

L'extension

listofitems

v1.6
1^{er} novembre 2018

Christian TELLECHEA^{*}
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

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

```
\usepackage{listofitems}
```

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

```
\input listofitems.tex
```

2 Lire une liste simple

Définir le séparateur Le $\langle \textit{séparateur} \rangle$ 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{ $\langle \textit{séparateur} \rangle$ }`. Un $\langle \textit{séparateur} \rangle$ 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 $\langle \textit{séparateur} \rangle$. Des commandes peuvent se trouver dans cet ensemble de tokens, y compris la primitive `\par`.

Le $\langle \textit{séparateur} \rangle$ « / » 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 $\langle \textit{séparateur} \rangle$ d'une liste simple, il faut, à l'aide de l'argument optionnel, choisir un autre $\langle \textit{séparateur} \rangle$ de listes imbriquées, par exemple « . » et écrire « `\setsepchar[.]{/}` ».

Il n'est pas possible de choisir « | » comme $\langle \textit{séparateur} \rangle$ 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{ $\langle \textit{macroliste} \rangle$ }{ $\langle \textit{liste} \rangle$ }` doit être appelée. Ce faisant, la $\langle \textit{liste} \rangle$ est lue et les éléments sont stockés dans une macro, notée $\langle \textit{macroliste} \rangle$ qui dès lors, se comporte comme un tableau composé des éléments de la $\langle \textit{liste} \rangle$. 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 $\langle \textit{macroliste} \rangle$ nommée `\foo`, on peut écrire

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

Si la $\langle \textit{liste} \rangle$ est contenue dans une macro, alors cette macro est développée par `listofitems`. On peut donc écrire `\readlist{ $\langle \textit{macroliste} \rangle$ }{ $\langle \textit{macro} \rangle$ }` 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 par conséquent, la $\langle \textit{macroliste} \rangle$ 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]` est³ « 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 $\langle \textit{liste} \rangle$ 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 $\langle \textit{liste} \rangle$ 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>`len contient⁴ 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>12</code> <code>abc</code> <code>x y</code> <code>z</code> <code>TeX</code> <code>!</code>
<code>\showitems*\foo</code>	<code>12</code> <code>abc</code> <code>x y</code> <code>{\bfseries z}</code> <code>TeX</code> <code>!</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 \fbbox{\strut#1}\endgroup
  \hskip0.25em\relax}
```

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

La macro `\fbbox` 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>12</code> <code>abc</code> <code>x y</code> <code>z</code> <code>TeX</code> <code>!</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

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.

extrêmes intervient *avant* que listofitems n'ignore les éléments vides :

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

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

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

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

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

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 *séparateurs*, chacun correspondant à un niveau d'imbrication. Pour déclarer une *liste de séparateurs* il faut définir le *séparateur* de cette *liste de séparateurs* à l'aide de l'argument optionnel de la macro `\setsepchar` et écrire `\setsepchar[<separateur>]{<liste des séparateurs>}`. Par défaut, le *séparateur* est « / ». Ainsi, si l'on donne l'ordre

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

on indique une profondeur récursive de 3 et on choisit comme séparateur de la *liste des séparateurs* 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 *profondeur* 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` ;
- [*i*] désigne l'élément n° *i* de la liste principale ;
- [*i*], [*j*] désigne l'élément n° *j* de la liste constituée par l'élément évoqué au point précédent ;

- [*i*], [*j*], [*k*] désigne l'élément n° *k* 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]</code> \par	b) <code>\baz[1,1]</code> est 1
b) <code>\string\baz[1,1]</code> est <code>\baz[1,1]</code> \par	c) <code>\baz[1,1,1]</code> est 1
c) <code>\string\baz[1,1,1]</code> est <code>\baz[1,1,1]</code> \par	b) <code>\bar[1,2]</code> est 2 a b
b) <code>\string\bar[1,2]</code> est <code>\baz[1,2]</code> \par	e) <code>\baz[1,2,3]</code> est b
e) <code>\string\baz[1,2,3]</code> est <code>\baz[1,2,3]</code> \par	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 : <code>\nombres[1]</code> \par	Terme 1 : 1
Terme 2 : <code>\nombres[2]</code> (facteurs : <code>\nombres[2,1]</code> et <code>\nombres[2,2]</code>)\par	Terme 2 : 2*3 (facteurs : 2 et 3)
Terme 3 : <code>\nombres[3]</code> (facteurs : <code>\nombres[3,1]</code> , <code>\nombres[3,2]</code> et <code>\nombres[3,3]</code>)	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 2 développements le même résultat que `<macrolist>len` qui le donne en 1.

<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) <code>\bazlen\</code> ou <code>\listlen\baz[]</code> \par	b) 3
b) <code>\listlen\baz[1]</code> \par	c) 3
c) <code>\listlen\baz[2]</code> \par	d) 5
d) <code>\listlen\baz[3]</code> \par	e) 1
e) <code>\listlen\baz[3,1]</code> \par	f) 2
f) <code>\listlen\baz[3,4]</code> \par% 2 éléments vides	g) 3
g) <code>\listlen\baz[3,5]</code>	

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

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.

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

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>].

<pre> \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 \itemtomacro\poeme[2,-4]\mot Un mot = \mot </pre>	<pre> 2e vers = Que le vent du matin vient glacer à mon front. Un mot = glacer </pre>
--	---

La macro `\gitemtomacro` 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.

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

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 $\langle tok1 \rangle \langle tok2 \rangle$ dans l' $\langle expression \rangle$.

```

\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\inmallparen
"\inmallparen"

```

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

Toute suggestion, remontée de bug, remarque, demande, ajout ou modification de fonctionnalité est bienvenue ; dans ce cas, j'invite les utilisateurs de `listofitems` à m'envoyer un email à `unbonpetit@netc.fr`.

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 % !TeX encoding = ISO-8859-1
2 % Ce fichier contient le code de l'extension "listofitems"
3 %
4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
5 %
6 \def\loiname           {listofitems}
7 \def\loiver            {1.6}
8 %
9 \def\loidate           {2018/11/01}
10 %
11 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
12 %
13 % Author      : Christian Tellechea, Steven B. Segletes
14 % Status     : Maintained
15 % Maintainer : Christian Tellechea
16 % Email      : unbonpetit@netc.fr
17 %             steven.b.segletes.civ@mail.mil
18 % Package URL: https://www.ctan.org/pkg/listofitems
19 % Bug tracker: https://framagit.org/unbonpetit/listofitems/issues
20 % Repository : https://framagit.org/unbonpetit/listofitems/tree/master
21 % Copyright  : Christian Tellechea 2016-2018
22 % Licence    : Released under the LaTeX Project Public License v1.3c
23 %             or later, see http://www.latex-project.org/lppl.txt
24 % Files      : 1) listofitems.tex
25 %             2) listofitems.sty
26 %             3) listofitems-fr.tex
27 %             4) listofitems-fr.pdf
28 %             5) listofitems-en.tex
29 %             6) listofitems-en.pdf
30 %             7) README
31 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
32 \ifdefined\ProvidesPackage\else
33 \immediate\write -1 {%
34 Package: \loidate\space v\loiver\space Grab items in lists using user-specified sep char (CT)
35 }%
36 \fi

```

```

37 \expandafter\edef\csname loi_restorecatcode\endcsname{\catcode\number‘\_=\number\catcode‘\_ \
    relax}
38 \catcode‘\_11
39
40 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
41 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% gestion des erreurs %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
42 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
43 \ifdefined PackageError
44 \def\loi_error#1{\PackageError\loiname{#1}{Read the manual}}% pour LaTeX
45 \else
46 \def\loi_error#1{\errmessage{Package \loiname space Error: #1^^J}}% pour TeX
47 \fi
48
49 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
50 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% vérification de la présence de etex %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
51 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
52 \begingroup
53 \edef__tempa{\meaning\TeXversion}\edef__tempb{\string\TeXversion}%
54 \ifx__tempa__tempb
55 \endgroup
56 \else
57 \endgroup
58 \loi_error{You are not using an eTeX engine, listofitems cannot work.}%
59 \loi_restorecatcode\expandafter\endinput
60 \fi
61
62 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
63 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macros auxiliaires %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
64 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
65 \chardef\loi_stop=0
66 \def\loi_quark{\loi_quark}
67 \long\def\loi_identity#1{#1}
68 \long\def\loi_gobarg#1{}
69 \long\def\loi_first#1#2{#1}
70 \long\def\loi_second#1#2{#2}
71 \long\def\loi_firsttonil#1#2\_nil{#1}
72 \long\def\loi_antefi#1#2\fi{#2\fi#1}
73 \long\def\loi_exparg#1#2{\expandafter\loi_exparg_a\expandafter{#2}{#1}}% \loi_exparg{<a>}{<b>} ↵
    devient <a>{<b>}
74 \long\def\loi_exparg_a#1#2{#2{#1}}
75 \long\def\loi_expafter#1#2{\expandafter\loi_expafter_a\expandafter{#2}{#1}}% \loi_expafter{<a ↵
    >}{<b>} devient <a><b>
76 \long\def\loi_expafter_a#1#2{#2#1}
77 \def\loi_macroname{\loi_ifinrange\escapechar[[0:255]]{\expandafter\loi_gobarg}{}\string}
78 \def\loi_argcsname#1#{\loi_argcsname_a{#1}}
79 \def\loi_argcsname_a#1#2{\loi_expafter{#1}{\csname#2\endcsname}}
80 \long\def\loi_addtomacro#1#2{\loi_exparg{\def#1}{#1#2}}
81
82 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
83 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macros de test %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
84 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
85 \long\def\loi_ifnum#1{\ifnum#1\expandafter\loi_first\else\expandafter\loi_second\fi}
86 \long\def\loi_ifx#1{\ifx#1\expandafter\loi_first\else\expandafter\loi_second\fi}
87 \long\def\loi_ifempty#1{\loi_exparg\loi_ifx{\expandafter\relax\detokenize{#1}\relax}}
88 \def\loi_ifstar#1#2{\def\loi_ifstar_a{\loi_ifx{*}\loi_nxttok}{\loi_first{#1}}{#2}}\futurelet\ ↵
    loi_nxttok\loi_ifstar_a}
89 \long\def\loi_ifprimitive#1{\edef\loi_tempa{\meaning#1}\edef\loi_tempb{\string#1}\loi_ifx{\ ↵
    loi_tempa\loi_tempb}}
90 \long\def\loi_ifcs#1{% #1 est-il une sc (n'étant pas une primitive) ?
91 \loi_ifempty{#1}
92 {\loi_second

```



```

93 }
94 {\loi_ifspacefirst{#1}
95   \loi_second% si espace en 1er, faux
96   {\loi_exparg\loi_ifempty{\loi_gobarg#1}% 1 seul token ?
97     {\begingroup \escapechar'\_
98       \if\expandafter\loi_firsttonil\detokenize{#1}\_nil\expandafter\loi_firsttonil\string\
99         relax\_nil
100       \loi_ifprimitive{#1}
101         {\endgroup\expandafter\loi_second}
102         {\endgroup\expandafter\loi_first}%
103       \else
104         \endgroup\expandafter\loi_second
105       \fi
106     }
107     {\loi_second% si plusieurs tokens, faux
108     }%
109   }%
110 }
111 \def\loi_ifinrange#1[[#2:#3]]{\loi_ifnum{\numexpr(#1-#2)*(#1-#3)>0 }\loi_second\loi_first}
112 \def\loi_ifstring#1\in#2{% si la chaine #1 est contenue dans #2
113   \def\loi_ifstring_a##1#1##2\_nil{\loi_ifempty{##2}\loi_second\loi_first}%
114   \loi_ifstring_a#2#1\_nil% appel de la macro auxiliaire
115 }
116
117 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
118 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macros \loi_foreach %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
119 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
120 \newcount\loi_cnt_foreach_nest \loi_cnt_foreach_nest=0
121 \def\end_foreach{\end_foreach}
122 \def\loi_def_foreachsep#1{%
123   \long\def\loi_foreach##1\in##2##3{%
124     \global\advance\loi_cnt_foreach_nest1
125     \loi_argcsname\def{loop_code_\number\loi_cnt_foreach_nest}{##3}%
126     \loi_foreach_a##1##2#1\end_foreach#1%
127     \loi_argcsname\let{loop_code_\number\loi_cnt_foreach_nest}\empty
128     \global\advance\loi_cnt_foreach_nest-1
129   }%
130   \long\def\loi_foreach_a##1##2#1{%
131     \def##1{##2}%
132     \loi_ifx{\end_foreach##1}
133     {}
134     {\csname loop_code_\number\loi_cnt_foreach_nest\endcsname% exécute le code
135     \loi_foreach_a##1%
136     }%
137   }%
138 }
139
140 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
141 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macros gérant l'appariement %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
142 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
143 \long\def\defpair#1{%
144   \let\loi_listofpair\empty
145   \loi_ifempty{#1}
146   {}
147   {\defpair_a#1\loi_quark\loi_quark}%
148 }
149 \long\def\defpair_a#1#2#3{%
150   \loi_ifx{\loi_quark#2}
151   {\def\loi_sanitizelist##1,\_nil{\def\loi_listofpair{##1}}%
152   \loi_sanitizelist#1\_nil

```

```

153 }
154 {\loi_if_validpair#2#3%
155   {\long\def\loi_paired_a{#2}\long\def\loi_paired_b{#3}%
156   \loi_ifx{\loi_paired_a\loi_paired_b}
157     {\loi_error{Paired tokens must not be equal, the pair \detokenize{#2#3} is ignored}%
158     \defpair_a{#1}%
159     }
160   {\defpair_a{#1#2#3,}%
161   }%
162 }
163 {\loi_error{Invalid paired tokens, the pair "\detokenize{#2}" and "\detokenize{#3}" is
164   ignored}%
165   \defpair_a{#1}%
166   }%
167 }
168 \long\def\loi_if_validpair#1#2{%
169   \def\loi_validpair{1}%
170   \loi_if_invalid_paiRToken{#1}{\def\loi_validpair{0}}%
171   \loi_if_invalid_paiRToken{#2}{\def\loi_validpair{0}}%
172   \loi_ifnum{\loi_validpair=1 }
173 }
174 \long\def\loi_if_invalid_paiRToken#1{%
175   \loi_ifempty{#1}
176     {\loi_identity
177     }
178     {\loi_ifspacefirst{#1}
179     {\loi_identity
180     }
181     {\loi_exparg\loi_ifempty{\loi_gobarg#1}% 1 seul token ?
182     {\ifcat\relax\noexpand#1\expandafter\loi_identity\else\expandafter\loi_gobarg\fi}
183     {\loi_identity}% si plusieurs tokens, faux
184     }%
185   }%
186 }
187 \long\def\loi_count_occur#1\in#2:#3{% compte le nombre d'occurrences de #1 dans #2 et met le
188   résultat dans la macro #3
189   \long\def\loi_count_occur_a##1##2#1##3\_nil{%
190     \loi_ifempty{##3}
191     {\def#3{##1}}
192     {\expandafter\loi_count_occur_a\number\numexpr##1+1\relax##3\_nil}%
193   }%
194   \loi_count_occur_a0#2#1\_nil
195 }
196 \long\def\loi_check_pair#1#2\in#3{% teste l'appariement de #1 et #2 dans #3
197   \loi_ifempty{#3}
198     {\loi_second
199     }
200     {\loi_count_occur#1\in#3:\loi_tempa
201     \loi_count_occur#2\in#3:\loi_tempb
202     \loi_ifnum{\loi_tempa=\loi_tempb\relax}%
203     }%
204 }
205 \long\def\loi_grabpaired_expr#1#2#3#4#5{% #1=liste de paires #2=expression #3=séparateur
206   #4=résultat #5=ce qui reste
207   \let#4\empty
208   \def\loi_remain{#2#3}%
209   \loi_foreach\loi_pair\in{#1}{\expandafter\loi_grabpaired_expr_a\loi_pair{#3}#4}%
210   \def\loi_remove_lastsep##1#3\_nil{\def#4{##1}}%
211   \expandafter\loi_remove_lastsep#4\_nil
212   \expandafter\long\expandafter\def\expandafter\loi_grab_remain#4##1\_nil{\loi_ifempty{##1}{\

```

```

let#5\empty}{\loi_exparg{\def#5}{\loi_gobarg#1}}}%
211 \loi_grab_remain#2\_nil
212 }
213 \long\def\loi_grabpaired_expr_a#1#2#3#4{% #1#2=paire en cours #3=séparateur #4=résultat
214 \loi_exparg{\loi_check_pair#1#2\in}#4% si les paires sont appariées dans le résultat
215 {}% passer à la paire suivante
216 {\long\def\loi_grabpaired_expr_b##1#3##2\_nil{%
217 \loi_addtomacro#4{##1#3}% ajouter au résultat ce qui est jusqu'au prochain séparateur
218 \def\loi_remain{##2}%
219 \loi_exparg{\loi_check_pair#1#2\in}{#4}
220 {}
221 {\loi_ifempty{##2}
222 {\loi_error{"\detokenize{#1}" and "\detokenize{#2}" are not paired}}
223 {\loi_grabpaired_expr_b##2\_nil}%
224 }%
225 }%
226 \expandafter\loi_grabpaired_expr_b\loi_remain\_nil
227 }%
228 }
229 \def\insidepair#1#2#3#4{% #1#2=paire #3=expr #4=macro recevant le resultat
230 \loi_if_validpair#1#2%
231 {\loi_ifcs{#3}
232 {\loi_exparg{\insidepair#1#2}{#3}#4%
233 }
234 {\loi_check_pair#1#2\in{#3}% si les paires sont appariées dans le résultat
235 {\def\insidepair_a##1#1##2\_nil{\insidepair_b##2\_nil{#1}}%
236 \def\insidepair_b##1#2##2\_nil##3{%
237 \loi_check_pair#1#2\in{##3##1#2}
238 {\loi_exparg{\def#4}{\loi_gobarg##3##1}}%
239 {\insidepair_b##2\_nil{##3##1#2}}%
240 }%
241 \insidepair_a#3\_nil
242 }
243 {\loi_error{"\detokenize{#1}" and "\detokenize{#2}" are not paired in "#3"}%
244 }%
245 }%
246 }
247 {\loi_error{Invalid paired tokens "\detokenize{#1}" and "\detokenize{#2}", empty \string#4 ↵
248 returned}% et bim
249 \let#4\empty% voilà, bien fait pour vos gueules
250 }%
251 }
252 %%%
253 %%% macro \loi_fornum %%%
254 %%%
255 \def\loi_fornum#1=#2to#3\do{%
256 \edef#1{\number\numexpr#2}\edef\loi_sgncmp{\ifnum#1<\numexpr#3\relax>+\else<-\fi}%
257 \expandafter\loi_fornum_a\cname\loi_fornum\_string#1\expandafter\endcsname\expandafter{\↵
258 \number\numexpr#3\expandafter}\loi_sgncmp#1%
259 }
260 \long\def\loi_fornum_a#1#2#3#4#5#6{\def#1{\unless\ifnum#5#3#2\relax\loi_antefi{#6\edef#5{\↵
261 \number\numexpr#5#41\relax}#1}\fi}#1}
262 %%%
263 %%% macro retirant les espaces extrêmes %%%
264 \long\def\loi_ifspacefirst#1{\expandafter\loi_ifspacefirst_a\detokenize{#10} \_nil}
265 \long\def\loi_ifspacefirst_a#1 #2\_nil{\loi_ifempty{#1}}
266 \expandafter\def\expandafter\loi_gobspace\space{}
267 \def\loi_removefirstspaces{\romannumeral\loi_removefirstspaces_a}

```

```

268 \long\def\loi_removefirstspaces_a#1{\loi_ifspacefirst{#1}{\expandafter\loi_removefirstspaces_a\
    \expandafter{\loi_gobspace#1}}{\loi_stop#1}}
269 \edef\loi_restorezerocatcode{\catcode0=\number\catcode0 \relax}
270 \catcode0 12
271 \long\def\loi_removelastspaces#1{\romannumeral\loi_removelastspaces_a#1^^00 ^^00\_nil}
272 \long\def\loi_removelastspaces_a#1 ^^00{\loi_removelastspaces_b#1^^00}
273 \long\def\loi_removelastspaces_b#1^^00#2\_nil{\loi_ifspacefirst{#2}{\loi_removelastspaces_a\
    #1^^00 ^^00\_nil}}{\loi_stop#1}}
274 \loi_restorezerocatcode
275 \long\def\loi_removeextremespaces#1{% #1=texte où les espaces extrêmes sont retirés
276 \romannumeral\expandafter\expandafter\expandafter\loi_removelastspaces\expandafter\
    \expandafter\expandafter
277 {\expandafter\expandafter\expandafter\loi_stop\loi_removefirstspaces{#1}}%
278 }
279
280 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
281 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro publique \setsepchar %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
282 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
283 \def\setsepchar{\futurelet\loi_nxttok\setsepchar_a}
284 \def\setsepchar_a{\loi_ifx{[\loi_nxttok]\setsepchar_b{\setsepchar_b[/]}}
285 \long\def\setsepchar_b[#1]#2{% #1=sepcar de <liste des sepcar> #2=<liste des sepcar>
286 \loi_ifempty{#1}
287 {\loi_error{Empty separator not allowed, separator "/" used}%
288 \setsepchar_b[/]{#2}%
289 }
290 {\def\loi_currentsep{#1}%
291 \removeextremespacesfalse
292 \loi_nestcnt1 % réinitialiser niveau initial à 1
293 \def\nestdepth{1}%
294 \loi_argcsname\let\loi_previndex[\number\loi_nestcnt]\empty
295 \def\loi_listname{\loi_listofsep}%
296 \let\loi_def\def \let\loi_edef\edef \let\loi_let\let
297 \let\loi_listofpair_saved\loi_list_ofpair
298 \let\loi_list_ofpair\empty
299 \loi_ifempty{#2}
300 {\loi_error{Empty list of separators not allowed, "," used}%
301 \readlist_e1{,}%
302 }
303 {\readlist_e1{#2}%
304 }%
305 \loi_argcsname\let\nestdepth{\loi_listofseplen[0]}%
306 \loi_argcsname\let\loi_currentsep{\loi_listofsep[1]}% 1er car de séparation
307 \let\loi_listofpair\loi_listofpair_saved
308 }%
309 }
310
311 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
312 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro normalisant l'index %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
313 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
314 \def\loi_normalizeindex#1#2#3{% #1=macroname #2=liste d'index #3=profondeur max --> renvoie {
    err}{indx norm}
315 \loi_ifempty{#2}
316 {\loi_stop}{}}
317 {\loi_normalizeindex_a1}{#3}{#1}#2, \loi_quark,%
318 }%
319 \def\loi_normalizeindex_a#1#2#3#4#5,% #1=compteur de profondeur #2=index précédents #3=
    profondeur max #4=macroname #5=index courant
320 \loi_ifx{\loi_quark#5}
321 {\loi_normalizeindex_c#2\loi_quark% supprimer la dernière virgule
322 }
323 {\loi_ifnum{#1>#3 }

```

```

324   {\loi_invalidindex{Too deeply nested index, index [.] retained}{#2}% si profondeur trop
      grande
325   }
326   {\loi_ifinrange\ifnum\numexpr#5<0 -1*\fi(#5)[[1:\csname #4len[#20]\endcsname]]% si abs
      (#5) hors de [1,len]
327   {\loi_exparg\loi_normalizeindex_b{\number\numexpr#5\ifnum\numexpr#5<0 +\csname #4len
      [#20]\endcsname+1\fi}{#1}{#2}{#3}{#4}}
328   {\loi_invalidindex{#5 is an invalid index, index [.] retained}{#2}}%
329   }%
330 }%
331 }
332 \def\loi_normalizeindex_b#1#2#3{\loi_exparg\loi_normalizeindex_a{\number\numexpr#2+1}{#3#1,}}%
      #1=index à rajouter #2=compteur de profondeur #3=index précédents
333 \def\loi_normalizeindex_c#1,\loi_quark{\loi_stop}{#1}}
334 \def\loi_invalidindex#1#2{\loi_ifempty{#2}{\loi_invalidindex_a{#1},}\loi_invalidindex_a
      {#1}{#2}}
335 \def\loi_invalidindex_a#1#2{\loi_invalidindex_b#1\loi_quark#2\loi_quark}
336 \def\loi_invalidindex_b#1[.]#2\loi_quark#3,\loi_quark#4\loi_quark,{\loi_stop{#1[#3]#2}{#3}}%
      #4= index ignorés
337
338 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
339 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro publique \readlist %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
340 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
341 \newcount\loi_nestcnt
342 \def\greadlist{\let\loi_def\gdef\let\loi_edef\xdef\def\loi_let{\global\let}\readlist_a}%
343 \def\readlist{\let\loi_def\def\let\loi_edef\edef\let\loi_let\let\readlist_a}
344 \def\readlist_a{%
345   \loi_nestcnt1 % niveau initial = 1
346   \loi_argcsname\let{\loi_previndex[\number\loi_nestcnt]}\empty
347   \loi_ifstar{\_removeextremespacestrue\readlist_b}{\_removeextremespacesfalse\readlist_b}%
348 }
349 \long\def\readlist_b#1#2{% #1=macro stockant les éléments #2=liste des éléments
      \loi_ifcs{#2}
350   {\loi_exparg{\readlist_b#1}{#2}%
351   }
352   {\loi_edef\loi_listname{\loi_macroname#1}%
      \loi_argcsname\loi_let{\loi_listname nest}\nestdepth
353   \loi_argcsname\loi_def{\loi_listname[]}{#2}% la liste entière
354   \loi_argcsname\loi_def{\loi_listname sep[]}{#2}% séparateur vide
355   \loi_ifempty{#2}
356   {\loi_def#1[#1]{}%
357   \loi_argcsname\loi_def{\loi_listname len}{0}\loi_argcsname\loi_def{\loi_listname len
      [0]}{0}%
358   \loi_error{Empty list ignored, nothing to do}%
359   }
360   {\loi_edef#1[#1]{\unexpanded{\romannumeral\expandafter\loi_checkindex\romannumeral\
      loi_normalizeindex}{\loi_listname}{##1}{\csname\loi_listname nest\endcsname}{\
      loi_listname}}%
361   \loi_argcsname\loi_edef{\loi_listname sep}[##1]{\unexpanded{\romannumeral\expandafter\
      loi_checkindex\romannumeral\loi_normalizeindex}{\loi_listname}{##1}{\csname\
      loi_listname nest\endcsname}{\loi_listname sep}}%
362   \readlist_c{#2}%
363   \loi_argcsname\loi_argcsname\loi_let{\loi_listname len}{\loi_listname len[0]}% longueur
      du niveau 0
364   }%
365 }%
366 }%
367 }%
368 }
369 \def\loi_checkindex#1#2#3{%
370   \expandafter\expandafter\expandafter\loi_stop\csname#3[#2]\expandafter\endcsname
371   \romannumeral\loi_ifempty{#1}{\loi_stop}{\loi_stop\loi_error{#1}}%
372 }

```

```

373 \def\readlist_c{%
374 \loi_argcsname\loi_let\loi_currentsep{\loi_listofsep[\number\loi_nestcnt]}%
375 \expandafter\readlist_d\loi_currentsep|\_nil
376 }
377 \long\def\readlist_d#1||#2\_nil#3{\readlist_e1{#3#1}}% #1=<sep courant simple> #3=liste -> ↵
    rajoute un élément vide pour le test \ifempty ci dessous
378 \long\def\readlist_e#1#2{% #1=compteur d'index #2=liste d'éléments à examiner terminée par <↵
    sep courant simple> >>RIEN laissé après
379 \loi_ifempty{#2}
380 {\loi_argcsname\loi_edef{\loi_listname len[\csname loi_previndex[\number\loi_nestcnt]\↵
    endcsname0}}{\number\numexpr#1-1\relax}%
381 \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
382 \advance\loi_nestcnt-1
383 \loi_argcsname\loi_let\loi_currentsep{\loi_listofsep[\number\loi_nestcnt]}%
384 }
385 {\loi_expafter{\readlist_f{#2}}}\loi_currentsep|\loi_quark|#2\_nil{#1}% aller isoler le ↵
    1er item
386 }%
387 }
388 \long\def\readlist_f#1#2#3|{|% #1=liste restante #2=<dernier sep utilisé> #3=<sep courant>
389 \loi_ifx{\loi_quark#3}% on a épuisé tous les <séparateurs> ? RESTE à lire <expr+sep1>\_nil{<↵
    compteur>}
390 {\loi_ifempty{#2}% si #2 vide, aucun <sep utilisé> n'a été trouvé, il reste à lire "<liste ↵
    complète>\_nil"
391 {\long\def\readlist_g##1\_nil##2{\loi_exparg{\readlist_h{##2}}}\loi_gobarg##1{#2}}% ↵
    ##2=compteur d'index
392 }
393 {\loi_ifx{\loi_listofpair\empty}% paires définies ?
394 {\long\def\readlist_g##1#2##2\_nil##3{\loi_exparg{\readlist_h{##3}}}\loi_gobarg ↵
    ##1{#2}}%
395 }
396 {\long\def\readlist_g##1\_nil##2{%
397 \loi_exparg{\loi_exparg\loi_grabpaired_expr\loi_listofpair}\loi_gobarg##1{#2}\↵
    loi_grabpaired_result\loi_grabpaired_remain
398 \loi_exparg{\loi_exparg{\readlist_h{##2}}}\loi_grabpaired_remain}\↵
    loi_grabpaired_result}{#2}}%
399 }%
400 }%
401 \readlist_g\relax% le \relax meuble l'argument délimité
402 }
403 {\long\def\readlist_g##1#3##2\_nil{%
404 \loi_ifempty{#2}% si <liste restante> ne contient pas le <sep courant>
405 {\readlist_f{#1}{#2}% recommencer avec le même <sep utile>
406 }%
407 {\loi_ifx{\loi_listofpair\empty}% si pas de paires définies
408 {\loi_exparg\readlist_f{\loi_gobarg##1#3}{#3}% raccourcir <liste restante> et <sep ↵
    courant>:=<sep utile>% ##BUGFIX v1.53
409 }%
410 {\loi_exparg\loi_grabpaired_expr\loi_listofpair{#1}{#3}\loi_grabpaired_result\↵
    loi_grabpaired_remain
411 \loi_exparg\readlist_f{\loi_grabpaired_result#3}{#3}%
412 }%
413 }%
414 }%
415 \readlist_g\relax#1#3\_nil% ##BUGFIX v1.53
416 }%
417 }
418 \long\def\readlist_h#1#2#3{% #1=compteur d'index #2=liste restante #3=élément courant
419 \loi_ifnum{0\loi_exparg\loi_ifspacefirst{\loi_currentsep}}1\if_removeextremespaces1\fi=11 }% ↵
    s'il faut retirer les espaces extrêmes

```

```

420 {\loi_exparg{\loi_exparg{\readlist_i{#1}{#2}}{\loi_removeextremespaces{#3}}}% redéfinir l'
élément courant
421 {\readlist_i{#1}{#2}{#3}}%
422 }
423 \long\def\readlist_i#1#2#3#4{% #1=compteur d'index #2=liste restante #3=élément courant #4=
sep utilisé
424 \loi_ifnum{0\if_ignoreemptyitems1\fi\loi_ifempty{#3}1{=11 }
425 {\readlist_e{#1}{#2}}% si l'on n'ignore pas les éléments vides
426 }%
427 {\loi_argcsname\loi_def{\loi_listname[\csname loi_previndex[\number\loi_nestcnt]\endcsname\
#1]}{#3}}% assignation de l'item ctuel à la macro
428 \loi_argcsname\loi_def{\loi_listname sep[\csname loi_previndex[\number\loi_nestcnt]\
endcsname#1]}{#4}}% assignation du <sep> actuel à la macro \macrolist>sep
429 \loi_ifnum{\loi_nestcnt<\nestdepth\relax}% si imbrication max non atteinte
430 {\advance\loi_nestcnt1
431 \loi_argcsname\edef{\loi_previndex[\number\loi_nestcnt]}{\csname loi_previndex[\number\
numexpr\loi_nestcnt-1]\endcsname#1,}%
432 \readlist_c{#3}}% recommencer avec l'élément courant
433 }
434 }%
435 \loi_exparg\readlist_e{\number\numexpr#1+1}{#2}}% puis chercher l'élément suivant dans la
liste restante
436 }%
437 }
438
439 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
440 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro \listlen %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
441 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
442 \def\listlen#1[#2]{%
443 \romannumeral\loi_ifempty{#2}
444 {\expandafter\expandafter\expandafter\loi_stop\csname\loi_macroname#1len[0]\endcsname}
445 {\loi_exparg\listlen_a{\romannumeral-'\. \loi_macroname#1}{#2}}%
446 }
447 \def\listlen_a#1#2{% #1=macro name #2=index non normalisé prendre <profondeur max-1>
448 \loi_exparg{\expandafter\listlen_b\romannumeral\loi_normalizeindex{#1}{#2}}{\number\numexpr\
csname#1nest\endcsname-1}{#1}}%
449 }
450 \def\listlen_b#1#2#3{% #1=err #2=index normalisé #3=macroname
451 \expandafter\expandafter\expandafter\loi_stop\csname#3len[#2,0]\endcsname
452 \romannumeral\loi_ifempty{#1}{\loi_stop}{\loi_stop\loi_error{#1}}%
453 }
454
455 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
456 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% macro \foreachitem %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
457 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
458 \def\foreachitem#1\in#2{%
459 \edef\foreachitem_a{\noexpand\foreachitem_c\noexpand#1{\expandafter\noexpand\csname\
loi_macroname#1cnt\endcsname}}{\loi_macroname#2}}%
460 \futurelet\loi_nxttok\foreachitem_b
461 }
462 \def\foreachitem_b{\loi_ifx{\loi_nxttok[]}\foreachitem_a{\foreachitem_a[]}}
463 \def\foreachitem_c#1#2#3[#4]{% prendre <profondeur max-1>
464 \loi_exparg{\expandafter\foreachitem_d\romannumeral\loi_normalizeindex{#3}{#4}}{\number\
numexpr\csname#3nest\endcsname-1}{#1}{#2}{#3}}%
465 }
466 \def\foreachitem_d#1#2{\loi_ifempty{#2}{\foreachitem_e{#1}{}}{\foreachitem_e{#1}{#2,}}}% #1=err
#2=index norm
467 \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
468 \loi_ifnum{\csname#5len[#20]\endcsname>0 }
469 {\loi_ifempty{#1}}{\loi_error{#1}}%

```

```

470 \loi_fornum#4=1to\csname#5len[#2]\endcsname\do{\loi_argcsname\let#3{#5[#2#4]}#6}%
471 }
472 {}%
473 }
474
475 %%%%%%%%%%
476 %%%%%%%%%% macro \showitem %%%%%%%%%%
477 %%%%%%%%%%
478 \def\showitems{\loi_ifstar{\let\showitems_cmd\detokenize\showitems_a}{\let\showitems_cmd\
loi_identity\showitems_a}}
479 \def\showitems_a#1{\def\showitems_b{\showitems_d#1}\futurelet\loi_nxttok\showitems_c}
480 \def\showitems_c{\loi_ifx{\loi_nxttok[]\showitems_b{\showitems_b[]}}
481 \def\showitems_d#1[#2]{\foreachitem\showitems_ater\in#1[#2]{\showitemsmacro{\expandafter\
showitems_cmd\expandafter{\showitems_ater}}}}
482 \unless\ifdefined\fbbox
483 \newdimen\fbboxrule \newdimen\fbboxsep \fbboxrule=.4pt \fbboxsep=3pt % réglages identiques à
LaTeX
484 \def\fbbox#1{% imitation de la macro \fbbox de LaTeX, voir pages 271 à 274 de "Apprendre à
programmer en TeX"
485 \hbox{%
486 \vrule width\fbboxrule
487 \vtop{%
488 \vbox{\hrule height\fbboxrule \kern\fbboxsep \hbox{\kern\fbboxsep#1\kern\fbboxsep}}%
489 \kern\fbboxsep \hrule height\fbboxrule
490 } \vrule width\fbboxrule
491 }%
492 }
493 \fi
494 \def\showitemsmacro#1{% encadrement par défaut
495 \begingroup\fbboxsep=0.25pt \fbboxrule=0.5pt \fbbox{\strut#1}\endgroup
496 \hskip0.25em\relax
497 }
498
499 %%%%%%%%%%
500 %%%%%%%%%% macro \itemtomacro %%%%%%%%%%
501 %%%%%%%%%%
502 \def\itemtomacro#1[#2]{% #1[#2]=item non encore lu: #3=macro
503 \edef\loi_listname{\loi_macroname#1}%
504 \loi_exparg{\expandafter\itemtomacro_a\romannumeral\expandafter\loi_normalizeindex\
expandafter{\loi_listname}{#2}}{\csname\loi_listname nest\endcsname}\let
505 }
506 \def\gitemtomacro#1[#2]{% #1[#2]=item
507 \xdef\loi_listname{\loi_macroname#1}%
508 \loi_exparg{\expandafter\itemtomacro_a\romannumeral\expandafter\loi_normalizeindex\
expandafter{\loi_listname}{#2}}{\csname\loi_listname nest\endcsname}{\global\let}%
509 }
510 \def\itemtomacro_a#1#2#3#4{%
511 \loi_ifempty{#1}{\loi_error{#1}}%
512 \loi_argcsname#3#4{\loi_listname[#2]}%
513 }
514
515 %%%%%%%%%%
516 %%%%%%%%%% réglages par défaut %%%%%%%%%%
517 %%%%%%%%%%
518 \newif\if_removeextremespaces
519 \newif\if_ignoreemptyitems
520 \let\ignoreemptyitems\_ignoreemptyitemstrue
521 \let\reademptyitems\_ignoreemptyitemsfalse
522 \setsepchar{,}
523 \defpair{}
524 \loi_def_foreachsep{,}

```



```

525 \reademptyitems
526
527 \loi_restorecatcode
528 \endinput
529
530 #####
531 ##### Historique #####
532 #####
533
534 v1.0 19/8/2016
535 - Première version publique
536
537 v1.1 01/09/2016
538 - Stockage des séparateurs dans <macro>list<sep
539 - bug corrigé dans \loi_restorecatcode
540
541 v1.2 22/10/2016
542 - macros \greadlist et \gitemtomacro pour la globalité
543
544 v1.3 18/11/2016
545 - bugs corrigés dans la gestion de la globalité
546
547 v1.4 05/10/2017
548 - test \loi_ifprimitive ajouté au test \loi_ifcs
549 - suppression de \loi_expafternil, création de \loi_expafter,
550 modification de \loi_argcsname
551 - correction d'un bug : \setsepchar{\par} ne provoque plus
552 d'erreur. \loi_ifnum devient \long
553
554 v1.5 06/10/2017
555 - correction d'un bug dans \loi_ifcs
556
557 v1.51 24/10/2017
558 - correction d'un bug dans \loi_ifcs
559
560 v1.52 13/01/2018
561 - le dernier séparateur est <vide>
562
563 v1.53 13/03/2018
564 - correction d'un bug dans \readlist_g
565
566 v1.6 01/11/2018
567 - possibilité d'appariement de tokens dans les items

```