

A Babel language definition file for French

frenchb.dtx v3.5s, 2024-02-29

Daniel Flipo
daniel.flipo@free.fr

Contents

1 The French language	2
1.1 Basic interface	2
1.2 Customisation	4
1.2.1 <code>\frenchsetup</code>	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	54
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 <code>frenchb.ldf</code> , <code>francais.ldf</code> , <code>canadien.ldf</code> and <code>acadian.ldf</code>	76
3 Change History	79

1 The French language

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

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

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

`babel-french` has been improved using helpful suggestions from many people, mainly from Jacques André, Michel Bovani, Thierry Bouche, Vincent Jalby, Denis Bitouzé, 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 v3.5s 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 v3.5s and was last revised on 2024-02-29.

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 `\ifBunicode` 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 `ltxluatex.tex` if it hasn’t been loaded before Babel.

```

209   \input ltxluatex.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 DISC = node_id("disc")
291 local KERN = node_id("kern")
292 local GLUE = node_id("glue")
293 local GLYPH = node_id("glyph")

```

```

294 local PENALTY      = node_id("penalty")
295 local nobreak      = new_node(PENALTY)
296 nobreak.penalty    = 10000
297 local nbspace      = new_node(GLYPH)
298 local insert_node_before = node.insert_before
299 local insert_node_after  = node.insert_after
300 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.

```

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

```

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

```

```

339 FBsp.guill.ch = {}
340 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`.

```

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

```

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

```

365 local FBspacing      = luatexbase.attributes['FB@spacing']
366 local addDPspace     = luatexbase.attributes['FB@addDPspace']
367 local addGUILspace   = luatexbase.attributes['FB@addGUILspace']
368 local FBucsNBSP      = luatexbase.attributes['FB@ucsNBSP']
369 local FBdialect      = luatexbase.attributes['FB@dialect']
370 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`.

```

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

```

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

```

373   node.kerning(head)
374   for item in node.traverse_id(GLYPH, head) do

```



```

375     local lang = item.lang
376     local char = item.char

```

Skip glyphs not concerned by French kernings.

```

377     if (lang == FR_fr or lang == FR_ca) and
378         (FB_punct_left[char] or FB_punct_right[char]) then
379         local fid = item.font
380         local attr = item.attr
381         local FRspacing = has_attribute(item, FBspacing)
382         FRspacing = FRspacing and FRspacing > 0
383         local FRucsNBSP = has_attribute(item, FBucsNBSP)
384         FRucsNBSP = FRucsNBSP and FRucsNBSP > 0
385         local FRdialect = has_attribute(item, FBdialect)
386         FRdialect = FRdialect and FRdialect > 0
387         local SIG = has_attribute(item, addGUILspace)
388         SIG = SIG and SIG > 0
389         if FRspacing and fid > 0 then
390             if FB_punct_left[char] then
391                 local prev = item.prev
392                 local prev_id, prev_subtype, prev_char
393                 if prev then
394                     prev_id = prev.id
395                     prev_subtype = prev.subtype
396                     if prev_id == GLYPH then
397                         prev_char = prev.char
398                     end
399                 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.

```

400         local is_glue = prev_id == GLUE
401         local glue_wd
402         if is_glue then
403             glue_wd = prev.width
404         end
405         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.

```

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

```

```

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

```

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

```

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

```

```
462             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.

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

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

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

```
498             elseif SIG then
```

```

499         local next = item.next
500         local next_id, next_subtype, next_char, nextnext, kern_wd
501         if next then
502             next_id = next.id
503             next_subtype = next.subtype

```

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

```

504             if next_id == PENALTY then
505                 nextnext = next.next
506                 if nextnext and nextnext.id == GLUE then
507                     head = remove_node(head,nextnext,true)
508                     head = remove_node(head,next,true)
509                     next = item.next
510                     if next then
511                         next_id = next.id
512                         next_subtype = next.subtype
513                         if next_id == GLYPH then
514                             next_char = next.char
515                         end
516                     end
517                 end
518             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}~»):

```

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

```

```

548         glue_wd = next.width
549     end

```

The `addgl` flag only depends on `next_char` and `is_glue`:

```

550         local addgl = (next_char and not FB_guill_null[next_char])
551         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).

```

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

```

586 <french>
587 \ifFB@luatex@punct
588 \def\activate@luatexpunct{%

```

```

589 \directlua{%
590   FR_fr = \the\l@french ; FR_ca = \the\l@acadian ;
591   local path = kpse.find_file("frenchb.lua", "lua")
592   if path then
593     local f = dofile(path)
594     luatexbase.add_to_callback("kerning",
595                               f, "frenchb.french_punctuation")
596   else
597     texio.write_nl('')
598     texio.write_nl('*****')
599     texio.write_nl('Error: frenchb.lua not found.')
600     texio.write_nl('*****')
601     texio.write_nl('')
602   end
603 }%
604 }
605 \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.

```

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

```

```

613             with this version of XeTeX!}
614 \fi

```

Six new character classes are defined for babel - french.

```

615 \newXeTeXintercharclass\FB@punctthick
616 \newXeTeXintercharclass\FB@punctthin
617 \newXeTeXintercharclass\FB@punctnul
618 \newXeTeXintercharclass\FB@guilo
619 \newXeTeXintercharclass\FB@guilf
620 \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.

```

621 \def\FBsavevariable@loop#1#2{\begingroup
622   \toks@\expandafter{\originalTeX #1}%
623   \edef\x{\endgroup
624     \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
625   \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.

```

626 \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F}
627 \def\FB@charlistUCC{}
628 \def\FB@charlistxeCJK{}
629 \edef\FB@charlistsave{\FB@charlist}
630 \ifLaTeXe
631   \AtBeginDocument{%
632     \@ifpackageloaded{ucharclasses}%
633     {\ifdefined\BasicLatinClass
634       \RenewCommandCopy{\FB@stdchar}{\BasicLatinClass}%
635       \def\FB@charlistUCC{"C0,"C2,"C6,"C7,"C8,"C9,"CA,"CB,"CE,"CF,%
636         "D4,"D6,"D9,"DB,"DC,"E0,"E2,"E6,"E7,"E8,"E9,"EA,"EB,"EE,%
637         "EF,"F4,"F6,"F9,"FB,"FC,"152,"153,"17F,"2019}%
638       \addto\FB@charlist{\FB@charlistUCC}%
639       \edef\FB@charlistsave{\FB@charlist}%
640       \fi
641     }{}%
642     \@ifpackageloaded{xeCJK}%
643     {\def\FB@charlistxeCJK{%
644       "29,"5D,"7B,"7D,"2C,"2D,"2E,"22,"25,"27,"60,"2019}%
645       \addto\FB@charlist{\FB@charlistxeCJK}%
646       \edef\FB@charlistsave{\FB@charlist}%
647     }{}%
648   }
649 \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.

```

650 \newcommand*{\FB@xetex@punct@french}{%
651   \babel@savevariable{\XeTeXinterchartokenstate}%
652   \bbl@for\FB@char\FB@charlistsave

```

```
653         {\FBsavevariable@loop{\XeTeXcharclass}{\FB@char}}%
```

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

```
654     \ifx\FB@charlistUCC\@empty\else
655         \bbl@for\FB@char\FB@charlistUCC
656         {\XeTeXcharclass \FB@char \FB@stdchar}%
657     \fi
```

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

```
658     \ifx\FB@charlistxeCJK\@empty\else
659         \bbl@for\FB@char\FB@charlistxeCJK
660         {\XeTeXcharclass\FB@char=\FB@stdchar}%
661     \fi
```

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

```
662     \bbl@for\FB@char {\[, \[, "A0, "202F}%
663         {\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).

```
664     \XeTeXinterchartokenstate=\@ne
665     \XeTeXcharclass \: = \FB@punctthick
666     \XeTeXinterchartoks \FB@stdchar \FB@punctthick = {%
667         \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
668     \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
669         \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.

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



```

687         \ifFB@spacing
688         \ifhmode
689             \ifdim\lastskip>1sp
690                 \unskip\penalty\@M\FBthinspace
691             \else
692                 \FDP@thinspace
693             \fi
694         \fi
695     \fi}%
696 \XeTeXinterchartoks \FB@guilo \FB@stdchar = {%
697     \ifFB@spacing\FB@guillspace\fi}%
698 \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
699     \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
700 \XeTeXinterchartoks \FB@stdchar \FB@guilf = {%
701     \ifFB@spacing\FB@guillspace\fi}%
702 \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
703     \ifFB@spacing\FB@guillspace\fi}%
704 \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
705     \ifFB@spacing\unskip\FB@guillspace\fi}%
706 }
707 \addto\extrasfrench{\FB@xetex@punct@french}

```

End of specific code for punctuation with modern XeTeX engines.

```
708 \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 :

```

709 \newif\ifFB@koma
710 \ifLaTeXe
711     \@ifclassloaded{scrartcl}{\FB@komatrue}{}
712     \@ifclassloaded{scrbook}{\FB@komatrue}{}
713     \@ifclassloaded{scrreprt}{\FB@komatrue}{}
714     \ifFB@koma\def\FB@std@capsep{: \ } \fi
715     \@ifclassloaded{beamer}{\def\FB@std@capsep{: \ }}{}
716     \@ifclassloaded{memoir}{\def\FB@std@capsep{: }}{}
717 \fi

718 \ifFB@active@punct
719     \initiate@active@char{:}%
720     \initiate@active@char{;}%
721     \initiate@active@char{!}%
722     \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.

```
723 \declare@shorthand{french}{;}{;%}
```

```

724 \iffB@spacing
725 \iffmode
726 \ifdim\lastskip>1sp
727 \unskip\penalty\@M\FBthinspace
728 \else
729 \FDP@thinspace
730 \fi
731 \fi
732 \fi

```

Now we can insert a ; character.

```
733 \string;}
```

The next three definitions are very similar.

```

734 \declare@shorthand{french}{!}{%
735 \iffB@spacing
736 \iffmode
737 \ifdim\lastskip>1sp
738 \unskip\penalty\@M\FBthinspace
739 \else
740 \FDP@thinspace
741 \fi
742 \fi
743 \fi
744 \string!}
745 \declare@shorthand{french}{?}{%
746 \iffB@spacing
747 \iffmode
748 \ifdim\lastskip>1sp
749 \unskip\penalty\@M\FBthinspace
750 \else
751 \FDP@thinspace
752 \fi
753 \fi
754 \fi
755 \string?}
756 \declare@shorthand{french}{:}{%
757 \iffB@spacing
758 \iffmode
759 \ifdim\lastskip>1sp
760 \unskip\penalty\@M\FBcolonspace
761 \else
762 \FDP@colonspace
763 \fi
764 \fi
765 \fi
766 \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.

```

767 \declare@shorthand{system}{:}{\string:}
768 \declare@shorthand{system}{!}{\string!}
769 \declare@shorthand{system}{?}{\string?}
770 \declare@shorthand{system}{;}{\string;}

```

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

```
771 \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.

```
772 \bbl@activate{:}\bbl@activate{;}%
773 \bbl@activate{!}\bbl@activate{?}%
774 }
775 \addto\noextrasfrench{%
776 \bbl@deactivate{:}\bbl@deactivate{;}%
777 \bbl@deactivate{!}\bbl@deactivate{?}%
778 }
779 \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.

```
780 \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 L^AT_EX.

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

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

`\rmfamilyFB` In L^AT_EX₂e `\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.

```
800 \ifLaTeXe
801 \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
802 \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on \rmfamilyORI}
803 \DeclareRobustCommand\sffamilyFB{\FB@spacing@on \sffamilyORI}
804 \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.

```
805 \DeclareRobustCommand*\NoAutoSpacing*{%
806 \FB@spacing@off
807 \ifFB@active@punct\shorthandoff{;:!?}\fi
808 }
```

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.

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

```
824 \let\xspace\relax
825 \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.

```
826 \providecommand\texorpdfstring[2]{#1}
827 \newcommand*\FB@og{\texorpdfstring{\@FB@og}{\guillemotleft\space}}
828 \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.

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

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

```
840 \ifFB@xetex@punct
841 \DeclareRobustCommand*\@FB@og{\leavevmode
842     \bgroup\FB@spacingfalse\guillemotleft\egroup
843     \FB@guillspace}
844 \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@\unskip\fi
845     \FB@guillspace
846     \bgroup\FB@spacingfalse\guillemotright\egroup}
847 \fi
848 \ifFB@active@punct
849 \DeclareRobustCommand*\@FB@og{\leavevmode
850     \guillemotleft
851     \FB@guillspace}
852 \DeclareRobustCommand*\@FB@fg{\ifdim\lastskip>\z@\unskip\fi
853     \FB@guillspace
854     \guillemotright}
855 \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.

```
856 \newcommand*\og{\@empty}
857 \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).

```

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

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

```

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

```

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

```

and the needed technical stuff to handle options:

```

881 \newcount\FBguill@level
882 \newtoks\FBold@everypar

```

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

```

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

```

```

895 \let\FBeveryparguill\FBguillopen
896 \let\FBverylineguill\FBguillnone
897 \let\FBeverypar@quote\relax
898 \let\FBveryline@quote\empty

```

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

```

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

```

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

```

910 \newcommand{\fr@quote}[1]{%
911   \leavevmode
912   \advance\FBguill@level by \@ne
913   \ifcase\FBguill@level
914   \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:

```

915   \ifx\FBeveryparguill\FBguillnone
916   \else
917     \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
918     \FB@addquote@everypar
919   \fi
920   \@ogi #1\@fgi
921 \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.

```

922   \ifx\FBverylineguill\FBguillopen
923     \def\FBveryline@quote{\FB@addGUILspace=\z@
924                           \guillemotleft\FBguillspace}%
925     \lcalleftbox{\FBveryline@quote}%
926     \let\FBeverypar@quote\relax
927     \@ogi #1\ifFBcloseguill\@fgi\fi
928   \else
929     \ifx\FBverylineguill\FBguillclose
930     \def\FBveryline@quote{\FB@addGUILspace=\z@
931                           \guillemotright\FBguillspace}%
932     \lcalleftbox{\FBveryline@quote}%
933     \let\FBeverypar@quote\relax
934     \@ogi #1\ifFBcloseguill\@fgi\fi
935   \else

```

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

```

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

```

Warn if `\FBguill@level > 2`:

```

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

```

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

```

962 \global\advance\FBguill@level by \m@ne
963 \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
964 \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
965   \global\let\FBeveryline@quote\empty
966   \ifx\FBeverylineguill\FBguillnone\else\localleftbox{}\fi
967 \fi
968 }

```

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.

```

969 \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.

```

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

```

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

```

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

```

2.5 Extra utilities

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

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

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

```

990 \newif\ifFB@poorman
991 \newdimen\FB@Mht
992 \ifLaTeXe
993   \AtEndOfPackage{\RequirePackage{scalegnt}}

```

`\FB@up@fake` holds the definition of fake superscripts. The scaling ratio is 0.65, raising is computed to put the top of lower case letters (like 'm') just under the top of upper case letters (like 'M'), precisely 12% down. The chosen settings look correct for most

fonts, but can be tuned by the end-user if necessary by changing `\FBsupR` and `\FBsupS` commands.

`\FB@lc` is defined as `\MakeLowercase` to inhibit the uppercasing of superscripts (this may happen in page headers with the standard classes but is wrong); `\FB@lc` can be re-defined to do nothing by option `LowercaseSuperscripts=false` of `\frenchsetup{}`.

```

994 \newcommand*\FBsupR{-0.12}
995 \newcommand*\FBsupS{0.65}
996 \newcommand*\FB@lc[1]{\MakeLowercase{#1}}
997 \DeclareRobustCommand*\FB@up@fake[1]{%
998   \settoheight{\FB@Mht}{M}%
999   \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
1000  \addtolength{\FB@Mht}{-\FBsupS ex}%
1001  \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
1002  }

```

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.

```

1003 \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
1004   \def\FB@suffix{#4}}
1005 \def\FB@x{x}
1006 \def\FB@j{j}
1007 \DeclareRobustCommand*\FB@up[1]{%
1008   \bgroup \FB@poormantrue
1009   \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.

```

1010   \edef\reserved@a{\lowercase{%
1011     \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
1012   \reserved@a
1013   {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
1014    \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
1015    \if\FB@poorman \FB@up@fake{#1}%
1016    \else \FB@up@real{#1}%
1017    \fi}%
1018   {\FB@up@fake{#1}}%
1019   \egroup}

```

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

```

1020 \newcommand*\FB@up@real[1]{\bgroup
1021   \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.

```

1022 \DeclareRobustCommand*\fup}[1]{%
1023   \texorpdfstring{\ifx\realsuperscript\undefined
1024     \FB@up{#1}%
1025   \else
1026     \bgroup\let\fakesuperscript\FB@up@fake
1027     \realsuperscript{\FB@lc{#1}}\egroup
1028   \fi
1029   }{#1}%
1030 }
```

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

```
1031 \providecommand*\up}{\fup}
```

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

```

1032 \else
1033 \providecommand*\up}[1]{\leavevmode\raiselex\hbox{\sevenrm #1}}
1034 \fi
```

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

```

\ier 1035 \def\ieme{\up{e}\xspace}
\iere 1036 \def\iemes{\up{es}\xspace}
\iemes 1037 \def\ier{\up{er}\xspace}
\iers 1038 \def\iers{\up{ers}\xspace}
\ieres 1039 \def\iere{\up{re}\xspace}
\ieres 1040 \def\ieres{\up{res}\xspace}
```

```

\FBmedkern
\FBthickkern 1041 \newcommand*\FBmedkern}{\kern+.2em}
1042 \newcommand*\FBthickkern}{\kern+.3em}
```

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

```

\fprimo) 1043 \newcommand*\FrenchEnumerate}[1]{%
  \nos 1044 #1\texorpdfstring{\up{o}\FBthickkern}{\textdegree\space}}
\Nos 1045 \newcommand*\FrenchPopularEnumerate}[1]{%
  \No 1046 #1\texorpdfstring{\up{o}}\FBthickkern}{\textdegree\space}}
\no Typing \primo should result in '°' (except in bookmarks where \textdegree is used
instead of o-superior),
1047 \def\primo{\FrenchEnumerate1}
1048 \def\secundo{\FrenchEnumerate2}
1049 \def\tertio{\FrenchEnumerate3}
1050 \def\quarto{\FrenchEnumerate4}
while typing \fprimo gives '°) (except in bookmarks where \textdegree is used
instead),.
1051 \def\fprimo){\FrenchPopularEnumerate1}
1052 \def\fsecundo){\FrenchPopularEnumerate2}
1053 \def\ftertio){\FrenchPopularEnumerate3}
1054 \def\fquarto){\FrenchPopularEnumerate4}
```

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

```

1055 \DeclareRobustCommand*\No}{%
1056   \texorpdfstring{N\up{o}\FBmedkern}{N\textdegree\space}}
1057 \DeclareRobustCommand*\no}{%
1058   \texorpdfstring{n\up{o}\FBmedkern}{n\textdegree\space}}
1059 \DeclareRobustCommand*\Nos}{%
1060   \texorpdfstring{N\up{os}\FBmedkern}{N\textdegree\space}}
1061 \DeclareRobustCommand*\nos}{%
1062   \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.

```

1063 \ifLaTeXe
1064   \DeclareRobustCommand*\bname}[1]{%
1065     \texorpdfstring{\leavevmode\beginngroup\kern0pt #1\endngroup}{#1}%
1066   }
1067   \DeclareRobustCommand*\bsc}[1]{%
1068     \texorpdfstring{\leavevmode\beginngroup\kern0pt \scshape #1\endngroup}%
1069     {\textsc{#1}}%
1070   }
1071 \else
1072   \newcommand*\bname}[1]{\leavevmode\beginngroup\kern0pt #1\endngroup}
1073   \let\bsc\bname
1074 \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.

```

1075 \ifFBunicode
1076   \providecommand*\textbackslash{{\char"005C}}
1077   \providecommand*\textasciicircum{{\char"005E}}
1078   \providecommand*\textasciitilde{{\char"007E}}
1079   \newcommand*\FB@degree{°}
1080 \else
1081   \ifLaTeXe
1082     \newcommand*\FB@degree{\r{}}
1083   \fi
1084 \fi
1085 \DeclareRobustCommand*\boi{\textbackslash}
1086 \DeclareRobustCommand*\circonflexe{\textasciicircum}
1087 \DeclareRobustCommand*\tild{\textasciitilde}
1088 \DeclareRobustCommand*\degree{%
1089   \texorpdfstring{\FB@degree}{\textdegree}}
1090 \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 0.3em, 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 T_S1-encoding.

```

1091 \DeclareRobustCommand*\degrees{\degree}
1092 \ifLaTeXe
1093   \AtBeginDocument{%
1094     \ifpackageloaded{fontspec}{\%
1095       \ifdefined\DeclareEncodingSubset
1096         \DeclareRobustCommand*\degrees{\%
1097           \texorpdfstring{\hbox{\UseTextSymbol{TS1}{\textdegree}}}{%
1098             \textdegree}}\%
1099       \else
1100         \def\Warning@degree@TSone{\FBWarning
1101           {Degrees would look better in TS1-encoding:%
1102             \MessageBreak add \protect
1103             \usepackage{textcomp} to the preamble.%
1104             \MessageBreak Degrees used}}
1105         \DeclareRobustCommand*\degrees{\%
1106           \texorpdfstring{\hbox to 0.3em{\hss\degree\hss}}{
1107             \Warning@degree@TSone
1108             \global\let\Warning@degree@TSone\relax}%
1109           {\textdegree}}\%
1110       \fi
1111     }%
1112   }
1113 \fi

```

2.6 Formatting numbers

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

Unfortunately, \newcount inside \if breaks Plain formats.

```

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

```

```

1127 \newcommand*{\std@math@comma}{%
1128   \mc@charfam=\Umathcharfam`,
1129   \mc@charslot=\Umathcharslot`,
1130   \Umathcode`= \mc@charclass \mc@charfam \mc@charslot
1131 }
1132 \else
1133   \std@mcc=\mathcode`,
1134   \dec@mcc=\std@mcc
1135   \@tempcnta=\std@mcc
1136   \divide\@tempcnta by "1000
1137   \multiply\@tempcnta by "1000
1138   \advance\dec@mcc by -\@tempcnta
1139   \newcommand*{\dec@math@comma}{\mathcode`= \dec@mcc}
1140   \newcommand*{\std@math@comma}{\mathcode`= \std@mcc}
1141 \fi
1142 \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`.

```

1143 \newif\if@FBpreamble
1144 \ifLaTeXe \@FBpreambletrue \fi
1145 \newif\if@preamble@DecimalMathComma
1146 \newcommand*{\DecimalMathComma}{%
1147   \if@FBpreamble \@preamble@DecimalMathCommatrue
1148   \else
1149     \ifFB@icomma
1150       \PackageWarning{french.ldf}{%
1151         icomma package loaded, \protect\DecimalMathComma\MessageBreak
1152         does nothing. Reported}%
1153     \else
1154       \ifFBfrench
1155         \dec@math@comma
1156         \let\dec@m@c\dec@math@comma
1157         \expandafter\addto\csname extras\language\endcsname
1158           {\dec@m@c}%
1159       \fi
1160     \fi
1161   \fi
1162 }
1163 \newcommand*{\StandardMathComma}{%
1164   \ifFB@icomma
1165     \PackageWarning{french.ldf}{%
1166       icomma package loaded, \protect\StandardMathComma\MessageBreak
1167       does nothing. Reported}%
1168   \else
1169     \ifFBfrench
1170       \std@math@comma
1171       \let\dec@m@c\relax
1172     \fi
1173   \fi
1174 }

```

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

```

1175 \ifLaTeXe\else
1176   \addto\noextrasfrench{\std@math@comma}
1177 \fi

```

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

```

1178 \newcommand*{\nombre}[1][\#1]\fb@warning{*** \noexpand\nombre
1179                                     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.

```

1180 \ifFB@luatex@punct
1181   \activate@luatexpunct
1182 \fi
1183 \let\FBstop@here\relax
1184 \def\FBclean@on@exit{%
1185   \let\ifLaTeXe\iffalse
1186   \let\LaTeXettrue\undefined
1187   \let\LaTeXefalse\undefined
1188   \let\FB@llc\loadlocalcfg
1189   \let\loadlocalcfg\@gobble}
1190 \ifx\magnification\@undefined
1191 \else
1192   \def\FBstop@here{%
1193     \FBclean@on@exit
1194     \ldf@finish\CurrentOption
1195     \let\loadlocalcfg\FB@llc
1196     \endinput}
1197 \fi
1198 \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`.

```

1199 \AtBeginDocument{%
1200   \@FBpreamblefalse
1201   \@ifpackageloaded{icomma}%
1202     {\FB@icommatrue
1203       \if@preamble@DecimalMathComma
1204         \PackageWarning{french.ldf}{%
1205           icomma package loaded, \protect\DecimalMathComma%
1206           \MessageBreak does nothing. Reported}%
1207       \fi
1208     }%
1209   {\if@preamble@DecimalMathComma
1210     \ifFB@mainlanguage@FR \dec@math@comma \fi
1211     \let\dec@m@c\dec@math@comma
1212     \addto\noextrasfrench{\dec@m@c}%
1213     \ifdefined\extrasacadian
1214       \addto\noextrasacadian{\dec@m@c}%
1215     \fi
1216   \fi

```

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

```

1217     \addto\noextrasfrench{\std@math@comma}%
1218     \ifdefined\noextrasacadian
1219         \addto\noextrasacadian{\std@math@comma}%
1220     \fi
1221 }%
1222 }

```

`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.

```

1223 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1224 \newcommand*{\Warning@nombre}[1]{%
1225     \ifdefined\numprint
1226         \numprint{#1}%
1227     \else
1228         \PackageWarning{french.ldf}{%
1229             \protect\nombre\space now relies on package numprint.sty,%
1230             \MessageBreak add \protect
1231             \usepackage[autolanguage]{numprint},\MessageBreak
1232             see file numprint.pdf for more options.\MessageBreak
1233             \protect\nombre\space called}%
1234         \global\let\Warning@nombre\relax
1235         {#1}%
1236     \fi
1237 }

1238 \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`.

```

1239 \providecommand*{\FBfigtabshape}{\scshape}

```

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

```

1240 \StartBabelCommands*{\BabelLanguages}{captions}
1241     [unicode, fontenc=TU EU1 EU2, charset=utf8]
1242     \SetString{\refname}{Références}
1243     \SetString{\abstractname}{Résumé}
1244     \SetString{\prefacename}{Préface}
1245     \SetString{\contentsname}{Table des matières}

```



```

1246 \SetString{\ccname}{Copie à }
1247 \SetString{\proofname}{Démonstration}
1248 \SetString{\partfirst}{Première}
1249 \SetString{\partsecond}{Deuxième}
1250 \SetStringLoop{ordinal#1}{%
1251   \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%
1252   Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1253   Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1254   Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1255 \StartBabelCommands*\BabelLanguages}{captions}
1256 \SetString{\refname}{R\ 'ef\ 'erences}
1257 \SetString{\abstractname}{R\ 'esum\ 'e}
1258 \SetString{\bibname}{Bibliographie}
1259 \SetString{\prefacename}{Pr\ 'eface}
1260 \SetString{\chaptername}{Chapitre}
1261 \SetString{\appendixname}{Annexe}
1262 \SetString{\contentsname}{Table des mati\`eres}
1263 \SetString{\listfigurename}{Table des figures}
1264 \SetString{\listtablename}{Liste des tableaux}
1265 \SetString{\indexname}{Index}
1266 \SetString{\figurename}{Figure}
1267 \SetString{\tablename}{Table}
1268 \SetString{\pagename}{page}
1269 \SetString{\seename}{voir}
1270 \SetString{\alsoname}{voir aussi}
1271 \SetString{\enclname}{P.~J. }
1272 \SetString{\ccname}{Copie \ `a }
1273 \SetString{\headtoname}{}
1274 \SetString{\proofname}{D\ 'emonstration}
1275 \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`.

```

1276 \SetString{\partfirst}{Premi\`ere}
1277 \SetString{\partsecond}{Deuxi\`eme}
1278 \SetString{\partnameord}{partie}
1279 \SetStringLoop{ordinal#1}{%
1280   \partfirst,\partsecond,Troisi\`eme,Quatri\`eme, Cinqui\`eme,%
1281   Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
1282   Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1283   Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1284   Vingti\`eme}
1285 \AfterBabelCommands{%
1286   \DeclareRobustCommand*\FB@emptypart{\def\thepart{\unskip}}%
1287   \DeclareRobustCommand*\FB@partname{%
1288     \ifFBPartNameFull
1289       \csname ordinal\romannumeral\value{part}\endcsname\space
1290       \partnameord\FB@emptypart
1291     \else
1292       Partie%
1293     \fi}%
1294   }
1295 \SetString{\partname}{\FB@partname}

```

```

1296 \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.).
1297 \AtBeginDocument{%
1298   \ifx\FBfigtabshape\relax
1299   \else
1300     \ifdefined\fnum@figure
1301       \let\fnum@figureORI\fnum@figure
1302       \renewcommand{\fnum@figure}{\ifFBfrench\FBfigtabshape\fi
1303                                     \fnum@figureORI}}%
1304     \fi
1305     \ifdefined\fnum@table
1306       \let\fnum@tableORI\fnum@table
1307       \renewcommand{\fnum@table}{\ifFBfrench\FBfigtabshape\fi
1308                                     \fnum@tableORI}}%
1309     \fi
1310   \fi
1311 }

```

2.8 Figure and table captions

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

```
1312 \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).

```

1313 \bgroup
1314 \catcode`:=12 \catcode`>=12 \relax
1315 \long\gdef\STD@makecaption#1#2{%
1316   \vskip\abovcaptionskip
1317   \sbox\@tempboxa{#1: #2}%
1318   \ifdim \wd\@tempboxa >\hsize
1319     #1: #2\par

```

```

1320 \else
1321 \global \@minipagefalse
1322 \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1323 \fi
1324 \vskip\belowcaptionskip}
1325 \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.

```

1326 \newif\if@FBwarning@capsep
1327 \ifFB@active@punct\@FBwarning@capseptrue\fi
1328 \newcommand*\CaptionSeparator{\space\textendash\space}
1329 \def\FBCaption@Separator{: }
1330 \long\def\FB@makecaption#1#2{%
1331 \vskip\abovecaptionskip
1332 \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1333 \ifdim \wd\@tempboxa >\hsize
1334 #1\FBCaption@Separator #2\par
1335 \else
1336 \global \@minipagefalse
1337 \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1338 \fi
1339 \vskip\belowcaptionskip}

```

Disable the standard warning with AMS and SMF classes.

```

1340 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1341 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1342 \@ifclassloaded{amsdtx}{\@FBwarning@capsepfalse}{}
1343 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1344 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1345 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1346 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}

```

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

```

1347 \@ifclassloaded{IEEEconf}{\@FBwarning@capsepfalse}{}
1348 \@ifclassloaded{IEEEtran}{\@FBwarning@capsepfalse}{}
1349 \@ifclassloaded{revtex4-2}{\@FBwarning@capsepfalse}{}
1350 \@ifclassloaded{svjour3}{\@FBwarning@capsepfalse}{}

```

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

```

1351 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1352 \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).

```

1353 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1354 \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').

```

1355 \AtBeginDocument{%
1356   \ifx\@makecaption\STD@makecaption
1357     \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`.

```

1358   \ifFBoldFigTabCaptions
1359   \else
1360     \def\FBCaption@Separator{\autospace@beforeFDP : }%
1361     \ifBFCustomiseFigTabCaptions
1362       \ifFB@mainlanguage@FR
1363         \def\FBCaption@Separator{\CaptionSeparator}%
1364       \fi
1365     \fi
1366   \fi
1367   \@FBwarning@capsepfalse
1368 \fi

```

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

```

1369   \@ifpackageloaded{caption}{\@FBwarning@capsepfalse}{}%
1370   \@ifpackageloaded{caption-light}{\@FBwarning@capsepfalse}{}%

```

Final warning if relevant:

```

1371   \if@FBwarning@capsep
1372     \FBWarning
1373     {Figures' and tables' captions might look like\MessageBreak
1374     `Figure 1:' in French instead of `Figure 1 :'.\MessageBreak
1375     If this happens, to fix this issue\MessageBreak
1376     switch to LuaLaTeX or XeLaTeX or\MessageBreak
1377     try to add \protect\usepackage{caption} or\MessageBreak
1378     ... leave it as it is; reported}%
1379   \fi
1380   \let\FB@makecaption\relax
1381   \let\STD@makecaption\relax
1382 }

```

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.

```

1383 \ifFBunicode
1384 \else
1385   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1386   \DeclareTextCommand{\FBtextellipsis}{PU}{\0040\046}
1387   \DeclareTextCommand{\FBtextellipsis}{PD1}{\203}
1388   \DeclareTextCommandDefault{\FBtextellipsis}{%
1389     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}%
1390   \def\bbl@frenchdots{\babel@save\textellipsis
1391     \let\textellipsis\FBtextellipsis}
1392   \addto\extrasfrench{\bbl@frenchdots}
1393 \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).

```

1394 \ifFB@active@punct
1395   \@ifpackageloaded{listings}
1396     {\AtBeginDocument{%
1397       \FBWarning{Please load the "listings" package\MessageBreak
1398         AFTER babel/french; reported}}%
1399     }{}
1400 \fi

```

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

```

1401 \newif\if@FBwarning@natbib
1402 \ifFB@active@punct
1403   \@ifpackageloaded{natbib}{\@FBwarning@natbibtrue}
1404 \fi
1405 \AtBeginDocument{%
1406   \if@FBwarning@natbib
1407     \@ifpackageloaded{natbib}{\@FBwarning@natbibfalse}%
1408   \fi
1409   \if@FBwarning@natbib
1410     \FBWarning{Please load the "natbib" package\MessageBreak
1411       BEFORE babel/french; reported}%
1412   \fi
1413 }

```

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

```

1414 \newif\if@FBwarning@beamerarticle
1415 \@ifpackageloaded{beamerarticle}{\@FBwarning@beamerarticlettrue}
1416 \AtBeginDocument{%
1417   \if@FBwarning@beamerarticle
1418     \@ifpackageloaded{beamerarticle}{%
1419       {\@FBwarning@beamerarticlefalse}%
1420   \fi
1421   \if@FBwarning@beamerarticle

```

```

1422     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1423             BEFORE babel/french; reported}%
1424   \fi
1425 }

```

2.11 Setup options: keyval stuff

All setup options are handled by command `\frenchsetup{}` using the keyval syntax. A list of flags is defined and set to a default value which will possibly be changed ‘AtEndOfPackage’ if French is the main language. After this, `\frenchsetup{}` eventually modifies the preset values of these flags.

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

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

```

1426 \newcommand*{\frenchsetup}[1]{%
1427   \setkeys{FB}{#1}%
1428 }%
1429 \@onlypreamble\frenchsetup

```

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

```

1430 \let\frenchbsetup\frenchsetup
1431 \@onlypreamble\frenchbsetup

```

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

```

1432 \newif\ifFBShowOptions
1433 \newif\ifFBStandardLayout           \FBStandardLayouttrue
1434 \newif\ifFBGlobalLayoutFrench      \FBGlobalLayoutFrenchtrue
1435 \newif\ifFBReduceListSpacing
1436 \newif\ifFBStandardListSpacing    \FBStandardListSpacingtrue
1437 \newif\ifFBListOldLayout
1438 \newif\ifFBListItemsAsPar
1439 \newif\ifFBCompactItemize
1440 \newif\ifFBStandardItemizeEnv      \FBStandardItemizeEnvtrue
1441 \newif\ifFBStandardEnumerateEnv    \FBStandardEnumerateEnvtrue
1442 \newif\ifFBStandardItemLabels      \FBStandardItemLabelstrue
1443 \newif\ifFBStandardLists           \FBStandardListstrue
1444 \newif\ifFBIndentFirst
1445 \newif\ifFBFrenchFootnotes
1446 \newif\ifFBAutoSpaceFootnotes
1447 \newif\ifFBOriginalTypewriter
1448 \newif\ifFBThinColonSpace
1449 \newif\ifFBThinSpaceInFrenchNumbers
1450 \newif\ifFBFrenchSuperscripts      \FBFrenchSuperscriptstrue

```

```

1451 \newif\ifFBLowercaseSuperscripts    \FBLowercaseSuperscriptstrue
1452 \newif\ifFBPartNameFull            \FBPartNameFulltrue
1453 \newif\ifFBCustomiseFigTabCaptions
1454 \newif\ifFBOldFigTabCaptions
1455 \newif\ifFBSmallCapsFigTabCaptions  \FBSmallCapsFigTabCaptionstrue
1456 \newif\ifFBSuppressWarning
1457 \newif\ifFBINGuillSpace

```

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

```

1458 \ifFB@koma
1459   \ifdefined\partformat
1460     \def\FB@partformat@fix{%
1461       \ifFBPartNameFull
1462         \babel@save\partformat
1463         \renewcommand*{\partformat}{\partname}%
1464       \fi}
1465     \addto\extrasfrench{\FB@partformat@fix}%
1466   \fi
1467 \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.

```

1468 \def\FB@french{french}
1469 \def\FB@acadian{acadian}
1470 \newif\ifFB@mainlanguage@FR
1471 \AtEndOfPackage{%
1472   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1473   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1474   \fi
1475   \ifFB@mainlanguage@FR
1476     \FBGlobalLayoutFrenchtrue
1477     \@ifclassloaded{beamer}%
1478       {\PackageInfo{french.ldf}{%
1479         No list customisation for the beamer class,%
1480         \MessageBreak reported}}%
1481     {\@ifpackageloaded{beamerarticle}%
1482       {\FBStandardItemLabelsfalse
1483        \FBStandardListSpacingfalse
1484        \PackageInfo{french.ldf}{%
1485          Minimal list customisation for the beamerarticle%
1486          \MessageBreak package; reported}}%

```

Otherwise customise lists “à la française”:

```
1487     {\FBStandardListSpacingfalse
1488     \FBStandardItemizeEnvfalse
1489     \FBStandardEnumerateEnvfalse
1490     \FBStandardItemLabelsfalse}%
1491   }
1492   \FBIndentFirsttrue
1493   \FBFrenchFootnotesttrue
1494   \FBAutoSpaceFootnotesttrue
1495   \FBCustomiseFigTabCaptionstrue
1496 \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.

```
1497 \RequirePackage{keyval}%
1498 \define@key{FB}{ShowOptions}[true]%
1499     {\csname FBShowOptions#1\endcsname}%
```

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

```
1500 \define@key{FB}{StandardLayout}[true]%
1501     {\ifFB@mainlanguage@FR
1502     \csname FBStandardLayout#1\endcsname
1503     \else
1504     \PackageWarning{french.ldf}%
1505     {Option `StandardLayout' skipped:\MessageBreak
1506     French is *not* babel's last option.\MessageBreak
1507     Reported}%
1508     \fi
1509     \ifFBStandardLayout
1510     \FBStandardListSpacingtrue
1511     \FBStandardItemizeEnvtrue
1512     \FBStandardItemLabelstrue
1513     \FBStandardEnumerateEnvtrue
1514     \FBIndentFirstfalse
1515     \FBFrenchFootnotesfalse
1516     \FBAutoSpaceFootnotesfalse
1517     \else
1518     \FBStandardListSpacingfalse
1519     \FBStandardItemizeEnvfalse
1520     \FBStandardItemLabelsfalse
1521     \FBStandardEnumerateEnvfalse
1522     \FBIndentFirsttrue
1523     \FBFrenchFootnotesttrue
1524     \FBAutoSpaceFootnotesttrue
1525     \fi}%
1526 \define@key{FB}{GlobalLayoutFrench}[true]%
1527     {\ifFB@mainlanguage@FR
1528     \csname FBGlobalLayoutFrench#1\endcsname
1529     \else
1530     \PackageWarning{french.ldf}%
1531     {Option `GlobalLayoutFrench' skipped:\MessageBreak
1532     French is *not* babel's last option.\MessageBreak
1533     Reported}%
```


1534 \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.

```
1535 \define@key{FB}{ReduceListSpacing}[true]%
1536     {\csname FBReduceListSpacing#1\endcsname
1537     \ifFBReduceListSpacing \FBStandardListSpacingfalse
1538     \else \FBStandardListSpacingtrue\fi
1539     }%
1540 \define@key{FB}{StandardListSpacing}[true]%
1541     {\csname FBStandardListSpacing#1\endcsname}%
1542 \define@key{FB}{ListOldLayout}[true]%
1543     {\csname FBListOldLayout#1\endcsname
1544     \ifFBListOldLayout
1545     \FBStandardEnumerateEnvtrue
1546     \renewcommand*{\FrenchLabelItem}{\textendash}%
1547     \fi}%
1548 \define@key{FB}{CompactItemize}[true]%
1549     {\csname FBCompactItemize#1\endcsname
1550     \ifFBCompactItemize
1551     \FBStandardItemizeEnvfalse
1552     \FBStandardEnumerateEnvfalse
1553     \else
1554     \FBStandardItemizeEnvtrue
1555     \FBStandardEnumerateEnvtrue
1556     \fi}%
1557 \define@key{FB}{StandardItemizeEnv}[true]%
1558     {\csname FBStandardItemizeEnv#1\endcsname}%
1559 \define@key{FB}{StandardEnumerateEnv}[true]%
1560     {\csname FBStandardEnumerateEnv#1\endcsname}%
1561 \define@key{FB}{StandardItemLabels}[true]%
1562     {\csname FBStandardItemLabels#1\endcsname}%
1563 \define@key{FB}{ItemLabels}%
1564     {\renewcommand*{\FrenchLabelItem}{#1}}%
1565 \define@key{FB}{ItemLabeli}%
1566     {\renewcommand*{\Frlabelitemi}{#1}}%
1567 \define@key{FB}{ItemLabelii}%
1568     {\renewcommand*{\Frlabelitemii}{#1}}%
1569 \define@key{FB}{ItemLabeliii}%
1570     {\renewcommand*{\Frlabelitemiii}{#1}}%
1571 \define@key{FB}{ItemLabeliv}%
1572     {\renewcommand*{\Frlabelitemiv}{#1}}%
1573 \define@key{FB}{StandardLists}[true]%
1574     {\csname FBStandardLists#1\endcsname
1575     \ifFBStandardLists
1576     \FBStandardListSpacingtrue
1577     \FBStandardItemizeEnvtrue
1578     \FBStandardEnumerateEnvtrue
1579     \FBStandardItemLabelstrue
1580     \else
1581     \FBStandardListSpacingfalse
1582     \FBStandardItemizeEnvfalse
1583     \FBStandardEnumerateEnvfalse
```

```

1584         \FBStandardItemLabelsfalse
1585         \fi}%
1586 \define@key{FB}{ListItemsAsPar}[true]%
1587         {\csname FBListItemsAsPar#1\endcsname}
1588 \define@key{FB}{IndentFirst}[true]%
1589         {\csname FBIndentFirst#1\endcsname}%
1590 \define@key{FB}{FrenchFootnotes}[true]%
1591         {\csname FBFrenchFootnotes#1\endcsname}%
1592 \define@key{FB}{AutoSpaceFootnotes}[true]%
1593         {\csname FBAutoSpaceFootnotes#1\endcsname}%
1594 \define@key{FB}{AutoSpacePunctuation}[true]%
1595         {\csname FBAutoSpacePunctuation#1\endcsname}%
1596 \define@key{FB}{OriginalTypewriter}[true]%
1597         {\csname FBOriginalTypewriter#1\endcsname}%
1598 \define@key{FB}{ThinColonSpace}[true]%
1599         {\csname FBThinColonSpace#1\endcsname
1600         \ifFBThinColonSpace
1601         \renewcommand*{\FBcolonspace}{\FBthinspace}%
1602         \fi}%
1603 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1604         {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1605 \define@key{FB}{FrenchSuperscripts}[true]%
1606         {\csname FBFrenchSuperscripts#1\endcsname}
1607 \define@key{FB}{LowercaseSuperscripts}[true]%
1608         {\csname FBLowercaseSuperscripts#1\endcsname}
1609 \define@key{FB}{PartNameFull}[true]%
1610         {\csname FBPartNameFull#1\endcsname}%
1611 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1612         {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1613 \define@key{FB}{OldFigTabCaptions}[true]%
1614         {\csname FBOldFigTabCaptions#1\endcsname
1615         \ifFBOldFigTabCaptions
1616         \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1617         \def\FBCaption@Separator{\CaptionSeparator}}%
1618         \addto\extrasfrench{\FB@capsep@fix}%
1619         \ifdefined\extrasacadian
1620         \addto\extrasacadian{\FB@capsep@fix}%
1621         \fi
1622         \fi}%
1623 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1624         {\csname FBSmallCapsFigTabCaptions#1\endcsname
1625         \ifFBSmallCapsFigTabCaptions
1626         \else \let\FBfigtabshape\relax \fi}%
1627 \define@key{FB}{SuppressWarning}[true]%
1628         {\csname FBSuppressWarning#1\endcsname
1629         \ifFBSuppressWarning
1630         \renewcommand{\FBWarning}[1]{}%
1631         \fi}%

```

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

```

1632 \define@key{FB}{INGuillSpace}[true]%
1633         {\csname FBINGuillSpace#1\endcsname
1634         \ifFBINGuillSpace

```

```

1635         \renewcommand*{\FBguillspace}{\space}%
1636         \fi}%
1637 \define@key{FB}{InnerGuillSingle}[true]%
1638         {\csname FBInnerGuillSingle#1\endcsname}%
1639 \define@key{FB}{EveryParGuill}[open]%
1640         {\expandafter\let\expandafter
1641         \FBeveryparguill\csname FBguill#1\endcsname
1642         \ifx\FBeveryparguill\FBguillopen
1643         \else\ifx\FBeveryparguill\FBguillclose
1644         \else\ifx\FBeveryparguill\FBguillnone
1645         \else
1646         \let\FBeveryparguill\FBguillopen
1647         \FBWarning{Wrong value for `EveryParGuill':
1648         try `open',\MessageBreak
1649         `close' or `none'. Reported}%
1650         \fi
1651         \fi
1652         \fi}%
1653 \define@key{FB}{EveryLineGuill}[open]%
1654         {\ifFB@luatex@punct
1655         \expandafter\let\expandafter
1656         \FBeverylineguill\csname FBguill#1\endcsname
1657         \ifx\FBeverylineguill\FBguillopen
1658         \else\ifx\FBeverylineguill\FBguillclose
1659         \else\ifx\FBeverylineguill\FBguillnone
1660         \else
1661         \let\FBeverylineguill\FBguillnone
1662         \FBWarning{Wrong value for `EveryLineGuill':
1663         try `open',\MessageBreak
1664         `close' or `none'. Reported}%
1665         \fi
1666         \fi
1667         \fi
1668         \else
1669         \FBWarning{Option `EveryLineGuill' skipped:%
1670         \MessageBreak this option is for
1671         LuaTeX *only*.\MessageBreak Reported}%
1672         \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.

```

1673 \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1674         {\ifFB@luatex@punct
1675         \csname FBucsNBSP#1\endcsname
1676         \ifFBucsNBSP \FB@ucsNBSP=\@ne \fi
1677         \else
1678         \FBWarning{Option `UnicodeNoBreakSpaces' skipped:%
1679         \MessageBreak this option is for
1680         LuaTeX *only*.\MessageBreak Reported}%
1681         \fi
1682         }%

```

Inputing 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@addGUIILspace` 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.

```
1683 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
1684 \define@key{FB}{og}%
1685     {\ifBunicode
```

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

```
1686     \ifFB@luatex@punct
1687     \FB@addGUIILspace=1 \relax
1688     \fi
```

then with XeTeX it is a bit more tricky:

```
1689     \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).

```
1690     \XeTeXcharclass"13 = \FB@guilo
1691     \XeTeXcharclass"AB = \FB@guilo
1692     \XeTeXcharclass"A0 = \FB@guilnul
1693     \XeTeXcharclass"202F = \FB@guilnul
1694     \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1695     \ifFB@active@punct
1696     \FBWarning{Option og=« not supported with this version
1697         of\MessageBreak LuaTeX/XeTeX; reported}%
1698     \fi
1699     \else
```

This is for conventional TeX engines:

```
1700     \newcommand*\FB@@og{%
1701     \ifFBfrench
1702     \ifFB@spacing\FB@og\ignorespaces
1703     \else\guillemotleft
1704     \fi
1705     \else\guillemotleft\fi}%
1706     \AtBeginDocument{%
1707     \ifdefined\uc@dclc
```

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

```
1708     \uc@dclc{171}{default}{\FB@@og}%
1709     \else
```

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

```
1710         \FB@parse#1\endparse
1711         \ifx\FB@second\@empty
```

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

```
1712         \ifdefined\mule@def
1713         \mule@def{11}{\FB@@og}%
1714         \else
1715         \ifdefined\DeclareInputText
1716         \@tempcnta`#1\relax
1717         \DeclareInputText{\the\@tempcnta}{\FB@@og}%
1718         \else
```

Package inputenc not loaded, no way...

```
1719         \FBWarning{Option `og' requires package
1720                     inputenc;MessageBreak reported}%
1721         \fi
1722         \fi
1723         \else
```

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

```
1724         \DeclareUnicodeCharacter{00AB}{\FB@@og}%
1725         \fi
1726         \fi}%
1727     \fi
1728     }%
```

Same code for the closing quote.

```
1729 \define@key{FB}{fg}%
1730     {\ifFBunicode
1731         \ifFB@luatex@punct
1732         \FB@addGUIlSpace=1 \relax
1733         \fi
1734         \ifFB@xetex@punct
1735         \XeTeXcharclass"14 = \FB@guilf
1736         \XeTeXcharclass"BB = \FB@guilf
1737         \XeTeXcharclass"A0 = \FB@guilnul
1738         \XeTeXcharclass"202F = \FB@guilnul
1739         \fi
1740         \ifFB@active@punct
1741         \FBWarning{Option fg=> not supported with this version
1742                     of\MessageBreak LuaTeX/XeTeX; reported}%
1743         \fi
1744         \else
1745         \newcommand*{\FB@@fg}{%
1746             \ifFBfrench
1747             \ifFB@spacing\FB@fg
1748             \else\guillemotright
1749             \fi
1750             \else\guillemotright\fi}%
1751         \AtBeginDocument{%
1752             \ifdefined\uc@dclc
1753             \uc@dclc{187}{default}{\FB@@fg}%
1754             \else
```

```

1755         \FB@parse#1\endparse
1756         \ifx\FB@second\@empty
1757           \ifdefined\mule@def
1758             \mule@def{27}{\FB@@fg}}%
1759         \else
1760           \ifdefined\DeclareInputText
1761             \@tempcnta`#1\relax
1762             \DeclareInputText{\the\@tempcnta}{\FB@@fg}%
1763           \else
1764             \FBWarning{Option `fg' requires package
1765                       inputenc;\MessageBreak reported}%
1766           \fi
1767         \fi
1768       \else
1769         \DeclareUnicodeCharacter{00BB}{\FB@@fg}%
1770       \fi
1771     \fi}%
1772   \fi
1773   }%
1774 }

```

`\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`.

1775 `\newcommand*{\FBprocess@options}{%`
Update flags if a package customising lists has been loaded, currently: `enumitem`, `paralist`, `enumerate`.

```

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

```

```

1798         {Setting StandardItemizeEnv=true for\MessageBreak
1799         compatibility with paralist package,\MessageBreak
1800         reported}%
1801     \fi
1802     \ifFBStandardEnumerateEnv
1803     \else
1804         \FBStandardEnumerateEnvtrue
1805         \PackageInfo{french.ldf}%
1806         {Setting StandardEnumerateEnv=true for\MessageBreak
1807         compatibility with paralist package,\MessageBreak
1808         reported}%
1809     \fi}{}%
1810 \@ifpackageloaded{enumerate}{%
1811     \ifFBStandardEnumerateEnv
1812     \else
1813         \FBStandardEnumerateEnvtrue
1814         \PackageInfo{french.ldf}%
1815         {Setting StandardEnumerateEnv=true for\MessageBreak
1816         compatibility with enumerate package,\MessageBreak
1817         reported}%
1818     \fi}{}%

```

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

```

1819 \def\FB@ufl{\update@frenchlists}
1820 \ifFB@mainlanguage@FR
1821     \update@frenchlists
1822 \else
1823     \ifFBStandardItemizeEnv
1824     \else
1825         \PackageWarning{french.ldf}%
1826         {babel-french will not customise lists' layout\MessageBreak
1827         when French is not the main language,\MessageBreak
1828         reported}%
1829     \fi
1830 \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.

```

1831 \ifFBAutoSpacePunctuation
1832     \autospace@beforeFDP
1833 \else
1834     \noautospace@beforeFDP
1835 \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.

```

1836 \ifFBOriginalTypewriter
1837 \else

```

```

1838 \let\ttfamilyORI\ttfamily
1839 \let\rmfamilyORI\rmfamily
1840 \let\sffamilyORI\sffamily
1841 \let\ttfamily\ttfamilyFB
1842 \let\rmfamily\rmfamilyFB
1843 \let\sffamily\sffamilyFB
1844 \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.

```

1845 \@ifpackageloaded{numprint}%
1846   {\ifnprt@autolanguage
1847     \providecommand*\npstylefrench{}%
1848     \ifFBThinSpaceInFrenchNumbers
1849       \renewcommand*\FBthousandsep{\,}%
1850     \fi
1851     \g@addto@macro\npstylefrench{\npthousandsep\FBthousandsep}%
1852   \fi
1853 }%

```

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.

```

1854 \ifFBFrenchSuperscripts
1855   \DeclareRobustCommand*\up*{%
1856     \texorpdfstring{\@ifstar{\FB@up@fake}{\fup}}{}%
1857   }
1858 \else
1859   \DeclareRobustCommand*\up*{%
1860     \texorpdfstring{\@ifstar{\FB@up@fake}{\textsuperscript}}{}%
1861   }
1862 \fi

```

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

```

1863 \ifFBLowercaseSuperscripts
1864 \else
1865   \renewcommand*\FB@lc[1]{##1}%
1866 \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`.

```

1867 \ifFB@koma
1868   \ifx\captionformat\FB@std@capsep
1869     \ifFBCustomiseFigTabCaptions
1870       \renewcommand*\captionformat{\CaptionSeparator}%
1871     \else
1872       \renewcommand*\captionformat{{\autospace@beforeFDP :\ }}%
1873     \fi
1874 \fi

```



```

1875 \fi
1876 \@ifclassloaded{memoir}%
1877   {\ifx\@contdelim\FB@std@capsep
1878     \ifFBCustomiseFigTabCaptions
1879       \captiondelim{\CaptionSeparator}%
1880     \else
1881       \captiondelim{{\autospace@beforeFDP : }}%
1882     \fi
1883   \fi}{}%
1884 \@ifclassloaded{beamer}%
1885   {\protected@edef\FB@capsep{%
1886     \csname beamer@tmpl@caption label separator\endcsname}%
1887   \ifx\FB@capsep\FB@std@capsep
1888     \ifFBCustomiseFigTabCaptions
1889       \defbeamertemplate{caption label separator}{FBcustom}{%
1890         \CaptionSeparator}%
1891     \setbeamertemplate{caption label separator}[FBcustom]%
1892   \else
1893     \defbeamertemplate{caption label separator}{FBcolon}{%
1894       {\autospace@beforeFDP : }}%
1895     \setbeamertemplate{caption label separator}[FBcolon]%
1896   \fi
1897   \fi}{}%

```

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

```

1898 \ifFBShowOptions
1899   \GenericWarning{* }{%
1900     *** List of possible options for babel-french ***\MessageBreak
1901     [Default values between brackets when french is loaded *LAST*]%
1902     \MessageBreak
1903     ShowOptions [false]\MessageBreak
1904     StandardLayout [false]\MessageBreak
1905     GlobalLayoutFrench [true]\MessageBreak
1906     PartNameFull [true]\MessageBreak
1907     IndentFirst [true]\MessageBreak
1908     ListItemsAsPar [false]\MessageBreak
1909     StandardListSpacing [false]\MessageBreak
1910     StandardItemizeEnv [false]\MessageBreak
1911     StandardEnumerateEnv [false]\MessageBreak
1912     StandardItemLabels [false]\MessageBreak
1913     ItemLabels=\textendash, \textbullet,
1914     \protect\ding{43},... [\textendash]\MessageBreak
1915     ItemLabeli=\textendash, \textbullet,
1916     \protect\ding{43},... [\textendash]\MessageBreak
1917     ItemLabelii=\textendash, \textbullet,
1918     \protect\ding{43},... [\textendash]\MessageBreak
1919     ItemLabeliii=\textendash, \textbullet,
1920     \protect\ding{43},... [\textendash]\MessageBreak
1921     ItemLabeliv=\textendash, \textbullet,
1922     \protect\ding{43},... [\textendash]\MessageBreak
1923     StandardLists [false]\MessageBreak
1924     ListOldLayout [false]\MessageBreak
1925     FrenchFootnotes [true]\MessageBreak
1926     AutoSpaceFootnotes [true]\MessageBreak

```

```

1927     AutoSpacePunctuation [true]\MessageBreak
1928     ThinColonSpace [false]\MessageBreak
1929     OriginalTypewriter [false]\MessageBreak
1930     UnicodeNoBreakSpaces [false]\MessageBreak
1931     og= <left quote character>, fg= <right quote character>%
1932     INGuillSpace [false]\MessageBreak
1933     EveryParGuill=open, close, none [open]\MessageBreak
1934     EveryLineGuill=open, close, none
1935             [open in LuaTeX, none otherwise]\MessageBreak
1936     InnerGuillSingle [false]\MessageBreak
1937     ThinSpaceInFrenchNumbers [false]\MessageBreak
1938     SmallCapsFigTabCaptions [true]\MessageBreak
1939     CustomiseFigTabCaptions [true]\MessageBreak
1940     OldFigTabCaptions [false]\MessageBreak
1941     FrenchSuperscripts [true]\MessageBreak
1942     LowercaseSuperscripts [true]\MessageBreak
1943     SuppressWarning [false]\MessageBreak
1944     \MessageBreak
1945     *****%
1946     \MessageBreak\protect\frenchsetup{ShowOptions}}
1947 \fi
1948 }

```

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.

```

1949 \AtBeginDocument{%
1950   \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`.

```

1951   \FBprocess@options

```

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

```

1952   \ifBucsNBS
1953     \renewcommand*\FBmedkern{\char"202F\relax}%
1954     \renewcommand*\FBthickkern{\char"A0\relax}%
1955     \ifFBThinSpaceInFrenchNumbers
1956       \renewcommand*\FBthousandsep{\char"202F\relax}%
1957     \else
1958       \renewcommand*\FBthousandsep{\char"A0\relax}%
1959     \fi
1960 \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*`!

```

1961 \begingroup
1962   \newcommand{\FBOTone}{OT1}%
1963   \ifx\encodingdefault\FBOTone
1964     \FBWarning{OT1 encoding should not be used for French.%
1965     \MessageBreak

```

```

1966          Add \protect\usepackage[T1]{fontenc} to the
1967          preamble\MessageBreak of your document; reported}%
1968      \fi
1969  \endgroup
1970 }

```

2.12 French lists

`\listFB` Vertical spacing in lists should be shorter in French texts than the defaults provided by LaTeX. Note that the easy way, just changing values of vertical spacing parameters when entering French and restoring them to their defaults on exit would not work; 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.

```

1971 \let\listORI\list
1972 \let\endlistORI\endlist
1973 \newdimen\FB@parskip
1974 \def\FB@listVsettings{%
1975     \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1976     \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`.

```

1977     \FB@parskip=\parskip
1978     \addtolength{\topsep}{-\FB@parskip}%
1979     \addtolength{\partopsep}{\FB@parskip}%
1980     \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1981     \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.

```

1982     \ifdim\FB@parskip>0pt
1983         \setlength{\parsep}{\FB@parskip}%
1984         \addtolength{\itemsep}{-\FB@parskip}%
1985     \fi
1986 }
1987 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1988 \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):

```

\FrenchLabelItem 1989 \newcommand*{\FrenchLabelItem}{\textemdash}
\FrenchLabelItem 1990 \newcommand*{\FrenchLabelItemii}{\FrenchLabelItem}
\FrenchLabelItem 1991 \newcommand*{\FrenchLabelItemiii}{\FrenchLabelItem}
\FrenchLabelItem 1992 \newcommand*{\FrenchLabelItemiv}{\FrenchLabelItem}
\FrenchLabelItem 1993 \newcommand*{\FrenchLabelItemv}{\FrenchLabelItem}

```

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

```

1994 \newdimen\listindentFB
1995 \setlength{\listindentFB}{-1pt}
1996 \newdimen\descindentFB
1997 \setlength{\descindentFB}{-1pt}
1998 \newdimen\labelindentFB
1999 \setlength{\labelindentFB}{-1pt}
2000 \newdimen\labelwidthFB
2001 \setlength{\labelwidthFB}{-1pt}

```

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

```

2002 \newdimen\leftmarginFB
2003 \def\FB@listHsettings{%
2004   \ifFBListItemsAsPar

```

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

```

2005   \itemindent=\labelindentFB
2006   \advance\itemindent by \labelwidthFB
2007   \advance\itemindent by \labelsep
2008   \leftmargin\z@
2009   \bbl@for\FB@dp {2, 3, 4, 5, 6}%
2010     {\csname leftmargin\romannumeral\FB@dp\endcsname =
2011       \labelindentFB}%
2012   \else

```

Default layout: labels hanging into the list left margin.

```

2013   \leftmarginFB=\labelwidthFB
2014   \advance\leftmarginFB by \labelsep
2015   \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
2016     {\csname leftmargin\romannumeral\FB@dp\endcsname =
2017       \leftmarginFB}%
2018   \advance\leftmarginFB by \listindentFB

```

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

```
2019 \listparindent=\parindent
2020 \fi
2021 \leftmargin=\csname leftmargin%
2022 \ifnum \@listdepth=\@ne i\else ii\fi\endcsname
2023 }
```

`\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).

```
2024 \def\FB@itemizesettings{%
2025   \ifFBStandardListSpacing
2026   \else
2027     \setlength{\topsep}{\z@}%
2028     \setlength{\partopsep}{\z@}%
2029     \FB@parskip=\parskip
2030     \addtolength{\topsep}{-\FB@parskip}%
2031     \addtolength{\partopsep}{\FB@parskip}%
2032     \setlength{\itemsep}{\z@}%
2033     \setlength{\parsep}{\z@}%
2034     \ifdim\FB@parskip>0pt
2035       \setlength{\parsep}{\FB@parskip}%
2036       \addtolength{\itemsep}{-\FB@parskip}%
2037     \fi
2038   \fi
2039   \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
2040   \ifFBListOldLayout
2041     \setlength{\leftmargin}{\labelwidth}%
2042     \addtolength{\leftmargin}{\labelsep}%
2043     \addtolength{\leftmargin}{\parindent}%
2044   \else
2045     \FB@listHsettings
2046   \fi
2047 }
```

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

```
2048 \def\itemizeFB{%
2049   \ifnum \@itemdepth >\thr@@\@toodeep\else
2050     \advance\@itemdepth by \@ne
2051     \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
2052     \expandafter
2053     \listORI
2054     \csname\@itemitem\endcsname
2055     \FB@itemizesettings
2056   \fi
2057 }
2058 \let\enditemizeFB\endlistORI

2059 \def\setlabelitemsFB{%
2060   \let\labelitemi\Frlabelitemi
```

```

2061 \let\labelitemii\Frlabelitemii
2062 \let\labelitemiii\Frlabelitemiii
2063 \let\labelitemiv\Frlabelitemiv
2064 \ifdim\labelwidthFB<\z@
2065   \settowidth{\labelwidthFB}{\FrenchLabelItem}%
2066 \fi
2067 }
2068 \def\setlistindentFB{%
2069   \ifdim\labelindentFB<\z@
2070     \ifdim\parindent=\z@
2071       \setlength{\labelindentFB}{1.5em}%
2072     \else
2073       \setlength{\labelindentFB}{\parindent}%
2074     \fi
2075 \fi
2076 \ifdim\listindentFB<\z@
2077   \ifdim\parindent=\z@
2078     \setlength{\listindentFB}{1.5em}%
2079   \else
2080     \setlength{\listindentFB}{\parindent}%
2081   \fi
2082 \fi
2083 \ifdim\descindentFB<\z@
2084   \ifFBListItemsAsPar
2085     \setlength{\descindentFB}{\labelindentFB}%
2086   \else
2087     \setlength{\descindentFB}{\listindentFB}%
2088   \fi
2089 \fi
2090 }

```

`\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`.

```

2091 \def\enumerateFB{%
2092   \ifnum \@enumdepth >\thr@@\@toodeep\else
2093     \advance\@enumdepth by \@ne
2094     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2095     \expandafter
2096     \list
2097       \csname label\@enumctr\endcsname
2098       {\FB@listHsettings
2099         \usecounter{\@enumctr}\def\makelabel##1{\hss\llap{##1}}}%
2100   \fi
2101 }
2102 \let\endenumerateFB\endlistORI

```

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

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

```

2103 \def\descriptionFB{%
2104     \list{}\FB@listHsettings
2105         \labelwidth=\z@
2106         \ifFBListItemsAsPar
2107             \itemindent=\descindentFB
2108         \else
2109             \itemindent=-\leftmargin
2110             \ifnum\@listdepth=\@ne
2111                 \ifdim\descindentFB=\z@
2112                     \ifdim\listindentFB>\z@
2113                         \leftmargini=\listindentFB
2114                         \leftmargin=\leftmargini
2115                         \itemindent=-\leftmargin
2116                     \fi
2117                 \else
2118                     \advance\itemindent by \descindentFB
2119                 \fi
2120             \fi
2121         \fi
2122         \let\makelabel\descriptionlabel}%
2123 }
2124 \let\enddescriptionFB\endlistORI

```

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

```

2125 \def\update@frenchlists{%
2126     \setlistindentFB
2127     \ifFBStandardListSpacing
2128     \else \let\list\listFB \fi
2129     \ifFBStandardItemizeEnv
2130     \else \let\itemize\itemizeFB \fi
2131     \ifFBStandardItemLabels
2132     \else \setlabelitemsFB \fi
2133     \ifFBStandardEnumerateEnv
2134     \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2135 }

```

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`.

```

2136 \def\FB@ufl{\relax}
2137 \def\bb@frenchlistlayout{%
2138     \ifFBGlobalLayoutFrench

```

```

2139 \else
2140 \babel@save\list \babel@save\itemize
2141 \babel@save\enumerate \babel@save\description
2142 \babel@save\labelitemi \babel@save\labelitemii
2143 \babel@save\labelitemiii \babel@save\labelitemiv
2144 \FB@ufl
2145 \fi
2146 }
2147 \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.

```

2148 \def\bbl@frenchindent{%
2149 \ifFBGlobalLayoutFrench\else\babel@save\@afterindentfalse\fi
2150 \ifBIndentFirst
2151 \ifFB@mainlanguage@FR\else\babel@save\@afterindentfalse\fi
2152 \let\@afterindentfalse\@afterindenttrue
2153 \@afterindenttrue
2154 \fi}
2155 \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 not customise footnotes at all. When the `footnotebackref` package is loaded, `babel-french` will not customise `\@footnotetext` in order to keep back referencing working.

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`.

```

2156 \AtBeginDocument{%
2157 \ifpackageloaded{bigfoot}%
2158 \PackageWarning{french.1df}%
2159 {bigfoot package in use.\MessageBreak
2160 babel-french will NOT customise footnotes;%
2161 \MessageBreak reported}}%
2162 {\let\@footnotemarkORI\@footnotemark
2163 \def\@footnotemarkFB{\leavevmode\unskip\unkern

```



```

2164                                     \, \@footnotemarkORI}%
2165     \ifFBAutoSpaceFootnotes
2166         \let\@footnotemark\@footnotemarkFB
2167     \fi}%
2168 \@ifpackageloaded{footnotebackref}%
2169     {\FBFrenchFootnotesfalse
2170     \PackageWarning{french. ldf}%
2171     {footnotebackref package loaded.\MessageBreak
2172     babel-french will NOT customise footnotes;%
2173     \MessageBreak reported}}%
2174     {}%
2175 }

```

\@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).

```

2176 \newdimen\parindentFFN
2177 \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.

```

2178 \newcommand*\dotFFN}{.}
2179 \newcommand*\kernFFN}{\kern .5em}
2180 \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.

```

2181 \ifFB@koma
2182     \let\@makefntextORI\@makefntext
2183     \let\@makefnmarkORI\@makefnmark

```

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

```

2184 \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
2185     {\thefootnotemark\dotFFN\kernFFN}
2186 \let\@makefntextFB\@makefntext
2187 \let\@makefnmarkFB\@makefnmark

```

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

```

2188 \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2189     {\textsuperscript{\thefootnotemark}}

```

```

2190 \let\@makefnctextTH\@makefnctext
2191 \let\@@makefnmarkTH\@@makefnmark

```

Restore the original definitions.

```

2192 \let\@makefnctext\@makefnctextORI
2193 \let\@@makefnmark\@@makefnmarkORI
2194 \fi

```

Definitions for the memoir class:

```
2195 \@ifclassloaded{memoir}
```

(see original definition in memman.pdf)

```

2196 {\newcommand{\@makefnctextFB}[1]{%
2197   \def\footscript##1{##1\dotFFN\kernFFN}%
2198   \setlength{\footmarkwidth}{\FBfnindent}%
2199   \setlength{\footmarksep}{-\footmarkwidth}%
2200   \setlength{\footparindent}{\parindentFFN}%
2201   \makefootmark #1}%
2202 }{}

```

Definitions for the beamer class:

```
2203 \@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.

```

2204 {\def\@makefnctextFB#1{%
2205   \def\insertfootnotetext{#1}%
2206   \def\insertfootnotemark{\insertfootnotemarkFB}%
2207   \usebeamertemplate***{footnote}}%
2208 \def\insertfootnotemarkFB{%
2209   \usebeamercolor[fg]{footnote mark}%
2210   \usebeamerfont*{footnote mark}%
2211   \llap{\@thefnmark}\dotFFN\kernFFN}%
2212 }{}

```

Now the default definition of \@makefnctextFB 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)!

```

2213 \providecommand*\insertfootnotemarkFB{%
2214   \parindent=\parindentFFN
2215   \rule\z@\footnotesep
2216   \setbox\@tempboxa\hbox{\@thefnmark}%
2217   \ifdim\wd\@tempboxa>\z@
2218     \llap{\@thefnmark}\dotFFN\kernFFN
2219   \fi}
2220 \providecommand\@makefnctextFB[1]{\insertfootnotemarkFB #1}

```

The rest of \@makefnctext's customisation is done at the \begin{document}. We save the original definition of \@makefnctext, and then redefine \@makefnctext 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.

```

2221 \providecommand\localleftbox[1]{}
2222 \AtBeginDocument{%
2223   \@ifpackageloaded{bigfoot}{}%
2224   {\ifdim\parindentFFN<10in
2225     \else
2226     \parindentFFN=\parindent
2227     \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2228     \fi
2229     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2230     \addtolength{\FBfnindent}{\parindentFFN}%
2231     \let\@makefntextORI\@makefntext
2232     \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).

```

2233     \let\@makefnmarkORI\@makefnmark
2234     \long\def\@makefntext#1{%
2235       \localleftbox{}%
2236       \let\FBeverypar@save\FBeverypar@quote
2237       \let\FBeverypar@quote\relax
2238       \ifFBFrenchFootnotes
2239         \ifx\footnote\thanks
2240           \let\@makefnmark\@makefnmarkTH
2241           \@makefntextTH{#1}
2242         \else
2243           \let\@makefnmark\@makefnmarkFB
2244           \@makefntextFB{#1}
2245         \fi
2246       \else
2247         \let\@makefnmark\@makefnmarkORI
2248         \@makefntextORI{#1}%
2249       \fi
2250       \let\FBeverypar@quote\FBeverypar@save
2251       \localleftbox{\FBeveryline@quote}}%
2252     \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.

```

2253     \@ifclassloaded{memoir}%
2254     {\ifFBFrenchFootnotes
2255       \setlength{\thanksmarkwidth}{\parindentFFN}%
2256       \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2257       \fi
2258     }{}%

```

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

```

2259     \@ifclassloaded{beamer}%
2260     {\ifFBFrenchFootnotes
2261       \ifdim\parindentFFN=1.5em\else
2262         \FBWarning{%
2263           \protect\parindentFFN\space is ineffective%

```

```

2264             \MessageBreak within the beamer class.%
2265             \MessageBreak Reported}%
2266         \fi
2267     \fi
2268 }{}%

```

Definition of \@makefntext for all other classes:

```

2269     \long\def\@makefntext#1{%
2270         \llocalleftbox{}%
2271         \let\FBeverypar@save\FBeverypar@quote
2272         \let\FBeverypar@quote\relax
2273         \ifFBFrenchFootnotes
2274             \@makefntextFB{#1}%
2275         \else
2276             \@makefntextORI{#1}%
2277         \fi
2278         \let\FBeverypar@quote\FBeverypar@save
2279         \llocalleftbox{\FBeverypar@quote}%
2280     \fi
2281 }%
2282 }

```

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

```

2283 \newcommand*\AddThinSpaceBeforeFootnotes{\FBAutoSpaceFootnotestruer}
2284 \newcommand*\FrenchFootnotes{\FBFrenchFootnotestruer}
2285 \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.

```

2286 \FBclean@on@exit
2287 \ldf@finish\CurrentOption
2288 \let\loadlocalcfg\FB@llc
2289 </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`.

```

2290 <*acadian>
2291 \PackageInfo{acadian.ldf}%
2292 {'acadian' dialect is currently\MessageBreak

```

```

2293  *absolutely identical* to the\MessageBreak
2294  `french' language; reported}
2295 </acadian>
2296 <*canadien>
2297 \PackageWarning{canadien.ldf}%
2298 {Option `canadien' for Babel is *deprecated*,\MessageBreak
2299  it might be removed sooner or later. Please\MessageBreak
2300  use `acadian' instead; reported}%
2301 \def\CurrentOption{acadian}

2302 \def\datecanadien{\dateacadian}
2303 \def\captionscanadien{\captionacadian}
2304 \def\extrascanadien{\extrasacadian}
2305 \def\noextrascanadien{\noextrasacadian}
2306 </canadien>
2307 <*français>
2308 \PackageWarning{français.ldf}%
2309 {Option `français' for Babel is *deprecated*,\MessageBreak
2310  it might be removed sooner or later. Please\MessageBreak
2311  use `french' instead; reported}%
2312 \chardef\l@français\l@french
2313 \def\CurrentOption{french}
2314 </français>

```

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

```

2315 <*frenchb>
2316 \def\bbl@tempa{frenchb}
2317 \ifx\CurrentOption\bbl@tempa
2318  \chardef\l@frenchb\l@french
2319  \def\CurrentOption{french}
2320  \PackageWarning{babel-french}%
2321  {Option `frenchb' for Babel is *deprecated*,\MessageBreak
2322   it might be removed sooner or later. Please\MessageBreak
2323   use `french' instead; reported}
2324 \else
2325  \def\bbl@tempa{français}
2326  \ifx\CurrentOption\bbl@tempa
2327   \chardef\l@français\l@french
2328  \def\CurrentOption{french}

```

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

```

2329  \ifx\magnification\@undefined
2330  \PackageWarning{babel-french}%
2331  {Option `français' for Babel is *deprecated*,\MessageBreak
2332   it might be removed sooner or later. Please\MessageBreak
2333   use `french' instead; reported}
2334  \fi
2335 \else
2336  \def\bbl@tempa{canadien}
2337  \ifx\CurrentOption\bbl@tempa
2338  \def\CurrentOption{acadian}
2339  \PackageWarning{babel-french}%
2340  {Option `canadien' for Babel is *deprecated*,\MessageBreak

```

```
2341         it might be removed sooner or later. Please\MessageBreak
2342         use `acadian' instead; reported}
2343     \fi
2344 \fi
2345 \fi
2346 </frenchb>
2347 <acadian|canadien|frenchb|français>\input french.ldf\relax
2348 <acadian|canadien>\let\extrasacadian\extrasfrench
2349 <acadian|canadien>\let\noextrasacadian\noextrasfrench
2350 <acadian|canadien|frenchb|français|french>\endinput
```

3 Change History

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

v3.5s	General: Footnotes: no customising of <code>\@footnotetext</code> when the <code>footnotebackref</code> package is loaded. Just warn the user.	72	default definition is changed in French.	53
	<code>frenchb.lua</code> : A ‘:’ followed by ‘-’ or a ligature should not trigger spacing.	25		
v3.5r	General: Compatibility with <code>ucharclasses</code> package added. . .	30		
v3.5q	<code>\listFB</code> : Bug correction: <code>\parsep</code> should be related to <code>\parskip</code> and <code>\listparindent</code> to <code>\parindent</code> . .	67		
v3.5p	<code>\DecimalMathComma</code> : <code>\DecimalMathComma</code> can again be used in the preamble for a global action. It now works as expected inside a group.	45		
	<code>\frquote</code> : <code>\FBeveryline@quote</code> : no need for a penalty inside a <code>\lcalleftbox</code>	39		
v3.5o	General: <code>\shorthandon</code> and <code>\shorthandoff</code> are no longer redefined in LuaTeX (it broke <code>\shorthandoff*</code>).	29		
	<code>\FB@xetex@punct@french</code> : <code>\shorthandon</code> and <code>\shorthandoff</code> are no longer redefined (it broke <code>\shorthandoff*</code>).	31		
	<code>frenchb.lua</code> : Opening guill.: look ahead when next is a penalty (nobreak space).	27		
v3.5n	<code>\bbl@frenchindent</code> : <code>\bbl@frenchindent</code> changed. <code>\bbl@nonfrenchindent</code> removed.	72		
	<code>\bsc</code> : Added command <code>\bname</code> (no small caps).	44		
	<code>\frenchsetup</code> : <code>\FBGlobalLayoutFrench</code> no longer set to false when French is not the main language.	55		
v3.5m	<code>\FBtextellipsis</code> : No longer redefine <code>\dots</code> , only <code>\textellipsis’s</code>			
v3.5l	General: No warning about <code>\@makecaption</code> for more classes.	51		
	<code>\captionfrench</code> : Redefine <code>\fnum@figure</code> and <code>\fnum@table</code> separately.	48		
v3.5k	General: <code>\degre</code> , <code>\degres</code> , <code>\circonflexe</code> , <code>\tild</code> , <code>\boi</code> and <code>\at</code> are now safe in bookmarks. .	44		
	<code>\pdfstringdefDisableCommands</code> dropped.	66		
	Reorganise warnings about ‘:’ in captions, according to enhancements in <code>caption.sty v3.5a</code> .	52		
	<code>\bsc</code> : <code>\bsc</code> now relies on <code>\texorpdfstring</code> to be safe in bookmarks.	44		
	<code>\captionfrench</code> : Small caps removed in <code>\figurename</code> and <code>\tablename</code> , use <code>\fnum@figure</code> and <code>\fnum@table</code> instead.	48		
	<code>\FB@fg</code> : <code>\FB@og</code> and <code>\FB@fg</code> now rely on <code>\texorpdfstring</code> to be safe in bookmarks.	37		
	<code>\frquote</code> : <code>\frquote</code> now relies on <code>\texorpdfstring</code> to be safe in bookmarks.	39		
	<code>\fup</code> : <code>\up</code> and <code>\fup</code> now rely on <code>\texorpdfstring</code> to be safe in bookmarks.	41		
	<code>\no</code> : <code>\no</code> , <code>\nos</code> , <code>\No</code> , <code>\Nos</code> , <code>\primo</code> , <code>\fprimo</code> , now rely on <code>\texorpdfstring</code> to be safe in bookmarks.	43		
v3.5j	General: For memoir, koma-script and beamer captions, <code>\FB@std@sep</code> has to be defined before activating the colon.	33		
v3.5i	<code>\FBprocess@options</code> : For memoir, koma-script and beamer classes, leave caption delimiter unchanged if it has been user customised. . .	64		
v3.5h	<code>frenchb.lua</code> : 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	Needed by <code>\frquote</code>	74
Code reorganised for better efficiency.	24	<code>\frquote</code> : New command	
v3.5g		<code>\FB@addquote@everypar</code> to manage <code>\frquote</code> failed when used immediately after a sectionning command.	38
<code>frenchb.lua</code> : The kerning callback is a bit specific: adding code with <code>add_to_callback</code> actually deletes the legacy kerning as pointed out by Marcel Krüger on SE.	24	v3.5a	
v3.5f		General: New optional layout for lists: lists' items can be typeset as paragraphs with indented labels while the default leaves the labels hanging into the left margin.	68
General: <code>\l@canadien</code> was defined too early in file 'canadien.ldf': <code>\l@acadian</code> might not be defined.	15	<code>\descriptionFB</code> : <code>ListItemsAsPar</code> option taken into account for description lists.	70
<code>\selectlanguage{canadien}</code> allowed again only for backward compatibility (deprecated).	77	<code>\frenchsetup</code> : New option <code>ListItemsAsPar</code> for displaying lists' items "as paragraphs".	54
<code>\DecimalMathComma</code> : Fixed bug with the acadian language. Warning added if used with the <code>icomma</code> package.	45	v3.4d	
v3.5e		<code>\frenchsetup</code> : New test for deciding about utf8 encoding for keys <code>og</code> and <code>fg</code> (the former one fails with LaTeX 2018 release).	60
<code>\frenchsetup</code> : <code>StandardLayout</code> and <code>GlobalLayoutFrench</code> options can no longer be toggled when French is not the main language.	55	v3.4c	
<code>\frquote</code> : Make resettings global on exit.	40	<code>\ifFBXeTeX</code> : Reverting to former test, beware of <code>\XeTeXrevision</code> left as <code>\relax</code> by careless testing.	16
new command <code>\NoEveryParQuote</code>	40	v3.4b	
reset <code>\FB@addGUILspace</code> attribute inside <code>\localleftbox</code> (LuaTeX). .	39	<code>\datefrench</code> : Do not redefine <code>\date</code> as <code>\frenchdate</code> in French.	40
v3.5d		v3.4a	
<code>\frenchsetup</code> : <code>ReduceListSpacing</code> option deprecated: see <code>StandardListSpacing</code>	54	General: <code>\LdfInit</code> checks	
v3.5c		<code>\FBclean@on@exit</code> instead of <code>\captionsfrench</code> (undefined in PLain). Prevents loading <code>french.ldf</code> again with <code>acadian</code> option.	14
General: Remove grouping inside <code>\@makefn</code> text, <code>\localleftbox</code> and <code>\FBeverypar@quote</code> saved and restored instead.	74	<code>babel-french</code> now requires eTeX. . .	14
<code>\frquote</code> : <code>\FBeverypar@quote</code> 's value now properly reset across level changes.	39	Lua function <code>token.get_meaning</code> requires LuaTeX 1.0.	21
<code>\noextrasfrench</code> : <code>\lccode</code> of quote <code>0x27</code> changed from <code>0x2019</code> to <code>0x27</code> for Unicode engines.	16	New <code>\FBgspchar</code> to customise the space character to be used for <code>\og</code> and <code>\fg</code> with the <code>UnicodeNoBreakSpaces</code> option. . .	37
v3.5b		New attribute <code>\FB@dialect</code> for the French dialect <code>acadian</code>	20
General: Reset <code>\FBeverypar@quote</code> locally inside <code>\@makefn</code> text.		New command <code>\FBsetspaces</code> to fine tune spacing independently in French and in French dialects. . .	18
		<code>Shrink/stretch</code> removed in <code>\FBthousandsep</code>	48
		Toks <code>\FBcolonsp</code> , <code>\FBthinsp</code> and <code>\FBguillsp</code> removed.	18

<code>\datefrench</code> : Specific code for Plain finally removed (babel bug reported).	40	<code>\iflanguage</code> test which is based on patterns.	16
<code>\extrasfrench</code> : Change <code>\(no)extras\CurrentOption</code> to <code>\(no)extrasfrench</code> . <code>\(no)extrasacadian</code> will be defined as <code>\(no)extrasfrench</code> in file <code>acadian.ldf</code>	16	v3.3a General: Compatibility code for pre 2015/10/01 LaTeX release removed, see <code>ltnews23.tex</code>	20
<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	Skip <code>\FBguillskip</code> for LuaTeX replaced by <code>\FBguillsp</code>	18
<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>\captionsfrench</code> : Commands <code>\frenchpartfirst</code> , <code>\frenchpartsecond</code> and <code>\frenchpartnameord</code> added.	48
v3.3d <code>frenchb.lua</code> : In default mode, for ‘:’ only, check if next node is a glyph or not. If it is, turn the ‘ <code>auto</code> ’ flag to false (avoids spurious spaces in URLs, MSDOS paths or 10:35).	25	<code>\FBthinspace</code> : Skips <code>\FBcolonskip</code> and <code>\FBthinskip</code> replaced by <code>\FBcolonsp</code> and <code>\FBthinsp</code>	17
v3.3c General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, <code>fontspec</code> is no longer required.	66	<code>\frenchsetup</code> : <code>\frenchsetup</code> is now an alias for <code>\frenchsetup</code>	54
New command <code>\FBthousandsep</code> to customise numprint.	48	Options <code>INGuillSpace</code> , <code>ThinColonSpace</code> no longer delayed <code>AtBeginDocument</code>	54
New configurable kerns <code>\FBmedkern</code> , and <code>\FBthickkern</code> suitable for HTML translation.	43	<code>\frquote</code> : <code>\FB@quotespace</code> (kern), changed into <code>\FB@guillspace</code>	39
Reorganise warnings when the caption, subcaption or floatrow packages are loaded before <code>babel/french</code>	52	v3.2h <code>\@makefnmark</code> : With <code>beamer.cls</code> , add <code>\llap</code> to <code>\@thefnmark</code> for notes numbered over 99.	74
Reset <code>\localleftbox</code> locally inside <code>\@makefnmark</code> . Needed by <code>\frquote</code> with LuaTeX.	74	<code>\bbl@frenchlistlayout</code> : Execute <code>\update@frenchlists</code> only if <code>GlobalLayoutFrench</code> is false. Delete stuff for lists in <code>\noextrasfrench</code>	71
<code>\frenchsetup</code> : New option ‘ <code>UnicodeNoBreakSpaces</code> ’ for html translators (LuaLaTeX only).	59	<code>\frenchsetup</code> : Option <code>GlobalLayoutFrench</code> skipped when French is not the main language.	55
<code>frenchb.lua</code> : Function ‘ <code>get_glue</code> ’ robustified. ‘ <code>french_punctuation</code> ’ can insert Unicode characters instead of glues.	22	v3.2g General: Changed Unicode definition of <code>\boi</code>	44
v3.3b General: Generate portmanteau files <code>acadian.ldf</code> , <code>canadien.ldf</code> , <code>frenchb.ldf</code> , and <code>français.ldf</code> and warn about deprecated options.	76	<code>fontspec</code> defines TU encoding now and no longer loads <code>xunicode.sty</code> . Test changed.	66
New ‘ <code>if</code> ’ <code>\ifFBfrench</code> to replace		Issue a warning if <code>beamerarticle.sty</code> is loaded after <code>babel</code>	53
		<code>\frenchsetup</code> : Minimal list customisation when <code>beamerarticle.sty</code> is loaded.	55
		Warn when wrong values are provided to options <code>EveryParGuill</code> or <code>EveryLineGuill</code>	58
		<code>\frquote</code> : Default options of <code>\frquote</code> are no longer engine-dependent.	38
		v3.2f <code>\DecimalMathComma</code> : Fixed conflict with the <code>icomma</code> package.	45

v3.2e	ones.	17
General: Add missing redefinitions for <code>\leftmarginv</code> , <code>\leftmarginvi</code> . Suggested by J.F. Burnol.	<code>\NoAutoSpacing</code> : <code>\NoAutoSpacing</code> made robust.	36
<code>\DecimalMathComma</code> : <code>\DecimalMathComma</code> didn't work with LuaTeX. Fixed now.	frenchb.lua: glue_spec removed; starting with LuaTeX 0.95, glue specifications fit in glue.	24
v3.2d	v3.2a	
<code>\descriptionFB</code> : Changed <code>\listindentFB</code> to <code>\descindentFB</code> which defaults to <code>\listindentFB</code> . <code>\leftmargini</code> reduced when <code>\descindentFB</code> is null.	<code>\@makefnctextFB</code> : beamer.cls requires a specific definition of <code>\@makefnctextFB</code> (pointed out by DB). The same is true for memoir and koma-script classes (done). .	73
v3.2c	<code>\fg</code> : <code>\xspace</code> moved from <code>\FB@fg</code> to <code>\fg</code> : <code>\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
General: New LuaTeX attribute <code>\FB@spacing</code>	v3.1m	
Newif <code>\ifFB@spacing</code> and new commands <code>\FB@spacingon</code> , <code>\FB@spacingoff</code> to control space tuning in French.	frenchb.lua: new <code>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
Switch <code>\ifFB@spacing</code> added to the four French shorthands.	v3.1l	
<code>\FB@xetex@punct@french</code> : Switch <code>\ifFB@spacing</code> added to all <code>\XeTeXinterchartoks</code> commands.	General: Add a variant of <code>\babel@savevariable</code> to save <code>\XeTeXcharclass(es)</code> in a loop. .	31
<code>\FBthinspace</code> : Change <code>.16667em</code> to <code>.5\fontdimen2\font</code> to get in XeTeX and pdfTeX the same spacing as in LuaTeX.	<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>\frenchsetup</code> : Add a warning about options <code>og/fg</code> for old XeTeX or LuaTeX engines requiring active characters.	frenchb.lua: <code>font.getfont(fid)</code> possibly returns nil even for a positive fid (i.e. AMS <code>lcircle1.pfb</code>). Reported by François Legendre. .	24
<code>\NoAutoSpacing</code> : New definition based on <code>\FB@spacing@off</code> common to all engines.	v3.1k	
<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>	General: (pdfTeX shorthands) test on <code>\lastskip</code> changed from <code>0pt</code> to <code>1sp</code> for active punctuation for consistency with XeTeX and LuaTeX.	33
v3.2b	<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
General: Load <code>lualatex.tex</code> for plain LuaTeX to ensure <code>\newattribute</code> is defined.	v3.1j	
Warning added when the subcaption package is loaded before <code>babel/french</code>	General: Loading <code>luatexbase.sty</code> is no longer needed with LaTeX release	
<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		

2015/10/01 or later.	20	Node HLIST added; node TEMP added for the first node of \hboxes.	22
\frquote: \fr@quote completely rewritten: \leavevmode added and explicitly save/restore \everypar and \localleftbox instead of using a group in order to ensure compatibility with package wrapfig.	39	v3.1f	
\PackageWarning is undefined in Plain, use \fb@warning instead.	39	General: \FBCaption@Separator changed when option CustomiseFigTabCaptions is set to false.	52
v3.1i		\FBprocess@options: Bug fix for the beamer class: figure and table captions are now consistent with babel-french's documentation. Pointed out by Denis Bitouzé.	64
General: Remove restriction about loading numprint.sty after babel.	53	Definition of \captionformat and \captiondelim changed when option CustomiseFigTabCaptions is set to false.	64
\frquote: \luatexlocalleftbox changed to \localleftbox by new LaTeX release 2015/10/01.	39	\FBthinspace: \FBthinspace is no longer a kern but a skip (babel-french adds a nobreak penalty before it).	17
nombre: \nombre command changed when numprint.sty is not loaded: only one warning, no error.	48	v3.1e	
v3.1h		\frenchsetup: Corrected typo: SmallCapsFigTabcaptions instead of SmallCapsFigTabCaptions. Pointed out by Céline Chevalier.	54
General: french.cfg from e-french conflicts with babel-french. Do NOT load it (no need for .cfg files with babel-french anyway).	76	v3.1d	
v3.1g		General: New section: issue warnings if packages listings, numprint and natbib are loaded too early or too late vs babel.	53
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	v3.1c	
Use Babel defined loops \bb\@for instead of \@for borrowed from file ltcntrl.dtx (\@for is undefined in Plain).	30	frenchb.lua: Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André.	25
\captionfrench: \partname's definition depends now on flag PartNameFull. No need to redefine it in \frenchbsetup.	48	v3.1b	
\frenchsetup: Bug fix for koma-scripts classes: a spurious dot was added by the \partformat command.	55	\captionfrench: Change \scshape to customisable \FBfigtabshape for \figurename and \tablename.	48
PartNameFull now just sets the flag, nothing to add to \captionfrench when false.	54	\frenchsetup: New option SmallCapsFigTabCaptions.	54
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	\ieres: Removed \lowercase from definitions of \ieme and co: \up already does the conversion.	43
flag addgl set to false for '»' at the beginning of an \hbox or a paragraph or a tabular 'l' and 'c' columns.	27	\no: Removed \lowercase from definitions of \FrenchEnumerate, ... \No and co: \up already does the conversion.	43
		frenchb.lua: Add a check for null fid in french_punctuation (Tikz \nullfont). Bug pointed out by Paul Gaborit.	24

v3.1a		
General: fontspec is not required for T1 fonts used with the luainputenc.sty package.	66	
Misplaced \fi for plain formats.	20	
New command \frquote for imbedded or long French quotations.	38	
\ frenchsetup: Codes 0x13 and 0x14 added for French quotes in T1-encoding. Support for older versions of LuaTeX and XeTeX dropped.	60	
New options InnerGuillSingle, EveryParGuill and EveryLineGuill to control \frquote.	54	
frenchb.lua: Added flag adagl which must also be true when prev or next is not a char (i.e. \kern0 in «\texttt{a}»).	27	
Codes 0x13 and 0x14 added for French quotes in T1-encoding.	22	
Look ahead when next is a kern (i.e. in «\texttt{a} »).	27	
v3.0c		
General: babel-french requires babel-3.9i.	14	
Just load luatexbase.sty instead of luaotfload.sty with plain formats.	20	
No need to define \l@french as \lang@french, babel.def (3.9j) takes care for this.	15	
\ frenchsetup: New option INGuillSpace.	54	
No list customisation when beamer class is loaded.	55	
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	
v3.0b		
General: frenchb.lua was not found by Lua function dofile (not kpathsea aware). Call function kpse.find_file first, as suggested by Paul Gaborit.	29	
Require luatexbase with LaTeX2e in case fontspec has not been loaded before babel.	20	
v3.0a		
General: \bbl@nonfrenchguillemets deleted, use \babel@save instead.	38	
		\LdfInit checks \captionfrench instead of \datefrench to avoid a conflict with papertex.cls which loads datetime.sty.
		14
		french.cfg will be loaded (if found) instead of frenchb.cfg. NO NEED for .cfg files in French anyway.
		76
		In Plain, provide a substitute for \PackageWarning and \PackageInfo.
		14
		Merging of \captionfrenchb, \captionfrançais with \captionfrench deleted in favor of new babel 3.9 syntax.
		50
		More informative, less TeXnical warning about \makecaption.
		52
		New flag \ifFB@luatex@punct for 'high punctuation' management with LuaTeX engines.
		17
		New handling of 'high punctuation' through callbacks with LuaTeX engines.
		20
		No warning about \makecaption for SMF classes.
		51
		Options processing completely reorganised, now \babel@save and \babel@savevariable are usable for French.
		54
		Support for options frenchb, français, canadien, acadian changed.
		14
		Test \ifXeTeX changed to \ifFBunicode and 'xltxtra' changed to 'fontspec'.
		66
		\CaptionSeparator: Remove \FBCaption@SeparatorORI, use \babel@save instead.
		50
		\captionfrench: Take advantage of babel's \SetString commands for captionnames.
		48
		\datefrench: Take advantage of babel's \SetString commands for \datefrench. Doesn't work with Plain (yet?).
		40
		\descriptionFB: Added \listindentFB to \itemindent. Suggested by Denis Bitouzé.
		70
		\extrasfrench: Take advantage of babel's \babel@savevariable to handle apostrophe's \lccode.
		16
		\FB@fg: Definitions of \FB@og and \FB@fg now depend on punctuation handling (LuaTeX / XeTeX / active).
		37
		\FBprocess@options: With

koma-script and memoir class,
customise \captionformat and
\captiondelim. 64

\frenchsetup: New options
OldFigTabCaptions and
CustomiseFigTabCaptions. 54