

A Babel language definition file for French

frenchb.dtx v3.5r, 2023-12-19

Daniel Flipo
daniel.flipo@free.fr

Contents

1	The French language	2
1.1	Basic interface	2
1.2	Customisation	4
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	35
2.3	Commands for French quotation marks	36
2.4	Date in French	40
2.5	Extra utilities	41
2.6	Formatting numbers	45
2.7	Caption names	48
2.8	Figure and table captions	50
2.9	Dots...	52
2.10	More checks about packages' loading order	53
2.11	Setup options: keyval stuff	53
2.12	French lists	67
2.13	French indentation of sections	72
2.14	Formatting footnotes	72
2.15	Clean up and exit	76
2.16	Files frenchb.ldf, francais.ldf, canadien.ldf and acadian.ldf	76
3	Change History	78

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. Version 3.0 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 v3.5r 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. 4);
3. vertical spacing in general LaTeX lists is shortened;
4. footnotes are displayed “à la française”.
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.

¹The file described in this section has version number v3.5r and was last revised on 2023-12-19.

²Always use `french` as option name for the French language, former aliases `frenchb` or `francais` 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. 4).

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 command `\frquote{}`: `\frquote{some text}` will output « some text ». Former commands `\og` and `\fg` are kept for backward compatibility: `\og some text\fg{}` is an alternative to `\frquote{some text}`.

If French quote characters are available on your keyboard, you can use them, to get proper spacing in LaTeX2e see option `og=«`, `fg=»` p. 7.

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 `>` or nothing, depending on option `EveryParGuill=open` (default) or `=close` or `=none`.
- 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’.

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.

2. `\frenchdate{<year>}{<month>}{<day>}` helps typesetting dates in French: `\frenchdate{2001}{01}{01}` will print 1^{er} janvier 2001 in a box without any linebreak.
3. A command `\up` is provided to typeset superscripts like `M\up{me}` (abbreviation for “Madame”), `1\up{er}` (for “premier”). Other commands are also provided for ordinals: `\ier`, `\iere`, `\iers`, `\ieres`, `\ieme`, `\iemes` (`3\iemes` prints 3^{es}). All these commands take advantage of real superscript letters when they are available in the current font.
4. Command `\bname{}` (boxed name) is provided to typeset family names: its argument will not be hyphenated except on explicit hyphens. `\bsc{}` (boxed small caps) is a variant that prints its argument in small capitals, it is meant for bibliographies, signatures, etc. Usage: `Albert~\bsc{Camus}`.
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 T_EXbook p. 134). The command `\DecimalMathComma` makes the comma behave as an ordinary character *when the current language is French* (no space added); as a counterpart, if `\DecimalMathComma` is active, an explicit thin space has to be added in lists and intervals: `$(x,\,y)$`, `$(0,\,1)$`. `\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 ‘`1\ier juin`’ will print ‘1^{er} juin’ (no need for a forced space after `1\ier`).

1.2 Customisation

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

1.2.1 `\frenchsetup{options}`

`\frenchbsetup{}` and `\frenchsetup{}` 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:

Text starting at ‘parindent’ <= Leftmargin — first item running on two lines or more... — first second level item on two lines... — next one... — second item...

Default French layout

Text starting at ‘parindent’ <= Leftmargin — first item running on two lines or more... — first second level item on two lines... — next one... — second item...

With `ListItemsAsPar=true`

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

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

`StandardEnumerateEnv=true (false*)`; 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 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).

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`.

`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).

`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 `\frquote{}`. 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» ⁹ (with or without spaces) to get properly typeset French quotes. This option works with LuaLaTeX, XeLaTeX and with pdfLaTeX (default encoding: utf8); with pdfLatex other 8-bits encodings (latin1,

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...`

⁹Or even «~guillemets~», but *only* with LuaLaTeX.

latin9, ansinew, applemac,...) are also supported when properly declared with `inputenc`.

`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 (~)¹⁰ 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.).

`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.

¹⁰Actually without stretch nor shrink.

`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{}`.

Version 3.5n introduces a new command `\bname{}` (an alternative to `\bsc{}`).

Version 3.5q corrects a bug in lists layout: `\listparindent` (formerly `0pt`) is defined as `\parindent` and if `\parskip > 0pt`, `\parsep` is now defined as `\parskip`. This ensures that paragraphs included in lists are now visible. The former behaviour can be recovered by adding `\parskip=0pt`, `\parindent=0pt` *inside* the list environment. Version 3.5r is compatible with `ucharclasses` which is now loaded by `fontsetup` with the XeTeX engine. The `frenchb.ins` file is no longer needed to extract the `.ldf` files from `frenchb.dtx` (see `README.md`).

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. 7.

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`, portman-teau 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' ¹¹. 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.

¹¹The 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

```

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

```

80 \newif\ifFBunicode
81 \newif\ifBLaTeX
82 \newif\ifBTeX
83 \begingroup\expandafter\expandafter\expandafter\endgroup
84 \expandafter\ifx\csname luatexversion\endcsname\relax
85 \else
86   \FBunicodetrue \BLaTeXtrue
87 \fi
88 \begingroup\expandafter\expandafter\expandafter\endgroup
89 \expandafter\ifx\csname XeTeXrevision\endcsname\relax
90 \else
91   \FBunicodetrue \BTeXtrue
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   \ifFBunicode
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 `0xFFF=4095` (formerly `0xFF=255`). The class for standard characters is 0.

```
125 \newcount\FB@stdchar
126 \newif\ifFB@xetex@punct
127 \ifdefined\XeTeXinterchartokenstate
128   \FB@xetex@puncttrue\FB@active@punctfalse
129   \ifdim\the\XeTeXversion\XeTeXrevision\p@ < 0.99994\p@
130     \chardef\FB@nonchar="FF \relax
131   \else
132     \chardef\FB@nonchar="FFF \relax
133   \fi
134   \FB@stdchar=\z@
135 \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 `\FBsetspace` command described below. A penalty will be added before these spaces to prevent line breaking.

```

136 \newcommand*{\FBguillspace}{\hskip .8\fontdimen2\font
137                plus .3\fontdimen3\font
138                minus .8\fontdimen4\font \relax}
139 \newcommand*{\FBcolonspace}{\space}
140 \newcommand*{\FBthinspace}{\hskip .5\fontdimen2\font \relax}

```

`\FBsetspace` 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 `\FBsetspace[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.

```

141 \ifLaTeXe
142 \newcommand*{\FBsetspace}[5][french]{%
143 \def\bbl@tempa{french}\def\bbl@tempb{#1}%
144 \ifx\bbl@tempa\bbl@tempb \def\bbl@tempb{}\fi
145 \@namedef{\bbl@tempb FB#2space}{\hskip #3\fontdimen2\font
146                plus #4\fontdimen3\font
147                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".

```

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

```

```

164 }
165 \@onlypreamble\FBsetspace
166 \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`.

```

167 \ifLaTeXe
168   \addto\extrasfrench{%
169     \ifFB@luatex@punct
170       \edef\bbl@tempa{\detokenize\expandafter{\language}}%
171       \edef\bbl@tempb{\detokenize{french}}%
172       \ifx\bbl@tempa\bbl@tempb \FB@dialect=\z@
173       \else \FB@dialect=\@ne
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!

```

175     \ifdefined\FB@once\else
176       \set@glue@table{colon}%
177       \set@glue@table{thin}%
178       \set@glue@table{guill}%
179       \def\FB@once{%
180         \fi
181       \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).

```

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

```

Same for `\FBcolonspace`:

```

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

```

And for `\FBguillspace`:

```

192   \ifcsname\language FBguillspace\endcsname
193     \babel@save\FBguillspace
194     \renewcommand*{\FBguillspace}{%

```

```

195             \csname\languagename FBguillspace\endcsname}%
196     \fi
197   }
198 \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.

```

199 \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).

```

200 \ifFB@luatex@punct
201   \newcommand*\FB@spacing@on{\FB@spacing=@ne}
202   \newcommand*\FB@spacing@off{\FB@spacing=@z@}
203 \else
204   \newcommand*\FB@spacing@on{\FB@spacingtrue}
205   \newcommand*\FB@spacing@off{\FB@spacingfalse}
206 \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.

```

207 \ifFB@luatex@punct
208   \ifdefined\newluafunction\else

```

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

```

209   \input l\luatex.tex
210 \fi

```

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

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

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

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

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

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

```

211 \newattribute\FB@spacing \FB@spacing=@ne
212 \newattribute\FB@addDPspace \FB@addDPspace=@ne
213 \newattribute\FB@addGUILspace \FB@addGUILspace=@z@
214 \newattribute\FB@ucsNBSP \FB@ucsNBSP=@z@
215 \newattribute\FB@dialect \FB@dialect=@z@

```

```

216 \ifLaTeXe
217   \PackageInfo{french.ldf}{No need for active punctuation
218             characters\MessageBreak with this version
219             of LuaTeX!\MessageBreak reported}
220 \else
221   \fb@info{No need for active punctuation characters\\
222           with this version of LuaTeX!}
223 \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.

```

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

```

frenchb.lua (*env.*) 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).

```
258 <*Lua>
259 local FB_punct_thin =
260   {[string.byte("!")] = true,
261    [string.byte("?")] = true,
262    [string.byte(";")] = true}
263 local FB_punct_thick =
264   {[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 ‘»’.

```
265 local FB_punct_left =
266   {[string.byte("!")] = true,
267    [string.byte("?")] = true,
268    [string.byte(";")] = true,
269    [string.byte(":")] = true,
270    [0x14] = true,
271    [0xBB] = true}
272 local FB_punct_right =
273   {[0x13] = true,
274    [0xAB] = true}
```

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

```
275 local FB_punct_null =
276   {[string.byte("!")] = true,
277    [string.byte("?")] = true,
278    [string.byte("[")] = true,
279    [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.

```
280   [0xA0] = true,
281   [0x202F] = true}
282 local FB_guil_null =
283   {[0xA0] = true,
284   [0x202F] = true}
```

Local definitions for nodes:

```
285 local new_node = node.new
286 local copy_node = node.copy
287 local node_id = node.id
288 local HLIST = node_id("hlist")
289 local TEMP = node_id("temp")
290 local KERN = node_id("kern")
291 local GLUE = node_id("glue")
292 local GLYPH = node_id("glyph")
293 local PENALTY = node_id("penalty")
```

```

294 local nobreak      = new_node(PENALTY)
295 nobreak.penalty    = 10000
296 local nbspace      = new_node(GLYPH)
297 local insert_node_before = node.insert_before
298 local insert_node_after  = node.insert_after
299 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.

```

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

```

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

```

```
339 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`.

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

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

```
364 local FBspacing      = luatexbase.attributes['FB@spacing']
365 local addDPspace     = luatexbase.attributes['FB@addDPspace']
366 local addGUILspace   = luatexbase.attributes['FB@addGUILspace']
367 local FBucsNBSP     = luatexbase.attributes['FB@ucsNBSP']
368 local FBdialect     = luatexbase.attributes['FB@dialect']
369 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`.

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

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

```
372   node.kerning(head)
373   for item in node.traverse_id(GLYPH, head) do
374     local lang = item.lang
```



```

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

```

399         local is_glue = prev_id == GLUE
400         local glue_wd
401         if is_glue then
402             glue_wd = prev.width
403         end
404         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 ≤ 1 sp for tabulars, possibly < 0) 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.

```

405         if FB_punct_thin[char] or FB_punct_thick[char] then
406             local SBDP = has_attribute(item, addDPspace)
407             local auto = SBDP and SBDP > 0
408             if FB_punct_thick[char] and auto then
409                 local next = item.next

```

```

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

```

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

```

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

```

461         elseif SIG then
462             local addgl = (prev_char and
463                 not FB_guil_null[prev_char])
464                 or
465                 (not prev_char and
466                 prev_id ~= TEMP and
467                 not (prev_id == HLIST and
468                     prev_subtype == 3)
469             )

```

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

```

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

```

496         elseif SIG then
497             local next = item.next
498             local next_id, next_subtype, next_char, nextnext, kern_wd
499             if next then

```

```

500         next_id = next.id
501         next_subtype = next.subtype

```

In case of coding «~ remove the penalty and the glue:

```

502         if next_id == PENALTY then
503             nextnext = next.next
504             if nextnext and nextnext.id == GLUE then
505                 head = remove_node(head,nextnext,true)
506                 head = remove_node(head,next,true)
507                 next = item.next
508                 if next then
509                     next_id = next.id
510                     next_subtype = next.subtype
511                     if next_id == GLYPH then
512                         next_char = next.char
513                     end
514                 end
515             end
516         end

```

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

```

517         if next_id == KERN then
518             kern_wd = next.kern
519             if kern_wd == 0 then
520                 nextnext = next.next
521                 if nextnext then
522                     next = nextnext
523                     next_id = nextnext.id
524                     next_subtype = nextnext.subtype
525                     if next_id == PENALTY then
526                         nextnext = next.next
527                         if nextnext and nextnext.id == GLUE then
528                             head = remove_node(head,next,true)
529                             head = remove_node(head,nextnext,true)
530                             next = item.next
531                             if next then
532                                 next_id = next.id
533                                 next_subtype = next.subtype
534                             end
535                         end
536                     end
537                 end
538             end
539         end
540         if next_id == GLYPH then
541             next_char = next.char
542         end
543     end
544     local is_glue = next_id == GLUE
545     if is_glue then
546         glue_wd = next.width
547     end

```

The addgl flag only depends on next_char and is_glue:

```

548         local addgl = (next_char and not FB_guil_null[next_char])
549         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).

```

550         if is_glue and glue_wd == 0 then
551             addgl = false
552         end
553         local fid = item.font
554         local t = FBsp.guill.gl.fr
555         nbspace.char = FBsp.guill.ch.fr
556         if FRdialect then
557             t = FBsp.guill.gl.ac
558             nbspace.char = FBsp.guill.ch.ac
559         end
560         local fbglue = new_glue_scaled(fid, t)
561         if addgl and fbglue then
562             if is_glue then
563                 head = remove_node(head,next,true)
564             end
565             if (FRucsNBSP) then
566                 nbspace.font = fid
567                 nbspace.attr = attr
568                 insert_node_after(head, item, copy_node(nbspace))
569             else
570                 nobreak.attr = attr
571                 fbglue.attr = attr
572                 insert_node_after(head, item, copy_node(fbglue))
573                 insert_node_after(head, item, copy_node(nobreak))
574             end
575         end
576     end
577 end
578 end
579 end
580 return head
581 end
582 return french_punctuation
583 </lua>

```

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 be done (see above, p. 19).

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`).

```

584 <*\french>
585 \ifFB@luatex@punct
586   \def\activate@luatexpunct{%
587     \directlua{%
588       FR_fr = \the\l@french ; FR_ca = \the\l@acadian ;
589       local path = kpse.find_file("frenchb.lua", "lua")

```

```

590     if path then
591         local f = dofile(path)
592         luatexbase.add_to_callback("kerning",
593             f, "frenchb.french_punctuation")
594     else
595         texio.write_nl('')
596         texio.write_nl('*****')
597         texio.write_nl('Error: frenchb.lua not found.')
598         texio.write_nl('*****')
599         texio.write_nl('')
600     end
601 }%
602 }
603 \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).

Unless `ucharclass` is loaded, the default value for `\XeTeXcharclass` is 0 for characters tokens and `\FB@nonchar` for all other tokens (glues, kerns, math and box boundaries, etc.). `ucharclass` defines a XeTeX class for every range of Unicode characters in order to facilitate font switching. Most French characters belong to range [“20, “7F] (class `\BasicLatinClass`) some (accented chars, diacritics,...) to range [“80, “FF] (class `\LatinSupplementClass`) and three (œ, Œ, and the long-s) to [“100, “17F] (class `\LatinExtendedAClass`).

We check `AtBeginDocument` whether `ucharclass` is loaded; if so, when switching to French, the class `\FB@stdchar` of all characters possibly used in French (except punctuation) will be forced to `\BasicLatinClass` which is the default for most of them, the class of the others (accented chars, ligatures, diacritics, etc.) will be saved and changed locally in French, then restored to their original value when leaving French. We switch `\XeTeXinterchartokenstate` to 1 and change the `\XeTeXcharclass` values of ; ! ? : (] « and » when entering French. Their initial values will be restored when leaving French.

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

```

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

```

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

```

613 \newXeTeXintercharclass\FB@punctthick
614 \newXeTeXintercharclass\FB@punctthin
615 \newXeTeXintercharclass\FB@punctnul
616 \newXeTeXintercharclass\FB@guilo
617 \newXeTeXintercharclass\FB@guilf
618 \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.

```

619 \def\FBsavevariable@loop#1#2{\begingroup
620   \toks@\expandafter{\originalTeX #1}%
621   \edef\x{\endgroup
622     \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
623   \x}

```

`\FB@charlistsave` holds the all list of characters which have their `\XeTeXcharclass` value modified in French: it always includes high punctuation, French quotes, opening delimiters and no-break spaces. If `ucharclasses` is loaded, non-ascii characters used in French have to be added; as `xeCJK` changes the class of some characters used in French, these have to be saved too if `xeCJK` is loaded.

```

624 \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F}
625 \def\FB@charlistUCC{}
626 \def\FB@charlistxeCJK{}
627 \edef\FB@charlistsave{\FB@charlist}
628 \ifLaTeXe
629   \AtBeginDocument{%
630     \ifpackageloaded{ucharclasses}%
631     {\ifdefined\BasicLatinClass
632       \RenewCommandCopy{\FB@stdchar}{\BasicLatinClass}%
633       \def\FB@charlistUCC{"C0,"C2,"C6,"C7,"C8,"C9,"CA,"CB,"CE,"CF,%
634         "D4,"D6,"D9,"DB,"DC,"E0,"E2,"E6,"E7,"E8,"E9,"EA,"EB,"EE,%
635         "EF,"F4,"F6,"F9,"FB,"FC,"152,"153,"17F,"2019}%
636       \addto\FB@charlist{\FB@charlistUCC}%
637       \edef\FB@charlistsave{\FB@charlist}%
638       \fi
639     }{}}%
640   \ifpackageloaded{xeCJK}%
641   {\def\FB@charlistxeCJK{%
642     "29,"5D,"7B,"7D,"2C,"2D,"2E,"22,"25,"27,"60,"2019}%
643     \addto\FB@charlist{\FB@charlistxeCJK}%
644     \edef\FB@charlistsave{\FB@charlist}%
645     }{}}%
646   }
647 \fi

```

`\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.

```

648 \newcommand*\FB@xetex@punct@french{%
649   \babel@savevariable{\XeTeXinterchartokenstate}%
650   \bbl@for\FB@char\FB@charlistsave
651     {\FBsavevariable@loop{\XeTeXcharclass}{\FB@char}}%

```

If `ucharclasses` is loaded, force non-ascii used in French to class `\FB@stdchar` (`=\BasicLatinClass`).

```

652 \ifx\FB@charlistUCC\@empty\else
653 \bbl@for\FB@char\FB@charlistUCC
654 {\XeTeXcharclass\FB@char\FB@stdchar}%
655 \fi

```

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

```

656 \ifx\FB@charlistxeCJK\@empty\else
657 \bbl@for\FB@char\FB@charlistxeCJK
658 {\XeTeXcharclass\FB@char=\FB@stdchar}%
659 \fi

```

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

```

660 \bbl@for\FB@char {\[, \[, "A0, "202F}%
661 {\XeTeXcharclass\FB@char=\FB@punctnul}%

```

Let's now define specific classes for punctuation 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).

```

662 \XeTeXinterchartokenstate=\@ne
663 \XeTeXcharclass \: = \FB@punctthick
664 \XeTeXinterchartoks \FB@stdchar \FB@punctthick = {%
665 \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
666 \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
667 \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.

```

668 \XeTeXinterchartoks \FB@nonchar \FB@punctthick = {%
669 \ifFB@spacing
670 \ifhmode
671 \ifdim\lastskip>1sp
672 \unskip\penalty\M\FBcolonspace
673 \else
674 \FDP@colonspace
675 \fi
676 \fi
677 \fi}%
678 \bbl@for\FB@char {\;, \!, \?}%
679 {\XeTeXcharclass\FB@char=\FB@punctthin}%
680 \XeTeXinterchartoks \FB@stdchar \FB@punctthin = {%
681 \ifFB@spacing\ifhmode\FDP@thinspace\fi\fi}%
682 \XeTeXinterchartoks \FB@guilf \FB@punctthin = {%
683 \ifFB@spacing\FDP@thinspace\fi}%
684 \XeTeXinterchartoks \FB@nonchar \FB@punctthin = {%
685 \ifFB@spacing
686 \ifhmode
687 \ifdim\lastskip>1sp
688 \unskip\penalty\M\FBthinspace

```



```

689         \else
690             \FDP@thinspace
691         \fi
692     \fi
693 \fi}%
694 \XeTeXinterchartoks \FB@guilo \FB@stdchar = {%
695     \ifFB@spacing\FB@guillspace\fi}%
696 \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
697     \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
698 \XeTeXinterchartoks \FB@stdchar \FB@guilf = {%
699     \ifFB@spacing\FB@guillspace\fi}%
700 \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
701     \ifFB@spacing\FB@guillspace\fi}%
702 \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
703     \ifFB@spacing\unskip\FB@guillspace\fi}%
704 }
705 \addto\extrasfrench{\FB@xetex@punct@french}

```

End of specific code for punctuation with modern XeTeX engines.

```
706 \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 :

```

707 \newif\ifFB@koma
708 \ifLaTeXe
709   \@ifclassloaded{scrartcl}{\FB@komatruel}{\FB@komatruer}{}
710   \@ifclassloaded{scrbook}{\FB@komatruel}{\FB@komatruer}{}
711   \@ifclassloaded{scrreprt}{\FB@komatruel}{\FB@komatruer}{}
712   \ifFB@koma\def\FB@std@capsep{: \ } \fi
713   \@ifclassloaded{beamer}{\def\FB@std@capsep{: \ } \fi}{}
714   \@ifclassloaded{memoir}{\def\FB@std@capsep{: } \fi}{}
715 \fi
716 \ifFB@active@punct
717   \initiate@active@char{:}%
718   \initiate@active@char{;}%
719   \initiate@active@char{!}%
720   \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.

```

721 \declare@shorthand{french}{;}{;%
722     \ifFB@spacing
723         \ifhmode
724             \ifdim\lastskip>1sp
725                 \unskip\penalty\M\FBthinspace

```

```

726     \else
727     \FDP@thinspace
728     \fi
729     \fi
730     \fi

```

Now we can insert a ; character.

```

731     \string;}

```

The next three definitions are very similar.

```

732 \declare@shorthand{french}{!}{%
733   \ifFB@spacing
734   \ifhmode
735     \ifdim\lastskip>1sp
736     \unskip\penalty\@M\FBthinspace
737   \else
738     \FDP@thinspace
739   \fi
740   \fi
741   \fi
742   \string!}
743 \declare@shorthand{french}{?}{%
744   \ifFB@spacing
745   \ifhmode
746     \ifdim\lastskip>1sp
747     \unskip\penalty\@M\FBthinspace
748   \else
749     \FDP@thinspace
750   \fi
751   \fi
752   \fi
753   \string?}
754 \declare@shorthand{french}{:}{%
755   \ifFB@spacing
756   \ifhmode
757     \ifdim\lastskip>1sp
758     \unskip\penalty\@M\FBcolonspace
759   \else
760     \FDP@colonspace
761   \fi
762   \fi
763   \fi
764   \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.

```

765 \declare@shorthand{system}{:}{\string:}
766 \declare@shorthand{system}{!}{\string!}
767 \declare@shorthand{system}{?}{\string?}
768 \declare@shorthand{system}{;}{\string;}

```

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

```

769 \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.

```

770   \bbl@activate{:}\bbl@activate{;}%
771   \bbl@activate{!}\bbl@activate{?}%
772   }
773   \addto\noextrsfrench{%
774     \bbl@deactivate{:}\bbl@deactivate{;}%
775     \bbl@deactivate{!}\bbl@deactivate{?}%
776   }
777 \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.

```
778 \newif\ifFBAutoSpacePunctuation \FBAutoSpacePunctuationtrue
```

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

Set the default now for Plain (done later for LaTeX).

```

779 \def\autospace@beforeFDP{%
780   \ifFB@luatex@punct \FB@addDPspace=\@ne \fi
781   \def\FDP@thinspace{\penalty\@M\FBthinspace}%
782   \def\FDP@colonspace{\penalty\@M\FBcolonspace}}
783 \def\noautospace@beforeFDP{%
784   \ifFB@luatex@punct \FB@addDPspace=\@z@ \fi
785   \let\FDP@thinspace\@empty
786   \let\FDP@colonspace\@empty}
787 \ifLaTeXe
788   \def\AutoSpaceBeforeFDP{\autospace@beforeFDP
789     \FBAutoSpacePunctuationtrue}
790   \def\NoAutoSpaceBeforeFDP{\noautospace@beforeFDP
791     \FBAutoSpacePunctuationfalse}
792   \AtEndOfPackage{\AutoSpaceBeforeFDP}
793 \else
794   \let\AutoSpaceBeforeFDP\autospace@beforeFDP
795   \let\NoAutoSpaceBeforeFDP\noautospace@beforeFDP
796   \AutoSpaceBeforeFDP
797 \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.

```
798 \ifLaTeXe
799 \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
800 \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on \rmfamilyORI}
801 \DeclareRobustCommand\sffamilyFB{\FB@spacing@on \sffamilyORI}
802 \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.

```
803 \DeclareRobustCommand*\NoAutoSpacing*{%
804   \FB@spacing@off
805   \ifFB@active@punct\shorthandoff{;:!?}\fi
806 }
```

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.

```
807 \ifLaTeXe
808 \else
809   \ifFBunicode
810     \def\guillemotleft{{\char"00AB}}
811     \def\guillemotright{{\char"00BB}}
812     \def\textquotedblleft{{\char"201C}}
813     \def\textquotedblright{{\char"201D}}
814   \else
815     \def\guillemotleft{\leavevmode\raise0.25ex
816       \hbox{$\scriptscriptstyle\ll$}}
817     \def\guillemotright{\raise0.25ex
818       \hbox{$\scriptscriptstyle\gg$}}
819     \def\textquotedblleft{``}
820     \def\textquotedblright{''}
821   \fi
822   \let\xspace\relax
823 \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` (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.

```
824 \providecommand\texorpdfstring[2]{#1}
825 \newcommand*\FB@og{\texorpdfstring{\@FB@og}{\guillemotleft\space}}
826 \newcommand*\FB@fg{\texorpdfstring{\@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.

```
827 \newcommand*\FB@guillspace{\penalty\M\FBguillspace}
828 \newcommand*\FBgspchar{\char"A0\relax}
829 \newif\ifFBucsNBSP
830 \ifFB@luatex@punct
831   \DeclareRobustCommand*\@FB@og{\leavevmode
832     \bgroup\FB@spacing=\z@ \guillemotleft\egroup
833     \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi}
834   \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@\unskip\fi
835     \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi
836     \bgroup\FB@spacing=\z@ \guillemotright\egroup}
837 \fi
```

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

```
838 \ifFB@xetex@punct
839   \DeclareRobustCommand*\@FB@og{\leavevmode
840     \bgroup\FB@spacingfalse\guillemotleft\egroup
841     \FB@guillspace}
842   \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@\unskip\fi
843     \FB@guillspace
844     \bgroup\FB@spacingfalse\guillemotright\egroup}
845 \fi
846 \ifFB@active@punct
847   \DeclareRobustCommand*\@FB@og{\leavevmode
848     \guillemotleft
849     \FB@guillspace}
850   \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@\unskip\fi
851     \FB@guillspace
852     \guillemotright}
853 \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.

```
854 \newcommand*\og{\@empty}
855 \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).

```

856 \ifLaTeXe
857   \def\bbbl@frenchguillemets{%
858     \renewcommand*\og{\FB@og}%
859     \renewcommand*\fg{\FB@fg\xspace}}
860   \renewcommand*\og{\textquotedblleft}
861   \renewcommand*\fg{\ifdim\lastskip>\z@\unskip\fi
862     \textquotedblright\xspace}
863 \else
864   \def\bbbl@frenchguillemets{\let\og\FB@og
865     \let\fg\FB@fg}
866   \def\og{\textquotedblleft}
867   \def\fg{\ifdim\lastskip>\z@\unskip\fi\textquotedblright}
868 \fi

869 \addto\extrasfrench{\babel@save\og \babel@save\fg
870   \bbbl@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...

```

871 \newcommand*\ogi{\FB@og}
872 \newcommand*\fgi{\FB@fg}
873 \newcommand*\@ogi{\ifmmode\hbox{\ogi}\else\ogi\fi}
874 \newcommand*\@fgi{\ifmmode\hbox{\fgi}\else\fgi\fi}
875 \newcommand*\ogii{\textquotedblleft}
876 \newcommand*\fgii{\textquotedblright}
877 \newcommand*\@ogii{\ifmmode\hbox{\ogii}\else\ogii\fi}
878 \newcommand*\@fgii{\ifmmode\hbox{\fgii}\else\fgii\fi}

```

and the needed technical stuff to handle options:

```

879 \newcount\FBguill@level
880 \newtoks\FBold@everypar

```

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

```

881 \def\FB@addquote@everypar{%
882   \let\FBnew@everypar\everypar
883   \FBold@everypar=\expandafter{\the\everypar}%
884   \FBnew@everypar={\the\FBold@everypar\FBeverypar@quote}%
885   \let\everypar\FBold@everypar
886   \let\FB@addquote@everypar\relax
887 }
888 \newif\ifFBcloseguill \FBcloseguilltrue
889 \newif\ifFBInnerGuillSingle
890 \def\FBguillopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
891 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
892 \let\FBguillnone\empty
893 \let\FBeveryparguill\FBguillopen
894 \let\FBeverylineguill\FBguillnone
895 \let\FBeverypar@quote\relax
896 \let\FBeveryline@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.

```

897 \ifLaTeXe
898   \DeclareRobustCommand\frquote{%
899     \texorpdfstring{\@ifstar{\FBcloseguillfalse\fr@quote}%
900                       {\FBcloseguilltrue \fr@quote}}%
901                       {\bm@fr@quote}%
902   }
903   \newcommand{\bm@fr@quote}[1]{%
904     \guillemotleft\space #1\space\guillemotright}
905 \else
906   \newcommand\frquote[1]{\fr@quote{#1}}
907 \fi

```

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

```

908 \newcommand{\fr@quote}[1]{%
909   \leavevmode
910   \advance\FBguill@level by \@ne
911   \ifcase\FBguill@level
912     \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:

```

913   \ifx\FBEveryparguill\FBguillnone
914   \else
915     \def\FBEverypar@quote{\FBEveryparguill\FB@guillspace}%
916     \FB@addquote@everypar
917   \fi
918   \@ogi #1\@fgi
919   \or

```

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

```

920   \ifx\FBEverylineguill\FBguilllopen
921     \def\FBEveryline@quote{\FB@addGUILLspace=\z@
922                           \guillemotleft\FBguillspace}%
923     \lcalleftbox{\FBEveryline@quote}%
924     \let\FBEverypar@quote\relax
925     \@ogi #1\ifFBcloseguill\@fgi\fi
926   \else
927     \ifx\FBEverylineguill\FBguillclose
928       \def\FBEveryline@quote{\FB@addGUILLspace=\z@
929                             \guillemotright\FBguillspace}%
930       \lcalleftbox{\FBEveryline@quote}%
931       \let\FBEverypar@quote\relax
932       \@ogi #1\ifFBcloseguill\@fgi\fi
933     \else

```

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

```

934       \let\FBEverypar@quote\relax
935       \ifFBInnerGuillSingle

```

```

936     \def\ogii{\leavevmode
937         \guilsinglleft\FB@guillspace}%
938     \def\fgii{\ifdim\lastskip>\z@\unskip\fi
939         \FB@guillspace\guilsinglright}%
940     \ifx\FBeveryparguill\FBguillopen
941         \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
942     \fi
943     \ifx\FBeveryparguill\FBguillclose
944         \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
945     \fi
946     \fi
947     \@ogii #1\ifFBcloseguill \@fgii \fi
948 \fi
949 \fi
950 \else

```

Warn if \FBguill@level > 2:

```

951     \ifx\PackageWarning@undefined
952         \fb@warning{\noexpand\frquote\space handles up to
953             two levels.\\ Quotation not printed.}%
954     \else
955         \PackageWarning{french.ldf}{%
956             \protect\frquote\space handles up to two levels.
957             \MessageBreak Quotation not printed. Reported}
958     \fi
959 \fi

```

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

```

960 \global\advance\FBguill@level by \m@ne
961 \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
962 \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
963 \global\let\FBeveryline@quote\empty
964 \ifx\FBeverylineguill\FBguillnone\else\localleftbox{}\fi
965 \fi
966 }

```

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.

```

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

```

2.4 Date in French

\frenchtoday The following code creates a macro \datefrench which in turn defines command \frenchdate \frenchtoday (\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.


```

968 \def\BabelLanguages{french,acadian}
969 \StartBabelCommands*\BabelLanguages}{date}
970   [unicode, fontenc=TU EU1 EU2, charset=utf8]
971   \SetString\monthiiname{février}
972   \SetString\monthviiname{août}
973   \SetString\monthxiiname{décembre}
974 \StartBabelCommands*\BabelLanguages}{date}
975   \SetStringLoop{month#1name}{%
976     janvier,f\`evrier,mars,avril,mai,juin,juillet,%
977     ao\`ut,septembre,octobre,novembre,d\`ecembre}
978   \SetString\today{\FB@date{\year}{\month}{\day}}
979 \EndBabelCommands

```

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

```

980 \newcommand*\FB@date}[3]{%
981   {\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
982   \csname month\romannumeral#2name\endcsname
983   \ifx#1\@empty\else\FBdatespace\number#1\fi}
984 \newcommand*\FBdatebox{\hbox}
985 \newcommand*\FBdatespace{\space}
986 \newcommand*\frenchdate{\FBdatebox\FB@date}
987 \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 `scalefnt` 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).

```

988 \newif\ifFB@poorman
989 \newdimen\FB@Mht
990 \ifLaTeXe
991   \AtEndOfPackage{\RequirePackage{scalefnt}}

```

`\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{}`.

```

992 \newcommand*\FBsupR}{-0.12}

```

```

993 \newcommand*{\FBsupS}{0.65}
994 \newcommand*{\FB@lc}[1]{\MakeLowercase{#1}}
995 \DeclareRobustCommand*{\FB@up@fake}[1]{%
996   \settoheight{\FB@Mht}{M}%
997   \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
998   \addtolength{\FB@Mht}{-\FBsupS ex}%
999   \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
1000 }

```

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.

```

1001 \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
1002   \def\FB@suffix{#4}}
1003 \def\FB@x{x}
1004 \def\FB@j{j}
1005 \DeclareRobustCommand*{\FB@up}[1]{%
1006   \bgroup \FB@poormantrue
1007   \expandafter\FB@split\f@family\@nil

```

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

```

1008   \edef\reserved@a{\lowercase{%
1009     \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
1010   \reserved@a
1011   {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
1012    \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
1013    \ifFB@poorman \FB@up@fake{#1}%
1014    \else \FB@up@real{#1}%
1015    \fi}%
1016   {\FB@up@fake{#1}}%
1017   \egroup}

```

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

```

1018 \newcommand*{\FB@up@real}[1]{\bgroup
1019   \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.

```

1020 \DeclareRobustCommand*{\fup}[1]{%
1021   \texorpdfstring{\ifx\realsuperscript\@undefined
1022     \FB@up{#1}%
1023     \else

```

```

1024             \bgroup\let\fakesuperscript\FB@up@fake
1025             \realsuperscript{\FB@lc{#1}}\egroup
1026             \fi
1027             }{#1}%
1028   }

```

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

```
1029 \providecommand*\up{\fup}
```

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

```

1030 \else
1031 \providecommand*\up}[1]{\leavevmode\raiselex\hbox{\sevenrm #1}}
1032 \fi

```

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

```

\ier 1033 \def\ieme{\up{e}\xspace}
\iere 1034 \def\iemes{\up{es}\xspace}
\iemes 1035 \def\ier{\up{er}\xspace}
\iers 1036 \def\iers{\up{ers}\xspace}
\ieres 1037 \def\iere{\up{re}\xspace}
1038 \def\ieres{\up{res}\xspace}

```

```

\FBmedkern
\FBthickkern 1039 \newcommand*\FBmedkern{\kern+.2em}
1040 \newcommand*\FBthickkern{\kern+.3em}

```

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

```

\fprimo) 1041 \newcommand*\FrenchEnumerate}[1]{%
\nos 1042 #1\texorpdfstring{\up{o}\FBthickkern}{\textdegree\space}}
\Nos 1043 \newcommand*\FrenchPopularEnumerate}[1]{%
\No 1044 #1\texorpdfstring{\up{o}}\FBthickkern}{\textdegree\space}}

```

`\no` Typing `\primo` should result in '°' (except in bookmarks where `\textdegree` is used instead of o-superior),

```

1045 \def\primo{\FrenchEnumerate1}
1046 \def\secundo{\FrenchEnumerate2}
1047 \def\tertio{\FrenchEnumerate3}
1048 \def\quarto{\FrenchEnumerate4}

```

while typing `\fprimo` gives '°) (except in bookmarks where `\textdegree` is used instead),.

```

1049 \def\fprimo{\FrenchPopularEnumerate1}
1050 \def\fsecundo{\FrenchPopularEnumerate2}
1051 \def\ftertio{\FrenchPopularEnumerate3}
1052 \def\fquarto{\FrenchPopularEnumerate4}

```

Let's provide four macros for the common abbreviations of "Numéro". In bookmarks ° is used instead of o-superior.

```

1053 \DeclareRobustCommand*\No}{%
1054 \texorpdfstring{N\up{o}\FBmedkern}{N\textdegree\space}}
1055 \DeclareRobustCommand*\no}{%
1056 \texorpdfstring{n\up{o}\FBmedkern}{n\textdegree\space}}

```

```

1057 \DeclareRobustCommand*\Nos}{%
1058   \texorpdfstring{N\up{os}\FBmedkern}{N\textdegree\space}}
1059 \DeclareRobustCommand*\nos}{%
1060   \texorpdfstring{n\up{os}\FBmedkern}{n\textdegree\space}}

```

`\bname` These commands are meant to easily enter family names (in small capitals for the latter) while avoiding hyphenation. A `\kern0pt` is used instead of `\mbox` because `\mbox` would break microtype's font expansion; as a positive side effect, composed names (such as Dupont-Durand) can now be hyphenated on explicit hyphens.

```

1061 \ifLaTeXe
1062   \DeclareRobustCommand*\bname}[1]{%
1063     \texorpdfstring{\leavevmode\begingroup\kern0pt #1\endgroup}{#1}%
1064   }
1065   \DeclareRobustCommand*\bsc}[1]{%
1066     \texorpdfstring{\leavevmode\begingroup\kern0pt \scshape #1\endgroup}%
1067     {\textsc{#1}}}%
1068   }
1069 \else
1070   \newcommand*\bname}[1]{\leavevmode\begingroup\kern0pt #1\endgroup}
1071   \let\bsc\bname
1072 \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.

```

1073 \ifFBunicode
1074   \providecommand*\textbackslash{{\char"005C}}
1075   \providecommand*\textasciicircum{{\char"005E}}
1076   \providecommand*\textasciitilde{{\char"007E}}
1077   \newcommand*\FB@degree{°}
1078 \else
1079   \ifLaTeXe
1080     \newcommand*\FB@degree{\r{}}
1081   \fi
1082 \fi
1083 \DeclareRobustCommand*\boi{\textbackslash}
1084 \DeclareRobustCommand*\circonflexe{\textasciicircum}
1085 \DeclareRobustCommand*\tild{\textasciitilde}
1086 \DeclareRobustCommand*\degree{%
1087   \texorpdfstring{\FB@degree}{\textdegree}}
1088 \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.

```

1089 \DeclareRobustCommand*\degrees{\degre}
1090 \ifLaTeXe
1091   \AtBeginDocument{%
1092     \@ifpackageloaded{fontspec}{\%
1093       \ifdefined\DeclareEncodingSubset
1094         \DeclareRobustCommand*\degrees{%
1095           \texorpdfstring{\hbox{\UseTextSymbol{TS1}{\textdegree}}}{%
1096             \textdegree}}%
1097       \else
1098         \def\Warning@degree@TSone{\FBWarning
1099           {Degrees would look better in TS1-encoding:%
1100             \MessageBreak add \protect
1101             \usepackage{textcomp} to the preamble.%
1102             \MessageBreak Degrees used}}
1103         \DeclareRobustCommand*\degrees{%
1104           \texorpdfstring{\hbox to 0.3em{\hss\degre\hss}}{
1105             \Warning@degree@TSone
1106             \global\let\Warning@degree@TSone\relax}%
1107           {\textdegree}}%
1108       \fi
1109     }%
1110   }
1111 \fi

```

2.6 Formatting numbers

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

```

1112 \newif\ifFB@icomma
1113 \newcount\mc@charclass
1114 \newcount\mc@charfam
1115 \newcount\mc@charslot
1116 \newcount\std@mcc
1117 \newcount\dec@mcc
1118 \ifFBLuaTeX
1119   \mc@charclass=\Umathcharclass`,
1120   \newcommand*\dec@math@comma{%
1121     \mc@charfam=\Umathcharfam`,
1122     \mc@charslot=\Umathcharslot`,
1123     \Umathcode`,= 0 \mc@charfam \mc@charslot
1124   }
1125   \newcommand*\std@math@comma{%
1126     \mc@charfam=\Umathcharfam`,
1127     \mc@charslot=\Umathcharslot`,
1128     \Umathcode`,= \mc@charclass \mc@charfam \mc@charslot
1129   }
1130 \else

```

```

1131 \std@mcc=\mathcode`,
1132 \dec@mcc=\std@mcc
1133 \@tempcnta=\std@mcc
1134 \divide\@tempcnta by "1000
1135 \multiply\@tempcnta by "1000
1136 \advance\dec@mcc by -\@tempcnta
1137 \newcommand*\dec@math@comma{\mathcode`,=\dec@mcc}
1138 \newcommand*\std@math@comma{\mathcode`,=\std@mcc}
1139 \fi
1140 \let\dec@m@c\relax

```

If `\DecimalMathComma` is issued in the document body (when the current language is French or Acadian) its effect will survive to a language switch, unless issued inside a group (see `\dec@m@c`'s expansion). The `icomma` inhibits `\DecimalMathComma`.

```

1141 \newif\if@FBpreamble
1142 \ifLaTeXe \@FBpreambletrue \fi
1143 \newif\if@preamble@DecimalMathComma
1144 \newcommand*\DecimalMathComma{%
1145   \if@FBpreamble \@preamble@DecimalMathCommatrue
1146   \else
1147     \if@icomma
1148       \PackageWarning{french.ldf}{%
1149         icomma package loaded, \protect\DecimalMathComma\MessageBreak
1150         does nothing. Reported}%
1151     \else
1152       \if@FBfrench
1153         \dec@math@comma
1154         \let\dec@m@c\dec@math@comma
1155         \expandafter\addto\csname extras\language\endcsname
1156         {\dec@m@c}%
1157       \fi
1158     \fi
1159   \fi
1160 }
1161 \newcommand*\StandardMathComma{%
1162   \if@icomma
1163     \PackageWarning{french.ldf}{%
1164       icomma package loaded, \protect\StandardMathComma\MessageBreak
1165       does nothing. Reported}%
1166   \else
1167     \if@FBfrench
1168       \std@math@comma
1169       \let\dec@m@c\relax
1170     \fi
1171   \fi
1172 }

```

This is for Plain formats *only* (see below).

```

1173 \ifLaTeXe\else
1174   \addto\noextrasfrench{\std@math@comma}
1175 \fi

```

Fake command `\nombre` for Plain based formats, warning users of `babel-french v. 1.x.` about the change:

```

1176 \newcommand*{\nombre}[1]{\fb@warning{*** \noexpand\nombre
1177                               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.

```

1178 \ifFB@luatex@punct
1179   \activate@luatexpunct
1180 \fi
1181 \let\FBstop@here\relax
1182 \def\FBclean@on@exit{%
1183   \let\ifLaTeXe\iffalse
1184   \let\LaTeXettrue\undefined
1185   \let\LaTeXefalse\undefined
1186   \let\FB@llc\loadlocalcfg
1187   \let\loadlocalcfg\@gobble}
1188 \ifx\magnification\@undefined
1189 \else
1190   \def\FBstop@here{%
1191     \FBclean@on@exit
1192     \ldf@finish\CurrentOption
1193     \let\loadlocalcfg\FB@llc
1194     \endinput}
1195 \fi
1196 \FBstop@here

```

What follows is for LaTeX2e *only*: the next piece of code would break Plain formats. If issued in the preamble, `\DecimalMathComma` works globally on all parts of the document that are typeset in a French dialect. Can be canceled anytime by `\StandardMathComma`.

```

1197 \AtBeginDocument{%
1198   \@FBpreamblefalse
1199   \@ifpackageloaded{icomma}%
1200     {\FB@icommatrue
1201       \if@preamble@DecimalMathComma
1202         \PackageWarning{french.ldf}{%
1203           icomma package loaded, \protect\DecimalMathComma%
1204           \MessageBreak does nothing. Reported}%
1205       \fi
1206     }%
1207   {\if@preamble@DecimalMathComma
1208     \ifFB@mainlanguage@FR \dec@math@comma \fi
1209     \let\dec@m@c\dec@math@comma
1210     \addto\extrasfrench{\dec@m@c}%
1211     \ifdefined\extrasacadian
1212       \addto\extrasacadian{\dec@m@c}%
1213     \fi
1214   \fi

```

The comma is reset to type `\mathpunct` when leaving French dialects (only if the `icomma` package is not loaded).

```

1215     \addto\noextrasfrench{\std@math@comma}%
1216     \ifdefined\noextrasacadian
1217       \addto\noextrasacadian{\std@math@comma}%

```

```

1218     \fi
1219     }%
1220 }

```

nombre We redefine `\nombre` for LaTeX2e. 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. 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.

```

1221 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1222 \newcommand*{\Warning@nombre}[1]{%
1223   \ifdefined\numprint
1224     \numprint{#1}%
1225   \else
1226     \PackageWarning{french.ldf}{%
1227       \protect\nombre\space now relies on package numprint.sty,%
1228       \MessageBreak add \protect
1229       \usepackage[autolanguage]{numprint},\MessageBreak
1230       see file numprint.pdf for more options.\MessageBreak
1231       \protect\nombre\space called}%
1232     \global\let\Warning@nombre\relax
1233     {#1}%
1234   \fi
1235 }

1236 \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`.

```

1237 \providecommand*{\FBfigtabshape}{\scshape}

```

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

```

1238 \StartBabelCommands*{\BabelLanguages}{captions}
1239   [unicode, fontenc=TU EU1 EU2, charset=utf8]
1240   \SetString{\refname}{Références}
1241   \SetString{\abstractname}{Résumé}
1242   \SetString{\prefacename}{Préface}
1243   \SetString{\contentsname}{Table des matières}
1244   \SetString{\ccname}{Copie à }
1245   \SetString{\proofname}{Démonstration}
1246   \SetString{\partfirst}{Première}
1247   \SetString{\partsecond}{Deuxième}
1248   \SetStringLoop{ordinal#1}{%
1249     \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%

```



```

1250     Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1251     Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1252     Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1253 \StartBabelCommands*\BabelLanguages}{captions}
1254   \SetString{\refname}{R\ 'ef\ 'erences}
1255   \SetString{\abstractname}{R\ 'esum\ 'e}
1256   \SetString{\bibname}{Bibliographie}
1257   \SetString{\prefacename}{Pr\ 'eface}
1258   \SetString{\chaptername}{Chapitre}
1259   \SetString{\appendixname}{Annexe}
1260   \SetString{\contentsname}{Table des mati\`eres}
1261   \SetString{\listfigurename}{Table des figures}
1262   \SetString{\listtablename}{Liste des tableaux}
1263   \SetString{\indexname}{Index}
1264   \SetString{\figurename}{Figure}
1265   \SetString{\tablename}{Table}
1266   \SetString{\pagename}{page}
1267   \SetString{\seename}{voir}
1268   \SetString{\alsoname}{voir aussi}
1269   \SetString{\enclname}{P.~J. }
1270   \SetString{\ccname}{Copie \ `a }
1271   \SetString{\headtoname}{ }
1272   \SetString{\proofname}{D\ 'emonstration}
1273   \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`.

```

1274   \SetString{\partfirst}{Premi\`ere}
1275   \SetString{\partsecond}{Deuxi\`eme}
1276   \SetString{\partnameord}{partie}
1277   \SetStringLoop{ordinal#1}{%
1278     \partfirst,\partsecond,Troisi\`eme,Quatri\`eme, Cinq\`ui\`eme,%
1279     Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
1280     Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1281     Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1282     Vingti\`eme}
1283 \AfterBabelCommands{%
1284   \DeclareRobustCommand*\FB@emptypart{\def\thepart{\unskip}}%
1285   \DeclareRobustCommand*\FB@partname{%
1286     \ifFBPartNameFull
1287       \csname ordinal\romannumeral\value{part}\endcsname\space
1288       \partnameord\FB@emptypart
1289     \else
1290       Partie%
1291     \fi}%
1292   }
1293   \SetString{\partname}{\FB@partname}
1294 \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.).

```

1295 \AtBeginDocument{%

```

```

1296 \ifx\FBfigtabshape\relax
1297 \else
1298   \ifdefined\fnun@figure
1299     \let\fnun@figureORI\fnun@figure
1300     \renewcommand{\fnun@figure}{\ifFBfrench\FBfigtabshape\fi
1301                                   \fnun@figureORI}}%
1302   \fi
1303   \ifdefined\fnun@table
1304     \let\fnun@tableORI\fnun@table
1305     \renewcommand{\fnun@table}{\ifFBfrench\FBfigtabshape\fi
1306                                   \fnun@tableORI}}%
1307   \fi
1308 \fi
1309 }

```

2.8 Figure and table captions

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

```
1310 \newcommand{\FBWarning}[1]{\PackageWarning{french.ldf}{#1}}
```

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

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

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

```

1311 \bgroup
1312 \catcode`:=12 \catcode`>=12 \relax
1313 \long\gdef\STD@makecaption#1#2{%
1314   \vskip\abovecaptionskip
1315   \sbox\@tempboxa{#1: #2}%
1316   \ifdim \wd\@tempboxa >\hsize
1317     #1: #2\par
1318   \else
1319     \global \@minipagefalse
1320     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1321   \fi
1322   \vskip\belowcaptionskip}
1323 \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.

```
1324 \newif\if@FBwarning@capsep
1325 \ifFB@active@punct\@FBwarning@capseptrue\fi
1326 \newcommand*\CaptionSeparator{\space\textendash\space}
1327 \def\FBCaption@Separator{: }
1328 \long\def\FB@makecaption#1#2{%
1329   \vskip\abovecaptionskip
1330   \box\@tempboxa{#1\FBCaption@Separator #2}%
1331   \ifdim \wd\@tempboxa >\hsize
1332     #1\FBCaption@Separator #2\par
1333   \else
1334     \global \@minipagefalse
1335     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1336   \fi
1337   \vskip\belowcaptionskip}
```

Disable the standard warning with AMS and SMF classes.

```
1338 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1339 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1340 \@ifclassloaded{amsdtx}{\@FBwarning@capsepfalse}{}
1341 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1342 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1343 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1344 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}
1345 \@ifclassloaded{IEEEconf}{\@FBwarning@capsepfalse}{}
1346 \@ifclassloaded{IEEEtran}{\@FBwarning@capsepfalse}{}
1347 \@ifclassloaded{revtex4-2}{\@FBwarning@capsepfalse}{}
1348 \@ifclassloaded{svjour3}{\@FBwarning@capsepfalse}{}
1349 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1350 \ifFB@koma \@FBwarning@capsepfalse \fi
```

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

```
1345 \@ifclassloaded{IEEEconf}{\@FBwarning@capsepfalse}{}
1346 \@ifclassloaded{IEEEtran}{\@FBwarning@capsepfalse}{}
1347 \@ifclassloaded{revtex4-2}{\@FBwarning@capsepfalse}{}
1348 \@ifclassloaded{svjour3}{\@FBwarning@capsepfalse}{}
1349 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1350 \ifFB@koma \@FBwarning@capsepfalse \fi
```

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)

```
1349 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1350 \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).

```
1351 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1352 \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’).

```

1353 \AtBeginDocument{%
1354   \ifx\@makecaption\STD@makecaption
1355     \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.

1356   \ifFBoldFigTabCaptions
1357   \else
1358     \def\FBCaption@Separator{\autospace@beforeFDP : }%
1359     \ifBFCustomiseFigTabCaptions
1360       \ifFB@mainlanguage@FR
1361         \def\FBCaption@Separator{\CaptionSeparator}%
1362       \fi
1363     \fi
1364   \fi
1365   \@FBwarning@capsepfalse
1366 \fi

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

1367   \@ifpackageloaded{caption}{\@FBwarning@capsepfalse}{}%
1368   \@ifpackageloaded{caption-light}{\@FBwarning@capsepfalse}{}%

Final warning if relevant:

1369   \if@FBwarning@capsep
1370     \FBWarning
1371       {Figures' and tables' captions might look like\MessageBreak
1372       `Figure 1:' in French instead of `Figure 1 :'.\MessageBreak
1373       If this happens, to fix this issue\MessageBreak
1374       switch to LuaLaTeX or XeLaTeX or\MessageBreak
1375       try to add \protect\usepackage{caption} or\MessageBreak
1376       ... leave it as it is; reported}%
1377   \fi
1378   \let\FB@makecaption\relax
1379   \let\STD@makecaption\relax
1380 }

```

2.9 Dots...

`\FBtextellipsis` Unless a ready-made character is available in the current font, LaTeX's default definition of `\textellipsis` 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 same way but without the last `\kern`.

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

```

1381 \ifFBunicode
1382 \else
1383   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1384   \DeclareTextCommand{\FBtextellipsis}{PU}{\9040\046}
1385   \DeclareTextCommand{\FBtextellipsis}{PD1}{\203}
1386   \DeclareTextCommandDefault{\FBtextellipsis}{%

```

```

1387     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}%
1388 \def\bbf@frenhdots{\babel@save\textellipsis
1389                 \let\textellipsis\FBtextellipsis}
1390 \addto\extrasfrench{\bbf@frenhdots}
1391 \fi

```

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).

```

1392 \ifFB@active@punct
1393   \ifpackageloaded{listings}
1394     {\AtBeginDocument{%
1395       \FBWarning{Please load the "listings" package\MessageBreak
1396                 AFTER babel/french; reported}}%
1397     }{}
1398 \fi

```

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

```

1399 \newif\if@FBwarning@natbib
1400 \ifFB@active@punct
1401   \ifpackageloaded{natbib}{}{\@FBwarning@natbibtrue}
1402 \fi
1403 \AtBeginDocument{%
1404   \if@FBwarning@natbib
1405     \ifpackageloaded{natbib}{}{\@FBwarning@natbibfalse}%
1406     \fi
1407   \if@FBwarning@natbib
1408     \FBWarning{Please load the "natbib" package\MessageBreak
1409               BEFORE babel/french; reported}%
1410   \fi
1411 }

```

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

```

1412 \newif\if@FBwarning@beamerarticle
1413 \ifpackageloaded{beamerarticle}{}{\@FBwarning@beamerarticlettrue}
1414 \AtBeginDocument{%
1415   \if@FBwarning@beamerarticle
1416     \ifpackageloaded{beamerarticle}{}%
1417     {\@FBwarning@beamerarticletfalse}%
1418   \fi
1419   \if@FBwarning@beamerarticle
1420     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1421               BEFORE babel/french; reported}%
1422   \fi
1423 }

```

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

```
1424 \newcommand*{\frenchsetup}[1]{%
1425   \setkeys{FB}{#1}%
1426 }%
1427 \@onlypreamble\frenchsetup
```

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

```
1428 \let\frenchbsetup\frenchsetup
1429 \@onlypreamble\frenchbsetup
```

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

```
1430 \newif\ifFBShowOptions
1431 \newif\ifFBStandardLayout           \FBStandardLayouttrue
1432 \newif\ifFBGlobalLayoutFrench      \FBGlobalLayoutFrenchtrue
1433 \newif\ifFBReduceListSpacing
1434 \newif\ifFBStandardListSpacing     \FBStandardListSpacingtrue
1435 \newif\ifFBListOldLayout
1436 \newif\ifFBListItemsAsPar
1437 \newif\ifFBCompactItemize
1438 \newif\ifFBStandardItemizeEnv      \FBStandardItemizeEnvtrue
1439 \newif\ifFBStandardEnumerateEnv    \FBStandardEnumerateEnvtrue
1440 \newif\ifFBStandardItemLabels      \FBStandardItemLabelstrue
1441 \newif\ifFBStandardLists           \FBStandardListstrue
1442 \newif\ifFBIndentFirst
1443 \newif\ifFBFrenchFootnotes
1444 \newif\ifFBAutoSpaceFootnotes
1445 \newif\ifFBOriginalTypewriter
1446 \newif\ifFBThinColonSpace
1447 \newif\ifFBThinSpaceInFrenchNumbers
1448 \newif\ifFBFrenchSuperscripts      \FBFrenchSuperscriptstrue
1449 \newif\ifFBLowercaseSuperscripts   \FBLowercaseSuperscriptstrue
1450 \newif\ifFBPartNameFull             \FBPartNameFulltrue
1451 \newif\ifFBCustomiseFigTabCaptions
1452 \newif\ifFBOldFigTabCaptions
1453 \newif\ifFBSmallCapsFigTabCaptions \FBSmallCapsFigTabCaptionstrue
1454 \newif\ifFBSuppressWarning
1455 \newif\ifFBINGuillSpace
```

The defaults values of these flags have been chosen so that `babel - french` does not change anything regarding the global layout. `\bbl@main@language`, set by the last

option of Babel, controls the global layout of the document. ‘AtEndOfPackage’ we check the main language in `\bbl@main@language`; if it is French (or a French dialect) the values of some flags have to be changed to ensure a French looking layout for the whole document (even in parts written in languages other than French); the end-user will then be able to customise the values of all these flags with `\frenchsetup{}`. The following patch is for koma-script classes: the `\partformat` command, defined as `\partname~\thepart\autodot`, is incompatible with our redefinition of `\partname`.

```

1456 \ifFB@koma
1457   \ifdefined\partformat
1458     \def\FB@partformat@fix{%
1459       \ifFBPartNameFull
1460         \babel@save\partformat
1461         \renewcommand*{\partformat}{\partname}%
1462       \fi}
1463   \addto\extrasfrench{\FB@partformat@fix}%
1464 \fi
1465 \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.

```

1466 \def\FB@french{french}
1467 \def\FB@acadian{acadian}
1468 \newif\ifFB@mainlanguage@FR
1469 \AtEndOfPackage{%
1470   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1471   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1472   \fi
1473   \ifFB@mainlanguage@FR
1474     \FBGlobalLayoutFrenchtrue
1475     \@ifclassloaded{beamer}%
1476       {\PackageInfo{french.ldf}{%
1477         No list customisation for the beamer class,%
1478         \MessageBreak reported}}%
1479     {\@ifpackageloaded{beamerarticle}%
1480      {\FBStandardItemLabelsfalse
1481       \FBStandardListSpacingfalse
1482       \PackageInfo{french.ldf}{%
1483         Minimal list customisation for the beamerarticle%
1484         \MessageBreak package; reported}}%

```

Otherwise customise lists “à la française”:

```

1485     {\FBStandardListSpacingfalse
1486     \FBStandardItemizeEnvfalse
1487     \FBStandardEnumerateEnvfalse
1488     \FBStandardItemLabelsfalse}%
1489   }
1490 \FBIndentFirsttrue
1491 \FBFrenchFootnotesttrue
1492 \FBAutoSpaceFootnotesttrue

```

```

1493   \FBCustomiseFigTabCaptionstrue
1494   \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.

```

1495   \RequirePackage{keyval}%
1496   \define@key{FB}{ShowOptions}[true]%
1497       {\csname FBShowOptions#1\endcsname}%

```

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

```

1498   \define@key{FB}{StandardLayout}[true]%
1499       {\ifFB@mainlanguage@FR
1500         \csname FBStandardLayout#1\endcsname
1501       \else
1502         \PackageWarning{french.ldf}%
1503           {Option `StandardLayout' skipped:\MessageBreak
1504             French is *not* babel's last option.\MessageBreak
1505             Reported}%
1506       \fi
1507   \ifFBStandardLayout
1508     \FBStandardListSpacingtrue
1509     \FBStandardItemizeEnvtrue
1510     \FBStandardItemLabelstrue
1511     \FBStandardEnumerateEnvtrue
1512     \FBIndentFirstfalse
1513     \FBFrenchFootnotesfalse
1514     \FBAutoSpaceFootnotesfalse
1515   \else
1516     \FBStandardListSpacingfalse
1517     \FBStandardItemizeEnvfalse
1518     \FBStandardItemLabelsfalse
1519     \FBStandardEnumerateEnvfalse
1520     \FBIndentFirsttrue
1521     \FBFrenchFootnotesttrue
1522     \FBAutoSpaceFootnotesttrue
1523   \fi}%
1524   \define@key{FB}{GlobalLayoutFrench}[true]%
1525       {\ifFB@mainlanguage@FR
1526         \csname FBGlobalLayoutFrench#1\endcsname
1527       \else
1528         \PackageWarning{french.ldf}%
1529           {Option `GlobalLayoutFrench' skipped:\MessageBreak
1530             French is *not* babel's last option.\MessageBreak
1531             Reported}%
1532       \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 at every language's switch.

```

1533   \define@key{FB}{ReduceListSpacing}[true]%
1534       {\csname FBReduceListSpacing#1\endcsname
1535     \ifFBReduceListSpacing \FBStandardListSpacingfalse
1536     \else \FBStandardListSpacingtrue\fi

```



```

1537     }%
1538 \define@key{FB}{StandardListSpacing}[true]%
1539     {\csname FBStandardListSpacing#1\endcsname}%
1540 \define@key{FB}{ListOldLayout}[true]%
1541     {\csname FBListOldLayout#1\endcsname
1542     \ifFBListOldLayout
1543     \FBStandardEnumerateEnvtrue
1544     \renewcommand*{\FrenchLabelItem}{\textendash}%
1545     \fi}%
1546 \define@key{FB}{CompactItemize}[true]%
1547     {\csname FBCompactItemize#1\endcsname
1548     \ifFBCompactItemize
1549     \FBStandardItemizeEnvfalse
1550     \FBStandardEnumerateEnvfalse
1551     \else
1552     \FBStandardItemizeEnvtrue
1553     \FBStandardEnumerateEnvtrue
1554     \fi}%
1555 \define@key{FB}{StandardItemizeEnv}[true]%
1556     {\csname FBStandardItemizeEnv#1\endcsname}%
1557 \define@key{FB}{StandardEnumerateEnv}[true]%
1558     {\csname FBStandardEnumerateEnv#1\endcsname}%
1559 \define@key{FB}{StandardItemLabels}[true]%
1560     {\csname FBStandardItemLabels#1\endcsname}%
1561 \define@key{FB}{ItemLabels}%
1562     {\renewcommand*{\FrenchLabelItem}{#1}}%
1563 \define@key{FB}{ItemLabeli}%
1564     {\renewcommand*{\Frlabelitemi}{#1}}%
1565 \define@key{FB}{ItemLabelii}%
1566     {\renewcommand*{\Frlabelitemii}{#1}}%
1567 \define@key{FB}{ItemLabeliii}%
1568     {\renewcommand*{\Frlabelitemiii}{#1}}%
1569 \define@key{FB}{ItemLabeliv}%
1570     {\renewcommand*{\Frlabelitemiv}{#1}}%
1571 \define@key{FB}{StandardLists}[true]%
1572     {\csname FBStandardLists#1\endcsname
1573     \ifFBStandardLists
1574     \FBStandardListSpacingtrue
1575     \FBStandardItemizeEnvtrue
1576     \FBStandardEnumerateEnvtrue
1577     \FBStandardItemLabelstrue
1578     \else
1579     \FBStandardListSpacingfalse
1580     \FBStandardItemizeEnvfalse
1581     \FBStandardEnumerateEnvfalse
1582     \FBStandardItemLabelsfalse
1583     \fi}%
1584 \define@key{FB}{ListItemsAsPar}[true]%
1585     {\csname FBListItemsAsPar#1\endcsname}
1586 \define@key{FB}{IndentFirst}[true]%
1587     {\csname FBIndentFirst#1\endcsname}%
1588 \define@key{FB}{FrenchFootnotes}[true]%
1589     {\csname FBFrenchFootnotes#1\endcsname}%

```

```

1590 \define@key{FB}{AutoSpaceFootnotes}[true]%
1591     {\csname FBAutoSpaceFootnotes#1\endcsname}%
1592 \define@key{FB}{AutoSpacePunctuation}[true]%
1593     {\csname FBAutoSpacePunctuation#1\endcsname}%
1594 \define@key{FB}{OriginalTypewriter}[true]%
1595     {\csname FBOriginalTypewriter#1\endcsname}%
1596 \define@key{FB}{ThinColonSpace}[true]%
1597     {\csname FBThinColonSpace#1\endcsname
1598     \ifFBThinColonSpace
1599     \renewcommand*{\FBcolonspace}{\FBthinspace}%
1600     \fi}%
1601 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1602     {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1603 \define@key{FB}{FrenchSuperscripts}[true]%
1604     {\csname FBFrenchSuperscripts#1\endcsname}%
1605 \define@key{FB}{LowercaseSuperscripts}[true]%
1606     {\csname FBLowercaseSuperscripts#1\endcsname}%
1607 \define@key{FB}{PartNameFull}[true]%
1608     {\csname FBPartNameFull#1\endcsname}%
1609 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1610     {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1611 \define@key{FB}{OldFigTabCaptions}[true]%
1612     {\csname FBOldFigTabCaptions#1\endcsname
1613     \ifFBOldFigTabCaptions
1614     \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1615     \def\FBCaption@Separator{\CaptionSeparator}}%
1616     \addto\extrasfrench{\FB@capsep@fix}%
1617     \ifdefined\extrasacadian
1618     \addto\extrasacadian{\FB@capsep@fix}%
1619     \fi
1620     \fi}%
1621 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1622     {\csname FBSmallCapsFigTabCaptions#1\endcsname
1623     \ifFBSmallCapsFigTabCaptions
1624     \else \let\FBfigtabshape\relax \fi}%
1625 \define@key{FB}{SuppressWarning}[true]%
1626     {\csname FBSuppressWarning#1\endcsname
1627     \ifFBSuppressWarning
1628     \renewcommand{\FBWarning}[1]{}%
1629     \fi}%

```

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

```

1630 \define@key{FB}{INGuillSpace}[true]%
1631     {\csname FBINGuillSpace#1\endcsname
1632     \ifFBINGuillSpace
1633     \renewcommand*{\FBguillspace}{\space}%
1634     \fi}%
1635 \define@key{FB}{InnerGuillSingle}[true]%
1636     {\csname FBInnerGuillSingle#1\endcsname}%
1637 \define@key{FB}{EveryParGuill}[open]%
1638     {\expandafter\let\expandafter
1639     \FBeveryparguill\csname FBguill#1\endcsname
1640     \ifx\FBeveryparguill\FBguillopen

```

```

1641         \else\ifx\FBeveryparguill\FBguillclose
1642             \else\ifx\FBeveryparguill\FBguillnone
1643                 \else
1644                     \let\FBeveryparguill\FBguillopen
1645                     \FBWarning{Wrong value for `EveryParGuill':
1646                         try `open',\MessageBreak
1647                         `close' or `none'. Reported}%
1648                 \fi
1649             \fi
1650         \fi}%
1651 \define@key{FB}{EveryLineGuill}[open]%
1652     {\ifFB@luatex@punct
1653         \expandafter\let\expandafter
1654             \FBeverylineguill\csname FBguill#1\endcsname
1655         \ifx\FBeverylineguill\FBguillopen
1656         \else\ifx\FBeverylineguill\FBguillclose
1657             \else\ifx\FBeverylineguill\FBguillnone
1658                 \else
1659                     \let\FBeverylineguill\FBguillnone
1660                     \FBWarning{Wrong value for `EveryLineGuill':
1661                         try `open',\MessageBreak
1662                         `close' or `none'. Reported}%
1663                 \fi
1664             \fi
1665         \fi
1666         \else
1667             \FBWarning{Option `EveryLineGuill' skipped:%
1668                 \MessageBreak this option is for
1669                 LuaTeX *only*.\MessageBreak Reported}%
1670         \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.

```

1671 \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1672     {\ifFB@luatex@punct
1673         \csname FBucsNBSP#1\endcsname
1674         \ifFBucsNBSP \FB@ucsNBSP=\@ne \fi
1675     \else
1676         \FBWarning{Option `UnicodeNoBreakSpaces' skipped:%
1677             \MessageBreak this option is for
1678             LuaTeX *only*.\MessageBreak Reported}%
1679     \fi
1680 }%

```

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

With pdfTeX (or old LuaTeX and XeTeX engines), quote characters are made active and expand to \og\ignorespaces and {\fg} respectively if the current language is French, and to \guillemotleft and \guillemotright otherwise (think of German quotes), this is done by \FB@@og and \FB@@fg; thus correct non-breaking spaces will

be added automatically to French quotes. The quote characters typed in depend on the input encoding, it can be single-byte (latin1, latin9, applemac,...) or multi-bytes (utf-8, utf8x); the next command is meant for checking whether a character is single-byte (`\FB@second` is empty) or not.

```
1681 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
1682 \define@key{FB}{og}%
1683     {\ifFBunicode
```

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

```
1684     \ifFB@luatex@punct
1685         \FB@addGUIILspace=1 \relax
1686     \fi
```

then with XeTeX it is a bit more tricky:

```
1687     \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).

```
1688         \XeTeXcharclass"13 = \FB@guilo
1689         \XeTeXcharclass"AB = \FB@guilo
1690         \XeTeXcharclass"A0 = \FB@guilnul
1691         \XeTeXcharclass"202F = \FB@guilnul
1692     \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1693     \ifFB@active@punct
1694         \FBWarning{Option og« not supported with this version
1695                 of\MessageBreak LuaTeX/XeTeX; reported}%
1696     \fi
1697 \else
```

This is for conventional TeX engines:

```
1698     \newcommand*{\FB@@og}{%
1699         \ifFBfrench
1700             \ifFB@spacing\FB@og\ignorespaces
1701             \else\guillemotleft
1702             \fi
1703         \else\guillemotleft\fi}%
1704 \AtBeginDocument{%
1705     \ifdefined\uc@dclc
```

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

```
1706         \uc@dclc{171}{default}{\FB@@og}%
1707     \else
```

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

```
1708         \FB@parse#1\endparse
1709         \ifx\FB@second\@empty
```

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

```
1710         \ifdefined\mule@def
1711         \mule@def{11}{\FB@@og}%
```

```

1712         \else
1713         \ifdefined\DeclareInputText
1714         \@tempcnta`#1\relax
1715         \DeclareInputText{\the\@tempcnta}{\FB@og}%
1716         \else
Package inputenc not loaded, no way...
1717         \FBWarning{Option `og' requires package
1718         inputenc;\MessageBreak reported}%
1719         \fi
1720         \fi
1721         \else
This means multi-byte character encoding, we assume UTF-8
1722         \DeclareUnicodeCharacter{00AB}{\FB@og}%
1723         \fi
1724         \fi}%
1725     \fi
1726 }%

```

Same code for the closing quote.

```

1727 \define@key{FB}{fg}%
1728     {\ifFBunicode
1729     \ifFB@luatex@punct
1730     \FB@addGUILspace=1 \relax
1731     \fi
1732     \ifFB@xetex@punct
1733     \XeTeXcharclass"14 = \FB@guilf
1734     \XeTeXcharclass"BB = \FB@guilf
1735     \XeTeXcharclass"A0 = \FB@guilnul
1736     \XeTeXcharclass"202F = \FB@guilnul
1737     \fi
1738     \ifFB@active@punct
1739     \FBWarning{Option fg=> not supported with this version
1740     of\MessageBreak LuaTeX/XeTeX; reported}%
1741     \fi
1742     \else
1743     \newcommand*{\FB@fg}{%
1744     \ifFBfrench
1745     \ifFB@spacing\FB@fg
1746     \else\guillemotright
1747     \fi
1748     \else\guillemotright\fi}%
1749     \AtBeginDocument{%
1750     \ifdefined\uc@dcl
1751     \uc@dcl{187}{default}{\FB@fg}%
1752     \else
1753     \FB@parse#1\endparse
1754     \ifx\FB@second\@empty
1755     \ifdefined\mule@def
1756     \mule@def{27}{\FB@fg}%
1757     \else
1758     \ifdefined\DeclareInputText
1759     \@tempcnta`#1\relax
1760     \DeclareInputText{\the\@tempcnta}{\FB@fg}%

```

```

1761             \else
1762             \FBWarning{Option `fg' requires package
1763             inputenc;\MessageBreak reported}%
1764             \fi
1765             \fi
1766             \else
1767             \DeclareUnicodeCharacter{00BB}{\FB@fg}%
1768             \fi
1769             \fi}%
1770     \fi
1771     }%
1772 }

```

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

```

1773 \newcommand*{\FBprocess@options}{%

```

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

```

1774 \@ifpackageloaded{enumitem}{%
1775   \ifFBStandardItemizeEnv
1776   \else
1777     \FBStandardItemizeEnvtrue
1778     \PackageInfo{french.ldf}%
1779     {Setting StandardItemizeEnv=true for\MessageBreak
1780     compatibility with enumitem package,\MessageBreak
1781     reported}%
1782   \fi
1783   \ifFBStandardEnumerateEnv
1784   \else
1785     \FBStandardEnumerateEnvtrue
1786     \PackageInfo{french.ldf}%
1787     {Setting StandardEnumerateEnv=true for\MessageBreak
1788     compatibility with enumitem package,\MessageBreak
1789     reported}%
1790   \fi}{}%
1791 \@ifpackageloaded{paralist}{%
1792   \ifFBStandardItemizeEnv
1793   \else
1794     \FBStandardItemizeEnvtrue
1795     \PackageInfo{french.ldf}%
1796     {Setting StandardItemizeEnv=true for\MessageBreak
1797     compatibility with paralist package,\MessageBreak
1798     reported}%
1799   \fi
1800   \ifFBStandardEnumerateEnv
1801   \else
1802     \FBStandardEnumerateEnvtrue
1803     \PackageInfo{french.ldf}%

```

```

1804         {Setting StandardEnumerateEnv=true for\MessageBreak
1805         compatibility with paralist package,\MessageBreak
1806         reported}%
1807     \fi}{}%
1808 \@ifpackageloaded{enumerate}{%
1809     \ifFBStandardEnumerateEnv
1810     \else
1811         \FBStandardEnumerateEnvtrue
1812         \PackageInfo{french.ldf}%
1813         {Setting StandardEnumerateEnv=true for\MessageBreak
1814         compatibility with enumerate package,\MessageBreak
1815         reported}%
1816     \fi}{}%

```

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

```

1817 \def\FB@ufl{\update@frenchlists}
1818 \ifFB@mainlanguage@FR
1819     \update@frenchlists
1820 \else
1821     \ifFBStandardItemizeEnv
1822     \else
1823         \PackageWarning{french.ldf}%
1824         {babel-french will not customize lists' layout\MessageBreak
1825         when French is not the main language,\MessageBreak
1826         reported}%
1827     \fi
1828 \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.

```

1829 \ifBBAutoSpacePunctuation
1830     \autospace@beforeFDP
1831 \else
1832     \noautospace@beforeFDP
1833 \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.

```

1834 \ifFBOriginalTypewriter
1835 \else
1836     \let\ttfamilyORI\ttfamily
1837     \let\rmfamilyORI\rmfamily
1838     \let\sffamilyORI\sffamily
1839     \let\ttfamily\ttfamilyFB
1840     \let\rmfamily\rmfamilyFB
1841     \let\sffamily\sffamilyFB
1842 \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.

```

1843 \ifpackageloaded{numprint}%
1844   {\ifnprt@autolanguage
1845     \providecommand*\npstylefrench}{}%
1846     \ifFBThinSpaceInFrenchNumbers
1847       \renewcommand*\FBthousandsep{\,}%
1848     \fi
1849     \g@addto@macro\npstylefrench{\npthousandsep\FBthousandsep}}%
1850   \fi
1851   }{}%
```

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.

```

1852 \ifFBFrenchSuperscripts
1853   \DeclareRobustCommand*\up}{%
1854     \texorpdfstring{\ifstar\FB@up@fake}\fup}}{}%
1855   }
1856 \else
1857   \DeclareRobustCommand*\up}{%
1858     \texorpdfstring{\ifstar\FB@up@fake}\textsuperscript}}{}%
1859   }
1860 \fi
```

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

```

1861 \ifFBLowercaseSuperscripts
1862 \else
1863   \renewcommand*\FB@lc}[1]{##1}%
1864 \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`.

```

1865 \ifFB@koma
1866   \ifx\captionformat\FB@std@capsep
1867     \ifFBCustomiseFigTabCaptions
1868       \renewcommand*\captionformat{\CaptionSeparator}%
1869     \else
1870       \renewcommand*\captionformat{{\autospace@beforeFDP : \ }}%
1871     \fi
1872   \fi
1873 \fi
1874 \ifclassloaded{memoir}%
1875   {\ifx\@contdelim\FB@std@capsep
1876     \ifFBCustomiseFigTabCaptions
1877       \captiondelim{\CaptionSeparator}%
1878     \else
1879       \captiondelim{{\autospace@beforeFDP : }}%
1880     \fi
```



```

1881     \fi}}}%
1882 \@ifclassloaded{beamer}%
1883   {\protected@edef\FB@capsep{%
1884     \csname beamer@@tpl@caption label separator\endcsname}%
1885   \ifx\FB@capsep\FB@std@capsep
1886     \ifFBCustomiseFigTabCaptions
1887       \defbeamertemplate{caption label separator}{FBcustom}{%
1888         \CaptionSeparator}%
1889       \setbeamertemplate{caption label separator}[FBcustom]%
1890     \else
1891       \defbeamertemplate{caption label separator}{FBcolon}{%
1892         {\autospace@beforeFDP : }}%
1893       \setbeamertemplate{caption label separator}[FBcolon]%
1894     \fi
1895   \fi}}}%

```

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

```

1896 \ifFBShowOptions
1897   \GenericWarning{* }{%
1898     *** List of possible options for babel-french ***\MessageBreak
1899     [Default values between brackets when french is loaded *LAST*]%
1900     \MessageBreak
1901     ShowOptions [false]\MessageBreak
1902     StandardLayout [false]\MessageBreak
1903     GlobalLayoutFrench [true]\MessageBreak
1904     PartNameFull [true]\MessageBreak
1905     IndentFirst [true]\MessageBreak
1906     ListItemsAsPar [false]\MessageBreak
1907     StandardListSpacing [false]\MessageBreak
1908     StandardItemizeEnv [false]\MessageBreak
1909     StandardEnumerateEnv [false]\MessageBreak
1910     StandardItemLabels [false]\MessageBreak
1911     ItemLabels=\textendash, \textbullet,
1912     \protect\ding{43},... [\textendash]\MessageBreak
1913     ItemLabeli=\textendash, \textbullet,
1914     \protect\ding{43},... [\textendash]\MessageBreak
1915     ItemLabelii=\textendash, \textbullet,
1916     \protect\ding{43},... [\textendash]\MessageBreak
1917     ItemLabeliii=\textendash, \textbullet,
1918     \protect\ding{43},... [\textendash]\MessageBreak
1919     ItemLabeliv=\textendash, \textbullet,
1920     \protect\ding{43},... [\textendash]\MessageBreak
1921     StandardLists [false]\MessageBreak
1922     ListOldLayout [false]\MessageBreak
1923     FrenchFootnotes [true]\MessageBreak
1924     AutoSpaceFootnotes [true]\MessageBreak
1925     AutoSpacePunctuation [true]\MessageBreak
1926     ThinColonSpace [false]\MessageBreak
1927     OriginalTypewriter [false]\MessageBreak
1928     UnicodeNoBreakSpaces [false]\MessageBreak
1929     og= <left quote character>, fg= <right quote character>%
1930     INGuillSpace [false]\MessageBreak
1931     EveryParGuill=open, close, none [open]\MessageBreak
1932     EveryLineGuill=open, close, none

```

```

1933             [open in LuaTeX, none otherwise]\MessageBreak
1934     InnerGuillSingle [false]\MessageBreak
1935     ThinSpaceInFrenchNumbers [false]\MessageBreak
1936     SmallCapsFigTabCaptions [true]\MessageBreak
1937     CustomiseFigTabCaptions [true]\MessageBreak
1938     OldFigTabCaptions [false]\MessageBreak
1939     FrenchSuperscripts [true]\MessageBreak
1940     LowercaseSuperscripts [true]\MessageBreak
1941     SuppressWarning [false]\MessageBreak
1942     \MessageBreak
1943     *****%
1944     \MessageBreak\protect\frenchsetup{ShowOptions}}
1945   \fi
1946 }

```

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.

```

1947 \AtBeginDocument{%
1948   \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`.

```

1949   \FBprocess@options

```

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

```

1950   \ifBucsNBSP
1951     \renewcommand*\FBmedkern{\char"202F\relax}%
1952     \renewcommand*\FBthickkern{\char"A0\relax}%
1953     \ifFBThinSpaceInFrenchNumbers
1954       \renewcommand*\FBthousandsep{\char"202F\relax}%
1955     \else
1956       \renewcommand*\FBthousandsep{\char"A0\relax}%
1957     \fi
1958   \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*`!

```

1959   \begingroup
1960     \newcommand{\FBOTone}{OT1}%
1961     \ifx\encodingdefault\FBOTone
1962       \FBWarning{OT1 encoding should not be used for French.%
1963         \MessageBreak
1964         Add \protect\usepackage[T1]{fontenc} to the
1965         preamble\MessageBreak of your document; reported}%
1966     \fi
1967   \endgroup
1968 }

```

2.12 French lists

`\listFB` Vertical spacing in lists should be shorter in French texts than the defaults provided
`\listORI` by LaTeX. Note that the easy way, just changing values of vertical spacing parameters
`\FB@listVsettings` when entering French and restoring them to their defaults on exit would not work;
 so we define the command `\FB@listVsettings` to hold the settings to be used by
 the French variant `\listFB` of `\list`. Note that switching to `\listFB` reduces vertical
 spacing in *all* environments built on `\list`: `itemize`, `enumerate`, `description`, but
 also `abstract`, `quotation`, `quote` and `verse`...

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

```
1969 \let\listORI\list
1970 \let\endlistORI\endlist
1971 \newdimen\FB@parskip
1972 \def\FB@listVsettings{%
1973     \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1974     \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
`\FB@parskip`.

```
1975     \FB@parskip=\parskip
1976     \addtolength{\topsep}{-\FB@parskip}%
1977     \addtolength{\partopsep}{\FB@parskip}%
1978     \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1979     \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
```

(v3.5q) If `\parskip` is not null, `\parsep` is set to `\parskip`, so paragraphs inside items
 will be preceded by the same vertical space as paragraphs located outside lists; the
 vertical skip before items (`\itemsep + \parsep`) doesn't need to be enlarged.

```
1980     \ifdim\FB@parskip>0pt
1981         \setlength{\parsep}{\FB@parskip}%
1982         \addtolength{\itemsep}{-\FB@parskip}%
1983     \fi
1984 }
1985 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1986 \let\endlistFB\endlistORI
```

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. 5.

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

```
\Frlabelitemi 1987 \newcommand*{\FrenchLabelItem}{\textemdash}
\Frlabelitemii 1988 \newcommand*{\Frlabelitemi}{\FrenchLabelItem}
\Frlabelitemiii 1989 \newcommand*{\Frlabelitemii}{\FrenchLabelItem}
\Frlabelitemiv 1990 \newcommand*{\Frlabelitemiii}{\FrenchLabelItem}
1991 \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`
`\labelwidthFB` They will get reasonable defaults later when entering French (see `\setlabelitemsFB` and `\setlistindentFB`) unless they have been customised.

```
1992 \newdimen\listindentFB
1993 \setlength{\listindentFB}{-1pt}
1994 \newdimen\descindentFB
1995 \setlength{\descindentFB}{-1pt}
1996 \newdimen\labelindentFB
1997 \setlength{\labelindentFB}{-1pt}
1998 \newdimen\labelwidthFB
1999 \setlength{\labelwidthFB}{-1pt}
```

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

```
2000 \newdimen\leftmarginFB
2001 \def\FB@listHsettings{%
2002   \ifFBListItemsAsPar
```

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

```
2003   \itemindent=\labelindentFB
2004   \advance\itemindent by \labelwidthFB
2005   \advance\itemindent by \labelsep
2006   \leftmargini\z@
2007   \bbl@for\FB@dp {2, 3, 4, 5, 6}%
2008     {\csname leftmargin\romannumeral\FB@dp\endcsname =
2009       \labelindentFB}%
2010   \else
```

Default layout: labels hanging into the list left margin.

```
2011   \leftmarginFB=\labelwidthFB
2012   \advance\leftmarginFB by \labelsep
2013   \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
2014     {\csname leftmargin\romannumeral\FB@dp\endcsname =
2015       \leftmarginFB}%
2016   \advance\leftmargini by \listindentFB
```

(v3.5q) Same 'parindent' for paragraphs in lists' items (was null as in standard lists).

```
2017   \listparindent=\parindent
2018   \fi
2019   \leftmargin=\csname leftmargin%
2020     \ifnum \@listdepth=\@ne i\else ii\fi\endcsname
2021 }
```

`\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 v2.5k).

```

2022 \def\FB@itemizesettings{%
2023   \ifFBStandardListSpacing
2024   \else
2025     \setlength{\topsep}{\z@}%
2026     \setlength{\partopsep}{\z@}%
2027     \FB@parskip=\parskip
2028     \addtolength{\topsep}{-\FB@parskip}%
2029     \addtolength{\partopsep}{\FB@parskip}%
2030     \setlength{\itemsep}{\z@}%
2031     \setlength{\parsep}{\z@}%
2032     \ifdim\FB@parskip>0pt
2033       \setlength{\parsep}{\FB@parskip}%
2034       \addtolength{\itemsep}{-\FB@parskip}%
2035     \fi
2036   \fi
2037   \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
2038   \ifFBListOldLayout
2039     \setlength{\leftmargin}{\labelwidth}%
2040     \addtolength{\leftmargin}{\labelsep}%
2041     \addtolength{\leftmargin}{\parindent}%
2042   \else
2043     \FB@listHsettings
2044   \fi
2045 }

```

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

```

2046 \def\itemizeFB{%
2047   \ifnum \@itemdepth >\thr@@\@toodeep\else
2048     \advance\@itemdepth by \@ne
2049     \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
2050     \expandafter
2051     \listORI
2052     \csname\@itemitem\endcsname
2053     \FB@itemizesettings
2054   \fi
2055 }
2056 \let\enditemizeFB\endlistORI

2057 \def\setlabelitemsFB{%
2058   \let\labelitemi\Frlabelitemi
2059   \let\labelitemii\Frlabelitemii
2060   \let\labelitemiii\Frlabelitemiii
2061   \let\labelitemiv\Frlabelitemiv
2062   \ifdim\labelwidthFB<\z@
2063     \settowidth{\labelwidthFB}{\FrenchLabelItem}%
2064   \fi
2065 }

```

```

2066 \def\setlistindentFB{%
2067   \ifdim\labelindentFB<\z@
2068     \ifdim\parindent=\z@
2069       \setlength{\labelindentFB}{1.5em}%
2070     \else
2071       \setlength{\labelindentFB}{\parindent}%
2072     \fi
2073 \fi
2074 \ifdim\listindentFB<\z@
2075   \ifdim\parindent=\z@
2076     \setlength{\listindentFB}{1.5em}%
2077   \else
2078     \setlength{\listindentFB}{\parindent}%
2079   \fi
2080 \fi
2081 \ifdim\descindentFB<\z@
2082   \ifFBListItemsAsPar
2083     \setlength{\descindentFB}{\labelindentFB}%
2084   \else
2085     \setlength{\descindentFB}{\listindentFB}%
2086   \fi
2087 \fi
2088 }

```

`\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`.

```

2089 \def\enumerateFB{%
2090   \ifnum \@enumdepth >\thr@@\@toodeep\else
2091     \advance\@enumdepth by \@ne
2092     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2093     \expandafter
2094     \list
2095       \csname label\@enumctr\endcsname
2096       {\FB@listHsettings
2097         \usecounter\@enumctr\def\makelabel##1{\hss\llap{##1}}}%
2098   \fi
2099 }
2100 \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.

```

2101 \def\descriptionFB{%
2102   \list{}{\FB@listHsettings
2103     \labelwidth=\z@

```

```

2104         \ifFBListItemsAsPar
2105         \itemindent=\descindentFB
2106     \else
2107         \itemindent=-\leftmargin
2108         \ifnum\@listdepth=\@ne
2109             \ifdim\descindentFB=\z@
2110                 \ifdim\listindentFB>\z@
2111                     \leftmargini=\listindentFB
2112                     \leftmargin=\leftmargini
2113                     \itemindent=-\leftmargin
2114                 \fi
2115             \else
2116                 \advance\itemindent by \descindentFB
2117             \fi
2118         \fi
2119     \fi
2120     \let\makelabel\descriptionlabel}%
2121 }
2122 \let\enddescriptionFB\endlistORI

```

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

```

2123 \def\update@frenchlists{%
2124   \setlistindentFB
2125   \ifFBStandardListSpacing
2126   \else \let\list\listFB \fi
2127   \ifFBStandardItemizeEnv
2128   \else \let\itemize\itemizeFB \fi
2129   \ifFBStandardItemLabels
2130   \else \setlabelitemsFB \fi
2131   \ifFBStandardEnumerateEnv
2132   \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2133 }

```

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

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

Lists' layout changes at language switches only if `GlobalLayoutFrench=false`.

```

2134 \def\FB@ufl{\relax}
2135 \def\bbl@frenchlistlayout{%
2136   \ifFBGlobalLayoutFrench
2137   \else
2138     \babel@save\list           \babel@save\itemize
2139     \babel@save\enumerate     \babel@save\description
2140     \babel@save\labelitemi    \babel@save\labelitemii
2141     \babel@save\labelitemiii  \babel@save\labelitemiv
2142     \FB@ufl

```

```

2143 \fi
2144 }
2145 \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`. Indentation changes at language switches in only two cases:

- a) `GlobalLayoutFrench=false`,
- b) `IndentFirst=true` and French isn't the main language.

```

2146 \def\bbl@frenchindent{%
2147 \ifFBGlobalLayoutFrench\else\babel@save\@afterindentfalse\fi
2148 \ifBBIIndentFirst
2149 \ifFB@mainlanguage@FR\else\babel@save\@afterindentfalse\fi
2150 \let\@afterindentfalse\@afterindenttrue
2151 \@afterindenttrue
2152 \fi}
2153 \addto\extrasfrench{\bbl@frenchindent}

```

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`.

```

2154 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2155 \PackageInfo{french.ldb}%
2156 {bigfoot package in use.\MessageBreak
2157 babel-french will NOT customise footnotes;%
2158 \MessageBreak reported}}%
2159 {\let\@footnotemarkORI\@footnotemark
2160 \def\@footnotemarkFB{\leavevmode\unskip\unkern
2161 \,\@footnotemarkORI}%
2162 \ifFBAutoSpaceFootnotes
2163 \let\@footnotemark\@footnotemarkFB
2164 \fi}%
2165 }

```

`\@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).

```
2166 \newdimen\parindentFFN
2167 \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.

```
2168 \newcommand*\dotFFN{.}
2169 \newcommand*\kernFFN{\kern .5em}
2170 \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.

```
2171 \ifFB@koma
2172 \let\@makefntextORI\@makefntext
2173 \let\@@makefnmarkORI\@@makefnmark
```

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

```
2174 \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
2175         {\thefootnotemark\dotFFN\kernFFN}
2176 \let\@makefntextFB\@makefntext
2177 \let\@@makefnmarkFB\@@makefnmark
```

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

```
2178 \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2179         {\textsuperscript{\thefootnotemark}}
2180 \let\@makefntextTH\@makefntext
2181 \let\@@makefnmarkTH\@@makefnmark
```

Restore the original definitions.

```
2182 \let\@makefntext\@makefntextORI
2183 \let\@@makefnmark\@@makefnmarkORI
2184 \fi
```

Definitions for the memoir class:

```
2185 \ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2186 {\newcommand{\@makefntextFB}[1]{%
2187     \def\footscript##1{##1\dotFFN\kernFFN}%
2188     \setlength{\footmarkwidth}{\FBfnindent}%
2189     \setlength{\footmarksep}{-\footmarkwidth}%
2190     \setlength{\footparindent}{\parindentFFN}%
```

```

2191     \makefootmark #1}%
2192   }{}

```

Definitions for the beamer class:

```
2193 \@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.

```

2194   {\def\@makefntextFB#1{%
2195     \def\insertfootnotetext{#1}%
2196     \def\insertfootnotemark{\insertfootnotemarkFB}%
2197     \usebeamertemplate***{footnote}}%
2198   \def\insertfootnotemarkFB{%
2199     \usebeamercolor[fg]{footnote mark}%
2200     \usebeamerfont*{footnote mark}%
2201     \llap{\@thefnmark}\dotFFN\kernFFN}%
2202   }{}

```

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)!

```

2203 \providecommand*\insertfootnotemarkFB{%
2204   \parindent=\parindentFFN
2205   \rule\z@\footnotesep
2206   \setbox\@tempboxa\hbox{\@thefnmark}%
2207   \ifdim\wd\@tempboxa>\z@
2208     \llap{\@thefnmark}\dotFFN\kernFFN
2209   \fi}
2210 \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.

```

2211 \providecommand\localleftbox[1]{}
2212 \AtBeginDocument{%
2213   \@ifpackageloaded{bigfoot}{}%
2214   {\ifdim\parindentFFN<10in
2215     \else
2216       \parindentFFN=\parindent
2217       \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2218     \fi
2219     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2220     \addtolength{\FBfnindent}{\parindentFFN}%
2221     \let\@makefntextORI\@makefntext
2222     \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).

```

2223     \let\@makefnmarkORI\@makefnmark
2224     \long\def\@makefntext#1{%
2225         \lcalleftbox{}}%
2226     \let\FBeverypar@save\FBeverypar@quote
2227     \let\FBeverypar@quote\relax
2228     \ifFBFrenchFootnotes
2229         \ifx\footnote\thanks
2230             \let\@makefnmark\@makefnmarkTH
2231             \@makefntextTH{#1}
2232         \else
2233             \let\@makefnmark\@makefnmarkFB
2234             \@makefntextFB{#1}
2235         \fi
2236     \else
2237         \let\@makefnmark\@makefnmarkORI
2238         \@makefntextORI{#1}%
2239     \fi
2240     \let\FBeverypar@quote\FBeverypar@save
2241     \lcalleftbox{\FBeveryline@quote}}%
2242 \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.

```

2243     \@ifclassloaded{memoir}%
2244     {\ifFBFrenchFootnotes
2245         \setlength{\thanksmarkwidth}{\parindentFFN}%
2246         \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2247         \fi
2248     }}%

```

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

```

2249     \@ifclassloaded{beamer}%
2250     {\ifFBFrenchFootnotes
2251         \ifdim\parindentFFN=1.5em\else
2252             \FBWarning{%
2253                 \protect\parindentFFN\space is ineffective%
2254                 \MessageBreak within the beamer class.%
2255                 \MessageBreak Reported}%
2256             \fi
2257         \fi
2258     }}%

```

Definition of \@makefntext for all other classes:

```

2259     \long\def\@makefntext#1{%
2260         \lcalleftbox{}}%
2261     \let\FBeverypar@save\FBeverypar@quote
2262     \let\FBeverypar@quote\relax
2263     \ifFBFrenchFootnotes
2264         \@makefntextFB{#1}%
2265     \else
2266         \@makefntextORI{#1}%
2267     \fi
2268     \let\FBeverypar@quote\FBeverypar@save
2269     \lcalleftbox{\FBeveryline@quote}}%

```

```

2270     \fi
2271   }%
2272 }

```

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`.

```

2273 \newcommand*\AddThinSpaceBeforeFootnotes{\FBAutoSpaceFootnotestruer}
2274 \newcommand*\FrenchFootnotes{\FBFrenchFootnotestruer}
2275 \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.

```

2276 \FBClean@on@exit
2277 \ldf@finish\CurrentOption
2278 \let\loadlocalcfg\FB@llc
2279 </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`.

```

2280 <*acadian>
2281 \PackageInfo{acadian.ldf}%
2282 {\`acadian' dialect is currently\MessageBreak
2283  *absolutely identical* to the\MessageBreak
2284  `french' language; reported}
2285 </acadian>
2286 <*canadien>
2287 \PackageWarning{canadien.ldf}%
2288 {Option `canadien' for Babel is *deprecated*,\MessageBreak
2289  it might be removed sooner or later. Please\MessageBreak
2290  use `acadian' instead; reported}%
2291 \def\CurrentOption{acadian}

2292 \def\datecanadien{\dateacadian}
2293 \def\captionscanadien{\captionacadian}
2294 \def\extrascanadien{\extrasacadian}
2295 \def\noextrascanadien{\noextrasacadian}
2296 </canadien>
2297 <*francais>
2298 \PackageWarning{francais.ldf}%

```

```

2299 {Option `francais' for Babel is *deprecated*,\MessageBreak
2300 it might be removed sooner or later. Please\MessageBreak
2301 use `french' instead; reported}%
2302 \chardef\l@francais\l@french
2303 \def\CurrentOption{french}
2304 </francais>

```

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

```

2305 <frenchb>
2306 \def\bbbl@tempa{frenchb}
2307 \ifx\CurrentOption\bbbl@tempa
2308 \chardef\l@frenchb\l@french
2309 \def\CurrentOption{french}
2310 \PackageWarning{babel-french}%
2311 {Option `frenchb' for Babel is *deprecated*,\MessageBreak
2312 it might be removed sooner or later. Please\MessageBreak
2313 use `french' instead; reported}
2314 \else
2315 \def\bbbl@tempa{francais}
2316 \ifx\CurrentOption\bbbl@tempa
2317 \chardef\l@francais\l@french
2318 \def\CurrentOption{french}

```

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

```

2319 \ifx\magnification\@undefined
2320 \PackageWarning{babel-french}%
2321 {Option `francais' for Babel is *deprecated*,\MessageBreak
2322 it might be removed sooner or later. Please\MessageBreak
2323 use `french' instead; reported}
2324 \fi
2325 \else
2326 \def\bbbl@tempa{canadien}
2327 \ifx\CurrentOption\bbbl@tempa
2328 \def\CurrentOption{acadian}
2329 \PackageWarning{babel-french}%
2330 {Option `canadien' for Babel is *deprecated*,\MessageBreak
2331 it might be removed sooner or later. Please\MessageBreak
2332 use `acadian' instead; reported}
2333 \fi
2334 \fi
2335 \fi
2336 </frenchb>
2337 <acadian|canadien|frenchb|francais>\input french.lda\relax
2338 <acadian|canadien>\let\extrasacadian\extrasfrench
2339 <acadian|canadien>\let\noextrasacadian\noextrasfrench
2340 <acadian|canadien|frenchb|francais|french>\endinput

```

3 Change History

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

v3.5r	General: Compatibility with ucharclasses package added. . . 30	v3.5k	General: \degre, \degres, \circonflexe, \tild, \boi and \at are now safe in bookmarks. . . 44
v3.5q	\listFB: Bug correction: \parsep should be related to \parskip and \listparindent to \parindent. . 67		\pdfstringdefDisableCommands dropped. 66
v3.5p	\DecimalMathComma: \DecimalMathComma can again be used in the preamble for a global action. It now works as expected inside a group. 45		Reorganise warnings about ‘:’ in captions, according to enhancements in caption.sty v3.5a. 51
	\frquote: \FBeveryline@quote: no need for a penalty inside a \lcalleftbox. 39		\bsc: \bsc now relies on \texorpdfstring to be safe in bookmarks. 44
v3.5o	General: \shorthandon and \shorthandoff are no longer redefined in LuaTeX (it broke \shorthandoff*). 29		\captionsfrench: Small caps removed in \figurename and \tablename, use \fnum@figure and \fnum@table instead. 48
	\FB@xetex@punct@french: \shorthandon and \shorthandoff are no longer redefined (it broke \shorthandoff*). 31		\FB@fg: \FB@og and \FB@fg now rely on \texorpdfstring to be safe in bookmarks. 37
	frenchb.lua: Opening guill.: look ahead when next is a penalty (nobreak space). 27		\frquote: \frquote now relies on \texorpdfstring to be safe in bookmarks. 39
v3.5n			\fup: \up and \fup now rely on \texorpdfstring to be safe in bookmarks. 41
	\bbl@frenchindent: \bbl@frenchindent changed. \bbl@nonfrenchindent removed. 72		\no: \no, \nos, \No, \Nos, \primo, \fprimo, now rely on \texorpdfstring to be safe in bookmarks. 43
	\bsc: Added command \bname (no small caps). 44	v3.5j	General: For memoir, koma-script and beamer captions, \FB@std@sep has to be defined before activating the colon. 33
	\frenchsetup: \FBGlobalLayoutFrench no longer set to false when French is not the main language. 55	v3.5i	\FBprocess@options: For memoir, koma-script and beamer classes, leave caption delimiter unchanged if it has been user customised. . . 64
v3.5m	\FBtextellipsis: No longer redefine \dots, only \textellipsis’s default definition is changed in French. 52	v3.5h	frenchb.lua: Added glues and penalties should inherit attributes from the related punctuation character; this is mandatory for Lua-UL to underline and highlight them. Thanks to Marcel Krüger for providing the fix. 24
v3.5l	General: No warning about \@makecaption for more classes. 51		Code reorganised for better efficiency. 24
	\captionsfrench: Redefine \fnum@figure and \fnum@table separately. 48		

v3.5g	frenchb.lua: The kerning callback is a bit specific: adding code with add_to_callback actually deletes the legacy kerning as pointed out by Marcel Krüger on SE.	24	lists' items can be typeset as paragraphs with indented labels while the default leaves the labels hanging into the left margin.	68
v3.5f	General: \@canadien was defined too early in file 'canadien.ldf': \@acadian might not be defined. \selectlanguage{canadien} allowed again only for backward compatibility (deprecated).	15	\descriptionFB: ListItemsAsPar option taken into account for description lists.	70
	\DecimalMathComma: Fixed bug with the acadian language. Warning added if used with the icomma package.	45	\frenchsetup: New option ListItemsAsPar for displaying lists' items "as paragraphs".	54
v3.5e	\frenchsetup: StandardLayout and GlobalLayoutFrench options can no longer be toggled when French is not the main language.	55	v3.4d	\frenchsetup: New test for deciding about utf8 encoding for keys og and fg (the former one fails with LaTeX 2018 release).
	\frquote: Make resettings global on exit.	40	v3.4c	\ifFBXeTeX: Reverting to former test, beware of \XeTeXrevision left as \relax by careless testing.
	new command \NoEveryParQuote. reset \FB@addGUIlSpace attribute inside \localleftbox (LuaTeX).	40	v3.4b	\datefrench: Do not redefine \date as \frenchdate in French.
		39	v3.4a	General: \LdfInit checks \FBclean@on@exit instead of \captionfrench (undefined in PLain). Prevents loading french.ldf again with acadian option.
v3.5d	\frenchsetup: ReduceListSpacing option depreciated: see StandardListSpacing.	54		babel-french now requires eTeX.
v3.5c	General: Remove grouping inside \@makefntext, \localleftbox and \FBeverypar@quote saved and restored instead.	74		Lua function token.get_meaning requires LuaTeX 1.0.
	\frquote: \FBeverypar@quote's value now properly reset across level changes.	39		New \FBgspchar to customise the space character to be used for \og and \fg with the UnicodeNoBreakSpaces option.
	\noextrasfrench: \lccode of quote 0x27 changed from 0x2019 to 0x27 for Unicode engines.	16		New attribute \FB@dialect for the French dialect acadian.
v3.5b	General: Reset \FBeverypar@quote locally inside \@makefntext. Needed by \frquote.	74		New command \FBsetspaces to fine tune spacing independently in French and in French dialects.
	\frquote: New command \FB@addquote@everypar to manage \everypar: \frquote failed when used immediately after a sectioning command.	38		Shrink/stretch removed in \FBthousandsep.
v3.5a	General: New optional layout for lists:			Toks \FBcolonsp, \FBthinsp and \FBguillsp removed.
				\datefrench: Specific code for Plain finally removed (babel bug reported).
				\extrasfrench: Change \(\no)extras\CurrentOption to \(\no)extrasfrench. \(\no)extrasacadian will be defined as \(\no)extrasfrench in file acadian.ldf.

<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.	55	<code>\frenchpartfirst</code> , <code>\frenchpartsecond</code> and <code>\frenchpartnameord</code> added.	48
<code>frenchb.lua</code> : Global ‘ <code>FBsp</code> ’ table added; local function ‘ <code>get_glue</code> ’ changed into global ‘ <code>FBget_glue</code> ’.	23	<code>\FBthinspace</code> : Skips <code>\FBcolonskip</code> and <code>\FBthinskip</code> replaced by <code>\FBcolonsp</code> and <code>\FBthinsp</code>	17
v3.3d		<code>\frenchsetup</code> : <code>\frenchbsetup</code> is now an alias for <code>\frenchsetup</code>	54
<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	Options <code>INGuillSpace</code> , <code>ThinColonSpace</code> no longer delayed <code>AtBeginDocument</code>	54
v3.3c		<code>\frquote</code> : <code>\FB@quotespace</code> (kern), changed into <code>\FB@guillspace</code>	39
General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, <code>fontspec</code> is no longer required.	66	v3.2h	
New command <code>\FBthousandsep</code> to customise numprint.	48	<code>\@makefnctextFB</code> : With <code>beamer.cls</code> , add <code>\llap</code> to <code>\@thefnmark</code> for notes numbered over 99.	74
New configurable kerns <code>\FBmedkern</code> , and <code>\FBthickkern</code> suitable for HTML translation.	43	<code>\bbl@frenchlistlayout</code> : Execute <code>\update@frenchlists</code> only if <code>GlobalLayoutFrench</code> is false. Delete stuff for lists in <code>\noextrsfrench</code>	71
Reorganise warnings when the caption, subcaption or floatrow packages are loaded before <code>babel/french</code>	51	<code>\frenchsetup</code> : Option <code>GlobalLayoutFrench</code> skipped when French is not the main language.	55
Reset <code>\localleftbox</code> locally inside <code>\@makefnctext</code> . Needed by <code>\frquote</code> with LuaTeX.	74	v3.2g	
<code>\frenchsetup</code> : New option ‘ <code>UnicodeNoBreakSpaces</code> ’ for html translators (LuaLaTeX only).	59	General: Changed Unicode definition of <code>\boi</code>	44
<code>frenchb.lua</code> : Function ‘ <code>get_glue</code> ’ robustified. ‘ <code>french_punctuation</code> ’ can insert Unicode characters instead of glues.	22	<code>fontspec</code> defines TU encoding now and no longer loads <code>xunicode.sty</code> . Test changed.	66
v3.3b		Issue a warning if <code>beamerarticle.sty</code> is loaded after <code>babel</code>	53
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 options.	76	<code>\frenchsetup</code> : Minimal list customisation when <code>beamerarticle.sty</code> is loaded.	55
New ‘if’ <code>\ifFBfrench</code> to replace <code>\iflanguage</code> test which is based on patterns.	16	Warn when wrong values are provided to options <code>EveryParGuill</code> or <code>EveryLineGuill</code>	58
v3.3a		<code>\frquote</code> : Default options of <code>\frquote</code> are no longer engine-dependent.	38
General: Compatibility code for pre 2015/10/01 LaTeX release removed, see <code>ltnews23.tex</code>	20	v3.2f	
Skip <code>\FBguillskip</code> for LuaTeX replaced by <code>\FBguillsp</code>	18	<code>\DecimalMathComma</code> : Fixed conflict with the <code>icomma</code> package.	45
<code>\captionsfrench</code> : Commands		v3.2e	
		General: Add missing redefinitions for <code>\leftmarginiv</code> , <code>\leftmarginvi</code> . Suggested by J.F. Burnol.	68
		<code>\DecimalMathComma</code> : <code>\DecimalMathComma</code> didn’t work with LuaTeX. Fixed now.	45
		v3.2d	
		<code>\descriptionFB</code> : Changed	

<ul style="list-style-type: none"> <code>\listindentFB</code> to <code>\descindentFB</code> which defaults to <code>\listindentFB</code>. <code>\leftmargini</code> reduced when <code>\descindentFB</code> is null. 70 	<ul style="list-style-type: none"> <code>\@makefntextFB</code> (pointed out by DB). The same is true for memoir and koma-script classes (done). . . 73
v3.2c	<code>\fg: \xspace</code> moved from <code>\FB@fg</code> to <code>\fg: \xspace</code> messes up <code>\frquote</code> , pointed out by Sonia Labetoulle. As a side effect <code>\xspace</code> is now active in <code>\fg</code> in and outside French. . . . 38
<ul style="list-style-type: none"> General: New LuaTeX attribute <code>\FB@spacing</code>. 20 Newif <code>\ifFB@spacing</code> and new commands <code>\FB@spacingon</code>, <code>\FB@spacingoff</code> to control space tuning in French. 20 Switch <code>\ifFB@spacing</code> added to the four French shorthands. 33 <code>\FB@xetex@punct@french</code>: Switch <code>\ifFB@spacing</code> added to all <code>\XeTeXinterchartoks</code> commands. 31 <code>\FB@thinspace</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>\frenchsetup</code>: Add a warning about options <code>og/fg</code> for old XeTeX or LuaTeX engines requiring active characters. 59 <code>\NoAutoSpacing</code>: New definition based on <code>\FB@spacing@off</code> common to all engines. 36 <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>. 36 	<ul style="list-style-type: none"> v3.1m <code>frenchb.lua: new_glue_scaled</code> returns nil in case of invalid font table (i.e. <code>lcircle1.pfb</code>). In such cases babel-french leaves the node list unchanged. 24
<ul style="list-style-type: none"> v3.2b General: Load <code>lualatex.tex</code> for plain LuaTeX to ensure <code>\newattribute</code> is defined. 20 Warning added when the subcaption package is loaded before <code>babel/french</code>. 51 <code>\ifFB@xetex@punct</code>: New counter <code>\FB@nonchar</code> needed for non characters: it's value will be 4095 for new engines and 255 for older ones. 17 <code>\NoAutoSpacing</code>: <code>\NoAutoSpacing</code> made robust. 36 <code>frenchb.lua: glue_spec</code> removed; starting with LuaTeX 0.95, glue specifications fit in glue. 24 	<ul style="list-style-type: none"> v3.1l General: Add a variant of <code>\babel@savevariable</code> to save <code>\XeTeXcharclass(es)</code> in a loop. . . 31 <code>\FB@xetex@punct@french</code>: Save and restore <code>\XeTeXinterchartokenstate</code>, <code>\shorthandon</code>, <code>\shorthandoff</code> using <code>\babel@savevariable</code> and <code>\babel@save</code>, <code>\XeTeXcharclass(es)</code> using <code>\FB@savevariable@loop</code>. 31 <code>frenchb.lua: font.getfont(fid)</code> possibly returns nil even for a positive fid (i.e. <code>AMS lcircle1.pfb</code>). Reported by François Legendre. . . 24
<ul style="list-style-type: none"> v3.2a <code>\@makefntextFB</code>: <code>beamer.cls</code> requires a specific definition of 	<ul style="list-style-type: none"> v3.1k General: (pdfTeX shorthands) <code>test on \lastskip</code> changed from <code>0pt</code> to <code>1sp</code> for active punctuation for consistency with XeTeX and LuaTeX. 33 <code>\FB@xetex@punct@french</code>: Thin glues (less than <code>1sp</code>) should not trigger space insertion before high punctuation. Add a check on <code>\lastkip</code>. 31
	<ul style="list-style-type: none"> v3.1j General: Loading <code>lualatexbase.sty</code> is no longer needed with LaTeX release 2015/10/01 or later. 20 <code>\frquote</code>: <code>\fr@quote</code> completely rewritten: <code>\leavevmode</code> added and explicitly save/retore <code>\everypar</code> and <code>\localleftbox</code> instead of using a group in order to ensure compatibility with package <code>wrapfig</code>. 39 <code>\PackageWarning</code> is undefined in Plain, use <code>\fb@warning</code> instead. . 39

v3.1i	General: Remove restriction about loading numprint.sty after babel.	53	babel-french’s documentation. Pointed out by Denis Bitouzé.	64
	\frquote: \luatexlocalleftbox changed to \localleftbox by new LaTeX release 2015/10/01.	39	Definition of \captionformat and \captiondelim changed when option CustomiseFigTabCaptions is set to false.	64
	nombre: \nombre command changed when numprint.sty is not loaded: only one warning, no error.	48	\FBthinspace: \FBthinspace is no longer a kern but a skip (babel-french adds a nobreak penalty before it).	17
v3.1h	General: french.cfg from e-french conflicts with babel-french. Do NOT load it (no need for .cfg files with babel-french anyway).	76	v3.1e	\frenchsetup: Corrected typo: SmallCapsFigTabcaptions instead of SmallCapsFigTabCaptions. Pointed out by Céline Chevalier.
v3.1g	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	v3.1d	General: New section: issue warnings if packages listings, numprint and natbib are loaded too early or too late vs babel.
	Use Babel defined loops \bb1@for instead of \@for borrowed from file ltcntrl.dtx (\@for is undefined in Plain).	30	v3.1c	frenchb.lua: Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André.
	\captionfrench: \partname’s definition depends now on flag PartNameFull. No need to redefine it in \frenchbsetup.	48	v3.1b	\captionfrench: Change \scshape to customisable \FBfigtabshape for \figurename and \tablename.
	\frenchsetup: Bug fix for koma-scripts classes: a spurious dot was added by the \partformat command.	55		\frenchsetup: New option SmallCapsFigTabCaptions.
	PartNameFull now just sets the flag, nothing to add to \captionfrench when false.	54		\ieres: Removed \lowercase from definitions of \ieme and co: \up already does the conversion.
	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		\no: Removed \lowercase from definitions of \FrenchEnumerate, ... \No and co: \up already does the conversion.
	flag addgl set to false for ‘»’ at the beginning of an \hbox or a paragraph or a tabular ‘l’ and ‘c’ columns.	27		frenchb.lua: Add a check for null fid in french_punctuation (Tikz \nullfont). Bug pointed out by Paul Gaborit.
	Node HLIST added; node TEMP added for the first node of \hboxes.	22	v3.1a	General: fontspec is not required for T1 fonts used with the luainputenc.sty package.
v3.1f	General: \FBCaption@Separator changed when option CustomiseFigTabCaptions is set to false.	51		Misplaced \fi for plain formats.
	\FBprocess@options: Bug fix for the beamer class: figure and table captions are now consistent with			New command \frquote for imbedded or long French quotations.
				\frenchsetup: Codes 0x13 and 0x14 added for French quotes in T1-encoding. Support for older

versions of LuaTeX and XeTeX dropped.	59	\PackageInfo.	14
New options InnerGuillSingle, EveryParGuill and EveryLineGuill to control \frquote.	54	Merging of \captionsfrenchb, \captionsfrançais with \captionsfrench deleted in favor of new babel 3.9 syntax.	50
frenchb.lua: Added flag addgl which must also be true when prev or next is not a char (i.e. \kern0 in «\texttt{a}»).	27	More informative, less TeXnical warning about \makecaption. ..	51
Codes 0x13 and 0x14 added for French quotes in T1-encoding. ...	22	New flag \ifFB@luatex@punct for ‘high punctuation’ management with LuaTeX engines.	17
Look ahead when next is a kern (i.e. in «\texttt{a} »).	27	New handling of ‘high punctuation’ through callbacks with LuaTeX engines.	20
v3.0c		No warning about \makecaption for SMF classes.	51
General: babel-french requires babel-3.9i.	14	Options processing completely reorganised, now \babel@save and \babel@savevariable are usable for French.	53
Just load luatexbase.sty instead of luaotfload.sty with plain formats.	20	Support for options frenchb, français, canadien, acadian changed.	14
No need to define \l@french as \lang@french, babel.def (3.9j) takes care for this.	15	Test \ifXeTeX changed to \ifFBunicode and ‘xltxtra’ changed to ‘fontspec’.	66
\frenchsetup: New option INGuillSpace.	54	\CaptionSeparator: Remove \FBCaption@SeparatorORI, use \babel@save instead.	50
No list customisation when beamer class is loaded.	55	\captionsfrench: Take advantage of babel’s \SetString commands for captionnames.	48
frenchb.lua: Null glues should not trigger space insertion before high punctuation. Bug pointed out by Benoit Rivet for the ‘lstlisting’ environment of the listings package.	25	\datefrench: Take advantage of babel’s \SetString commands for \datefrench. Doesn’t work with Plain (yet?).	40
v3.0b		\descriptionFB: Added \listindentFB to \itemindent. Suggested by Denis Bitouzé.	70
General: frenchb.lua was not found by Lua function dofile (not kpathsea aware). Call function kpse.find_file first, as suggested by Paul Gaborit.	29	\extrasfrench: Take advantage of babel’s \babel@savevariable to handle apostrophe’s \lccode. ...	16
Require luatexbase with LaTeX2e in case fontspec has not been loaded before babel.	20	\FB@fg: Definitions of \FB@og and \FB@fg now depend on punctuation handling (LuaTeX / XeTeX / active).	37
v3.0a		\FBprocess@options: With koma-script and memoir class, customise \captionformat and \captiondelim.	64
General: \bbl@nonfrenchguillemets deleted, use \babel@save instead. \LdfInit checks \captionsfrench instead of \datefrench to avoid a conflict with papertex.cls which loads datetime.sty.	14	\frenchsetup: New options OldFigTabCaptions and CustomiseFigTabCaptions.	54
french.cfg will be loaded (if found) instead of frenchb.cfg. NO NEED for .cfg files in French anyway. ..	76		
In Plain, provide a substitute for \PackageWarning and			