A Babel language definition file for French
frenchb.dtx v3.5q, 2023/03/08

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, Ulrike Fisher and Marcel Krüger. Thanks to all of them!

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

Changes between version 3.0 and v3.5q are listed in subsection 1.4 p. 11.

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

1.1 Basic interface

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

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

`\usepackage[german,spanish,french,british]{babel}` [2]

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

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

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

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

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

3. vertical spacing in general LaTeX lists is shortened;

4. footnotes are displayed "à la française''.

---

[1]The file described in this section has version number v3.5q and was last revised on 2023/03/08.

[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 1.2.3 p. 9.

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 1.2.2 p. 9.

5. the space after \dots is removed in French.

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

1. French quotation marks can be entered using the command \frquote{}: \frquote{some text} will output « some text ». Former commands \og and \fg are kept for backward compatibility: \og some text\fg{} is an alternative to \frquote{some text}.

   If French quote characters are available on your keyboard, you can use them, to get proper spacing in LaTeX2e see option og=«, fg=» p. 8.

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

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

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

   • 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.

---

[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 p. 7.

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. Command \bname{} (boxed name) is provided to typeset family names: its argument will not be hyphenated except on explicit hyphens. \bsc{} (boxed small caps) is a variant that prints its argument in small capitals, it is meant for bibliographies, signatures, etc. Usage: Albert~\bsc{Camus}.

5. Commands \primo, \secundo, \tertio and \quarto print 1°, 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) or for angles in math mode.

8. In math mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the TEXbook p. 134). The command \DecimalMathComma makes the comma behave as an ordinary character *when the current language is French* (no space added); as a counterpart, if \DecimalMathComma is active, an explicit thin space has to be added in lists and intervals: $(x,\,y)$, $[0,\,1]$. \StandardMathComma switches back to the standard behaviour of the comma in French.

   The icomma package is an alternative workaround.

9. A command \nombre was provided in 1.x versions to easily format numbers in slices of three digits separated either by a comma in English or with a space in French; \nombre is now mapped to \numprint from numprint.sty, which should be loaded *after* Babel, see numprint.pdf for more information.

10. babel-french has been designed to take advantage of the xspace package if present: adding \usepackage{xspace} in the preamble will force macros like \fg, \ier, \ieme, \dots, …, to respect the spaces you type after them, for instance typing '1\ier juin' will print '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}

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

The other options are listed below. Their default value is shown between braces, sometimes followed 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; it useless unless French is the main language. This option can be used to avoid conflicts with classes or packages which customise lists or footnotes.

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

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

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

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

```
┌──────────────────────────────────┐   ┌──────────────────────────────────┐
│   Text starting at 'parindent'   │   │   Text starting at 'parindent'   │
│ <= Leftmargin                    │   │ <= Leftmargin                    │
│     — first item running on two  │   │     — first item running on two  │
│       lines or more…             │   │ lines or more…                   │
│         — first second level item│   │         — first second level item│
│           on two lines…          │   │   on two lines…                  │
│         — next one…              │   │         — next one…              │
│     — second item…               │   │     — second item…               │
└──────────────────────────────────┘   └──────────────────────────────────┘
        Default French layout                With ListItemsAsPar=true
```

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

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

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

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

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

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

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

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

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

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

---

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

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

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

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

AutoSpacePunctuation=false (true); in French, the user *should* input a space before the four characters ':;!?' but as many people forget about it (even among native French writers!), the default behaviour of babel-french is to automatically typeset non-breaking spaces the width of which is either \FBthinspace (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 [7], 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} [8] or {\NoAutoSpacing ???} (needed for pdfTeX only).

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

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

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

---

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

[8]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 \frquote{}. This option tells babel-french which characters are opening and closing French guillemets (they depend on the input encoding), then you can type either « guillemets » or «guillemets» [9] (with or without spaces) to get properly typeset French quotes. This option works with LuaLaTeX, XeLaTeX and with pdfLaTeX (default encoding: utf8); with pdflatex other 8-bits encodings (latin1, latin9, ansinew, applemac,…) are also supported when properly declared with inputenc.

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

EveryParGuill=open, close, none (open); sets whether an opening quote («) or a closing one (») or nothing should be printed by \frquote{} at the beginning of every 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{} com-mand, \npthousandsep is defined as a non-breaking space (~) [10] in French; when set to true, this option redefines \npthousandsep as a thin space (\,).

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

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

---

[9]Or even «~guillemets~», but *only* with LuaLaTeX.
[10]Actually without stretch nor shrink.

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

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

`LowercaseSuperscripts=false (true)`; by default babel-french inhibits the 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 koma-script and beamer classes. In case this procedure fails a warning is issued.
When French is not the main language, the colon is preserved for all languages including French but babel-french tries hard to insert a proper space before it and warns if it fails to do so.
Three options are provided to customise figure and table captions:

- `CustomiseFigTabCaptions` is set to `true` when French is the main language (hence separator = ' – ') and to `false` otherwise (hence separator = ': ' with a proper space before the colon in French if possible); toogle this option if needed;

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

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

1.3   Hyphenation checks

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

- run `pdfLaTeX` on the following file:

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

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

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

- you get `sig-nal con-tainer`, this probably means that the hyphenation patterns you are using are for US-English, not for French;

- you get no hyphen at all in `évé-ne-ment`, this probably means that you are using CM fonts and the macro \accent to produce accented characters. Using 8-bits fonts with built-in accented characters avoids this kind of mismatch.

1.4 Changes

What's new in version 3.5?

Version 3.5a offers a new option `ListItemsAsPar`. The default layout of lists is unchanged (for backward compatibility), but users should try this new option which ensures a layout of lists closer to French typographic standards: see f.i. how lists are typeset in the book "Lexique des règles typographiques en usage à l'Imprimerie Nationale''.
Version 3.5b fixes a bug due to wrong \everypar's management in \frquote{}; it showed up when \frquote{} immediately followed a sectionning command.
Starting with version 3.5d, a new option `StandardListSpacing` has been added to supersede `ReduceListSpacing`.
A new command \NoEveryParQuote has been added in version 3.5e: it is meant to be used inside a group or environment to suppress unwanted guillemets (typically when lists are embedded in \frquote{}).
Version 3.5g fixes a long standing bug affecting LuaTeX: legacy kerning was disabled for Type1 fonts since v3.1g (2015).
Version 3.5j also fixes a long standing bug affecting koma-script, memoir et beamer classes: redefintions of the caption separator (commands \captionformat, \captiondelim, etc.) are now taken into account properly.
Version 3.5k is a cleanup release:

- the translations in French of \figurename and \tablename no longer hold font changing commands (switch to small caps), the font switch has been moved to \fnum@figure and \fnum@table as suggested by Axel Sommerfeldt.

- Package caption can now be loaded whether before or after babel, indifferently.

- \pdfstringdefDisableCommands is no longer used: as suggested by the La-TeX3 team, all commands requiring special care in hyperref's bookmarks are now defined using \textorpdfstring{}{}.

Version 3.5n introduces a new command \bname{} (an alternative to \bsc{}).
Version 3.5q corrects a bug in lists layout: \listparindent (formely 0pt) is defined as \parindent and if \parskip > 0pt, \parsep is now defined as \parskip. This ensures that paragraphs included in lists are now visible. The former behaviour can be recovered by adding \parskip=0pt, \parindent=0pt *inside* the list environment.

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 komascript classes. The layout of footnotes "à la française'' should be unchanged but footnotes' customisations offered by these classes (i.e. font or color changes) are now available even when option FrenchFootnotes is true.

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

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

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

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

What's new in version 3.1?

New command \frquote{} meant to enter French quotations, especially long ones (spreading over several paragraphs) and/or embedded ones. see p. 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' [11]. Functionalities and user interface are unchanged.

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

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

---

[11]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
58   \adddialect\l@canadien\l@acadian
59 \else
60   \adddialect\l@acadian\l@french
61   \adddialect\l@canadien\l@french
62 \fi
```

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

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

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

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

\ifFBunicode French hyphenation patterns are now coded in Unicode, see file hyph-fr.tex. XeTeX
\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.

```
80 \newif\ifFBunicode
81 \newif\ifFBLuaTeX
82 \newif\ifFBXeTeX
83 \begingroup\expandafter\expandafter\expandafter\endgroup
84 \expandafter\ifx\csname luatexversion\endcsname\relax
85 \else
86   \FBunicodetrue \FBLuaTeXtrue
87 \fi
88 \begingroup\expandafter\expandafter\expandafter\endgroup
89 \expandafter\ifx\csname XeTeXrevision\endcsname\relax
90 \else
91   \FBunicodetrue \FBXeTeXtrue
92 \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=»}.

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

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

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

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

2.2 Punctuation

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

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

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

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

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

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

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

\FBguillspace These three commands are meant for basic French. Other French dialects can use
\FBcolonspace different settings, see below. According to the I.N. specifications, the ':' requires
\FBthinspace an inter-word space before it, the other three require just a thin space. We define
\FBcolonspace as \space (inter-word space) and \FBthinspace as an half inter-word

space with no shrink nor stretch. \FBguillspace is defined btw. as spacing for French quotes is handled together with high punctuation for LuaTeX and XeTeX. \FBguillspace has been fine tuned by Thierry Bouche to 80% of an inter-word space with reduced stretchability. All three are user customisable in the preamble, best using the \FBsetspaces command described below. A penalty will be added before these spaces to prevent line breaking.

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

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

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

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

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

Remember that the *same* \extrasfrench command is executed when switching to French or to a French dialect (Acadian). Acadian and French may share the same patterns (or not), and may use different spacing for high punctuation and/or quotes. Basically, for pdfLaTeX and XeLaTeX, the spacing is set for French, then potentially tuned differently for Acadian. LuaTeX relies on an attribute \FB@dialect to decide what spacing is needed for French or Acadian (see LuaTeX table FBsp). As a rough test on \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.

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

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

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

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

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

Same for \FBcolonspace:

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

And for \FBguillspace:

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

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

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

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

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

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

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

```
208     \input ltluatex.tex
209   \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.

```
210   \newattribute\FB@spacing        \FB@spacing=\@ne
211   \newattribute\FB@addDPspace     \FB@addDPspace=\@ne
212   \newattribute\FB@addGUILspace \FB@addGUILspace=\z@
213   \newattribute\FB@ucsNBSP        \FB@ucsNBSP=\z@
214   \newattribute\FB@dialect        \FB@dialect=\z@
215   \ifLaTeXe
216     \PackageInfo{french.ldf}{No need for active punctuation
217                 characters\MessageBreak with this version
218                 of LuaTeX!\MessageBreak reported}
```

20

```
219  \else
220    \fb@info{No need for active punctuation characters\\
221           with this version of LuaTeX!}
222  \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.

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

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

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

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

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

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

```
279    [0xA0]             = true,
280    [0x202F]           = true}
281 local FB_guil_null =
282   {[0xA0]             = true,
283    [0x202F]           = true}
```

Local definitions for nodes:

```
284 local new_node        = node.new
285 local copy_node       = node.copy
286 local node_id         = node.id
287 local HLIST           = node_id("hlist")
288 local TEMP            = node_id("temp")
289 local KERN            = node_id("kern")
290 local GLUE            = node_id("glue")
291 local GLYPH           = node_id("glyph")
292 local PENALTY         = node_id("penalty")
293 local nobreak         = new_node(PENALTY)
294 nobreak.penalty       = 10000
295 local nbspace         = new_node(GLYPH)
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 'AtBe-ginDocument' 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 -- Main function (to be added to the kerning callback).
370 local function french_punctuation (head)
```

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

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

Skip glyphs not concerned by French kernings.

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

```
398          local is_glue = prev_id == GLUE
399          local glue_wd
400          if is_glue then
401            glue_wd = prev.width
402          end
403          local realglue = is_glue and glue_wd > 1
```

For characters for which FB_punct_thin or FB_punct_thick is *true*, the amount of spacing to be typeset before them is controlled by commands \FBthinspace and \FBcolonspace respectively. Two options: if a space has been typed in before (turned into *glue* in the node list), we remove the *glue* and add a nobreak penalty and the required *glue*. Otherwise (auto option), the penalty and the required *glue* are inserted if attribute \FB@addDPspace is set, unless any of these four conditions is met: a) node is ':' and the next one is of type GLYPH (avoids spurious spaces in http://mysite, C:\ or 10:35); b) the previous character is part of type FB_punct_null (avoids spurious spaces in strings like (!) or ??); c) a null glue (actually <= 1 sp for tabulars, possibly < 0) 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.

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

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

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

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

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

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

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

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

```
500                    next_subtype = next.subtype
```

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

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

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

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

The addgl flag only depends on next_char and is_glue:

```
547              local addgl = (next_char and not FB_guil_null[next_char])
548                           or (next and not next_char)
```

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

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

As a language tag is part of glyph nodes in LuaTeX, no more switching has to be done in \extrasfrench, setting the dialect attribute has already be done (see above, p. 19).

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

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

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

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

Six new character classes are defined for babel-french.

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

30

```
618    \def\FBsavevariable@loop#1#2{\begingroup
619      \toks@\expandafter{\originalTeX #1}%
620      \edef\x{\endgroup
621        \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}%
622      \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
)	]	{	}	,	-	.	”	%	'	'	'

```
623    \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F,%
624                     "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.

```
625      \newcommand*{\FB@xetex@punct@french}{%
626        \babel@savevariable{\XeTeXinterchartokenstate}%
627        \bbl@for\FB@char\FB@charlist
628            {\FBsavevariable@loop{\XeTeXcharclass}{\FB@char}}%
```

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).

```
629        \XeTeXinterchartokenstate=\@ne
630        \XeTeXcharclass `\: = \FB@punctthick
631        \XeTeXinterchartoks \z@ \FB@punctthick = {%
632            \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
633        \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
634            \ifFB@spacing\FDP@colonspace\fi}%
```

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

```
635        \XeTeXinterchartoks \FB@nonchar \FB@punctthick = {%
636            \ifFB@spacing
637              \ifhmode
638                \ifdim\lastskip>1sp
639                  \unskip\penalty\@M\FBcolonspace
640                \else
641                  \FDP@colonspace
642                \fi
643              \fi
644            \fi}%
645      \bbl@for\FB@char
```

```
646            {`\;,`\!,`\?}%
647            {\XeTeXcharclass\FB@char=\FB@punctthin}%
648      \XeTeXinterchartoks \z@ \FB@punctthin = {%
649          \ifFB@spacing\ifhmode\FDP@thinspace\fi\fi}%
650      \XeTeXinterchartoks \FB@guilf \FB@punctthin = {%
651          \ifFB@spacing\FDP@thinspace\fi}%
652      \XeTeXinterchartoks \FB@nonchar \FB@punctthin = {%
653          \ifFB@spacing
654            \ifhmode
655              \ifdim\lastskip>1sp
656                \unskip\penalty\@M\FBthinspace
657              \else
658                \FDP@thinspace
659              \fi
660            \fi
661          \fi}%
662      \XeTeXinterchartoks \FB@guilo \z@ = {%
663          \ifFB@spacing\FB@guillspace\fi}%
664      \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
665          \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
666      \XeTeXinterchartoks \z@ \FB@guilf = {%
667          \ifFB@spacing\FB@guillspace\fi}%
668      \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
669          \ifFB@spacing\FB@guillspace\fi}%
670      \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
671          \ifFB@spacing\unskip\FB@guillspace\fi}%
```

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

```
672      \bbl@for\FB@char
673            {`\[,`\(,"A0,"202F}%
674            {\XeTeXcharclass\FB@char=\FB@punctnul}%
```

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

```
675      \bbl@for\FB@char
676            {`\{,`\,,`\.,`\-,`\),`\],`\},`\%,"22,"27,"60,"2019}%
677            {\XeTeXcharclass\FB@char=\z@}%
678      }
679      \addto\extrasfrench{\FB@xetex@punct@french}
```

End of specific code for punctuation with modern XeTeX engines.

```
680 \fi
```

2.2.3 Punctuation with standard (pdf)TeX

In standard (pdf)TeX we need to make the four characters ; ! ? and : 'active' and provide their definitions. Before doing so, we have to save some definitions involving :.

```
681 \newif\ifFB@koma
682 \ifLaTeXe
683   \@ifclassloaded{scrartcl}{\FB@komatrue}{}
684   \@ifclassloaded{scrbook}{\FB@komatrue}{}
```

```
685  \@ifclassloaded{scrreprt}{\FB@komatrue}{}
686  \ifFB@koma\def\FB@std@capsep{:\ }\fi
687  \@ifclassloaded{beamer}{\def\FB@std@capsep{:\ }}{}
688  \@ifclassloaded{memoir}{\def\FB@std@capsep{: }}{}
689 \fi

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

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

Now we can insert a ; character.

```
705    \string;}
```

The next three definitions are very similar.

```
706  \declare@shorthand{french}{!}{%
707    \ifFB@spacing
708      \ifhmode
709        \ifdim\lastskip>1sp
710          \unskip\penalty\@M\FBthinspace
711        \else
712          \FDP@thinspace
713        \fi
714      \fi
715    \fi
716    \string!}
717  \declare@shorthand{french}{?}{%
718    \ifFB@spacing
719      \ifhmode
720        \ifdim\lastskip>1sp
721          \unskip\penalty\@M\FBthinspace
722        \else
723          \FDP@thinspace
724        \fi
725      \fi
726    \fi
```

```
727     \string?}
728   \declare@shorthand{french}{:}{%
729     \ifFB@spacing
730       \ifhmode
731         \ifdim\lastskip>1sp
732           \unskip\penalty\@M\FBcolonspace
733         \else
734           \FDP@colonspace
735         \fi
736       \fi
737     \fi
738     \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.

```
739   \declare@shorthand{system}{:}{\string:}
740   \declare@shorthand{system}{!}{\string!}
741   \declare@shorthand{system}{?}{\string?}
742   \declare@shorthand{system}{;}{\string;}
```

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

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

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

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

```
752 \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).

```
753 \def\autospace@beforeFDP{%
754   \ifFB@luatex@punct \FB@addDPspace=\@ne \fi
```

34

```
755  \def\FDP@thinspace{\penalty\@M\FBthinspace}%
756  \def\FDP@colonspace{\penalty\@M\FBcolonspace}}
757 \def\noautospace@beforeFDP{%
758   \ifFB@luatex@punct \FB@addDPspace=\z@ \fi
759   \let\FDP@thinspace\@empty
760   \let\FDP@colonspace\@empty}
761 \ifLaTeXe
762   \def\AutoSpaceBeforeFDP{\autospace@beforeFDP
763                            \FBAutoSpacePunctuationtrue}
764   \def\NoAutoSpaceBeforeFDP{\noautospace@beforeFDP
765                              \FBAutoSpacePunctuationfalse}
766   \AtEndOfPackage{\AutoSpaceBeforeFDP}
767 \else
768   \let\AutoSpaceBeforeFDP\autospace@beforeFDP
769   \let\NoAutoSpaceBeforeFDP\noautospace@beforeFDP
770   \AutoSpaceBeforeFDP
771 \fi
```

\rmfamilyFB
\sffamilyFB
\ttfamilyFB

In LaTeX2e \ttfamily (and hence \texttt) will be redefined 'AtBeginDocument' as \ttfamilyFB so that no space is added before the four ; : ! ? characters, 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 characters* with the 'og'/'fg' options in \frenchsetup{}. This is also a workaround for the weird behaviour of these characters in verbatim mode.

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

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

2.3   Commands for French quotation marks

\guillemotleft
\guillemotright
\textquoteddblleft
\textquoteddblright

pdfLaTeX users are supposed to use 8-bit output encodings (T1, LY1,...) to typeset French, those who still stick to OT1 should load aeguill or a similar package. In both

cases the commands \guillemotleft and \guillemotright will print the French
opening and closing quote characters from the output font. For XeLaTeX and LuaLaTeX,
\guillemotleft and \guillemotright are defined by package fontspec (v. 2.5d
and up).

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

```
781 \ifLaTeXe
782 \else
783   \ifFBunicode
784     \def\guillemotleft{{\char"00AB}}
785     \def\guillemotright{{\char"00BB}}
786     \def\textquotedblleft{{\char"201C}}
787     \def\textquotedblright{{\char"201D}}
788   \else
789     \def\guillemotleft{\leavevmode\raise0.25ex
790                         \hbox{$\scriptscriptstyle\ll$}}
791     \def\guillemotright{\raise0.25ex
792                          \hbox{$\scriptscriptstyle\gg$}}
793     \def\textquotedblleft{``}
794     \def\textquotedblright{''}
795   \fi
796   \let\xspace\relax
797 \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   (including spacing) are printed by \FB@og and \FB@fg, the expansion of the top level
            commands \og and \fg is different in and outside French.

\FB@og and \FB@fg are now designed to work in bookmarks.

```
798 \providecommand\texorpdfstring[2]{#1}
799 \newcommand*{\FB@og}{\texorpdfstring{\@FB@og}{\guillemotleft\space}}
800 \newcommand*{\FB@fg}{\texorpdfstring{\@FB@fg}{\space\guillemotright}}
```

The internal definitions \@FB@og and \@FB@fg need some engine-dependent tuning:
for LuaTeX, \FB@spacing is set to 0 locally to prevent the quotes characters from
adding space when option og=«, fg=» is set.

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

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

```
812 \ifFB@xetex@punct
813   \DeclareRobustCommand*{\@FB@og}{\leavevmode
814         \bgroup\FB@spacingfalse\guillemotleft\egroup
```

```
815        \FB@guillspace}
816    \DeclareRobustCommand*{\@FB@fg}{\ifdim\lastskip>\z@\unskip\fi
817        \FB@guillspace
818        \bgroup\FB@spacingfalse\guillemotright\egroup}
819 \fi
820 \ifFB@active@punct
821    \DeclareRobustCommand*{\@FB@og}{\leavevmode
822        \guillemotleft
823        \FB@guillspace}
824    \DeclareRobustCommand*{\@FB@fg}{\ifdim\lastskip>\z@\unskip\fi
825        \FB@guillspace
826        \guillemotright}
827 \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.

```
828 \newcommand*{\og}{\@empty}
829 \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).

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

843 \addto\extrasfrench{\babel@save\og \babel@save\fg
844                     \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...

```
845 \newcommand*{\ogi}{\FB@og}
846 \newcommand*{\fgi}{\FB@fg}
847 \newcommand*{\@ogi}{\ifmmode\hbox{\ogi}\else\ogi\fi}
848 \newcommand*{\@fgi}{\ifmmode\hbox{\fgi}\else\fgi\fi}
849 \newcommand*{\ogii}{\textquotedblleft}
```

37

```
850 \newcommand*{\fgii}{\textquotedblright}
851 \newcommand*{\@ogii}{\ifmmode\hbox{\ogii}\else\ogii\fi}
852 \newcommand*{\@fgii}{\ifmmode\hbox{\fgii}\else\fgii\fi}
```

and the needed technical stuff to handle options:

```
853 \newcount\FBguill@level
854 \newtoks\FBold@everypar
```

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

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

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

```
871 \ifLaTeXe
872   \DeclareRobustCommand\frquote{%
873     \texorpdfstring{\@ifstar{\FBcloseguillfalse\fr@quote}%
874                            {\FBcloseguilltrue \fr@quote}}%
875                 {\bm@fr@quote}%
876   }
877   \newcommand{\bm@fr@quote}[1]{%
878     \guillemotleft\space #1\space\guillemotright}
879 \else
880   \newcommand\frquote[1]{\fr@quote{#1}}
881 \fi
```

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

```
882 \newcommand{\fr@quote}[1]{%
883   \leavevmode
884   \advance\FBguill@level by \@ne
885   \ifcase\FBguill@level
886   \or
```

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

```
887     \ifx\FBeveryparguill\FBguillnone
888     \else
889       \def\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
890       \FB@addquote@everypar
```

38

```
891    \fi
892    \@ogi #1\@fgi
893  \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.

```
894    \ifx\FBeverylineguill\FBguillopen
895      \def\FBeveryline@quote{\FB@addGUILspace=\z@
896                            \guillemotleft\FBguillspace}%
897      \localleftbox{\FBeveryline@quote}%
898      \let\FBeverypar@quote\relax
899      \@ogi #1\ifFBcloseguill\@fgi\fi
900    \else
901      \ifx\FBeverylineguill\FBguillclose
902        \def\FBeveryline@quote{\FB@addGUILspace=\z@
903                              \guillemotright\FBguillspace}%
904        \localleftbox{\FBeveryline@quote}%
905        \let\FBeverypar@quote\relax
906        \@ogi #1\ifFBcloseguill\@fgi\fi
907      \else
```

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

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

Warn if \FBguill@level > 2:

```
925    \ifx\PackageWarning\@undefined
926      \fb@warning{\noexpand\frquote\space handles up to
927                  two levels.\\ Quotation not printed.}%
928    \else
929      \PackageWarning{french.ldf}{%
930         \protect\frquote\space handles up to two levels.
931         \MessageBreak Quotation not printed.  Reported}
932    \fi
933  \fi
```

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

```
934  \global\advance\FBguill@level by \m@ne
935  \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
936  \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
937      \global\let\FBeveryline@quote\empty
938      \ifx\FBeverylineguill\FBguillnone\else\localleftbox{}\fi
939  \fi
940 }
```

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

```
941 \newcommand*{\NoEveryParQuote}{\let\FBeveryparguill\FBguillnone}
```

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. Coding \StartBabelCommands*{french,acadian} would *only* define \date\CurrentOption, leaving the second language undefined in Babel's sens.

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

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

```
954 \newcommand*{\FB@date}[3]{%
955  {{\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
956  \csname month\romannumeral#2name\endcsname
957  \ifx#1\@empty\else\FBdatespace\number#1\fi}}
958 \newcommand*{\FBdatebox}{\hbox}
959 \newcommand*{\FBdatespace}{\space}
960 \newcommand*{\frenchdate}{\FBdatebox\FB@date}
961 \newcommand*{\acadiandate}{\FBdatebox\FB@date}
```

2.5 Extra utilities

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

\up \up eases the typesetting of superscripts like '1ᵉʳ'. Up to version 2.0 of babel-
\fup 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).

```
962 \newif\ifFB@poorman
963 \newdimen\FB@Mht
964 \ifLaTeXe
965   \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 re-
defined to do nothing by option LowercaseSuperscripts=false of \frenchsetup{}.

```
966   \newcommand*{\FBsupR}{-0.12}
967   \newcommand*{\FBsupS}{0.65}
968   \newcommand*{\FB@lc}[1]{\MakeLowercase{#1}}
969   \DeclareRobustCommand*{\FB@up@fake}[1]{%
970     \settoheight{\FB@Mht}{M}%
971     \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
972     \addtolength{\FB@Mht}{-\FBsupS ex}%
973     \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
974     }
```

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.

```
975   \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
976                             \def\FB@suffix{#4}}
977   \def\FB@x{x}
978   \def\FB@j{j}
979   \DeclareRobustCommand*{\FB@up}[1]{%
980     \bgroup \FB@poormantrue
981       \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.

```
982        \edef\reserved@a{\lowercase{%
983          \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
984        \reserved@a
985          {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
986           \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
987           \ifFB@poorman \FB@up@fake{#1}%
988           \else           \FB@up@real{#1}%
989           \fi}%
990          {\FB@up@fake{#1}}%
991      \egroup}
```

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

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

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

```
994      \DeclareRobustCommand*{\fup}[1]{%
995        \texorpdfstring{\ifx\realsuperscript\@undefined
996                        \FB@up{#1}%
997                      \else
998                        \bgroup\let\fakesuperscript\FB@up@fake
999                          \realsuperscript{\FB@lc{#1}}\egroup
1000                     \fi
1001                     }{#1}%
1002    }
```

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

```
1003    \providecommand*{\up}{\fup}
```

Poor man's definition of \up for Plain.

```
1004 \else
1005    \providecommand*{\up}[1]{\leavevmode\raise1ex\hbox{\sevenrm #1}}
1006 \fi
```

\ieme   Some handy macros for those who don't know how to abbreviate ordinals:
\ier
\iere   1007 \def\ieme{\up{e}\xspace}
\iemes   1008 \def\iemes{\up{es}\xspace}
\iers   1009 \def\ier{\up{er}\xspace}
\ieres   1010 \def\iers{\up{ers}\xspace}
         1011 \def\iere{\up{re}\xspace}
         1012 \def\ieres{\up{res}\xspace}

\FBmedkern
\FBthickkern 1013 \newcommand*{\FBmedkern}{\kern+.2em}
         1014 \newcommand*{\FBthickkern}{\kern+.3em}
```

Some support macros relying on \up for numbering,

```
1015 \newcommand*{\FrenchEnumerate}[1]{%
1016    #1\texorpdfstring{\up{o}\FBthickkern}{\textdegree\space}}
1017 \newcommand*{\FrenchPopularEnumerate}[1]{%
1018    #1\texorpdfstring{\up{o})\FBthickkern}{\textdegree\space}}
```

Typing \primo should result in '° ' (except in bookmarks where \textdegree is used instead of o-superior),

```
1019 \def\primo{\FrenchEnumerate1}
1020 \def\secundo{\FrenchEnumerate2}
1021 \def\tertio{\FrenchEnumerate3}
1022 \def\quarto{\FrenchEnumerate4}
```

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

```
1023 \def\fprimo){\FrenchPopularEnumerate1}
1024 \def\fsecundo){\FrenchPopularEnumerate2}
1025 \def\ftertio){\FrenchPopularEnumerate3}
1026 \def\fquarto){\FrenchPopularEnumerate4}
```

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

```
1027 \DeclareRobustCommand*{\No}{%
1028   \texorpdfstring{N\up{o}\FBmedkern}{N\textdegree\space}}
1029 \DeclareRobustCommand*{\no}{%
1030    \texorpdfstring{n\up{o}\FBmedkern}{n\textdegree\space}}
1031 \DeclareRobustCommand*{\Nos}{%
1032    \texorpdfstring{N\up{os}\FBmedkern}{N\textdegree\space}}
1033 \DeclareRobustCommand*{\nos}{%
1034    \texorpdfstring{n\up{os}\FBmedkern}{n\textdegree\space}}
```

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

```
1035 \ifLaTeXe
1036   \DeclareRobustCommand*{\bname}[1]{%
1037     \texorpdfstring{\leavevmode\begingroup\kern0pt #1\endgroup}{#1}%
1038   }
1039   \DeclareRobustCommand*{\bsc}[1]{%
1040     \texorpdfstring{\leavevmode\begingroup\kern0pt \scshape #1\endgroup}%
1041                    {\textsc{#1}}%
1042   }
1043 \else
1044   \newcommand*{\bname}[1]{\leavevmode\begingroup\kern0pt #1\endgroup}
1045   \let\bsc\bname
1046 \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.

```
1047 \ifFBunicode
```

```
1048 \providecommand*{\textbackslash}{{\char"005C}}
1049 \providecommand*{\textasciicircum}{{\char"005E}}
1050 \providecommand*{\textasciitilde}{{\char"007E}}
1051 \newcommand*{\FB@degre}{°}
1052 \else
1053   \ifLaTeXe
1054     \newcommand*{\FB@degre}{\r{}}
1055   \fi
1056 \fi
1057 \DeclareRobustCommand*{\boi}{\textbackslash}
1058 \DeclareRobustCommand*{\circonflexe}{\textasciicircum}
1059 \DeclareRobustCommand*{\tild}{\textasciitilde}
1060 \DeclareRobustCommand*{\degre}{%
1061   \texorpdfstring{\FB@degre}{\textdegree}}
1062 \newcommand*{\at}{@}
```

\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). \degres works in math-mode (angles).

If TeX 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.

```
1063 \DeclareRobustCommand*{\degres}{\degre}
1064 \ifLaTeXe
1065   \AtBeginDocument{%
1066     \@ifpackageloaded{fontspec}{}{%
1067       \ifdefined\DeclareEncodingSubset
1068         \DeclareRobustCommand*{\degres}{%
1069           \texorpdfstring{\hbox{\UseTextSymbol{TS1}{\textdegree}}}%
1070                           {\textdegree}}%
1071       \else
1072         \def\Warning@degree@TSone{\FBWarning
1073             {Degrees would look better in TS1-encoding:%
1074               \MessageBreak add \protect
1075               \usepackage{textcomp} to the preamble.%
1076               \MessageBreak Degrees used}}
1077         \DeclareRobustCommand*{\degres}{%
1078           \texorpdfstring{\hbox to 0.3em{\hss\degre\hss}%
1079                           \Warning@degree@TSone
1080                           \global\let\Warning@degree@TSone\relax}%
1081                           {\textdegree}}%
1082       \fi
1083     }%
1084   }
1085 \fi
```

2.6 Formatting numbers

\StandardMathComma As mentioned in the TeXbook p. 134, the comma is of type \mathpunct in math mode:
\DecimalMathComma it is automatically followed by a thin space. This is convenient in lists and intervals but

unpleasant when the comma is used as a decimal separator in French: it has to be entered as {,}. \DecimalMathComma makes the comma be an ordinary character (of type \mathord) in French (or Acadian) *only* (no space added); \StandardMathComma switches back to the standard behaviour of the comma.

Unfortunately, \newcount inside \if breaks Plain formats.

```
1086 \newif\ifFB@icomma
1087 \newcount\mc@charclass
1088 \newcount\mc@charfam
1089 \newcount\mc@charslot
1090 \newcount\std@mcc
1091 \newcount\dec@mcc
1092 \ifFBLuaTeX
1093   \mc@charclass=\Umathcharclass`\,
1094   \newcommand*{\dec@math@comma}{%
1095     \mc@charfam=\Umathcharfam`\,
1096     \mc@charslot=\Umathcharslot`\,
1097     \Umathcode`\,= 0 \mc@charfam \mc@charslot
1098   }
1099   \newcommand*{\std@math@comma}{%
1100     \mc@charfam=\Umathcharfam`\,
1101     \mc@charslot=\Umathcharslot`\,
1102     \Umathcode`\,= \mc@charclass \mc@charfam \mc@charslot
1103   }
1104 \else
1105   \std@mcc=\mathcode`\,
1106   \dec@mcc=\std@mcc
1107   \@tempcnta=\std@mcc
1108   \divide\@tempcnta by "1000
1109   \multiply\@tempcnta by "1000
1110   \advance\dec@mcc by -\@tempcnta
1111   \newcommand*{\dec@math@comma}{\mathcode`\,=\dec@mcc}
1112   \newcommand*{\std@math@comma}{\mathcode`\,=\std@mcc}
1113 \fi
1114 \let\dec@m@c\relax
```

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

```
1115 \newif\if@FBpreamble \ifLaTeXe \@FBpreambletrue \fi
1116 \newif\if@preamble@DecimalMathComma
1117 \newcommand*{\DecimalMathComma}{%
1118   \if@FBpreamble \@preamble@DecimalMathCommatrue
1119   \else
1120     \ifFB@icomma
1121       \PackageWarning{french.ldf}{%
1122         icomma package loaded, \protect\DecimalMathComma\MessageBreak
1123         does nothing.  Reported}%
1124     \else
1125       \ifFBfrench
1126         \dec@math@comma
1127         \let\dec@m@c\dec@math@comma
1128         \expandafter\addto\csname extras\languagename\endcsname
1129           {\dec@m@c}%
```

```
1130        \fi
1131      \fi
1132    \fi
1133 }
1134 \newcommand*{\StandardMathComma}{%
1135    \ifFB@icomma
1136      \PackageWarning{french.ldf}{%
1137        icomma package loaded, \protect\StandardMathComma\MessageBreak
1138        does nothing.  Reported}%
1139    \else
1140      \ifFBfrench
1141        \std@math@comma
1142        \let\dec@m@c\relax
1143      \fi
1144    \fi
1145 }
```

Ii issued in the preamble, \DecimalMathComma works globally on all parts of the document that are typeset in a French dialect. Can be canceled anytime by \StandardMathComma.

```
1146 \ifLaTeXe
1147    \AtBeginDocument{%
1148      \@FBpreamblefalse
1149      \@ifpackageloaded{icomma}%
1150        {\FB@icommatrue
1151          \if@preamble@DecimalMathComma
1152            \PackageWarning{french.ldf}{%
1153              icomma package loaded, \protect\DecimalMathComma%
1154              \MessageBreak does nothing.  Reported}%
1155          \fi
1156        }%
1157        {\if@preamble@DecimalMathComma
1158          \ifFB@mainlanguage@FR \dec@math@comma \fi
1159          \let\dec@m@c\dec@math@comma
1160          \addto\extrasfrench{\dec@m@c}%
1161          \ifdefined\extrasacadian
1162            \addto\extrasacadian{\dec@m@c}%
1163          \fi
1164        \fi
```

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

```
1165          \addto\noextrasfrench{\std@math@comma}%
1166          \ifdefined\noextrasacadian
1167            \addto\noextrasacadian{\std@math@comma}%
1168          \fi
1169        }%
1170    }
1171 \else
1172    \addto\noextrasfrench{\std@math@comma}
1173 \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

46

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:

```
1174 \newcommand*{\nombre}[1]{{#1}\fb@warning{*** \noexpand\nombre
1175                                   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.

```
1176 \ifFB@luatex@punct
1177   \activate@luatexpunct
1178 \fi
1179 \let\FBstop@here\relax
1180 \def\FBclean@on@exit{%
1181   \let\ifLaTeXe\undefined
1182   \let\LaTeXetrue\undefined
1183   \let\LaTeXefalse\undefined
1184   \let\FB@llc\loadlocalcfg
1185   \let\loadlocalcfg\@gobble}
1186 \ifx\magnification\@undefined
1187 \else
1188   \def\FBstop@here{%
1189     \FBclean@on@exit
1190     \ldf@finish\CurrentOption
1191     \let\loadlocalcfg\FB@llc
1192     \endinput}
1193 \fi
1194 \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.

```
1195 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1196 \newcommand*{\Warning@nombre}[1]{%
1197   \ifdefined\numprint
1198     \numprint{#1}%
1199   \else
1200     \PackageWarning{french.ldf}{%
1201       \protect\nombre\space now relies on package numprint.sty,%
1202       \MessageBreak add \protect
1203       \usepackage[autolanguage]{numprint},\MessageBreak
1204       see file numprint.pdf for more options.\MessageBreak
1205       \protect\nombre\space called}%
1206     \global\let\Warning@nombre\relax
1207     {#1}%
1208   \fi
1209 }
```

```
1210 \newcommand*{\FBthousandsep}{\kern \fontdimen2\font \relax}
```

2.7 Caption names

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

\captionsfrench Let's first define \captionsfrench which sets all strings used in the four standard document classes provided with LaTeX.

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

```
1211 \providecommand*{\FBfigtabshape}{\scshape}
```

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

```
1212 \StartBabelCommands*{\BabelLanguages}{captions}
1213      [unicode, fontenc=TU EU1 EU2, charset=utf8]
1214    \SetString{\refname}{Références}
1215    \SetString{\abstractname}{Résumé}
1216    \SetString{\prefacename}{Préface}
1217    \SetString{\contentsname}{Table des matières}
1218    \SetString{\ccname}{Copie à }
1219    \SetString{\proofname}{Démonstration}
1220    \SetString{\partfirst}{Première}
1221    \SetString{\partsecond}{Deuxième}
1222    \SetStringLoop{ordinal#1}{%
1223      \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%
1224      Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1225      Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1226      Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1227 \StartBabelCommands*{\BabelLanguages}{captions}
1228    \SetString{\refname}{R\'ef\'erences}
1229    \SetString{\abstractname}{R\'esum\'e}
1230    \SetString{\bibname}{Bibliographie}
1231    \SetString{\prefacename}{Pr\'eface}
1232    \SetString{\chaptername}{Chapitre}
1233    \SetString{\appendixname}{Annexe}
1234    \SetString{\contentsname}{Table des mati\`eres}
1235    \SetString{\listfigurename}{Table des figures}
1236    \SetString{\listtablename}{Liste des tableaux}
1237    \SetString{\indexname}{Index}
1238    \SetString{\figurename}{Figure}
1239    \SetString{\tablename}{Table}
1240    \SetString{\pagename}{page}
1241    \SetString{\seename}{voir}
1242    \SetString{\alsoname}{voir aussi}
1243    \SetString{\enclname}{P.~J. }
1244    \SetString{\ccname}{Copie \`a }
1245    \SetString{\headtoname}{}
1246    \SetString{\proofname}{D\'emonstration}
1247    \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.

```
1248    \SetString{\partfirst}{Premi\`ere}
1249    \SetString{\partsecond}{Deuxi\`eme}
```

```
1250    \SetString{\partnameord}{partie}
1251    \SetStringLoop{ordinal#1}{%
1252      \partfirst,\partsecond,Troisi\`eme,Quatri\`eme, Cinqui\`eme,%
1253      Sixi\`eme,Septi\`eme,Huiti\`eme,Neuvi\`eme,Dixi\`eme,%
1254      Onzi\`eme,Douzi\`eme,Treizi\`eme,Quatorzi\`eme,Quinzi\`eme,%
1255      Seizi\`eme,Dix-septi\`eme,Dix-huiti\`eme,Dix-neuvi\`eme,%
1256      Vingti\`eme}
1257    \AfterBabelCommands{%
1258      \DeclareRobustCommand*{\FB@emptypart}{\def\thepart{\unskip}}%
1259      \DeclareRobustCommand*{\FB@partname}{%
1260        \ifFBPartNameFull
1261          \csname ordinal\romannumeral\value{part}\endcsname\space
1262          \partnameord\FB@emptypart
1263        \else
1264          Partie%
1265        \fi}%
1266    }
1267    \SetString{\partname}{\FB@partname}
1268 \EndBabelCommands
```

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

```
1269 \AtBeginDocument{%
1270    \ifx\FBfigtabshape\relax
1271    \else
1272      \ifdefined\fnum@figure
1273        \let\fnum@figureORI\fnum@figure
1274        \renewcommand{\fnum@figure}{{\ifFBfrench\FBfigtabshape\fi
1275                                     \fnum@figureORI}}%
1276      \fi
1277      \ifdefined\fnum@table
1278        \let\fnum@tableORI\fnum@table
1279        \renewcommand{\fnum@table}{{\ifFBfrench\FBfigtabshape\fi
1280                                    \fnum@tableORI}}%
1281      \fi
1282    \fi
1283 }
```

2.8   Figure and table captions

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

```
1284 \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 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).

```
1285 \bgroup
1286   \catcode`:=12 \catcode`>=12 \relax
1287   \long\gdef\STD@makecaption#1#2{%
1288     \vskip\abovecaptionskip
1289     \sbox\@tempboxa{#1: #2}%
1290     \ifdim \wd\@tempboxa >\hsize
1291       #1: #2\par
1292     \else
1293       \global \@minipagefalse
1294       \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1295     \fi
1296     \vskip\belowcaptionskip}
1297 \egroup
```

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

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

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

Enable the standard warning only if high punctuation is active.

```
1298 \newif\if@FBwarning@capsep
1299 \ifFB@active@punct\@FBwarning@capseptrue\fi
1300 \newcommand*{\CaptionSeparator}{\space\textendash\space}
1301 \def\FBCaption@Separator{: }
1302 \long\def\FB@makecaption#1#2{%
1303   \vskip\abovecaptionskip
1304   \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1305   \ifdim \wd\@tempboxa >\hsize
1306     #1\FBCaption@Separator #2\par
1307   \else
1308     \global \@minipagefalse
1309     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1310   \fi
1311   \vskip\belowcaptionskip}
```

Disable the standard warning with AMS and SMF classes.

```
1312 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1313 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1314 \@ifclassloaded{amsdtx}{\@FBwarning@capsepfalse}{}
```

```
1315 \@ifclassloaded{amsldoc}{\@FBwarning@capsepfalse}{}
1316 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1317 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1318 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}
```

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

```
1319 \@ifclassloaded{IEEEconf}{\@FBwarning@capsepfalse}{}
1320 \@ifclassloaded{IEEEtran}{\@FBwarning@capsepfalse}{}
1321 \@ifclassloaded{revtex4-2}{\@FBwarning@capsepfalse}{}
1322 \@ifclassloaded{svjour3}{\@FBwarning@capsepfalse}{}
```

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

```
1323 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1324 \ifFB@koma \@FBwarning@capsepfalse \fi
```

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

```
1325 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1326 \ifdefined\@makecaption\else\@FBwarning@capsepfalse\fi
```

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

```
1327 \AtBeginDocument{%
1328   \ifx\@makecaption\STD@makecaption
1329     \global\let\@makecaption\FB@makecaption
```

If `OldFigTabCaptions=true`, do not overwrite `\FBCaption@Separator` (already saved as ': ' for other languages and set to `\CaptionSeparator` by `\extrasfrench` when French is the main language); otherwise locally force `\autospace@beforeFDP` in case `AutoSpacePunctuation=false`.

```
1330     \ifFBOldFigTabCaptions
1331     \else
1332       \def\FBCaption@Separator{{\autospace@beforeFDP : }}%
1333       \ifFBCustomiseFigTabCaptions
1334         \ifFB@mainlanguage@FR
1335           \def\FBCaption@Separator{\CaptionSeparator}%
1336         \fi
1337       \fi
1338     \fi
1339     \@FBwarning@capsepfalse
1340   \fi
```

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

```
1341     \@ifpackageloaded{caption}{\@FBwarning@capsepfalse}{}%
1342     \@ifpackageloaded{caption-light}{\@FBwarning@capsepfalse}{}%
```

Final warning if relevant:

```
1343   \if@FBwarning@capsep
1344     \FBWarning
1345       {Figures' and tables' captions might look like\MessageBreak
```

```
1346          `Figure 1:' in French instead of `Figure 1 :'.\MessageBreak
1347          If this happens, to fix this issue\MessageBreak
1348          switch to LuaLaTeX or XeLaTeX or\MessageBreak
1349          try to add \protect\usepackage{caption} or\MessageBreak
1350          ... leave it as it is; reported}%
1351   \fi
1352   \let\FB@makecaption\relax
1353   \let\STD@makecaption\relax
1354 }
```

2.9 Dots…

\FBtextellipsis Unless a ready-made character is available in the current font, LaTeX's default def-
inition of \textellipsis includes a \kern at the end; this space is not wanted in
some cases (before a closing brace for instance) and \kern breaks hyphenation of
the next word. We define \FBtextellipsis for French (in LaTeX only) the same way
but without the last \kern.

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

```
1355 \ifFBunicode
1356 \else
1357   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1358   \DeclareTextCommand{\FBtextellipsis}{PU}{\9040\046}
1359   \DeclareTextCommand{\FBtextellipsis}{PD1}{\203}
1360   \DeclareTextCommandDefault{\FBtextellipsis}{%
1361       .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}%
1362   \def\bbl@frenchdots{\babel@save\textellipsis
1363                       \let\textellipsis\FBtextellipsis}
1364   \addto\extrasfrench{\bbl@frenchdots}
1365 \fi
```

2.10 More checks about packages' loading order

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

```
1366 \ifFB@active@punct
1367   \@ifpackageloaded{listings}
1368     {\AtBeginDocument{%
1369       \FBWarning{Please load the "listings" package\MessageBreak
1370               AFTER babel/french; reported}}%
1371     }{}
1372 \fi
```

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

```
1373 \newif\if@FBwarning@natbib
1374 \ifFB@active@punct
1375   \@ifpackageloaded{natbib}{}{\@FBwarning@natbibtrue}
1376 \fi
1377 \AtBeginDocument{%
1378   \if@FBwarning@natbib
```

```
1379        \@ifpackageloaded{natbib}{}{\@FBwarning@natbibfalse}%
1380    \fi
1381    \if@FBwarning@natbib
1382      \FBWarning{Please load the "natbib" package\MessageBreak
1383               BEFORE babel/french; reported}%
1384    \fi
1385 }
```

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

```
1386 \newif\if@FBwarning@beamerarticle
1387 \@ifpackageloaded{beamerarticle}{}{\@FBwarning@beamerarticletrue}
1388 \AtBeginDocument{%
1389    \if@FBwarning@beamerarticle
1390      \@ifpackageloaded{beamerarticle}{}%
1391                               {\@FBwarning@beamerarticlefalse}%
1392    \fi
1393    \if@FBwarning@beamerarticle
1394      \FBWarning{Please load the "beamerarticle" package\MessageBreak
1395               BEFORE babel/french; reported}%
1396    \fi
1397 }
```

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.

```
1398 \newcommand*{\frenchsetup}[1]{%
1399    \setkeys{FB}{#1}%
1400 }%
1401 \@onlypreamble\frenchsetup
```

Keep the former name \frenchbsetup working for compatibility.

```
1402 \let\frenchbsetup\frenchsetup
1403 \@onlypreamble\frenchbsetup
```

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

```
1404 \newif\ifFBShowOptions
```

```
1405 \newif\ifFBStandardLayout          \FBStandardLayouttrue
1406 \newif\ifFBGlobalLayoutFrench       \FBGlobalLayoutFrenchtrue
1407 \newif\ifFBReduceListSpacing
1408 \newif\ifFBStandardListSpacing      \FBStandardListSpacingtrue
1409 \newif\ifFBListOldLayout
1410 \newif\ifFBListItemsAsPar
1411 \newif\ifFBCompactItemize
1412 \newif\ifFBStandardItemizeEnv       \FBStandardItemizeEnvtrue
1413 \newif\ifFBStandardEnumerateEnv     \FBStandardEnumerateEnvtrue
1414 \newif\ifFBStandardItemLabels       \FBStandardItemLabelstrue
1415 \newif\ifFBStandardLists            \FBStandardListstrue
1416 \newif\ifFBIndentFirst
1417 \newif\ifFBFrenchFootnotes
1418 \newif\ifFBAutoSpaceFootnotes
1419 \newif\ifFBOriginalTypewriter
1420 \newif\ifFBThinColonSpace
1421 \newif\ifFBThinSpaceInFrenchNumbers
1422 \newif\ifFBFrenchSuperscripts       \FBFrenchSuperscriptstrue
1423 \newif\ifFBLowercaseSuperscripts    \FBLowercaseSuperscriptstrue
1424 \newif\ifFBPartNameFull             \FBPartNameFulltrue
1425 \newif\ifFBCustomiseFigTabCaptions
1426 \newif\ifFBOldFigTabCaptions
1427 \newif\ifFBSmallCapsFigTabCaptions  \FBSmallCapsFigTabCaptionstrue
1428 \newif\ifFBSuppressWarning
1429 \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.

```
1430 \ifFB@koma
1431   \ifdefined\partformat
1432     \def\FB@partformat@fix{%
1433         \ifFBPartNameFull
1434           \babel@save\partformat
1435           \renewcommand*{\partformat}{\partname}%
1436         \fi}
1437     \addto\extrasfrench{\FB@partformat@fix}%
1438   \fi
1439 \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.

```
1440 \def\FB@french{french}
```

```
1441 \def\FB@acadian{acadian}
1442 \newif\ifFB@mainlanguage@FR
1443 \AtEndOfPackage{%
1444   \ifx\bbl@main@language\FB@french \FB@mainlanguage@FRtrue
1445   \else \ifx\bbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1446   \fi
1447   \ifFB@mainlanguage@FR
1448     \FBGlobalLayoutFrenchtrue
1449     \@ifclassloaded{beamer}%
1450       {\PackageInfo{french.ldf}{%
1451           No list customisation for the beamer class,%
1452           \MessageBreak reported}}%
1453       {\@ifpackageloaded{beamerarticle}%
1454         {\FBStandardItemLabelsfalse
1455          \FBStandardListSpacingfalse
1456          \PackageInfo{french.ldf}{%
1457             Minimal list customisation for the beamerarticle%
1458             \MessageBreak package; reported}}%
```

Otherwise customise lists "à la française'':

```
1459         {\FBStandardListSpacingfalse
1460          \FBStandardItemizeEnvfalse
1461          \FBStandardEnumerateEnvfalse
1462          \FBStandardItemLabelsfalse}%
1463       }
1464     \FBIndentFirsttrue
1465     \FBFrenchFootnotestrue
1466     \FBAutoSpaceFootnotestrue
1467     \FBCustomiseFigTabCaptionstrue
1468   \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.

```
1469   \RequirePackage{keyval}%
1470   \define@key{FB}{ShowOptions}[true]%
1471         {\csname FBShowOptions#1\endcsname}%
```

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

```
1472   \define@key{FB}{StandardLayout}[true]%
1473         {\ifFB@mainlanguage@FR
1474             \csname FBStandardLayout#1\endcsname
1475          \else
1476            \PackageWarning{french.ldf}%
1477              {Option `StandardLayout' skipped:\MessageBreak
1478               French is *not* babel's last option.\MessageBreak
1479               Reported}%
1480          \fi
1481          \ifFBStandardLayout
1482            \FBStandardListSpacingtrue
1483            \FBStandardItemizeEnvtrue
1484            \FBStandardItemLabelstrue
1485            \FBStandardEnumerateEnvtrue
1486            \FBIndentFirstfalse
```

```
1487            \FBFrenchFootnotesfalse
1488            \FBAutoSpaceFootnotesfalse
1489          \else
1490            \FBStandardListSpacingfalse
1491            \FBStandardItemizeEnvfalse
1492            \FBStandardItemLabelsfalse
1493            \FBStandardEnumerateEnvfalse
1494            \FBIndentFirsttrue
1495            \FBFrenchFootnotestrue
1496            \FBAutoSpaceFootnotestrue
1497          \fi}%
1498  \define@key{FB}{GlobalLayoutFrench}[true]%
1499          {\ifFB@mainlanguage@FR
1500            \csname FBGlobalLayoutFrench#1\endcsname
1501          \else
1502            \PackageWarning{french.ldf}%
1503              {Option `GlobalLayoutFrench' skipped:\MessageBreak
1504               French is *not* babel's last option.\MessageBreak
1505               Reported}%
1506          \fi}%
```

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

```
1507  \define@key{FB}{ReduceListSpacing}[true]%
1508          {\csname FBReduceListSpacing#1\endcsname
1509           \ifFBReduceListSpacing \FBStandardListSpacingfalse
1510           \else \FBStandardListSpacingtrue\fi
1511           }%
1512  \define@key{FB}{StandardListSpacing}[true]%
1513          {\csname FBStandardListSpacing#1\endcsname}%
1514  \define@key{FB}{ListOldLayout}[true]%
1515          {\csname FBListOldLayout#1\endcsname
1516           \ifFBListOldLayout
1517             \FBStandardEnumerateEnvtrue
1518             \renewcommand*{\FrenchLabelItem}{\textendash}%
1519           \fi}%
1520  \define@key{FB}{CompactItemize}[true]%
1521          {\csname FBCompactItemize#1\endcsname
1522           \ifFBCompactItemize
1523             \FBStandardItemizeEnvfalse
1524             \FBStandardEnumerateEnvfalse
1525           \else
1526             \FBStandardItemizeEnvtrue
1527             \FBStandardEnumerateEnvtrue
1528           \fi}%
1529  \define@key{FB}{StandardItemizeEnv}[true]%
1530          {\csname FBStandardItemizeEnv#1\endcsname}%
1531  \define@key{FB}{StandardEnumerateEnv}[true]%
1532          {\csname FBStandardEnumerateEnv#1\endcsname}%
1533  \define@key{FB}{StandardItemLabels}[true]%
1534          {\csname FBStandardItemLabels#1\endcsname}%
1535  \define@key{FB}{ItemLabels}%
1536          {\renewcommand*{\FrenchLabelItem}{#1}}%
```

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

```
1590              \addto\extrasfrench{\FB@capsep@fix}%
1591              \ifdefined\extrasacadian
1592                \addto\extrasacadian{\FB@capsep@fix}%
1593              \fi
1594          \fi}%
1595  \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1596          {\csname FBSmallCapsFigTabCaptions#1\endcsname
1597           \ifFBSmallCapsFigTabCaptions
1598           \else \let\FBfigtabshape\relax \fi}%
1599  \define@key{FB}{SuppressWarning}[true]%
1600          {\csname FBSuppressWarning#1\endcsname
1601           \ifFBSuppressWarning
1602             \renewcommand{\FBWarning}[1]{}%
1603           \fi}%
```

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

```
1604  \define@key{FB}{INGuillSpace}[true]%
1605          {\csname FBINGuillSpace#1\endcsname
1606           \ifFBINGuillSpace
1607             \renewcommand*{\FBguillspace}{\space}%
1608           \fi}%
1609  \define@key{FB}{InnerGuillSingle}[true]%
1610          {\csname FBInnerGuillSingle#1\endcsname}%
1611  \define@key{FB}{EveryParGuill}[open]%
1612          {\expandafter\let\expandafter
1613            \FBeveryparguill\csname FBguill#1\endcsname
1614           \ifx\FBeveryparguill\FBguillopen
1615           \else\ifx\FBeveryparguill\FBguillclose
1616              \else\ifx\FBeveryparguill\FBguillnone
1617                    \else
1618                      \let\FBeveryparguill\FBguillopen
1619                      \FBWarning{Wrong value for `EveryParGuill':
1620                                try `open',\MessageBreak
1621                                `close' or `none'. Reported}%
1622                  \fi
1623              \fi
1624          \fi}%
1625  \define@key{FB}{EveryLineGuill}[open]%
1626          {\ifFB@luatex@punct
1627             \expandafter\let\expandafter
1628               \FBeverylineguill\csname FBguill#1\endcsname
1629             \ifx\FBeverylineguill\FBguillopen
1630             \else\ifx\FBeverylineguill\FBguillclose
1631                  \else\ifx\FBeverylineguill\FBguillnone
1632                        \else
1633                          \let\FBeverylineguill\FBguillnone
1634                          \FBWarning{Wrong value for `EveryLineGuill':
1635                                    try `open',\MessageBreak
1636                                    `close' or `none'. Reported}%
1637                      \fi
1638                  \fi
1639              \fi
1640          \else
```

58

```
1641            \FBWarning{Option `EveryLineGuill' skipped:%
1642                         \MessageBreak this option is for
1643                         LuaTeX *only*.\MessageBreak Reported}%
1644         \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.

```
1645   \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1646         {\ifFB@luatex@punct
1647            \csname FBucsNBSP#1\endcsname
1648            \ifFBucsNBSP \FB@ucsNBSP=\@ne \fi
1649          \else
1650            \FBWarning{Option `UnicodeNoBreakSpaces' skipped:%
1651                         \MessageBreak this option is for
1652                         LuaTeX *only*.\MessageBreak Reported}%
1653         \fi
1654         }%
```

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 is meant for checking whether a character is single-byte (\FB@second is empty) or not.

```
1655   \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%

1656   \define@key{FB}{og}%
1657         {\ifFBunicode
```

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

```
1658            \ifFB@luatex@punct
1659              \FB@addGUILspace=1 \relax
1660            \fi
```

then with XeTeX it is a bit more tricky:

```
1661            \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).

```
1662              \XeTeXcharclass"13   = \FB@guilo
1663              \XeTeXcharclass"AB   = \FB@guilo
1664              \XeTeXcharclass"A0   = \FB@guilnul
1665              \XeTeXcharclass"202F = \FB@guilnul
1666            \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1667                \ifFB@active@punct
1668                  \FBWarning{Option og=« not supported with this version
1669                          of\MessageBreak LuaTeX/XeTeX; reported}%
1670                \fi
1671              \else
```

This is for conventional TeX engines:

```
1672                \newcommand*{\FB@@og}{%
1673                  \ifFBfrench
1674                    \ifFB@spacing\FB@og\ignorespaces
1675                    \else\guillemotleft
1676                    \fi
1677                  \else\guillemotleft\fi}%
1678                \AtBeginDocument{%
1679                  \ifdefined\uc@dclc
```

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

```
1680                    \uc@dclc{171}{default}{\FB@@og}%
1681                  \else
```

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

```
1682                    \FB@parse#1\endparse
1683                    \ifx\FB@second\@empty
```

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

```
1684                      \ifdefined\mule@def
1685                        \mule@def{11}{\FB@@og}%
1686                      \else
1687                        \ifdefined\DeclareInputText
1688                          \@tempcnta`#1\relax
1689                          \DeclareInputText{\the\@tempcnta}{\FB@@og}%
1690                        \else
```

Package inputenc not loaded, no way...

```
1691                          \FBWarning{Option `og' requires package
1692                                  inputenc;\MessageBreak reported}%
1693                        \fi
1694                      \fi
1695                    \else
```

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

```
1696                      \DeclareUnicodeCharacter{00AB}{\FB@@og}%
1697                    \fi
1698                  \fi}%
1699              \fi
1700            }%
```

Same code for the closing quote.

```
1701    \define@key{FB}{fg}%
1702            {\ifFBunicode
1703              \ifFB@luatex@punct
1704                \FB@addGUILspace=1 \relax
1705              \fi
1706              \ifFB@xetex@punct
```

60

```
1707              \XeTeXcharclass"14   = \FB@guilf
1708              \XeTeXcharclass"BB   = \FB@guilf
1709              \XeTeXcharclass"A0   = \FB@guilnul
1710              \XeTeXcharclass"202F = \FB@guilnul
1711           \fi
1712         \ifFB@active@punct
1713           \FBWarning{Option fg=» not supported with this version
1714                      of\MessageBreak LuaTeX/XeTeX; reported}%
1715           \fi
1716       \else
1717         \newcommand*{\FB@@fg}{%
1718           \ifFBfrench
1719             \ifFB@spacing\FB@fg
1720             \else\guillemotright
1721             \fi
1722           \else\guillemotright\fi}%
1723         \AtBeginDocument{%
1724           \ifdefined\uc@dclc
1725             \uc@dclc{187}{default}{\FB@@fg}%
1726           \else
1727             \FB@parse#1\endparse
1728             \ifx\FB@second\@empty
1729               \ifdefined\mule@def
1730                 \mule@def{27}{{\FB@@fg}}%
1731               \else
1732                 \ifdefined\DeclareInputText
1733                   \@tempcnta`#1\relax
1734                   \DeclareInputText{\the\@tempcnta}{\FB@@fg}%
1735                 \else
1736                   \FBWarning{Option `fg' requires package
1737                              inputenc;\MessageBreak reported}%
1738                 \fi
1739               \fi
1740             \else
1741               \DeclareUnicodeCharacter{00BB}{\FB@@fg}%
1742             \fi
1743           \fi}%
1744       \fi
1745     }%
1746 }
```

\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.

```
1747 \newcommand*{\FBprocess@options}{%
```

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

```
1748   \@ifpackageloaded{enumitem}{%
1749     \ifFBStandardItemizeEnv
```

61

```
1750      \else
1751        \FBStandardItemizeEnvtrue
1752        \PackageInfo{french.ldf}%
1753          {Setting StandardItemizeEnv=true for\MessageBreak
1754            compatibility with enumitem package,\MessageBreak
1755            reported}%
1756      \fi
1757      \ifFBStandardEnumerateEnv
1758      \else
1759        \FBStandardEnumerateEnvtrue
1760        \PackageInfo{french.ldf}%
1761          {Setting StandardEnumerateEnv=true for\MessageBreak
1762            compatibility with enumitem package,\MessageBreak
1763            reported}%
1764      \fi}{}%
1765    \@ifpackageloaded{paralist}{%
1766      \ifFBStandardItemizeEnv
1767      \else
1768        \FBStandardItemizeEnvtrue
1769        \PackageInfo{french.ldf}%
1770          {Setting StandardItemizeEnv=true for\MessageBreak
1771            compatibility with paralist package,\MessageBreak
1772            reported}%
1773      \fi
1774      \ifFBStandardEnumerateEnv
1775      \else
1776        \FBStandardEnumerateEnvtrue
1777        \PackageInfo{french.ldf}%
1778          {Setting StandardEnumerateEnv=true for\MessageBreak
1779            compatibility with paralist package,\MessageBreak
1780            reported}%
1781      \fi}{}%
1782    \@ifpackageloaded{enumerate}{%
1783      \ifFBStandardEnumerateEnv
1784      \else
1785        \FBStandardEnumerateEnvtrue
1786        \PackageInfo{french.ldf}%
1787          {Setting StandardEnumerateEnv=true for\MessageBreak
1788            compatibility with enumerate package,\MessageBreak
1789            reported}%
1790      \fi}{}%
```

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

```
1791    \def\FB@ufl{\update@frenchlists}
1792    \ifFB@mainlanguage@FR
1793      \update@frenchlists
1794    \else
1795      \ifFBStandardItemizeEnv
1796      \else
1797        \PackageWarning{french.ldf}%
1798          {babel-french will not customize lists' layout\MessageBreak
1799            when French is not the main language,\MessageBreak
1800            reported}%
```

```
1801      \fi
1802    \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.

```
1803    \ifFBAutoSpacePunctuation
1804      \autospace@beforeFDP
1805    \else
1806      \noautospace@beforeFDP
1807    \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.

```
1808    \ifFBOriginalTypewriter
1809    \else
1810      \let\ttfamilyORI\ttfamily
1811      \let\rmfamilyORI\rmfamily
1812      \let\sffamilyORI\sffamily
1813      \let\ttfamily\ttfamilyFB
1814      \let\rmfamily\rmfamilyFB
1815      \let\sffamily\sffamilyFB
1816    \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.

```
1817    \@ifpackageloaded{numprint}%
1818      {\ifnprt@autolanguage
1819        \providecommand*{\npstylefrench}{}%
1820        \ifFBThinSpaceInFrenchNumbers
1821          \renewcommand*{\FBthousandsep}{\,}%
1822        \fi
1823        \g@addto@macro\npstylefrench{\npthousandsep{\FBthousandsep}}%
1824      \fi
1825      }{}%
```

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.

```
1826    \ifFBFrenchSuperscripts
1827      \DeclareRobustCommand*{\up}{%
1828        \texorpdfstring{\@ifstar{\FB@up@fake}{\fup}}{}%
1829        }
1830    \else
1831      \DeclareRobustCommand*{\up}{%
1832        \texorpdfstring{\@ifstar{\FB@up@fake}{\textsuperscript}}{}%
1833        }
1834    \fi
```

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

```
1835   \ifFBLowercaseSuperscripts
1836   \else
1837     \renewcommand*{\FB@lc}[1]{##1}%
1838   \fi
```

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

```
1839   \ifFB@koma
1840       \ifx\captionformat\FB@std@capsep
1841         \ifFBCustomiseFigTabCaptions
1842           \renewcommand*{\captionformat}{\CaptionSeparator}%
1843         \else
1844           \renewcommand*{\captionformat}{{\autospace@beforeFDP :\ }}%
1845         \fi
1846       \fi
1847   \fi
1848   \@ifclassloaded{memoir}%
1849       {\ifx\@contdelim\FB@std@capsep
1850         \ifFBCustomiseFigTabCaptions
1851           \captiondelim{\CaptionSeparator}%
1852         \else
1853           \captiondelim{{\autospace@beforeFDP : }}%
1854         \fi
1855       \fi}{}%
1856   \@ifclassloaded{beamer}%
1857       {\protected@edef\FB@capsep{%
1858           \csname beamer@@tmpl@caption label separator\endcsname}%
1859         \ifx\FB@capsep\FB@std@capsep
1860           \ifFBCustomiseFigTabCaptions
1861             \defbeamertemplate{caption label separator}{FBcustom}{%
1862                 \CaptionSeparator}%
1863             \setbeamertemplate{caption label separator}[FBcustom]%
1864           \else
1865             \defbeamertemplate{caption label separator}{FBcolon}{%
1866                 {\autospace@beforeFDP : }}%
1867             \setbeamertemplate{caption label separator}[FBcolon]%
1868           \fi
1869       \fi}{}%
```

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

```
1870   \ifFBShowOptions
1871     \GenericWarning{* }{%
1872     *** List of possible options for babel-french ***\MessageBreak
1873     [Default values between brackets when french is loaded *LAST*]%
1874     \MessageBreak
1875     ShowOptions [false]\MessageBreak
1876     StandardLayout [false]\MessageBreak
1877     GlobalLayoutFrench [true]\MessageBreak
1878     PartNameFull [true]\MessageBreak
1879     IndentFirst [true]\MessageBreak
```

```
1880        ListItemsAsPar [false]\MessageBreak
1881        StandardListSpacing [false]\MessageBreak
1882        StandardItemizeEnv [false]\MessageBreak
1883        StandardEnumerateEnv [false]\MessageBreak
1884        StandardItemLabels [false]\MessageBreak
1885        ItemLabels=\textemdash, \textbullet,
1886            \protect\ding{43},... [\textendash]\MessageBreak
1887        ItemLabeli=\textemdash, \textbullet,
1888            \protect\ding{43},... [\textendash]\MessageBreak
1889        ItemLabelii=\textemdash, \textbullet,
1890            \protect\ding{43},... [\textendash]\MessageBreak
1891        ItemLabeliii=\textemdash, \textbullet,
1892            \protect\ding{43},... [\textendash]\MessageBreak
1893        ItemLabeliv=\textemdash, \textbullet,
1894            \protect\ding{43},... [\textendash]\MessageBreak
1895        StandardLists [false]\MessageBreak
1896        ListOldLayout [false]\MessageBreak
1897        FrenchFootnotes [true]\MessageBreak
1898        AutoSpaceFootnotes [true]\MessageBreak
1899        AutoSpacePunctuation [true]\MessageBreak
1900        ThinColonSpace [false]\MessageBreak
1901        OriginalTypewriter [false]\MessageBreak
1902        UnicodeNoBreakSpaces [false]\MessageBreak
1903        og= <left quote character>, fg= <right quote character>%
1904        INGuillSpace [false]\MessageBreak
1905        EveryParGuill=open, close, none [open]\MessageBreak
1906        EveryLineGuill=open, close, none
1907                    [open in LuaTeX, none otherwise]\MessageBreak
1908        InnerGuillSingle [false]\MessageBreak
1909        ThinSpaceInFrenchNumbers [false]\MessageBreak
1910        SmallCapsFigTabCaptions [true]\MessageBreak
1911        CustomiseFigTabCaptions [true]\MessageBreak
1912        OldFigTabCaptions [false]\MessageBreak
1913        FrenchSuperscripts [true]\MessageBreak
1914        LowercaseSuperscripts [true]\MessageBreak
1915        SuppressWarning [false]\MessageBreak
1916        \MessageBreak
1917        ***********************************************%
1918        \MessageBreak\protect\frenchsetup{ShowOptions}}}
1919    \fi
1920 }
```

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.

```
1921 \AtBeginDocument{%
1922     \providecommand*{\xspace}{\relax}%
```

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

```
1923     \FBprocess@options
```

When option UnicodeNoBreakSpaces is true (LuaLaTeX only) we need to redefine

\FBmedkern, \FBthickkern and \FBthousandsep as Unicode characters.

```
1924    \ifFBucsNBSP
1925      \renewcommand*{\FBmedkern}{\char"202F\relax}%
1926      \renewcommand*{\FBthickkern}{\char"A0\relax}%
1927      \ifFBThinSpaceInFrenchNumbers
1928        \renewcommand*{\FBthousandsep}{\char"202F\relax}%
1929      \else
1930        \renewcommand*{\FBthousandsep}{\char"A0\relax}%
1931      \fi
1932    \fi
```

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

```
1933    \begingroup
1934      \newcommand{\FBOTone}{OT1}%
1935      \ifx\encodingdefault\FBOTone
1936        \FBWarning{OT1 encoding should not be used for French.%
1937                  \MessageBreak
1938                  Add \protect\usepackage[T1]{fontenc} to the
1939                  preamble\MessageBreak of your document; reported}%
1940      \fi
1941    \endgroup
1942 }
```

2.12   French lists

\listFB
\listORI
\FB@listVsettings

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

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

```
1943 \let\listORI\list
1944 \let\endlistORI\endlist
1945 \newdimen\FB@parskip
1946 \def\FB@listVsettings{%
1947      \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1948      \setlength{\partopsep}{0.4ex plus 0.2ex minus 0.2ex}%
```

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

```
1949      \FB@parskip=\parskip
1950      \addtolength{\topsep}{-\FB@parskip}%
1951      \addtolength{\partopsep}{\FB@parskip}%
```

66

```
1952        \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1953        \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
```

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

```
1954        \ifdim\FB@parskip>0pt
1955          \setlength{\parsep}{\FB@parskip}%
1956          \addtolength{\itemsep}{-\FB@parskip}%
1957        \fi
1958 }
1959 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1960 \let\endlistFB\endlistORI
```

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

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

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

- In French the labels of itemize-lists are vertically aligned as shown p. 6.

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

```
\Frlabelitemi    1961 \newcommand*{\FrenchLabelItem}{\textemdash}
\Frlabelitemii   1962 \newcommand*{\Frlabelitemi}{\FrenchLabelItem}
\Frlabelitemiii  1963 \newcommand*{\Frlabelitemii}{\FrenchLabelItem}
\Frlabelitemiv   1964 \newcommand*{\Frlabelitemiii}{\FrenchLabelItem}
                 1965 \newcommand*{\Frlabelitemiv}{\FrenchLabelItem}
```

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

```
1966 \newdimen\listindentFB
1967 \setlength{\listindentFB}{-1pt}
1968 \newdimen\descindentFB
1969 \setlength{\descindentFB}{-1pt}
1970 \newdimen\labelindentFB
1971 \setlength{\labelindentFB}{-1pt}
1972 \newdimen\labelwidthFB
1973 \setlength{\labelwidthFB}{-1pt}
```

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

```
1974 \newdimen\leftmarginFB
1975 \def\FB@listHsettings{%
1976   \ifFBListItemsAsPar
```

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

```
1977     \itemindent=\labelindentFB
1978     \advance\itemindent by \labelwidthFB
1979     \advance\itemindent by \labelsep
1980     \leftmargini\z@
1981     \bbl@for\FB@dp {2, 3, 4, 5, 6}%
1982       {\csname leftmargin\romannumeral\FB@dp\endcsname =
1983          \labelindentFB}%
1984   \else
```

Default layout: labels hanging into the list left margin.

```
1985     \leftmarginFB=\labelwidthFB
1986     \advance\leftmarginFB by \labelsep
1987     \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1988       {\csname leftmargin\romannumeral\FB@dp\endcsname =
1989          \leftmarginFB}%
1990     \advance\leftmargini by \listindentFB
```

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

```
1991     \listparindent=\parindent
1992   \fi
1993   \leftmargin=\csname leftmargin%
1994     \ifnum\@listdepth=\@ne i\else ii\fi\endcsname
1995 }
```

\itemizeFB New environment for French itemize-lists.
\FB@itemizesettings \FB@itemizesettings does two things: first suppress all vertical spaces including glue unless option StandardListSpacing is set, then set horizontal indentations according to \FB@listHsettings unless option ListOldLayout is true (compatibility with lists up to v2.5k).

```
1996 \def\FB@itemizesettings{%
1997     \ifFBStandardListSpacing
1998     \else
1999       \setlength{\topsep}{\z@}%
2000       \setlength{\partopsep}{\z@}%
2001       \FB@parskip=\parskip
2002       \addtolength{\topsep}{-\FB@parskip}%
2003       \addtolength{\partopsep}{\FB@parskip}%
2004       \setlength{\itemsep}{\z@}%
2005       \setlength{\parsep}{\z@}%
2006       \ifdim\FB@parskip>0pt
2007         \setlength{\parsep}{\FB@parskip}%
2008         \addtolength{\itemsep}{-\FB@parskip}%
2009       \fi
2010     \fi
2011     \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
2012     \ifFBListOldLayout
2013       \setlength{\leftmargin}{\labelwidth}%
2014       \addtolength{\leftmargin}{\labelsep}%
2015       \addtolength{\leftmargin}{\parindent}%
2016     \else
2017       \FB@listHsettings
2018     \fi
2019 }
```

68

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

```
2020 \def\itemizeFB{%
2021     \ifnum \@itemdepth >\thr@@\@toodeep\else
2022         \advance\@itemdepth by \@ne
2023         \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
2024         \expandafter
2025         \listORI
2026         \csname\@itemitem\endcsname
2027         \FB@itemizesettings
2028     \fi
2029 }
2030 \let\enditemizeFB\endlistORI

2031 \def\setlabelitemsFB{%
2032     \let\labelitemi\Frlabelitemi
2033     \let\labelitemii\Frlabelitemii
2034     \let\labelitemiii\Frlabelitemiii
2035     \let\labelitemiv\Frlabelitemiv
2036     \ifdim\labelwidthFB<\z@
2037         \settowidth{\labelwidthFB}{\FrenchLabelItem}%
2038     \fi
2039 }
2040 \def\setlistindentFB{%
2041     \ifdim\labelindentFB<\z@
2042         \ifdim\parindent=\z@
2043             \setlength{\labelindentFB}{1.5em}%
2044         \else
2045             \setlength{\labelindentFB}{\parindent}%
2046         \fi
2047     \fi
2048     \ifdim\listindentFB<\z@
2049         \ifdim\parindent=\z@
2050             \setlength{\listindentFB}{1.5em}%
2051         \else
2052             \setlength{\listindentFB}{\parindent}%
2053         \fi
2054     \fi
2055     \ifdim\descindentFB<\z@
2056         \ifFBListItemsAsPar
2057             \setlength{\descindentFB}{\labelindentFB}%
2058         \else
2059             \setlength{\descindentFB}{\listindentFB}%
2060         \fi
2061     \fi
2062 }
```

\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.

```
2063 \def\enumerateFB{%
2064     \ifnum \@enumdepth >\thr@@\@toodeep\else
```

```
2065        \advance\@enumdepth by \@ne
2066        \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2067        \expandafter
2068        \list
2069          \csname label\@enumctr\endcsname
2070          {\FB@listHsettings
2071            \usecounter\@enumctr\def\makelabel##1{\hss\llap{##1}}}%
2072     \fi
2073 }
2074 \let\endenumerateFB\endlistORI
```

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

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

```
2075 \def\descriptionFB{%
2076      \list{}{\FB@listHsettings
2077              \labelwidth=\z@
2078              \ifFBListItemsAsPar
2079                \itemindent=\descindentFB
2080              \else
2081                \itemindent=-\leftmargin
2082                \ifnum\@listdepth=\@ne
2083                  \ifdim\descindentFB=\z@
2084                    \ifdim\listindentFB>\z@
2085                      \leftmargini=\listindentFB
2086                      \leftmargin=\leftmargini
2087                      \itemindent=-\leftmargin
2088                    \fi
2089                  \else
2090                    \advance\itemindent by \descindentFB
2091                  \fi
2092                \fi
2093              \fi
2094              \let\makelabel\descriptionlabel}%
2095 }
2096 \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).

```
2097 \def\update@frenchlists{%
2098   \setlistindentFB
2099   \ifFBStandardListSpacing
2100   \else \let\list\listFB \fi
2101   \ifFBStandardItemizeEnv
2102   \else \let\itemize\itemizeFB \fi
2103   \ifFBStandardItemLabels
2104   \else \setlabelitemsFB \fi
2105   \ifFBStandardEnumerateEnv
```

```
2106    \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2107 }
```

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. .

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

```
2108 \def\FB@ufl{\relax}
2109 \def\bbl@frenchlistlayout{%
2110    \ifFBGlobalLayoutFrench
2111    \else
2112       \babel@save\list          \babel@save\itemize
2113       \babel@save\enumerate     \babel@save\description
2114       \babel@save\labelitemi    \babel@save\labelitemii
2115       \babel@save\labelitemiii \babel@save\labelitemiv
2116       \FB@ufl
2117    \fi
2118 }
2119 \addto\extrasfrench{\bbl@frenchlistlayout}
```

2.13   French indentation of sections

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

Indentation changes at language switches in only two cases:

a) GlobalLayoutFrench=false,

b) IndentFirst=true and French isn't the main language.

```
2120 \def\bbl@frenchindent{%
2121    \ifFBGlobalLayoutFrench\else\babel@save\@afterindentfalse\fi
2122    \ifFBIndentFirst
2123       \ifFB@mainlanguage@FR\else\babel@save\@afterindentfalse\fi
2124       \let\@afterindentfalse\@afterindenttrue
2125       \@afterindenttrue
2126    \fi}
2127 \addto\extrasfrench{\bbl@frenchindent}
```

2.14   Formatting footnotes

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

The layout of footnotes is controlled by two flags `\ifFBAutoSpaceFootnotes` and `\ifFBFrenchFootnotes` which are set by options of `\frenchsetup{}` (see section). 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.

```
2128 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2129                 {\PackageInfo{french.ldf}%
2130                   {bigfoot package in use.\MessageBreak
2131                    babel-french will NOT customise footnotes;%
2132                    \MessageBreak reported}}%
2133                 {\let\@footnotemarkORI\@footnotemark
2134                  \def\@footnotemarkFB{\leavevmode\unskip\unkern
2135                                       \,\@footnotemarkORI}%
2136                  \ifFBAutoSpaceFootnotes
2137                    \let\@footnotemark\@footnotemarkFB
2138                  \fi}%
2139                 }
```

\@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).

```
2140 \newdimen\parindentFFN
2141 \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.

```
2142 \newcommand*{\dotFFN}{.}
2143 \newcommand*{\kernFFN}{\kern .5em}
2144 \newdimen\FBfnindent
```

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

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

```
2145 \ifFB@koma
2146   \let\@makefntextORI\@makefntext
2147   \let\@@makefnmarkORI\@@makefnmark
```

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

```
2148   \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
```

72

```
2149                {\thefootnotemark\dotFFN\kernFFN}
2150    \let\@makefntextFB\@makefntext
2151    \let\@@makefnmarkFB\@@makefnmark
```

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

```
2152    \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2153                {\textsuperscript{\thefootnotemark}}
2154    \let\@makefntextTH\@makefntext
2155    \let\@@makefnmarkTH\@@makefnmark
```

Restore the original definitions.

```
2156    \let\@makefntext\@makefntextORI
2157    \let\@@makefnmark\@@makefnmarkORI
2158 \fi
```

Definitions for the memoir class:

```
2159 \@ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2160    {\newcommand{\@makefntextFB}[1]{%
2161        \def\footscript##1{##1\dotFFN\kernFFN}%
2162        \setlength{\footmarkwidth}{\FBfnindent}%
2163        \setlength{\footmarksep}{-\footmarkwidth}%
2164        \setlength{\footparindent}{\parindentFFN}%
2165        \makefootmark #1}%
2166    }{}
```

Definitions for the beamer class:

```
2167 \@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.

```
2168    {\def\@makefntextFB#1{%
2169        \def\insertfootnotetext{#1}%
2170        \def\insertfootnotemark{\insertfootnotemarkFB}%
2171        \usebeamertemplate***{footnote}}%
2172     \def\insertfootnotemarkFB{%
2173        \usebeamercolor[fg]{footnote mark}%
2174        \usebeamerfont*{footnote mark}%
2175        \llap{\@thefnmark}\dotFFN\kernFFN}%
2176    }{}
```

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)!

```
2177 \providecommand*{\insertfootnotemarkFB}{%
2178    \parindent=\parindentFFN
2179    \rule\z@\footnotesep
2180    \setbox\@tempboxa\hbox{\@thefnmark}%
2181    \ifdim\wd\@tempboxa>\z@
2182      \llap{\@thefnmark}\dotFFN\kernFFN
2183    \fi}
2184 \providecommand\@makefntextFB[1]{\insertfootnotemarkFB #1}
```

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

The LuaTeX command \localleftbox and \FBeverypar@quote used by \frquote{} have to be reset inside footnotes; done for LaTeX based formats only.

```
2185 \providecommand\localleftbox[1]{}
2186 \AtBeginDocument{%
2187    \@ifpackageloaded{bigfoot}{}%
2188      {\ifdim\parindentFFN<10in
2189       \else
2190         \parindentFFN=\parindent
2191         \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2192       \fi
2193       \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2194       \addtolength{\FBfnindent}{\parindentFFN}%
2195       \let\@makefntextORI\@makefntext
2196       \ifFB@koma
```

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

```
2197         \let\@@makefnmarkORI\@@makefnmark
2198         \long\def\@makefntext#1{%
2199           \localleftbox{}%
2200           \let\FBeverypar@save\FBeverypar@quote
2201           \let\FBeverypar@quote\relax
2202           \ifFBFrenchFootnotes
2203             \ifx\footnote\thanks
2204               \let\@@makefnmark\@@makefnmarkTH
2205               \@makefntextTH{#1}
2206             \else
2207               \let\@@makefnmark\@@makefnmarkFB
2208               \@makefntextFB{#1}
2209             \fi
2210           \else
2211             \let\@@makefnmark\@@makefnmarkORI
2212             \@makefntextORI{#1}%
2213           \fi
2214           \let\FBeverypar@quote\FBeverypar@save
2215           \localleftbox{\FBeveryline@quote}}%
2216       \else
```

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

```
2217         \@ifclassloaded{memoir}%
2218           {\ifFBFrenchFootnotes
2219             \setlength{\thanksmarkwidth}{\parindentFFN}%
2220             \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2221           \fi
2222           }{}%
```

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

```
2223        \@ifclassloaded{beamer}%
2224          {\ifFBFrenchFootnotes
2225            \ifdim\parindentFFN=1.5em\else
2226              \FBWarning{%
2227                \protect\parindentFFN\space is ineffective%
2228                \MessageBreak within the beamer class.%
2229                \MessageBreak Reported}%
2230            \fi
2231          \fi
2232          }{}%
```

Definition of \@makefntext for all other classes:

```
2233        \long\def\@makefntext#1{%
2234          \localleftbox{}%
2235          \let\FBeverypar@save\FBeverypar@quote
2236          \let\FBeverypar@quote\relax
2237          \ifFBFrenchFootnotes
2238            \@makefntextFB{#1}%
2239          \else
2240            \@makefntextORI{#1}%
2241          \fi
2242          \let\FBeverypar@quote\FBeverypar@save
2243          \localleftbox{\FBeveryline@quote}}%
2244      \fi
2245    }%
2246 }
```

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.

```
2247 \newcommand*{\AddThinSpaceBeforeFootnotes}{\FBAutoSpaceFootnotestrue}
2248 \newcommand*{\FrenchFootnotes}{\FBFrenchFootnotestrue}
2249 \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.

```
2250 \FBclean@on@exit
2251 \ldf@finish\CurrentOption
2252 \let\loadlocalcfg\FB@llc
2253 </french>
```

2.16 Files `frenchb.ldf`, `francais.ldf`, `canadien.ldf` and `acadian.ldf`

Babel now expects a *<lang>*.ldf file for each *<lang>*. So we create portmanteau .ldf files for options canadien, francais, frenchb and acadian. These files themselves only load french.ldf which does the real work. Warn users about options canadien, frenchb and francais being deprecated and force recommended options acadian or french.

```
2254 <*acadian>
2255 \PackageInfo{acadian.ldf}%
2256   {`acadian' dialect is currently\MessageBreak
2257    *absolutely identical* to the\MessageBreak
2258    `french' language; reported}
2259 </acadian>
2260 <*canadien>
2261 \PackageWarning{canadien.ldf}%
2262   {Option `canadien' for Babel is *deprecated*,\MessageBreak
2263    it might be removed sooner or later.  Please\MessageBreak
2264    use `acadian' instead; reported}%
2265 \def\CurrentOption{acadian}

2266 \def\datecanadien{\dateacadian}
2267 \def\captionscanadien{\captionsacadian}
2268 \def\extrascanadien{\extrasacadian}
2269 \def\noextrascanadien{\noextrasacadian}
2270 </canadien>
2271 <*francais>
2272 \PackageWarning{francais.ldf}%
2273   {Option `francais' for Babel is *deprecated*,\MessageBreak
2274    it might be removed sooner or later.  Please\MessageBreak
2275    use `french' instead; reported}%
2276 \chardef\l@francais\l@french
2277 \def\CurrentOption{french}
2278 </francais>
```

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

```
2279 <*frenchb>
2280 \def\bbl@tempa{frenchb}
2281 \ifx\CurrentOption\bbl@tempa
2282   \chardef\l@frenchb\l@french
2283   \def\CurrentOption{french}
2284   \PackageWarning{babel-french}%
2285     {Option `frenchb' for Babel is *deprecated*,\MessageBreak
2286      it might be removed sooner or later.  Please\MessageBreak
2287      use `french' instead; reported}
2288 \else
2289   \def\bbl@tempa{francais}
2290   \ifx\CurrentOption\bbl@tempa
2291     \chardef\l@francais\l@french
2292     \def\CurrentOption{french}
```

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

```
2293     \ifx\magnification\@undefined
```

```
2294        \PackageWarning{babel-french}%
2295          {Option `francais' for Babel is *deprecated*,\MessageBreak
2296           it might be removed sooner or later.  Please\MessageBreak
2297           use `french' instead; reported}
2298      \fi
2299    \else
2300      \def\bbl@tempa{canadien}
2301      \ifx\CurrentOption\bbl@tempa
2302        \def\CurrentOption{acadian}
2303        \PackageWarning{babel-french}%
2304          {Option `canadien' for Babel is *deprecated*,\MessageBreak
2305           it might be removed sooner or later.  Please\MessageBreak
2306           use `acadian' instead; reported}
2307      \fi
2308    \fi
2309 \fi
2310 </frenchb>
2311 <acadian|canadien|frenchb|francais>\input french.ldf\relax
2312 <acadian|canadien>\let\extrasacadian\extrasfrench
2313 <acadian|canadien>\let\noextrasacadian\noextrasfrench
```

3 Change History

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

81