

A Babel language definition file for French

frenchb.dtx v3.5f, 2019/09/07

Daniel Flipo
daniel.flipo@free.fr

Contents

1 The French language	2
1.1 Basic interface	2
1.2 Customisation	5
1.2.1 <code>\frenchsetup</code>	5
1.2.2 Caption names	9
1.2.3 Figure and table captions	10
1.3 Hyphenation checks	10
1.4 Changes	11
2 The code	14
2.1 Initial setup	14
2.2 Punctuation	17
2.2.1 Punctuation with LuaTeX	20
2.2.2 Punctuation with XeTeX	30
2.2.3 Punctuation with standard (pdf)TeX	33
2.2.4 Punctuation switches common to all engines	35
2.3 Commands for French quotation marks	36
2.4 Date in French	40
2.5 Extra utilities	41
2.6 Formatting numbers	45
2.7 Caption names	48
2.8 Figure and table captions	49
2.9 Dots...	53
2.10 More checks about packages' loading order	53
2.11 Setup options: keyval stuff	54
2.12 French lists	68
2.13 French indentation of sections	73
2.14 Formatting footnotes	74
2.15 Clean up and exit	78
2.16 Files frenchb.ldf, francais.ldf, canadien.ldf and acadian.ldf	78
3 Change History	80

1 The French language

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

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

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

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

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

Changes between version 3.0 and v3.5f 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;

¹The file described in this section has version number v3.5f and was last revised on 2019/09/07.

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

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

4. footnotes are displayed “à la française”.
5. the separator following the table or figure number in captions is printed as ‘ – ’ instead of ‘: ’; for changing this see 1.2.3 p. 10.

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

1. French hyphenation patterns are made active;
2. ‘high punctuation’ characters (: ; ! ?) automatically add correct spacing⁵ in French; this is achieved using callbacks in Lua(La)TeX or ‘XeTeXinterchar’ mechanism in Xe(La)TeX; with TeX’82 and pdf(La)TeX these four characters are made active in the whole document;
3. `\today` prints the date in French;
4. the caption names are translated into French (LaTeX only). For customisation of caption names see section 1.2.2 p. 9.
5. the space after `\dots` is removed in French.

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

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

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

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

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

- with all engines: the inner quotation is surrounded by double quotes (“texte”) unless option `InnerGuillSingle=true`, then a) the inner quotation is printed as < texte > and b) if the inner quotation spreads

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

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

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.

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 1^{er} janvier 2001 in a box without any linebreak.
3. A command `\up` is provided to typeset superscripts like `M\up{me}` (abbreviation for “Madame”), `1\up{er}` (for “premier”). Other commands are also provided for ordinals: `\ier`, `\iere`, `\iers`, `\ieres`, `\ieme`, `\iemes` (3*\iemes* prints 3^{es}). All these commands take advantage of real superscript letters when they are available in the current font.
4. Family names should be typeset in small capitals and never be hyphenated, the macro `\bsc` (boxed small caps) does this, e.g., `L.\bsc{Lamport}` will print the same as `L.\mbox{\textsc{Lamport}}`. Note that composed names (such as Dupont-Durant) may now be hyphenated on explicit hyphens, this differs from babel-french v. 1.x.
5. Commands `\primo`, `\secundo`, `\tertio` and `\quarto` print 1°, 2°, 3°, 4°. `\FrenchEnumerate{6}` prints 6°.
6. Abbreviations for “Numéro(s)” and “numéro(s)” (N° N^{os} n° and n^{os}) are obtained via the commands `\No`, `\Nos`, `\no`, `\nos`.
7. Two commands are provided to typeset the symbol for “degré”: `\degre` prints the raw character and `\degres` should be used to typeset temperatures (e.g., “20~\degres C” with a non-breaking space), or for alcohols” strengths (e.g., “45\degres” with *no* space in French).
8. In math mode the comma has to be surrounded with braces to avoid a spurious space being inserted after it, in decimal numbers for instance (see the T_EXbook p. 134). The command `\DecimalMathComma` makes the comma behave as an ordinary character *when the current language is French* (no space added); as a counterpart, if `\DecimalMathComma` is active, an explicit space has to be added in lists and intervals: `$(0,\ 1)$`, `$(x,\ y)$`. `\StandardMathComma` switches back to the standard behaviour of the comma in French.

The `icomma` package is an alternative workaround.

9. A command `\nombre` was provided in 1.x versions to easily format numbers in slices of three digits separated either by a comma in English or with a space in French; `\nombre` is now mapped to `\numprint` from `numprint.sty`, see `numprint.pdf` for more information.
10. `babel-french` has been designed to take advantage of the `xspace` package if present: adding `\usepackage{xspace}` in the preamble will force macros like `\fg`, `\ier`, `\ieme`, `\dots`, ..., to respect the spaces you type after them, for instance typing `'1\ier juin'` will print '1^{er} juin' (no need for a forced space after `1\ier`).

1.2 Customisation

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

1.2.1 `\frenchsetup{options}`

`\frenchsetup{}` and `\frenchbsetup{}` are synonymous; the latter should be preferred as the language name for French in `babel` is no longer `frenchb` but `french`.

`\frenchsetup{ShowOptions}` prints all available options to the `.log` file, it is just meant as a remainder of the list of offered options. As usual with `keyval` syntax, boolean options (as `ShowOptions`) can be entered as `ShowOptions=true` or just `ShowOptions`, the `=true` part can be omitted.

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

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

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

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

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

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

Text starting at ‘parindent’
 \leq Leftmargin

- first item running on two lines or more. . .
- first second level item on two lines. . .
- next one. . .
- second item. . .

Default French layout

Text starting at ‘parindent’
 \leq Leftmargin

- first item running on two lines or more. . .
- first second level item on two lines. . .
- next one. . .
- second item. . .

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*)` ; starting with version 2.6 babel-french redefines the `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 the example `\ding{43}` requires `\usepackage{pifont}`.

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

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

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

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

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

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

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

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

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

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

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

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

`{\NoAutoSpacing http://mysite}`⁸ or `{\NoAutoSpacing ???}` (needed for pdfTeX only).

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

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

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

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

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

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

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

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

`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 paragrah included in the inner quotation.

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

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

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

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

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

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

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

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

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

1.2.2 Caption names

All caption names can easily be customised in French using the simplified syntax introduced by `babel 3.9`, for instance `\def\frenchproofname{Preuve}` or

⁹Actually without stretch nor shrink.

`\def\acadianproofname{Preuve}` for the acadian dialect. The older syntax `\addto\captionsfrench{\def\proofname{Preuve}}` still works. Keep in mind that *only* french can be used to redefine captions, even if babel’s option was entered as frenchb or francais.

1.2.3 Figure and table captions

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

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

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

Three options are provided to customise figure and table captions:

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

1.3 Hyphenation checks

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

- run pdfLaTeX on the following file:

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

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

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

Frequent mismatches:

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

1.4 Changes

What’s new in version 3.5?

Version 3.5a offers a new option `ListItemsAsPar`. The default layout of lists is unchanged (for backward compatibility), but users should try this new option which ensures a layout of lists closer to French typographic standards: see f.i. how lists are typeset in the book “Lexique des règles typographiques en usage à l’Imprimerie Nationale”.

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

What’s new in version 3.4?

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

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

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

Version 3.4 requires eT_EX and LuaT_EX 1.0.4 or newer.

What's new in version 3.3?

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

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

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

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

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

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

What's new in version 3.2?

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

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

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

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

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

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

What's new in version 3.1?

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

What's new in version 3.0?

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

- babel 3.9 is required now to process `frenchb.ldf`, this change allows for cleaner definitions of dates and captions for the Unicode engines LuaTeX and XeTeX and also provides a simpler syntax for end-users, see section 1.2.2 p.9.
- `\frenchsetup{}` options management has been completely reworked; two new options added.
- Canadian French didn't work as a normal babel's dialect, it should now; btw. the French language should now be loaded as `french`, *not as* `frenchb` or `français` and preferably as a *global* option of `\documentclass`. Some tolerance still exists in v3.0, but do not rely on it.
- babel-french no longer loads `frenchb.cfg`: customisation should definitely be done using `\frenchsetup{}` options.
- Description lists labels are now indented; try setting `\descindentFB=0pt` (or `\listindentFB=0pt` for all lists) in the preamble if you don't like it.
- The last but not least change affects the (recent) LuaTeX-based engines, (this means version 0.76 as included in TL2013 and up): active characters are no longer used in French for 'high punctuation' ¹⁰. Functionalities and user interface are unchanged.

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

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

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

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

2 The code

2.1 Initial setup

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

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

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

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

Quit if eTeX is not available.

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

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

```
31 \let\bbl@tempa\relax
32 \ifdefined\babeltags
33 \else
34   \let\bbl@tempa\endinput
35   \ifdefined\PackageError
36     \PackageError{french.ldf}
37     {babel-french requires babel v.3.16.\MessageBreak
```

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

65 \newif\ifLaTeXe
66 \let\bbl@tempa\relax
67 \ifdefined\magnification
68 \else
69     \ifdefined\@compatibilitytrue
70         \LaTeXtrue
71     \else
72         \PackageError{french.ldf}
73             {LaTeX-2.09 format is no longer supported.\MessageBreak
74             Aborting here}

```

```

75      {Please upgrade to LaTeX2e!}
76      \let\bbl@tempa\endinput
77      \fi
78 \fi
79 \bbl@tempa

```

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

```

80 \newif\ifFBunicode
81 \newif\ifBLuaTeX
82 \newif\ifFBXeTeX
83 \begingroup\expandafter\expandafter\expandafter\endgroup
84 \expandafter\ifx\csname luatexversion\endcsname\relax
85 \else
86   \FBunicodetrue \BLuaTeXtrue
87 \fi
88 \begingroup\expandafter\expandafter\expandafter\endgroup
89 \expandafter\ifx\csname XeTeXrevision\endcsname\relax
90 \else
91   \FBunicodetrue \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 \ifBLaTeX
109   \ifnum\luatexversion<100
110     \ifx\PackageWarning\@undefined
111       \fb@warning{Please upgrade LuaTeX to version 1.0.4 or above!\\%
112         babel-french will make high punctuation characters (;:!)\\%
113         active with LuaTeX < 1.0.4.}%
114     \else
115       \PackageWarning{french.ldf}{Please upgrade LuaTeX
116         to version 1.0.4 or above!\MessageBreak
117         babel-french will make high punctuation characters%
118         \MessageBreak (;:!) active with LuaTeX < 1.0.4;%
119         \MessageBreak reported}%
120     \fi
121   \else
122     \FB@luatex@puncttrue\FB@active@punctfalse
123   \fi
124 \fi
```

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

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

```

132 \FB@nonchar=4095 \relax
133 \fi
134 \fi

```

\FBguillspace These three commands are meant for basic French. Other French dialects can use different settings, see below. According to the I.N. specifications, the ‘:’ requires an inter-word space before it, the other three require just a thin space. We define **\FBcolonspace** as `\space` (inter-word space) and **\FBthinspace** as an half inter-word space with no shrink nor stretch. **\FBguillspace** is defined btw. as spacing for French quotes is handled together with high punctuation for LuaTeX and XeTeX. **\FBguillspace** has been fine tuned by Thierry Bouche to 80% of an inter-word space with reduced stretchability. All three are user customisable in the preamble, best using the **\FBsetspaces** command described below. A penalty will be added before these spaces to prevent line breaking.

```

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

```

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

```

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

```

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

```

147 \ifFB@luatex@punct
148 \ifx\bbl@tempb\FB@acadian
149 \directlua{
150   FBsp.#2.gl.ac[1] = #3
151   FBsp.#2.gl.ac[2] = #4
152   FBsp.#2.gl.ac[3] = #5
153   if #3 > 0.6 then
154     FBsp.#2.ch.ac = 0xA0

```

```

155         elseif #3 > 0.2 then
156             FBsp.#2.ch.ac = 0x202F
157         else
158             FBsp.#2.ch.ac = 0x200B
159         end
160     }%
161 \fi
162 \fi
163 }
164 \@onlypreamble\FBsetspace
165 \fi

```

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

```

166 \ifLaTeXe
167   \addto\extrasfrench{%
168     \ifFB@luatex@punct
169       \edef\bbl@tempa{\detokenize\expandafter{\language}}%
170       \edef\bbl@tempb{\detokenize{french}}%
171       \ifx\bbl@tempa\bbl@tempb \FB@dialect=0 \relax
172     \else
173       \FB@dialect=1 \relax
174     \fi
175   }%
176 \fi

```

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

```

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

```

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

```

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

```

Same for `\FBcolonspace`:

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

And for `\FBguillspace`:

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

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

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

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

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

2.2.1 Punctuation with LuaTeX

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

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

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

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

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

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

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

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

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

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

```

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

```

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

```

228 \newcommand*\set@glue@table}[1]{%
229   \directlua {
230     local s = token.get_meaning("FB#1space")
231     local t = FBget_glue(s)
232     if t then
233       FBsp.#1.gl.fr = t
234       if not FBsp.#1.gl.ac[1] then
235         FBsp.#1.gl.ac = t
236       end
237       if FBsp.#1.gl.fr[1] > 0.6 then
238         FBsp.#1.ch.fr = 0xA0
239       elseif FBsp.#1.gl.fr[1] > 0.2 then

```

```

240         FBsp.#1.ch.fr = 0x202F
241     else
242         FBsp.#1.ch.fr = 0x200B
243     end
244     if not FBsp.#1.ch.ac then
245         FBsp.#1.ch.ac = FBsp.#1.ch.fr
246     end
247 else
248     texio.write_nl('term and log', '')
249     texio.write_nl('term and log',
250         '*** french.ldf warning: Unexpected syntax in FB#1space,')
251     texio.write_nl('term and log',
252         '*** french.ldf warning: LuaTeX table FBsp unchanged.')
```

```

253     texio.write_nl('term and log',
254         '*** french.ldf warning: Consider using FBsetspace to ')
255     texio.write('term and log', 'customise FB#1space.')
256     texio.write_nl('term and log', '')
257 end
258 }%
259 }
260 \fi
261 </french>
```

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

```

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

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

```

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

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

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

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

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

Local definitions for nodes:

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

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

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

```

319     end
320   elseif string.match(toks, "[^%w]F?B?thinspace") then
321     t = {0.5, 0, 0}
322   elseif string.match(toks, "[^%w]space") then
323     t = {1, 1, 1}
324   end
325   return t
326 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](#).

```

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

```

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

```



```

361         return nil
362     end
363 else
364     return nil
365 end
366 end

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

```

434         t = FBsp.colon.gl.fr
435         nbspace.char = FBsp.colon.ch.fr
436     end
437 else
438     if FRdialect then
439         t = FBsp.thin.gl.ac
440         nbspace.char = FBsp.thin.ch.ac
441     else
442         t = FBsp.thin.gl.fr
443         nbspace.char = FBsp.thin.ch.fr
444     end
445 end
446 fbglue = new_glue_scaled(fid, t)

```

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

```

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

```

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

```

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

```

465     if is_glue and glue_wd <= 1 then
466         addgl = false
467     end
468     local t = FBsp.guill.gl.fr
469     nbspace.char = FBsp.guill.ch.fr
470     if FRdialect then
471         t = FBsp.guill.gl.ac
472         nbspace.char = FBsp.guill.ch.ac

```

```

473         end
474         local fbglue = new_glue_scaled(fid, t)
475         if addgl and fbglue then
476             if is_glue then
477                 head = remove_node(head, prev, true)
478             end
479             if (FRucsNBSP) then
480                 nbspace.font = fid
481                 insert_node_before(head, item, copy_node(nbspace))
482             else
483                 insert_node_before(head, item, copy_node(nobreak))
484                 insert_node_before(head, item, copy_node(fbglue))
485             end
486         end
487     end
488 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*.

```

489     if FRspacing and FB_punct_right[char]
490         and fid > 0 and SIG then
491         local next = item.next
492         local next_id, next_subtype, next_char, nextnext, kern_wd
493         if next then
494             next_id = next.id
495             next_subtype = next.subtype
496             if next_id == GLYPH then
497                 next_char = next.char

```

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

```

498         elseif next_id == KERN then
499             kern_wd = next.kern
500             if kern_wd == 0 then
501                 nextnext = next.next
502                 if nextnext then
503                     next = nextnext
504                     next_id = nextnext.id
505                     next_subtype = nextnext.subtype
506                     if next_id == GLYPH then
507                         next_char = nextnext.char
508                     end
509                 end
510             end
511         end
512     end
513     local is_glue = next_id == GLUE
514     if is_glue then

```

```

515         glue_wd = next.width
516     end
517     local addgl = (next_char and not FB_guil_null[next_char]) or
518         (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).

```

519     if is_glue and glue_wd == 0 then
520         addgl = false
521     end
522     local fid = item.font
523     local t = FBsp.guill.gl.fr
524     nbospace.char = FBsp.guill.ch.fr
525     if FRdialect then
526         t = FBsp.guill.gl.ac
527         nbospace.char = FBsp.guill.ch.ac
528     end
529     local fbglue = new_glue_scaled(fid, t)
530     if addgl and fbglue then
531         if is_glue then
532             head = remove_node(head,next,true)
533         end
534         if (FRucsNBSP) then
535             nbospace.font = fid
536             insert_node_after(head, item, copy_node(nbospace))
537         else
538             insert_node_after(head, item, copy_node(fbglue))
539             insert_node_after(head, item, copy_node(nobreak))
540         end
541     end
542 end
543 end
544 return head
545 end
546 return french_punctuation
547 </lua>

```

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

```

548 < *french>
549 \ifFB@luatex@punct
550   \newcommand*{\FB@luatex@punct@french}{%
551     \babel@save\shorthandon
552     \babel@save\shorthandoff
553     \def\shorthandoff##1{%
554       \ifx\PackageWarning\@undefined
555         \fb@warning{\noexpand\shorthandoff{;!?} is helpless with
556           LuaTeX,\ \ use \noexpand\NoAutoSpacing

```

```

557         *inside a group* instead.}%
558     \else
559         \PackageWarning{french.ldb}{\protect\shorthandoff{;! ?} is
560         helpless with LuaTeX,\MessageBreak use \protect\NoAutoSpacing
561         \space *inside a group* instead;\MessageBreak reported}%
562     \fi}%
563     \def\shorthandon##1{%
564 }
565 \addto\extrasfrench{\FB@luatex@punct@french}

```

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

```

566 \def\activate@luatexpunct{%
567     \directlua{%
568         FR_fr = \the\l@french ; FR_ca = \the\l@acadian ;
569         local path = kpse.find_file("frenchb.lua", "lua")
570         if path then
571             local f = dofile(path)
572             luatexbase.add_to_callback("kerning",
573                 f, "frenchb.french_punctuation")
574         else
575             texio.write_nl('')
576             texio.write_nl('*****')
577             texio.write_nl('Error: frenchb.lua not found.')
578             texio.write_nl('*****')
579             texio.write_nl('')
580         end
581     }%
582 }
583 \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.ldb`. We use the same mechanism for French quotes (`«` and `»`), when automatic spacing for quotes is required by options `og=«` and `fg=»` in `\frenchsetup{}` (see section 2.11).

The default value for `\XeTeXcharclass` is 0 for characters tokens and `\FB@nonchar` for all other tokens (glues, kerns, math and box boundaries, etc.). These defaults should not be changed otherwise the spacing before the ‘high punctuation’ characters and inside quotes might not be correct.

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

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

```

584 \ifFB@xetex@punct

```

```

585 \ifLaTeXe
586 \PackageInfo{french.ldf}{No need for active punctuation characters%
587 \MessageBreak with this version of XeTeX!%
588 \MessageBreak reported}
589 \else
590 \fb@info{No need for active punctuation characters\\
591 with this version of XeTeX!}
592 \fi

```

Six new character classes are defined for babel-french.

```

593 \newXeTeXintercharclass\FB@punctthick
594 \newXeTeXintercharclass\FB@punctthin
595 \newXeTeXintercharclass\FB@punctnul
596 \newXeTeXintercharclass\FB@guilo
597 \newXeTeXintercharclass\FB@guilf
598 \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.

```

599 \def\FBsavevariable@loop#1#2{\begingroup
600 \toks@{\expandafter{\originalTeX #1}%
601 \edef\x{\endgroup
602 \def\noexpand\originalTeX{\the\toks@ #2=\the#1#2\relax}}}%
603 \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
)]	{	}	,	-	.	"	%	'	'	'

```

604 \def\FB@charlist{"21,"3A,"3B,"3F,"AB,"BB,"28,"5B,"A0,"202F,%
605 "29,"5D,"7B,"7D,"2C,"2D,"2E,"22,"25,"27,"60,"2019}

```

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

```

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

```

```

617     \else
618         \PackageWarning{french.ldf}{\protect\shorthandoff{;:!?} is
619             helpless with XeTeX,\MessageBreak use \protect\NoAutoSpacing
620             \space *inside a group* instead;\MessageBreak reported}%
621     \fi}%
622     \def\shorthandon##1{%

```

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

```

623     \XeTeXinterchartokenstate=1
624     \XeTeXcharclass '\: = \FB@punctthick
625     \XeTeXinterchartoks \z@ \FB@punctthick = {%
626         \ifFB@spacing\ifhmode\FDP@colonspace\fi\fi}%
627     \XeTeXinterchartoks \FB@guilf \FB@punctthick = {%
628         \ifFB@spacing\FDP@colonspace\fi}%

```

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

```

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

```



```

657         \ifFB@spacing\FB@guillspace\fi}%
658     \XeTeXinterchartoks \FB@guilo \FB@nonchar = {%
659         \ifFB@spacing\FB@guillspace\ignorespaces\fi}%
660     \XeTeXinterchartoks \z@ \FB@guilf = {%
661         \ifFB@spacing\FB@guillspace\fi}%
662     \XeTeXinterchartoks \FB@punctthin \FB@guilf = {%
663         \ifFB@spacing\FB@guillspace\fi}%
664     \XeTeXinterchartoks \FB@nonchar \FB@guilf = {%
665         \ifFB@spacing\unskip\FB@guillspace\fi}%

```

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

```

666     \bbl@for\FB@char
667         {\[, \'[, \'(, "A0, "202F}%
668         {\XeTeXcharclass\FB@char=\FB@punctnul}%

```

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

```

669     \bbl@for\FB@char
670         {\{, \'[, \'., \'-, \' ), \' ), \' ), \' ), "22, "27, "60, "2019}%
671         {\XeTeXcharclass\FB@char=\z@}%
672     }
673     \addto\extrasfrench{\FB@xetex@punct@french}

```

End of specific code for punctuation with modern XeTeX engines.

```

674 \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.

```

675 \ifFB@active@punct
676     \initiate@active@char{:}%
677     \initiate@active@char{;}%
678     \initiate@active@char{!}%
679     \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`.

```

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

```

```

688     \fi
689     \fi

```

Now we can insert a ; character.

```

690     \string;}

```

The next three definitions are very similar.

```

691 \declare@shorthand{french}{!}{%
692   \ifFB@spacing
693     \ifhmode
694       \ifdim\lastskip>1sp
695         \unskip\penalty\@M\FBthinspace
696       \else
697         \FDP@thinspace
698       \fi
699     \fi
700   \fi
701   \string!}
702 \declare@shorthand{french}{?}{%
703   \ifFB@spacing
704     \ifhmode
705       \ifdim\lastskip>1sp
706         \unskip\penalty\@M\FBthinspace
707       \else
708         \FDP@thinspace
709       \fi
710     \fi
711   \fi
712   \string?}
713 \declare@shorthand{french}{:}{%
714   \ifFB@spacing
715     \ifhmode
716       \ifdim\lastskip>1sp
717         \unskip\penalty\@M\FBcolonspace
718       \else
719         \FDP@colonspace
720       \fi
721     \fi
722   \fi
723   \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.

```

724 \declare@shorthand{system}{:}{\string:}
725 \declare@shorthand{system}{!}{\string!}
726 \declare@shorthand{system}{?}{\string?}
727 \declare@shorthand{system}{;}{\string;}
728 %}

```

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

```

729 \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.

```

730   \bbl@activate{:}\bbl@activate{;}%
731   \bbl@activate{!}\bbl@activate{?}%
732   }
733   \addto\noextrasfrench{%
734     \bbl@deactivate{:}\bbl@deactivate{;}%
735     \bbl@deactivate{!}\bbl@deactivate{?}%
736   }
737 \fi

```

2.2.4 Punctuation switches common to all engines

A new ‘if’ `\ifFBAutoSpacePunctuation` needs to be defined now to control the two possible ways of dealing with ‘high punctuation’. its default value is true, but it can be set to false by `\frenchsetup{AutoSpacePunctuation=false}` for finer control.

```

738 \newif\ifFBAutoSpacePunctuation \FBAutoSpacePunctuationtrue

```

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

```

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

```

`\rmfamilyFB` In L^AT_EX₂e `\ttfamily` (and hence `\texttt`) will be redefined ‘AtBeginDocument’ `\sffamilyFB` as `\ttfamilyFB` so that no space is added before the four ; : ! ? characters, `\ttfamilyFB` even if `AutoSpacePunctuation` is true. When `AutoSpacePunctuation` is false, the

eventually typed spaces are left unchanged (not turned into thin spaces, no penalty added). `\rmfamily` and `\sffamily` need to be redefined also (`\ttfamily` is not always used inside a group, its effect can be cancelled by `\rmfamily` or `\sffamily`). These redefinitions can be canceled if necessary, for instance to recompile older documents, see option [OriginalTypewriter](#) below.

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

```
758 \ifLaTeXe
759   \DeclareRobustCommand\ttfamilyFB{\FB@spacing@off \ttfamilyORI}
760   \DeclareRobustCommand\rmfamilyFB{\FB@spacing@on  \rmfamilyORI}
761   \DeclareRobustCommand\sffamilyFB{\FB@spacing@on  \sffamilyORI}
762 \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.

```
763 \DeclareRobustCommand*\NoAutoSpacing{%
764   \FB@spacing@off
765   \ifFB@active@punct\shorthandoff{;:!?}\fi
766 }
```

2.3 Commands for French quotation marks

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

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

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

```

780 \def\textquotedblright{''}
781 \fi
782 \let\xspace\relax
783 \fi

```

\FBgspchar The next step is to provide correct spacing after ‘«’ and before ‘»’; no line break is allowed neither *after* the opening one, nor *before* the closing one. French quotes **\FB@og** (including spacing) are printed by **\FB@og** and **\FB@fg**, the expansion of the top level commands **\og** and **\fg** is different in and outside French.

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

```

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

```

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

```

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

```

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

```

811 \newcommand*{\og}{\@empty}
812 \newcommand*{\fg}{\@empty}

```

The definitions of **\og** and **\fg** for quotation marks are switched on and off through the **\extrafrench** **\noextrafrench** 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).

```

813 \ifLaTeXe
814   \def\bbl@frenchguillemets{%
815     \renewcommand*{\og}{\FB@og}%
816     \renewcommand*{\fg}{\FB@fg\xspace}}
817   \renewcommand*{\og}{\textquotedblleft}
818   \renewcommand*{\fg}{\ifdim\lastskip>\z@unskip\fi
819     \textquotedblright\xspace}
820 \else
821   \def\bbl@frenchguillemets{\let\og\FB@og
822     \let\fg\FB@fg}
823   \def\og{\textquotedblleft}
824   \def\fg{\ifdim\lastskip>\z@unskip\fi\textquotedblright}
825 \fi

826 \addto\extrasfrench{\babel@save\og \babel@save\fg \bbl@frenchguillemets}

```

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

```

827 \newcommand*{\ogi}{\FB@og}
828 \newcommand*{\fgi}{\FB@fg}
829 \newcommand*{\@ogi}{\ifmmode\hbox{\ogi}\else\ogi\fi}
830 \newcommand*{\@fgi}{\ifmmode\hbox{\fgi}\else\fgi\fi}
831 \newcommand*{\ogii}{\textquotedblleft}
832 \newcommand*{\fgii}{\textquotedblright}
833 \newcommand*{\@ogii}{\ifmmode\hbox{\ogii}\else\ogii\fi}
834 \newcommand*{\@fgii}{\ifmmode\hbox{\fgii}\else\fgii\fi}

```

and the needed technical stuff to handle options:

```

835 \newcount\FBguill@level
836 \newtoks\FBbold@everypar

```

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

```

837 \def\FB@addquote@everypar{%
838   \let\FBnew@everypar\everypar
839   \FBbold@everypar=\expandafter{\the\everypar}%
840   \FBnew@everypar={\the\FBbold@everypar\FBeverypar@quote}%
841   \let\everypar\FBbold@everypar
842   \let\FB@addquote@everypar\relax
843 }
844 \newif\ifFBcloseguill \FBcloseguilltrue
845 \newif\ifFBinnerguillsingle
846 \def\FBguillopen{\bgroup\NoAutoSpacing\guillemotleft\egroup}
847 \def\FBguillclose{\bgroup\NoAutoSpacing\guillemotright\egroup}
848 \let\FBguillnone\empty
849 \let\FBeveryparguill\FBguillopen

```

```

850 \let\FBverylineguill\FBguillnone
851 \let\FBverypar@quote\relax
852 \let\FBveryline@quote\empty

```

The main command `\frquote` accepts (in LaTeX2e only) a starred version which suppresses the closing quote; it is meant to be used for inner quotations which end together with the outer one, then only one closing guillemet (the outer one) should be printed.

```

853 \ifLaTeXe
854   \DeclareRobustCommand\frquote{%
855     \@ifstar{\FBcloseguillfalse\fr@quote}%
856             {\FBcloseguilltrue\fr@quote}}
857 \else
858   \newcommand\frquote[1]{\fr@quote{#1}}
859 \fi

```

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

```

860 \newcommand{\fr@quote}[1]{%
861   \leavevmode
862   \advance\FBguill@level by \@ne
863   \ifcase\FBguill@level
864   \or

```

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

```

865   \ifx\FBveryparguill\FBguillnone
866   \else
867     \def\FBverypar@quote{\FBveryparguill\FB@guillspace}%
868     \FB@addquote@everypar
869   \fi
870   \@ogi #1\@fgi
871 \or

```

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

```

872   \ifx\FBverylineguill\FBguilllopen
873     \def\FBveryline@quote{\FB@addGUILspace=0 \guillemotleft
874                           \FB@guillspace}%
875     \llocalleftbox{\FBveryline@quote}%
876     \let\FBverypar@quote\relax
877     \@ogi #1\ifFBcloseguill\@fgi\fi
878   \else
879     \ifx\FBverylineguill\FBguillclose
880       \def\FBveryline@quote{\FB@addGUILspace=0 \guillemotright
881                             \FB@guillspace}%
882       \llocalleftbox{\FBveryline@quote}%
883       \let\FBverypar@quote\relax
884       \@ogi #1\ifFBcloseguill\@fgi\fi
885     \else

```

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

```

886 \let\FBeverypar@quote\relax
887 \ifFBInnerGuillSingle
888 \def\ogii{\leavevmode
889 \guilsinglleft\FB@guillspace}%
890 \def\fgii{\ifdim\lastskip>\z@\unskip\fi
891 \FB@guillspace\guilsinglright}%
892 \ifx\FBeveryparguill\FBguillopen
893 \def\FBeverypar@quote{\guilsinglleft\FB@guillspace}%
894 \fi
895 \ifx\FBeveryparguill\FBguillclose
896 \def\FBeverypar@quote{\guilsinglright\FB@guillspace}%
897 \fi
898 \fi
899 \@ogii #1\ifFBcloseguill \@fgii \fi
900 \fi
901 \fi
902 \else

```

Warn if \FBguill@level > 2:

```

903 \ifx\PackageWarning\@undefined
904 \fb@warning{\noexpand\frquote\space handles up to
905 two levels.\\ Quotation not printed.}%
906 \else
907 \PackageWarning{french.ldf}{%
908 \protect\frquote\space handles up to two levels.
909 \MessageBreak Quotation not printed. Reported}
910 \fi
911 \fi

```

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

```

912 \global\advance\FBguill@level by \m@ne
913 \ifcase\FBguill@level \global\let\FBeverypar@quote\relax
914 \or \gdef\FBeverypar@quote{\FBeveryparguill\FB@guillspace}%
915 \global\let\FBeveryl@ne@quote\empty
916 \ifx\FBeveryl@ne@quote\FBguillnone\else\localleftbox{}\fi
917 \fi
918 }

```

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.

```

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

```

2.4 Date in French

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

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

```

920 \def\BabelLanguages{french,acadian}
921 \StartBabelCommands*{\BabelLanguages}{date}
922     [unicode, fontenc=TU EU1 EU2, charset=utf8]
923     \SetString\monthiiname{février}
924     \SetString\monthviiname{août}
925     \SetString\monthxiiname{décembre}
926 \StartBabelCommands*{\BabelLanguages}{date}
927     \SetStringLoop{month#1name}{%
928         janvier,f\'evrier,mars,avril,mai,juin,juillet,%
929         ao\^ut,septembre,octobre,novembre,d\'ecembre}
930     \SetString\today{\FB@date{\year}{\month}{\day}}
931 \EndBabelCommands

```

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

```

932 \newcommand*{\FB@date}[3]{%
933     {{\number#3}\ifnum1=#3{\ier}\fi\FBdatespace
934     \csname month\romannumeral#2name\endcsname
935     \ifx#1\@empty\else\FBdatespace\number#1\fi}}
936 \newcommand*{\FBdatebox}{\hbox}
937 \newcommand*{\FBdatespace}{\space}
938 \newcommand*{\frenchdate}{\FBdatebox\FB@date}
939 \newcommand*{\acadiandate}{\FBdatebox\FB@date}

```

2.5 Extra utilities

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

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

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

```

940 \newif\ifFB@poorman
941 \newdimen\FB@Mht
942 \ifLaTeXe
943     \AtEndOfPackage{\RequirePackage{scalegnt}}

```

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

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

```

944 \newcommand*\FBsupR{-0.12}
945 \newcommand*\FBsupS{0.65}
946 \newcommand*\FB@lc[1]{\MakeLowercase{#1}}
947 \DeclareRobustCommand*\FB@up@fake[1]{%
948   \settoheight{\FB@Mht}{M}%
949   \addtolength{\FB@Mht}{\FBsupR \FB@Mht}%
950   \addtolength{\FB@Mht}{-\FBsupS ex}%
951   \raisebox{\FB@Mht}{\scalefont{\FBsupS}{\FB@lc{#1}}}%
952 }

```

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.

```

953 \def\FB@split#1#2#3#4\@nil{\def\FB@firstthree{#1#2#3}%
954                               \def\FB@suffix{#4}}
955 \def\FB@x{x}
956 \def\FB@j{j}
957 \DeclareRobustCommand*\FB@up[1]{%
958   \bgroup \FB@poormantrue
959   \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.

```

960 \edef\reserved@a{\lowercase{%
961   \noexpand\IfFileExists{\f@encoding\FB@firstthree -sup.fd}}}%
962 \reserved@a
963 {\ifx\FB@suffix\FB@x \FB@poormanfalse\fi
964  \ifx\FB@suffix\FB@j \FB@poormanfalse\fi
965  \ifFB@poorman \FB@up@fake{#1}%
966  \else \FB@up@real{#1}%

```

```

967         \fi}%
968         {\FB@up@fake{#1}}%
969     \egroup}

```

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

```

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

```

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

```

972 \DeclareRobustCommand*{\fup}[1]{%
973     \ifx\realsuperscript\undefined
974         \FB@up{#1}%
975     \else
976         \bgroup\let\fakesuperscript\FB@up@fake
977             \realsuperscript{\FB@lc{#1}}\egroup
978     \fi}

```

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

```

979 \providecommand*{\up}{\relax}

```

Poor man's definition of \up for Plain.

```

980 \else
981     \providecommand*{\up}[1]{\leavevmode\raiselex\hbox{\sevenrm #1}}
982 \fi

```

```

\ieme Some handy macros for those who don't know how to abbreviate ordinals:
\ier  983 \def\ieme{\up{e}\xspace}
\iere 984 \def\iemes{\up{es}\xspace}
\iemes 985 \def\ier{\up{er}\xspace}
\iers  986 \def\iers{\up{ers}\xspace}
\ieres 987 \def\iere{\up{re}\xspace}
        988 \def\ieres{\up{res}\xspace}

```

```

\FBmedkern
\FBthickkern 989 \newcommand*{\FBmedkern}{\kern+.2em}
              990 \newcommand*{\FBthickkern}{\kern+.3em}

```

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

```

\no 991 \newcommand*{\FrenchEnumerate}[1]{#1\up{o}\FBthickkern}
\Nos 992 \newcommand*{\FrenchPopularEnumerate}[1]{#1\up{o})\FBthickkern}
\nos
\primo Typing \primo should result in 'º ',
\frimo 993 \def\primo{\FrenchEnumerate1}
        994 \def\secundo{\FrenchEnumerate2}
        995 \def\tertio{\FrenchEnumerate3}
        996 \def\quarto{\FrenchEnumerate4}

```

while typing `\fprimo`) gives ‘^o’.

```
997 \def\fprimo){\FrenchPopularEnumerate1}
998 \def\fsecundo){\FrenchPopularEnumerate2}
999 \def\ftertio){\FrenchPopularEnumerate3}
1000 \def\fquarto){\FrenchPopularEnumerate4}
```

Let’s provide four macros for the common abbreviations of “Numéro”.

```
1001 \DeclareRobustCommand*\No}{N\up{o}\FBmedkern}
1002 \DeclareRobustCommand*\no}{n\up{o}\FBmedkern}
1003 \DeclareRobustCommand*\Nos}{N\up{os}\FBmedkern}
1004 \DeclareRobustCommand*\nos}{n\up{os}\FBmedkern}
```

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

```
1005 \DeclareRobustCommand*\bsc}[1]{\leavevmode\begingroup\kern0pt
1006                                     \scshape #1\endgroup}
1007 \ifLaTeXe\else\let\scshape\relax\fi
```

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

```
1008 \ifBUnicode
1009   \newcommand*\at{{\char"0040}}
1010   \newcommand*\circonflexe{{\char"005E}}
1011   \newcommand*\tild{{\char"007E}}
1012   \newcommand*\boi{{\char"005C}}
1013   \newcommand*\degree{{\char"00B0}}
1014 \else
1015   \ifLaTeXe
1016     \DeclareTextSymbol{\at}{T1}{64}
1017     \DeclareTextSymbol{\circonflexe}{T1}{94}
1018     \DeclareTextSymbol{\tild}{T1}{126}
1019     \DeclareTextSymbolDefault{\at}{T1}
1020     \DeclareTextSymbolDefault{\circonflexe}{T1}
1021     \DeclareTextSymbolDefault{\tild}{T1}
1022     \DeclareRobustCommand*\boi{\textbackslash}
1023     \DeclareRobustCommand*\degree{\r{}}
1024   \else
1025     \def\T@one{T1}
1026     \ifx\f@encoding\T@one
1027       \newcommand*\degree{{\char6}}
1028     \else
1029       \newcommand*\degree{{\char23}}
1030     \fi
1031     \newcommand*\at{{\char64}}
```

```

1032 \newcommand*\circflexe{{\char94}}
1033 \newcommand*\tild{{\char126}}
1034 \newcommand*\boi{{\backslash$}}
1035 \fi
1036 \fi

```

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

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

```

1037 \ifLaTeXe
1038 \newcommand*\degrees{\degree}
1039 \ifFBunicode
1040 \DeclareRobustCommand*\degrees{\degree}
1041 \else
1042 \def\Warning@degree@TSone{\FBWarning
1043     {Degrees would look better in TS1-encoding:%
1044     \MessageBreak add \protect
1045     \usepackage{textcomp} to the preamble.%
1046     \MessageBreak Degrees used}}
1047 \AtBeginDocument{\ifx\DeclareEncodingSubset\undefined
1048     \DeclareRobustCommand*\degrees{%
1049         \leavevmode\hbox to 0.3em{\hss\degree\hss}%
1050         \Warning@degree@TSone
1051         \global\let\Warning@degree@TSone\relax}%
1052     \else
1053         \DeclareRobustCommand*\degrees{%
1054             \hbox{\UseTextSymbol{TS1}{\textdegree}}}%
1055     \fi
1056 }
1057 \fi
1058 \else
1059 \newcommand*\degrees{%
1060     \leavevmode\hbox to 0.3em{\hss\degree\hss}}
1061 \fi

```

2.6 Formatting numbers

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

```

1062 \newif\ifFB@icomma
1063 \newcount\mc@charclass
1064 \newcount\mc@charfam
1065 \newcount\mc@charslot
1066 \newcount\std@mcc
1067 \newcount\dec@mcc
1068 \ifBLuaTeX
1069   \mc@charclass=\Umathcharclass'\,
1070   \newcommand*\dec@math@comma{%
1071     \mc@charfam=\Umathcharfam'\,
1072     \mc@charslot=\Umathcharslot'\,
1073     \Umathcode'\,= 0 \mc@charfam \mc@charslot
1074   }
1075   \newcommand*\std@math@comma{%
1076     \mc@charfam=\Umathcharfam'\,
1077     \mc@charslot=\Umathcharslot'\,
1078     \Umathcode'\,= \mc@charclass \mc@charfam \mc@charslot
1079   }
1080 \else
1081   \std@mcc=\mathcode'\,
1082   \dec@mcc=\std@mcc
1083   \@tempcnta=\std@mcc
1084   \divide\@tempcnta by "1000
1085   \multiply\@tempcnta by "1000
1086   \advance\dec@mcc by -\@tempcnta
1087   \newcommand*\dec@math@comma{\mathcode'\,=\dec@mcc}
1088   \newcommand*\std@math@comma{\mathcode'\,=\std@mcc}
1089 \fi

```

\DecimalMathComma operates in French or Acadian independently.

```

1090 \newcommand*\DecimalMathComma{%
1091   \ifFB@icomma
1092     \PackageWarning{french.ldf}{%
1093       icomma package loaded, \protect\DecimalMathComma\MessageBreak
1094       does nothing. Reported}%
1095   \else
1096     \ifFBfrench
1097       \dec@math@comma
1098       \expandafter\addto\csname extras\language\endcsname
1099       {\dec@math@comma}%
1100     \fi
1101   \fi
1102 }
1103 \newcommand*\StandardMathComma{%
1104   \ifFB@icomma
1105     \PackageWarning{french.ldf}{%
1106       icomma package loaded, \protect\StandardMathComma\MessageBreak
1107       does nothing. Reported}%
1108   \else
1109     \std@math@comma
1110     \expandafter\addto\csname extras\language\endcsname

```

```

1111      {\std@math@comma}%
1112 \fi
1113 }
1114 \ifLaTeXe
1115 \AtBeginDocument{\@ifpackageloaded{icomma}%
1116                  {\FB@icommatrue}%
1117                  {\addto\noextrasfrench{\std@math@comma}%
1118                  \ifdefined\noextrasacadian
1119                  \addto\noextrasacadian{\std@math@comma}%
1120                  \fi
1121                  }%
1122 }
1123 \else
1124 \addto\noextrasfrench{\std@math@comma}
1125 \fi

```

\nombre The command `\nombre` is now borrowed from `numprint.sty` for LaTeX2e. There is no point to maintain the former tricky code when a package is dedicated to do the same job and more. For Plain based formats, `\nombre` no longer formats numbers, it prints them as is and issues a warning about the change.
Fake command `\nombre` for Plain based formats, warning users of babel-french v. 1.x. about the change:

```

1126 \newcommand*{\nombre}[1]{\{#1\}\fb@warning{*** \noexpand\nombre
1127                               no longer formats numbers\string! ***}}

```

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

```

1128 \ifFB@luatex@punct
1129 \activate@luatexpunct
1130 \fi
1131 \let\FBstop@here\relax
1132 \def\FBclean@on@exit{%
1133 \let\ifLaTeXe\undefined
1134 \let\LaTeXetrue\undefined
1135 \let\LaTeXefalse\undefined
1136 \let\FB@llc\loadlocalcfg
1137 \let\loadlocalcfg@gobble}
1138 \ifx\magnification\@undefined
1139 \else
1140 \def\FBstop@here{%
1141 \FBclean@on@exit
1142 \ldf@finish\CurrentOption
1143 \let\loadlocalcfg\FB@llc
1144 \endinput}
1145 \fi
1146 \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.

```

1147 \renewcommand*{\nombre}[1]{\Warning@nombre{#1}}
1148 \newcommand*{\Warning@nombre}[1]{%
1149   \ifdefined\numprint
1150     \numprint{#1}%
1151   \else
1152     \PackageWarning{french.ldf}{%
1153       \protect\nombre\space now relies on package numprint.sty,%
1154       \MessageBreak add \protect
1155       \usepackage[autolanguage]{numprint},\MessageBreak
1156       see file numprint.pdf for more options.\MessageBreak
1157       \protect\nombre\space called}%
1158     \global\let\Warning@nombre\relax
1159     {#1}%
1160   \fi
1161 }

1162 \newcommand*{\FBthousandsep}{\kern \fontdimen2\font \relax}

```

2.7 Caption names

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

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

Let's give a chance to a class or a package read before babel-french to define \FBfigtabshape as \relax, otherwise \FBfigtabshape will be defined as \scshape (can be changed with \frenchsetup{SmallCapsFigTabCaptions=false}).

```

1163 \providecommand*{\FBfigtabshape}{\scshape}

```

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

```

1164 \StartBabelCommands*{\BabelLanguages}{captions}
1165   [unicode, fontenc=TU EU1 EU2, charset=utf8]
1166   \SetString{\refname}{Références}
1167   \SetString{\abstractname}{Résumé}
1168   \SetString{\prefacename}{Préface}
1169   \SetString{\contentsname}{Table des matières}
1170   \SetString{\ccname}{Copie à }
1171   \SetString{\proofname}{Démonstration}
1172   \SetString{\partfirst}{Première}
1173   \SetString{\partsecond}{Deuxième}
1174   \SetStringLoop{ordinal#1}{%
1175     \frenchpartfirst,\frenchpartsecond,Troisième,Quatrième,%
1176     Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,Onzième,%
1177     Douzième,Treizième,Quatorzième,Quinzième,Seizième,%
1178     Dix-septième,Dix-huitième,Dix-neuvième,Vingtième}
1179 \StartBabelCommands*{\BabelLanguages}{captions}
1180   \SetString{\refname}{R\ 'ef\ 'erences}
1181   \SetString{\abstractname}{R\ 'esum\ 'e}

```



```

1182 \SetString{\bibname}{Bibliographie}
1183 \SetString{\prefacename}{Préface}
1184 \SetString{\chaptername}{Chapitre}
1185 \SetString{\appendixname}{Annexe}
1186 \SetString{\contentsname}{Table des matières}
1187 \SetString{\listfigurename}{Table des figures}
1188 \SetString{\listtablename}{Liste des tableaux}
1189 \SetString{\indexname}{Index}
1190 \SetString{\figurename}{\FBfigtabshape Figure}
1191 \SetString{\tablename}{\FBfigtabshape Table}
1192 \SetString{\pagename}{page}
1193 \SetString{\seename}{voir}
1194 \SetString{\alsoname}{voir aussi}
1195 \SetString{\enclname}{P.-J. }
1196 \SetString{\ccname}{Copie \a }
1197 \SetString{\headtoname}{}
1198 \SetString{\proofname}{Démonstration}
1199 \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`.

```

1200 \SetString{\partfirst}{Première}
1201 \SetString{\partsecond}{Deuxième}
1202 \SetString{\partnameord}{partie}
1203 \SetStringLoop{ordinal#1}{%
1204   \partfirst,\partsecond,Troisième,Quatrième,%
1205   Cinquième,Sixième,Septième,Huitième,Neuvième,Dixième,%
1206   Onzième,Douzième,Treizième,Quatorzième,Quinzième,%
1207   Seizième,Dix-septième,Dix-huitième,Dix-neuvième,%
1208   Vingtième}
1209 \AfterBabelCommands{%
1210   \DeclareRobustCommand*\FB@emptypart{\def\thepart{}}%
1211   \DeclareRobustCommand*\FB@partname{%
1212     \ifFBPartNameFull
1213       \csname ordinal\romannumeral\value{part}\endcsname\space
1214       \partnameord\FB@emptypart
1215     \else
1216       Partie%
1217     \fi}%
1218   }
1219   \SetString{\partname}{\FB@partname}
1220 \EndBabelCommands

```

2.8 Figure and table captions

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

```

1221 \newcommand{\FBWarning}[1]{\PackageWarning{french.ldf}{#1}}

```

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

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

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

```

1222 \bgroup
1223 \catcode': =12 \catcode'> =12 \relax
1224 \long\gdef\STD@makecaption#1#2{%
1225   \vskip\abovcaptionskip
1226   \sbox\@tempboxa{#1: #2}%
1227   \ifdim \wd\@tempboxa >\hsize
1228     #1: #2\par
1229   \else
1230     \global \@minipagefalse
1231     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1232   \fi
1233   \vskip\belowcaptionskip}
1234 \egroup

```

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

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

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

Enable the standard warning only if high punctuation is active.

```

1235 \newif\if@FBwarning@capsep
1236 \ifFB@active@punct\@FBwarning@capseptrue\fi
1237 \newcommand*\CaptionSeparator{\space\textendash\space}
1238 \def\FBCaption@Separator{: }
1239 \long\def\FB@makecaption#1#2{%
1240   \vskip\abovcaptionskip
1241   \sbox\@tempboxa{#1\FBCaption@Separator #2}%
1242   \ifdim \wd\@tempboxa >\hsize
1243     #1\FBCaption@Separator #2\par
1244   \else

```

```

1245 \global \@minipagefalse
1246 \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
1247 \fi
1248 \vskip\belowcaptionskip}

```

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

```

1249 \@ifclassloaded{acmart}{\@FBwarning@capsepfalse}{}
1250 \@ifclassloaded{amsart}{\@FBwarning@capsepfalse}{}
1251 \@ifclassloaded{amsbook}{\@FBwarning@capsepfalse}{}
1252 \@ifclassloaded{amstdx}{\@FBwarning@capsepfalse}{}
1253 \@ifclassloaded{amslatex}{\@FBwarning@capsepfalse}{}
1254 \@ifclassloaded{amproc}{\@FBwarning@capsepfalse}{}
1255 \@ifclassloaded{smfart}{\@FBwarning@capsepfalse}{}
1256 \@ifclassloaded{smfbook}{\@FBwarning@capsepfalse}{}

```

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

```

1257 \newif\ifFB@koma
1258 \@ifclassloaded{memoir}{\@FBwarning@capsepfalse}{}
1259 \@ifclassloaded{scrartcl}{\@FBwarning@capsepfalse\FB@komatrue}{}
1260 \@ifclassloaded{scrbook}{\@FBwarning@capsepfalse\FB@komatrue}{}
1261 \@ifclassloaded{scrreprt}{\@FBwarning@capsepfalse\FB@komatrue}{}

```

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

```

1262 \@ifclassloaded{beamer}{\@FBwarning@capsepfalse}{}
1263 \ifdefined\@makecaption\else\FB@warning@capsepfalse\fi

```

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

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

```

1264 \newcounter{FBcaption@count}
1265 \@ifpackageloaded{caption3}{\addtocounter{FBcaption@count}{4}}{}
1266 \@ifpackageloaded{subcaption}{\addtocounter{FBcaption@count}{2}}{}
1267 \@ifpackageloaded{floatrow}{\stepcounter{FBcaption@count}}{}

```

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

```

1268 \AtBeginDocument{%
1269 \ifx\@makecaption\STD@makecaption
1270 \global\let\@makecaption\FB@makecaption

```

If `OldFigTabCaptions=true`, do not overwrite \FBCaption@Separator (already saved as ':' for other languages and set to \CaptionSeparator by \extrasfrench

when French is the main language); otherwise add a space before the ‘:’ in French in order to avoid problems when `AutoSpacePunctuation=false`.

```

1271 \ifFBoldFigTabCaptions
1272 \else
1273 \def\FBCaption@Separator{\ifFBfrench\space\fi : }%
1274 \fi
1275 \ifFBCustomiseFigTabCaptions
1276 \ifFB@mainlanguage@FR
1277 \def\FBCaption@Separator{\CaptionSeparator}%
1278 \fi
1279 \fi
1280 \@FBwarning@capsepfalse
1281 \fi

```

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

```

1282 \@ifpackageloaded{caption3}{%
1283 \ifnum\value{FBCaption@count}=0 \@FBwarning@capsepfalse\fi
1284 }{}%
1285 \if@FBwarning@capsep
1286 \ifnum\value{FBCaption@count}>0

```

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

```

1287 \FBWarning
1288 {Figures' and tables' captions might look like\MessageBreak
1289 'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1290 If you have loaded any of the packages caption,\MessageBreak
1291 subcaption or floatrow BEFORE babel/french,\MessageBreak
1292 please move them AFTER babel/french.\MessageBreak
1293 If one of them is loaded by your class,\MessageBreak
1294 you can still add AFTER babel/french\MessageBreak
1295 \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1296 \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1297 ... live with it; reported}%
1298 \else

```

caption3.sty hasn't been loaded at all.

```

1299 \FBWarning
1300 {Figures' and tables' captions might look like\MessageBreak
1301 'Figure 1:' in French instead of 'Figure 1 :'.\MessageBreak
1302 If it happens, see your class documentation to\MessageBreak
1303 fix this issue or add AFTER babel/french\MessageBreak
1304 \protect\usepackage[labelsep=period]{caption} or\MessageBreak
1305 \protect\usepackage[labelsep=endash]{caption} or\MessageBreak
1306 or ... live with it; reported}%
1307 \fi
1308 \fi
1309 \let\FB@makecaption\relax
1310 \let\STD@makecaption\relax
1311 }

```

2.9 Dots...

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

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

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

```
1312 \iffBUnicode
1313   \let\FBtextellipsis\textellipsis
1314 \else
1315   \DeclareTextSymbol{\FBtextellipsis}{LY1}{133}
1316   \DeclareTextCommandDefault{\FBtextellipsis}{%
1317     .\kern\fontdimen3\font.\kern\fontdimen3\font.\xspace}
1318 \fi
```

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

```
1319 \newcommand*{\Tdots@}{\@xp\textellipsis}
1320 \newcommand*{\Mdots@}{\@xp\mdots@}
1321 \AtBeginDocument{\DeclareRobustCommand*{\dots}{\relax
1322   \csname\ifmmode M\else T\fi dots@endcsname}%
1323   \ifdefined\@xp\else\let\@xp\relax\fi
1324   \ifdefined\mdots@\else\let\Mdots@\mathellipsis\fi
1325 }
1326 \def\bbl@frenchdots{\babel@save\Tdots@ \let\Tdots@\FBtextellipsis}
1327 \addto\extrasfrench{\bbl@frenchdots}
```

2.10 More checks about packages' loading order

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

```
1328 \iffB@active@punct
1329   \@ifpackageloaded{listings}
1330     {\AtBeginDocument{%
1331       \FBWarning{Please load the "listings" package\MessageBreak
1332         AFTER babel/french; reported}}}%
1333     }{}
1334 \fi
```

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

```
1335 \newif\iffBwarning@natbib
```

```

1336 \ifFB@active@punct
1337   \@ifpackageloaded{natbib}{\@FBwarning@natbibtrue}
1338 \fi
1339 \AtBeginDocument{%
1340   \if@FBwarning@natbib
1341     \@ifpackageloaded{natbib}{\@FBwarning@natbibfalse}%
1342   \fi
1343   \if@FBwarning@natbib
1344     \FBWarning{Please load the "natbib" package\MessageBreak
1345               BEFORE babel/french; reported}%
1346   \fi
1347 }

```

Package beamerarticle should be loaded before babel-french to avoid list's conflicts, see p. 56.

```

1348 \newif\if@FBwarning@beamerarticle
1349 \@ifpackageloaded{beamerarticle}{\@FBwarning@beamerarticlettrue}
1350 \AtBeginDocument{%
1351   \if@FBwarning@beamerarticle
1352     \@ifpackageloaded{beamerarticle}{\@FBwarning@beamerarticlefalse}%
1353   \fi
1354   \if@FBwarning@beamerarticle
1355     \FBWarning{Please load the "beamerarticle" package\MessageBreak
1356               BEFORE babel/french; reported}%
1357   \fi
1358 }

```

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.

```

1360 \newcommand*\frenchsetup[1]{%
1361   \setkeys{FB}{#1}%
1362 }%
1363 \@onlypreamble\frenchsetup

```

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

```
1364 \let\frenchbsetup\frenchsetup
1365 \@onlypreamble\frenchbsetup
```

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

```
1366 \newif\ifFBShowOptions
1367 \newif\ifFBStandardLayout      \FBStandardLayouttrue
1368 \newif\ifFBGlobalLayoutFrench \FBGlobalLayoutFrenchtrue
1369 \newif\ifFBReduceListSpacing
1370 \newif\ifFBStandardListSpacing \FBStandardListSpacingtrue
1371 \newif\ifFBListOldLayout
1372 \newif\ifFBListItemsAsPar
1373 \newif\ifFBCompactItemize
1374 \newif\ifFBStandardItemizeEnv   \FBStandardItemizeEnvtrue
1375 \newif\ifFBStandardEnumerateEnv \FBStandardEnumerateEnvtrue
1376 \newif\ifFBStandardItemLabels   \FBStandardItemLabelstrue
1377 \newif\ifFBStandardLists        \FBStandardListstrue
1378 \newif\ifFBIndentFirst
1379 \newif\ifFBFrenchFootnotes
1380 \newif\ifFBAutoSpaceFootnotes
1381 \newif\ifFBOriginalTypewriter
1382 \newif\ifFBThinColonSpace
1383 \newif\ifFBThinSpaceInFrenchNumbers
1384 \newif\ifFBFrenchSuperscripts   \FBFrenchSuperscriptstrue
1385 \newif\ifFBLowercaseSuperscripts \FBLowercaseSuperscriptstrue
1386 \newif\ifFBPartNameFull         \FBPartNameFulltrue
1387 \newif\ifFBCustomiseFigTabCaptions
1388 \newif\ifFBOldFigTabCaptions
1389 \newif\ifFBSmallCapsFigTabCaptions \FBSmallCapsFigTabCaptionstrue
1390 \newif\ifFBSuppressWarning
1391 \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`.

```
1392 \ifFB@koma
1393   \ifdefined\partformat
1394     \def\FB@partformat@fix{%
1395       \ifFBPartNameFull
1396         \babel@save\partformat
1397         \renewcommand*{\partformat}{\partname}%
1398       \fi}
1399     \addto\extrasfrench{\FB@partformat@fix}%
1400   \fi
1401 \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.

```

1402 \def\FB@french{french}
1403 \def\FB@acadian{acadian}
1404 \newif\ifFB@mainlanguage@FR
1405 \AtEndOfPackage{%
1406   \ifx\bbbl@main@language\FB@french \FB@mainlanguage@FRtrue
1407   \else \ifx\bbbl@main@language\FB@acadian \FB@mainlanguage@FRtrue \fi
1408   \fi
1409   \ifFB@mainlanguage@FR
1410     \FBGlobalLayoutFrenchtrue
1411     \@ifclassloaded{beamer}%
1412       {\PackageInfo{french.ldb}{%
1413         No list customisation for the beamer class,%
1414         \MessageBreak reported}}%
1415       {\@ifpackageloaded{beamerarticle}%
1416        {\FBStandardItemLabelsfalse
1417         \FBStandardListSpacingfalse
1418         \PackageInfo{french.ldb}{%
1419           Minimal list customisation for the beamerarticle%
1420           \MessageBreak package; reported}}}%

```

Otherwise customise lists “à la française”:

```

1421     {\FBStandardListSpacingfalse
1422     \FBStandardItemizeEnvfalse
1423     \FBStandardEnumerateEnvfalse
1424     \FBStandardItemLabelsfalse}%
1425   }
1426   \FBIndentFirsttrue
1427   \FBFrenchFootnotesttrue
1428   \FBAutoSpaceFootnotesttrue
1429   \FBCustomiseFigTabCaptionstrue
1430 \else
1431   \FBGlobalLayoutFrenchfalse
1432 \fi

```

babel-french being an option of babel, it cannot load a package (keyval) while french.ldb is read, so we defer the loading of keyval and the options setup at the end of babel’s loading.

```

1433 \RequirePackage{keyval}%
1434 \define@key{FB}{ShowOptions}[true]%
1435   {\csname FBShowOptions#1\endcsname}%

```

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

```

1436 \define@key{FB}{StandardLayout}[true]%
1437   {\ifFB@mainlanguage@FR
1438     \csname FBStandardLayout#1\endcsname

```



```

1439         \else
1440             \PackageWarning{french.ldf}%
1441                 {Option ‘StandardLayout’ skipped:\MessageBreak
1442                     French is *not* babel’s last option.\MessageBreak
1443                     Reported}%
1444         \fi
1445     \ifFBStandardLayout
1446         \FBStandardListSpacingtrue
1447         \FBStandardItemizeEnvtrue
1448         \FBStandardItemLabelstrue
1449         \FBStandardEnumerateEnvtrue
1450         \FBIndentFirstfalse
1451         \FBFrenchFootnotesfalse
1452         \FBAutoSpaceFootnotesfalse
1453         \FBGlobalLayoutFrenchfalse
1454     \else
1455         \FBStandardListSpacingfalse
1456         \FBStandardItemizeEnvfalse
1457         \FBStandardItemLabelsfalse
1458         \FBStandardEnumerateEnvfalse
1459         \FBIndentFirsttrue
1460         \FBFrenchFootnotestruetrue
1461         \FBAutoSpaceFootnotestruetrue
1462     \fi}%
1463 \define@key{FB}{GlobalLayoutFrench}[true]%
1464 {\ifFB@mainlanguage@FR
1465     \csname FBGlobalLayoutFrench#1\endcsname
1466 \else
1467     \PackageWarning{french.ldf}%
1468         {Option ‘GlobalLayoutFrench’ skipped:\MessageBreak
1469             French is *not* babel’s last option.\MessageBreak
1470             Reported}%
1471 \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.

```

1472 \define@key{FB}{ReduceListSpacing}[true]%
1473 {\csname FBReduceListSpacing#1\endcsname
1474     \ifFBReduceListSpacing \FBStandardListSpacingfalse
1475     \else \FBStandardListSpacingtrue\fi
1476 }%
1477 \define@key{FB}{StandardListSpacing}[true]%
1478 {\csname FBStandardListSpacing#1\endcsname}%
1479 \define@key{FB}{ListOldLayout}[true]%
1480 {\csname FBListOldLayout#1\endcsname
1481     \ifFBListOldLayout
1482         \FBStandardEnumerateEnvtrue
1483         \renewcommand*{\FrenchLabelItem}{\textendash}%
1484     \fi}%
1485 \define@key{FB}{CompactItemize}[true]%

```

```

1486         {\csname FBCompactItemize#1\endcsname
1487         \ifFBCompactItemize
1488             \FBStandardItemizeEnvfalse
1489             \FBStandardEnumerateEnvfalse
1490         \else
1491             \FBStandardItemizeEnvtrue
1492             \FBStandardEnumerateEnvtrue
1493         \fi}%
1494 \define@key{FB}{StandardItemizeEnv}[true]%
1495     {\csname FBStandardItemizeEnv#1\endcsname}%
1496 \define@key{FB}{StandardEnumerateEnv}[true]%
1497     {\csname FBStandardEnumerateEnv#1\endcsname}%
1498 \define@key{FB}{StandardItemLabels}[true]%
1499     {\csname FBStandardItemLabels#1\endcsname}%
1500 \define@key{FB}{ItemLabels}%
1501     {\renewcommand*\FrenchLabelItem{#1}}%
1502 \define@key{FB}{ItemLabeli}%
1503     {\renewcommand*\Frlabelitemi{#1}}%
1504 \define@key{FB}{ItemLabelii}%
1505     {\renewcommand*\Frlabelitemii{#1}}%
1506 \define@key{FB}{ItemLabeliii}%
1507     {\renewcommand*\Frlabelitemiii{#1}}%
1508 \define@key{FB}{ItemLabeliv}%
1509     {\renewcommand*\Frlabelitemiv{#1}}%
1510 \define@key{FB}{StandardLists}[true]%
1511     {\csname FBStandardLists#1\endcsname
1512     \ifFBStandardLists
1513         \FBStandardListSpacingtrue
1514         \FBStandardItemizeEnvtrue
1515         \FBStandardEnumerateEnvtrue
1516         \FBStandardItemLabelstrue
1517     \else
1518         \FBStandardListSpacingfalse
1519         \FBStandardItemizeEnvfalse
1520         \FBStandardEnumerateEnvfalse
1521         \FBStandardItemLabelsfalse
1522     \fi}%
1523 \define@key{FB}{ListItemsAsPar}[true]%
1524     {\csname FBListItemsAsPar#1\endcsname}%
1525 \define@key{FB}{IndentFirst}[true]%
1526     {\csname FBIndentFirst#1\endcsname}%
1527 \define@key{FB}{FrenchFootnotes}[true]%
1528     {\csname FBFrenchFootnotes#1\endcsname}%
1529 \define@key{FB}{AutoSpaceFootnotes}[true]%
1530     {\csname FBAutoSpaceFootnotes#1\endcsname}%
1531 \define@key{FB}{AutoSpacePunctuation}[true]%
1532     {\csname FBAutoSpacePunctuation#1\endcsname}%
1533 \define@key{FB}{OriginalTypewriter}[true]%
1534     {\csname FBOriginalTypewriter#1\endcsname}%
1535 \define@key{FB}{ThinColonSpace}[true]%
1536     {\csname FBThinColonSpace#1\endcsname

```

```

1537         \ifFBThinColonSpace
1538             \renewcommand*{\FBcolonspace}{\FBthinspace}%
1539         \fi}%
1540 \define@key{FB}{ThinSpaceInFrenchNumbers}[true]%
1541     {\csname FBThinSpaceInFrenchNumbers#1\endcsname}%
1542 \define@key{FB}{FrenchSuperscripts}[true]%
1543     {\csname FBFrenchSuperscripts#1\endcsname}%
1544 \define@key{FB}{LowercaseSuperscripts}[true]%
1545     {\csname FBLowercaseSuperscripts#1\endcsname}%
1546 \define@key{FB}{PartNameFull}[true]%
1547     {\csname FBPartNameFull#1\endcsname}%
1548 \define@key{FB}{CustomiseFigTabCaptions}[true]%
1549     {\csname FBCustomiseFigTabCaptions#1\endcsname}%
1550 \define@key{FB}{OldFigTabCaptions}[true]%
1551     {\csname FBOldFigTabCaptions#1\endcsname}%
1552     \ifFBOldFigTabCaptions
1553         \def\FB@capsep@fix{\babel@save\FBCaption@Separator
1554             \def\FBCaption@Separator{\CaptionSeparator}}%
1555         \addto\extrasfrench{\FB@capsep@fix}%
1556         \ifdefined\extrasacadian
1557             \addto\extrasacadian{\FB@capsep@fix}%
1558         \fi
1559     \fi}%
1560 \define@key{FB}{SmallCapsFigTabCaptions}[true]%
1561     {\csname FBSmallCapsFigTabCaptions#1\endcsname}%
1562     \ifFBSmallCapsFigTabCaptions
1563         \let\FBfigtabshape\scshape
1564     \else
1565         \let\FBfigtabshape\relax
1566     \fi}%
1567 \define@key{FB}{SuppressWarning}[true]%
1568     {\csname FBSuppressWarning#1\endcsname}%
1569     \ifFBSuppressWarning
1570         \renewcommand{\FBWarning}[1]{}%
1571     \fi}%

```

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

```

1572 \define@key{FB}{INGuillSpace}[true]%
1573     {\csname FBINGuillSpace#1\endcsname}%
1574     \ifFBINGuillSpace
1575         \renewcommand*{\FBguillspace}{\space}%
1576     \fi}%
1577 \define@key{FB}{InnerGuillSingle}[true]%
1578     {\csname FBInnerGuillSingle#1\endcsname}%
1579 \define@key{FB}{EveryParGuill}[open]%
1580     {\expandafter\let\expandafter
1581         \FBEveryparguill\csname FBguill#1\endcsname
1582         \if\FBEveryparguill\FBguillopen
1583         \else\if\FBEveryparguill\FBguillclose
1584         \else\if\FBEveryparguill\FBguillnone

```

```

1585             \else
1586                 \let\FBeveryparguill\FBguillopen
1587                 \FBWarning{Wrong value for 'EveryParGuill':
1588                     try 'open',\MessageBreak
1589                     'close' or 'none'. Reported}%
1590             \fi
1591         \fi
1592     \fi}%
1593 \define@key{FB}{EveryLineGuill}[open]%
1594     {\ifFB@luatex@punct
1595         \expandafter\let\expandafter
1596             \FBeverylineguill\csname FBguill#1\endcsname
1597         \ifx\FBeverylineguill\FBguillopen
1598             \else\ifx\FBeverylineguill\FBguillclose
1599                 \else\ifx\FBeverylineguill\FBguillnone
1600                     \else
1601                         \let\FBeverylineguill\FBguillnone
1602                         \FBWarning{Wrong value for 'EveryLineGuill':
1603                             try 'open',\MessageBreak
1604                             'close' or 'none'. Reported}%
1605                     \fi
1606                 \fi
1607             \fi
1608         \else
1609             \FBWarning{Option 'EveryLineGuill' skipped:%
1610                 \MessageBreak this option is for
1611                 LuaTeX *only*.\MessageBreak Reported}%
1612         \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.

```

1613 \define@key{FB}{UnicodeNoBreakSpaces}[true]%
1614     {\ifFB@luatex@punct
1615         \csname FBucsNBSP#1\endcsname
1616         \ifFBucsNBSP \FB@ucsNBSP=1 \fi
1617     \else
1618         \FBWarning{Option 'UnicodeNoBreakSpaces' skipped:%
1619             \MessageBreak this option is for
1620             LuaTeX *only*.\MessageBreak Reported}%
1621     \fi
1622 }%

```

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

```
1623 \def\FB@parse#1#2\endparse{\def\FB@second{#2}}%
1624 \define@key{FB}{og}%
1625     {\ifFBunicode
```

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

```
1626     \ifFB@luatex@punct
1627     \FB@addGUILspace=1 \relax
1628     \fi
```

then with XeTeX it is a bit more tricky:

```
1629     \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).

```
1630     \XeTeXcharclass"13 = \FB@guilo
1631     \XeTeXcharclass"AB = \FB@guilo
1632     \XeTeXcharclass"A0 = \FB@guilnul
1633     \XeTeXcharclass"202F = \FB@guilnul
1634     \fi
```

Issue a warning with older Unicode engines requiring active characters.

```
1635     \ifFB@active@punct
1636     \FBWarning{Option og=< not supported with this version
1637         of\MessageBreak LuaTeX/XeTeX; reported}%
1638     \fi
1639     \else
```

This is for conventional TeX engines:

```
1640     \newcommand*{\FB@@og}{%
1641     \ifFBfrench
1642     \ifFB@spacing\FB@og\ignorespaces
1643     \else\guillemotleft
1644     \fi
1645     \else\guillemotleft\fi}%
1646     \AtBeginDocument{%
1647     \ifdefined\uc@dccl
```

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

```
1648     \uc@dccl{171}{default}}{\FB@@og}%
1649     \else
```

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

```
1650     \FB@parse#1\endparse
1651     \ifx\FB@second\@empty
```

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

```

1652             \ifdefined\mule@def
1653             \mule@def{11}{\FB@@og}%
1654         \else
1655             \ifdefined\DeclareInputText
1656             \@tempcnta'#1\relax
1657             \DeclareInputText{\the\@tempcnta}{\FB@@og}%
1658         \else

```

Package inputenc not loaded, no way...

```

1659             \FBWarning{Option 'og' requires package
1660                 inputenc;\MessageBreak reported}%
1661         \fi
1662     \fi
1663 \else

```

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

```

1664             \DeclareUnicodeCharacter{00AB}{\FB@@og}%
1665         \fi
1666     \fi}%
1667 \fi
1668 }%

```

Same code for the closing quote.

```

1669 \define@key{FB}{fg}{%
1670     {\ifFBunicode
1671         \ifFB@luatex@punct
1672             \FB@addGUIlspace=1 \relax
1673         \fi
1674         \ifFB@xetex@punct
1675             \XeTeXcharclass"14 = \FB@guilf
1676             \XeTeXcharclass"BB = \FB@guilf
1677             \XeTeXcharclass"A0 = \FB@guilnul
1678             \XeTeXcharclass"202F = \FB@guilnul
1679         \fi
1680         \ifFB@active@punct
1681             \FBWarning{Option fg=> not supported with this version
1682                 of\MessageBreak LuaTeX/XeTeX; reported}%
1683         \fi
1684     \else
1685         \newcommand*{\FB@@fg}{%
1686             \ifFBfrench
1687                 \ifFB@spacing\FB@fg
1688                 \else\guillemotright
1689                 \fi
1690             \else\guillemotright\fi}%
1691     \AtBeginDocument{%
1692         \ifdefined\uc@dclc
1693         \uc@dclc{187}{default}{\FB@@fg}%
1694         \else
1695             \FB@parse#1\endparse

```

```

1696         \ifx\FB@second\@empty
1697         \ifdefined\mule@def
1698             \mule@def{27}{\FB@@fg}}%
1699         \else
1700             \ifdefined\DeclareInputText
1701                 \@tempcnta'#1\relax
1702                 \DeclareInputText{\the\@tempcnta}{\FB@@fg}%
1703             \else
1704                 \FBWarning{Option 'fg' requires package
1705                     inputenc;\MessageBreak reported}%
1706             \fi
1707         \fi
1708     \else
1709         \DeclareUnicodeCharacter{00BB}{\FB@@fg}%
1710     \fi
1711 \fi}%
1712 \fi
1713 }%
1714 }

```

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

```

1715 \newcommand*\FBprocess@options{%

```

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

```

1716 \@ifpackageloaded{enumitem}{%
1717     \ifFBStandardItemizeEnv
1718     \else
1719         \FBStandardItemizeEnvtrue
1720         \PackageInfo{french.ldf}%
1721             {Setting StandardItemizeEnv=true for\MessageBreak
1722             compatibility with enumitem package,\MessageBreak
1723             reported}%
1724     \fi
1725     \ifFBStandardEnumerateEnv
1726     \else
1727         \FBStandardEnumerateEnvtrue
1728         \PackageInfo{french.ldf}%
1729             {Setting StandardEnumerateEnv=true for\MessageBreak
1730             compatibility with enumitem package,\MessageBreak
1731             reported}%
1732     \fi}{}%
1733 \@ifpackageloaded{paralist}{%
1734     \ifFBStandardItemizeEnv
1735     \else
1736         \FBStandardItemizeEnvtrue

```

```

1737     \PackageInfo{french.ldf}%
1738     {Setting StandardItemizeEnv=true for\MessageBreak
1739     compatibility with paralist package,\MessageBreak
1740     reported}%
1741   \fi
1742   \ifFBStandardEnumerateEnv
1743   \else
1744     \FBStandardEnumerateEnvtrue
1745     \PackageInfo{french.ldf}%
1746     {Setting StandardEnumerateEnv=true for\MessageBreak
1747     compatibility with paralist package,\MessageBreak
1748     reported}%
1749   \fi}{}%
1750 \@@ifpackageloaded{enumerate}{%
1751   \ifFBStandardEnumerateEnv
1752   \else
1753     \FBStandardEnumerateEnvtrue
1754     \PackageInfo{french.ldf}%
1755     {Setting StandardEnumerateEnv=true for\MessageBreak
1756     compatibility with enumerate package,\MessageBreak
1757     reported}%
1758   \fi}{}%

```

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

```

1759 \def\FB@ufl{\update@frenchlists}
1760 \ifFB@mainlanguage@FR
1761   \update@frenchlists
1762 \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.

```

1763 \ifFBAutoSpacePunctuation
1764   \autospace@beforeFDP
1765 \else
1766   \noautospace@beforeFDP
1767 \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.

```

1768 \ifFBOriginalTypewriter
1769 \else
1770   \let\ttfamilyORI\ttfamily
1771   \let\rmfamilyORI\rmfamily
1772   \let\sffamilyORI\sffamily
1773   \let\ttfamily\ttfamilyFB
1774   \let\rmfamily\rmfamilyFB

```



```

1775 \let\sfffamily\sfffamilyFB
1776 \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.

```

1777 \@ifpackageloaded{numprint}%
1778   {\ifnprt@autolanguage
1779     \providecommand*\npstylefrench{}}%
1780     \ifFBThinSpaceInFrenchNumbers
1781       \renewcommand*\FBthousandsep{\,}%
1782     \fi
1783     \g@addto@macro\npstylefrench{\npthousandsep\FBthousandsep}%
1784   \fi
1785 }{}%

```

[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.

```

1786 \ifFBFrenchSuperscripts
1787   \DeclareRobustCommand*\up*{\@ifstar\FB@up@fake}\fup}%
1788 \else
1789   \DeclareRobustCommand*\up*{\@ifstar\FB@up@fake}%
1790                               {\textsuperscript}}%
1791 \fi

```

[LowercaseSuperscripts](#): if [false](#) \FB@lc is redefined to do nothing.

```

1792 \ifFBLowercaseSuperscripts
1793 \else
1794   \renewcommand*\FB@lc[1]{##1}%
1795 \fi

```

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

```

1796 \ifFBCustomiseFigTabCaptions
1797   \ifFB@koma
1798     \renewcommand*\captionformat{\CaptionSeparator}%
1799   \fi
1800   \@ifclassloaded{memoir}%
1801     {\captiondelim{\CaptionSeparator}}{}%
1802   \@ifclassloaded{beamer}%
1803     {\defbeamertemplate{caption label separator}{FBcustom}{%
1804       \CaptionSeparator}%
1805     \setbeamertemplate{caption label separator}{FBcustom}}{}%
1806 \else

```

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

```

1807   \ifFB@koma
1808     \renewcommand*\captionformat{\autospace@beforeFDP : }%
1809   \fi

```

```

1810 \@ifclassloaded{memoir}%
1811     {\captiondelim{{\autospace@beforeFDP : }}%
1812     }{}%
1813 \@ifclassloaded{beamer}%
1814     {\defbeamertemplate{caption label separator}{FBcolon}{%
1815         {\autospace@beforeFDP : }}%
1816         \setbeamertemplate{caption label separator}{FBcolon}%
1817     }{}%
1818 \fi

```

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

```

1819 \ifFBShowOptions
1820     \GenericWarning{* }{%
1821         ***** List of possible options for babel-french *****\MessageBreak
1822         [Default values between brackets when french is loaded *LAST*]%
1823         \MessageBreak
1824         ShowOptions=true [false]\MessageBreak
1825         StandardLayout=true [false]\MessageBreak
1826         GlobalLayoutFrench=false [true]\MessageBreak
1827         PartNameFull=false [true]\MessageBreak
1828         IndentFirst=false [true]\MessageBreak
1829         ListItemsAsPar=true [false]\MessageBreak
1830         StandardListSpacing=true [false]\MessageBreak
1831         StandardItemizeEnv=true [false]\MessageBreak
1832         StandardEnumerateEnv=true [false]\MessageBreak
1833         StandardItemLabels=true [false]\MessageBreak
1834         ItemLabels=\textendash, \textbullet,
1835         \protect\ding{43},... [\textendash]\MessageBreak
1836         ItemLabeli=\textendash, \textbullet,
1837         \protect\ding{43},... [\textendash]\MessageBreak
1838         ItemLabelii=\textendash, \textbullet,
1839         \protect\ding{43},... [\textendash]\MessageBreak
1840         ItemLabeliii=\textendash, \textbullet,
1841         \protect\ding{43},... [\textendash]\MessageBreak
1842         ItemLabeliv=\textendash, \textbullet,
1843         \protect\ding{43},... [\textendash]\MessageBreak
1844         StandardLists=true [false]\MessageBreak
1845         ListOldLayout=true [false]\MessageBreak
1846         FrenchFootnotes=false [true]\MessageBreak
1847         AutoSpaceFootnotes=false [true]\MessageBreak
1848         AutoSpacePunctuation=false [true]\MessageBreak
1849         ThinColonSpace=true [false]\MessageBreak
1850         OriginalTypewriter=true [false]\MessageBreak
1851         UnicodeNoBreakSpaces=true [false]\MessageBreak
1852         og= <left quote character>, fg= <right quote character>%
1853         INGuillSpace=true [false]\MessageBreak
1854         EveryParGuill=open, close, none [open]\MessageBreak
1855         EveryLineGuill=open, close, none
1856             [open in LuaTeX, none otherwise]\MessageBreak
1857         InnerGuillSingle=true [false]\MessageBreak
1858         ThinSpaceInFrenchNumbers=true [false]\MessageBreak

```

```

1859 SmallCapsFigTabCaptions=false [true]\MessageBreak
1860 CustomiseFigTabCaptions=false [true]\MessageBreak
1861 OldFigTabCaptions=true [false]\MessageBreak
1862 FrenchSuperscripts=false [true]\MessageBreak
1863 LowercaseSuperscripts=false [true]\MessageBreak
1864 SuppressWarning=true [false]\MessageBreak
1865 \MessageBreak
1866 *****%
1867 \MessageBreak\protect\frenchsetup{ShowOptions}}
1868 \fi
1869 }

```

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.

```

1870 \AtBeginDocument{%
1871   \providecommand*\xspace{\relax}%

```

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

```

1872   \ifdefined\pdfstringdefDisableCommands
1873     \pdfstringdefDisableCommands{%
1874       \let\up\relax
1875       \let\up\relax
1876       \let\degre\textdegree
1877       \let\degres\textdegree
1878       \def\ieme{e\xspace}%
1879       \def\iemes{es\xspace}%
1880       \def\ier{er\xspace}%
1881       \def\iers{ers\xspace}%
1882       \def\iere{re\xspace}%
1883       \def\ieres{res\xspace}%
1884       \def\FrenchEnumerate#1{#1\degre\space}%
1885       \def\FrenchPopularEnumerate#1{#1\degre)\space}%
1886       \def\No{N\degre\space}%
1887       \def\no{n\degre\space}%
1888       \def\Nos{N\degre\space}%
1889       \def\nos{n\degre\space}%
1890       \def\FB@og{\guillemotleft\space}%
1891       \def\FB@fg{\space\guillemotright}%
1892       \def\frquote#1{\FB@og #1\FB@fg}%
1893       \def\at{@}%
1894       \def\circonflexe{\string^}%
1895       \def\tild{\string~}%
1896       \def\boi{\textbackslash}%
1897       \let\bsc\textsc
1898     }%
1899   \fi

```

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

```

1900   \FBprocess@options

```

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

```

1901 \ifFBucsNBSP
1902 \renewcommand*\FBmedkern{\char"202F\relax}%
1903 \renewcommand*\FBthickkern{\char"A0\relax}%
1904 \ifFBThinSpaceInFrenchNumbers
1905 \renewcommand*\FBthousandsep{\char"202F\relax}%
1906 \else
1907 \renewcommand*\FBthousandsep{\char"A0\relax}%
1908 \fi
1909 \fi

```

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

```

1910 \begingroup
1911 \newcommand{\FBOTone}{OT1}%
1912 \ifx\encodingdefault\FBOTone
1913 \FBWarning{OT1 encoding should not be used for French.%
1914 \MessageBreak
1915 Add \protect\usepackage[T1]{fontenc} to the
1916 preamble\MessageBreak of your document; reported}%
1917 \fi
1918 \endgroup
1919 }

```

2.12 French lists

`\listFB` Vertical spacing in lists should be shorter in French texts than the defaults provided by LaTeX. Note that the easy way, just changing values of vertical spacing parameters `\listORI` 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.

```

1920 \let\listORI\list
1921 \let\endlistORI\endlist
1922 \def\FB@listVsettings{%
1923 \setlength{\itemsep}{0.4ex plus 0.2ex minus 0.2ex}%
1924 \setlength{\parsep}{0.4ex plus 0.2ex minus 0.2ex}%
1925 \setlength{\topsep}{0.8ex plus 0.4ex minus 0.4ex}%
1926 \setlength{\partopsep}{0.4ex plus 0.2ex minus 0.2ex}%

```

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

\@tempdima.

```
1927 \@tempdima=\parskip
1928 \addtolength{\topsep}{-\@tempdima}%
1929 \addtolength{\partopsep}{\@tempdima}%
1930 }
1931 \def\listFB#1#2{\listORI{#1}{\FB@listVsettings #2}}
1932 \let\endlistFB\endlist
```

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

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

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

```
\Frlabelitemi 1933 \newcommand*\FrenchLabelItem{\textemdash}
\Frlabelitemii 1934 \newcommand*\Frlabelitemi{\FrenchLabelItem}
\Frlabelitemiii 1935 \newcommand*\Frlabelitemii{\FrenchLabelItem}
\Frlabelitemiv 1936 \newcommand*\Frlabelitemiii{\FrenchLabelItem}
1937 \newcommand*\Frlabelitemiv{\FrenchLabelItem}
```

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

```
1938 \newdimen\listindentFB
1939 \setlength{\listindentFB}{-1pt}
1940 \newdimen\descindentFB
1941 \setlength{\descindentFB}{-1pt}
1942 \newdimen\labelindentFB
1943 \setlength{\labelindentFB}{-1pt}
1944 \newdimen\labelwidthFB
1945 \setlength{\labelwidthFB}{-1pt}
```

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

```
1946 \newdimen\leftmarginFB
1947 \def\FB@listHsettings{%
1948 \ifFBListItemsAsPar
```

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

```

1949 \itemindent=\labelindentFB
1950 \advance\itemindent by \labelwidthFB
1951 \advance\itemindent by \labelsep
1952 \leftmargini\z@
1953 \bbl@for\FB@dp {2, 3, 4, 5, 6}%
1954 {\csname leftmargin\romannumeral\FB@dp\endcsname=\labelindentFB}%
1955 \else

```

Default layout: labels hanging into the left margin.

```

1956 \leftmarginFB=\labelwidthFB
1957 \advance\leftmarginFB by \labelsep
1958 \bbl@for\FB@dp {1, 2, 3, 4, 5, 6}%
1959 {\csname leftmargin\romannumeral\FB@dp\endcsname=\leftmarginFB}%
1960 \advance\leftmargini by \listindentFB
1961 \fi
1962 \leftmargin=\csname leftmargin\ifnum\@listdepth=\@ne i\else
1963                                     ii\fi\endcsname
1964 }

```

`\itemizeFB` New environment for French itemize-lists.

`\FB@itemizesettings` `\FB@itemizesettings` does two things: first suppress all vertical spaces including glue unless option `StandardListSpacing` is set, then set horizontal indentations according to `\FB@listHsettings` unless option `ListOldLayout` is `true` (compatibility with lists up to v. 2.5k).

```

1965 \def\FB@itemizesettings{%
1966   \ifFBStandardListSpacing
1967   \else
1968     \setlength{\itemsep}{\z@}%
1969     \setlength{\parsep}{\z@}%
1970     \setlength{\topsep}{\z@}%
1971     \setlength{\partopsep}{\z@}%
1972     \@tempdima=\parskip
1973     \addtolength{\topsep}{-\@tempdima}%
1974     \addtolength{\partopsep}{\@tempdima}%
1975   \fi
1976   \settowidth{\labelwidth}{\csname\@itemitem\endcsname}%
1977   \ifFBListOldLayout
1978     \setlength{\leftmargin}{\labelwidth}%
1979     \addtolength{\leftmargin}{\labelsep}%
1980     \addtolength{\leftmargin}{\parindent}%
1981   \else
1982     \FB@listHsettings
1983   \fi
1984 }

```

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

```

1985 \def\itemizeFB{%
1986   \ifnum \@itemdepth >\thr@@\toodeep\else
1987     \advance\@itemdepth by \@ne

```

```

1988      \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
1989      \expandafter
1990      \listORI
1991      \csname\@itemitem\endcsname
1992      \FB@itemizesettings
1993      \fi
1994 }
1995 \let\enditemizeFB\endlistORI

1996 \def\setlabelitemsFB{%
1997   \let\labelitemi\Frlabelitemi
1998   \let\labelitemii\Frlabelitemii
1999   \let\labelitemiii\Frlabelitemiii
2000   \let\labelitemiv\Frlabelitemiv
2001   \ifdim\labelwidthFB<\z@
2002     \settowidth{\labelwidthFB}{\FrenchLabelItem}%
2003   \fi
2004 }
2005 \def\setlistindentFB{%
2006   \ifdim\labelindentFB<\z@
2007     \ifdim\parindent=\z@
2008       \setlength{\labelindentFB}{1.5em}%
2009     \else
2010       \setlength{\labelindentFB}{\parindent}%
2011     \fi
2012   \fi
2013   \ifdim\listindentFB<\z@
2014     \ifdim\parindent=\z@
2015       \setlength{\listindentFB}{1.5em}%
2016     \else
2017       \setlength{\listindentFB}{\parindent}%
2018     \fi
2019   \fi
2020   \ifdim\descindentFB<\z@
2021     \ifFBListItemsAsPar
2022       \setlength{\descindentFB}{\labelindentFB}%
2023     \else
2024       \setlength{\descindentFB}{\listindentFB}%
2025     \fi
2026   \fi
2027 }

```

\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.

```

2028 \def\enumerateFB{%
2029   \ifnum \@enumdepth >\thr@@\@toodeep\else
2030     \advance\@enumdepth by \@ne
2031     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
2032     \expandafter

```

```

2033 \list
2034 \csname label\@enumctr\endcsname
2035 {\FB@listHsettings
2036 \usecounter\@enumctr\def\makelabel##1{\hss\llap{##1}}}%
2037 \fi
2038 }
2039 \let\endenumerateFB\endlistORI

```

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

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

```

2040 \def\descriptionFB{%
2041 \list{}{\FB@listHsettings
2042 \labelwidth=\z@
2043 \ifFBListItemsAsPar
2044 \itemindent=\descindentFB
2045 \else
2046 \itemindent=-\leftmargin
2047 \ifnum\@listdepth=1
2048 \ifdim\descindentFB=\z@
2049 \ifdim\listindentFB>\z@
2050 \leftmargini=\listindentFB
2051 \leftmargin=\leftmargini
2052 \itemindent=-\leftmargin
2053 \fi
2054 \else
2055 \advance\itemindent by \descindentFB
2056 \fi
2057 \fi
2058 \fi
2059 \let\makelabel\descriptionlabel}%
2060 }
2061 \let\enddescriptionFB\endlistORI

```

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

```

2062 \def\update@frenchlists{%
2063 \setlistindentFB
2064 \ifFBStandardListSpacing
2065 \else \let\list\listFB \fi
2066 \ifFBStandardItemizeEnv
2067 \else \let\itemize\itemizeFB \fi
2068 \ifFBStandardItemLabels
2069 \else \setlabelitemsFB \fi
2070 \ifFBStandardEnumerateEnv

```



```

2071 \else \let\enumerate\enumerateFB \let\description\descriptionFB \fi
2072 }

```

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. 64.

```

2073 \def\FB@ufl{\relax}
2074 \def\bbl@frenchlistlayout{%
2075   \ifFBGlobalLayoutFrench
2076   \else
2077     \babel@save\list           \babel@save\itemize
2078     \babel@save\enumerate     \babel@save\description
2079     \babel@save\labelitemi    \babel@save\labelitemii
2080     \babel@save\labelitemiii  \babel@save\labelitemiv
2081     \FB@ufl
2082   \fi
2083 }
2084 \addto\extrasfrench{\bbl@frenchlistlayout}

```

2.13 French indentation of sections

`\bbl@frenchindent` In French the first paragraph of each section should be indented, this is another difference with US-English. This is controlled by the flag `\if@afterindent`. We will need to save the value of the flag `\if@afterindent` 'AtBeginDocument' before eventually changing its value.

```

2085 \def\bbl@frenchindent{%
2086   \ifFBGlobalLayoutFrench
2087   \else
2088     \babel@save\@afterindentfalse
2089   \fi
2090   \ifFBIndentFirst
2091     \let\@afterindentfalse\@afterindenttrue
2092     \@afterindenttrue
2093   \fi}
2094 \def\bbl@nonfrenchindent{%
2095   \ifFBGlobalLayoutFrench
2096   \ifFBIndentFirst
2097     \@afterindenttrue
2098   \fi
2099   \fi}
2100 \addto\extrasfrench{\bbl@frenchindent}
2101 \addto\noextrasfrench{\bbl@nonfrenchindent}

```

2.14 Formatting footnotes

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

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

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

```

2102 \AtBeginDocument{\@ifpackageloaded{bigfoot}%
2103                 {\PackageInfo{french.ldf}%
2104                 {bigfoot package in use.\MessageBreak
2105                 babel-french will NOT customise footnotes;%
2106                 \MessageBreak reported}}}%
2107                 {\let\@footnotemarkORI\@footnotemark
2108                 \def\@footnotemarkFB{\leavevmode\unskip\unkern
2109                 \,\@footnotemarkORI}%
2110                 \ifBFAutoSpaceFootnotes
2111                 \let\@footnotemark\@footnotemarkFB
2112                 \fi}%
2113                 }

```

\@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).

```

2114 \newdimen\parindentFFN
2115 \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.

```

2116 \newcommand*{\dotFFN}{.}
2117 \newcommand*{\kernFFN}{\kern .5em}
2118 \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.

```
2119 \ifFB@koma
2120   \let\@makefntextORI\@makefntext
2121   \let\@@makefnmarkORI\@@makefnmark
```

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

```
2122   \deffootnote[\FBfnindent]{0pt}{\parindentFFN}%
2123           {\thefootnotemark\dotFFN\kernFFN}
2124   \let\@makefntextFB\@makefntext
2125   \let\@@makefnmarkFB\@@makefnmark
```

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

```
2126   \deffootnote[\parindentFFN]{0pt}{\parindentFFN}%
2127           {\textsuperscript{\thefootnotemark}}
2128   \let\@makefntextTH\@makefntext
2129   \let\@@makefnmarkTH\@@makefnmark
```

Restore the original definitions.

```
2130   \let\@makefntext\@makefntextORI
2131   \let\@@makefnmark\@@makefnmarkORI
2132 \fi
```

Definitions for the memoir class:

```
2133 \@ifclassloaded{memoir}
```

(see original definition in `memman.pdf`)

```
2134   {\newcommand{\@makefntextFB}[1]{%
2135     \def\footscript##1{##1\dotFFN\kernFFN}%
2136     \setlength{\footmarkwidth}{\FBfnindent}%
2137     \setlength{\footmarksep}{-\footmarkwidth}%
2138     \setlength{\footparindent}{\parindentFFN}%
2139     \makefootmark #1}%
2140   }{}
```

Definitions for the beamer class:

```
2141 \@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.

```
2142   {\def\@makefntextFB#1{%
2143     \def\insertfootnotetext{#1}%
2144     \def\insertfootnotemark{\insertfootnotemarkFB}%
2145     \usebeamertemplate***{footnote}}%
2146   \def\insertfootnotemarkFB{%
2147     \usebeamercolor[fg]{footnote mark}%
2148     \usebeamerfont*{footnote mark}%
2149     \llap{\@thefnmark}\dotFFN\kernFFN}%
2150   }{}
```

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

```

2151 \providecommand*\insertfootnotemarkFB{%
2152   \parindent=\parindentFFN
2153   \rule\z@\footnotesep
2154   \setbox\@tempboxa\hbox{\@thefnmark}%
2155   \ifdim\wd\@tempboxa>\z@
2156     \llap{\@thefnmark}\dotFFN\kernFFN
2157   \fi}
2158 \providecommand\@makefnmarkFB[1]{\insertfootnotemarkFB #1}

```

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

```

2159 \providecommand\localleftbox[1]{}
2160 \AtBeginDocument{%
2161   \@ifpackageloaded{bigfoot}{}%
2162   {\ifdim\parindentFFN<10in
2163     \else
2164       \parindentFFN=\parindent
2165       \ifdim\parindentFFN<1.5em \parindentFFN=1.5em \fi
2166     \fi
2167     \settowidth{\FBfnindent}{\dotFFN\kernFFN}%
2168     \addtolength{\FBfnindent}{\parindentFFN}%
2169     \let\@makefnmarkORI\@makefnmark
2170     \ifFB@koma

```

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

```

2171       \let\@makefnmarkORI\@makefnmark
2172       \long\def\@makefnmark#1{%
2173         \localleftbox{}%
2174         \let\FBeverypar@save\FBeverypar@quote
2175         \let\FBeverypar@quote\relax
2176         \ifFBFrenchFootnotes
2177           \ifx\footnote\thanks
2178             \let\@makefnmark\@makefnmarkTH
2179             \@makefnmarkTH{#1}
2180           \else
2181             \let\@makefnmark\@makefnmarkFB
2182             \@makefnmarkFB{#1}
2183           \fi
2184         \else

```

```

2185         \let\@makefnmark\@makefnmarkORI
2186         \@makefnmarkORI{#1}%
2187     \fi
2188     \let\FBeverypar@quote\FBeverypar@save
2189     \localleftbox{\FBeveryline@quote}}%
2190 \else

```

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

```

2191     \@ifclassloaded{memoir}%
2192     {\ifFBFrenchFootnotes
2193         \setlength{\thanksmarkwidth}{\parindentFFN}%
2194         \setlength{\thanksmarksep}{-\thanksmarkwidth}%
2195     \fi
2196     }{}%

```

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

```

2197     \@ifclassloaded{beamer}%
2198     {\ifFBFrenchFootnotes
2199         \ifdim\parindentFFN=1.5em\else
2200             \FBWarning{%
2201                 \protect\parindentFFN\space is ineffective%
2202                 \MessageBreak within the beamer class.%
2203                 \MessageBreak Reported}%
2204         \fi
2205     \fi
2206     }{}%

```

Definition of \@makefnmark for all other classes:

```

2207     \long\def\@makefnmark#1{%
2208         \localleftbox{}%
2209         \let\FBeverypar@save\FBeverypar@quote
2210         \let\FBeverypar@quote\relax
2211         \ifFBFrenchFootnotes
2212             \@makefnmarkFB{#1}%
2213         \else
2214             \@makefnmarkORI{#1}%
2215         \fi
2216         \let\FBeverypar@quote\FBeverypar@save
2217         \localleftbox{\FBeveryline@quote}}%
2218     \fi
2219 }%
2220 }

```

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 \@makefnmark.

```

2221 \newcommand*{\AddThinSpaceBeforeFootnotes}{\FBAutoSpaceFootnotestru}

```

```

2222 \newcommand*{\FrenchFootnotes}{\FBFrenchFootnotesttrue}
2223 \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.

```

2224 \FBclean@on@exit
2225 \ldf@finish\CurrentOption
2226 \let\loadlocalcfg\FB@llc
2227 </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`.

```

2228 <*acadian>
2229 \PackageInfo{acadian.ldf}%
2230 {'acadian' dialect is currently\MessageBreak
2231  *absolutely identical* to the\MessageBreak
2232  'french' language; reported}
2233 </acadian>
2234 <*canadien>
2235 \PackageWarning{canadien.ldf}%
2236 {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2237  it might be removed sooner or later. Please\MessageBreak
2238  use 'acadian' instead; reported}%
2239 \def\CurrentOption{acadian}

2240 \def\datecanadien{\dateacadian}
2241 \def\captionscanadien{\captionacadian}
2242 \def\extrascanadien{\extrasacadian}
2243 \def\noextrascanadien{\noextrasacadian}
2244 </canadien>
2245 <*francais>
2246 \PackageWarning{francais.ldf}%
2247 {Option 'francais' for Babel is *deprecated*,\MessageBreak
2248  it might be removed sooner or later. Please\MessageBreak
2249  use 'french' instead; reported}%
2250 \chardef\l@francais\l@french
2251 \def\CurrentOption{french}
2252 </francais>

```

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

```
2253 < *frenchb>
2254 \def\bbl@tempa{frenchb}
2255 \ifx\CurrentOption\bbl@tempa
2256   \chardef\l@frenchb\l@french
2257   \def\CurrentOption{french}
2258   \PackageWarning{babel-french}%
2259     {Option 'frenchb' for Babel is *deprecated*,\MessageBreak
2260     it might be removed sooner or later. Please\MessageBreak
2261     use 'french' instead; reported}
2262 \else
2263   \def\bbl@tempa{francais}
2264   \ifx\CurrentOption\bbl@tempa
2265     \chardef\l@francais\l@french
2266     \def\CurrentOption{french}
```

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

```
2267   \ifx\magnification@\undefined
2268     \PackageWarning{babel-french}%
2269       {Option 'francais' for Babel is *deprecated*,\MessageBreak
2270       it might be removed sooner or later. Please\MessageBreak
2271       use 'french' instead; reported}
2272   \fi
2273 \else
2274   \def\bbl@tempa{canadien}
2275   \ifx\CurrentOption\bbl@tempa
2276     \def\CurrentOption{acadian}
2277     \PackageWarning{babel-french}%
2278       {Option 'canadien' for Babel is *deprecated*,\MessageBreak
2279       it might be removed sooner or later. Please\MessageBreak
2280       use 'acadian' instead; reported}
2281   \fi
2282 \fi
2283 \fi
2284 < /frenchb>
2285 <acadian|canadien|frenchb|francais>\input french.ldf\relax
2286 <acadian|canadien>\let\extrasacadian\extrasfrench
2287 <acadian|canadien>\let\noextrasacadian\noextrasfrench
```

3 Change History

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

v3.5f

General: `\l@canadien` was defined too early in file ‘canadien.ldf’:
`\l@acadian` might not be defined. 15
`\selectlanguage{canadien}` allowed again only for backward compatibility (deprecated). 78
`\DecimalMathComma`: Fixed bug with the acadian language. Warning added if used with the `icomma` package. 45

v3.5e

`\frenchsetup`: `StandardLayout` and `GlobalLayoutFrench` options can no longer be toggled when French is not the main language. 56
`\frquote`: Make resettings global on exit. 40
new command `\NoEveryParQuote`. reset `\FB@addGUIlSpace` attribute inside `\localleftbox` (LuaTeX). 39

v3.5d

General: Add `\frquote` to redefinitions for bookmarks. . . . 67
`\frenchsetup`: `ReduceListSpacing` option depreciated: see `StandardListSpacing`. 54

v3.5c

General: Remove grouping inside `\@makefntext`, `\localleftbox` and `\FBeverypar@quote` saved and restored instead. 76
`\frquote`: `\FBeverypar@quote`’s value now properly reset across level changes. 39
`\noextrsfrench`: `\lccode` of quote 0x27 changed from 0x2019 to 0x27 for Unicode engines. 16

v3.5b

General: Reset `\FBeverypar@quote` locally inside `\@makefntext`. Needed by `\frquote`. 76
`\frquote`: New command `\FB@addquote@everypar` to manage `\everypar`: `\frquote` failed when used immediately after a sectionning command. . . 38

v3.5a

General: New optional layout for lists: lists’ items can be typeset as paragraphs with indented labels while the default leaves the labels hanging into the left margin. . . . 69
`\descriptionFB`: `ListItemsAsPar` option taken into account for description lists. 72
`\frenchsetup`: New option `ListItemsAsPar` for displaying lists’ items “as paragraphs”. 54

v3.4d

`\frenchsetup`: New test for deciding about utf8 encoding for keys `og` and `fg` (the former one fails with LaTeX 2018 release). 60

v3.4c

`\ifFBXeTeX`: Reverting to former test, beware of `\XeTeXrevision` left as `\relax` by careless testing. . . . 16

v3.4b

`\datefrench`: Do not redefine `\date` as `\frenchdate` in French. 40

v3.4a

General: `\LdfInit` checks `\FBclean@on@exit` instead of `\captionfrench` (undefined in PLain). Prevents loading `french.ldf` again with acadian option. 14
babel-french now requires eTeX. . . 14
Lua function `token.get_meaning` requires LuaTeX 1.0. 21
New `\FBgspchar` to customise the space character to be used for `\og` and `\fg` with the `UnicodeNoBreakSpaces` option. . 37
New attribute `\FB@dialect` for the French dialect acadian. 20
New command `\FBsetspaces` to fine tune spacing independently in French and in French dialects. . . 18
Shrink/stretch removed in `\FBthousandsep`. 48
Toks `\FBcolonsp`, `\FBthinsp` and `\FBguillsp` removed. 18

frenchb.lua: Global 'FBsp' table added; local function 'get_glue' changed into global 'FBget_glue'.	23		
\datefrench: Specific code for Plain finally removed (babel bug reported).	40		
\extrasfrench: Change \noextras\CurrentOption to \noextrasfrench. \noextrasacadian will be defined as \noextrasfrench in file acadian.ldf.	16		
\frenchsetup: Patch for koma-script classes moved here, after \ifFBPartNameFull is defined, so that it applies to \extrasacadian too; \AtEndOfPackage is too late.	55		
v3.3d			
frenchb.lua: In default mode, for ':' only, check if next node is a glyph or not. If it is, turn the 'auto' flag to false (avoids spurious spaces in URLs, MSDOS paths or 10:35).	26		
v3.3c			
General: LaTeX 2017-04-15 defines TU encoding for Unicode engines, fontspec is no longer required.	68		
New command \FBthousandsep to customise numprint.	48		
New configurable kerns \FBmedkern, and \FBthickkern suitable for HTML translation.	43		
Reorganise warnings when the caption, subcaption or floatrow packages are loaded before babel/french.	51		
Reset \localleftbox locally inside \makefnfntext. Needed by \frquote with LuaTeX.	76		
frenchb.lua: Function 'get_glue' robustified. 'french_punctuation' can insert Unicode characters instead of glues.	22		
\frenchsetup: New option 'UnicodeNoBreakSpaces' for html translators (LuaLaTeX only).	60		
v3.3b			
General: Generate portmanteau files acadian.ldf, canadien.ldf, frenchb.ldf, and francais.ldf			
		and warn about deprecated options.	78
		New 'if' \ifFBfrench to replace \iflanguage test which is based on patterns.	16
	v3.3a		
	General: Compatibility code for pre 2015/10/01 LaTeX release removed, see ltnnews23.tex.		20
	Skip \FBguillskip for LuaTeX replaced by toks \FBguillsp.		18
	\captionsfrench: Commands \frenchpartfirst, \frenchpartsecond and \frenchpartnameord added.		48
	\FBthinspace: Skips \FBcolonskip and \FBthinskip replaced by toks \FBcolonsp and \FBthinsp.		18
	\frenchsetup: \frenchbsetup is now an alias for \frenchsetup.		54
	Options INGuillSpace, ThinColonSpace no longer delayed AtBeginDocument.		54
	\frquote: \FB@quotespace (kern), changed into \FB@guillspace.		39
	v3.2h		
	\@makefnfntextFB: With beamer.cls, add \llap to \@thefnmark for notes numbered over 99.		75
	\bbl@frenchlistlayout: Execute \update@frenchlists only if GlobalLayoutFrench is false. Delete stuff for lists in \noextrasfrench.		73
	\frenchsetup: Option GlobalLayoutFrench skipped when French is not the main language.		56
	v3.2g		
	General: Add \boi to redefinitions for bookmarks.		67
	Changed Unicode definition of \boi.		44
	fontspec defines TU encoding now and no longer loads xunicode.sty. Test changed.		68
	Issue a warning if beamerarticle.sty is loaded after babel.		54
	\frenchsetup: Minimal list customisation when beamerarticle.sty is loaded.		56

Warn when wrong values are provided to options EveryParGuill or EveryLineGuill.	59	v3.2b	<code>\FB@spacing@off</code> and <code>\FB@spacing@on</code>	36
<code>\frquote</code> : Default options of <code>\frquote</code> are no longer engine-dependent.	38		General: Load <code>lualatex.tex</code> for plain LuaTeX to ensure <code>\newattribute</code> is defined.	20
v3.2f			Warning added when the subcaption package is loaded before <code>babel/french</code>	51
<code>\DecimalMathComma</code> : Fixed conflict with the <code>icomma</code> package.	45		<code>frenchb.lua</code> : <code>glue_spec</code> removed; starting with LuaTeX 0.95, glue specifications fit in glue.	24
v3.2e			<code>\ifFB@xetex@punct</code> : New counter <code>\FB@nonchar</code> needed for non characters: it's value will be 4095 for new engines and 255 for older ones.	17
General: Add missing redefinitions for <code>\leftmarginv</code> , <code>\leftmarginvi</code> . Suggested by J.F. Burnol.	69		<code>\NoAutoSpacing</code> : <code>\NoAutoSpacing</code> made robust.	36
<code>\DecimalMathComma</code> : <code>\DecimalMathComma</code> didn't work with LuaTeX. Fixed now.	45		v3.2a	
v3.2d			<code>\@makefnfntextFB</code> : <code>beamer.cls</code> requires a specific definition of <code>\@makefnfntextFB</code> (pointed out by DB). The same is true for <code>memoir</code> and <code>koma-script</code> classes (done). .	74
<code>\descriptionFB</code> : Changed <code>\listindentFB</code> to <code>\descindentFB</code> which defaults to <code>\listindentFB</code> . <code>\leftmargini</code> reduced when <code>\descindentFB</code> is null.	72		<code>\fg</code> : <code>\xspace</code> moved from <code>\FB@fg</code> to <code>\fg</code> : <code>\xspace</code> messes up <code>\frquote</code> , pointed out by Sonia Labetoulle. As a side effect <code>\xspace</code> is now active in <code>\fg</code> in and outside French.	38
v3.2c			v3.1m	
General: New LuaTeX attribute <code>\FB@spacing</code>	20		<code>frenchb.lua</code> : <code>new_glue_scaled</code> returns nil in case of invalid font table (i.e. <code>lcircle1.pfb</code>). In such cases <code>babel-french</code> leaves the node list unchanged.	24
Newif <code>\ifFB@spacing</code> and new commands <code>\FB@spacingon</code> , <code>\FB@spacingoff</code> to control space tuning in French.	20		v3.1l	
Switch <code>\ifFB@spacing</code> added to the four French shorthands.	33		General: Add a variant of <code>\babel@savevariable</code> to save <code>\XeTeXcharclass(es)</code> in a loop. .	31
<code>\FB@xetex@punct@french</code> : Switch <code>\ifFB@spacing</code> added to all <code>\XeTeXinterchartoks</code> commands.	31		<code>frenchb.lua</code> : <code>font.getfont(fid)</code> possibly returns nil even for a positive fid (i.e. <code>AMS lcircle1.pfb</code>). Reported by François Legendre. .	24
<code>\FBthinspace</code> : Change <code>.16667em</code> to <code>.5\fontdimen2\font</code> to get in XeTeX and pdfTeX the same spacing as in LuaTeX.	18		<code>\FB@luatex@punct@french</code> : Use <code>\babel@save</code> to save and restore <code>\shorthandon</code> and <code>\shorthandoff</code>	29
<code>\frenchsetup</code> : Add a warning about options <code>og/fg</code> for old XeTeX or LuaTeX engines requiring active characters.	60		<code>\FB@xetex@punct@french</code> : Save and restore	
<code>\NoAutoSpacing</code> : New definition based on <code>\FB@spacing@off</code> common to all engines.	36			
<code>\ttfamilyFB</code> : New definitions of <code>\ttfamilyFB</code> and <code>co</code> , common to all engines, based on				

<code>\XeTeXinterchartokenstate,</code> <code>\shorthandon, \shorthandoff</code> using <code>\babel@savevariable</code> and <code>\babel@save,</code> <code>\XeTeXcharclass(es)</code> using <code>\FB@savevariable@loop.</code>	31	Use Babel defined loops <code>\bbl@for</code> instead of <code>\@for</code> borrowed from file <code>ltxcntrl.dtx</code> (<code>\@for</code> is undefined in Plain).	30
v3.1k General: (pdfTeX shorthands) test on <code>\lastskip</code> changed from 0pt to 1sp for active punctuation for consistency with XeTeX and LuaTeX.	33	<code>frenchb.lua</code> : Flag <code>addgl</code> set to false for ‘«’ at the end of an <code>\hbox</code> or a paragraph or when followed by a null glue (i.e. springs).	28
<code>\FB@xetex@punct@french</code> : Thin glues (less than 1sp) should not trigger space insertion before high punctuation. Add a check on <code>\lastkip.</code>	31	flag <code>addgl</code> set to false for ‘»’ at the beginning of an <code>\hbox</code> or a paragraph or a tabular ‘l’ and ‘c’ columns.	27
v3.1j General: Loading <code>luatexbase.sty</code> is no longer needed with LaTeX release 2015/10/01 or later.	20	Node <code>HLIST</code> added; node <code>TEMP</code> added for the first node of <code>\hboxes.</code>	23
<code>\frquote</code> : <code>\fr@quote</code> completely rewritten: <code>\leavevmode</code> added and explicitly <code>save/retore</code> <code>\everypar</code> and <code>\localleftbox</code> instead of using a group in order to ensure compatibility with package <code>wrapfig.</code>	39	<code>\captionsfrench</code> : <code>\partname’s</code> definition depends now on flag <code>PartNameFull</code> . No need to redefine it in <code>\frenchbsetup.</code>	48
<code>\PackageWarning</code> is undefined in Plain, use <code>\fb@warning</code> instead.	39	<code>\frenchsetup</code> : Bug fix for koma-scripts classes: a spurious dot was added by the <code>\partformat</code> command.	55
v3.1i General: <code>\nombre</code> command changed when <code>numprint.sty</code> is not loaded: only one warning, no error.	48	<code>PartNameFull</code> now just sets the flag, nothing to add to <code>\captionsfrench</code> when false. ..	54
Remove restriction about loading <code>numprint.sty</code> after <code>babel.</code>	53	v3.1f General: <code>\FBCaption@Separator</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.	51
<code>\frquote</code> : <code>\luatexlocalleftbox</code> changed to <code>\localleftbox</code> by new LaTeX release 2015/10/01. .	39	<code>\FBprocess@options</code> : Bug fix for the beamer class: figure and table captions are now consistent with babel-french’s documentation. Pointed out by Denis Bitouzé. ...	65
v3.1h General: <code>french.cfg</code> from e-french conflicts with babel-french. Do NOT load it (no need for .cfg files with babel-french anyway).	78	Definition of <code>\captionformat</code> and <code>\captiondelim</code> changed when option <code>CustomiseFigTabCaptions</code> is set to false.	65
v3.1g General: Lua function <code>french_punctuation</code> is now inserted at the end of the ‘kerning’ callback (no priority) instead of ‘ <code>hpack_filter</code> ’ and ‘ <code>pre_linebreak_filter</code> ’.	30	<code>\FBthinspace</code> : <code>\FBthinspace</code> is no longer a kern but a skip (babel-french adds a nobreak penalty before it).	18
		v3.1e <code>\frenchsetup</code> : Corrected typo: <code>SmallCapsFigTabCaptions</code> instead of <code>SmallCapsFigTabCaptions</code> . Pointed out by Céline Chevalier. .	54
		v3.1d General: New section: issue warnings	

if packages listings, numprint and natbib are loaded too early or too late vs babel.	53	v3.0c	General: babel-french requires babel-3.9i.	14
v3.1c			Just load luatexbase.sty instead of luaotfload.sty with plain formats.	20
frenchb.lua: Previous bug fix for null glues (v3.0c) did not work properly. Fixed now (I hope!). Pointed out by Jacques André. ...	25		No need to define \l@french as \lang@french, babel.def (3.9j) takes care for this.	15
v3.1b			frenchb.lua: Null glues should not trigger space insertion before high punctuation. Bug pointed out by Benoit Rivet for the 'lstlisting' environment of the listings package.	25
frenchb.lua: Add a check for null fid in french_punctuation (Tikz \nullfont). Bug pointed out by Paul Gaborit.	25		\frenchsetup: New option INGuillSpace.	54
\captionsfrench: Change \scshape to customisable \FBfigtabshape for \figurename and \tablename.	48		No list customisation when beamer class is loaded.	56
\fprimo): Removed \lowercase from definitions of \FrenchEnumerate, ... \No and co: \up already does the conversion.	43	v3.0b	General: frenchb.lua was not found by Lua function dofile (not kpathsea aware). Call function kpse.find_file first, as suggested by Paul Gaborit.	30
\frenchsetup: New option SmallCapsFigTabCaptions.	54		Require luatexbase with LaTeX2e in case fontspec has not been loaded before babel.	20
\ieres: Removed \lowercase from definitions of \ieme and co: \up already does the conversion. ...	43	v3.0a	General:	
v3.1a			\bbl@nonfrenchguillemets deleted, use \babel@save instead.	38
General: fontspec is not required for T1 fonts used with the luainputenc.sty package.	68		\LdfInit checks \captionsfrench instead of \datefrench to avoid a conflict with papertex.cls which loads datetime.sty.	14
Misplaced \fi for plain formats. ...	20		french.cfg will be loaded (if found) instead of frenchb.cfg. NO NEED for .cfg files in French anyway. ...	78
New command \frquote for imbedded or long French quotations.	38		In Plain, provide a substitute for \PackageWarning and \PackageInfo.	14
frenchb.lua: Added flag addgl which must also be true when prev or next is not a char (i.e. \kern0 in «\texttt{a}»).	27		Merging of \captionsfrenchb, \captionsfrançais with \captionsfrench deleted in favor of new babel 3.9 syntax.	49
Codes 0x13 and 0x14 added for French quotes in T1-encoding. ...	22		More informative, less TeXnical warning about \@makecaption. ...	51
Look ahead when next is a kern (i.e. in «\texttt{a}»).	28		New flag \ifFB@luatex@punct for 'high punctuation' management	
\frenchsetup: Codes 0x13 and 0x14 added for French quotes in T1-encoding. Support for older versions of LuaTeX and XeTeX dropped.	60			
New options InnerGuillSingle, EveryParGuill and EveryLineGuill to control \frquote.	54			

with LuaTeX engines.	17	<code>\datefrench</code> : Take advantage of babel's <code>\SetString</code> commands for <code>\datefrench</code> . Doesn't work with Plain (yet?).	40
New handling of 'high punctuation' through callbacks with LuaTeX engines.	20	<code>\descriptionFB</code> : Added <code>\listindentFB</code> to <code>\itemindent</code> . Suggested by Denis Bitouzé. ...	72
No warning about <code>\@makecaption</code> for SMF classes.	51	<code>\extrasfrench</code> : Take advantage of babel's <code>\babel@savevariable</code> to handle apostrophe's <code>\lccode</code> . ..	16
Options processing completely reorganised, now <code>\babel@save</code> and <code>\babel@savevariable</code> are usable for French.	54	<code>\FB@fg</code> : Definitions of <code>\FB@og</code> and <code>\FB@fg</code> now depend on punctuation handling (LuaTeX / XeTeX / active).	37
Support for options <code>frenchb</code> , <code>francais</code> , <code>canadien</code> , <code>acadian</code> changed.	14	<code>\FBprocess@options</code> : With koma-script and memoir class, customise <code>\captionformat</code> and <code>\captiondelim</code>	65
Test <code>\ifXeTeX</code> changed to <code>\ifFBunicode</code> and 'xltextra' changed to 'fontspec'.	68	<code>\frenchsetup</code> : New options <code>OldFigTabCaptions</code> and <code>CustomiseFigTabCaptions</code>	54
<code>\CaptionSeparator</code> : Remove <code>\FBCaption@SeparatorORI</code> , use <code>\babel@save</code> instead.	50		
<code>\captionsfrench</code> : Take advantage of babel's <code>\SetString</code> commands for <code>captionnames</code>	48		