

L'extension

listofitems

v1.65
9 mars 2024

Christian TELLECHEA *

Traducteur : Steven B. SEGLETES †

Cette petite extension pour est destinée à lire une liste d'éléments dont le séparateur peut être choisi par l'utilisateur. Une fois la liste lue, ses éléments sont stockés dans une structure qui se comporte comme un tableau unidimensionnel et ainsi, il devient très facile d'accéder à un élément de la liste par son numéro. Par exemple, si la liste est stockée dans la macro `\foo`, l'élément n° 3 est désigné par `\foo[3]`.

Un élément peut à son tour être une liste disposant d'un autre séparateur que celui de la liste de niveau supérieur, ce qui ouvre la voie à des imbrications et donne une syntaxe rappelant celle des tableaux à plusieurs dimensions du type `\foo[3,2]` pour accéder à l'élément n° 2 de la liste contenue dans l'élément n° 3 de la liste de plus haut niveau.

*. unbonpetit@netc.fr

†. steven.b.segletes.civ@mail.mil

1 Avant-propos

Important : pour toute modification du code source de cette extension (remontée de bug, demande d'ajout ou modification de fonctionnalité), j'invite les utilisateurs de listofitems à envoyer un email à unbonpetit@netc.fr. Il est notamment inutile d'envoyer un email à une autre adresse ou de poster sur un site internet spécialisé.

Cette extension ne requiert aucun package, doit être utilisée avec un moteur ϵ -TeX fournissant la primitive `\expanded`, et doit être appelée sous (pdf)(Xe)(lua)LaTeX par

```
\usepackage{listofitems}
```

et sous (pdf)(Xe)(lua)TeX par

```
\input listofitems.tex
```

Je remercie Steven B. SEGLETES pour la traduction de ce manuel mais aussi pour l'énergie, la motivation et la persévérance qu'il déploie pour apporter de l'aide aux utilisateurs de listofitems en répondant à leurs questions sur un site spécialisé. J'avoue que, vu l'ampleur et la répétitivité de la tâche, je serais lassé depuis belle lurette.

2 Lire une liste simple

Définir le séparateur Le \langle separateur \rangle par défaut est la virgule et si l'on souhaite en changer, il faut, avant de lire une liste d'éléments, le définir avec `\setsepchar{ \langle separateur \rangle }`. Un \langle separateur \rangle est un ensemble de tokens dont les catcodes sont différents de 1 et 2 (les accolades ouvrantes et fermantes), 14 (habituellement %) et 15. Le token de catcode 6 (habituellement #) n'est accepté que s'il est suivi d'un entier auquel cas l'ensemble désigne l'argument d'une macro ; en aucun cas ce token ne doit se trouver seul dans le \langle separateur \rangle . Des commandes peuvent se trouver dans cet ensemble de tokens, y compris la primitive `\par`.

Le \langle separateur \rangle « / » est réservé par défaut pour les listes imbriquées (voir page 3). Il ne faut donc pas écrire « `\setsepchar{/}` » car listofitems comprendrait que l'on souhaite lire une liste imbriquée. Pour définir « / » comme \langle separateur \rangle d'une liste simple, il faut, à l'aide de l'argument optionnel, choisir un autre \langle separateur \rangle de listes imbriquées, par exemple « . » et écrire « `\setsepchar[.]{/}` ».

Il n'est pas possible de choisir « | » comme \langle separateur \rangle car il entrerait en conflit avec l'opérateur logique **OU** noté « || » (voir plus bas). On peut cependant contourner cette limitation, à ses risques et périls, en écrivant « `\setsepchar{|}` ».

Lire la liste Pour lire la liste d'éléments, la commande `\readlist{ \langle macroliste \rangle }{ \langle liste \rangle }` doit être appelée. Ce faisant, la \langle liste \rangle est lue et les éléments sont stockés dans une macro, notée \langle macroliste \rangle qui dès lors, se comporte comme un tableau composé des éléments de la \langle liste \rangle . Un élément est un ensemble de tokens dont les accolades *doivent* être équilibrées. Les tokens de catcode 6 seuls, 14 et 15 ne sont pas autorisés dans les listes. Par exemple, pour définir la \langle macroliste \rangle nommée `\foo`, on peut écrire

```
\setsepchar{,}
\readlist\foo{12,abc,x y ,{\bfseries z},,\TeX,,!}
```

Si la \langle liste \rangle est contenue dans une macro, alors cette macro est développée par listofitems. On peut donc écrire `\readlist{ \langle macroliste \rangle }{ \langle macro \rangle }` ce qui donnerait

```
\setsepchar{,}
\def\liste{12,abc,x y ,{\bfseries z},,\TeX,,!}
\readlist\foo\liste
```

La macro `\greadlist` agit comme `\readlist`, mais effectue des assignations *globales* et donc, la \langle macroliste \rangle est utilisable hors du groupe où a été exécutée `\greadlist`.

Accéder à un élément La macro `\foo` attend un argument numérique *obligatoire* entre crochets, que nous notons i et qui désigne le rang de l'élément auquel on souhaite accéder. Ainsi, `\foo[1]` est ³ « 12 ». De la même façon, `\foo[4]` est « `{\bfseries z}` ».

Le nombre i peut également être négatif auquel cas le comptage se fait à partir de la fin de la liste : -1 représente le dernier rang, -2 l'avant-dernier, etc. Si le nombre d'éléments est n , alors l'argument $-n$ est le premier élément.

D'une façon générale, si une \langle liste \rangle a une longueur n , alors l'index i peut se trouver dans l'intervalle $[1; n]$ ou $[-n; -1]$ et dans le cas contraire, une erreur de compilation survient.

Si l'index est vide, alors `\foo[]` se développe en la \langle liste \rangle entière.

3. Il faut 2 développements à `\foo[i]` pour obtenir l'élément n° i .

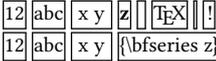
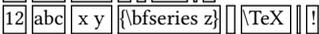
Accéder à un séparateur Lorsque `\readlist\foo{<liste>}` est exécuté, la macro `\foosep` est créée. Elle s'utilise avec la syntaxe `\foosep[<index>]` et permet d'accéder au séparateur qui suit l'élément de rang `<index>`. Le dernier séparateur (celui qui suit le dernier élément) est vide. Si l'`<index>` est vide, `\foosep[]` a un développement vide.

Choisir plusieurs séparateurs possibles Pour spécifier plusieurs séparateurs possibles, il faut utiliser l'opérateur **OU** noté « `||` ». On peut par exemple utiliser cette fonctionnalité pour isoler les termes dans une somme algébrique :

<code>\setsepchar{+ -}</code>	
<code>\readlist\terme{17-8+4-11}</code>	1) 17 (séparateur = -)
1) <code>\terme[1]</code> (séparateur = <code>\termesep[1]</code>)\par	2) 8 (séparateur = +)
2) <code>\terme[2]</code> (séparateur = <code>\termesep[2]</code>)\par	3) 4 (séparateur = -)
3) <code>\terme[3]</code> (séparateur = <code>\termesep[3]</code>)\par	4) 11 (séparateur =)
4) <code>\terme[4]</code> (séparateur = <code>\termesep[4]</code>)	

Nombre d'éléments Si l'on écrit `\readlist<macroliste>{<liste>}` alors la macro `<macroliste>` en contient ⁴ le nombre d'éléments de la `<liste>`. Dans l'exemple avec `\foo`, la macro `\foolen` contient 8.

Afficher tous les éléments À des fins de débogage, la macro `\showitems<macroliste>` compose tous les éléments d'une liste tandis que sa version étoilée affiche ces éléments « détokénisés ⁵ ».

<code>\showitems\foo\par</code>	<code>\showitems*\foo</code>
	

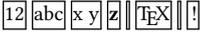
La présentation de chaque élément est confiée à la macro `\showitemsmacro` dont le code est

```
\newcommand\showitemsmacro[1]{%
  \begingroup\fbboxsep=0.25pt \fbboxrule=0.5pt \fbox{\strut#1}\endgroup
  \hskip0.25em\relax}
```

Il est donc possible – et souhaitable – de la redéfinir si l'on cherche un autre effet.

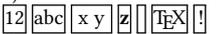
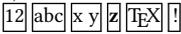
La macro `\fbox` et ses dimensions afférentes `\fbboxsep` et `\fbboxrule` sont définies par `listofitems` lorsqu'on ne compile pas sous \LaTeX de façon à obtenir le même résultat qu'avec \LaTeX .

Suppression des espaces extrêmes Par défaut, `listofitems` lit et prend en compte le (ou les) espaces se trouvant au début et à la fin d'un élément. Pour que ces espaces soient ignorés lors de la lecture de la `<liste>`, il faut exécuter la version étoilée `\readlist*{<macro>}{<liste>}` :

<code>\setsepchar{,}</code>	
<code>\readlist*\foo{12,abc, x y ,{\bfseries z} , ,\TeX,,!}</code>	
<code>\showitems\foo</code>	

Gestion des éléments vides Par défaut, `listofitems` prend en compte les éléments vides. Ainsi, dans l'exemple précédent, le 2-développement de `\foo[7]` est vide. Pour que des éléments vides (ceux délimités par deux séparateurs consécutifs dans la liste) soient ignorés, il faut, avant de lancer la macro `\readlist`, exécuter la macro `\ignoreemptyitems`. La macro `\reademptyitems` revient au comportement par défaut.

Cette option peut être utilisée seule ou combinée avec `\readlist*` auquel cas la suppression des espaces extrêmes intervient *avant* que `listofitems` n'ignore les éléments vides :

<code>\setsepchar{,}</code>	
<code>\ignoreemptyitems</code>	
<code>\readlist\foo{12,abc, x y ,{\bfseries z} , ,\TeX,,!}</code>	
a) nombre d'éléments = <code>\foolen\par</code>	a) nombre d'éléments = 7
<code>\showitems\foo</code>	
<code>\readlist*\foo{12,abc, x y ,{\bfseries z} , ,\TeX,,!}</code>	b) nombre d'éléments = 6
b) nombre d'éléments = <code>\foolen\par</code>	
<code>\showitems\foo</code>	

4. C'est-à-dire qu'elle est purement développable et se développe en un nombre

5. La primitive `\detokenize` qui procède à cette dénaturation insère un espace après chaque séquence de contrôle.

Itérer sur la liste Une fois une liste lue par `\readlist` et stockée dans une $\langle macroliste \rangle$, la commande `\foreachitem` $\langle variable \rangle$ `\in` $\langle macroliste \rangle$ $\{ \langle code \rangle \}$ itère sur la liste : la $\langle variable \rangle$ est une macro choisie par l'utilisateur qui prendra tour à tour la valeur de chaque élément. La macro $\langle variable \rangle cnt$ représente le numéro de l'élément contenu dans $\langle variable \rangle$.

<code>\setsepchar{ }%</code> séparateur = espace	Le mot numéro 1 : Une
<code>\readlist\phrase{Une phrase de test.}</code>	Le mot numéro 2 : phrase
<code>\foreachitem\mot\in\phrase{Le mot numéro \motcnt{} : \mot\par}</code>	Le mot numéro 3 : de
	Le mot numéro 4 : test.

Assigner un élément à une macro La commande `\itemtomacro` $\langle macroliste \rangle$ $[index]$ $\langle macro \rangle$ assigne à la $\langle macro \rangle$ l'élément désigné par $\langle macroliste \rangle$ $[index]$. La $\langle macro \rangle$ ainsi définie est purement développable, sous réserve que l'élément qu'elle contient le soit.

<code>\setsepchar{ }%</code> séparateur = espace	
<code>\readlist\phrase{Une phrase de test.}</code>	
<code>\itemtomacro\phrase[2]\unmot</code>	macro:->phrase
<code>\meaning\unmot\par</code>	macro:->test.
<code>\itemtomacro\phrase[-1]\motdelafin</code>	
<code>\meaning\motdelafin</code>	

3 Listes imbriquées

On parle de liste « imbriquée » lorsque l'on demande à `listofitems` de lire une liste où les éléments sont à leur tour compris comme une liste (dont le séparateur est différent de la liste de niveau supérieur). Le nombre d'imbrication n'est pas limité, mais dans la pratique, un niveau d'imbrication de 2, voire 3, semble un maximum.

Définir les séparateurs Pour indiquer que les éléments de la liste doivent eux-mêmes être compris comme des listes et que la recherche des éléments sera récursive, il faut spécifier plusieurs $\langle séparateurs \rangle$, chacun correspondant à un niveau d'imbrication.

Pour déclarer une $\langle liste de séparateurs \rangle$ il faut définir le $\langle séparateur \rangle$ de cette $\langle liste de séparateurs \rangle$ à l'aide de l'argument optionnel de la macro `\setsepchar` et écrire

```
\setsepchar[ $\langle séparateur \rangle$ ]{ $\langle liste des séparateurs \rangle$ }
```

Par défaut, le $\langle séparateur \rangle$ est « / ». Ainsi, si l'on donne l'ordre

```
\setsepchar{\\,/ }
```

on indique une profondeur récursive de 3 et on choisit comme séparateur de la $\langle liste des séparateurs \rangle$ le caractère par défaut « / » :

- les éléments de niveau 1 sont trouvés entre les séparateurs « \\ » ;
- les éléments de niveau 2 sont trouvés dans les éléments de niveau 1 entre les séparateurs « , » ;
- enfin, les éléments de niveau 3 sont trouvés dans ceux de niveau 2 entre les séparateurs « □ ».

La $\langle profondeur \rangle$ de recherche est contenue dans la macro purement développable `\nestdepth`.

Lire et accéder aux éléments Pour les listes imbriquées, les index obéissent à la règle suivante :

- $[]$ désigne la liste principale, c'est-à-dire l'argument de `\readlist` ;
- $[\langle i \rangle]$ désigne l'élément n° $\langle i \rangle$ de la liste principale ;
- $[\langle i \rangle , \langle j \rangle]$ désigne l'élément n° $\langle j \rangle$ de la liste constituée par l'élément évoqué au point précédent ;
- $[\langle i \rangle , \langle j \rangle , \langle k \rangle]$ désigne l'élément n° $\langle k \rangle$ de la liste constituée par l'élément évoqué au point précédent ;
- etc.

Comme pour les liste non imbriquées, les index peuvent être négatifs.

Pour lire les éléments, la syntaxe de `\readlist` est exactement la même qu'avec les listes simples :

<code>\setsepchar{\\,/ }</code>	
<code>\readlist\baz{1,2 a b,3 c\\4 d e f,5,6\\7,,8, ,9 xy z}</code>	a) <code>\baz[1]</code> est 1,2 a b,3 c
a) <code>\string\baz[1]</code> est <code>\baz[1]\par</code>	b) <code>\baz[1,1]</code> est 1
b) <code>\string\baz[1,1]</code> est <code>\baz[1,1]\par</code>	c) <code>\baz[1,1]</code> est 1
c) <code>\string\baz[1,1,1]</code> est <code>\baz[1,1,1]\par</code>	b) <code>\bar[1,2]</code> est 2 a b
b) <code>\string\bar[1,2]</code> est <code>\baz[1,2]\par</code>	e) <code>\baz[1,2,3]</code> est b
e) <code>\string\baz[1,2,3]</code> est <code>\baz[1,2,3]\par</code>	f) <code>\baz[-2,1,-1]</code> est f
f) <code>\string\baz[-2,1,-1]</code> est <code>\baz[-2,1,-1]</code>	

L'opérateur « || » Cet opérateur peut se trouver dans n'importe quel niveau d'imbrication.

<code>\setsepchar[,]{+ -,* /}</code>	
<code>\readlist\nombres{1+2*3-4/5*6}</code>	
Terme 1 : \nombres[1]\par	Terme 1 : 1
Terme 2 : \nombres[2] (facteurs : \nombres[2,1] et \nombres[2,2])\par	Terme 2 : 2*3 (facteurs : 2 et 3)
Terme 3 : \nombres[3] (facteurs : \nombres[3,1], \nombres[3,2] et \nombres[3,3])	Terme 3 : 4/5*6 (facteurs : 4, 5 et 6)

Nombre d'éléments La macro `\listlen<macroList>[<index>]` nécessite 2 développements pour donner le nombre d'éléments de la liste spécifiée par l'<index>.

La <profondeur> de l'<index> doit être strictement inférieure à celle de la <liste>.

Dans le cas d'un <index> vide, `\listlen<macroList>[]` donne en deux développements le même résultat que `<macroList>\len` qui le donne en un seul.

<code>\setsepchar{\\,/ }</code>	
<code>\readlist\baz{1,2 a b,3 c\\4 d e f,5,6\\7,,8, ,9 xy z}</code>	a) 3 ou 3
a) \bazlen\ ou \listlen\baz[]\par	b) 3
b) \listlen\baz[1]\par	c) 3
c) \listlen\baz[2]\par	d) 5
d) \listlen\baz[3]\par	e) 1
e) \listlen\baz[3,1]\par	f) 2
f) \listlen\baz[3,4]\par% 2 éléments vides	g) 3
g) \listlen\baz[3,5]	

Afficher les éléments La macro `\showitems<macroList>[<index>]` affiche les éléments de la liste spécifiée par <index>, selon le même principe que `\listlen`.

La <profondeur> de l'<index> doit être strictement inférieure à celle de la <liste>.

<code>\setsepchar{\\,/ }</code>	a) 1,2 a b,3 c 4 d e f,5,6 7,,8, ,9 xy z
<code>\readlist\baz{1,2 a b,3 c\\4 d e f,5,6\\7,,8, ,9 xy z}</code>	b) 2 a b 3 c
a) \showitems\baz[]\par	c) 4 d e f 5 6
b) \showitems\baz[1]\par	d) 7 8 9 xy z
c) \showitems\baz[2]\par	e) 7
d) \showitems\baz[3]\par	f)
e) \showitems\baz[3,1]\par	g) 9 xy z
f) \showitems\baz[3,4]\par% 2 éléments vides	
g) \showitems\baz[3,5]	

Éléments vides et espaces extrêmes La suppression des éléments vides et/ou des espaces extrêmes intervient dans tous les éléments, quel que soit le degré d'imbrication. Il est clair que choisir un espace comme séparateur est inutile si l'on veut utiliser `\readlist*`. C'est pourquoi dans cet exemple, « * » est choisi comme séparateur. Dans cet exemple, on ne supprime que les espaces extrêmes en gardant les éléments vides.

<code>\setsepchar{\\,/ *}</code>	a) 1, 2*a*b ,3*c 4*d*e*f,5,6 7,,8, ,9* xy *z
<code>\readlist*\baz{1, 2*a*b ,3*c\\4*d*e*f,5,6\\7,,8, ,9* xy *z}</code>	b) 2*a*b 3*c
a) \showitems\baz[]\par	c) 4*d*e*f 5 6
b) \showitems\baz[1]\par	d) 7 8 9* xy *z
c) \showitems\baz[2]\par	e) 7
d) \showitems\baz[3]\par	f)
e) \showitems\baz[3,1]\par	g) 9 xy z
f) \showitems\baz[3,4]\par	
g) \showitems\baz[3,5] % "xy" sans espaces extrêmes	

Itérer sur une liste La syntaxe `\foreachitem <variable> \in <macro>[<index>]{<code>}` reste valable où désormais, l'<index> spécifie sur quel élément (compris comme une liste) on veut itérer.

La <profondeur> de l'<index> doit être strictement inférieure à celle de la <liste>.

Assigner un élément à une macro La syntaxe `\itemtomacro<macroliste>[<index>]<macro>` reste valable pour assigner à `<macro>` l'élément spécifié par `<macroliste>[<index>]`.

```
\setsepchar[,]{\\, }
\readlist\poeme{J'arrive tout couvert encore de rosée\\%
Que le vent du matin vient glacer à mon front.\\%
Souffrez que ma fatigue à vos pieds reposée\\%
Rêve des chers instants qui la délasseront.}% 2e strophe de «
Green », Paul Verlaine
\itemtomacro\poeme[2]\vers
2e vers = \vers
Un mot = glacer

\itemtomacro\poeme[2,-4]\mot
Un mot = \mot
```

La macro `\itemtomacro` fait une assignation globale.

4 Tokens appariés

Pour le découpage des items, il est possible à partir de la version 1.6, de tenir compte de la présence de caractères *appariés*. Ainsi, si une liste de caractères appariés est définie, chaque item s'étend jusqu'au prochain (*séparateur*) qui équilibre les tokens appariés.

Pour définir une liste de tokens appariés, on utilise

```
\defpair{<tok1><tok2><tok3><tok4>...}
```

où les tokens sont lus deux par deux pour former les paires d'appariement. Un `<token>` d'appariement doit être constitué d'un seul caractère ; les macros, primitives, espaces, accolades, token `«#»`, ensembles de plusieurs tokens entre accolades sont interdits. Deux tokens formant une paire *doivent* être différents.

```
\setsepchar{+|-}
\defpair{()[]}
\readlist\termes{1+2*[3+4*(5+6-7)+8]-9+10}
\showitems\termes
```

1 2*[3+4*(5+6-7)+8] 9 10

Pour revenir au comportement par défaut, c'est-à-dire sans tokens appariés, il faut exécuter

```
\defpair{}
```

Dans une expression, pour stocker dans une macro ce qui se trouve entre deux tokens appariés, on peut faire appel à

```
\insidepair<tok1><tok2>{<expression>}\macro
```

qui mettra dans la `\macro` ce qui se trouve entre la paire `<tok1><tok2>` dans l'`<expression>`.

```
\setsepchar{+|-}
\defpair{()}
\readlist\termes{1+2*(3+4*(5+6-7)+8)-9+10}
\showitems\termes

\itemtomacro\termes[2]\parenterm
Dans la parenthèse extérieure :
\insidepair()\parenterm\inbigparen
"\inbigparen"

Dans la parenthèse intérieure :
\insidepair()\inbigparen\insmallparen
"\insmallparen"
```

1 2*(3+4*(5+6-7)+8) 9 10
 Dans la parenthèse extérieure : "3+4*(5+6-7)+8"
 Dans la parenthèse intérieure : "5+6-7"

5 Le code

Le code ci-dessous est l'exact verbatim du fichier `listofitems.tex`. J'espère que les quelques commentaires qui le parsèment de-ci de-là seront suffisants pour que l'utilisateur ou le curieux comprenne la machinerie interne de ce package :

```

1 % Ce fichier contient le code de l'extension "listofitems"
2 %
3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4 %
5 \def\loiname      {listofitems}
6 \def\loiver       {1.65}
7 %
8 \def\loidate      {2024/03/09}
9 %
10 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
11 %
12 % Author      : Christian Tellechea
13 % Status      : Maintained
14 % Maintainer  : Christian Tellechea
15 % Email       : unbonpetit@netc.fr
16 % Package URL: https://www.ctan.org/pkg/listofitems
17 % Copyright   : Christian Tellechea 2016-2024
18 % Licence     : Released under the LaTeX Project Public License v1.3c
19 %              or later, see http://www.latex-project.org/lppl.txt
20 % Files       : 1) listofitems.tex
21 %              2) listofitems.sty
22 %              3) listofitems-fr.tex
23 %              4) listofitems-fr.pdf
24 %              5) listofitems-en.tex
25 %              6) listofitems-en.pdf
26 %              7) README
27 %              8) listofitems-test-tex.tex
28 %              9) listofitems-test-tex.pdf
29 %             10) listofitems-test-latex.tex
30 %             11) listofitems-test-latex.pdf
31 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
32 \csname loiloadonce\endcsname
33 \let\loiloadonce\endinput
34 \expandafter\edef\csname loi_restorecatcode\endcsname{%
35   \catcode\number'\_=\number\catcode'\_ \relax
36 }
37 \catcode'\_11
38
39 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
40 %%% gestion des erreurs + annonce package %%%
41 %%%
42 \ifdefined\loi_fromsty
43   \def\loi_error#1{\PackageError\loiname{#1}{Read the \loiname\space manual}}% pour LaTeX
44 \else
45   \def\loi_error#1{\errmessage{Package \loiname\space Error: #1^^J}}% pour TeX
46   \immediate\write -1 {Package: \loidate\space v\loiver\space Grab items in lists using user-specified sep char (CT)}%
47 \fi
48
49 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
50 %%% vérification des prérequis %%%
51 %%%
52 \def\loi_checkprimitive#1#2{% Vérifie que #1 est une primitive et sinon, émet le message #2 et exécute \endinput
53   \begingroup
54     \edef\__tempa{\meaning#1}%
55     \edef\__tempb{\string #1}%
56     \expandafter
57   \endgroup
58   \ifx\__tempa\__tempb\else
59     \loi_error{#2}%
60     \loi_restorecatcode
61     \expandafter\endinput
62   \fi
63 }
64 \loi_checkprimitive\TeXversion
65 {%
66   You are not using an eTeX engine, listofitems cannot work.%
67 }%
68 \loi_checkprimitive\expanded
69 {%
70   The \string\expanded\space primitive is not provided by your TeX engine, , listofitems cannot work.%

```

```

71 }%
72
73 %%%
74 %%% macros auxiliaires %%%
75 %%%
76 \def\loi_quark{\loi_quark}
77 \long\def\loi_identity#1{#1}
78 \long\def\loi_gobarg#1{}
79 \long\def\loi_first#1#2{#1}
80 \long\def\loi_second#1#2{#2}
81 \long\def\loi_firsttonil#1#2\_nil{#1}
82 \long\def\loi_antefi#1#2\fi{#2\fi#1}
83 \long\def\loi_exparg#1#2{%
84   \expandafter\loi_exparg_a\expandafter{#2}{#1}% \loi_exparg{<a>}{<b>} devient <a>{<*b>}
85 }
86 \long\def\loi_exparg_a#1#2{#2{#1}}
87 \long\def\loi_expafter#1#2{%
88   \expandafter\loi_expafter_a\expandafter{#2}{#1}% \loi_expafter{<a>}{<b>} devient <a><*b>
89 }
90 \long\def\loi_expafter_a#1#2{#2#1}
91 \def\loi_macroname{%
92   \loi_ifinrange\escapechar[[0:255]]%
93   {%
94     \expandafter\loi_gobarg
95   }
96   {%
97   }%
98   \string
99 }
100 \def\loi_argcsname#1#\loi_argcsname_a{#1}
101 \def\loi_argcsname_a#1#2{\loi_expafter{#1}{\csname#2\endcsname}}
102 \long\def\loi_addtomacro#1#2{\loi_exparg{\def#1}{#1#2}}
103
104 %%%
105 %%% macros de test %%%
106 %%%
107 \long\def\loi_ifnum#1{%
108   \ifnum#1%
109     \expandafter\loi_first
110   \else
111     \expandafter\loi_second
112   \fi
113 }
114 \long\def\loi_ifx#1{%
115   \ifx#1%
116     \expandafter\loi_first
117   \else
118     \expandafter\loi_second
119   \fi
120 }
121 \long\def\loi_ifempty#1{%
122   \loi_exparg\loi_ifx{\expandafter\relax\detokenize{#1}\relax}%
123 }
124 \def\loi_ifstar#1#2{%
125   \def\loi_ifstar_a{\loi_ifx{*}\loi_nxttok}{\loi_first{#1}}{#2}}%
126   \futurelet\loi_nxttok\loi_ifstar_a
127 }
128 \edef\loi_escapechar{\expandafter\loi_gobarg\string\}
129 \long\def\loi_ifcsexpandable#1{% #1 est-il constitué d'une seule sc _développable_ ?
130   \loi_ifempty{#1}
131   {%
132     \loi_second
133   }
134   {\loi_ifspacefirst{#1}
135     {%
136       \loi_second% si espace en 1er, faux
137     }
138     {%
139       \csname loi_\if\loi_escapechar\expandafter\loi_firsttonil\detokenize{#1}\_nil first\else second\fi\endcsname
140       {%

```

```

141 \loi_exparg\loi_ifempty{\loi_gobarg#1}% 1 seul arg commençant par "\ " ?
142 {%
143 \def\loi_tempa{#1}\loi_exparg{\def\loi_tempb}{#1}% est-il développable ?
144 \expandafter\unless\loi_ifx{\loi_tempa\loi_tempb}%
145 }
146 {%
147 \loi_second
148 }%
149 }
150 {%
151 \loi_second
152 }%
153 }%
154 }%
155 }
156 \def\loi_ifinrange#1[[#2:#3]]{%
157 \expandafter\unless\loi_ifnum{\numexpr(#1-#2)*(#1-#3)>0 }%
158 }
159 \def\loi_ifstring#1#2{% si la chaine #1 est contenue dans #2
160 \def\loi_ifstring_a#1#1##2\_nil{%
161 \loi_ifempty{##2}
162 \loi_second
163 \loi_first
164 }%
165 \loi_ifstring_a#2#1\_nil% appel de la macro auxiliaire
166 }
167
168 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
169 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro \loi_foreach %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
170 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
171 \newcount\loi_cnt_foreach_nest
172 \loi_cnt_foreach_nest=0
173 \def\end_foreach{\end_foreach}% quark
174 \def\loi_def_foreachsep#1{%
175 \long\def\loi_foreach##1\in##2##3{%
176 \global\advance\loi_cnt_foreach_nest1
177 \loi_argcsname\def{loop_code\_number\loi_cnt_foreach_nest}{##3}%
178 \loi_foreach_a##1##2#1\end_foreach#1%
179 \loi_argcsname\let{loop_code\_number\loi_cnt_foreach_nest}\empty
180 \global\advance\loi_cnt_foreach_nest-1
181 }%
182 \long\def\loi_foreach_a##1##2#1{%
183 \def##1{##2}%
184 \loi_ifx{\end_foreach##1}
185 {}
186 {%
187 \csname loop_code\_number\loi_cnt_foreach_nest\endcsname% exécute le code
188 \loi_foreach_a##1%
189 }%
190 }%
191 }
192
193 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
194 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macros gérant l'appariement %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
195 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
196 \long\def\defpair#1{%
197 \let\loi_listofpair\empty
198 \loi_ifempty{#1}
199 {%
200 }
201 {%
202 \defpair_a{#1}\loi_quark\loi_quark
203 }%
204 }
205 \long\def\defpair_a#1#2#3{%
206 \loi_ifx{\loi_quark#2}
207 {%
208 \def\loi_sanitizelist##1,\_nil{\def\loi_listofpair{##1}}%
209 \loi_sanitizelist#1\_nil
210 }

```

```

211 {%
212 \loi_if_validpair#2#3%
213 {%
214 \long\def\loi_paired_a{#2}\long\def\loi_paired_b{#3}%
215 \loi_ifx{\loi_paired_a\loi_paired_b}
216 {%
217 \loi_error{Paired tokens must not be equal, the pair \detokenize{#2#3} is ignored}%
218 \defpair_a{#1}%
219 }
220 {%
221 \defpair_a{#1#2#3,}%
222 }%
223 }
224 {%
225 \loi_error{Invalid paired tokens, the pair "\detokenize{#2}" and "\detokenize{#3}" is ignored}%
226 \defpair_a{#1}%
227 }%
228 }%
229 }
230 \long\def\loi_if_validpair#1#2{%
231 \def\loi_validpair{1}%
232 \loi_if_invalid_pairtoken{#1}
233 {%
234 \def\loi_validpair{0}%
235 }%
236 \loi_if_invalid_pairtoken{#2}
237 {%
238 \def\loi_validpair{0}%
239 }%
240 \loi_ifnum{\loi_validpair=1 }
241 }
242 \long\def\loi_if_invalid_pairtoken#1{%
243 \loi_ifempty{#1}
244 {%
245 \loi_identity
246 }
247 {%
248 \loi_ifspacefirst{#1}
249 {%
250 \loi_identity
251 }
252 {%
253 \loi_exparg\loi_ifempty{\loi_gobarg#1}% 1 seul token ?
254 {%
255 \ifcat\relax\noexpand#1%
256 \expandafter\loi_identity
257 \else
258 \expandafter\loi_gobarg
259 \fi
260 }
261 {%
262 \loi_identity% si plusieurs tokens, faux
263 }%
264 }%
265 }%
266 }
267 \long\def\loi_count_occure#1\in#2\_nil#3{% compte le nombre d'occurrences de #1 dans #2 et met le résultat dans la ✓
macro #3 ##BUGFIX v1.65
268 \long\def\loi_count_occure_a##1##2##1##3\_nil{%
269 \loi_ifempty{##3}
270 {%
271 \def#3{##1}%
272 }
273 {%
274 \expandafter\loi_count_occure_a\number\numexpr##1+1\relax##3\_nil
275 }%
276 }%
277 \loi_count_occure_a0#2#1\_nil
278 }
279 \long\def\loi_check_pair#1#2\in#3{% teste l'appariement de #1 et #2 dans #3

```

```

280 \loi_ifempty{#3}
281   {%
282   \loi_second
283   }
284   {%
285   \loi_count_occur#1\in#3\_nil\loi_tempa
286   \loi_count_occur#2\in#3\_nil\loi_tempb
287   \loi_ifnum{\loi_tempa=\loi_tempb\relax}%
288   }%
289 }
290 \long\def\loi_grabpaired_expr#1#2#3#4#5{% #1=liste de paires #2=expression #3=séparateur #4=résultat #5=reste
291   \let#4\empty
292   \def\loi_remain{#2#3}%
293   \loi_foreach\loi_pair\in{#1}{%
294     \expandafter\loi_grabpaired_expr_a\loi_pair{#3}#4%
295   }%
296   \def\loi_remove_lastsep##1#3\_nil{\def#4{##1}}%
297   \expandafter\loi_remove_lastsep#4\_nil
298   \loi_expafter{\long\def\loi_grab_remain}#4##1\_nil{%
299     \loi_ifempty{##1}
300     {%
301       \let#5\empty
302     }
303     {%
304       \loi_exparg{\def#5}{\loi_gobarg##1}%
305     }%
306   }%
307   \loi_grab_remain#2\_nil
308 }
309 \long\def\loi_grabpaired_expr_a#1#2#3#4{% #1#2=paire en cours #3=séparateur #4=résultat
310   \loi_exparg{\loi_check_pair#1#2\in}#4% si les paires sont appariées dans le résultat
311   {%
312   }% passer à la paire suivante
313   {%
314   \long\def\loi_grabpaired_expr_b##1#3##2\_nil{%
315     \loi_addtomacro#4{##1#3}% ajouter au résultat ce qui est jusqu'au prochain séparateur
316     \def\loi_remain{##2}%
317     \loi_exparg{\loi_check_pair#1#2\in}{#4}
318     {%
319     }
320     {%
321     \loi_ifempty{##2}
322     {%
323       \loi_error{"\detokenize{#1}" and "\detokenize{#2}" are not paired}%
324     }
325     {%
326     \loi_grabpaired_expr_b##2\_nil
327     }%
328   }%
329   }%
330   \expandafter\loi_grabpaired_expr_b\loi_remain\_nil
331   }%
332 }
333 \def\insidepair#1#2#3#4{% #1#2=paire #3=expr #4=macro recevant le resultat
334   \loi_if_validpair#1#2%
335   {%
336   \loi_ifcsexpandable{#3}
337   {%
338   \loi_exparg{\insidepair#1#2}{#3}#4%
339   }
340   {%
341   \loi_check_pair#1#2\in{#3}% si les paires sont appariées dans le résultat
342   {%
343   \def\insidepair_a##1##2\_nil{\insidepair_b##2\_nil{#1}}%
344   \def\insidepair_b##1##2##2\_nil##3{%
345     \loi_check_pair#1#2\in{##3##1#2}
346     {%
347     \loi_exparg{\def#4}{\loi_gobarg##3##1}%
348     \def\loi_remainaftertparen{##2}%
349     }%

```

```

350     {%
351     \insidepair_b##2\_nil{##3##1#2}%
352     }%
353     }%
354     \insidepair_a#3\_nil
355     }
356     {%
357     \loi_error{"\detokenize{#1}" and "\detokenize{#2}" are not paired in "#3"}%
358     }%
359     }%
360     }
361     {%
362     \loi_error{Invalid paired tokens "\detokenize{#1}" and "\detokenize{#2}", empty \string#4 returned}% et bim
363     \let#4 empty% voilà, bien fait pour vos gueules
364     }%
365     }
366
367     %%%%%%%%%%%
368     %%%%%%%%%%% macro \loi_fornum %%%%%%%%%%%
369     %%%%%%%%%%%
370     \def\loi_fornum#1=#2to#3\do{%
371     \edef#1{\number\numexpr#2}%
372     \expandafter\loi_fornum_a
373     \csname loi_fornum_\string#1\expandafter\endcsname\expandafter
374     {\number\numexpr#3\expandafter}%
375     \expanded{\ifnum#1<\numexpr#3\relax>+\else<-\fi}%
376     #1%
377     }
378     \long\def\loi_fornum_a#1#2#3#4#5#6{%
379     \def#1{%
380     \unless\ifnum#5#3#2\relax
381     \loi_antefi{#6\edef#5{\number\numexpr#5#41\relax}#1}%
382     \fi%
383     }%
384     #1%
385     }
386
387     %%%%%%%%%%%
388     %%%%%%%%%%% macro retirant les espaces extrêmes %%%%%%%%%%%
389     %%%%%%%%%%%
390     \long\def\loi_ifspacefirst#1{%
391     \expandafter\loi_ifspacefirst_a\detokenize{#10} \_nil
392     }
393     \long\def\loi_ifspacefirst_a#1 #2\_nil{%
394     \loi_ifempty{#1}%
395     }
396     \loi_expafter{\def\loi_gobspace}\space{}
397     \long\def\loi_removefirstspaces#1{##BUGFIX v1.63
398     \loi_ifspacefirst{#1}
399     {\loi_exparg\loi_removefirstspaces{\loi_gobspace#1}}
400     {\unexpanded{#1}}%
401     }%
402     \begingroup
403     \catcode0 12
404     \long\gdef\loi_removelastspaces#1{\loi_removelastspaces_a#1^^00 ^^00\_nil}
405     \long\gdef\loi_removelastspaces_a#1 ^^00{\loi_removelastspaces_b#1^^00}
406     \long\gdef\loi_removelastspaces_b#1^^00#2\_nil{\loi_ifspacefirst{#2}{\loi_removelastspaces_a#1^^00 ^^00\_nil}{\
unexpanded{#1}}}
407     \endgroup
408     \long\def\loi_removeextremespaces#1{\expanded{\loi_exparg\loi_removelastspaces{\expanded{\loi_removefirstspaces\
{#1}}}}}
409
410     %%%%%%%%%%%
411     %%%%%%%%%%% macro publique \setsepchar %%%%%%%%%%%
412     %%%%%%%%%%%
413     \def\setsepchar{\futurelet\loi_nxttok\setsepchar_a}
414     \def\setsepchar_a{%
415     \loi_ifx{[\loi_nxttok}
416     {%
417     \setsepchar_b

```

```

418 }
419 {%
420 \setsepchar_b[/]%
421 }%
422 }
423 \long\def\setsepchar_b[#1]#2{% #1=sepcar de <liste des sepcar> #2=<liste des sepcar>
424 \loi_ifempty{#1}
425 {%
426 \loi_error{Empty separator not allowed, separator "/" used}%
427 \setsepchar_b[/]{#2}%
428 }
429 {%
430 \def\loi_currentsep{#1}%
431 \_loi_removeextremespacesfalse
432 \loi_nestcnt1 % réinitialiser niveau initial à 1
433 \def\nestdepth{1}%
434 \loi_argcsname\let\loi_previndex[\number\loi_nestcnt]\empty
435 \def\loi_listname{\loi_listofsep}%
436 \let\loi_def\def
437 \let\loi_edef\edef
438 \let\loi_let\let
439 \let\loi_listofpair_saved\loi_list_ofpair
440 \let\loi_list_ofpair\empty
441 \loi_ifempty{#2}
442 {%
443 \loi_error{Empty list of separators not allowed, "," used}%
444 \readlist_g1{,}%
445 }
446 {%
447 \readlist_g1{#2}%
448 }%
449 \loi_argcsname\let\nestdepth{\loi_listofseplen[0]}%
450 \loi_argcsname\let\loi_currentsep{\loi_listofsep[1]}% 1er car de séparation
451 \let\loi_listofpair\loi_listofpair_saved
452 }%
453 }
454
455 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
456 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro normalisant l'index %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
457 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
458 \def\loi_normalizeindex#1#2#3{% #1=correction de profondeur #2=macroname #3=liste d'index --> renvoie {err}{indx /
norm}
459 \loi_ifempty{#3}
460 {%
461 {}{}%
462 }
463 {%
464 \loi_exparg{\loi_normalizeindex_a1{}}{\number\numexpr\csname#2nest\endcsname-#1\relax}{#2}#3,\loi_quark,%
465 }%
466 }%
467 \def\loi_normalizeindex_a#1#2#3#4#5,% #1=compteur de profondeur #2=index précédents #3=profondeur max #4=macroname /
#5=index courant
468 \loi_ifx{\loi_quark#5}
469 {%
470 \loi_normalizeindex_c#2\loi_quark% supprimer la dernière virgule
471 }
472 {%
473 \loi_ifnum{#1>#3 }
474 {%
475 \loi_invalidindex{Too deeply nested index, index [.] retained}{#2}% si profondeur trop grande
476 }
477 {%
478 \loi_ifinrange\ifnum\numexpr#5<0 -1*\fi(#5)[[1:\csname #4len[#20]\endcsname]]% si abs(#5) hors de [1,len]
479 {%
480 \loi_exparg\loi_normalizeindex_b
481 {\number\numexpr#5\ifnum\numexpr#5<0 +\csname #4len[#20]\endcsname+1\fi}%
482 {#1}%
483 {#2}%
484 {#3}%
485 {#4}%

```

```

486     }
487     {%
488     \loi_invalidindex{#5 is an invalid index, index [.] retained}{#2}%
489     }%
490 }%
491 }%
492 }
493 \def\loi_normalizeindex_b#1#2#3{%
494 \loi_exparg\loi_normalizeindex_a{\number\numexpr#2+1}{#3#1,}% #1=index à rajouter #2=compteur de profondeur #3=✓
    index précédents
495 }
496 \def\loi_normalizeindex_c#1,\loi_quark{{}{#1}}
497 \def\loi_invalidindex#1#2{%
498 \loi_ifempty{#2}
499     {%
500     \loi_invalidindex_a{#1},%
501     }
502     {%
503     \loi_invalidindex_a{#1}{#2}% BUGFIX 1.64
504     }%
505 }
506 \def\loi_invalidindex_a#1#2{%
507 \loi_invalidindex_b#1\loi_quark#2\loi_quark
508 }
509 \def\loi_invalidindex_b#1[.]#2\loi_quark#3,\loi_quark#4\loi_quark,{% #4= index ignorés
510 {#1[#3]#2}{#3}%
511 }
512
513 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
514 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro publique \readlist %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
515 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
516 \newcount\loi_nestcnt
517 \def\greadlist{%
518 \let\loi_def\gdef
519 \let\loi_edef\xdef
520 \def\loi_let{\global\let}%
521 \readlist_a
522 }%
523 \def\readlist{%
524 \let\loi_def\def
525 \let\loi_edef\edef
526 \let\loi_let\let
527 \readlist_a%
528 }
529 \def\readlist_a{%
530 \loi_nestcnt1 % niveau initial = 1
531 \loi_argcsname\let\loi_previndex[\number\loi_nestcnt]}empty
532 \loi_ifstar
533     {%
534     \_loi_removeextremespacestrue
535     \readlist_b
536     }
537     {%
538     \_loi_removeextremespacesfalse
539     \readlist_b
540     }%
541 }
542 \long\def\readlist_b#1#2{% #1=macro stockant les éléments #2=liste des éléments
543 \loi_ifcsexpandable{#2}
544     {%
545     \loi_exparg{\readlist_b#1}{#2}%
546     }
547     {%
548     \loi_edef\loi_listname{\loi_macroname#1}%
549     \loi_exparg{\readlist_c#1{#2}}{\loi_listname}%%
550     }%
551 }
552 \long\def\readlist_c#1#2#3{% #1=macro stockant les éléments #2=liste des éléments #3=macroname
553 \loi_argcsname\loi_let{#3nest}\nestdepth
554 \loi_argcsname\loi_def{#3[]}{#2}% la liste entière

```

```

555 \loi_argcsname\loi_def{#3sep[]}{% séparateur vide
556 \loi_ifempty{#2}
557 {%
558 \loi_def#1[##1]{}%
559 \loi_argcsname\loi_def{#3len}{0}\loi_argcsname\loi_def{#3len[0]}{0}%
560 \loi_error{Empty list ignored, nothing to do}%
561 }
562 {%
563 \loi_def#1[##1]{\expanded{\expandafter\readlist_d\expanded{\loi_normalizeindex0{#3}{##1}}{#3}}}%
564 \loi_argcsname\loi_def{#3sep}[##1]{\expanded{\expandafter\readlist_d\expanded{\loi_normalizeindex0{#3}{##1}}{#3sep}/
}}}%
565 \readlist_e{#2}%
566 \loi_argcsname\loi_argcsname\loi_let{#3len}{#3len[0]}% longueur du niveau 0
567 }%
568 }
569 \def\readlist_d#1#2#3{%
570 \unexpanded\expandafter\expandafter\expandafter{\csname#3[#2]\expandafter\endcsname\expandafter}%
571 \expanded{\loi_ifempty{#1}{\unexpanded{\unexpanded{\loi_error{#1}}}}}%
572 }
573 \def\readlist_e{%
574 \loi_argcsname\loi_let\loi_currentsep{\loi_listofsep[\number\loi_nestcnt]}%
575 \expandafter\readlist_f\loi_currentsep|\_nil
576 }
577 \long\def\readlist_f#1|#2\_nil#3{\readlist_g1{#3#1}}% #1=<sep courant simple> #3=liste -> rajoute un élément vide ✓
pour le test \ifempty ci dessous
578 \long\def\readlist_g#1#2{% #1=compteur d'index #2=liste d'éléments à examiner terminée par <sep courant simple> >>✓
RIEN laissé après
579 \loi_ifempty{#2}
580 {%
581 \loi_argcsname\loi_edef{\loi_listname len[\csname loi_previndex[\number\loi_nestcnt]\endcsname0]}{\number\numexpr/
#1-1\relax}%
582 \loi_argcsname\loi_let{\loi_listname sep[\csname loi_previndex[\number\loi_nestcnt]\endcsname\number\numexpr#1-1\
relax]}\empty% le dernier <sep> est <vide> ##NEW v1.52
583 \advance\loi_nestcnt-1
584 \loi_argcsname\loi_let\loi_currentsep{\loi_listofsep[\number\loi_nestcnt]}%
585 }
586 {%
587 \loi_expafter{\readlist_h{#2}{}}\loi_currentsep|\loi_quark|#2\_nil{#1}% aller isoler le 1er item
588 }%
589 }
590 \long\def\readlist_h#1#2#3|{% #1=liste restante #2=<dernier sep utilisé> #3=<sep courant>
591 \loi_ifx{\loi_quark#3}% on a épuisé tous les <séparateurs> ? RESTE à lire <expr+sep1>\_nil{compteur}
592 {%
593 \loi_ifempty{#2}% si #2 vide, aucun <sep utilisé> n'a été trouvé, il reste à lire "<liste complète>\_nil"
594 {%
595 \long\def\readlist_i##1\_nil##2{\loi_exparg{\readlist_j{##2}{}}{\loi_gobarg##1}{#2}}% ##2=compteur d'index
596 }
597 {%
598 \loi_ifx{\loi_listofpair\empty}% paires définies ?
599 {%
600 \long\def\readlist_i##1#2##2\_nil##3{\loi_exparg{\readlist_j{##3}{##2}}{\loi_gobarg##1}{#2}}%
601 }
602 {%
603 \long\def\readlist_i##1\_nil##2{%
604 \loi_exparg{\loi_exparg\loi_grabpaired_expr\loi_listofpair}{\loi_gobarg##1}{#2}\loi_grabpaired_result\
loi_grabpaired_remain
605 \loi_exparg{\loi_exparg{\readlist_j{##2}}{\loi_grabpaired_remain}}{\loi_grabpaired_result}{#2}}%
606 }%
607 }%
608 \readlist_i\relax% le \relax meuble l'argument délimité
609 }
610 {%
611 \long\def\readlist_i##1#3##2\_nil{%
612 \loi_ifempty{##2}% si <liste restante> ne contient pas le <sep courant>
613 {%
614 \readlist_h{#1}{#2}% recommencer avec le même <sep utile>
615 }%
616 {%
617 \loi_ifx{\loi_listofpair\empty}% si pas de paires définies
618 }%

```

```

619 % ##BUGFIX v1.53 (manger le \relax)
620 % ##BUGFIX v 1.64 (##2 n'étant pas vide, ne pas ajouter #3 après ##1
621 %           puisque #3 subsiste toujours avant le \_nil)
622 \loi_exparg\readlist_h{\loi_gobarg##1}{#3}% raccourcir <liste restante> et <sep courant>:=<sep utile>
623 }%
624 {%
625 \loi_exparg\loi_grabpaired_expr\loi_listofpair{#1#3}{#3}\loi_grabpaired_result\loi_grabpaired_remain
626 \loi_ifx{\loi_grabpaired_remain\empty}% si liste non raccourcie #BUGFIX 1.63
627 {\loi_exparg\readlist_h{\loi_grabpaired_result}{#2}}% garder le précédent <sep>
628 {\loi_exparg\readlist_h{\loi_grabpaired_result}{#3}}%
629 }%
630 }%
631 }%
632 \readlist_i\relax#1#3\_nil% ##BUGFIX v1.53 (ajouter \relax)
633 }%
634 }
635 \long\def\readlist_j#1#2#3{% #1=compteur d'index #2=liste restante #3=élément courant
636 \loi_ifnum{0\loi_exparg\loi_ifspacefirst{\loi_currentsep}{1}\if_loi_removeextremespaces1\fi=11 }% s'il faut retirer ✓
637   les espaces extrêmes
638   {%
639   \loi_exparg{\loi_exparg{\readlist_k{#1}{#2}}}{\loi_removeextremespaces{#3}}% redéfinir l'élément courant
640   }%
641   {%
642   \readlist_k{#1}{#2}{#3}%
643   }%
644 }
645 \long\def\readlist_k#1#2#3#4{% #1=compteur d'index #2=liste restante #3=élément courant #4=sep utilisé
646 \loi_ifnum{0\if_loi_ignoreemptyitems1\fi\loi_ifempty{#3}1}=11 }
647   {%
648   \readlist_g{#1}{#2}% si l'on n'ignore pas les éléments vides
649   }%
650   {%
651   \loi_argcsname\loi_def{\loi_listname[\csname loi_previndex[\number\loi_nestcnt]\endcsname#1]}{#3}% assignation
652   \loi_argcsname\loi_def{\loi_listname sep[\csname loi_previndex[\number\loi_nestcnt]\endcsname#1]}{#4}% assignation ✓
653   du <sep> actuel à la macro \<macrolist>sep
654   \loi_ifnum{\loi_nestcnt<\nestdepth\relax}% si imbrication max non atteinte
655   {%
656   \advance\loi_nestcnt1
657   \loi_argcsname\edef{\loi_previndex[\number\loi_nestcnt]}{\csname loi_previndex[\number\numexpr\loi_nestcnt-1]\endcsname#1,%
658   \readlist_e{#3}% recommencer avec l'élément courant
659   }
660   }%
661   \loi_exparg\readlist_g{\number\numexpr#1+1}{#2}% puis chercher l'élément suivant dans la liste restante
662   }%
663 }
664 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
665 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro \listlen %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
666 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
667 \def\listlen#1[#2]{%
668   \expanded{%
669     \loi_ifempty{#2}
670     {%
671       \csname\loi_macroname#1len[0]\endcsname
672     }
673     {%
674       \loi_exparg\listlen_a{\expanded{\loi_macroname#1}}{#2}%
675     }
676   }%
677 }
678 \def\listlen_a#1#2{% #1=macro name #2=index non normalisé prendre <profondeur max-1>
679 \expandafter\listlen_b\expanded{\loi_normalizeindex1{#1}{#2}}{#1}%
680 }
681 \def\listlen_b#1#2#3{% #1=err #2=index normalisé #3=macroname
682 \csname#3len[#2,0]\expandafter\endcsname
683 \expanded{%
684 \loi_ifempty{#1}
685 }%

```

```

686     }
687     {%
688     \unexpanded{\unexpanded{\loi_error{#1}}}%
689     }%
690 }%
691 }
692
693 %%%%%%%%%%% macro \foreachitem %%%%%%%%%%%
694 %%%%%%%%%%% macro \foreachitem %%%%%%%%%%%
695 %%%%%%%%%%%
696 \def\foreachitem#1\in#2{%
697 \edef\foreachitem_a{\noexpand\foreachitem_c\noexpand#1{\expandafter\noexpand\csname\loi_macroname#1cnt\endcsname}{\loi_macroname#2}}%
698 \futurelet\loi_nxttok\foreachitem_b
699 }
700 \def\foreachitem_b{\loi_ifx{\loi_nxttok[]}\foreachitem_a{\foreachitem_a[]}}
701 \def\foreachitem_c#1#2#3[#4]{% prendre <profondeur max-1>
702 \expandafter\foreachitem_d\expanded{\loi_normalizeindex1{#3}{#4}}#1{#2}{#3}%
703 }
704 \def\foreachitem_d#1#2{%
705 \loi_ifempty{#2}
706     {%
707     \foreachitem_e{#1}{}%
708     }
709     {%
710     \foreachitem_e{#1}{#2,}% #1=err #2=index norm
711     }%
712 }%
713 \long\def\foreachitem_e#1#2#3#4#5#6{% #1=err #2=index norm #3=macroiter #4=compteur associé #5=nom de macrolist #6=code
714 \loi_ifnum{\csname#5len[#20]\endcsname>0 }
715     {%
716     \loi_ifempty{#1}
717         {%
718         }
719         {%
720         \loi_error{#1}%
721         }%
722         \loi_fornum#4=1to\csname#5len[#20]\endcsname\do{\loi_argcsname\let#3{#5[#2#4]}#6}%
723         }
724     }%
725 }%
726 }
727
728 %%%%%%%%%%% macro \showitem %%%%%%%%%%%
729 %%%%%%%%%%% macro \showitem %%%%%%%%%%%
730 %%%%%%%%%%%
731 \def\showitems{%
732 \loi_ifstar
733     {%
734     \let\showitems_cmd\detokenize
735     \showitems_a
736     }
737     {%
738     \let\showitems_cmd\loi_identity
739     \showitems_a
740     }%
741 }
742 \def\showitems_a#1{%
743 \def\showitems_b{\showitems_d#1}%
744 \futurelet\loi_nxttok\showitems_c
745 }
746 \def\showitems_c{%
747 \loi_ifx{\loi_nxttok[]
748     {%
749     \showitems_b
750     }
751     {%
752     \showitems_b[]}%
753 }

```

```

754 \def\showitems_d#1[#2]{%
755   \foreachitem\showitems_iter\in#1[#2]
756     {%
757       \showitemsmacro{\expandafter\showitems_cmd\expandafter{\showitems_iter}}%
758     }%
759 }
760 \unless\ifdefined\fbbox
761   \newdimen\fbboxrule
762   \newdimen\fbboxsep
763   \fbboxrule=.4pt
764   \fbboxsep=3pt % réglages identiques à LaTeX
765   \def\fbbox#1{% imitation de la macro \fbbox de LaTeX, voir pages 271 à 274 de "Apprendre à programmer en TeX"
766     \hbox{%
767       \vrule width\fbboxrule
768       \vtop{%
769         \vbox{\hrule height\fbboxrule \kern\fbboxsep \hbox{\kern\fbboxsep#1\kern\fbboxsep}}%
770         \kern\fbboxsep \hrule height\fbboxrule
771       }%
772       \vrule width\fbboxrule
773     }%
774 }
775 \fi
776 \def\showitemsmacro#1{% encadrement par défaut
777   \beginngroup
778     \fbboxsep=0.25pt
779     \fbboxrule=0.5pt
780     \fbbox{\strut#1}%
781   \endgroup
782   \hskip0.25em\relax
783 }
784
785 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
786 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro \itemtomacro %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
787 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
788 \def\itemtomacro#1[#2]{% #1[#2]=item non encore lu: #3=macro
789   \edef\loi_listname{\loi_macroname#1}%
790   \expandafter\itemtomacro_a\expanded{\loi_normalizeindex0{\loi_listname}{#2}}\let
791 }
792 \def\gitomacro#1[#2]{% #1[#2]=item
793   \xdef\loi_listname{\loi_macroname#1}%
794   \expandafter\itemtomacro_a\expanded{\loi_normalizeindex0{\loi_listname}{#2}}{\global\let}%
795 }
796 \def\itemtomacro_a#1#2#3#4{%
797   \loi_ifempty{#1}{}\loi_error{#1}%
798   \loi_argc#3#4{\loi_listname[#2]}%
799 }
800
801 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
802 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% réglages par défaut %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
803 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
804 \newif\if_loi_removeextremespaces
805 \newif\if_loi_ignoreemptyitems
806 \let\ignoreemptyitems\_loi_ignoreemptyitemstrue
807 \let\reademptyitems\_loi_ignoreemptyitemsfalse
808 \loi_def_foreachsep{,}
809 \loi_restorecatcode
810 \reademptyitems
811 \setsepchar{,}
812 \defpair{}
813 \endinput
814
815 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
816 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% historique %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
817 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
818 v1.0 19/8/2016
819 - Première version publique
820 -----
821 v1.1 01/09/2016
822 - Stockage des séparateurs dans <macrolist>sep
823 - bug corrigé dans \loi_restorecatcode

```

824 -----

825 v1.2 22/10/2016

826 - macros `\readlist` et `\gitemtomacro` pour la globalité

827 -----

828 v1.3 18/11/2016

829 - bugs corrigés dans la gestion de la globalité

830 -----

831 v1.4 05/10/2017

832 - test `\loi_ifprimitive` ajouté au test `\loi_ifcs`

833 - suppression de `\loi_expafatternil`, création de `\loi_expafter`,

834 modification de `\loi_argcsname`

835 - correction d'un bug : `\setsepchar{\par}` ne provoque plus d'erreur.

836 `\loi_ifnum` devient `\long`

837 -----

838 v1.5 06/10/2017

839 - correction d'un bug dans `\loi_ifcs`

840 -----

841 v1.51 24/10/2017

842 - correction d'un bug dans `\loi_ifcs`

843 -----

844 v1.52 13/01/2018

845 - le dernier séparateur est <vide>

846 -----

847 v1.53 13/03/2018

848 - correction d'un bug dans `\readlist_i`

849 -----

850 v1.6 01/11/2018

851 - possibilité d'appariement de tokens dans les items

852 -----

853 v1.61 03/03/2019

854 - la macro `\loi_ifcs` contient une erreur de conception. Il faut

855 tester si le token est un `sc` && s'il est développable pour

856 renvoyer vrai car il existe des `sc` non développables && qui ne

857 sont `_pas_` des primitives.

858 Macro rebaptisée `\loi_ifcsexpandable`

859 -----

860 v1.62 18/05/2019

861 - utilisation de la nouvelle primitive `\expanded` au lieu du

862 désormais obsolète `\romannumeral`

863 - bug corrigé dans `\loi_ifcsexpandable`

864 -----

865 v1.63 21/08/2019

866 - bug corrigé dans `\readlist_h` avec les tokens appariés

867 - bug corrigé `\loi_removefirstspaces` est désormais `\long`

868 -----

869 v1.64 10/02/2024

870 - bug corrigé dans `\readlist_h`. Bug découvert plus d'un an après (sic)

871 avoir été signalé dans la question :

872 `tex.stackexchange.com/questions/670379`

873 et donc désormais, le code suivant ne plante plus

874 `\setsepchar{10||00}%`

875 `\readlist\mylist{A10B}%`

876 - bug corrigé dans `\loi_normalizeindex` : la profondeur maxi de

877 l'index peut être diminuée de 1 (pour notamment `\foreachitem` et

878 `\listlen`)

879 - la primitive `\expanded` est désormais obligatoire

880 - code source UFT8, manuel compilé avec luaLaTeX

881 - code plus lisiblement formaté

882 -----

883 v1.65 09/03/2024

884 - bug corrigé dans la définition de `\loi_count_occur`, l'argument ne

885 doit `_pas_` être délimité par `"` :