

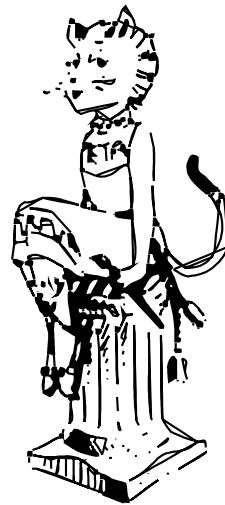
Voir en page x comment on peut imprimer ce manuel...

TOUT CE QUE VOUS AVEZ TOUJOURS
VOULU SAVOIR SUR

L^AT_EX

SANS JAMAIS O_S^E_R LE DEMANDER

..



Ou comment utiliser L^AT_EX quand
on n'y connaît goutte

Vincent LOZANO
lozano@enise.fr

Version du 5 mars 2008
Dernière mise à jour sur :
<http://cours.enise.fr/info/latex>

Enfin voilà deux jeunes corps enlacés qui jouissent de leur jeunesse en fleur ; déjà ils pressentent les joies de la volupté et Vénus va ensemer le champ de la jeune femme. Les amants se pressent avidement, mêlent leur salive et confondent leur souffle en entrechoquant leurs dents. Vains efforts, puisque aucun des deux ne peut rien détacher du corps de l'autre, non plus qu'y pénétrer et s'y fondre tout entier. Car tel est quelquefois le but de leur lutte, on le voit à la passion qu'ils mettent à serrer étroitement les liens de Vénus, quand tout l'être se pâme de volupté. Enfin quand le désir concentré dans les veines a fait irruption, un court moment d'apaisement succède à l'ardeur violente ; puis c'est un nouvel accès de rage, une nouvelle frénésie. Car savent-ils ce qu'ils désirent, ces insensés ? Ils ne peuvent trouver le remède capable de vaincre leur mal, ils souffrent d'une blessure secrète et inconnaissable.

Lucrèce *De natural Rerum*, Livre IV

Introduction

*Mieux vaut la malice d'un homme
que la bonté d'une femme.*¹

L'ecclésiastique Si 42 14.

Il était une fois...

Tout a commencé lorsqu'au tout début des années 1990, j'utilisais sur un ordinateur PC 286 une version du logiciel Word Perfect pour m'initier à ce qu'on appelait alors le « traitement de texte ». Ce logiciel — qui existe toujours, édité par la société Corel — proposait à l'époque sous le désormais célèbre MS-DOS, une interface composée d'un vague aperçu du document, et surtout laissait à l'utilisateur la possibilité de « voir les codes » c'est-à-dire de visualiser le document avec une sorte de langage à balises en permettant un contrôle très souple.

Un peu plus tard, avec la prolifération de Windows 3.1 et l'engouement soudain pour les interfaces graphiques, je me laissais convaincre — faible que j'étais — d'utiliser le logiciel de traitement de texte devenu très célèbre aujourd'hui dans sa version d'alors : la version 2.0 (avec une petite lettre derrière qui avait toute son importance à l'époque)... Cette version, je ne l'appris qu'un peu plus tard, avait la particularité intéressante de comporter un bug très sérieux qui empêchait à partir d'un certain volume de données, la sauvegarde ! Il n'y avait alors aucune solution pour sauvegarder ni récupérer son document ; les plus teigneux d'entre nous se hasardaient à supprimer quelques lignes et tentaient à nouveau une sauvegarde, mais en vain...

À cette époque où les logiciels édités par la société dont nous taïrons le nom ici, faisaient l'objet de railleries non dissimulées,² la plupart des utilisateurs qui m'entouraient acceptaient malgré tout la situation : il était normal d'utiliser des logiciels qui se vautraient lamentablement et notoirement sans crier gare. Cette particularité a fait naître en moi une certitude : *je n'accepterai pas d'utiliser de tels logiciels*. J'étais alors élève ingénieur et je pressentais qu'une partie de mon travail serait consacré à l'élaboration de documents et à l'utilisation de systèmes informatiques en général, il me fallait des outils robuste pour y parvenir.

C'est au cours de mon DEA (appelé aujourd'hui master recherche) à l'université Jean MONNET et à l'École des Mines de Saint-Étienne que j'ai découvert à la fois Unix (dans sa version Solaris) puis Linux. C'est alors que le mot « latèque » fut lâché pas loin de moi au début de ma thèse (1993-94). Il était apparemment question d'un logiciel indispensable pour produire des formules mathématiques, et surtout il semblait évident que L^AT_EX était *le* choix incontournable pour produire des documents scientifiques. À vrai dire la question n'avait même pas l'air de se poser !

¹Les épigraphes de ce document sont tirées de l'Ancien et du Nouveau testament. Ces citations sont insérées par pure provocation de ma part, et ont — parfois — un lien avec le titre du chapitre.

²Parmi celles-ci, même si elles n'apparurent que quelques années après, on pourra noter la célèbre intervention du patron de General Motors en réponse à une provocation de Bill GATES et le non moins célèbre « Piège dans le cyberspace » de Roberto DI COSMO.

J'entrepris donc d'installer cette « chose » qu'était $\text{\LaTeX}$ à la fois sur un système Mac avec une distribution nommée $\text{\OzTeX}$ et sur un système Solaris, avec la distribution fournie par l'association Gutenberg. Il avait fallu pour cela souder l'administrateur système pour qu'il accepte de créer un utilisateur privilégié `texadm` dont le but était d'administrer la distribution...

Début 1994, je commençais ma thèse avec bien évidemment la ferme intention de la rédiger avec $\text{\LaTeX}$. Courant 1995, enthousiasmé par ce que je découvrais, j'entrepris de rédiger un guide d'initiation à $\text{\LaTeX}$ pour mes collègues de laboratoire, guide qui est à l'origine du présent manuel. C'est au cours de l'année 1997, après environ deux ans de pratique et d'initiation au monde de la typographie, que je me confortais dans l'idée que $\text{\LaTeX}$ était effectivement le logiciel de choix pour la rédaction d'un document « sérieux » : contrôle global de la mise en page, gestion de la bibliographie, des indexes (nom communs et auteurs), légèreté des fichiers manipulés et surtout : *la beauté du résultat*. Aujourd'hui encore, c'est pour moi l'argument le plus fort et le plus irréfutable pour utiliser $\text{\LaTeX}$.

Aujourd'hui maître de conférence en informatique à l'école nationale d'ingénieurs de Saint-Étienne, j'utilise $\text{\LaTeX}$ pour la rédaction de documents scientifiques et de supports pédagogiques. Après maintenant plusieurs années de pratique, je continue à apprendre et à découvrir, tout en étant encore ébloui par l'ensemble des extensions proposées par les contributeurs du projet, ensemble d'extensions qui font de $\text{\LaTeX}$ un joyeux bazar, mais aussi un outil extraordinaire évoluant dans le sens de la véritable ergonomie,³ un outil unique dont le souci permanent est « la belle ouvrage ».

Organisation du manuel

Ce manuel est une introduction au « traitement de texte » $\text{\LaTeX}$; il ne s'agit pas d'un manuel de référence, mais il a pour but de donner les bases pour utiliser $\text{\LaTeX}$ et si possible d'y prendre goût. Ainsi trouvera-t-on les informations nécessaires pour *commencer* en $\text{\LaTeX}$ et quelques conseils sur la rédaction des documents. Pour votre confort, nous avons eu l'idée lumineuse de diviser ce manuel en chapitres et annexes. La première partie présente les bases de $\text{\LaTeX}$:

Principes de base expose les concepts fondamentaux de $\text{\LaTeX}$ à lire impérativement pour comprendre le reste.

Ce qu'il faut savoir présente les outils standard, ceux qu'il faut connaître pour rédiger un document simple.

Mathématiques ou comment produire des équations.

Un pas vers la sorcellerie pénètre un peu plus profondément dans les rouages de $\text{\LaTeX}$; à lire si vous voulez utiliser $\text{\LaTeX}$ de manière satisfaisante.

Graphisme permet de comprendre comment insérer des graphiques dans vos documents.

Documents scientifiques donne quelques conseils pour rédiger articles, bibliographies, index et transparents.

Documents en français fournit quelques notions de typographie élémentaires et présente les principaux aspects du package `french`.

³Pas celle qui consiste à ajouter une entrée dans un menu, ou un son à l'apparition d'une boîte de dialogue.

À vous de jouer ! une conclusion sous forme de conseils pour chercher des informations sur $\text{T}_{\text{E}}\text{X}$ et $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$.



La deuxième partie a pour but d'aborder les aspects plus complexes de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ en prenant comme prétexte d'expliquer comment ce manuel a été produit. Ne la lisez pas avant d'avoir lu la première... Toute exposition — même non prolongée — à la deuxième partie peut provoquer des troubles du comportement et des traumatismes irréversibles.

Viennent ensuite les annexes :

Générer des documents en PDF comme son nom l'indique explique la méthode utilisée pour générer la version pdf de ce manuel ;

Mémento est un fourre-tout qui propose une liste non exhaustive d'extensions utiles, les raccourcis de $\text{AucT}_{\text{E}}\text{X}$, et la configuration de aspell pour emacs ;

Symboles une liste des symboles mathématiques disponibles en standard et avec l'extension amssymb.

Il est conseillé de lire dans l'ordre les premiers chapitres jusqu'aux mathématiques. Les suivants peuvent se lire indépendamment les uns des autres. Encore une fois, la deuxième partie du manuel n'est à lire qu'après avoir maîtrisé les concepts de base. Un index en fin de document constitue un bon point d'entrée pour retrouver des informations. Enfin, à l'instar des auteurs de la FAQ française de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$, je n'ai pas fait d'effort particulier pour traduire systématiquement tous les termes du jargon de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ et de l'informatique en général.

Ce qu'il faudrait que vous sachiez

La lecture de ce manuel qui s'adresse aux débutants, ne demande aucun pré-requis à propos de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. Le lecteur devra cependant posséder une connaissance de base d'un système d'exploitation en tant qu'utilisateur, c'est-à-dire savoir manipuler des fichiers. Être capable de créer un fichier PostScript encapsulé sur son système, à partir d'un logiciel de dessin ou de manipulation d'image, est également souhaitable.

Ce que vous ne saurez pas

Ce fabuleux manuel que vous avez entre les mains souffre tout de même de quelques lacunes ; parmi celles-ci :

- il manque une explication claire de la manière dont $\text{T}_{\text{E}}\text{X}$ et $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ gèrent les fontes. Vous ne trouverez d'ailleurs nulle part le mot METAFONT ;
- vous n'apprendrez pas comment installer et administrer une distribution $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ sur un système UNIX ;
- vous ne trouverez pas de « catalogue » ou d'inventaire des extensions existantes, utiles ou inutiles, compatibles ou incompatibles, etc. ;
- la question de l'œuf ou la poule est également occultée, ainsi que celle des liens entre Dieu et la science ;
- ...



Il est important de noter que le titre de ce manuel est un mensonge éhonté. D'autre part, les images de la couverture mises à disposition par $\text{K}_{\text{N}}\text{U}\text{T}\text{H}$ ⁴ sont dessinées

⁴<http://www-cs-faculty.stanford.edu/~uno/graphics.html>

par Duane BIBBY et n'ont rien à voir avec le sujet du présent manuel puisqu'il s'agit de dessins censés représenter les programmes T_EX et METAFONT.

T_EX ?

Le mathématicien Donald Ervin KNUTH — à qui l'on doit de nombreux ouvrages de mathématiques et d'algorithmique (notamment *The Art of Computer Programming* [1]) — a conçu dans les années 70 un système de traitement de texte nommé T_EX après avoir été déçu par la manière dont ses articles étaient imprimés par les systèmes du moment. T_EX — accessible au public depuis le début des années 80 — est un environnement complexe de programmation composé d'un processeur de macro (*macro processor*) et de quelques centaines de primitives. Un premier ensemble de macros pré-compilées est apparu assez rapidement sous le nom de *format plain*.

On pourra noter que T_EX n'est ni un *traitement de texte* (KNUTH le nomme « typesetting system » que l'on pourrait traduire par système de composition) ni un langage de programmation compilé. Voici quelques citations de KNUTH à propos de T_EX⁵

« Des mots anglais comme « technology » sont dérivés de racines grecques commençant par les lettres ... ; ce même mot grec voulant dire à la fois art et technologie. D'où le nom T_EX, qui est la forme en majuscules de »

Au sujet de la prononciation du « X » de T_EX :

« [...] C'est le son « ch » en allemand comme dans ach ; c'est le « j » espagnol [...]. Lorsque vous le dites correctement à votre ordinateur, l'écran doit devenir légèrement humide. »

Votre humble serviteur se contente lui de le prononcer « TeK » pour contrecarrer l'aspect caoutchouteux et éviter d'avoir à nettoyer son écran régulièrement.

Enfin pour ce qui est du logo lui-même KNUTH fait remarquer que ce déplacement du E est là pour rappeler qu'il s'agit de typographie, et insiste sur le fait que dans une situation où l'on veut parler de T_EX sans avoir les moyens d'abaisser le E, il faudra écrire « TeX ».

La version actuelle de T_EX est 3.14159 (les versions comme vous l'avez compris tendent vers π) ; KNUTH estime que le dernier bug a été trouvé et corrigé dans T_EX le 27 novembre 1985 (préface du livre T_EX : le programme) et propose une récompense de \$20.48 à qui en trouve un nouveau (!).

L^AT_EX ?

En 1985, quelques années après la diffusion publique de T_EX, Leslie LAMPORT crée un format composé de macros permettant d'avoir une vision de plus haut niveau d'un document, appelé L^AT_EX et portant le numéro de version 2.09. Aujourd'hui,

⁵Tiré du chapitre introductif « The Name of the Game » du T_EXBook.

L^AT_EX est un standard de fait, et seuls quelques sorciers produisent encore des documents uniquement avec T_EX. Cependant, L^AT_EX étant une « surcouche » de T_EX — contenant donc des appels à des macros de T_EX — il est parfois utile de connaître quelques-uns des concepts de T_EX pour se tirer d'un mauvais pas. Voici ce que dit LAMPORT à ce propos dans son livre [2] :

« Imaginez L^AT_EX comme une maison dont la charpente et les clous seraient fournis par T_EX. Vous n'en avez pas besoin pour vivre dans la maison, mais ils sont pratiques pour y ajouter une nouvelle pièce. »

Un peu plus loin :

« L^AT_EX a été conçu pour permettre à un auteur de faire abstraction des soucis de mise en page, et se concentrer sur l'écriture. Si vous passez beaucoup de temps sur la forme, vous faites un mauvais usage de L^AT_EX. »

Aujourd'hui et depuis 1994, une équipe mi-européenne mi-américaine (autour de Frank MITTELBACH) a pris en main le développement de L^AT_EX ; la version de L^AT_EX parue en 1994 se nomme L^AT_EX 2_ε. Le but à long terme est de concevoir un système nommé L^AT_EX3.

Licence

On peut souligner que T_EX et L^AT_EX sont des logiciels faisant partie de la famille des logiciels libres et sont donc — entre autres — gratuits. Ce qui caractérise les logiciels libres (*free software*) est également l'aspect *ouvert* des logiciels. Il est donc possible d'avoir les sources Web⁶ de T_EX. Les macros de L^AT_EX sont quant à elles distribuées sous forme de code source T_EX. Le fait de pouvoir obtenir les sources d'un logiciel peut sembler secondaire à la plupart des utilisateurs ; il faut comprendre que c'est parce que *rien n'est caché* que l'amélioration de l'existant et la création d'extensions sont possibles.

Le fait qu'un logiciel soit libre ne veut pas dire qu'on puisse en faire tout à fait ce que l'on veut. Il reste la propriété de son auteur et toute modification doit être documentée ; chacune de ces modifications doit également donner lieu à un nom de fichier différent de celui du fichier initial avant modification. Ceci pour assurer cohérence et portabilité au système (voir à ce sujet <ftp://ftp.lip6.fr/pub/TeX/CTAN/macros/latex/base/lppl.txt> pour la licence de L^AT_EX 2_ε).

Cinq bonnes raisons pour ne pas utiliser L^AT_EX

Il existe plusieurs raisons pour lesquelles il est « impératif » de ne pas utiliser L^AT_EX :

⁶Le langage Web conçu par KNUTH est qualifié de langage de « programmation littéraire. » À partir d'un document source Web, on peut produire le code Pascal ou C du programme ainsi qu'une documentation en T_EX de ce code.

1. vous utilisez un traitement de texte uniquement pour faire vos cartes de vœux, votre courrier, pour noter quelques idées, etc. ;
 2. vous adorez les souris (1 ou 3 boutons indifféremment) et vous pensez que la seule manière d'écrire des équations est de les utiliser (les souris) de manière intensive ;
 3. vous pensez qu'UNIX c'est « prise de tête » et « pas convivial » et/ou vous avez une aversion particulière pour tout langage de programmation ;
 4. vous trouvez normal : 1° que votre logiciel préféré ne puisse pas lire le document que vous aviez produit avec la version précédente, et/ou 2° que la nouvelle version vous oblige à changer de système d'exploitation, et 3° que la nouvelle version dudit système d'exploitation vous oblige à changer d'ordinateur, et 4° que votre nouvel ordinateur...
 5. vous ne savez pas où se trouve la touche `\` sur votre clavier ;
- si vous vous reconnaissez dans une de ces catégories, mieux vaut vous contenter de votre système actuel.

Quelques bonnes raisons d'utiliser $\text{\LaTeX}$

Il n'est pas question ici de convaincre le lecteur de la supériorité de $\text{\TeX}$ et $\text{\LaTeX}$ par rapport à un autre système, de toutes manières, vous lisez ce manuel, donc vous êtes inconsciemment convaincu. Laissons donc la parole au concepteur de $\text{\TeX}$:

« By preparing a manuscript in $\text{\TeX}$ format, you will be telling a computer exactly how the manuscript is to be transformed into pages whose typographic quality is comparable to that of the world's finest printers. »

D. E. Knuth in the $\text{\TeX}$ book [3]

Les documents générés par $\text{\TeX}$ ou $\text{\LaTeX}$ sont d'une qualité typographique exceptionnelle (avec possibilité de réglage très fin⁷), ceci grâce notamment à :

- un dessin de fontes très soigné ;
- des détails typographiques (tirets, ligatures,...) ;
 - avez-vous — bien — regardé ces tirets (page 19–23) ?
 - et le « fi » de fin, le « ffl » de souf ou le « fl » de trèfle ?
- un algorithme de césure très performant ;
- des formules mathématiques particulièrement réussies ;
- ...

D'autre part, $\text{\LaTeX}$ est un des rares logiciels de traitement de texte orienté vers la production de documents *scientifiques*. Car outre les équations et autres formules, $\text{\LaTeX}$ possède `ogrnd (n)-1(o)1rmfouctionualitflseourla rfldauction` et `graphx` pour la gestion de

peu de mémoire, mais parce que son mécanisme de *macros* et de *référence croisée* (*cross reference* en anglais) permettent de garder un contrôle global et très souple du document.

référence croisée : $\text{\LaTeX}$ permet de faire référence de manière *symbolique* à toute partie du document faisant l'objet d'une numérotation. Ainsi, le numéro des titres, figures, tableaux, équations, pages, références bibliographiques, items d'énumération, théorèmes,... peut être mentionné à plusieurs endroits dans un document de manière très simple, sans se soucier du numéro lui-même.

macros : sans doute l'aspect le plus puissant de $\text{\LaTeX}$. Il faut savoir que tout processus qui mène à la génération d'un document est une séquence de commandes ou macros. Chaque utilisateur peut donc modifier l'allure d'un document, en modifiant l'une des ces macros. On peut bien évidemment définir ses propres macros pour mettre en page une partie spécifique d'un document. L'idée forte autour des macros est qu'on peut *a priori* séparer le fond de la forme lors de la rédaction d'un document.

Les limites du Wysiwyg

$\text{\LaTeX}$ est le contraire d'un Wysiwyg,⁸ puisqu'un source $\text{\LaTeX}$ est un document texte composé du texte lui-même et des commandes de mise en page. LAMPORT présente ce type d'approche comme étant une mise en page *logique* par opposition à la mise en page *visuelle*.⁹

On pourrait cependant dire que $\text{\LaTeX}$ est un Wywsiewyg (what you will see is exactly what you get) puisqu'après compilation on peut visualiser à l'écran une image exacte du document futur sur papier.

Voici donc un exemple¹⁰ parmi d'autres qui met en évidence les limites du Wysiwyg et les avantages de la mise en page logique : supposons que dans un document apparaisse un certain nombre de fois, une fonction quelconque ayant deux arguments. La notation étant un point délicat dans l'élaboration de documents scientifiques, on pourra définir une macro `\mafct` permettant de produire une telle fonction. Ainsi les séquences `\mafct{1}{2.5}` et `\mafct{x}{t}` produiront respectivement $\mathcal{F} ; (1, 2.5)$ et $\mathcal{F} ; (x, t)$. Mais si l'on a besoin de changer de notation, il suffira de redéfinir la commande `\mafct` pour produire aux endroits nécessaires : $F [1, 2.5]$ et $F [x, t]$. Et le tour est joué !

Un autre exemple : imaginons que votre document comporte beaucoup de mots techniques que vous voulez mettre en évidence d'une manière particulière. Vous écrirez alors dans votre document `\jargon{implémentation}` en ayant préalablement défini la macro `\jargon` de manière à ce qu'elle mette en italique le mot du vocabulaire technique. Les 235 mots de jargon auxquels vous faites référence dans votre document pourront être mis en évidence autrement qu'en italique si vous changez d'avis, et cela sans avoir à passer sur les 235 occurrences des mots du jargon, mais juste en changeant la définition de la macro `\jargon`. Avec un peu d'entraînement

⁸Pour « what you see is what you get » terme apparu au début des années 90 pour désigner les logiciels permettant à l'utilisateur de voir à l'écran sur qu'il obtiendrait sur le papier.

⁹Pour faire un peu de mauvais esprit, les logiciels du type Wysiwyg ont d'ailleurs été qualifiés par Kernighan (dixit LAMPORT dans son livre sur $\text{\LaTeX}$) de « what you see is all what you've got » !

¹⁰LAMPORT propose un exemple analogue à celui-ci dans son manuel.

vous arriverez même à faire en sorte que cette macro insère automatiquement le mot du jargon dans l'index de votre document...

Voici un exemple un peu plus tordu : dans le titre du paragraphe intitulé « **Cinq** bonnes raisons de ne pas [...] » un peu plus haut dans ce chapitre, je n'ai pas écrit « **Cinq** »¹¹ en toutes lettres dans le document source. En réalité le titre du paragraphe est produit par : « `\ref{nbraisons}` bonnes raisons... » qui affiche en français le nombre correspondant aux nombres de bonnes raisons de ne pas utiliser L^AT_EX. Si jamais j'avais à rajouter d'autres entrées dans cette liste de bonnes raisons, il ne sera pas nécessaire de refaire la numérotation...



Vous trouverez le long de ce manuel, d'autres exemples mettant en évidence, les faiblesses du Wysiwyg. Ce « nota », vous avertissant d'un point important en est un autre exemple. Car au moment où l'auteur tape ces lignes, la présence du « panneau danger » est un détail — il s'agit simplement d'un nota. Et l'auteur a écrit :

```
\begin{nota}
  Vous trouvez le long de ce manuel...
\end{nota}
```

Pour en finir avec les macros, on peut dire qu'il s'agit d'une « généralisation » des styles du célèbre logiciel « Mot » de la société « Micrologiciel ». La lecture de ce document et en particulier la deuxième partie devrait vous convaincre que les macros permettent d'aller bien au-delà de ces fameux styles...

Pour les accros du Wysiwyg, une équipe de développeur a mis en œuvre une version *What you see is what you Mean* (sic) de L^AT_EX nommé LyX, dont je vous invite à prendre connaissance à <http://www.lyx.org>.

Comment imprimer ce manuel ?

Avec une imprimante,¹² en utilisant exclusivement les fichiers proposés sur <http://www.enise.fr/cours/info/latex>. Ce manuel est conçu pour être imprimé en deux pages logiques par page physique (*2up printing* comme disent nos amis anglo-saxons). En d'autres termes, chaque page de ce document doit apparaître sur une page au format A5. Quatre solutions — et quatre fichiers associés — sont possibles :

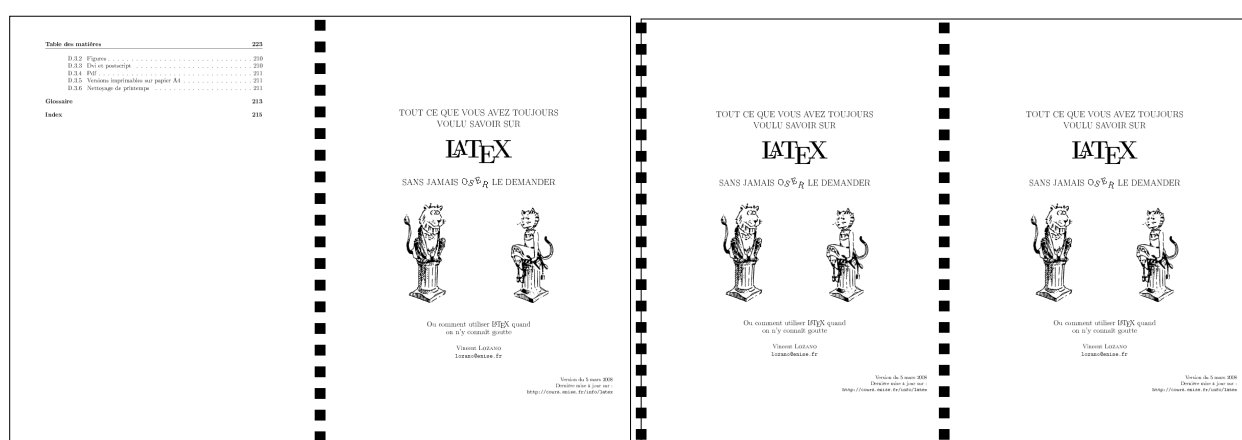
1. une version destinée à être reliée du long coté de la feuille A4, c'est la solution pour ceux qui ne disposent pas de massicot ; le fichier portant le nom `guide-latex-0.pdf` est en outre généré pour présenter la page impaire (dite belle page) à droite (cf. figure 1a page xi) ;
2. une version pour ceux qui disposent d'un massicot : dans ce cas, il faudra « plier » le document et le relier au niveau de cette pliure (voir figure 1b page xi). Le fichier correspondant se nomme `guide-latex-1.pdf` ;
3. une version contenant 2 exemplaires (fichier `guide-latex-2.pdf`). On reliera ces deux exemplaires comme indiqué à la figure 1c page xi ;
4. une version au format PostScript `guide-latex.ps.gz` pour ceux qui voudraient faire « à leur sauce »...

¹¹Là non plus d'ailleurs.

¹²Arf arf (comme disait Frank ZAPPA).



(a) Reliure longue



(b) Reliure courte « 1 exemplaire »

(c) Reliure courte « 2 exemplaires »

FIG. 1 – Les versions imprimables



Attention, les deux derniers fichiers au format Pdf, sont prévus pour être imprimés en recto verso en demandant à l'imprimante de retourner la feuille du côté court. Il faudra donc veiller à ce que le pilote d'imprimante utilise cette option.

Que pouvez-vous faire de ce manuel ?

Nom de l'auteur : Vincent LOZANO ;

Titre : Tout ce que vous avez toujours voulu savoir sur L^AT_EX sans jamais avoir osé le demander ;

Date : 5 mars 2008

Copyright : ce manuel est libre selon les termes de la Licence Art Libre (<http://www.artlibre.org>) (LAL).

La LAL stipule en résumé que vous pouvez copier ce manuel. Vous pouvez également le diffuser à condition :

- d'indiquer qu'il est sous la LAL ;

- d’indiquer le nom de l’auteur de l’original : Vincent LOZANO et de ceux qui auraient apporté des modifications ;
- d’indiquer que les sources peuvent être téléchargés sur <http://cours.enise.fr/latex> ;

Enfin vous pouvez le modifier à condition :

- de respecter les conditions de diffusion énoncées ci-dessus ;
- d’indiquer qu’il s’agit d’une version modifiée et si possible la nature de la modification ;
- de diffuser vos modifications sous la même licence ou sous une licence compatible.

En avant !

Comme beaucoup de logiciels puissants, L^AT_EX n’est pas toujours simple à utiliser. En fait lorsque l’on va dans son sens, L^AT_EX est toujours agréable et permet effectivement comme le souligne LAMPORT de ne pas se soucier de problèmes de mise en page. Lorsque l’on veut changer un comportement, et que la solution consiste à choisir une autre option d’une commande, tout va encore très bien. Cependant, même si les choix de L^AT_EX répondent à des conventions en vigueur chez tous les bons imprimeurs, il arrive un jour où l’on désire avoir une mise en page particulière qu’apparemment L^AT_EX est incapable de fournir. À ce stade, plusieurs solutions s’offrent à vous :

- inclure un *package* qui répond à votre problème (L^AT_EX étant un système ouvert, une multitude de packages plus ou moins standardisés sont disponibles pour réaliser des opérations variées voire farfelues) ;
- demander à un T_EXnicien¹³ de vous dépanner ;
- si les deux premières solutions sont inefficaces, vous n’avez plus qu’à faire le détective et mettre le nez dans le code¹⁴ pour trouver la commande qui vous fait du tort et la modifier. Vous aurez besoin à ce stade de connaissance de la première couche du système, à savoir T_EX. On touche sans doute ici à un des défauts de L^AT_EX : si d’autres logiciels sont incapables de faire des choses compliquées, il est parfois difficile de faire faire à L^AT_EX des choses simples (vous en serez probablement convaincu après la lecture de la deuxième partie de ce manuel).

Conventions typographiques

Certaines conventions utilisées dans ce manuel nécessitent d’être quelque peu éclaircies. Les d’extraits de code L^AT_EX qui parsèment le document peuvent apparaître comme ceci :

```
% attention les yeux
Ceci est \emph{déjà} du code \LaTeX.
```

¹³ou un T_EXpert, mais c’est assez rare.

¹⁴C’est la solution la plus plaisante pour ceux qui ont certaines velléités pour ~~visser~~ du code...

Le choix s'est porté sur la fonte «machine à écrire» de L^AT_EX. Le code est également souvent présenté sous la forme suivante, avec un petit numéro sur la barre centrale auquel on se réfère parfois :

```
% attention les yeux
Ceci est \emph{déjà} du
code \LaTeX.
```



Ceci est *déjà* du code L^AT_EX.



Certaines parties sont présentées sous forme de « notes » pour éclaircir un point sans que la lecture soit indispensable au premier abord.



Si la lecture est indispensable, on aura recours au pictogramme ci-contre pour attirer l'attention du lecteur distrait...

Les logiciels et les packages de L^AT_EX sont typographiés comme indiqués ci-avant. Les mots en anglais sont produits *like this*. Pour mettre en évidence les parties génériques d'une commande on utilisera [cette notation](#). Par exemple :

Ceci est `\emph{texte à mettre en phase}` du code `\LaTeX`.

Quelques rares fois sont insérées des commandes Unix, comme ceci :

```
| grep -wi bidule /tmp/truc.dat | sort -n
```

On trouve même dans une des annexes, des commandes pour emacs :

```
| M-x doctor
```

Et, comble de l'horreur, des extraits de Makefile :

```
bidule : bidule.o truc.o
→ gcc -o $@ $^
```

Dans la version papier apparaissent des renvois sur des chapitres ou des paragraphes, comme celui-ci dirigeant le lecteur vers la production de formules I mathématiques avec L^AT_EX.

Ch. 3 p. 33 J

Remerciements

La rédaction de cet ouvrage qui est initialement le guide local du laboratoire d'informatique graphique et d'ingénierie de la vision situé à Saint-Etienne, a débuté en 1995. Je tiens ici à remercier les membres de cette équipe de recherche qui m'ont fait part de leurs remarques et encouragements. Les personnes participant au forum `fr.comp.text.tex` m'ont indirectement apporté énormément d'informations qui ont enrichi ce document, qu'elles en soient ici remerciées.

Je voudrais également remercier Benjamin BAYART qui m'a aidé à créer certaines des extensions utilisées dans ce manuel, en particulier la version initiale de la boîte entourant les « mini » tables des matières en tête de chapitre ; ainsi que Guillaume CONNAN pour ses remarques sur l'annexe concernant le format Pdf et pour ses encouragements.

Un merci particulier à Denis BITOUZÉ pour sa lecture attentive ses conseils précieux et les nombreuses corrections qu'il a apportées à la première partie du manuel. Denis m'a mis sur la voie de la rédemption, j'ai fait une croix sur a4wide, eqnarray, et toutes ces horreurs qui faisaient de moi un pauvre pécheur...

La lecture des «grands classiques» de la littérature autour de T_EX et L^AT_EX m'a inconsciemment influencé dans la rédaction de ce document. La lecture de T_EXbook de KNUTH [3] m'a bien évidemment donné l'idée de créer le panneau danger , la lecture quasi compulsive du *L^AT_EX Companion* de GOOSSENS, MITTELBACH et SAMARIN [4] a très certainement influencé beaucoup de passages de cet ouvrage aussi bien pour le fond que la forme. Enfin la lecture de plusieurs manuels en ligne a également dû orienter certains de mes choix (Le «Not So Short Introduction to L^AT_EX» possède par exemple un chapitre que l'on peut traduire par «ce qu'il faut savoir»)...

Avant de commencer je tiens à signaler que même si ce document a mis plusieurs années à mûrir, il est dans un style tout à fait douteux. La preuve, sur ma machine, la commande :

```
grep -E -i 'on peut|permet' *.tex | wc -l
```

donne 343 (plus d'une occurrence par page) ce qui dénote un style assez pauvre.

Bonne lecture et bon courage!¹⁵

¹⁵Cette ligne est une illustration de la mise en page logique, elle est censée être au 2/3 du blanc restant sur la page, quelle que soit la taille de ce blanc bien sûr.

Sommaire

I	« Tout »sur L ^A T _E X	1
1	Principes de base	3
1.1	Installation	3
1.2	Cycle de production	4
1.3	Le document source : un document type	7
1.4	C'est parti !	9
1.5	Premiers outils	12
1.6	Premières erreurs	12
2	Ce qu'il faut savoir	15
2.1	Mise en évidence	15
2.2	Environnements	18
2.3	Notes de marge	22
2.4	Titres	23
2.5	Notes de bas de page	24
2.6	Entête et pied de page	24
2.7	Flottants	25
2.8	Références	26
2.9	Fichiers auxiliaires	27
2.10	Où il est question de césure	29
3	Mathématiques	33
3.1	Les deux façons d'écrire des maths	33
3.2	Commandes usuelles	34
3.3	Fonctions	36
3.4	Des symboles les uns sur les autres	38
3.5	Deux principes importants	39
3.6	Array : simple et efficace	40
3.7	Équations et environnements	42
3.8	Changer le style en mode mathématique	43
4	Un pas vers la sorcellerie	47
4.1	Compteurs	47
4.2	Longueurs	50
4.3	Espaces	54
4.4	Boîtes	56
4.5	Définitions	62
4.6	Mais encore ?	65

5	Graphisme	67
5.1	Apéritifs	67
5.2	Du format des fichiers graphiques	68
5.3	Le package graphicx	68
5.4	Quelques extensions utiles	71
5.5	Utiliser make	74
5.6	À part ça	76
6	Documents scientifiques	77
6.1	Article	77
6.2	Bibliographie	78
6.3	Index	82
6.4	Diviser votre document	84
7	Des documents en français	87
7.1	Le problème des lettres accentuées	87
7.2	Rédiger un document en français avec L ^A T _E X	89
7.3	Le package babel et la typographie	89
7.4	Courrier et fax	93
8	À vous de jouer !	97
8.1	Livres et autres manuels	97
8.2	Local	98
8.3	E ² Tépi, Ouèbe et niouses	98
II	« Tout »sur (« Tout »sur L ^A T _E X)	101
9	Outillage nécessaire	105
9.1	Hercule Poirot	105
9.2	Outils de bas niveaux	107
9.3	Structures de contrôle et tests	110
9.4	Fontes	114
9.5	Listes et nouveaux environnements	119
9.6	Des environnements qui mettent en boîte	125
10	Cosmétique	127
10.1	Allure de l'index	127
10.2	Allures des titres	129
10.3	Géométrie	134
10.4	En-tête et pied de page	136
10.5	Environnements avec caractères spéciaux	141
10.6	About those so called "french guillemets"	145
10.7	Une boîte pour la minitable des matières	146
11	De nouveaux jouets	153
11.1	Quelques bricoles	153
11.2	Des nota	158
11.3	Des citations	162
11.4	Des lettrines	166

11.5	Un sommaire	169
11.6	Un glossaire	171
11.7	Des onglets	174
11.8	Exemples $\text{\LaTeX}$	179
III	Annexes	185
A	Générer des « pdf »	187
A.1	Principe général	187
A.2	Ce qui change	187
A.3	Trucs et astuces	188
A.4	Hyperliens	189
A.5	Interaction avec psfrag et pstricks	190
B	Mémento	195
B.1	Extensions	195
B.2	Les fichiers auxiliaires	196
B.3	AucTeX	197
B.4	Aspell	198
C	Symboles	201
C.1	Symboles standard	202
C.2	Symboles de l' $\mathcal{AMS}$	205
C.3	Symboles du package textcomp	207
D	Notes de production	211
D.1	Distribution du moment	211
D.2	Les sources du manuel	211
D.3	Compilation	212
	Bibliographie	215
	Glossaire	217
	Index	219

I

Tout ce que vous avez toujours
voulu savoir sur L^AT_EX sans jamais
oser le demander

1

Principes de base

Sommaire

- 1.1 Installation
- 1.2 Cycle de production
- 1.3 Le document source : un document type
- 1.4 C'est parti !
- 1.5 Premiers outils
- 1.6 Premières erreurs

1

*Lorsqu'un homme a un écoulement
sortant de son corps,
cet écoulement est impur.*
Le Lévitique Lv 15 2.

CE CHAPITRE expose les mécanismes de base de $\text{\LaTeX}$. Vous y trouverez donc une courte introduction à l'installation de $\text{\LaTeX}$, une présentation d'une « session » $\text{\LaTeX}$ classique, la structure d'un document type, des remarques sur les accents, quelques outils à connaître, et enfin, une présentation de l'attitude à avoir devant les messages d'erreurs de compilation.

1.1 Installation

Vous voulez utiliser $\text{\LaTeX}$? Il vous faudra installer une *distribution* correspondant à votre système d'exploitation.¹ Les distributions fournissent des programmes permettant d'automatiser la configuration et l'installation de $\text{\LaTeX}$, $\text{\TeX}$ et tous les utilitaires connexes.

Sous Unix : on trouve encore la distribution teTeX bien que son développement ait été stoppé en 2006. Aujourd'hui on installe généralement la $\text{\TeX Live}$ (<http://www.tug.org/texlive>) sur un système Unix ;

Sous MacOS : la distribution de référence est MacTeX (<http://www.tug.org/mactex>) ;

Sous Windows : le plus simple est sans doute de choisir proTeXt (<http://www.tug.org/protext>) qui installe la distribution MiKTeX (<http://www.miktex.org>) et quelques outils de développement dont un programme de visualisation de fichiers au format PostScript (gsview).

Il faut parfois ajouter à ces distributions (si elles n'en contiennent pas déjà un) un éditeur de texte puisque vous le découvrirez bien assez tôt, utiliser $\text{\LaTeX}$ c'est taper du texte et des commandes dans des fichiers :

¹Si vous ne savez pas ce qu'est un système d'exploitation, le vôtre est MacOS, si vous ne savez pas *quel est exactement* le système de votre machine vous avez Fenêtre, sinon vous avez un Unix...

- emacs ou vi sous Unix sont deux éditeurs de référence qui, bien que le premier soit nettement supérieur au second, continuent de faire l'objet d'une guerre stérile entre utilisateurs la plupart du temps de mauvaise foi ;
- kile et texmaker sont des environnements de développement intégré grâce auxquels les utilisateurs débutants pourront se sentir à l'aise pour commencer : ils ont en effet la particularité de centraliser dans une même interface : édition, compilation & visualisation. Ces environnements permettent également de découvrir les commandes de L^AT_EX par l'intermédiaire de menus, boîtes de dialogue et autres onglets (voir la figure 1.1a page suivante pour en avoir un aperçu) ;
- T_EXnicCenter est l'équivalent (aperçu à la figure 1.1b) sous la marque «à la fenêtre» ;
- T_EXshop et iT_EXmax sont les équivalents sous la marque «à la pomme»

Vous apprendrez également bien assez vite que la production d'un document avec L^AT_EX consiste à traduire (on dit aussi *compiler*) un source — donc créé par un éditeur de texte — en un format destiné à l'assemblage ou à l'impression. Il existe donc, plus ou moins intégré aux distributions, des outils célèbres pour la visualisation des différents fichiers résultants de la compilation :²

Format DVI : xdvi, kdvi sous Unix et yap sous Windows font partie des programmes permettant de visualiser le résultat de la compilation d'un fichier L^AT_EX ;

Format Postscript : la suite ghostscript disponible sous des noms qui peuvent varier selon la plateforme, permet de visualiser des fichiers au format PostScript ;

Format PDF : mis à part le célèbre acrobat reader, il existe sous Unix des utilitaires permettant de visualiser le format Pdf : xpdf, evince, ...



Il faudra veiller à ce que la distribution choisie comprenne le module « français » de L^AT_EX assurant (*hyphenation* en anglais) correcte des mots. On vérifiera les « logs » (voir § 1.6 page 12) au moment de la compilation d'un document pour contrôler le chargement des motifs pour le français :

```
LaTeX2e <2005/12/01>
Babel <v3.8h> and hyphenation patterns for english, usenglishmax,
dumylang, french, loaded.
```

1.2 Cycle de production

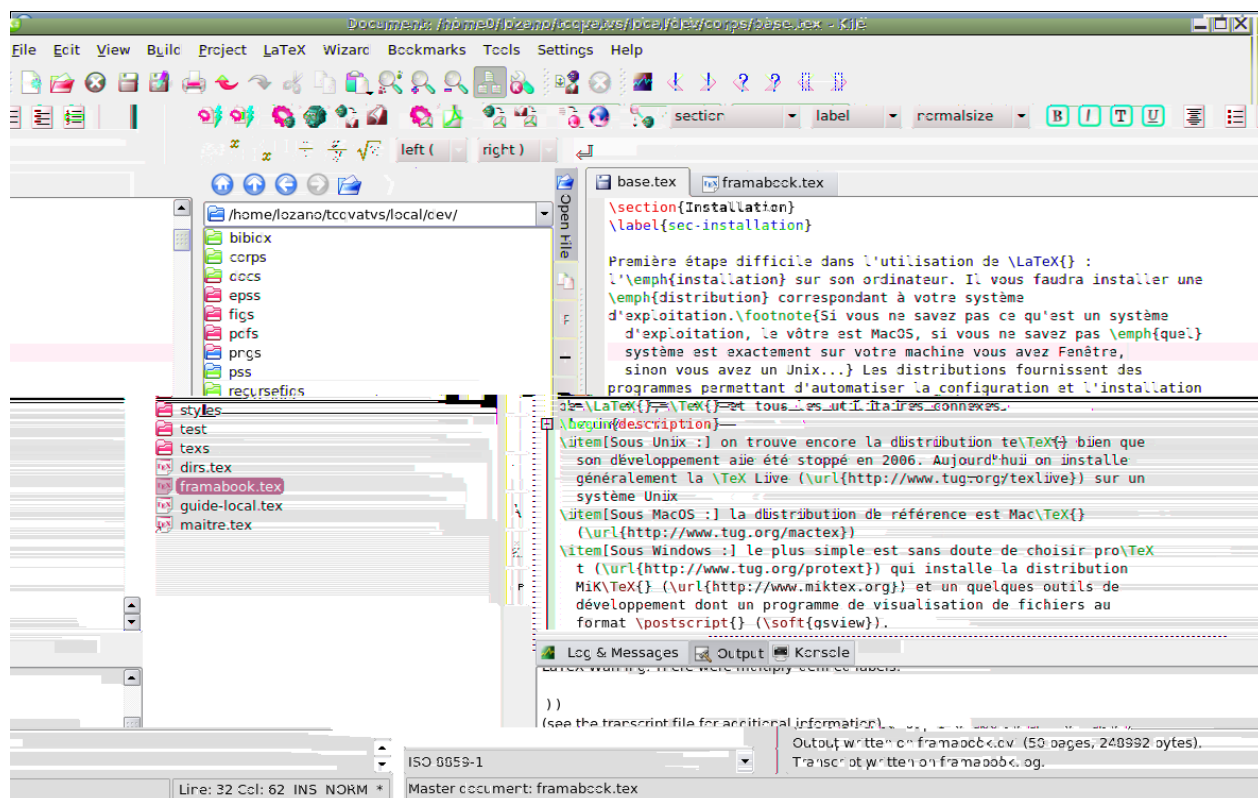
Même si L^AT_EX n'est pas à proprement parler un langage de programmation compilé, on peut malgré tout faire une analogie entre le cycle de production d'un document L^AT_EX et le cycle *édition-compilation-exécution* d'un développement de logiciel avec un langage de programmation classique.

1.2.1 Édition

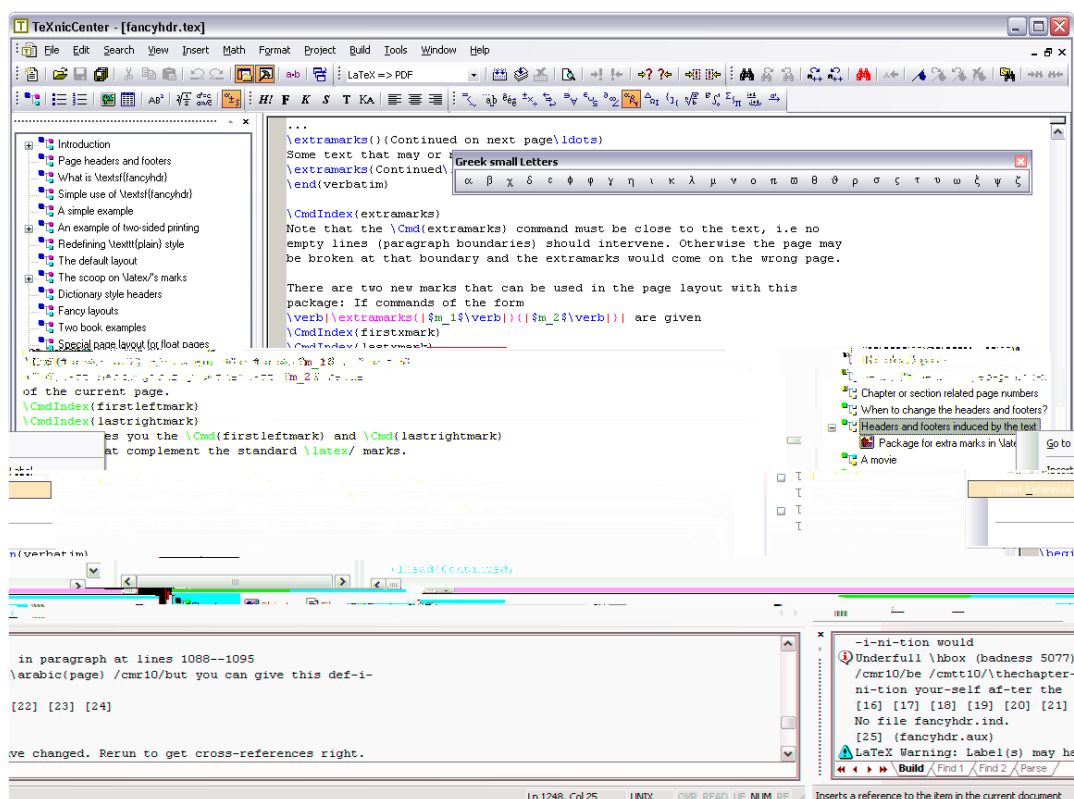
Un document *source* L^AT_EX est un fichier texte.³ Ainsi la manipulation d'un fichier L^AT_EX ne demande pas de logiciel particulier, si ce n'est un éditeur de texte classique. Donc, pour manipuler un document L^AT_EX :

²Ces formats sont présentés peu plus loin dans ce chapitre.

³C'est-à-dire un fichier ne contenant que le code des caractères qui le composent.



(a) Kile



(b) TeXnicCenter

FIG. 1.1 – Deux exemples d'environnement de développement intégré : Kile sous Linux et TeXnicCenter sous Windows. Ils permettent de centraliser dans une même interface : édition, compilation & visualisation.

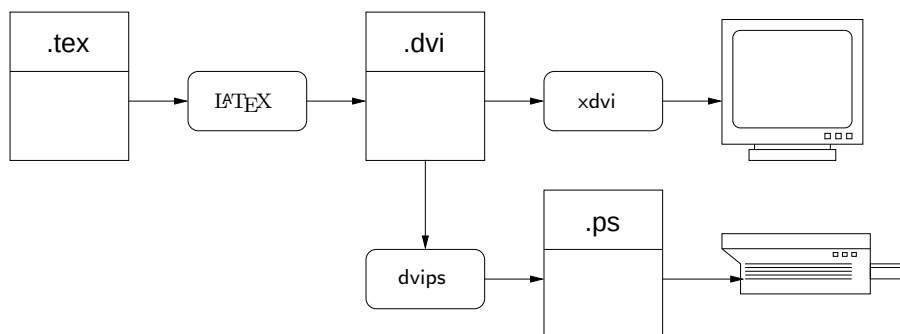


FIG. 1.2 – Cycle de production sur machines UNIX

1

```
| emacs nom de fichier.tex &
```

ou :

```
| vi nom de fichier.tex
```

devrait suffire pour entrer dans ce monde sauvage et inconnu qu'est la saisie d'un document L^AT_EX. Sous Windows, on s'équipera d'un éditeur de texte de son choix.⁴ Notez qu'il est recommandé de donner l'extension `.tex` aux sources L^AT_EX.

1.2.2 Compilation

On lance la compilation grâce à la commande :

```
| latex nom de fichier.tex
```

La compilation génère un jour ou l'autre des erreurs. Il en sera question à la section 1.6 page 12. En tout cas, après suppression des erreurs de compilation, on obtient un fichier portant l'extension `dvi` pour *device independant*. Ce qui signifie que le fichier contient des informations indépendantes du périphérique de sortie (écran, imprimantes, ...). Ce fichier de type binaire contenant une « image » du document portable sur tout système T_EX quel que soit le système d'exploitation. Il existe ensuite des programmes permettant soit :

- de visualiser le document : `dvi` → bitmap écran ;
- de l'imprimer : `dvi` → langage imprimante ;
- de le convertir : `dvi` → fichier PostScript.

La figure 1.2 illustre les divers programmes entrant en jeu dans la production du document final sur une machine UNIX.



Il est également possible de générer un document au format PDF. On pourra consulter l'annexe A pour obtenir des informations à ce sujet.

1.2.3 Visualisation

La visualisation s'effectue simplement — après compilation sans erreur — grâce au programme `xdvi` en tapant la commande :

```
| xdvi nom de fichier.dvi &
```

⁴Il existe une version d'Emacs pour Micrologiciel Fenêtre, avis aux amateurs.

il s'agit d'un logiciel tournant sous X Window, très intuitif, qui donne un aperçu très lisible du document. La distribution TeX pour Windows propose un visualiseur nommé yap.⁵



On notera qu'il n'est pas nécessaire de relancer xdvi ou yap après chaque compilation. Ils mettent en effet à jour l'affichage automatiquement.

1.2.4 Impression

Pour imprimer un document, on utilise le programme dvips comme suit :

```
| dvips nom de fichier.dvi
```

Il est aussi possible de générer un fichier PostScript en redirigeant la sortie de dvips :

```
| dvips nom de fichier.dvi -o nom de fichier.ps
```

Le fichier `nom de fichier.ps` est un fichier ASCII pur contenant des commandes PostScript. Par exemple si le fichier source est nommé `truc.tex`, on écrira :

```
| latex truc.tex
| dvips truc.dvi -o bidule.ps
```

pour compiler d'abord `truc.tex`, et pour produire ensuite `bidule.ps` à partir de `truc.dvi`. Pour préparer un fichier destiné à d'autres imprimantes, il faut spécifier en ligne de commande :

```
| dvips -Pconfig nom de fichier.dvi
```

où `config` désigne une imprimante particulière ou une option définie par l'administrateur du système TeX que vous utilisez.



Le format PostScript défini par Adobe est un langage d'impression très répandu sur les systèmes UNIX. De nombreux utilitaires autour du Postscript sont disponibles, voir par exemple les très bons `psutils` disponibles par exemple à <ftp://ftp.lip6.fr/pub/TeX/CTAN/support/psutils> ou toute autre bonne crèmerie CTAN⁶ près de chez vous ou sous la forme d'un paquet Debian.

1.3 Le document source : un document type

Nous allons présenter dans cette section un document type. Tous les documents TeX ont en effet une structure commune, de la forme suivante :

```
\documentclass[cOption1,cOption2,...]{classe}
\usepackage[pOption1,pOption2]{package}
...
préambule
...
\begin{document}
...
le texte
...
\end{document}
```

⁵Pour «yet another previewer», soit «encore un visualiseur» (humour d'informaticien)...

⁶voir le chapitre 8 page 97 pour la signification de ce terme.

Ainsi tout document $\text{\LaTeX}$ peut se décomposer comme suit :

- spécification de la **classe** du document ;
- préambule :
 - utilisation de **packages** particuliers ;
 - initialisations et déclarations diverses ;
- corps du document (entre `\begin{document}` et `\end{document}`).

Voici quelques détails sur chacune de ces parties.

1.3.1 Classe du document

La classe est une indication donnée à $\text{\LaTeX}$ qui va déterminer la mise en page de certaines parties du document. Suivant la classe utilisée, certaines commandes seront disponibles ou non (`\chapter` disponible pour la classe `book` mais indisponible pour la classe `article`, par exemple). D'autre part, une commande donnée aura une signification spécifique selon la classe choisie (titres, tables des matières,...). En première approche,⁷ tout document $\text{\LaTeX}$ commence donc nécessairement par l'instruction `\documentclass` avec entre accolades une classe de document qui peut-être :

- `article` pour un article ;
- `proc` pour un article dans le style IEEE proceedings ;
- `report` pour un rapport de plusieurs dizaines de pages ;
- `book` pour un livre ou une thèse ;
- `letter` pour une lettre ;
- `slides` pour produire des transparents

On peut évidemment définir sa propre classe de document. Les options de la classe sont précisées entre crochets et peuvent être les suivantes (non-exhaustif) :

- `11pt`, `12pt` pour changer la taille des caractères de manière globale ;
- `twoside` pour générer un document en recto verso ;
- `draft` pour générer le document en mode brouillon ;
- ...

On pourra donc par exemple, entrer :

```
\documentclass{article}
```

pour avoir toutes les options par défaut (10pt, une colonne, mode recto,...).

```
\documentclass[12pt]{article}
```

pour un article en 12pt (par défaut la taille est de 10pt), ou encore :

```
\documentclass[twoside,draft]{report}
```

pour un rapport en recto verso et mode brouillon.

1.3.2 Le préambule

Le préambule est la zone située entre la clause `\documentclass` et la clause `\begin{document}`. Cette zone est la zone où l'on peut spécifier les extensions que l'on veut inclure (voir le paragraphe suivant), l'initialisation de variables globales (marges,...), la définition de styles (titres, numérotation,...), ou de macros particulières.

⁷En réalité on peut insérer d'autres incantations magiques avant le `\documentclass...`

1.3.3 Ajout d'extension

La commande `\usepackage` de $\text{\LaTeX}$ pourrait être comparée à une directive `#include` du langage C. Elle permet de rajouter des fonctionnalités à $\text{\LaTeX}$ sous forme de macros et/ou d'environnement.⁸ À ce stade, il faut juste noter que l'on peut inclure plusieurs packages en une seule ligne :

```
\usepackage{module1, module2, module3, ...}
```

Si `module1`, `module2` et `module3` ont en commun une option `opt1`, on peut entrer :

```
\usepackage[opt1]{module1, module2, module3}
```

Par contre si l'option `opt1` ne concerne que l'extension `module2`, il sera nécessaire d'entrer les deux lignes suivantes :

```
\usepackage{module1, module3}
\usepackage[opt1]{module2}
```

Voici deux exemples :

```
% package graphicx avec option draft et xdvi
\usepackage[xdvi,draft]{graphicx}
% packages array et subfig
\usepackage{array,subfig}
```



Toutes les options (de classe, de packages, ou de commandes) sont par définition des arguments *optionnels*. On peut donc déjà retenir le fait que tout argument $\text{\LaTeX}$ donné entre crochets [...] est un argument facultatif.

1.4 C'est parti !

Nous allons tenter dans cette section, de présenter, à partir d'un document ne contenant que quelques commandes de mise en page, les principes de base de $\text{\LaTeX}$.

```
\documentclass{article}
\begin{document}
Un outil qui vous tombe des mains tombe
toujours dans l'endroit le plus
inaccessible, ou sur le composant le plus
fragile.

Cette loi est l'une des lois de \emph{Murphy}.
\end{document}
```

Un outil qui vous tombe des mains tombe
toujours dans l'endroit le plus inaccessible, ou
sur le composant le plus fragile.

Cette loi est l'une des lois de *Murphy*.

Cet exemple illustre un certain nombre de principes parmi les plus importants de $\text{\LaTeX}$, à savoir :

Ligne vierge $\equiv$ saut de paragraphe : une ligne vierge indique à $\text{\LaTeX}$ la fin d'un paragraphe. Ainsi dans l'exemple précédent, le premier paragraphe commence à «Un outil» et finit avec «fragile.». La commande `\par` est équivalente à la ligne vierge et peut donc également être utilisée pour commencer ou finir un paragraphe.

⁸Ce terme est expliqué au chapitre suivant.

L^AT_EX ignore les sauts de lignes : ce ne sont pas les sauts de lignes dans le document source qui définissent les sauts de lignes dans le document final. L^AT_EX *coupe*, *indente* et *justifie* automatiquement chaque paragraphe, sauf contre-ordre de votre part.

L^AT_EX ignore les espaces multiples : taper un espace ou dix huit mille sept cent quatre vingt quatre espaces est équivalent, comme le montrent les espaces insérés avant « tombe » et avant « l'endroit ». Ceci est aussi vrai pour les sauts de paragraphes : entrer une ligne vierge ou plusieurs revient au même.

\ est le caractère d'échappement : il indique à L^AT_EX que le « mot » qui suit est une séquence de contrôle, c'est-à-dire une commande (ou macro) dans le sens le plus général du terme. Ici, il s'agit de mettre en évidence le mot Murphy. Ceci est effectué grâce à la commande \emph.

{ et } sont des délimiteurs de *groupe*, notions expliquées un peu plus bas.

1.4.1 Quelques caractères sont spéciaux

Comme le suggère l'intervention du caractère \, il existe d'autres symboles ayant une signification spéciale pour L^AT_EX. Il s'agit des 10 caractères suivants :

\ \$ & % # ^ _ { } ~

Voici un petit exemple, utilisant une partie de ces symboles :

```
% paragraphe sans intérêt
\textbf{To be} a subscript :  $x_{i+1}$ ,
or a superscript :  $e^{i\pi}$  ?
that's question~1 ! % or question 2 ?
```

To be a subscript : x_{i+1} , or a superscript : e^i ? that's question 1!

Pour l'instant, il faut donc savoir que :

- % indique à L^AT_EX d'ignorer le restant de la ligne. C'est donc le symbole du commentaire (équivalent au // du C++);
- \$ est le symbole de début et fin de formule. Lorsque L^AT_EX rencontre un \$ il commute en mode **mathématique** jusqu'au symbole \$ suivant;
- ~ est l'espace insécable⁹, il empêche L^AT_EX de faire une césure à cet endroit particulier. Il existe un nombre important de situations où il est nécessaire d'insérer un caractère insécable (tout ce qui est du style : figure~4). Cependant, il n'existe pas de règles systématisant l'usage d'un tel caractère;
- _ et ^ permettent respectivement de passer en indice et en exposant. Attention, ces symboles ne sont autorisés qu'en mode mathématique;
- { et } sont respectivement les caractères de début et fin de groupe. Deux types de groupement sont donnés à titre d'exemple : l'un, en mode mathématique, pour grouper la « sous-formule » à mettre en indice ou en exposant; l'autre pour grouper les mots à mettre en gras.

On peut produire une partie des caractères spéciaux dans le texte grâce aux commandes suivantes :

\\$ \& \% \# \{ \} _

qui donnent respectivement : \$ & % # { } _ . La section 2.2.5 page 21 explique comment produire les autres caractères spéciaux (\ ~ ^).

⁹Voir aussi, le paragraphe 2.10 page 29 sur le contrôle de la césure.

1.4.2 Appel des commandes

Vous avez compris que pour appeler une commande ou macro, il est nécessaire d'insérer le caractère d'échappement — *escape char* en anglais — et de le faire suivre par le nom de la macro que vous voulez utiliser. Mais comment fait L^AT_EX pour repérer la fin du nom de la macro ? Prenons comme exemple la macro `\TeX` qui produit le logo T_EX.

The \TeX book is for \TeX hackers.

`\TeX\` has some powerful macros.

`\LaTeX{}` is a document preparation system

The T_FXbook is for T_FXhackers.

T_EX has some powerful macros.

L^AT_EX is a document preparation system

On peut résumer le mécanisme en deux règles. Il y a deux types de caractères qui indiquent à $\text{\LaTeX}$ la fin du nom de la macro :

- les espaces ; ils sont cependant ignorés dans la production du document ;
- tout caractère autre que les caractères de la catégorie « lettre » (alphabet majuscule et minuscule).

 Le caractère `\` (où est le caractère espace) est appelé un espace de contrôle ; cet espace n'est jamais ignoré par $\text{\LaTeX}$. C'est pourquoi : `et\ \ \ \hop\` produira ; «`et \ hop !`». En fait, il est bon de prendre l'habitude d'appeler les macros sous la forme `\fonction{arguments}` et donc d'utiliser la troisième forme de l'exemple précédent plutôt que la deuxième. Cela évite de se poser le problème de l'espace ignoré¹⁰. On écrira donc «`the  $\text{\TeX}$ book`» avec «`the \TeX{}book`» et « `$\text{\LaTeX}$  is a ...`» avec «`\LaTeX{} is a ...`».

1.4.3 Accents

Les français ont souvent une appréhension à utiliser L^AT_EX à cause des accents. Pas d'aïe ! vous n'aurez pas à saisir les caractères accentués comme indiqué dans le tableau 1.1. Il est quand même bon de noter qu'il est possible d'accentuer (et «cédiller») n'importe quel type de caractère, y compris les majuscules.

TAB. 1.1 – Saisie des accents avec des caractères 7 bits

accent aigu	\'z	ž
accent grave	\'z	ž
accent circonflexe	\^z	ž
cédille	\c{z}	ž
tréma	\"z}	ž

Attention ! S'il est possible de saisir des documents avec des caractères accentués il ne faut pas perdre de vue qu'il faut alors faire appel à un encodage qui est pour l'instant local à une région du globe. On utilise en France le codage Iso8859 avec l'extension latin1 qui permet de manipuler nos jolis accents. Avant de lire précisément le chapitre dédié aux documents rédigés en **français**, nous vous suggérons de rajouter dans votre préambule :

¹⁰Mais pourquoi il nous en parle, alors !?

```
\usepackage[latin1]{inputenc} % codage du fichier source
\usepackage[T1]{fontenc}      % codage des fontes TeX
\usepackage[français]{babel}  % document en français
```

pour «attaquer» un document en français.

1.5 Premiers outils

Voici quelques macros et ligatures à connaître car souvent utilisées dans un document. Tout d'abord, L^AT_EX distingue trois types de tirets :

- - pour «Saint-Étienne» ;
- -- pour «page 12–24» ;
- --- pour ouvrir une parenthèse — comme cela.

Les guillemets doivent être entrés comme ceci :

- ‘ ‘ et ’ ’ pour les textes en “anglais” ;
- « et » si votre clavier le permet,¹¹ pour les textes en «français». La partie française du package babel (cf. chapitre 7 page 87) permet la saisie de caractères à l'aide des commandes \og et \fg, ainsi : \og français\fg{ }.

Voici pour finir quelques commandes utiles :

- \today produit la date du jour (de la compilation) : 5 mars 2008 ;
- \S donne le signe paragraphe : § ;
- \ldots permet de saisir les points de suspension dans un document anglais. . . Ils doivent être saisis avec trois points : . . . dans un document français (voir le chapitre 7 pour quelques notions de typographie française)...

Enfin, souvenez-vous qu'en anglais, on ne saisit pas d'espace avant les ponctuations doubles (: ; ! ?) — contrairement au français. Rappelez-vous aussi que dans ce doux pays qu'est la France, on roule surtout à droite.

1.6 Premières erreurs



Dans ce qui suit nous vous proposons d'examiner les états d'âme de L^AT_EX pendant qu'il compile votre document. Lorsqu'on lance la compilation en ligne de commande, on voit directement cette sortie dans le terminal. De manière à utiliser L^AT_EX de manière la plus enrichissante nous vous incitons à trouver dans votre propre environnement la manière d'examiner les « logs » de L^AT_EX, qui vous indiqueront les messages d'erreurs et autres avertissements survenant lors de la compilation.

1.6.1 Symptômes

Si vous utilisez L^AT_EX en interactif vous serez amenés un jour ou l'autre à voir apparaître à l'écran, un message barbare de ce type :

```
1 This is TeX, Version 3.1415 (C version 6.1)
2 (erreur.tex
3 LaTeX2e <1995/12/01>
4 (/usr/local/lib/texmf/tex/cls/article.cls
5 Document Class: article 1995/11/30 v1.3p Standard LaTeX document class
```

¹¹     sur un clavier sous Linux, par exemple.

```

6 (/usr/local/lib/texmf/tex/clo/size10.clo)) (erreur.aux)
7 ! Undefined control sequence.
8 l.5  paragraphe de ce \empha
9                                     {document}
10 ?

```

Ce message qui vous semble sûrement incompréhensible, est le résultat produit sur le terminal après avoir exécuté $\text{\LaTeX}$ sur le document `erreur.tex` que voici :

```

\documentclass{article}

\begin{document}
Il me semble bien qu'il y ait une erreur dans le premier
paragraphe de ce \empha{document} somme toute assez court.
\end{document}

```

1

1.6.2 Diagnostic

On peut donc expliquer de manière simple le message d'erreur :

ligne 1 vous utilisez $\text{\TeX}$ version π à 10^{-4} près ;

ligne 2 vous compilez le fichier `erreur.tex` ;

ligne 3 vous utilisez $\text{\LaTeX}$ 2_ε version de décembre 95 ;

ligne 4–5 vous utilisez la classe de document standard `article` ;

ligne 6 par défaut, la taille 10pt est utilisée ;

ligne 7 le message d'erreur proprement dit ;

ligne 8–9 la ligne où s'est produite l'erreur ainsi que son numéro dans le document source `erreur.tex` ;

ligne 10 le prompt `?` particulièrement angoissant de $\text{\TeX}$

La « coupure » formée par les lignes 8 et 9, indique précisément l'endroit où $\text{\LaTeX}$ a perdu les pédales. Le message :

```
! Undefined control sequence.
```

vous indique que la commande que vous avez entrée n'est pas connue par $\text{\LaTeX}$. Et effectivement, la commande `\empha` n'existe pas.

1.6.3 Soins

Mais que répondre à $\text{\LaTeX}$, lorsqu'il nous affiche son fameux prompt « `?` » ? Voici, trois solutions, les plus couramment utilisées pour communiquer un peu avec $\text{\LaTeX}$:

- appuyer sur <Entrée> pour ignorer l'erreur ;
- taper `x` permet de quitter la compilation ;
- taper `r` pour demander à $\text{\LaTeX}$ de continuer en ignorant tous les autres messages d'erreur ;
- taper `i` pour insérer une correction et continuer la compilation. Sachant que cette correction ne sera pas insérée dans le document source ;
- taper `h` pour demander un plus d'information quant à l'erreur ; voici ce que vous dit $\text{\TeX}$ pour le `Undefined control sequence` :

The control sequence at the end of the top line of your error message was never `\def`'ed. If you have misspelled it (e.g., `'\hobx'`), type `'I'` and the correct spelling (e.g., `'I\hbox'`). Otherwise just continue, and I'll forget about whatever was undefined.

1.6.4 Une collection de message

$\text{T}_{\text{E}}\text{X}$ et $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ disposent d'un nombre important de messages d'erreur qui correspondent à diverses situations. Ces messages ne sont pas toujours compréhensibles au premier abord. Cependant on peut dire que la plupart des erreurs viennent le plus souvent :

- d'une erreur de syntaxe sur les mots réservés de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$;
- de paires d'accolades mal construites ;
- d'une commande mathématique utilisée en mode texte ;
- d'un mode mathématique non refermé ;
- d'un package que vous avez oublié d'inclure ;
- d'une fin de journée di $\square$ cile ;
- ...

Y a plus qu'à !

Vous avez maintenant compris comment on pouvait créer un document imprimable à partir d'un source $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. Ce chapitre vous a également permis de comprendre le principe de l'appel des commandes. Il ne vous reste qu'à entamer le chapitre suivant pour connaître les di $\square$ érentes fonctionnalités que vous propose le langage $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$.

2

Ce qu'il faut savoir

Sommaire

- 2.1 Mise en évidence
- 2.2 Environnements
- 2.3 Notes de marge
- 2.4 Titres
- 2.5 Notes de bas de page
- 2.6 Entête et pied de page
- 2.7 Flottants
- 2.8 Références
- 2.9 Fichiers auxiliaires
- 2.10 Où il est question de césure

2

*Quand on châtie le railleur, le simple s'assagit ;
quand on instruit le sage, celui-ci gagne en savoir.*

Les proverbes Pr 21 11.

IL SERA QUESTION dans ce chapitre, des commandes de mise en page de base à connaître pour générer un document avec $\text{\LaTeX}$. Nous traiterons en vrac des mises en évidences, des environnements standard $\text{\LaTeX}$, des titres, des notes de bas de page, des entête et pied de page et des environnements flottants. Nous terminerons le chapitre par un exposé du système de références suivi d'une présentation des fichiers auxiliaires générés par $\text{\LaTeX}$. Enfin, ceux qui auront tenu jusque là, auront la chance de pouvoir lire quelques considérations sur la césure.

Toutes ces commandes seront à utiliser avec leur comportement par défaut, c'est-à-dire que nous ne présenterons pas ici la manière de les redéfinir. Vous serez par contre en mesure de produire un document classique avec les mises en page traditionnelles. Pour taper un article plus évolué, vous aurez besoin d'informations sur la manière de produire des formules mathématiques (chapitre 3), quelques infos sur les documents scientifiques (chapitre 6), et éventuellement sur l'inclusion de graphiques (chapitre 5).

2.1 Mise en évidence

Pour comprendre un tant soit peu le mécanisme de sélection de fontes de $\text{\LaTeX}$, il faut savoir qu'on distingue au moins quatre paramètres dans une fonte :

famille : c'est la forme globale de la fonte. $\text{\LaTeX}$ utilise par défaut 3 types de familles : roman, sans sérif et machine à écrire. Le mot anglais utilisé par $\text{\LaTeX}$ est *family*

style : c'est l'allure (en anglais *shape*) de la fonte : *italique penché* et PETITES CAPITALES.

TAB. 2.1 – Déclarations de changement de fontes

Commande	Déclarations	Output
<code>\textrm{...}</code>	<code>{\rmfamily ...}</code>	roman
<code>\textsf{...}</code>	<code>{\sffamily ...}</code>	sans sérif
<code>\texttt{...}</code>	<code>{\ttfamily ...}</code>	machine à écrire
<code>\textup{...}</code>	<code>{\upshape ...}</code>	droit
<code>\textit{...}</code>	<code>{\itshape ...}</code>	<i>italique</i>
<code>\textsl{...}</code>	<code>{\slshape ...}</code>	<i>penché</i>
<code>\textsc{...}</code>	<code>{\scshape ...}</code>	PETITES CAPITALES
<code>\textmd{...}</code>	<code>{\mdseries ...}</code>	medium
<code>\textbf{...}</code>	<code>{\bfseries ...}</code>	gras

graisse : c'est l'épaisseur (*serie* pour L^AT_EX) des traits. Par défaut 2 épaisseurs : médium et gras;

taille : taille des caractères.

2.1.1 Family-shape-series

On distingue deux types de macros pour changer les trois premiers paramètres (cf. tableau 2.1) : les *commandes* et les *déclarations*. Les commandes agissent sur leur argument donné entre accolades. Les déclarations agissent comme des interrupteurs en changeant la valeur d'un de ces paramètres jusqu'à nouvel ordre. En règle générale, on utilisera les commandes pour mettre en évidence un mot ou un groupe de mots :

une `\emph{variable}` de type `\texttt{char}` est `\textsc{Toujours}` codée sur `\textbf{8 bits}`. une *variable* de type `char` est TOUJOURS codée sur 8 bits.

Notez l'utilisation dans l'exemple précédent, de la commande `\emph` (dont la déclaration équivalente est `\em` qui permet de mettre en évidence de manière élégante un groupe de mots. Il est fortement conseillé d'utiliser les *commandes* plutôt que les *déclarations*. Cependant lorsqu'une longue portion de texte est à changer, il sera parfois plus judicieux d'utiliser les commandes :¹

`{\em The music of \bfseries Magma \mdseries is like a mirror where everyone can see a reflection of who he is.}` *The music of Magma is like a mirror where everyone can see a reflection of who he is.*

L'exemple suivant illustre l'utilisation de groupes. La déclaration `\slshape` se situe dans un groupe, elle est donc *locale* à ce groupe. D'autre part, un groupe hérite les paramètres du groupe qui l'englobe. Ainsi, « silence » est écrit en fonte sans sérif (groupe englobant) et *penché* (déclaration locale).

¹Ainsi que lors de la définition de commandes.

`\sffamily` Le jazz est une musique où le
`{\slshape silence\}` a toujours raison ;
 c'est pour cela qu'il n'a pas d'autre
 issue que l'impossible.

Le jazz est une musique où le *silence* a toujours
 raison ; c'est pour cela qu'il n'a pas d'autre issue
 que l'impossible.

2.1.2 Correction italique

Une autre raison pour laquelle il est recommandé d'utiliser les commandes plutôt que les déclarations, est que les commandes effectuent la *correction italique* contrairement aux déclarations. La correction italique est un espace qu'il est nécessaire de rajouter à la fin d'un groupe de mots en italique, pour éviter que celui-ci ne « touche » le mot suivant. Cet espacement est fonction du caractère mis en jeu :

le `{\em chef}` a toujours raison.\par
 le `{\em chef\}` a toujours raison.\par
 le `\emph{chef}` a toujours raison.\par

le *chef* a toujours raison.
 le *chef* a toujours raison.
 le *chef* a toujours raison.

On voit donc clairement que la commande `\emph` effectue la correction, alors qu'il est nécessaire de la faire explicitement à l'aide de la macro `\/`, quand on utilise la forme déclaration.

2.1.3 Tailles

On dispose des macros données au tableau 2.2 pour changer la taille de la fonte en cours. Ces macros sont des *déclarations* et il existe pour chacune d'entre elles un environnement portant le même nom.

TAB. 2.2 – Changement de taille

<code>\Huge</code>	immense	<code>\normalsize</code>	normal
<code>\huge</code>	énorme	<code>\small</code>	petit
<code>\LARGE</code>	très très gros	<code>\footnotesize</code>	plus petit
<code>\Large</code>	très gros	<code>\scriptsize</code>	rikiki
<code>\large</code>	gros	<code>\tiny</code>	minuscule

2.1.4 Quelques recommandations

L'usage veut que dans la mesure du possible, on utilise avec parcimonie les changements de fontes. Il est en effet de mauvais goût d'effectuer des mises en évidence intempestives et inutiles ; le plus généralement elles surchargent le document au lieu de le rendre plus lisible. Voici trois suggestions (toujours d'usage !) sur l'utilisation des changements de fontes :

- préférer la commande `\emph` (par défaut *italique*) que tout autre commande pour mettre en évidence ;
- réserver le gras pour une remarque particulièrement importante ;

- utiliser les petites capitales ne sont à utiliser quasiment exclusivement que pour les noms dans un document en français : Donald K_NUTH ;
- la famille machine à écrire est souvent utilisée pour produire du texte en langage de programmation ou équivalent.

À bon entendeur...

D'autre part nous vous donnons ci-dessous deux considérations quant à l'utilisation du changement de taille et du souligné (commande `\underline`) :

« Perhaps poets who wish to speak in a still small voice will cause future books to make use of frequent font variations, but nowadays it's only an occasional font freak (like the author of this manual) who likes such experiments. »
Donald K_NUTH in the T_EXbook [3]

« Note that underlining for emphasis is considered bad practice in the publishing world. Underlining is only used when the output device can't do highlighting in another way — for example, when using a typewriter. »
Michel GOOSSENS et al. in the L^AT_EX Companion [4]

2.2 Environnements

L^AT_EX propose une série d'outils sous la forme d'*environnements*. Il s'agit d'une structure de bloc dont la syntaxe est la suivante :

```
\begin{nom env}
...
\end{nom env}
```

où `nom env` est le nom d'un environnement. Le premier environnement rencontré jusqu'ici est l'environnement `document`. Entre le `\begin` et le `\end` on insère une portion de texte qui va subir une mise en page particulière.



Notons tout de suite que toute déclaration est locale à un environnement ; et qu'il est bien sûr possible de définir ses propres environnements éventuellement à partir d'autres existants.

Le reste de cette section sera consacré à la description des environnements normalisés de L^AT_EX.

2.2.1 Centrage et alignement

Pour centrer quelques lignes, on utilise l'environnement `center` :

```
... fin de phrase.
\begin{center}
  quelques lignes \
parfaitement centrées \
entre les marges
\end{center}
et le paragraphe continue...
```

```
... fin de phrase.
                quelques lignes
                parfaitement centrées
                entre les marges
et le paragraphe continue...
```

De même on peut aisément aligner un paragraphe à droite grâce à l'environnement `flushright` :

```
... fin de phrase.
\begin{flushright}
  deux lignes\\
  alignées à droite
\end{flushright}
et le paragraphe continue...
```

```
... fin de phrase.
deux lignes
alignées à droite
et le paragraphe continue...
```

Noter l'emploi dans les deux précédents exemples de la commande `\\` pour passer à la ligne. En dehors de cas particuliers (tableaux, titre et auteur d'un document, centrage et alignements notamment), cette commande est à proscrire : pour passer à la ligne, il faut laisser une ligne vierge ou utiliser la commande `\par`.

En général, on emploie l'environnement `flushleft` avec des commandes `\\`. Mais on peut l'utiliser pour produire un paragraphe comme celui-ci, non justifié à droite, en laissant à $\text{\LaTeX}$ le soin d'insérer les sauts de lignes.



La grande majorité des environnements passent à la ligne pour insérer leur contenu. Cependant, il est important de comprendre qu'un environnement *interrompt* le paragraphe dans lequel il est inséré, mais ne le termine pas — vous pourrez d'ailleurs remarquer que la phrase « et le paragraphe continue » n'est pas indentée. En outre, $\text{\LaTeX}$ insère gracieusement autour de chaque environnement un espace vertical.

On peut noter qu'aux trois environnements précédents correspondent respectivement les trois déclarations : `\centering`, `\raggedleft` et `\raggedright`. On peut par exemple écrire :

```
Emacs stands for :

{\centering Emacs\\Makes\\
  A\\Computer\\Slow\\}
```

```
Emacs stands for :
Emacs
Makes
  A
Computer
Slow
```

2.2.2 Listes

$\text{\LaTeX}$ offre la possibilité d'utiliser trois principaux types de *listes* sous forme d'environnement : `itemize`, `enumerate` et `description`. Il est possible de définir ses **propres listes**, si celles de $\text{\LaTeX}$ de vous conviennent pas. Mais voici les listes standard :

Tout d'abord `itemize` qui est une liste d'« items » non numérotés dont le premier niveau est marqué par un tiret (–) en version française et par un point (•) par défaut :

```
... toujours la fin d'une phrase.
\begin{itemize}
\item dans un calcul complexe, un facteur
  du numérateur passe toujours au
  dénominateur
\item une virgule est toujours mal placée
\end{itemize}
et le truc continue, imperturbablement...
```

```
... toujours la fin d'une phrase.
– dans un calcul complexe,un facteurdu nu-
  mérateur passe toujours au dénominateur
– une virgule est toujours mal placée
et le truc continue, imperturbablement...
```

Vient ensuite l'environnement `enumerate`, sur le même principe que le précédent mais où les items sont numérotés. Étant donné que ces environnements peuvent être inclus les uns dans les autres, nous vous présenterons `enumerate` et `description` dans un même exemple :

```
... encore la fin d'une phrase.  
\begin{description}  
\item[\TeX] The \TeX{}book  
\item[\LaTeX] deux livres importants :  
  \begin{enumerate}  
    \item \LaTeX{} : A Document preparation  
      System  
    \item The \LaTeX{} Companion  
  \end{enumerate}  
\end{description}
```

Voici un tableau :

```
\begin{tabular}{|r|c|}
\hline
deux & trois \\
cinq & six \\
\hline
\end{tabular}
```

2.11

Voici un tableau :

deux	trois
cinq	six

on peut donc comprendre grâce à cet exemple, les choses suivantes :

- l'environnement `tabular` attend un paramètre qui est en quelque sorte une « chaîne de format ». À chaque colonne doit correspondre un caractère de positionnement :
- `r` : alignement à droite ;
- `c` : centrage ;
- `l` : alignement à gauche ;
- le caractère `&` est le séparateur des colonnes ;
- la commande `\\` permet de passer à la ligne ;
- les bordures verticales s'insèrent dans la chaîne de mise en page grâce au caractère `|` ;
- les bordures horizontales à l'aide de la commande `\hline`.

On peut donc jouer sur le nombre de `\hline` et de `|` pour changer l'allure des bordures. Le package `array` permet quelques fantaisies avec les tableaux.



Si la plupart des environnements commencent une nouvelle ligne, ce n'est pas le cas de l'environnement `tabular`. Il crée juste une boîte dans le texte courant.

On peut en outre préciser le positionnement vertical du tableau grâce à un argument optionnel :

```
un :
\begin{tabular}[b]{|c|} a\\ \end{tabular}
et deux :
\begin{tabular}[t]{|r|} c\\ \end{tabular}
```

2.12

un :

a
b

 et deux :

c
d

Vous avez donc compris que l'argument `b` (resp. `t`) « pose » (resp. « accroche ») le tableau sur (resp. à) la ligne. Sans cet argument le tableau est centré verticalement, comme dans le premier exemple de la section.

Les tableaux peuvent évidemment ne pas être insérés dans des « phrases » et constituer des « paragraphes » à eux seuls, par exemple en figurant centrés au moyen d'un environnement `center`.

2.2.5 Simulation de terminal

L'environnement `verbatim` insère son contenu mot pour mot. Il offre donc la possibilité de rentrer n'importe quel caractère même spécial, et donc, par exemple d'écrire une portion de code C++:³

```
\begin{verbatim}
class pixel{
    int x,y;
public:
    pixel(int i=0, int j=0); ...};
\end{verbatim}
```

```
class pixel{
    int x,y;
public:
    pixel(int i=0, int j=0); ...};
```

³Notez que le package `listings` est bien mieux adapté à ce genre de problème.



On peut tout écrire dans un environnement verbatim *sauf* la séquence de caractères : `\end{verbatim}` !

Il existe deux commandes permettant de produire une portion de texte comme le fait l'environnement verbatim : il s'agit de `\verb` et `\verb*`. La forme «étoilée» remplace le caractère « » par «□».

L'argument de ces commandes n'est pas donné entre accolades (`{ }`) mais par tout autre caractère : 1° autre que les caractères spéciaux et 2° n'étant pas contenu dans l'argument.

La déclaration : `\verb+\include<stdlib.h>+` permet d'inclure les prototypes de la librairie standard du C.

La déclaration : `\include<stdlib.h>` permet d'inclure les prototypes de la librairie standard du C.



La commande `\verb` ne peut en aucun cas se trouver dans l'argument d'une commande, quelle qu'elle soit.

2

2.2.6 Citations

Les environnements `quote` et `quotation` permettent d'insérer une citation dans le texte. Voici d'abord `quote` :

```
... encore la fin d'une phrase.
\begin{quote}
  Tout est relatif.\hfill\textbf{Einstein}.

  Il n'est pas certain que tout soit certain.
  \hfill\textbf{Pascal}.
\end{quote}
et le paragraphe interrompu, continue...
```

```
... encore la fin d'une phrase.
  Tout est relatif.      Einstein.
  Il n'est pas certain que tout soit
  certain.              Pascal.
et le paragraphe interrompu, continue...
```

La commande `\hfill` insère un espace qui s'étend horizontalement de manière infinie. L'environnement `quotation` diffère quelque peu de `quote` :

```
... encore la fin d'une phrase.
\begin{quotation}
  L'homme est plein d'imperfections mais on
  ne peut que se montrer indulgent si l'on
  songe à l'époque où il fut créé.\par
  \raggedleft Alphonse \textsc{Allais}.
\end{quotation}
et ce brave paragraphe qui continue...
```

```
... encore la fin d'une phrase.
  L'homme est plein d'imperfec-
  tions mais on ne peut que se
  montrer indulgent si l'on songe à
  l'époque où il fut créé.
  Alphonse ALLAIS.
et ce brave paragraphe qui continue...
```

En fait ces deux environnements sont présentés par Leslie LAMPORT, l'un (`quote`) pour une ou plusieurs citations courtes, et l'autre (`quotation`) pour une citation longue.

2.3 Notes de marge

La commande `\marginpar` crée un mini-paragraphe dans la marge, la syntaxe est la suivante :

TAB. 2.3 – *Sectioning commands*

<code>\part{...}</code>	<code>\chapter{...}</code>	
<code>\section{...}</code>	<code>\subsection{...}</code>	<code>\subsubsection{...}</code>
<code>\paragraph{...}</code>	<code>\subparagraph{...}</code>	

`\marginpar{texte}`

Pour distinguer la page droite de la page gauche en mode recto-verso, on pourra utiliser :

`\marginpar[textegauche][textedroite]`

où *textegauche* et *textedroite* seront respectivement les textes qui apparaîtront en marge selon la parité du numéro de la page. Ainsi :

`\marginpar[Youhou !][Coucou !]`

donne ce que vous pouvez constater dans la marge.

Coucou

2

2.4 Titres

Le tableau 2.3 montre les commandes de *section* disponibles dans $\text{\LaTeX}$. La commande `\chapter` n'est pas disponible pour la classe de document `article` ; et aucune commande de titres ne peut être utilisée dans la classe `letter`. Pour l'instant, il faut savoir les deux choses suivantes :

- chaque titre résultant d'une commande de section est automatiquement *numéroté* et mis dans la table des matières le cas échéant ;
- la commande `\tableofcontents` produit une *table des matières* à l'endroit où est insérée cette commande.



D'autre part toutes les commandes de titres ont un style associé que l'on peut éventuellement redéfinir. Enfin, ces commandes effectuent automatiquement les espacements verticaux avant et après le titre ; ainsi toute ligne vierge insérée avant ou après la commande est ignorée.

... tiens c'est la fin d'une phrase.
`\section{La quantification}`
 Le processus de quantification...

... tiens c'est la fin d'une phrase.

3.2 La quantification

Le processus de quantification...

Il existe une forme « étoilée » (par exemple : `\section*`) de chaque commande de titres permettant d'insérer un titre non numéroté. Mais attention, ce titre n'apparaîtra pas dans la *table des matières*. Les commandes de section prennent également un argument optionnel permettant de préciser une entrée de table matières différente du titre de la section. Par exemple :

`\section[Paulette]{C'était bien, c'était chouetteuuuu}`

insère « Paulette » dans la table des matières en lieu et place du titre inséré dans le document.

2.5 Notes de bas de page

L'insertion d'une note de bas de page s'effectue de manière simple par la commande `\footnote{texte}`. La numérotation est automatique, et par défaut, les notes sont numérotées à l'intérieur d'un chapitre. Voici ce que donne $\text{\LaTeX}$:

Contre toute attente, c'est la commande `\verb+footnote+\footnote{Comme son nom l'indique...}` qui insère une note de bas de page.

Contre toute attente, c'est la commande `footnotea` qui insère une note de bas de page.

^aComme son nom l'indique...

Il arrive lorsqu'on travaille en milieu particulièrement hostile que la commande `\footnote` ne produise pas l'effet désiré. Il est alors nécessaire de procéder en deux temps :

1. poser une *marque de note*, commande `\footnotemark` ;
2. entrer le *texte* de la note de base de page — commande `\footnotetext` — lorsque les conditions sont plus favorables.

Par exemple, il semble délicat de mettre une note de bas de page dans un tableau, on écrira donc :

```
\begin{tabular}{cc}
  un & deux\footnotemark \\
  trois & quatre \\
\end{tabular}\footnotetext{Une note.}
```

un	deux ^a
trois	quatre

^aUne note.

2.6 Entête et pied de page

Les commandes standard de $\text{\LaTeX}$ permettant de personnaliser les entêtes et pied de page sont assez rudimentaires mais méritent d'être mentionnées ici, puisqu'elles peuvent suffire dans certains cas.



Nous ne nous attarderons pas plus sur ces commandes, car il nous semble que le package `fancyhdr` — documenté dans `fancyhdr.dvi` — est beaucoup plus confortable à utiliser et offre des fonctionnalités bien plus intéressantes que les options standard de $\text{\LaTeX}$. L'utilisation de ce package pour produire les entête et pied de page du manuel que vous avez sous les yeux est expliquée à la section 10.4.

Sans faire appel à un package particulier, on peut spécifier le style d'entête et pied de page à l'aide de la commande `\pagestyle` :

`\pagestyle{style}`

dans le préambule du document ; le paramètre *style* pouvant prendre les valeurs suivantes :

- `empty` ni entête, ni pied de page ;
- `plain` c'est le style par défaut, le pied de page contient les numéros de pages centrés ;
- `headings` suivant le style de document, un certain nombre d'informations est inséré dans l'entête et le pied de page (par exemple dans le style `report` en recto-verso, est inséré en entête : soit le titre du chapitre en cours, soit le titre de la section en cours) ;

– myheadings un style qui permet de personnaliser les informations à insérer.
Il existe d'autre part une commande `\thispagestyle`, qui permet de changer ou de spécifier le style de la page courante.

2.7 Flottants

L^AT_EX offre à ses valeureux utilisateurs la possibilité d'utiliser des environnements *flottants*. Ces environnements ont la particularité de rendre « flottants » leur contenu. C'est-à-dire que L^AT_EX choisit à partir d'un algorithme qui tient compte d'un certain nombre de paramètres, la position de l'environnement dans le document.



Contrairement à ce que leur nom laisse croire, les environnements de L^AT_EX `figure` et `table` ne sont pas spécialement conçus pour insérer des figures et des tables ! En fait ils sont conçus uniquement pour faire flotter leur contenu et laisser la possibilité d'insérer une légende. Le contenu à proprement parler peut être constitué de ce que bon vous semble, pas nécessairement du graphique.

2

2.7.1 Figure et table

L'environnement `figure` est en général utilisé pour les graphiques, et l'environnement `table`, pour les tableaux. Chacun de ces environnements possède une légende. La syntaxe d'utilisation est la suivante :

Ce paragraphe contient un environnement flottant de type « figure ». Le contenu est donc susceptible de se déplacer dans la page.

```
\begin{figure}
  \begin{center}
    0 + 0 \\
    =
  \end{center}
  \caption{La tête à toto}
\end{figure}
```

Ce paragraphe contient un environnement flottant de type « figure ». Le contenu

$$0 + 0 \\ =$$

Figure 1: La tête à toto

est donc susceptible de se déplacer dans la page.

Vous noterez que c'est la commande `\caption` qui produit la légende. Le texte « Figure 1: » est inséré automatiquement avec le numéro correspondant à la figure. Le « style » de la légende est bien entendu personnalisable.

2.7.2 Placement

L^AT_EX tente de placer le contenu flottant en fonction des paramètres qu'on indique entre crochets après le `\begin` du flottant :

- h : là où il apparaît dans la source ;
- t : en haut de la page ;
- b : en bas de la page ;
- p : seul sur une page

Notons qu'il arrive parfois que l'on s'arrache les cheveux, pour placer les environnements flottants. Pour ne pas s'énerver, il faut comprendre — et accepter — que L^AT_EX utilise plusieurs paramètres pour placer les `figure` et `table`. Notons parmi ces paramètres :

- le nombre maximum d'environnements flottants en haut et en bas de page ;

- le pourcentage maximum de la surface de la page qu'occupe un flottant en haut et en bas de la page ;
- les espacements avant et après le flottant.

Si vous avez des problèmes⁴ pour placer vos figures, nous vous conseillons de suivre ces quelques recommandations :

- si vous tenez à écrire «*comme le montre la figure :*» en attendant la figure à la suite, *n'utilisez pas l'environnement figure* !
- utilisez plutôt le système de *référence* et écrivez «comme le montre la figure 3» ;
- on a toujours tendance à faire des figures énormes : rétrécissez-les !
- si vous avez des tableaux à rallonge, mettez-les en annexe, puisque de toutes façons ils gêneront le lecteur ;
- les paramètres de L^AT_EX sont étudiés pour équilibrer le texte et les figures dans le document. Donc, si votre document est une bande dessinée, attendez vous au pire...
- ne vous souciez du placement des figures qu'au moment d'imprimer votre document final.

2.7.3 Liste des figures

La commande `\listoffigures` (resp. `\listoftables`) insère une liste des figures (resp. des tableaux) de votre document. La liste est imprimée là où apparaît la commande. Ces commandes produisent un fichier portant l'extension `.lof` (resp. `.lot`). En outre, de manière analogue aux commandes de sections qui alimentent la table des matières, la commande `\caption` prend un argument optionnel permet-

1. On pose une étiquette avec la commande `\label` :

`\label{étiquette}`

où `étiquette` est une chaîne de caractères ne comprenant pas de caractères spéciaux.

2. On fait référence au numéro de l'objet référencé avec la commande `\ref` :

`\ref{étiquette}`

On fait ensuite référence à la page avec `\pageref` :

`\pageref{étiquette}`

2.8.2 Que référencer ?

Les objets que l'on peut référencer sont les suivants :

- les titres ;
- les flottants (figure, table, ...);
- les équations (cf. chapitre 3);
- les items de liste énumérée (enumerate par exemple);
- etc.

Voici un exemple synthétisant les trois commandes de référencement :

```
\section{Second degré}\label{sec-2degre}
Ce sont les équations du type :
\begin{equation}
ax^2 + bx + c = 0 \label{equ}
\end{equation}
L'équation \ref{equ} de la section
\ref{sec-2degre} page \pageref{sec-2degre}
patati patala...
```

3.5 Second degré

Ce sont les équations du type :

$$ax^2 + bx + c = 0 \quad (2.12)$$

L'équation 2.12 de la section 3.5 page 13 patati patala...

Dans cet exemple on fait référence à une `\section` et une `\equation` (cf. chapitre 3). En outre, on fait référence à la page où apparaît la section en question.



Lorsque vous placez un `\label` dans un environnement flottant, placez le toujours après la commande `\caption`. Sinon, la référence « pointera » sur la section et non sur la figure.

2.9 Fichiers auxiliaires

Pour bien comprendre le mécanisme de référencement, il nous reste à examiner ce que $\text{\LaTeX}$ écrit sur votre disque lorsqu'il compile un fichier source. Pour l'instant, voici les fichiers que vous pourrez rencontrer :

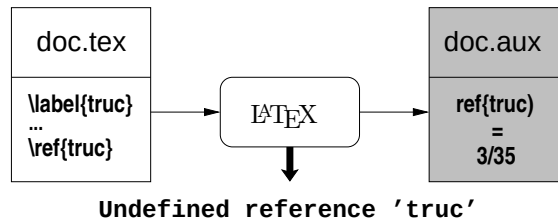
- `dvi` l'image de votre document ;
- `log` c'est le bavardage de $\text{\LaTeX}$ lors de la dernière compilation. En général, il correspond peu ou prou à ce que vous avez sur votre terminal au moment de la compilation ;
- `aux` le fichier auxiliaire, il stocke les informations concernant les références, les numéros de pages, les titres, ... ;

- `toc` le fichier contenant la table des matières ;
- `lof` le fichier contenant la liste des figures.

2.9.1 Interaction avec les références

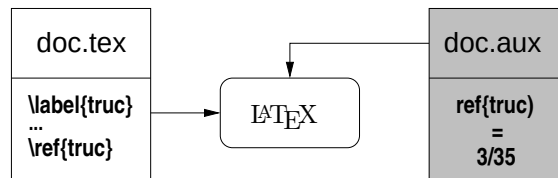
$\text{\LaTeX}$ gère les références de la manière suivante : lors d'une première compilation, il stocke les références dans le fichier `nom-doc.aux` où `nom-doc` est le nom de votre document. À l'aide d'un exemple où l'on aurait placé une étiquette `truc` pour la section 3 à la page 35 d'un document, voyons le principe du mécanisme de résolution des références.

- la première compilation avec $\text{\LaTeX}$ stocke dans le fichier auxiliaire `.aux` le numéro de l'étiquette (le numéro de la section dans notre exemple) et le numéro de la page où cette étiquette apparaît :



$\text{\LaTeX}$ envoie donc lors de cette compilation un avertissement précisant que l'étiquette `truc` est indéfinie ;

- on effectue donc une deuxième compilation qui va cette fois exploiter le contenu du fichier auxiliaire :



Les références peuvent être incorrectement définies dans les situations suivantes :

- vous avez inséré une nouvelle étiquette, et c'est la première compilation que vous effectuez (les références sont *indéfinies*) ; et vous aurez pour cette nouvelle étiquette un message :

Reference '`vlunch`' on page 2 undefined on input line 17.

- les changements que vous avez apportés à votre document ont sans doute changé la numérotation des pages ou le placement des objets (figures, équations,...), les références sont alors *mal définies*, et vous serez averti par un message en fin de compilation :

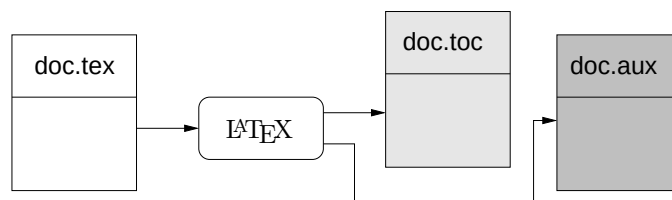
Label(s) may have changed.
Rerun to get cross-references right.

- vous faites référence à une étiquette qui n'existe pas. Dans ce cas, 18 compilations ne changeront rien à votre problème.

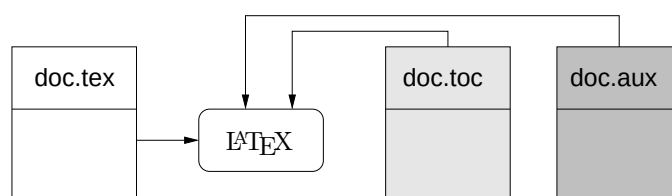
2.9.2 Interaction avec la table des matières

On retrouve un peu le même principe avec la table des matières. Lorsque vous insérez la commande `\tableofcontents` dans votre document, la table des matières va être créée en deux étapes, comme suit :

1. un premier parcours pour récupérer les informations liées aux *titres* de tout le document et stocker dans le fichier `nom-du-document.toc` :



2. un deuxième passage pour inclure `nom-du-document.toc` — donc la table des matières — dans le document final :



Vous serez alors confrontés au phénomène suivant : lorsqu'au cours de la rédaction d'un document contenant déjà l'ordre `\tableofcontents`, vous insérez une commande de section, elle n'apparaîtra dans la table des matières qu'après deux compilations.

2.9.3 Petits conseils

Prenez l'habitude de créer un répertoire pour chaque document que vous rédigez. $\text{\LaTeX}$ crée en effet plusieurs fichiers autour de votre `.tex`⁵. D'autre part, ne vous souciez pas trop, lors de la rédaction de votre document, de savoir si les références ou la table des matières sont à jour : elles le seront bien un jour ou l'autre ! En fait, il faut s'assurer que les références sont correctes avant d'imprimer.

Enfin, de même qu'on effectue de temps en temps un `make clean` lorsqu'on n'est plus sûr de ses fichiers objets, il est bon quand il vous semble que tout va mal, d'effacer les **fichiers auxiliaires** et de reprendre la compilation.

2.10 Où il est question de césure

$\text{\LaTeX}$ s'appuie sur $\text{\TeX}$ pour effectuer la césure des mots en fonction d'une langue déterminée. Cet algorithme décrit à l'annexe H du $\text{\TeX}$ book constitue un des aspects les plus réussis de $\text{\TeX}$. Une manière de reconnaître un document généré par $\text{\LaTeX}$ est d'examiner la manière dont sont coupés les paragraphes ; beaucoup d'autres logiciels se contentent d'insérer des blancs entre les mots. Il existe cependant des situations où $\text{\LaTeX}$ ne peut effectuer une césure correcte. Dans ce cas, $\text{\LaTeX}$ vous avertira par l'un des deux messages terrifiants :

⁵Et encore il n'a pas encore été question de bibliographie, ni d'index et de glossaires...

Underfull \hbox (badness 1810) detected at line 33

ou bien :

Overfull \hbox (14.24376pt too wide) detected at line 41

À un très bas niveau, T_EX produit votre document en assemblant des *boîtes*. Chaque caractère est contenu dans une boîte qui lui est propre ; les mots sont formés par assemblage de ces boîtes. Et ainsi de suite, pour les lignes qui forment des paragraphes puis des pages.

Pour résumer et présenter les choses de manière simple, disons que T_EX est en mode horizontal pour assembler les « mots » et manipule alors des \hbox ; il est en mode vertical et manipule des \vbox lorsqu'il crée les pages. Aussi, lors de l'assemblage de ces boîtes, si T_EX juge que le résultat ne sera pas esthétique, il vous

Guider la césure

On peut aider L^AT_EX à couper un mot en lui indiquant les endroits où la césure peut être effectuée, en insérant aux endroits nécessaires, la commande `\-`. Par exemple, si L^AT_EX a du mal à couper le mot «nonmaiçavapamieu», vous pouvez entrer :

```
non\ -mai\ -ça\ -va\ -pa\ -mieu
```

Si vous utilisez ce mot fréquemment, vous pouvez, pour vous épargner d'indiquer les césures comme ci-dessus, entrer dans le préambule la commande `\hyphenation` :

```
\hyphenation{non-mai-ça-va-pa-mieu}
```

qui indique à L^AT_EX comment couper ce mot étrange.

Forcer la césure

Vous pouvez forcer la césure, en insérant la commande `\linebreak[nombre]`, mais cela peut avoir des résultats catastrophiques. Si vous voyez ce que je veux dire. Le paramètre `nombre` permet de moduler la commande `\linebreak`. Vous avez la possibilité de formuler un souhait timide : `\linebreak[0]` ou un ordre à ne pas discuter : `\linebreak[4]`.

La commande `\pagebreak[nombre]` est la commande correspondant aux coupures de pages. D'autre part, deux commandes sont disponibles pour effectuer un saut de page :

- `\clearpage` finit la page actuelle ;
- `\cleardoublepage` finit la page actuelle, et assure de commencer sur une page impaire, en mode recto verso.

Ces deux commandes forcent L^AT_EX à insérer toutes les figures flottantes en cours de placement.



Une autre intervention manuelle pratique dans certaines situations, consiste à agrandir la hauteur de la page actuelle avec la commande `\enlargethispage` suivi d'une **dimension** puis d'insérer un saut de ligne :

```
\enlargethispage{10cm}      ← au niveau de la page trop courte
[...le texte un peu trop long...]
\clearpage                  ← fin explicite de la page allongée de 10cm
```

Empêcher la césure

Il existe trois moyens de forcer L^AT_EX à ne pas couper le texte :

1. insérer l'espace insécable ~ ;
2. mettre un mot dans une boîte⁶ avec la commande `\mbox{mot}` ;
3. utiliser l'ordre `\nolinebreak` :

```
\nolinebreak[nombre]
```

pour empêcher les sauts de lignes, et la commande `\nopagebreak` :

```
\nopagebreak[nombre]
```

où `nombre` a la même signification que pour les commandes `\linebreak` et `\pagebreak`.

⁶Car T_EX ne coupe jamais une boîte.

Conclusion

Ce chapitre a présenté les fonctionnalités standard de $\text{\LaTeX}$. Si vous avez lu attentivement jusqu'ici, vous devriez pouvoir produire n'importe quel document simple (sans formule ni graphique, pour l'instant). Si vous n'êtes pas encore en mesure de personnaliser vos documents, ils seront tout de même d'une très bonne qualité typographique en vous évitant de vous poser des questions métaphysiques sur la « bonne » largeur d'une marge ou le « bon » écart entre un titre et le texte,... En effet, les comportements par défaut de $\text{\LaTeX}$ répondent, pour la plupart, à des règles en usage dans le monde de l'imprimerie.

3

Mathématiques

Sommaire

- 3.1 Les deux façons d'écrire des maths
- 3.2 Commandes usuelles
- 3.3 Fonctions
- 3.4 Des symboles les uns sur les autres
- 3.5 Deux principes importants
- 3.6 Array : simple et efficace
- 3.7 Équations et environnements
- 3.8 Changer le style en mode mathématique

3

Voici les noms des douze apôtres :
en tête Simon que l'on appelle Pierre [...]
L'Évangile selon Saint Matthieu Mt 10 2.

UN DES ASPECTS pratique et rigolo¹ de $\text{\LaTeX}$ est bien sûr la génération de formules mathématiques ; elles seront naturellement belles, sans que vous n'ayez à faire quoique ce soit.² De plus, si vous avez un mauvais souvenir d'un certain éditeur d'équations, réjouissez vous : vous n'avez pas besoin de souris pour écrire des équations ! La génération d'équations avec $\text{\LaTeX}$ est un domaine particulièrement vaste. Nous présenterons ici les bases requises pour produire les formules « usuelles ». Ce chapitre ne constitue donc qu'une petite introduction à la manipulation des formules avec $\text{\LaTeX}$.



Les commandes standard de $\text{\LaTeX}$ permettent de produire la plupart des équations mathématiques usuelles. Il est cependant conseillé d'utiliser les extensions de l'*American Mathematical Society* nommées `amsmath` et `amssymb` simplifiant la mise en forme dans beaucoup de situations.

3.1 Les deux façons d'écrire des maths

$\text{\LaTeX}$ distingue deux manières d'écrire des mathématiques. L'une consiste à insérer une formule dans le texte, comme ceci : $ax + b = c$, l'autre à écrire une ou plusieurs formules dans un environnement, par exemple :

$$dU = \delta\mathcal{W} + \delta\mathcal{Q}$$

sachant que chacun de ces deux modes respectent un certain nombre de principes quant à la taille et la position des différents symboles. Voici un exemple avec les deux modes :

¹Si, si ! Il y a même des gens qui font des formules juste pour le plaisir !

²Ou alors juste deux ou trois petites choses...

Déterminer la fonction dérivée de $f(x)$:

```
\begin{displaymath}
f(x)=\sqrt{\frac{x-1}{x+1}}
\end{displaymath}
```

si elle existe.

Déterminer la fonction dérivée de $f(x)$:

$$f(x) = \sqrt{\frac{x-1}{x+1}}$$

si elle existe.

Cet exemple nous montre donc que l'on entre en mode mathématique « interne » grâce au symbole \$, et que le même symbole \$ permet d'en sortir. D'autre part, on utilise ici l'environnement `displaymath` qui est le plus simple pour produire des équations. Ce dernier peut être saisi grâce aux commandes `\[` et `\]` (cf. § 3.7.1 page 42).

Nous vous présenterons au § 3.7 les différents environnements de $\text{\LaTeX}$.

3.2 Commandes usuelles

3.2.1

Indice et exposant

Comme mentionné au § 1.4.1 page 10, `_` et `^` sont les commandes permettant de produire respectivement *indice* et *exposant*. Il est nécessaire de « grouper » les arguments entre accolades pour que ces commandes agissent sur plusieurs symboles.

<code>x_2</code>	x_2	<code>x_{2y}</code>	x_{2y}	<code>x_{t_0}</code>	x_{t_0}
<code>x^2</code>	x^2	<code>x^{2y}</code>	x^{2y}	<code>x_{t^0}</code>	x_{t^0}
		<code>x^{2y}_{t_0}</code>	$x^{2y}_{t_0}$	<code>x_{t^0}^{2y}</code>	$x_{t^0}^{2y}$

3.2.2 Fraction et racine

Voici comment produire *racines* et *fractions* :

- la commande `\frac{num}{denom}` produit une fraction formée par le numérateur `num` et le dénominateur `denom` ;
- la commande `\sqrt[n]{expr}` affiche la racine `n`^e de son argument `arg`.

Notons que ces deux commandes ne produisent pas le même affichage selon le mode mathématique : interne ou équation. Ainsi voici une fraction : $\frac{1}{\sin x + 1}$ et une racine : $\sqrt{3x^2 - 1}$ et leur équivalent en mode équation :

$$\frac{1}{\sin x + 1} \quad \sqrt{3x^2 - 1}$$

Pour en finir avec ces deux commandes, voyons comment elles peuvent être imbriquées et l'effet que cela produit :

```
\begin{displaymath}
\sqrt{\frac{1+\sqrt[3]{3x+1}}{3x+\frac{1-x}{1+x}}}
\end{displaymath}
```

$$\sqrt{\frac{1 + \sqrt[3]{3x+1}}{3x + \frac{1-x}{1+x}}}$$

3.2.3 Symboles

Symboles usuels

Le tableau 3.1 donne les macros produisant une partie des symboles dont vous pourriez avoir besoin.

TAB. 3.1 – Symboles mathématiques usuels

<code>\pm</code>	$\pm$	<code>\otimes</code>	$\otimes$	<code>\cong</code>	$\cong$	<code>\imath</code>	$\imath$
<code>\mp</code>	$\mp$	<code>\oslash</code>	$\oslash$	<code>\subset</code>	$\subset$	<code>\jmath</code>	$\jmath$
<code>\div</code>	$\div$	<code>\odot</code>	$\odot$	<code>\supset</code>	$\supset$	<code>\ell</code>	ℓ
<code>\ast</code>	$*$	<code>\leq</code>	$\leq$	<code>\subseteq</code>	$\subseteq$	<code>\aleph</code>	$\aleph$
<code>\times</code>	$\times$	<code>\geq</code>	$\geq$	<code>\supseteq</code>	$\supseteq$	<code>\nabla</code>	∇
<code>\bullet</code>	$\bullet$	<code>\equiv</code>	$\equiv$	<code>\in</code>	$\in$	<code>\ </code>	$\ $
<code>\circ</code>	$\circ$	<code>\ll</code>	$\ll$	<code>\ni</code>	$\ni$	<code>\partial</code>	∂
<code>\star</code>	$\star$	<code>\gg</code>	$\gg$	<code>\emptyset</code>	$\emptyset$	<code>\wedge</code>	$\wedge$
<code>\setminus</code>	$\setminus$	<code>\sim</code>	$\sim$	<code>\forall</code>	$\forall$	<code>\vee</code>	$\vee$
<code>\oplus</code>	$\oplus$	<code>\simeq</code>	$\simeq$	<code>\infty</code>	∞	<code>\cup</code>	$\cup$
<code>\ominus</code>	$\ominus$	<code>\approx</code>	$\approx$	<code>\exists</code>	$\exists$	<code>\cap</code>	$\cap$



Nous avons recensé près de 450 symboles disponibles avec les packages latexsym et amssymb. Notre but n'est donc pas de les présenter ici ! Le tableau 3.1 est une sélection parmi les symboles standard. Nous avons jugé qu'ils faisaient partie des symboles les plus utiles — ce qui, malgré la présence tout à fait fortuite de l'aleph dans ce tableau, démontre que le niveau en mathématiques de l'auteur de ce document avoisine le ras des pâquerettes.

Points de suspension

On utilise couramment pour économiser de l'encre des points de suspension dans des formules. Il en existe de trois types. La commande `\dots` produit des points « posés » sur la ligne :

`\vec{c}_0, \vec{c}_1, \dots, \vec{c}_N` est l'ensemble des N couleurs.

$C = \{e_0, e_1, \dots, e_N\}$ est l'ensemble des N couleurs.

La commande `\cdots` produit des points centrés verticalement sur le signe égal :

$\vec{\mu} = \frac{1}{N} (\vec{c}_0 + \vec{c}_1 + \cdots + \vec{c}_N)$ est la moyenne des N couleurs.

$\tilde{c} = \frac{1}{N} (e_0 + e_1 + \cdots + e_N)$ est la moyenne des N couleurs.

Enfin les commandes `\vdots` et `\ddots` sont à utiliser essentiellement dans les matrices (cf. § 3.6 et l'exemple 3.15). Ces deux commandes produisent respectivement : $\vdots$ et $\ddots$.

Flèches

Voici un moyen simple pour mémoriser les commandes permettant de générer des flèches :

- toutes les commandes finissent par `arrow` ;
- le préfixe obligatoire `left` ou `right` indique la direction ;
- le préfixe facultatif `long` donne une version longue ;
- la première lettre de la commande mise en majuscule rend la flèche double ;
- on peut mettre des flèches aux deux extrémités en collant les deux mots `left` et `right`.

ainsi :

<code>\rightarrow</code>	donne	$\rightarrow$
<code>\Longleftarrow</code>	donne	$\Longleftarrow$
<code>\Leftarrow</code>	donne	$\Leftarrow$
<code>\Longleftrightarrow</code>	donne	$\Longleftrightarrow$

Lettres grecques

Les lettres grecques s'utilisent de la manière la plus simple qui soit : en les appelant par leur nom. Ainsi : `\alpha` donne « α ». Mettre une majuscule à la première lettre de la commande donne la lettre correspondante : `\Gamma` donne « Γ ». Attention, toutes les majuscules ne sont pas disponibles dans l'alphabet grec, on mettra par exemple en majuscule, avec la commande `\Alpha` (qui n'existe pas).

L'ensemble des réels

Une question «cruciale» que se posent les rédacteurs de documents scientifiques est : «Comment peut/doit-on écrire l'ensemble des réels?». Les avis sont partagés à ce sujet. Historiquement, on a vu qu'initialement, dans les ouvrages de mathématiques, le symbole $\mathbb{R}$ était typographié en gras («Soit $x \in \mathbb{R}$ ») et que les professeurs pour écrire les réels sur un tableau avec une craie avaient recours à l'artifice de repasser plusieurs fois la lettre «R»; cette pratique pénible aurait évolué vers l'écriture «bien belle» «Soit $x \in \mathbb{R}$ ». Il y a donc les adeptes du $\mathbb{R}$, du $\mathbb{R}$, etc. Pour choisir par soi-même, voir les packages :

- `bbm` qui propose la commande `\mathbbm{R}` produisant $\mathbb{R}$, la commande `\mathbbbmss{R}` produisant $\mathbb{R}$, etc.
- `bbold` qui propose la commande `\mathbbbm{R}` produisant $\mathbb{R}$, etc.
- `amssymb` qui propose les commandes `\mathbb{R}` produisant $\mathbb{R}$ et `\mathbf{R}` produisant $\mathbb{R}$.

3.3 Fonctions

$\sin^2 x + \cos^2 x = 1$

3.6

$\sin^2 x + \cos^2 x = 1$

La différence réside dans le fait que L^AT_EX traite la chaîne `cos` comme une suite de variable (donc produites en italiques) alors que la fonction `\cos` produit «cos» en roman. Une autre différence importante est le placement d'éventuels indices (cf. l'exemple de la fonction `\max` ci-dessous). Parmi les fonctions mathématiques standard de L^AT_EX, on trouvera :

- toutes les fonctions trigonométriques : `\sin`, `\cos` et `\tan`. En rajoutant `arc` devant, vous aurez les réciproques, et `h` derrière vous obtiendrez les versions hyperboliques.
- les logarithmes népérien et décimal définis respectivement par les fonctions `\ln` et `\log`.
- les fonctions `\sup`, `\inf`, `\max`, `\min`, et `\arg` qui vous permettront de générer des formules de ce genre :

```
\begin{displaymath}
T=\arg \max_{t<0} f(t)
\end{displaymath}
```

3.7

$T = \arg \max_{t < 0} f(t)$

3

Notez l'utilisation de l'opérateur indice `_` et le placement résultant avec la commande `\max`.

3.3.2 Intégrales, sommes et autres limites

L^AT_EX utilise une syntaxe simple pour produire *intégrales*, *sommes*, etc. La syntaxe est la suivante :

`\op_{inf}^{sup}`

où `op` est l'un des opérateurs `sum`, `prod`, `int` ou `lim` et `inf` et `sup` sont les bornes inférieure et supérieure de la somme ou de l'intégrale. Ainsi on peut donc écrire :

Somme des termes d'une suite géométrique :

```
\begin{displaymath}
\sum_{i=0}^n q^i = \frac{1-q^{n+1}}{1-q}
\end{displaymath}
```

3.8

Somme des termes d'une suite géométrique :

$$\sum_{i=0}^n q^i = \frac{1 - q^{n+1}}{1 - q}$$

Le produit `\prod` s'utilise de manière analogue avec la commande `\prod`. Un exemple avec une intégrale, en veux-tu en voilà :

On définit le logarithme népérien de $x > 0$ comme suit :

```
\begin{displaymath}
\ln(x) = \int_1^x \frac{1}{t} dt, \mathrm{d}t
\end{displaymath}
```

3.9

On définit le logarithme népérien de $x > 0$ comme suit :

$$\ln(x) = \int_1^x \frac{1}{t} dt$$

La commande `\,` permet d'insérer un léger blanc avant le «dt» (cf. § 3.5.1). Si vous êtes plutôt *curviligne*, vous pouvez utiliser `\oint` qui donne : $\oint$. Bon, je vous donne juste un exemple avec une limite mais c'est bien parce que c'est vous :

$f(x)$ admet une limite ℓ en x_0 :

```
\begin{displaymath}
\lim_{x \rightarrow x_0} f(x) = \ell
\end{displaymath}
```

$f(x)$ admet une limite ℓ en x_0 :

$$\lim_{x \rightarrow x_0} f(x) = \ell$$

J'espère que vous avez apprécié le beau ℓ ; pour se fixer les idées sur les deux modes mathématiques, voici les mêmes formules mais incrustées dans le texte. Donc d'abord la sommation : $\sum_{i=0}^n q^i = \frac{1-q^{n+1}}{1-q}$, ensuite l'intégrale : $\int_1^x \frac{1}{t} dt = \ln(x)$, et enfin la limite : $\lim_{x \rightarrow x_0} f(x) = \ell$.

3.4 Des symboles les uns sur les autres

3.4.1 L'opérateur not

L'opérateur `\not` permet de produire la « négation » d'une relation :

⌵

Soit $x \notin I$ un réel...

3.11

Soit $x \notin I$ un réel...

le résultat est donc un « slash » sur le symbole suivant. Attention, cet opérateur n'est pas très performant : `\not\longrightarrow` donne : $\not\longrightarrow$, mais est satisfaisant pour les symboles d'une largeur raisonnable.

3.4.2 Accents

Il est souvent utile,³ d'accentuer les symboles en guise de notation particulière. Voici les accents disponibles :

<code>\hat{x}</code>	$\hat{x}$	<code>\check{x}</code>	$\check{x}$	<code>\breve{x}</code>	$\breve{x}$
<code>\acute{x}</code>	$\acute{x}$	<code>\grave{x}</code>	$\grave{x}$	<code>\tilde{x}</code>	$\tilde{x}$
<code>\bar{x}</code>	$\bar{x}$	<code>\dot{x}</code>	$\dot{x}$	<code>\ddot{x}</code>	$\ddot{x}$

3.4.3 Vecteurs

Il existe deux⁴ façons d'obtenir un vecteur :

- `\vec` pour les petits symboles car `\vec` est une commande d'accentuation ;
- `\overrightarrow` dans les autres cas.

Soit $\overrightarrow{A \! B}$ défini dans la base $(\vec{i}, \vec{j})$.

3.12

Soit $\overrightarrow{AB}$ défini dans la base $(\vec{i}, \vec{j})$.

Notez que $\vec{A \! B}$ aurait donné : $\vec{AB}$ (voir aussi le paragraphe 3.5.1 pour la signification de la commande `\!`). Remarquez également les commandes `\imath` et `\jmath` qui fournissent les lettres « i » et « j » sans point : $\imath$ et $\jmath$.

³En réalité les mathématiciens dignes de ce nom raffolent de ce genre de petits chapeaux au dessus des symboles ; certains en superposent même deux, voire trois...

⁴Le package `esvect` d'Eddie SAUDRAIS permet de produire des vecteurs avec des flèches mieux dessinées que celles proposées ici.

TAB. 3.2 – Espacement en mode mathématique

<code>\!</code>	<i>(rien)</i>	<code>\,</code>	<code>\:</code>
<code>\;</code>	<code>\sq</code>	<code>\quad</code>	<code>\qquad</code>

3.4.4 Commande `\stackrel`

La commande `\stackrel` permet de poser deux symboles l'un sur l'autre :

`\stackrel{symp_1}{symp_2}`

met le `symp_1` sur `symp_2`. Par exemple :

`x\stackrel{f}{\longmapsto}y`

donne : $x \xrightarrow{f} y$.

3.5 Deux principes importants

3

Pour bien comprendre la manière dont $\text{\LaTeX}$ génère les formules, il faut saisir les deux principes suivants :

Espaces : $\text{\LaTeX}$ ignore les espaces entre les symboles mathématiques ; ainsi : `\$x+1\$` produira la même formule que `\$x + 1\$`. C'est $\text{\LaTeX}$ qui insère les espaces à l'endroit qu'il juge que le plus judicieux ;

Texte⁵ : tout groupe de symboles est considéré comme un groupe de variables ou fonctions ; ainsi `\$x=t` avec `t>0\$` produira $x = t_{vect} > 0$ et non ce que vous espériez : $x = t$ avec $t > 0$.

Une fois ces deux principes acquis, voyons comment on peut faire avec.

3.5.1 Espaces en mode mathématique

Tout d'abord, sachez que $\text{\LaTeX}$ fait un choix d'espacement qui est en général correct. Cependant le jour où vous aurez à jouer l'~~exercice~~ de mouche, les commandes du tableau 3.2 vous permettront d'insérer un ou des espaces dans des formules. Dans ce tableau, on montre l'effet des commandes d'espacement entre deux symboles .

Pour ce qui concerne les mouches, sachez que l'auteur de ce manuel a sournoisement inséré un certain nombre d'espacements au numérateur du calcul de la somme des termes de la suite géométrique (§ 3.3.2 page 37), pour aligner les deux q de la fraction. Voici ce que donnait la formule par défaut :

$$\sum_{i=0}^n q^i = \frac{1 - q^{n+1}}{1 - q}$$

et voyons si les histoires de q vous donnent le sens de l'observation.

⁵L'insertion de texte dans une formule ne devient un problème que dans un environnement de la famille `\displaymath`, puisque vous pouvez toujours écrire «`\$x=t\$` avec `\$t>0\$`», bien sûr !

3.5.2 Texte en mode mathématique

Le moyen le plus simple d'insérer du texte dans une formule est de le mettre « en boîte » et d'insérer quelques espaces :

```
Soient les suites $(u_n)$ et $(v_n)$ :
\begin{displaymath}
  u_n = \ln n \quad
  \mbox{et} \quad v_n = (1 + \frac{1}{n})^n
\label{ex-maths-suite}
\end{displaymath}
```

Soient les suites (u_n) et (v_n) :

$$u_n = \ln n \quad \text{et} \quad v_n = (1 + \frac{1}{n})^n$$

Vous trouverez des détails sur la commande `\mbox` à la section 4.4.1 page 57. Si vous avez pensé à mettre en route le package `amsmath` vous serez en mesure d'utiliser la commande `\text` en lieu et place de `\mbox`.

3.6

Array : simple et efficace

L'environnement `array` est un environnement qui vous permettra de produire la grande majorité de vos formules. Comme son nom l'indique il range des objets en ligne et colonne. En fait c'est le pendant de l'environnement `tabular` du mode texte. Et comme `tabular`, `array` ne passe pas à la ligne.

3.6.1 Comment ça marche

La syntaxe rappelle celle de `tabular` :

```
\begin{array}[vpos]{format} ... \end{array}
```

où `format` précise pour chaque colonne l'alignement : `c` pour centré, `l` pour aligné à gauche et `r` pour aligné à droite ; l'argument optionnel `vpos` spécifie quant à lui le positionnement vertical du **tableau**. Comme dans les tableaux, on notera l'utilisation des commandes :

- `&` comme séparateur de colonne ;
- `\\` pour passer à la ligne.

```
Soit $A=\begin{array}{rc}
-1 & 1 \\
3 & 4
\end{array}$ la matrice ...
```

Soit $A = \begin{array}{cc} -1 & 1 \\ 3 & 4 \end{array}$ la matrice ...

Voici un exemple utilisant les points de suspensions :

```
\begin{displaymath} A = \left[ \begin{array}{ccc}
a_{00} & \dots & a_{0n} \\
\vdots & \ddots & \vdots \\
a_{n0} & \dots & a_{nn}
\end{array} \right] \end{displaymath}
```

$$A = \begin{array}{ccc} a_{00} & \dots & a_{0n} \\ \vdots & \ddots & \vdots \\ a_{n0} & \dots & a_{nn} \end{array}$$

3.6.2 Array et les délimiteurs

On utilise couramment l'environnement `array` pour produire des matrices. Il faut alors avoir recours à des *délimiteurs*. Ces délimiteurs sont de la famille des parenthèses et permettent d'englober un objet mathématique entre crochets, accolades, etc. La syntaxe est la suivante :

`\leftdelim1 mobjet \rightdelim2`

où `delim1` et `delim2` sont deux délimiteurs et `mobjet` un objet mathématique.

Parmi les délimiteurs, voici les plus usités :

(et)	(\Pi)	[et]	(\Pi)
\{ et \}	\{\Pi\}	\lfloor et \rfloor	(\Pi)
\lceil et \rceil	(\Pi)	\langle et \rangle	(\Pi)
	(\Pi)	\	(\Pi)

L'intérêt des délimiteurs est qu'ils s'adaptent automatiquement à la taille des objets qu'ils entourent :

soit `$I=`
`\left[\begin{array}{cc}`
`1&0 \quad 0&1`
`\end{array}\right]` la matrice identité.

soit $I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ la matrice identité.

On peut également reprendre l'exemple 3.13 page ci-contre avec des délimiteurs pour ajuster la taille des parenthèses :

Soient les suites `$(u_n)$` et `$(v_n)$` :

`\begin{displaymath}`
`u_n=\ln n\quad\mbox{et}`
`\quad v_n=\left(1+\frac{1}{n}\right)^n`
`\end{displaymath}`

Soient les suites (u_n) et (v_n) :

$$u_n = \ln n \quad \text{et} \quad v_n = 1 + \frac{1}{n}^n$$


Il doit toujours y avoir une commande `\right` pour une commande `\left`. Cependant, il n'est pas nécessaire d'avoir les mêmes symboles à droite et à gauche.

Voici un exemple où on utilise la commande `\right.` pour spécifier que l'on n'utilise pas de symbole à droite :

soit `$ S_i=\left\{\begin{array}{rl}`
`-1 & \mbox{si } \$i\$ est pair} \quad \backslash\backslash`
`1 & \mbox{sinon.}\end{array}\right.`

soit $S_i = \begin{cases} -1 & \text{si } i \text{ est pair} \\ 1 & \text{sinon.} \end{cases}$

3.6.3 Pour vous simplifier la vie...

Le package `amsmath` permet de saisir plus simplement les matrices avec notamment deux environnements : `pmatrix` (p pour parenthèse) et `bmatrix` (b pour *bracket*).

```
\begin{displaymath}
\bar{\bar{\sigma}}=\begin{bmatrix}
\sigma_{11} & \sigma_{12} \\
\sigma_{21} & \sigma_{22}
\end{bmatrix}
\end{displaymath}
```

3 19

$$= \begin{matrix} 11 & 12 \\ 21 & 22 \end{matrix}$$

3.7 Équations et environnements

Nous présenterons dans ce paragraphe trois environnements standard de L^AT_EX permettant de produire des formules.

3.7.1 L'environnement `displaymath`

Vous l'avez compris, si vous avez lu jusqu'ici, `displaymath` affiche une formule centrée, interrompant le paragraphe. Un raccourci agréable de :

```
\begin{displaymath}...\end{displaymath}
```

est : `\[...\]`. Ainsi :

```
Distance colorimétrique :\[
\Delta E=\sqrt{
\Delta L^{*2}+ \Delta a^{*2}+\Delta b^{*2}}
\]
```

Distance colorimétrique :

$$E = \sqrt{L^{*2} + a^{*2} + b^{*2}}$$

3 20

3.7.2 L'environnement `equation`

L'environnement `equation` est l'équivalent du précédent, sauf qu'il numérote la formule.

```
À retenir : si $a>0$ et $b>0$,\begin{equation}
\ln(ab)=\ln(a)+\ln(b)
\end{equation}
```

À retenir : si $a > 0$ et $b > 0$,

$$\ln(ab) = \ln(a) + \ln(b) \quad (3.1)$$

3 21



L'option de classe de document `leqno` met le numéro des équations à gauche. Et l'option `fleqn` aligne les équations à gauche, au lieu de les centrer.

3.7.3 Formules multi-lignes



Dans une précédente édition, nous finissions la présentation des environnements standard par l'environnement `eqnarray` qui permet de produire des formules de plusieurs lignes. **Sachez que c'est mal.** Il existe d'ailleurs des écrits à ce sujet (lire par exemple [5] ou [6]), vous expliquant comment produire des documents « propres ». Prenez bien conscience qu'utiliser `eqnarray` (et bien d'autres choses) **est un péché**, et que si vous cédez malgré tout à la tentation, l'inquisition vous retrouvera un jour ou l'autre par l'intermédiaire d'un moteur de recherche. Aucune confession ou indulgence ne pourra vous sortir de ce mauvais pas, vous être prévenu.

Nous vous présentons donc ici l'environnement `align` du package `amsmath` :

- `\` passe à la ligne ;

- chaque ligne est numérotée sauf si la commande `\nonumber` est présente dans la ligne ;
- on procède à l'alignement avec deux⁶ opérateurs `&`.

```
\begin{align}
(a+b)^2 &= (a+b)(a+b)\nonumber\\
&= a^2+b^2+2ab
\end{align}
```

$$(a+b)^2 = (a+b)(a+b) = a^2 + b^2 + 2ab \quad (3.2)$$



Il existe une forme « étoilée » de l'environnement : `align*` où aucune des lignes n'est numérotée. Si vous voulez faire référence à certaines lignes d'un `align`, il vous faudra poser autant de `\label` nécessaires sur chaque ligne correspondante.

Pour faire numéroté une équation s'étalant sur plusieurs lignes on peut utiliser l'environnement `split` (lui aussi fourni avec `amsmath`) :

```
\begin{equation}
\begin{split}
(a+b)^2 &= (a+b)(a+b)\\
&= a^2+b^2+2ab
\end{split}
\end{equation}
```

$$(a+b)^2 = (a+b)(a+b) = a^2 + b^2 + 2ab \quad (3.3)$$

3

3.8 Changer le style en mode mathématique

3.8.1 Fontes

$\text{\LaTeX}$ fournit plusieurs commandes permettant de changer de fontes dans les modes mathématiques. Par défaut tout symbole ou suite de caractères (autre qu'une fonction) est produit en italique dans le document final. Or dans certains cas, il est utile de pouvoir forcer le style de fonte. Voici comment réaliser un tel exploit :

Soit $\mathit{A \in \Phi}$	Soit $A \in \Phi$
Soit $\mathrm{A \in \Phi}$	Soit $A \in \Phi$
Soit $\mathbf{A \in \Phi}$	Soit $A \in \Phi$
Soit $\mathsf{A \in \Phi}$	Soit $A \in \Phi$
Soit $\mathtt{A \in \Phi}$	Soit $A \in \Phi$
Soit $\mathcal{A \in \Phi}$	Soit $\mathcal{A} \in \Phi$



La commande `\mathcal` doit prendre exclusivement des lettres majuscules latines comme argument. Dans le cas contraire, les résultats seront farfelus. Par exemple, la séquence :

```
\mathcal{abcd\Gamma}
```

donne $\mathcal{abcd\Gamma}$.

3.8.2 Taille des symboles

$\text{\LaTeX}$ distingue quatre *styles* d'écriture des formules. Ces modes sont utilisés suivant la « situation » dans laquelle se trouve $\text{\LaTeX}$ lorsqu'il produit une partie d'une formule :

⁶Puisqu'il y a trois colonnes.

texte pour une formule insérée dans le texte courant ;

equation pour une formule sous forme d'équation ;

indice pour l'écriture des indices ;

sous-indice pour les indices d'indices

chacun de ces modes peut être enclenché explicitement par l'utilisateur grâce aux déclarations suivantes :

- `\textstyle` pour le mode texte ;
- `\displaystyle` pour le mode équation ;
- `\scriptstyle` pour le mode indice ;
- `\scriptscriptstyle` pour le mode indice d'indice

Voici deux exemples illustrant comment forcer le mode *texte* en mode *équation* et inversement :

deux produits : `\prod_{1}^nf_i`
et `\displaystyle\prod_{1}^nf_i`

et inversement :

`\[ \prod_{1}^nf_i`
`\mbox{ et } \textstyle\prod_{1}^nf_i \]`

deux produits : $\prod_1^n f_i$ et $\prod_1^n f_i$
et inversement :
 $\prod_1^n f_i$ et $\prod_1^n f_i$

3.8.3 Créer de nouveaux opérateurs

Imaginez que vous ayez besoin de créer un opérateur spécial nommé « burps ». Il suffira de procéder comme suit :

`\newcommand{\burps}{%`
`\mathop{\texttrm{burps}}}`
`$x=\burps_i f(i)$`

$x = \text{burps}_i f(i)$

Un autre exemple, pour franciser la fonction « arcsinus » (produisant par défaut $\arcsin$), on pourra écrire :

`$\theta = \arcsin x$`
`\newcommand{\arcsin}{%`
`\mathop{\texttrm{Arcsin}}\nolimits}`
 `$\theta = \arcsin x$`

$= \arcsin x = \text{Arcsin } x$

La commande `\nolimits` indique que l'opérateur concerné ne fera pas usage d'arguments en indice ou exposant comme le font les opérateurs `\lim`, `\int`, etc. En outre les deux exemples précédents utilisent les commandes `\newcommand` et `\renewcommand` dont il est question au paragraphe 4.5 page 62.

Enfin une autre voie possible si vous avez pris soin de charger le paquetage `amsmath` est de déclarer dans le préambule :

`\DeclareMathOperator*\vlunch{\vlunch}`
`\DeclareMathOperator{\zirgl}{Zirgl}`

`\[x=\vlunch_i f(\theta)\]`
où `$\theta = \zirgl y$`

$x = \text{vlunch}_i f()$
où $= \text{Zirgl } y$

Conclusion

Ce chapitre présente les fonctions de base pour produire des formules. Ces commandes suffisent pour la plupart des documents scientifiques. Si vous êtes amenés à rédiger des documents truffés de formules complexes, il est possible que les seules macros de $\text{\LaTeX}$ ne suffisent plus. C'est pourquoi la célèbre *American Mathematical Society* a conçu pour vous un package nommé $\mathcal{A}\mathcal{M}\text{\TeX}$ (mise en route : `\usepackage{amsmath}`) capable de générer des formules particulièrement « tordues. »

4

Un pas vers la sorcellerie

Sommaire

- 4.1 Compteurs
- 4.2 Longueurs
- 4.3 Espaces
- 4.4 Boîtes
- 4.5 Définitions
- 4.6 Mais encore ?

*Et lorsque l'Agneau ouvrit le septième sceau
il se fit un silence dans le ciel,
environ une demi-heure...*

L'Apocalypse Ap 8 1.

A

TAB. 4.1 – Les compteurs de L^AT_EX

part	paragraph	figure	enumi
chapter	subparagraph	table	enumii
section	page	footnote	enumiii
subsection	equation	mpfootnote	enumiv
subsubsection			

sont associés. Les compteurs `enumi`, ..., `enumiv` sont associés aux items de niveaux 1 à 4 de l'environnement `enumerate`. Le compteur `mpfootnote` est le compteur de note de bas de page de l'environnement `minipage` dont il est question au paragraphe 4.4.3.

4.1.2 Manipulation

Nous vous donnons, dans les paragraphes qui suivent, les outils de base pour manipuler les compteurs. Il est important de noter que les compteurs sont des variables *globales*. Ainsi les trois commandes décrites plus bas ont une portée globale. Il est également utile de noter que ces variables sont des *entiers*.

Création

On peut *créer* un nouveau compteur grâce à la commande :

```
\newcounter{cpteur}[cpt_maitre]
```

qui crée un nouveau compteur `cpteur`. Si l'argument optionnel `cpt_maitre` est présent, le compteur `cpteur` est remis à zéro à chaque fois que le compteur maître `cpt_maitre` est incrémenté.

Affectation

On affecte une valeur à un compteur de la manière suivante :

```
\setcounter{compteur}{valeur}
```

où `compteur` est le compteur que l'on veut modifier, et `valeur` la valeur que l'on veut lui affecter.

Incrémentation

On peut incrémenter ou décrémenter un compteur grâce à la commande :

```
\addtocounter{compteur}{valeur}
```

où `valeur` est un nombre positif (resp. négatif) pour réaliser une incrémentation (resp. décrémentation). Illustrons l'utilisation de cette commande en entrant la ligne suivante dans notre document :

```
\addtocounter{footnote}{357}
```

pour changer³⁵⁹ la numérotation des notes de bas de page. Pour que tout rentre dans l'ordre, avec les notes de bas de page suivantes, nous avons préféré entrer dans notre source, la commande :

```
\addtocounter{footnote}{-357}
```

et normalement,³ nous devrions avoir une numérotation correcte.

4.1.3 Affichage

Pour afficher un compteur on utilise la commande :

```
\thenom-du-compteur
```

En fait, toute commande ou environnement qui donne lieu à l'affichage d'un compteur fait appel à ce type de commande. Ainsi, on a par exemple :

- \thepage produit : «49» et est appelée notamment à chaque saut de page,
- \thefootnote produit : «3» et est appelée par \footnote,
- \thesubsection produit : «4.1.3» et est appelée par \subsection,
- ...

Les commandes de la famille \the sont généralement définies à partir des commandes de formatage suivantes :

- \arabic{compteur},
- \roman{compteur} et \Roman{compteur},
- \alph{compteur} et \Alph{compteur}

en voici quelques exemples :

- \arabic{page} produit : «49»;
- \alph{footnote} produit : «c» et \Alph{section} produit : «A»;
- \Roman{subsection} produit : «III» et \roman{page} produit : «xlix»;
- ...

Il est courant de redéfinir les commandes de la famille \the pour personnaliser un document. Par exemple, dans la classe de document utilisée pour ce manuel, la commande \thefigure est définie comme suit :

```
\arabic{chapter}.\arabic{figure}
```

ce qui produit dans les légendes des figures une numérotation formée par : 1) le numéro du chapitre en chiffre arabe, 2) un point, et 3) le numéro de la figure en chiffre arabe. Il est possible de redéfinir cet affichage en définissant la commande \thefigure par exemple comme suit :

```
(\Roman{chapter}):\arabic{section}.\arabic{figure}
```

Ce qui permet d'obtenir un numéro de figure — relativement immonde — dans les légendes quelque peu différent du style prédéfini. Ici, on a donc redéfini la commande \thefigure pour produire une numérotation formée par le numéro du chapitre entre parenthèses et en chiffres romains, suivi du numéro de section et du numéro de la figure en chiffre arabe, séparés par un point. Le «FIG.» ainsi que le tiret qui



FIG. (IV):1.1 – La légende

³⁵⁹Même si ce changement est un peu ridicule...

³On croise les doigts!

suit le numéro de la figure sont quant à eux définis au niveau de la commande `\caption...`

4.2 Longueurs

Si les compteurs sont dédiés à la *numérotation* des objets d'un document, les longueurs définissent l'*encombrement* d'une entité. Il s'agit en quelque sorte, d'un type de donnée de $\text{\LaTeX}$ destiné à exprimer les dimensions d'un objet.

4.2.1 Unités

Toutes les dimensions doivent avoir une unité ; une dimension de type *rigide*⁴ a la forme suivante :

`nombre``unité`

où `nombre` est un nombre positif ou négatif avec éventuellement une partie décimale, et `unité` une unité de mesure reconnue par $\text{\LaTeX}$. Voici une liste non exhaustive des unités *légal*es :

cm pour *centimètre* ;

mm pour *millimètre* ;

in pour les *allerginch* au système métrique (environ 2.54cm) ;

pt pour *point* : couramment utilisé en typographie : $\frac{1}{72.27}$ inch ;

em : la largeur de la lettre 'M' de la fonte courante ;

ex : la hauteur de la lettre 'x' de la fonte courante

Notez que les unités em (resp. ex) sont généralement utilisées pour des dimensions horizontales (resp. verticales) et permettent de manipuler des dimensions dépendantes de la taille de la fonte courante. Voici quelques exemples de dimensions :

1cm	:	
1in	:	
3mm	:	
2em	:	
10pt	:	

4.2.2 Quelques longueurs de $\text{\LaTeX}$

Il existe dans $\text{\LaTeX}$ et dans chaque extension des longueurs prédéfinies. Ces longueurs déterminent en général, les dimensions de certaines parties du document. Ainsi :

- `\parindent` est la dimension de l'indentation en début de paragraphe. Cette dimension est prédéfinie à 15pt ;
- `\textwidth` et `\textheight` définissent la largeur (resp. la hauteur) du texte ;
- `\baselineskip` représente la distance entre la base de la ligne et la base de la ligne suivante (10pt dans ce document) ;

⁴On verra plus loin qu'il existe des dimensions *élastiques*.

- `\parskip` la distance séparant deux paragraphes ; cette distance est initialisée à 0pt plus 1pt⁵ ;
- ...

Il est important de comprendre qu'il est possible d'exprimer une dimension en fonction d'une de ces dimensions « internes ». Ainsi :

`0.5\textwidth`

représente la moitié de largeur de la page, et :

`3\parindent`

équivalent à trois fois l'indentation des paragraphes. Notez aussi que l'on peut écrire `-\baselineskip` pour : `-1\baselineskip`

4.2.3 Manipulation des longueurs

Comme pour les compteurs, il existe quelques commandes permettant de manipuler les dimensions.

Création

La commande suivante crée une nouvelle longueur :

`\newlength{dim}`

où `dim` est le nom de la nouvelle dimension initialisée à 0pt (cf. exemple page suivante).



Attention, quel que soit l'endroit où intervient la commande `\newlength`, la longueur définie est toujours **globale**. De plus, déclarer deux fois la même longueur provoque une erreur. Par contre, la modification d'une longueur est locale au groupe (`{...}`) où elle survient.

Affectation

On peut affecter une valeur à une longueur avec la commande :

`\setlength{dim}{val}`

qui affecte la valeur `val` à la longueur `dim`.

Incrémentation

On incrémente une longueur comme suit :

`\addtolength{dim}{val}`

qui a pour effet d'augmenter la longueur `dim` de la valeur `val`.

Alors que vous lisez fébrilement ce paragraphe, nous nous sommes permis d'augmenter la longueur `\parindent` de 30 points avec :

`\addtolength{\parindent}{30pt}` Alors que vous lisez fébrilement ce paragraphe...

pour illustrer l'utilisation de l'incrémentation des longueurs. Après ce paragraphe, on a écrit :

⁵Cf. les dimensions élastiques pour avoir la signification du plus.

```
\addtolength{\parindent}{-30pt}
```

pour que tout rentre dans l'ordre.

Obtenir les dimensions d'un objet

Comme il en a été vaguement question **précédemment**, au niveau de T_EX, les différents objets qui composent le document sont assemblés dans des *boîtes*. Ces boîtes sont positionnées les unes par rapport aux autres en alignant leur *point de référence*. Ces points alignés forment une ligne imaginaire confondue avec la base de la ligne. Toute boîte est caractérisée par trois dimensions :

- sa *largeur* ;
- sa *hauteur* : du point de référence au haut de la boîte ;
- sa *profondeur* : du point de référence jusqu'au bas de la boîte.

Voici par exemple comment sont assemblées les boîtes du mot « Ingénierie » :



les symboles «_o» représentent les points de référence. On voit ici que toutes les boîtes ont une profondeur nulle sauf celle de la lettre 'g'.

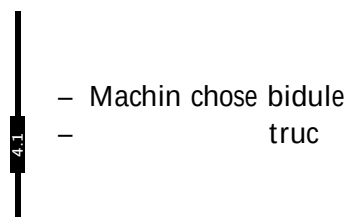
Mais fermons la parenthèse concernant les boîtes !

Il est donc possible d'extraire les caractéristiques d'un objet (une lettre, un mot, une boîte, etc.) à l'aide des commandes suivantes :

```
\settowidth{dim}{obj}
\settoheight{dim}{obj}
\settodepth{dim}{obj}
```

trois commandes qui affectent à la dimension *dim* respectivement la largeur, la hauteur et la profondeur de l'objet *obj*. Par exemple :

```
\newlength{\malongueur}
\settowidth{\malongueur}{Machin chose}
\begin{itemize}
\item Machin chose bidule
\item \hspace{\malongueur} truc
\end{itemize}
```



La longueur *\malongueur* contient alors la largeur de « Machin chose » et est utilisé pour insérer un blanc (voir le paragraphe sur les **espaces**).

4.2.4 Longueurs élastiques

Les dimensions présentées jusqu'ici sont des dimensions *rigides*,⁶ il existe cependant des longueurs *élastiques* ou *ressort*. Au niveau de T_EX, un grand nombre de dimensions sont définies comme suit :

val plus *p_val* minus *m_val*

⁶Sauf *\parskip*.

cette syntaxe permet de définir une longueur ayant la dimension `val`, mais pouvant selon les circonstances s'agrandir ou se rétracter. Et ainsi, si on appelle `dim` la dimension créée, on a :

$$val - m_val \ 6 \ dim \ 6 \ val + p_val$$

Par exemple, la longueur `\parskip` qui sépare deux paragraphes consécutifs, est fixée à :

0pt plus 1pt

ce qui signifie qu'au cas où la page est un peu lâche, L^AT_EX insérera entre les paragraphes un blanc vertical de 1 point. On ne peut pas directement⁷ créer de telles longueur avec L^AT_EX. Cependant ces dimensions ont surtout un intérêt lors de la création de macros complexes ou d'extensions. Par contre, les utilisateurs de L^AT_EX auront la chance inouïe de pouvoir manipuler une autre famille de longueurs élastiques tout aussi intéressante. Cette famille possède les deux particularités suivantes :

1. une longueur nulle ;
2. la capacité de s'étirer indéfiniment avec une certaine force.

L^AT_EX 2_ε dispose d'une commande permettant de spécifier une longueur élastique en précisant son degré d'élasticité :

`\stretch{nbre}`

où `nbre` est la force du ressort. Ce nombre peut être signé et avoir une partie décimale. Voici un exemple :

```
zéro\hspace{\stretch{1}}%          4.2  zéro          tiers          un
tiers\hspace{\stretch{2}}un
```

Ce code L^AT_EX introduit des espaces⁸ de longueurs élastiques entre les mots « zéro tiers un ». Le deuxième ressort a une raideur deux fois plus importante que le premier. L'espacement est donc double. Vous noterez aussi que ces ressorts ont une élasticité *relative* mais *infinie* ; c'est pourquoi les mots « zéro » et « un » sont « poussés » contre les marges. Enfin, sachez que `\fill` est un raccourci agréable de `\stretch{1}`.

4.2.5 Affichage

Il est parfois utile d'afficher la valeur d'une longueur. Pour ce faire on peut avoir recours à la commande `\showthe` qui interrompt la compilation pour afficher la valeur de la longueur passée en paramètre. Ainsi :

`\showthe\linewidth`

afficher la valeur de la longueur `\linewidth` en interrompant la **compilation**. On aura sur le terminal quelque chose du genre :

```
[ ... ]          ← Le laïus initial
Document Class: book 2001/04/21 v1.4 Standard LaTeX document class
(/usr/share/texmf/tex/latex/base/bk12.clo) (./test.aux)
```

⁷Pas avec la commande `\setlength`.

⁸C'est la commande `\hspace` qui produit une espace horizontale de longueur définie par son argument.

```
> 17.62482pt.          ← la valeur de la longueur
l.10 \showthe\parindent ← la longueur à ajouter

?
```

Lorsque la compilation est lancée dans un terminal de commande, une pression sur la touche <Entrée> fait reprendre la compilation.



Comme indiqué à la page 12, votre environnement de développement ne vous permet peut être pas directement d'avoir accès aux messages de L^AT_EX. À vous de chercher où se trouvent ces informations...

4.3 Espaces

On appelle *espaces* les blancs que l'on insère à divers endroits dans un document. Il existe des commandes permettant d'insérer des blancs de longueur prédéfinie ou choisie par l'utilisateur. Il s'agit bien sûr de longueur au sens de L^AT_EX.

4.3.1 Commandes de base

Pour insérer une espace⁹ entre les objets, on dispose de commandes de la forme suivante :

`\dirspace{dim}`

où `dim` est une longueur rigide ou élastique, et `dir` vaut :

- v pour une espace verticale ;
- h pour une espace horizontale.

Ainsi :

un saut\hspace{1cm}de \texttt{1cm}

\vspace{2\baselineskip}

et deux lignes vierges.

un saut de 1cm

et deux lignes vierges.



Dans certaines situations, T_EX supprime les espaces. Il est alors nécessaire d'utiliser la forme « étoilée » des commandes d'espacement, à savoir `\hspace*` et `\vspace*`.

Les situations en question sont :

- le début et la fin de page ;
- le début et la fin d'une ligne s'il ne s'agit pas de la première ou de la dernière ligne du paragraphe.

4.3.2 Quelques espaces prédéfinies

On dispose de plusieurs commandes d'espacement, regroupées en deux catégories selon le mode horizontal ou vertical.

⁹Nous utilisons ici le genre féminin du mot *espace* qui désigne alors les petites tiges métalliques utilisées autrefois en imprimerie pour séparer les mots et les lettres. Aujourd'hui, le genre féminin est utilisé dans le monde de la typographie et de la photo-composition.

Espaces horizontales

Voici quelques espaces rigides :

`\enspace` : □ soit 0.5\quad
`\quad` : □ soit 1em
`\qqquad` : □□□ soit 2\quad

et quelques espaces élastiques :

`\hfill` : soit `\hspace{\fill}`
`\hrulefill` : comme `\hfill` mais trace une ligne
`\dotfill` : comme `\hfill` mais trace des points

Voici quelques exemples montrant l'utilisation des espaces horizontales. Tout d'abord, notez que les espaces entourant la commande `\hspace` ne sont pas ignorées :

zéro\hspace{1cm}un\par		zéro	un
zéro \hspace{1cm}un\par		zéro	un
zéro \hspace{1cm} un\par		zéro	un

Voici ensuite, les espaces élastiques de L^AT_EX :

zéro \hfill{} un\par		zéro	un
zéro \hrulefill{} un\par		zéro	un
zéro \dotfill{} un\par		zéro	un

Et pour finir, la force relative des ressorts :

zéro \dotfill{} demi \hfill{} un\par		zéro	demi	un
zéro \hrulefill{} tiers		zéro	tiers	un
\hspace{\stretch{2}} un\par		zéro		

Vous aurez donc compris que les « ressorts » prédéfinis de L^AT_EX (à savoir `\hfill`, `\hrulefill`, et `\dotfill`) ont une raideur de 1.

Espaces verticales

Voici trois grands classiques de la famille des espaces verticales :

- `\smallskip` pour un *petit* saut vertical ;
- `\medskip` pour un saut vertical *moyen* ;
- `\bigskip` pour un *grand* saut vertical.

Ces espaces s'utilisent comme la commande `\vspace`, avec pour e_{ff}et :

défaut :	petite :	moyenne :	grande :
§ suivant...	§ suivant...	§ suivant...	§ suivant...

Il existe une espace verticale élastique prédéfinie : `\vfill` équivalent à :

`\par\vspace{\fill}`

c'est-à-dire, un saut de paragraphe, suivi d'une espace verticale de dimension `\fill`.

`\hrulefill{}`

haut

haut

`\vfill`

fragile

`\hfill{}fragile\hfill{}``\vspace{\stretch{2}}``\hfill{}bas`

bas

`\hrulefill{}`

Il est important d'utiliser la commande `\vspace` **entre deux paragraphes** au risque d'avoir des résultats surprenants. Il vaut donc mieux prendre l'habitude d'insérer un saut de paragraphe — une ligne vierge ou une commande `\par` — avant et/ou après `\vspace`.

4.4 Boîtes

La dernière section de ce chapitre sera dédiée aux *boîtes*, et vous verrez que le titre du présent chapitre sera amplement justifié ! Comme nous l'avons aperçu précédemment, les boîtes sont des entités qui contiennent d'autres éléments (une boîte pouvant en contenir une autre). Ces entités peuvent d'autre part être positionnées selon la fantaisie¹⁰ de l'utilisateur.

Il existe deux principaux types de boîtes (au niveau de T_EX c'est un peu plus subtil) chacun d'eux ayant un comportement spécifique. Nous qualifierons ces deux catégories comme suit :

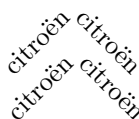
- boîte *simple*
- boîte *paragraphe*

Nous verrons qu'une manipulation habile de ces boîtes permet de produire des mises en page sophistiquées particulièrement utiles notamment lors de la conception de transparents.

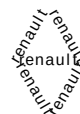
Voici un premier exemple avec des boîtes simples qui, j'en suis sûr, vous a sauté aux yeux, c'est le logo de T_EX : T_EX. Il s'agit en fait des trois lettres, T, E et X « mises en boîte » et assemblées avec des décalages horizontaux et verticaux :



notez que la boîte du 'E' est décalée vers le bas et que les trois boîtes se superposent. Un autre exemple :



et pour éviter les querelles culturelles :

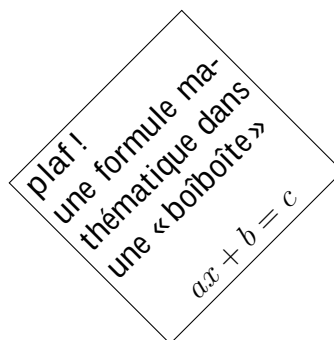


ici chaque mot est dans une boîte. Chaque boîte est ensuite placée par rapport aux autres avec moult décalages et rotations. Pour en finir avec les exemples préliminaires, nous vous donnons ici deux exemples utilisant les boîtes *paragraphes* :

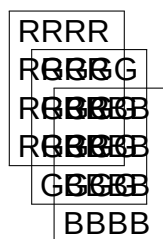
¹⁰Humm... ainsi qu'avec patience et bonne humeur...

Ceci est une boîte paragraphe de 4cm de large. Une telle boîte est capable de contenir à peu près tout ce qu'on trouve dans un document L^AT_EX.

le texte continue, et,



Voici une image en couleur, avec ses 3 composantes :



4.4.1 Boîtes simples

La première catégorie — les boîtes simples — se comporte comme des mots dans un paragraphe. Voici leurs caractéristiques :

- on peut imposer sa *largeur*,
- sa *hauteur* est donnée par ce qu'elle contient,
- Elle ne doit pas contenir de saut de paragraphe

Sans bordure La commande `\makebox` permet de construire une boîte simple.

`\makebox[larg][pos]{contenu}`

où `larg` est la largeur désirée, `pos` la position (c=centré, l=aligné à gauche ou r=à droite) de `contenu` dans la boîte. Voici quelques exemples :

et `\makebox[2cm][c]{hop !}` une boîte\par
et `\makebox[3cm][r]{rehop !}` une autre



et hop! une boîte
et rehop! une autre

Les deux arguments `larg` et `pos` sont optionnels et s'ils sont omis, la largeur de la boîte est celle du texte. Le cas échéant on saisit :

`\mbox{texte}`

au lieu de `\makebox[] [] {texte}`. On notera également que l'option `s` de la commande `\makebox` permet d'étirer le contenu pour qu'il fasse exactement la dimension imposée :

`\makebox[5cm][s]{Ouaaahhh quelle fatigue !}`



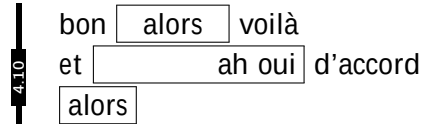
Ouaaahhh quelle fatigue!

Avec bordure On construit une boîte entourée par une bordure grâce à la commande `\framebox` qui suit la même syntaxe que `\makebox` :

`\framebox[larg][pos]{texte}`

le raccourci `\fbox{texte}` existe comme pour les boîtes sans bordure. Ce qui donne :

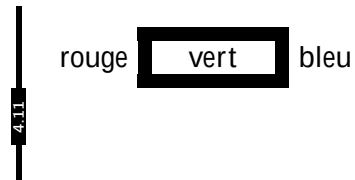
```
bon \framebox[1.5cm][c]{alors} voilà\par
et \framebox[2.8cm][r]{ah oui} d'accord\par
\fbox{alors}
```



Deux longueurs sont disponibles pour changer l'allure des `\framebox` :

- `\fboxsep` la distance entre la bordure et le texte,
- `\fboxrule` l'épaisseur du trait.

```
\setlength{\fboxrule}{5pt}
rouge \framebox[2cm]{vert} bleu\par
\setlength{\fboxrule}{0.4pt}
\setlength{\fboxsep}{8pt}
cyan \framebox[2cm]{magenta} jaune
```



```

\begin{flushleft}
  ligne 1 : XXXXX\\
  ligne 2 :                               ligne 1 : XXXXX
  XX\raisebox{0.8\baselineskip}{0}XX\\
  ligne 3 : XXXXX\\                      ligne 2 : XXOXX
  ligne 4 : XXXXX\\                      ligne 3 : XXXXX
  ligne 5 :                               ligne 4 : XXOXX
  XX%                                   ligne 5 : XXOXX
\raisebox{0.8\baselineskip}[1ex][2ex]{0}XX\\ ligne 6 : XXXXX
  ligne 6 : XXXXX\\
\end{flushleft}

```

On «soulève» un ‘O’ au milieu de la ligne 2. La bordure met en évidence la place occupée par la boîte soulevée.¹¹ Au milieu de la ligne 5, on soulève le même ‘O’ mais cette fois en imposant les dimensions (montrées par la bordure). L^AT_EX considère donc que la boîte résultant de la translation fait 1ex de haut et 2ex de profondeur, il effectue les sauts de lignes en conséquence.

Translation horizontale

Les translations horizontales ne sont pas à proprement parler des caractéristiques des boîtes, puisqu’on les obtient en insérant des **espaces** appropriées. Voici un exemple :

Non à la \makebox[1.5cm]{censure}%
 \hspace{-1.5cm}\makebox[1.5cm]{////////}
 sur Internet.

4.14

Non à la ~~censure~~ sur Internet.

Notez que ce n’est pas forcément la meilleure façon de «hachurer» un mot, mais cela illustre la manière de déplacer un boîte horizontalement, à l’aide d’un \hspace négatif.

Boîte simple de largeur nulle

Il est parfois utile de manipuler les boîtes de largeur nulle, par exemple dans le cas où l’on souhaite superposer des éléments. En imposant une dimension nulle en guise de premier argument optionnel de la commande \makebox :

\newcommand{\grogra}{\huge\bfseries}
 avant\makebox[0cm][c]{\grogra C}après
 avant\makebox[0cm][l]{\grogra G}après
 avant\makebox[0cm][r]{\grogra D}après

4.15

avant^Caprès
 avant^Gaprès
 avant^Daprès

on produit bien une superposition mais l’alignement n’est pas exactement celui auquel on s’attendait ; en effet l’argument *l* met le contenu à *droite* du point d’insertion de la boîte, et à *gauche* pour l’argument *r*.

¹¹Cette bordure est insérée ici pour la compréhension du mécanisme.

Rotation

Il existe plusieurs extensions de $\text{\LaTeX}$ pour faire subir des rotations à des éléments de texte ; nous avons choisi de vous présenter la commande `\rotatebox` de l'extension `graphicx` présentée au chapitre 5. La syntaxe en est la suivante :

`\rotatebox{angle}{texte}`

où `angle` est l'angle dans le sens trigonométrique, et `texte` l'élément de texte à faire tourner :

Attention `\rotatebox{30}{virage}` dangereux.



Attention `\virage` dangereux.



La version actuelle de `xdvi` n'est pas en mesure d'afficher les objets qui ont subi une rotation.¹² Cette lacune (avec quelques petites autres) sera peut-être corrigée dans les prochaines versions. La parade est de visualiser la sortie PostScript avec `ghostview` ou `gv`, ou la sortie Pdf.

4.4.3 Boîtes paragraphe

Les boîtes dites *boîtes paragraphe* ont la particularité de pouvoir contenir des sauts de ligne et des sauts de paragraphe (contrairement aux boîtes dites *simples*). Il existe deux manières de créer des boîtes paragraphe ; la première avec la commande `\parbox` :

`\parbox[bpos][hauteur][tpos]{largeur}{contenu}`

où `contenu` est l'élément de texte à mettre en boîte, `largeur` la largeur de la boîte à créer, `bpos` un argument optionnel précisant le point de référence. Cet argument optionnel est à rapprocher de celui de l'environnement `tabular`. Par exemple :

```
Voici --- \parbox{2.1cm}{une boîte\paragraphe}
--- une --- \parbox[t]{2.1cm}{autre boîte\paragraphe}
--- et --- \parbox[b]{2.1cm}{une boîte\paragraphe}
```

qui donne (avec des bordures pour mettre en évidence les dimensions des boîtes) :

Voici — une boîte
paragraphe — une — autre boîte
paragraphe — et — une boîte
paragraphe

Pour construire une boîte paragraphe en imposant sa hauteur, on utilise l'argument optionnel `hauteur`. On peut alors éventuellement préciser la position verticale `tpos` du texte dans la boîte. Par défaut `tpos` vaut `bpos`, et il peut prendre les valeurs habituelles `c` pour centré, `t` et `b` pour haut et bas ; plus une valeur : `s` pour spécifier que le texte peut s'étirer (`stretch`) sur toute la hauteur de la boîte — dans ce cas c'est à l'utilisateur de positionner le texte. Par exemple :

```
---\parbox[b][2cm]{2cm}{haut\par milieu\par bas}}
\parbox[b][2cm][t]{2cm}{haut\parmilieu\par bas}}
\parbox[b][2cm][c]{2cm}{haut\par milieu\par bas}}
\parbox[b][2cm][s]{2cm}{haut\par
\vspace{\stretch{2}} milieu\par\vfill bas}}---
```

¹²L'objet est affiché mais sans la rotation.

donne avec les `\fbox` pour y voir un peu plus clair :

haut milieu bas	haut milieu bas	haut milieu bas	haut milieu bas
-----------------------	-----------------------	-----------------------	-----------------------

Pour créer une boîte paragraphe, il peut être utile d'utiliser l'environnement `minipage`, qui simule la création d'une page avec d'éventuelles notes de bas de page, tableaux, listes, etc.¹³ La syntaxe est analogue à `\parbox` sauf qu'il s'agit d'un environnement :

```
\begin{minipage}[bpos][hauteur][tpos]{largeur}
... \marg{texte} ...
\end{minipage}
```

Voici un exemple :

Pour une raison x , y ou z , on peut vouloir raconter sa vie avec une `minipage`. Comme dans cet exemple là →

Ce que j'ai à dire^a n'est pas à franchement parler :

- ni très intéressant ;
 - ni particulièrement indispensable
- mais bon, j'en parle quand même.
J'avais autre chose à raconter mais ça m'est sorti de la tête...

^aC'est un bien grand mot.

Dans cet exemple on a créé une `minipage` faisant la moitié (55%) de la largeur du texte, et contenant un environnement `itemize` et une `\footnote`. La boîte ainsi créée est centrée par rapport au paragraphe « Pour une raison... » car l'argument optionnel `pos` est absent :

```
\parbox{0.40\textwidth}{...
  Comme dans cet exemple là $\longrightarrow$\hfill
\begin{minipage}{0.55\textwidth}
  Ce que j'ai à dire\footnote{C'est un bien grand mot.} n'est pas à
  franchement parler :
  \begin{itemize}
    \item ni très intéressant ;
    \item ni particulièrement indispensable
  \end{itemize}
  mais bon, j'en parle quand même.

  J'avais autre chose à raconter mais ça m'est sorti de la tête...
\end{minipage}
```



Dans les boîtes paragraphe la longueur `\parindent` est mise à zéro. Ce qui explique que « J'avais autre chose ... » dans l'exemple précédent n'est pas indenté. Enfin, contrairement aux cas des `\parbox`s, lorsqu'on fait référence dans une `minipage` à la dimension `\textwidth`, il s'agit de celle de la boîte et non de celle du texte.

¹³Cet environnement ne peut toutefois pas contenir de flottants.

4.4.4 Petites astuces

Toutes les fonctions concernant les boîtes peuvent prendre en paramètre de longueur les dimensions suivantes :

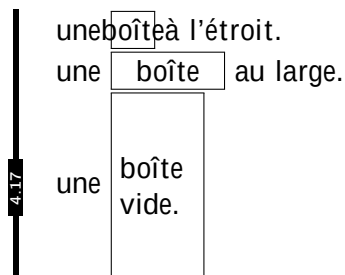
- `\width` : la largeur du texte contenu,
- `\height` : la hauteur du texte contenu,
- `\depth` : la profondeur du texte contenu,
- `\totalheight` (hauteur + profondeur) du texte.

Il est alors possible de préciser les dimensions de la boîte relativement au texte qu'elle contient. Ce qui peut être utile dans certaines situations :

une `\framebox[0.7\width]{boîte}` à l'étroit.

une `\framebox[1.8\width]{boîte}` au large.

une `\fbox{%
 \parbox[c][3\height]{1cm}{boîte\\vide.}}`



4

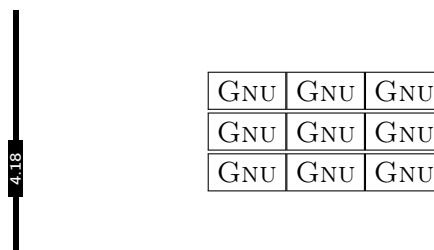
4.4.5 Sauvegarde et réutilisation

Il est possible de stocker un extrait de code $\text{\LaTeX}$ dans une boîte pour le réutiliser — ceci par exemple lorsque ce code exige de $\text{\LaTeX}$ des ressources importantes ; dans ce cas on procède en 3 étapes :

1. déclaration d'une boîte avec la commande `\newsavebox`,
2. stockage avec `\sbox` ou `\savebox`,
3. réutilisation avec `\usebox`.

Par exemple voici une texture de Gnu :

```
\newsavebox{\gnu}
\sbox{\gnu}{\fbox{\textsc{Gnu}}}\
\begin{center}
  \usebox{\gnu}\usebox{\gnu}\usebox{\gnu}\\
  \usebox{\gnu}\usebox{\gnu}\usebox{\gnu}\\
  \usebox{\gnu}\usebox{\gnu}\usebox{\gnu}
\end{center}
```



On peut faire une analogie entre le couple de commande `\sbox` et `\savebox` et le couple `\mbox` et `\makebox` (cf. § 4.4.1).

4.5 Définitions

Une nouvelle fois, laissons parler le maître :

«... they have come to be known as macros because they are so powerful; one little macro can represent an enormous amount of material, so it has a sort of macroscopic effect.»

D. E. Knuth in the $\text{\TeX}$ book

Lorsque dans un document, on peut définir une «entité» indépendante et que cette entité apparaît plus d'un «certain nombre de fois» il est nécessaire de se poser la question de savoir s'il n'est pas judicieux d'en faire une *macro*. Voilà une phrase vague ! Pour résumer, les macros sont là pour vous éviter de refaire x fois les mêmes choses. Avec un peu d'expérience, on peut définir des commandes très pratiques et avec le temps de plus en plus complexes.

4.5.1 Commandes

La commande `\newcommand` permet de définir une macro, son utilisation est très simple :

`\newcommand{nomcom}[nargs]{code LATEX}`

où **nargs** est le nombre d'*arguments* — au sens arguments d'une fonction d'un langage de programmation — et **code L^AT_EX** le code définissant votre commande. Voici un exemple de macro, définissant le symbole d'un espace de représentation utilisé en colorimétrie :

`\newcommand{\lab}{ $L^*a^*b^*$ }`

Soit `\lab{}` l'espace...

4.19

Soit $L^*a^*b^*$ l'espace...

Notez que cette commande ne prend pas d'argument, il n'est donc pas nécessaire ici d'utiliser l'argument optionnel **nargs**. Pour améliorer un peu l'utilisation de cette commande, on peut la définir comme suit :

`\newcommand{\Lab}{%`

`\ensuremath{L^*a^*b^*}}`

`L'espace \Lab{ } et  $\vec{c} \in \Lab{}$ .`

4.20

L'espace $L^*a^*b^*$ et $e \in L^*a^*b^*$.

La commande `\ensuremath` permet de s'assurer que la commande sera utilisée dans un environnement mathématique, quel que soit le contexte, comme dans l'exemple ci-dessous.



Les macros ou commandes de L^AT_EX ne sont pas tout à fait des fonctions au sens d'un langage de programmation, elles s'apparentent plutôt au `#define` du C. Et en ce sens, elles suivent le mécanisme d'*expansion*. Ainsi, dans le premier exemple de la fonction `\Lab`, `\Lab` se « déploie » en $L^*a^*b^*$. On comprend donc pourquoi, `$... \Lab$` aurait généré une erreur de compilation.

Voici une commande utilisant un argument : elle permet de dessiner une touche de clavier :¹⁴

`\newcommand{\Touche}[1]{\Ovalbox{#1}}`

Appuyer sur `\Touche{Tab}`
puis sur `\Touche{Entrée}`

4.21

Appuyer sur Tab puis sur Entrée

On voit donc que cette commande attend *un* argument (c'est le sens de « `[1]` ») et qu'on fait référence à cet argument dans la définition de la commande avec `#1`.

Si l'on souhaite définir une fonction avec plusieurs arguments (9 au maximum), *no problemo* :

¹⁴Cette commande fait appel à la commande `\Ovalbox` définie dans le package `fancybox`.

```
\newcommand{\fraction}[2]{%
  \raisebox{0.5ex}{#1}%
  \slash\raisebox{-0.5ex}{#2}}
\fraction{1}{2} et \fraction{3}{4} font
\fraction{5}{4}
```

1/2 et 3/4 font 5/4

On remarquera donc que :

- la macro `\fraction` prend 2 arguments,
- on fait référence au n^{e} argument avec `#n`,
- les caractères % s'il vous paraissent saugrenus, permettent d'insérer des sauts de ligne dans le code sans insérer d'espace dans le document (voir aussi le paragraphe 9.2.1 page 107 à ce sujet).

Il est également possible de définir une commande dont le premier argument est optionnel. La syntaxe est alors la suivante :

```
\newcommand{\nomcom}[narg][arg default]{code LATEX}
```

où `narg` est le nombre d'arguments, sachant que `#1` sera l'argument par défaut, `arg default` est la valeur que prend `#1` par défaut, et `code LATEX` le code de la commande. Voici par exemple une autre approche de la commande vue précédemment, qui dessine une touche de clavier :

```
\newcommand{\Touche}[1][Entrée]{\ovalbox{#1}}
Appuyer sur \Touche[Tab]{} puis sur \Touche{}
```

Appuyer sur Tab puis sur Entrée

On voit donc que l'argument 1 est facultatif et sa valeur par défaut est : « Entrée ». On notera également que l'utilisation de l'argument optionnel requiert des crochets et non des accolades.



On peut très bien imaginer que l'on ait à définir une commande ayant un argument optionnel et un ou plusieurs arguments obligatoires. Dans ce cas le premier argument obligatoire sera `#2`. D'autre part, notez qu'on ne peut rendre optionnel que le **premier argument** d'une commande.

4.5.2 Environnement

Il est possible de définir ses propres environnements de la manière suivante :

```
\newenvironment{\nom env}[narg]{clause begin}{clause end}
```

où `nom env` est le nom de l'environnement ainsi défini, `narg` le nombre d'arguments, et `clause begin` et `clause end` les « pré » et « post » traitements de l'environnement. Il est pratique de définir des environnements à partir d'autres, par exemple les environnements de L^AT_EX :

```
\newenvironment{bonmot}%
{\small\slshape\begin{flushright}}%
{\end{flushright}\normalsize\upshape}
\begin{bonmot}
  L'homme a reçu de la nature une clef\
  avec laquelle il remonte la femme\
  toutes les vingt-quatre heures.
\end{bonmot}
```

*L'homme a reçu de la nature une clef
avec laquelle il remonte la femme
toutes les vingt-quatre heures.*

ensemble de commandes. $\text{\LaTeX}$ cherchera alors le fichier `moncru.sty`. Une autre solution est d'utiliser la commande `\input{moncru.sty}`.

Dans la plupart des distributions de $\text{\LaTeX}$, un fichier nommé `texmf.cnf` définit un certain nombre de paramètres permettant de configurer le moteur $\text{\LaTeX}$ et notamment les chemins de recherche des fichiers. Sur mon système ($\text{\TeX}$ sous Debian) ce fichier contient notamment :

```
HOMETEXMF = $HOME/texmf
```

indiquant que le système cherchera les fichiers à inclure dans le répertoire `texmf` de votre répertoire privé. Vous devrez alors placer vos extensions «maison» dans le répertoire `~/texmf/tex/latex`.

Un dernier conseil : pour pouvoir définir vos commandes ou environnements de manière plus confortable, nous vous recommandons de jeter un petit coup d'œil sur :

- l'extension `ifthen` qui propose des structures de contrôle de type «si-alors-sinon» et «faire-tant-que»,
- le package `calc` qui permet d'effectuer des opérations arithmétiques sur les compteurs et les longueurs.
- enfin l'environnement `list` qui peut être un bon point de départ pour se définir un environnement de type liste.

Ces extensions et leur utilisation sont présentées en détail dans la deuxième partie de ce manuel.

5

Graphisme

Sommaire

- 5.1 Apéritifs
- 5.2 Du format des fichiers graphiques
- 5.3 Le package graphicx
- 5.4 Quelques extensions utiles
- 5.5 Utiliser make
- 5.6 À part ça

*Tu ne te feras aucune image sculptée de rien
qui ressemble à ce qui est dans les cieux là-haut [...]
Tu ne te prosterner pas devant ces images ni ne les serviras.*

Le Deutéronome Dt 5 8.

AUJOURD'HUI il est tout à fait naturel d'insérer des dessins, figures et autres images dans un document. Ceci est dû aux imprimantes de plus en plus performantes et bon marché. Il faut cependant se replacer dans le contexte des années 80 à l'essor de T_EX. C'est l'époque de l'apparition des imprimantes et le matériel de qualité professionnelle n'était pas accessible au particulier. Cependant beaucoup de solutions d'impression émergeront s'appuyant la plupart sur le langage PostScript devenu *ipso facto* un standard.

Il existe plusieurs solutions autour de L^AT_EX pour insérer des graphiques dans un document. Parmi elles on notera l'utilisation de metafont (l'utilitaire qui gère les fontes de L^AT_EX), la programmation d'un environnement picture ou la mise en œuvre d'un code PICT_EX. Ces solutions ne seront pas décrites ici car nous considérons qu'elles sont d'une utilisation un peu déroutante au premier abord ; il est tout de même bon de connaître leur existence. L'approche adoptée dans ce manuel pour manipuler des graphiques est d'insérer dans le source L^AT_EX un fichier au format PostScript encapsulé contenant le graphique en question, ce dernier ayant été créé par un logiciel de dessin tel que xfig, gnuplot, gimp, etc.

5.1 Apéritifs

Il n'est pas inutile de connaître la commande `\rule` qui permet de faire des traits :

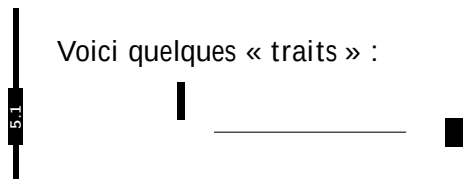
```
\rule[hpos]{largeur}{hauteur}
```

où `hpos` impose une éventuelle translation verticale du trait, les deux autres arguments ont un nom suffisamment explicite :

Voici quelques \og traits \fg{} :

```
\begin{center}
  \rule[1ex]{1mm}{5mm}\quad\rule{1in}{0.4pt}
  \quad\rule[-0.5em]{1em}{1em}
\end{center}
```

Voici quelques « traits » :



5.2 Du format des fichiers graphiques

Pour inclure des dessins ou des images dans vos documents, il faut insérer un *fichier*. La configuration de $\text{\LaTeX}$ permet d'incorporer des fichiers de type PS pour PostScript et EPS pour Encapsuled PostScript. Ce fichier peut être généré par n'importe quel programme. Si le format PostScript vous semble contraignant, sachez que :

- tout *bon* logiciel de dessin « vectoriel » vous permet d'exporter vos schémas au format EPS. Ce format est devenu la référence en matière d'impression.
- toute image peut être convertie au format EPS. Sur un système UNIX, le programme `convert` permet d'effectuer cette opération.¹ On pourra aussi recourir au logiciel `gimp` (logiciel libre de retouche et de création d'image numérique), présent également sur d'autres systèmes d'exploitation.

2

5.3 Le package `graphicx`

$\text{\LaTeX}$, ou plutôt $\text{\TeX}$, n'a pas été initialement conçu pour manipuler des graphiques (images, dessins,...). De ce fait, une multitude d'extensions ont été proposées, aucune n'ayant vraiment réussi à s'imposer ou à être vraiment indépendante des systèmes d'exploitation.

5.3.1 Un standard

Aujourd'hui, les concepteurs de $\text{\LaTeX}$ semblent s'être mis d'accord pour standardiser une extension *graphique*. Deux extensions ont donc vu le jour à fin de l'année 1994 :

- `graphics` l'extension « standard » ;
- `graphicx` l'extension « plus plus ».

Nous avons choisi de vous présenter `graphicx`. Il faut bien comprendre que si l'interface de cette extension est indépendante du système d'exploitation, la partie du code gérant les différents types de fichiers graphiques est dépendante du système sous-jacent. Aussi, est-il nécessaire de préciser un *driver* de package. Les drivers existants correspondent aux implantations connues de $\text{\TeX}$ sur des plate-formes diverses.² Sous UNIX, le driver utilisé est généralement `dvips`, il est choisi par défaut dans la distribution `teTeX`, si bien que la ligne :

```
\usepackage{graphicx}
```

¹ Cherchez du côté de la suite « ImageMagick » pour obtenir ce programme s'il n'est pas présent sur votre système.

² On notera entre autres : `xdvi` et `dvips` pour le monde UNIX, `texture` et `OzTeX` pour le Mac, `emTeX` et `dviwin` pour windows.

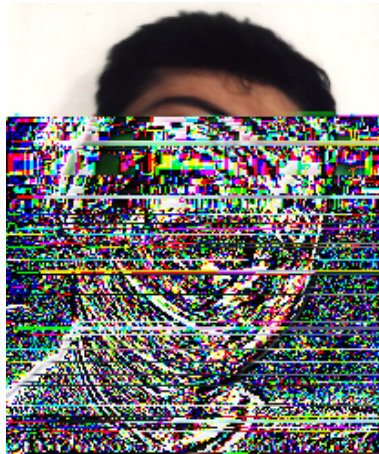


FIG. 5.1 – Robert (après quelques bières).

suivant pour mettre en route l'extension graphicx. La commande pour inclure un dessin ou une figure est la suivante :

```
\includegraphics[option]{fichier}
```

où **fichier** est un fichier contenant votre figure, et **option** est une liste d'options séparées par des virgules. La commande `\includegraphics` ne crée pas de mise en page particulière, elle insère juste une boîte contenant le graphique dans le texte. Ainsi :

avant

```
\includegraphics{punch}
```

et après.

5.2

avant  et après.



Pour assurer la portabilité de vos sources et ainsi pouvoir insérer des fichiers graphiques dans des formats différents, il est impératif de *ne pas préciser l'extension du fichier dans la commande `\includegraphics`*.

En général, on combine `\includegraphics` avec un environnement **figure**. Par exemple, la figure 5.1 a été produite grâce au code suivant :

```
\begin{figure}
  \centering\includegraphics[width=5cm]{punch}
  \caption{Robert (après quelques bières).}
  \label{fig-exemple}
\end{figure}
```



Notez bien que l'environnement **figure** assure que le graphisme « flotte » dans la page et que ça n'est pas la commande `\includegraphics` qui assure ce rôle. Au cas où ça aurait échappé à certains :

L'environnement `figure` assure que le graphisme « flotte » dans la page ; ça n'est pas la commande `\includegraphics` qui assure ce rôle.

5.3.2 Options

Le package **graphicx** possède plusieurs options permettant de contrôler l'insertion des graphiques. Parmi les options disponibles voici les plus utilisées :

Changement d'échelle

Il existe trois manières d'agir sur la taille d'un graphique.

- `scale=ratio`, où `ratio` est un nombre positif ou négatif, permet de changer la taille globale de la figure ;
- `width=dimen` permet d'imposer la largeur du graphique ;
- `height=dimen` permet d'imposer la hauteur du graphique.

```
\begin{center}
\includegraphics[scale=0.2]{magma}
\includegraphics[width=8.5mm]{magma}
\includegraphics[width=2cm,height=3mm]{magma}
\end{center}
```



Rotation

Vous pouvez si vous le désirez, faire effectuer une rotation à votre figure en utilisant l'option `angle`, dont la syntaxe est la suivante :

`angle=ndegre`

où `ndegre` est un angle précisé en degrés dans le sens trigonométrique.

```
\includegraphics[angle=45,scale=0.2]{magma}
```



On trouvera dans le fichier `grfguide.pdf`³ une description détaillée de cette extension. On pourra également consulter le fichier `fepslatex.pdf`⁴.

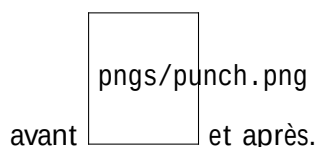
Mode brouillon

L'option `draft` permet de produire les figures en mode « brouillon » : seul un cadre avec le nom du fichier inclus est produit dans le document final.

avant

```
\includegraphics[draft,scale=.2]{punch}
```


 et après.



Le mode `draft` est enclenché par défaut lorsque l'option de document `draft` est spécifiée. Si vous voulez contrer l'effet de l'option de document⁵, il est possible d'utiliser l'option `final` de la commande `\includegraphics` ou au moment d'inclure l'extension avec `\usepackage`.

³Utiliser `locate` ou `find` pour le trouver sur votre système.

⁴<http://tug.ctan.org/tex-archive/info/epslatex/french/fepslatex.pdf>

⁵Par exemple, si vous voulez voir vos figures mais aussi les « OverfullBoxMark. »

5.4 Quelques extensions utiles

Voici dans les paragraphes qui suivent trois extensions utiles pour la production de documents contenant des graphiques.

5.4.1 subfig

Cette extension permet de gérer des figures comportant plusieurs sous-figures, avec numérotation automatique et possibilité de faire référence aux sous-figures elles-mêmes. Par exemple :



FIG. 5.2 – Uniweria Zëkt

```
\begin{figure}[htbp]
  \begin{center}
    \leavevmode
    \subfloat[Magma]{%
      \label{fig-uniweria-magma}
      \includegraphics[width=2cm]{magma}}
    \hspace{2cm}
    \subfloat[UZMK]{%
      \label{fig-uniweria-uzmk}
      \includegraphics[height=2cm]{uzmk}}
    \caption{Uniweria Zëkt}
    \label{fig-uniweria}
  \end{center}
\end{figure}
```

Pour ce qui concerne les références, on peut soit référencer la figure globale par `\ref{fig-uniweria}` qui donne : 5.2, soit les sous-figures par leur label respectif : `\ref{fig-uniweria-magma}` et `\ref{fig-uniweria-uzmk}` qui donnent : 5.2a et 5.2b.



Une manière élégante de gérer les `\subfigures` est d'encapsuler chacune d'elles dans un environnement `minipage`. On trouvera dans le fichier `subfig.pdf` accompagnant la distribution, comment personnaliser l'environnement `subfigure`, notamment les espaces inter-légendes.

5.4.2 Le package wrapfig

Le package `wrapfig` propose l'environnement `wrapfigure` permettant de faire flotter une figure dans un paragraphe. Il ne s'agit pas d'un environnement flottant au sens de l'environnement `figure` de $\text{\LaTeX}$ puisqu'on spécifie la position de la figure dans le paragraphe. La syntaxe est la suivante :

```
\begin{wrapfigure}{position}{largeur}
```

```
...
```

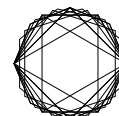
```
\end{wrapfigure}
```

où *position* est la position de la figure (l ou r) et *largeur* la largeur de la figure à insérer. Voici un exemple :

```
\begin{wrapfigure}{r}{1.5cm}
\includegraphics[width=1cm]{polygons}
\end{wrapfigure}
```

Le package `\ltxcom{wrapfig}` n'est ---à ma connaissance--- pas documenté sous la forme d'un fichier `\texttt{dvi}` ; par contre il est possible...

Le package `\wrapfig` n'est — à ma connaissance — pas documenté sous la forme d'un fichier `dvi` ; par contre il est possible de trouver des informations très détaillées dans le fichier `.sty` lui-même qui se trouve dans l'arborescence `TEX` dans : [...] /misc/wrapfig.sty. On notera au passage — car il faut parler pour faire un paragraphe un peu long — que la règle veut que tout package soit « auto-documenté » grâce à une extension connue sous le nom de `docstrip`. Ainsi toute extension — *package* en anglais — contient aussi bien le code que la documentation. Une procédure d'installation permet d'extraire l'un et l'autre. L'auteur de `wrapfig` n'a vraisemblablement pas suivi cette règle, tant pis...



5.4.3 Le package `psfrag`

Une autre extension intéressante est l'extension `psfrag`. Elle a pour but de pouvoir réunir la puissance d'un fichier PostScript et la beauté des équations de `LATEX`. Un problème se pose en effet et lorsque l'on veut intégrer des formules à un dessin, car la génération d'équations n'est pas prévue dans la plupart de ces logiciels. La solution adoptée par les auteurs de `psfrag` est d'utiliser la commande `\psfrag` pour insérer les formules à la place de chaînes de caractères présentes dans le dessin. Ainsi, pour avoir la figure 5.3b au lieu de la figure 5.3a, on a procédé comme suit :

1. ajout avant le `\includegraphics{courbe}` la ligne :

```
\psfrag{exp(-x)*sin(10*x)}[r][r]{$e^{-x}\sin(10x)$}
```

qui permet de remplacer la chaîne de caractères faisant office de légende par une belle équation ;

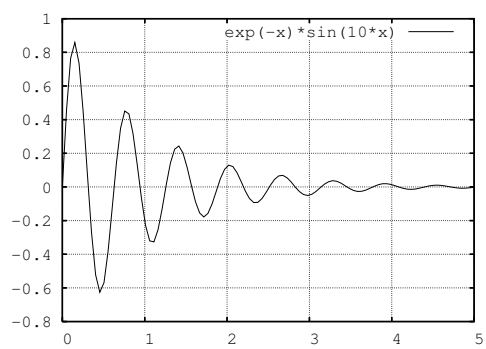
2. le résultat n'est pas visible dans le fichier `.dvi`, par contre `dvips` se charge d'exploiter les instructions précédentes pour modifier le fichier PostScript généré.

Le positionnement de la formule se fait en faisant correspondre deux points de référence, l'un appartenant à l'équation, l'autre à la chaîne de caractères à remplacer. C'est à l'utilisateur d'indiquer où se trouve ces points de référence, par l'intermédiaire de deux arguments optionnels à la commande `\psfrag`. Supposons qu'on définisse ces points de référence comme suit :

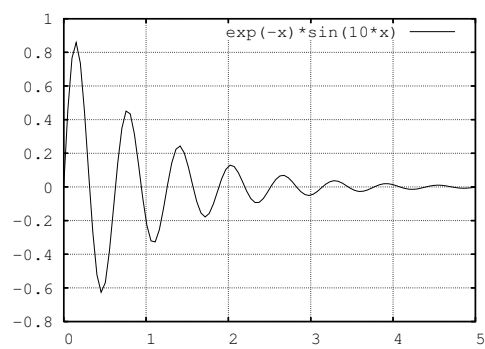
```
\psfrag{chaîne}[l][c]{équation}
```

On aura alors l'assemblage suivant :





(a) avant



(b) après

FIG.

- sans bordure :

```
\colorbox{couleur du fond}{contenu}
```

- avec bordure :

```
\fcolorbox{couleur bordure}{couleur fond}{contenu}
```

Les deux commandes pour les boîtes en couleur sont sensibles à la longueur \fboxsep. « *Quid* des couleurs qui n'ont pas de nom ? » vous entends-je marmonner *in petto*... Ce à quoi je réponds sur le champ :

Voici un `{\color[rgb]{.2,.4,.5}\bfseries bleu gris}`...  Voici un **bleu gris**...

Il est aussi possible de donner un « petit nom » à cette dernière couleur :

`\definecolor{bleugris}{rgb}{.2,.4,.5}`
Voici un `{\color{bleugris}\bfseries bleu gris}`...  Voici un **bleu gris**...



Vous noterez qu'en lieu et place du modèle de couleur « rgb » il est possible d'utiliser le modèle gray de manière à définir des nuances de gris. De même, en utilisant le modèle html, on pourra utiliser la syntaxe du langage Html pour spécifier les couleurs.

5.5 Utiliser make



Ce paragraphe est destiné aux utilisateurs d'un système d'exploitation disposant de l'utilitaire Gnu make (pour de plus amples informations sur cet utilitaire, n'hésitez pas lire à l'incontournable [7]). Les autres peuvent passer leur chemin...

Voici donc une idée de makefile qui vous permettra d'automatiser la génération :

- des fichiers au format Eps à partir des images « bitmaps » ;
- des fichiers au format Eps à partir de fichiers stockés dans le format d'un logiciel de dessin vectoriel.

On souhaite pour ce faire, élaborer une cible que l'on précisera en ligne de commande :

```
make figs
```

On suppose que les images et les fichiers de dessins sont respectivement stockés dans les sous-répertoire Imgs et Figs du répertoire contenant le document maître et que les fichiers Eps fabriqués seront également stockés dans un sous-répertoire Epss (voir la figure 5.4 page ci-contre). On commencera donc par définir un ensemble de variables précisant les différents répertoires à utiliser :

```
FIGSDIR=Figs
EPSSDIR=Epss
IMGSDIR=Imgs
```

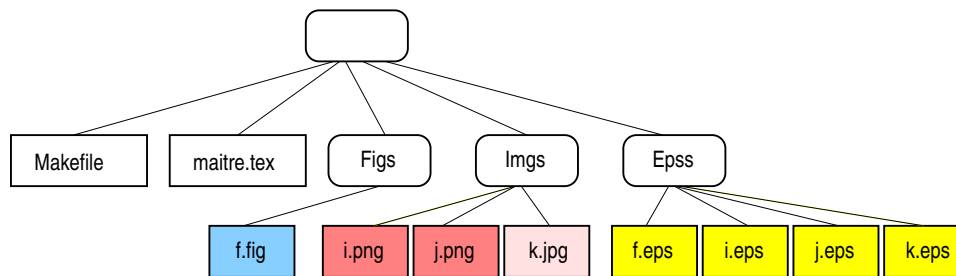


FIG. 5.4 – Proposition d'arborescence pour stocker les fichiers graphiques : un répertoire pour les images et un répertoire pour les graphiques vectoriels. Les fichiers au format Eps sont enregistrés dans un répertoire à part.

5.5.1 Convertir les images

Tout d'abord on fait la liste des fichiers au format Jpeg et Png (c'est un exemple) en stockant cette liste dans deux variables comme suit :

```
PNGS=$(notdir $(wildcard $(IMGSDIR)/*.png))
JPGS=$(notdir $(wildcard $(IMGSDIR)/*.jpg))
```

La fonction `wildcard` permet d'obtenir la liste des fichiers du répertoire `$(IMGSDIR)` tandis que la fonction `notdir` supprime la partie « répertoire » de chacun des fichiers. Finalement la variable `PNGS` contiendra :

i.png j.png

et `JPGS` :

k.jpg

On peut ensuite à partir de ces deux variables créer la liste des fichiers Eps à fabriquer (rappelez-vous qu'ils doivent résider dans un répertoire à part) :

```
IMG2EPSS=$(patsubst %, $(EPSSDIR)/%, \
$(PNGS:.png=.eps) $(JPGS:.jpg=.eps))
```

L'expression à droite de l'affectation permet de changer l'extension en eps dans les deux listes précédentes et de préfixer chaque nom par le répertoire de stockage des fichiers Eps. `IMG2EPSS` contiendra donc :

Epss/i.eps Epss/j.eps Epss/k.eps

Cette liste constitue les «prérequis» (au sens de make) pour fabriquer les images. On définit donc la cible `figs` comme suit :

```
figs : $(IMG2EPSS)
```

Il faudra également expliquer à make comment on peut fabriquer un fichier au format postscript encapsulé à partir d'une image. Ceci pourra s'écrire à l'aide de la règle suivante :

```
$(EPSSDIR)/%.eps : $(IMGSDIR)/%.png
→ convert $< EPS:$@
$(EPSSDIR)/%.eps : $(IMGSDIR)/%.jpg
→ convert $< EPS:$@
```

qui précise qu'on utilisera l'utilitaire convert⁶ pour convertir un fichier Png ou Jpeg en Eps.

5.5.2 Convertir les fichiers de dessin

La conversion des fichiers de dessins suit exactement le même principe. Supposons qu'on dispose de sources au format Fig provenant de xfig et au format Svg provenant de Inkscape. On aura alors dans le makefile :

```
FIGS=$(notdir $(wildcard $(FIGSDIR)/*.fig))
SVGS=$(notdir $(wildcard $(FIGSDIR)/*.svg))
FIGS2EPSS=$(patsubst %, $(EPSSDIR)/%, \
    $(FIGS:.fig=.eps) $(SVGS:.svg=.eps))
```

Les règles de conversion sont bien évidemment différentes puisqu'elles font appel l'une à fig2dev (utilitaire connexe à xfig) et à Inkscape lui-même pour l'autre :

```
$(EPSSDIR)/%.eps : $(FIGSDIR)/%.fig
    ↳ fig2dev -L eps $< > $@
$(EPSSDIR)/%.eps : $(FIGSDIR)/%.svg
    ↳ inkscape -E $@ $<
```

La cible permettant de fabriquer les fichiers de dessins et les images devient finalement :

```
figs : $(IMGS2EPSS) $(FIGS2EPSS)
```

5.6 À part ça

On peut trouver un grand nombre d'extensions permettant de produire des graphiques correspondant à un besoin particulier (arbres, circuits électroniques, histogrammes,...). Vous pouvez en e et avoir besoin, un jour, de g  n  rer des graphiques de mani  re automatique    partir d'une commande. Jetez alors un coup d'oeil sur les diff  rentes extensions disponibles (pstricks, METAPOST,...) ainsi que sur l'environnement picture et ses extensions epic et eepic. Bon courage !

⁶Du package ImageMagick disponible    <http://www.imagemagick.org>, qui convertit    peu pr  s tout format d'images

6

Documents scientifiques

Sommaire

- 6.1 Article
- 6.2 Bibliographie
- 6.3 Index
- 6.4 Diviser votre document

*Les sages thésaurisent la science
mais la bouche du fou
est un danger permanent.*

Les proverbes Pr 10 14.

VOICI venu le moment de vous parler des quelques caractéristiques des documents dits *scientifiques*. Si le problème des formules et autres équations a été abordé avec brio au chapitre 3, il reste tout de même un gros morceau à avaler : la bibliographie. Sachez quand même que si l'ingurgitation peut être difficile, la suite vous permettra de vous simplifier grandement le travail. Nous profiterons également de ce chapitre pour expliquer le principe de la génération d'index.

Nous vous parlerons donc des quelques particularités de la rédaction d'article, ensuite viendra un exposé sur la génération d'une bibliographie, la génération d'index, enfin la méthode utile à connaître pour diviser un gros document en petites parties.

6

6.1 Article

Pour rédiger un article, rien de bien nouveau, tout ce qui a été vu jusqu'ici s'applique. On notera juste l'utilisation dans le préambule, des commandes :

- \title pour définir le titre,
- \date pour définir la date,
- \author pour définir les auteurs,
- \thanks pour spécifier l'affiliation des auteurs.

Pour insérer le titre à partir de ces définitions, il est nécessaire d'ajouter la commande \maketitle après le \begin{document} :

```
\documentclass{article}
\title{Le seuillage à 128 : une révolution !}
\author{M. C. Orlanrien\\
        Institut du Pixel\\
        42007 Saint-Etienne---FRANCE}
\date{2 Avril 1927}

\begin{document}
\maketitle% c'est ici qu'est inséré le titre
```

```
...
\end{document}
```

Nous répétons :¹ c'est la commande `\maketitle` qui génère et insère le titre et non les définitions du préambule.

En règle générale, les conférences ou revues qui fournissent un fichier de style, proposent quelques variantes, par exemple une commande `\address` pour séparer les auteurs et leur adresse respective, etc. Mais l'idée de base reste la même.

6.2 Bibliographie

Il existe deux manières de rédiger une bibliographie avec $\text{\LaTeX}$: l'une que l'on peut qualifier de « manuelle » consiste à insérer un environnement `thebibliography` dans le document, l'autre que nous allons décrire ici, utilise le programme `BIBTEX`. Voici le principe :

1. on crée un ou plusieurs fichiers de données contenant une description de chaque entrée de bibliographie (article, conférence,...) au format `BIBTEX`. C'est l'inévitable tâche de *saisie*,
2. dans le document, on fait référence aux entrées par la commande `\cite`,
3. la bibliographie sera formatée automatiquement selon un style particulier que vous choisirez.

L'avantage de cette méthode est que vous saisissez une fois pour toutes les entrées de votre bibliographie. De plus, vous n'avez pas à vous soucier de sa mise en page, dans la mesure où vous utilisez des *fichiers de style* ; il en existe plusieurs dizaines correspondant à toutes sortes de standards, revues et autres conférences. On trouve aussi sur internet beaucoup de bases de données bibliographiques au format `BIBTEX` que l'on peut utiliser directement dans ses documents.

Nous répétons qu'il existe des standards en matière de bibliographie, mais que malheureusement certaines revues prennent un malin plaisir à pondre leur propre style de bibliographie. Le jour où vous publierez dans ce genre de revue, vous aurez à créer ou adapter un fichier de style. Pour ce faire, cherchez du côté de l'utilitaire `makebst`.

6.2.1 Fichier `.bib`

La première opération est de constituer² le fichier de bibliographie qui doit de préférence porter l'extension `.bib`. Ce fichier doit suivre une syntaxe particulière. Tout d'abord, il faut savoir que `BIBTEX` distingue chaque entrée par son *type*. Ainsi, chaque entrée correspond à un type de document : livre, article, conférence, rapport technique,... En tout plus d'une douzaine de types de document différents.



Accompagnant les distributions $\text{\LaTeX}$, on trouve normalement un fichier nommé `BIBTEXing` (généralement sous le nom `btxdoc.dvi`) écrit par Oran PATASHNIK il y a une vingtaine d'années et contenant une source importante d'information sur la manière de construire un fichier au format `BIBTEX`.

Chaque *type* d'entrée contient à son tour un certain nombre de *champs* décrivant l'entrée. La structure d'une entrée de bibliographie est la suivante :

¹Parce qu'il paraît qu'enseigner c'est répéter

²Le module `AucTeX` d'Emacs possède un mode `BIBTEX` très pratique.

```
@entree{clef,
  champ1 = {...},
  champ2 = {...},
  ...
  champn = {...}
}
```

où *entree* est le type de document (article, inproceedings,...) et *champ₁*, *champ₂*, ..., *champ_n* sont les différents champs de l'entrée de bibliographie. Ces différents mots réservés de *BIBTEX* peuvent être saisis en majuscules ou en minuscules.

Le symbole *clef* doit identifier le document de manière univoque. Ce symbole est à rapprocher du symbole identifiant une étiquette avec `\label`. Pour vous permettre de commencer à utiliser rapidement *BIBTEX* nous vous donnons, ici, un exemple pour les trois principales entrées que vous serez amené à utiliser :

Article dans une revue

Un article dans une revue doit être saisi comme suit :

```
@article{qtz:UchArb,
  author = {Uchiyama, Toshio and Arbib, Michael A.},
  title = {Color Image Segmentation Using Competitive Learning},
  journal=pami,
  volume =16, number=2, pages={1197--1206},
  month=dec, year=1994}
```

Notez que :

1. les champs *author*, *title*, *journal*, *year* sont obligatoires ;
2. pour les auteurs,³ il est impératif de suivre l'ordre *nom*, *prénom* et de séparer tous les auteurs par *and* ;
3. pour les auteurs ayant un nom composé ou à particule, on saisira :

```
author="de la Motte Beuvron, Alain"
```

donc en respectant l'ordre *particule*, *nom*, *prénom* et en utilisant la virgule en guise de séparateur comme indiqué ci-dessus ;

4. tous les mois de l'année peuvent être produits grâce aux chaînes : jan, feb, mar, etc.

Nous avons créé par commodité l'*abréviation* *pami* qui est définie au début de notre fichier *.bib* par :

```
@string{pami="IEEE transactions on Pattern Analysis and Machine
Intelligence"}
```

Article dans une conférence

Eh oui, *BIBTEX* distingue un article dans une *revue*, et un article dans une *conférence*. La structure est sensiblement la même, si ce n'est qu'on utilise le champ *booktitle* pour le titre de la conférence, à la place du titre de la revue :

³Ces remarques concernant les auteurs sont valables pour les autres entrées (conférences, livres, etc.

```
@Inproceedings{qtz:BouOrch,
  author={Bouman, Charles A. and Orchard, Michael T.}
  title={Color Image Display with a Limited Palette Size},
  booktitle={SPIE Conference on Visual Communications
    and Image Processing},
  volume=1199,pages={522--533},
  year=1989}
```

Ici les champs `author`, `title`, `booktitle`, `year` sont obligatoires, et l'on doit choisir entre `volume` et `number`.

Un extrait de livre

On cite souvent un extrait—chapitre(s), ou page(s)—d'un livre plutôt que le livre lui-même :

```
@inBook{col:McA,
  author = {MacAdam, David L.},
  title = {Color Measurement},
  chapter = 4,
  pages = {48--49},
  publisher = {Springer-Verlag},
  year = 1985}
```

Sont obligatoires : `author`, `title`, `chapter` ou `pages`, `publisher` (l'éditeur) et `year`.



Encore une fois nous ne saurions trop vous conseiller d'exploiter le mode `BIBTEX` du module `AuctEX` d'Emacs. Ce mode vous propose notamment un menu contenant tous les types d'entrée. La sélection d'un item de ce menu insère un « squelette » d'entrée dans votre fichier `.bib`. Ce module est téléchargeable à <ftp://ftp.lip6.fr/pub/Tex/CTAN/support/auctex> et également disponible sous la forme d'un paquet Debian.

a

6.2.2 Citation

Une fois le (ou les) fichier(s) de bibliographie constitué(s), on peut faire référence aux entrées par l'intermédiaire des clefs, avec la commande `\cite` :

```
\cite{clef}
```

La commande `\cite` a pour effet :

1. d'insérer un renvoi dont la forme dépend du style choisi ([2], [Loz95],...),
2. d'ajouter l'article cité dans la bibliographie de votre document.



Un article—au sens large du terme—n'apparaît dans la bibliographie que s'il fait l'objet d'une commande `\cite`. Pour qu'un article apparaisse sans pour autant être cité, il faut utiliser la commande `\nocite{clef}`. L'article référencé par `clef` sera alors inséré dans la bibliographie. Par ailleurs, la commande `\nocite{*}` insère *toutes* les entrées de votre fichier de biblio.

Avant de passer à l'étape de génération proprement dite, il est nécessaire d'insérer à la fin du document `LATEX` un appel à la commande `\bibliographystyle` pour stipuler un style de bibliographie, puis un appel à la commande `\bibliography` pour insérer effectivement la bibliographie. Pour le style :

```
\bibliographystyle{style}
```

Les trois styles prédéfinis⁴ de L^AT_EX sont :

- plain les citations sont sous la forme [2], et la bibliographie est classée par auteur,
- unsrt idem mais pas de tri, les documents apparaissent dans l'ordre où ils sont cités ; très utilisé pour les actes de conférences.
- alpha les citations sont sous la forme « auteurs abrégés + année ».

Il faut ensuite spécifier quels sont les fichiers contenant les informations bibliographiques sur lesquelles « pointent » les commandes \cite de votre document :

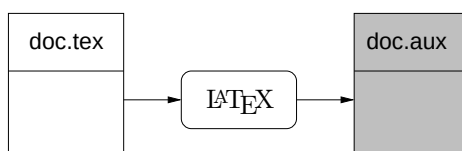
```
\bibliography{fichier1,fichier2,...}
```

indiquera à B_IB_TE_X de considérer les fichiers fichier1.bib, fichier2.bib,... lors de son traitement.

6.2.3 Génération

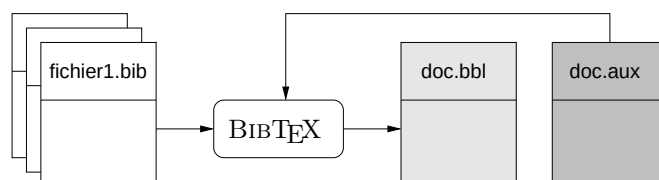
Pour générer la bibliographie :

1. effectuer une première compilation avec L^AT_EX pour que le fichier auxiliaire doc.aux contienne les informations de citations :

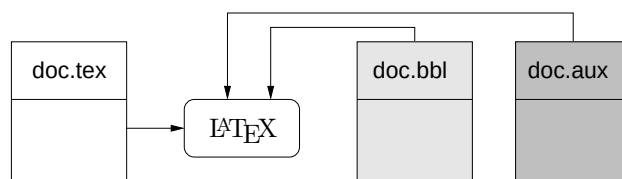


2. lancer B_IB_TE_X pour générer la bibliographie dans le fichier doc.bbl :

| bibtex doc



3. effectuer une deuxième compilation avec L^AT_EX pour insérer la bibliographie :



4. résoudre les références croisées par une troisième compilation.

Si vous êtes curieux, vous verrez que le fichier doc.bbl contient un environnement thebibliography prêt à l'emploi⁵ et que le fichier doc.blg est l'équivalent du .log : un fichier « log » contenant les éventuelles erreurs ou warnings de la dernière utilisation de B_IB_TE_X.



Le programme Bib_TE_X est sensible à la variable d'environnement BIBINPUTS. Il peut donc parfois être nécessaire d'ajouter la ligne :

⁴Cherchez sur les sites CTAN, dans le répertoire biblio/bibtex/contrib il y a plusieurs dizaines d'autres styles disponibles.

⁵C'est-à-dire celui que vous auriez dû vous palucher si vous n'utilisiez pas B_IB_TE_X.

```
| export BIBINPUTS=$HOME/LaTeX/biblio/:
```

dans votre `.bash_profile` pour que `BIBTEX` cherche vos fichiers de bibliographie dans le répertoire `$HOME/LaTeX/biblio` (c'est un exemple).

6.3 Index

La génération d'index s'appuie sur deux concepts :

1. l'ajout de commandes `\index` dans le document `LATEX` pour ajouter des entrées dans l'index ;
2. l'utilisation du programme `makeindex` qui va trier et mettre en page l'index proprement dit.

C'est la commande `\printindex` qui insère l'index dans le document. Cette commande est analogue à la commande `\tableofcontents`.

6.3.1 Ce qu'il faut faire

Voici un petit mémo pour faire un index.

1. rajouter deux commandes dans le document maître :

<code>\makeindex</code>	← pour dire à <code>L^AT_EX</code> de générer les index
<code>\begin{document}</code>	
... le document ...	
<code>\printindex</code>	← Pour réellement l'insérer dans le document
<code>\end{document}</code>	

2. ajouter une entrée dans l'index :

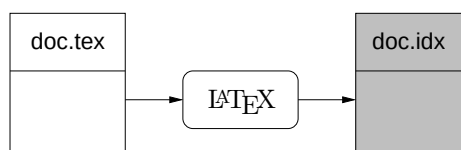
<code>\index{bidule}</code>	← insère « bidule » dans l'index
-----------------------------	----------------------------------

3. pour générer l'index pour le document `doc.tex`, on lancera successivement les trois commandes suivantes :

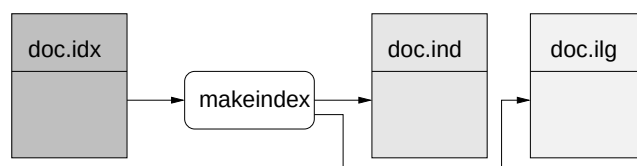
```
latex doc
makeindex doc
latex doc
```

6.3.2 Détail du fonctionnement

La première compilation du document `doc.tex` (à la condition que la séquence de contrôle `\makeindex` soit présente dans son préambule) génère un fichier `doc.idx` contenant les entrées de l'index « en vrac » :



On utilise ensuite `makeindex` pour classer et supprimer les doublons dans ce fichier `doc.idx`, le résultat est mis dans `doc.ind` ; une trace de l'exécution est stockée dans `doc.ilg` :

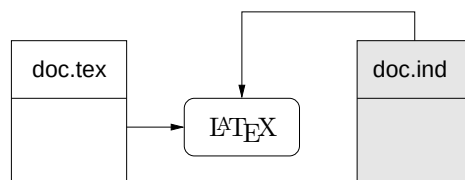


makeindex est bavard sur le terminal; voici ce qu'il dit pour générer l'index de ce document :

```

This is makeindex, version 2.13 [07-Mar-1997] (using kpathsea).
Scanning input file guide.idx....done (982 entries accepted, 0 rejected).
Sorting entries.....done (11254 comparisons).
Generating output file guide.ind....done (745 lines written, 0 warnings).
Output written in guide.ind.
Transcript written in guide.ilg.
  
```

Il faut donc veiller aux éventuels rejets ou avertissements (*warnings*) et se reporter au fichier log `doc.ilg` le cas échéant. La deuxième compilation avec $\text{\LaTeX}$ permet d'insérer l'index formaté (fichier `doc.ind`) à l'endroit spécifié par la commande `\printindex` dans `doc.tex` :



L'utilitaire `makeindex` reconnaît l'option `-s` qui permet de spécifier un *style* pour l'index. Ces styles—définis dans des fichiers portant l'extension `.ist`—changent la mise en page de l'index. On utilise un fichier style de la manière suivante :

```
makeindex -s fichier-style document-maitre
```

Cherchez sur votre distribution quels sont les fichiers de styles et testez-les.

6.3.3 Différents types d'entrée d'index

On peut utiliser des entrées un peu plus sophistiquées que la forme vue jusqu'à maintenant (`<mot>` et `<page>`). Il existe au moins trois autres entrées :

1. les entrées hiérarchiques :

```
\index{bidule!chouette}
```

insère une sous-entrée 'chouette' à 'bidule'

2. les entrées à cheval sur plusieurs pages :

```

\index{bidule|()}          ← à la page i
\index{bidule|)}          ← à la page j
  
```

génère une entrée de type : bidule i–j

3. les entrées symboliques :

```
\index{alpha@\alpha}
```

ajoute la lettre grecque α et la classe à l'entrée 'alpha'. De même :

```
\index{eplucher@éplucher}
```

ajoute le mot « éplucher » et le classe parmi les mots commençant par « e »
 Cette dernière forme peut également être utilisée pour mettre dans l'index une entrée avec une mise en forme particulière, par exemple :

```
\index{bonjour@\textbf{bonjour}}
```

ajoute **bonjour** (**bonjour** en gras) dans l'index, et classe cette entrée à 'bonjour'. Enfin on peut vouloir afficher les numéros de pages avec une mise en évidence particulière. On utilisera alors la forme :

```
\index{entrée|commande de mise en forme}
```

Par exemple :

```
\index{bidule|textbf}
```

affichera le numéro de la page où apparaît "bidule" en gras (notez qu'il n'y a pas de caractère \ pour la commande de mise en forme).

6.3.4 Glossaire

On a parfois besoin de préciser la signification de certains termes d'un document ; la partie d'un manuel qui regroupe l'explication de ces termes s'appelle un *glossaire*. Pour en générer, il faut procéder de manière analogue à un *index* avec quelques petites variations présentées au paragraphe 11.6 page 171.

a

6.4 Diviser votre document

Lorsqu'on manipule un gros document, on peut le diviser naturellement en chapitres ou parties. Il est alors conseillé de créer un document *maître* chargé d'inclure ces chapitres ou parties. Le document maître a l'allure suivante :

```
\documentclass{book}

\begin{document}
\frontmatter % tout ce qui est introductif
\include{preface}
\tableofcontents
\mainmatter % le « corps » du document
\include{chapitre1}
\include{chapitre2}
\backmatter % tout ce qui vient en fin de document
\bibliographystyle{plain}
\bibliography{machin,bidule,truc}
\end{document}
```

L'intérêt des commandes \include réside dans le fait qu'elles vous permettent de travailler sur un nombre réduit de chapitres à la fois, tout en gardant l'intégrité du document. On utilise pour cela la commande \includeonly :

```
\includeonly{preface,savoir}
```

dans le préambule, permet de compiler uniquement la préface (fichier preface.tex) et le chapitre contenu dans le document savoir.tex.



Chaque commande `\include` commence une nouvelle page. Et il n'y a apparemment pas moyen de passer outre. Vous aurez donc compris que `\include` est à utiliser avec les commandes de section qui sautent une page (`\chapter` et cie). Pour insérer un fichier sans saut de page, utilisez la commande `\input`. Par contre, vous ne bénéficierez pas du mécanisme de document maître.

Enfin, il est utile de noter que les commandes `\frontmatter`, `\mainmatter` et `\backmatter` ne sont pas indispensables, mais permettent automatiquement d'adopter une numérotation en roman pour les pages introductives et d'autres petites choses (elles ne sont cependant accessibles que dans la classe `book`).

7

Des documents en français

Sommaire

- 7.1 Le problème des lettres accentuées
- 7.2 Rédiger un document en français avec $\text{\LaTeX}$
- 7.3 Le package babel et la typographie
- 7.4 Courrier et fax

*L'homme répondit : c'est la femme
que tu as mise auprès de moi
qui m'a donné de l'arbre, et j'ai mangé !*

La Genèse Gn 3 12.

LA COMPOSITION d'un document en français suit des règles qu'il est bon de connaître. Ces règles ne sont pas à proprement parler des directives dont on ne peut se soustraire, il s'agit la plupart du temps de règles d'usage, qu'il est conseillé de suivre pour rendre un document lisible ne perturbant pas le lecteur. Ces conseils d'usage donnent généralement un aspect sérieux voire professionnel à un document. Il existe plusieurs ouvrages traitant de la typographie française, je citerais ici le lexique de l'imprimerie nationale [8] et le manuel d'Yves PEYROUSSEAU [9].

Ce chapitre contient des informations sommaires sur la manière dont sont codées les fontes dans $\text{\LaTeX}$ pour obtenir les accents de la langue française. Suivent quelques règles de typographie et une présentation du package babel permettant de simplifier la saisie de documents en français. Ce chapitre se termine sur une présentation d'une classe de document lettre ayant pour but de composer des lettres et des fax.

7

7.1 Le problème des lettres accentuées

Il y a quelques années, lorsque $\text{\TeX}$ a été conçu, les fontes utilisées ne comportaient pas de lettres accentuées. Chacune de ces fontes était codée à l'aide de 7 bits par caractère, et donc contenaient quelques 128 caractères codables. Provenant des États-Unis, ces 128 caractères ne comportaient évidemment pas les caractères accentués de la langue française. C'est la raison pour laquelle, pendant un long moment de valeureux utilisateurs francophones de $\text{\TeX}$ et de $\text{\LaTeX}$ étaient contraints de saisir leur document en `fran\c{c}ais` avec des caractères assez pénibles `\'a` taper.

Aujourd'hui, ces petits désagréments ne sont plus qu'un mauvais souvenir. Depuis 1990 un codage des fontes tenant compte de caractères accentués de plusieurs langues a été adopté et porte le nom de *Cork encoding* ou de *codage T1*. Le tableau 7.1a page suivante donne à titre indicatif les caractères correspondants à chaque valeur. Dans ces tableaux, les cases sont numérotées à partir de 0, les valeurs augmentent de droite à gauche et de haut en bas.

`	´	^	~	¨	ˆ	°	ˇ	˘	˙	˚	¸	ˆ	˜	<	>
“	”	„	«	»	—	—		□	□	□	□	fi	fl	□	□
□	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
p	q	r	s	t	u	v	w	x	y	z	{		}	~	.
Ā	Ą	Ć	Č	Ď	Ě	Ę	Ğ	Ĺ	Ł	Ł	Ń	Ň	Ŋ	Ō	Ŕ
Ř	Ś	Ŝ	Ş	Š	Ť	Ŧ	Ů	Ű	Ý	Ž	Ž	Ž	ı	đ	š
ă	ą	ć	č	ď	ě	ę	ğ	ĺ	ł	ł	ń	ň	ŋ	ő	ř
ŕ	ś	ŝ	ş	š	ť	ŧ	ű	ű	ÿ	ž	ž	ž	ı	ı	£
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Œ	Ø	Ù	Ú	Û	Ü	Ý	Þ	□
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
ð	ñ	ò	ó	ô	õ	ö	œ	ø	ù	ú	û	ü	ý	þ	ß

(a) Codage Cork (T1)

NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	NL	VT	NP	CR	SO	SI
DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
SP	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
ð	ñ	ò	ó	ô	õ	÷	ø	ù	ú	û	ü	ý	þ	ÿ	

(b) Caractères iso-latin1

TAB. 7.1 – Codage des fontes T1 et codage de caractères Iso-latin1. Le tableau du bas montre un codage de caractères (incluant les lettres accentuées de plusieurs langues européennes) aujourd’hui très répandu : le codage iso-latin1.

On peut remarquer que le codage des fontes est différent de celui des caractères ; le tableau 7.1b montre le codage dit *iso-latin1* qui fait maintenant office de standard pour le codage des caractères de la plupart des langues européennes. Les packages L^AT_EX contiennent donc une opération de « traduction » du codage des caractères (par ex. iso-latin1) en codage des fontes (par ex: codage T1).

7.2 Rédiger un document en français avec L^AT_EX

Il existe deux packages L^AT_EX permettant de « franciser » un document : le package french et le package babel. Pour des raisons tout à fait partisans, nous nous intéresserons au deuxième. On active le package babel comme suit :

```
\usepackage[francais]{babel}
```

Cet ordre dans le préambule met en route cinq fonctionnalités qui sont :

Césure : babel gère la césure des paragraphes en tenant compte de la langue française,¹ et plus particulièrement des mots accentués du français ;

Typographie : les règles de typographie française sont appliquées notamment en ce qui concerne les guillemets et les signes de ponctuation ;

Mise en page : il s'agit essentiellement de réintroduire l'indentation du début de paragraphe qui suit un titre de section,² changement du symbole et d'espace-ment pour les environnements de types listes, ...

Traduction : tous les mots susceptibles d'être traduits (« Chapitre », « Table des matières », etc.) sont traduits en français ;

Macros : un ensemble de macros sont disponibles avec le package babel, ces macros permettent de saisir correctement certaines constructions courantes en français, telles que n^o, 1^{er}, 2^o, 37° C, ...

7

7.3 Le package babel et la typographie

L'ensemble des « règles » liées à la typographie française dépasse largement le cadre de ce chapitre. Heureusement le package babel permet pratiquement de les utiliser sans les connaître. Il suffit simplement de respecter quelques règles de *saisie* du document L^AT_EX pour que la composition respecte les règles de typographie les plus courantes. Ainsi, à titre d'exemple, babel insérera un quart de cadratin insécable avant le point virgule ; ce qui est une pratique courante en typographie française.



Si cette insertion automatique ne vous convenait pas, il est possible d'appeler la commande `\NoAutoSpaceBeforeFDP`. Vous serez alors responsable de l'insertion ou non de l'espace avant les signes de ponctuations.

7.3.1 Ponctuation

Les règles à connaître pour la ponctuation peuvent se résumer aux deux propositions suivantes :

¹On ne coupe pas de la même manière les mots anglais et les mots français.

²Ce qui n'est pas le cas en typographie anglaise.

1. un espace doit apparaître avant et après tous les signes de ponctuation *doubles*, c'est-à-dire les signes ; : ! ? « et »
2. on saisit un espace après (et pas avant) les signes de ponctuation *simples*, c'est-à-dire les signes . , (et)

Le respect de cette saisie permet à babel d'insérer les espaces nécessaires avant et après les signes de ponctuation. À ce sujet, il est intéressant de remarquer que les espaces avant les points d'interrogation et d'exclamation sont des espaces fines :

```
fouilla ! et \selectlanguage{english}
fouilla !\selectlanguage{french}
```

7.1

fouilla! et fouilla !

7.3.2 L-a, e dans l'a, t-i, t-i, a!

Je cite Serge GAINSBURG en guise de titre de ce paragraphe sur les deux «jolies» ligatures de la langue française : 'æ' et 'œ'. Au sujet de la saisie de ces ligatures, on peut au choix : saisir \oe et \ae (\AE et \OE en majuscules) :

L\ae titia va au Sacré-C\oe ur.

7.2

Lætitia va au Sacré-Cœur.

Ou saisir directement 'æ' sur le clavier s'il le permet. À titre indicatif, AltGr+a donne l'e dans l'a sur un système Linux digne de ce nom. Et pour une histoire compliquée l'e dans l'o ne se trouvant pas dans la norme iso-latin1, mon clavier n'est pas en mesure de fournir la ligature 'œ'.

7.3.3 Outils du package babel

Un grand nombre de «petites choses» restent toujours très floues quant à la manière correcte de les «typographier». Je pense à toutes ces abréviations courantes telles que : Monsieur, Madame, premier, deuxième, primo, etc. Heureusement le package babel répond à certaines de nos interrogations.

1\ier	1 ^{er}
3\ieme	3 ^e
37\degres{ } C	37° C
\primo, \secundo, \tertio, \quarto	1°, 2°, 3°, 4°
\no 4	n° 4
\No 4	N° 4

Lettrine

ON TROUVE dans certains documents des *lettrines* comme celle en début de ce paragraphe. Le package french de Bernard GAULLE et le package lettrine définissent une telle commande. Nous vous proposons dans la deuxième partie de ce document un exemple de code L^AT_EX permettant de générer une telle commande.

Sommaire

Dans un document français, on insère généralement la table des matières en fin de document et le sommaire, qui est une table des matières résumée, en début de

document. Le package french propose la commande `\sommaire` qui permet — comme son nom l’indique — un sommaire dans le document. Encore une fois, nous vous proposons dans la deuxième partie de ce document d’étudier une manière de générer un tel sommaire.

7.3.4 Recommandations d’usage

Les recommandations qui suivent ne sont pas à proprement parler des fonctionnalités du package babel ; ces recommandations sont des conseils que vous pourrez retrouver dans des ouvrages ayant trait à la typographie :

- les guillemets à la française se saisissent soit avec «et» si votre clavier permet de les générer, soit avec les signes inférieurs et supérieurs : << et >>, soit avec les commandes `\og` et `\fg`. Les guillemets “à l’anglaise” se saisissent avec les quote et backquote : ‘ ’ et ’ ’ ; dans tous les cas, utiliser " pour les guillemets n’est pas recommandé ;
- les locutions latines se saisissent *a priori* en italique ;
- on abrège :

<i>et cætera</i>	avec	etc.	etc. et non pas «etc...»
Monsieur	avec	M. ³	M. Machin
Messieurs	avec	MM.	MM. Machin et Bidule
Madame	avec	M ^{\up{me}}	M ^{me} Machin
Mademoiselle	avec	M ^{\up{lle}}	M ^{lle} Machin
kilomètre(s)	avec	km	25 km (pas de ‘s’)
kilogramme(s)	avec	kg	25 kg

- le séparateur de partie décimale et partie entière est la *virgule* en français, et le point en anglais. On «doit» donc écrire : 123,54.
- on insère un quart de cadratin tous les milliers, et millièmes :

`\nombre{12345678,23434}`

12 345 678,234 34

- il est d’usage d’écrire les noms propres en petites capitales, comme ceci : John COLTRANE. Ici on a utilisé la commande `\textsc{Coltrane}` ; le package babel contient la macro `\bsc` : `\bsc{COLTRANE}`, `\bsc{Coltrane}` et `\bsc{coltrane}` donnent le résultat escompté ;
- on écrit les sigles sans point et en lettre capitales (RATP, SNCF, ENISE). Certains sigles qui «se prononcent bien» peuvent même s’écrire en minuscules : Assedic, Inserm, etc.

7.3.5 Le cas de l’euro

Le symbole de l’euro peut être produit à l’aide de la commande `\texteuro` du package textcomp. On obtient alors le caractère : € ou € en utilisant la fonte sans sérif. Une autre approche consiste à utiliser la package eurosym fournissant les commandes :

- `\euro{}` : €
- `\EUR{35}` : 35€

³Et non pas «Mr» qui est l’abréviation anglaise de *mister*.

7.3.6 Au sujet des majuscules

En dehors des cas bien connus où l'on doit mettre ou ne pas mettre de majuscule (il faut en mettre en début de phrase, ne pas en mettre pour commencer une parenthèse, selon le contexte après les deux points, etc.), voici trois points importants au sujet des majuscules (capitales comme disent les typographes).

Tout d'abord les majuscules doivent être accentuées (je ne m'énerve pas, j'explique) lire à ce sujet ce que dit Yves PERROUSEAUX dans son manuel. Il y est expliqué que les accents présents sur les majuscules depuis le ^{XVI}^e siècle ont disparu avec l'arrivée des machines à écrire et de composition typographique d'origine anglo-saxonne. On peut également trouver dans tous les bons ouvrages de typographie des exemples de phrases ambiguës lorsque les accents ne sont pas mis.

Ensuite, *dans un titre*, on ne mettra une majuscule qu'à la première lettre (contrairement à l'anglais où on met une majuscule à chaque mot). Enfin, il faut insister sur le fait que l'usage des majuscules est un domaine dont les nuances sont assez subtiles à saisir. Notons ici quelques points pour appréhender ces « règles » :

- on écrit maître de conférences (donc sans majuscule) ;
- l'université Jean Monnet (pas de majuscule à université) ;
- mais l'Université lorsqu'on parle de la structure en tant qu'entité propre ;
- le ministre de l'Intérieur ;
- l'académie de Lyon ;
- l'Assemblée nationale et le Sénat parce que ce sont des organismes uniques ;
- les Espagnols (pour le peuple) et le français (pour la langue)

Je ne résiste pas à l'envie de citer Jacques ANDRÉ :

« [...] Voici typiquement le genre de phrase que l'on trouve dans notre rapport d'activité :

Jean Transent, Maître de Conférence en Analyse de Données à l'Université de Nancy(Bien connue de la Communauté Scientifique Internationale) a donné, lors du séminaire de Biologie Informatique de Mardi 23 Juin, une conférence sur les Applications de l'Intelligence Artificielle à l'emploi de la Télévision Haute Définition en Robotique Avancée.

Dans cette phrase, il y a 23 majuscules. Il ne devrait y en avoir que trois (Jean, Transent et Nancy). si si... »

Jacques ANDRÉ [10]

et Yves PERROUSEAUX :

« Les dénominations d'une dignité, d'une charge, d'un grade ou d'une fonction **sont des noms communs** :

...

- le président du conseil général, etc.

C'est un nom commun, au même titre que le concierge ou les femmes de ménage du conseil général.

»

Yves PERROUSEAUX [9]

7.4 Courrier et fax

Le noyau de L^AT_EX comprend une classe de document pour rédiger des lettres. Cependant cette classe n'est pas très souple et mal adaptée au français.⁴ Pour les lettres françaises, nous conseillons l'utilisation de la classe `lettre` de Denis MEGÉVAND de l'observatoire de Genève. La classe et sa documentation peuvent se trouver à : <ftp://obsftp.unige.ch/pub/tex/macros/lettre> et dans le paquet `tetex-frogg` de la distribution Debian Sarge.

7.4.1 Commandes disponibles

Voici quelques unes des entités que l'on peut définir dans la classe `lettre` :

- Adresse de l'expéditeur en utilisant la commande `\address` ;
- Ville originaire `\lieu` permet d'écrire en haut à droite, l'endroit d'où l'on écrit la lettre ;
- Téléphone et fax sont précisés avec les commandes `\telephone` et `\fax` respectivement ;
- Signature à l'aide de la commande `\signature` ;
- Objet de la lettre avec la commande `\conc` (pour concernant) ;
- Pièces jointes grâce à la commande `\encl` (de l'anglais *enclosed*)

7.4.2 Structure d'un document basé sur la classe `lettre`

Nous donnons à la figure 7.1 page suivante l'«ossature» d'un document L^AT_EX basé sur la classe `lettre`. Les commandes `\opening` et `\closing` sont obligatoires et ont pour respectivement pour objet d'introduire les formules de politesse de début et de fin de lettre.

7.4.3 Fichiers «instituts»

La classe `lettre` est livrée avec un fichier `default.ins` qui définit par défaut l'adresse de l'observatoire de Genève. L'administrateur du système L^AT_EX que vous utilisez devra donc adapter ce fichier à votre organisation.

On peut cependant définir son propre fichier «institut» et l'inclure dans ses lettres. Lorsqu'on veut envoyer des lettres à titre personnel,⁵ il est en effet plus logique d'utiliser ses propres coordonnées ; on pourra alors définir un fichier nommé `moi.ins` contenant par exemple :

```
\address{%
  M. Expéditeur\\
  27, rue du cube parfait\\
  19683 Huit}
\lieu{Huit sur Loire}
\telephone{1234567890}
\fax{0987654321}
\signature{Tar \textsc{Tempion}}
```

⁴Appréciation personnelle sur la version 1.2z de la classe `letter` du 9 février 1999.

⁵Ce qui n'a pas véritablement lieu d'être puisque vous êtes tout de même au boulot pour bosser et non pas pour envoyer du courrier personnel.

```

\documentclass[12pt]{lettre}
\usepackage[francais]{babel}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}

\begin{document}

\begin{letter}{%
  M\up{me} \textsc{Destinataire}\\
  4, rue de Square\\
  65536 Carré
}
\address{%
  M. Expéditeur\\
  27, rue du cube parfait\\
  19683 Huit}

\lieu{Huit sur Loire}
\telephone{1234567890}
\fax{0987654321}
\signature{Tar \textsc{Tempion}}

\conc{au sujet du bidule}

\opening{Madame,}

... Le corps de la lettre ...

\closing{Veuillez agréer, madame,
  l'expression de mes salutations
  distinguées}
\encl{Deux ou trois choses.}
\end{letter}
\end{document}

```

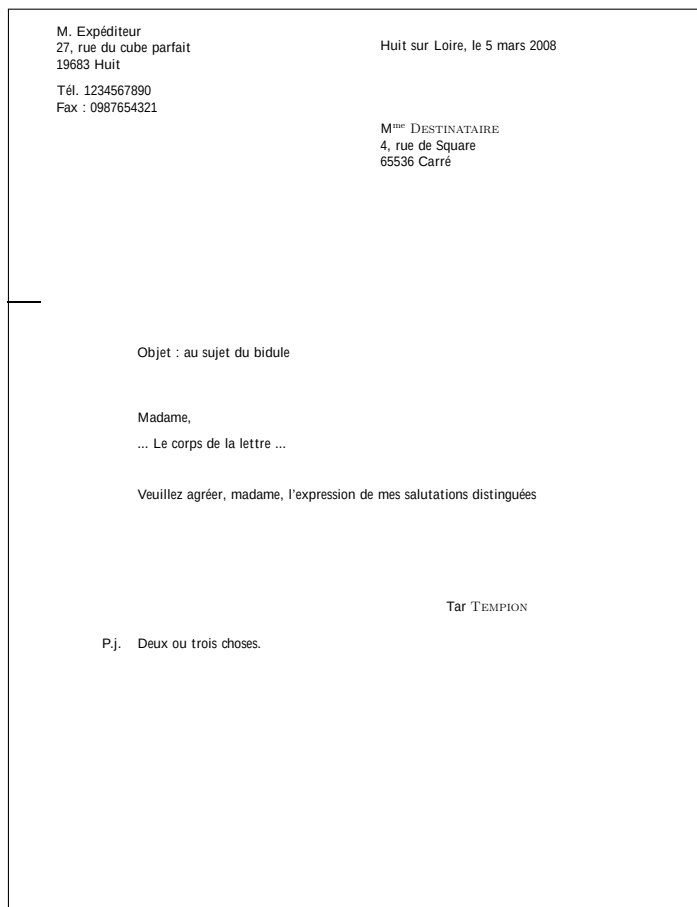


FIG. 7.1 – Ossature d'un document basé sur la classe lettre.

Il suffit alors d'insérer dans le préambule du document, la commande `\institut` qui cherche un fichier portant l'extension `.ins` :

```
\institut{moi}
```

7.4.4 Fax

La classe lettre contient également un environnement pour préparer un fax avec un en-tête correspondant à votre organisation. Le principe général et les mots clés sont les mêmes à condition d'utiliser l'environnement `telexfax` en lieu et place de l'environnement `letter`.

Notez qu'on peut ici aussi utiliser les fichier «instituts». Enfin la commande `\addpages` permet de gérer le cas où vous joignez un document déjà imprimé à votre fax. Par exemple si vous avez à envoyer n pages à votre fax initial, il faudra ajouter la commande `\addpages{n}`. La figure 7.2 page ci-contre montre le document minimal pour créer un fax.

```

\documentclass[12pt]{lettre}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}
\usepackage[T1]{fontenc}

\begin{document}

\begin{telefax}{01234567}{% numéro de fax
  M. le destinataire\\
  33 rue du fax\\
  98000 CCIT groupe 3}

\address{\centering
  Institut du pixel\\
  128 rue du niveau de gris\\
  65535 Érode sur Loire}

\name{René Kspéditeur}

\conc{la bazar}

\opening{Cher Monsieur,}

... patila patala

\closing{À bientôt.}

\end{telefax}
\end{document}

```

Institut du pixel 128 rue du niveau de gris 65535 Érode sur Loire	T É L É F A X				
TÉLÉPHONE : 987.64.20	TÉLÉFAX : 987.75.31				
<table border="0" style="width: 100%;"> <tr> <td style="width: 50%;"> À : M. le destinataire 33 rue du fax 98000 CCIT groupe 3 </td> <td style="width: 50%; text-align: right;"> Télécopie : 01234567 </td> </tr> <tr> <td> De : René Kspéditeur </td> <td style="text-align: right;"> Nombre de pages : </td> </tr> </table>		À : M. le destinataire 33 rue du fax 98000 CCIT groupe 3	Télécopie : 01234567	De : René Kspéditeur	Nombre de pages :
À : M. le destinataire 33 rue du fax 98000 CCIT groupe 3	Télécopie : 01234567				
De : René Kspéditeur	Nombre de pages :				
<i>En cas de mauvaise transmission, appelez s.v.p. l'opérateur téléfax</i>					
Sassonne-le-Creux, le 5 mars 2008					
Objet : la bazar					
Cher Monsieur, ... patila patala					
À bientôt.					
René Kspéditeur					

FIG. 7.2 – Ossature d'un document « fax »

8

À vous de jouer !

Sommaire

- 8.1 Livres et autres manuels
- 8.2 Local
- 8.3 EffTépe, Ouèbe et niouses

*Tu ne coucheras pas avec un homme
comme on couche avec une femme.
C'est une abomination.
Le Lévitique Lv 18 22.*

S'IL EST vrai que $\text{\LaTeX}$ permet de faire à peu près tout ce que l'on veut, il est souvent difficile de savoir comment le lui demander. Nous tenterons ici de vous donner quelques points d'entrée pour chercher plus de documentation sur le monstre.

8.1 Livres et autres manuels

La documentation «standard» concernant $\text{\TeX}$ et $\text{\LaTeX}$ est constituée des ouvrages suivants :

- *$\text{\LaTeX}$: A Document Preparation System* de L. LAMPORT [2] ;
- *The $\text{\LaTeX}$ Companion* [4] de M. GOOSSENS, F. MITTELBACH et A. SAMARIN
- *The $\text{\LaTeX}$ Graphics Companion* [11] des mêmes auteurs ;
- *The $\text{\TeX}$ book* [3] de KNUTH ;
- La FAQ française de $\text{\LaTeX}$ disponible à l'url suivante :
<http://www.grappa.univ-lille3.fr/FAQ-LaTeX>.

On y trouve aujourd'hui environ 70 questions répondant essentiellement à des problèmes classiques de mise en page. Ce document est actuellement maintenu par B. BAYARD

Deux ouvrages d'initiation en français sont parus récemment:

- *$\text{\LaTeX}$* de D. BITOUZÉ et J.C. CHARPENTIER [12] ;
- *$\text{\LaTeX}$ pour l' impatient* de C. CHEVALIER et al. [13].

Les documents en ligne référencés par le $\text{\LaTeX}$ navigator (cf. plus bas) traitant de $\text{\LaTeX} 2_{\epsilon}$ en français, sont les suivants :

- *Apprends LaTeX!* de M. BAUDOIN ;
- *Joli manuel pour $\text{\LaTeX} 2_{\epsilon}$* de B. BAYARD
- *Aide mémoire pour $\text{\LaTeX}$* de C. WILLEMS et F. GERAERDS ;
- *Guide d'introduction français au traitement de texte $\text{\LaTeX}$* écrit par F. GERAERDS ;
- *Une courte (?) introduction a $\text{\LaTeX} 2_{\epsilon}$* de T. OETIKER, H. PARTL, I. HYNÁ, E. SCHLEGL, traduit de l'allemand.

Ces documents sont disponibles au format dvi et/ou PostScript et parfois pdf.

8.2 Local

Avant de vous jeter sur ce pauvre réseau international, sachez que si vous avez la chance d'utiliser la distribution $\text{\TeX}$, vous trouverez sous l'arborescence $\text{\TeX}$ (`/usr/share/texmf/doc1`), un répertoire `doc` contenant un certain nombre de documentations intéressantes :

- `latex` : la doc de toutes les extensions installées au format `dvi` ;
- `fonts` : des docs sur les fontes disponibles ;
- ...

D'autre part, vous trouverez dans le menu aide d'Emacs un guide de référence au format info très pratique ; avec notamment la syntaxe de toutes les commandes et environnements standard de $\text{\LaTeX}$.

Enfin, n'hésitez pas à utiliser les commandes de recherche de fichiers disponibles sur votre système pour essayer de trouver des informations sur un package ou une fonte à partir de son nom.

8.3 EffTépé, Ouèbe et niouses

Comme pour la plupart des logiciels gratuits, on trouve une foulditude d'informations sur internet.

8.3.1 Sites FTP

Il a été créé une archive *standard* pour $\text{\TeX}$ et Cie portant le doux nom de CTAN pour Comprehensive TeX Archive Network. Cette archive est copiée sur plusieurs sites français, donc inutile d'aller encombrer les quelques lignes qui traversent l'océan, vous trouverez votre bonheur dans :

- <ftp://ftp.lip6.fr/pub/TeX/CTAN> à ma connaissance un serveur français très complet, avec des « miroirs » de beaucoup de sites.
- <ftp://ftp.inria.fr/pub/TeX/CTAN> une autre réplique CTAN.

L'arborescence du CTAN est la même quel que soit le site, et on notera entre autres :

- `macros/latex` la racine de $\text{\LaTeX}$ et ses extensions,
- `graphics` la racine de ce qui touche de près ou de loin au graphisme,
- `support` les logiciels qui tournent autour de $\text{\LaTeX}$: correcteurs orthographiques, convertisseurs, éditeurs,...
- ...

8.3.2 Sites Web

Encore une fois, ne cherchez pas plus loin :

<http://tex.loria.fr>

Même si ce site ne semble plus être maintenu, il contient de très nombreux liens vers des documents de référence. On pourra également se référer au site :

<http://www.latex-project.org>

pour avoir des infos sur les modifications du format $\text{\LaTeX} 2_{\epsilon}$ et de l'avancement du projet $\text{\LaTeX} 3$. Notez toutefois que ces sites sont en langue anglaise.

¹Ou quelque chose d'approchant suivant l'installation effectuée.

8.3.3 Les newsgroups

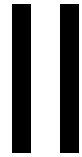
Il existent deux groupes de discussion sur T_EX et L^AT_EX :

- comp.text.tex environ 150 messages par jour ! Mais on peut y apprendre des choses, et
- fr.comp.text.tex beaucoup moins de transit et en français.

Ces groupes de discussion constituent une source d'information extraordinaire si on veut bien faire l'effort de trier un peu les messages parfois un peu trop nombreux. Vous pouvez *en dernier recours* — c.-à-d., après vous être documenté — poser une question sur le groupe. Si vous êtes clair et concis, la réponse ne se fait en général guère attendre.

*
* *

À vous de jouer !



Tout ce que vous avez toujours
voulu savoir sur (Tout ce que
vous avez toujours voulu savoir
sur $\text{\LaTeX}$ sans jamais oser le
demander) sans jamais oser le
demander

Introduction

*Que tes pieds sont beaux dans ta chaussure, fille de prince!
Les contours de ta hanche sont comme des colliers, Œuvre des mains d'un artiste.
Ton sein est une coupe arrondie, Où le vin parfumé ne manque pas
Ton corps est un tas de froment, Entouré de lis.²*

Le Cantique des cantiques Ct 7 2.

CETTE PARTIE s'intitule « Tout ce que vous avez toujours voulu savoir sur (Tout ce que vous avez toujours voulu savoir sur L^AT_EX sans jamais oser le demander) sans jamais oser le demander ». Elle a pour but d'expliquer comment les chapitres précédents ont été produits, et donc présente les différentes commandes et environnements qui ont été définis pour générer le manuel que vous avez sous les yeux. Mais son objectif est plus large puisque nous espérons fournir ici au courageux lecteur, une base solide pour la création de ses propres styles...

L'idée a germé dans mon esprit d'écrire les chapitres suivants après avoir eu plusieurs questions de lecteurs me demandant s'ils pouvaient réutiliser tel ou tel aspect du style de ce document. Le chantier que représente la rédaction des pages qui suivent a été pour moi titanesque dans la mesure où j'ai dû présenter des aspects de L^AT_EX qui ne font plus partie des connaissances de base et qui sont donc à ce titre plus difficiles à expliquer.³ Enfin — et non des moindres — ce qui reste généralement de l'ordre du bazar privé a dû ici être « rationalisé » pour être présentable. Ce qui n'a pas été une mince affaire.

J'ai voulu dans cette partie présenter la démarche que j'ai adoptée pour générer ce document. Je ne prétends pas qu'il s'agit du seul moyen possible pour obtenir la mise en page que vous avez sous les yeux. Par exemple, certaines parties de ce document auraient pu être produites à l'aide de paquets existants fournissant des fonctionnalités analogues ou même meilleures que celles des outils développés ici.

L'idée sous-jacente à cette partie est donc bien de guider l'utilisateur curieux vers des pistes d'exploration de L^AT_EX, de montrer comment on peut à l'aide de quelques outils, mettre au point des commandes originales correspondant exactement à ses propres besoins. Ces pistes sont suffisamment générales pour être suivies telles quelles ou adaptées pour des cas similaires ou non. Il s'agit donc de découvrir les grands classiques des fonctionnalités internes de « L^AT_EX » et d'avoir la satisfaction — sans pour autant prôner la « ré-invention » de la roue — de créer ses propres outils

J'ai tenté, autant que possible, de présenter des commandes n'utilisant que L^AT_EX. Il a cependant parfois été nécessaire d'utiliser certaines des fonctionnalités de T_EX ce qui a donné l'occasion de les présenter ici. Cette partie se compose donc de trois chapitres :

Outils nécessaires qui présente les commandes à connaître permettant de s'équiper pour la suite. On y trouve par exemple quelques pistes sur la structure des fichiers d'une distribution (II)-58es :

ment, et une présentation détaillée de la création de nouveaux environnements basés sur les listes ;

Cosmétique qui présente les outils qui ont été mis en œuvre pour changer l'allure des titres, des en-têtes et pieds de page, des marges, et quelques autres petites choses ;

De nouveaux jouets qui est l'occasion d'expliquer la création des onglets de ce manuel, du glossaire, des exemples, du sommaire, des lettrines, des notas, et de quelques autres bricoles.

Certaines explications données dans les chapitres qui suivent sont tellement fulmeuses que même l'auteur ne les comprend pas. Certaines solutions apportées aux problèmes ne sont que partielles. Enfin, certaines choses restent mystérieuses pour votre serviteur ; dans ces situations, un panneau « dos d'âne » est inséré dans le paragraphe.

9

Outillage nécessaire

Sommaire

- 9.1 Hercule Poirot
- 9.2 Outils de bas niveaux
- 9.3 Structures de contrôle et tests
- 9.4 Fontes

et là aussi, on trouvera généralement un répertoire par package, contenant les sources dans un fichier au format texte, portant l'extension `.sty` et éventuellement des fichiers connexes. Enfin, pour comprendre le comportement par défaut de $\text{\LaTeX}$, indépendamment des packages que l'on peut inclure, on pourra avoir recours au source de $\text{\LaTeX}$ dans :

```
/usr/share/texmf-texlive/tex/latex/base/latex.ltx
```

et aux sources des classes de documents dans :

```
/usr/share/texmf-texlive/tex/latex/base/book.cls
```

pour la classe `book`.

9.1.2 Examiner les macros

Un moyen pratique de trouver la définition d'une commande consiste à le demander à $\text{\LaTeX}$ lors d'une session interactive. On lance directement dans un terminal

```
*\show\hbox
```

```
> \hbox=\hbox.
```

← c'est une primitive

```
<*> \show\hbox
```

On constate ici que \hbox

- deux espaces avant «*de*» introduits par le saut de ligne à la fin de «est impératif» et celui à la fin de `\begin{hyperimportant}`;
- deux espaces après «*sauvegardes*» induits par le saut de ligne à la fin de «sauvegardes» et par celui à la fin de `\end{hyperimportant}`.

La preuve, si on supprime ces sauts de ligne :

Il est impératif\begin{hyperimportant} de
multiplier les
sauvegardes\end{hyperimportant} de vos
documents personnels

9.4

Il est impératif *de multiplier les sauve-*
gardes de vos documents personnels

Pour éviter d'avoir à se soucier de ce genre de problème on a généralement recours à deux commandes permettant de supprimer ces espaces doubles. On fait appel, pour éliminer ceux situés *avant* la séquence à `\ignorespaces` et pour ceux situés *après*, à la commande `\unskip`.

La commande `\ignorespaces`

Cette commande procède à l'expansion des commandes qui suivent en ignorant tous les espaces qui la suivent :

```
\newcommand{\truc}{ } \newcommand{\bidule}{ }
```

```
a\truc\bidule b\par
a\ignorespaces\truc\bidule b
```

9.5

a b
ab

Dans l'exemple ci-dessus, les commandes `\truc` et `\bidule` ont pour seul but de produire un espace lorsqu'elles seront appelées. Par conséquent, la ligne :

```
a\truc\bidule b
```

produira 'a□□b' c'est-à-dire les deux lettres a et b séparées par deux espaces. L'appel avec la commande `\ignorespaces` ignore — comme son nom l'indique — les deux espaces produits par les commandes `\truc` et `\bidule`. On peut donc utiliser cette commande dans notre exemple précédent :

```
\newenvironment{hyperimportant}{%
  \bfseries\itshape\ignorespaces}\upshape\mdseries}
```

qui devrait supprimer un espace :

Il est impératif
\begin{hyperimportant}
de multiplier les sauvegardes
\end{hyperimportant}
de vos documents personnels.

9.6

Il est impératif *de multiplier les sauve-*
gardes de vos documents personnels.

La commande `\unskip`

Si vous êtes attentif, vous noterez que deux espaces entre «*sauvegardes*» et «*de*» résistent à nos assauts. C'est là qu'intervient la primitive T_EX `\unskip` qui enlève le dernier espace inséré :

```
\newcommand{\truc}{ } \newcommand{\bidule}{ }
a\truc\bidule b\par
a\truc\bidule\unskip b
```

9.7

a b
a b

Finalement la définition «correcte» de notre environnement est la suivante :

```
\newenvironment{hyperimportant}{%
  \bfseries\itshape\ignorespaces}{\unskip\upshape\mdseries}
```

qui devrait supprimer tous les espaces indésirables :

```
Il est impératif
\begin{hyperimportant}
  de multiplier les sauvegardes
\end{hyperimportant}
de vos documents personnels.
```

9.8

Il est impératif *de multiplier les sauve-*
gardes de vos documents personnels.

9.2.2 Le caractère @

Lorsque vous vous lancerez dans l'exploration des sources des packages vous remarquerez que le nom d'une grande partie des commandes qui y sont définies contient le caractère @. Or dans un document .tex, il n'est pas autorisé d'exécuter une commande dont le nom contient ce dernier. Ceci permet de protéger ou de limiter la portée des commandes des packages. Par exemple la commande \cb@defpoint, définie dans le package changebar, ne peut pas être appelée par un utilisateur du package. Pour redéfinir ces commandes internes, il est nécessaire d'effectuer la petite manipulation suivante :

```
\makeatletter
% ici on peut bidouiller
\renewcommand{\@ttention}{oulala...}
\makeatother
% ici on ne peut plus
```

La commande hypothétique \@ttention peut uniquement être manipulée si le caractère @ est une lettre. C'est le rôle de la commande \makeatletter qui transforme le caractère @ en une lettre comme les autres, tandis que \makeatother lui réaffecte sa fonction spéciale.



Cette manipulation n'est pas nécessaire dans les fichiers de styles inclus avec la commande \usepackage pour lesquels la lettre @ peut être utilisé comme un caractère.

La manière dont T_EX peut changer la catégorie des caractères est expliquée au chapitre suivant au paragraphe 10.5.1 page 142.

9.2.3 Le \let de T_EX

Il est parfois utile de modifier une commande interne de L^AT_EX pour ajouter une fonctionnalité à son comportement par défaut. Par exemple pour modifier la commande interne \bidule,¹ on peut procéder comme suit :

¹Oui oui ça n'est pas une commande interne, mais un exemple idiot de nom de commande qui n'existe pas...

1. sauvegarder la commande grâce à l’instruction `\let` de T_EX :

```
\let\biduleORIG\bidule
```

2. redéfinir la commande `\bidule` en se basant sur la définition initiale :

```
\renewcommand{\bidule}{quelque chose en plus\biduleORIG}
```

3. si nécessaire, revenir à la définition initiale grâce à :

```
\let\bidule\biduleORIG
```

9.3 Structures de contrôle et tests

Les structures introduites par le package `ifthen` suivent la syntaxe :

```
\ifthenelse{ expression booléenne }
{ ... code LaTeX si vrai ... }
{ ... code LaTeX si faux ... }
```

et :

```
\whiledo{ expression booléenne }
{ ... code LaTeX tant que c’est vrai ... }
```

L’**expression booléenne** peut être constituée selon le contexte de différentes commandes du package `ifthen`, et parmi elles :

- l’expression `nombre1>nombre2`, l’expression `nombre1<nombre2` ou l’expression `nombre1=nombre2` permettant de comparer les deux valeurs numériques `nombre1` et `nombre2` ;
- `\equal{C1}{C2}` qui renvoie vrai ou faux selon que la chaîne de caractère `C1` est égale à la chaîne `C2` ;
- `\isodd{nombre}` qui renvoie vrai si le `nombre` est impair, faux sinon ;
- `\value{compteur}` qui renvoie la valeur d’un `compteur` sous la forme d’un nombre exploitable dans les conditions booléennes ;
- `\lengthtest{test longueur}` qui renvoie l’évaluation de `test longueur`, test contenant les opérateurs `<`, `>` ou `=` et des longueurs L^AT_EX comme opérandes.

On notera également qu’on pourra utiliser les connecteurs logiques `\OR`, `\AND` et `\NOT` qui jouent le rôle qu’on attend d’eux dans une expression booléenne. On peut grouper des expressions avec les opérateurs `\(` et `\)`.

9.3.1 Booléens et opérateurs associés

Le package `ifthen` propose à ses vaillants utilisateurs la possibilité de manipuler des booléens. On peut en déclarer un avec la commande `\newboolean` :

```
\newboolean{id booléen}
```

qui définit une «variable» booléenne identifiée par `id booléen`. On pourra ensuite lui affecter la valeur `true` ou `false` avec la commande `\setboolean` :

```
\setboolean{id booléen}{valeur}
```

Et bien sûr on pourra utiliser le booléen ainsi créé avec les **structures de contrôle**, par exemple comme ceci :

```
\ifthenelse{\boolean{id booléen}}
{ code LaTeX si id booléen vaut vrai }
{ code LaTeX s'il vaut faux }
```

Il est bon de connaître la version T_EX de ce qui précède. On trouve en effet dans les packages L^AT_EX du code écrit en T_EX, et en particulier l'utilisation de la structure de contrôle «Si Alors Sinon». Voici un exemple pour définir un nouveau booléen avec monsieur T_EX :

```
\newif\ifimprimantecouleur
```

On le positionne à faux avec :

```
\imprimantecouleurfalse
```

et à vrai avec :

```
\imprimantecouleurfalse
```

On peut ensuite exploiter ce booléen dans une structure «Si Alors Sinon» à la mode T_EX comme suit :

```
\ifimprimantecouleur
... % code si on a une imprimante couleur
\else
... % code si c'est une imprimante noir et blanc
\fi
```

9.3.2 Exemples

On souhaite écrire une commande pour produire le développement de la fonction factorielle,² de manière à pouvoir écrire :

On peut exprimer la factorielle de 10 comme suit :

```
\begin{displaymath}
10!=\itfactorielle{10}
\end{displaymath}
```

On peut exprimer la factorielle de 10 comme suit :

$$10! = 10 \times 9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$$

Une façon de résoudre le problème est d'écrire une commande contenant une boucle \whiledo :

```
\newcommand{\itfactorielle}[1]{%
\setcounter{cptfact}{#1}% on stocke l'argument dans un compteur
\whiledo{\value{cptfact}>1}{% tant qu'il est > 1
\thecptfact\times% on l'a che avec un ×
\addtocounter{cptfact}{-1}}% on décrémente le compteur
1}% on a che 1 à la fin
```

Il faudra bien sûr déclarer le compteur :

```
\newcounter{cptfact}
```

On notera que dans la condition booléenne de la boucle «TantQue», on fait appel à la commande \value pour comparer la valeur du compteur avec la valeur 1. Un peu plus tordu : on peut implémenter cette commande de manière récursive :

²Y en a qui n'ont pas grand chose à faire de leur journée...

```

\newcommand{\recfactorielle}[1]{% version récursive :
\setcounter{cptfact}{#1}% on a ecte l'argument au compteur
\ifthenelse{#1>1}{% si cette valeur est supérieure à 1
  \thecptfact\times% on l'a che suivie de ×
  \addtocounter{cptfact}{-1}% on décrémente le compteur
  \recfactorielle{\thecptfact}}% on fait un appel récursif
{1}}% sinon (valeur=1) on a che 1

```

Cette commande produit évidemment le même résultat que la précédente. On notera que dans la condition du `\ifthenelse` on compare un nombre (`#1`) avec un autre (`1`). Enfin on pourra remarquer que la présence de la commande `\times` impose le mode mathématique pour exécuter ces commandes. On peut contourner le problème, si nécessaire, avec la commande `\ensuremath`.

Le `\whiledo` et le `\ifthenelse` ont été utilisés dans le document que vous avez sous les yeux pour générer les tableaux de symboles à la page 209 et 210, ainsi que les tableaux sur le codage à la page 88 du chapitre sur les documents en français. Nous avons tout d'abord créé une commande permettant d'insérer un symbole :

```

\affsymb{pzd}{249} \affsymb{pzd}{75}
\affsymb{pzd}{221} \affsymb{pzd}{88}

```

910	ù 249	K 75	Ý 221	X 88
-----	----------	---------	----------	---------

Cette commande est la suivante :

```

\newcommand{\affsymb}[2]{%
\framebox{% un cadre
\parbox[][16pt][b]{1em}{% autour d'une boîte paragraphe
  \centering% de 16 pt de hauteur, 1em de large,
  \Pisymbol{#1}{#2}\\% dont le contenu centré
  \tiny#2}}}% est composé du symbole et de son numéro

```

L'argument `#1` est le nom de la police (`pzd` ou `psy`), et l'argument `#2` est le numéro de **symbole**. Sinon, rien de particulier dans cette commande, si vous avez suivi jusqu'ici (notamment en lisant le chapitre 4 et plus particulièrement le paragraphe 4.4 page 56)... Nous avons ensuite défini une commande permettant d'insérer un série de symboles :

e

```

Voici les symboles Zapf Dingbats, à partir
du \No 40, sur 3 lignes et 6 colonnes :
\begin{center}
  \symboles[40]{pzd}{3}{6}
\end{center}

```

Voici les symboles Zapf Dingbats, à partir du N° 40, sur 3 lignes et 6 colonnes :

(40	) 41	* 42	+ 43	, 44	- 45
. 46	/ 47	0 48	1 49	2 50	3 51
4 52	5 53	6 54	7 55	8 56	9 57

Voici le code de la commande `\symboles` :

```

\newcommand{\symboles}[4][0]{%
  \setcounter{clig}{0}% Mise à zéro des compteurs de ligne
  \setcounter{ccol}{0}% et de colonne
  \setcounter{cligmax}{#3}% arguments 3 et 4 pour fixer
  \setcounter{ccolmax}{#4}% le nombre max de colonnes et de lignes
  % Pour chaque ligne :

```

```

\whiledo{\value{clig}<\value{cligmax}}{%
  \setcounter{ccol}{0}% remise à zéro du compteur de colonne
  % et pour chaque colonne :
  \whiledo{\value{ccol}<\value{ccolmax}}{%
    % on calcule le numéro du symbole
    \setcounter{csym}{%
      \value{clig}*\value{ccolmax}+\value{ccol}+1}
    % si sa valeur est inférieure à 256
    \ifthenelse{\value{csym}<256}{%
      \affsymb{#2}{\the{csym}}}% on l'a che
      \mbox{}}}% sinon on crée un boîte vide
    \stepcounter{ccol}% on passe à la colonne suivante
  \stepcounter{clig}% on passe à la ligne suivante
  % on saute une ligne, sauf à la fin
  \ifthenelse{\value{clig}<\value{cligmax}}{\\\}{}}

```

Il faudra bien sûr déclarer les cinq compteurs avec la commande `\newcounter`.

Et je sais que vous êtes tout particulièrement curieux de voir ce que fait cette commande lorsque le compteur dépasse les bornes :

```

\begin{center}
  \symboles[240]{psy}{3}{6}
\end{center}

```

Et je sais que vous êtes tout particulièrement curieux de voir ce que fait cette commande lorsque le compteur dépasse les bornes :

240	ñ	ò	ó	ô	õ
241	÷	ø	ù	ú	û
242					
243					
244					
245					
246					
247					
248					
249					
250					
251					
252					
253					
254					
255					

9.12

9.3.3 Tester la parité des pages

C'est une pratique courante — on le verra par la suite — de créer des commandes ayant un comportement différent selon la parité de la page. À l'entrée « Finding if you're on an odd or an even page » de la Faq anglaise [14] est expliqué que le naïf :

```

\ifthenelse{\isodd{\value{page}}}
{ ... la page est impaire ... }
{ ... la page est paire ... }

```

peut ne pas donner le résultat escompté. Le compteur de page lorsqu'il est examiné à la frontière entre deux pages peut ne pas être à jour : le compteur de page s'il est interrogé au début d'une page renverra le numéro de la page précédente... Ceci est dû à la manière dont T_EX procède pour effectuer les sauts de page. Pour contourner ce problème plusieurs solutions sont possibles. Celle adoptée dans ce document consiste à utiliser le package `chnpage` qui insère artificiellement un label à l'endroit où l'on veut tester la parité de la page.

Par conséquent dans le cas où le test de parité à réaliser peut être évalué à la frontière entre deux pages, il faudra écrire :

```

\checkoddpag%
\ifcpoddpag
  ... la page est impaire ...
\else

```

```
... la page est paire...
\fi
```

9.4 Fontes

9.4.1 Le jeu des «trois» familles

Pour conserver une homogénéité dans l'allure des caractères dans un document $\text{\LaTeX}$, sont définies trois familles :

1. la famille roman celle que vous êtes en train de lire ;
2. la famille sans sérif que vous êtes également en train de lire à l'instant même ;
3. et la famille machine à écrire, également appelée « typewriter » lorsqu'on est anglophone, que — cela ne vous aura sans doute pas échappé — vous êtes en train de lire.

Il est important de noter que ces trois familles de fontes sont par défaut trois familles de la police baptisée par son auteur (KNUTH lui-même) «Computer Modern». Elles sont conçues pour s'harmoniser au sein d'un même document. Dans cet ordre d'idée, il faudra toujours veiller à ce que ces trois familles (roman, sans sérif, et machine à écrire) soient visuellement «compatibles» entre elles. Les distributions de $\text{\LaTeX}$ proposent généralement des packages permettant d'utiliser les fontes PostScript dans un document, avec notamment le célèbre³ package times utilisant :

1. Times pour la famille roman celle que vous êtes en train de lire ;
2. Helvetica pour la famille sans sérif que vous êtes également en train de lire à l'instant même ;
3. Courier pour la famille machine à écrire.

De même le package newcent utilise :

1. NewCentury pour la famille roman celle que vous êtes en train de lire ;
2. AvantGarde pour la famille sans sérif que vous êtes également en train de lire à l'instant même ;
3. Courier pour la famille machine à écrire.

9.4.2 Désignation des fontes et de leurs attributs

Une fonte⁴ ou police de caractères est définie dans $\text{\LaTeX}$ par plusieurs caractéristiques dont il a été question au paragraphe 2.1 page 15. De manière à désigner la fonte à l'aide des commandes que nous allons découvrir dans ce paragraphe, on utilisera :

- un codage qui sera à quelques exceptions près le codage T1 ;
- une série de caractères identifiant la famille : cmr pour «computer modern roman», ptm pour «PostScript times», etc.
- une série de caractères désignant la «graisse» de la fonte, m pour «médium», b pour «bold» (gras), bx pour «bold extended» (gras étendu, c'est-à-dire gras avec des caractères plus large), etc.

³Mais obsolète. Aujourd'hui il est conseillé d'utiliser le package mathptmx

⁴Terme faisant référence au plomb de l'imprimerie...

- une série de caractères définissant l'allure (*shape* en anglais) de la fonte : `n` pour «normal», `it` pour «italique», `sl` pour «slanted» (penché), etc.

Fontes «computer modern»

Il s'agit d'un ensemble de fontes dessinées par Donald K_{NUTH}. et utilisées par défaut dans L^AT_EX. Les commandes `\emph`, `\textbf`, etc. sélectionnent donc automatiquement ces polices.

Computer Modern roman (cmr)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	it	italique
<i>machin Bidule Chouette chose</i>	m	sl	penché
MACHIN BIDULE CHOUETTE CHOSE	m	sc	petites capitales
machin Bidule Chouette chose	bx	n	gras étendu normal
<i>machin Bidule Chouette chose</i>	bx	it	gras étendu italique
<i>machin Bidule Chouette chose</i>	bx	sl	gras étendu penché
machin Bidule Chouette chose	b	n	gras normal

Computer Modern sans sérif (cmss)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	sl	penché
machin Bidule Chouette chose	bx	n	gras étendu normal
machin Bidule Chouette chose	sbc	n	semi gras condensé normal

Computer Modern typewriter (cmtt)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	it	italique
<i>machin Bidule Chouette chose</i>	m	sl	penché
machin Bidule Chouette chose	m	sc	petites capitales

Computer Modern fibonacci (cmfib)	Codage T1		
machin Bidule Chouette chose	m	n	normal

Computer Modern funny roman (cmfr)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	it	italique

Computer Modern dunhil (cmdh)	Codage T1		
machin Bidule Chouette chose	m	n	normal

Fontes en béton

Elles ont été dessinées par K_{nuth} pour son ouvrage intitulé « Mathématiques concrètes ». Le package `beton`⁵ permet de les charger dans un document.

⁵Notez ici le jeu de mots désopilant, *concrete* veut aussi dire «béton» en langue anglaise.

Concrete fonts (ccr)	Codage T1		
machin Bidule Chouette chose	m	n	normal
machin Bidule Chouette chose	m	sc	petites capitales
<i>machin Bidule Chouette chose</i>	m	sl	penché
<i>machin Bidule Chouette chose</i>	m	it	italique

Fontes « gothiques »

Les fontes ci-dessous sont dites de la famille gothique et ne sont à utiliser que dans un contexte bien précis faute de rendre le texte parfaitement illisible, comme celui que vous êtes en train de lire actuellement, d'ailleurs vous avez probablement déjà arrêté, donc je peux dire des gros mots : caca...

Gothique (ygoth)	Codage U		
machin Bidule Chouette chose	m	n	

Fraktur (yfrak)	Codage U		
ma...in Bidule Chouette ...ose	m	n	

Schwabacher (yswab)	Codage U		
machin Bidule Chouette chose	m	n	

Fontes PostScript

Les fontes ci-dessous sont généralement disponibles gratuitement et résident la plupart du temps dans les imprimantes.

Times (ptm)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	it	italique
<i>machin Bidule Chouette chose</i>	m	sl	penché
MACHIN BIDULE CHOUETTE CHOSE	m	sc	petites capitales
machin Bidule Chouette chose	b	n	gras

Palatino (ppl)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	it	italique
<i>machin Bidule Chouette chose</i>	m	sl	penché
MACHIN BIDULE CHOUETTE CHOSE	m	sc	petites capitales
machin Bidule Chouette chose	b	n	gras

charter (bch)	Codage T1		
machin Bidule Chouette chose	m	n	normal
machin Bidule Chouette chose	m	it	italique
	m	sl	penché
MACHIN BIDULE CHOUETTE CHOSE	m	sc	petites capitales
machin Bidule Chouette chose	b	n	gras

New century (pnc)	Codage T1		
machin Bidule Chouette chose	m	n	normal
<i>machin Bidule Chouette chose</i>	m	it	italique
<i>machin Bidule Chouette chose</i>	m	sl	penché
MACHIN BIDULE CHOUETTE CHOSE	m	sc	petites capitales

N mathpazo : pour le Palatino;

N ... : et d'autres, n'avez qu'à fouiller votre distribution...

Si l'on examine le contenu du fichier newcent.sty on trouve simplement :

```
\renewcommand{\rmdefault}{pnc}
\renewcommand{\sfdefault}{pag}
\renewcommand{\ttdefault}{pcr}
```

ce qui signifie que — comme expliqué au § 9.4.1 page 114 — on a redéfini les trois familles « roman », « sans sérif » et « machine à écrire » en les nommant par leur nom $\text{\LaTeX}$ standardisé : pcn pour PostScript NewCentrury, pag pour PostScript AvantGarde etc. Les noms standardisés sont donnés dans les tableaux de la section précédente.

Localement

Il est toujours possible de changer localement de fonte dans un texte en spécifiant les paramètres nécessaires :

```
{\fontfamily{cmfr}\selectfont
```

On passe en "Funny Roman" et même qu'on peut faire de l'*italique*...

c'est dingue !} Et hop nous voila de nouveau en \verb+\rmdefault+

On passe en "Funny Roman" et même qu'on peut faire de l'*italique*... c'est dingue! Et hop nous voila de nouveau en \rmdefault

Les appels qu'il est possible de faire avant la commande \selectfont sont :

- \fontencoding pour le codage;
- \fontfamily avec comme argument la famille (cmr pour Computer Modern, ptm pour PostScript Times, etc.);
- \fontseries pour préciser la graisse (argument b pour gras, m pour la graisse moyenne, etc.);
- \fontshape pour l'allure de la fonte (argument n pour normal, sl pour penché, etc.);
- \fontsize avec deux arguments : la taille des caractères et l'espace entre deux lignes consécutives.

Voici un autre exemple :

```
{\fontfamily{ppl}\fontseries{b}%
\fontsize{2cm}{2.5cm}\selectfont
gros !}
```

Et nous voila de nouveau en \verb+\rmdefault+

gros !

Et nous voila de nouveau en \rmdefault

Finalement pour utiliser à plusieurs endroits dans un document, une fonte dont tous les attributs sont fixes, on pourra avoir recours à la commande \DeclareFixedFont prenant six arguments (nom, codage, famille, graisse, allure, taille) et permettant d'être ensuite utilisé comme une déclaration :

```
\DeclareFixedFont{\toupiti}{T1}{pag}{m}{n}{3pt}
```

Avant {\toupiti bon bé là à moins d'avoir une bonne loupe vous ne serez pas capable de lire ce texte} après.

Avant bon bé là à moins d'avoir une bonne loupe vous ne serez pas capable de lire ce texte après.

9.5 Listes et nouveaux environnements

À plusieurs reprises dans ce document, nous avons eu recours à l'environnement `list` permettant de créer des environnements basés sur le principe des listes (numérotés, de descriptions, etc.). Nous donnons ici les bases nécessaires pour pouvoir utiliser cet environnement en s'appuyant sur des exemples.

9.5.1 Principe

Pour définir un environnement basé sur les listes, on utilisera la syntaxe suivante :

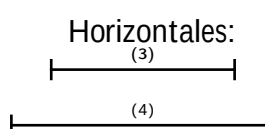
```
\newenvironment{maliste}%
{\begin{list}%
  { ... code pour l'item par défaut ... }
  { ... caractéristiques de la liste ... }
}%
{\end{list}}
```

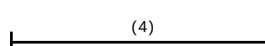
L'environnement `list` prend donc *deux* arguments. Le premier permet de définir l'allure de l'étiquette (ou item) par défaut. Le second permet de définir la liste elle-même et en particulier :

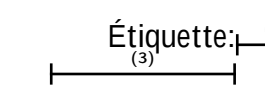
Sa géométrie : les marges, les espaces entre les paragraphes composant la liste, les espaces entre la liste et l'environnement dans lequel elle est insérée, etc.

la production de l'étiquette : c'est-à-dire la manière dont on va effectivement produire le titre de chaque entrée de la liste.

La liste suivante tente d'illustrer les différentes dimensions que l'on peut modifier pour définir sa propre liste :

Horizontales:  la dimension `\itemindent` (1) permet d'introduire une indentation pour le premier paragraphe d'une entrée de liste.

 la marge de gauche peut être définie par la dimension `\leftmargin` (4) et celle de droite par `\rightmargin`;

Étiquette:  la dimension `\labelsep` (2) détermine la dimension séparant l'étiquette du début du paragraphe. La dimension (3) `\labelwidth` définit quant à elle la largeur de la boîte contenant l'étiquette.

 Si on commence un nouveau paragraphe dans une entrée de liste, ce paragraphe est indenté de `\listparindent` (5) qui vaut 0 point par défaut.

Remarque « assez » importante:  si la largeur du texte de l'étiquette est inférieure à `\labelwidth` alors ce texte est inséré dans une boîte de largeur `\labelwidth`. Dans le cas contraire, comme ici, le texte de l'étiquette sera inséré dans une boîte de la largeur nécessaire et le paragraphe sera indenté en conséquence.



La commande `\makeatlabel` prenant un argument, permet de produire l'étiquette. Ainsi lorsqu'on entre la commande `\item[texte étiquette]`, il est fait appel à la commande `\makeatlabel{texte étiquette}`.

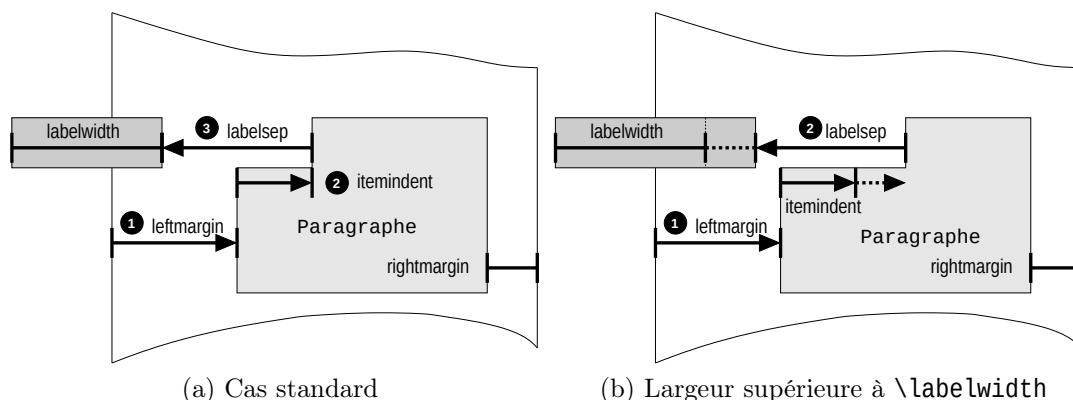


FIG. 9.1 – Positionnement de l'entrée de liste

9.5.2 Réglage de l'étiquette

Pour comprendre le fonctionnement de l'environnement `list` et plus particulièrement le principe du positionnement relatif de l'étiquette et du paragraphe adjacent, on peut imaginer que les éléments sont positionnés dans l'ordre suivant :

1. le paragraphe est d'abord positionné par rapport à la marge de gauche à l'aide de la longueur `\leftmargin` ;
2. la première ligne du paragraphe est ensuite indenté à l'aide de la longueur `\itemindent` ;
3. puis l'étiquette est positionnée *relativement* au début du paragraphe ainsi indenté à l'aide de la longueur `\labelsep`.

Il découle de ceci que l'entrée de liste (ou étiquette) peut être produite dans la marge de gauche... La figure 9.1 illustre le positionnement de l'entrée de liste par rapport au paragraphe dans les deux situations suivantes :

- cas de la figure 9.1a où la largeur de l'entrée de liste est inférieure à la dimension `\labelwidth`. Dans ce cas l'entrée de liste est positionnée à une distance de `\labelsep` du paragraphe, lui même étant indenté de `\itemindent` et positionné par rapport à la marge gauche à l'aide de `\leftmargin` ;
- cas de la figure 9.1b : la largeur de l'entrée de liste est supérieure à la dimension `\labelwidth`. Dans ce cas l'entrée de liste est toujours positionnée à la distance `\labelsep` du paragraphe, mais celui-ci est indenté d'une valeur supérieure à `\itemindent`.

9.5.3 Réglages verticaux

On peut également régler les blancs verticaux dans l'environnement `list`. Ces paramètres permettent notamment de définir les espaces entre les paragraphes constituant les entrées de liste, mais également les blancs que l'on veut insérer avant ou après la liste. Il s'agit de :

- `\itemsep` : l'espace entre chaque entrée de liste ;
- `\parsep` : l'espace entre deux paragraphes à l'intérieur d'une entrée de liste ;
- `\topsep` : le blanc inséré avant et après l'environnement créé, auquel est ajouté `\partopsep` si celui-ci commence un nouveau paragraphe.

9.5.4 Valeurs par défaut

Tous les paramètres de l'environnement `list` ont des valeurs par défaut. Les longueurs pour les réglages horizontaux sont par défaut les suivantes sur le système de votre serveur :

dimension	valeur par défaut
<code>\itemindent</code>	0pt
<code>\listparindent</code>	0pt
<code>\rightmargin</code>	0pt
<code>\leftmargin</code>	25pt
<code>\labelwidth</code>	20pt
<code>\labelsep</code>	5pt

Et pour les réglages verticaux :

dimension	valeur par défaut
<code>\itemsep</code>	4.0pt plus 2.0pt minus 1.0pt
<code>\parsep</code>	4.0pt plus 2.0pt minus 1.0pt
<code>\topsep</code>	8.0pt plus 2.0pt minus 4.0pt
<code>\partopsep</code>	2.0pt plus 1.0pt minus 1.0pt

La commande `\makelabel` est quant à elle définie par :

```
\hfil #1
```

par conséquent, dans la boîte de largeur `\labelwidth`, le contenu de l'étiquette est poussé à droite. Ainsi si on définit une liste simple avec :

```
\newenvironment{listebasique}
{\begin{list}{}{}}
{\end{list}}
```

On aura :

```
Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
\begin{listebasique}
  \item[X] o o o o o o o o o o o o o o o o
           o o o o o o o o o o o o o o o o

           u u u u u u u u u u u u u u u u u
  \item[Machin] v v v v v v v v v v v v v v v
\end{listebasique}
Après après après après après après après
après après après après après après après
```

avec des étiquettes, et sans étiquette :

```
Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
\begin{listebasique}
  \item[] e e e e e e e e e e e e e e e e
          e e e e e e e e e e e e e e e e
\end{listebasique}
Après après après après après après après
après après après après après après après
```

```
Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
X o o o o o o o o o o o o o o o o o o o o o
  o o o o o o o o o o o o o o o o
  u u u u u u u u u u u u u u u u u u u u u
Machin v v v v v v v v v v v v v v v
Après après après après après après après
après après après après après après après
```

```
Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
e e e e e e e e e e e e e e e e e e e e e
e e e e e e e e e e e e e e e e e e e e e
Après après après après après après après
après après après après après après après
```

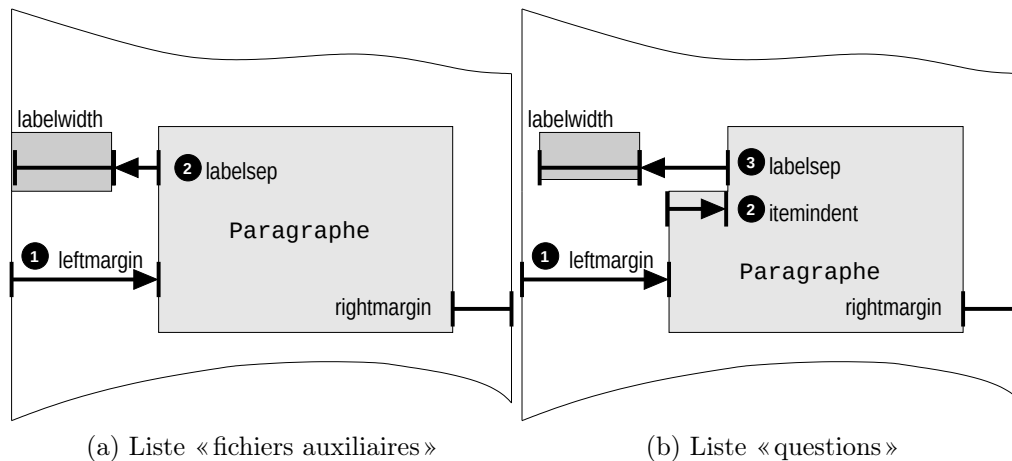


FIG. 9.2 – Positionnement des étiquettes dans les listes exemples.

9.5.5 Exemples

La liste décrivant les fichiers auxiliaires de $\text{\LaTeX}$ située à la page 196 a été produite avec le code suivant :

<pre>\begin{ficaux} \item[tex] fichier source \LaTeX{} ; blabla blabla blabla blabla blabla blabla blabla blabla blabla blabla blabla \item[aux] fichier auxiliaire ... \item[log] le fichier de trace ... \item[dvi] fichier \emph{device independant},... \end{ficaux}</pre>	tex fichier source $\text{\LaTeX}$; blabla blabla blabla blabla blabla blabla blabla blabla blabla aux fichier auxiliaire ... log le fichier de trace ... dvi fichier <i>device independant</i>,...
--	--

L'environnement `ficaux` a lui été conçu comme suit :

```
\newenvironment{ficaux}{%
  \begin{list}{}{%
    \setlength{\labelwidth}{1cm}% largeur de la boîte englobant le label
    \setlength{\labelsep}{8pt}% espace entre paragraphe et l'étiquette
    \setlength{\leftmargin}{\labelwidth+\labelsep}% marge de gauche
    \renewcommand{\makelabel}[1]{% production de l'étiquette :
      \framebox[\labelwidth]{\texttt{##1}}}}{\end{list}}
```

Dans cet exemple la relation :

```
\leftmargin=\labelwidth+\labelsep
```

permet de positionner l'entrée de liste comme indiqué à la figure 9.2a.

Un autre exemple : la création d'un environnement de liste numérotée pour produire des questions dans un énoncé de travaux pratiques ou autre devoir surveillé...

<code>\begin{question}</code>	
<code>\item 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$/_1 \quad 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0$
<code>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0$
<code>\item 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$/_2 \quad 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0$
<code>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0$
<code>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$0 0 0 0 0 0 0 0 0 0 0 0$
<code>\item 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$/_3 \quad 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0$
<code>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</code>	$0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0$
<code>\end{question}</code>	

Cet environnement a été produit par le code suivant :

```
\newenvironment{question}{\begin{list}}{\end{list}}{\%
  \usecounter{cptquestion}%
  \setlength{\labelwidth}{2em}%
  \setlength{\labelsep}{1em}\setlength{\itemindent}{15pt}%
  \setlength{\leftmargin}{.8cm}\setlength{\rightmargin}{10pt}
  \renewcommand{\makelabel}[1]{\etiquettequestion{##1}}}
```

Le positionnement correspondant est montré à la figure 9.2b page précédente. On notera que dans la définition de la liste est fait usage de la commande `\usecounter` permettant de créer une liste numérotée et de préciser quel compteur est utilisé. On devra donc déclarer le compteur en question :

```
\newcounter{cptquestion}
```

Enfin, chaque entrée de liste composée du numéro de la question et d'un «joli» crayon est produite par la commande :

```
\newcommand{\etiquettequestion}[1]{\%
  \makebox[\labelwidth]{\Pisymbol{pzd}{47}$\_thecptquestion$}}
```

Finalement la commande :

```
\renewcommand{\makelabel}[1]{\etiquettequestion{##1}}}
```

redéfinit la commande `\makelabel` comme faisant appel à notre «joli» crayon. Le premier et unique argument est passé à `\etiquettequestion` à l'aide de l'expression `##1`, car `#1` désignerait, dans le contexte de la définition de environnement `question`, le premier argument de celui-ci.

Dans ce manuel, on trouve dans le mémento une liste de packages (page 195) produite par le code suivant :

```
\newenvironment{packages}{\begin{list}}{\end{list}}{\%
  \setlength{\labelwidth}{2.5cm}%
  \setlength{\itemindent}{0pt}%
  \setlength{\leftmargin}{\labelwidth+\labelsep}%
  \renewcommand{\makelabel}[1]{\%
    $\blacktriangle$ \ltxpack{##1} \hfil:}}{\end{list}}
```

La commande `\ltxpack` est définie au paragraphe 11.1.2 page 154. Pour information, l'environnement ci-dessus donne :

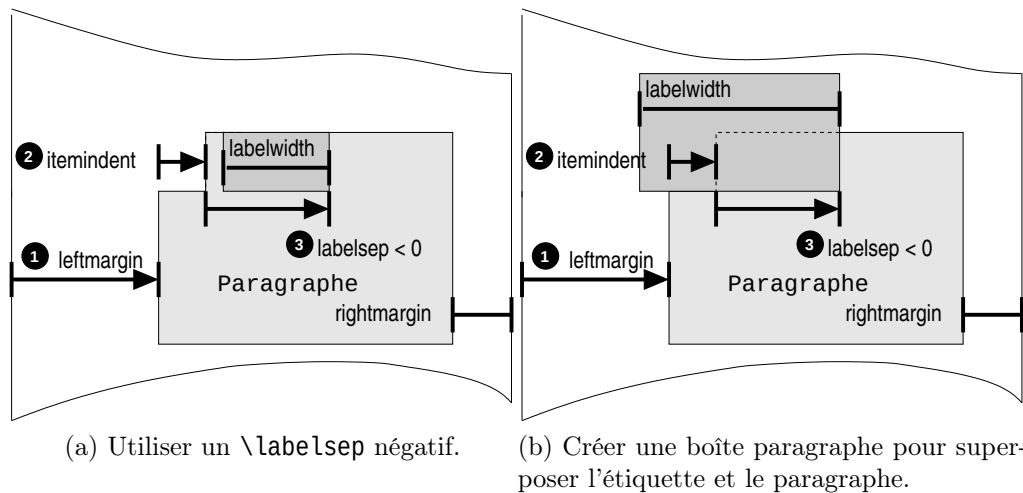


FIG. 9.3 – Positionnement de l'étiquette dans la liste « glossaire ».

<pre>\begin{packages} \item[bidule] cette extension permet d'insérer des bidules dans son document sans avoir à savoir s'il s'agit de machin ou de truc. \end{packages}</pre>	9.20	N bidule : cette extension permet d'insérer des bidules dans son document sans avoir à savoir s'il s'agit de machin ou de truc.
---	---	---

9.5.6 Un exemple un peu plus tordu...

Nous allons détailler dans cette section la manière dont la liste composant le glossaire de la page 217 a été générée. Cette liste dont l'allure est donnée ci-dessous exploite deux idées :

1^{re} idée

La longueur `\labelsep` peut être négative ce qui permet de superposer l'entrée de liste avec le paragraphe. Ceci est illustré à la figure 9.3a ;

Deuxième idée

On peut créer une boîte paragraphe pour la boîte produisant l'étiquette. La boîte ainsi créée pourra donc être composée de deux lignes : celle du haut contenant le texte de l'étiquette, celle du dessous étant vide, se superpose avec le paragraphe (figure 9.3b).

Le code permettant de générer la liste ci-dessus et celle du glossaire de ce manuel est donc :

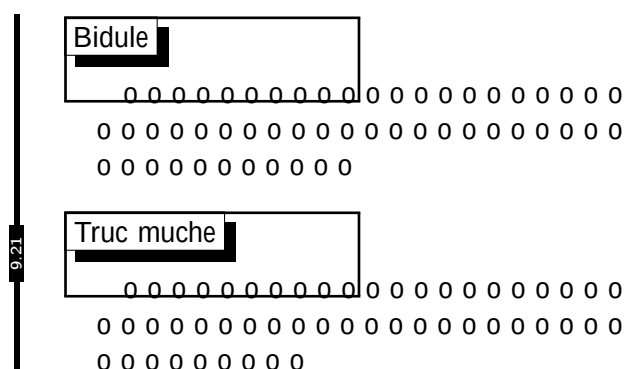
```
\newenvironment{glossaire}{\begin{list}}{\end{list}}{
\setlength{\labelwidth}{.5\textwidth}%
\setlength{\labelsep}{-.8\labelwidth}%
\setlength{\itemindent}{\parindent}%
\setlength{\leftmargin}{25pt}%
\setlength{\rightmargin}{0pt}%
\setlength{\itemsep}{.8\baselineskip}%
\renewcommand{\makelabel}[1]{\boiteentreeglossaire{##1}}}
\end{list}}
```

La valeur `.8\baselineskip` pour `\itemsep` permet d'aérer le glossaire en insérant des blancs entre chaque entrée. La commande permettant de générer la boîte contenant l'entrée de liste à superposer est :

```
\newcommand{\boiteentreeglossaire}[1]{%
  \parbox[b]{\labelwidth}{%
    \setlength{\fboxsep}{3pt}%
    \setlength{\fboxrule}{.4pt}%
    \shadowbox{\sffamily#1}\hfill\mbox{}}}
```

Pour comprendre comment est positionnée cette boîte, nous avons quelque peu modifié la commande précédente pour dessiner un cadre autour :

```
\begin{leglossaire}
  \renewcommand{\boiteentreeglossaire}{%
    \fboiteentreeglossaire}%
  \item[Bidule] o o o o o o o o o o o o o o o
    o o o o o o o o o o o o o o o o o o o
    o o o o o o o o o o o o o o o o o o o
  \item[Truc muche] o o o o o o o o o o o o o
    o o o o o o o o o o o o o o o o o o o
    o o o o o o o o o o o o o o o o o o o
\end{leglossaire}
```



9.6 Des environnements qui mettent en boîte

L'environnement `lrbox` qui fait l'objet de ce paragraphe permet de stocker le « contenu » d'un environnement dans une boîte. C'est un outil très pratique qui est souvent une solution adaptée à des problèmes courants comme l'encadrement d'objets. Nous allons voir sur un exemple comment on peut utiliser cette construction.

9.6.1 Principe

On déclare une boîte comme expliqué au paragraphe 4.4.5 page 62 :

```
\newsavebox{maboite}
```

On peut ensuite écrire :

```
\begin{lrbox}{maboite}...contenu...\end{lrbox}
```

qui est parfaitement équivalent à :

```
\savebox{maboite}{...contenu...}
```

Bon mais alors si c'est la même chose, à quoi ça sert ?? Ça sert pour la *définition d'un environnement*.

9.6.2 Exemple

Supposons, par exemple l'existence d'une environnement :

```
\newenvironment{remarque}{%
  % clause begin
  \begin{center}\begin{minipage}{.8\textwidth}}{%
```

```
% clause end
\end{minipage}\end{center}}
```

Et :

On peut faire la remarque (édifiante) suivante :

```
\begin{remarque}
  Jazz is not dead. It just smells funny.
  \hfill Frank \textsc{Zappa}
\end{remarque}
```

Et on reprend notre propos.

On peut faire la remarque (édifiante) suivante :

Jazz is not dead. It just smells funny.
Frank ZAPPA

Et on reprend notre propos.

À la question : *comment pourrait-on faire pour encadrer cette remarque ?* nous répondons sans hésiter : « avec l'environnement `lrbox` ». Dans la mesure où il n'y a pas de moyen avec $\text{\LaTeX}$ de commencer une `\fbox` dans la clause `begin` et de la finir dans la clause `end`, on utilisera le principe suivant :

1. stocker le contenu à encadrer à l'aide de l'environnement `lrbox`
2. l'encadrer en réutilisant la boîte ainsi construite.

On aura donc :

```
\newsavebox{\boiteremarque}
\newenvironment{remarque}{%
  % clause begin
  \begin{lrbox}{\boiteremarque}% début mise en boîte
  \begin{minipage}{.8\textwidth}}{%
  % clause end
  \end{minipage}
  \end{lrbox}% fin mise en boîte
  % production de la boîte encadrée
  \begin{center}
    \fbox{\usebox{\boiteremarque}}
  \end{center}}
```

e

On peut faire la remarque (édifiante) suivante :

```
\begin{remarque}
  Jazz is not dead. It just smells funny.
  \hfill Frank \textsc{Zappa}
\end{remarque}
```

Et on reprend notre propos.

On peut faire la remarque (édifiante) suivante :

Jazz is not dead. It just smells funny.
Frank ZAPPA

Et on reprend notre propos.

Cette idée est utilisée à plusieurs reprises dans les deux chapitres qui suivent.



Il faut noter que la boîte que définit l'environnement `lrbox` est une **boîte simple** à l'instar des boîtes créées par les commandes de la famille `\mbox` et par conséquent ne peut contenir de saut de paragraphe.

10

Cosmétique

Sommaire

- 10.1 Allure de l'index
- 10.2 Allures des titres
- 10.3 Géométrie
- 10.4 En-tête et pied de page
- 10.5 Environnements avec caractères spéciaux
- 10.6 About those so called “french guillemets”
- 10.7 Une boîte pour la minitable des matières

*Je me dis: Je monterai sur le palmier, J'en saisirai les rameaux!
Que tes seins soient comme les grappes de la vigne,
Le parfum de ton souffle comme celui des pommes,
Et ta bouche comme un vin excellent qui coule aisément pour mon bien-aimé,
Et glisse sur les lèvres de ceux qui s'endorment!*

Le Cantique des cantiques Ct 7 10.

L'IDÉE GÉNÉRALE de ce chapitre, laissant présager des macros parfumées, est de présenter les outils standard de L^AT_EX qui ont été personnalisés pour produire certaines parties du document. Ces personnalisations s'entendent à plusieurs niveaux : en utilisant des options de packages (par exemple pour les en-têtes de page), ou parfois en «mettant le nez» dans la définition des macros, comme pour l'allure des chapitres et des sections, ou en modifiant plus en profondeur ces macros comme dans le cas de la minitable des matières. Une partie du chapitre est consacrée aux outils que l'on peut mettre au point à partir du package fancyvrb. Enfin un combat en règle contre les guillemets français est mené en guise de clôture de ce chapitre.

10.1 Allure de l'index

Pour changer l'allure de l'index, il faut comprendre que lorsqu'on tape fébrilement avec ses petits doigts la commande :

`| makeindex document`

on génère alors un fichier `document.ind` contenant quelque chose ressemblant à :

```
\begin{theindex}                                ← préambule

\item Cosmic debries, 12,34

\indexspace                                     ← espace inter-groupe
```

`\item Debra kadabra, 23`

← entrée, séparateur, page

`\end{theindex}`

← postamble

En réalité, ce code est généré à partir d'entités génériques ayant des valeurs prédéfinies et pouvant être modifiées. Pour s'en convaincre, il suffit de savoir que le programme `makeindex` peut générer un fichier `.ind` contenant autre chose que du code $\text{\LaTeX}$. Pour comprendre cette affaire d'entités génériques, on pourrait décrire le travail de `makeindex` comme suit :

1. Écrire le préambule en examinant la valeur de l'entité `preamble` ;
2. Pour chaque entrée du fichier `.idx` :
 - (a) écrire le contenu de l'entité `item_0` ;
 - (b) écrire l'entrée (« Cosmic debries » dans l'exemple précédent) ;
 - (c) écrire le séparateur (valeur de l'entité `delim_0`) ;
 - (d) écrire le numéro de page
3. À chaque fin de groupe (changement de lettre) écrire le contenu de l'entité `group_skip` ;
4. Écrire le postamble en examinant la valeur de l'entité `postamble`.

Les valeurs des entités auxquelles il est fait allusion sont par défaut les suivantes :

<code>preamble</code>	<code>"\begin{theindex}\n"</code>
<code>item_0</code>	<code>"\n \item"</code>
<code>delim_0</code>	<code>", "</code>
<code>group_skip</code>	<code>"\n\n \indexspace\n"</code>
<code>postamble</code>	<code>"\n\n\end{theindex}\n"</code>

Ces valeurs peuvent être modifiées par l'intermédiaire d'un fichier de style auquel on met généralement l'extension `.ist` et que l'on utilisera lors de l'appel à `makeindex` de la manière suivante :

| `makeindex -s style.ist document`

Ainsi pour produire l'index de ce document, nous avons dans un premier temps redéfini les séparateurs de niveau 1 et 2 :

```
delim_0 " \dotfill \ "
delim_1 " \dotfill \ "
```

on remplace donc la virgule qui sépare par défaut l'entrée d'index et son numéro de page par des points de suspension. Ensuite, en lisant scrupuleusement la documentation `makeindex`,¹ on apprend qu'écrire :

```
headings_flag 1
```

est la manière polie de demander à `makeindex` de produire entre les groupes d'entrées la lettre correspondant au groupe. Cette lettre sera (en majuscule) et encadrée par les contenus respectifs des entités `heading_prefix` et `heading_suffix`. Qu'à cela ne tienne, pour produire nos jolies boîtes ombrées, nous écrivons dans le fichier de style :

```
heading_prefix "{\large\sffamily\bfseries\shadowbox{"
heading_suffix "}\hfill}\nopagebreak\n"
```

¹Voir également le nota qui suit pour les références bibliographiques utiles.

Ce qui vous l'avez compris, produira par exemple pour la lettre «c» :

```
{\large\sffamily\bfseries%
\shadowbox{C}\hfil}\nopagebreak
```



Cette commande sera précédée par le contenu de `group_skip` qui, nous l'avons dit un peu plus haut, vaut par défaut `\indexspace`. Nous avons après quelques mois de recherche,² déniché la définition de cette commande dans `book.cls` et l'avons modifiée pour augmenter légèrement l'espace entre les groupes :

```
\renewcommand\indexspace{\par \vskip 20pt plus5pt minus3pt\relax}
```



Ce paragraphe ne donne bien évidemment qu'un aperçu très succinct des fonctionnalités proposées par `makeindex`. Outre les informations que l'on peut trouver dans le `LATEX` companion, la page de manuel de cet utilitaire dans un environnement Debian donne une liste exhaustive des entités génériques que l'on peut définir. Un fichier nommé `ind.dvi` écrit par P. CHEN et M. A. HARRINSON constitue également un bon point de départ pour la personnalisation de l'index.

10.2 Allures des titres

Nous proposons ici d'exposer la manière dont on a modifié l'allure des titres standard (partie, chapitre, section, etc.) de `LATEX`.

10.2.1 Numérotation des titres et table des matières

Avant toute chose il faut savoir qu'on peut agir sur la table des matières à l'aide de deux compteurs :

1. `secnumdepth` (*section numbering depth*) stipulant la profondeur de la numérotation des titres dans le document ;
2. `tocdepth` (*table of contents depth*) définissant quel est le niveau (ou profondeur) de titre maxi dans la table des matières.

Pour utiliser ces deux compteurs, il faut en outre avoir connaissance de la manière dont `LATEX` associe une profondeur à chaque titre. Et bien réjouissez-vous, le tableau suivant vous fournit cette information :

Titre	profondeur	Titre	profondeur
part	-1		
chapter	0	subsubsection	3
section	1	paragraph	4
subsection	2	subparagraph	5

Ainsi, affecter 1 à `secnumdepth` et 2 à `tocdepth` numérottera les titres jusqu'aux \sections et insérera dans la table des matières, toutes les titres jusqu'aux \subsection...

²Je plaisante, juste quelques semai... minutes veux-je dire.

10.2.2 Sections et niveaux inférieurs

Dans le fichier `book.cls` du système $\text{T}_{\text{E}}\text{X}$, on trouve le code suivant :³

```
\newcommand{\section}{%
  \@startsection%
  {section}% nom du titre
  {1}% niveau de titre
  {0pt}% indentation
  {-3.5ex plus -1ex minus -.2ex}% espace vertical avant
  {2.3ex plus.2ex}% espace vertical après
  {\normalfont\Large\bfseries}} % allure du titre
```

Ce code permet de définir comment sera produit le titre d'une section. On constate que la commande `\section` fait appel à la commande `\@startsection`, cette dernière attendant six arguments :

- le nom du titre : `section`, `subsection`, etc.
- son niveau : 1 pour `section`, 2 pour `subsection`, 3 pour `subsubsection`, etc.
- son indentation ;
- le blanc vertical avant le titre ;
- le blanc vertical après le titre ;
- un ensemble de *déclarations* pour formater le titre lui-même.

On pourra donc noter que la mise en page par défaut de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ pour les sections dans la classe `book` est la suivante :

- pas d'indentation (0pt)
- espace avant le titre de 3.5ex avec un tolérance de plus -1ex et moins -.2ex ;
- espace après le titre de 2.3ex avec une tolérance de plus .2ex ; on pourra noter que si l'espace est négatif, le paragraphe commence juste après le titre, et non sur un nouveau paragraphe ;
- les titres sont en gros et en gras dans la fonte « normale ».

Pour définir l'allure des sections de ce document, nous avons introduit trois longueurs pour l'indentation de sections, subsections et subsubsections :

```
\newlength{\sectiontitleindent}
\newlength{\subsectiontitleindent}
\newlength{\subsubsectiontitleindent}
```

Ces longueurs ont pour valeur :

```
\setlength{\sectiontitleindent}{-1cm}
\setlength{\subsectiontitleindent}{-.5cm}
\setlength{\subsubsectiontitleindent}{-.25cm}
```

D'autre part, nous avons défini une fonte particulière pour les titres, définie comme suit :

```
\newcommand{\sectionfont}{%
  \fontencoding{\encodingdefault}%
  \fontfamily{pag}%
  \fontseries{bc}%
  \fontshape{n}%
  \selectfont}
```

³Légèrement simplifié...

Cette commande permet de sélectionner la fonte PostScript Avant-Garde en gras condensé (cf. 9.4 page 114). Finalement, pour définir l'allure de nos sections on utilisera :

```
\renewcommand{\section}{%
  \@startsection%
  {section}%
  {1}%
  {\sectiontitleindent}%
  {-3.5ex plus -1ex minus -.2ex}%
  {2.3ex plus.2ex}%
  {\sectionfont\Large}}
```

Des commandes équivalentes ont été écrites pour les titres de niveaux inférieurs.

10.2.3 Chapitres

C'est en fouillant dans le fichier `book.cls` qu'on peut trouver des informations sur la manière dont L^AT_EX produit les en-têtes de chapitres.

Principe

Dans le fichier `book.cls`, on trouve la commande :

```
\newcommand{\chapter}{%
  ...
  \thispagestyle{plain}%
  ...
  \secdef\@chapter\@schapter} % la ligne qui nous intéresse
```

La commande `\chapter` fait donc elle-même appel à deux commandes distinctes :

1. `\@chapter` pour les titres de chapitre qui sont numérotés ;
2. `\@schapter` pour les titres de chapitre non numérotés (s pour *star* ou étoile faisant référence à la commande `\chapter*`).

En cherchant vaillamment la définition de ces deux commandes (toujours dans le fichier `book.cls`), on trouve quelque chose du genre :

```
\def\@chapter[#1]#2{%
  ...
  \refstepcounter{chapter}%
  \typeout{\@chapapp\space\thechapter.}% message sur le terminal
  \addcontentsline{toc}{chapter}% ajout du titre dans la toc
  ...
  \if@twocolumn
  ...
  \else% le cas d'un document à une colonne
  \@makechapterhead{#2}% la ligne qui nous intéresse
  \fi}
```

ce qui nous met sur la voie... en effet la commande `\@makechapterhead` (qu'on peut traduire littéralement par « faire l'en-tête de chapitre ») est la commande qu'il va nous falloir redéfinir pour changer l'allure des en-têtes. Une recherche supplémentaire

nous met également sur la piste de la commande `\@makeschapterhead` produisant l'en-tête d'un chapitre non numéroté. Ces deux commandes attendent un argument qui est le titre du chapitre.

Petits outils nécessaires

Nous avons défini un environnement `cadrechap` dont le propos est simplement d'élargir la marge de droite de deux centimètres :

```
\newenvironment{cadrechap}%
{\begin{list}{}{}%
  \setlength{\leftmargin}{0pt}%
  \setlength{\rightmargin}{-2cm}% on se met au large
  \setlength{\itemindent}{0pt}%
  \setlength{\labelsep}{0pt}%
  }\item}%
{\end{list}}
```

Il sera également fait usage du booléen `@mainmatter` permettant de savoir si on se trouve dans la partie «centrale» du document. C'est le cas lorsque la commande `\mainmatter` a été appelée (cf. 6.4 page 84).

En-tête des chapitres à proprement parler

Pour ce manuel, nous avons redéfini la commande `\@makechapterhead` comme suit :

```
\renewcommand{\@makechapterhead}[1]{%
  \begin{cadrechap}
    \if@mainmatter
      {\chapnumfont\thechapter}\hfill{\chapfont#1}
    \else
      \mbox{} \hfill{\chapfont#1}
    \fi
  \end{cadrechap}
  \vspace{1cm}}
```

Par conséquent, si on est dans le «`\mainmatter`» :

- on affiche le numéro du chapitre (`\thechapter`) dans une fonte particulière (`\chapnumfont`) ;
- on insère un ressort horizontal (`\hfill`) ;
- on affiche le titre du chapitre (`#1`) dans une police particulière (`\chapfont`).

Si on n'est pas dans la partie centrale, on n'affiche pas de numéro de chapitre, mais juste le titre poussé à droite. Les commandes permettant de mettre en évidence le numéro du chapitre et son titre sont respectivement :

```
\DeclareFixedFont{\chapnumfont}{T1}{phv}{bc}{n}{80pt}
\DeclareFixedFont{\chapfont}{T1}{phv}{b}{n}{24.88pt}
```

Finalement, si on écrit :

```
\begin{cadrechap}
  {\chapnumfont7}\hfill{\chapfont Le titre de la mort qui tue}
\end{cadrechap}
```

On obtient :

7

Le titre de la mort qui tue

10.2.4 Parties

Dans le fichier `book.cls` on trouve la définition de la commande `\part` :

```
\newcommand\part{%
  \cleardoublepage
  \thispagestyle{plain}
  [...]
  \null\vfil
  \secdef\@part\@spart}
```

qui nous informe qu'à l'instar des chapitres, la commande `\part` fait appel à deux commandes distinctes pour produire les parties numérotées et non numérotées (grâce à un appel aux commandes `\@part` et `\@spart` respectivement). Dans un premier temps nous avons imposé que le style de page pour les débuts de partie soit vide (c'est-à-dire sans numéro de page ni en-tête etc.), nous avons donc écrit :

```
\newcommand\part{%
  \cleardoublepage
  \thispagestyle{empty}% à la place de plain par défaut
  [...]
  \null\vfil% un boîte vide et un ressort vertical
  \secdef\@part\@spart}
```

On peut ensuite examiner la définition de la commande `\@part` qui produit la page de partie :

```
\def\@part[#1]#2{%
  [...]
  {\centering % centrage
  [...]
  \huge\bfseries \partname\nobreakspace\thepart
  \par
  \vskip 20\pt
  [...]
  \Huge \bfseries #2\par}%
  \@endpart}
```

En examinant ce code on constate que la page de partie est constituée d'une ligne en gros caractères gras, du nom « Partie » suivie du numéro de la partie :⁴

```
\huge\bfseries \partname\nobreakspace\thepart
```

suivie 20 points plus bas du titre de la partie (contenu dans l'argument #1). Pour ce manuel, nous avons redéfini la commande `\@part` comme suit :

⁴En réalité, après avoir enclenché le package `babel` et l'option `french` ces deux commandes sont redéfinies pour produire quelque chose du style : « Première partie »

```

\def\@part[#1]#2{%
  [...]
  {\centering
   \interlinepenalty \@M
   \normalfont
   [...]
   \partnumfont \thepart % juste le numéro de la partie
   \par
   \vskip 50\p@% 50 point au lieu de 20...
   \partfont #2\par}% le titre avec une fonte personnalisée
\@endpart}

```

Pour garder une homogénéité avec les en-têtes de chapitre on a défini les commandes de fontes comme suit :

```

\newcommand{\partfont}{%
  \fontencoding{\encodingdefault}\fontfamily{phv}%
  \fontseries{bc}\fontshape{n}%
  \fontsize{32}{34}%
  \selectfont}
\DeclareFixedFont{\partnumfont}{T1}{phv}{bc}{n}{80}%

```



On notera également que la commande `\@part` se termine par l'appel à une autre commande : `\@endpart`. En examinant le fichier `book.cls` on pourra se rendre compte que cette dernière permet de s'opposer au ressort vertical de la commande `\part` et de sauter une page blanche...

10.3 Géométrie

Les différentes dimensions de chaque page de ce document ont été définies à l'aide du package `geometry` et de la commande :

```

\geometry{%
  a4paper,
  body={150mm,250mm},
  left=25mm,top=25mm,
  headheight=7mm,headsep=4mm,
  marginparsep=4mm,
  marginparwidth=27mm}

```

qui définit respectivement (voir aussi figure 10.1 page suivante) :

- un corps de texte faisant 150 mm de largeur par 250 mm ;
- le positionnement du corps du texte dans la page, à 25 mm du bord gauche du papier, et 25 mm du bord supérieur du papier ;
- la hauteur de l'en-tête (7mm) et l'espace entre l'en-tête et le texte lui-même (4 mm) ;
- la taille du papier : standard A4 ;
- la largeur de la marge pour les notes de marges (2.7 cm).

De manière générale, comme le montre la figure 10.1 page ci-contre, le package `geometry` permet de définir un certain nombre de dimensions que l'on peut passer soit en option à la commande `\usepackage` soit à l'aide de la commande `\geometry`.

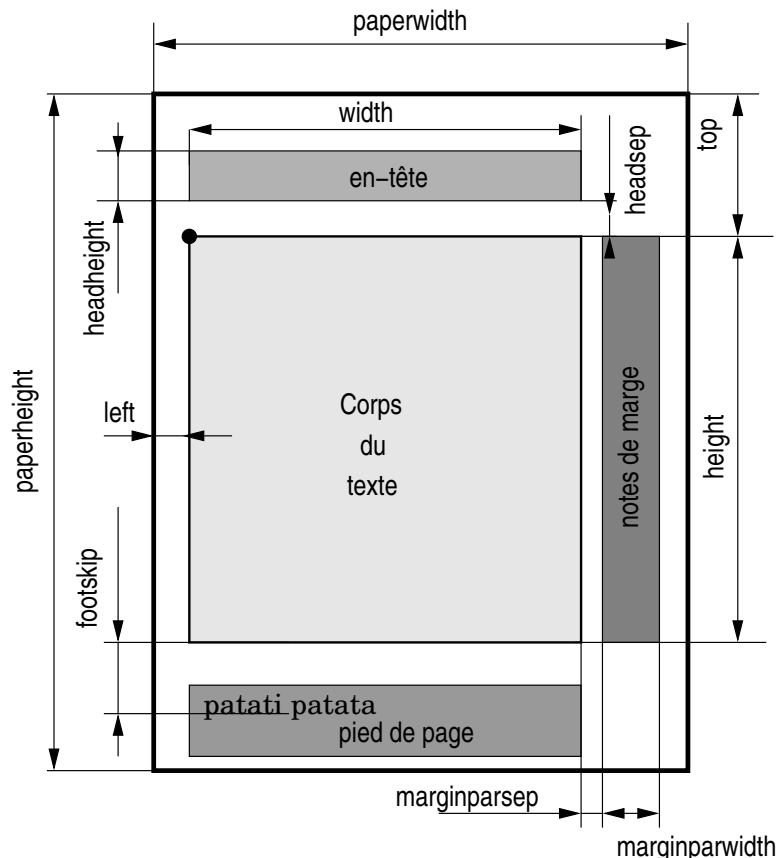


FIG. 10.1 – Quelques unes des dimensions pour définir la géométrie d'un document.

Dimension du papier :

- a4paper, a5paper, etc. pour utiliser un format de papier prédéfini,
- paperwidth=**dim** et paperheight=**dim** pour spécifier une dimension de papier libre, par exemple pour un document qui sera massicoté.

Texte :

- soit avec : body={**largeur**, **hauteur**}
- soit avec : width=**largeur** et height=**hauteur**.
- le texte est positionné à l'intérieur de la page par rapport à un point de référence spécifié avec top=**pos_vert** et left=**pos_horiz**

Haut et bas de page :

- la hauteur de la surface réservée à l'en-tête peut être définie à l'aide de la formule magique headheight=**hauteur** et sa position par rapport au corps du texte à l'aide de headsep=**espace**.
- la position du pied de page peut être définie à l'aide de footskip=**espace** qui impose l'espace entre le bas du corps du texte et la première ligne du contenu du pied de page.

Note de marge dans le même esprit la largeur et la position de la surface réservée aux notes de marge peuvent être définies à l'aide de marginparwidth=**largeur** et marginparsep=**espace**.



Dans le package geometry les dimensions concernant l'en-tête, le pied de page et la zone pour les notes de marge, sont par défaut comptabilisées *en plus* du corps du texte. Des options permettent d'inclure l'une ou l'intégralité de ces dimensions dans

le corps du texte pour le calcul, en disant par exemple : « je veux que la largeur soit de 10 centimètres, notes de marge comprises. » (voir la documentation du package pour les détails).

10.4 En-tête et pied de page

Les zones au-dessus et en dessous du corps du texte appelées en-tête et pied de page peuvent être personnalisées à l'aide du package fancyhdr. Le principe de base est simple,⁵ il suffit d'utiliser la commande :

```
\pagestyle{fancy}
```

pour spécifier qu'on veut utiliser des en-têtes et des pieds de page définis grâce au package fancyhdr. Par défaut le package produit des traits horizontaux en dessous de l'en-tête et au-dessus du pied de page dont les épaisseurs sont définies par les commandes \footrulewidth et \headrulewidth. On peut ensuite utiliser les commandes :

- \fancyhead pour définir l'en-tête ;
- \fancyfoot pour définir le pied de page ;

Ces deux commandes peuvent prendre un argument optionnel constitué d'une ou deux séquences des caractères suivants :

- E ou O pour spécifier la parité de la page (paire=*even*, impaire=*odd*) ;
- R, L, C pour spécifier où l'on veut produire l'information : respectivement à droite, à gauche ou au centre ;

Voici un exemple :

```
\fancyhf{} % on efface tout et on recommence
% EN TÊTE :
% initiales à droite sur page paire, à gauche sur page impaire :
\fancyhead[RE,LO]{VL}
% numéro de page au centre :
\fancyhead[C]{\thepage}
% numéro de section à droite sur page impaire, à gauche sur page paire :
\fancyhead[LE,R0]{\thesection}
% PIED DE PAGE :
% une image à droite sur page impaire, à gauche sur page paire :
\fancyfoot[R0,LE]{\includegraphics[height=4ex]{punch}}
% titre à gauche sur page impaire, à droite sur page paire :
\fancyfoot[LO,RE]{%
  Tout ce que vous avez toujours voulu savoir sur \LaTeX{}
}
% épaisseur des traits
\renewcommand{\footrulewidth}{3pt}
```

10.4.1 Cas de la première page des chapitres

Dans la classe book, L^AT_EX fait automatiquement appel au style plain pour les premières pages de chapitre. Pour demander au package fancyhdr de définir un style particulier pour ces pages, on écrit :

⁵Vous ne serez sans doute pas tout à fait d'accord avec le terme « simple » après avoir lu la suite...



`\markright{texte}`

Nous allons imaginer que les arguments `textex` sont stockés dans une pile et une file. Dans cet ordre d'idée :

- `\markboth` empile `texteg`, et stocke `texted` dans la file ;
- `\markright` stocke `texte` dans la file.

Ces deux commandes de « marquage » peuvent être appelées plusieurs fois ou jamais, sur une même page. Les données de la pile et de la file seront exploitées au moment de générer les en-têtes et pied de page, lorsque T_EX achève la mise en forme de la page, et ceci grâce aux commandes :

- `\leftmark` renvoie le sommet de la pile, c'est-à-dire `texteg` du *dernier appel* à `\markboth` ;
- `\rightmark` renvoie le début de la file, c'est-à-dire `texted` du *premier appel* à `\markboth` ou `texte` du *premier appel* à `\markright`.



Une petite subtilité au sujet de la « file » que nous présentons ici : tant qu'aucune commande de « marquage » n'ajoute de données au cours d'une page, la file contiendra *la dernière information insérée* dans les pages précédentes. La « file » est vidée dès qu'une commande `\markboth` ou `\markright` survient.

Un autre moyen de comprendre ce mécanisme de marqueurs pourrait être de dire :

- `\leftmark` contient la dernière information que j'ai insérée sur la pile (à l'aide du premier argument de `\markboth`) ;
- `\rightmark` contient la première information de la « file », si on en a mis une sur cette page, ou la dernière qui a été enfilé (à l'aide du deuxième argument de `\markboth` ou de l'argument de `\markright`).



Il peut être utile de savoir que l'auteur a utilisé ces commandes pour la production d'un trombinoscope composé de plusieurs dizaines de noms et photos par page. L'idée était d'exploiter le mécanisme de marquage pour faire apparaître dans l'en-tête le premier et le dernier nom de la page, comme dans un dictionnaire. Il suffit pour cela d'appeler pour chaque personne (nom et photo) la commande :

```
\markboth{nom du gugusse}{nom du gugusse}
```

puis d'insérer dans les en-têtes la commande `\rightmark` sur les pages de gauche (impaires) et `\leftmark` sur les pages de droite (paires)...

Interactions avec les commandes de paragraphe

À chaque début de chapitre, de section, de sous-section, etc. une commande interne de L^AT_EX fait appel aux commandes de marquages présentées au paragraphe précédent, pour stocker des informations susceptibles d'enrichir l'en-tête ou le pied de page. Ces commandes se nomment :

- `\chaptermark` pour les chapitres ;
- `\sectionmark` pour les sections ;
- ...

elles attendent un argument qui contient le titre du chapitre ou du paragraphe. Dans ce manuel, les deux commandes précédentes ont été définies comme suit :

```
\renewcommand{\sectionmark}[1]{% #1 contient le titre de la section
  \markright{\sectionfont\thesection\ #1}}
\renewcommand{\chaptermark}[1]{% #2 contient le titre du chapitre
  \markboth{\sectionfont#1}{}}
```



Puis :

```
\fancyhead[LE,R0]{\thepage}
\fancyhead[L0]{\rightmark}
\fancyhead[RE]{\leftmark}
```

Par conséquent :

- à droite des pages paires, on trouve (\leftmark) le dernier titre de chapitre rencontré;
- à gauche des pages impaires, on trouve (\rightmark) le numéro et le titre de la première \section de cette page, ou le numéro et le titre de la dernière \section rencontrée...

Si vous ne me croyez pas voyez par vous-même le haut de cette page.

10.4.4 Organisation du document

Il est nécessaire de savoir que dans un document tel que celui que vous lisez, il existe trois parties qui sont reconnues par $\text{\LaTeX}$: le *front matter*, le *main matter* et le *back matter* désignant respectivement le début du document (comportant généralement la préface et le sommaire), la partie principale, et la partie clôturant le document (comportant généralement la table des matières, le ou les index, la ou les bibliographies, le glossaire, etc). On doit alors explicitement écrire un document $\text{\LaTeX}$ comme suit :

```
\documentclass{classe du document}
\begin{document}
\frontmatter % introduction
[...]
\mainmatter % partie principale
[...]
\backmatter % pour clore le document
[...]
\end{document}
```

Nous verrons dans la suite de ce chapitre que nous serons amenés à modifier les trois commandes permettant de passer d'une partie à une autre. Pour l'instant, il faut savoir que la classe book définit un booléen :

```
\newif\if@mainmatter
```

utilisé par défaut dans $\text{\LaTeX}$ pour savoir si on se trouve dans le « main matter » ou pas. Nous avons en outre défini pour notre document un autre booléen :

```
\newif\if@frontmatter
```

qui nous permettra d'effectuer des traitements particuliers lorsqu'on sera dans la partie introductive du document. Les trois commandes délimitant les trois parties sont définies par :

```
\renewcommand\frontmatter{%
\cleardoublepage
\@frontmattertrue
\@mainmatterfalse
\pagenumbering{roman}% numérotation en romain}
```

```

}
\renewcommand\mainmatter{%
  \cleardoublepage
  \@mainmattertrue
  \@frontmatterfalse
  \pagenumbering{arabic}% numérotation en chiffres arabes
}
\renewcommand\backmatter{%
  \cleardoublepage
  \@frontmatterfalse
  \@mainmatterfalse
}

```

En farfouillant dans le code de L^AT_EX on peut comprendre que `\pagenumbering`, la commande permettant de changer la numérotation, réinitialise le compteur de page à 1.

10.4.5 Numéroté l'introduction en roman « petites capitales »

Votre serviteur a tenu à ce que les pages de l'introduction soient numérotées en chiffres romains et petites capitales. On ne peut malheureusement pas écrire :

```
\renewcommand{\thepage}{\textsc{\roman{page}}}
```

Puisque cela provoque une incompatibilité avec la gestion de l'index. L'idée retenue est de procéder comme suit :

1. utiliser la numérotation en chiffre romain minuscule ;
2. dans le pied de page ajouter `\textsc{\thepage}` ;
3. modifier la commande `\index` pour que les numéros de pages s'ajoute en petites capitales.

D'où dans la définition de `\frontmatter` on ajoutera :

```

\let\indexORI\index% sauvegarde de la définition initiale
\renewcommand{\index}[1]{\indexORI{##1|textsc}}
\fancyfoot{}
\fancyhead[LE,RO]{\textsc{\thepage}}

```

Et dans la définition de `\mainmatter` :

```
\let\index\indexORI% pour revenir à la définition initiale
```

Pour être parfaitement rigoureux on va modifier l'allure des premières pages de chapitre :

```

\fancypagestyle{plain}{%
  \fancyhf{}
  \if@frontmatter% introduction
    \fancyfoot[C]{\textsc{\thepage}}
  \else
    \fancyfoot[C]{\thepage}
  \fi
\renewcommand{\headrulewidth}{0pt}
\renewcommand{\footrulewidth}{0pt}}
\makeatother

```

10.4.6 Interaction avec index, bibliographie et table des matières

Dans la classe book, deux environnements sont définis :

- thebibliography permettant de produire la bibliographie ;
- theindex pour produire l'index ;

et la commande :

- \tableofcontents pour produire la table des matières.

Ces environnements et cette commande sont conçus pour produire des en-têtes avec le numéro de la page et le nom du chapitre en majuscules à savoir \bibname, \indexname et \contentsname. Voici par exemple un extrait de \tableofcontents :

```
\newcommand\tableofcontents{%
  [...]
  \chapter*{\contentsname
    \@mkboth{%
      \MakeUppercase\contentsname}%
    {\MakeUppercase\contentsname}}%
  \@starttoc{toc}%
  [...]
}
```

J'ai souhaité que dans ce manuel les en-têtes ne soient pas en majuscules. Deux solutions sont possibles :

1. utiliser la commande \nouppercase du package fancyhdr et écrire dans la définition de \backmatter :

```
\fancyhead[LO]{\nouppercase\rightmark}% en-tête en minuscule
\fancyhead[RE]{\nouppercase\leftmark}%
```

2. recopier la définition de \tableofcontents provenant de book.cls, et la modifier pour supprimer les commandes \MakeUppercase. Faire la même chose pour l'index et la bibliographie.

Dans ce document, c'est la deuxième solution qui a été adoptée. On en a également profité pour insérer l'index et la bibliographie dans la table des matières, ce qui n'est pas le comportement par défaut de L^AT_EX et de la classe book. Nous avons donc pour l'environnement theindex :

```
\renewenvironment{theindex}
{%
  [...]
  % insertion dans la table des matières
  \addcontentsline{toc}{chapter}{\indexname}
  \@mkboth{\indexname}{\indexname}% suppression de \MakeUppercase
  \thispagestyle{plain}
  [...]
}{\if@restonecol\onecolumn\else\clearpage\fi}
```

10.5 Environnements avec caractères spéciaux

Les packages fancyvrb et listings ont tous deux la particularité de produire du texte avec des caractères spéciaux. Le premier permet de produire des environnements de type verbatim avec beaucoup plus de souplesse. Il permet notamment

de personnaliser d'éventuelles bordures, les marges, et surtout on peut «s'échapper vers $\text{\LaTeX}$ » au beau milieu de l'environnement, ou comme disent les anglophones : *to escape to $\text{\LaTeX}$* . En d'autres termes, bien qu'étant dans un environnement où les caractères $\backslash$, $\{$ et $\}$ sont sans effet, il est malgré tout possible de faire appel à des commandes $\text{\LaTeX}$.

Le second (listings) a pour objectif de produire des extraits de langage de programmation. Il propose également parmi un grand nombre de fonctionnalités, la possibilité de s'échapper vers $\text{\LaTeX}$. Nous vous proposons donc ici de découvrir ces deux environnements à l'aide d'exemples «en grandeur nature» utilisés dans ce manuel.

10.5.1 Digression vers les caractères...

Il peut ne pas être inutile⁶ de faire ici une petite digression sur la manière dont $\text{\TeX}$ mange et digère les caractères qu'on lui fournit. Il faut savoir que les caractères peuvent entrer dans seize catégories différentes. Chaque caractère peut n'appartenir qu'à une catégorie à la fois. Chacune de ces catégories permet de basculer $\text{\TeX}$ vers un traitement particulier. Par exemple lorsque le caractère $\backslash$ est rencontré, $\text{\TeX}$ va lire les caractères qui suivent pour connaître le nom de la commande (ou *séquence de contrôle*), lorsqu'il rencontre le caractère $\{$, $\text{\TeX}$ va ouvrir un nouveau groupe, lorsque que le caractère $\%$ est lu, $\text{\TeX}$ va ignorer les caractères jusqu'à la fin de la ligne, c'est-à-dire jusqu'à ce qu'il rencontre un caractère catégorisé «fin de ligne», etc. Parmi les catégories reconnues par $\text{\TeX}$:

Catégorie 0 caractère de contrôle ($\backslash$ dans $\text{\LaTeX}$) ;

Catégorie 1 début de groupe ($\{$ dans $\text{\LaTeX}$) ;

Catégorie 2 fin de groupe ($\}$ dans $\text{\LaTeX}$) ;

Catégorie 11 lettre ;

Catégorie 14 commentaire ($\%$ dans $\text{\LaTeX}$) ;

On peut donc s'amuser à changer le contenu de chaque catégorie. Dans l'exemple ci-dessous, on a transformé les caractères $\backslash$, $\{$ et $\}$ en lettres, et on a décidé que les caractères $/$, $($ et $)$ appartiendraient respectivement aux catégories : caractère de contrôle, début de groupe et fin de groupe. Le caractère $\#$ a également été changé de catégorie, c'est désormais un caractère de commentaire.

01

```
{ \catcode'\/=0 \catcode'\(=1 \catcode'\)=2
  \catcode'\#=14
  \catcode'\{=11 \catcode'\}=11 \catcode'\\=11
  # ça on devrait pas le voir...
  \bidule{truc muche} /textbf(en gras)
  )\par
  on retourne dans le monde \LaTeX{}
```

\bidule{truc muche} en gras
on retourne dans le monde $\text{\LaTeX}$...

En outre, il est intéressant de savoir que $\text{\TeX}$ peut rendre actifs certains caractères (qui entrent alors dans la catégorie 13). Ces caractères peuvent alors être définis comme des commandes voici un exemple idiot :

⁶Les français sont parait-il des spécialistes de la litote. Mais ne nous égarons pas...

```
\catcode'\+=13
\newcommand{+}{plus}
3 + 4 = 7
```

3 plus 4 = 7

Dans cet exemple on a rendu « actif » le caractère +, puis on l'a défini comme une commande. Vous noterez qu'ici on a pu créer une commande que l'on utilise sans faire appel au caractère \.



Il peut être utile de savoir que lorsqu'on charge le package babel et l'extension française, les caractères de ponctuations double sont également rendus actifs notamment pour empêcher la césure avant ceux-ci. En outre le caractère ~ est dans la catégorie des caractères actifs dans L^AT_EX. On peut d'ailleurs voir sa définition dans une session L^AT_EX interactive :

```
*\show~
> ~=macro:
->\nobreakspace {}.
<*> \show~
```

10.5.2 Environnements maison basés sur fancyvrb

Les environnements du type de verbatim ont pour but de changer l'appartenance des caractères à leur catégorie respective. En outre, le package fancyvrb permet de définir quels caractères permettent de repasser le contrôle à L^AT_EX. Dans ce document, l'environnement unixcom a été défini comme suit :

```
\DefineVerbatimEnvironment{unixcom}{Verbatim}{%
  commandchars=$« »,
  frame=single, framerule=.4pt, framesep=1.5mm,
  gobble=2,
  xleftmargin=15pt}
```

Cet environnement est donc de type verbatim mais dans lequel on peut « exécuter » des commandes L^AT_EX à l'aide des caractères \$ de catégorie 0, « de catégorie 1 et » de catégorie 2 — on peut bien évidemment choisir n'importe quel caractère pour ce faire. Il faut cependant garder à l'esprit que ceux-ci doivent être à la fois lisibles pour l'utilisateur et peu utilisés à d'autres fins que de repasser le contrôle à L^AT_EX.

Pour afficher le contenu d'une variable :

```
\begin{unixcom}
  echo ${nom variable}
\end{unixcom}
```

Pour afficher le contenu d'une variable :

```
| echo ${nom variable}
```

La commande `\marg` permet de produire son argument entre chevrons simples et en penché. Les autres paramètres de la commande `\DefineVerbatimEnvironment` ont pour but de préciser l'allure de la bordure (paramètres `frame...`), la marge de gauche (paramètre `xleftmargin`) et le fait que les premiers caractères de chaque ligne seront systématiquement ignorés (`gobble`). Comme le montre la documentation du package fancyvrb beaucoup d'autres options sont disponibles.

Un autre environnement de ce genre a été créé pour saisir les commandes d'Emacs dans l'annexe consacrée à Auc_TE_X. L'environnement en question (`emacscom`) a été créé comme suit :

```
\DefineVerbatimEnvironment{emacscom}{Verbatim}{%
  commandchars=$« »,
```

```
frame=leftline, framerule=1mm, framesep=2mm,
gobble=2, xleftmargin=15pt}
```

qui donne par exemple :

```
Pour jouer à Tetris dans \soft{Emacs} :
\begin{emacscom}
  M-x tetris
\end{emacscom}
```

```
Pour jouer à Tetris dans Emacs :
  M-x tetris
```

10.5.3 Environnements pour les langages de programmation

Le package listings reconnaît la syntaxe d'un grand nombre de langage de programmation. Une manière simple d'utiliser ce package consiste à créer son propre environnement à l'aide d'une commande analogue à `\newenvironment` :

```
\lstnewenvironment{C}{\lstset{language=C}}{}
```

On pourra alors simplement écrire :

```
\begin{C}
/* hello world en C */
int main()
{
  printf("bonjour monde\n");
  return 0;
}
\end{C}
```

```
/□ hello world en C □/
int main()
{
  printf("bonjour□monde\n");
  return 0;
}
```

Bien évidemment une foultitude d'option de configuration permet d'adapter cet environnement à vos besoins. Le plus simple et le plus efficace est sans doute de lire la documentation accompagnant le package. À titre d'exemple, il faut savoir que l'on peut changer la mise en évidence des mots réservés et des commentaires du langage considéré. Ainsi, en écrivant :

```
\lstnewenvironment{Cbis}{%
  \lstset{language=C,
    basicstyle=\rmfamily\slshape,
    commentstyle=\rmfamily\upshape,}}{}
```

On aura :

```
\begin{Cbis}
/* hello world en C */
int main()
{
  printf("bonjour monde\n");
  return 0;
}
\end{Cbis}
```

```
/□ hello world en C □/
int main()
{
  printf("bonjour_monde\n");
  return 0;
}
```

Et puisqu'il est question des caractères spéciaux et de l'«échappement vers $\text{\LaTeX}$ », il faut savoir qu'à l'instar de `fancyvrb`, le package listings permet de spécifier un caractère permettant cet échappement. Ainsi :

```
\lstnewenvironment{Cter}{\lstset{language=C, escapechar=@}}{}
```

Permet d’insérer des commandes $\text{\LaTeX}$ dans le listing :

<pre>\begin{Cter} int main() { printf("bonjour monde\n"); return @\fbox{code de retour}@; } \end{Cter}</pre>	108	<pre>int main() { printf("bonjour □monde\n"); return code de retour; }</pre>
--	---	--

10.6 About those so called “french guillemets”

Un des plaisirs de la typographie française est sans aucun doute l’utilisation de ces merveilleux guillemets «à la française»...⁷ Cependant le package babel ne gère pas la césure correctement s’il on saisit dans un document :

```
\begin{minipage}{5cm}
  Cette courte phrase dans une boîte a pour unique
  but de montrer que ces symboles ne se comportent
  pas comme de « gentils » guillemets.
\end{minipage}
```

on aura la minipage suivante :

Cette courte phrase a pour unique but de montrer que ces petits symboles ne se comportent pas comme de « gentils » guillemets.

Ce qui est pour le moins gênant... Il est bien sûr possible de saisir ces guillemets avec les commandes `\og` et `\fg` fournies par le package babel, mais cela est au goût de votre serviteur trop contraignant dans la mesure où ce caractère est directement accessible depuis le clavier.⁸ Il existe une solution — qui avait été adopté par le package french — palliant le problème de la césure qui consiste à rendre **actifs** les caractères ‘«’ et ‘»’. Nous avons donc écrit :

```
\catcode'\«=13
\catcode'\»=13
```

puis défini les deux commandes suivantes :

```
\newcommand{\fermerguillemets}{\unskip\kern.15em\symbol{20}}
\newcommand{\ouvrerguillemets}{\symbol{19}\ignorespaces\kern.15em}
```

On notera l’utilisation de la commande $\text{\TeX}$ `\kern` permettant d’insérer un blanc insécable d’une longueur donnée, de la commande **unskip** et enfin de la commande `\symbol` qui insère ici les 19^e et 20^e caractères de la fonte courante :

⁷On notera d’ailleurs que la mode qui consiste aujourd’hui à faire subir à ses majeurs et index des mouvements d’oreilles de lapin, pendant qu’on parle pour dire «entre guillemets» sans le dire, est clairement emprunter aux américains. Je propose donc ici publiquement que l’on se force à utiliser plutôt l’index et le pouce pour faire ce geste, il faudra par contre se munir de deux autres bras pour être en mesure d’imiter les deux chevrons « et », ou peut-être qu’un habile tortillage de quatre doigts d’une main peut donner un résultat...

⁸Alt-Gr-z et Alt-Gr-x.

```
\setcounter{car}{1}
\whiledo{\value{car}<64}{%
  \symbol{\value{car}}$_{\thecar}$
\stepcounter{car}}
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14
>15 "16 "17 „18 «19 »20 -21 —22 23 24 25 26 27
fi28 fl29 □30 □31 □32 !33 "34 #35 $36 %37 &38 '39
(40 )41 *42 +43 ,44 -45 .46 /47 048 149 250 351 452
553 654 755 856 957 :58 ;59 <60 =61 >62 ?63
```

Enfin, on a affecté aux caractères les deux commandes précédentes :

```
\let»=\fermerguillemets
\let«=\ouvrerguillemets
```

Cette façon de faire a trois inconvénients mineurs que je suis bien incapable de résoudre aujourd'hui. Tout d'abord, les moteurs sachant gérer le codage Utf 8 et permettant à T_EX de rendre actif le caractère « (alors codé sur 2 octets), sont aujourd'hui peu répandus et j'avoue humblement que je ne les ai pas encore testés. Par conséquent la manipulation proposée ici est limitée aux codages utilisant 1 octet par caractère. Ensuite on ne peut utiliser ces guillemets dans un titre au risque d'avoir des artéfacts dans les « signets/*bookmarks* » d'un fichier pdf. Enfin, ces guillemets ne fonctionnent pas avec l'environnement ltxexemple défini à la fin du chapitre suivant. Un drame quoi !

10.7 Une boîte pour la minitable des matières

Le package mini-toc permet — comme son nom l'indique — de produire des « minitables des matières » que l'on peut insérer dans un document à un endroit donné, généralement en début de chapitre. Après avoir utilisé l'ordre `\dominitoc` dans le préambule, on fait ensuite appel à la commande `\minitoc` pour insérer cette minitable des matières à l'endroit voulu. La documentation du package explique tout cela en détail et présente notamment les différents styles que l'on peut utiliser. Pour ce manuel, j'ai trouvé l'idée d'une table des matières en début de chapitre séduisante, mais les styles proposés par le package ne me convenaient pas. En fait je souhaitais pouvoir mettre les titres de sections dans un boîte comme ceci :

Sommaire x.1 Le premier titre x.2 Le deuxième titre x.3 etc.

C'est-à-dire une boîte avec un titre — ici le titre est « Sommaire ». À ma connaissance, L^AT_EX ne propose pas de telles boîtes et suite à une question posée sur les forums de discussions, une bonne âme — en l'occurrence Benjamin BAYART — me propose un code T_EX répondant au cahier des charges. Je vous propose dans ce paragraphe, une version⁹ L^AT_EX d'une boîte avec titre...

10.7.1 L'interface de la commande

Plusieurs solutions sont possibles pour créer une telle commande. En s'inspirant de l'interface des boîtes de L^AT_EX, on peut créer une macro dont la syntaxe d'utilisation serait :

⁹Tout à fait discutable et limitée dans ses fonctionnalités comme tout « logiciel » pondu par un bricoleur...

```
\titlebox{\footnotesize Un titre}{%
  Le contenu de la boîte}
```

1010

Un titre
Le contenu de la boîte

ou encore :

```
\setlength{\fboxsep}{5pt}
\setlength{\fboxrule}{2pt}
\titlebox{Un autre titre}{%
  \begin{minipage}{3cm}\begin{center}
    truc\ \ muche
  \end{center}\end{minipage}}
```

1011

Un autre titre
truc
muche

10.7.2 Quand même un peu de T_EX

La primitive de T_EX `\leaders` permet de remplir un espace élastique avec ce qui vous passe par la tête. Sa syntaxe :

`\leaders` *ce qui vous chante* *space*

permet donc de remplir l'*espace* avec *ce qui vous chante*. Par exemple :

```
\framebox[3cm]{%
  \leaders\hbox{o}\hfill}
```

1012

oooooooooooooooo

La primitive `\hbox` de T_EX (utilisée par `\mbox` et `\makebox`) permet de créer des boîtes horizontales :

```
\framebox[3cm]{%
  \leaders\hbox to 3pt{o}\hfill}
```

1013

oooooooooooooooooooo

Les `\leaders` peuvent également être utilisés avec la primitive `\hrule` de T_EX permettant de dessiner des traits :

```
\framebox[3cm]{%
  \leaders\hrule height 4pt\hfill}
```

1014

—————

Ici, le ressort `\hfill` s'étire les trois centimètres de la `\framebox` et est rempli par un trait de hauteur quatre points.

```
\framebox[3cm]{%
  \leaders\hbox to 5pt{%
    \leaders\hrule width 1pt\hfill%
    \kern 2pt}\hfill}
```

1015

Dans l'exemple ci-dessus, l'espace de trois centimètres est rempli par des boîtes de cinq points de large, contenant chacune d'elles des `\leaders` comme dans l'exemple précédent, et un blanc de deux points de large. Avec T_EX, on peut régler la *raideur* du ressort de la manière suivante :

```
\framebox[5cm]{%
  \hskip 0pt plus 2fill X\hskip 0pt plus 3fill}
```

1016

X

La dimension :

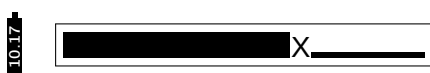
`\hskip 0pt plus nfill`

permet de définir une longueur élastique de raideur relative n . Dans l'exemple précédent la lettre 'X' se trouve donc au $2/5$ de la boîte... En utilisant ce type de blanc élastique et des `\leaders`, on peut définir la commande suivante :

```
\newcommand{\traitressort}[2][1]{%
  \leaders\hrule height#2\hskip0pt plus #1fill\relax}
```

pouvant être par exemple utilisée comme suit :

```
\framebox[5cm]{%
  \traitressort[2]{2ex}X\traitressort{2pt}}
```



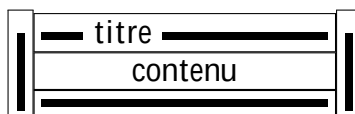
on a donc dans la boîte de cinq centimètres :

- un blanc élastique de raideur 2, rempli d'un trait de quatre points de hauteur ;
- la lettre X ;
- un blanc élastique de raideur 1, rempli d'un trait de deux points de hauteur.

Nous allons bien entendu nous servir de cette commande pour la suite...

10.7.3 Conception de la boîte

Pour concevoir la boîte à proprement parler, nous allons créer trois `\parbox` comme suit :



Il y a :

- deux `\parbox`s pour contenir les deux traits verticaux à droite et à gauche ;
- une `\parbox` pour le centre, contenant le trait du haut interrompu par le titre, le contenu, et en bas un trait horizontal.

Nous allons voir maintenant comment on peut construire ces trois boîtes et les positionner correctement les unes par rapport aux autres.

10.7.4 Le code

Nous allons avoir besoin d'une boîte pour stocker la `\parbox` centrale :

```
\newsavebox{\boitetitre}
```

et de deux dimensions :

```
\newlength{\largeurboitetitre}
\newlength{\hauteurboitetitre}
```

qui portent un nom suffisamment explicite m'évitant ainsi des phrases alambiquées expliquant la signification de telle ou telle variable. La première tâche que l'on va demander à la commande `\titlebox` est de stocker son contenu et de le mesurer :

```
\newcommand{\titlebox}[2]{%
  \begin{lrbox}{\boitetitre}% stockage du contenu
    \kern\fboxsep#2\kern\fboxsep
  \end{lrbox}}
```

```
% mesure de la largeur de la parbox centrale
\settowidth{\largeurboitetitre}{\usebox{\boitetitre}}%
% mesure de la hauteur de la parbox centrale
\settoheight{\hauteurboitetitre}{\usebox{\boitetitre}}%
\settodepth{\tempdim}{\usebox{\boitetitre}}%
\addtolength{\hauteurboitetitre}{\tempdim+2\fbboxrule+2\fbboxsep}%
... }
```

`\kern` est une commande $\text{\TeX}$ permettant d'insérer un blanc insécable, ici de largeur `\fbboxsep`. Notez que pour mesurer la hauteur totale on a recours à une longueur temporaire qui nous permet de faire la somme de la hauteur (*height*) et la profondeur (*depth*). On ajoute ensuite à cette hauteur totale deux fois l'épaisseur du trait et deux fois l'espace `\fbboxsep`. Vous vous souvenez sans doute que les dimensions `\fbboxrule` et `\fbboxsep` définissent respectivement l'épaisseur du trait et l'espace entre la bordure et le contenu d'une **boîte simple**. Par conséquent, on a :

- `\largeurboitetitre` correspond à la largeur de la `\parbox` centrale augmentée de deux fois `\fbboxsep` ;
- `\hauteurboitetitre` correspond à la hauteur totale augmentée de l'espace occupée par les deux traits horizontaux : $2(\text{\fbboxsep} + \text{\fbboxrule})$.

On peut donc construire une première version de la commande :

```
\newcommand{\titleboxI}[2]{%
...
\parbox{\fbboxrule}{% le trait de gauche
  \rule{\fbboxrule}{\hauteurboitetitre}}%
\parbox{\largeurboitetitre}{% la boîte centrale
  \begin{flushleft}
    \usebox{\boitetitre}
  \end{flushleft}}%
\parbox{\fbboxrule}{% le trait de droite
  \rule{\fbboxrule}{\hauteurboitetitre}}}}
```

Ce qui donnera pour l'instant :

<code>\titleboxI{titre}{Bidule truc mucho}</code>	10.18	Bidule truc mucho
<code>\titleboxI{encore}{% \parbox{4cm}{truc\bidule\machin}}</code>	10.18	 truc bidule machin

Il reste donc à modifier le contenu de la `\parbox` centrale pour ajouter les deux traits horizontaux, celui du bas, et celui du haut coupé par le titre. L'idée est d'entasser trois boîtes :

1. une boîte contenant le titre et des « traits ressorts » :
2. la boîte stockant le contenu (`\boitetitre`) ;
3. un trait de largeur `\largeurboitetitre`.

Voici une première approche :

```
\newcommand{\titleboxII}[2]{%
...
\parbox{\largeurboitetitre}{% la boîte centrale
```

```

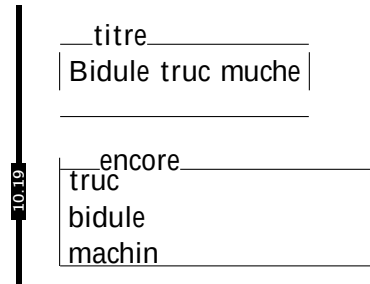
\begin{flushleft}
  \makebox[\largeurboitetitre]{%
    \traitressort{\fboxrule}#1\traitressort[5]{\fboxrule}}\
  \usebox{\boitetitre}\
  \rule{\largeurboitetitre}{\fboxrule}
\end{flushleft}}
...}

```

qui donnera :

```
\titleboxII{titre}{Bidule truc muche}
```

```
\titleboxII{encore}{%
  \parbox{4cm}{truc\bidule\machin}}
```



On dirait que c'est pas « tout à fait ça ». Il faudrait penser à faire en sorte que la commande `\e` effectue un saut vertical équivalent à la dimension `\fboxsep`. On en profite au passage pour faire subir au titre une translation verticale vers le bas :

```

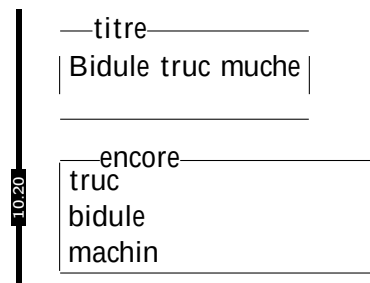
\newcommand{\titleboxIII}[2]{%
  ...
  \parbox{\largeurboitetitre}{% la boîte centrale
    \begin{flushleft}
      \makebox[\largeurboitetitre]{%
        \traitressort{\fboxrule}%
        \raisebox{-.5ex}[0pt][0pt]{#1}%
        \traitressort[5]{\fboxrule}}\[\fboxsep]
      \usebox{\boitetitre}\[\fboxsep]
      \rule{\largeurboitetitre}{\fboxrule}
    \end{flushleft}}
  ...}

```

ce qui donnera :

```
\titleboxIII{titre}{Bidule truc muche}
```

```
\titleboxIII{encore}{%
  \parbox{4cm}{truc\bidule\machin}}
```



On dirait que ça n'a pas arrangé grand chose... Il faut savoir que lorsque T_EX entasse des boîtes en mode vertical, il insère de lui même des espaces entre ces boîtes de manière à ce que les lignes soit espacées de la longueur `\baselineskip`. On trouve dans le T_EXbook, à la page 79 du chapitre traitant des *glues* :

« Exception : no interline glue is inserted before or after a rule box. You can also inhibit interline glue by saying `\nointerlineskip` between two boxes. »

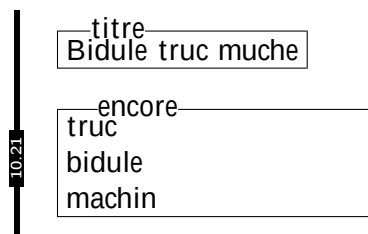
L'ordre `\nointerlineskip` résout donc le problème :

```
\newcommand{\titleboxIV}[2]{%
...
\parbox{\largeurboitetitre}{% la boîte centrale
\begin{flushleft}
\makebox[\largeurboitetitre]{%
\traitressort{\fboxrule}%
\raisebox{-.5ex}[0pt][0pt]{#1}%
\traitressort[5]{\fboxrule}}\[\fboxsep]\nointerlineskip
\usebox{\boitetitre}\[\fboxsep]\nointerlineskip
\rule{\largeurboitetitre}{\fboxrule}
\end{flushleft}}
...}
```

ce qui donnera :

```
\titleboxIV{titre}{Bidule truc mucho}
```

```
\titleboxIV{encore}{%
\parbox{4cm}{truc\bidule\machin}}
```



Ce qui répond au cahier des charges.



D'autres améliorations — laissées en guise d'exercice — peuvent être apportées à cette commande. On pourra par exemple définir un argument optionnel permettant de régler l'abaissement du titre (on a mis ici `-.5ex` en dur). Il est également possible de régler le rapport des traits entourant le titre. Enfin, il est tout à fait envisageable de régler l'espace autour du titre (ici il n'y en a pas).

10.7.5 Utilisation avec package minitoc

L'utilisation de la commande `\titlebox` précédemment définie, dans le package `minitoc` se fait simplement en revêtant le chapeau de monsieur POIROT. En inspectant à la loupe le fichier de style, on trouve la définition d'une commande nommée `\minitoc@`. J'ai simplement recopié le code de cette macro et inséré un appel à la merveilleuse commande `\titlebox`.

11

De nouveaux jouets

Sommaire

- 11.1 Quelques bricoles
- 11.2 Des nota
- 11.3 Des citations
- 11.4 Des lettrines
- 11.5 Un sommaire
- 11.6 Un glossaire
- 11.7 Des onglets
- 11.8 Exemples L^AT_EX

*Je suis à mon bien-aimé, Et ses désirs se portent vers moi.
Viens, mon bien-aimé, sortons dans les champs, Demeurons dans les villages!
Dès le matin nous irons aux vignes, Nous verrons si la vigne pousse,
Si la fleur s'ouvre, Si les grenadiers fleurissent.
Là je te donnerai mon amour.*

Le Cantique des cantiques Ct 7 11.

NOUS PRÉSENTONS dans ce chapitre les outils qui ont été créés spécialement pour ce manuel. Pour comprendre la plupart des commandes et environnements définis ici, il est impératif d'avoir lu les deux précédents chapitres... Il est question dans ce chapitre de la manière dont le nota avec panneau danger a été créé, des lettrines apparaissant en début de chapitres, du sommaire, du glossaire, des onglets contenant le numéro du chapitre courant, et enfin de l'environnement permettant de produire du code L^AT_EX et son interprétation côte à côte.

11.1 Quelques bricoles

11.1.1 Arguments et convention typographique

Dans un document parlant de langage informatique, il est important de faire ressortir clairement les arguments de commande ou de fonction. Par exemple on écrira :

```
Pour compiler le fichier \bwarg{fichier} :  
\begin{flushleft}  
  \ttfamily latex \bwarg{fichier}  
\end{flushleft}
```

```
Pour compiler le fichier ⟨fichier⟩ :  
⋮ latex ⟨fichier⟩
```

La commande `\bwarg` écrit son argument en *fonte penchée*, entre les symboles `⟨` et `⟩` produits respectivement par les commandes `\langle` et `\rangle` en mode mathé-

matique. De plus vous aurez sans doute remarqué qu'on peut utiliser une notation indicée comme dans l'exemple ci-dessous :

Pour copier des fichiers :

```
\begin{flushleft}\ttfamily
  cp \bwarg[1]{fichier} ... \bwarg[n]{fichier}
  \bwarg[dst]{fichier}
\end{flushleft}
```

Pour copier des fichiers :

```
cp <fichier1> ... <fichiern> <fichierdst>
```

La commande `\bwarg`¹ est définie comme suit :

```
\newcommand{\marg}[2][]{%
  {\normalfont%
    \textsl{$\langle$#2%
    \ifthenelse{\equal{#1}{}}{}% si l'argument optionnel est présent
    {$\_mathit{#1}$}% on l'a che en indice
    $\rangle$}}}%
}
```

La commande `\normalfont` permet de revenir à la fonte par défaut dans le document. Ce qui explique que « `<fichier>` » n'apparaît pas en fonte machine à écrire dans l'exemple 11.1.

Dans la version électronique du document — version lue sur un écran — il a été décidé, pour la mise en évidence, d'utiliser la *couleur* plutôt que les caractères `<` et `>`. Ainsi on peut définir la commande `\colarg` :

```
\newcommand{\colarg}[2][]{%
  \normalfont\color{blue!90}#2% un bleu
  \ifthenelse{\equal{#1}{}}{}{$\_mathit{#1}$}}
}
```

On pourra ensuite grâce à un **booléen** habilement positionné, définir une commande générique `\marg` faisant appel à l'une ou l'autre des versions (noir & blanc ou couleur) :

```
\ifversionenligne
  \let\marg\colarg
\else
  \let\marg\bwarg
\fi
```

Cette construction fait appel à la commande `\let` de T_EX présentée de manière lumineuse à la section 9.2.3 page 109.

11.1.2 Autour de la génération de l'index

Lorsque dans le texte du présent manuel, il est question d'une commande, d'un environnement, d'un package, d'une classe de document, etc. il est fait appel à une commande particulière insérant automatiquement une entrée dans l'index. Ainsi par exemple :

Le package `\ltxpack{varioref}` permet d'utiliser la commande `\ltxcom{vref}`...

Le package `varioref` permet d'utiliser la commande `\vref`...

La commande `\ltxpack` est définie comme suit. Tout d'abord :

¹Pour « black & white » argument...

```
\newcommand{\ltx@pack}[1]{%
  \upshape\textsf{#1}}
```

définissant la commande `\ltx@pack` permettant simplement de produire le nom du package en sans sérif. On définit ensuite :

```
\newcommand{\ltxpack}[1]{%
  \ltx@pack{#1}%
  \protect\index{extensions!\protect\texttt{#1}}%
  \protect\index{#1@\protect\textsf{#1 extension}}}
```

qui appelle la commande précédente, et qui insère deux entrées dans l'index. Une de la forme « **nom du package** extension » et l'autre comme **sous-entrée** de « extensions ». La commande `\protect` permet ici d'éviter les ennuis si la commande `\ltxpack` est elle-même en argument d'une autre commande. Dans un même ordre d'idée, la commande `\ltxcom` est définie tout d'abord par :

```
\newcommand{\ltx@com}[1]{%
  \texttt{\symbol{92}#1}}
```

permettant de produire en fonte machine à écrire, le nom de la commande précédé du caractère `\`. La commande `\symbol` est une commande $\text{\LaTeX}$ permettant d'insérer ici le 92^e caractère de la fonte sélectionnée (en l'occurrence le backslash). On peut alors définir la commande finale :

```
\newcommand{\ltxcom}[1]{%
  \ltx@com{#1}%
  \index{#1@\protect\texttt{\symbol{92}#1}}}
```

qui appelle la commande précédente et introduit une entrée dans l'index. L'idée à retenir, c'est qu'il est peut être utile de définir des commandes pour insérer automatiquement des entrées dans l'index. On pourrait par exemple définir une commande :

```
\newcommand{\jargonanglais}[1]{%
  \emph{#1}%
  \index{#1}}
```

permettant à la fois de formater les mots du jargon en anglais, et de les insérer dans l'index, voire dans un index spécial. De même si un mot revient souvent dans un document on peut définir une commande pour le produire et l'insérer dans l'index. Par exemple dans ce manuel, on a défini :

```
\newcommand{\postscript}{%
  PostScript%
  \protect\index{PostScript}}
```

11.1.3 Des renvois

La version « papier » de ce document est parsemée de renvois comme celui-ci parlant de `\ glossaire`, qui n'a strictement rien à voir avec le propos du moment² si ce n'est qu'il s'agit d'un renvoi. La commande mise en œuvre pour les renvois a été baptisée `\voir` et attend deux arguments : § D.3.6 p. 217 J

²Je veux dire que nous ne sommes pas en train de parler de glossaire, ou alors vous ne suivez pas du tout...

```
\voir{label cible}{texte objet du renvoi}
```

Par exemple, le renvoi précédent a été produit par :

```
\voir{chap-glossaire}{glossaire}
```

Toute la «difficulté» de la conception de cette commande réside dans l'orientation des triangles qui dépend de la parité de la page. Cette difficulté peut être levée grâce à l'utilisation du package `chnpage` comme expliqué au paragraphe 9.3.3 page 113.

Le reste concerne la mise en page des triangles. Pour cela deux commandes ont été définies, produisant chacune les renvois dans la marge et les marques dans le texte. D'où la forme de la commande `\voir` :

```
\newcommand{\voir}[3][\S]{%
  \checkoddpage%
  \ifcoddpage
    \v@irpageimpaire{#1}{#2}{#3}}{% renvoi sur page impaire
  \else
    \v@irpagepaire{#1}{#2}{#3}} % renvoi sur page paire
\fi
```

On notera qu'outre les deux arguments obligatoires, la commande accepte un argument optionnel défini par défaut comme étant le caractère de paragraphe (§). Les deux commandes `\v@irpageimpaire` et `\v@irpagepaire` sont symétriques l'une de l'autre et ont pour objet :

1. d'encadrer le texte objet du renvoi par des triangles correctement orientés ;
2. de produire une note marginale avec la cible du renvoi.

Les triangles sont obtenus à l'aide de symboles contenus dans le package `amssymb` :

Oh les \og joulis\fg{} triangles :
 $\blacktriangleleft$ et
 $\blacktriangleright$!

114

Oh les « joulis » triangles : J et I !

Le petit triangle à plat est obtenu grâce à la commande suivante :

```
\newcommand{\petit@triangle}{%
  \makebox[2.5pt]{\tiny$\blacktriangle$}}
```

On peut alors écrire :

Mais, il sont tous \petit@triangle{}petits
ces \petit@triangle{}triangles...

115

Mais, il sont tous $\text{\tiny}\blacktriangle$ petits ces $\text{\tiny}\blacktriangle$ triangles...

Il n'est pas inutile de noter l'usage de la boîte de deux points et demi de large pour faire croire à ce naïf de $\text{\LaTeX}$ que c'est la largeur effective de `\blacktriangle`. Ce qui n'est bien entendu pas le cas, mais contribue à coller ce «petit triangle» au mot qui le suit. Je sens que vous n'êtes pas convaincu, donc :

Mais, il sont tous $\text{\tiny}\blacktriangle$ %
petits ces $\text{\tiny}\blacktriangle$ triangles...

116

Mais, il sont tous $\text{\tiny}\blacktriangle$ petits ces $\text{\tiny}\blacktriangle$ triangles...

Voici finalement la commande permettant de faire un renvoi dans le cas d'une page paire :

```
\newcommand{\v@irpagepaire}[3]{%
  \petit@triangle#3{\small$\blacktriangleleft$}% le renvoi est à gauche
  \marginpar{{\small% on écrit en petit dans la marge
    % le numéro et la page de la cible du renvoi ...
    $\blacktriangleright$ #1~\ref{#2} p.~\pageref{#2}}}}
```

Pour la page impaire il faut faire la même chose que pour la page paire, mais en tenant compte que c'est une page impaire :-)

Dans la version «en ligne» les renvois apparaissent comme un lien hypertexte. Ceci est facilement réalisable grâce à la commande `\hyperref` du package éponyme. On aura donc quelque chose du genre :

```
\newcommand{\voir}[3][\S]{%
  \hyperref[#2]{#3}}
```

L'argument optionnel ne sert ici à rien d'autre qu'à assurer la compatibilité avec la version «papier» de la commande `\voir`.

11.1.4 Changement de marges

À plusieurs reprises dans ce document, j'ai eu recours à des changements de marges provisoires. C'est le cas par exemple des exemples de code $\text{\LaTeX}$ avec le résultat en face, ou pour les épigraphes. Pour ce faire, Marie-Paul KLUTH³ qui maintient la Faq française de $\text{\LaTeX}$ avait suggéré un environnement ressemblant à celui-ci :

```
\newenvironment{changemargin}[2]{%
{\begin{list}{}{%
  \setlength{\listparindent}{\parindent}%
  \setlength{\itemindent}{\parindent}%
  \setlength{\leftmargin}{#1}%
  \setlength{\rightmargin}{#2}%
}\item }%
{\end{list}}}
```

L'idée est donc de définir une liste dont on change les marges. Les deux arguments qu'attend cet environnement correspondent respectivement aux dimensions des marges gauche et droite. Une idée intéressante serait celle d'un environnement dans lequel les marges ont des dimensions différentes selon la parité de la page. Un tel environnement peut être défini comme suit :

```
\newenvironment{agrandirmarges}[2]{%
\begin{list}{}{%
  \setlength{\topsep}{0pt}%
  \setlength{\listparindent}{\parindent}%
  \setlength{\itemindent}{\parindent}%
  \setlength{\parsep}{0pt plus 1pt}%
  \checkoddpage%
  \ifcoddpage
    \setlength{\leftmargin}{-#1}\setlength{\rightmargin}{-#2}
  \else
    \setlength{\leftmargin}{-#2}\setlength{\rightmargin}{-#1}
```

³Qui possède un lom prédestilé...

FIG. 11.1 – Une figure qui ne sert à rien si ce n’est à montrer qu’on peut momentanément changer les marges gauche et droite quand on a besoin de place...

```
\fi}\item }%
{\end{list}}
```

Notez qu’on utilise ici la commande `\isodd` pour tester la parité de la page. La figure 11.1 montre un exemple d’utilisation de cet environnement avec le code suivant :

```
\begin{figure}[tb]
\begin{agrandirmarges}{1cm}{2cm}
% ici on a 1cm de plus côté « reliure »
%      et 2cm de plus côté « bord »
...
\caption{Une figure qui ne sert à rien...}
\end{agrandirmarges}
\end{figure}
```

11.2 Des nota

Les pictogrammes⁴ ont été « conçus » avec le logiciel xfig et sont représentés à la figure 11.2 en trois centimètres de large. Les « nota » insérés ça et là dans le

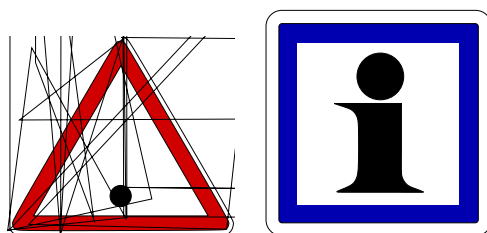


FIG. 11.2 – Les pictogrammes du manuel

document, ont été produits par un environnement défini par votre serviteur, basé sur une fonctionnalité de niveau T_EX découverte lors de mes laborieuses lecture du T_EXBook : la commande `\parshape`. Cette commande permet de donner une forme arbitraire à un paragraphe :

⁴L'idée de ces pictogrammes a été inspirée par la lecture de T_EXbook, comme je l'explique en introduction.

<pre> \parshape=5 2cm 3cm 2cm 3cm 1cm 2cm 1cm 2cm 0pt \textwidth a </pre>		<pre> a </pre>
---	--	--

Le nombre suivant le signe '=' permet de spécifier le nombre de lignes auxquelles on imposera une déformation. Suivent ensuite des couples de dimensions indiquant le retrait et la longueur de chaque ligne déformée. Dans l'exemple ci-dessus :

- les deux premières lignes auront un retrait de deux centimètres et mesureront chacune trois centimètres ;
- les deux lignes suivantes seront indentées d'un centimètre et mesureront deux centimètres ;
- la cinquième et dernière spécification détermine l'allure de toutes les lignes restantes : retrait de zéro centimètre et longueur de la ligne égale à la longueur prédéfinie \textwidth.

Pour insérer un nota dans un paragraphe, on va donc déplacer les deux premières lignes à l'aide de cette commande.

<pre> \setlength{\larnota}{.9cm} \setlength{\largligne}{\textwidth-\larnota} \parshape=3 \larnota\largligne \larnota\largligne 0pt\textwidth \noindent Attention ce paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe et ensuite continuer comme si de rien n'était... </pre>		Attention ce paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe et ensuite continuer comme si de rien n'était...
--	--	---

Bon, il reste à essayer de mettre l'image dans le « trou » laissé par la commande \parshape. Essayons simplement :

<pre> \setlength{\larnota}{.9cm} \setlength{\largligne}{\textwidth-\larnota} \parshape=3 \larnota\largligne\larnota\largligne 0pt\textwidth\noindent% \includegraphics[width=\larnota]{\ficnota} Attention ce paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe [...] </pre>		 Attention ce paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe [...]
---	--	---

Évidemment, l'image se pose sur la ligne comme n'importe quel autre caractère. On la met dans une boîte de largeur nulle dont le contenu est aligné à droite :

```

\setlength{\larnota}{.9cm}
\setlength{\largligne}{\textwidth-\larnota}%
\parshape=3
\larnota\largligne\larnota\largligne%
0pt\textwidth\noindent%
\makebox[0pt][r]{%
  \includegraphics[width=\larnota]{\ficnota}}%
Attention ce joli paragraphe a uniquement
pour but de montrer que l'on peut décaler deux
lignes dans un paragraphe [...]

```



Attention ce joli paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe [...]

Il reste à faire subir au pictogramme, une translation verticale (le pictogramme est pratiquement carré, on se sert donc de la dimension `\indnota`) :

```

\setlength{\larnota}{.9cm}
\setlength{\largligne}{\textwidth-\larnota}
\parshape=3
\larnota\largligne\larnota\largligne
0pt\textwidth\noindent%
\raisebox{-\larnota}{%
  \makebox[0pt][r]{%
    \includegraphics[width=\larnota]{\ficnota}}}%
Attention ce joli paragraphe a uniquement
pour but de montrer que l'on peut décaler deux
lignes dans un paragraphe [...]

```



Attention ce joli paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe [...]

Bon, encore raté, il faut faire croire à $\text{\LaTeX}$ que la boîte qu'on translate verticalement est de taille nulle :

```

\setlength{\larnota}{.9cm}
\setlength{\largligne}{\textwidth-\larnota}
\parshape=3
\larnota\largligne\larnota\largligne
0pt\textwidth\noindent%
\raisebox{-\larnota}[0pt][0pt]{%
  \makebox[0pt][r]{%
    \includegraphics[width=\larnota]{\ficnota}}}%
Attention ce joli paragraphe a uniquement
pour but de montrer que l'on peut décaler deux
lignes dans un paragraphe [...]

```



Attention ce joli paragraphe a uniquement pour but de montrer que l'on peut décaler deux lignes dans un paragraphe [...]

On y est presque. Deux ajustements sont nécessaires :

- la boîte est un peu trop basse, puisque la ligne de référence est le bas de la ligne du texte. Par conséquent on peut enlever `1ex` (la hauteur d'un caractère) à la translation ;
- il serait bon d'ajouter un espace entre le pictogramme et le texte. On définit pour cela une longueur `\padnota`.

```

\setlength{\padnota}{5pt}
\setlength{\larnota}{.9cm}
\setlength{\indnota}{\larnota+\padnota}
\setlength{\largligne}{\textwidth-\indnota}
\parshape=3
\indnota\largligne\indnota\largligne
0pt\textwidth\noindent%
\raisebox{-\larnota+2.2ex}[0pt][0pt]{%
  \makebox[0pt][r]{%
    \includegraphics[width=\larnota]{\ficnota}%
    \hspace*{\padnota}}}%
Attention ce joli paragraphe a uniquement
pour but de montrer qu'on peut décaler deux
lignes dans un paragraphe [...]

```



Attention ce joli paragraphe a uniquement pour but de montrer qu'on peut décaler deux lignes dans un paragraphe [...]

Et ouala! Il ne reste «qu'à» modifier ce code pour créer un nouvel environnement. La technique choisie ici est de se baser sur l'environnement `list` présenté au paragraphe 9.5 page 119. Le code complet de l'environnement est le suivant :

```

\newenvironment{pictonote}[1]{% on passe le nom du fichier en argument
  \begin{list}{}{%
    \setlength{\labelsep}{0pt}%
    \setlength{\rightmargin}{15pt}}
  \item%
    \setlength{\indentationnota}{%
      \@totalleftmargin+\largeurnota+\paddingnota}%
    \setlength{\largeurlignenota}{%
      \linewidth-\largeurnota-\paddingnota}%
    \parshape=3%
    \indentationnota\largeurlignenota%
    \indentationnota\largeurlignenota%
    \@totalleftmargin\linewidth%
    \raisebox{-\largeurnota+2.2ex}[0pt][0pt]{%
      \makebox[0pt][r]{%
        \includegraphics[width=\largeurnota]{#1}%
        \hspace{\paddingnota}}}%
    \ignorespaces}%
  \end{list}}

```



On notera l'utilisation de la dimension `\@totalleftmargin` permettant d'obtenir la largeur de la marge de gauche dans une liste, en tenant compte d'éventuelles imbrications. En effet la dimension `\leftmargin` dans une liste correspond à la marge de gauche *relativement* à l'environnement contenant la dite liste.

On pourra ensuite utiliser cet environnement en lui passant en paramètre un fichier particulier, ou mieux définir un nouvel environnement par exemple :

```

\newenvironment{nota}{%
  \begin{pictonote}{votre fichier}\end{pictnote}}

```

Pour terminer cette section sur les nota il faut savoir que cet environnement a un défaut : Si un nota contient un saut de paragraphe, un espace est de nouveau laissé libre pour un pictogramme

Oui oui comme dans celui-ci, vous voyez bien qu'il y a un emplacement prévu pour le pictogramme, alors qu'ici on n'a pas vraiment l'intention d'en mettre un, non ? Pour éviter ce genre de désagrément il faut réinitialiser les retraits au niveau T_EX grâce aux incantations vaudoues suivantes :

```
\par\parshape=1\@totalleftmargin\linewidth
```

qui pourra faire l'objet d'une commande à insérer manuellement au début de chaque nouveau paragraphe, comme je viens de le faire ici :-)

11.3 Des citations

11.3.1 Épigraphe

Les épigraphes provocatrices de ce manuel ont été produites par un environnement que j'ai nommé *epigraphe*. Par exemple au début du code L^AT_EX du chapitre 2, on trouve :

```
\chapter{Ce qu'il faut savoir}
\label{chap-savoir}
\begin{epigraphe}{Les proverbes Pr \textbf{21} 11}
  Quand on châtie le railleur, le simple s'assagit ;\
  quand on instruit le sage, celui-ci gagne en savoir.
\end{epigraphe}
```

L'environnement *epigraphe* est défini comme suit :

```
\newenvironment{epigraphe}[1]
{% clause begin
  \vspace*{-1.5cm}%
  \small\ssfamily% mise en évidence
  \savebox{\nomepigraphe}{\#1}% une boîte pour sauvegarder
                           % l'origine de la citation
  \slshape% tout est penché
  \begin{changemargin}{0pt}{-2cm}% on se met au large
    \begin{flushright}}% tout est poussé à droite
  {% clause end
    \[4pt]\usebox{\nomepigraphe}.% insertion de l'origine
  \end{flushright}%
  \end{changemargin}\par\vspace*{0.6cm}}
```

Il faut bien entendu **déclarer la boîte** qu'on utilise pour sauvegarder l'origine de notre citation :

```
\newsavebox{\nomepigraphe}
```

Les blanc verticaux (*\vspace*) insérés avant et après cet environnement permettent de caler correctement l'épigraphe entre le début du chapitre et la minitable des matières.

11.3.2 Citations

Quelques citations parsèment le manuel que vous avez sous les yeux. Elles ont été produites avec un environnement fait maison :

```

XXXXXXXXXXXXXXXXX
\begin{unecitation}[Georges \textsc{Bataille}]
  La vieillesse renouvelle la terreur à
  l'infini. Elle ramène l'être sans finir au
  commencement. Le commencement qu'au bord
  de la tombe j'entrevois est le \emph{porc}
  qu'en moi la mort ni l'insulte ne peuvent
  tuer. La terreur au bord de la tombe est
  divine et je m'enfonce dans la terreur dont
  je suis l'enfant.
\end{unecitation}
XXXXXXXXXXXXXXXXX

```

XXXXXXXXXXXXXXXXX

« La vieillesse renouvelle la terreur à l'infini. Elle ramène l'être sans finir au commencement. Le commencement qu'au bord de la tombe j'entrevois est le **porc** qu'en moi la mort ni l'insulte ne peuvent tuer. La terreur au bord de la tombe est divine et je m'enfonce dans la terreur dont je suis l'enfant. »

Georges BATAILLE

XXXXXXXXXXXXXXXXX

Nous allons créer pas à pas cet environnement de manière à mettre le doigt sur quelques problèmes classiques auxquels on peut être confronté avec monsieur L^AT_EX. On va définir l'environnement `citation` en s'appuyant sur celui du paragraphe 9.6 page 125 (permettant de faire une remarque encadrée) et sur celui du § 4.5.2 page 64 qui produit une citation avec son auteur :

```

% un boite pour l'auteur de la citation
\newsavebox{\auteurcitation}
\newsavebox{\boitecitation}
\newenvironment{citationi}[1]{% clause begin
  % on sauve l'argument 1 pour l'auteur
  \savebox{\auteurcitation}{#1}%
  \begin{lrbox}{\boitecitation}
    \begin{minipage}{.8\linewidth}
      \small\slshape% on passe en petit et penché
    {% clause end : on pousse l'auteur de la citation à droite
      \par\mbox{}{\hfill\usebox{\auteurcitation}}
    }
    \end{minipage}
  \end{lrbox}
  \begin{center}
    \usebox{\boitecitation}
  \end{center}
}

```

Ce qui donne :

```

Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
\begin{citationi}{%
  Michel \textsc{Bakounine}, 1845}
  Dans presque tous les pays les femmes
  sont esclaves ; tant qu'elles ne seront
  pas complètement émancipées, notre propre
  liberté sera impossible.
\end{citationi}
Après après après après après après après
après après après après après après après

```

Avant avant avant avant avant avant avant
 avant avant avant avant avant avant avant
*Dans presque tous les pays les femmes
 sont esclaves ; tant qu'elles ne seront
 pas complètement émancipées, notre
 propre liberté sera impossible.*
 Michel BAKOUNINE, 1845
 Après après après après après après après
 après après après après après après après

On peut également changer l'indentation du paragraphe dans la minipage (qui vaut par défaut 0pt dans cet environnement) en modifiant dans l'environnement la valeur de la longueur `\parindent` :

```
...
\begin{lrbox}{\boitecitation}
  \begin{minipage}{.8\linewidth}%
    \setlength{\parindent}{10pt}%    ← pour indenter la 1re ligne
  ...
```

On insère ensuite des guillemets en début et fin de citation, avec le code suivant :

```
...
\begin{minipage}{.8\linewidth}%
  \setlength{\parindent}{10pt}%
  \small\slshape«\ignorespaces% on passe en petit et penché
{\unskip»
  \par\mbox{} \hfill\usebox{\auteurcitation}
  ...
```



On se réfèrera aux paragraphes 9.2.1 et 9.2.1 page 108 pour la signification des commandes `\ignorespaces` et `\unskip`.

Ce qui donne :

```
Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
\begin{citationiii}{%
  Michel \textsc{Bakounine}, 1845}
  Dans presque tous les pays les femmes
  sont esclaves ; tant qu'elles ne seront
  pas complètement émancipées, notre propre
  liberté sera impossible.
\end{citationiii}
Après après après après après après après
après après après après après après après
```

```
Avant avant avant avant avant avant avant
avant avant avant avant avant avant avant
  « Dans presque tous les pays les
  femmes sont esclaves ; tant qu'elles ne
  seront pas complètement émancipées,
  notre propre liberté sera impossible. »
  Michel BAKOUNINE, 1845
Après après après après après après après
après après après après après après après
```

Ensuite — on y est presque — on se propose de rendre l'argument « auteur de la citation » optionnel de manière à pouvoir produire une citation sans auteur si besoin est. L'idée est de déclarer un booléen de manière à mémoriser le fait qu'un auteur est présent ou pas :

```
\newboolean{auteurcitationpresent}
```

On modifie ensuite la définition de l'environnement comme suit :

```
\newenvironment{unecitation}[1][] {% argument optionnel vide par défaut
  % Clause begin :
  % on note si on a un auteur pour la citation ou pas
  \ifthenelse{\equal{#1}{}}{%
    \setboolean{auteurcitationpresent}{false}}{%
    \setboolean{auteurcitationpresent}{true}%
    \savebox{\auteurcitation}{#1}% on le sauve si nécessaire
  ...
```

Puis on modifie la clause `\end` de l'environnement en insérant l'auteur uniquement s'il est présent en argument :

```
{ »% clause end de l'environnement
  % s'il y a un auteur on le met poussé tout à droite
  \ifthenelse{\boolean{auteurcitation}}{%
    {\par\nopagebreak\hfill\usebox{\auteurcitation}}
  }% sinon on ne fait rien ...
```

Enfin, on met la citation sur un fond. On peut par exemple déclarer cette couleur grâce au package `xcolor` :

```
\definecolor{coulcitation}{rgb}{0.60,0.70,0.90}%
```

Il suffit alors de remplacer la commande `\usebox` de la clause end par quelque chose du genre :

```
\setlength{\fboxsep}{10pt}%
\colorbox{coulcitation}{\usebox{\boitecitation}}
```

Ce qui donnera finalement :

avant avant avant avant avant avant avant
`\begin{citationiv}[Pierre \textsc{Desproges}]`
 Il était tellement obsédé qu'à la fin
 il sautait même des repas.
`\end{citationiv}`
 Après après après après après après après

avant avant avant avant avant avant avant

*« Il était tellement obsédé qu'à la fin
 il sautait même des repas. »*

Pierre DESPROGES

Après après après après après après après

11.4 Des lettrines

Les documents soignés font souvent appel aux *lettrines* qui permettent, selon les règles d'usage en typographie, de produire la première lettre du chapitre en gros, ainsi que le mot ou groupe de mots qui suit. Par exemple :

`\lettrine{Les documents}` soignés font souvent
 appel aux lettrines, qui selon les règles
 d'usage en typographie...

LES DOCUMENTS soignés font souvent appel
 aux lettrines, qui selon les règles d'usage en
 typographie...

Il y a deux difficultés dont une a déjà été surmontée :

- comment appliquer un traitement à une seule lettre d'un argument d'une commande ;
- comment décaler les lignes pour laisser la place à la lettrine. On utilisera pour cela la commande `\parshape` comme pour le nota (§ 11.2 page 158).

11.4.1 La commande `\glurps` ou un pas vers T_EX

Au niveau de L^AT_EX c'est la commande `\newcommand` qui permet de créer de
~~nouvelles~~ LesA

On remarque donc que si on ne délimite pas explicitement les arguments avec les caractères { et }, le premier argument (#1) est remplacé par le premier caractère rencontré, le deuxième argument (#2) par le deuxième caractère, etc. Encore plus intéressant, on peut définir très simplement le format d'appel de la commande, par exemple :

```
\def\bidule#1#2/{(1=#1) et (2=#2)}
```

Ici, on indique que pour appeler la commande \bidule il faut lui faire suivre deux arguments suivis du caractère /.

```
\bidule abc/d      (1=a) et (2=bc)d
\bidule ab/cd      (1=a) et (2=b)cd
```

Par conséquent cette dernière commande prendra comme premier argument, le premier caractère rencontré, et comme deuxième : tout ce qu'elle trouve jusqu'au caractère /. On peut donc créer une ébauche de commande pour une lettrine :

```
\def\glurps#1#2/{\Huge#1}\textsc{#2}}
\newcommand{\lettrinedev}[1]{\glurps#1/}
\lettrinedev{Bon bé} ouala le travail
```

On va même pousser le vice jusqu'à mettre la grosse lettre un peu plus bas :

```
\def\glurps#1#2/{%
  {\Huge#1}%
  \raisebox{\baselineskip}{\textsc{#2}}}
\newcommand{\lettrinedev}[1]{\glurps#1/}
\lettrinedev{Bon bé} ouala le travail
```

11.4.2 Insertion de la lettrine dans un paragraphe

Pour insérer la lettrine dans un paragraphe on aura recours à la commande \parshape. La figure 11.3 montre que l'on doit définir deux dimensions pour insérer la lettrine :

1. l'indentation de la première ligne, correspondant à la largeur de la «grosse lettre» plus celle de la suite de la lettrine ;
2. l'indentation de la deuxième ligne, correspondant à la largeur de la «grosse lettre» éventuellement augmentée d'une espace pour aérer un peu.

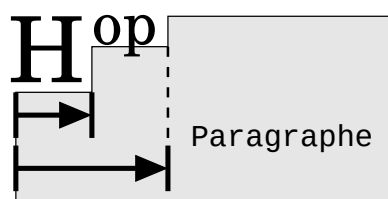


FIG. 11.3 – Insertion de la lettrine dans un paragraphe

On définit les dimensions suivantes :

- `\indletH` et `\larligH` respectivement l'indentation et la largeur de la première ligne (la ligne «du Haut») du paragraphe contenant une lettrine ;
- `\indletB` et `\larligB` la même chose pour la deuxième ligne (la ligne «du Bas»)

Ceci permet d'écrire quelque chose du genre :

```
\setlength{\indletB}{.8cm}% au pif
\setlength{\larligB}{\textwidth-\indletB}
\setlength{\indletH}{1.5cm}% au pif
\setlength{\larligH}{\textwidth-\indletH}
\parshape=3
\indletH\larligH
\indletB\larligB
0pt\textwidth
\noindent Ce paragraphe est prêt à recevoir
une jolie lettrine qui occupera deux lignes
environ voire même exactement...
```

Ce paragraphe est prêt à recevoir une
jolie lettrine qui occupera deux lignes en-
viron voire même exactement...

L'emplacement est prêt, il reste à insérer la «grosse lettre» et ce qui suit à la bonne place. On commence par créer une commande pour la fonte à utiliser :

```
\DeclareFixedFont{%
  \lettrinefont}{T1}{ppl}{m}{n}{2\baselineskip}
{\lettrinefont Pour écrire les grosses lettres}
```

**Pour écrire les
grosses lettres**

Ensuite on va modifier la commande `\glurps`⁵ et la commande `\lettrine` pour qu'elles calculent elles-mêmes les dimensions définies ci-avant (`\indletH`, `\larligH`, ...).

```
\newsavebox{\lalettrine}% une boîte pour la lettrine
\def\creerlettrine#1#2/{%
  \savebox{\lalettrine}{%
    {\lettrinefont#1}\raisebox{\baselineskip}{\textsc{#2}}}%
  \settowidth{\indletB}{\lettrinefont#1}%
  \settowidth{\indletH}{\usebox{\lalettrine}}}
```

La commande `\creerlettrine` (digne héritière de `\glurps`) sauve donc la lettrine dans une boîte et en profite pour sauvegarder la largeur de la «grosse lettre» dans la dimension `\indletB`, et la dimension de l'ensemble dans `\indletH`. Tentons maintenant, une première version de la lettrine :

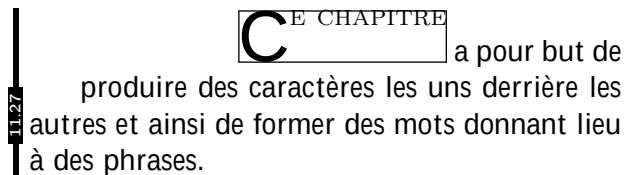
```
\newcommand{\lettrineI}[1]{%
  \creerlettrine#1/%
  \setlength{\larligH}{\textwidth-\indletH}%
  \setlength{\larligB}{\textwidth-\indletB}%
  \parshape=3\indletH\larligH\indletB\larligB%
  0cm\textwidth%
  \noindent\usebox{\lalettrine}}
```

Qui donne :⁶

⁵Vous pourrez noter qu'il est tout à fait inadmissible d'utiliser des noms aussi ridicules pour les commandes que vous serez amené à définir...

⁶Des traits ont été ajoutés pour bien situer la boîte englobant la lettrine.

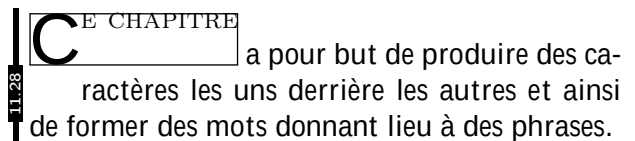
`\lettrineI{Ce chapitre}` a pour but de produire des caractères les uns derrière les autres et ainsi de former des mots donnant lieu à des phrases.



Les lecteurs très attentifs auront noté que ce nota n'est pas à la bonne place. Il semblerait que des translations horizontales et verticales soient nécessaires. On commencera ici par se décaler vers la gauche. À la dernière ligne de la définition de la commande `\lettrine`, on écrira donc :

`\noindent\hspace{-\indletH}% décalage équivalent à la largeur totale`
qui produira :

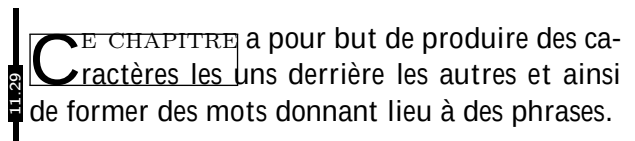
`\lettrineII{Ce chapitre}` a pour but de produire des caractères les uns derrière les autres et ainsi de former des mots donnant lieu à des phrases.



Puis on intègre la translation verticale :

`\noindent\hspace{-\indletH}%`
`\raisebox{-\baselineskip}[0pt][0pt]{\usebox{\lalettrine}}`
qui produira :

`\lettrineIII{Ce chapitre}` a pour but de produire des caractères les uns derrière les autres et ainsi de former des mots donnant lieu à des phrases.



Les lecteurs encore éveillés auront remarqué que la « grosse lettre » est un peu trop rapprochée du texte. On peut donc augmenter légèrement la dimension `\indletB`. Voici finalement le code de la lettrine :

```
\newcommand{\lettrine}[1]{%
  \creerlettrine#1/%
  \addtolength{\indletB}{3pt}% pour avoir un peu d'espace
  \setlength{\larligH}{\textwidth-\indletH}%
  \setlength{\larligB}{\textwidth-\indletB}%
  \parshape=3%
  \indletH\larligH\indletB\larligB0cm\textwidth%
  \noindent\hspace{-\indletH}%
  \raisebox{-\baselineskip}[0pt][0pt]{\usebox{\lalettrine}}}
```

11.5 Un sommaire

C'est une pratique courante que le sommaire d'un document rédigé en français soit placé en tête de document et contienne un condensé de la table des matières. Cette dernière est quant à elle généralement insérée à la fin. En fouillant dans le fichier `book.cls` on trouve une instruction commune aux commandes `\tableofcontents` et `\listoffigures` : la commande `\@starttoc`.

```
\@starttoc{toc}
```

pour la commande `\tableofcontents` et :

```
\@starttoc{lof}
```

pour la commande `\listoffigures`. La commande interne de L^AT_EX `\@starttoc` permet de commencer un table (matières, figures, etc.) à partir d'un fichier auxiliaire portant l'extension donné en argument, ici `toc` pour la table des matières et `lof` pour la liste des figures. Nous avons donc créé dans un premier temps la commande `\sommaire` comme suit :

```
\newcommand{\sommaire}{%
  \chapter*{Sommaire}
  \@starttoc{som}}
```

Le fichier portant l'extension `som` contiendra les entrées du sommaire. L'étape suivante consiste à remplir le fichier `som`. Pour cela, il faut savoir que les commandes `\chapter`, `\section`, etc. font toutes appel à une commande pour insérer une entrée dans la table des matières. Ainsi quand on écrit :

```
\section{Bidule truc muche}
```

il sera fait appel à la commande :

```
\addcontentsline{toc}{section}{Bidule truc muche}
```

pour insérer le titre dans la table des matières. De même lorsqu'on écrit :

```
\chapter{Machin chose}
```

il sera fait automatiquement appel à :

```
\addcontentsline{toc}{chapter}{Machin chose}
```

Cette dernière commande écrit dans le fichier de table des matières (portant donc l'extension `toc`) la ligne :

```
\contentsline{chapter}{\numberline {1}Machine chose}{3}{chapter.1}
```

Et (Hercule POIROT a eu du pain sur la planche), il se trouve que cette dernière commande, fait finalement appel à une commande de la forme :

```
\l@type d'entrée
```

pour produire l'entrée dans la table. Par exemple la commande précédente fait appel à `\l@chapter`. Ceci parce qu'on trouve la définition suivante :

```
\def\contentsline#1{\csname l@#1\endcsname}
```

dans `latex.ltx`. Nous passons volontairement sous silence le détail de ces commandes, et on se contentera ici de savoir que la commande `\l@section` produit une entrée de table de matières pour les sections, `\l@subsection` produit une entrée pour les subsections, etc. Pour finir notre sommaire, il reste à écrire dans le fichier de sommaire. Nous avons choisi d'insérer dans le sommaire les titres de parties, chapitres et sections. Pour arriver à nos fins, nous avons ajouté cette insertion à la commande `\addcontentsline` originale :

```
% sauvegarde de l' originale
\let\acLORIG\addcontentsline
% redéfinition
```

```
\renewcommand{\addcontentsline}[3]{%
  \aclORIG{#1}{#2}{#3}% appel de l'originale
  \ifthenelse{% on insère sections, chapitres et parties
    \equal{#2}{section} \or \equal{#2}{chapter}
    \or \equal{#2}{part}}{%
    \aclORIG{som}{#2}{#3}}{}}
```



Il faut noter que notre sommaire sera mis en page exactement de la même manière que la table des matières, puisque tous deux font appel à la même commande interne :

```
\aclORIG{som}{section}{...}
```

c'est donc la commande `\l@section` qui sera appelée pour mettre en page l'entrée. Pour mettre en page le sommaire différemment, il aurait fallu écrire :

```
\aclORIG{som}{somsection}{...}
```

puis définir la commande `\l@somsection` pour mettre en page les entrées de sections dans le sommaire...

11.6 Un glossaire

Il est souvent utile d'agrémenter un document d'un glossaire ayant pour but de rendre moins mystérieux les termes du jargon qu'on appelle aussi vocabulaire technique. Nous proposons ici une méthode s'appuyant sur le programme `makeindex` présenté au paragraphe 6.3 page 82 et sur le paragraphe 10.1 page 127 présentant les outils permettant de changer l'allure de l'index.

11.6.1 Tordre le cou à `makeindex`

De même que pour la création d'un index, on doit mettre dans le préambule du document la commande `\makeglossary`. Cette commande demande à monsieur $\text{\LaTeX}$ de bien vouloir créer un fichier portant l'extension `glo`, réceptacle des entrées de glossaire. On pourra écrire :

```
\glossary{machin chouette chose}
```

pour ajouter «machin chouette chose» dans le glossaire. En réalité cette commande a pour effet d'ajouter dans le fichier d'extension `glo`, la ligne :

```
\glossaryentry{machin chouette chose}{n° page}
```

La phase suivante consiste à produire le glossaire proprement dit grâce au programme `makeindex` et au fichier d'entrées de glossaire :

```
| makeindex -sstyle glossaire document.glo -odocument.glx
```

qui produit, en utilisant comme fichier de style `gglo.ist` le glossaire suivant :

```
\begin{theglossary}
\item machin chouette chose\pfill 27
\end{theglossary}
```

On note donc que le glossaire — qu'il va falloir insérer explicitement dans notre document — est constitué par l'environnement `theglossary` et que chaque entrée donne lieu à un `\item` et son numéro de page (ici 27 pour l'exemple). Pour arriver à nos fins, il nous faudra :

1. définir l'environnement `theglossary` car rien n'est fait par défaut dans $\text{\LaTeX}$;
2. produire les entrées avec un terme suivi de sa définition, sous la forme :

```
\begin{theglossary}
\item[terme] blabla de définition
\end{theglossary}
```

3. supprimer les numéros de pages puisqu'ils n'apparaissent pas dans un glossaire.

Ces trois tâches font l'objet des paragraphes suivants.

11.6.2 Un environnement pour le glossaire

Au paragraphe 9.5.6 page 124 du chapitre 9, nous avons proposé une liste particulière permettant de présenter les entrées dans une boîte avec une ombre. La définition était la suivante :

```
\newenvironment{leglossaire}{\begin{list}}{\end{list}}{\%
\setlength{\labelwidth}{.5\textwidth}\%
\setlength{\labelsep}{-.8\labelwidth}\%
\setlength{\itemindent}{\parindent}\%
\setlength{\leftmargin}{25pt}\%
\setlength{\rightmargin}{0pt}\%
\setlength{\itemsep}{.8\baselineskip}\%
\renewcommand{\makelabel}[1]{\boiteentreeglossaire{##1}}}%
}
```

où `\boiteentreeglossaire` est :

```
\newcommand{\boiteentreeglossaire}[1]{\%
\parbox[b]{\labelwidth}{\%
\setlength{\fboxsep}{3pt}\%
\setlength{\fboxrule}{.4pt}\%
\shadowbox{\sffamily#1}\hfill\mbox{}}}
```

On se reportera au paragraphe mentionné ci-dessus pour les explications de ces commandes. En tous les cas on aura :



```
\begin{leglossaire}
\item[Sphère] Patate bien régulière.
\end{leglossaire}
```



Sphère

Patate bien régulière.

11.6.3 Produire le fichier .glx

De manière à ce que `makeindex` produise le fichier d'extension `glx` avec cet environnement on doit écrire le fichier de style suivant :

```
preamble "\n\begin{leglossaire}"
postamble "\n\end{leglossaire}"
```

Doit également apparaître dans ce fichier de style — que l'on nommera par exemple `glossaire.ist` — le mot clef désignant les entrées de glossaire dans le fichier `.glo` :

```
keyword "\\glossaryentry"
```

Pour que chaque entrée soit constituée d'un terme et de sa définition, nous avons défini la commande suivante :

```
\newcommand{\entreeglossaire}[2]{\glossary{[#1] #2}}
```

De telle sorte que :

```
\entreeglossaire{Sphere}{Patate bien régulière.}
```

aura pour effet d'écrire :

```
\glossary{[Sphère] Patate bien régulière.}
```

dans le fichier `.glo`. Après appel de `makeindex` :

```
| makeindex -s glossaire.ist document.glo -odocument.glx
```

on aura dans le fichier `.glx` :

```
\begin{leglossaire}
\item [Sphère] Patate bien régulière., n° de page
\end{leglossaire}
```

Si vous m'avez bien suivi jusqu'ici, vous devriez râler et me dire : « faudrait p'têt' voir à enlever cette vilaine virgule et ce numéro de page... » Certes. La virgule est automatiquement mise par `makeindex` comme délimiteur « de premier niveau » entre une entrée d'index et le numéro de page où apparaît cette entrée. Pour utiliser autre chose qu'une virgule (ici rien en l'occurrence) on écrit dans le fichier de style :

```
delim_0 ""
```

Il reste à régler le cas du numéro de page. La solution que j'ai adoptée consiste à utiliser une commande « absorbante » en guise de mise en forme du numéro de page (cf. § 6.3.3 page 83). Voici la commande `\entreeglossaire` modifiée :

```
\newcommand{\pasdenumerodepage}[1]{% « mange » l'argument
\newcommand{\entreeglossaire}[2]{%
\glossary{[#1] #2|pasdenumerodepage}}
```

ceci permet de créer le fichier `.glx` de la forme :

```
\begin{leglossaire}
\item [Sphère] Patate bien régulière.\pagedenumerodepage{27}
\end{leglossaire}
```

qui enverra le numéro de page (ici 27 pour l'exemple) dans le vide intersidéral.

11.6.4 Recollons les morceaux

Pour en finir avec le glossaire, il nous faut créer une commande suffisamment souple qui aura essentiellement pour but de produire un chapitre contenant le glossaire, c'est-à-dire d'insérer dans le document le fichier `.glx`. On peut décomposer les tâches à effectuer par la commande en question, comme suit :

- créer un nouveau chapitre (avec comme titre «glossaire»);
- lire les entrées de glossaire (commande `\entreeglossaire`) depuis un fichier;
- insérer le fichier `.glx`

Voici la commande effectuant ces traitements :

```
\newcommand{\printglossary}[1][glossaire.tex]{%
  \chapter*{Glossaire}
  \label{chap-glossaire}
  \markboth{Glossaire}{Glossaire}% entête de page
  % insertion dans la table des matières :
  \addcontentsline{toc}{chapter}{Glossaire}
  % insertion des entrées de glossaire :
  \InputIfFileExists{#1}{%
    \typeout{Données du glossaire}}{%
    \typeout{Pas de fichier glossaire.tex}}
  % insertion du fichier .glx
  \InputIfFileExists{\jobname.glx}{%
    \typeout{Glossaire trié}}{%
    \typeout{Pas de fichier \jobname.glx}}
}
```

On notera les points suivants concernant cette commande :

- le fichier d'entrée de glossaire est par défaut le fichier nommé `glossaire.tex`;
- on a décidé d'insérer le glossaire dans la table des matières;
- pour insérer un fichier, on fait appel à la commande `\InputIfFileExists` définie dans le format $\LaTeX$ ayant la forme suivante :

```
\InputIfFileExists{fichier à inclure}
{code  $\LaTeX$  si le fichier existe}
{code  $\LaTeX$  si le fichier n'existe pas}
```

- Enfin `\jobname` contient le nom du fichier (ou document maître) en cours de compilation et la commande `\typeout` affiche un message sur la console de compilation.

Enfin, le fichier `glossaire.ist` contient finalement :

```
delim_0 ""
preamble "\n\\begin{leglossaire}"
postamble "\n\\end{leglossaire}"
keyword "\\glossaryentry"
```



11.7 Des onglets

J'avais initialement prévu de mettre en page ce document sur du papier plus petit que le standard A4 pour ensuite le massicoter. D'où l'idée de créer des onglets, petits carrés colorés se retrouvant au ras de la feuille après massicotage. Le fait que l'établissement dans lequel je travaille s'est séparé de son massicot hydraulique d'une part, et que d'autre part tout le monde n'a pas facilement accès à ce type de matériel m'a fait changer d'avis quant au bien fondé que ce document devait impérativement être massicoté. Les onglets sont malgré tout restés dans le document, bien que leur place n'est plus tout à fait au ras de la feuille. Voici comment ils ont été générés...

11.7.1 Idée retenue

Le cahier des charges est le suivant :

- les onglets apparaissent sur chaque page du côté opposé à la reliure ;
- ils sont produits «à l'envers» sur les pages paires ;
- ils doivent être à une hauteur proportionnelle au numéro du chapitre.

D'où l'idée :

- d'utiliser les fonctionnalités du package fancyhdr pour produire les onglets ;
- utiliser des translations verticales pour les positionner ;

11.7.2 Les boîtes dans la marge

De manière à produire les numéros de chapitres dans la marge, j'ai simplement créé des boîtes paragraphe de largeur et hauteur imposées :⁷

```
\newlength{\ongletwidth}
\newlength{\ongletheight}
\setlength{\ongletheight}{32pt}
\setlength{\ongletwidth}{.96cm}
```

Voici la commande produisant la boîte :

```
\newcommand{\b@iteonglet}{%
  \colorbox[gray]{.7}{% une boîte avec un fond gris contenant
    % la boîte paragraphe de largeur et hauteur fixée :
    \parbox[t]{\ongletheight}[s]{\ongletwidth}{%
      \vfill%
      \centering%
      % on applique un e et miroir selon la parité de la page
      \ifthenelse{\isodd{\value{page}}}{%
        \ongletfont\thechapter}{%
        \reflectbox{\ongletfont\thechapter}}%
      \vfill}}}
```

On notera l'utilisation de la commande `\reflectbox` du package `graphicx`. On pourra également remarquer que le contenu de la boîte paragraphe est centré en hauteur grâce à deux ressorts verticaux et qu'on insère finalement dans cette boîte le numéro du chapitre dans une **mystérieuse fonte** enclenchée par la commande `\ongletfont`.

Et voici la boîte : `\b@iteonglet...`

Et voici la boîte : ...

11.31

11

11

Il est important de noter que les boîtes produites par `\colorbox` sont assujetties à la dimension `\fboxsep` :

```
\setlength{\fboxsep}{15pt}
En voici une autre : \b@iteonglet...
```

En voici une autre : ...

11.32

11

⁷On remarquera cette maladie des informaticiens de nommer les variables avec des noms à moitié en français et en anglais...

11.7.3 Position des onglets

Pour positionner les onglets, il faut effectuer deux translations :

- une horizontale pour pousser l’onglet jusqu’au bout de la zone de note marginale ;
- une verticale en fonction du numéro de chapitre.

On utilisera le mécanisme de définition des en-têtes de page du package fancyhdr pour positionner les onglets. L’idée est simple, on écrit :

```
\fancyhead[RO]{\bfseries\thepage\onglet}
\fancyhead[LE]{\onglet\bfseries\thepage}
```

pour dire qu’en plus du numéro de page en gras, on mettra dans l’en-tête le résultat de la commande `\onglet` (à droite sur les pages impaires et à gauche sur les pages paires). Nous allons voir comment construire pas à pas cette commande.

Positionnement horizontal

Dans le cas des pages impaires, il faut dans un premier temps pousser la boîte de l’onglet vers la droite, et vers la gauche pour les pages paires. Qu’à cela ne tienne :⁸

```
\newcommand{\ongletI}{%
\ifthenelse{\isodd{\value{page}}}{%
% page impaire
\hspace*{\marginparwidth}\hspace*{\marginparsep}}{%
% page paire
\hspace*{-\marginparwidth}\hspace*{-\marginparsep}}%
\b@iteonglet}
```

Voici ce que donnerait cette commande sur cette page :

176



Pour placer correctement le numéro de la page dans l’entête, il faut donc « leurrer » L^AT_EX en utilisant une boîte de largeur nulle contenant la boîte de l’onglet et les espaces de translation horizontale :

```
\newcommand{\ongletII}{%
\makebox[0pt][l]{%
\ifthenelse{\isodd{\value{page}}}{%
\hspace*{\marginparwidth}\hspace*{\marginparsep}}{%
\hspace*{-\marginparwidth}\hspace*{-\marginparsep}}%
\b@iteonglet}}
```

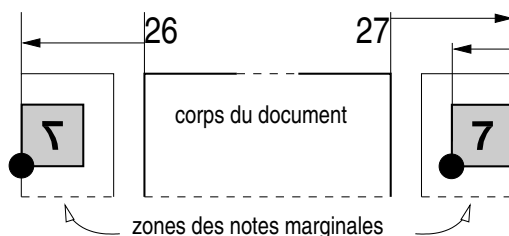
qui donnerait sur cette page :

176



⁸Je suis toujours très ému de commencer une phrase par le beau ‘Q’ de la police Computer Modern...

Pour les pages impaires, s'il on veut pousser la boîte de l'onglet contre le bord de la zone réservée aux notes marginales, on doit également tenir compte de la largeur totale de cette boîte, comme le montre les points de références dans le schéma ci-dessous :



Par conséquent, le positionnement horizontal est obtenu grâce à la commande :

```
\newcommand{\ongletIII}{%
  \makebox[0pt][l]{%
    \ifthenelse{\isodd{\value{page}}}{% page impaire
      \hspace*{\marginparwidth}\hspace*{\marginparsep}%
      \hspace*{-\ongletwidth}\hspace*{-2\fbboxsep}}{% page paire
      \hspace*{-\marginparwidth}\hspace*{-\marginparsep}}%
    \b@iteonglet}}
```

puisque à la largeur de la boîte de l'onglet, il faut rajouter deux fois la dimension `\fbboxsep` induite par la commande `\colorbox`.

Positionnement vertical

Reste maintenant à traiter le problème du positionnement vertical... La commande `\raisebox` va nous permettre de positionner verticalement l'onglet. De plus avec la forme suivante :

```
\raisebox{déplacement}[0pt][0pt]{objet à déplacer}
```

on fait croire à $\text{\LaTeX}$ que l'**objet à déplacer** a une hauteur nulle. Il n'y aura par conséquent pas de déplacement des objets aux alentours et en particulier ceux constituant l'en-tête de la page. Par exemple, en écrivant :

```
\newcommand{\ongletIV}{%
  \makebox[0pt][l]{%
    \ifthenelse{\isodd{\value{page}}}{%
      \hspace*{\marginparwidth}\hspace*{\marginparsep}%
      \hspace*{-\ongletwidth}\hspace*{-2\fbboxsep}}{%
      \hspace*{-\marginparwidth}\hspace*{-\marginparsep}}%
      \raisebox{-5cm}[0pt][0pt]{\b@iteonglet}}}
```

On place à partir d'ici un onglet à cinq centimètres vers le bas :

La méthode choisie pour définir le placement de la boîte est la suivante : pour le chapitre de numéro c :

- se déplacer vers le bas d'une dimension fixe donnée d_f ;

- ajouter à ce déplacement un déplacement proportionnel à c . Le déplacement de l'onglet pour le chapitre c peut s'écrire :

$$c \times \text{hauteuronglet} \times \alpha$$

où α est un facteur permettant d'espacer les onglets. Si $\alpha = 1$, les onglets seront espacés d'exactly la longueur de la boîte, avec $\alpha = 2$ ils seront séparés par un espacement égal à deux fois la hauteur de l'onglet, etc.

Voici pour le premier déplacement :

```
% position de la première étiquette
\newlength{\ongletvshift}
\setlength{\ongletvshift}{2cm}
```

Puis pour le facteur alpha :

```
\newcommand{\ongletsep}{1.37}
```

On déclare une dimension permettant de positionner l'onglet :

```
\newlength{\ongletpos}
```

On peut maintenant écrire, la commande `\onglet` :

```
\newcommand{\onglet}{%
  \makebox[0pt][l]{%
    \ifthenelse{\isodd{\value{page}}}{%
      \hspace*{\marginparwidth}\hspace*{\marginparsep}%
      \hspace*{-\ongletwidth}\hspace*{-2\fbboxsep}%
    }{%
      \hspace*{-\marginparwidth}\hspace*{-\marginparsep}%
    }%
    % calcul de la position verticale
    \setlength{\ongletvpos}{%
      -\ongletvshift
      -\ongletheight*\real{\thechapter}*\real{\ongletsep}}%
    % positionnement de l'onglet
    \raisebox{\ongletvpos}[0pt][0pt]{\b@iteonglet}}
```

Ouf! ça marche... Oui pour le chapitre ne faisant pas partie des annexes. Mais pour ceux en faisant partie, numérotés A, B, etc. ça ne collera pas. Puisqu'on ne pourra pas calculer la position verticale en fonction de ces numéros. En petit coup d'œil dans `book.cls` nous confirme cela :

```
\newcommand\appendix{\par
\setcounter{chapter}{0}%
\setcounter{section}{0}%
[...]
\gdef\thechapter{\@Alph{c}@chapter}}
```

on peut comprendre ici que lorsqu'on commence les annexes avec la commande `\appendix`, le compteur de chapitre est remis à zéro, et produit en lettre majuscule. Il faut donc trouver une parade pour que l'onglet continue à se décaler même après les annexes. La solution que j'ai adoptée consiste simplement à utiliser un nouveau compteur permettant de numéroté les chapitres et les annexes. Pour cela, on le déclare :

```
\newcounter{chapitre}
```

On précise tout de suite qu'on veut le produire en caractère arabe :

```
\renewcommand{\thechapitre}{\arabic{chapitre}}
```

On ajoute la mise à zéro de ce compteur dans la commande `\frontmatter` :

```
\renewcommand\frontmatter{%
  \cleardoublepage
  \setcounter{chapitre}{1}
  [...]
}
```

Puis à chaque appel de la commande `\chapter`, on incrémente le compteur :

```
\renewcommand{\chapter}{%
  \cleardoublepage
  \stepcounter{chapitre}
  \thispagestyle{plain}
  [...]
}
```

11.8 Exemples $\text{\LaTeX}$

Pour clore ce chapitre, nous présenterons l'environnement `\ltxenv{ltxexemple}` permettant d'introduire des exemples de code `\LaTeX{}` et le résultat dans le document...

Pour clore ce chapitre, nous présenterons l'environnement `ltxexemple` permettant d'introduire des exemples de code $\text{\LaTeX}$ et le résultat dans le document...

11.8.1 Outils nécessaires

Le package `fancyvrb` propose deux fonctionnalités axées autour des fichiers. Tout d'abord, la commande `\VerbatimInput` permet d'insérer le contenu d'un fichier. Par exemple :

```
\VerbatimInput[lastline=4]{corps/jouets.tex}
```

donne (jouets.tex est le fichier source de ce chapitre) :

```
\chapter{De nouveaux jouets}
\label{chap-jouets}
```

```
\begin{epigraphe}{Le Cantique des cantiques Ct \textbf{7} 11}
```

Ensuite, l'environnement `VerbatimOut` réalisant la tâche inverse :

```
\begin{VerbatimOut}{fichier}
  code  $\text{\LaTeX}$ 
\end{VerbatimOut}
```

stockera le code $\text{\LaTeX}$ dans le fichier nommé `fichier`.

Le deuxième outil nécessaire est défini en s'inspirant de la définition des commandes `\settowidth`, `\settoheight`, etc. dans le fichier `latex.ltx`. Il s'agit d'une commande permettant de récupérer la hauteur totale d'un objet, c'est-à-dire sa hauteur additionnée de sa profondeur :

```

\newcommand{\hauteur totale}[2]{%
  \setbox\@tempboxa\hbox{\{#2\}}% sauvegarde de l'objet à mesurer
  \setlength{#1}{\ht\@tempboxa}% récupération de la hauteur
  \addtolength{#1}{\dp\@tempboxa}% à laquelle on ajoute la profondeur
  \setbox\@tempboxa\box\voidb@x}% vidange de la boîte temporaire

```

On notera l'utilisation des commandes $\text{T}_{\text{E}}\text{X}$, $\backslash\text{ht}$ et $\backslash\text{dp}$ renvoyant respectivement la hauteur et la largeur d'une boîte. La commande $\backslash\text{setbox}$ et l'équivalent $\text{T}_{\text{E}}\text{X}$ de $\backslash\text{savebox}$. La boîte $\backslash\@tempboxa$ est une boîte temporaire utilisée par $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ et enfin, la boîte $\backslash\text{voidb@x}$ est une boîte vide.

11.8.2 Le principe de l'environnement `ltxexemple`

L'environnement `ltxexemple` que nous présentons en début de section, est un peu plus complexe que ce que nous avons rencontré jusqu'à maintenant. En effet lorsqu'on écrit :

```
\begin{ltxexemple} contenu \end{ltxexemple}
```

il faut d'une part pouvoir produire `contenu` tel quel (en verbatim), et d'autre part pouvoir l'interpréter c'est-à-dire le traiter comme $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ le ferait. Finalement, on se rend compte qu'il faudrait demander à $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ de traiter *deux fois* `contenu` ce qui n'est pas envisageable. Une solution pour contourner ce problème est précisément de sauvegarder `contenu` dans un fichier pour pouvoir le réutiliser, soit en tant que verbatim, soit pour être interprété par $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. La première difficulté à surmonter est donc de créer un environnement sauvegardant son contenu :

```

\newenvironment{ltxexemple}{%
  \VerbatimEnvironment
  \begin{VerbatimOut}{\jobname.exe}}{% clause begin
  \end{VerbatimOut}}% clause end

```

Ne me demandez pas à quoi sert la commande $\backslash\text{VerbatimEnvironment}$ non documentée dans le package, mais nécessaire au bon fonctionnement de l'environnement défini ci-dessus.

Tout cela est bien joli, mais cet environnement ne fait que sauvegarder son contenu dans un fichier. Il faut donc écrire dans la clause «end» de l'environnement :

```

\end{VerbatimOut}%
\VerbatimInput{\jobname.exe}% le contenu tel quel
\input{\jobname.exe}% le contenu interprété par LaTeX

```

Ainsi :

```

\begin{ltxexemplei}
  du code \LaTeX{}...
\end{ltxexemplei}

```

donne :

```

du code \LaTeX{}...
du code  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ...

```

Il reste donc à modifier la mise en page des deux parties. Ce qui fait l'objet du paragraphe suivant.

11.8.3 Mises en boîte

L'idée de base consiste à mettre dans deux boîtes :

```
\newsavebox{\b@iteentree}
\newsavebox{\b@itesortie}
```

pour stocker l'«entrée» (le code) et la «sortie» (le code interprété par L^AT_EX). On écrira donc dans la clause «end» de l'environnement :

```
[...]
\end{VerbatimOut}%
\begin{ltxexempleenv}% pour agrandir les marges
  \savebox{\b@iteentree}{% sauvegarde du code en verbatim
    \begin{minipage}{.53\linewidth}
      \VerbatimInput{\jobname.exe}
    \end{minipage}}%
  \savebox{\b@itesortie}{% code interprété
    \begin{minipage}{.44\linewidth}
      \setlength{\parindent}{10pt}% par défaut Opt
      \input{\jobname.exe}
    \end{minipage}}%
  \usebox{\b@iteentree}
  \usebox{\b@itesortie}
\end{ltxexempleenv}
```

L'environnement `ltxexempleenv` est analogue à celui que nous avons présenté au paragraphe concernant l'**agrandissement des marges**. La seule différence est qu'il bascule en `\small` et effectue quelques réglages sur les blancs verticaux. On notera que la boîte «entrée» occupera 53% de la largeur de la page et, la boîte «sortie» 44%. Ainsi, le code :

```
\begin{ltxexempleii}
  du code \LaTeX{}...
  \par\noindent
  et c'est tout.
\end{ltxexempleii}
```

Donnera :

```
du code \LaTeX{}...
\par\noindent
et c'est tout.
```

```
du code LATEX...
et c'est tout.
```

Nous avons mis des bordures aux ras des deux boîtes «entrée» et «sortie» pour mettre en évidence leurs dimensions. On peut également noter ici que la clause «begin» de l'environnement est en réalité définie comme suit :

```
\pagebreak[3]% on suggère de changer de page ici
\VerbatimEnvironment%
\begin{VerbatimOut}[gobble=2]{\jobname.exe}
```

L'option `[gobble=2]` permet de «manger» systématiquement deux caractères au début de chaque ligne car tout bon éditeur⁹ ajoute des espaces pour l'indentation du source. Ainsi le code :

⁹Je veux bien sûr parler de *xy*, pardon, Emacs...

```
\begin{lttexempleii}
du code \LaTeX{...}
\par\noindent
et c'est tout.
\end{lttexempleii}
```

donnerait :

```
code \LaTeX{...}
ar\noindent
c'est tout.
```

```
code LATEX... arc'est tout.
```

La suite de la mise en page consiste en la création du trait central. Ceci fait l'objet du paragraphe 11.8.5 page 184. Avant cela nous nous attarderons sur la numérotation des exemples.

11.8.4 Numérotation des exemples

Pour numéroter les exemples, il est nécessaire de déclarer un **compteur** :

```
\newcounter{c@exemple}[chapter]
```

qui sera donc remis à zéro à chaque chapitre. On précise la manière dont il s'affichera lorsqu'on y fera référence :

```
\renewcommand{\thec@exemple}{\thechapter.\arabic{c@exemple}}
```

À chaque appel à l'environnement `lttexemple`, on fera appel à :

```
\refstepcounter{c@exemple}
```

pour incrémenter le compteur et mettre à jour le système de référence. La petite boîte noire que vous avez pu apercevoir au milieu des exemples a pour largeur `\l@rgeurnumex` (définie à 16 points) et est produite par :

```
\newcommand{\affichenumex}{%
\raisebox{-1.7pt}[0pt][0pt]{%
\setlength{\fboxsep}{.7pt}%
\colorbox{black}{%
\makebox[\l@rgeurnumex]{%
\color{white}%
\tiny\texsf{\thec@exemple}}}}}
```

← la boîte noire

← une boîte de largeur fixée (16 pt)

← le contenu en blanc

Ainsi :



La petite boîte (`\affichenumex`)



La petite boîte (`\affichenumex`)

Une nouvelle version de l'environnement `lttexemple` pourrait donc être :

```
\newenvironment{lttexempleiii}{%
\VerbatimEnvironment%
\begin{VerbatimOut}[gobble=2]{\jobname.exa}}{%
\end{VerbatimOut}%
\begin{lttexempleenv}%
\refstepcounter{c@exemple}%
\savebox{\b@iteentree}{ [...] }%
\savebox{\b@itesortie}{ [...] }%
```

← incrémentation du compteur

```

\usebox{\b@iteentree}%
\kern2pt%
\parbox{3pt}{\rotatebox{90}{\affichenumex}}%
\kern2pt%
\usebox{\b@itesortie}
\end{ltxexempleenv}}%

```

qui donnera :

```

\setlength{\fboxsep}{-2pt}
\setlength{\fboxrule}{.5mm}
Ceci est un \fbox{EXEMPLE} idiot...

```

Ceci est un **EXEMPLE** idiot...

La dernière difficulté est de gérer le système de référencement. En effet, on ne peut pas écrire :

Voici un exemple.\label{monexemple}.

Puisque la séquence «\label{monexemple}» apparaîtra dans l'exemple :

Voici un exemple.\label{monexemple}

Voici un exemple.

Ce qui n'est pas souhaitable... La solution adoptée ici a été de passer l'éventuelle étiquette de \label à l'environnement par le truchement d'une commande. On a défini :

```

\newcommand{\l@belex}{} % la valeur courante du label
\newcommand{\labelexemple}[1]{% commande pour la mettre à jour
  \renewcommand{\l@belex}{#1}}

```

Ainsi avant l'utilisation d'un environnement ltxexemple il suffira d'appeler :

\labelexemple{étiquette}

pour pouvoir ensuite y faire référence avec \ref{étiquette} où une commande équivalente. Dans la définition de l'environnement ltxexemple, on a ajouté :

```

\ifthenelse{\equal{\l@belex}{} }{}{% si le label courant n'est pas vide
  \label{\l@belex}% on pose un label

```

qui signifie en français : «si la commande \l@belex est définie à une valeur non vide, on pose une étiquette (commande \label) avec cette valeur». Il faut ensuite repositionner la commande \l@belex à une valeur vide car dans le cas contraire l'étiquette serait définie plusieurs fois (message «Label 'xxx' multiply defined» de L^AT_EX). On pourrait donc écrire :

```

\ifthenelse{\equal{\l@belex}{} }{}{% si le label courant n'est pas vide
  \label{\l@belex}% on pose un label
  \renewcommand{\l@belex}{}

```

ce qui est correct du point de vue de la syntaxe. Cependant la portée de la commande \renewcommand est ici locale au groupe dans lequel le test \ifthenelse intervient. Pour contourner ce problème on utilisera la construction T_EX :

```

\global\def\l@belex{}

```

en lieu et place du \renewcommand pour effectuer une redéfinition à portée globale...

11.8.5 Le trait central

Il reste donc à traiter la partie ayant pour but de tracer le vilain trait entre les deux boîtes. La première chose à faire est de mesurer la hauteur totale des deux boîtes, et de conserver la plus importante. Ceci est réalisé grâce au code suivant :

```
% mesure de la boîte d'entrée
\hauteurtotale{\tempodim}{\usebox{\b@iteentree}}%
% mesure de la boîte de sortie
\hauteurtotale{\hauteurdutrait}{\usebox{\b@itesortie}}%
% on garde la plus grande
\ifthenelse{\hauteurdutrait>\tempodim}{%
  \setlength{\tempodim}{\hauteurdutrait}}{}
```

On aura bien sûr préalablement déclaré les dimensions `\hauteurdutrait` et `\tempodim`. La commande `\settotoalheight` est quant à elle présentée au paragraphe 11.8.1.

Le trait noir à tracer entre l'entrée et la sortie mesure exactement `\hauteurtrait` moins `\l@rgeurnumex` (la largeur de la boîte contenant le numéro de l'exemple). On stocke cette dimension :

```
% hauteur du trait sans la boîte du numéro
\setlength{\hauteurdutrait}{\tempodim-\l@rgeurnumex}
```

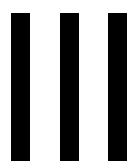
Nous avons décidé après un vote à bulletin secret en assemblée générale, de dessiner 70% du trait au dessus du numéro et 30% en dessous. Par conséquent le trait central est produit dans un `\parbox` comme suit :

```
% le trait central
\parbox{3pt}{%
  \begin{center}
    \rule{3pt}{.7\hauteurdutrait}\\ \nointerlineskip% 70% au dessus
    \rotatebox{90}{\affichenumex}\\ \nointerlineskip%
    \rule{3pt}{.3\hauteurdutrait}% 30% en dessous
  \end{center}}
```

La commande `\nointerlineskip` permet de supprimer tout blanc vertical supplémentaire qui pourrait être inséré par la commande `\\`. Plus de détails sont donnés à la section présentant l'implémentation de la **boîte avec titre** pour la minitable des matières.

Voilà, c'est tout pour ce merveilleux environnement `\ltxenv{ltexexemple}`. Notez que la dimension `\texttt{3pt}` pourrait faire l'objet de la définition d'une longueur...

Voilà, c'est tout pour ce merveilleux environnement `ltexexemple`. Notez que la dimension `3pt` pourrait faire l'objet de la définition d'une longueur...



Annexes

A

Générer des « pdf »

Sommaire

- A.1 Principe général
- A.2 Ce qui change
- A.3 Trucs et astuces
- A.4 Hyperliens
- A.5 Interaction avec psfrag et pstricks

CETTE ANNEXE présente un moyen de générer des documents au format Pdf (portable document format). Ce format créé par la société Adobe présente l'avantage d'être effectivement portable d'un ordinateur à un autre, et de manière plus générale, d'un système d'exploitation à un autre. Il est donc intéressant aujourd'hui de pouvoir générer de tels fichiers à partir d'un source L^AT_EX.

A.1 Principe général

Il y a au moins trois façons de générer des fichiers au format Pdf à partir d'un document L^AT_EX :

1. à l'aide de pdflatex qui s'utilise en lieu et place du programme latex pour traduire le source L^AT_EX en Pdf ;
2. à l'aide de dvipdf permettant de traduire le fichier Dvi en Pdf ;
3. à l'aide de ps2pdf pour traduire une sortie PostScript en Pdf.



Votre serviteur qui a une certaine expérience de la première solution s'attardera sur pdflatex. Un des pré-requis pour une utilisation correcte de ce logiciel est

- soit l'utilisation du package lmodern ;
- soit l'installation de l'extension « CM-Super font » de Vladimir VOLOVICH. La distribution Etch de la Debian contient un paquet prêt à l'emploi. On peut également trouver des documentations sur le ouèbe permettant d'installer cette extension sur une distribution Debian Sarge avec teT_EX (http://sravier.free.fr/linux/debian_latex_cm-super.html).

A.2 Ce qui change

Pour compiler le fichier source L^AT_EX et produire un fichier au format Pdf, on utilisera le programme pdflatex :

| pdflatex monfichier.tex

commande qui, si le document source ne contient pas d'erreur, créera le fichier nommé `monfichier.pdf`. Voici ensuite quelques remarques importantes :

Graphiques : ils devront être inclus au format Png ou Jpeg pour les images et Pdf pour les dessins.¹

Liens : à condition d'inclure le package `hyperref`, le document Pdf contiendra automatiquement des liens à chaque occurrence de la commande `\ref`, dans la table des matières, dans l'index, etc. De plus une table des matières déroulante sera générée pour le programme `acrobat reader`.

A.3 Trucs et astuces

Puisqu'on génère souvent du Dvi ou du Pdf à partir du même source et que l'on doit inclure des fichiers graphiques à des formats différents selon la situation, on utilisera le package `ifpdf` et l'astuce suivante :

```
\ifpdf
% un truc spécifique à la sortie en pdf
\else
% un machin spécifique à la sortie en dvi
\fi
```

A.3.1 Gestion des graphiques

On pourra écrire ensuite quelque chose du genre :

```
\ifpdf
\graphicspath{{pngs/}{pdfs/}}
\else
\graphicspath{{epss/}}
\fi
```

Si on a pris soin de ranger les fichiers graphiques dans les répertoires `pngs`, `pdfs` et `epss`... Ce nouveau «if» permet également des constructions du style :

```
\ifpdf
\includegraphics[pdftex]{graphicx}
\else
\includegraphics{graphicx}
\fi
```

Ce qui ne doit pas être nécessaire avec les dernières moutures de $\text{\LaTeX}$.

A.3.2 Vignettes

Les versions récentes de `pdflatex` permettent de créer des vignettes (*thumbnail*) pour les visualiseurs `evince` et `Acrobat reader` pour ne citer qu'eux. Auparavant, il était nécessaire d'utiliser le package `thumpdf` :

```
\usepackage{thumpdf}
```

¹Les fichiers du logiciel Xfig peuvent être convertis en Pdf

puis exécuter :

```
| thumbpdf monfichier.pdf
```

Cette commande crée un fichier nommé `monfichier.tpt` qui sera inclus à la compilation suivante avec `pdflatex`.

A.3.3 Pagination

Pour faire apparaître les numéros de pages du document dans le navigateur Acrobat Reader, il est nécessaire d'ajouter l'option `pdfpagelabels` à l'inclusion du package `hyperref` (cf. § suivant).

A.3.4 Signets

Les signets (*bookmarks* en anglais) des visualiseurs de fichier pdf sont des sortes d'«explorateurs de table des matières» qui donnent accès directement à une section d'un niveau déterminé. Il y a eu deux difficultés à contourner pour produire ce manuel :

1. faire en sorte que le contenu du «backmatter» (biblio, glossaire, index) soit au même niveau hiérarchique que les «parties» dans cet explorateur. Par défaut, ces informations se trouvent en effet «cachées» dans la partie des annexes, car à la même profondeur que les `\chapter` ;
2. faire en sorte que le lien vers l'index dans les signets pointe effectivement sur l'index...

Pour régler le premier problème, il suffit de faire croire à $\LaTeX$ qu'à partir du «backmatter» les *chapitres* sont d'un même niveau de profondeur dans la table des matières, que les *parties*. L'incantation vaudou correspondante est :

```
\renewcommand{\toclevel@chapter}{-1}
```

à placer à un endroit judicieux dans un fichier de style. L'endroit où l'on redéfinit le `\backmatter` est sans nul doute un bon choix.

Pour en finir avec les *bookmarks*, afin que celui de l'index pointe effectivement sur l'index (!), on devra cette fois avoir recours à une incantation chamanique :

```
\let\printindexORIG\printindex
\renewcommand{\printindex}{%
  \cleardoublepage
  \phantomsection% création d'une fausse section
  \addcontentsline{toc}{chapter}{Index}
  \printindexORIG}
```

permettant de surcharger la commande `\printindex` en y ajoutant une fausse section à l'aide de la commande `\phantomsection` fournie avec le package `hyperref`. Ne m'en demandez pas plus :-)

A.4 Hyperliens

Le package `hyperref` permet d'insérer dans les fichiers `.dvi` et `.pdf` des commandes spéciales qui pourront être exploitées par les navigateurs (`xdvi` et Acrobat

reader entre autres). On pourra alors cliquer sur le texte produit par les commandes telles que `\ref` pour se rendre automatiquement à la zone référencée. Dans le document que vous avez sous les yeux, la version électronique possède des liens sur lesquels on peut cliquer pour :

- toutes les références générées par `\ref`, `\pageref` et `\vref`;
- les notes de bas de page;
- les url produites par la commande `\url`;
- les renvois bibliographiques;
- les pages pour chaque entrée d'index.

Pour mettre en branle ce système d'hyperliens, on écrira :

```
\ifpdf
\usepackage[pdftex=true,
             hyperindex=true,
             colorlinks=true]{hyperref}
\else
\usepackage[hypertex=true,
             hyperindex=true,
             colorlinks=false]{hyperref}
\fi
```

Les liens seront mis en couleur uniquement pour la version Pdf. Dans le cas contraire la version à imprimer apparaîtra en gris plus ou moins clair sur imprimante laser noir et blanc...

L'ordre dans lequel on inclura les différents packages pour un document influera sur le bon fonctionnement de l'extension `hyperref`. Il arrive même que l'endroit choisi pour l'inclusion provoque une erreur de compilation. À vous de trouver la bonne séquence :-)

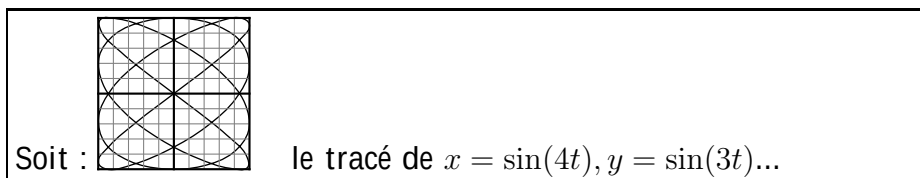
A.5 Interaction avec `psfrag` et `pstricks`

A.5.1 `pstricks`

To trick en anglais, ou « tricher » en français... En gros en écrivant ça :

```
Soit :
\begin{pspicture}[](-1,-1)(1,1)
\parametricplot[linewidth=.5pt,plotstyle=ccurve]%
{0}{360}{4 t mul sin 3 t mul sin}
\psgrid[gridlabels=0pt](-1,-1)(1,1)
\end{pspicture}
\quad le tracé de  $x=\sin(4t)$ ,  $y=\sin(3t)$ ...
```

On obtient ça :



Dingue, non ? Certes. Le principe de l'extension pstricks est d'insérer du code PostScript dans le fichier .dvi, code qui pourra être également traité par le programme dvips. Là où ça se corse c'est lorsque l'on veut utiliser ces bestioles avec pdflatex. En effet ce dernier créant directement un fichier .pdf à partir du .tex, insérer du PostScript dans le fichier au format Pdf n'aura aucun effet... Il est malgré tout possible de contourner le problème :

1. d'abord générer un document L^AT_EX minimal contenant les commandes pstricks ;
2. compiler ce document avec L^AT_EX pour générer un .dvi ;
3. demander à dvips de créer un fichier au format PostScript encapsulé avec l'option -E ;
4. convertir ce fichier au format Pdf ;
5. inclure ce fichier au moment d'utiliser pdflatex.

Tout cela est évidemment un peu tordu mais peut être automatisé à l'aide d'un Makefile, d'un petit script UNIX, et d'une commande... Tout d'abord :

```
\newcommand{\includepstricksgraphics}[1]{%
  \ifpdf\includegraphics{#1}\else\input{#1}\fi}
```

L'idée est donc d'extraire la portion de code contenant des commandes pstricks pour les stocker dans un fichier bidule.tex, puis lorsqu'on écrit :

```
\includepstricksgraphics{bidule}
```

on inclura bidule.pdf si on utilise pdflatex et bidule.tex si on utilise L^AT_EX. Ensuite, le « petit » script UNIX qu'on peut adapter à ses besoins :

```
#!/bin/sh
# on enlève l'extension du 1er argument
FILE=${1%.*}
# création d'un fichier temporaire psttemp.tex
cat > psttemp.tex <<EOF
\documentclass{manuel}          ← mettre la classe et les packages adéquats
\thispagestyle{empty}
\begin{document}
\input{FILE}
\end{document}
EOF
# Création du fichier dvi
latex psttemp
# Création du fichier eps
dvips -E $TMPFILE.dvi -o psttemp.eps
# Création du fichier pdf
epstopdf psttemp.eps --debug --outfile=$FILE.pdf
# effacement des fichiers temporaires
rm -f psttemp.*
```

Ce script sauvé sous le nom pstricks.sh pourra être invoqué comme suit :

```
./pstricks.sh bidule.tex
```

et devrait créer le fichier `bidule.pdf` que `pdflatex` aura la sagesse d'inclure grâce à la commande `\includepstricksgraphics` dont le code est donné plus haut. Pour ce qui est du `Makefile`, il n'est pas très difficile à partir du script précédent de définir une règle ayant pour but de transformer un fichier `.tex` en un fichier `.pdf`. Avec la version Gnu de `make`, on aura quelque chose du genre :

```
%.pdf : %.tex
    ./pstricks.sh $<
```

Le programme `dvips` n'est pas toujours en mesure de calculer correctement la boîte englobante pour le PostScript encapsulé. En particulier la section 41 de la documentation de `pstricks`² indique que `dvips` n'est pas capable de tenir compte du code postscript généré pour estimer cette boîte englobante. Dans ce cas, il est conseillé soit d'ajouter du texte autour du graphique et `dvips` arrive à s'en sortir, soit d'utiliser l'environnement `TeXtoEPS`. Le document temporaire du script précédent devient alors :

```
cat > psttemp.tex <<EOF
\documentclass{manuel}
\usepackage{pst-eps}
\thispagestyle{empty}
\begin{document}
\begin{TeXtoEPS}                                ← Pour aider PSTricks à calculer la boîte
\input{$FILE}
\end{TeXtoEPS}
\end{document}
EOF
```

A.5.2 psfrag

La limitation et le principe sont les mêmes que pour `pstricks`. Pour utiliser `psfrag` avec `pdflatex`, il est nécessaire de procéder comme suit :

1. d'abord générer un document $\text{\LaTeX}$ minimal contenant des commandes `psfrag` ;
2. compiler ce document avec $\text{\LaTeX}$ pour générer un `.dvi` ;
3. demander à `dvips` de créer un fichier au format PostScript encapsulé avec l'option `-E` ;
4. convertir ce fichier au format Pdf ;
5. inclure ce fichier au moment d'utiliser `pdflatex`.

Il y a cependant un petit « hic » car la figure dont on calcule la boîte englobante avec `dvips` contient du texte généré par `psfrag` indiquant les remplacements qui seront effectués. On doit en tenir compte. Dans le script shell, on crée une fonction :

```
function genere_eps
{
    cat > $TMPFILE.tex <<EOF
\documentclass{manuel}                        ← mettre la classe et les packages qui vont bien
\thispagestyle{empty}
\begin{document}
    \input{$1}
```

²<http://tug.org/PSTricks>

```

\end{document}
EOF
echo "Création du fichier dvi"
latex $TMPFILE > $LOGFILE
echo "Création du fichier $TMPFILE.eps"
dvips -E $TMPFILE.dvi -o $TMPFILE.eps >> $LOGFILE 2>&1
}

```

On utilise ensuite cette fonction par deux fois comme suit, dans le script :

```

FILE=${1%.*}
TMPFILE=truc
LOGFILE=truc.log
sanspsfrag=$TMPFILE-sanspsf.tex

# on enlève les lignes contenant la commande \psfrag
# et on récupère la boîte englobante du fichier eps sans les psfrag
grep -v \\\psfrag $FILE.tex > $sanspsfrag
genere_eps $sanspsfrag
bonnebb=$(grep "^%BoundingBox" $TMPFILE.eps | head -1)

# on récupère la boîte englobante du fichier eps avec psfrag
genere_eps $FILE
mauvaisebb=$(grep "^%BoundingBox" $TMPFILE.eps | head -1)

# on remplace la boîte englobante par la bonne
sed -i "s/$mauvaisebb/$bonnebb/" $TMPFILE.eps

echo "Création du fichier pdf"
epstopdf $TMPFILE.eps --debug --outfile=pdfs/${FILE##*/}.pdf >>
$LOGFILE 2>&1

# un petit coup de toilette
rm -f $TMPFILE.* $LOGFILE $sanspsfrag

```



Ce script a plusieurs limitations. Parmi elles : il échouera si une commande `\psfrag` s'étend sur plusieurs lignes. On demande en effet à `grep` d'enlever les lignes contenant `\psfrag` sans vérifier que la commande ne se termine pas une ou plusieurs lignes plus bas...

B

Mémento

Sommaire

- B.1 Extensions
- B.2 Les fichiers auxiliaires
- B.3 AucTeX
- B.4 Aspell

VOUS TROUVEREZ ici quelques informations «en vrac» sur les extensions de $\text{\LaTeX}$, une liste assez complète des fichiers auxiliaires qui gravitent autour de votre fichier source. Suivent quelques explications succinctes sur la merveilleuse extension AucTeX d'Emacs. Enfin pour ceux qui ont la chance de travailler dans un environnement UNIX, cette annexe s'achève sur la configuration d'Emacs pour travailler avec le correcteur orthographique aspell.

B.1 Extensions

Comme il en est question dans la préface de ce document, $\text{\TeX}$ et $\text{\LaTeX}$ sont des systèmes *ouverts*. Autour du noyau $\text{\LaTeX}$ gravitent un certain nombre de packages standard qui constituent la base du système. Mais tout utilisateur peut faire évoluer $\text{\LaTeX}$ en lui rajoutant des fonctionnalités diverses. On trouve donc une multitude d'outils sous formes d'extensions (ou *packages* en anglais) ou sous forme de classe de documents. Certaines sont devenues des standards, «toutes» sont disponibles sur les serveurs dédiés à la distribution de $\text{\LaTeX}$ (cf. chapitre 8) ou sur des pages personnelles, d'autres sont fournies avec les Call for papers et autres author's guides.

Nous vous donnons ici une liste de packages «classiques» et vous invitons à vous référer à la documentation qui est généralement jointe au package. Notez que le site du loria propose une liste des packages «généraux» à <http://www.loria.fr/services/tex/packages.html#latex> ; il existe d'autre part un moteur de recherche contenant plus de mille extensions référencées, à <ftp://ftp.loria.fr/pub/unix/tex/ctan/help/Catalogue/catalogue.html>.

- N french : utilisé pour «franciser» les documents. Ne coupe pas une phrase entre un mot et une double ponctuation. Propose aussi quelques commandes axées sur la typographie française (voir chapitre 7) ;
- N amsmath : le package pour faire des formules et équations perfectionnées ;
- N array : améliore l'utilisation de `tabular` ;
- N hline : étend les bordures de tableaux de base de $\text{\LaTeX}$;
- N fancyhdr : permet de personnaliser en-tête et pied de page. Jetez un coup d'œil sur ceux de ce manuel ;

sty fichier contenant des définitions de commandes modifiant la mise en page, ou fournissant des outils particuliers ;

cls un fichier définissant une classe. standard ;

Pour archiver un document L^AT_EX,

Vous pouvez effacer : tous les fichiers auxiliaires, les fichiers *log*, ainsi que les fichiers de tables des matières et listes de figures et tables.

Vous pouvez aussi effacer : le fichier *bbl* si vous êtes capable de le générer à partir d'un fichier *bib* et B_IB_TE_X. Les fichiers d'index peuvent généralement être effacés puisqu'ils sont en principe produits par *makeindex*. Le fichier *dvi* n'est pas indispensable puisque vous êtes censé avoir le source L^AT_EX

Vous devez garder : le source L^AT_EX et les éventuels fichiers de styles que vous avez définis (*sty* et *cls*) ; mais si vous en êtes au stade de la définition de classe, le conseil est probablement un peu saugrenu...

B.3 AucTeX

AucTeX est un module d'Emacs qui permet de faciliter la saisie de documents L^AT_EX. Ce module est automatiquement chargé lorsqu'on ouvre un document portant l'extension *.tex*, *.sty* ou *.cls*. On peut distinguer trois types d'aide dans AucTeX :

1. l'aide au formatage du source (couleur, indentation,...)
2. les raccourcis clavier pour insérer des commandes ou des environnements,
3. l'aide à la compilation.

B.3.1 Formatage du source

Les couleurs et la touche *tab* jouent le même rôle que dans un bu^ler C ou C++. On notera que **M-q** «formate» un paragraphe, c.-à-d., découpe automatiquement le paragraphe en lignes de longueurs à peu près égales.

B.3.2 Raccourcis

fontes

C-c C-f (Changer Fonte) suivi de :

- **C-e** insère `\emph{}`
- **C-b** insère `\textbf{}`
- **C-t** insère `\texttt{}`
- **C-s** insère `\textsl{}`
- **C-c** insère `\textsc{}`
- ...

Section

C-c C-s insère une Section en vous demandant son niveau, son titre et son label dans le minibu^ler.

Commandes et Environnement

- `M-Tab` tente de compléter le nom en cours (*automatic completion*).
- `C-c RET` insère une commande.
- `C-c C-e` insère un Environnement.²
- `C-u C-c C-e` change un environnement.
- `C-c ]` ferme un environnement en ajoutant la commande `\end` qui manque.

B.3.3 Compilation

`C-c C-c` tente de suivre de cycle de compilation d'un document, en lançant suivant la situation, `lATEX`, `BIBTEX`, `xdvi`,... Notez aussi que `AucTEX` permet de gérer le mécanisme du document maître (cf. 6.4). Pour cela il vous demandera de saisir le nom du document maître lorsque vous ouvrirez un nouveau document dans Emacs. Dans le cas contraire il faudra expliquer gentiment à `AucTEX` qui est le document maître avec :

`M-x TeX-master-file-ask` ou `C-c _`

vous devrez alors saisir le nom du fichier maître. En agissant ainsi, lorsque vous lancerez une compilation avec `C-c C-c` sur un des documents « esclaves », c'est sur le *master* qu'elle agira.

B.4 Aspell

Aspell est un correcteur orthographique multilingue qu'on peut interfacer avec l'éditeur de texte à tout faire Emacs. Pour l'utiliser dans Emacs, deux commandes à connaître :

`M-x ispell-change-dictionary`

sélectionne la langue du dictionnaire (français ou english), et :

`M-x ispell-buffer`

commence une session de correction sur le buffer. Il est également possible de vérifier l'orthographe d'un seul mot avec la commande :

`M-x ispell-word` ou `M-$`



À l'heure où j'écris ces lignes, la distribution Debian ne configure pas Emacs pour utiliser le programme Aspell par défaut. Il faut donc ajouter dans votre fichier `.emacs`, la ligne :

```
(setq-default ispell-program-name "aspell")
```

Il est particulièrement utile de noter que l'on peut configurer le programme Aspell pour lui demander explicitement d'ignorer ou non les arguments des commandes `lATEX`. On pourra par exemple ne pas vérifier l'argument d'une commande ne contenant pas de français. Ainsi, si l'on définit la commande :

² Pour certains environnements et certaines commandes dont la syntaxe est connue par `AucTEX` il vous sera demandé quelques précisions (valeurs des arguments, légendes, format du tableau,...).

```
\newcommand{\bidule}[2]{% commandes prenant 2 arguments
... }
```

Il su□ra d' crire dans son fichier `~/.aspell.conf` :

```
add-tex-command bidule pP
```

pour demander   Aspell de v rifier le second param tre (P), mais d'ignorer le contenu du premier (p). Pour ignorer les deux, on aurait  crit :

```
add-tex-command bidule pp
```

Enfin, Emacs dispose  galement d'un mode de correction de mot «  la vol e»³ qu'on peut activer ou d sactiver avec la commande :

```
| M-x flyspell-mode
```

³D'aucuns diront «  la ou rde»...

C

Symboles

Sommaire

- C.1 Symboles standard
- C.2 Symboles de l'*AMS*
- C.3 Symboles du package textcomp

VOUS TROUVEREZ dans cette annexe, une liste de « tous » les symboles mathématiques disponibles dans $\text{\LaTeX}$. Nous avons séparé ces symboles en quatre catégories :

- les symboles standard du tableau C.1 au tableau C.10 ;
- les symboles de $\text{\LaTeX}$ disponibles avec le package latexsym donnés par le tableau C.11 ;
- les symboles de l'*American Mathematical Society* disponibles avec le package amssymb, du tableau C.12 au tableau C.19 ;
- les symboles disponibles avec le package textcomp (tableaux C.20 et C.21) ;
- les symboles des polices PostScript bien connues ZapfDingbats et Symbol. Les symboles de ces fontes sont accessibles en incluant le package pifont et en utilisant la commande :

`\Pisymbol{pzd}{numéro}`

pour les symboles de la police Zapf, et :

`\Pisymbol{psy}{numéro}`

pour ceux de la police Symbol. Le nombre `numéro` est le numéro de la case correspondant au symbole choisi dans la table C.22 page 209 ou C.23.

C.1 Symboles standard

TAB. C.1 – Les lettres grecques.

<code>\alpha</code>	α	<code>\beta</code>	β	<code>\gamma</code>	γ	<code>\delta</code>	δ
<code>\epsilon</code>	ϵ	<code>\varepsilon</code>	ε	<code>\zeta</code>	ζ	<code>\eta</code>	η
<code>\theta</code>	θ	<code>\vartheta</code>	ϑ	<code>\iota</code>	ι	<code>\kappa</code>	κ
<code>\lambda</code>	λ	<code>\mu</code>	μ	<code>\nu</code>	ν	<code>\xi</code>	ξ
<code>o</code>	o	<code>\pi</code>	π	<code>\varpi</code>	ϖ	<code>\rho</code>	ρ
<code>\varrho</code>	ϱ	<code>\sigma</code>	σ	<code>\varsigma</code>	ς	<code>\tau</code>	τ
<code>\upsilon</code>	υ	<code>\phi</code>	ϕ	<code>\varphi</code>	φ	<code>\chi</code>	χ
<code>\psi</code>	ψ	<code>\omega</code>	ω				
<code>\Gamma</code>	Γ	<code>\Delta</code>	Δ	<code>\Theta</code>	Θ	<code>\Lambda</code>	Λ
<code>\Xi</code>	Ξ	<code>\Pi</code>	Π	<code>\Sigma</code>	Σ	<code>\Upsilon</code>	Υ
<code>\Phi</code>	Φ	<code>\Psi</code>	Ψ	<code>\Omega</code>	Ω		

TAB. C.2 – Les opérateurs binaires.

<code>\pm</code>	$\pm$	<code>\cdot</code>	$\cdot$	<code>\setminus</code>	$\setminus$	<code>\ominus</code>	$\ominus$
<code>\mp</code>	$\mp$	<code>\cap</code>	$\cap$	<code>\wr</code>	$\wr$	<code>\otimes</code>	$\otimes$
<code>\times</code>	$\times$	<code>\cup</code>	$\cup$	<code>\diamond</code>	$\diamond$	<code>\oslash</code>	$\oslash$
<code>\div</code>	$\div$	<code>\uplus</code>	$\uplus$	<code>\bigtriangleup</code>	$\bigtriangleup$	<code>\odot</code>	$\odot$
<code>\ast</code>	$\ast$	<code>\sqcap</code>	$\sqcap$	<code>\bigtriangledown</code>	$\bigtriangledown$	<code>\bigcirc</code>	$\bigcirc$
<code>\star</code>	$\star$	<code>\sqcup</code>	$\sqcup$	<code>\triangleleft</code>	$\triangleleft$	<code>\dagger</code>	$\dagger$
<code>\circ</code>	$\circ$	<code>\vee</code>	$\vee$	<code>\triangleright</code>	$\triangleright$	<code>\ddagger</code>	$\ddagger$
<code>\bullet</code>	$\bullet$	<code>\wedge</code>	$\wedge$	<code>\oplus</code>	$\oplus$	<code>\amalg</code>	$\amalg$

TAB. C.3 – Les symboles de tailles variables.

<code>\sum</code>	<code>\prod</code>	<code>\coprod</code>	<code>\int</code>	<code>\oint</code>
<code>\bigcap</code>	<code>\bigcup</code>	<code>\bigsqcup</code>	<code>\bigvee</code>	<code>\bigwedge</code>
<code>\bigodot</code>	<code>\bigotimes</code>	<code>\bigoplus</code>	<code>\biguplus</code>	

TAB. C.4 – Les points.

<code>\ldots</code>	$\ldots$	<code>\cdots</code>	$\cdots$	<code>\vdots</code>	$\vdots$	<code>\ddots</code>	$\ddots$
---------------------	----------	---------------------	----------	---------------------	----------	---------------------	----------

TAB. C.5 – Les relations.

<code>\leq</code>	$\leq$	<code>\geq</code>	$\geq$	<code>\equiv</code>	$\equiv$	<code>\models</code>	$\models$
<code>\prec</code>	$\prec$	<code>\succ</code>	$\succ$	<code>\sim</code>	$\sim$	<code>\perp</code>	$\perp$
<code>\preceq</code>	$\preceq$	<code>\succeq</code>	$\succeq$	<code>\simeq</code>	$\simeq$	<code>\mid</code>	$\mid$
<code>\ll</code>	$\ll$	<code>\gg</code>	$\gg$	<code>\asymp</code>	$\asymp$	<code>\parallel</code>	$\parallel$
<code>\subset</code>	$\subset$	<code>\supset</code>	$\supset$	<code>\approx</code>	$\approx$	<code>\bowtie</code>	$\bowtie$
<code>\subseteq</code>	$\subseteq$	<code>\supseteq</code>	$\supseteq$	<code>\cong</code>	$\cong$	<code>\smile</code>	$\smile$
<code>\sqsubseteq</code>	$\sqsubseteq$	<code>\sqsupseteq</code>	$\sqsupseteq$	<code>\neq</code>	$\neq$	<code>\frown</code>	$\frown$
<code>\in</code>	$\in$	<code>\ni</code>	$\ni$	<code>\doteq</code>	$\doteq$		
<code>\vdash</code>	$\vdash$	<code>\dashv</code>	$\dashv$	<code>\propto</code>	$\propto$		

TAB. C.6 – Les flèches.

<code>\leftarrow</code>	$\leftarrow$	<code>\longleftarrow</code>	$\longleftarrow$	<code>\uparrow</code>	$\uparrow$
<code>\Leftarrow</code>	$\Leftarrow$	<code>\Longleftarrow</code>	$\Longleftarrow$	<code>\Uparrow</code>	$\Uparrow$
<code>\rightarrow</code>	$\rightarrow$	<code>\longrightarrow</code>	$\longrightarrow$	<code>\downarrow</code>	$\downarrow$
<code>\Rightarrow</code>	$\Rightarrow$	<code>\Longrightarrow</code>	$\Longrightarrow$	<code>\Downarrow</code>	$\Downarrow$
<code>\leftrightharpoonup</code>	$\leftrightharpoonup$	<code>\longleftrightharpoonup</code>	$\longleftrightharpoonup$	<code>\updownarrow</code>	$\updownarrow$
<code>\Leftrightarrow</code>	$\Leftrightarrow$	<code>\Longleftrightarrow</code>	$\Longleftrightarrow$	<code>\Updownarrow</code>	$\Updownarrow$
<code>\mapsto</code>	$\mapsto$	<code>\longmapsto</code>	$\longmapsto$	<code>\nearrow</code>	$\nearrow$
<code>\hookrightarrow</code>	$\hookrightarrow$	<code>\hookleftarrow</code>	$\hookleftarrow$	<code>\searrow</code>	$\searrow$
<code>\leftharpoonup</code>	$\leftharpoonup$	<code>\rightharpoonup</code>	$\rightharpoonup$	<code>\swarrow</code>	$\swarrow$
<code>\leftharpoondown</code>	$\leftharpoondown$	<code>\rightharpoondown</code>	$\rightharpoondown$	<code>\nwarrow</code>	$\nwarrow$

TAB. C.7 – Divers.

<code>\aleph</code>	$\aleph$	<code>\prime</code>	$\prime$	<code>\forall</code>	$\forall$	<code>\infty</code>	∞
<code>\hbar</code>	$\hbar$	<code>\emptyset</code>	$\emptyset$	<code>\exists</code>	$\exists$	<code>\triangle</code>	$\triangle$
<code>\imath</code>	$\imath$	<code>\nabla</code>	∇	<code>\neg</code>	$\neg$	<code>\clubsuit</code>	$\clubsuit$
<code>\jmath</code>	$\jmath$	<code>\surd</code>	$\surd$	<code>\flat</code>	$\flat$	<code>\diamondsuit</code>	$\diamondsuit$
<code>\ell</code>	ℓ	<code>\top</code>	$\top$	<code>\natural</code>	$\natural$	<code>\heartsuit</code>	$\heartsuit$
<code>\wp</code>	$\wp$	<code>\bot</code>	$\bot$	<code>\sharp</code>	$\sharp$	<code>\spadesuit</code>	$\spadesuit$
<code>\Re</code>	$\Re$	<code>\ </code>	$\ $	<code>\backslash</code>	$\backslash$		
<code>\Im</code>	$\Im$	<code>\angle</code>	$\angle$	<code>\partial</code>	∂		

TAB. C.8 – Les fonctions.

<code>\arccos</code>	<code>\cos</code>	<code>\csc</code>	<code>\exp</code>	<code>\ker</code>	<code>\limsup</code>	<code>\min</code>	<code>\sinh</code>
<code>\arcsin</code>	<code>\cosh</code>	<code>\deg</code>	<code>\gcd</code>	<code>\lg</code>	<code>\ln</code>	<code>\Pr</code>	<code>\sup</code>
<code>\arctan</code>	<code>\cot</code>	<code>\det</code>	<code>\hom</code>	<code>\lim</code>	<code>\log</code>	<code>\sec</code>	<code>\tan</code>
<code>\arg</code>	<code>\coth</code>	<code>\dim</code>	<code>\inf</code>	<code>\liminf</code>	<code>\max</code>	<code>\sin</code>	<code>\tanh</code>

TAB. C.9 – Les délimiteurs.

<code>\uparrow</code>	↑	<code>\Uparrow</code>	↗	<code>\downarrow</code>	↓	<code>\Downarrow</code>	↘
<code>\{</code>	{	<code>\}</code>	}	<code>\updownarrow</code>	↕	<code>\Updownarrow</code>	↕
<code>\lfloor</code>	⌊	<code>\rfloor</code>	⌋	<code>\lceil</code>	⌈	<code>\rceil</code>	⌋
<code>\langle</code>	⟨	<code>\rangle</code>	⟩	<code>/</code>	/	<code>\backslash</code>	\
<code> </code>		<code>\ </code>					

TAB. C.10 – Les grands délimiteurs.

<code>\rmoustache</code>		<code>\lmoustache</code>		<code>\rgroup</code>		<code>\lgroup</code>
<code>\arrowvert</code>		<code>\Arrowvert</code>		<code>\bracevert</code>		

TAB. C.11 – Les symboles de latexsym

<code>\lhd</code>	C	<code>\rhd</code>	B	<code>\unlhd</code>	E	<code>\unrhd</code>	D
<code>\sqsubset</code>	@	<code>\sqsubset</code>	@	<code>\Join</code>	⋈	<code>\mho</code>	f
<code>\Box</code>		<code>\Diamond</code>		<code>\leadsto</code>			

C.2 Symboles de l' $\mathcal{AMS}$

TAB. C.12 – Les flèches de l' $\mathcal{AMS}$

<code>\dashrightarrow</code>	99K	<code>\dashleftarrow</code>	L99	<code>\leftleftarrows</code>
<code>\leftrightharpoons</code>		<code>\Lleftarrow</code>	W	<code>\twoheadleftarrow</code>
<code>\leftarrowtail</code>		<code>\looparrowleft</code>	"	<code>\leftrightharpoons</code>
<code>\curvearrowleft</code>	x	<code>\circlearrowleft</code>		<code>\Lsh</code>
<code>\upuparrows</code>		<code>\upharpoonleft</code>		<code>\downharpoonleft</code>
<code>\multimap</code>	(	<code>\leftrightsquigarrow</code>	!	<code>\rightrightarrows</code>
<code>\rightleftarrows</code>		<code>\rightrightarrows</code>		<code>\rightleftarrows</code>
<code>\twoheadrightarrow</code>		<code>\rightarrowtail</code>		<code>\looparrowright</code>
<code>\rightleftharpoons</code>		<code>\curvearrowright</code>	y	<code>\circlearrowright</code>
<code>\Rsh</code>		<code>\downdownarrows</code>		<code>\upharpoonright</code>
<code>\downharpoonright</code>		<code>\rightsquigarrow</code>		

TAB. C.13 – Les relations de l' $\mathcal{AMS}$

<code>\leq</code>	$\leq$	<code>\leqslant</code>	6	<code>\eqslantless</code>	0
<code>\lesssim</code>	.	<code>\lessapprox</code>	/	<code>\approxeq</code>	u
<code>\lessdot</code>	l	<code>\lll</code>	n	<code>\lessgtr</code>	7
<code>\lesseqgtr</code>	Q	<code>\lesseqqgtr</code>	S	<code>\doteqdot</code>	+
<code>\risingdotseq</code>	:	<code>\fallingdotseq</code>	;	<code>\backsim</code>	v
<code>\backsimeq</code>	w	<code>\subteq</code>	j	<code>\Subset</code>	b
<code>\sqsubset</code>	@	<code>\preccurlyeq</code>	4	<code>\curlyeqprec</code>	2
<code>\precsim</code>	-	<code>\precapprox</code>	w	<code>\vartriangleleft</code>	C
<code>\trianglelefteq</code>	E	<code>\vdash</code>		<code>\Vdash</code>	
<code>\smallsmile</code>	`	<code>\smallfrown</code>	a	<code>\bumpeq</code>	l
<code>\Bumpeq</code>	m	<code>\geqq</code>	=	<code>\geqslant</code>	>
<code>\eqslantgtr</code>	1	<code>\gtrsim</code>	&	<code>\gtrapprox</code>	'
<code>\gtrdot</code>	m	<code>\ggg</code>	o	<code>\gtrless</code>	?
<code>\gtreqless</code>	R	<code>\gtreqqless</code>	T	<code>\eqcirc</code>	P
<code>\circeq</code>	\$	<code>\triangleq</code>	,	<code>\thicksim</code>	s
<code>\thickapprox</code>	t	<code>\supseteq</code>	k	<code>\Supset</code>	c
<code>\sqsupset</code>	A	<code>\succcurlyeq</code>	<	<code>\curlyeqsucc</code>	3
<code>\succsim</code>	%	<code>\succapprox</code>	v	<code>\vartriangleright</code>	B
<code>\trianglerighteq</code>	D	<code>\Vdash</code>		<code>\shortmid</code>	p
<code>\shortparallel</code>	q	<code>\between</code>	G	<code>\pitchfork</code>	t
<code>\varpropto</code>	—	<code>\blacktriangleleft</code>	J	<code>\therefore</code>	)
<code>\backepsilon</code>	•	<code>\blacktriangleright</code>	I	<code>\because</code>	*

TAB. C.18 – Opérateurs binaires de l' $\mathcal{AMS}$

<code>\dotplus</code>	u	<code>\smallsetminusminus</code>	r	<code>\Cap</code>	e
<code>\Cup</code>	d	<code>\barwedge</code>	Z	<code>\veebar</code>	Y
<code>\doublebarwedge</code>	[	<code>\boxminus</code>		<code>\boxtimes</code>	
<code>\boxdot</code>		<code>\boxplus</code>		<code>\divideontimes</code>	>
<code>\ltimes</code>	n	<code>\rtimes</code>	o	<code>\leftthreetimes</code>	h
<code>\rightthreetimes</code>	i	<code>\curlywedge</code>	f	<code>\curlyvee</code>	g
<code>\circleddash</code>	•	<code>\circledast</code>	~	<code>\circledcirc</code>	}
<code>\symcenterdot</code>		<code>\intercal</code>			

TAB. C.19 – Symboles divers de l' $\mathcal{AMS}$

<code>\hbar</code>	~	<code>\hslash</code>	}	<code>\vartriangle</code>	M
<code>\triangledown</code>	O	<code>\square</code>		<code>\lozenge</code>	
<code>\circledS</code>	s	<code>\angle</code>	\	<code>\measuredangle</code>	]
<code>\nexists</code>	@	<code>\mho</code>	f	<code>\Finv</code>	`
<code>\Game</code>	a	<code>\Bbbk</code>		<code>\backprime</code>	8
<code>\varnothing</code>	?	<code>\blacktriangle</code>	N	<code>\blacktriangledown</code>	H
<code>\blacksquare</code>		<code>\blacklozenge</code>		<code>\bigstar</code>	F
<code>\sphericalangle</code>	^	<code>\complement</code>	{	<code>\eth</code>	g
<code>\diagup</code>		<code>\diagdown</code>			

C.3 Symboles du package `textcomp`

TAB. C.20 – Symboles du package `textcomp`.

<code>\textacutedbl</code>	”	<code>\textascendercompwordmark</code>	
<code>\textasciicute</code>	´	<code>\textasciibreve</code>	˘
<code>\textasciicaron</code>	ˇ	<code>\textasciidieresis</code>	¨
<code>\textasciigrave</code>	`	<code>\textasciimacron</code>	—
<code>\textasterisksymcentered</code>		<code>\textbaht</code>	฿
<code>\textbardbl</code>		<code>\textbigcircle</code>	◯
<code>\textblank</code>	□	<code>\textborn</code>	□
<code>\textbrokenbar</code>		<code>\textbullet</code>	•
<code>\textcapitalcompwordmark</code>		<code>\textcelsius</code>	°
<code>\textcent</code>	¢	<code>\textcentoldstyle</code>	¢
<code>\textcircledP</code>	Ⓟ	<code>\textcolonmonetary</code>	₯
<code>\textcopyright</code>	©	<code>\textcopyright</code>	©

TAB. C.21 – Symboles du package textcomp (suite).

<code>\textcurrency</code>	¤	<code>\textdagger</code>	†
<code>\textdaggerdbl</code>	‡	<code>\textdblhyphen</code>	–
<code>\textdblhyphenchar</code>	–	<code>\textdegree</code>	°
<code>\textdied</code>	☞	<code>\textdiscount</code>	☐
<code>\textdiv</code>	÷	<code>\textdivorced</code>	☐
<code>\textdollar</code>	\$	<code>\textdollaroldstyle</code>	☐
<code>\textdong</code>	₫	<code>\textdownarrow</code>	↓
<code>\texteightoldstyle</code>	☐	<code>\textestimated</code>	e
<code>\texteuro</code>	€	<code>\textfiveoldstyle</code>	☐
<code>\textflorin</code>	f	<code>\textfouroldstyle</code>	☐
<code>\textfractionsolidus</code>	/	<code>\textgravedbl</code>	☐
<code>\textguarani</code>	₲		
<code>\textinterrobang</code>	‡	<code>\textinterrobangdown</code>	☐
<code>\textlangle</code>	☐	<code>\textlbrackdbl</code>	☐
<code>\textleaf</code>	☐	<code>\textleftarrow</code>	←
<code>\textlira</code>	₺	<code>\textlnot</code>	¬
<code>\textlquill</code>	☐	<code>\textmarried</code>	☐
<code>\textmho</code>	☐	<code>\textminus</code>	–
<code>\textmu</code>	μ	<code>\textmusicalnote</code>	♪
<code>\textnaira</code>	₦	<code>\textnineoldstyle</code>	☐
<code>\textnumero</code>	№	<code>\textohm</code>	Ω
<code>\textonehalf</code>	½	<code>\textoneoldstyle</code>	☐
<code>\textonequarter</code>	¼	<code>\textonesuperior</code>	¹
<code>\textopenbullet</code>	°	<code>\textordfeminine</code>	ª
<code>\textordmasculine</code>	°	<code>\textparagraph</code>	¶
<code>\textperiodsymcentered</code>		<code>\textpertenthousand</code>	‰
<code>\textperthousand</code>	‰	<code>\textpeso</code>	₱
<code>\textpilcrow</code>	¶	<code>\textpm</code>	±
<code>\textquotesingle</code>	'	<code>\textquotestraightbase</code>	,
<code>\textquotestraightdblbase</code>	„	<code>\texttriangle</code>	☐
<code>\textrbrackdbl</code>	☐	<code>\textrecipe</code>	☐
<code>\textreferencemark</code>	☐	<code>\textregistered</code>	®
<code>\textrightarrow</code>	→	<code>\textrquill</code>	☐
<code>\textsection</code>	§	<code>\textservicemark</code>	☐
<code>\textsevenoldstyle</code>	☐	<code>\textsixoldstyle</code>	☐
<code>\textsterling</code>	£	<code>\textsurd</code>	√
<code>\textthreeoldstyle</code>	☐	<code>\textthreequarters</code>	¾
<code>\textthreequartersemdash</code>	☐	<code>\textthreesuperior</code>	³
<code>\texttildelow</code>	☐	<code>\texttimes</code>	×
<code>\texttrademark</code>	™	<code>\texttwelveudash</code>	☐
<code>\texttwooldstyle</code>	☐	<code>\texttwosuperior</code>	²
<code>\textuparrow</code>	↑	<code>\textwon</code>	₩
<code>\textyen</code>	¥	<code>\textzerooldstyle</code>	☐

TAB. C.22 – La police Zapf Dingbats

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	¯
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
p	q	r	s	t	u	v	w	x	y	z	{		}	~	
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
	ı	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	®	¯	
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

TAB. C.23 – Le police Symbol

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
p	q	r	s	t	u	v	w	x	y	z	{		}	~	
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	ı	ı	£	ı	¥	ı	§	..	©	ª	«	¬	-	®	—
161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	
°	±	2	3	´	μ	¶	·	ı	1	o	»	¼	½	¾	¿
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

D

Notes de production

Sommaire

- D.1 Distribution du moment
- D.2 Les sources du manuel
- D.3 Compilation

J'AI RASSEMBLÉ ici des éléments permettant d'exploiter les sources de ce document, comment les compiler, avec quelle distribution de L^AT_EX, comment les fichiers sont organisés, etc.

D.1 Distribution du moment

Le présent ouvrage a été compilé sur deux systèmes Linux contenant la distribution T_EXlive :

1. la distribution T_EXlive de la Ubuntu 07-10
2. une distribution T_EXlive pour Debian Etch provenant de <http://people.debian.org/~preining/TeX/>

Toute autre expérience fructueuse sur des systèmes différents est bien entendu la bienvenue... De plus pour la génération du document au format pdf, seule la famille Computer Modern dans sa version « CM-Super » a été testée.

D.2 Les sources du manuel

D.2.1 Structure

Les sources du manuel sont organisées selon le principe suivant :

- sources L^AT_EX dans le répertoire corps avec un fichier par chapitre ;
- les styles (sty et cls) dans le répertoire styles ;
- les images dans un répertoire pngs ;
- le répertoire teks contient des sources L^AT_EX à inclure dans le document (modèles de lettre, de fax, code contenant des appels à Psfrag ou Pstricks) ;
- les sources xfig dans le répertoire figs ;
- tout ce qui a trait à l'index, à la bibliographie et au glossaire est stocké dans le répertoire bibidx ;

Les sources xfig et certains « bouts » de fichiers L^AT_EX sont traduits au format pdf, postscript encapsulé ou non, par un makefile les stockant :

- dans le répertoire epss
- dans le répertoire pdfs

- dans le répertoire pss
selon le moteur utilisé (latex ou pdflatex).

D.2.2 Styles

Le fichier `manuel.cls` contient la définition de la classe du manuel. Ce fichier fait appel à une série de packages « du commerce » et une série de packages « maison ». On trouve, pour ces derniers, un fichier source pour :

- chaque « nouveau jouet » : onglets, nota, sommaire, glossaire, boite avec un titre (`titlebox.sty`), exemples, lettrine, renvois (`voir.sty`), citations et épi-graphes
- le sommaire
- la géométrie globale du document
- l'allure des en-têtes et pieds de page
- l'allure des sections/chapitres/etc.
- des commandes en vrac utilisées dans le document (`manumac.sty`)

Sauf indication contraire, ces fichiers portent un nom ressemblant étrangement à ce qu'ils contiennent.

D.3 Compilation

D.3.1 Makefile

L'arborescence racine des sources contient un fichier Makefile pour le manuel, qu'il faut copier :

```
| cp Makefile.manuel Makefile
```

D.3.2 Figures

Les figures peuvent être compilées grâce aux commandes :

```
#1. produire les figures
make figs
#2. générer les figures « récursives » de l'introduction
command latex --interaction nonstopmode guide-local
command latex --interaction nonstopmode guide-local
# 2 fois pour avoir la table des matières
make recursefigs
```



Il faut disposer du logiciel transfig connexe à xfig pour traduire les sources en pdf et eps. Sur le site <http://cours.enise.fr/info/latex> est disponible une archive contenant les figures déjà traduites...

D.3.3 Dvi et postscript

```
| latex guide-local
| make bibindex
| latex guide-local
| latex guide-local
| dvips guide-local -o guide-local
```

D.3.4 Pdf

Rien de particulier

```
pdflatex guide-local  
make bibindex  
pdflatex guide-local  
pdflatex guide-local
```

D.3.5 Versions imprimables sur papier A4

Les différentes versions imprimables dont il est question au paragraphe [page x](#) sont générées à partir d'un fichier postscript à jour :

```
dvips guide-local -o guide-local
```

et des utilitaires psutils. Les versions « reliure longue », « reliure courte 1 exemplaire » et « reliure courte 2 exemplaires » sont respectivement générées par les commandes suivantes :

```
make print-0  
make print-1  
make print-2
```

D.3.6 Nettoyage de printemps

En bref :

```
make cleanfigs  
make cleantex  
make cleandocs
```

← efface tous les eps/pdfs/...
← efface tous les fichiers auxiliaires
← efface tous les documents générés
(dvi/ps/pdf/...)

Bibliographie

- [1] KNUTH (D. E.), *The Art of Computer Programming*, vol. 1–3. Addison-Wesley, 1997–98.

Trois volumes sur « l'art de programmer ». Un quatrième tome est en préparation. Cet ensemble de livres a été accueilli par la communauté scientifique comme un des ouvrages les plus importants de ce siècle (cf. <http://www.amsci.org/amsci/bookshelf/centurylist.html> à ce sujet et <http://www-cs-staff.stanford.edu/~knuth/taocp.html> sur la page web de KNUTH pour plus d'info sur « TAOCP »).

- [2] LAMPORT (L.), *L^AT_EX : A Document Preparation System*. Addison-Wesley, édition 2^e, 1994.

Le livre de l'auteur de L^AT_EX dans sa seconde édition couvrant L^AT_EX 2". Bien évidemment une très bonne introduction, avec en fin d'ouvrage un guide de référence des commandes.

- [3] KNUTH (D. E.), *The T_EXBook*. Addison-Wesley, 1988.

LA bible de T_EX. Un livre plein de « virages dangereux » expliquant très précisément le fonctionnement interne de T_EX. C'est un ouvrage de référence assez difficile à lire et qui ne constitue pas une introduction à T_E

- [10] ANDRÉ (J.), « Petites leçons de typographie », 1990.
On doit pouvoir trouver ce document à l'url <http://jacques-andre.fr> Il s'agit d'un article intéressant sur la typographie avec beaucoup d'exemples sur l'emploi des majuscules, la ponctuation, l'usage du souligné et les caractères français.
- [11] GOOSSENS (M.), RAHTZ (S.) et MITTELBACH (F.), *The L^AT_EX Graphics Companion*. Addison-Wesley, 1997.
Par les auteurs du *L^AT_EX Companion*, un livre sur l'utilisation des graphiques au sens large du terme, avec notamment une exploration des packages permettant de dessiner avec L^AT_EX et une présentation de l'utilisation des fontes Postscript.
- [12] BITOUZÉ (D.) et CHARPENTIER (J.-C.), « L^AT_EX », dans *Collection Synthex*. Pearson Education France, septembre 2006.
- [13] APPEL (W.), CHEVALIER (E.), CORNET (E.) *et al.*, « L^AT_EX pour l' impatient », dans *Technique et pratique*. H & K, 2007.
- [14] *the UK List of T_EX Frequently Asked Questions on the Web*.
une mine d'informations en anglais, disponible à <http://www.tex.ac.uk/cgi-bin/texfaq2html>, listant les fameuses questions « fréquemment posées » sur T_EX & L^AT_EX (contrairement à ce que le titre indique).

Glossaire

Compilation

Même si ce terme n'est pas très rigoureux d'un point de vue scientifique, on appelle compilation la phase permettant de traduire le source $\text{\LaTeX}$ en un fichier au format DVI ou PDF.

Document maître

C'est le document source qui contient le `begin{document}` dans le contexte d'un document divisé en plusieurs fichiers.

Document source

Un document texte contenant le texte et les commandes $\text{\LaTeX}$. $\text{\TeX}$ C'est le document à ne pas perdre car il est à la source de la production papier, écran, etc. au même titre qu'un code source en langage C est la source d'un programme exécutable.

Dvi

Format de fichier *Device Independent* mis au point par KNUTH de manière à créer, à partir du document source, un document dont le format est indépendant de la plateforme et du matériel.

Fichiers auxiliaires

Les fichiers produits par $\text{\LaTeX}$ lors d'une compilation. Ils portent le nom du document source, et ont une extension de trois lettres rappelant leur rôle.

Format

C'est un ensemble de commandes ou macro précompilées et stockées dans un fichier portant généralement l'extension `.fmt`. Les plus connus de ces ensembles sont le format `plain` de $\text{\TeX}$ et le format $\text{\LaTeX}$.

Macro

C'est l'outil permettant de faire faire des choses compliquées à $\text{\LaTeX}$ en passant en ordre simple. Les macros, appelées aussi commandes, ressemblent un peu aux routines des langages de programmation.

PDF

Pour *portable document format*, format de fichier créé par la société Adobe, dont le but est de pouvoir échanger facilement des documents d'un système à un autre. Le format PDF peut être créé de plusieurs façons à partir d'un source $\text{\LaTeX}$ (cf. [A](#) page 187).

PostScript

Langage défini par la société Adobe pour décrire un document destiné à l'impression. Ce langage composé de primitives de bas niveau peut être interprété par des logiciels pour réaliser des aperçus avant impression ou directement par des circuits électroniques embarqués sur les imprimantes pour générer l'image à imprimer.

Références

Système permettant de manipuler les numéros des paragraphes, équations, chapitres, etc. de manière symbolique, pour s'affranchir de la difficulté de les mettre à jour lorsque l'on change la mise en page.

L^AT_EX

C'est l'ensemble de macros défini par Leslie LAMPORT au dessus de T_EX. La version utilisée aujourd'hui est L^AT_EX 2_ε.

T_EX

Le moteur de base, L^AT_EX étant un ensemble de macros formant une surcouche. La version de T_EX est stabilisée à la version 3.14159, à chaque nouvelle version K_NU_TH ajoute une décimale.

Symboles

.....	37
.....	37
.....	37
\(	110
\)	110
\,	37
\-	31
... extension	118
\/	17
\=	20
@	109
\[	34
\$	10, 34
§	12
T _E XnicCenter	4
T _E Xshop	4
\]	34

A

a4wide extension	XIV
accents	11
et fontes	87
mathématique	38
saisie	11
sur les majuscules	92
accolades	41
Acrobat reader	188, 190
acrobat reader	4, 188
\acute	38
\addcontentsline	170
\addpages	94
\address	78, 93
\addtocounter	48
\addtolength	51
\AE	90
\ae	90
algorithms extension	196
align environnement	42, 43
align* environnement	43
alignement	

à droite	18
à gauche	19
\Alph	49
\alph	49
\alpha	36
amsmath extension	33, 40–44, 195
amssymb extension	.v, 33, 35, 36, 156, 201
\AND	110
\appendix	178
\arabic	49
\arccos	204
\arcsin	204
\arctan	204
\arg	37, 204
argument	
de commandes	63
optionnel	9, 64
array extension	21, 195
array environnement	40, 41
article	
rédaction	77
style	78
Aspell	198, 199
aspell	.v, 195
AucT _E X	195
AucT _E X	78, 80, 197, 198
\author	77

B

babel extension	12, 87, 89–91, 133, 143, 145
\backmatter	85, 141, 189
\bar	38
\baselineskip longueur	50, 150
bash	65
bbm extension	36
bbold extension	36
beton extension	115
\bfseries	16
BibT _E X	82
BIBINPUTS	

- variable d'environnement 82
- bibliographie
 - article 79
 - citations 80
 - conférence 80
 - livre 80
 - saisie 78
 - style 78
 - alpha, 81
 - plain, 81
 - unsrt, 81
- \bibliography 81
- \bibliographystyle 81
- \bibname 141
- BIBTEX 78–82, 196–198
- bibunits extension 196
- bidouillage 109
- bidule extension 124
- \bidule 108–110, 166, 167
- \blacktriangle 156
- bmatrix environnement 41
- \boiteentreeglossaire 172
- book.cls 129–131, 133, 134, 141, 169, 178
- bookmarks 146, 189
- boîte
 - bordure 58
 - dimensions 52
 - et césure 31
 - exemples 56
 - paragraphe 60
 - positionnement 58
 - sauvegarde 62
 - simple 57
- \breve 38
- brouillon mode 30
- \bsc 91
- \bwarg 154
- \bwmarg 153
- 
- cadre de boîte 58
- cadrechap environnement 132
- calc extension 66
- \caption 25–27, 50
- caractère
 - d'échappement 10
 - spéciaux 10
- caractère @ 109
- \cdots 35
- center environnement 18, 21
- \centering 19
- centrage 18
- changebar extension 109
- \chapfont 132
- \chapnumfont 132
- \chapter .. 23, 85, 131, 170, 179, 189
- \chapter* 131
- chapterbib extension 196
- \chaptermark 138
- \check 38
- chnpage extension 113, 156
- citations 22, 80
- \cite 78, 80, 81
- \cleardoublepage 31, 137
- \clearpage 31
- \closing 93
- codage
 - iso-latin1 87
 - OT1 87
 - T1 87
- \colarg 154
- \colorbox 175, 177
- commande
 - appel 11
 - définitions 63
 - redéfinition 65
- commentaires 10
- compilation 6
 - et références 28
- compteur
 - açage 49
 - manipulation 48
- \conc 93
- concrete 115
- \contentsname 141
- convert 68, 76
- \cos 37, 204
- \cosh 204
- \cot 204
- \coth 204
- courrier 93
- \creerlettrine 168
- \csc 204
- césure 29

D

`\date` 77
 date du jour 12
`\ddot` 38
`\ddots` 35
`\DeclareFixedFont` 118
`\DefineVerbatimEnvironment` .. 143
`\deg` 204
`\degres` 90
 depth 149
`\depth longueur` 62
 description environnement .. 19, 20
 dessin 67
`\det` 204
`\dim` 204
 dimension d'un objet 52
 displaymath environnement .. 34, 42
`\displaystyle` 44
 diviser document 84
 docstrip extension 72
 document
 diviser 84
 document environnement 18
 documentation 97
`\documentclass` 8
`\dominitoc` 146
`\dot` 38
`\dotfill` 55
`\dots` 35
`\doublebox` 196
`\dp` 180
 draft option de classe 30
 dvipdf 187
 dvips 6, 7, 68, 72, 191, 192
 dviwin 68
 définition 62
 commandes 63
 environnement 64
 délimiteurs 41

E

e dans l'a 90
 édition 4
 eepic extension 76
`\em` 16
 Emacs ... 6, 78, 80, 98, 143, 144, 182,
 195, 197–199

emacs V, XIII, 4
 emacscom environnement 143
`\emph` 10, 16, 17, 115
 emphase fontes 17
 emTeX 68
`\encl` 93
`\enlargethispage` 31
`\enspace` 55
`\ensuremath` 63, 112
`\entreeglossaire` 173, 174
 entête 24
 enumerate environnement .. 19, 20, 27
 environnement
 définition 64
 redéfinition 65
 environnements
 TeXtoEPS 192
 VerbatimOut 179
 align* 43
 align 42, 43
 array 40, 41
 bmatrix 41
 cadrechap 132
 center 18, 21
 description 19, 20
 displaymath 34, 42
 document 18
 emacscom 143
 enumerate 19, 20, 27
 epigraphe 162
 eqnarray XIV, 42
 equation 42
 fiaux 122
 figure 25–27, 69, 71
 flushleft 19
 flushright 18
 hyperref 189
 itemize 19, 61
 letter 94
 list 66, 119–121, 161
 lrbox 125, 126
 ltexexemple 184
 ltexexempleenv 181
 ltexexemple .. 146, 179, 180, 182,
 183
 minipage 61, 71
 picture 67, 76
 pmatrix 41
 question 123

- varioref 154, 196
 - wrapfig 71, 72
 - xcolor 73, 165
 - textcomp 201
 - inclusion 9
 - options 9
- F**
- faire-tant-que 66
 - fancybox extension 63, 196
 - \fancyfoot 136
 - fancyhdr extension .. 24, 136, 141, 175, 176, 195
 - \fancyhead 136
 - fancyvrb extension . 127, 141, 143, 144, 179, 196
 - fax 93
 - \fax 93
 - \fbox 58, 61, 126, 196
 - \fboxrule longueur 58, 149
 - \fboxsep longueur .. 58, 74, 149, 150, 175, 177
 - \fg 12, 91, 145
 - ficaux environnement 122
 - fichier
 - .aux 27, 28
 - .bbl 82
 - .bib 78, 80, 81
 - .blg 82
 - .dvi 6, 27
 - .lof 26, 28
 - .log 27
 - .lot 26
 - .toc 28, 29
 - auxiliaire 27
 - graphique 68
 - postscript 6
 - source 7
 - fichiers
 - book.cls 129–131, 133, 134, 141, 169, 178
 - gglo.ist 171
 - glossaire.ist 174
 - ind.dvi 129
 - latex.ltx 137, 170, 180
 - newcent.sty 118
 - fig2dev 76
 - figure 25, 67
 - et mathématiques 72
 - incrustée 71
 - liste de 26
 - placement 25
 - figure environnement . 25–27, 69, 71
 - \fill 53, 55
 - flushleft environnement 19
 - flushright environnement 18
 - flèches 35
 - fonction mathématiques 36
 - \fontencoding 118
 - fontes
 - correction italique 17
 - emphase 17
 - gras 17
 - machine à écrire 16, 18
 - mathématiques 43
 - mise en évidence 15
 - penchée 16
 - petites majuscules 16, 18
 - sans serif 16
 - souligné 18
 - taille 17, 18
 - usage 17
 - \fontfamily 118
 - \fontseries 118
 - \fontshape 118
 - \fontsize 118
 - \footnote 24, 49, 61
 - \footnotemark 24
 - \footnotesize 17
 - \footnotetext 24
 - \footrulewidth 136
 - format
 - fichiers graphiques 68
 - \frac 34
 - fraction 34
 - \fraction 64
 - \framebox 58, 147
 - french extension .. IV, 89–91, 145, 195
 - \frontmatter 85, 140, 179
- G**
- \Gamma 36
 - \gcd 204
 - geometry extension 134, 135, 196
 - \geometry 134
 - gglo.ist 171

ghostscript 4
ghostview 60
gimp 67, 68
glossaire 84
glossaire.ist 174
\glurps 166, 168
gnuplot 67
graphics extension 68
graphicx extension . 60, 68, 69, 73, 175
graphique 67
 et mathématiques 72
gras fontes 17
\grave 38
grep 193
groupes 16
groupes de discussion 99
gsview 3
guillemets 12, 91
gv 60

H

\hat 38
\hauteurboitetitre longueur .. 149
\hauteurdutrait longueur 184
\hauteurtrait longueur 184
\hbox 30, 106, 107, 147
\headrulewidth 136
height 149
\height longueur 62
\hfill 22, 55, 132, 147
hhline extension 195
\hline 21
\hom 204
\hrule 147
\hrulefill 55
\hspace 59
\hspace* 54
\ht 180
\Huge 17
\huge 17
hyperref extension 188–190
\hyperref 157
hyperref environnement 189
hyphenation 4
\hyphenation 31

I

iTeXmax 4
\ieme 90
\ier 90
ifpdf extension 188
ifthen extension 66, 110, 196
\ifthenelse 112, 183
\ignorespaces 108, 164
image 67, 68
\imath 38
impression 7
imprimantes 7
\include 85
\includegraphics 69, 70
\includeonly 85
\includepstricksgraphics 192
inclusion
 d'extensions 9
 d'images 69
 de graphiques 69
ind.dvi 129
index 82
\index 82
\indexname 141
\indexspace 129
indice 10, 34
\indletB longueur 168, 169
\indletH longueur 168
\indnota longueur 160
\inf 37, 204
info 98
Inkscape 76
\input 66, 85
\InputIfFileExists 174
\institut 94
\int 44
intégrale 37
\isodd 110, 158
italique fontes 17
\item 172
\itemindent longueur 119–121
itemize environnement 19, 61
\itemsep longueur 120, 121, 125
\itshape 16

J

\jmath 38

\jobname174

K

kdvi4

\ker204

\kern145, 149

kile4

\kill20

L

\Lab63

\label27, 43, 79, 183

\labelsep longueur119–121, 124

\labelwidth longueur119–121

\langle41, 153

\LARGE17

\Large17

\large17

\largeurboitetitre longueur ..149

\larligB longueur168

\larligH longueur168

latex187

latex.ltx137, 170, 180

latexsym extension35, 201, 204

\lceil41

\ldots12

\leaders147, 148

\leavevmode106, 107

\left41

\leftmargin longueur .119–121, 161

\leftmark138, 139

\lengthtest110

\let109, 110, 154

letter environnement94

lettres grecques36

lettrine90

lettrine extension90

\lettrine168, 169

\lfloor41

\lg204

\lieu93

like thisXIII

\lim44, 204

\liminf204

limite37

\limsup204

\linebreak31

\linewidth longueur53

list environnement 66, 119–121, 161

liste19, 66

 d'items19

 des figures26

 des tables26

 description20

 énumération20

listings extension21, 141, 142, 144

\listoffigures26, 169, 170

\listoftables26

make 67, 74, 75, 192
makebst 78
makeindex 82, 83, 128, 129,
171–173, 196, 197
metafont 67
pdflatex 187–189, 191, 192
ps2pdf 187
psfrag 187, 190, 192
pstricks 187, 190, 192

newcent.sty 118
 \newcommand 44, 63, 64, 166
 \newcounter 48, 113
 \newenvironment 64, 144
 \newlength 51
 \newsavebox 62
 newsgroup 99
 \No 90
 \no 90
 \NoAutoSpaceBeforeFDP 89
 \nocite 80
 \nointerlineskip 151, 184
 \nolimits 44
 \nolinebreak 31
 \nonumber 43
 \nopagebreak 31
 \normalfont 154
 \normalsize 17
 \NOT 110
 \not 38
 note de bas de page 24
 \nouppercase 141
 numérotation 47



odd 136
 \OE 90
 œ 90
 \oe 90
 \log 12, 91, 145
 \oint 37
 \onglet 176, 178
 \ongletfont 175
 \opening 93
 options
 de graphicx 69
 de classe 8
 opérateur \not 38
 \OR 110
 \Ovalbox 63, 196
 \ovalbox 196
 overcite extension 196
 overfull 29
 \overrightarrow 38
 OzTeX 68



packages extension XIII
 \padnota longueur 160
 \pagebreak 31
 \pagenumbering 140
 \pageref 27, 190
 \pagestyle 24
 \par 9, 19, 56
 \paragraph 23
 paragraphe
 séparation 53
 \parbox 60, 61, 148, 149, 184
 parenthèses 41
 \parindent longueur 61, 164
 \parindent 50, 51
 \parsep longueur 120, 121
 \parshape 158, 159, 166, 167
 \parskip 51, 53
 \part 23, 133, 134
 \partopsep longueur 120, 121
 pdflatex 187–189, 191, 192
 petites majuscules fontes 18
 \phantomsection 189
 \pi 36
 picture environnement 67, 76
 pied de page 24
 pifont extension 201
 pmatrix environnement 41
 points 55
 points de suspension 12, 35
 positionnement de boîte 58
 PostScript .. 3, 4, 6, 7, 60, 67, 68, 72,
 97, 114, 118, 191, 192, 201
 \Pr 204
 préambule 8
 \primo 90
 \printindex 82, 83, 189
 \prod 37
 produit 37
 \protect 155
 ps2pdf 187
 psfrag extension 72, 73, 192
 \psfrag 72
 psfrag 187, 190, 192
 pstricks extension 76, 191, 192
 pstricks 187, 190, 192
 psutils 213

Q

<code>\qqquad</code>	39, 55
<code>\quad</code>	39, 55
<code>\quarto</code>	90
question environnement	123
quotation environnement	22
quote environnement	22

R

racine	34
<code>\raggedleft</code>	19
<code>\raggedright</code>	19
<code>\raisebox</code>	58, 177
<code>\rangle</code>	41, 153
<code>\rceil</code>	41
recto verso	8
redéfinitions	65
<code>\ref</code>	27, 188, 190, 196
<code>\reflectbox</code>	175
<code>\renewcommand</code>	44, 65, 183, 184
<code>\renewenvironment</code>	65
ressort	52
<code>\rfloor</code>	41
<code>\right</code>	41
<code>\right.</code>	41
<code>\rightmargin</code> longueur	119, 121
<code>\rightmark</code>	138, 139
<code>\rmfamily</code>	16
<code>\Roman</code>	49
<code>\roman</code>	49
<code>\rotatebox</code>	60
rotation	
de boîtes	60
de graphiques	70
<code>\rule</code>	67
référence	26
aux subfigures	71
et fichier auxiliaires	28
non définie	28
à un objet	27
à une page	27

S

<code>\S</code>	12
saut	
de ligne	9

de paragraphe	9
sauvegarde de boîte	62
<code>\savebox</code>	62, 180
<code>\sbox</code>	62
<code>\scriptscriptstyle</code>	44
<code>\scriptsize</code>	17
<code>\scriptstyle</code>	44
<code>\scshape</code>	16
<code>\sec</code>	204
<code>\section</code> .. 23, 27, 129, 130, 139, 170	
section numbering depth	129
<code>\section*</code>	23
<code>\sectionmark</code>	138
<code>\secundo</code>	90
<code>\selectfont</code>	118
<code>\setboolean</code>	110
<code>\setbox</code>	180
<code>\setcounter</code>	48
<code>\setlength</code>	51
<code>\settodepth</code>	52
<code>\settoheight</code>	52, 180
<code>\settotoalheight</code>	184
<code>\settowidth</code>	52, 180
<code>\sffamily</code>	16
<code>\shadowbox</code>	196
shape	115
<code>\showthe</code>	53
si-alors-sinon	66
<code>\signature</code>	93
simulation de terminal	21
<code>\sin</code>	37, 204
<code>\sinh</code>	204
sites internet	98
<code>\slshape</code>	16
<code>\small</code>	17, 181
sommaire	90
<code>\sommaire</code>	91, 170
somme	37
souligné	
fontes	18
sous-figures	71
split environnement	43
<code>\sqrt</code>	34
<code>\stackrel</code>	39
<code>\stretch</code>	53
subfig extension	71
<code>\subfigure</code>	71
subfigure environnement	71
<code>\subparagraph</code>	23

`\subsection` 23, 49, 129
`\subsubsection` 23
`\sup` 37, 204
`\symbol` 145, 155
`\symboles` 112
 symboles mathématiques 35

T

tab le of contents depth 129
 tabbing environnement 20
 table 25
 liste des 26
 placement 25
 table environnement 25, 27
 table des matières 29
 numérotation 129
 profondeur 129
 tableau 20
 mathématique 40
`\tableofcontents` ... 23, 29, 82, 141,
 169, 170
 tabular environnement ... 20, 21, 40,
 60, 195
 tabulations 20
 taille
 des fontes 17
 des graphiques 70
`\tan` 37, 204
`\tanh` 204
 telefax environnement 94
`\telephone` 93
`\tempodim longueur` 184
`\tertio` 90
`\TeX` 11
 TEXINPUTS
 variable d'environnement 65
 texmaker 4
`\text` 40
`\textbf` 16, 115
 textcomp extension .. 91, 201, 207, 208
 textcomp extension 201
`\texteuro` 91
`\textheight` 50
`\textit` 16
`\textmd` 16
 TeXtoEPS environnement 192
`\textrm` 16
`\textsc` 16

`\textsf` 16
`\textsl` 16
`\textstyle` 44
`\texttt` 16
`\textup` 16
 texture 68
`\textwidth longueur` 61, 159
`\textwidth` 50
`\thanks` 77
`\the` 49
 thebibliography environnement 78,
 82, 141
`\thechapter` 132
`\thefigure` 49
`\thefootnote` 49
 theglossary environnement 172
 theindex environnement 141
`\thepage` 49
`\thesubsection` 49
`\thispagestyle` 25
 thumbpdf extension 188
`\tilde` 38
 times extension 114
`\times` 112
`\tiny` 17
 tirets 12
`\title` 77
`\titlebox` 148, 151
 titre 23
 d'un document 78
 numérotation 129
`\today` 12
`\topsep longueur` 120, 121
`\totalheight longueur` 62
 traits 55, 67
 transfig 212
 translation de boîte 58, 59
`\truc` 108
`\ttfamily` 16
`\typeout` 174
 typographie
 lettrine 90
 majuscules 92
 ponctuation 89
 règles 17, 91

U

underfull 29

<code>\underline</code>	18
unité des longueurs	50
UNIX ..V, VIII, 6, 7, 68, 105, 191, 195	
unixcom environnement	143
<code>\unskip</code>	108, 145, 164
<code>\up</code>	91
<code>\upshape</code>	16
url extension	196
<code>\url</code>	190
<code>\usebox</code>	62, 165
<code>\usecounter</code>	123
<code>\usepackage</code>	9, 65, 70, 109, 134

Vurl

Table des matières

I	« Tout » sur L ^A T _E X	1
1	Principes de base	3
1.1	Installation	3
1.2	Cycle de production	4
1.2.1	Édition	4
1.2.2	Compilation	6
1.2.3	Visualisation	6
1.2.4	Impression	7
1.3	Le document source : un document type	7
1.3.1	Classe du document	8
1.3.2	Le préambule	8
1.3.3	Ajout d'extension	9
1.4	C'est parti !	9
1.4.1	Quelques caractères sont spéciaux	10
1.4.2	Appel des commandes	11
1.4.3	Accents	11
1.5	Premiers outils	12
1.6	Premières erreurs	12
1.6.1	Symptômes	12
1.6.2	Diagnostic	13
1.6.3	Soins	13
1.6.4	Une collection de message	14
2	Ce qu'il faut savoir	15
2.1	Mise en évidence	15
2.1.1	Family-shape-series	16
2.1.2	Correction italique	17
2.1.3	Tailles	17
2.1.4	Quelques recommandations	17
2.2	Environnements	18
2.2.1	Centrage et alignement	18
2.2.2	Listes	19
2.2.3	Tabulations	20
2.2.4	Tableaux	20
2.2.5	Simulation de terminal	21
2.2.6	Citations	22
2.3	Notes de marge	22
2.4	Titres	23
2.5	Notes de bas de page	24
2.6	Entête et pied de page	24
2.7	Flottants	25

2.7.1	Figure et table	25
2.7.2	Placement	25
2.7.3	Liste des figures	26
2.8	Références	26
2.8.1	Principe	26
2.8.2	Que référencer ?	27
2.9	Fichiers auxiliaires	27
2.9.1	Interaction avec les références	28
2.9.2	Interaction avec la table des matières	29
2.9.3	Petits conseils	29
2.10	Où il est question de césure	29
2.10.1	Contrôler la césure	30
3	Mathématiques	33
3.1	Les deux façons d'écrire des maths	33
3.2	Commandes usuelles	34
3.2.1	Indice et exposant	34
3.2.2	Fraction et racine	34
3.2.3	Symboles	35
3.3	Fonctions	36
3.3.1	Fonctions standards	36
3.3.2	Intégrales, sommes et autres limites	37
3.4	Des symboles les uns sur les autres	38
3.4.1	L'opérateur not	38
3.4.2	Accents	38
3.4.3	Vecteurs	38
3.4.4	Commande stackrel	39
3.5	Deux principes importants	39
3.5.1	Espaces en mode mathématique	39
3.5.2	Texte en mode mathématique	40
3.6	Array : simple et échappée	40
3.6.1	Comment ça marche	40
3.6.2	Array et les délimiteurs	41
3.6.3	Pour vous simplifier la vie...	41
3.7	Équations et environnements	42
3.7.1	L'environnement displaymath	42
3.7.2	L'environnement equation	42
3.7.3	Formules multi-lignes	42
3.8	Changer le style en mode mathématique	43
3.8.1	Fontes	43
3.8.2	Taille des symboles	43
3.8.3	Créer de nouveaux opérateurs	44
4	Un pas vers la sorcellerie	47
4.1	Compteurs	47
4.1.1	Compteurs disponibles	47
4.1.2	Manipulation	48
4.1.3	Alignage	49
4.2	Longueurs	50

4.2.1	Unités	50
4.2.2	Quelques longueurs de $\text{\LaTeX}$	50
4.2.3	Manipulation des longueurs	51
4.2.4	Longueurs élastiques .	

7	Des documents en français	87
7.1	Le problème des lettres accentuées	87
7.2	Rédiger un document en français avec L ^A T _E X	89
7.3	Le package babel et la typographie	89
7.3.1	Ponctuation	89
7.3.2	L-a, e dans l'a, t-i, t-i, a!	90
7.3.3	Outils du package babel	90
7.3.4	Recommandations d'usage	91
7.3.5	Le cas de l'euro	91
7.3.6	Au sujet des majuscules	92
7.4	Courrier et fax	93
7.4.1	Commandes disponibles	93
7.4.2	Structure d'un document basé sur la classe lettre	93
7.4.3	Fichiers «instituts»	93
7.4.4	Fax	94
8	À vous de jouer !	97
8.1	Livres et autres manuels	97
8.2	Local	98
8.3	E ² Tépé, Ouèbe et niouses	98
8.3.1	Sites FTP	98
8.3.2	Sites Web	98
8.3.3	Les newsgroups	99
II	« Tout »sur (« Tout »sur L ^A T _E X)	101
9	Outillage nécessaire	105
9.1	Hercule Poirot	105
9.1.1	Fouiller dans les fichiers	105
9.1.2	Examiner les macros	106
9.2	Outils de bas niveaux	107
9.2.1	Pour qui sont ces pourcents ?	107
9.2.2	Le caractère @	109
9.2.3	Le \let de T _E X	109
9.3	Structures de contrôle et tests	110
9.3.1	Booléens et opérateurs associés	110
9.3.2	Exemples	111
9.3.3	Tester la parité des pages	113
9.4	Fontes	114
9.4.1	Le jeu des « trois »familles	114
9.4.2	Désignation des fontes et de leurs attributs	114
9.4.3	Changer de fontes	117
9.5	Listes et nouveaux environnements	119
9.5.1	Principe	119
9.5.2	Réglage de l'étiquette	120
9.5.3	Réglages verticaux	120
9.5.4	Valeurs par défaut	121
9.5.5	Exemples	122

9.5.6	Un exemple un peu plus tordu...	124
9.6	Des environnements qui mettent en boîte	125
9.6.1	Principe	125
9.6.2	Exemple	125
10	Cosmétique	127
10.1	Allure de l'index	127
10.2	Allures des titres	129
10.2.1	Numérotation des titres et table des matières	129
10.2.2	Sections et niveaux inférieurs	130
10.2.3	Chapitres	131
10.2.4	Parties	133
10.3	Géométrie	134
10.4	En-tête et pied de page	136
10.4.1	Cas de la première page des chapitres	136
10.4.2	Pages vierges avant le début d'un chapitre	137
10.4.3	Mécanisme de marqueurs	137
10.4.4	Organisation du document	139
10.4.5	Numéroter l'introduction en roman « petites capitales »	140
10.4.6	Interaction avec index, bibliographie et table des matières	141
10.5	Environnements avec caractères spéciaux	141
10.5.1	Digression vers les caractères...	142
10.5.2	Environnements maison basés sur fancyvrb	143
10.5.3	Environnements pour les langages de programmation	144
10.6	About those so called "french guillemets"	145
10.7	Une boîte pour la minitable des matières	146
10.7.1	L'interface de la commande	146
10.7.2	Quand même un peu de T _E X	147
10.7.3	Conception de la boîte	148
10.7.4	Le code	148
10.7.5	Utilisation avec package minitoc	151
11	De nouveaux jouets	153
11.1	Quelques bricoles	153
11.1.1	Arguments et convention typographique	153
11.1.2	Autour de la génération de l'index	154
11.1.3	Des renvois	155
11.1.4	Changement de marges	157
11.2	Des nota	158
11.3	Des citations	162
11.3.1	Épigraphes	162
11.3.2	Citations	162
11.4	Des lettrines	166
11.4.1	La commande \glurps ou un pas vers T _E X	166
11.4.2	Insertion de la lettrine dans un paragraphe	167
11.5	Un sommaire	169
11.6	Un glossaire	171
11.6.1	Tordre le cou à makeindex	171
11.6.2	Un environnement pour le glossaire	172

11.6.3	Produire le fichier .glx	172
11.6.4	Recollons les morceaux	173
11.7	Des onglets	174
11.7.1	Idée retenue	175
11.7.2	Les boîtes dans la marge	175
11.7.3	Position des onglets	176
11.8	Exemples L ^A T _E X	179
11.8.1	Outils nécessaires	179
11.8.2	Le principe de l'environnement ltxexemple	180
11.8.3	Mises en boîte	181
11.8.4	Numérotation des exemples	182
11.8.5	Le trait central	184
III	Annexes	185
A	Générer des « pdf »	187
A.1	Principe général	187
A.2	Ce qui change	187
A.3	Trucs et astuces	188
A.3.1	Gestion des graphiques	188
A.3.2	Vignettes	188
A.3.3	Pagination	189
A.3.4	Signets	189
A.4	Hyperliens	189
A.5	Interaction avec psfrag et pstricks	190
A.5.1	pstricks	190
A.5.2	psfrag	192
B	Mémento	195
B.1	Extensions	195
B.2	Les fichiers auxiliaires	196
B.3	AucT _E X	197
B.3.1	Formatage du source	197
B.3.2	Raccourcis	197
B.3.3	Compilation	198
B.4	Aspell	198
C	Symboles	201
C.1	Symboles standard	202
C.2	Symboles de l' $\mathcal{AMS}$	205
C.3	Symboles du package textcomp	207
D	Notes de production	211
D.1	Distribution du moment	211
D.2	Les sources du manuel	211
D.2.1	Structure	211
D.2.2	Styles	212
D.3	Compilation	212
D.3.1	Makefile	212

Table des matières	237
D.3.2 Figures	212
D.3.3 Dvi et postscript	212
D.3.4 Pdf	213
D.3.5 Versions imprimables sur papier A4	213
D.3.6 Nettoyage de printemps	213
Bibliographie	215
Glossaire	217
Index	219