

A Babel language definition file for French

frenchb.dtx v3.5h, 2020/04/18

Daniel Flipo
daniel.flipo@free.fr

Contents

1	The French language	2
1.1	Basic interface	2
1.2	Customisation	5
1.2.1	\frenchsetup	5
1.2.2	Caption names	9
1.2.3	Figure and table captions	9
1.3	Hyphenation checks	10
1.4	Changes	10
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	32
2.2.4	Punctuation switches common to all engines	34
2.3	Commands for French quotation marks	35
2.4	Date in French	39
2.5	Extra utilities	40
2.6	Formatting numbers	44
2.7	Caption names	46
2.8	Figure and table captions	48
2.9	Dots...	51
2.10	More checks about packages' loading order	52
2.11	Setup options: keyval stuff	52
2.12	French lists	66
2.13	French indentation of sections	70
2.14	Formatting footnotes	71
2.15	Clean up and exit	75
2.16	Files frenchb.ldf, francais.ldf, canadien.ldf and acadian.ldf	75
3	Change History	77

1 The French language

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

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

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

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

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

Changes between version 3.0 and v3.5h are listed in subsection 1.4 p. 10.

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

1.1 Basic interface

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

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

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

²

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

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

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

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

¹The file described in this section has version number v3.5h and was last revised on 2020/04/18.

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

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

5. the separator following the table or figure number in captions is printed as ‘ – ’ instead of ‘ : ’; for changing this see 1.2.3 p. 9.

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

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

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

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

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

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

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

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

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

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

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

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

2. `\frenchdate{<year>}{<month>}{<day>}` helps typesetting dates in French: `\frenchdate{2001}{01}{01}` will print 1er 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 3es). All these commands take advantage of real superscript letters when they are available in the current font.
4. Family names should be typeset in small capitals and never be hyphenated, the macro `\bsc` (boxed small caps) does this, e.g., `L.\bsc{Lamport}` will print the same as `L.\mbox{\textsc{Lamport}}`. Note that composed names (such as Dupont-Durant) may now be hyphenated on explicit hyphens, this differs from `babel-french` v. 1.x.
5. Commands `\primo`, `\secundo`, `\tertio` and `\quarto` print 1o, 2o, 3o, 4o. `\FrenchEnumerate{6}` prints 6o.
6. Abbreviations for “Numéro(s)” and “numéro(s)” (No Nos no and nos) are obtained via the commands `\No`, `\Nos`, `\no`, `\nos`.
7. Two commands are provided to typeset the symbol for “degré”: `\degre` prints the raw character and `\degres` should be used to typeset temperatures (e.g., “20~\degres C” with a non-breaking space), or for alcohols’ strengths (e.g., “45\degres” with no space in French).
8. In math mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the `TEXbook` p. 134). The command `\DecimalMathComma` makes the comma behave as an ordinary character *when the current language is French* (no space added); as a counterpart, if `\DecimalMathComma` is active, an explicit space has to be added in lists and intervals: $[0, \ 1]$, $(x, \ y)$. `\StandardMathComma` switches back to the standard behaviour of the comma in French.
The `icomma` package is an alternative workaround.
9. A command `\nombre` was provided in 1.x versions to easily format numbers in slices of three digits separated either by a comma in English or with a space in French; `\nombre` is now mapped to `\numprint` from `numprint.sty`, 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 `'1er juin'` (no need for a forced space after `1\ier`).

1.2 Customisation

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

1.2.1 `\frenchsetup{options}`

`\frenchsetup{}` and `\frenchbsetup{}` are synonymous; the latter should be preferred as the language name for French in `Babel` is no longer `frenchb` but `french`. `\frenchsetup{ShowOptions}` prints all available options to the `.log` file, it is just meant as a reminder 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’ \leq Leftmargin — first item running on two lines or more... — first second level item on two lines... — next one... — second item...	Text starting at ‘parindent’ \leq Leftmargin — first item running on two lines or more... — first second level item on two lines... — next one... — second item...
Default French layout	With <code>ListItemsAsPar=true</code>

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

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

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

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

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

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

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

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

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

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

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`.

all four options `StandardListSpacing=true`, `StandardItemizeEnv=true`, `StandardEnumerateEnv=true` and `StandardItemLabels=true`.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

⁹Actually without stretch nor shrink.

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

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

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

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

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

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

1.2.2 Caption names

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

1.2.3 Figure and table captions

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

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

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

Three options are provided to customise figure and table captions:

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

1.3 Hyphenation checks

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

- run pdfLaTeX on the following file:

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

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

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

Frequent mismatches:

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

1.4 Changes

What’s new in version 3.5?

Version 3.5a offers a new option `ListItemsAsPar`. The default layout of lists is unchanged (for backward compatibility), but users should try this new option which

ensures a layout of lists closer to French typographic standards: see f.i. how lists are typeset in the book “Lexique des règles typographiques en usage à l’Imprimerie Nationale”.

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

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).

What’s new in version 3.4?

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

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

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

Version 3.4 requires eTeX and LuaTeX 1.0.4 or newer.

What’s new in version 3.3?

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

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

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

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

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

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

What's new in version 3.2?

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

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

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

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

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

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

What's new in version 3.1?

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

What's new in version 3.0?

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

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

- The last but not least change affects the (recent) LuaTeX-based engines, (this means version 0.76 as included in TL2013 and up): active characters are no longer used in French for ‘high punctuation’ ¹⁰. Functionalities and user interface are unchanged.

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

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

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

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

2 The code

2.1 Initial setup

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

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

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

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

Quit if eTeX is not available.

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

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

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

```

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

```

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

```

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

```

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

```

57 \ifdefined\l@acadian
58   \addialect\l@canadien\l@acadian
59 \else
60   \addialect\l@acadian\l@french
61   \addialect\l@canadien\l@french
62 \fi

```

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

```

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

```

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

```

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

```


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

```

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

```

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

```

93 \newif\ifBfrench

```

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

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

```

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

```

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

```

104 \addto\extrasfrench{\bbl@frenchspacing}
105 \addto\noextrasfrench{\bbl@nonfrenchspacing}

```

2.2 Punctuation

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

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

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

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

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

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

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

`\FBguillspace` These three commands are meant for basic French. Other French dialects can use different settings, see below. According to the I.N. specifications, the ‘:’ requires an inter-word space before it, the other three require just a thin space. We define `\FBcolonspace` as `\space` (inter-word space) and `\FBthinspace` as an half inter-word space with no shrink nor stretch. `\FBguillspace` is defined btw. as spacing for French quotes is

handled together with high punctuation for LuaTeX and XeTeX. `\FBguillspace` has been fine tuned by Thierry Bouche to 80% of an inter-word space with reduced stretchability. All three are user customisable in the preamble, best using the `\FBsetspaces` command described below. A penalty will be added before these spaces to prevent line breaking.

```

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

```

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

```

140 \ifLaTeXe
141   \newcommand*{\FBsetspaces}[5][french]{%
142     \def\bbl@tempa{french}\def\bbl@tempb{#1}%
143     \ifx\bbl@tempa\bbl@tempb \def\bbl@tempb{}\fi
144     \@namedef{\bbl@tempb FB#2space}{\hskip #3\fontdimen2\font
145                                     plus #4\fontdimen3\font
146                                     minus #5\fontdimen4\font \relax}%

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

Same for `\FBcolonspace`:

```

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

```

And for `\FBguillspace`:

```

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

```

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

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

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

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

2.2.1 Punctuation with LuaTeX

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

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

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

```
213   \input ltxlua.tex
214 \fi
```

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

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

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

`\FB@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.

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

Local definitions for nodes:

```

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

```



```

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

```

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

```

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

```

Let’s initialize the global LuaTeX table `FBsp`: it holds the characteristics of the glues used in French and Acadian for high punctuation and quotes and the corresponding no-breaking space characters for option **UnicodeNoBreakSpaces**.

```

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

```

```
343 FBsp.guill.ch.fr = 0xA0 ; FBsp.guill.ch.ac = nil
```

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

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

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

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

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

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

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

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

Skip glyphs not concerned by French kernings.

```

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

```

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

```

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

```

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

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

```

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

```

```

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

```

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

```

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

```

Let's consider '»' now (the only remaining glyph of `FB_punct_left` class): we just have to remove any *glue* possibly preceding '», then to insert the nobreak penalty and the

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

```

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

```

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

```

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

```

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

```

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

```

```

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

```

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

```

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

```

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

```

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

```

```

553         end
554     end
555 end
556 end
557 end
558 end
559 return head
560 end
561 return french_punctuation
562 </lua>

```

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

```

563 < *french>
564 \ifFB@luatexpunct
565   \newcommand*{\FB@luatexpunct@french}{%
566     \babel@save\shorthandon
567     \babel@save\shorthandoff
568     \def\shorthandoff##1{%
569       \ifx\PackageWarning\@undefined
570         \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
571           LuaTeX, \\\ use \noexpand\NoAutoSpacing
572           *inside a group* instead.}%
573       \else
574         \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?}
575           is helpless with LuaTeX, \MessageBreak
576           use \protect\NoAutoSpacing \space *inside a group*
577           instead; \MessageBreak reported}%
578       \fi}%
579     \def\shorthandon##1{}}%
580   }
581   \addto\extrasfrench{\FB@luatexpunct@french}

```

The next definition will be used to activate Lua punctuation: it loads `frenchb.lua` and adds function `french_punctuation` to the kerning callback; "adding" anything actually disables the built-in kerning for Type1 fonts (which is now added to `french_punctuation`).

```

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

```



```

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

```

End of specific code for punctuation with LuaTeX engines.

2.2.2 Punctuation with XeTeX

If `\XeTeXinterchartokenstate` is available, we use the “inter char” mechanism to provide correct spacing in French before the four characters `;` `!` `?` and `:`. The basis of the following code was borrowed from the `polyglossia` package, see `gloss-french.ldf`. We use the same mechanism for French quotes (`«` and `»`), when automatic spacing for quotes is required by options `og=«` and `fg=»` in `\frenchsetup{}` (see section 2.11). The default value for `\XeTeXcharclass` is 0 for characters tokens and `\FB@nonchar` for all other tokens (glues, kerns, math and box boundaries, etc.). These defaults should not be changed otherwise the spacing before the ‘high punctuation’ characters and inside quotes might not be correct.

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

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

```

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

```

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

```

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

```

As `\babel@savevariable@loop#1#2{\begingroup` doesn’t work inside a `\bbl@for` loop, we define a variant to save the `\XeTeXcharclass` values which will be modified in French.

```

615 \def\FBsavevariable@loop#1#2{\begingroup
616   \toks@ \expandafter{\originalTeX #1}%
617   \edef\x{\endgroup
618     \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
619   \x}

```

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

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

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

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

```

620 \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F,%
621 "29,"5D,"7B,"7D,"2C,"2D,"2E,"22,"25,"27,"60,"2019}

```

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

```

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

```

Let's now set the classes and interactions between classes. When false, the flag `\ifFB@spacing` switches off any interaction between classes (this flag is controlled by user-level command `\NoAutoSpacing`; this flag is also set to false when the current font is a typewriter font).

```

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

```

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

```

646 \XeTeXinterchartoks \FB@nonchar \FB@punctthick = {%
647 \ifFB@spacing
648 \ifhmode
649 \ifdim\lastskip>1sp
650 \unskip\penalty\@M\FBcolonspace

```

```

651         \else
652         \FDP@colonspace
653         \fi
654     \fi
655     \fi}%
656 \bbl@for\FB@char
657 {`\;,\`!\,\`?}%
658 {\XeTeXcharclass\FB@char=\FB@punctthin}%
659 \XeTeXinterchartoks \z@ \FB@punctthin = {%
660     \ifFB@spacing\ifhmode\FDP@thinspace\fi\fi}%
661 \XeTeXinterchartoks \FB@guilf \FB@punctthin = {%
662     \ifFB@spacing\FDP@thinspace\fi}%
663 \XeTeXinterchartoks \FB@nonchar \FB@punctthin = {%
664     \ifFB@spacing
665     \ifhmode
666     \ifdim\lastskip>1sp
667     \unskip\penalty\@M\FBthinspace
668     \else
669     \FDP@thinspace
670     \fi
671     \fi
672     \fi}%
673 \XeTeXinterchartoks \FB@guilo \z@ = {%
674     \ifFB@spacing\FB@guillspace\fi}%
675 \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
676     \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
677 \XeTeXinterchartoks \z@ \FB@guilf = {%
678     \ifFB@spacing\FB@guillspace\fi}%
679 \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
680     \ifFB@spacing\FB@guillspace\fi}%
681 \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
682     \ifFB@spacing\unskip\FB@guillspace\fi}%

```

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

```

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

```

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

```

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

```

End of specific code for punctuation with modern XeTeX engines.

```
691 \fi
```

2.2.3 Punctuation with standard (pdf)TeX

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

```

692 \ifFB@active@punct
693   \initiate@active@char{:}%
694   \initiate@active@char{;}%
695   \initiate@active@char{!}%
696   \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.

```

697 \declare@shorthand{french}{;}{;%
698   \ifFB@spacing
699     \ifhmode
700       \ifdim\lastskip>1sp
701         \unskip\penalty\@M\FBthinspace
702       \else
703         \FDP@thinspace
704       \fi
705     \fi
706   \fi

```

Now we can insert a ; character.

```

707   \string;}

```

The next three definitions are very similar.

```

708 \declare@shorthand{french}{!}{;%
709   \ifFB@spacing
710     \ifhmode
711       \ifdim\lastskip>1sp
712         \unskip\penalty\@M\FBthinspace
713       \else
714         \FDP@thinspace
715       \fi
716     \fi
717   \fi
718   \string!}
719 \declare@shorthand{french}{?}{;%
720   \ifFB@spacing
721     \ifhmode
722       \ifdim\lastskip>1sp
723         \unskip\penalty\@M\FBthinspace
724       \else
725         \FDP@thinspace
726       \fi
727     \fi
728   \fi
729   \string?}
730 \declare@shorthand{french}{:}{;%
731   \ifFB@spacing
732     \ifhmode
733       \ifdim\lastskip>1sp
734         \unskip\penalty\@M\FBcolonspace

```

```

735         \else
736         \FDP@colonspace
737         \fi
738     \fi
739     \fi
740     \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.

```

741 \declare@shorthand{system}{:}{\string:}
742 \declare@shorthand{system}{!}{\string!}
743 \declare@shorthand{system}{?}{\string?}
744 \declare@shorthand{system}{;}{\string;}

```

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

```

745 \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.

```

746 \bbl@activate{:}{\bbl@activate{;}}%
747 \bbl@activate{!}{\bbl@activate{?}}%
748 }
749 \addto\noextrasfrench{%
750 \bbl@deactivate{:}{\bbl@deactivate{;}}%
751 \bbl@deactivate{!}{\bbl@deactivate{?}}%
752 }
753 \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.

```

754 \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 \TeX .

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

```

755 \def\autospace@beforeFDP{%
756   \ifFB@luatex@punct\FB@addDPspace=1 \fi
757   \def\FDP@thinspace{\penalty\@M\FBthinspace}%
758   \def\FDP@colonspace{\penalty\@M\FBcolonspace}}
759 \def\noautospace@beforeFDP{%
760   \ifFB@luatex@punct\FB@addDPspace=0 \fi
761   \let\FDP@thinspace\@empty
762   \let\FDP@colonspace\@empty}
763 \ifLaTeXe

```

```

764 \def\AutoSpaceBeforeFDP{\autospace@beforeFDP
765 \FBAutoSpacePunctuationtrue}
766 \def\NoAutoSpaceBeforeFDP{\noautospace@beforeFDP
767 \FBAutoSpacePunctuationfalse}
768 \AtEndOfPackage{\AutoSpaceBeforeFDP}
769 \else
770 \let\AutoSpaceBeforeFDP\autospace@beforeFDP
771 \let\NoAutoSpaceBeforeFDP\noautospace@beforeFDP
772 \AutoSpaceBeforeFDP
773 \fi

```

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

To be consistent with what is done for the ; : ! ? characters, **\ttfamilyFB** also switches off insertion of spaces inside French guillemets *when they are typed in as characters* with the ‘og’/‘fg’ options in **\frenchsetup{}**. This is also a workaround for the weird behaviour of these characters in verbatim mode.

```

774 \ifLaTeXe
775 \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
776 \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on \rmfamilyORI}
777 \DeclareRobustCommand\sffamilyFB{\FB@spacing@on \sffamilyORI}
778 \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.

```

779 \DeclareRobustCommand*\NoAutoSpacing{%
780 \FB@spacing@off
781 \ifFB@active@punct\shorthandoff{;:!?}\fi
782 }

```

2.3 Commands for French quotation marks

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

```

783 \ifLaTeXe
784 \else
785   \ifFBunicode
786     \def\guillemotleft{{\char"00AB}}
787     \def\guillemotright{{\char"00BB}}
788     \def\textquotedblleft{{\char"201C}}
789     \def\textquotedblright{{\char"201D}}
790   \else
791     \def\guillemotleft{\leavevmode\raise0.25ex
792                       \hbox{$\scriptscriptstyle\ll$}}
793     \def\guillemotright{\raise0.25ex
794                        \hbox{$\scriptscriptstyle\gg$}}
795     \def\textquotedblleft{``}
796     \def\textquotedblright{' '}
797   \fi
798   \let\xspace\relax
799 \fi

```

\FBgspchar The next step is to provide correct spacing after ‘«’ and before ‘»’; no line break is allowed
\FB@og neither *after* the opening one, nor *before* the closing one. French quotes (including
\FB@fg spacing) are printed by \FB@og and \FB@fg, the expansion of the top level commands
\og and \og is different in and outside French.
The definitions of \FB@og and \FB@fg need some engine-dependent tuning: for LuaTeX,
\FB@spacing is set to 0 locally to prevent the quotes characters from adding space
when option **og=«**, **fg=»** is set.

```

800 \newcommand*{\FB@guillspace}{\penalty\@M\FBguillspace}
801 \newcommand*{\FBgspchar}{\char"A0\relax}
802 \newif\ifFBucsNBSP
803 \ifFB@luatex@punct
804   \DeclareRobustCommand*{\FB@og}{\leavevmode
805     \bgroup\FB@spacing=0 \guillemotleft\egroup
806     \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi}
807   \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>z@ \unskip\fi
808     \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi
809     \bgroup\FB@spacing=0 \guillemotright\egroup}
810 \fi

```

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

```

811 \ifFB@xetex@punct
812   \DeclareRobustCommand*{\FB@og}{\leavevmode
813     \bgroup\FB@spacingfalse\guillemotleft\egroup
814     \FB@guillspace}
815   \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>z@ \unskip\fi
816     \FB@guillspace
817     \bgroup\FB@spacingfalse\guillemotright\egroup}
818 \fi
819 \ifFB@active@punct
820   \DeclareRobustCommand*{\FB@og}{\leavevmode
821     \guillemotleft
822     \FB@guillspace}
823   \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>z@ \unskip\fi
824     \FB@guillspace

```



```

825         \guillemotright}
826 \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 these commands are not yet defined.

```

827 \newcommand*\og{}\@empty}
828 \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).

```

829 \ifLaTeXe
830   \def\bbl@frenchguillemets{%
831     \renewcommand*\og{}\FB@og}%
832     \renewcommand*\fg{}\FB@fg\xspace}}
833   \renewcommand*\og{}\textquotedblleft}
834   \renewcommand*\fg{}\ifdim\lastskip>\z@unskip\fi
835     \textquotedblright\xspace}
836 \else
837   \def\bbl@frenchguillemets{\let\og\FB@og
838     \let\fg\FB@fg}
839   \def\og{}\textquotedblleft}
840   \def\fg{}\ifdim\lastskip>\z@unskip\fi\textquotedblright}
841 \fi

842 \addto\extrasfrench{\babel@save\og \babel@save\fg
843   \bbl@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...

```

844 \newcommand*\ogi{}\FB@og}
845 \newcommand*\fgi{}\FB@fg}
846 \newcommand*\@ogi{}\ifmmode\hbox{\ogi}\else\ogi\fi}
847 \newcommand*\@fgi{}\ifmmode\hbox{\fgi}\else\fgi\fi}
848 \newcommand*\ogii{}\textquotedblleft}
849 \newcommand*\fgii{}\textquotedblright}
850 \newcommand*\@ogii{}\ifmmode\hbox{\ogii}\else\ogii\fi}
851 \newcommand*\@fgii{}\ifmmode\hbox{\fgii}\else\fgii\fi}

```

and the needed technical stuff to handle options:

```

852 \newcount\FBguill@level
853 \newtoks\FB@everypar

```

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

```

854 \def\FB@addquote@everypar{%
855   \let\FBnew@everypar\everypar

```

```

856 \FBold@everypar=\expandafter{\the\everypar}%
857 \FBnew@everypar={\the\FBold@everypar\FBeverypar@quote}%
858 \let\everypar\FBold@everypar
859 \let\FB@addquote@everypar\relax
860 }
861 \newif\ifFBcloseguill \FBcloseguilltrue
862 \newif\ifFBInnerGuillSingle
863 \def\FBguillopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
864 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
865 \let\FBguillnone\empty
866 \let\FBeveryparguill\FBguillopen
867 \let\FBverylineguill\FBguillnone
868 \let\FBeverypar@quote\relax
869 \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.

```

870 \ifLaTeXe
871   \DeclareRobustCommand\frquote{%
872     \ifstar{\FBcloseguillfalse\fr@quote}%
873     {\FBcloseguilltrue\fr@quote}}
874 \else
875   \newcommand\frquote[1]{\fr@quote{#1}}
876 \fi

```

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

```

877 \newcommand{\fr@quote}[1]{%
878   \leavevmode
879   \advance\FBguill@level by \@ne
880   \ifcase\FBguill@level
881   \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:

```

882   \ifx\FBeveryparguill\FBguillnone
883   \else
884     \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
885     \FB@addquote@everypar
886   \fi
887   \@ogi #1\@fgi
888   \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.

```

889   \ifx\FBverylineguill\FBguillopen
890     \def\FBveryline@quote{\FB@addGUllspace=0 \guillemotleft
891       \FB@guillspace}%
892     \lcalleftbox{\FBveryline@quote}%
893     \let\FBeverypar@quote\relax
894     \@ogi #1\ifFBcloseguill\@fgi\fi
895   \else
896     \ifx\FBverylineguill\FBguillclose
897     \def\FBveryline@quote{\FB@addGUllspace=0 \guillemotright
898       \FB@guillspace}%

```

```

899      \lcalleftbox{\FBeveryline@quote}%
900      \let\FBeverypar@quote\relax
901      \@ogi #1\ifFBcloseguill\@fgi\fi
902      \else

```

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

```

903      \let\FBeverypar@quote\relax
904      \ifFBInnerGuillSingle
905      \def\ogii{\leavevmode
906      \guilsinglleft\FB@guillspace}%
907      \def\fgii{\ifdim\lastskip>\z@\unskip\fi
908      \FB@guillspace\guilsinglright}%
909      \ifx\FBeveryparguill\FBguillopen
910      \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
911      \fi
912      \ifx\FBeveryparguill\FBguillclose
913      \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
914      \fi
915      \fi
916      \@ogii #1\ifFBcloseguill \@fgii \fi
917      \fi
918      \fi
919      \else

```

Warn if \FBguill@level > 2:

```

920      \ifx\PackageWarning\@undefined
921      \fb@warning{\noexpand\frquote\space handles up to
922      two levels.\\ Quotation not printed.}%
923      \else
924      \PackageWarning{french.ldf}{%
925      \protect\frquote\space handles up to two levels.
926      \MessageBreak Quotation not printed. Reported}
927      \fi
928      \fi

```

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

```

929      \global\advance\FBguill@level by \m@ne
930      \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
931      \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
932      \global\let\FBeveryline@quote\empty
933      \ifx\FBeverylineguill\FBguillnone\else\lcalleftbox{}\fi
934      \fi
935      }

```

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.

```

936 \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. Note that coding `\StartBabelCommands*{french,acadian}` would *only* define `\csname date\CurrentOption\endcsname`, leaving the second language undefined in Babel's sens.

```

937 \def\BabelLanguages{french,acadian}
938 \StartBabelCommands*{\BabelLanguages}{date}
939     [unicode, fontenc=TU EU1 EU2, charset=utf8]
940     \SetString\monthiiname{février}
941     \SetString\monthviiiname{août}
942     \SetString\monthxiiname{décembre}
943 \StartBabelCommands*{\BabelLanguages}{date}
944     \SetStringLoop{month#1name}{%
945         janvier,f\'evrier,mars,avril,mai,juin,juillet,%
946         ao\'ut,septembre,octobre,novembre,d\'ecembre}
947     \SetString\today{\FB@date{\year}{\month}{\day}}
948 \EndBabelCommands

```

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

```

949 \newcommand*{\FB@date}[3]{%
950     {{\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
951     \csname month\romannumeral#2name\endcsname
952     \ifx#1\@empty\else\FBdatespace\number#1\fi}}
953 \newcommand*{\FBdatebox}{\hbox}
954 \newcommand*{\FBdatespace}{\space}
955 \newcommand*{\frenchdate}{\FBdatebox\FB@date}
956 \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 '1er'. Up to version 2.0 of `babel-french` **\fup** `\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).

```

957 \newif\ifFB@poorman
958 \newdimen\FB@Mht
959 \ifLaTeXe
960     \AtEndOfPackage{\RequirePackage{scalegnt}}

```

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

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

```

961 \newcommand*{\FBsupR}{-0.12}
962 \newcommand*{\FBsupS}{0.65}
963 \newcommand*{\FB@lc}[1]{\MakeLowercase{#1}}
964 \DeclareRobustCommand*{\FB@up@fake}[1]{%
965   \settoheight{\FB@Mht}{M}%
966   \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
967   \addtolength{\FB@Mht}{-\FBsupS ex}%
968   \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
969 }

```

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.

```

970 \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
971   \def\FB@suffix{#4}}
972 \def\FB@x{x}
973 \def\FB@j{j}
974 \DeclareRobustCommand*{\FB@up}[1]{%
975   \bgroup \FB@poormantrue
976   \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.

```

977   \edef\reserved@a{\lowercase{%
978     \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
979   \reserved@a
980   {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
981     \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
982     \if\FB@poorman \FB@up@fake{#1}%
983     \else \FB@up@real{#1}%
984     \fi}%
985   {\FB@up@fake{#1}}%
986 \egroup}

```

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

```

987 \newcommand*{\FB@up@real}[1]{\bgroup
988   \fontfamily{\FB@firstthree -sup}\selectfont \FB@lc{#1}\egroup}
\fbup is defined as \FB@up unless \realsuperscript is defined by realscripts.sty.
989 \DeclareRobustCommand*{\fbup}[1]{%
990   \ifx\realsuperscript\@undefined
991     \FB@up{#1}%
992   \else
993     \bgroup\let\fakesuperscript\FB@up@fake
994     \realsuperscript{\FB@lc{#1}}\egroup
995   \fi}

```

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

```

996 \providecommand*{\up}{\relax}

```

Poor man's definition of \up for Plain.

```

997 \else
998   \providecommand*{\up}[1]{\leavevmode\raise1ex\hbox{\sevenrm #1}}
999 \fi

```

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

```

\ier 1000 \def\ieme{\up{e}\xspace}
\iere 1001 \def\iemes{\up{es}\xspace}
\iemes 1002 \def\ier{\up{er}\xspace}
\iers 1003 \def\iers{\up{ers}\xspace}
\ieres 1004 \def\iere{\up{re}\xspace}
1005 \def\ieres{\up{res}\xspace}

```

\FBmedkern

\FBthickkern

```

1006 \newcommand*{\FBmedkern}{\kern+.2em}
1007 \newcommand*{\FBthickkern}{\kern+.3em}

```

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

```

\no 1008 \newcommand*{\FrenchEnumerate}[1]{#1\up{o}\FBthickkern}
\nos 1009 \newcommand*{\FrenchPopularEnumerate}[1]{#1\up{o})\FBthickkern}
\nos Typing \primo should result in 'o',
\primo 1010 \def\primo{\FrenchEnumerate1}
\frimo 1011 \def\secundo{\FrenchEnumerate2}
1012 \def\tertio{\FrenchEnumerate3}
1013 \def\quarto{\FrenchEnumerate4}

```

while typing \frimo) gives 'o'.

```

1014 \def\frimo{\FrenchPopularEnumerate1}
1015 \def\fsecundo{\FrenchPopularEnumerate2}
1016 \def\ftertio{\FrenchPopularEnumerate3}
1017 \def\fquarto{\FrenchPopularEnumerate4}

```

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

```

1018 \DeclareRobustCommand*{\No}{N\up{o}\FBmedkern}
1019 \DeclareRobustCommand*{\no}{n\up{o}\FBmedkern}
1020 \DeclareRobustCommand*{\Nos}{N\up{os}\FBmedkern}
1021 \DeclareRobustCommand*{\nos}{n\up{os}\FBmedkern}

```

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

```
1022 \DeclareRobustCommand*\bsc}[1]{\leavevmode\beginngroup\kern0pt
1023                                     \scshape #1\endgroup}
1024 \ifLaTeXe\else\let\scshape\relax\fi
```

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

```
1025 \ifBUnicode
1026   \newcommand*\at{{\char"0040}}
1027   \newcommand*\circonflexe{{\char"005E}}
1028   \newcommand*\tild{{\char"007E}}
1029   \newcommand*\boi{{\char"005C}}
1030   \newcommand*\degree{{\char"00B0}}
1031 \else
1032   \ifLaTeXe
1033     \DeclareTextSymbol{\at}{T1}{64}
1034     \DeclareTextSymbol{\circonflexe}{T1}{94}
1035     \DeclareTextSymbol{\tild}{T1}{126}
1036     \DeclareTextSymbolDefault{\at}{T1}
1037     \DeclareTextSymbolDefault{\circonflexe}{T1}
1038     \DeclareTextSymbolDefault{\tild}{T1}
1039     \DeclareRobustCommand*\boi{\textbackslash}
1040     \DeclareRobustCommand*\degree{\r{ }}
1041   \else
1042     \def\T@one{T1}
1043     \ifx\f@encoding\T@one
1044       \newcommand*\degree{{\char6}}
1045     \else
1046       \newcommand*\degree{{\char23}}
1047     \fi
1048     \newcommand*\at{{\char64}}
1049     \newcommand*\circonflexe{{\char94}}
1050     \newcommand*\tild{{\char126}}
1051     \newcommand*\boi{{\backslash}}
1052   \fi
1053 \fi
```

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

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

```

1054 \ifLaTeXe
1055   \newcommand*{\degres}{\degre}
1056   \ifFBunicode
1057     \DeclareRobustCommand*{\degres}{\degre}
1058   \else
1059     \def\Warning@degree@TSone{\FBWarning
1060       {Degrees would look better in TS1-encoding:%
1061       \MessageBreak add \protect
1062       \usepackage{textcomp} to the preamble.%
1063       \MessageBreak Degrees used}}
1064     \AtBeginDocument{\ifx\DeclareEncodingSubset\@undefined
1065       \DeclareRobustCommand*{\degres}{%
1066         \leavevmode\hbox to 0.3em{\hss\degre\hss}%
1067         \Warning@degree@TSone
1068         \global\let\Warning@degree@TSone\relax}%
1069       \else
1070         \DeclareRobustCommand*{\degres}{%
1071           \hbox{\UseTextSymbol{TS1}{\textdegree}}}%
1072       \fi
1073     }
1074   \fi
1075 \else
1076   \newcommand*{\degres}{%
1077     \leavevmode\hbox to 0.3em{\hss\degre\hss}}
1078 \fi

```

2.6 Formatting numbers

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

```

1079 \newif\ifFB@icomma
1080 \newcount\mc@charclass
1081 \newcount\mc@charfam
1082 \newcount\mc@charslot
1083 \newcount\std@mcc
1084 \newcount\dec@mcc
1085 \ifBLaTeX
1086   \mc@charclass=\Umathcharclass`,
1087   \newcommand*{\dec@math@comma}{%
1088     \mc@charfam=\Umathcharfam`,
1089     \mc@charslot=\Umathcharslot`,
1090     \Umathcode`,= 0 \mc@charfam \mc@charslot
1091   }
1092   \newcommand*{\std@math@comma}{%
1093     \mc@charfam=\Umathcharfam`,
1094     \mc@charslot=\Umathcharslot`,
1095     \Umathcode`,= \mc@charclass \mc@charfam \mc@charslot

```



```

1096 }
1097 \else
1098   \std@mcc=\mathcode`\,
1099   \dec@mcc=\std@mcc
1100   \@tempcnta=\std@mcc
1101   \divide\@tempcnta by "1000
1102   \multiply\@tempcnta by "1000
1103   \advance\dec@mcc by -\@tempcnta
1104   \newcommand*\dec@math@comma{\mathcode`\,=\dec@mcc}
1105   \newcommand*\std@math@comma{\mathcode`\,=\std@mcc}
1106 \fi

\DecimalMathComma operates in French or Acadian independently.
1107 \newcommand*\DecimalMathComma{%
1108   \ifFB@icomma
1109     \PackageWarning{french.ldf}{%
1110       icomma package loaded, \protect\DecimalMathComma\MessageBreak
1111       does nothing. Reported}%
1112   \else
1113     \ifFBfrench
1114       \dec@math@comma
1115       \expandafter\addto\csname extras\language\endcsname
1116       {\dec@math@comma}%
1117     \fi
1118   \fi
1119 }
1120 \newcommand*\StandardMathComma{%
1121   \ifFB@icomma
1122     \PackageWarning{french.ldf}{%
1123       icomma package loaded, \protect\StandardMathComma\MessageBreak
1124       does nothing. Reported}%
1125   \else
1126     \std@math@comma
1127     \expandafter\addto\csname extras\language\endcsname
1128     {\std@math@comma}%
1129   \fi
1130 }
1131 \ifLaTeXe
1132   \AtBeginDocument{\@ifpackageloaded{icomma}%
1133     {\FB@icommatrue}%
1134     {\addto\noextrasfrench{\std@math@comma}%
1135     \ifdefined\noextrasacadian
1136       \addto\noextrasacadian{\std@math@comma}%
1137     \fi
1138     }%
1139   }
1140 \else
1141   \addto\noextrasfrench{\std@math@comma}
1142 \fi

```

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

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

```
1143 \newcommand*{\nombre}[1]{\fb@warning{*** \noexpand\nombre
1144                               no longer formats numbers\string! ***}}
```

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

```
1145 \ifFB@luatex@punct
1146   \activate@luatexpunct
1147 \fi
1148 \let\FBstop@here\relax
1149 \def\FBclean@on@exit{%
1150   \let\ifLaTeXe\undefined
1151   \let\LaTeXettrue\undefined
1152   \let\LaTeXefalse\undefined
1153   \let\FB@llc\loadlocalcfg
1154   \let\loadlocalcfg\@gobble}
1155 \ifx\magnification\@undefined
1156 \else
1157   \def\FBstop@here{%
1158     \FBclean@on@exit
1159     \ldf@finish\CurrentOption
1160     \let\loadlocalcfg\FB@llc
1161     \endinput}
1162 \fi
1163 \FBstop@here
```

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

```
1164 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1165 \newcommand*{\Warning@nombre}[1]{%
1166   \ifdefined\numprint
1167     \numprint{#1}%
1168   \else
1169     \PackageWarning{french.ldf}{%
1170       \protect\nombre\space now relies on package numprint.sty,%
1171       \MessageBreak add \protect
1172       \usepackage[autolanguage]{numprint},\MessageBreak
1173       see file numprint.pdf for more options.\MessageBreak
1174       \protect\nombre\space called}%
1175     \global\let\Warning@nombre\relax
1176     {#1}%
1177   \fi
1178 }

1179 \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`.

```
1180 \providecommand*\FBfigtabshape{\scshape}
```

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

```
1181 \StartBabelCommands*\BabelLanguages}{captions}
1182   [unicode, fontenc=TU EU1 EU2, charset=utf8]
1183   \SetString{\refname}{Références}
1184   \SetString{\abstractname}{Résumé}
1185   \SetString{\prefacename}{Préface}
1186   \SetString{\contentsname}{Table des matières}
1187   \SetString{\ccname}{Copie à }
1188   \SetString{\proofname}{Démonstration}
1189   \SetString{\partfirst}{Première}
1190   \SetString{\partsecond}{Deuxième}
1191   \SetStringLoop{ordinal#1}{%
1192     \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%
1193     Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1194     Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1195     Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1196 \StartBabelCommands*\BabelLanguages}{captions}
1197   \SetString{\refname}{R\`ef\`erences}
1198   \SetString{\abstractname}{R\`esum\`e}
1199   \SetString{\bibname}{Bibliographie}
1200   \SetString{\prefacename}{Pr\`eface}
1201   \SetString{\chaptername}{Chapitre}
1202   \SetString{\appendixname}{Annexe}
1203   \SetString{\contentsname}{Table des mati\`eres}
1204   \SetString{\listfigurename}{Table des figures}
1205   \SetString{\listtablename}{Liste des tableaux}
1206   \SetString{\indexname}{Index}
1207   \SetString{\figurename}{\FBfigtabshape Figure}
1208   \SetString{\tablename}{\FBfigtabshape Table}
1209   \SetString{\pagename}{page}
1210   \SetString{\seename}{voir}
1211   \SetString{\alsoname}{voir aussi}
1212   \SetString{\enclname}{P.~J. }
1213   \SetString{\ccname}{Copie \`a }
1214   \SetString{\headtoname}{}
1215   \SetString{\proofname}{D\`emonstration}
1216   \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`.

```
1217   \SetString{\partfirst}{Premi\`ere}
1218   \SetString{\partsecond}{Deuxi\`eme}
1219   \SetString{\partnameord}{partie}
1220   \SetStringLoop{ordinal#1}{%
1221     \partfirst,\partsecond,Troisi\`eme,Quatri\`eme, Cinqui\`eme,%
1222     Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
```

```

1223 Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1224 Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1225 Vingt\`eme}
1226 \AfterBabelCommands{%
1227   \DeclareRobustCommand*\FB@emptypart{}\def\thepart{\unskip}}%
1228   \DeclareRobustCommand*\FB@partname{%
1229     \ifFBPartNameFull
1230       \csname ordinal\romannumeral\value{part}\endcsname\space
1231       \partnameord\FB@emptypart
1232     \else
1233       Partie%
1234     \fi}%
1235   }
1236   \SetString{\partname}{\FB@partname}
1237 \EndBabelCommands

```

2.8 Figure and table captions

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

```

1238 \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).

```

1239 \bgroup
1240 \catcode\:=12 \catcode\>=12 \relax
1241 \long\gdef\STD@makecaption#1#2{%
1242   \vskip\abovcaptionskip
1243   \sbox\@tempboxa{#1: #2}%
1244   \ifdim \wd\@tempboxa >\hsize
1245     #1: #2\par
1246   \else
1247     \global \@minipagefalse
1248     \hb@xt\@hsize{\hfil\box\@tempboxa\hfil}%
1249   \fi

```

```

1250 \vskip\belowcaptionskip}
1251 \egroup

```

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

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

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

Enable the standard warning only if high punctuation is active.

```

1252 \newif\if@FBwarning@capsep
1253 \ifFB@active@punct\@FBwarning@capseptrue\fi
1254 \newcommand*\CaptionSeparator{\space\textendash\space}
1255 \def\FBCaption@Separator{: }
1256 \long\def\FB@makecaption#1#2{%
1257   \vskip\abovecaptionskip
1258   \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1259   \ifdim \wd\@tempboxa >\hsize
1260     #1\FBCaption@Separator #2\par
1261   \else
1262     \global \@minipagefalse
1263     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1264   \fi
1265   \vskip\belowcaptionskip}

```

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

```

1266 \@ifclassloaded{acmart}{\@FBwarning@capsepfalse}{}
1267 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1268 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1269 \@ifclassloaded{amsdtx}{\@FBwarning@capsepfalse}{}
1270 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1271 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1272 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1273 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}

```

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

```

1274 \newif\ifFB@koma
1275 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1276 \@ifclassloaded{scrartcl}{\@FBwarning@capsepfalse\FB@komatrue}{}
1277 \@ifclassloaded{scrbook}{\@FBwarning@capsepfalse\FB@komatrue}{}
1278 \@ifclassloaded{scrreprt}{\@FBwarning@capsepfalse\FB@komatrue}{}

```

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

```

1279 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1280 \ifdefined\@makecaption\else\@FBwarning@capsepfalse\fi

```

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

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

```
1281 \newcounter{FBcaption@count}
1282 \@ifpackageloaded{caption3}{\addtocounter{FBcaption@count}{4}}{}
1283 \@ifpackageloaded{subcaption}{\addtocounter{FBcaption@count}{2}}{}
1284 \@ifpackageloaded{floatrow}{\stepcounter{FBcaption@count}}{}
```

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

```
1285 \AtBeginDocument{%
1286   \ifx\@makecaption\STD@makecaption
1287     \global\let\@makecaption\FB@makecaption
```

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

```
1288   \ifBOldFigTabCaptions
1289   \else
1290     \def\FBCaption@Separator{\ifBFfrench\space\fi : }%
1291   \fi
1292   \ifBCustomiseFigTabCaptions
1293     \ifB@mainlanguage@FR
1294     \def\FBCaption@Separator{\CaptionSeparator}%
1295   \fi
1296   \fi
1297   \@FBwarning@capsepfalse
1298 \fi
```

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

```
1299 \@ifpackageloaded{caption3}{%
1300   \ifnum\value{FBcaption@count}=0 \@FBwarning@capsepfalse\fi
1301 }{}%
1302 \if@FBwarning@capsep
1303   \ifnum\value{FBcaption@count}>0
```

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

```
1304   \FBWarning
1305     {Figures' and tables' captions might look like\MessageBreak
1306     `Figure 1:' in French instead of `Figure 1 :'.\MessageBreak
1307     If you have loaded any of the packages caption,\MessageBreak
1308     subcaption or floatrow BEFORE babel/french,\MessageBreak
1309     please move them AFTER babel/french.\MessageBreak
1310     If one of them is loaded by your class,\MessageBreak
1311     you can still add AFTER babel/french\MessageBreak
1312     \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1313     \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1314     ... live with it; reported}%
1315   \else
```

caption3.sty hasn't been loaded at all.

```

1316     \FBWarning
1317     {Figures' and tables' captions might look like\MessageBreak
1318     `Figure 1:' in French instead of `Figure 1 :'.\MessageBreak
1319     If it happens, see your class documentation to\MessageBreak
1320     fix this issue or add AFTER babel/french\MessageBreak
1321     \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1322     \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1323     or ... live with it; reported}%
1324   \fi
1325 \fi
1326 \let\FB@makecaption\relax
1327 \let\STD@makecaption\relax
1328 }

```

2.9 Dots...

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

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

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

```

1329 \ifFBunicode
1330   \let\FBtextellipsis\textellipsis
1331 \else
1332   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1333   \DeclareTextCommandDefault{\FBtextellipsis}{%
1334     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}
1335 \fi

```

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

```

1336 \newcommand*{\Tdots@}{\@xp\textellipsis}
1337 \newcommand*{\Mdots@}{\@xp\mdots}
1338 \AtBeginDocument{\DeclareRobustCommand*{\dots}{\relax
1339   \csname\ifmmode M\else T\fi dots@\endcsname}%
1340   \ifdefined\@xp\else\let\@xp\relax\fi
1341   \ifdefined\mdots\else\let\Mdots@mathellipsis\fi
1342 }
1343 \def\bbl@frenchdots{\babel@save\Tdots@ \let\Tdots@\FBtextellipsis}
1344 \addto\extrasfrench{\bbl@frenchdots}

```

2.10 More checks about packages' loading order

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

```
1345 \ifFB@active@punct
1346   \@ifpackageloaded{listings}
1347     {\AtBeginDocument{%
1348       \FBWarning{Please load the "listings" package\MessageBreak
1349                 AFTER babel/french; reported}}}%
1350   }{}
1351 \fi
```

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

```
1352 \newif\if@FBwarning@natbib
1353 \ifFB@active@punct
1354   \@ifpackageloaded{natbib}{}{\@FBwarning@natbibtrue}
1355 \fi
1356 \AtBeginDocument{%
1357   \if@FBwarning@natbib
1358     \@ifpackageloaded{natbib}{}{\@FBwarning@natbibfalse}%
1359   \fi
1360   \if@FBwarning@natbib
1361     \FBWarning{Please load the "natbib" package\MessageBreak
1362               BEFORE babel/french; reported}%
1363   \fi
1364 }
```

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

```
1365 \newif\if@FBwarning@beamerarticle
1366 \@ifpackageloaded{beamerarticle}{}{\@FBwarning@beamerarticlettrue}
1367 \AtBeginDocument{%
1368   \if@FBwarning@beamerarticle
1369     \@ifpackageloaded{beamerarticle}{}%
1370                               {\@FBwarning@beamerarticlefalse}%
1371   \fi
1372   \if@FBwarning@beamerarticle
1373     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1374               BEFORE babel/french; reported}%
1375   \fi
1376 }
```

2.11 Setup options: keyval stuff

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

Option processing can occur either in `\frenchsetup{}`, but *only for options explicitly set by `\frenchsetup{}`*, or 'AtBeginDocument'; any option affecting `\extrasfrench{}` *must* be processed by `\frenchsetup{}`: when French is the main language,

`\extrasfrench{}` is executed by Babel when it switches the main language and this occurs *before* reading the stuff postponed by babel-french ‘AtBeginDocument’. Reexecuting `\extrasfrench{}` is an option which was used up to v2.6h, it has been dropped in v3.0a because of its side-effects (f.i. `\babel@save` and `\babel@savevariable` did not work for French).

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

```
1377 \newcommand*{\frenchsetup}[1]{%
1378   \setkeys{FB}{#1}%
1379 }%
1380 \@onlypreamble\frenchsetup
```

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

```
1381 \let\frenchbsetup\frenchsetup
1382 \@onlypreamble\frenchbsetup
```

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

```
1383 \newif\ifFBShowOptions
1384 \newif\ifFBStandardLayout           \FBStandardLayouttrue
1385 \newif\ifFBGlobalLayoutFrench       \FBGlobalLayoutFrenchtrue
1386 \newif\ifFBReduceListSpacing
1387 \newif\ifFBStandardListSpacing      \FBStandardListSpacingtrue
1388 \newif\ifFBListOldLayout
1389 \newif\ifFBListItemsAsPar
1390 \newif\ifFBCompactItemize
1391 \newif\ifFBStandardItemizeEnv        \FBStandardItemizeEnvtrue
1392 \newif\ifFBStandardEnumerateEnv     \FBStandardEnumerateEnvtrue
1393 \newif\ifFBStandardItemLabels       \FBStandardItemLabeltrue
1394 \newif\ifFBStandardLists            \FBStandardListtrue
1395 \newif\ifFBIndentFirst
1396 \newif\ifFBFrenchFootnotes
1397 \newif\ifFBAutoSpaceFootnotes
1398 \newif\ifFBOriginalTypewriter
1399 \newif\ifFBThinColonSpace
1400 \newif\ifFBThinSpaceInFrenchNumbers
1401 \newif\ifFBFrenchSuperscripts       \FBFrenchSuperscripttrue
1402 \newif\ifFBLowercaseSuperscripts    \FBLowercaseSuperscripttrue
1403 \newif\ifFBPartNameFull             \FBPartNameFulltrue
1404 \newif\ifFBCustomiseFigTabCaptions
1405 \newif\ifFBOldFigTabCaptions
1406 \newif\ifFBSmallCapsFigTabCaptions \FBSmallCapsFigTabCaptiontrue
1407 \newif\ifFBSuppressWarning
1408 \newif\ifFBINGuillSpace
```

The defaults values of these flags have been chosen so that babel-french does not change anything regarding the global layout. `\bbl@main@language`, set by the last option of Babel, controls the global layout of the document. ‘AtEndOfPackage’ we check the main language in `\bbl@main@language`; if it is French (or a French dialect) the values of some flags have to be changed to ensure a French looking layout for the whole document (even in parts written in languages other than French); the end-user will then be able to customise the values of all these flags with `\frenchsetup{}`.

The following patch is for koma-script classes: the `\partformat` command, defined as `\partname~\thepart\autodot`, is incompatible with our redefinition of `\partname`.

```

1409 \ifFB@koma
1410   \ifdefined\partformat
1411     \def\FB@partformat@fix{%
1412       \ifFBPartNameFull
1413         \babel@save\partformat
1414         \renewcommand*{\partformat}{\partname}%
1415       \fi}
1416   \addto\extrasfrench{\FB@partformat@fix}%
1417 \fi
1418 \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.

```

1419 \def\FB@french{french}
1420 \def\FB@acadian{acadian}
1421 \newif\ifFB@mainlanguage@FR
1422 \AtEndOfPackage{%
1423   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1424   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1425   \fi
1426   \ifFB@mainlanguage@FR
1427     \FBGlobalLayoutFrenchtrue
1428     \@ifclassloaded{beamer}%
1429       {\PackageInfo{french.ldf}{%
1430         No list customisation for the beamer class,%
1431         \MessageBreak reported}}%
1432       {\@ifpackageloaded{beamerarticle}%
1433         {\FBStandardItemLabelsfalse
1434          \FBStandardListSpacingfalse
1435          \PackageInfo{french.ldf}{%
1436            Minimal list customisation for the beamerarticle%
1437            \MessageBreak package; reported}}}%

```

Otherwise customise lists “à la française”:

```

1438       {\FBStandardListSpacingfalse
1439        \FBStandardItemizeEnvfalse
1440        \FBStandardEnumerateEnvfalse
1441        \FBStandardItemLabelsfalse}%
1442     }
1443     \FBIndentFirsttrue
1444     \FBFrenchFootnotesttrue
1445     \FBAutoSpaceFootnotesttrue
1446     \FBCustomiseFigTabCaptionstrue
1447   \else
1448     \FBGlobalLayoutFrenchfalse
1449   \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.

```
1450 \RequirePackage{keyval}%
1451 \define@key{FB}{ShowOptions}[true]%
1452 {\csname FBShowOptions#1\endcsname}%
```

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

```
1453 \define@key{FB}{StandardLayout}[true]%
1454 {\ifFB@mainlanguage@FR
1455   \csname FBStandardLayout#1\endcsname
1456   \else
1457     \PackageWarning{french.ldf}%
1458       {Option `StandardLayout' skipped:\MessageBreak
1459         French is *not* babel's last option.\MessageBreak
1460         Reported}%
1461   \fi
1462   \ifFBStandardLayout
1463     \FBStandardListSpacingtrue
1464     \FBStandardItemizeEnvtrue
1465     \FBStandardItemLabelstrue
1466     \FBStandardEnumerateEnvtrue
1467     \FBIndentFirstfalse
1468     \FBFrenchFootnotesfalse
1469     \FBAutoSpaceFootnotesfalse
1470     \FBGlobalLayoutFrenchfalse
1471   \else
1472     \FBStandardListSpacingfalse
1473     \FBStandardItemizeEnvfalse
1474     \FBStandardItemLabelsfalse
1475     \FBStandardEnumerateEnvfalse
1476     \FBIndentFirsttrue
1477     \FBFrenchFootnotestruetrue
1478     \FBAutoSpaceFootnotestruetrue
1479   \fi}%
1480 \define@key{FB}{GlobalLayoutFrench}[true]%
1481 {\ifFB@mainlanguage@FR
1482   \csname FBGlobalLayoutFrench#1\endcsname
1483   \else
1484     \PackageWarning{french.ldf}%
1485       {Option `GlobalLayoutFrench' skipped:\MessageBreak
1486         French is *not* babel's last option.\MessageBreak
1487         Reported}%
1488   \fi}%
1489 \define@key{FB}{ReduceListSpacing}[true]%
1490 {\csname FBReduceListSpacing#1\endcsname
1491   \ifFBReduceListSpacing \FBStandardListSpacingfalse
1492   \else \FBStandardListSpacingtrue\fi
1493 }%
1494 \define@key{FB}{StandardListSpacing}[true]%
```

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.

```
1489 \define@key{FB}{ReduceListSpacing}[true]%
1490 {\csname FBReduceListSpacing#1\endcsname
1491   \ifFBReduceListSpacing \FBStandardListSpacingfalse
1492   \else \FBStandardListSpacingtrue\fi
1493 }%
1494 \define@key{FB}{StandardListSpacing}[true]%
```

```

1495         {\csname FBStandardListSpacing#1\endcsname}%
1496 \define@key{FB}{ListOldLayout}[true]%
1497     {\csname FBListOldLayout#1\endcsname
1498     \ifFBListOldLayout
1499         \FBStandardEnumerateEnvtrue
1500         \renewcommand*{\FrenchLabelItem}{\textendash}%
1501         \fi}%
1502 \define@key{FB}{CompactItemize}[true]%
1503     {\csname FBCompactItemize#1\endcsname
1504     \ifFBCompactItemize
1505         \FBStandardItemizeEnvfalse
1506         \FBStandardEnumerateEnvfalse
1507     \else
1508         \FBStandardItemizeEnvtrue
1509         \FBStandardEnumerateEnvtrue
1510     \fi}%
1511 \define@key{FB}{StandardItemizeEnv}[true]%
1512     {\csname FBStandardItemizeEnv#1\endcsname}%
1513 \define@key{FB}{StandardEnumerateEnv}[true]%
1514     {\csname FBStandardEnumerateEnv#1\endcsname}%
1515 \define@key{FB}{StandardItemLabels}[true]%
1516     {\csname FBStandardItemLabels#1\endcsname}%
1517 \define@key{FB}{ItemLabels}%
1518     {\renewcommand*{\FrenchLabelItem}{#1}}%
1519 \define@key{FB}{ItemLabeli}%
1520     {\renewcommand*{\Frlabelitemi}{#1}}%
1521 \define@key{FB}{ItemLabelii}%
1522     {\renewcommand*{\Frlabelitemii}{#1}}%
1523 \define@key{FB}{ItemLabeliii}%
1524     {\renewcommand*{\Frlabelitemiii}{#1}}%
1525 \define@key{FB}{ItemLabeliv}%
1526     {\renewcommand*{\Frlabelitemiv}{#1}}%
1527 \define@key{FB}{StandardLists}[true]%
1528     {\csname FBStandardLists#1\endcsname
1529     \ifFBStandardLists
1530         \FBStandardListSpacingtrue
1531         \FBStandardItemizeEnvtrue
1532         \FBStandardEnumerateEnvtrue
1533         \FBStandardItemLabelstrue
1534     \else
1535         \FBStandardListSpacingfalse
1536         \FBStandardItemizeEnvfalse
1537         \FBStandardEnumerateEnvfalse
1538         \FBStandardItemLabelsfalse
1539     \fi}%
1540 \define@key{FB}{ListItemsAsPar}[true]%
1541     {\csname FBListItemsAsPar#1\endcsname}%
1542 \define@key{FB}{IndentFirst}[true]%
1543     {\csname FBIndentFirst#1\endcsname}%
1544 \define@key{FB}{FrenchFootnotes}[true]%
1545     {\csname FBFrenchFootnotes#1\endcsname}%
1546 \define@key{FB}{AutoSpaceFootnotes}[true]%
1547     {\csname FBAutoSpaceFootnotes#1\endcsname}%

```

```

1548 \define@key{FB}{AutoSpacePunctuation}[true]%
1549     {\csname FBAutoSpacePunctuation#1\endcsname}%
1550 \define@key{FB}{OriginalTypewriter}[true]%
1551     {\csname FBOriginalTypewriter#1\endcsname}%
1552 \define@key{FB}{ThinColonSpace}[true]%
1553     {\csname FBThinColonSpace#1\endcsname
1554     \ifFBThinColonSpace
1555     \renewcommand*{\FBcolonspace}{\FBthinspace}%
1556     \fi}%
1557 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1558     {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1559 \define@key{FB}{FrenchSuperscripts}[true]%
1560     {\csname FBFrenchSuperscripts#1\endcsname}%
1561 \define@key{FB}{LowercaseSuperscripts}[true]%
1562     {\csname FBLowercaseSuperscripts#1\endcsname}%
1563 \define@key{FB}{PartNameFull}[true]%
1564     {\csname FBPartNameFull#1\endcsname}%
1565 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1566     {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1567 \define@key{FB}{OldFigTabCaptions}[true]%
1568     {\csname FBOldFigTabCaptions#1\endcsname
1569     \ifFBOldFigTabCaptions
1570     \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1571     \def\FBCaption@Separator{\CaptionSeparator}}%
1572     \addto\extrasfrench{\FB@capsep@fix}%
1573     \ifdefined\extrasacadian
1574     \addto\extrasacadian{\FB@capsep@fix}%
1575     \fi
1576     \fi}%
1577 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1578     {\csname FBSmallCapsFigTabCaptions#1\endcsname
1579     \ifFBSmallCapsFigTabCaptions
1580     \let\FBfigtabshape\scshape
1581     \else
1582     \let\FBfigtabshape\relax
1583     \fi}%
1584 \define@key{FB}{SuppressWarning}[true]%
1585     {\csname FBSuppressWarning#1\endcsname
1586     \ifFBSuppressWarning
1587     \renewcommand{\FBWarning}[1]{}%
1588     \fi}%

```

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

```

1589 \define@key{FB}{INGuillSpace}[true]%
1590     {\csname FBINGuillSpace#1\endcsname
1591     \ifFBINGuillSpace
1592     \renewcommand*{\FBguillspace}{\space}%
1593     \fi}%
1594 \define@key{FB}{InnerGuillSingle}[true]%
1595     {\csname FBInnerGuillSingle#1\endcsname}%
1596 \define@key{FB}{EveryParGuill}[open]%
1597     {\expandafter\let\expandafter
1598     \FBeveryparguill\csname FBguill#1\endcsname
1599     \ifx\FBeveryparguill\FBguillopen

```

```

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

```

1630 \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1631 {\ifFB@luatex@punct
1632   \csname FBucsNBSP#1\endcsname
1633   \ifFBucsNBSP \FB@ucsNBSP=1 \fi
1634 \else
1635   \FBWarning{Option `UnicodeNoBreakSpaces' skipped:%
1636               \MessageBreak this option is for
1637               LuaTeX *only*.\MessageBreak Reported}%
1638   \fi
1639 }%

```

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

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

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

```
1640 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
1641 \define@key{FB}{og}%
1642      {\ifFBunicode
```

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

```
1643      \ifFB@luatex@punct
1644      \FB@addGUIILspace=1 \relax
1645      \fi
```

then with XeTeX it is a bit more tricky:

```
1646      \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).
```

```
1647      \XeTeXcharclass"13 = \FB@guilo
1648      \XeTeXcharclass"AB = \FB@guilo
1649      \XeTeXcharclass"A0 = \FB@guilnul
1650      \XeTeXcharclass"202F = \FB@guilnul
1651      \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1652      \ifFB@active@punct
1653      \FBWarning{Option og=« not supported with this version
1654                  of\MessageBreak LuaTeX/XeTeX; reported}%
1655      \fi
1656      \else
```

This is for conventional TeX engines:

```
1657      \newcommand*{\FB@og}{%
1658      \ifFBfrench
1659      \ifFB@spacing\FB@og\ignorespaces
1660      \else\guillemotleft
1661      \fi
1662      \else\guillemotleft\fi}%
1663      \AtBeginDocument{%
1664      \ifdefined\uc@dclc
```

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

```
1665      \uc@dclc{171}{default}\FB@og}%
1666      \else
```

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

```
1667      \FB@parse#1\endparse
1668      \ifx\FB@second\@empty
```

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

```
1669      \ifdefined\mule@def
1670      \mule@def{11}\FB@og}%
1671      \else
```

```

1672             \ifdefined\DeclareInputText
1673             \@tempcnta`#1\relax
1674             \DeclareInputText{\the\@tempcnta}{\FB@og}%
1675             \else
Package inputenc not loaded, no way...
1676             \FBWarning{Option `og' requires package
1677             inputenc;\MessageBreak reported}%
1678             \fi
1679             \fi
1680             \else
This means multi-byte character encoding, we assume UTF-8
1681             \DeclareUnicodeCharacter{00AB}{\FB@og}%
1682             \fi
1683             \fi}%
1684             \fi
1685             }%
Same code for the closing quote.
1686 \define@key{FB}{fg}%
1687 {\iffBunicode
1688   \iffB@luatex@punct
1689   \FB@addGUIIspace=1 \relax
1690   \fi
1691   \iffB@xetex@punct
1692   \XeTeXcharclass"14 = \FB@guilf
1693   \XeTeXcharclass"BB = \FB@guilf
1694   \XeTeXcharclass"A0 = \FB@guilnul
1695   \XeTeXcharclass"202F = \FB@guilnul
1696   \fi
1697   \iffB@active@punct
1698   \FBWarning{Option fg=» not supported with this version
1699   of\MessageBreak LuaTeX/XeTeX; reported}%
1700   \fi
1701   \else
1702   \newcommand*{\FB@fg}{%
1703     \iffBfrench
1704     \iffB@spacing\FB@fg
1705     \else\guillemotright
1706     \fi
1707     \else\guillemotright\fi}%
1708   \AtBeginDocument{%
1709     \ifdefined\uc@dcl
1710     \uc@dcl{187}{default}{\FB@fg}%
1711     \else
1712     \FB@parse#1\endparse
1713     \ifx\FB@second\@empty
1714     \ifdefined\mule@def
1715     \mule@def{27}{\FB@fg}%
1716     \else
1717     \ifdefined\DeclareInputText
1718     \@tempcnta`#1\relax
1719     \DeclareInputText{\the\@tempcnta}{\FB@fg}%
1720     \else

```



```

1721             \FBWarning{Option `fg' requires package
1722                 inputenc;\MessageBreak reported}%
1723         \fi
1724     \fi
1725     \else
1726         \DeclareUnicodeCharacter{00BB}{\FB@fg}%
1727     \fi
1728 \fi}%
1729 \fi
1730 }%
1731 }

```

\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.

```

1732 \newcommand*{\FBprocess@options}{%
Update flags if a package customising lists has been loaded, currently: enumitem,
paralist, enumerate.
1733 \@ifpackageloaded{enumitem}{%
1734     \ifFBStandardItemizeEnv
1735     \else
1736         \FBStandardItemizeEnvtrue
1737         \PackageInfo{french.ldf}%
1738             {Setting StandardItemizeEnv=true for\MessageBreak
1739             compatibility with enumitem package,\MessageBreak
1740             reported}%
1741     \fi
1742     \ifFBStandardEnumerateEnv
1743     \else
1744         \FBStandardEnumerateEnvtrue
1745         \PackageInfo{french.ldf}%
1746             {Setting StandardEnumerateEnv=true for\MessageBreak
1747             compatibility with enumitem package,\MessageBreak
1748             reported}%
1749     \fi}%
1750 \@ifpackageloaded{paralist}{%
1751     \ifFBStandardItemizeEnv
1752     \else
1753         \FBStandardItemizeEnvtrue
1754         \PackageInfo{french.ldf}%
1755             {Setting StandardItemizeEnv=true for\MessageBreak
1756             compatibility with paralist package,\MessageBreak
1757             reported}%
1758     \fi
1759     \ifFBStandardEnumerateEnv
1760     \else
1761         \FBStandardEnumerateEnvtrue
1762         \PackageInfo{french.ldf}%
1763             {Setting StandardEnumerateEnv=true for\MessageBreak

```

```

1764         compatibility with paralist package,\MessageBreak
1765         reported}%
1766     \fi}}%
1767 \@ifpackageloaded{enumerate}{%
1768     \ifFBStandardEnumerateEnv
1769     \else
1770         \FBStandardEnumerateEnvtrue
1771         \PackageInfo{french.ldf}%
1772         {Setting StandardEnumerateEnv=true for\MessageBreak
1773         compatibility with enumerate package,\MessageBreak
1774         reported}%
1775     \fi}}%

```

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

```

1776 \def\FB@ufl{\update@frenchlists}
1777 \ifFB@mainlanguage@FR
1778     \update@frenchlists
1779 \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.

```

1780 \ifFBAutoSpacePunctuation
1781     \autospace@beforeFDP
1782 \else
1783     \noautospace@beforeFDP
1784 \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.

```

1785 \ifFBOriginalTypewriter
1786 \else
1787     \let\ttfamilyORI\ttfamily
1788     \let\rmfamilyORI\rmfamily
1789     \let\sffamilyORI\sffamily
1790     \let\ttfamily\ttfamilyFB
1791     \let\rmfamily\rmfamilyFB
1792     \let\sffamily\sffamilyFB
1793 \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.

```

1794 \@ifpackageloaded{numprint}%
1795 {
1796     \ifnprt@autolanguage
1797     \providecommand*{\npstylefrench}{}%
1798     \ifFBThinSpaceInFrenchNumbers
1799     \renewcommand*{\FBthousandsep}{\,}%

```

```

1799     \fi
1800     \g@addto@macro\npstylefrench{\npthousandsep{\FBthousandsep}}%
1801     \fi
1802     }{}%

```

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.

```

1803 \ifFBFrenchSuperscripts
1804   \DeclareRobustCommand*\up*{\@ifstar{\FB@up@fake}{\fup}}%
1805 \else
1806   \DeclareRobustCommand*\up*{\@ifstar{\FB@up@fake}%
1807                                   {\textsuperscript}}%
1808 \fi

```

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

```

1809 \ifFBLowercaseSuperscripts
1810 \else
1811   \renewcommand*\FB@lc{1}{##1}%
1812 \fi

```

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

```

1813 \ifFBCustomiseFigTabCaptions
1814   \ifFB@koma
1815     \renewcommand*\captionformat{\CaptionSeparator}%
1816   \fi
1817   \@ifclassloaded{memoir}%
1818     {\captiondelim{\CaptionSeparator}}{}%
1819   \@ifclassloaded{beamer}%
1820     {\defbeamertemplate{caption label separator}{FBcustom}{%
1821       \CaptionSeparator}%
1822     \setbeamertemplate{caption label separator}[FBcustom]}{}%
1823 \else

```

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

```

1824   \ifFB@koma
1825     \renewcommand*\captionformat{\autospace@beforeFDP : }%
1826   \fi
1827   \@ifclassloaded{memoir}%
1828     {\captiondelim{\autospace@beforeFDP : }}%
1829   }{}%
1830   \@ifclassloaded{beamer}%
1831     {\defbeamertemplate{caption label separator}{FBcolon}{%
1832       {\autospace@beforeFDP : }}%
1833     \setbeamertemplate{caption label separator}[FBcolon]%
1834     }{}%
1835 \fi

```

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

```

1836 \ifFBShowOptions
1837   \GenericWarning{* }{}%
1838   *** List of possible options for babel-french ***\MessageBreak
1839   [Default values between brackets when french is loaded *LAST*]%

```

```

1840 \MessageBreak
1841 ShowOptions [false]\MessageBreak
1842 StandardLayout [false]\MessageBreak
1843 GlobalLayoutFrench [true]\MessageBreak
1844 PartNameFull [true]\MessageBreak
1845 IndentFirst [true]\MessageBreak
1846 ListItemsAsPar [false]\MessageBreak
1847 StandardListSpacing [false]\MessageBreak
1848 StandardItemizeEnv [false]\MessageBreak
1849 StandardEnumerateEnv [false]\MessageBreak
1850 StandardItemLabels [false]\MessageBreak
1851 ItemLabels=\textendash, \textbullet,
1852 \protect\ding{43},... [\textendash]\MessageBreak
1853 ItemLabeli=\textendash, \textbullet,
1854 \protect\ding{43},... [\textendash]\MessageBreak
1855 ItemLabelii=\textendash, \textbullet,
1856 \protect\ding{43},... [\textendash]\MessageBreak
1857 ItemLabeliii=\textendash, \textbullet,
1858 \protect\ding{43},... [\textendash]\MessageBreak
1859 ItemLabeliv=\textendash, \textbullet,
1860 \protect\ding{43},... [\textendash]\MessageBreak
1861 StandardLists [false]\MessageBreak
1862 ListOldLayout [false]\MessageBreak
1863 FrenchFootnotes [true]\MessageBreak
1864 AutoSpaceFootnotes [true]\MessageBreak
1865 AutoSpacePunctuation [true]\MessageBreak
1866 ThinColonSpace [false]\MessageBreak
1867 OriginalTypewriter [false]\MessageBreak
1868 UnicodeNoBreakSpaces [false]\MessageBreak
1869 og= <left quote character>, fg= <right quote character>%
1870 INGuillSpace [false]\MessageBreak
1871 EveryParGuill=open, close, none [open]\MessageBreak
1872 EveryLineGuill=open, close, none
1873 [open in LuaTeX, none otherwise]\MessageBreak
1874 InnerGuillSingle [false]\MessageBreak
1875 ThinSpaceInFrenchNumbers [false]\MessageBreak
1876 SmallCapsFigTabCaptions [true]\MessageBreak
1877 CustomiseFigTabCaptions [true]\MessageBreak
1878 OldFigTabCaptions [false]\MessageBreak
1879 FrenchSuperscripts [true]\MessageBreak
1880 LowercaseSuperscripts [true]\MessageBreak
1881 SuppressWarning [false]\MessageBreak
1882 \MessageBreak
1883 *****%
1884 \MessageBreak\protect\frenchsetup{ShowOptions}}
1885 \fi
1886 }

```

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.

```
1887 \AtBeginDocument{%
```

```
1888 \providecommand*{\xspace}{\relax}%
```

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

```
1889 \ifdefined\pdfstringdefDisableCommands
1890 \pdfstringdefDisableCommands{%
1891     \let\up\relax
1892     \let\up\relax
1893     \let\degre\textdegree
1894     \let\degres\textdegree
1895     \def\ieme{e\xspace}%
1896     \def\iemes{es\xspace}%
1897     \def\ier{er\xspace}%
1898     \def\iers{ers\xspace}%
1899     \def\iere{re\xspace}%
1900     \def\ieres{res\xspace}%
1901     \def\FrenchEnumerate#1{#1\degre\space}%
1902     \def\FrenchPopularEnumerate#1{#1\degre)\space}%
1903     \def\No{N\degre\space}%
1904     \def\no{n\degre\space}%
1905     \def\Nos{N\degre\space}%
1906     \def\nos{n\degre\space}%
1907     \def\FB@og{\guillemotleft\space}%
1908     \def\FB@fg{\space\guillemotright}%
1909     \def\frquote#1{\FB@og #1\FB@fg}%
1910     \def\at{@}%
1911     \def\circonflexe{\string^}%
1912     \def\tild{\string~}%
1913     \def\boi{\textbackslash}%
1914     \let\bsc\textsc
1915 }%
1916 \fi
```

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

```
1917 \FBprocess@options
```

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

```
1918 \ifFBucsNBSP
1919 \renewcommand*{\FBmedkern}{\char"202F\relax}%
1920 \renewcommand*{\FBthickkern}{\char"A0\relax}%
1921 \ifFBThinSpaceInFrenchNumbers
1922 \renewcommand*{\FBthousandsep}{\char"202F\relax}%
1923 \else
1924 \renewcommand*{\FBthousandsep}{\char"A0\relax}%
1925 \fi
1926 \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*`!

```
1927 \begingroup
1928 \newcommand{\FBOTone}{OT1}%
1929 \ifx\encodingdefault\FBOTone
1930 \FBWarning{OT1 encoding should not be used for French.%
```

```

1931 \MessageBreak
1932 Add \protect\usepackage[T1]{fontenc} to the
1933 preamble\MessageBreak of your document; reported}%
1934 \fi
1935 \endgroup
1936 }

```

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.

```

1937 \let\listORI\list
1938 \let\endlistORI\endlist
1939 \def\FB@listVsettings{%
1940     \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1941     \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
1942     \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1943     \setlength{\partopsep}{0.4ex plus 0.2ex minus 0.2ex}%

```

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

```

1944     \@tempdima=\parskip
1945     \addtolength{\topsep}{-\@tempdima}%
1946     \addtolength{\partopsep}{\@tempdima}%
1947 }
1948 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1949 \let\endlistFB\endlist

```

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

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

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

```

\Frlabelitemi 1950 \newcommand*{\FrenchLabelItem}{\textendash}
\Frlabelitemii 1951 \newcommand*{\Frlabelitemii}{\FrenchLabelItem}
\Frlabelitemiii 1952 \newcommand*{\Frlabelitemiii}{\FrenchLabelItem}
\Frlabelitemiv 1953 \newcommand*{\Frlabelitemiv}{\FrenchLabelItem}
1954 \newcommand*{\Frlabelitemiv}{\FrenchLabelItem}

```

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

```

1955 \newdimen\listindentFB
1956 \setlength{\listindentFB}{-1pt}
1957 \newdimen\descindentFB
1958 \setlength{\descindentFB}{-1pt}
1959 \newdimen\labelindentFB
1960 \setlength{\labelindentFB}{-1pt}
1961 \newdimen\labelwidthFB
1962 \setlength{\labelwidthFB}{-1pt}

```

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

```

1963 \newdimen\leftmarginFB
1964 \def\FB@listHsettings{%
1965   \ifFBListItemsAsPar

```

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

```

1966     \itemindent=\labelindentFB
1967     \advance\itemindent by \labelwidthFB
1968     \advance\itemindent by \labelsep
1969     \leftmargini\z@
1970     \bbl@for\FB@dp {2, 3, 4, 5, 6}%
1971       {\csname leftmargin\romannumeral\FB@dp\endcsname =
1972         \labelindentFB}%
1973   \else

```

Default layout: labels hanging into the left margin.

```

1974     \leftmarginFB=\labelwidthFB
1975     \advance\leftmarginFB by \labelsep
1976     \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1977       {\csname leftmargin\romannumeral\FB@dp\endcsname =
1978         \leftmarginFB}%
1979     \advance\leftmargini by \listindentFB
1980   \fi
1981   \leftmargin=\csname leftmargin%
1982     \ifnum\@listdepth=\@ne i\else ii\fi\endcsname
1983 }

```

`\itemizeFB` New environment for French itemize-lists.
`\FB@itemizesettings` `\FB@itemizesettings` does two things: first suppress all vertical spaces including glue
unless option `StandardListSpacing` is set, then set horizontal indentations according

to `\FB@listHsettings` unless option `ListOldLayout` is `true` (compatibility with lists up to v. 2.5k).

```

1984 \def\FB@itemizesettings{%
1985   \ifFBStandardListSpacing
1986   \else
1987     \setlength{\itemsep}{\z@}%
1988     \setlength{\parsep}{\z@}%
1989     \setlength{\topsep}{\z@}%
1990     \setlength{\partopsep}{\z@}%
1991     \@tempdima=\parskip
1992     \addtolength{\topsep}{-\@tempdima}%
1993     \addtolength{\partopsep}{\@tempdima}%
1994   \fi
1995   \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
1996   \ifFBListOldLayout
1997     \setlength{\leftmargin}{\labelwidth}%
1998     \addtolength{\leftmargin}{\labelsep}%
1999     \addtolength{\leftmargin}{\parindent}%
2000   \else
2001     \FB@listHsettings
2002   \fi
2003 }

```

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

```

2004 \def\itemizeFB{%
2005   \ifnum \@itemdepth >\thr@@\@toodeep\else
2006     \advance\@itemdepth by \@ne
2007     \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
2008     \expandafter
2009     \listORI
2010     \csname\@itemitem\endcsname
2011     \FB@itemizesettings
2012   \fi
2013 }
2014 \let\enditemizeFB\endlistORI

2015 \def\setlabelitemsFB{%
2016   \let\labelitemi\Frlabelitemi
2017   \let\labelitemii\Frlabelitemii
2018   \let\labelitemiii\Frlabelitemiii
2019   \let\labelitemiv\Frlabelitemiv
2020   \ifdim\labelwidthFB<\z@
2021     \settowidth{\labelwidthFB}{\FrenchLabelItem}%
2022   \fi
2023 }

2024 \def\setlistindentFB{%
2025   \ifdim\labelindentFB<\z@
2026     \ifdim\parindent=\z@
2027       \setlength{\labelindentFB}{1.5em}%
2028     \else
2029       \setlength{\labelindentFB}{\parindent}%
2030     \fi
2031   \fi

```



```

2032 \ifdim\listindentFB<\z@
2033 \ifdim\parindent=\z@
2034 \setlength{\listindentFB}{1.5em}%
2035 \else
2036 \setlength{\listindentFB}{\parindent}%
2037 \fi
2038 \fi
2039 \ifdim\descindentFB<\z@
2040 \ifFBListItemsAsPar
2041 \setlength{\descindentFB}{\labelindentFB}%
2042 \else
2043 \setlength{\descindentFB}{\listindentFB}%
2044 \fi
2045 \fi
2046 }

```

\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`.

```

2047 \def\enumerateFB{%
2048 \ifnum \@enumdepth >\thr@@\@toodeep\else
2049 \advance\@enumdepth by \@ne
2050 \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2051 \expandafter
2052 \list
2053 \csname label\@enumctr\endcsname
2054 {\FB@listHsettings
2055 \usecounter\@enumctr\def\makelabel##1{\hss\llap{##1}}}%
2056 \fi
2057 }
2058 \let\endenumerateFB\endlistORI

```

\descriptionFB Same tuning for the `description` environment (see `classes.dtx` for the original definition). Customisable `\descindentFB`, which defaults to `\listindentFB`, is added to `\itemindent` (first level only). When `\descindentFB=0pt` (1st level labels start at the left margin), `\leftmargini` is reduced to `\listindentFB` instead of `\listindentFB + \leftmarginFB`. When option `ListItemsAsPar` is turned to `true`, the `description` items are also displayed as paragraphs; `\descindentFB=0pt` can be used to push labels to the left margin.

```

2059 \def\descriptionFB{%
2060 \list{}{\FB@listHsettings
2061 \labelwidth=\z@
2062 \ifFBListItemsAsPar
2063 \itemindent=\descindentFB
2064 \else
2065 \itemindent=-\leftmargin
2066 \ifnum\@listdepth=1
2067 \ifdim\descindentFB=\z@
2068 \ifdim\listindentFB>\z@
2069 \leftmargini=\listindentFB

```

```

2070             \leftmargin=\leftmargini
2071             \itemindent=-\leftmargin
2072             \fi
2073         \else
2074             \advance\itemindent by \descindentFB
2075             \fi
2076         \fi
2077     \fi
2078     \let\makelabel\descriptionlabel}%
2079 }
2080 \let\enddescriptionFB\endlistORI

```

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

```

2081 \def\update@frenchlists{%
2082   \setlistindentFB
2083   \ifFBStandardListSpacing
2084   \else \let\list\listFB \fi
2085   \ifFBStandardItemizeEnv
2086   \else \let\itemize\itemizeFB \fi
2087   \ifFBStandardItemLabels
2088   \else \setlabelitemsFB \fi
2089   \ifFBStandardEnumerateEnv
2090   \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2091 }

```

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

In order to ensure compatibility with packages customising lists, the command \update@frenchlists should not be included in the first call to \extrasfrench which occurs *before* the relevant flags are finally set, so we define \FB@ufl as \relax, it will be redefined later 'AtBeginDocument' by \FBprocess@options as \update@frenchlists, see p. 62.

```

2092 \def\FB@ufl{\relax}
2093 \def\bbl@frenchlistlayout{%
2094   \ifFBGlobalLayoutFrench
2095   \else
2096     \babel@save\list           \babel@save\itemize
2097     \babel@save\enumerate     \babel@save\description
2098     \babel@save\labelitemi    \babel@save\labelitemii
2099     \babel@save\labelitemiii  \babel@save\labelitemiv
2100     \FB@ufl
2101   \fi
2102 }
2103 \addto\extrasfrench{\bbl@frenchlistlayout}

```

2.13 French indentation of sections

\bbl@frenchindent In French the first paragraph of each section should be indented, this is another difference with US-English. This is controlled by the flag \if@afterindent.

We will need to save the value of the flag `\if@afterindent` ‘AtBeginDocument’ before eventually changing its value.

```

2104 \def\bbl@frenchindent{%
2105   \ifFBGlobalLayoutFrench
2106   \else
2107     \babel@save\@afterindentfalse
2108   \fi
2109   \ifFBIndentFirst
2110     \let\@afterindentfalse\@afterindenttrue
2111     \@afterindenttrue
2112   \fi}
2113 \def\bbl@nonfrenchindent{%
2114   \ifFBGlobalLayoutFrench
2115     \ifFBIndentFirst
2116       \@afterindenttrue
2117     \fi
2118   \fi}
2119 \addto\extrasfrench{\bbl@frenchindent}
2120 \addto\noextrasfrench{\bbl@nonfrenchindent}

```

2.14 Formatting footnotes

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

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

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

```

2121 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2122                  {\PackageInfo{french.ldf}%
2123                     {bigfoot package in use.\MessageBreak
2124                     babel-french will NOT customise footnotes;%
2125                     \MessageBreak reported}}%
2126                  {\let\@footnotemarkORI\@footnotemark
2127                   \def\@footnotemarkFB{\leavevmode\unskip\unkern
2128                                         \,\@footnotemarkORI}%
2129                   \ifFBAutoSpaceFootnotes
2130                     \let\@footnotemark\@footnotemarkFB
2131                   \fi}%
2132                  }

```

`\@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).

```
2133 \newdimen\parindentFFN
```

```
2134 \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.

```
2135 \newcommand*{\dotFFN}{.}
```

```
2136 \newcommand*{\kernFFN}{\kern .5em}
```

```
2137 \newdimen\FBfnindent
```

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

```
2138 \ifFB@koma
```

```
2139   \let\@makefnfntextORI\@makefnfntext
```

```
2140   \let\@makefnmarkORI\@makefnmark
```

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

```
2141   \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
```

```
2142       {\thefootnotemark\dotFFN\kernFFN}
```

```
2143   \let\@makefnfntextFB\@makefnfntext
```

```
2144   \let\@makefnmarkFB\@makefnmark
```

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

```
2145   \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
```

```
2146       {\textsuperscript{\thefootnotemark}}
```

```
2147   \let\@makefnfntextTH\@makefnfntext
```

```
2148   \let\@makefnmarkTH\@makefnmark
```

Restore the original definitions.

```
2149   \let\@makefnfntext\@makefnfntextORI
```

```
2150   \let\@makefnmark\@makefnmarkORI
```

```
2151 \fi
```

Definitions for the `memoir` class:

```
2152 \ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2153   {\newcommand{\@makefnfntextFB}[1]{%
```

```
2154     \def\footscript##1{##1\dotFFN\kernFFN}%
```

```
2155     \setlength{\footmarkwidth}{\FBfnindent}%
```

```
2156     \setlength{\footmarksep}{-\footmarkwidth}%
```

```
2157     \setlength{\footparindent}{\parindentFFN}%
```

```
2158     \makefootmark #1}%
```

```
2159 }{}
```

Definitions for the beamer class:

```
2160 \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.

```
2161 {\def\@makefnfntextFB#1{%
2162   \def\insertfootnotetext{#1}%
2163   \def\insertfootnotemark{\insertfootnotemarkFB}%
2164   \usebeamertemplate***{footnote}}%
2165   \def\insertfootnotemarkFB{%
2166     \usebeamercolor[fg]{footnote mark}%
2167     \usebeamerfont*{footnote mark}%
2168     \llap{\@thefnmark}\dotFFN\kernFFN}%
2169   }{}}
```

Now the default definition of `\@makefnfntextFB` 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)!

```
2170 \providecommand*\insertfootnotemarkFB{%
2171   \parindent=\parindentFFN
2172   \rule{\z@}{footnotesep}
2173   \setbox\@tempboxa\hbox{\@thefnmark}%
2174   \ifdim\wd\@tempboxa>\z@
2175     \llap{\@thefnmark}\dotFFN\kernFFN
2176   \fi}
2177 \providecommand\@makefnfntextFB[1]{\insertfootnotemarkFB #1}
```

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

```
2178 \providecommand\localleftbox[1]{}
2179 \AtBeginDocument{%
2180   \@ifpackageloaded{bigfoot}{}%
2181   {\ifdim\parindentFFN<10in
2182     \else
2183       \parindentFFN=\parindent
2184       \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2185     \fi
2186     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2187     \addtolength{\FBfnindent}{\parindentFFN}%
2188     \let\@makefnfntextORI\@makefnfntext
2189     \ifFB@koma
```

Definition of `\@makefnfntext` for koma-script classes: running `makefnfntextORI` 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).

```

2190      \let\@makefnmarkORI\@makefnmark
2191      \long\def\@makefnmark#1{%
2192        \localleftbox{}%
2193        \let\FBeverypar@save\FBeverypar@quote
2194        \let\FBeverypar@quote\relax
2195        \ifFBFrenchFootnotes
2196          \ifx\footnote\thanks
2197            \let\@makefnmark\@makefnmarkTH
2198            \@makefnmarkTH{#1}
2199          \else
2200            \let\@makefnmark\@makefnmarkFB
2201            \@makefnmarkFB{#1}
2202          \fi
2203        \else
2204          \let\@makefnmark\@makefnmarkORI
2205          \@makefnmarkORI{#1}%
2206        \fi
2207        \let\FBeverypar@quote\FBeverypar@save
2208        \localleftbox{\FBeveryline@quote}}%
2209      \else

```

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

```

2210      \ifclassloaded{memoir}%
2211      {\ifFBFrenchFootnotes
2212        \setlength{\thanksmarkwidth}{\parindentFFN}%
2213        \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2214        \fi
2215      }{}%

```

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

```

2216      \ifclassloaded{beamer}%
2217      {\ifFBFrenchFootnotes
2218        \ifdim\parindentFFN=1.5em\else
2219          \FBWarning{%
2220            \protect\parindentFFN\space is ineffective%
2221            \MessageBreak within the beamer class.%
2222            \MessageBreak Reported}%
2223        \fi
2224      \fi
2225      }{}%

```

Definition of \@makefnmark for all other classes:

```

2226      \long\def\@makefnmark#1{%
2227        \localleftbox{}%
2228        \let\FBeverypar@save\FBeverypar@quote
2229        \let\FBeverypar@quote\relax
2230        \ifFBFrenchFootnotes
2231          \@makefnmarkFB{#1}%
2232        \else
2233          \@makefnmarkORI{#1}%
2234        \fi

```

```

2235         \let\FBeverypar@quote\FBeverypar@save
2236         \localleftbox{\FBeveryline@quote}}}%
2237     \fi
2238 }%
2239 }

```

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

```

2240 \newcommand*{\AddThinSpaceBeforeFootnotes}{\FBAutoSpaceFootnotestruer}
2241 \newcommand*{\FrenchFootnotes}{\FBFrenchFootnotestruer}
2242 \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.

```

2243 \FBclean@on@exit
2244 \ldf@finish\CurrentOption
2245 \let\loadlocalcfg\FB@llc
2246 </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`.

```

2247 <*acadian>
2248 \PackageInfo{acadian.ldf}%
2249   {'acadian' dialect is currently\MessageBreak
2250    *absolutely identical* to the\MessageBreak
2251    `french' language; reported}
2252 </acadian>
2253 <*canadien>
2254 \PackageWarning{canadien.ldf}%
2255   {Option `canadien' for Babel is *deprecated*,\MessageBreak
2256    it might be removed sooner or later. Please\MessageBreak
2257    use `acadian' instead; reported}%
2258 \def\CurrentOption{acadian}

2259 \def\datecanadien{\dateacadian}
2260 \def\captionscanadien{\captionacadian}
2261 \def\extrascanadien{\extrasacadian}
2262 \def\noextrascanadien{\noextrasacadian}
2263 </canadien>

```

```

2264 <*français>
2265 \PackageWarning{français.ldf}%
2266   {Option `français' for Babel is *deprecated*,\MessageBreak
2267     it might be removed sooner or later. Please\MessageBreak
2268     use `french' instead; reported}%
2269 \chardef\l@français\l@french
2270 \def\CurrentOption{french}
2271 </français>

```

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

```

2272 <*frenchb>
2273 \def\bbl@tempa{frenchb}
2274 \ifx\CurrentOption\bbl@tempa
2275   \chardef\l@frenchb\l@french
2276   \def\CurrentOption{french}
2277   \PackageWarning{babel-french}%
2278     {Option `frenchb' for Babel is *deprecated*,\MessageBreak
2279       it might be removed sooner or later. Please\MessageBreak
2280       use `french' instead; reported}
2281 \else
2282   \def\bbl@tempa{français}
2283   \ifx\CurrentOption\bbl@tempa
2284     \chardef\l@français\l@french
2285     \def\CurrentOption{french}

```

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

```

2286   \ifx\magnification\@undefined
2287     \PackageWarning{babel-french}%
2288       {Option `français' for Babel is *deprecated*,\MessageBreak
2289         it might be removed sooner or later. Please\MessageBreak
2290         use `french' instead; reported}
2291   \fi
2292 \else
2293   \def\bbl@tempa{canadien}
2294   \ifx\CurrentOption\bbl@tempa
2295     \def\CurrentOption{acadian}
2296     \PackageWarning{babel-french}%
2297       {Option `canadien' for Babel is *deprecated*,\MessageBreak
2298         it might be removed sooner or later. Please\MessageBreak
2299         use `acadian' instead; reported}
2300   \fi
2301 \fi
2302 \fi
2303 </frenchb>
2304 <acadian/canadien/frenchb/français>\input french.ldf\relax
2305 <acadian/canadien>\let\extrasacadian\extrasfrench
2306 <acadian/canadien>\let\noextrasacadian\noextrasfrench

```


3 Change History

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

v3.5h	frenchb.lua: Added glues and penalties should inherit attributes from the related punctuation character; this is mandatory for Lua-UL to underline and highlight them. Thanks to Marcel Krüger for providing the fix. 24	level changes. 38
	Code reorganised for better efficiency. 24	\noextrasfrench: \lccode of quote 0x27 changed from 0x2019 to 0x27 for Unicode engines. 16
v3.5g	frenchb.lua: The kerning callback is a bit specific: adding code with add_to_callback actually deletes the legacy kerning as pointed out by Marcel Krüger on SE. 24	v3.5b
v3.5f	General: \l@canadien was defined too early in file ‘canadien.ldf’: \l@acadian might not be defined. \selectlanguage{canadien} allowed again only for backward compatibility (deprecated). 75	General: Reset \FBeverypar@quote locally inside \@makefntext. Needed by \frquote. 73
	\Decima\MathComma: Fixed bug with the acadian language. Warning added if used with the icomma package. 44	\frquote: New command \FB@addquote@everypar to manage \everypar: \frquote failed when used immediately after a sectionning command. 37
v3.5e	\frenchsetup: StandardLayout and GlobalLayoutFrench options can no longer be toggled when French is not the main language. 54	v3.5a
	\frquote: Make resettings global on exit. 39	General: New optional layout for lists: lists’ items can be typeset as paragraphs with indented labels while the default leaves the labels hanging into the left margin. 67
	new command \NoEveryParQuote. 39	\descriptionFB: ListItemsAsPar option taken into account for description lists. 69
	reset \FB@addGUIspace attribute inside \localleftbox (LuaTeX). . 38	\frenchsetup: New option ListItemsAsPar for displaying lists’ items “as paragraphs”. 53
v3.5d	General: Add \frquote to redefinitions for bookmarks. 65	v3.4d
	\frenchsetup: ReduceListSpacing option depreciated: see StandardListSpacing. 53	\frenchsetup: New test for deciding about utf8 encoding for keys og and fg (the former one fails with LaTeX 2018 release). 58
v3.5c	General: Remove grouping inside \@makefntext, \localleftbox and \FBeverypar@quote saved and restored instead. 73	v3.4c
	\frquote: \FBeverypar@quote’s value now properly reset across	\ifFBXeTeX: Reverting to former test, beware of \XeTeXrevision left as \relax by careless testing. 16
		v3.4b
		\datefrench: Do not redefine \date as \frenchdate in French. 40
		v3.4a
		General: \LdfInit checks \FBclean@on@exit instead of \captionsfrench (undefined in PLain). Prevents loading french.ldf again with acadian option. 14
		babel-french now requires eTeX. ... 14
		Lua function token.get_meaning requires LuaTeX 1.0. 21
		New \FBgspchar to customise the space character to be used for \og

and\fg with the UnicodeNoBreakSpaces option.	36	can insert Unicode characters instead of glues.	22
New attribute \FB@dialect for the French dialect acadian.	20	\frenchsetup: New option 'UnicodeNoBreakSpaces' for html translators (LuaLaTeX only).	58
New command \FBsetspaces to fine tune spacing independently in French and in French dialects.	18	v3.3b General: Generate portmanteau files acadian.ldf, canadien.ldf, frenchb.ldf, and français.ldf and warn about deprecated options.	75
Shrink/stretch removed in \FBthousandsep.	46	New 'if' \ifFBfrench to replace \iflanguage test which is based on patterns.	16
Toks \FBcolonsp, \FBthinsp and \FBguillsp removed.	18	v3.3a General: Compatibility code for pre 2015/10/01 LaTeX release removed, see ltnews23.tex.	20
frenchb.lua: Global 'FBsp' table added; local function 'get_glue' changed into global 'FBget_glue'. . .	23	Skip \FBguillskip for LuaTeX replaced by toks \FBguillsp.	18
\datefrench: Specific code for Plain finally removed (babel bug reported).	40	\captionsfrench: Commands \frenchpartfirst, \frenchpartsecond and \frenchpartnameord added.	47
\extrasfrench: Change \(no)extras\CurrentOption to \(no)extrasfrench. \(no)extrasacadian will be defined as \((no)extrasfrench in file acadian.ldf.	16	\FBthinspace: Skips \FBcolonskip and \FBthinskip replaced by toks \FBcolonsp and \FBthinsp.	17
\frenchsetup: Patch for koma-script classes moved here, after \ifFBPartNameFull is defined, so that it applies to \extrasacadian too: \AtEndOfPackage is too late.	53	\frenchsetup: \frenchbsetup is now an alias for \frenchsetup.	53
v3.3d frenchb.lua: In default mode, for ':' only, check if next node is a glyph or not. If it is, turn the 'auto' flag to false (avoids spurious spaces in URLs, MSDOS paths or 10:35).	25	Options INGuillSpace, ThinColonSpace no longer delayed AtBeginDocument.	53
v3.3c General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, fontspec is no longer required.	65	\frquote: \FB@quotespace (kern), changed into \FB@guillspace.	38
New command \FBthousandsep to customise numprint.	46	v3.2h \@makefnfntextFB: With beamer.cls, add \llap to \@thefnmark for notes numbered over 99.	73
New configurable kerns \FBmedkern, and \FBthickkern suitable for HTML translation.	42	\bbl@frenchlistlayout: Execute \update@frenchlists only if GlobalLayoutFrench is false. Delete stuff for lists in \noextrasfrench. . .	70
Reorganise warnings when the caption, subcaption or floatrow packages are loaded before babel/french.	49	\frenchsetup: Option GlobalLayoutFrench skipped when French is not the main language. . . .	54
Reset \localleftbox locally inside \@makefnfntext. Needed by \frquote with LuaTeX.	73	v3.2g General: Add \boi to redefinitions for bookmarks.	65
frenchb.lua: Function 'get_glue' robustified. 'french_punctuation'		Changed Unicode definition of \boi. fontspec defines TU encoding now and no longer loads xunicode.sty. Test changed.	43 65

Issue a warning if beamerarticle.sty is loaded after babel.	52	\ttfamilyFB: New definitions of \ttfamilyFB and co, common to all engines, based on \FB@spacing@off and \FB@spacing@on.	35
\frenchsetup: Minimal list customisation when beamerarticle.sty is loaded.	54	v3.2b	
Warn when wrong values are provided to options EveryParGuill or EveryLineGuill.	57	General: Load ltuatex.tex for plain LuaTeX to ensure \newattribute is defined.	20
\frquote: Default options of \frquote are no longer engine-dependent.	37	Warning added when the subcaption package is loaded before babel/french.	49
v3.2f		frenchb.lua: glue_spec removed; starting with LuaTeX 0.95, glue specifications fit in glue.	24
\DecimalMathComma: Fixed conflict with the icomma package.	44	\ifFB@xetex@punct: New counter \FB@nonchar needed for non characters: it's value will be 4095 for new engines and 255 for older ones. ...	17
v3.2e		\NoAutoSpacing: \NoAutoSpacing made robust.	35
General: Add missing redefinitions for \leftmarginv, \leftmarginvi. Suggested by J.F. Burnol.	67	v3.2a	
\DecimalMathComma: \DecimalMathComma didn't work with LuaTeX. Fixed now.	44	\@makefnctextFB: beamer.cls requires a specific definition of \@makefnctextFB (pointed out by DB). The same is true for memoir and koma-script classes (done). ..	72
v3.2d		\fg: \xspace moved from \FB@fg to \fg: \xspace messes up \frquote, pointed out by Sonia Labetoulle. As a side effect \xspace is now active in \fg in and outside French.	37
\descriptionFB: Changed \listindentFB to \descindentFB which defaults to \listindentFB. \leftmargini reduced when \descindentFB is null.	69	v3.1m	
v3.2c		frenchb.lua: new_glue_scaled returns nil in case of invalid font table (i.e. lcircle1.pfb). In such cases babel-french leaves the node list unchanged.	24
General: New LuaTeX attribute \FB@spacing.	20	v3.1l	
Newif \ifFB@spacing and new commands \FB@spacingon, \FB@spacingoff to control space tuning in French.	20	General: Add a variant of \babel@savevariable to save \XeTeXcharclass(es) in a loop. .	30
Switch \ifFB@spacing added to the four French shorthands.	33	frenchb.lua: font.getfont(fid) possibly returns nil even for a positive fid (i.e. AMS lcircle1.pfb). Reported by François Legendre. ...	24
\FB@xetex@punct@french: Switch \ifFB@spacing added to all \XeTeXinterchartoks commands.	31	\FB@luatex@punct@french: Use \babel@save to save and restore \shorthandon and \shorthandoff.	29
\FBthinspace: Change .16667em to .5\fontdimen2\font to get in XeTeX and pdfTeX the same spacing as in LuaTeX.	17	\FB@xetex@punct@french: Save and restore	
\frenchsetup: Add a warning about options og/fg for old XeTeX or LuaTeX engines requiring active characters.	58		
\NoAutoSpacing: New definition based on \FB@spacing@off common to all engines.	35		

<code>\XeTeXinterchartokenstate,</code> <code>\shorthandon, \shorthandoff</code> using <code>\babel@savevariable</code> and <code>\babel@save,</code> <code>\XeTeXcharclass(es)</code> using <code>\FB@savevariable@loop</code>	for ‘«’ at the end of an <code>\hbox</code> or a paragraph or when followed by a null glue (i.e. springs).
31	27
v3.1k	
General: (pdfTeX shorthands) test on <code>\lastskip</code> changed from 0pt to 1sp for active punctuation for consistency with XeTeX and LuaTeX.	flag <code>addgl</code> set to false for ‘»’ at the beginning of an <code>\hbox</code> or a paragraph or a tabular ‘l’ and ‘c’ columns.
33	27
<code>\FB@xetex@punct@french</code> : Thin glues (less than 1sp) should not trigger space insertion before high punctuation. Add a check on <code>\lastkip</code>	Node HLIST added; node TEMP added for the first node of <code>\hboxes</code>
31	22
v3.1j	
General: Loading <code>luatexbase.sty</code> is no longer needed with LaTeX release 2015/10/01 or later.	<code>\captionsfrench</code> : <code>\partname</code> ’s definition depends now on flag <code>PartNameFull</code> . No need to redefine it in <code>\frenchbsetup</code>
20	47
<code>\frquote</code> : <code>\fr@quote</code> completely rewritten: <code>\leavevmode</code> added and explicitly save/retore <code>\everypar</code> and <code>\localleftbox</code> instead of using a group in order to ensure compatibility with package <code>wrapfig</code> .	<code>\frenchsetup</code> : Bug fix for koma-scripts classes: a spurious dot was added by the <code>\partformat</code> command.
38	53
<code>\PackageWarning</code> is undefined in Plain, use <code>\fb@warning</code> instead. .	PartNameFull now just sets the flag, nothing to add to <code>\captionsfrench</code> when false. . .
38	53
v3.1i	
General: <code>\nombre</code> command changed when <code>numprint.sty</code> is not loaded: only one warning, no error.	v3.1f
46	
Remove restriction about loading <code>numprint.sty</code> after <code>babel</code>	General: <code>\FBCaption@Separator</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.
52	50
<code>\frquote</code> : <code>\luatexlocalleftbox</code> changed to <code>\localleftbox</code> by new LaTeX release 2015/10/01. . . .	<code>\FBprocess@options</code> : Bug fix for the beamer class: figure and table captions are now consistent with babel-french’s documentation. Pointed out by Denis Bitouzé. . . .
38	63
v3.1h	
General: <code>french.cfg</code> from e-french conflicts with babel-french. Do NOT load it (no need for .cfg files with babel-french anyway).	Definition of <code>\captionformat</code> and <code>\captiondelim</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.
75	63
v3.1g	
General: Lua function <code>french_punctuation</code> is now inserted at the end of the ‘kerning’ callback (no priority) instead of ‘ <code>hpack_filter</code> ’ and ‘ <code>pre_linebreak_filter</code> ’.	<code>\FBthinspace</code> : <code>\FBthinspace</code> is no longer a kern but a skip (babel-french adds a nobreak penalty before it).
29	17
Use Babel defined loops <code>\bb@for</code> instead of <code>\@for</code> borrowed from file <code>ltntrl.dtx</code> (<code>\@for</code> is undefined in Plain).	v3.1e
30	
<code>frenchb.lua</code> : Flag <code>addgl</code> set to false	<code>\frenchsetup</code> : Corrected typo: <code>SmallCapsFigTabCaptions</code> instead of <code>SmallCapsFigTabCaptions</code> . Pointed out by Céline Chevalier.
	53
	v3.1d
	General: New section: issue warnings if packages <code>listings</code> , <code>numprint</code> and <code>natbib</code> are loaded too early or too late vs <code>babel</code>
	52
	v3.1c
	<code>frenchb.lua</code> : Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André.
	25

v3.1b		
frenchb.lua: Add a check for null fid in french_punctuation (Tikz \nullfont). Bug pointed out by Paul Gaborit.	24	
\captionsfrench: Change \scshape to customisable \FBfigtabshape for \figurename and \tablename. .	47	
\fprimo): Removed \lowercase from definitions of \FrenchEnumerate, ... \No and co: \up already does the conversion. .	42	
\frenchsetup: New option SmallCapsFigTabCaptions.	53	
\ieres: Removed \lowercase from definitions of \ieme and co: \up already does the conversion.	42	
v3.1a		
General: fontspec is not required for T1 fonts used with the luainputenc.sty package.	65	
Misplaced \fi for plain formats. .	20	
New command \frquote for imbedded or long French quotations.	37	
frenchb.lua: Added flag addgl which must also be true when prev or next is not a char (i.e. \kern0 in «\texttt{a}»).	27	
Codes 0x13 and 0x14 added for French quotes in T1-encoding.	22	
Look ahead when next is a kern (i.e. in « \texttt{a} »).	27	
\frenchsetup: Codes 0x13 and 0x14 added for French quotes in T1-encoding. Support for older versions of LuaTeX and XeTeX dropped.	58	
New options InnerGuillSingle, EveryParGuill and EveryLineGuill to control \frquote.	53	
v3.0c		
General: babel-french requires babel-3.9i.	14	
Just load luatexbase.sty instead of luaotfload.sty with plain formats. .	20	
No need to define \l@french as \lang@french, babel.def (3.9j) takes care for this.	15	
frenchb.lua: Null glues should not trigger space insertion before high punctuation. Bug pointed out by		
		Benoit Rivet for the 'lstlisting' environment of the listings package. 25
		\frenchsetup: New option INGuillSpace. 53
		No list customisation when beamer class is loaded. 54
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. .	37	
\LdfInit checks \captionsfrench instead of \datefrench to avoid a conflict with papertex.cls which loads datetime.sty.	14	
french.cfg will be loaded (if found) instead of frenchb.cfg. NO NEED for .cfg files in French anyway.	75	
In Plain, provide a substitute for \PackageWarning and \PackageInfo.	14	
Merging of \captionsfrenchb, \captionsfrançais with \captionsfrench deleted in favor of new babel 3.9 syntax.	48	
More informative, less TeXnical warning about \@makecaption. .	50	
New flag \ifFB@luatex@punct for 'high punctuation' management with LuaTeX engines.	17	
New handling of 'high punctuation' through callbacks with LuaTeX engines.	20	
No warning about \@makecaption for SMF classes.	49	
Options processing completely reorganised, now \babel@save and \babel@savevariable are usable for French.	52	
Support for options frenchb, français, canadien, acadian changed.	14	
Test \ifXeTeX changed to \ifBUnicode and \xltextra' changed to 'fontspec'.	65	

<code>\CaptionSeparator</code> : Remove <code>\FBCaption@SeparatorORI</code> , use <code>\babel@save</code> instead.	48	babel's <code>\babel@savevariable</code> to handle apostrophe's <code>\lccode</code>	16
<code>\captionsfrench</code> : Take advantage of babel's <code>\SetString</code> commands for captionnames.	47	<code>\FB@fg</code> : Definitions of <code>\FB@og</code> and <code>\FB@fg</code> now depend on punctuation handling (LuaTeX / XeTeX / active).	36
<code>\datefrench</code> : Take advantage of babel's <code>\SetString</code> commands for <code>\datefrench</code> . Doesn't work with Plain (yet?).	40	<code>\FBprocess@options</code> : With koma-script and memoir class, customise <code>\captionformat</code> and <code>\captiondelim</code>	63
<code>\descriptionFB</code> : Added <code>\listindentFB</code> to <code>\itemindent</code> . Suggested by Denis Bitouzé.	69	<code>\frenchsetup</code> : New options OldFigTabCaptions and CustomiseFigTabCaptions.	53
<code>\extrasfrench</code> : Take advantage of			