guillspace
837         \guillemotright}
838 \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.

```

839 \newcommand*\og{\@empty}
840 \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).

```

841 \ifLaTeXe
842 \def\bbf@frenchguillemets{%
843     \renewcommand*\og{\FB@og}%
844     \renewcommand*\fg{\FB@fg\xspace}}
845 \renewcommand*\og{\textquotedblleft}
846 \renewcommand*\fg{\ifdim\lastskip>\z@unskip\fi
847     \textquotedblright\xspace}
848 \else
849 \def\bbf@frenchguillemets{\let\og\FB@og
850     \let\fg\FB@fg}
851 \def\og{\textquotedblleft}
852 \def\fg{\ifdim\lastskip>\z@unskip\fi\textquotedblright}
853 \fi

854 \addto\extrasfrench{\babel@save\og \babel@save\fg
855     \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...

```
856 \newcommand*{\ogi}{\FB@og}
857 \newcommand*{\fgi}{\FB@fg}
858 \newcommand*{@ogi}{\ifmmode\hbox{\ogi}\else\ogi\fi}
859 \newcommand*{@fgi}{\ifmmode\hbox{\fgi}\else\fgi\fi}
860 \newcommand*{\ogii}{\textquotedblleft}
861 \newcommand*{\fgii}{\textquotedblright}
862 \newcommand*{@ogii}{\ifmmode\hbox{\ogii}\else\ogii\fi}
863 \newcommand*{@fgii}{\ifmmode\hbox{\fgii}\else\fgii\fi}
```

and the needed technical stuff to handle options:

```
864 \newcount\FBguill@level
865 \newtoks\FBold@everypar
```

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

```
866 \def\FB@addquote@everypar{%
867   \let\FBnew@everypar\everypar
868   \FBold@everypar=\expandafter{\the\everypar}%
869   \FBnew@everypar={\the\FBold@everypar\FBeverypar@quote}%
870   \let\everypar\FBold@everypar
871   \let\FB@addquote@everypar\relax
872 }
873 \newif\ifFBcloseguill \FBcloseguilltrue
874 \newif\ifFBInnerGuillSingle
875 \def\FBguillopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
876 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
877 \let\FBguillnone\empty
878 \let\FBeveryparguill\FBguillopen
879 \let\FBverylineguill\FBguillnone
880 \let\FBeverypar@quote\relax
881 \let\FBveryline@quote\empty
```

The main command `\frquote` accepts (in LaTeX2e only) a starred version which suppresses the closing quote; it is meant to be used for inner quotations which end together with the outer one, then only one closing guillemet (the outer one) should be printed. `\frquote` (without star) is now designed to work in bookmarks too.

```
882 \ifLaTeXe
883   \DeclareRobustCommand\frquote{%
884     \texorpdfstring{@ifstar{\FBcloseguillfalse\fr@quote}%
885                       {\FBcloseguilltrue \fr@quote}}%
886     {\bm@fr@quote}%
887   }
888   \newcommand{\bm@fr@quote}[1]{%
889     \guillemotleft\space #1\space\guillemotright}
890 \else
891   \newcommand\frquote[1]{\fr@quote{#1}}
892 \fi
```

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

```
893 \newcommand{\fr@quote}[1]{%
894   \leavevmode
895   \advance\FBguill@level by \@ne
```

```

896 \ifcase\FBguill@level
897 \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:

```

898 \ifx\FBEveryparguill\FBguillnone
899 \else
900 \def\FBeverypar@quote{\FBEveryparguill\FB@guillspace}%
901 \FB@addquote@everypar
902 \fi
903 \@ogi #1\@fgi
904 \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.

```

905 \ifx\FBEverylineguill\FBguilllopen
906 \def\FBEveryline@quote{\FB@addGUIILspace=0 \guillemotleft
907 \FB@guillspace}%
908 \localleftbox{\FBEveryline@quote}%
909 \let\FBeverypar@quote\relax
910 \@ogi #1\ifFBcloseguill\@fgi\fi
911 \else
912 \ifx\FBEverylineguill\FBguillclose
913 \def\FBEveryline@quote{\FB@addGUIILspace=0 \guillemotright
914 \FB@guillspace}%
915 \localleftbox{\FBEveryline@quote}%
916 \let\FBeverypar@quote\relax
917 \@ogi #1\ifFBcloseguill\@fgi\fi
918 \else

```

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

```

919 \let\FBeverypar@quote\relax
920 \ifFBInnerGuillSingle
921 \def\ogii{\leavevmode
922 \guilsinglleft\FB@guillspace}%
923 \def\fgii{\ifdim\lastskip>\z@\unskip\fi
924 \FB@guillspace\guilsinglright}%
925 \ifx\FBEveryparguill\FBguilllopen
926 \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
927 \fi
928 \ifx\FBEveryparguill\FBguillclose
929 \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
930 \fi
931 \fi
932 \@ogii #1\ifFBcloseguill \@fgii \fi
933 \fi
934 \fi
935 \else

```

Warn if `\FBguill@level > 2`:

```

936 \ifx\PackageWarning\@undefined
937 \fb@warning{\noexpand\frquote\space handles up to
938 two levels.\\ Quotation not printed.}%
939 \else

```

```

940     \PackageWarning{french.ldf}{%
941       \protect\frquote\space handles up to two levels.
942       \MessageBreak Quotation not printed. Reported}
943   \fi
944 \fi

```

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

```

945 \global\advance\FBguill@level by \m@ne
946 \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
947 \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
948   \global\let\FBeveryl@line@quote\empty
949   \ifx\FBeveryl@line@guill\FBguillnone\else\localleftbox{}\fi
950 \fi
951 }

```

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

```

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

```

2.4 Date in French

`\frenchtoday` The following code creates a macro `\datefrench` which in turn defines command `\frenchdate` (`\today` is defined as `\frenchtoday` in French). The corresponding `\datefrench` 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. Coding `\StartBabelCommands*{french,acadian}` would *only* define `\date\CurrentOption`, leaving the second language undefined in Babel's sens.

```

953 \def\BabelLanguages{french,acadian}
954 \StartBabelCommands*\BabelLanguages}{date}
955   [unicode, fontenc=TU EU1 EU2, charset=utf8]
956   \SetString\monthiiname{février}
957   \SetString\monthviiiname{août}
958   \SetString\monthxiiname{décembre}
959 \StartBabelCommands*\BabelLanguages}{date}
960   \SetStringLoop{month#1name}{%
961     janvier,f\'evrier,mars,avril,mai,juin,juillet,%
962     ao^ut,septembre,octobre,novembre,d\'ecembre}
963   \SetString\today{\FB@date{\year}{\month}{\day}}
964 \EndBabelCommands

```

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

```

965 \newcommand*\FB@date}[3]{%
966   {\number#3}\ifnum1=#3\ier}\fi\FBdatespace
967   \csname month\romannumeral#2name\endcsname
968   \ifx#1\@empty\else\FBdatespace\number#1\fi}
969 \newcommand*\FBdatebox}{\hbox}

```



```

970 \newcommand*\FBdatespace{\space}
971 \newcommand*\frenchdate{\FBdatebox\FB@date}
972 \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 scalefont 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).

```

973 \newif\ifFB@poorman
974 \newdimen\FB@Mht
975 \ifLaTeXe
976   \AtEndOfPackage{\RequirePackage{scalefont}}

```

\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{}

```

977 \newcommand*\FBsupR{-0.12}
978 \newcommand*\FBsupS{0.65}
979 \newcommand*\FB@lc[1]{\MakeLowercase{#1}}
980 \DeclareRobustCommand*\FB@up@fake[1]{%
981   \settoheight{\FB@Mht}{M}%
982   \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
983   \addtolength{\FB@Mht}{-\FBsupS ex}%
984   \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
985 }

```

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.

```

986 \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
987 \def\FB@suffix{#4}}
988 \def\FB@x{x}
989 \def\FB@j{j}
990 \DeclareRobustCommand*\FB@up}[1]{%
991 \bgroup \FB@poormantrue
992 \expandafter\FB@split\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.

```

993 \edef\reserved@a{\lowercase{%
994 \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
995 \reserved@a
996 {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
997 \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
998 \ifFB@poorman \FB@up@fake{#1}%
999 \else \FB@up@real{#1}%
1000 \fi}%
1001 {\FB@up@fake{#1}}%
1002 \egroup}

```

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

```

1003 \newcommand*\FB@up@real}[1]{\bgroup
1004 \fontfamily{\FB@firstthree -sup}\selectfont \FB@lc{#1}\egroup}

```

`\fup` is defined as `\FB@up` unless `\realsuperscript` is defined by `realscripts.sty`. `\fup` just prints its argument in bookmarks.

```

1005 \DeclareRobustCommand*\fup}[1]{%
1006 \texorpdfstring{\ifx\realsuperscript\@undefined
1007 \FB@up{#1}%
1008 \else
1009 \bgroup\let\fakesuperscript\FB@up@fake
1010 \realsuperscript{\FB@lc{#1}}\egroup
1011 \fi
1012 }{#1}%
1013 }

```

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

```

1014 \providecommand*\up}{\fup}

```

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

```

1015 \else
1016 \providecommand*\up}[1]{\leavevmode\raiselex\hbox{\sevenrm #1}}
1017 \fi

```

```

\ieme Some handy macros for those who don’t know how to abbreviate ordinals:
\ier 1018 \def\ieme{\up{e}\xspace}
\iere 1019 \def\iemes{\up{es}\xspace}
\iemes
\iers
\ieres

```

```

1020 \def\ier{\up{er}\xspace}
1021 \def\iers{\up{ers}\xspace}
1022 \def\iere{\up{re}\xspace}
1023 \def\ieres{\up{res}\xspace}

```

\FBmedkern
\FBthickkern

```

1024 \newcommand*\FBmedkern{\kern+.2em}
1025 \newcommand*\FBthickkern{\kern+.3em}

```

\primo Some support macros relying on `\up` for numbering,

\fprimo) 1026 `\newcommand*\FrenchEnumerate[1]{%`
\nos 1027 `#1\textorpdfstring{\up{o}\FBthickkern}{\textdegree\xspace}}`
\Nos 1028 `\newcommand*\FrenchPopularEnumerate[1]{%`
\No 1029 `#1\textorpdfstring{\up{o})\FBthickkern}{\textdegree\xspace}}`
\no Typing `\primo` should result in ‘^o’ (except in bookmarks where `\textdegree` is used instead of o-superior),

```

1030 \def\primo{\FrenchEnumerate1}
1031 \def\secundo{\FrenchEnumerate2}
1032 \def\tertio{\FrenchEnumerate3}
1033 \def\quarto{\FrenchEnumerate4}

```

while typing `\fprimo` gives ‘^o’ (except in bookmarks where `\textdegree` is used instead),.

```

1034 \def\fprimo{\FrenchPopularEnumerate1}
1035 \def\fsecundo{\FrenchPopularEnumerate2}
1036 \def\ftertio{\FrenchPopularEnumerate3}
1037 \def\fquarto{\FrenchPopularEnumerate4}

```

Let’s provide four macros for the common abbreviations of “Numéro”. In bookmarks ^o is used instead of o-superior.

```

1038 \DeclareRobustCommand*\No{%
1039   \textorpdfstring{N\up{o}\FBmedkern}{N\textdegree\xspace}}
1040 \DeclareRobustCommand*\no{%
1041   \textorpdfstring{n\up{o}\FBmedkern}{n\textdegree\xspace}}
1042 \DeclareRobustCommand*\Nos{%
1043   \textorpdfstring{N\up{os}\FBmedkern}{N\textdegree\xspace}}
1044 \DeclareRobustCommand*\nos{%
1045   \textorpdfstring{n\up{os}\FBmedkern}{n\textdegree\xspace}}

```

\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}`.

```

1046 \ifLaTeXe
1047   \DeclareRobustCommand*\bsc[1]{%
1048     \textorpdfstring{\leavevmode\begingroup\kern0pt \scshape #1\endgroup}%
1049     {\textsc{#1}}}%
1050   }
1051 \else

```

```

1052 \newcommand*{\bsc}[1]{\leavevmode\begingroup\kern0pt #1\endgroup}
1053 \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.

```

1054 \iffBunicode
1055 \providecommand*\textbackslash{{\char"005C}}
1056 \providecommand*\textasciicircum{{\char"005E}}
1057 \providecommand*\textasciitilde{{\char"007E}}
1058 \newcommand*\FB@degree{°}
1059 \else
1060 \ifLaTeXe
1061 \newcommand*\FB@degree{\r{}}
1062 \fi
1063 \fi
1064 \DeclareRobustCommand*\boi{\textbackslash}
1065 \DeclareRobustCommand*\circonflexe{\textasciicircum}
1066 \DeclareRobustCommand*\tild{\textasciitilde}
1067 \DeclareRobustCommand*\degree{%
1068 \texorpdfstring{\FB@degree}{\textdegree}}
1069 \newcommand*\at{@}

```

\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). `\degrees` works in math-mode (angles). 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.

```

1070 \DeclareRobustCommand*\degrees{\degree}
1071 \ifLaTeXe
1072 \AtBeginDocument{%
1073 \ifpackageloaded{fontspec}{\%
1074 \ifdefined\DeclareEncodingSubset
1075 \DeclareRobustCommand*\degrees{%
1076 \texorpdfstring{\hbox{\UseTextSymbol{TS1}{\textdegree}}}{%
1077 \textdegree}}%
1078 \else
1079 \def\Warning@degree@TSone{\FBWarning
1080 {Degrees would look better in TS1-encoding:%
1081 \MessageBreak add \protect
1082 \usepackage{textcomp} to the preamble.%
1083 \MessageBreak Degrees used}}
1084 \DeclareRobustCommand*\degrees{%
1085 \texorpdfstring{\hbox to 0.3em{\hss\degree\hss}}{
1086 \Warning@degree@TSone
1087 \global\let\Warning@degree@TSone\relax}%
1088 \textdegree}}%
1089 \fi

```

```

1090 }%
1091 }
1092 \fi

```

2.6 Formatting numbers

`\StandardMathComma` As mentioned in the \TeX book p. 134, the comma is of type `\mathpunct` in math mode: 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 (or Acadian) *only* (no space added); `\StandardMathComma` switches back to the standard behaviour of the comma.

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

```

1093 \newif\ifFB@icomma
1094 \newcount\mc@charclass
1095 \newcount\mc@charfam
1096 \newcount\mc@charslot
1097 \newcount\std@mcc
1098 \newcount\dec@mcc
1099 \ifFBLuaTeX
1100 \mc@charclass=\Umathcharclass`,
1101 \newcommand*\dec@math@comma}{%
1102 \mc@charfam=\Umathcharfam`,
1103 \mc@charslot=\Umathcharslot`,
1104 \Umathcode`,= 0 \mc@charfam \mc@charslot
1105 }
1106 \newcommand*\std@math@comma}{%
1107 \mc@charfam=\Umathcharfam`,
1108 \mc@charslot=\Umathcharslot`,
1109 \Umathcode`,= \mc@charclass \mc@charfam \mc@charslot
1110 }
1111 \else
1112 \std@mcc=\mathcode`,,
1113 \dec@mcc=\std@mcc
1114 \@tempcnta=\std@mcc
1115 \divide\@tempcnta by "1000
1116 \multiply\@tempcnta by "1000
1117 \advance\dec@mcc by -\@tempcnta
1118 \newcommand*\dec@math@comma}{\mathcode`,=\dec@mcc}
1119 \newcommand*\std@math@comma}{\mathcode`,=\std@mcc}
1120 \fi

```

`\DecimalMathComma` operates in French or Acadian independently.

```

1121 \newcommand*\DecimalMathComma}{%
1122 \ifFB@icomma
1123 \PackageWarning{french.ldf}{%
1124 icomma package loaded, \protect\DecimalMathComma\MessageBreak
1125 does nothing. Reported}%
1126 \else
1127 \ifFBfrench
1128 \dec@math@comma
1129 \expandafter\addto\csname extras\language\endcsname
1130 {\dec@math@comma}%

```

```

1131 \fi
1132 \fi
1133 }
1134 \newcommand*{\StandardMathComma}{%
1135 \ifFB@icomma
1136 \PackageWarning{french.ldf}{%
1137 icomma package loaded, \protect\StandardMathComma\MessageBreak
1138 does nothing. Reported}%
1139 \else
1140 \std@math@comma
1141 \expandafter\addto\csname extras\language\endcsname
1142 {\std@math@comma}%
1143 \fi
1144 }
1145 \ifLaTeXe
1146 \AtBeginDocument{\@ifpackageloaded{icomma}%
1147 {\FB@icommatrue}%
1148 {\addto\noextrasfrench{\std@math@comma}%
1149 \ifdefined\noextrasacadian
1150 \addto\noextrasacadian{\std@math@comma}%
1151 \fi
1152 }%
1153 }
1154 \else
1155 \addto\noextrasfrench{\std@math@comma}
1156 \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:

```

1157 \newcommand*{\nombre}[1]{\fb@warning{*** \noexpand\nombre
1158 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.

```

1159 \ifFB@luatex@punct
1160 \activate@luatexpunct
1161 \fi
1162 \let\FBstop@here\relax
1163 \def\FBclean@on@exit{%
1164 \let\ifLaTeXe\undefined
1165 \let\LaTeXetrue\undefined
1166 \let\LaTeXefalse\undefined
1167 \let\FB@llc\loadlocalcfg
1168 \let\loadlocalcfg@gobble}
1169 \ifx\magnification\@undefined
1170 \else
1171 \def\FBstop@here{%
1172 \FBclean@on@exit

```

```

1173 \ldf@finish\CurrentOption
1174 \let\loadlocalcfg\FB@llc
1175 \endinput}
1176 \fi
1177 \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.

```

1178 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1179 \newcommand*{\Warning@nombre}[1]{%
1180   \ifdefined\numprint
1181     \numprint{#1}%
1182   \else
1183     \PackageWarning{french.ldf}{%
1184       \protect\nombre\space now relies on package numprint.sty,%
1185       \MessageBreak add \protect
1186       \usepackage[autolanguage]{numprint},\MessageBreak
1187       see file numprint.pdf for more options.\MessageBreak
1188       \protect\nombre\space called}%
1189     \global\let\Warning@nombre\relax
1190     {#1}%
1191   \fi
1192 }

1193 \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.

`\figurename` and `\tablename` are printed in small caps in French, unless either `SmallCapsFigTabCaptions` is set to `false` or a class or package loaded before `babel-french` defines `\FBfigtabshape` as `\relax`.

```
1194 \providecommand*{\FBfigtabshape}{\scshape}
```

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

```

1195 \StartBabelCommands*{\BabelLanguages}{captions}
1196   [unicode, fontenc=TU EU1 EU2, charset=utf8]
1197   \SetString{\refname}{Références}
1198   \SetString{\abstractname}{Résumé}
1199   \SetString{\prefacename}{Préface}
1200   \SetString{\contentsname}{Table des matières}
1201   \SetString{\ccname}{Copie à }
1202   \SetString{\proofname}{Démonstration}
1203   \SetString{\partfirst}{Première}
1204   \SetString{\partsecond}{Deuxième}
1205   \SetStringLoop{ordinal#1}{%
1206     \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%

```

```

1207     Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1208     Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1209     Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1210 \StartBabelCommands*\BabelLanguages}{captions}
1211   \SetString{\refname}{R\ 'ef\ 'erences}
1212   \SetString{\abstractname}{R\ 'esum\ 'e}
1213   \SetString{\bibname}{Bibliographie}
1214   \SetString{\prefacename}{Pr\ 'eface}
1215   \SetString{\chaptername}{Chapitre}
1216   \SetString{\appendixname}{Annexe}
1217   \SetString{\contentsname}{Table des mati\`eres}
1218   \SetString{\listfigurename}{Table des figures}
1219   \SetString{\listtablename}{Liste des tableaux}
1220   \SetString{\indexname}{Index}
1221   \SetString{\figurename}{Figure}
1222   \SetString{\tablename}{Table}
1223   \SetString{\pagename}{page}
1224   \SetString{\seename}{voir}
1225   \SetString{\alsoname}{voir aussi}
1226   \SetString{\enclname}{P.~J. }
1227   \SetString{\ccname}{Copie \ `a }
1228   \SetString{\headtoname}{ }
1229   \SetString{\proofname}{D\ 'emonstration}
1230   \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`.

```

1231   \SetString{\partfirst}{Premi\`ere}
1232   \SetString{\partsecond}{Deuxi\`eme}
1233   \SetString{\partnameord}{partie}
1234   \SetStringLoop{ordinal#1}{%
1235     \partfirst,\partsecond,Troisi\`eme,Quatri\`eme, Cinq\`ui\`eme,%
1236     Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
1237     Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1238     Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1239     Vingti\`eme}
1240 \AfterBabelCommands{%
1241   \DeclareRobustCommand*\FB@emptypart{\def\thepart{\unskip}}%
1242   \DeclareRobustCommand*\FB@partname{%
1243     \ifFBPartNameFull
1244       \csname ordinal\romannumeral\value{part}\endcsname\space
1245       \partnameord\FB@emptypart
1246     \else
1247       Partie%
1248     \fi}%
1249   }
1250   \SetString{\partname}{\FB@partname}
1251 \EndBabelCommands

```

`\figurename` and `\tablename` no longer include font commands; to print them in small caps in French (the default), we now customise `\fnum@figure` and `\fnum@table` when available (not in `beamer.cls` f.i.).

```

1252 \AtBeginDocument{%

```



```

1253 \ifx\FBfigtabshape\relax
1254 \else
1255   \ifdefined\fnun@figure
1256     \let\fnun@figureORI\fnun@figure
1257     \renewcommand{\fnun@figure}{\ifFBfrench\FBfigtabshape\fi
1258                                   \fnun@figureORI}}%
1259   \fi
1260   \ifdefined\fnun@table
1261     \let\fnun@tableORI\fnun@table
1262     \renewcommand{\fnun@table}{\ifFBfrench\FBfigtabshape\fi
1263                                   \fnun@tableORI}}%
1264   \fi
1265 \fi
1266 }

```

2.8 Figure and table captions

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

```
1267 \newcommand{\FBWarning}[1]{\PackageWarning{french.ldb}{#1}}
```

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

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

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

```

1268 \bgroup
1269 \catcode`:=12 \catcode`>=12 \relax
1270 \long\gdef\STD@makecaption#1#2{%
1271   \vskip\abovcaptionskip
1272   \sbox\@tempboxa{#1: #2}%
1273   \ifdim \wd\@tempboxa >\hsize
1274     #1: #2\par
1275   \else
1276     \global \@minipagefalse
1277     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1278   \fi
1279   \vskip\belowcaptionskip}
1280 \egroup

```

No warning is issued for SMF and AMS 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.

```
1281 \newif\if@FBwarning@capsep
1282 \ifFB@active@punct\@FBwarning@capseptrue\fi
1283 \newcommand*\CaptionSeparator{\space\textendash\space}
1284 \def\FBCaption@Separator{: }
1285 \long\def\FB@makecaption#1#2{%
1286   \vskip\abovecaptionskip
1287   \box\@tempboxa{#1\FBCaption@Separator #2}%
1288   \ifdim \wd\@tempboxa >\hsize
1289     #1\FBCaption@Separator #2\par
1290   \else
1291     \global \@minipagefalse
1292     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1293   \fi
1294   \vskip\belowcaptionskip}
```

Disable the standard warning with AMS and SMF classes.

```
1295 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1296 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1297 \@ifclassloaded{amstex}{\@FBwarning@capsepfalse}{}
1298 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1299 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1300 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1301 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}

```

Disable the standard warning for some classes that do not use ‘:’ as caption separator.

```
1302 \@ifclassloaded{IEEEconf}{\@FBwarning@capsepfalse}{}
1303 \@ifclassloaded{IEEEtran}{\@FBwarning@capsepfalse}{}
1304 \@ifclassloaded{revtex4-2}{\@FBwarning@capsepfalse}{}
1305 \@ifclassloaded{svjour3}{\@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)

```
1306 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1307 \ifFB@koma \@FBwarning@capsepfalse \fi

```

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).

```
1308 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1309 \ifdefined\@makecaption\else\@FBwarning@capsepfalse\fi

```

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’).

```

1310 \AtBeginDocument{%
1311   \ifx\@makecaption\STD@makecaption
1312     \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 locally force \autospace@beforeFDP in case
AutoSpacePunctuation=false.

1313   \ifFBOldFigTabCaptions
1314   \else
1315     \def\FBCaption@Separator{\autospace@beforeFDP : }%
1316     \ifFBCustomiseFigTabCaptions
1317       \ifFB@mainlanguage@FR
1318         \def\FBCaption@Separator{\CaptionSeparator}%
1319       \fi
1320     \fi
1321   \fi
1322   \@FBwarning@capsepfalse
1323 \fi

No Warning if caption.sty or caption-light.sty has been loaded.

1324   \@ifpackageloaded{caption}{\@FBwarning@capsepfalse}{}%
1325   \@ifpackageloaded{caption-light}{\@FBwarning@capsepfalse}{}%

Final warning if relevant:

1326   \if@FBwarning@capsep
1327     \FBWarning
1328       {Figures' and tables' captions might look like\MessageBreak
1329       `Figure 1:' in French instead of `Figure 1 :'.\MessageBreak
1330       If this happens, to fix this issue\MessageBreak
1331       switch to LuaLaTeX or XeLaTeX or\MessageBreak
1332       try to add \protect\usepackage{caption} or\MessageBreak
1333       ... leave it as it is; reported}%
1334   \fi
1335   \let\FB@makecaption\relax
1336   \let\STD@makecaption\relax
1337 }

```

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.

```

1338 \ifFBunicode
1339   \let\FBtextellipsis\textellipsis

```

```

1340 \else
1341   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1342   \DeclareTextCommandDefault{\FBtextellipsis}{%
1343     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}
1344 \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 ‘At-BeginDocument’, if `amsmath` has not been loaded. `\Mdots@` doesn’t change when switching from/to French, while `\Tdots@` is redefined as `\FBtextellipsis` in French.

```

1345 \newcommand*\Tdots@{\@xp\textellipsis}
1346 \newcommand*\Mdots@{\@xp\mdots@}
1347 \AtBeginDocument{\DeclareRobustCommand*\dots{\relax
1348   \csname\ifmmode M\else T\fi dots@\endcsname}%
1349   \ifdefined\@xp\else\let\@xp\relax\fi
1350   \ifdefined\mdots@\else\let\Mdots@\mathellipsis\fi
1351 }
1352 \def\bbbl@frenchdots{\babel@save\Tdots@ \let\Tdots@\FBtextellipsis}
1353 \addto\extrasfrench{\bbbl@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).

```

1354 \ifFB@active@punct
1355   \@ifpackageloaded{listings}
1356     {\AtBeginDocument{%
1357       \FBWarning{Please load the "listings" package\MessageBreak
1358         AFTER babel/french; reported}}%
1359     }{}
1360 \fi

```

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

```

1361 \newif\if@FBwarning@natbib
1362 \ifFB@active@punct
1363   \@ifpackageloaded{natbib}{\@FBwarning@natbibtrue}
1364 \fi
1365 \AtBeginDocument{%
1366   \if@FBwarning@natbib
1367     \@ifpackageloaded{natbib}{\@FBwarning@natbibfalse}%
1368   \fi
1369   \if@FBwarning@natbib
1370     \FBWarning{Please load the "natbib" package\MessageBreak
1371       BEFORE babel/french; reported}%
1372   \fi
1373 }

```

Package `beamerarticle` should be loaded before `babel-french` to avoid list’s conflicts, see p. 54.

```

1374 \newif\if@FBwarning@beamerarticle
1375 \@ifpackageloaded{beamerarticle}{\@FBwarning@beamerarticlettrue}

```

```

1376 \AtBeginDocument{%
1377   \if@FBwarning@beamerarticle
1378     \@ifpackageloaded{beamerarticle}{}%
1379                                     {\@FBwarning@beamerarticlefalse}%
1380   \fi
1381   \if@FBwarning@beamerarticle
1382     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1383               BEFORE babel/french; reported}%
1384   \fi
1385 }

```

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 ‘AtEnd-OfPackage’ 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 `\extrsfrench{}` *must* be processed by `\frenchsetup{}`: when French is the main language, `\extrsfrench{}` is executed by Babel when it switches the main language and this occurs *before* reading the stuff postponed by babel - french ‘AtBeginDocument’. Re-executing `\extrsfrench{}` 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.

```

1386 \newcommand*{\frenchsetup}[1]{%
1387   \setkeys{FB}{#1}%
1388 }%
1389 \@onlypreamble\frenchsetup

```

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

```

1390 \let\frenchbsetup\frenchsetup
1391 \@onlypreamble\frenchbsetup

```

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

```

1392 \newif\ifFBShowOptions
1393 \newif\ifFBStandardLayout           \FBStandardLayouttrue
1394 \newif\ifFBGlobalLayoutFrench       \FBGlobalLayoutFrenchtrue
1395 \newif\ifFBReduceListSpacing
1396 \newif\ifFBStandardListSpacing      \FBStandardListSpacingtrue
1397 \newif\ifFBListOldLayout
1398 \newif\ifFBListItemsAsPar
1399 \newif\ifFBCompactItemize
1400 \newif\ifFBStandardItemizeEnv        \FBStandardItemizeEnvtrue
1401 \newif\ifFBStandardEnumerateEnv      \FBStandardEnumerateEnvtrue
1402 \newif\ifFBStandardItemLabels        \FBStandardItemLabelstrue
1403 \newif\ifFBStandardLists             \FBStandardListstrue
1404 \newif\ifFBIndentFirst

```

```

1405 \newif\ifFBFrenchFootnotes
1406 \newif\ifFBAutoSpaceFootnotes
1407 \newif\ifFBOriginalTypewriter
1408 \newif\ifFBThinColonSpace
1409 \newif\ifFBThinSpaceInFrenchNumbers
1410 \newif\ifFBFrenchSuperscripts      \FBFrenchSuperscriptstrue
1411 \newif\ifFBLowercaseSuperscripts  \FBLowercaseSuperscriptstrue
1412 \newif\ifFBPartNameFull           \FBPartNameFulltrue
1413 \newif\ifFBCustomiseFigTabCaptions
1414 \newif\ifFBOldFigTabCaptions
1415 \newif\ifFBSmallCapsFigTabCaptions \FBSmallCapsFigTabCaptionstrue
1416 \newif\ifFBSuppressWarning
1417 \newif\ifFBINGuillSpace

```

The defaults values of these flags have been choosen 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`.

```

1418 \ifFB@koma
1419   \ifdefined\partformat
1420     \def\FB@partformat@fix{%
1421       \ifFBPartNameFull
1422         \babel@save\partformat
1423         \renewcommand*{\partformat}{\partname}%
1424       \fi
1425     \addto\extrasfrench{\FB@partformat@fix}%
1426   \fi
1427 \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.

```

1428 \def\FB@french{french}
1429 \def\FB@acadian{acadian}
1430 \newif\ifFB@mainlanguage@FR
1431 \AtEndOfPackage{%
1432   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1433   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1434   \fi
1435   \ifFB@mainlanguage@FR
1436     \FBGlobalLayoutFrenchtrue
1437     \@ifclassloaded{beamer}%
1438     {\PackageInfo{french.ldf}{%
1439       No list customisation for the beamer class,%
1440       \MessageBreak reported}}%

```

```

1441     {\@ifpackageloaded{beamerarticle}%
1442      {\FBStandardItemLabelsfalse
1443       \FBStandardListSpacingfalse
1444       \PackageInfo{french.ldf}{%
1445        Minimal list customisation for the beamerarticle%
1446        \MessageBreak package; reported}}%

```

Otherwise customise lists “à la française”:

```

1447     {\FBStandardListSpacingfalse
1448      \FBStandardItemizeEnvfalse
1449      \FBStandardEnumerateEnvfalse
1450      \FBStandardItemLabelsfalse}%
1451   }
1452   \FBIndentFirsttrue
1453   \FBFrenchFootnotesttrue
1454   \FBAutoSpaceFootnotesttrue
1455   \FBCustomiseFigTabCaptionstrue
1456   \else
1457   \FBGlobalLayoutFrenchfalse
1458   \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.

```

1459   \RequirePackage{keyval}%
1460   \define@key{FB}{ShowOptions}[true]%
1461     {\csname FBShowOptions#1\endcsname}%

```

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

```

1462   \define@key{FB}{StandardLayout}[true]%
1463     {\ifFB@mainlanguage@FR
1464      \csname FBStandardLayout#1\endcsname
1465      \else
1466        \PackageWarning{french.ldf}%
1467          {Option `StandardLayout' skipped:\MessageBreak
1468           French is *not* babel's last option.\MessageBreak
1469           Reported}%
1470      \fi
1471     \ifFBStandardLayout
1472       \FBStandardListSpacingtrue
1473       \FBStandardItemizeEnvtrue
1474       \FBStandardItemLabelstrue
1475       \FBStandardEnumerateEnvtrue
1476       \FBIndentFirstfalse
1477       \FBFrenchFootnotesfalse
1478       \FBAutoSpaceFootnotesfalse
1479       \FBGlobalLayoutFrenchfalse
1480     \else
1481       \FBStandardListSpacingfalse
1482       \FBStandardItemizeEnvfalse
1483       \FBStandardItemLabelsfalse
1484       \FBStandardEnumerateEnvfalse
1485       \FBIndentFirsttrue
1486       \FBFrenchFootnotesttrue

```

```

1487         \FBAutoSpaceFootnotesttrue
1488         \fi}%
1489 \define@key{FB}{GlobalLayoutFrench}[true]%
1490     {\ifFB@mainlanguage@FR
1491      \csname FBGlobalLayoutFrench#1\endcsname
1492     \else
1493      \PackageWarning{french.ldb}%
1494      {Option `GlobalLayoutFrench' skipped:\MessageBreak
1495       French is *not* babel's last option.\MessageBreak
1496       Reported}%
1497     \fi}%

```

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

```

1498 \define@key{FB}{ReduceListSpacing}[true]%
1499     {\csname FBReduceListSpacing#1\endcsname
1500     \ifFBReduceListSpacing \FBStandardListSpacingfalse
1501     \else \FBStandardListSpacingtrue\fi
1502     }%
1503 \define@key{FB}{StandardListSpacing}[true]%
1504     {\csname FBStandardListSpacing#1\endcsname}%
1505 \define@key{FB}{ListOldLayout}[true]%
1506     {\csname FBListOldLayout#1\endcsname
1507     \ifFBListOldLayout
1508      \FBStandardEnumerateEnvtrue
1509      \renewcommand*\FrenchLabelItem{\textendash}%
1510     \fi}%
1511 \define@key{FB}{CompactItemize}[true]%
1512     {\csname FBCompactItemize#1\endcsname
1513     \ifFBCompactItemize
1514      \FBStandardItemizeEnvfalse
1515      \FBStandardEnumerateEnvfalse
1516     \else
1517      \FBStandardItemizeEnvtrue
1518      \FBStandardEnumerateEnvtrue
1519     \fi}%
1520 \define@key{FB}{StandardItemizeEnv}[true]%
1521     {\csname FBStandardItemizeEnv#1\endcsname}%
1522 \define@key{FB}{StandardEnumerateEnv}[true]%
1523     {\csname FBStandardEnumerateEnv#1\endcsname}%
1524 \define@key{FB}{StandardItemLabels}[true]%
1525     {\csname FBStandardItemLabels#1\endcsname}%
1526 \define@key{FB}{ItemLabels}%
1527     {\renewcommand*\FrenchLabelItem{#1}}%
1528 \define@key{FB}{ItemLabeli}%
1529     {\renewcommand*\Frlabelitemi{#1}}%
1530 \define@key{FB}{ItemLabelii}%
1531     {\renewcommand*\Frlabelitemii{#1}}%
1532 \define@key{FB}{ItemLabeliii}%
1533     {\renewcommand*\Frlabelitemiii{#1}}%
1534 \define@key{FB}{ItemLabeliv}%
1535     {\renewcommand*\Frlabelitemiv{#1}}%
1536 \define@key{FB}{StandardLists}[true]%

```



```

1537     {\csname FBStandardLists#1\endcsname
1538     \ifFBStandardLists
1539         \FBStandardListSpacingtrue
1540         \FBStandardItemizeEnvtrue
1541         \FBStandardEnumerateEnvtrue
1542         \FBStandardItemLabelstrue
1543     \else
1544         \FBStandardListSpacingfalse
1545         \FBStandardItemizeEnvfalse
1546         \FBStandardEnumerateEnvfalse
1547         \FBStandardItemLabelsfalse
1548     \fi}%
1549 \define@key{FB}{ListItemsAsPar}[true]%
1550     {\csname FBListItemsAsPar#1\endcsname}
1551 \define@key{FB}{IndentFirst}[true]%
1552     {\csname FBIndentFirst#1\endcsname}%
1553 \define@key{FB}{FrenchFootnotes}[true]%
1554     {\csname FBFrenchFootnotes#1\endcsname}%
1555 \define@key{FB}{AutoSpaceFootnotes}[true]%
1556     {\csname FBAutoSpaceFootnotes#1\endcsname}%
1557 \define@key{FB}{AutoSpacePunctuation}[true]%
1558     {\csname FBAutoSpacePunctuation#1\endcsname}%
1559 \define@key{FB}{OriginalTypewriter}[true]%
1560     {\csname FBOriginalTypewriter#1\endcsname}%
1561 \define@key{FB}{ThinColonSpace}[true]%
1562     {\csname FBThinColonSpace#1\endcsname
1563     \ifFBThinColonSpace
1564         \renewcommand*\FBcolonspace{\FBthinspace}%
1565     \fi}%
1566 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1567     {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1568 \define@key{FB}{FrenchSuperscripts}[true]%
1569     {\csname FBFrenchSuperscripts#1\endcsname}
1570 \define@key{FB}{LowercaseSuperscripts}[true]%
1571     {\csname FBLowercaseSuperscripts#1\endcsname}
1572 \define@key{FB}{PartNameFull}[true]%
1573     {\csname FBPartNameFull#1\endcsname}%
1574 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1575     {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1576 \define@key{FB}{OldFigTabCaptions}[true]%
1577     {\csname FBOldFigTabCaptions#1\endcsname
1578     \ifFBOldFigTabCaptions
1579         \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1580         \def\FBCaption@Separator{\CaptionSeparator}}%
1581         \addto\extrasfrench{\FB@capsep@fix}%
1582         \ifdefined\extrasacadian
1583             \addto\extrasacadian{\FB@capsep@fix}%
1584         \fi
1585     \fi}%
1586 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1587     {\csname FBSmallCapsFigTabCaptions#1\endcsname
1588     \ifFBSmallCapsFigTabCaptions
1589         \else \let\FBfigtabshape\relax \fi}%

```

```

1590 \define@key{FB}{SuppressWarning}[true]%
1591     {\csname FBSuppressWarning#1\endcsname
1592     \ifFBSuppressWarning
1593     \renewcommand{\FBWarning}[1]{}%
1594     \fi}%

```

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

```

1595 \define@key{FB}{INGuillSpace}[true]%
1596     {\csname FBINGuillSpace#1\endcsname
1597     \ifFBINGuillSpace
1598     \renewcommand*{\FBguillspace}{\space}%
1599     \fi}%
1600 \define@key{FB}{InnerGuillSingle}[true]%
1601     {\csname FBInnerGuillSingle#1\endcsname}%
1602 \define@key{FB}{EveryParGuill}[open]%
1603     {\expandafter\let\expandafter
1604     \FBeveryparguill\csname FBguill#1\endcsname
1605     \ifx\FBeveryparguill\FBguillopen
1606     \else\ifx\FBeveryparguill\FBguillclose
1607     \else\ifx\FBeveryparguill\FBguillnone
1608     \else
1609     \let\FBeveryparguill\FBguillopen
1610     \FBWarning{Wrong value for `EveryParGuill':
1611     try `open',\MessageBreak
1612     `close' or `none'. Reported}%
1613     \fi
1614     \fi
1615     \fi}%
1616 \define@key{FB}{EveryLineGuill}[open]%
1617     {\ifFB@luatex@punct
1618     \expandafter\let\expandafter
1619     \FBeverylineguill\csname FBguill#1\endcsname
1620     \ifx\FBeverylineguill\FBguillopen
1621     \else\ifx\FBeverylineguill\FBguillclose
1622     \else\ifx\FBeverylineguill\FBguillnone
1623     \else
1624     \let\FBeverylineguill\FBguillnone
1625     \FBWarning{Wrong value for `EveryLineGuill':
1626     try `open',\MessageBreak
1627     `close' or `none'. Reported}%
1628     \fi
1629     \fi
1630     \fi
1631     \else
1632     \FBWarning{Option `EveryLineGuill' skipped:%
1633     \MessageBreak this option is for
1634     LuaTeX *only*.\MessageBreak Reported}%
1635     \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.

```

1636 \define@key{FB}{UnicodeNoBreakSpaces}[true]%

```

```

1637     {\ifFB@luatex@punct
1638         \csname FBucsNBS#1\endcsname
1639         \ifFBucsNBS \FB@ucsNBS=1 \fi
1640     \else
1641         \FBWarning{Option `UnicodeNoBreakSpaces' skipped:%
1642             \MessageBreak this option is for
1643             LuaTeX *only*.\MessageBreak Reported}%
1644     \fi
1645 }%

```

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@addGUIspace` 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.

```

1646 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
1647 \define@key{FB}{og}%
1648     {\ifFBunicode

```

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

```

1649     \ifFB@luatex@punct
1650         \FB@addGUIspace=1 \relax
1651     \fi

```

then with XeTeX it is a bit more tricky:

```

1652     \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).

```

1653         \XeTeXcharclass"13 = \FB@guilo
1654         \XeTeXcharclass"AB = \FB@guilo
1655         \XeTeXcharclass"A0 = \FB@guilnul
1656         \XeTeXcharclass"202F = \FB@guilnul
1657     \fi

```

Issue a warning with older Unicode engines requiring active characters.

```

1658     \ifFB@active@punct
1659         \FBWarning{Option og« not supported with this version
1660             of\MessageBreak LuaTeX/XeTeX; reported}%
1661     \fi
1662 \else

```

This is for conventional TeX engines:

```

1663     \newcommand*{\FB@@og}{%

```

```

1664         \ifFBfrench
1665             \ifFB@spacing\FB@og\ignorespaces
1666             \else\guillemotleft
1667             \fi
1668         \else\guillemotleft\fi}%
1669     \AtBeginDocument{%
1670         \ifdefined\uc@dclc
Package inputenc with utf8x (ucs) encoding loaded, use \uc@dclc:
1671             \uc@dclc{171}{default}{\FB@@og}%
1672         \else
if encoding is not utf8x, check if the argument of og is a single-byte character:
1673             \FB@parse#1\endparse
1674             \ifx\FB@second\@empty
This means 8-bit character encoding. Package MULEenc (from CJK) defines \mule@def
to map characters to control sequences.
1675             \ifdefined\mule@def
1676                 \mule@def{11}{\FB@@og}%
1677             \else
1678                 \ifdefined\DeclareInputText
1679                     \@tempcnta`#1\relax
1680                     \DeclareInputText{\the\@tempcnta}{\FB@@og}%
1681                 \else
Package inputenc not loaded, no way...
1682                 \FBWarning{Option `og' requires package
1683                     inputenc;\MessageBreak reported}%
1684                 \fi
1685             \fi
1686         \else
This means multi-byte character encoding, we assume UTF-8
1687             \DeclareUnicodeCharacter{00AB}{\FB@@og}%
1688             \fi
1689         \fi}%
1690     \fi
1691 }%
Same code for the closing quote.
1692 \define@key{FB}{fg}%
1693     {\ifFBunicode
1694         \ifFB@luatex@punct
1695             \FB@addGUIlSpace=1 \relax
1696         \fi
1697         \ifFB@xetex@punct
1698             \XeTeXcharclass"14 = \FB@guilf
1699             \XeTeXcharclass"BB = \FB@guilf
1700             \XeTeXcharclass"A0 = \FB@guilnul
1701             \XeTeXcharclass"202F = \FB@guilnul
1702         \fi
1703         \ifFB@active@punct
1704             \FBWarning{Option fg=> not supported with this version
1705                 of\MessageBreak LuaTeX/XeTeX; reported}%
1706         \fi

```

```

1707         \else
1708             \newcommand*{\FB@fg}{%
1709                 \ifFBfrench
1710                     \ifFB@spacing\FB@fg
1711                     \else\guillemotright
1712                     \fi
1713                 \else\guillemotright\fi}%
1714         \AtBeginDocument{%
1715             \ifdefined\uc@dclc
1716                 \uc@dclc{187}{default}{\FB@fg}%
1717             \else
1718                 \FB@parse#1\endparse
1719                 \ifx\FB@second\@empty
1720                     \ifdefined\mule@def
1721                         \mule@def{27}{\FB@fg}%
1722                     \else
1723                         \ifdefined\DeclareInputText
1724                             \@tempcnta`#1\relax
1725                             \DeclareInputText{\the\@tempcnta}{\FB@fg}%
1726                         \else
1727                             \FBWarning{Option `fg' requires package
1728                                 inputenc;\MessageBreak reported}%
1729                             \fi
1730                         \fi
1731                     \else
1732                         \DeclareUnicodeCharacter{00BB}{\FB@fg}%
1733                         \fi
1734                     \fi}%
1735         \fi
1736     }%
1737 }

```

\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 \captionfrench *have already been processed* by Babel at \begin{document} *before* \FBprocess@options.

```
1738 \newcommand*{\FBprocess@options}{%
```

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

```

1739     \@ifpackageloaded{enumitem}{%
1740         \ifFBStandardItemizeEnv
1741         \else
1742             \FBStandardItemizeEnvtrue
1743             \PackageInfo{french.ldf}%
1744                 {Setting StandardItemizeEnv=true for\MessageBreak
1745                 compatibility with enumitem package,\MessageBreak
1746                 reported}%
1747         \fi
1748         \ifFBStandardEnumerateEnv
1749         \else

```

```

1750     \FBStandardEnumerateEnvtrue
1751     \PackageInfo{french.ldf}%
1752     {Setting StandardEnumerateEnv=true for\MessageBreak
1753      compatibility with enumitem package,\MessageBreak
1754      reported}%
1755     \fi}{}%
1756 \@ifpackageloaded{paralist}{%
1757   \ifFBStandardItemizeEnv
1758   \else
1759     \FBStandardItemizeEnvtrue
1760     \PackageInfo{french.ldf}%
1761     {Setting StandardItemizeEnv=true for\MessageBreak
1762      compatibility with paralist package,\MessageBreak
1763      reported}%
1764   \fi
1765   \ifFBStandardEnumerateEnv
1766   \else
1767     \FBStandardEnumerateEnvtrue
1768     \PackageInfo{french.ldf}%
1769     {Setting StandardEnumerateEnv=true for\MessageBreak
1770      compatibility with paralist package,\MessageBreak
1771      reported}%
1772   \fi}{}%
1773 \@ifpackageloaded{enumerate}{%
1774   \ifFBStandardEnumerateEnv
1775   \else
1776     \FBStandardEnumerateEnvtrue
1777     \PackageInfo{french.ldf}%
1778     {Setting StandardEnumerateEnv=true for\MessageBreak
1779      compatibility with enumerate package,\MessageBreak
1780      reported}%
1781   \fi}{}%

```

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

```

1782 \def\FB@ufl{\update@frenchlists}
1783 \ifFB@mainlanguage@FR
1784   \update@frenchlists
1785 \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.

```

1786 \ifBFAutoSpacePunctuation
1787   \autospace@beforeFDP
1788 \else
1789   \noautospace@beforeFDP
1790 \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.

```
1791 \ifFBOriginalTypewriter
1792 \else
1793 \let\ttfamilyORI\ttfamily
1794 \let\rmfamilyORI\rmfamily
1795 \let\sffamilyORI\sffamily
1796 \let\ttfamily\ttfamilyFB
1797 \let\rmfamily\rmfamilyFB
1798 \let\sffamily\sffamilyFB
1799 \fi
```

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

```
1800 \@ifpackageloaded{numprint}%
1801   {\ifnprt@autolanguage
1802     \providecommand*\npstylefrench{}%
1803     \ifFBThinSpaceInFrenchNumbers
1804       \renewcommand*\FBthousandsep{\,}%
1805     \fi
1806     \g@addto@macro\npstylefrench{\npthousandsep\FBthousandsep}%
1807   \fi
1808   }%}
```

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.

```
1809 \ifFBFrenchSuperscripts
1810   \DeclareRobustCommand*\up*{%
1811     \texorpdfstring{\@ifstar{\FB@up@fake}{\fup}}{}%
1812   }
1813 \else
1814   \DeclareRobustCommand*\up*{%
1815     \texorpdfstring{\@ifstar{\FB@up@fake}{\textsuperscript}}{}%
1816   }
1817 \fi
```

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

```
1818 \ifBBLowercaseSuperscripts
1819 \else
1820 \renewcommand*\FB@lc[1]{##1}%
1821 \fi
```

This is for koma-script, memoir and beamer classes. If the caption delimiter has been user customised, leave it unchanged. Otherwise, force the colon to behave properly in French (add locally `\autospace@beforeFDP` in case of `AutoSpacePunctuation=false`) and change the caption delimiter to `\CaptionSeparator` if `CustomiseFigTabCaptions` has been set to `true`.

```
1822 \ifFB@koma
1823   \ifx\captionformat\FB@std@capsep
1824     \ifBBCustomiseFigTabCaptions
1825       \renewcommand*\captionformat{\CaptionSeparator}%
1826     \else
```

```

1827         \renewcommand*{\captionformat}{\autospace@beforeFDP : \ }}%
1828     \fi
1829 \fi
1830 \fi
1831 \@ifclassloaded{memoir}%
1832     {\ifx\@contdelim\FB@std@capsep
1833         \ifFBCustomiseFigTabCaptions
1834             \captiondelim{\CaptionSeparator}%
1835         \else
1836             \captiondelim{\autospace@beforeFDP : }}%
1837     \fi
1838 \fi}%
1839 \@ifclassloaded{beamer}%
1840     {\protected@edef\FB@capsep%
1841         \csname beamer@@@tpl@caption label separator\endcsname%
1842     \ifx\FB@capsep\FB@std@capsep
1843         \ifFBCustomiseFigTabCaptions
1844             \defbeamertemplate{caption label separator}{FBcustom}{%
1845                 \CaptionSeparator}%
1846             \setbeamertemplate{caption label separator}[FBcustom]%
1847         \else
1848             \defbeamertemplate{caption label separator}{FBcolon}{%
1849                 \autospace@beforeFDP : }}%
1850             \setbeamertemplate{caption label separator}[FBcolon]%
1851     \fi
1852 \fi}%

```

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

```

1853 \ifFBShowOptions
1854     \GenericWarning{* }{%
1855         *** List of possible options for babel-french ***\MessageBreak
1856         [Default values between brackets when french is loaded *LAST*]%
1857         \MessageBreak
1858         ShowOptions [false]\MessageBreak
1859         StandardLayout [false]\MessageBreak
1860         GlobalLayoutFrench [true]\MessageBreak
1861         PartNameFull [true]\MessageBreak
1862         IndentFirst [true]\MessageBreak
1863         ListItemsAsPar [false]\MessageBreak
1864         StandardListSpacing [false]\MessageBreak
1865         StandardItemizeEnv [false]\MessageBreak
1866         StandardEnumerateEnv [false]\MessageBreak
1867         StandardItemLabels [false]\MessageBreak
1868         ItemLabels=\textendash, \textbullet,
1869         \protect\ding{43},... [\textendash]\MessageBreak
1870         ItemLabeli=\textendash, \textbullet,
1871         \protect\ding{43},... [\textendash]\MessageBreak
1872         ItemLabelii=\textendash, \textbullet,
1873         \protect\ding{43},... [\textendash]\MessageBreak
1874         ItemLabeliii=\textendash, \textbullet,
1875         \protect\ding{43},... [\textendash]\MessageBreak
1876         ItemLabeliv=\textendash, \textbullet,
1877         \protect\ding{43},... [\textendash]\MessageBreak
1878         StandardLists [false]\MessageBreak

```



```

1879 ListOldLayout [false]\MessageBreak
1880 FrenchFootnotes [true]\MessageBreak
1881 AutoSpaceFootnotes [true]\MessageBreak
1882 AutoSpacePunctuation [true]\MessageBreak
1883 ThinColonSpace [false]\MessageBreak
1884 OriginalTypewriter [false]\MessageBreak
1885 UnicodeNoBreakSpaces [false]\MessageBreak
1886 og= <left quote character>, fg= <right quote character>%
1887 INGuillSpace [false]\MessageBreak
1888 EveryParGuill=open, close, none [open]\MessageBreak
1889 EveryLineGuill=open, close, none
1890 [open in LuaTeX, none otherwise]\MessageBreak
1891 InnerGuillSingle [false]\MessageBreak
1892 ThinSpaceInFrenchNumbers [false]\MessageBreak
1893 SmallCapsFigTabCaptions [true]\MessageBreak
1894 CustomiseFigTabCaptions [true]\MessageBreak
1895 OldFigTabCaptions [false]\MessageBreak
1896 FrenchSuperscripts [true]\MessageBreak
1897 LowercaseSuperscripts [true]\MessageBreak
1898 SuppressWarning [false]\MessageBreak
1899 \MessageBreak
1900 *****%
1901 \MessageBreak\protect\frenchsetup{ShowOptions}}
1902 \fi
1903 }

```

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.

```

1904 \AtBeginDocument{%
1905   \providecommand*\xspace{\relax}%

```

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

```

1906   \FBprocess@options

```

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

```

1907   \ifFBucsNBSP
1908     \renewcommand*\FBmedkern{\char"202F\relax}%
1909     \renewcommand*\FBthickkern{\char"A0\relax}%
1910     \ifFBThinSpaceInFrenchNumbers
1911       \renewcommand*\FBthousandsep{\char"202F\relax}%
1912     \else
1913       \renewcommand*\FBthousandsep{\char"A0\relax}%
1914     \fi
1915 \fi

```

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

```

1916 \begingroup
1917   \newcommand{\FBOTone}{OT1}%

```

```

1918 \ifx\encodingdefault\FB0Tone
1919 \FBWarning{OT1 encoding should not be used for French.%
1920 \MessageBreak
1921 Add \protect\usepackage[T1]{fontenc} to the
1922 preamble\MessageBreak of your document; reported}%
1923 \fi
1924 \endgroup
1925 }

```

2.12 French lists

`\listFB` Vertical spacing in lists should be shorter in French texts than the defaults provided by 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; `\listORI` 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 `Opt`, but will be noticeable when `\parskip` is *not* null.

```

1926 \let\listORI\list
1927 \let\endlistORI\endlist
1928 \def\FB@listVsettings{%
1929 \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1930 \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
1931 \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1932 \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`.

```

1933 \@tempdima=\parskip
1934 \addtolength{\topsep}{-\@tempdima}%
1935 \addtolength{\partopsep}{\@tempdima}%
1936 }
1937 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1938 \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 endash '–' is preferred for all levels. The item label to be used in French, stored in `\FrenchLabelItem`, 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 1939 \newcommand*{\FrenchLabelItem}{\textemdash}
  \Frlabelitemii 1940 \newcommand*{\Frlabelitemi}{\FrenchLabelItem}
\Frlabelitemiii 1941 \newcommand*{\Frlabelitemii}{\FrenchLabelItem}
  \Frlabelitemiv 1942 \newcommand*{\Frlabelitemiii}{\FrenchLabelItem}
  1943 \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 preamble.

`\labelindentFB` They will get reasonable defaults later when entering French (see `\setlabelitemsFB` and `\setlistindentFB`) unless they have been customised.

```

1944 \newdimen\listindentFB
1945 \setlength{\listindentFB}{-1pt}
1946 \newdimen\descindentFB
1947 \setlength{\descindentFB}{-1pt}
1948 \newdimen\labelindentFB
1949 \setlength{\labelindentFB}{-1pt}
1950 \newdimen\labelwidthFB
1951 \setlength{\labelwidthFB}{-1pt}

```

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

```

1952 \newdimen\leftmarginFB
1953 \def\FB@listHsettings{%
1954   \ifFBListItemsAsPar

```

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

```

1955   \itemindent=\labelindentFB
1956   \advance\itemindent by \labelwidthFB
1957   \advance\itemindent by \labelsep
1958   \leftmargini\z@
1959   \bbl@for\FB@dp {2, 3, 4, 5, 6}%
1960     {\csname leftmargin\romannumeral\FB@dp\endcsname =
1961       \labelindentFB}%
1962   \else

```

Default layout: labels hanging into the left margin.

```

1963   \leftmarginFB=\labelwidthFB
1964   \advance\leftmarginFB by \labelsep
1965   \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1966     {\csname leftmargin\romannumeral\FB@dp\endcsname =
1967       \leftmarginFB}%
1968   \advance\leftmargini by \listindentFB
1969   \fi
1970   \leftmargin=\csname leftmargin%
1971     \ifnum\@listdepth=\@ne i\else ii\fi\endcsname
1972 }

```

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

`\FB@itemizesettings` `\FB@itemizesettings` does two things: first suppress all vertical spaces including glue

unless option `StandardListSpacing` is set, then set horizontal indentations according to `\FB@listHsettings` unless option `ListOldLayout` is `true` (compatibility with lists up to v. 2.5k).

```

1973 \def\FB@itemizesettings{%
1974   \ifFBStandardListSpacing
1975   \else
1976     \setlength{\itemsep}{\z@}%
1977     \setlength{\parsep}{\z@}%
1978     \setlength{\topsep}{\z@}%
1979     \setlength{\partopsep}{\z@}%
1980     \@tempdima=\parskip
1981     \addtolength{\topsep}{-\@tempdima}%
1982     \addtolength{\partopsep}{\@tempdima}%
1983   \fi
1984   \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
1985   \ifBListOldLayout
1986     \setlength{\leftmargin}{\labelwidth}%
1987     \addtolength{\leftmargin}{\labelsep}%
1988     \addtolength{\leftmargin}{\parindent}%
1989   \else
1990     \FB@listHsettings
1991   \fi
1992 }

```

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

```

1993 \def\itemizeFB{%
1994   \ifnum \@itemdepth >\thr@@\toodeep\else
1995     \advance\@itemdepth by \@ne
1996     \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
1997     \expandafter
1998     \listORI
1999     \csname\@itemitem\endcsname
2000     \FB@itemizesettings
2001   \fi
2002 }
2003 \let\enditemizeFB\endlistORI

2004 \def\setlabelitemsFB{%
2005   \let\labelitemi\Frlabelitemi
2006   \let\labelitemii\Frlabelitemii
2007   \let\labelitemiii\Frlabelitemiii
2008   \let\labelitemiv\Frlabelitemiv
2009   \ifdim\labelwidthFB<\z@
2010     \settowidth{\labelwidthFB}{\FrenchLabelItem}%
2011   \fi
2012 }

2013 \def\setlistindentFB{%
2014   \ifdim\labelindentFB<\z@
2015     \ifdim\parindent=\z@
2016       \setlength{\labelindentFB}{1.5em}%
2017     \else
2018       \setlength{\labelindentFB}{\parindent}%
2019     \fi

```

```

2020 \fi
2021 \ifdim\listindentFB<\z@
2022   \ifdim\parindent=\z@
2023     \setlength{\listindentFB}{1.5em}%
2024   \else
2025     \setlength{\listindentFB}{\parindent}%
2026   \fi
2027 \fi
2028 \ifdim\descindentFB<\z@
2029   \ifBListItemsAsPar
2030     \setlength{\descindentFB}{\labelindentFB}%
2031   \else
2032     \setlength{\descindentFB}{\listindentFB}%
2033   \fi
2034 \fi
2035 }

```

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

```

2036 \def\enumerateFB{%
2037   \ifnum \@enumdepth >\thr@@\@toodeep\else
2038     \advance\@enumdepth by \@ne
2039     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2040     \expandafter
2041     \list
2042       \csname label\@enumctr\endcsname
2043       {\FB@listHsettings
2044         \usecounter\@enumctr\def\makelabel##1{\hss\llap{##1}}}%
2045   \fi
2046 }
2047 \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` (1rst 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.

```

2048 \def\descriptionFB{%
2049   \list{}{\FB@listHsettings
2050     \labelwidth=\z@
2051     \ifBListItemsAsPar
2052       \itemindent=\descindentFB
2053     \else
2054       \itemindent=-\leftmargin
2055     \ifnum\@listdepth=1
2056       \ifdim\descindentFB=\z@
2057         \ifdim\listindentFB>\z@

```

```

2058             \leftmargini=\listindentFB
2059             \leftmargin=\leftmargini
2060             \itemindent=-\leftmargin
2061             \fi
2062         \else
2063             \advance\itemindent by \descindentFB
2064         \fi
2065     \fi
2066 \fi
2067 \let\makelabel\descriptionlabel}%
2068 }
2069 \let\enddescriptionFB\endlistORI

```

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

```

2070 \def\update@frenchlists{%
2071   \setlistindentFB
2072   \ifFBStandardListSpacing
2073   \else \let\list\listFB \fi
2074   \ifFBStandardItemizeEnv
2075   \else \let\itemize\itemizeFB \fi
2076   \ifFBStandardItemLabels
2077   \else \setlabelitemsFB \fi
2078   \ifFBStandardEnumerateEnv
2079   \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2080 }

```

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. 62.

```

2081 \def\FB@ufl{\relax}
2082 \def\bbl@frenchlistlayout{%
2083   \ifFBGlobalLayoutFrench
2084   \else
2085     \babel@save\list           \babel@save\itemize
2086     \babel@save\enumerate     \babel@save\description
2087     \babel@save\labelitemi    \babel@save\labelitemii
2088     \babel@save\labelitemiii  \babel@save\labelitemiv
2089     \FB@ufl
2090   \fi
2091 }
2092 \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.

```

2093 \def\bbl@frenchindent{%
2094   \ifFBGlobalLayoutFrench
2095   \else
2096     \babel@save\@afterindentfalse
2097   \fi
2098   \ifFBIndentFirst
2099     \let\@afterindentfalse\@afterindenttrue
2100     \@afterindenttrue
2101   \fi}
2102 \def\bbl@nonfrenchindent{%
2103   \ifFBGlobalLayoutFrench
2104     \ifFBIndentFirst
2105       \@afterindenttrue
2106     \fi
2107   \fi}
2108 \addto\extrasfrench{\bbl@frenchindent}
2109 \addto\noextrasfrench{\bbl@nonfrenchindent}

```

2.14 Formatting footnotes

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

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

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

```

2110 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2111   {\PackageInfo{french.ldb}%
2112     {bigfoot package in use.\MessageBreak
2113       babel-french will NOT customise footnotes;%
2114       \MessageBreak reported}}%
2115   {\let\@footnotemarkORI\@footnotemark
2116     \def\@footnotemarkFB{\leavevmode\unskip\unkern
2117       \,\@footnotemarkORI}%
2118     \ifFBAutoSpaceFootnotes
2119       \let\@footnotemark\@footnotemarkFB
2120     \fi}%
2121   }

```

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

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

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

```
2122 \newdimen\parindentFFN
```

```
2123 \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.

```
2124 \newcommand*\dotFFN{.}
```

```
2125 \newcommand*\kernFFN{\kern .5em}
```

```
2126 \newdimen\FBfnindent
```

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

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

```
2127 \ifFB@koma
```

```
2128 \let\@makefntextORI\@makefntext
```

```
2129 \let\@@makefnmarkORI\@@makefnmark
```

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

```
2130 \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
```

```
2131 \thefootnotemark\dotFFN\kernFFN}
```

```
2132 \let\@makefntextFB\@makefntext
```

```
2133 \let\@@makefnmarkFB\@@makefnmark
```

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

```
2134 \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
```

```
2135 \textsuperscript{\thefootnotemark}}
```

```
2136 \let\@makefntextTH\@makefntext
```

```
2137 \let\@@makefnmarkTH\@@makefnmark
```

Restore the original definitions.

```
2138 \let\@makefntext\@makefntextORI
```

```
2139 \let\@@makefnmark\@@makefnmarkORI
```

```
2140 \fi
```

Definitions for the `memoir` class:

```
2141 \ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2142 {\newcommand{\@makefntextFB}[1]{%
```

```
2143 \def\footscript##1{##1\dotFFN\kernFFN}%
```

```
2144 \setlength{\footmarkwidth}{\FBfnindent}%
```

```
2145 \setlength{\footmarksep}{-\footmarkwidth}%
```

```
2146 \setlength{\footparindent}{\parindentFFN}%
```

```
2147 \makefootmark #1}%
```



```
2148 }{}
```

Definitions for the beamer class:

```
2149 \@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.

```
2150 {\def\@makefntextFB#1{%
2151   \def\insertfootnotetext{#1}%
2152   \def\insertfootnotemark{\insertfootnotemarkFB}%
2153   \usebeamertemplate***{footnote}}%
2154 \def\insertfootnotemarkFB{%
2155   \usebeamercolor[fg]{footnote mark}%
2156   \usebeamerfont*{footnote mark}%
2157   \llap{\@thefnmark}\dotFFN\kernFFN}%
2158 }
```

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)!

```
2159 \providecommand*\insertfootnotemarkFB{%
2160   \parindent=\parindentFFN
2161   \rule\z@\footnotesep
2162   \setbox\@tempboxa\hbox{\@thefnmark}%
2163   \ifdim\wd\@tempboxa>\z@
2164     \llap{\@thefnmark}\dotFFN\kernFFN
2165   \fi}
2166 \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.

```
2167 \providecommand\localleftbox[1]{}
2168 \AtBeginDocument{%
2169   \@ifpackageloaded{bigfoot}{}%
2170   {\ifdim\parindentFFN<10in
2171     \else
2172     \parindentFFN=\parindent
2173     \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2174     \fi
2175     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2176     \addtolength{\FBfnindent}{\parindentFFN}%
2177     \let\@makefntextORI\@makefntext
2178     \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).

```
2179     \let\@makefntextORI\@makefntext
```

```

2180     \long\def\@makefntext#1{%
2181         \lcalleftbox{}}%
2182         \let\FBeverypar@save\FBeverypar@quote
2183         \let\FBeverypar@quote\relax
2184         \ifFBFrenchFootnotes
2185             \ifx\footnote\thanks
2186                 \let\@@makefnmark\@@makefnmarkTH
2187                 \@makefntextTH{#1}
2188             \else
2189                 \let\@@makefnmark\@@makefnmarkFB
2190                 \@makefntextFB{#1}
2191             \fi
2192         \else
2193             \let\@@makefnmark\@@makefnmarkORI
2194             \@makefntextORI{#1}%
2195         \fi
2196         \let\FBeverypar@quote\FBeverypar@save
2197         \lcalleftbox{\FBeveryline@quote}}%
2198     \else

```

Special add-on for the memoir class: \@makefntext is redefined as \makethanksmark by \maketitle, hence these settings to match the other notes' vertical alignment.

```

2199     \@ifclassloaded{memoir}%
2200     {
2201         \setlength{\thanksmarkwidth}{\parindentFFN}%
2202         \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2203         \fi
2204     }%

```

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

```

2205     \@ifclassloaded{beamer}%
2206     {
2207         \ifdim\parindentFFN=1.5em\else
2208             \FBWarning{%
2209                 \protect\parindentFFN\space is ineffective%
2210                 \MessageBreak within the beamer class.%
2211                 \MessageBreak Reported}%
2212             \fi
2213         \fi
2214     }%

```

Definition of \@makefntext for all other classes:

```

2215     \long\def\@makefntext#1{%
2216         \lcalleftbox{}}%
2217         \let\FBeverypar@save\FBeverypar@quote
2218         \let\FBeverypar@quote\relax
2219         \ifFBFrenchFootnotes
2220             \@makefntextFB{#1}%
2221         \else
2222             \@makefntextORI{#1}%
2223         \fi
2224         \let\FBeverypar@quote\FBeverypar@save
2225         \lcalleftbox{\FBeveryline@quote}}%
2226     \fi

```

```
2227 }%
2228 }
```

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 `\@makefnctext`.

```
2229 \newcommand*{\AddThinSpaceBeforeFootnotes}{\FBAutoSpaceFootnotestruer}
2230 \newcommand*{\FrenchFootnotes}{\FBFrenchFootnotestruer}
2231 \newcommand*{\StandardFootnotes}{\FBFrenchFootnotesfalse}
```

2.15 Clean up and exit

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

```
2232 \FBclean@on@exit
2233 \ldf@finish\CurrentOption
2234 \let\loadlocalcfg\FB@llc
2235 </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`.

```
2236 <*acadian>
2237 \PackageInfo{acadian.ldf}%
2238 {\`acadian' dialect is currently\MessageBreak
2239  *absolutely identical* to the\MessageBreak
2240  `french' language; reported}
2241 </acadian>
2242 <*canadien>
2243 \PackageWarning{canadien.ldf}%
2244  {Option `canadien' for Babel is *deprecated*,\MessageBreak
2245   it might be removed sooner or later. Please\MessageBreak
2246   use `acadian' instead; reported}%
2247 \def\CurrentOption{acadian}

2248 \def\datecanadien{\dateacadian}
2249 \def\captionscanadien{\captionzacadian}
2250 \def\extrascanadien{\extrasacadian}
2251 \def\noextrascanadien{\noextrasacadian}
2252 </canadien>
2253 <*francais>
2254 \PackageWarning{francais.ldf}%
2255  {Option `francais' for Babel is *deprecated*,\MessageBreak
```

```

2256   it might be removed sooner or later.  Please\MessageBreak
2257   use `french' instead; reported}%
2258 \chardef\l@francais\l@french
2259 \def\CurrentOption{french}
2260 </francais>

```

Compatibility code for Babel pre-3.13: frenchb.lda could be loaded with options acadian, canadien, frenchb or francais.

```

2261 <*frenchb>
2262 \def\bbl@tempa{frenchb}
2263 \ifx\CurrentOption\bbl@tempa
2264   \chardef\l@frenchb\l@french
2265   \def\CurrentOption{french}
2266   \PackageWarning{babel-french}%
2267     {Option `frenchb' for Babel is *deprecated*,\MessageBreak
2268     it might be removed sooner or later.  Please\MessageBreak
2269     use `french' instead; reported}
2270 \else
2271   \def\bbl@tempa{francais}
2272   \ifx\CurrentOption\bbl@tempa
2273     \chardef\l@francais\l@french
2274     \def\CurrentOption{french}

```

Plain formats: no warning when francais.sty loads frenchb.lda (Babel pre-3.13).

```

2275   \ifx\magnification\@undefined
2276     \PackageWarning{babel-french}%
2277       {Option `francais' for Babel is *deprecated*,\MessageBreak
2278       it might be removed sooner or later.  Please\MessageBreak
2279       use `french' instead; reported}
2280   \fi
2281 \else
2282   \def\bbl@tempa{canadien}
2283   \ifx\CurrentOption\bbl@tempa
2284     \def\CurrentOption{acadian}
2285     \PackageWarning{babel-french}%
2286       {Option `canadien' for Babel is *deprecated*,\MessageBreak
2287       it might be removed sooner or later.  Please\MessageBreak
2288       use `acadian' instead; reported}
2289   \fi
2290 \fi
2291 \fi
2292 </frenchb>
2293 <acadian|canadien|frenchb|francais>\input french.lda\relax
2294 <acadian|canadien>\let\extrasacadian\extrasfrench
2295 <acadian|canadien>\let\noextrasacadian\noextrasfrench

```

3 Change History

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

v3.5l	General: No warning about <code>\@makecaption</code> for more classes.	50	character; this is mandatory for Lua-UL to underline and highlight them. Thanks to Marcel Krüger for providing the fix.	24
	<code>\captionsfrench</code> : Redefine <code>\fnum@figure</code> and <code>\fnum@table</code> separately.	47	Code reorganised for better efficiency.	24
v3.5k	General: <code>\degre</code> , <code>\degres</code> , <code>\circonflexe</code> , <code>\tild</code> , <code>\boi</code> and <code>\at</code> are now safe in bookmarks.	44	v3.5g	<code>frenchb.lua</code> : The kerning callback is a bit specific: adding code with <code>add_to_callback</code> actually deletes the legacy kerning as pointed out by Marcel Krüger on SE.
	<code>\pdfstringdefDisableCommands</code> dropped.	65		
	Reorganise warnings about ‘:’ in captions, according to enhancements in <code>caption.sty</code> v3.5a.	50	v3.5f	General: <code>\l@canadien</code> was defined too early in file ‘ <code>canadien.ldf</code> ’: <code>\l@acadian</code> might not be defined.
	<code>\bsc</code> : <code>\bsc</code> now relies on <code>\texorpdfstring</code> to be safe in bookmarks.	43		<code>\selectlanguage{canadien}</code> allowed again only for backward compatibility (deprecated).
	<code>\captionsfrench</code> : Small caps removed in <code>\figurename</code> and <code>\tablename</code> , use <code>\fnum@figure</code> and <code>\fnum@table</code> instead.	47		<code>\DecimalMathComma</code> : Fixed bug with the acadian language. Warning added if used with the <code>icomma</code> package.
	<code>\FB@fg</code> : <code>\FB@og</code> and <code>\FB@fg</code> now rely on <code>\texorpdfstring</code> to be safe in bookmarks.	36	v3.5e	<code>\frenchsetup</code> : <code>StandardLayout</code> and <code>GlobalLayoutFrench</code> options can no longer be toggled when French is not the main language.
	<code>\frquote</code> : <code>\frquote</code> now relies on <code>\texorpdfstring</code> to be safe in bookmarks.	38		<code>\frquote</code> : Make resettings global on exit.
	<code>\fup</code> : <code>\up</code> and <code>\fup</code> now rely on <code>\texorpdfstring</code> to be safe in bookmarks.	41		new command <code>\NoEveryParQuote</code>
	<code>\no</code> : <code>\no</code> , <code>\nos</code> , <code>\No</code> , <code>\Nos</code> , <code>\primo</code> , <code>\fprimo</code> , now rely on <code>\texorpdfstring</code> to be safe in bookmarks.	43		reset <code>\FB@addGUIlSpace</code> attribute inside <code>\localleftbox</code> (LuaTeX).
v3.5j	General: For memoir, koma-script and beamer captions, <code>\FB@std@sep</code> has to be defined before activating the colon.	33	v3.5d	<code>\frenchsetup</code> : <code>ReduceListSpacing</code> option deprecated: see <code>StandardListSpacing</code>
v3.5i	<code>\FBprocess@options</code> : For memoir, koma-script and beamer classes, leave caption delimiter unchanged if it has been user customised.	63	v3.5c	General: Remove grouping inside <code>\@makefnctext</code> , <code>\localleftbox</code> and <code>\FBeverypar@quote</code> saved and restored instead.
v3.5h	<code>frenchb.lua</code> : Added glues and penalties should inherit attributes from the related punctuation			<code>\frquote</code> : <code>\FBeverypar@quote</code> ’s value now properly reset across level changes.
				<code>\noextrasfrench</code> : <code>\lccode</code> of quote <code>0x27</code> changed from <code>0x2019</code> to <code>0x27</code> for Unicode engines.

v3.5b	General: Reset <code>\FBeverypar</code> locally inside <code>\@makefntext</code> . Needed by <code>\frquote</code>	73	Toks <code>\FBcolonsp</code> , <code>\FBthinsp</code> and <code>\FBguillsp</code> removed.	18	
	<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	<code>frenchb.lua</code> : Global ‘ <code>FBsp</code> ’ table added; local function ‘ <code>get_glue</code> ’ changed into global ‘ <code>FBget_glue</code> ’.	23	
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.	67	<code>\datefrench</code> : Specific code for Plain finally removed (babel bug reported).	40	
	<code>\descriptionFB</code> : <code>ListItemsAsPar</code> option taken into account for description lists.	69	<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	
	<code>\frenchsetup</code> : New option <code>ListItemsAsPar</code> for displaying lists’ items “as paragraphs”.	53	<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	
v3.4d	<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	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 ‘ <code>auto</code> ’ flag to false (avoids spurious spaces in URLs, MSDOS paths or 10:35).	25
v3.4c	<code>\ifFBXeTeX</code> : Reverting to former test, beware of <code>\XeTeXrevision</code> left as <code>\relax</code> by careless testing.	16	v3.3c	General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, <code>fontspec</code> is no longer required.	65
v3.4b	<code>\datefrench</code> : Do not redefine <code>\date</code> as <code>\frenchdate</code> in French.	40		New command <code>\FBthousandsep</code> to customise numprint.	47
v3.4a	General: <code>\LdfInit</code> checks <code>\FBClean@on@exit</code> instead of <code>\captionfrench</code> (undefined in PLain). Prevents loading <code>french.ldf</code> again with <code>acadian</code> option.	14		New configurable kerns <code>\FBmedkern</code> , and <code>\FBthickkern</code> suitable for HTML translation.	43
	<code>babel-french</code> now requires eTeX.	14		Reorganise warnings when the caption, subcaption or floatrow packages are loaded before <code>babel/french</code>	50
	Lua function <code>token.get_meaning</code> requires LuaTeX 1.0.	21		Reset <code>\localleftbox</code> locally inside <code>\@makefntext</code> . Needed by <code>\frquote</code> with LuaTeX.	73
	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>frenchb.lua</code> : Function ‘ <code>get_glue</code> ’ robustified. ‘ <code>french_punctuation</code> ’ can insert Unicode characters instead of glues.	22
	New attribute <code>\FB@dialect</code> for the French dialect <code>acadian</code>	20		<code>\frenchsetup</code> : New option ‘ <code>UnicodeNoBreakSpaces</code> ’ for html translators (LuaLaTeX only).	58
	New command <code>\FBsetspaces</code> to fine tune spacing independently in French and in French dialects.	18	v3.3b	General: Generate portmanteau files <code>acadian.ldf</code> , <code>canadien.ldf</code> , <code>frenchb.ldf</code> , and <code>français.ldf</code> and warn about deprecated	
	Shrink/stretch removed in <code>\FBthousandsep</code>	47			

options.	75	v3.2f	<code>\DecimalMathComma</code> : Fixed conflict with the <code>icomma</code> package.	45
New ‘if’ <code>\ifFBfrench</code> to replace <code>\iflanguage</code> test which is based on patterns.	16	v3.2e	General: Add missing redefinitions for <code>\leftmarginiv</code> , <code>\leftmarginvi</code> . Suggested by J.F. Burnol.	67
v3.3a			<code>\DecimalMathComma</code> : <code>\DecimalMathComma</code> didn’t work with LuaTeX. Fixed now.	45
General: Compatibility code for pre 2015/10/01 LaTeX release removed, see <code>ltnews23.tex</code>	20	v3.2d	<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.	69
Skip <code>\FBguillskip</code> for LuaTeX replaced by <code>\FBguillsp</code>	18	v3.2c	General: New LuaTeX attribute <code>\FB@spacing</code>	20
<code>\captionfrench</code> : Commands <code>\frenchpartfirst</code> , <code>\frenchpartsecond</code> and <code>\frenchpartnameord</code> added.	47		Newif <code>\ifFB@spacing</code> and new commands <code>\FB@spacingon</code> , <code>\FB@spacingoff</code> to control space tuning in French.	20
<code>\FBthinspace</code> : Skips <code>\FBcolonskip</code> and <code>\FBthinspace</code> replaced by <code>\FBcolonsp</code> and <code>\FBthinsp</code>	17		Switch <code>\ifFB@spacing</code> added to the four French shorthands.	33
<code>\frenchsetup</code> : <code>\frenchbsetup</code> is now an alias for <code>\frenchsetup</code>	53		<code>\FB@xetex@punct@french</code> : Switch <code>\ifFB@spacing</code> added to all <code>\XeTeXinterchartoks</code> commands.	31
Options <code>INGuillSpace</code> , <code>ThinColonSpace</code> no longer delayed <code>AtBeginDocument</code>	53		<code>\FBthinspace</code> : Change <code>.16667em</code> to <code>.5\fontdimen2\font</code> to get in XeTeX and pdfTeX the same spacing as in LuaTeX.	17
<code>\frquote</code> : <code>\FB@quotespace</code> (kern), changed into <code>\FB@guillspace</code> . ..	38		<code>\frenchsetup</code> : Add a warning about options <code>og/fg</code> for old XeTeX or LuaTeX engines requiring active characters.	59
v3.2h			<code>\NoAutoSpacing</code> : New definition based on <code>\FB@spacing@off</code> common to all engines.	35
<code>\@makefntextFB</code> : With <code>beamer.cls</code> , add <code>\llap</code> to <code>\@thefnmark</code> for notes numbered over 99.	73		<code>\ttfamilyFB</code> : New definitions of <code>\ttfamilyFB</code> and <code>co</code> , common to all engines, based on <code>\FB@spacing@off</code> and <code>\FB@spacing@on</code>	35
<code>\bbl@frenchlistlayout</code> : Execute <code>\update@frenchlists</code> only if <code>GlobalLayoutFrench</code> is false. Delete stuff for lists in <code>\noextrsfrench</code>	70	v3.2b	General: Load <code>lthuatex.tex</code> for plain LuaTeX to ensure <code>\newattribute</code> is defined.	20
<code>\frenchsetup</code> : Option <code>GlobalLayoutFrench</code> skipped when French is not the main language.	54		Warning added when the subcaption package is loaded before <code>babel/french</code>	50
v3.2g				
General: Changed Unicode definition of <code>\boi</code>	44			
<code>fontspec</code> defines TU encoding now and no longer loads <code>xunicode.sty</code> . Test changed.	65			
Issue a warning if <code>beamerarticle.sty</code> is loaded after <code>babel</code>	52			
<code>\frenchsetup</code> : Minimal list customisation when <code>beamerarticle.sty</code> is loaded.	54			
Warn when wrong values are provided to options <code>EveryParGuill</code> or <code>EveryLineGuill</code>	58			
<code>\frquote</code> : Default options of <code>\frquote</code> are no longer engine-dependent.	38			

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.	35	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.	38
\@makefntextFB: beamer.cls requires a specific definition of \@makefntextFB (pointed out by DB). The same is true for memoir and koma-script classes (done).	72	\PackageWarning is undefined in Plain, use \fb@warning instead.	38
\fg: \xspace moved from \FB@fg to \fg: \xspace messes up \frquote, pointed out by Sonia Labetoulle. As a side effect \xspace is now active in \fg in and outside French.	37	v3.1i	
v3.1m		General: \nombre command changed when numprint.sty is not loaded: only one warning, no error.	47
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	Remove restriction about loading numprint.sty after babel.	52
v3.1l		\frquote: \luatexlocalleftbox changed to \localleftbox by new LaTeX release 2015/10/01.	39
General: Add a variant of \babel@savevariable to save \XeTeXcharclass(es) in a loop.	30	v3.1h	
frenchb.lua: font.getfont(fid) possibly returns nil even for a positive fid (i.e. AMS lcircle1.pfb). Reported by François Legendre.	24	General: french.cfg from e-french conflicts with babel-french. Do NOT load it (no need for .cfg files with babel-french anyway).	75
\FB@luatex@punct@french: Use \babel@save to save and restore \shorthandon and \shorthandoff.	29	v3.1g	
\FB@xetex@punct@french: Save and restore \XeTeXinterchartokenstate, \shorthandon, \shorthandoff using \babel@savevariable and \babel@save, \XeTeXcharclass(es) using \FB@savevariable@loop.	31	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
v3.1k		Use Babel defined loops \bbl@for instead of \@for borrowed from file ltcntrl.dtx (\@for is undefined in Plain).	30
General: (pdfTeX shorthands) test on \lastskip changed from 0pt to 1sp for active punctuation for consistency with XeTeX and LuaTeX.	33	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).	27
		flag addgl set to false for '»' at the beginning of an \hbox or a paragraph or a tabular 'l' and 'c' columns.	27
		Node HLIST added; node TEMP added for the first node of \hboxes.	22
		\captionsfrench: \partname's definition depends now on flag	

PartNameFull. No need to redefine it in <code>\frenchbsetup</code>	47	definitions of <code>\ieme</code> and <code>co</code> : <code>\up</code> already does the conversion.	42
<code>\frenchsetup</code> : Bug fix for koma-scripts classes: a spurious dot was added by the <code>\partformat</code> command.	54	<code>\no</code> : Removed <code>\lowercase</code> from definitions of <code>\FrenchEnumerate</code> , ... <code>\No</code> and <code>co</code> : <code>\up</code> already does the conversion.	43
PartNameFull now just sets the flag, nothing to add to <code>\captionsfrench</code> when false.	53	v3.1a	
v3.1f		General: <code>fontspec</code> is not required for T1 fonts used with the <code>luainputenc.sty</code> package.	65
General: <code>\FBCaption@Separator</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.	50	Misplaced <code>\fi</code> for plain formats.	20
<code>\FBprocess@options</code> : Bug fix for the beamer class: figure and table captions are now consistent with babel-french's documentation. Pointed out by Denis Bitouzé.	63	New command <code>\frquote</code> for imbedded or long French quotations.	38
Definition of <code>\captionformat</code> and <code>\captiondelim</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.	63	<code>frenchb.lua</code> : Added flag <code>addgl</code> which must also be true when <code>prev</code> or <code>next</code> is not a char (i.e. <code>\kern0</code> in <code>«\texttt{a}»</code>).	27
<code>\FBthinspace</code> : <code>\FBthinspace</code> is no longer a kern but a skip (babel-french adds a <code>nobreak</code> penalty before it).	17	Codes 0x13 and 0x14 added for French quotes in T1-encoding.	22
v3.1e		Look ahead when next is a kern (i.e. in <code>«\texttt{a}»</code>).	27
<code>\frenchsetup</code> : Corrected typo: <code>SmallCapsFigTabCaptions</code> instead of <code>SmallCapsFigTabCaptions</code> . Pointed out by Céline Chevalier.	53	<code>\frenchsetup</code> : Codes 0x13 and 0x14 added for French quotes in T1-encoding. Support for older versions of LuaTeX and XeTeX dropped.	59
v3.1d		New options <code>InnerGuillSingle</code> , <code>EveryParGuill</code> and <code>EveryLineGuill</code> to control <code>\frquote</code>	53
General: New section: issue warnings if packages <code>listings</code> , <code>numprint</code> and <code>natbib</code> are loaded too early or too late vs babel.	52	v3.0c	
v3.1c		General: babel-french requires babel-3.9i.	14
<code>frenchb.lua</code> : Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André.	25	Just load <code>luatexbase.sty</code> instead of <code>luaotfload.sty</code> with plain formats.	20
v3.1b		No need to define <code>\l@french</code> as <code>\lang@french</code> , <code>babel.def</code> (3.9j) takes care for this.	15
<code>frenchb.lua</code> : Add a check for null fid in <code>french_punctuation</code> (Tikz <code>\nullfont</code>). Bug pointed out by Paul Gaborit.	24	<code>frenchb.lua</code> : Null glues should not trigger space insertion before high punctuation. Bug pointed out by Benoit Rivet for the 'Istlisting' environment of the <code>listings</code> package.	25
<code>\captionsfrench</code> : Change <code>\scshape</code> to customisable <code>\FBfigtabshape</code> for <code>\figurename</code> and <code>\tablename</code>	47	<code>\frenchsetup</code> : New option <code>INGuillSpace</code>	53
<code>\frenchsetup</code> : New option <code>SmallCapsFigTabCaptions</code>	53	No list customisation when beamer class is loaded.	54
<code>\ieres</code> : Removed <code>\lowercase</code> from		v3.0b	
		General: <code>frenchb.lua</code> was not found by Lua function <code>dofile</code> (not <code>kpathsea</code> aware). Call function <code>kpse.find_file</code> first, as suggested by Paul Gaborit.	29

Require luatexbase with LaTeX2e in case fontspec has not been loaded before babel.	20	and\babel@savevariable are usable for French.	53
v3.0a		Support for options frenchb, francais, canadien, acadian changed.	14
General: \bbl@nonfrenchguillemets deleted, use \babel@save instead.	37	Test \ifXeTeX changed to \ifFBunicode and 'xltxtra' changed to 'fontspec'.	65
\LdfInit checks \captionfrench instead of \datefrench to avoid a conflict with papertex.cls which loads datetime.sty.	14	\CaptionSeparator: Remove \FBCaption@SeparatorORI, use \babel@save instead.	49
french.cfg will be loaded (if found) instead of frenchb.cfg. NO NEED for .cfg files in French anyway. ..	75	\captionfrench: Take advantage of babel's \SetString commands for captionnames.	47
In Plain, provide a substitute for \PackageWarning and \PackageInfo.	14	\datefrench: Take advantage of babel's \SetString commands for \datefrench. Doesn't work with Plain (yet?).	40
Merging of \captionfrenchb, \captionfrançais with \captionfrench deleted in favor of new babel 3.9 syntax.	49	\descriptionFB: Added \listindentFB to \itemindent. Suggested by Denis Bitouzé.	69
More informative, less TeXnical warning about \@makecaption. ..	50	\extrasfrench: Take advantage of babel's \babel@savevariable to handle apostrophe's \lccode. ...	16
New flag \ifFB@luatex@punct for 'high punctuation' management with LuaTeX engines.	17	\FB@fg: Definitions of \FB@og and \FB@fg now depend on punctuation handling (LuaTeX / XeTeX / active). ...	36
New handling of 'high punctuation' through callbacks with LuaTeX engines.	20	\FBprocess@options: With koma-script and memoir class, customise \captionformat and \captiondelim.	63
No warning about \@makecaption for SMF classes.	50	\frenchsetup: New options OldFigTabCaptions and CustomiseFigTabCaptions.	53
Options processing completely reorganised, now \babel@save			