A Babel language definition file for French
frenchb.dtx v3.4d, 2018/04/12

Daniel Flipo
daniel.flipo@free.fr

Contents

1 The French language

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

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

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

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

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

Changes between version 3.0 and v3.4d are listed in subsection 1.4 p. 11.

An extensive documentation is available in French here:
http://daniel.flipo.free.fr/frenchb

1.1 Basic interface

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

The French language can be loaded with babel by a command like:
`\usepackage[german,spanish,french,british]{babel}` [2]

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

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

1. the first paragraph of each section is indented (LaTeX only);

2. the default items in itemize environment are set to '—' instead of '•', and all vertical spacing and glue is deleted; it is possible to change '—' to something else ('–' for instance) using `\frenchsetup{}` (see section 1.2 p. 5);

3. vertical spacing in general LaTeX lists is shortened;

4. footnotes are displayed "à la française".

---

[1]The file described in this section has version number v3.4d and was last revised on 2018/04/12.

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

[3] For 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 .

Regarding local typography, the command \selectlanguage{french} switches to the French language[4], 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 .

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=» .

   \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 .

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

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

---

[4] \selectlanguage{francais} and \selectlanguage{frenchb} are no longer supported.
[5] Well, the automatic insertion may add unwanted spaces in some cases, for correction see AutoSpacePunctuation option and \NoAutoSpacing command .

- 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 1°, 2°, 3°, 4°. `\FrenchEnumerate{6}` prints 6°.

6. Abbreviations for "Numéro(s)" and "numéro(s)" (N^o N^{os} n^o and n^{os}) are obtained via the commands `\No, \Nos, \no, \nos`.

7. Two commands are provided to typeset the symbol for "degré": `\degre` prints the raw character and `\degres` should be used to typeset temperatures (e.g., "20~`\degres` C" with a non-breaking space), or for alcohols" strengths (e.g., "45`\degres`" with *no* space in French).

8. In math mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the 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, 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 remainder of the list of offered options. As usual with keyval syntax, boolean options (as ShowOptions) can be entered as ShowOptions=true or just ShowOptions, the =true part can be omitted.

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

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

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

GlobalLayoutFrench=false (true∗) should no longer be used; it was intended to emulate, when French is the main language, what prior versions of babel-french (pre-2.2) did: lists, and first paragraphs of sections would be displayed the standard way in other languages than French, and "à la française" in French. Note that the layout of footnotes is language independent anyway (see below FrenchFootnotes and AutoSpaceFootnotes).

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

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

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

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

`StandardEnumerateEnv=true (false∗)` ; starting with version 2.6 babel-french
redefines the enumerate and `description` environments to make left mar-
gins match those of the French version of `itemize` lists. Setting this option
to `true` reverts to the standard definition of enumerate and `description`.

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

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

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

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

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

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

`StandardLists=true (false∗)` forbids babel-french to customise any kind of
list. Try the option `StandardLists` in case of conflicts with classes or
packages that customise lists too. This option is just a shorthand setting
all four options `ReduceListSpacing=false`, `StandardItemizeEnv=true`,
`StandardEnumerateEnv=true` and `StandardItemLabels=true`.

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

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

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

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

AutoSpacePunctuation=false (true) ; in French, the user *should* input a space before the four characters ':;!?' but as many people forget about it (even among native French writers!), the default behaviour of babel-french is to automatically typeset non-breaking spaces the width of which is either \FBthinspace (defauts to a thin space) before ';' '!' '?' or \FBcolonspace (defauts to \space) before ':'; the defaults follow the French 'Imprimerie Nationale's recommendations. This is convenient in most cases but can lead to addition of spurious spaces in URLs, in MS-DOS paths or in timetables (10:55) —this no longer occurs with LuaTeX—, except if they are typed in \texttt or verbatim mode. When the current font is a monospaced (typewriter) font, no spurious space is added in that case [6], so the default behaviour of of babel-french in that area should be fine in most circumstances.

Choosing `AutoSpacePunctuation=false` will ensure that a proper space is added before ':;!?' *if and only if* a (normal) space has been typed in. This option gives full control on space insertion before ':;!?'. Those who are unsure about their typing in this area should stick to the default option and use the provided \NoAutoSpacing command inside a group in case an unwanted space is added by babel-french (i.e. {\NoAutoSpacing http://mysite} [7] or {\NoAutoSpacing ???} (needed for pdfTeX only).

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

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

UnicodeNoBreakSpaces=true (false) ; (experimental) this option should be set to `true` *only while converting LuaLaTeX files* to HTML. It ensures that non-breaking spaces added by babel-french are inserted in the PDF file as U+A0

---

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

[7]Actually, 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...

or U+202F (thin) instead of penalties and glues. Note that `lwarp` (v. 0.37 and up) is fully compatible with babel-french for translating PDFLaTeX or XeLaTeX files to HTML.

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

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

`EveryParGuill=open, close, none (open)` ; sets whether an opening quote («) or a closing one (») or nothing should be printed by `\frquote{}` at the beginning of every parapraph 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 parapraph included in the inner quotation.

`ThinSpaceInFrenchNumbers=true (false)` ; if `numprint` has been loaded with the `autolanguage` option, while typesetting numbers with the `\numprint{}` command, `\npthousandsep` is defined as a non-breaking space (~) [8] in French; when set to true, this option redefines `\npthousandsep` as a thin space (`\,`).

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

---

[8]Actually without stretch nor shrink.

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

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

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

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

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

Options' order – Please remember that options are read in the order they appear in the `\frenchsetup{}` command. Someone wishing that babel-french leaves the layout of lists and footnotes untouched but caring for indentation of first paragraph of sections should choose
`\frenchsetup{`StandardLayout,IndentFirst`}` to get the expected layout. The reverse order `\frenchsetup{`IndentFirst,StandardLayout`}` would lead to option IndentFirst being overwritten by StandardLayout.

1.2.2 Caption names

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

1.2.3 Figure and table captions

In French, captions in figures and tables should never be printed as 'Figure 1: ' which is the default in standard LaTeX2e classes (a space should *always* preceed a colon in French), anyway 'Figure 1 – ' is preferred.
When French is the main language, the default behaviour of babel-french is to change the separator (colon) used in figures' and tables' captions *for all languages* to \CaptionSeparator which defaults to ' – ' and can be redefined in the preamble with \renewcommand*{\CaptionSeparator}{...}. This works for the standard LaTeX2e classes, for the memoir and koma-script classes. In case this procedure fails a warning is issued.

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

Three options are provided to customise figure and table captions:

- if `CustomiseFigTabCaptions` is set to `false` the colon will be used as separator in all languages, with a proper space before the colon in French (if possible);

- the second option, `OldFigTabCaptions`, can be set to `true` to print figures' and tables' captions as they were with versions pre 3.0 of babel-french (using `\CaptionSeparator` in French and colon in other languages); this option only makes sense with the standard LaTeX classes `article`, `report` and `book`;

- the last option, `SmallCapsFigTabCaptions`, can be set to `false` to typeset `\figurename` and `\tablename` in French as "Figure" and "Table" rather than in small caps (the default).

1.3 Hyphenation checks

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

- run pdfLaTeX on the following file, with the encoding suitable for your machine (*my-encoding* will be `latin1` for Unix machines, `ansinew` for PCs running Windows, `applemac` or `latin1` for Macintoshs, or `utf8`...

  ```
  %%% Test file for French hyphenation.
  \documentclass[french]{article}
  \usepackage[my-encoding]{inputenc}
  \usepackage[T1]{fontenc} % Use LM fonts
  \usepackage{lmodern}     % for French
  \usepackage{babel}
  \begin{document}
  \showhyphens{signal container \'ev\'enement alg\`ebre}
  \showhyphens{signal container événement algèbre}
  \end{document}
  ```

- check the hyphenations proposed by 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 3.4?

Version 3.4a adds a new command \frenchdate (see p.) 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. .
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 convertion into HTML of non-breaking spaces added by babel-french. Usage of lwarp (v.0.37 and up) is recommended for HTML output, it works fine on files compiled with XeLaTeX or pdfLaTeX formats. A new experimental option UnicodeNoBreakSpaces has been added for LuaLaTeX in version 3.3c, see p. .
According to current babel's standards, every dialect should have it's 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.
Tne 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. 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 .

- \frenchsetup{} options management has been completely reworked; two new options added.

- Canadian French didn't work as a normal babel's dialect, it should now; btw. the French language should now be loaded as french, *not as* frenchb or francais and preferably as a *global* option of \documentclass. Some tolerance still exists in v3.0, but do not rely on it.

- babel-french no longer loads frenchb.cfg: customisation should definitely be done using \frenchsetup{} options.

- Description lists labels are now indented; try setting `\descindentFB=0pt` (or `\listindentFB=0pt` for all lists) in the preamble if you don't like it.

- The last but not least change affects the (recent) LuaTeX-based engines, (this means version 0.76 as included in TL2013 and up): active characters are no longer used in French for 'high punctuation' [9]. Functionalities and user interface are unchanged.

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

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

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

---

[9] The 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           {Orignal 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@<languagename> also for eTeX, LuaTeX and XeTeX formats which set \lang@<languagename>.

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

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

```
57 \ifdefined\l@acadian \else \adddialect\l@acadian\l@french \fi
```

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

```
58 \providehyphenmins{french}{\tw@\thr@@}
59 \providehyphenmins{acadian}{\tw@\thr@@}
```

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

```
60 \newif\ifLaTeXe
61 \let\bbl@tempa\relax
62 \ifdefined\magnification
63 \else
64    \ifdefined\@compatibilitytrue
65       \LaTeXetrue
66    \else
67       \PackageError{french.ldf}
68          {LaTeX-2.09 format is no longer supported.\MessageBreak
69           Aborting here}
70          {Please upgrade to LaTeX2e!}
71       \let\bbl@tempa\endinput
72    \fi
73 \fi
74 \bbl@tempa
```

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

```
75 \newif\ifFBunicode
76 \newif\ifFBLuaTeX
77 \newif\ifFBXeTeX
78 \begingroup\expandafter\expandafter\expandafter\endgroup
79 \expandafter\ifx\csname luatexversion\endcsname\relax
80 \else
81   \FBunicodetrue \FBLuaTeXtrue
82 \fi
83 \begingroup\expandafter\expandafter\expandafter\endgroup
84 \expandafter\ifx\csname XeTeXrevision\endcsname\relax
85 \else
86   \FBunicodetrue \FBXeTeXtrue
87 \fi
```

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

```
88 \newif\ifFBfrench
```

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

In French, character "apostrophe" is a letter in expressions like l'ambulance (French
hyphenation patterns provide entries for this kind of words). This means that the
\lccode of "apostrophe" has to be non null in French for proper hyphenation of those
expressions, and has to be reset to null when exiting French.

The following code ensures correct hyphenation of words like d'aventure, l'utopie,
with all TeX engines (XeTeX, LuaTeX, pdfTeX) using hyph-fr.tex patterns.

```
89 \def\extrasfrench{%
90     \FBfrenchtrue
91     \babel@savevariable{\lccode`\'}%
92     \ifFBunicode
93         \babel@savevariable{\lccode"2019}%
94         \lccode`\'="2019\lccode"2019="2019
95     \else
96         \lccode`\'=`\'
97     \fi
98 }
99 \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.

```
100 \addto\extrasfrench{\bbl@frenchspacing}
101 \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.

```
102 \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).

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

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

\FBguillspace These three commands are meant for basic French. Other French dialects can use
\FBcolonspace different settings, see below. According to the I.N. specifications, the ':' requires
\FBthinspace

17

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.

```
131 \newcommand*{\FBguillspace}{\hskip .8\fontdimen2\font
132                              plus .3\fontdimen3\font
133                              minus .8\fontdimen4\font \relax}
134 \newcommand*{\FBcolonspace}{\space}
135 \newcommand*{\FBthinspace}{\hskip .5\fontdimen2\font \relax}
```

\FBsetspaces This command makes it easy to fine tune \FBguillspace, \FBcolonspace and \FBthinspace in French (defaut) 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.

```
136 \ifLaTeXe
137   \newcommand*{\FBsetspaces}[5][french]{%
138     \def\bbl@tempa{french}\def\bbl@tempb{#1}%
139     \ifx\bbl@tempa\bbl@tempb \def\bbl@tempb{}\fi
140     \@namedef{\bbl@tempb FB#2space}{\hskip #3\fontdimen2\font
141                                      plus #4\fontdimen3\font
142                                      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".

```
143     \ifFB@luatex@punct
144       \ifx\bbl@tempb\FB@acadian
145         \directlua{
146           FBsp.#2.gl.ac[1] = #3
147           FBsp.#2.gl.ac[2] = #4
148           FBsp.#2.gl.ac[3] = #5
149           if #3 > 0.6 then
150              FBsp.#2.ch.ac = 0xA0
151           elseif #3 > 0.2 then
152              FBsp.#2.ch.ac = 0x202F
153           else
154              FBsp.#2.ch.ac = 0x200B
155           end
156         }%
```

```
157        \fi
158      \fi
159    }
160    \@onlypreamble\FBsetspaces
161 \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 \languagename 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.

```
162 \ifLaTeXe
163   \addto\extrasfrench{%
164     \ifFB@luatex@punct
165       \edef\bbl@tempa{\detokenize\expandafter{\languagename}}%
166       \edef\bbl@tempb{\detokenize{french}}%
167       \ifx\bbl@tempa\bbl@tempb \FB@dialect=0 \relax
168       \else                    \FB@dialect=1 \relax
169       \fi
```

The first time whe enter French, we have to set the LuaTeX tables for French (\FB@dialet=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!

```
170       \ifdefined\FB@once\else
171         \set@glue@table{colon}%
172         \set@glue@table{thin}%
173         \set@glue@table{guill}%
174         \def\FB@once{}%
175       \fi
176     \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).

```
177     \ifcsname\languagename FBthinspace\endcsname
178       \babel@save\FBthinspace
179       \renewcommand*{\FBthinspace}{%
180             \csname\languagename FBthinspace\endcsname}%
181     \fi
```

Same for \FBcolonspace:

```
182     \ifcsname\languagename FBcolonspace\endcsname
183       \babel@save\FBcolonspace
184       \renewcommand*{\FBcolonspace}{%
185             \csname\languagename FBcolonspace\endcsname}%
186     \fi
```

And for \FBguillspace:

```
187    \ifcsname\languagename FBguillspace\endcsname
188      \babel@save\FBguillspace
189      \renewcommand*{\FBguillspace}{%
190              \csname\languagename FBguillspace\endcsname}%
191    \fi
192  }
193 \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.

```
194 \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).

```
195 \newcommand*{\FB@spacing@on}{%
196   \ifFB@luatex@punct
197     \FB@spacing=1 \relax
198   \else
199     \FB@spacingtrue
200   \fi}
201 \newcommand*{\FB@spacing@off}{%
202   \ifFB@luatex@punct
203     \FB@spacing=0 \relax
204   \else
205     \FB@spacingfalse
206   \fi}
```

2.2.1 Punctuation with LuaTeX

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

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

This code is for Plain: load ltluatex.tex if it hasn't been loaded before babel.

```
209     \input ltluatex.tex
210   \fi
```

We define five LuaTeX attributes to control spacing in French and/or Acadian for 'high punctuation' and quotes, making sure that \newattribute is defined.
\FB@spacing=0 switches off any space tuning both before high punctuation characters and inside French quotes (i.e. function french_punctuation doesn't alter the node list at all).
\FB@addDPspace=0 switches off automatic insertion of spaces before high punctuation characters (but typed spaces are still turned into non-breaking thin- or word-spaces).

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

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

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

```
211  \newattribute\FB@spacing      \FB@spacing=1 \relax
212  \newattribute\FB@addDPspace    \FB@addDPspace=1 \relax
213  \newattribute\FB@addGUILspace \FB@addGUILspace=0 \relax
214  \newattribute\FB@ucsNBSP       \FB@ucsNBSP=0 \relax
215  \newattribute\FB@dialect       \FB@dialect=0 \relax
216  \ifLaTeXe
217    \PackageInfo{french.ldf}{No need for active punctuation
218                 characters\MessageBreak with this version
219                 of LuaTeX!\MessageBreak reported}
220  \else
221    \fb@info{No need for active punctuation characters\\
222             with this version of LuaTeX!}
223  \fi
```

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

```
224  \newcommand*{\set@glue@table}[1]{%
225    \directlua {
226      local s = token.get_meaning("FB#1space")
227      local t = FBget_glue(s)
228      if t then
229         FBsp.#1.gl.fr = t
230         if not FBsp.#1.gl.ac[1] then
231            FBsp.#1.gl.ac =  t
232         end
233         if FBsp.#1.gl.fr[1] > 0.6 then
234            FBsp.#1.ch.fr = 0xA0
235         elseif FBsp.#1.gl.fr[1] > 0.2 then
236            FBsp.#1.ch.fr = 0x202F
237         else
238            FBsp.#1.ch.fr = 0x200B
239         end
240         if not FBsp.#1.ch.ac then
241            FBsp.#1.ch.ac = FBsp.#1.ch.fr
242         end
243      else
244         texio.write_nl('term and log', '')
```

```
245            texio.write_nl('term and log',
246              '*** french.ldf warning: Unexpected syntax in FB#1space,')
247            texio.write_nl('term and log',
248              '*** french.ldf warning: LuaTeX table FBsp unchanged.')
249            texio.write_nl('term and log',
250              '*** french.ldf warning: Consider using FBsetspaces to ')
251            texio.write('term and log', 'customise FB#1space.')
252            texio.write_nl('term and log', '')
253         end
254      }%
255   }
256 \fi
257 </french>
```

frenchb.lua This is frenchb.lua. It holds Lua code to deal with 'high punctuation' and quotes. This code is based on suggestions from Paul Isambert.

First we define two flags to control spacing before French 'high punctuation' (thin space or inter-word space).

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

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

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

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

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

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

22

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

Local definitions for nodes:

```
285 local new_node        = node.new
286 local copy_node       = node.copy
287 local node_id         = node.id
288 local HLIST           = node_id("hlist")
289 local TEMP            = node_id("temp")
290 local KERN            = node_id("kern")
291 local GLUE            = node_id("glue")
292 local GLYPH           = node_id("glyph")
293 local PENALTY         = node_id("penalty")
294 local nobreak         = new_node(PENALTY)
295 nobreak.penalty       = 10000
296 local insert_node_before = node.insert_before
297 local insert_node_after  = node.insert_after
298 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.

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

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

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

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

```
363 local FBspacing    = luatexbase.attributes['FB@spacing']
364 local addDPspace   = luatexbase.attributes['FB@addDPspace']
```

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

```
369 local function french_punctuation (head)
370   for item in node.traverse_id(GLYPH, head) do
371     local lang = item.lang
372     local char = item.char
373     local fid  = item.font
374     local FRspacing = has_attribute(item, FBspacing)
375     FRspacing = FRspacing and FRspacing > 0
376     local FRucsNBSP = has_attribute(item, FBucsNBSP)
377     FRucsNBSP = FRucsNBSP and FRucsNBSP > 0
378     local FRdialect = has_attribute(item, FBdialect)
379     FRdialect = FRdialect and FRdialect > 0
380     local SIG  = has_attribute(item, addGUILspace)
381     SIG = SIG and SIG >0
382     if lang ~= FR_fr and lang ~= FR_ca then
383       FRspacing = nil
384     end
385     local nbspace  = new_node("glyph")
386     if FRspacing and FB_punct_left[char] and fid > 0 then
387       local prev = item.prev
388       local prev_id, prev_subtype, prev_char
389       if prev then
390         prev_id = prev.id
391         prev_subtype = prev.subtype
392         if prev_id == GLYPH then
393           prev_char = prev.char
394         end
395       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.

```
396       local is_glue = prev_id == GLUE
397       local glue_wd
398       if is_glue then
399         glue_wd = prev.width
400       end
401       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

25

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.

```
402        if FB_punct_thin[char] or FB_punct_thick[char] then
403            local SBDP = has_attribute(item, addDPspace)
404            local auto = SBDP and SBDP > 0
405            if FB_punct_thick[char] and auto then
406                local next = item.next
407                local next_id
408                if next then
409                    next_id = next.id
410                end
411                if next_id and next_id == GLYPH then
412                    auto = false
413                end
414            end
415            if auto then
416                if (prev_char and FB_punct_null[prev_char]) or
417                    (is_glue and glue_wd <= 1) or
418                    (prev_id == HLIST and prev_subtype == 3) or
419                    (prev_id == TEMP) then
420                    auto = false
421                end
422            end
423            local fbglue
424            local t
425            if FB_punct_thick[char] then
426                if FRdialect then
427                    t = FBsp.colon.gl.ac
428                    nbspace.char = FBsp.colon.ch.ac
429                else
430                    t = FBsp.colon.gl.fr
431                    nbspace.char = FBsp.colon.ch.fr
432                end
433            else
434                if FRdialect then
435                    t = FBsp.thin.gl.ac
436                    nbspace.char = FBsp.thin.ch.ac
437                else
438                    t = FBsp.thin.gl.fr
439                    nbspace.char = FBsp.thin.ch.fr
440                end
```

```
441            end
442            fbglue = new_glue_scaled(fid, t)
```

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

```
443            if (realglue or auto) and fbglue then
444                if realglue then
445                    head = remove_node(head,prev,true)
446                end
447                if (FRucsNBSP) then
448                    nbspace.font  = fid
449                    insert_node_before(head, item, copy_node(nbspace))
450                else
451                    insert_node_before(head, item, copy_node(nobreak))
452                    insert_node_before(head, item, copy_node(fbglue))
453                end
454            end
```

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

```
455        elseif SIG then
456            local addgl = (prev_char and not FB_guil_null[prev_char]) or
457                          (not prev_char and
458                           prev_id ~= TEMP and
459                           not (prev_id == HLIST and prev_subtype == 3)
460                          )
```

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

```
461            if is_glue and glue_wd <= 1 then
462                addgl = false
463            end
464            local t = FBsp.guill.gl.fr
465            nbspace.char = FBsp.guill.ch.fr
466            if FRdialect then
467                t = FBsp.guill.gl.ac
468                nbspace.char = FBsp.guill.ch.ac
469            end
470            local fbglue = new_glue_scaled(fid, t)
471            if addgl and fbglue then
472                if is_glue then
473                    head = remove_node(head,prev,true)
474                end
475                if (FRucsNBSP) then
476                    nbspace.font = fid
477                    insert_node_before(head, item, copy_node(nbspace))
478                else
479                    insert_node_before(head, item, copy_node(nobreak))
```

```
480                    insert_node_before(head, item, copy_node(fbglue))
481              end
482          end
483      end
484  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*.

```
485      if FRspacing and FB_punct_right[char]
486              and fid > 0 and SIG then
487          local next = item.next
488          local next_id, next_subtype, next_char, nextnext, kern_wd
489          if next then
490              next_id = next.id
491              next_subtype = next.subtype
492              if next_id == GLYPH then
493                  next_char = next.char
```

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

```
494              elseif next_id == KERN then
495                  kern_wd = next.kern
496                  if kern_wd == 0 then
497                      nextnext = next.next
498                      if nextnext then
499                          next = nextnext
500                          next_id = nextnext.id
501                          next_subtype = nextnext.subtype
502                          if next_id == GLYPH then
503                              next_char = nextnext.char
504                          end
505                      end
506                  end
507              end
508          end
509          local is_glue = next_id == GLUE
510          if is_glue then
511              glue_wd = next.width
512          end
513          local addgl = (next_char and not FB_guil_null[next_char]) or
514                        (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).

```
515          if is_glue and glue_wd == 0 then
516              addgl = false
517          end
518          local fid = item.font
```

```
519        local t = FBsp.guill.gl.fr
520        nbspace.char = FBsp.guill.ch.fr
521        if FRdialect then
522            t = FBsp.guill.gl.ac
523            nbspace.char = FBsp.guill.ch.ac
524        end
525        local fbglue = new_glue_scaled(fid, t)
526        if addgl and fbglue then
527            if is_glue then
528                head = remove_node(head,next,true)
529            end
530            if (FRucsNBSP) then
531                nbspace.font = fid
532                insert_node_after(head, item, copy_node(nbspace))
533            else
534                insert_node_after(head, item, copy_node(fbglue))
535                insert_node_after(head, item, copy_node(nobreak))
536            end
537        end
538    end
539  end
540  return head
541 end
542 return french_punctuation
543 </lua>
```

\FB@luatex@punct@french As a language tag is part of glyph nodes in LuaTeX, no more switching has to be done in \extrasfrench, setting the dialect attribute has already be 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.

```
544 <*french>
545 \ifFB@luatex@punct
546   \newcommand*{\FB@luatex@punct@french}{%
547      \babel@save\shorthandon
548      \babel@save\shorthandoff
549      \def\shorthandoff##1{%
550        \ifx\PackageWarning\@undefined
551          \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
552            LuaTeX,\\ use \noexpand\NoAutoSpacing
553            *inside a group* instead.}%
554        \else
555          \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?} is
556            helpless with LuaTeX,\MessageBreak use \protect\NoAutoSpacing
557            \space *inside a group* instead;\MessageBreak reported}%
558        \fi}%
559      \def\shorthandon##1{}%
560   }
561   \addto\extrasfrench{\FB@luatex@punct@french}
```

The next definition will be used to activate Lua punctuation: it loads `frenchb.lua` and adds function `french_punctuation` at the end of the `kerning` callback (no priority).

```
562  \def\activate@luatexpunct{%
563    \directlua{%
564      FR_fr = \the\l@french ; FR_ca = \the\l@acadian ;
565      local path = kpse.find_file("frenchb.lua", "lua")
566      if path then
567        local f = dofile(path)
568        luatexbase.add_to_callback("kerning",
569                    f, "frenchb.french_punctuation")
570      else
571        texio.write_nl('')
572        texio.write_nl('****************************')
573        texio.write_nl('Error: frenchb.lua not found.')
574        texio.write_nl('****************************')
575        texio.write_nl('')
576      end
577    }%
578  }
579 \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 inital values when leaving French.

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

```
580 \ifFB@xetex@punct
581   \ifLaTeXe
582   \PackageInfo{french.ldf}{No need for active punctuation characters%
583                \MessageBreak with this version of XeTeX!%
584                \MessageBreak reported}
585   \else
586   \fb@info{No need for active punctuation characters\\
587           with this version of XeTeX!}
588   \fi
```

Six new character classes are defined for babel-french.

```
589    \newXeTeXintercharclass\FB@punctthick
590    \newXeTeXintercharclass\FB@punctthin
591    \newXeTeXintercharclass\FB@punctnul
592    \newXeTeXintercharclass\FB@guilo
593    \newXeTeXintercharclass\FB@guilf
594    \newXeTeXintercharclass\FB@guilnul
```

As \babel@savevariable doesn't work inside a \bbl@for loop, we define a variant
to save the \XeTeXcharclass values which will be modified in French.

```
595    \def\FBsavevariable@loop#1#2{\begingroup
596      \toks@\expandafter{\originalTeX #1}%
597      \edef\x{\endgroup
598        \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
599      \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
)	]	{	}	,	-	.	"	%	'	`	'

```
600    \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F,%
601                     "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.

```
602    \newcommand*{\FB@xetex@punct@french}{%
603      \babel@savevariable{\XeTeXinterchartokenstate}%
604      \babel@save{\shorthandon}%
605      \babel@save{\shorthandoff}%
606      \bbl@for\FB@char\FB@charlist
607           {\FBsavevariable@loop{\XeTeXcharclass}{\FB@char}}%
608      \def\shorthandoff##1{%
609        \ifx\PackageWarning\@undefined
610          \fb@warning{\noexpand\shorthandoff{;:!?} is helpless with
611             XeTeX,\\ use \noexpand\NoAutoSpacing
612             *inside a group* instead.}%
613        \else
614          \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?} is
615             helpless with XeTeX,\MessageBreak use \protect\NoAutoSpacing
616             \space *inside a group* instead;\MessageBreak reported}%
617        \fi}%
618      \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).

```
619        \XeTeXinterchartokenstate=1
620        \XeTeXcharclass '\: = \FB@punctthick
621        \XeTeXinterchartoks \z@ \FB@punctthick = {%
622            \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
623        \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
624            \ifFB@spacing\FDP@colonspace\fi}%
```

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

```
625        \XeTeXinterchartoks \FB@nonchar \FB@punctthick = {%
626            \ifFB@spacing
627              \ifhmode
628                \ifdim\lastskip>1sp
629                  \unskip\penalty\@M\FBcolonspace
630                \else
631                  \FDP@colonspace
632                \fi
633              \fi
634            \fi}%
635        \bbl@for\FB@char
636            {'\;,'\!,'\?}%
637            {\XeTeXcharclass\FB@char=\FB@punctthin}%
638        \XeTeXinterchartoks \z@ \FB@punctthin = {%
639            \ifFB@spacing\ifhmode\FDP@thinspace\fi\fi}%
640        \XeTeXinterchartoks \FB@guilf \FB@punctthin = {%
641            \ifFB@spacing\FDP@thinspace\fi}%
642        \XeTeXinterchartoks \FB@nonchar \FB@punctthin = {%
643            \ifFB@spacing
644              \ifhmode
645                \ifdim\lastskip>1sp
646                  \unskip\penalty\@M\FBthinspace
647                \else
648                  \FDP@thinspace
649                \fi
650              \fi
651            \fi}%
652        \XeTeXinterchartoks \FB@guilo \z@ = {%
653            \ifFB@spacing\FB@guillspace\fi}%
654        \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
655            \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
656        \XeTeXinterchartoks \z@ \FB@guilf = {%
657            \ifFB@spacing\FB@guillspace\fi}%
658        \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
659            \ifFB@spacing\FB@guillspace\fi}%
660        \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
661            \ifFB@spacing\unskip\FB@guillspace\fi}%
```

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

```
662       \bbl@for\FB@char
663              {'\[,'\(,"A0,"202F}%
664              {\XeTeXcharclass\FB@char=\FB@punctnul}%
```

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

```
665       \bbl@for\FB@char
666              {'\{,'\,,'\.,'\-,'\),'\],'\},'\%,"22,"27,"60,"2019}%
667              {\XeTeXcharclass\FB@char=\z@}%
668     }
669   \addto\extrasfrench{\FB@xetex@punct@french}
```

End of specific code for punctuation with modern XeTeX engines.

```
670 \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.

```
671 \ifFB@active@punct
672   \initiate@active@char{:}%
673   \initiate@active@char{;}%
674   \initiate@active@char{!}%
675   \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.

```
676   \declare@shorthand{french}{;}{%
677     \ifFB@spacing
678       \ifhmode
679         \ifdim\lastskip>1sp
680           \unskip\penalty\@M\FBthinspace
681         \else
682           \FDP@thinspace
683         \fi
684       \fi
685     \fi
```

Now we can insert a ; character.

```
686     \string;}
```

The next three definitions are very similar.

```
687   \declare@shorthand{french}{!}{%
688     \ifFB@spacing
689       \ifhmode
```

```
690        \ifdim\lastskip>1sp
691          \unskip\penalty\@M\FBthinspace
692        \else
693          \FDP@thinspace
694        \fi
695      \fi
696    \fi
697    \string!}
698  \declare@shorthand{french}{?}{%
699    \ifFB@spacing
700      \ifhmode
701        \ifdim\lastskip>1sp
702          \unskip\penalty\@M\FBthinspace
703        \else
704          \FDP@thinspace
705        \fi
706      \fi
707    \fi
708    \string?}
709  \declare@shorthand{french}{:}{%
710    \ifFB@spacing
711      \ifhmode
712        \ifdim\lastskip>1sp
713          \unskip\penalty\@M\FBcolonspace
714        \else
715          \FDP@colonspace
716        \fi
717      \fi
718    \fi
719    \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.

```
720    \declare@shorthand{system}{:}{\string:}
721    \declare@shorthand{system}{!}{\string!}
722    \declare@shorthand{system}{?}{\string?}
723    \declare@shorthand{system}{;}{\string;}
724 %}
```

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

```
725    \addto\extrasfrench{\languageshorthands{french}%
```

These characters are 'turned on' once, later their definition may vary. Don't misun-
derstand the following code: they keep being active all along the document, even
when leaving French.

```
726      \bbl@activate{:}\bbl@activate{;}%
727      \bbl@activate{!}\bbl@activate{?}%
728    }
729  \addto\noextrasfrench{%
730      \bbl@deactivate{:}\bbl@deactivate{;}%
```

```
731     \bbl@deactivate{!}\bbl@deactivate{?}%
732   }
733 \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'. it's default value is true, but it can be set to false by \frenchsetup{AutoSpacePunctuation=false} for finer control.

```
734 \newif\ifFBAutoSpacePunctuation   \FBAutoSpacePunctuationtrue
```

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

```
735 \def\autospace@beforeFDP{%
736   \ifFB@luatex@punct\FB@addDPspace=1 \fi
737   \def\FDP@thinspace{\penalty\@M\FBthinspace}%
738   \def\FDP@colonspace{\penalty\@M\FBcolonspace}}
739 \def\noautospace@beforeFDP{%
740   \ifFB@luatex@punct\FB@addDPspace=0 \fi
741   \let\FDP@thinspace\@empty
742   \let\FDP@colonspace\@empty}
743 \ifLaTeXe
744   \def\AutoSpaceBeforeFDP{\autospace@beforeFDP
745                           \FBAutoSpacePunctuationtrue}
746   \def\NoAutoSpaceBeforeFDP{\noautospace@beforeFDP
747                             \FBAutoSpacePunctuationfalse}
748   \AtEndOfPackage{\AutoSpaceBeforeFDP}
749 \else
750   \let\AutoSpaceBeforeFDP\autospace@beforeFDP
751   \let\NoAutoSpaceBeforeFDP\noautospace@beforeFDP
752   \AutoSpaceBeforeFDP
753 \fi
```

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

35

characters with the 'og'/'fg' options in `\frenchsetup{}`. This is also a workaround for the weird behaviour of these characters in verbatim mode.

```
754 \ifLaTeXe
755   \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
756   \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on  \rmfamilyORI}
757   \DeclareRobustCommand\sffamilyFB{\FB@spacing@on  \sffamilyORI}
758 \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.

```
759 \DeclareRobustCommand*{\NoAutoSpacing}{%
760   \FB@spacing@off
761   \ifFB@active@punct\shorthandoff{;:!?}\fi
762 }
```

2.3 Commands for French quotation marks

\guillemotleft pdfLaTeX users are supposed to use 8-bit output encodings (T1, LY1,...) to typeset
\guillemotright French, those who still stick to OT1 should load aeguill or a similar package. In
\textquoteddblleft both cases the commands \guillemotleft and \guillemotright will print the
\textquoteddblright French opening and closing quote characters from the output font. For XeLaTeX and LuaLaTeX, \guillemotleft and \guillemotright are defined by package fontspec (v. 2.5d and up).

We provide the following definitions for non-LaTeX users only as fall-back, they are welcome to change them for anything better.

```
763 \ifLaTeXe
764 \else
765   \ifFBunicode
766     \def\guillemotleft{{\char"00AB}}
767     \def\guillemotright{{\char"00BB}}
768     \def\textquotedblleft{{\char"201C}}
769     \def\textquotedblright{{\char"201D}}
770   \else
771     \def\guillemotleft{\leavevmode\raise0.25ex
772                         \hbox{$\scriptscriptstyle\ll$}}
773     \def\guillemotright{\raise0.25ex
774                         \hbox{$\scriptscriptstyle\gg$}}
775     \def\textquotedblleft{''}
776     \def\textquotedblright{''}
777   \fi
778   \let\xspace\relax
779 \fi
```

\FBgspchar The next step is to provide correct spacing after '«' and before '»'; no line break is
\FB@og allowed neither *after* the opening one, nor *before* the closing one. French quotes
\FB@fg

36

(including 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.

```
780 \newcommand*{\FB@guillspace}{\penalty\@M\FBguillspace}
781 \newcommand*{\FBgspchar}{\char"A0\relax}
782 \newif\ifFBucsNBSP
783 \ifFB@luatex@punct
784   \DeclareRobustCommand*{\FB@og}{\leavevmode
785         \bgroup\FB@spacing=0 \guillemotleft\egroup
786         \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi}
787   \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>\z@\unskip\fi
788         \ifFBucsNBSP\FBgspchar\else\FB@guillspace\fi
789         \bgroup\FB@spacing=0 \guillemotright\egroup}
790 \fi
```

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

```
791 \ifFB@xetex@punct
792   \DeclareRobustCommand*{\FB@og}{\leavevmode
793         \bgroup\FB@spacingfalse\guillemotleft\egroup
794         \FB@guillspace}
795   \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>\z@\unskip\fi
796         \FB@guillspace
797         \bgroup\FB@spacingfalse\guillemotright\egroup}
798 \fi
799 \ifFB@active@punct
800   \DeclareRobustCommand*{\FB@og}{\leavevmode
801         \guillemotleft
802         \FB@guillspace}
803   \DeclareRobustCommand*{\FB@fg}{\ifdim\lastskip>\z@\unskip\fi
804         \FB@guillspace
805         \guillemotright}
806 \fi
```

\og The user level macros for quotation marks are named \og ("ouvrez guillemets") and
\fg \fg ("fermez guillemets"). Another option for typesetting quotes in French is to use the command \frquote (see below). Dummy definition of \og and \fg just to ensure that this commands are not yet defined.

```
807 \newcommand*{\og}{\@empty}
808 \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).

```
809 \ifLaTeXe
810   \def\bbl@frenchguillemets{\renewcommand*{\og}{\FB@og}%
```

```
811                            \renewcommand*{\fg}{\FB@fg\xspace}}
812    \renewcommand*{\og}{\textquotedblleft}
813    \renewcommand*{\fg}{\ifdim\lastskip>\z@\unskip\fi
814                       \textquotedblright\xspace}
815 \else
816    \def\bbl@frenchguillemets{\let\og\FB@og
817                            \let\fg\FB@fg}
818    \def\og{\textquotedblleft}
819    \def\fg{\ifdim\lastskip>\z@\unskip\fi\textquotedblright}
820 \fi

821 \addto\extrasfrench{\babel@save\og \babel@save\fg \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...

```
822 \newcommand*{\ogi}{\FB@og}
823 \newcommand*{\fgi}{\FB@fg}
824 \newcommand*{\ogii}{\textquotedblleft}
825 \newcommand*{\fgii}{\textquotedblright}
```

and the needed technical stuff to handle options:

```
826 \newcount\FBguill@level
827 \newtoks\FB@everypar
828 \newif\ifFBcloseguill  \FBcloseguilltrue
829 \newif\ifFBInnerGuillSingle
830 \def\FBguillopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
831 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
832 \let\FBguillnone\empty
833 \let\FBeveryparguill\FBguillopen
834 \let\FBeverylineguill\FBguillnone
```

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.

```
835 \ifLaTeXe
836    \DeclareRobustCommand\frquote{%
837        \@ifstar{\FBcloseguillfalse\fr@quote}%
838                {\FBcloseguilltrue\fr@quote}}
839 \else
840    \newcommand\frquote[1]{\fr@quote{#1}}
841 \fi
```

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

```
842 \newcommand{\fr@quote}[1]{%
843    \leavevmode
844    \advance\FBguill@level by \@ne
845    \ifcase\FBguill@level
846    \or
```

This for level 1 (outer) quotations: save \everypar before customising it, set \FBeverypar@quote for level 1 quotations and add it to \everypar, then print the quotation:

```
847    \FB@everypar=\everypar
848    \ifx\FBeveryparguill\FBguillnone
849    \else
850      \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
851      \everypar=\expandafter{\the\everypar \FBeverypar@quote}%
852    \fi
853    \ogi #1\fgi
854  \or
```

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

```
855    \ifx\FBeverylineguill\FBguillopen
856      \localleftbox{\guillemotleft\FB@guillspace}%
857      \let\FBeverypar@quote\relax
858      \ogi #1\ifFBcloseguill\fgi\fi
859    \else
860      \ifx\FBeverylineguill\FBguillclose
861        \localleftbox{\guillemotright\FB@guillspace}%
862        \let\FBeverypar@quote\relax
863        \ogi #1\ifFBcloseguill\fgi\fi
864      \else
```

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

```
865        \let\FBeverypar@quote\relax
866        \ifFBInnerGuillSingle
867          \def\ogii{\leavevmode
868                   \guilsinglleft\FB@guillspace}%
869          \def\fgii{\ifdim\lastskip>\z@\unskip\fi
870                   \FB@guillspace\guilsinglright}%
871          \ifx\FBeveryparguill\FBguillopen
872            \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
873          \fi
874          \ifx\FBeveryparguill\FBguillclose
875            \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
876          \fi
877        \fi
878        \ogii #1\ifFBcloseguill \fgii \fi
879      \fi
880    \fi
881  \else
```

Warn if \FBguill@level > 2:

```
882    \ifx\PackageWarning\@undefined
883      \fb@warning{\noexpand\frquote\space handles up to
884                 two levels.\\ Quotation not printed.}%
885    \else
886      \PackageWarning{french.ldf}{%
```

```
887          \protect\frquote\space handles up to two levels.
888          \MessageBreak Quotation not printed.  Reported}
889    \fi
890  \fi
```

Clean on exit: adjust \FBguill@level and restore \localleftbox and \everypar.

```
891  \advance\FBguill@level by \m@ne
892  \ifx\FBeverylineguill\FBguillnone\else\localleftbox{}\fi
893  \ifx\FBeveryparguill\FBguillnone\else\everypar=\FB@everypar\fi
894 }
```

2.4 Date in French

\frenchtoday
\frenchdate
\datefrench

The following code creates a macro \datefrench which in turn defines command \frenchtoday (\today is defined as \frenchtoday in French). The corresponding commands for the French dialect, \dateacadian and \acadiantoday are also created btw. This new implementation relies on commands \SetString and \SetStringLoop, therefore requires babel 3.10 or newer.

Explicitly defining \BabelLanguages as the list of all French dialects defines *both* \datefrench and \dateacadian; this is required as french.ldf is read only once even if both language options french and acadian are supplied to babel. Note that coding \StartBabelCommands∗{french,acadian} would *only* define \csname date\CurrentOption\endcsname, leaving the second language undefined in babel's sens.

```
895 \def\BabelLanguages{french,acadian}
896 \StartBabelCommands*{\BabelLanguages}{date}
897    [unicode, fontenc=TU EU1 EU2, charset=utf8]
898  \SetString\monthiiname{février}
899  \SetString\monthviiiname{août}
900  \SetString\monthxiiname{décembre}
901 \StartBabelCommands*{\BabelLanguages}{date}
902  \SetStringLoop{month#1name}{%
903      janvier,f\'evrier,mars,avril,mai,juin,juillet,%
904      ao\^ut,septembre,octobre,novembre,d\'ecembre}
905  \SetString\today{\FB@date{\year}{\month}{\day}}
906 \EndBabelCommands
```

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

```
907 \newcommand*{\FB@date}[3]{%
908  {{\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
909  \csname month\romannumeral#2name\endcsname
910  \ifx#1\@empty\else\FBdatespace\number#1\fi}}
911 \newcommand*{\FBdatebox}{\hbox}
912 \newcommand*{\FBdatespace}{\space}
913 \newcommand*{\frenchdate}{\FBdatebox\FB@date}
914 \newcommand*{\acadiandate}{\FBdatebox\FB@date}
```

2.5 Extra utilities

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

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

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

```
915 \newif\ifFB@poorman
916 \newdimen\FB@Mht
917 \ifLaTeXe
918   \AtEndOfPackage{\RequirePackage{scalefnt}}
```

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

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

```
919   \newcommand*{\FBsupR}{-0.12}
920   \newcommand*{\FBsupS}{0.65}
921   \newcommand*{\FB@lc}[1]{\MakeLowercase{#1}}
922   \DeclareRobustCommand*{\FB@up@fake}[1]{%
923     \settoheight{\FB@Mht}{M}%
924     \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
925     \addtolength{\FB@Mht}{-\FBsupS ex}%
926     \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
927     }
```

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.

```
928  \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
929                          \def\FB@suffix{#4}}
930  \def\FB@x{x}
931  \def\FB@j{j}
932  \DeclareRobustCommand*{\FB@up}[1]{%
933    \bgroup \FB@poormantrue
934      \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.

```
935      \edef\reserved@a{\lowercase{%
936        \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
937      \reserved@a
938        {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
939         \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
940         \ifFB@poorman \FB@up@fake{#1}%
941         \else          \FB@up@real{#1}%
942         \fi}%
943        {\FB@up@fake{#1}}%
944    \egroup}
```

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

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

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

```
947  \DeclareRobustCommand*{\fup}[1]{%
948    \ifx\realsuperscript\@undefined
949      \FB@up{#1}%
950    \else
951      \bgroup\let\fakesuperscript\FB@up@fake
952            \realsuperscript{\FB@lc{#1}}\egroup
953    \fi}
```

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

```
954  \providecommand*{\up}{\relax}
```

Poor man's definition of \up for Plain.

```
955 \else
956   \providecommand*{\up}[1]{\leavevmode\raise1ex\hbox{\sevenrm #1}}
957 \fi
```

\ieme   Some handy macros for those who don't know how to abbreviate ordinals:
\ier
\iere   958 \def\ieme{\up{e}\xspace}
\iemes   959 \def\iemes{\up{es}\xspace}
\iers   960 \def\ier{\up{er}\xspace}
\ieres   961 \def\iers{\up{ers}\xspace}

42
```

```
962 \def\iere{\up{re}\xspace}
963 \def\ieres{\up{res}\xspace}
```

```
964 \newcommand*{\FBmedkern}{\kern+.2em}
965 \newcommand*{\FBthickkern}{\kern+.3em}
```

\No And some more macros relying on \up for numbering, first two support macros.
\no
\Nos
```
966 \newcommand*{\FrenchEnumerate}[1]{#1\up{o}\FBthickkern}
967 \newcommand*{\FrenchPopularEnumerate}[1]{#1\up{o})\FBthickkern}
```
\nos
Typing \primo should result in 'ᵒ',
\primo
\fprimo)
```
968 \def\primo{\FrenchEnumerate1}
969 \def\secundo{\FrenchEnumerate2}
970 \def\tertio{\FrenchEnumerate3}
971 \def\quarto{\FrenchEnumerate4}
```

while typing \fprimo) gives 'ᵒ).

```
972 \def\fprimo){\FrenchPopularEnumerate1}
973 \def\fsecundo){\FrenchPopularEnumerate2}
974 \def\ftertio){\FrenchPopularEnumerate3}
975 \def\fquarto){\FrenchPopularEnumerate4}
```

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

```
976 \DeclareRobustCommand*{\No}{N\up{o}\FBmedkern}
977 \DeclareRobustCommand*{\no}{n\up{o}\FBmedkern}
978 \DeclareRobustCommand*{\Nos}{N\up{os}\FBmedkern}
979 \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}.

```
980 \DeclareRobustCommand*{\bsc}[1]{\leavevmode\begingroup\kern0pt
981                                  \scshape #1\endgroup}
982 \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. \degre can be accessed by the command \r{} for ring accent.

```
983 \ifFBunicode
984   \newcommand*{\at}{{\char"0040}}
985   \newcommand*{\circonflexe}{{\char"005E}}
986   \newcommand*{\tild}{{\char"007E}}
987   \newcommand*{\boi}{{\char"005C}}
988   \newcommand*{\degre}{{\char"00B0}}
```

```
 989 \else
 990   \ifLaTeXe
 991     \DeclareTextSymbol{\at}{T1}{64}
 992     \DeclareTextSymbol{\circonflexe}{T1}{94}
 993     \DeclareTextSymbol{\tild}{T1}{126}
 994     \DeclareTextSymbolDefault{\at}{T1}
 995     \DeclareTextSymbolDefault{\circonflexe}{T1}
 996     \DeclareTextSymbolDefault{\tild}{T1}
 997     \DeclareRobustCommand*{\boi}{\textbackslash}
 998     \DeclareRobustCommand*{\degre}{\r{}}
 999   \else
1000     \def\T@one{T1}
1001     \ifx\f@encoding\T@one
1002       \newcommand*{\degre}{{\char6}}
1003     \else
1004       \newcommand*{\degre}{{\char23}}
1005     \fi
1006     \newcommand*{\at}{{\char64}}
1007     \newcommand*{\circonflexe}{{\char94}}
1008     \newcommand*{\tild}{{\char126}}
1009     \newcommand*{\boi}{$\backslash$}
1010   \fi
1011 \fi
```

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

```
1012 \ifLaTeXe
1013   \newcommand*{\degres}{\degre}
1014   \ifFBunicode
1015     \DeclareRobustCommand*{\degres}{\degre}
1016   \else
1017     \def\Warning@degree@TSone{\FBWarning
1018             {Degrees would look better in TS1-encoding:%
1019              \MessageBreak add \protect
1020              \usepackage{textcomp} to the preamble.%
1021              \MessageBreak Degrees used}}
1022     \AtBeginDocument{\ifx\DeclareEncodingSubset\@undefined
1023                       \DeclareRobustCommand*{\degres}{%
1024                         \leavevmode\hbox to 0.3em{\hss\degre\hss}%
1025                       \Warning@degree@TSone
1026                       \global\let\Warning@degree@TSone\relax}%
1027                     \else
1028                       \DeclareRobustCommand*{\degres}{%
1029                         \hbox{\UseTextSymbol{TS1}{\textdegree}}}%
```

```
1030                              \fi
1031                              }
1032    \fi
1033 \else
1034    \newcommand*{\degres}{%
1035        \leavevmode\hbox to 0.3em{\hss\degre\hss}}
1036 \fi
```

2.6 Formatting numbers

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 *only* (no space added); \StandardMathComma switches
back to the standard behaviour of the comma.

Unfortunately, \newcount inside \if breaks Plain formats.

```
1037 \newif\ifFB@icomma
1038 \newcount\mc@charclass
1039 \newcount\mc@charfam
1040 \newcount\mc@charslot
1041 \newcount\std@mcc
1042 \newcount\dec@mcc
1043 \ifFBLuaTeX
1044    \mc@charclass=\Umathcharclass`\,
1045    \newcommand*{\dec@math@comma}{%
1046        \mc@charfam=\Umathcharfam`\,
1047        \mc@charslot=\Umathcharslot`\,
1048        \Umathcode`\,= 0 \mc@charfam \mc@charslot
1049    }
1050    \newcommand*{\std@math@comma}{%
1051        \mc@charfam=\Umathcharfam`\,
1052        \mc@charslot=\Umathcharslot`\,
1053        \Umathcode`\,= \mc@charclass \mc@charfam \mc@charslot
1054    }
1055 \else
1056    \std@mcc=\mathcode`\,
1057    \dec@mcc=\std@mcc
1058    \@tempcnta=\std@mcc
1059    \divide\@tempcnta by "1000
1060    \multiply\@tempcnta by "1000
1061    \advance\dec@mcc by -\@tempcnta
1062    \newcommand*{\dec@math@comma}{\mathcode`\,=\dec@mcc}
1063    \newcommand*{\std@math@comma}{\mathcode`\,=\std@mcc}
1064 \fi
1065 \newcommand*{\DecimalMathComma}{%
1066    \ifFBfrench\dec@math@comma\fi
1067    \ifFB@icomma\else\addto\extrasfrench{\dec@math@comma}\fi
1068 }
1069 \newcommand*{\StandardMathComma}{%
```

45

```
1070    \std@math@comma
1071    \ifFB@icomma\else\addto\extrasfrench{\std@math@comma}\fi
1072 }
1073 \ifLaTeXe
1074    \AtBeginDocument{\@ifpackageloaded{icomma}%
1075                        {\FB@icommatrue}%
1076                        {\addto\noextrasfrench{\std@math@comma}}%
1077    }
1078 \else
1079    \addto\noextrasfrench{\std@math@comma}
1080 \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:

```
1081 \newcommand*{\nombre}[1]{{#1}\fb@warning{*** \noexpand\nombre
1082                                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.

```
1083 \ifFB@luatex@punct
1084    \activate@luatexpunct
1085 \fi
1086 \let\FBstop@here\relax
1087 \def\FBclean@on@exit{%
1088    \let\ifLaTeXe\undefined
1089    \let\LaTeXetrue\undefined
1090    \let\LaTeXefalse\undefined
1091    \let\FB@llc\loadlocalcfg
1092    \let\loadlocalcfg\@gobble}
1093 \ifx\magnification\@undefined
1094 \else
1095    \def\FBstop@here{%
1096        \FBclean@on@exit
1097        \ldf@finish\CurrentOption
1098        \let\loadlocalcfg\FB@llc
1099        \endinput}
1100 \fi
1101 \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.

```
1102 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1103 \newcommand*{\Warning@nombre}[1]{%
```

```
1104    \ifdefined\numprint
1105      \numprint{#1}%
1106    \else
1107      \PackageWarning{french.ldf}{%
1108        \protect\nombre\space now relies on package numprint.sty,%
1109        \MessageBreak add \protect
1110        \usepackage[autolanguage]{numprint},\MessageBreak
1111        see file numprint.pdf for more options.\MessageBreak
1112        \protect\nombre\space called}%
1113      \global\let\Warning@nombre\relax
1114      {#1}%
1115    \fi
1116 }

1117 \newcommand*{\FBthousandsep}{\kern \fontdimen2\font \relax}
```

2.7 Caption names

The next step consists in defining the French equivalents for the LaTeX caption names.

\captionsfrench Let's first define \captionsfrench which sets all strings used in the four standard document classes provided with LaTeX.
Let's give a chance to a class or a package read before babel-french to define \FBfigtabshape as \relax, otherwise \FBfigtabshape will be defined as \scshape (can be changed with \frenchsetup{SmallCapsFigTabCaptions=false}).

```
1118 \providecommand*{\FBfigtabshape}{\scshape}
```

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

```
1119 \StartBabelCommands*{\BabelLanguages}{captions}
1120      [unicode, fontenc=TU EU1 EU2, charset=utf8]
1121    \SetString{\refname}{Références}
1122    \SetString{\abstractname}{Résumé}
1123    \SetString{\prefacename}{Préface}
1124    \SetString{\contentsname}{Table des matières}
1125    \SetString{\ccname}{Copie à }
1126    \SetString{\proofname}{Démonstration}
1127    \SetString{\partfirst}{Première}
1128    \SetString{\partsecond}{Deuxième}
1129    \SetStringLoop{ordinal#1}{%
1130      \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%
1131      Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1132      Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1133      Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1134 \StartBabelCommands*{\BabelLanguages}{captions}
1135    \SetString{\refname}{R\'ef\'erences}
1136    \SetString{\abstractname}{R\'esum\'e}
1137    \SetString{\bibname}{Bibliographie}
1138    \SetString{\prefacename}{Pr\'eface}
1139    \SetString{\chaptername}{Chapitre}
1140    \SetString{\appendixname}{Annexe}
```

```
1141    \SetString{\contentsname}{Table des mati\`eres}
1142    \SetString{\listfigurename}{Table des figures}
1143    \SetString{\listtablename}{Liste des tableaux}
1144    \SetString{\indexname}{Index}
1145    \SetString{\figurename}{{\FBfigtabshape Figure}}
1146    \SetString{\tablename}{{\FBfigtabshape Table}}
1147    \SetString{\pagename}{page}
1148    \SetString{\seename}{voir}
1149    \SetString{\alsoname}{voir aussi}
1150    \SetString{\enclname}{P.~J. }
1151    \SetString{\ccname}{Copie \`a }
1152    \SetString{\headtoname}{}
1153    \SetString{\proofname}{D\'emonstration}
1154    \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`.

```
1155    \SetString{\partfirst}{Premi\`ere}
1156    \SetString{\partsecond}{Deuxi\`eme}
1157    \SetString{\partnameord}{partie}
1158    \SetStringLoop{ordinal#1}{%
1159      \partfirst,\partsecond,Troisi\`eme,Quatri\`eme,%
1160      Cinqui\`eme,Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
1161      Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1162      Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1163      Vingti\`eme}
1164    \AfterBabelCommands{%
1165      \DeclareRobustCommand*{\FB@emptypart}{\def\thepart{}}%
1166      \DeclareRobustCommand*{\FB@partname}{%
1167        \ifFBPartNameFull
1168          \csname ordinal\romannumeral\value{part}\endcsname\space
1169          \partnameord\FB@emptypart
1170        \else
1171          Partie%
1172        \fi}%
1173      }
1174    \SetString{\partname}{\FB@partname}
1175 \EndBabelCommands
```

2.8   Figure and table captions

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

```
1176 \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 preceed the colon in French). This flaw may occur with pdfLaTeX as ':' is made active too late. With LuaLaTeX and XeLaTeX, this

48

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).

```
1177 \bgroup
1178   \catcode`:=12 \catcode`>=12 \relax
1179   \long\gdef\STD@makecaption#1#2{%
1180     \vskip\abovecaptionskip
1181     \sbox\@tempboxa{#1: #2}%
1182     \ifdim \wd\@tempboxa >\hsize
1183       #1: #2\par
1184     \else
1185       \global \@minipagefalse
1186       \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1187     \fi
1188     \vskip\belowcaptionskip}
1189 \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.

```
1190 \newif\if@FBwarning@capsep
1191 \ifFB@active@punct\@FBwarning@capseptrue\fi
1192 \newcommand*{\CaptionSeparator}{\space\textendash\space}
1193 \def\FBCaption@Separator{: }
1194 \long\def\FB@makecaption#1#2{%
1195   \vskip\abovecaptionskip
1196   \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1197   \ifdim \wd\@tempboxa >\hsize
1198     #1\FBCaption@Separator #2\par
1199   \else
1200     \global \@minipagefalse
1201     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1202   \fi
1203   \vskip\belowcaptionskip}
```

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

```
1204 \@ifclassloaded{acmart}{\@FBwarning@capsepfalse}{}
1205 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1206 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1207 \@ifclassloaded{amsdtx}{\@FBwarning@capsepfalse}{}
1208 \@ifclassloaded{amsldoc}{\@FBwarning@capsepfalse}{}
1209 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1210 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1211 \@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) .

```
1212 \newif\ifFB@koma
1213 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1214 \@ifclassloaded{scrartcl}{\@FBwarning@capsepfalse\FB@komatrue}{}
1215 \@ifclassloaded{scrbook}{\@FBwarning@capsepfalse\FB@komatrue}{}
1216 \@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).

```
1217 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1218 \ifdefined\@makecaption\else\@FBwarning@capsepfalse\fi
```

The caption, subcaption and floatrow packages are compatible with babel-french if they are loaded after babel.
Check if packages caption3 subcaption or floatrow are loaded now (before babel-french) and step counter FBcaption@count accordingly; it's value will be checked \AtBeginDocument. N.B.: caption loads caption3, subcaption loads caption3 and floatrow loads caption3.

```
1219 \newcounter{FBcaption@count}
1220 \@ifpackageloaded{caption3}{\addtocounter{FBcaption@count}{4}}{}
1221 \@ifpackageloaded{subcaption}{\addtocounter{FBcaption@count}{2}}{}
1222 \@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').

```
1223 \AtBeginDocument{%
1224   \ifx\@makecaption\STD@makecaption
1225     \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.

```
1226     \ifFBOldFigTabCaptions
1227     \else
1228       \def\FBCaption@Separator{\ifFBfrench\space\fi : }%
```

50

```
1229      \fi
1230      \ifFBCustomiseFigTabCaptions
1231        \ifFB@mainlanguage@FR
1232          \def\FBCaption@Separator{\CaptionSeparator}%
1233        \fi
1234      \fi
1235      \@FBwarning@capsepfalse
1236    \fi
```

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

```
1237   \@ifpackageloaded{caption3}{%
1238     \ifnum\value{FBcaption@count}=0 \@FBwarning@capsepfalse\fi
1239     }{}%
1240   \if@FBwarning@capsep
1241     \ifnum\value{FBcaption@count}>0
```

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

```
1242       \FBWarning
1243       {Figures' and tables' captions might look like\MessageBreak
1244        'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1245        If you have loaded any of the packages caption,\MessageBreak
1246        subcaption or floatrow BEFORE babel/french,\MessageBreak
1247        please move them AFTER babel/french.\MessageBreak
1248        If one of them is loaded by your class,\MessageBreak
1249        you can still add AFTER babel/french\MessageBreak
1250        \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1251        \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1252        ... live with it; reported}%
1253     \else
```

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

```
1254       \FBWarning
1255       {Figures' and tables' captions might look like\MessageBreak
1256        'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1257        If it happens, see your class documentation to\MessageBreak
1258        fix this issue or add AFTER babel/french\MessageBreak
1259        \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1260        \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1261        or ... live with it; reported}%
1262     \fi
1263   \fi
1264   \let\FB@makecaption\relax
1265   \let\STD@makecaption\relax
1266 }
```

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.

```
1267 \ifFBunicode
1268   \let\FBtextellipsis\textellipsis
1269 \else
1270   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1271   \DeclareTextCommandDefault{\FBtextellipsis}{%
1272     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}
1273 \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.

```
1274 \newcommand*{\Tdots@}{\@xp\textellipsis}
1275 \newcommand*{\Mdots@}{\@xp\mdots@}
1276 \AtBeginDocument{\DeclareRobustCommand*{\dots}{\relax
1277                 \csname\ifmmode M\else T\fi dots@\endcsname}%
1278                 \ifdefined\@xp\else\let\@xp\relax\fi
1279                 \ifdefined\mdots@\else\let\Mdots@\mathellipsis\fi
1280                }
1281 \def\bbl@frenchdots{\babel@save\Tdots@ \let\Tdots@\FBtextellipsis}
1282 \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).

```
1283 \ifFB@active@punct
1284   \@ifpackageloaded{listings}
1285     {\AtBeginDocument{%
1286       \FBWarning{Please load the "listings" package\MessageBreak
1287               AFTER babel/french; reported}}%
1288     }{}
1289 \fi
```

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

```
1290 \newif\if@FBwarning@natbib
1291 \ifFB@active@punct
1292   \@ifpackageloaded{natbib}{}{\@FBwarning@natbibtrue}
1293 \fi
1294 \AtBeginDocument{%
1295   \if@FBwarning@natbib
1296     \@ifpackageloaded{natbib}{}{\@FBwarning@natbibfalse}%
```

```
1297      \fi
1298    \if@FBwarning@natbib
1299      \FBWarning{Please load the "natbib" package\MessageBreak
1300                BEFORE babel/french; reported}%
1301    \fi
1302 }
```

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

```
1303 \newif\if@FBwarning@beamerarticle
1304 \@ifpackageloaded{beamerarticle}{}{\@FBwarning@beamerarticletrue}
1305 \AtBeginDocument{%
1306    \if@FBwarning@beamerarticle
1307      \@ifpackageloaded{beamerarticle}{}%
1308                                      {\@FBwarning@beamerarticlefalse}%
1309    \fi
1310    \if@FBwarning@beamerarticle
1311      \FBWarning{Please load the "beamerarticle" package\MessageBreak
1312                BEFORE babel/french; reported}%
1313    \fi
1314 }
```

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.

```
1315 \newcommand*{\frenchsetup}[1]{%
1316    \setkeys{FB}{#1}%
1317 }%
1318 \@onlypreamble\frenchsetup
```

Keep the former name \frenchbsetup working for compatibility.

```
1319 \let\frenchbsetup\frenchsetup
1320 \@onlypreamble\frenchbsetup
```

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

```
1321 \newif\ifFBShowOptions
1322 \newif\ifFBStandardLayout            \FBStandardLayouttrue
1323 \newif\ifFBGlobalLayoutFrench        \FBGlobalLayoutFrenchtrue
1324 \newif\ifFBReduceListSpacing
1325 \newif\ifFBListOldLayout
1326 \newif\ifFBCompactItemize
1327 \newif\ifFBStandardItemizeEnv        \FBStandardItemizeEnvtrue
1328 \newif\ifFBStandardEnumerateEnv      \FBStandardEnumerateEnvtrue
1329 \newif\ifFBStandardItemLabels        \FBStandardItemLabelstrue
1330 \newif\ifFBStandardLists             \FBStandardListstrue
1331 \newif\ifFBIndentFirst
1332 \newif\ifFBFrenchFootnotes
1333 \newif\ifFBAutoSpaceFootnotes
1334 \newif\ifFBOriginalTypewriter
1335 \newif\ifFBThinColonSpace
1336 \newif\ifFBThinSpaceInFrenchNumbers
1337 \newif\ifFBFrenchSuperscripts        \FBFrenchSuperscriptstrue
1338 \newif\ifFBLowercaseSuperscripts     \FBLowercaseSuperscriptstrue
1339 \newif\ifFBPartNameFull              \FBPartNameFulltrue
1340 \newif\ifFBCustomiseFigTabCaptions
1341 \newif\ifFBOldFigTabCaptions
1342 \newif\ifFBSmallCapsFigTabCaptions   \FBSmallCapsFigTabCaptionstrue
1343 \newif\ifFBSuppressWarning
1344 \newif\ifFBINGuillSpace
```

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

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

```
1345 \ifFB@koma
1346   \ifdefined\partformat
1347     \def\FB@partformat@fix{%
1348          \ifFBPartNameFull
1349             \babel@save\partformat
1350             \renewcommand*{\partformat}{\partname}%
1351          \fi}
1352     \addto\extrasfrench{\FB@partformat@fix}%
1353   \fi
1354 \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.

```
1355 \def\FB@french{french}
1356 \def\FB@acadian{acadian}
1357 \newif\ifFB@mainlanguage@FR
1358 \AtEndOfPackage{%
1359   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1360   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1361   \fi
1362   \ifFB@mainlanguage@FR
1363     \FBGlobalLayoutFrenchtrue
1364     \@ifclassloaded{beamer}%
1365       {\PackageInfo{french.ldf}{%
1366           No list customisation for the beamer class,%
1367           \MessageBreak reported}}%
1368       {\@ifpackageloaded{beamerarticle}%
1369         {\FBStandardItemLabelsfalse
1370          \FBReduceListSpacingtrue
1371          \PackageInfo{french.ldf}{%
1372             Minimal list customisation for the beamerarticle%
1373             \MessageBreak package; reported}}%
```

Otherwise customise lists "à la française":

```
1374         {\FBReduceListSpacingtrue
1375          \FBStandardItemizeEnvfalse
1376          \FBStandardEnumerateEnvfalse
1377          \FBStandardItemLabelsfalse}%
1378       }
1379     \FBIndentFirsttrue
1380     \FBFrenchFootnotestrue
1381     \FBAutoSpaceFootnotestrue
1382     \FBCustomiseFigTabCaptionstrue
1383   \else
1384     \FBGlobalLayoutFrenchfalse
1385   \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.

```
1386   \RequirePackage{keyval}%
1387   \define@key{FB}{ShowOptions}[true]%
1388           {\csname FBShowOptions#1\endcsname}%
1389   \define@key{FB}{StandardLayout}[true]%
1390           {\csname FBStandardLayout#1\endcsname
1391            \ifFBStandardLayout
1392              \FBReduceListSpacingfalse
1393              \FBStandardItemizeEnvtrue
1394              \FBStandardItemLabelstrue
1395              \FBStandardEnumerateEnvtrue
1396              \FBIndentFirstfalse
1397              \FBFrenchFootnotesfalse
```

```
1398              \FBAutoSpaceFootnotesfalse
1399              \FBGlobalLayoutFrenchfalse
1400            \else
1401              \FBReduceListSpacingtrue
1402              \FBStandardItemizeEnvfalse
1403              \FBStandardItemLabelsfalse
1404              \FBStandardEnumerateEnvfalse
1405              \FBIndentFirsttrue
1406              \FBFrenchFootnotestrue
1407              \FBAutoSpaceFootnotestrue
1408            \fi}%
1409   \define@key{FB}{GlobalLayoutFrench}[true]%
1410          {\csname FBGlobalLayoutFrench#1\endcsname
```

If this key is set to true when French is the main language, nothing to do: all flags keep their default value. If this key is set to false, nothing to do either: \babel@save will do the job. Warn and reset in case this key is set to true while the main language is *not* French.

```
1411              \ifFBGlobalLayoutFrench
1412                \ifFB@mainlanguage@FR
1413                \else
1414                  \FBGlobalLayoutFrenchfalse
1415                  \PackageWarning{french.ldf}%
1416                    {Option 'GlobalLayoutFrench' skipped:\MessageBreak
1417                     French is *not* babel's last option.\MessageBreak
1418                     Reported}%
1419                \fi
1420              \fi}%
1421   \define@key{FB}{ReduceListSpacing}[true]%
1422          {\csname FBReduceListSpacing#1\endcsname}%
1423   \define@key{FB}{ListOldLayout}[true]%
1424          {\csname FBListOldLayout#1\endcsname
1425            \ifFBListOldLayout
1426              \FBStandardEnumerateEnvtrue
1427              \renewcommand*{\FrenchLabelItem}{\textendash}%
1428            \fi}%
1429   \define@key{FB}{CompactItemize}[true]%
1430          {\csname FBCompactItemize#1\endcsname
1431            \ifFBCompactItemize
1432              \FBStandardItemizeEnvfalse
1433              \FBStandardEnumerateEnvfalse
1434            \else
1435              \FBStandardItemizeEnvtrue
1436              \FBStandardEnumerateEnvtrue
1437            \fi}%
1438   \define@key{FB}{StandardItemizeEnv}[true]%
1439          {\csname FBStandardItemizeEnv#1\endcsname}%
1440   \define@key{FB}{StandardEnumerateEnv}[true]%
1441          {\csname FBStandardEnumerateEnv#1\endcsname}%
1442   \define@key{FB}{StandardItemLabels}[true]%
1443          {\csname FBStandardItemLabels#1\endcsname}%
```

```
1444  \define@key{FB}{ItemLabels}%
1445        {\renewcommand*{\FrenchLabelItem}{#1}}%
1446  \define@key{FB}{ItemLabeli}%
1447        {\renewcommand*{\Frlabelitemi}{#1}}%
1448  \define@key{FB}{ItemLabelii}%
1449        {\renewcommand*{\Frlabelitemii}{#1}}%
1450  \define@key{FB}{ItemLabeliii}%
1451        {\renewcommand*{\Frlabelitemiii}{#1}}%
1452  \define@key{FB}{ItemLabeliv}%
1453        {\renewcommand*{\Frlabelitemiv}{#1}}%
1454  \define@key{FB}{StandardLists}[true]%
1455        {\csname FBStandardLists#1\endcsname
1456         \ifFBStandardLists
1457           \FBReduceListSpacingfalse
1458           \FBCompactItemizefalse
1459           \FBStandardItemizeEnvtrue
1460           \FBStandardEnumerateEnvtrue
1461           \FBStandardItemLabelstrue
1462         \else
1463           \FBReduceListSpacingtrue
1464           \FBCompactItemizetrue
1465           \FBStandardItemizeEnvfalse
1466           \FBStandardEnumerateEnvfalse
1467           \FBStandardItemLabelsfalse
1468         \fi}%
1469  \define@key{FB}{IndentFirst}[true]%
1470        {\csname FBIndentFirst#1\endcsname}%
1471  \define@key{FB}{FrenchFootnotes}[true]%
1472        {\csname FBFrenchFootnotes#1\endcsname}%
1473  \define@key{FB}{AutoSpaceFootnotes}[true]%
1474        {\csname FBAutoSpaceFootnotes#1\endcsname}%
1475  \define@key{FB}{AutoSpacePunctuation}[true]%
1476        {\csname FBAutoSpacePunctuation#1\endcsname}%
1477  \define@key{FB}{OriginalTypewriter}[true]%
1478        {\csname FBOriginalTypewriter#1\endcsname}%
1479  \define@key{FB}{ThinColonSpace}[true]%
1480        {\csname FBThinColonSpace#1\endcsname
1481         \ifFBThinColonSpace
1482           \renewcommand*{\FBcolonspace}{\FBthinspace}%
1483         \fi}%
1484  \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1485        {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1486  \define@key{FB}{FrenchSuperscripts}[true]%
1487        {\csname FBFrenchSuperscripts#1\endcsname}
1488  \define@key{FB}{LowercaseSuperscripts}[true]%
1489        {\csname FBLowercaseSuperscripts#1\endcsname}
1490  \define@key{FB}{PartNameFull}[true]%
1491        {\csname FBPartNameFull#1\endcsname}%
1492  \define@key{FB}{CustomiseFigTabCaptions}[true]%
1493        {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1494  \define@key{FB}{OldFigTabCaptions}[true]%
```

```
1495            {\csname FBOldFigTabCaptions#1\endcsname
1496             \ifFBOldFigTabCaptions
1497               \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1498                   \def\FBCaption@Separator{\CaptionSeparator}}%
1499               \addto\extrasfrench{\FB@capsep@fix}%
1500               \ifdefined\extrasacadian
1501                 \addto\extrasacadian{\FB@capsep@fix}%
1502               \fi
1503             \fi}%
1504   \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1505            {\csname FBSmallCapsFigTabCaptions#1\endcsname
1506             \ifFBSmallCapsFigTabCaptions
1507               \let\FBfigtabshape\scshape
1508             \else
1509               \let\FBfigtabshape\relax
1510             \fi}%
1511   \define@key{FB}{SuppressWarning}[true]%
1512            {\csname FBSuppressWarning#1\endcsname
1513             \ifFBSuppressWarning
1514               \renewcommand{\FBWarning}[1]{}%
1515             \fi}%
```

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

```
1516   \define@key{FB}{INGuillSpace}[true]%
1517            {\csname FBINGuillSpace#1\endcsname
1518             \ifFBINGuillSpace
1519               \renewcommand*{\FBguillspace}{\space}%
1520             \fi}%
1521   \define@key{FB}{InnerGuillSingle}[true]%
1522            {\csname FBInnerGuillSingle#1\endcsname}%
1523   \define@key{FB}{EveryParGuill}[open]%
1524            {\expandafter\let\expandafter
1525              \FBeveryparguill\csname FBguill#1\endcsname
1526             \ifx\FBeveryparguill\FBguillopen
1527             \else\ifx\FBeveryparguill\FBguillclose
1528                 \else\ifx\FBeveryparguill\FBguillnone
1529                    \else
1530                      \let\FBeveryparguill\FBguillopen
1531                      \FBWarning{Wrong value for 'EveryParGuill':
1532                                  try 'open',\MessageBreak
1533                                  'close' or 'none'. Reported}%
1534                    \fi
1535                \fi
1536             \fi}%
1537   \define@key{FB}{EveryLineGuill}[open]%
1538            {\ifFB@luatex@punct
1539               \expandafter\let\expandafter
1540                 \FBeverylineguill\csname FBguill#1\endcsname
1541               \ifx\FBeverylineguill\FBguillopen
1542               \else\ifx\FBeverylineguill\FBguillclose
```

58

```
1543                    \else\ifx\FBeverylineguill\FBguillnone
1544                        \else
1545                         \let\FBeverylineguill\FBguillnone
1546                         \FBWarning{Wrong value for 'EveryLineGuill':
1547                                    try 'open',\MessageBreak
1548                                    'close' or 'none'. Reported}%
1549                        \fi
1550                    \fi
1551                \fi
1552            \else
1553              \FBWarning{Option 'EveryLineGuill' skipped:%
1554                        \MessageBreak this option is for
1555                        LuaTeX *only*.\MessageBreak Reported}%
1556        \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.

```
1557    \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1558            {\ifFB@luatex@punct
1559              \csname FBucsNBSP#1\endcsname
1560              \ifFBucsNBSP \FB@ucsNBSP=1 \fi
1561            \else
1562              \FBWarning{Option 'UnicodeNoBreakSpaces' skipped:%
1563                        \MessageBreak this option is for
1564                        LuaTeX *only*.\MessageBreak Reported}%
1565            \fi
1566            }%
```

Inputing French quotes as *single characters* when they are available on the keyboard (through a compose key for instance) is more comfortable than typing \og and \fg. Life is simple here with modern LuaTeX or XeTeX engines: we just have to activate the \FB@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 checks if a character is single-byte (then \FB@second is empty) or not.

```
1567    \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
```

```
1568    \define@key{FB}{og}%
1569            {\ifFBunicode
```

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

```
1570                \ifFB@luatex@punct
1571                  \FB@addGUILspace=1 \relax
1572                \fi
```

then with XeTeX it is a bit more tricky:

```
1573              \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).

```
1574              \XeTeXcharclass"13   = \FB@guilo
1575              \XeTeXcharclass"AB   = \FB@guilo
1576              \XeTeXcharclass"A0   = \FB@guilnul
1577              \XeTeXcharclass"202F = \FB@guilnul
1578            \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1579            \ifFB@active@punct
1580              \FBWarning{Option og=« not supported with this version
1581                        of\MessageBreak LuaTeX/XeTeX; reported}%
1582            \fi
1583          \else
```

This is for conventional TeX engines:

```
1584            \newcommand*{\FB@@og}{%
1585              \ifFBfrench
1586                \ifFB@spacing\FB@og\ignorespaces
1587                \else\guillemotleft
1588                \fi
1589              \else\guillemotleft\fi}%
1590            \AtBeginDocument{%
1591              \ifdefined\uc@dclc
```

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

```
1592                \uc@dclc{171}{default}{\FB@@og}%
1593              \else
```

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

```
1594                \FB@parse#1\endparse
1595                \ifx\FB@second\@empty
```

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

```
1596                  \ifdefined\mule@def
1597                    \mule@def{11}{\FB@@og}%
1598                  \else
1599                    \ifdefined\DeclareInputText
1600                      \@tempcnta`#1\relax
1601                      \DeclareInputText{\the\@tempcnta}{\FB@@og}%
1602                    \else
```

Package inputenc not loaded, no way...

```
1603                      \FBWarning{Option 'og' requires package
1604                                inputenc;\MessageBreak reported}%
1605                    \fi
1606                  \fi
1607                \else
```

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

```
1608                    \DeclareUnicodeCharacter{00AB}{\FB@@og}%
1609                \fi
1610              \fi}%
1611            \fi
1612          }%
```

Same code for the closing quote.

```
1613    \define@key{FB}{fg}%
1614          {\ifFBunicode
1615             \ifFB@luatex@punct
1616               \FB@addGUILspace=1 \relax
1617             \fi
1618             \ifFB@xetex@punct
1619               \XeTeXcharclass"14   = \FB@guilf
1620               \XeTeXcharclass"BB   = \FB@guilf
1621               \XeTeXcharclass"A0   = \FB@guilnul
1622               \XeTeXcharclass"202F = \FB@guilnul
1623             \fi
1624             \ifFB@active@punct
1625               \FBWarning{Option fg=» not supported with this version
1626                          of\MessageBreak LuaTeX/XeTeX; reported}%
1627             \fi
1628           \else
1629             \newcommand*{\FB@@fg}{%
1630                \ifFBfrench
1631                  \ifFB@spacing\FB@fg
1632                  \else\guillemotright
1633                  \fi
1634                \else\guillemotright\fi}%
1635             \AtBeginDocument{%
1636                \ifdefined\uc@dclc
1637                  \uc@dclc{187}{default}{\FB@@fg}%
1638                \else
1639                  \FB@parse#1\endparse
1640                  \ifx\FB@second\@empty
1641                    \ifdefined\mule@def
1642                      \mule@def{27}{{\FB@@fg}}%
1643                    \else
1644                      \ifdefined\DeclareInputText
1645                        \@tempcnta`#1\relax
1646                        \DeclareInputText{\the\@tempcnta}{\FB@@fg}%
1647                      \else
1648                        \FBWarning{Option `fg' requires package
1649                                   inputenc;\MessageBreak reported}%
1650                      \fi
1651                    \fi
1652                  \else
1653                    \DeclareUnicodeCharacter{00BB}{\FB@@fg}%
1654                  \fi
1655                \fi}%
```

```
1656                \fi
1657            }%
1658 }
```

\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: cur-
rently 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 al-
ready been processed* by babel at \begin{document} *before* \FBprocess@options.

```
1659 \newcommand*{\FBprocess@options}{%
```

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

```
1660    \@ifpackageloaded{enumitem}{%
1661      \ifFBStandardItemizeEnv
1662      \else
1663        \FBStandardItemizeEnvtrue
1664        \PackageInfo{french.ldf}%
1665           {Setting StandardItemizeEnv=true for\MessageBreak
1666            compatibility with enumitem package,\MessageBreak
1667            reported}%
1668      \fi
1669      \ifFBStandardEnumerateEnv
1670      \else
1671        \FBStandardEnumerateEnvtrue
1672        \PackageInfo{french.ldf}%
1673           {Setting StandardEnumerateEnv=true for\MessageBreak
1674            compatibility with enumitem package,\MessageBreak
1675            reported}%
1676      \fi}{}%
1677    \@ifpackageloaded{paralist}{%
1678      \ifFBStandardItemizeEnv
1679      \else
1680        \FBStandardItemizeEnvtrue
1681        \PackageInfo{french.ldf}%
1682           {Setting StandardItemizeEnv=true for\MessageBreak
1683            compatibility with paralist package,\MessageBreak
1684            reported}%
1685      \fi
1686      \ifFBStandardEnumerateEnv
1687      \else
1688        \FBStandardEnumerateEnvtrue
1689        \PackageInfo{french.ldf}%
1690           {Setting StandardEnumerateEnv=true for\MessageBreak
1691            compatibility with paralist package,\MessageBreak
1692            reported}%
1693      \fi}{}%
1694    \@ifpackageloaded{enumerate}{%
1695      \ifFBStandardEnumerateEnv
1696      \else
```

62

```
1697        \FBStandardEnumerateEnvtrue
1698        \PackageInfo{french.ldf}%
1699           {Setting StandardEnumerateEnv=true for\MessageBreak
1700            compatibility with enumerate package,\MessageBreak
1701            reported}%
1702     \fi}{}%
```

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

```
1703  \def\FB@ufl{\update@frenchlists}
1704  \ifFB@mainlanguage@FR
1705     \update@frenchlists
1706  \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.

```
1707  \ifFBAutoSpacePunctuation
1708     \autospace@beforeFDP
1709  \else
1710     \noautospace@beforeFDP
1711  \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.

```
1712  \ifFBOriginalTypewriter
1713  \else
1714     \let\ttfamilyORI\ttfamily
1715     \let\rmfamilyORI\rmfamily
1716     \let\sffamilyORI\sffamily
1717     \let\ttfamily\ttfamilyFB
1718     \let\rmfamily\rmfamilyFB
1719     \let\sffamily\sffamilyFB
1720  \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.

```
1721  \@ifpackageloaded{numprint}%
1722     {\ifnprt@autolanguage
1723        \providecommand*{\npstylefrench}{}%
1724        \ifFBThinSpaceInFrenchNumbers
1725           \renewcommand*{\FBthousandsep}{\,}%
1726        \fi
1727        \g@addto@macro\npstylefrench{\npthousandsep{\FBthousandsep}}%
1728     \fi
1729     }{}%
```

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.

```
1730    \ifFBFrenchSuperscripts
1731      \DeclareRobustCommand*{\up}{\@ifstar{\FB@up@fake}{\fup}}%
1732    \else
1733      \DeclareRobustCommand*{\up}{\@ifstar{\FB@up@fake}%
1734                                        {\textsuperscript}}%
1735    \fi
```

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

```
1736    \ifFBLowercaseSuperscripts
1737    \else
1738      \renewcommand*{\FB@lc}[1]{##1}%
1739    \fi
```

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

```
1740    \ifFBCustomiseFigTabCaptions
1741      \ifFB@koma
1742        \renewcommand*{\captionformat}{\CaptionSeparator}%
1743      \fi
1744      \@ifclassloaded{memoir}%
1745        {\captiondelim{\CaptionSeparator}}{}%
1746      \@ifclassloaded{beamer}%
1747        {\defbeamertemplate{caption label separator}{FBcustom}{%
1748            \CaptionSeparator}%
1749         \setbeamertemplate{caption label separator}[FBcustom]}{}%
1750    \else
```

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

```
1751      \ifFB@koma
1752        \renewcommand*{\captionformat}{{\autospace@beforeFDP : }}%
1753      \fi
1754      \@ifclassloaded{memoir}%
1755        {\captiondelim{{\autospace@beforeFDP : }}%
1756        }{}%
1757      \@ifclassloaded{beamer}%
1758        {\defbeamertemplate{caption label separator}{FBcolon}{%
1759            {\autospace@beforeFDP : }}%
1760         \setbeamertemplate{caption label separator}[FBcolon]%
1761        }{}%
1762    \fi
```

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

```
1763    \ifFBShowOptions
1764      \GenericWarning{* }{%
1765      ***** List of possible options for babel-french ****\MessageBreak
1766      [Default values between brackets when french is loaded *LAST*]%
1767      \MessageBreak
1768      ShowOptions=true [false]\MessageBreak
```

```
1769      StandardLayout=true [false]\MessageBreak
1770      GlobalLayoutFrench=false [true]\MessageBreak
1771      PartNameFull=false [true]\MessageBreak
1772      IndentFirst=false [true]\MessageBreak
1773      ReduceListSpacing=false [true]\MessageBreak
1774      StandardItemizeEnv=true [false]\MessageBreak
1775      StandardEnumerateEnv=true [false]\MessageBreak
1776      StandardItemLabels=true [false]\MessageBreak
1777      ItemLabels=\textemdash, \textbullet,
1778          \protect\ding{43},... [\textendash]\MessageBreak
1779      ItemLabeli=\textemdash, \textbullet,
1780          \protect\ding{43},... [\textendash]\MessageBreak
1781      ItemLabelii=\textemdash, \textbullet,
1782          \protect\ding{43},... [\textendash]\MessageBreak
1783      ItemLabeliii=\textemdash, \textbullet,
1784          \protect\ding{43},... [\textendash]\MessageBreak
1785      ItemLabeliv=\textemdash, \textbullet,
1786          \protect\ding{43},... [\textendash]\MessageBreak
1787      StandardLists=true [false]\MessageBreak
1788      ListOldLayout=true [false]\MessageBreak
1789      CompactItemize=false [true]\MessageBreak
1790      FrenchFootnotes=false [true]\MessageBreak
1791      AutoSpaceFootnotes=false [true]\MessageBreak
1792      AutoSpacePunctuation=false [true]\MessageBreak
1793      ThinColonSpace=true [false]\MessageBreak
1794      OriginalTypewriter=true [false]\MessageBreak
1795      UnicodeNoBreakSpaces=true [false]\MessageBreak
1796      og= <left quote character>, fg= <right quote character>%
1797      INGuillSpace=true [false]\MessageBreak
1798      EveryParGuill=open, close, none [open]\MessageBreak
1799      EveryLineGuill=open, close, none
1800                     [open in LuaTeX, none otherwise]\MessageBreak
1801      InnerGuillSingle=true [false]\MessageBreak
1802      ThinSpaceInFrenchNumbers=true [false]\MessageBreak
1803      SmallCapsFigTabCaptions=false [true]\MessageBreak
1804      CustomiseFigTabCaptions=false [true]\MessageBreak
1805      OldFigTabCaptions=true [false]\MessageBreak
1806      FrenchSuperscripts=false [true]\MessageBreak
1807      LowercaseSuperscripts=false [true]\MessageBreak
1808      SuppressWarning=true [false]\MessageBreak
1809      \MessageBreak
1810      ********************************************%
1811      \MessageBreak\protect\frenchsetup{ShowOptions}}}
1812   \fi
1813 }
```

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.

```
1814 \AtBeginDocument{%
```

```
1815    \providecommand*{\xspace}{\relax}%
```

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

```
1816    \ifdefined\pdfstringdefDisableCommands
1817      \pdfstringdefDisableCommands{%
1818        \let\up\relax
1819        \let\fup\relax
1820        \let\degre\textdegree
1821        \let\degres\textdegree
1822        \def\ieme{e\xspace}%
1823        \def\iemes{es\xspace}%
1824        \def\ier{er\xspace}%
1825        \def\iers{ers\xspace}%
1826        \def\iere{re\xspace}%
1827        \def\ieres{res\xspace}%
1828        \def\FrenchEnumerate#1{#1\degre\space}%
1829        \def\FrenchPopularEnumerate#1{#1\degre)\space}%
1830        \def\No{N\degre\space}%
1831        \def\no{n\degre\space}%
1832        \def\Nos{N\degre\space}%
1833        \def\nos{n\degre\space}%
1834        \def\FB@og{\guillemotleft\space}%
1835        \def\FB@fg{\space\guillemotright}%
1836        \def\at{@}%
1837        \def\circonflexe{\string^}%
1838        \def\tild{\string~}%
1839        \def\boi{\textbackslash}%
1840        \let\bsc\textsc
1841      }%
1842    \fi
```

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

```
1843    \FBprocess@options
```

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

```
1844    \ifFBucsNBSP
1845      \renewcommand*{\FBmedkern}{\char"202F\relax}%
1846      \renewcommand*{\FBthickkern}{\char"A0\relax}%
1847      \ifFBThinSpaceInFrenchNumbers
1848        \renewcommand*{\FBthousandsep}{\char"202F\relax}%
1849      \else
1850        \renewcommand*{\FBthousandsep}{\char"A0\relax}%
1851      \fi
1852    \fi
```

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

```
1853    \begingroup
1854      \newcommand{\FBOTone}{OT1}%
```

```
1855        \ifx\encodingdefault\FBOTone
1856          \FBWarning{OT1 encoding should not be used for French.%
1857                     \MessageBreak
1858                     Add \protect\usepackage[T1]{fontenc} to the
1859                     preamble\MessageBreak of your document; reported}%
1860        \fi
1861      \endgroup
1862 }
```

2.12 French lists

\listFB Vertical spacing in lists should be shorter in French texts than the defaults provided by
\listORI LaTeX. Note that the easy way, just changing values of vertical spacing parameters
\FB@listVsettings when entering French and restoring them to their defaults on exit would not work; so
we define the command \FB@listVsettings to hold the settings to be used by the
French variant \listFB of \list. Note that switching to \listFB reduces vertical
spacing in *all* environments built on \list: itemize, enumerate, description, but
also abstract, quotation, quote and verse. . .
The amount of vertical space before and after a list is given by \topsep + \parskip
(+ \partopsep if the list starts a new paragraph). IMHO, \parskip should be added
only when the list starts a new paragraph, so I subtract \parskip from \topsep and
add it back to \partopsep; this will normally make no difference because \parskip's
default value is 0pt, but will be noticeable when \parskip is *not* null.

```
1863 \let\listORI\list
1864 \let\endlistORI\endlist
1865 \def\FB@listVsettings{%
1866      \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1867      \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
1868      \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1869      \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.

```
1870      \@tempdima=\parskip
1871      \addtolength{\topsep}{-\@tempdima}%
1872      \addtolength{\partopsep}{\@tempdima}%
1873 }
1874 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1875 \let\endlistFB\endlist
```

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

- The '•' is never used in French itemize-lists, an emdash '—' or an en-
  dash '–' is preferred for all levels. The item label to be used in French is
  stored in \FrenchLabelItem}, it defaults to '—' and can be changed using
  \frenchsetup{} (see section 2.11).

- Vertical spacing between items, before and after the list, should be *null* with *no glue* added;

- In French the labels of itemize-lists are vertically aligned as follows:

> Text starting at 'parindent'
> <= Leftmargin
>   — first item. . .
>   — first second level item
>   — next one. . .
>   — second item. . .

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

\Frlabelitemi
\Frlabelitemii
\Frlabelitemiii
\Frlabelitemiv

```
1876 \newcommand*{\FrenchLabelItem}{\textemdash}
1877 \newcommand*{\Frlabelitemi}{\FrenchLabelItem}
1878 \newcommand*{\Frlabelitemii}{\FrenchLabelItem}
1879 \newcommand*{\Frlabelitemiii}{\FrenchLabelItem}
1880 \newcommand*{\Frlabelitemiv}{\FrenchLabelItem}
```

\listindentFB Let's define three lengths \listindentFB, \descindentFB and \labelwidthFB to
\descindentFB customise lists' horizontal indentations. They are given silly negative values here
\labelwidthFB in order to eventually enable their customisation in the preamble. They will get
reasonnable defaults later when entering French (see \bbl@frenchlabelitems)
unless they have been customised.

```
1881 \newlength\listindentFB
1882 \setlength{\listindentFB}{-1pt}
1883 \newlength\descindentFB
1884 \setlength{\descindentFB}{-1pt}
1885 \newlength\labelwidthFB
1886 \setlength{\labelwidthFB}{-1pt}
```

\FB@listHsettings \FB@listHsettings holds the new horizontal settings chosen for French lists itemize
\leftmarginFB and enumerate starting with version 2.6a. They are based on the look resquested in
French for itemize-lists.

```
1887 \newlength\leftmarginFB
1888 \def\FB@listHsettings{%
1889    \leftmarginFB\labelwidthFB
1890    \advance\leftmarginFB \labelsep
1891    \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1892      {\csname leftmargin\romannumeral\FB@dp\endcsname \leftmarginFB}%
1893    \advance\leftmargini \listindentFB
1894    \leftmargin\csname leftmargin\ifnum\@listdepth=\@ne i\else
1895                                          ii\fi\endcsname
1896 }
```

\itemizeFB New environment for French itemize-lists.
\FB@itemizesettings \FB@itemizesettings does two things: first suppress all vertical spaces including
glue when option ReduceListSpacing is set, then set horizontal indentations accord-
ing to \FB@listHsettings unless option ListOldLayout is true (compatibility with
lists up to v. 2.5k).

68

```
1897 \def\FB@itemizesettings{%
1898     \ifFBReduceListSpacing
1899       \setlength{\itemsep}{\z@}%
1900       \setlength{\parsep}{\z@}%
1901       \setlength{\topsep}{\z@}%
1902       \setlength{\partopsep}{\z@}%
1903       \@tempdima=\parskip
1904       \addtolength{\topsep}{-\@tempdima}%
1905       \addtolength{\partopsep}{\@tempdima}%
1906     \fi
1907     \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
1908     \ifFBListOldLayout
1909       \setlength{\leftmargin}{\labelwidth}%
1910       \addtolength{\leftmargin}{\labelsep}%
1911       \addtolength{\leftmargin}{\parindent}%
1912     \else
1913       \FB@listHsettings
1914     \fi
1915 }
```

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

```
1916 \def\itemizeFB{%
1917     \ifnum \@itemdepth >\thr@@\@toodeep\else
1918       \advance\@itemdepth\@ne
1919       \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
1920       \expandafter
1921       \listORI
1922       \csname\@itemitem\endcsname
1923       \FB@itemizesettings
1924     \fi
1925 }
1926 \let\enditemizeFB\endlistORI

1927 \def\labelitemsFB{%
1928     \let\labelitemi\Frlabelitemi
1929     \let\labelitemii\Frlabelitemii
1930     \let\labelitemiii\Frlabelitemiii
1931     \let\labelitemiv\Frlabelitemiv
1932     \ifdim\labelwidthFB<\z@
1933       \settowidth{\labelwidthFB}{\FrenchLabelItem}%
1934     \fi
1935     \ifdim\listindentFB<\z@
1936       \ifdim\parindent=\z@
1937         \setlength{\listindentFB}{1.5em}%
1938       \else
1939         \setlength{\listindentFB}{\parindent}%
1940       \fi
1941     \fi
1942     \ifdim\descindentFB<\z@
1943       \setlength{\descindentFB}{\listindentFB}%
1944     \fi
```

```
1945 }
```

\enumerateFB The definition of \enumerateFB, new to version 2.6a, follows the one of \enumerate in standard LaTeX classes (see ltlists.dtx), vertical spaces are customised (or not) via \list (=\listFB or \listORI) and horizontal spaces (leftmargins) are borrowed from itemize lists via \FB@listHsettings.

```
1946 \def\enumerateFB{%
1947   \ifnum \@enumdepth >\thr@@\@toodeep\else
1948     \advance\@enumdepth\@ne
1949     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
1950     \expandafter
1951     \list
1952       \csname label\@enumctr\endcsname
1953       {\FB@listHsettings
1954        \usecounter\@enumctr\def\makelabel##1{\hss\llap{##1}}}%
1955   \fi
1956 }
1957 \let\endenumerateFB\endlistORI
```

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

```
1958 \def\descriptionFB{%
1959       \list{}{\FB@listHsettings
1960             \labelwidth\z@
1961             \itemindent-\leftmargin
1962             \ifnum\@listdepth=1
1963               \ifdim\descindentFB=\z@
1964                 \ifdim\listindentFB>\z@
1965                   \leftmargini\listindentFB
1966                   \leftmargin\leftmargini
1967                   \itemindent-\leftmargin
1968                 \fi
1969               \else
1970                 \advance\itemindent by \descindentFB
1971               \fi
1972             \fi
1973             \let\makelabel\descriptionlabel}%
1974 }
1975 \let\enddescriptionFB\endlistORI
```

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

```
1976 \def\update@frenchlists{%
1977   \ifFBReduceListSpacing \let\list\listFB \fi
1978   \ifFBStandardItemizeEnv
1979   \else \let\itemize\itemizeFB \fi
1980   \ifFBStandardItemLabels
```

```
1981   \else \labelitemsFB \fi
1982   \ifFBStandardEnumerateEnv
1983   \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
1984 }
```

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. .

```
1985 \def\FB@ufl{\relax}
1986 \def\bbl@frenchlistlayout{%
1987   \ifFBGlobalLayoutFrench
1988   \else
1989     \babel@save\list        \babel@save\itemize
1990     \babel@save\enumerate   \babel@save\description
1991     \babel@save\labelitemi  \babel@save\labelitemii
1992     \babel@save\labelitemiii \babel@save\labelitemiv
1993     \FB@ufl
1994   \fi
1995 }
1996 \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
\bbl@nonfrenchindent   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.

```
1997 \def\bbl@frenchindent{%
1998   \ifFBGlobalLayoutFrench
1999   \else
2000     \babel@save\@afterindentfalse
2001   \fi
2002   \ifFBIndentFirst
2003     \let\@afterindentfalse\@afterindenttrue
2004     \@afterindenttrue
2005   \fi}
2006 \def\bbl@nonfrenchindent{%
2007   \ifFBGlobalLayoutFrench
2008     \ifFBIndentFirst
2009       \@afterindenttrue
2010     \fi
2011   \fi}
2012 \addto\extrasfrench{\bbl@frenchindent}
2013 \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.

```
2014 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2015                 {\PackageInfo{french.ldf}%
2016                   {bigfoot package in use.\MessageBreak
2017                    babel-french will NOT customise footnotes;%
2018                    \MessageBreak reported}}%
2019                 {\let\@footnotemarkORI\@footnotemark
2020                  \def\@footnotemarkFB{\leavevmode\unskip\unkern
2021                                  \,\@footnotemarkORI}%
2022                  \ifFBAutoSpaceFootnotes
2023                    \let\@footnotemark\@footnotemarkFB
2024                  \fi}%
2025               }
```

\@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).

```
2026 \newdimen\parindentFFN
2027 \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.

```
2028 \newcommand*{\dotFFN}{.}
2029 \newcommand*{\kernFFN}{\kern .5em}
2030 \newlength\FBfnindent
```

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

72

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

2031 \ifFB@koma
2032   \let\@makefntextORI\@makefntext
2033   \let\@@makefnmarkORI\@@makefnmark

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

2034   \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
2035   {\thefootnotemark\dotFFN\kernFFN}
2036   \let\@makefntextFB\@makefntext
2037   \let\@@makefnmarkFB\@@makefnmark

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

2038   \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2039   {^{\thefootnotemark}}
2040   \let\@makefntextTH\@makefntext
2041   \let\@@makefnmarkTH\@@makefnmark

Restore the original definitions.

2042   \let\@makefntext\@makefntextORI
2043   \let\@@makefnmark\@@makefnmarkORI
2044 \fi

Definitions for the memoir class:

2045 \@ifclassloaded{memoir}

(see original definition in memman.pdf)

2046   {\newcommand{\@makefntextFB}[1]{%
2047   \def\footscript##1{##1\dotFFN\kernFFN}%
2048   \setlength{\footmarkwidth}{\FBfnindent}%
2049   \setlength{\footmarksep}{-\footmarkwidth}%
2050   \setlength{\footparindent}{\parindentFFN}%
2051   \makefootmark #1}%
2052   }{}

Definitions for the beamer class:

2053 \@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.

2054   {\def\@makefntextFB#1{%
2055   \def\insertfootnotetext{#1}%
2056   \def\insertfootnotemark{\insertfootnotemarkFB}%
2057   \usebeamertemplate***{footnote}}%
2058   \def\insertfootnotemarkFB{%
2059   \usebeamercolor[fg]{footnote mark}%
2060   \usebeamerfont*{footnote mark}%
2061   \llap{\@thefnmark}\dotFFN\kernFFN}%
2062   }{}

73

Now the default definition of \@makefntextFB for standard LaTeX and AMS classes. The next command prints the footnote mark according to the specifications of the French 'Imprimerie Nationale'. Keep in mind that \@thefnmark might be empty (i.e. in AMS classes' titles)!

```
2063 \providecommand*{\insertfootnotemarkFB}{%
2064   \parindent=\parindentFFN
2065   \rule\z@\footnotesep
2066   \setbox\@tempboxa\hbox{\@thefnmark}%
2067   \ifdim\wd\@tempboxa>\z@
2068     \llap{\@thefnmark}\dotFFN\kernFFN
2069   \fi}
2070 \providecommand\@makefntextFB[1]{\insertfootnotemarkFB #1}
```

The rest of \@makefntext's customisation is done at the \begin{document}. We save the original definition of \@makefntext, and then redefine \@makefntext according to the value of flag \ifFBFrenchFootnotes (true or false). Koma-script classes require a special treatment.

The LuaTeX command \localleftbox used by \frquote{} has to be reset inside footnotes, done for LaTeX based formats only.

```
2071 \providecommand\localleftbox[1]{}
2072 \AtBeginDocument{%
2073   \@ifpackageloaded{bigfoot}{}%
2074     {\ifdim\parindentFFN<10in
2075      \else
2076       \parindentFFN=\parindent
2077       \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2078      \fi
2079     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2080     \addtolength{\FBfnindent}{\parindentFFN}%
2081     \let\@makefntextORI\@makefntext
2082     \ifFB@koma
```

Definition of \@makefntext for koma-script classes: running makefntextORI inside a group to reset \localleftbox{} would mess up the layout of footnotes whenever the first manadatory argument of \deffootnote{} (used as \leftskip) is non-nil (default is 1em, 0pt in French).

```
2083        \let\@@makefnmarkORI\@@makefnmark
2084        \long\def\@makefntext#1{%
2085          \ifFBFrenchFootnotes
2086            \ifx\footnote\thanks
2087              \let\@@makefnmark\@@makefnmarkTH
2088              \begingroup\localleftbox{}\@makefntextTH{#1}\endgroup
2089            \else
2090              \let\@@makefnmark\@@makefnmarkFB
2091              \begingroup\localleftbox{}\@makefntextFB{#1}\endgroup
2092            \fi
2093          \else
2094            \let\@@makefnmark\@@makefnmarkORI
2095            \@makefntextORI{#1}%
2096          \fi}%
```

```
2097            \else
```

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

```
2098            \@ifclassloaded{memoir}%
2099              {\ifFBFrenchFootnotes
2100                \setlength{\thanksmarkwidth}{\parindentFFN}%
2101                \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2102              \fi
2103              }{}%
```

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

```
2104            \@ifclassloaded{beamer}%
2105              {\ifFBFrenchFootnotes
2106                \ifdim\parindentFFN=1.5em\else
2107                  \FBWarning{%
2108                    \protect\parindentFFN\space is ineffective%
2109                    \MessageBreak within the beamer class.%
2110                    \MessageBreak Reported}%
2111                \fi
2112              \fi
2113              }{}%
```

Definition of \@makefntext for all classes other than koma-script:

```
2114            \long\def\@makefntext#1{\begingroup\localleftbox{}%
2115              \ifFBFrenchFootnotes
2116                \@makefntextFB{#1}%
2117              \else
2118                \@makefntextORI{#1}%
2119              \fi\endgroup}%
2120          \fi
2121      }%
2122 }
```

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

```
2123 \newcommand*{\AddThinSpaceBeforeFootnotes}{\FBAutoSpaceFootnotestrue}
2124 \newcommand*{\FrenchFootnotes}{\FBFrenchFootnotestrue}
2125 \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.

```
2126 \FBclean@on@exit
2127 \ldf@finish\CurrentOption
2128 \let\loadlocalcfg\FB@llc
2129 </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 them-
selves 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.

```
2130 <*acadian>
2131 \PackageInfo{acadian.ldf}%
2132   {'acadian' dialect is currently\MessageBreak
2133    *absolutely identical* to the\MessageBreak
2134    'french' language; reported}
2135 </acadian>
2136 <*canadien>
2137 \PackageWarning{canadien.ldf}%
2138   {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2139    it might be removed sooner or later.  Please\MessageBreak
2140    use 'acadian' instead; reported}%
2141 \let\l@canadien\l@acadian
2142 \def\CurrentOption{acadian}
2143 </canadien>
2144 <*francais>
2145 \PackageWarning{francais.ldf}%
2146   {Option 'francais' for Babel is *deprecated*,\MessageBreak
2147    it might be removed sooner or later.  Please\MessageBreak
2148    use 'french' instead; reported}%
2149 \let\l@francais\l@french
2150 \def\CurrentOption{french}
2151 </francais>
```

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

```
2152 <*frenchb>
2153 \def\bbl@tempa{frenchb}
2154 \ifx\CurrentOption\bbl@tempa
2155   \let\l@frenchb\l@french
2156   \def\CurrentOption{french}
2157   \PackageWarning{babel-french}%
2158     {Option 'frenchb' for Babel is *deprecated*,\MessageBreak
2159      it might be removed sooner or later.  Please\MessageBreak
2160      use 'french' instead; reported}
2161 \else
2162   \def\bbl@tempa{francais}
2163   \ifx\CurrentOption\bbl@tempa
```

```
2164      \let\l@francais\l@french
2165      \def\CurrentOption{french}
```

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

```
2166      \ifx\magnification\@undefined
2167        \PackageWarning{babel-french}%
2168          {Option 'francais' for Babel is *deprecated*,\MessageBreak
2169           it might be removed sooner or later.  Please\MessageBreak
2170           use 'french' instead; reported}%
2171      \fi
2172    \else
2173      \def\bbl@tempa{canadien}
2174      \ifx\CurrentOption\bbl@tempa
2175        \let\l@canadien\l@acadian
2176        \def\CurrentOption{acadian}
2177        \PackageWarning{babel-french}%
2178          {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2179           it might be removed sooner or later.  Please\MessageBreak
2180           use 'acadian' instead; reported}
2181      \fi
2182    \fi
2183  \fi
2184  </frenchb>
2185  <acadian|canadien|frenchb|francais>\input french.ldf\relax
2186  <acadian|canadien>\let\extrasacadian\extrasfrench
2187  <acadian|canadien>\let\noextrasacadian\noextrasfrench
```

3 Change History

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

v3.4d

> \frenchsetup: New test for deciding about utf8 encoding for keys og and fg (the former one fails with LaTeX 2018 release). 59

v3.4c

> \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
>
> New attribute \FB@dialect for the French dialect acadian. 20
>
> New command \FBsetspaces to fine tune spacing independently in French and in French dialects. . . 18
>
> Shrink/stretch removed in \FBthousandsep. 47
>
> Toks \FBcolonsp, \FBthinsp and \FBguillsp removed. 18
>
> frenchb.lua: Global 'FBsp' table added; local function 'get_glue' changed into global 'FBget_glue'. 23
>
> \datefrench: Specific code for Plain finally removed (babel bug reported). 40
>
> \extrasfrench: Change \(no)extras\CurrentOption to \(no)extrasfrench. \(no)extrasacadian will be defined as \(no)extrasfrench in file acadian.ldf. 16

> \frenchsetup: Patch for koma-script classes moved here, after \ifFBPartNameFull is defined, so that it applies to \extrasacadian too: \AtEndOfPackage is too late. 54

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

v3.3c

> General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, fontspec is no longer required. . . 66
>
> New command \FBthousandsep to customise numprint. 47
>
> New configurable kerns \FBmedkern, and \FBthickkern suitable for HTML translation. . . 43
>
> Reorganise warnings when the caption, subcaption or floatrow packages are loaded before babel/french. 50
>
> Reset \localleftbox locally inside \@makefntext. Needed by \frquote with LuaTeX. 74
>
> frenchb.lua: Function 'get_glue' robustified. 'french_punctuation' can insert Unicode characters instead of glues. 22
>
> \frenchsetup: New option 'UnicodeNoBreakSpaces' for html translators (LuaLaTeX only). 59

v3.3b

> General: Generate portmanteau files acadian.ldf, canadien.ldf, frenchb.ldf, and francais.ldf and warn about deprecated options. 76
>
> New 'if' \ifFBfrench to replace \iflanguage test which is based on patterns. 16

v3.3a

> General: Compatibility code for pre 2015/10/01 LaTeX release removed, see ltnews23.tex. 20

81