

# Introduction à L<sup>A</sup>T<sub>E</sub>X

Jean-Marc LICHTLE

11 juillet 2004

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Version décrite . . . . .	4
1.2	Compatibilité d'autres système de création de documents . . . . .	4
<b>2</b>	<b>Présentation</b>	<b>6</b>
2.1	L <sup>A</sup> T <sub>E</sub> X et vos habitudes de travail . . . . .	6
2.2	Le travail avec L <sup>A</sup> T <sub>E</sub> X. . . . .	7
<b>3</b>	<b>Installation de L<sup>A</sup>T<sub>E</sub>X</b>	<b>8</b>
<b>4</b>	<b>Structure d'un document L<sup>A</sup>T<sub>E</sub>X.</b>	<b>8</b>
4.1	Les balises, syntaxe générale . . . . .	8
4.2	Le préambule . . . . .	8
4.2.1	Le type de document . . . . .	8
4.2.2	La langue du document . . . . .	9
4.2.3	Les caractères accentués . . . . .	9
4.3	Les environnements . . . . .	10
4.4	Le plan d'un document . . . . .	11
4.4.1	Notions générales . . . . .	11
4.4.2	"Profondeur" et numérotation de la table des matières . . . . .	12
<b>5</b>	<b>Mise en page</b>	<b>12</b>
5.1	Format de la page . . . . .	12
5.2	Entête et pied de page . . . . .	14
<b>6</b>	<b>Étoffer votre texte</b>	<b>15</b>
6.1	Résumé de document . . . . .	15
6.2	Sauts de ligne . . . . .	16
6.3	Les aménagements de police de caractère . . . . .	16
6.3.1	La forme des caractères . . . . .	16
6.3.2	La tailles des caractères . . . . .	17

6.3.3	Définir une police de caractères pour l'ensemble du document . . . . .	17
6.4	les environnements de liste . . . . .	18
6.5	Inclure des éléments graphiques . . . . .	18
6.6	Utiliser les références internes . . . . .	20
6.7	Les tableaux . . . . .	21
6.8	Les barres de révision . . . . .	22
6.9	les notes en marges et notes en pied de page . . . . .	23
6.10	Les tabulations et l'environnement tabbing . . . . .	24
<b>7</b>	<b>L<sup>A</sup>T<sub>E</sub>X et les mathématiques</b>	<b>25</b>
<b>8</b>	<b>Définir une commande L<sup>A</sup>T<sub>E</sub>X personnelle</b>	<b>25</b>
8.1	Application à la création d'un symbole €. . . . .	26
8.2	Application à la mise en place de logos dans le texte . . . . .	27
<b>9</b>	<b>Bibliographies et index</b>	<b>27</b>
9.1	Gestion des bibliographies . . . . .	27
9.1.1	Le fichier .bib . . . . .	27
9.1.2	Compilations spécifiques . . . . .	28
9.2	Mise en place d'un index . . . . .	29
9.2.1	Modification code source . . . . .	29
9.2.2	Compilation spécifique . . . . .	29
9.2.3	Utilisation avancée . . . . .	29
9.3	Compilation automatique . . . . .	30
<b>10</b>	<b>Création de documents au format .pdf</b>	<b>30</b>
10.1	Format de page . . . . .	31
10.2	Police de caractères . . . . .	31
<b>11</b>	<b>Compilation avec pdf<sub>l</sub>atex</b>	<b>31</b>
11.1	Modification élémentaire du source en vue de la compilation par pdf <sub>l</sub> atex . . . . .	32
11.2	Source compilable par L <sup>A</sup> T <sub>E</sub> X et pdf <sub>l</sub> atex . . . . .	33
11.3	Aléas lors des compilations alternées avec La <sub>T</sub> E <sub>X</sub> et pdf <sub>l</sub> atex . . . . .	34
<b>12</b>	<b>Synthèse</b>	<b>34</b>
<b>13</b>	<b>Faire du courrier</b>	<b>36</b>
13.1	Compilation, L <sup>A</sup> T <sub>E</sub> X ou pdf <sub>l</sub> AT <sub>E</sub> X ? . . . . .	36
13.2	La classe lettre de L'Observatoire de Genève . . . . .	36
13.3	Récupérer les fichiers strictement nécessaires . . . . .	36
13.4	Une documentation complète . . . . .	36
13.5	Installation . . . . .	37
13.6	Paramétrage du fichier default.ins . . . . .	37
13.7	Premier essai . . . . .	37
13.8	Les champs disponibles pour la zone entête de lettre . . . . .	40

13.9 Les champs disponibles pour la zone pied de lettre . . . . .	41
<b>14 Utilisation de la classe lettre pour la rédaction de téléfax</b>	<b>43</b>
14.1 Premier essai . . . . .	43
14.2 Les champs disponibles pour la zone entête de telefax . . . . .	44
14.3 Les champs disponibles pour la zone pied de telefax . . . . .	46
<b>15 Utilisation avancée</b>	<b>46</b>
15.1 Inclusion d'un fichier d'entête . . . . .	46
15.2 Inclusion de parties de documents stockés dans des fichiers externes . . . . .	48
15.3 Définir une nouvelle macro commande paramétrable . . . . .	48
<b>16 Remerciements</b>	<b>49</b>
<b>17 L'auteur</b>	<b>50</b>
<b>18 Copyleft</b>	<b>51</b>

## Table des figures

1 Démarche de création d'un document L <sup>A</sup> T <sub>E</sub> X. . . . .	5
2 Layout standard et fullpage . . . . .	13
3 Le même logo, cette fois redimensionné (réduit) . . . . .	20
4 Encore le même logo réduit, cette fois étiré . . . . .	20
5 Toujours le même logo réduit, cette fois tourné . . . . .	20
6 Notre premier courrier . . . . .	38
7 Notre courrier s'étoffe . . . . .	42
8 Notre premier téléfax . . . . .	45
9 Une version plus évoluée . . . . .	47
10 joli essai . . . . .	49

### Note de mise à jour

Le document que vous avez entre les mains est la 5ème version mise à jour en juin 2004. Cette version est la suite du document original datant de 2002 et qui a été largement diffusé sur Internet, notamment sur le site de lea-linux.org. Les ajouts de cette nouvelle version concernent notamment :

- Une méthode permettant de fixer de façon fiable et simple la police appliquée à l'ensemble du document
- Un complément portant sur la "profondeur" de la numérotation de sections et celle de la table des matières
- Une méthode pour créer un symbole € qui soit acceptable et qui soutienne la comparaison avec un traitement de texte très courant édité du côté de REDMOND (USA) ainsi qu'une variante permettant la mise en place facile de logo contenus dans le texte.

- 
- La description de l'une des méthodes permettant la mise en place d'une bibliographie.
  - Un chapitre sur la création d'un index
  - Un chapitre entièrement refondu portant sur l'utilisation avancée, notamment les inclusions de fichiers et les macro commandes paramétrables.
  - Quelques retouches de vocabulaire (et de grammaire !)

# 1 Introduction

## 1.1 Version décrite

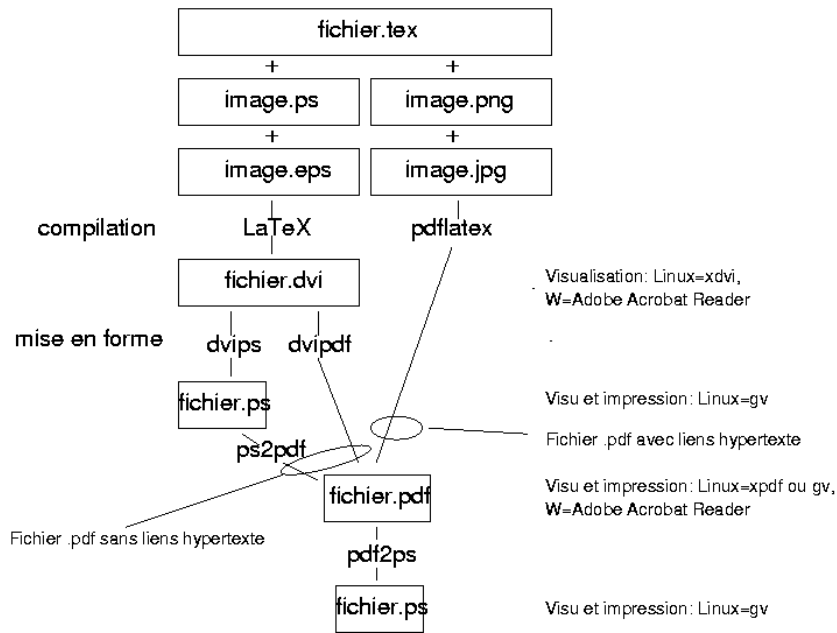
Ce document décrit l'utilisation de  $\LaTeX$  en environnement LINUX Mandrake 8.0 et suivantes (testé jusqu'à la 9.2). J'ai vérifié par ailleurs que les explications données ici s'appliquent également en à  $\LaTeX$  livré avec FreeBSD 5.2.1.  $\LaTeX$  existe dans une multitude de versions chacune adaptée à un système d'exploitation. Vous ne devriez donc pas avoir de problèmes à vous procurer la version qui tourne sur votre machine même si vous êtes l'un de ces aficionados du système d'exploitation créé par Daniel PORTES et ces gens. La version  $\LaTeX$  décrite est la 2 $\epsilon$ .

## 1.2 Compatibilité d'autres système de création de documents

$\LaTeX$  compile vers son propre format de document signé d'un suffixe .dvi qui signifie device independant. Ce format n'est pas propre à  $\LaTeX$  mais est utilisé par d'autres logiciels dont Groff, information intéressante s'il en est mais qui, tout à fait entre nous, ne présente qu'un intérêt anecdotique. Les logiciels livrés avec l'installation  $\LaTeX$  de base permettent toutefois d'exporter très facilement vers les formats beaucoup plus usuels :

- .ps Format PostScript créé pour la gestion des imprimantes mais qui peut être relu est exploité par une masse de logiciels. Le logiciel dvips est une des solutions pour créer du .ps, exemple : `dvips -o test.ps test.dvi` réalise l'exportation vers un fichier test.ps de votre fichier test.dvi compilé par  $\LaTeX$ , -o signifie «output». Dans le même ordre d'idée `dvips test.dvi` enverra le fruit de la compilation vers votre imprimante, en fait vers le filtre d'imprimante, que certains appellent le driver, et qui sait convertir le .ps en langage d'imprimante. Une autre solution pour imprimer consiste à utiliser gv (ghostview)
- .pdf Format portable document file introduit par Adobe. L'exportation vers ce format peut se faire, par exemple, avec ps2pdf ce qui suppose que vous ayez créé au préalable un fichier PostScript ou alors dvipdf qui utilise directement le fruit de la compilation  $\LaTeX$ . Nous verrons au chapitre 11 , page 31 que le format .pdf peut également être créé directement en compilant au moyen de pdflatex plutôt qu'avec le compilateur  $\LaTeX$ .
- .html Format hyper text markup language utilisé pour la création de documents hypertextes tels que ceux utilisés sur Internet ou tout autre intranet. Dans notre cas c'est la commande latex2html qui prend en charge la transcription. Reportez vous éventuellement à l'aide en ligne très complète, un peu plus de 600 lignes, accessible par latex2html -help. L'un des intérêts de ce format est de pouvoir très facilement être relu par les logiciels de traitement de texte classiques. Un autre aspect intéressant est la réalisation

FIG. 1 – Démarche de création d'un document  $\LaTeX$ .



de présentations très professionnelles, l'exportation pouvant se faire, c'est une des options, section par section. Dans ce cas on retrouve en tête de chaque page html un ensemble de flèches de navigation très pratique lors d'un exposé. En effet ces flèches se substituent aux flèches du navigateur et permettent d'utiliser ce dernier en mode plein écran tout en conservant l'entier contrôle de la navigation.

Il existe bien d'autres formats, tapez par exemple `dvi` suivi d'une frappe sur la touche `TAB` pour obtenir la liste des commandes d'exportation. Vous y découvrirez par exemple la possibilité d'exporter directement au format fax, par défaut en haute résolution.

N'hésitez pas par ailleurs à «fouiller» tous les logiciels, très nombreux, présents dans le monde du libre, (je ne fouille pas le secteur des softs commerciaux pour lesquels je ne possède pas de licence...) vous y découvrirez des «passerelles» insoupçonnées, par exemple `Abiword` qui permet de convertir, certes de façon maladroite, le format d'un célèbre traitement de texte cher à Daniel Portes en  $\LaTeX$ .

Le diagramme 1 schématise la démarche qui va vous permettre de créer un document  $\LaTeX$ . Gardez toujours cette structure à l'esprit. L'essentiel de la présente étude, parce que c'est la démarche classique, va vous permettre de réaliser un document final en suivant la branche de gauche. La description de la démarche par la branche de droite fera l'objet du chapitre 11 spécifique à la création directe de documents au format `.pdf`.

---

## 2 Présentation

### 2.1 $\LaTeX$ et vos habitudes de travail

*$\LaTeX$  est un logiciel d'abord complexe et dont l'utilisation est réservée aux universitaires et autres spécialistes.*

Faux ! Vous vous êtes peut-être déjà cassé les dents sur des premiers essais infructueux, cela ne signifie toutefois pas que  $\LaTeX$  ne soit pas accessible au plus grand nombre. Simplement  $\LaTeX$  est un système de création de documents totalement à l'écart des sentiers battus, des cliquadrômes et autres interfaces graphiques stérilisantes. Son abord nécessite un investissement initial, un peu comme un ticket d'entrée. L'objectif de ce document est de vous fournir ce ticket d'entrée, de vous permettre de créer les premiers documents, de vous convaincre de la validité du concept.

$\LaTeX$  n'est pas un "traitement de texte" au sens habituel de ce terme. Vous ne rédigerez jamais un document avec  $\LaTeX$ .  $\LaTeX$  est un compilateur qui traite un texte source et le transforme en un document exploitable par des logiciels de visualisation ou d'impression.  $\LaTeX$  a donc, dans la chaîne de création du document final, un rôle à la fois fondamental, puisque c'est lui qui effectue toute la mise en page du document, et peu spectaculaire puisqu'il n'intervient ni lors de la création du texte source, ni lors de la visualisation.

Les principales étapes de la création d'un document avec  $\LaTeX$  sont :

- La création et l'édition du texte source au moyen de l'éditeur de votre choix, Emacs, VI ou autre ;
- la compilation avec  $\LaTeX$  ;
- l'exploitation du document créé, sa visualisation au moyen de xdvi ou son impression via dvips.

Le texte source est le texte de votre document final agrémenté de balises de mise en forme, un peu comme un texte source html. Toute la difficulté de la création d'un document avec  $\LaTeX$  est donc de connaître un minimum de syntaxe de façon à pouvoir mettre en place les balises nécessaires et traiter les erreurs de compilation qui ne manqueront pas de se produire en cas d'emploi erroné des balises.

La lecture de ce document vous permettra d'apprendre à utiliser certaines de ces balises. Il vous permettra de créer des rapports, mais aussi, comme nous le verrons bientôt, des courriers, des téléfax et, dans une version future, des planches pour rétroprojecteur. Par convention les fichiers sources ont un suffixe .tex mais il n'y a aucune obligation. L'avantage d'utiliser l'extension conventionnelle est qu'elle est prise en compte implicitement lors de la compilation. La ligne de commande

```
[jml@localhost] $ latex test
```

compilera automatiquement le fichier test.tex alors qu'une terminaison inhabituelle .latex par exemple vous aurait obligé à préciser l'extension dans la ligne de commande. La compilation crée un fichier du même nom que le fichier source mais avec une extension .dvi. Il s'agit d'un format spécifique à  $\LaTeX$  et à d'autres softs qui sera exploité à son tour par des logiciels assurant l'affichage à l'écran ou l'impression. Ici aussi l'extension est implicite. La syntaxe

```
[jml@localhost] $ xdvi test
```

affiche test.dvi dans une fenêtre du visualiseur.

L'essentiel de ce document va décrire le travail avec le compilateur standard  $\LaTeX$ . Le chapitre 11 page 31 décrit l'utilisation d'un autre compilateur, pdflatex, à utiliser préférentiel-

lement à  $\LaTeX$  si vous souhaitez produire un document final au format .pdf, surtout si vous souhaitez que ce document utilise les liens hyper texte, un peu comme une page .html. C'est le format idéal pour transmettre un rapport (ou tout autre document) par courrier électronique à un correspondant qui utiliserait encore le système d'exploitation symbolisé par les fenêtres. Acrobat Reader est un lecteur de fichier .pdf gratuit et qui s'installe sans difficulté sur ce système. Sous Linux les fichiers .pdf peuvent être lus au moyen de gv (ghostview) mais cet afficheur ne sait pas exploiter les liens hyper texte. Xpdf est un autre afficheur qui exploite correctement les liens hyper texte. A noter que Acrobat Reader est présent aussi sous Linux, FreeBSD ou d'autres UNIX, par exemple IRIX de SGI.

A titre personnel je n'utilise presque plus que le format pdf obtenu directement par pdflatex, tout en veillant, par une petite compilation de test, que  $\LaTeX$  reste capable de traiter mon code source. J'assure ainsi une compatibilité maximale de mon code tout en bénéficiant des avantages du pdf, portabilité fantastique et liens hyper texte.

Ajouté le  
13 juin  
2004

## 2.2 Le travail avec $\LaTeX$ .

La création d'un document  $\LaTeX$  implique l'utilisation simultanée de trois logiciels, l'éditeur de texte qui intervient sur le code source,  $\LaTeX$  lui même qui recompile sur demande le texte source et qui remet à jour le fichier .dvi correspondant et xdvi, le visualiseur qui vous permet de suivre l'évolution de votre document, la pertinence de vos choix de mise en forme etc..

Pour ma part je m'organise comme suit :

- Sous KDE la combinaison ALT-F2 me permet d'ouvrir une fenêtre de dialogue dans laquelle je demande le lancement d'Emacs.
- Toujours ALT-F2 pour obtenir une fenêtre rxvt qui me servira à lancer la compilation à la demande puis, si la compilation est un succès, la visualisation avec xdvi. Les compilations successives peuvent être relancées simplement en défilant l'historique de frappe avec la flèche «haut» suivie de la frappe de «entrée» (je suppose ici l'emploi de bash).
- INFO : xdvi est une petite merveille qui est très sensible à tout passage de souris. Dès que xdvi se rend compte qu'il a le focus il vérifie si la version affichée est bien la dernière version en date. Dans le cas contraire il recharge automatiquement la dernière version et affiche celle-ci à la page à laquelle vous aviez arrêté votre lecture. Il s'en suit qu'il est inutile de refermer et d'ouvrir xdvi en permanence. Vous avez donc tout intérêt à laisser xdvi ouvert, soit en le lançant directement à partir de KDE, soit en le lançant avec le suffixe & depuis rxvt, soit, si vous avez oublié le &, en faisant un CTRL-z dans la fenêtre rxvt pour interrompre xdvi et reprendre la main dans rxvt, suivi d'un bg (pour back ground) qui relance xdvi en arrière plan.

La création d'un document  $\LaTeX$  est donc un passage permanent entre les trois fenêtres, la fenêtre d'édition, celle du compilateur et la fenêtre de visualisation. Une autre solution efficace consiste à compiler à partir de Emacs en utilisant le menu Command / LaTeX Interactive. Cette procédure crée une nouvelle fenêtre Emacs qui contiendra le résultat de la compilation, exactement comme si vous aviez lancé depuis une fenêtre rxvt.

Vous pouvez par ailleurs visualiser avec xdvi directement depuis emacs en utilisant le menu Command/View. N'oubliez pas de valider le choix par défaut qui s'affiche dans la ligne de commande d'Emacs par ENTER faute de quoi rien ne se passe !

---

## 3 Installation de L<sup>A</sup>T<sub>E</sub>X

Suivant le type d'installation réalisé sur votre linuxette L<sup>A</sup>T<sub>E</sub>X sera ou ne sera pas installé. Il suffit de taper latex à l'invite système dans une fenêtre terminal pour être fixé.

S'il devait arriver que L<sup>A</sup>T<sub>E</sub>X ne soit pas installé voilà, dans l'ordre, la suite des packages à installer (sous LINUX) :

- ed, un éditeur ligne de commande, exactement le truc tellement spartiate que personne n'a envie de l'utiliser mais il est nécessaire pour le package suivant,
- tetex, le moteur du système,
- tetex-latex, la couche L<sup>A</sup>T<sub>E</sub>X qui fournit des macro commandes basées sur le langage T<sub>E</sub>X,
- tetex-dvips, la moulinette qui permet de créer des fichiers imprimables au format .ps en partant du produit de la compilation par L<sup>A</sup>T<sub>E</sub>X,
- tetex-xdvi, une autre moulinette qui prend en charge l'affichage à l'écran du même produit compilé,
- latex2html, le logiciel qui vous permettra d'exporter un document L<sup>A</sup>T<sub>E</sub>X au format .html.

Lorsque vous aurez installé tout ça vous serez paré pour faire connaissance avec L<sup>A</sup>T<sub>E</sub>X. Nous découvrirons au fil du document des compléments précieux qu'il conviendra d'installer au fur et à mesure pour peaufiner votre installation.

## 4 Structure d'un document L<sup>A</sup>T<sub>E</sub>X.

### 4.1 Les balises, syntaxe générale

Les balises du langage L<sup>A</sup>T<sub>E</sub>X sont introduites par une barre oblique inverse \. Exemple : `\title{Titre du document}`.

Il est possible de positionner des commentaires. Ceux-ci sont introduits par un %. Tous les caractères situés après cette balise sur la ligne sont ignorés par le compilateur. Ce symbole est très pratique pour neutraliser provisoirement une ligne d'entête sans pour autant l'effacer ce qui permettra de revenir en arrière sur la modification.

### 4.2 Le préambule

#### 4.2.1 Le type de document

Le premier travail va consister à définir le type de document que vous souhaitez créer. Cette information, valable pour l'ensemble du fichier, figure dans le préambule. Les types de documents habituellement disponibles sont article, letter, seminar (pour la rédaction de planches rétroprojecteur).

Le type article est destiné à la rédaction de rapports courts, d'articles ou de tout document de ce genre. C'est le format de base pour débiter avec L<sup>A</sup>T<sub>E</sub>X, celui qui a servi à la rédaction de l'article que vous lisez par exemple. Les types book et report sont des variantes qui se distinguent par le fait qu'elles sont mieux adaptées à la rédaction de documents plus volumineux (titre isolé sur une page, titres de chapitres etc..).

Le type letter présent dans L<sup>A</sup>T<sub>E</sub>X est utilisable mais présente un intérêt moindre que le type

lettre mis à disposition par l'Observatoire de Genève (voir plus loin au chapitre 13, page 36) et qui pourra servir à la rédaction de courriers et de téléfax.

La première ligne de notre préambule sera donc :

```
\documentclass{article}
```

On pourra préciser immédiatement le format de papier en option en utilisant la syntaxe :

```
\documentclass[a4paper]{article}
```

Vous aurez noté la syntaxe : l'argument est placé entre {} et les options, éventuellement séparée par des virgules, entre []. Nous aurions ainsi pu écrire

```
\documentclass[a4paper, french]{article}
```

pour préciser à la fois le format de papier et la langue du document.

### 4.2.2 La langue du document

Pour ma part j'utilise volontiers la syntaxe :

```
\usepackage[francais]{babel}
```

(et non l'option french dans \documentclass) qui va veiller au respect de la typographie française, du format de date etc. Comme dans le cas précédent nous notons la présence d'un argument

```
{babel}
```

et d'une option

```
[francais].
```

Attention, dans ce cas francais s'écrit sans cédille ! Les options peuvent être multiples, nous aurions pu écrire

```
[francais, german]
```

pour préparer un texte bilingue français allemand. Le choix de la langue s'effectue ensuite au moyen d'une balise

```
\selectlanguage{francais}
```

par exemple. Notez que :

- L'ordre des options est important, sans précision particulière, par exemple tant que vous ne mettez pas de balise

```
\selectlanguage
```

c'est la dernière langue listée qui sera utilisée, dans notre cas l'allemand.

- La position de la balise \selectlanguage n'est pas indifférente. Elle doit figurer dans le corps du texte, non dans le préambule.

### 4.2.3 Les caractères accentués

Nos amis les rosbeefs utilisent un jeu de caractère désespérément triste et sans le moindre accent. Avantage : ayant moins de caractères il peut se contenter d'un codage 7 bits soit 128 caractères. C'est pratique pour eux, pas pour nous. Une première solution pour créer des caractères accentués est d'utiliser des séquences absconses du genre \'e pour créer un é ! C'est intéressant si on travaille sur un document qui doit pouvoir rester compatible avec d'autres systèmes d'exploitation éventuellement non francisés. Avouez qu'on aimerait autant se passer de ce genre de frappes. Ceux qui pratiquent le langage html connaissent une technique analogue, le é s'écrit en effet &eacute;. Pour éviter cela il est possible de forcer le codage des caractères dans un jeu 8 bits qui reproduise les caractères

accentués. Ceci s'obtient par :

```
\usepackage[latin1]{inputenc}
```

L'option `latin1` correspond au jeu classique utilisé par les langues latines et satisfait tout à fait à nos besoins de franchouillards à bérets. Petite information à l'attention des germanophones, c'est mon cas, le ß cher à nos voisins d'Outre-Rhin s'obtient par `\ss`. Il suffisait de le savoir....

Nota : Lorsque vous tapez `\ss` dans un mot laissez un espace après le double s afin de marquer, pour  $\text{\LaTeX}$  la fin de la séquence introduite par `\`. Cet espace ne sera pas reproduit dans le texte. Par contre, si le ß tombe à la fin d'un mot alors il convient d'orthographier `\ss\` en mettant une barre oblique accolée entre le `ss` et l'espace. La barre oblique indique la fin de séquence, l'espace prend alors toute sa valeur et sera reproduit dans le texte. Là aussi il suffisait de le savoir....

`inputenc` règle donc le problème du codage des caractères dans le fichier source, pas celui de ce codage dans le fichier `.dvi` résultant de la compilation. Pour cela nous allons demander explicitement au compilateur d'utiliser un codage de sortie qui tienne compte des accents avec la ligne :

```
\usepackage[T1]{fontenc}.
```

Il existe différents codage, `T1` est le plus récent, c'est donc le codage de base pour celui qui souhaite se lancer dans  $\text{\LaTeX}$ .

Les 4 lignes que nous venons de mettre en place constituent l'entête de base autour de laquelle vont s'articuler toutes les variantes imaginables de document.

```
\documentclass{seminar}
```

 introduira un document destiné à la création de planches rétroprojecteur,

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

 demandera une taille de base de la police employée de 12 pt au lieu des 10 pt utilisés par défaut etc. Il est grand temps maintenant de taper dans la butte et de commencer à construire notre texte.

### 4.3 Les environnements

$\text{\LaTeX}$  utilise plusieurs familles de balises qui se distinguent par leur portée. Lorsque nous écrivons `\textbf{Texte en gras}` pour obtenir **Texte en gras** nous utilisons une balise locale qui modifie le texte passé en argument. Ceci est surtout valable pour des ajustements locaux. Une balise `\selectlanguage{francais}` vue plus haut transforme le texte à partir de la position de la balise jusqu'à la fin ou jusqu'à rencontrer une balise qui annule la précédente, `\selectlanguage{german}` par exemple. Pour définir précisément une action plus large, sur le paragraphe ou l'ensemble du document il convient de mettre en place une balise de début et une balise de fin selon une syntaxe simple `\begin{} \end{}`. La création d'un document par exemple passe par la mise en place de deux balises `\begin{document}` et `\end{document}`. Cette notion de balise enveloppante est courante pour définir des environnements tels que document, liste, figure, table, tabulations, barre de révision etc. Attention ! L'erreur classique consiste à mettre en place une balise de début et d'oublier la balise de fin. Pour tenter de l'éviter faites comme moi, tapez les deux balises à la suite l'une de l'autre et "remplissez" entre les balises. Vous noterez que le premier type de balise à action ponctuelle existe dans deux variantes que vous devrez vous habituer à distinguer, la variante exposée plus haut est une version dans laquelle le modificateur est externe. Il existe une

syntaxe dans laquelle le modificateur est contenu dans l'accolade. **Texte large** s'obtient par exemple avec `\LARGE Texte large`. Des combinaisons des deux grammaires sont d'ailleurs possibles, par exemple **Texte gras et large** est généré par `\textbf{\LARGE Texte gras et large}`. Simple non ?

## 4.4 Le plan d'un document

### 4.4.1 Notions générales

Le plan d'un document classique comporte des notions connues, titre, nom d'auteur, date, table des matières, résumé, mais aussi sections, sous-sections, sous-sous-sections etc. Vous l'aurez deviné, toutes ces notions trouvent leur traduction dans  $\LaTeX$  sous forme d'une balise adaptée à les décrire. Mais plutôt qu'un long exposé essayez plutôt le fichier de test suivant que vous sauvegarderez sous `test.tex` (pour faire preuve d'originalité!).

```
\documentclass[a4paper]{article}
\usepackage[français, german]{babel} % Typographie, date en format F ou D
\usepackage[latin1]{inputenc} % Accents standard clavier
\usepackage[T1]{fontenc} % Accents dans le dvi
\begin{document} % Balise de début de document
\selectlanguage{français} % Sélection du langage
\title{Premier essai} % Titre général du document
\author{moi-même} % Le nom d'auteur qui apparaîtra dans le titre
\date{01/01/2002} % La date qui apparaîtra dans le titre
\maketitle{} % Force LaTeX à mettre en place le titre
\tableofcontents{} % Mise en place de la table des matières
\section{Titre du premier paragraphe} % Titre de paragraphe
Mon premier essai transformé! % Votre texte
\end{document} % Ne pas oublier de refermer les balises ouvertes!
```

Salez, poivrez, mettez au four préchauffé.. euh, excusez moi, je pensais à autre chose. Plus sérieusement quelques indications complémentaires sur ce premier fichier test :

- Les balises `\begin{document}` et `\end{document}` sont obligatoires.
- Si vous ne fixez pas la date comme ci-dessus  $\LaTeX$  met automatiquement la date du jour de la compilation sauf à préciser `\date{}` ce qui bloquerait l'impression de la date
- La balise `\maketitle{}` fixe l'endroit où  $\LaTeX$  placera le titre (ensemble titre + auteur + date). Si vous omettez cette balise le titre ne sera pas généré, si vous la placez ailleurs.. mais essayez donc !
- La balise `\tableofcontents{}` place la table des matières. Comme la langue française a été sélectionnée soigneusement le titre de cette table des matières devrait être "Table des matières". Si vous voyez apparaître Contents ou Inhaltsverzeichnis après compilation ne cherchez pas, il y a une erreur dans la sélection de langue.
- Les différentes sections, sous-sections etc.. qui vont constituer l'articulation de votre document sont introduites par les balises `\section{}`, `\subsection{}` etc. Leur numérotation est automatique. Par défaut le titre passé en argument à la balise apparaît aussi bien dans le texte que dans la table des matières. Vous pouvez toutefois passer en option, donc entre `[]` le titre que vous souhaitez voir apparaître dans la table des

---

matières si vous souhaitez un texte plus concis par exemple. Faites l'essai d'ajouter une deuxième section avec `\section [Section2] {Titre du deuxième paragraphe}`.

- Vous serez certainement surpris de constater que certaines modifications, comme celle que je viens de proposer ci-dessus et qui agit directement sur la table des matières ne sont pas prises en compte correctement par  $\LaTeX$ , disons plutôt à l'instant ou vous l'attendez. Il ne s'agit pas d'une erreur, simplement d'un point de technique que vous devez conserver à l'esprit. A la première compilation  $\LaTeX$  génère la table des matières et la range dans un fichier du même nom que le fichier source mais avec une extension `.toc`, pour «table of contents». Ce n'est qu'à la deuxième compilation que le contenu de ce fichier sera effectivement pris en compte dans le document final. Pensez donc à faire systématiquement une double compilation avant toute impression finale de votre document faute de quoi vous risquez de laisser les informations actualisées dans un fichier `.toc` qui ne serait pas recopiées dans le fichier de sortie.

Notez que nous retrouverons cette notion plus tard, à l'occasion de l'examen d'autres points comme la numérotation des pages d'un fax, la mise à jour de références internes etc..

#### 4.4.2 "Profondeur" et numérotation de la table des matières

Vous pourrez travailler longtemps avant d'avoir besoin de ce chapitre puis, un jour, vous tomberez sur l'un des problèmes suivants :

- Vous aurez utilisé les balises `\section`, `\subsection`, `\subsubsection`, `\paragraph`, et `\subparagraph`, pour structurer votre document et constaterez que la table des matières ne détaille pas aussi loin que vous le souhaitez cette structure.
- Vous avez, comme ci-dessus, mis en place une belle structure mais constatez que la numérotation s'arrête, par exemple aux balises `\subsubsection` et ignore les paragraphes et sous paragraphes.

La solution consiste ici à modifier les valeurs par défaut des deux paramètres `\secnumdepth` et `\tocdepth`. Si la langue des Beatles ne vous est pas tout à fait étrangère vous aurez certainement deviné que :

**secnumdepth** règle la profondeur de la numérotation. Par exemple une valeur de 3 va numéroté les sections, sous-sections et sous-sous-sections.

**tocdepth** ajuste la profondeur de la table des matières.

En supposant une valeur 3 pour `\secnumdepth`, une valeur de 2 affectée à `\tocdepth` limitera la définition de la tables des matières aux sections et sous-sections, les ss-sections étant toutefois numérotées dans le texte. En réglant `\tocdepth` à 4 on verrait apparaître dans la table des matières des titres de paragraphes non numérotés.

## 5 Mise en page

### 5.1 Format de la page

La question de la mise en page est d'un abord assez difficile. Vous aurez certainement remarqué que les pages que vous obtenez pour l'instant sont assez peu remplies. On aime ou on aime pas, c'est une question de goût. S'il s'agit de créer un document de travail

Ajout le  
13 juin  
2004

## 5.1 Format de la page

qui sera relu, raturé, annoté alors cette disposition est correcte. S'il s'agit d'imprimer un document final alors il peut s'avérer intéressant de remplir un peu mieux les pages, mais comment? Pour approcher la réponse à cette question je vous propose de demander à L<sup>A</sup>T<sub>E</sub>X de nous fournir l'aide souhaitée avec la commande `\layout`. Celle-ci, intégrée à un fichier source L<sup>A</sup>T<sub>E</sub>X dessine une page et les différents champs réservés à l'entête, au pied de page, au corps du texte et aux notes en marge. La commande `\layout` nécessite de charger le package `layout`. Je vous propose donc de créer un nouveau fichier que nous appellerons `mep.tex` (`mep` pour mise en page) libellé comme suit :

```
\documentclass[a4paper]{article}
\usepackage[francais]{babel}
\usepackage[francais]{layout}
\begin{document}
\layout
\end{document}
```

Admettez qu'il est difficile de faire plus simple! L'objectif est ici simplement de provoquer l'affichage de la mise en page standard avec les commentaires en français. Vous devriez obtenir quelque chose comme la figure 2 page 13.

Refaisons maintenant la même opération (édition, compilation, affichage) en ajoutant :

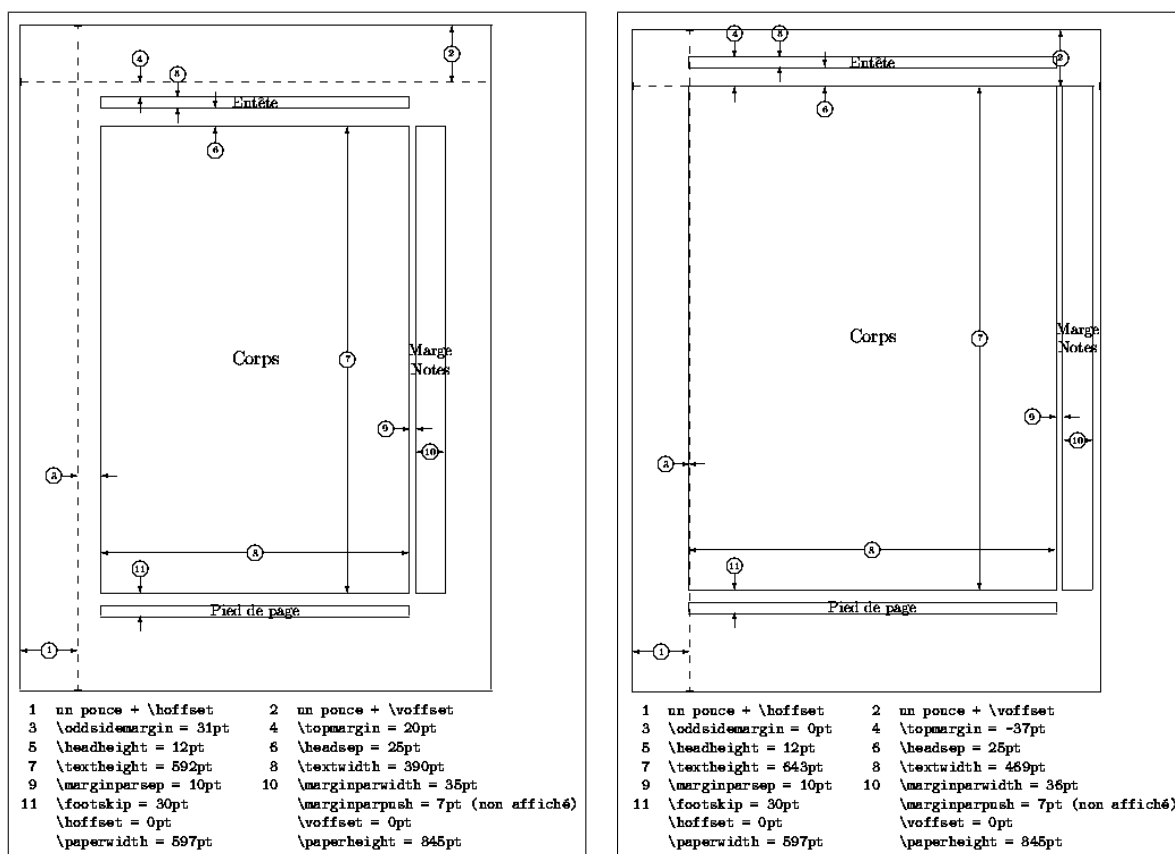


FIG. 2 – Layout standard et fullpage

```
\usepackage{fullpage}
```

dans le préambule, juste avant la balise d'ouverture du texte.

Vous aurez noté immédiatement quelques différences :

- Le rectangle qui figure l'entête de page est passé dans la bande d'un pouce qui borde la page en haut. Nous avons préalablement une marge de 20 pt sous la bande, maintenant une marge négative de 37 pt. Cette modification a augmenté sensiblement la hauteur utile du texte
- La position du pied de page est inchangée
- La marge de gauche est limitée strictement à la bande d'un pouce alors que nous avions précédemment un offset de 31 pt
- La largeur du texte a augmenté, décalant d'autant la position des notes en marge.

Cette petite expérience a montré l'action induite par le package `fullpage`. Elle vous donne aussi, et c'est le principal avantage de cet exercice, les noms de quelques paramètres principaux de mise en page. La nouvelle mise en page est nettement plus efficace que la mise en page de base. Elle présente toutefois encore un petit inconvénient, le texte pourrait être encore un peu plus long, la position du pied de page n'a pas changé. Pour allonger le texte il suffit maintenant d'ajouter une ligne

`\textheight=24cm` ou `\setlength{\textheight}{24cm}` dans le préambule, après l'appel du package `fullpage`. Les deux syntaxes sont autorisées, la plus courte est bien sûr préférable à mon avis... Vous pouvez ainsi retoucher tous les paramètres de la mise en page.

Les dimensions peuvent être données dans tout système d'unité reconnu par  $\text{\LaTeX}$ . Par défaut l'unité utilisée est le point, abréviation «pt», valeur  $1/72$  de pouce soit environ 0,35 mm mais vous pouvez utiliser également le mm et le cm si vous préférez.

## 5.2 Entête et pied de page

Pour agir sur les entêtes et bas de page il est nécessaire d'utiliser un package supplémentaire, `fancyhdr`. Son utilisation nécessite de le charger et de l'activer. Les deux lignes à ajouter dans le préambule sont donc :

```
\usepackage{fancyhdr}
\pagestyle{fancy}
```

Recompiler le fichier, vous devriez voir immédiatement l'effet de ces deux lignes. Une barre horizontale souligne maintenant la zone d'entête qui contient à gauche le nom de la section courante, à droite celui de la sous-section, de la sous-sous-section, du paragraphe suivant le cas en cours. Il s'agit là d'un format par défaut sur lequel vous pouvez agir comme nous le verrons bientôt.

Voilà, par exemple, la présentation que j'utilise sur mon lieu de travail. Celle-ci comporte en entête, à gauche, un petit logo de l'entreprise, suivi à droite de l'identification du site. Le pied de page comporte trois zones, de gauche à droite :

- l'indication Page suivie du numéro, en police réduite (`tiny`) et en italique (`textsl`)
- un autre petit logo représentant mes initiales
- la date du jour dans la même police que le numéro de page

Le tout est séparé du texte par une ligne horizontale d'épaisseur 0.4 pt.

```
\lhead{\includegraphics*[width=2cm]{rehau.ps}}
\rhead{\tiny\textsl{Usine de BOURGES}}\hfill \tiny\textsl{La Chapelle}}
\lfoot{\tiny\textsl{Page \thepage}}
```

---

```
\cfoot{\includegraphics*[width=2cm]{jml4.ps}}
\rfoot{\tiny\textsl{\today}}
\renewcommand{\footrulewidth}{0.4pt}
```

Dans les quelques lignes qui précèdent vous aurez bien sûr deviné que `head` signifie entête, que `foot` désigne le pied de page et que `l`, `c` et `r` sont des raccourcis pour gauche (`left`), centre (`center`) et droite (`right`). Ces lignes sont insérées entre les lignes précédentes, c'est à dire dans le préambule, entre celle qui effectue l'appel à `fancyhdr` et celle qui active cette extension. Attention, ne vous arrachez pas les cheveux lors de vos essais. Vous risquez en effet, si vous utilisez des commandes `\tableofcontents{}` ou `\layout` de vous heurter rapidement au constat suivant :

- la page sur laquelle apparaît la table des matières n'est pas modifiée et ne reçoit donc ni entête, ni pied de page personnalisé
- il s'en suit que si votre document tient sur une page qui contient la table des matières ou si vous ne pensez pas à aller voir la page 2 vous avez l'impression que «quelque chose ne marche pas!».

Cela est du au fait que la mise en place d'une table des matières réinitialise les champs entête et pied de page sur les pages concernées. Pour contourner cette intervention tapageuse si celle-ci vous gêne il suffit de taper `\thispagestyle{fancy}` juste après l'appel de `\tableofcontents{}` pour forcer la remise en place des entêtes et bas de page. L'appel à «`layout`» provoque également (évidemment) une ré-initialisation des champs entête et pied de page. Note : `\thispagestyle{fancy}` n'est fonctionnel qu'avec la classe de document `article`. Il semblerait qu'il ne soit pas possible d'avoir une page de couverture "fancy" avec une classe `report` par exemple.

## 6 Étoffer votre texte

### 6.1 Résumé de document

`indexAbstract`

Si vous construisez un document important il peut être intéressant de mettre en place en tête de document un résumé. Ce résumé est placé entre deux balises `\begin{abstract}` et `\end{abstract}`. Ce texte sera placé sous un titre «Résumé» qui sera centré au dessus de votre synthèse. Notez que si le texte «Résumé» ne correspond pas à vos souhaits il vous est loisible de le changer en modifiant la valeur de la variable `abstractname` grâce à une ligne `\renewcommand{\abstractname}{nouveau titre}`. C'est ce que j'ai fait sur le présent document pour lequel j'utilise l'abstract pour placer les notes de mise à jour. Le titre de ce paragraphe est donc créé par `\renewcommand{\abstractname}{Note de mise à jour}`. Il s'agit là d'une remarque très générale. N'hésitez surtout pas à lire le fichier `article.cls` (avec Emacs par exemple, C-s vous permet de rechercher rapidement des séquences de caractères) de façon à faire connaissance progressivement avec le nom de toutes les variables. Avec un peu d'expérience vous arriverez à flairer la signification de `listfigurename`, `contentsname` ou autre.

Attention toutefois, nous sommes français, il s'en suit qu'il serait illusoire de chercher la séquence «Résumé» dans `article.cls`. En effet ce titre est redéfini par la francisation du texte (la

ligne `\usepackage{francais}[babel]` par exemple). La recherche doit s'effectuer sur «abstract». Plus généralement essayez de ne pas perdre de temps à chercher des séquences de caractères contenant des accents. Vous ne pouvez jamais être certain du codage des accents (voir ?? page ??) qui a été employé et risquez donc de «passer à côté» de la séquence souhaitée.

## 6.2 Sauts de ligne

On voit écrit partout que  $\LaTeX$  s'occupe de la mise en page et que l'auteur peut se concentrer sur son texte. La perfection n'est pas de ce monde. Il s'en suit que vous pouvez vous trouver dans la nécessité d'introduire un saut de ligne, voire même un saut avec espacement. La commande `\\` éventuellement complétée par `\smallskip`, `\medskip` ou `\bigskip` sont là pour forcer  $\LaTeX$  à faire ce que vous souhaitez. `\\` fait un saut de ligne simple. Précédé de `\bigskip` la commande `\\` produit un saut avec espace. Notez que les balises `\smallskip` et `\medskip` définissent elles aussi des sauts de lignes, mais avec des espaces moins importants.

La valeur des différents espaces est fixée par la valeurs des variables `bigskipamount`, `medskipamount` et `smallskipamount`. Notez bien que les hauteurs de saut doivent être définies juste avant l'instruction de saut, selon de schéma suivant `\\bigskip\\`. Cette séquence permet par exemple d'obtenir le saut suivant

qui est légèrement plus grand qu'un saut de ligne normal. Une autre possibilité de réaliser des sauts de ligne est de taper la touche "ENTREE" deux fois de suite de telle sorte à introduire une ligne vide dans le code source.

A la différence de `\\` cette variante permet de réaliser un saut de ligne avec indentation. C'est d'ailleurs de cette façon que j'ai introduit le paragraphe que vous lisez en ce moment, après avoir placé à `\bigskip` juste après .... dans le code source.

## 6.3 Les aménagements de police de caractère

### 6.3.1 La forme des caractères

$\LaTeX$  utilise des règles qui lui sont propres pour mettre en gras, augmenter la taille de la police etc, par exemple pour les titres de documents, les titres de chapitres etc. Vous avez aussi la possibilité d'intervenir sur le rendu final en forçant un comportement spécifique, pour obtenir du texte en **gras**, de l'*italique*, de l'*emphase*. Ces trois transformations sont obtenues respectivement par les balises `\textbf{}`, `\textit{}`, `\emph{}`. Vous ne voyez pas de différence entre `\textit{}` et `\emph{}`? J'explique. Dans du texte normal `\textit{}` et `\emph{}` auront tous deux pour effet de mettre le texte en italique. Dans un texte qui serait déjà en italique par contre, la balise `\emph{}` va forcer le texte en police normale ce qui va le distinguer du texte environnant et le mettre en valeur, en «emphase».

Modifié  
le 20  
juin  
2004

### 6.3.2 La tailles des caractères

On peut jouer les tailles de polices, de `tiny` à **huge** en passant par `scriptsize`, `footnotesize`, `small`, `normalsize`, `large`, `Large`, `LARGE` et **huge** ! Pour passer un passage en petits caractères on peut, par exemple, mettre en place la syntaxe `{\tiny ceci sera écrit en taille tiny}` ce qui donne : `ceci sera écrit en taille tiny.`

La liste suivante donne une idées des tailles de caractères qu'il est possible d'obtenir en fonction des balises employées :

<code>tiny</code>	essai de texte
<code>scriptsize</code>	essai de texte
<code>footnotesize</code>	essai de texte
<code>small</code>	essai de texte
<code>normalsize</code>	essai de texte
<code>large</code>	essai de texte
<code>Large</code>	essai de texte
<code>LARGE</code>	essai de texte
<code>huge</code>	essai de texte

Observez que les deux types de balises examinés jusque là, celles qui modifient la graisse et la forme de la police et celles qui en modifient la taille, n'emploient pas la même syntaxe. Les modificateurs de forme de caractères sont externes aux accolades, les modificateurs de taille sont contenus dans les accolades, avec le texte.

### 6.3.3 Définir une police de caractères pour l'ensemble du document

Par défaut  $\LaTeX$  utilise une police roman. Ce choix peut être modifié par l'auteur du document. Vous trouverez, à l'occasion de vos recherches dans les livres ou sur Internet différentes communications sur le sujet, par obligatoirement très claires (à mon avis). Pour ma part j'ai identifié et testé la méthode suivante. J'ajoute les lignes suivantes dans l'entête de mon document :

```
\renewcommand{\familydefault}{phv}
\selectfont
```

Combinées ces deux lignes fixent la police appliquée à l'ensemble du document.

La première ligne change la valeur de la variable `\familydefault` en la forçant à le nouvelle valeur, ici `phv`. Cette ligne, laissée seule, transformerait la police des titres de tables, de liste de figures, d'entête et de pied de page mais pas le titre général du document ni le corps de celui-ci. C'est la deuxième ligne qui va mettre à jour la police du titre et du corps.

Vous pouvez ainsi utiliser des polices très variées, par exemple :

**phv** une police Adobe Helvetica (la police du présent document)

**ptm** une autre police Adobe, cette fois en Times

**pcr** une dernière Adobe, type courier

**cmr** Computer Modern Roman

ajout le  
12 juin  
2004

**cmss** CM mais sans sérifs

**cmtt** CM Type writer, donne l'illusion d'une frappe machine à écrire

## 6.4 les environnements de liste

Les listes et énumérations sont des constituants habituels des documents.  $\LaTeX$  met à votre disposition quelques environnements spécifiques pour la réalisation de ces présentations :

**itemize** L'environnement pour les listes simples. Il est introduit par une balise `\begin{itemize}` et terminé par un `\end{itemize}`. Chaque ligne est introduite par une balise `\item`. Deux syntaxe sont possibles, `\item` suivi de la ligne qui constitue l'élément de liste et `\item{avec l'élément de liste borné par des accolades}`.

**enumerate** Se distingue du cas précédent par la syntaxe (on remplace `itemize` par `enumerate`) et le résultat puisque les lignes de la liste sont maintenant numérotées.

**description** Utilise les balises `\begin{description}` et `\end{description}`. L'environnement `description` produit très exactement ce que vous lisez en ce moment. La syntaxe est légèrement plus complexe puisque chaque `item` prend un argument entre crochets. Le paragraphe que vous lisez en ce moment commence, dans le source, par `\item [description]` Pour produire etc...

## 6.5 Inclure des éléments graphiques

$\LaTeX$  est capable d'importer de nombreux formats de fichiers graphiques. Le format PostScript est un format très courant, développé à l'origine pour générer les impressions laser. Il est reconnu par la majorité des logiciels à vocation graphique ou qui génèrent des sorties pour imprimantes. C'est donc ce format qui sera utilisé ici, plus précisément l'encapsuled PostScript, une variante du `.ps` ordinaire qui contient, en plus des informations habituelles des renseignements sur la taille de l'image. Vous vous reporterez utilement à un article «GnuPlot LyX et  $\LaTeX$ » sur <http://www.lea-linux.org> et qui traite plus particulièrement de la génération de graphiques au moyen de GnuPlot et de leur intégration dans  $\LaTeX$ . Cet article dont je suis l'auteur, sans prétendre faire le tour complet de la question, compare pour cette application différents formats graphiques, dont le PostScript et le format `.tex`. C'est à l'occasion de la rédaction du papier sur GnuPlot que m'est venu l'envie d'approfondir ma connaissance de  $\LaTeX$  ce qui conduit à l'article que vous lisez actuellement.



L'inclusion d'éléments graphiques est complexe quelque soit le logiciel avec lequel on travaille. Ne me dites surtout pas que vous n'avez jamais rencontré de problème avec votre logiciel préféré, je ne vous croirais pas !

Une inclusion simple peut être réalisée au moyen d'une simple balise `\includegraphics{}`.

Cette balise fait partie d'un package «graphicx» qu'il convient donc de charger en précisant `\usepackage{graphicx}` dans le préambule du source  $\LaTeX$ . Installé dans un environnement `\begin{center} \end{center}` l'instruction `\includegraphics{}` conduit au centrage du graphique.

Attention : il existe deux variantes de package aux noms très proches, `graphics` et `graphicx`. La deuxième autorise la rotation et la mise à l'échelle du fichier importé, c'est donc vers lui que va ma préférence. Pour aller plus loin,

- si vous souhaitez mettre en place un titre de graphique,
- si vous souhaitez que votre élément graphique puisse figurer dans une liste des graphiques (un peu comme une table des matières mais qui référence les éléments graphiques plutôt que les titres de sections),
- si vous souhaitez pouvoir y faire référence dans le texte (voir figure x page y),

alors vous devez passer par un environnement `figure`. Cet environnement fait partie des environnements «flottants» au même titre que l'environnement `table` qui sera vu plus loin. La notion d'environnement flottant est simple et complexe à la fois. Simple parce que  $\LaTeX$  se débrouille pour placer le flottant en fonction de règles de mise en page complexes et dont l'exposé sortirait du cadre de la présente étude. Complexe parce que vous allez passer, surtout au début, un temps certain à lutter contre ce satané fichu logiciel ... qui fait ce qu'il veut et vous ripe vos beaux dessins ou bon lui semble, généralement en haut de page !

Adoptez donc l'attitude du philosophe, ayez suffisamment de raison pour ne pas lutter contre ce que vous ne pouvez pas changer.  $\LaTeX$  vous fera de toutes façons une mise en page impeccable, à charge pour l'auteur du texte (vous..) de référencer correctement ses flottants et d'éviter surtout les formules du genre «la figure qui suit montre précisément le résultat de cette action..» parce que, précisément, vous n'avez pas la moindre idée de l'endroit où va être cette figure ! Le truc consiste donc à utiliser les références internes, point que nous examinerons à la section 6.6 page 20.

Pour tester la mise en place d'un élément graphique essayez la séquence de commandes qui suit :

```
\begin{figure}
\caption{Le même logo, cette fois redimensionné}
\center
\includegraphics[width=2cm]{jml.ps}
\end{figure}
```

Les différentes lignes sont relativement explicites. L'environnement `figure`, le fameux environnement qui va flotter, est borné par les balises `\begin{figure}` et `\end{figure}`. Une balise `\caption{}` permet de donner un titre au graphique. Elle peut être placée soit avant, soit après le graphique. Vous aurez deviné que ce choix n'est pas indifférent et que ce titre apparaîtra donc au dessus ou en dessous du graphique. Notez que tous les titres de figures seront précédés d'une mention «FIG x», x étant le numéro séquentiel attribué par  $\LaTeX$ . Le texte «FIG» peut lui-même être changé en attribuant une nouvelle valeur à la variable «figurename». Quand je vous disais que la lecture du code source de la classe article était passionnante ! La balise `\center` a une fonction évidente, la balise `\includegraphics` reçoit dans ce cas un argument `[width=2cm]` qui redimensionne le graphique en limitant sa largeur.

Le résultat de ces lignes est montré par la figure 3. Le redimensionnement d'une image peut

être obtenu en limitant sa largeur, mais aussi sa hauteur. Dans ce cas l'argument de la balise `includegraphics` devient par exemple `[height=2cm]`. La combinaison des deux, par exemple `[width=2cm, height=3cm]` peut donner les résultats curieux comme le montre la figure 4. L'association d'une dimension et d'un angle de rotation donne un résultat lui aussi intéressant comme le montre la figure 5 obtenue avec une syntaxe `\includegraphics[angle=45, width=2cm]{jml.ps}`. Au delà de ces possibilités qui permettent la mise en place d'élé-

FIG. 3 – Le même logo, cette fois redimensionné (réduit)




FIG. 4 – Encore le même logo réduit, cette fois étiré



FIG. 5 – Toujours le même logo réduit, cette fois tourné



ments graphiques de grande taille tels que courbes de résultats, photos, dessins etc. vous avez la possibilité d'insérer des éléments graphiques directement dans le texte comme ceci

 . Ce petit logo a simplement été mis en place avec la syntaxe `\includegraphics[width=2cm, angle=15]{jml4.ps}`.

## 6.6 Utiliser les références internes

$\LaTeX$  offre une possibilité très intéressante de mettre en place des références internes. Comme d'habitude ces références seront créées au vu des balises que vous ne manquerez pas de mettre en place de façon judicieuse dans votre code source :

**label{}** Met en place une ancre qui servira à repérer la partie du document à laquelle vous souhaitez faire référence. Les accolades contiennent un libellé quelconque qui servira à identifier l'ancrage.

**ref{}** Renvoie dans le texte le numéro de section, de figure, de table auquel vous faites référence. Cette référence prendra, par exemple, la forme : la figure~\ref{la figure} montre le résultat ainsi obtenu...

**pageref{}** Donne l'information du numéro de la page qui contient la référence visée. La forme du source pourrait dans ce cas être : la figure~\ref{la figure} page~\{pageref{}}].

Vous noterez la présence d'espaces insécables identifiés par un symbole tilde (~). L'intérêt de cet espace est d'éviter que L<sup>A</sup>T<sub>E</sub>X fasse une coupure inopportune de fin de ligne, ou pire de fin de page.

Vous ne rencontrerez certainement pas beaucoup de difficultés pour faire référence à des passages de texte. Pensez simplement que L<sup>A</sup>T<sub>E</sub>X a besoin de faire deux compilations successives pour mettre à jour correctement ses références, la première lui permet de repérer les ancres auxquelles vous faites références, la suivante reporte les bons numéros de section dans le texte source.

Pour les graphiques, tables et tous les flottants la prise de tête est au rendez-vous. Pour vous éviter le recours à l'aspirine je vous donne le truc : la balise label doit être collée immédiatement dans la ligne qui suit la balise caption. De cette façon la balise label renverra bien le numéro de figure, de tableau et non le numéro de la section dans laquelle se situe cet élément, d'autant plus qu'il flotte... Une autre possibilité, à mon avis encore plus sûre, consiste à mettre la balise label dans la balise caption (`\caption{mon titre \label{la balise correspondante}}`). De la sorte vous ne risquez pas, si vous décidez de déplacer la balise caption, de dissocier la balise label correspondante.

## 6.7 Les tableaux

Comme les listes, les tableaux constituent des éléments essentiels d'un exposé, surtout si celui-ci porte sur un sujet scientifique ou technique. L<sup>A</sup>T<sub>E</sub>X dispose ici aussi d'un environnement spécifique. La démarche est un peu parallèle à la mise en place d'un élément graphique, vous pouvez créer un simple tableau ou alors placer celui-ci dans un environnement flottant qui permettra d'y faire référence, de l'illustrer d'un titre etc.

Le code source suivant permet la création d'un tableau élémentaire.

```
\begin{tabular}{|c|c|}
\hline
Région & Résultats \\
\hline
\hline
Nord & 12 \\
\cline{2-2}
Sud & 13 \\
\hline
Total & 25 \\
\hline
\end{tabular}
```

Le résultat de ce code est montré par le tableau 1  
Commentaire des ces quelques lignes de code :

TAB. 1 – Un premier tableau

Région	Résultats
Nord	12
Sud	13
Total	25

**tabular** La balise `\begin{tabular}` est suivie d'une définition des colonnes du tableau donnée entre accolades. Les `|` forcent  $\LaTeX$  à générer des traits verticaux pour séparer les colonnes. Les caractères «`c`» servent à préciser que le contenu de la colonne sera centré. On peut placer le contenu à gauche avec «`l`» ou à droite avec «`r`».

**hline** Sert à tracer des lignes horizontales. Le doublement de cette instruction va doubler la ligne (étonnant non ?!). Notez à ce sujet que vous pouvez également doubler les traits verticaux si vous doublez le symbole pipe «`||`» dans la définition donnée en argument à la balise `\tabular{}`.

Le contenu des lignes du tableau est défini dans les lignes successives du tableau source. Les champs (colonnes) sont séparés par des `&`. Chaque ligne se termine par un code de saut de ligne `\\`.

**cline** Semblable à `\hline` mais trace des lignes horizontales sur une partie seulement du tableau. La commande `\cline` prends deux arguments situés dans l'accolade, le numéro de colonne sur laquelle commence le tracé et celui sur laquelle se termine celui-ci.

J'ai donné là quelques grandes lignes qui permettent de réaliser des tableaux élémentaires. Vous vous reporterez utilement à la documentation spécialisée qui traite du sujet et qui vous apprendra comment exploiter toutes les possibilités offertes par  $\LaTeX$  sur cette question. Tout ce qui a été vu plus haut pour les figures (section 6.5) reste valable, en particulier les informations relatives à la création de tables flottantes, délimitées dans ce cas par les balises `\begin{table}` et `\end{table}`, celles relatives à la mise en place de titres au moyen de la balise `\caption{}` etc. Je ne vous ferais pas l'affront, puisque vous suivez parfaitement, de vous exposer tout les avantages que vous pouvez éventuellement tirer de la modification du contenu de la variable `tablename`.

## 6.8 Les barres de révision

Il arrive souvent qu'un document soit révisé régulièrement. C'est la cas notamment dans l'industrie lorsque la politique qualité impose la révision annuelle des procédures, modes opératoires etc. La nouvelle mouture du document doit donc, pour être efficace, porter en marge une «barre de révision» qui indique précisément le ou les passages qui ont fait l'objet d'une modification. Ces barres de révision sont mises en place par une paire de balises `\begin{changebar}` `\end{changebar}`.

Changebar n'est pas reconnue comme une balise de base de  $\LaTeX$ . Il convient donc de charger la définition de celle-ci en incluant le package `changebar` dans le préambule du source. Pour ma part j'aime bien avoir la barre de révision à gauche. J'inclus donc l'instruction

```
\usepackage[outerbars]changebar dans mon préambule. L'argument outerbars a pour effet
```

de faire passer la barre à gauche du texte ce qui libère la marge de droite pour les notes en marge qui me servent à commenter la modification, voir section 6.9.

Attention ! Pour une raison que j'ignore xdvi n'affiche pas les barres de révision. Si vous souhaitez vérifier la modification ainsi introduite dans le document final vous devez, par exemple, faire une exportation du document compilé (format .dvi) en PostScript avec la commande `dvips -o fichier.ps fichier.dvi`. Cette exportation .ps pourra à son tour être visualisée avec `gv fichier.ps` qui lance ghostview un excellent visualisateur pour fichier PostScript ou .pdf.

L'emploi des barres de révision est valide si on compile son document avec  $\LaTeX$ . Par contre pdf<sub>l</sub>atex semble incapable de produire une barre de révision, que celle-ci soit à gauche ou à droite. J'ai passé des heures à chercher la solution, pour l'instant sans succès.

note en  
marge,  
illustra-  
tion de  
mon  
propos

Ajout le  
13 juin  
2004

## 6.9 les notes en marges et notes en pied de page

Réglons de suite la cas des notes en marges évoquées plus haut. Une note en marge se met en place au moyen d'une balise `\marginpar{}`. Le texte passé en argument sera affiché dans la marge droite du document. Vous observerez que la place est très réduite en marge. Ne perdez donc pas de vue que :

- Vous pouvez avoir avantage à utiliser une police réduite, par exemple une taille `footnotesize` qui donnera une présentation homogène avec les notes de pieds de page. La syntaxe devient dans ce cas `\marginpar{\footnotesize note en marge...}`
- Vous avez la possibilité de préciser les césures en insérant des séquences `\-` dans les mots les plus longs que  $\LaTeX$  peut être amené à couper. Il est évident que ces séquences devront être disposées à des endroits compatibles avec les règles du bon français. N'hésitez pas à en préciser plusieurs,  $\LaTeX$  ne tiendra compte que des césures qui l'intéressent et reproduira le reste du texte intact.
- Vous devez en tout état de cause limiter le contenu des notes en marge, celles-ci sont difficilement lisibles. Préférez éventuellement les notes de pied de page qui sont plus faciles à lire.

Les notes de pied de page ont l'avantage d'être imprimées sur toute la largeur du document ce qui les rend plus facile à lire que les notes en marge. Plusieurs possibilités s'offrent à vous pour créer une note de pied de page :

- Vous pouvez insérer en une seule opération la référence et le texte d'une note de bas de page au moyen d'une balise `\footnote{}`<sup>1</sup>. Les accolades contiennent dans ce cas le texte de la note. Vous avez, comme dans le cas des notes en marge, la possibilité de jouer sur la taille des caractères, par exemple en ajoutant une balise `\tiny` en début de texte, comme ceci<sup>2</sup>. Cette méthode de création s'applique dans le cas général d'un texte. Vous allez par contre au devant d'ennuis si vous essayez de mettre en place une note de pied de page depuis un environnement qui n'accepte pas la syntaxe de base, depuis un tableau par exemple.
- Dans le cas où la syntaxe de base ne donne pas l'effet escompté il faut procéder en deux temps, créer une référence par `\footnotemark` puis créer le contenu de la note par `\footnotetext{}`. Le texte de la note est contenu dans la deuxième balise,

---

<sup>1</sup>Voilà une note pour l'exemple

<sup>2</sup>attention toutefois à ce que l'ensemble reste lisible

la première ne servant qu'à mettre en place le repère. Attention dans ce cas à bien créer autant de balises de chaque type et à faire correspondre les références et les contenus en les plaçant dans le même ordre ! En procédant de la sorte  $\LaTeX$  prend en charge la numérotation séquentielle des notes, les notes créées selon la première ou celles créées par la deuxième méthode étant numérotées de façon cohérente.

- Si vous craignez un mélange des références et du contenu des notes par suite de mises à jours successives vous pouvez opter pour une syntaxe plus évoluée `\footnotemark[n]` `\footnotetext[n]` qui attribue d'office un numéro  $n$  à la note. Avantage, les mélanges sont impossibles. Inconvénient vous n'êtes plus cohérents avec la numérotation  $\LaTeX$ . Vous créez deux systèmes de numéros et  $\LaTeX$ , sans doute vexé de ne plus avoir l'initiative (on va dire comme ça hein !), ne réagira pas à la présence de deux notes portant le même numéro ! L'une des parades est de numéroté à partir de 100. On aurait pu imaginer numéroté au moyen des lettres de l'alphabet, ça ne fonctionne pas,  $\LaTeX$  ne sait traiter que des chiffres à cet endroit, croyez moi sur parole, j'ai fait l'essai.

## 6.10 Les tabulations et l'environnement `tabbing`

L'environnement `tabbing` permet la mise en place de tabulations. Celles-ci permettent d'aligner des informations un peu comme dans un tableau qui n'aurait pas de bordures. Cet environnement permet par ailleurs des effets assez amusants comme des effets de barré. Mais voyons plutôt ce que donne le code source suivant :

```
\begin{tabbing}
xxxxxxxxxxxxxxxxxxx\=xxxxxxxxxxxxxxxxxxx\= \kill
essai \>50 \>fin première ligne\\
test \>45 \>fin deuxième ligne\\
\>début à la tab. 2\\
\>\>et voilà la tab. 3!\\
\>\includegraphics*[width=2cm]{jml4}
\end{tabbing}
```

qui donne le résultat suivant :

essai	50	fin première ligne
test	45	fin deuxième ligne
	début à la tab. 2	
		et voilà la tab. 3!



Commentaires :

- L'environnement est borné par des balises `\begin{tabbing}` et `\end{tabbing}` selon une logique désormais familière.
- La ligne `xxxxx...` définit les tabulations. Les `x` ne sont là que pour réserver des emplacements de caractères entre les tabulations. Celles-ci sont balisées par des séquences `\=`. La ligne se termine par une balise `\kill` qui marque la fin de la définition sans l'afficher.
- Les différentes lignes qui vont constituer le `tabbing` sont listées ensuite. Des balises `\>` demandent à  $\LaTeX$  de sauter à la tabulation suivante. Chaque ligne de définition se termine par un saut de ligne `\\`.

– L’environnement `tabbing` permet même de mettre en place des éléments graphiques !  
 Le petit logo `jml` qui figure en dernière ligne est un petit fichier PostScript.  
 Cet environnement ouvre par ailleurs la possibilité de réaliser des effets assez intéressants.  
 Devinez par exemple ce qui va se produire si le texte qui occupe une colonne dépasse la  
 tabulation suivante !

## 7 $\LaTeX$ et les mathématiques

S’il est un domaine dans lequel  $\LaTeX$  excelle c’est bien celui des mathématiques. Alors que la rédaction de formules mathématiques est au mieux délicate, souvent impossible avec la plupart des traitements de textes et éditeurs courants qui fabriquent des fichiers monstrueux,  $\LaTeX$  offre des possibilités très vastes pour ce travail, possibilités que nous allons simplement effleurer dans les lignes qui suivent. Un exercice très simple consiste à mettre un chiffre en exposant, par exemple pour exprimer une unité «au carré». Cela se fait avec  $\$^2\$$ , le signe  $\$$  servant de balise pour marquer le début et la fin du mode mathématique. Ainsi une surface de 270 mètres carrés s’écrit  $270\text{ m}\$^2\$$  ce qui donne  $270\text{ m}^2$ . Plus complexe,  $\sqrt[3]{27} = 3$  est le fruit de la syntaxe  $\$\sqrt[3]{27}=3\$$ . Dans le même ordre d’idée vous obtenez  $\sum_{i=1}^{\infty} x_i = 0$  avec  $\$\sum_{i=1}^{\infty} x_i=0\$$ . Pour les cas plus complexes, notamment les systèmes de plusieurs équations  $\LaTeX$  offre un environnement spécifique `eqnarray`, ce qui peut se traduire par «réseau d’équation». Le système suivant :

$$\sqrt[3]{27} = 3 \tag{1}$$

$$\sum_{i=1}^{\infty} x_i = 0 \tag{2}$$

est obtenu avec le code  $\LaTeX$  suivant :

```
\begin{eqnarray}
\sqrt[3]{27}=3\
\sum_{i=1}^{\infty} x_i=0
\end{eqnarray}
```

Vous aurez noté immédiatement deux intérêts évidents à l’emploi de cette présentation :

- Les équations sont «calées» de telle sorte que les signes = soient alignés verticalement, c’est plus bô.
- Les équations sont numérotées ce qui permet d’apporter ensuite des explications spécifiques à chacune d’elle en la désignant de façon univoque et fiable.

## 8 Définir une commande $\LaTeX$ personnelle

Chaque année apporte son lot de nouveautés. Après le bogue de l’an 2000<sup>3</sup> nous venons d’avoir le passage à l’euro. Mais au fait, comment obtient-on le symbole  $\text{€}$  qui ressemble vaguement au symbole de l’euro ? La réponse est à la fois simple et compliquée. La

---

<sup>3</sup>Un bogue plutôt spécifique à Daniel PORTES dans son système d’exploitation qui s’appelle Fenêtres ou quelque chose comme ça...

## 8.1 Application à la création d'un symbole €.

---

séquence `\epsilon` devrait dessiner le  $\epsilon$  mais ça ne marche qu'en mode mathématiques. D'où la nécessité d'encadrer cette séquence avec des `$` comme ceci  `$\epsilon$`, pour forcer ce mode. Pas franchement le truc simple non ? Et si votre texte comporte une douzaine d'indications de prix alors vous êtes vraiment mal.. Heureusement  $\LaTeX$  offre la possibilité de définir simplement une nouvelle commande. Insérez simplement la ligne

```
\newcommand{\e}{\begin{LARGE}$\epsilon$\end{LARGE}}
```

dans l'entête de votre fichier. A partir de ce moment le symbole  $\epsilon$  s'obtient simplement en tapant `\e` !

Analyse de la syntaxe de cette ligne :

**newcommand** `\newcommand{}` prévient  $\LaTeX$  qu'il s'agit d'une nouvelle définition de commande. Il existe des variantes, `renewcommand` pour redéfinir une commande qui existait déjà, `providemcommand` qui définit une nouvelle commande seulement si elle n'existait pas précédemment. La commande `\newcommand` prend au moins deux arguments, elle peut en prendre plus, la chaîne qui va déclencher cette commande, ici `\e`, et ce qui va être fait par cette commande, en clair dans ce cas la chaîne de caractère qui va se substituer à `\e` à la compilation.

**epsilon** La commande `\epsilon` va générer, en mode mathématiques, le caractère  $\epsilon$ .

**LARGE** Les balises `\begin{LARGE}` et `\end{LARGE}` augmentent un peu la taille du caractère, en effet par défaut la commande  `$\epsilon$` va générer un caractère assez petit comme ceci :  $\epsilon$ .

## 8.1 Application à la création d'un symbole €.

Le paragraphe précédent, héritage d'une version antérieure de ce document, avait surtout pour mérite de montrer comment créer une commande personnelle. Il faut bien avouer que l'aspect du symbole Euro ainsi obtenu n'est pas franchement bon. Il existe de nombreuses méthodes pour obtenir un symbole correct, certaines font appel à des packages supplémentaires comme `eurosym`. Pour ma part je me contenterais ici de décrire une méthode impliquant l'emploi de l'alphabet grec. La première chose à faire va donc être de donner consigne à  $\LaTeX$  de charger, en plus du package `francais`, le package `greek`. Pour ce faire ajouter la ligne

```
\usepackage[greek, francais]{babel}
```

dans l'entête du document ou modifier la ligne initiale existante. A compter de ce moment la séquence `\textgreek{\euro}` va générer, à l'emplacement où elle aura été tapée dans le texte, un joli symbole €. Inconvénient : la syntaxe de la commande n'est pas franchement de celle qu'on a envie d'apprendre par cœur.

C'est à cet endroit qu'intervient la définition d'une commande personnelle puisque nous allons, à l'image de ce qui a été vu plus haut, définir une nouvelle commande `\E` qui va remplacer la séquence `\textgreek{\euro}`. Il suffit pour ce faire de créer une ligne `\newcommand{\E}{\textgreek{\euro}}` dans l'entête du document pour que la frappe de la séquence `\E` génère automatiquement le caractère €.


Ajout le  
16 juin  
2004

## 8.2 Application à la mise en place de logos dans le texte

Il est courant que les entreprises définissent des chartes graphiques pour leurs documents. Parmi les éléments de ces chartes on retrouve souvent l'obligation, lorsque c'est possible, de mettre le nom de l'entreprise sous forme de logo plutôt qu'en mode texte. Il est bien évidemment possible, à chaque fois que le logo doit apparaître, de taper :

`\includegraphics*[width=1.5cm]{mon_logo}` mais c'est loin d'être la meilleure solution. La bonne méthode, extrapolée directement de ce qui précède, consiste à :

- Ajouter une ligne d'entête  
`\newcommand{\logo}{\includegraphics*[width=1.5cm]{mon_logo} }`
- Mettre simplement en place une balise `\logo` à chaque emplacement où le logo doit apparaître

Avec mon logo personnel j'obtiens ainsi le logo  en suivant cette méthode.

## 9 Bibliographies et index

### 9.1 Gestion des bibliographies

Je n'aurais jamais imaginé écrire ce chapitre dans un document d'introduction à  $\LaTeX$  et puis un beau jour la question est tombée "Papa, je dois mettre en place une bibliographie dans mon document" suivie d'un nom de soft de gestion de biblio qui tourne, paraît-il sur un système d'exploitation à l'emblème des fenêtres! "Comment est-ce que je dois faire?". Exactement le genre de question qui, posée par votre fille, s'impose à vous et exige des recherches immédiates et efficaces.

#### 9.1.1 Le fichier .bib

On découvre ainsi assez rapidement qu'il existe des softs de gestion de bases de données bibliographiques, détail que j'ignorais encore il y a peu de temps. L'analyse de ces logiciels dépasse naturellement le cadre de la présente. Ces logiciels sont, très généralement, capables d'exporter leurs informations au format Bib $\TeX$  spécifique à  $\LaTeX$ . Il en découle un fichier d'extension .bib et qui comporte les informations souhaitées. Les lignes qui suivent constituent un extrait d'un de ces fichiers :

```
@Book{latexplp,  
  author = {Christian ROLLAND},  
  title = {LaTeX par la pratique},  
  publisher = {O'Reilly},  
  year = {1999}  
}  
  
@Book{megevand,  
  author = {Denis MEGEVAND},  
  title = {De la correspondance avec LaTeX 2e},  
  publisher = {Observatoire genève},  
  year = {2002}
```

}

Cet extrait comporte deux enregistrements, l'un relatif à un manuel très précieux édité par O'REILLY, le suivant à un document auquel je ferais référence dans la partie relative à la correspondance sous  $\LaTeX$ . Il n'est en fait absolument pas impératif d'employer un logiciel spécifique pour créer ce fichier, un simple éditeur comme VI ou EMACS peut faire l'affaire, avec cet avantage pour ce dernier qu'il reconnaît le format .bib et adapte donc sa barre de menu en conséquence. Vous noterez la composition finalement très simple de ces enregistrements, essentiellement constitué d'une balise d'identification ("latexplp" ou "megevand" dans mon exemple) qui servira, dans le texte, à établir le lien avec l'entrée de la table. Chacune de ces balises est suivie de champs dont la syntaxe ne nécessite pas de longues explications.

L'emploi de cette base de données bibliographiques dans le document  $\LaTeX$  est très simple :

- Ajouter une ligne `\usepackage{openbib}` dans l'entête du document à la suite des autres "usepackage".
- Ajouter une ligne `\bibliographystyleplain` juste avant `\begin{document}`.
- Ajouter une ligne `\bibliography{nom_du_fichier_bib}` à l'endroit où vous souhaitez voir apparaître la bibliographie. Nota : ne pas préciser l'extension .bib, celle-ci est implicite.
- Ajouter une référence `\cite{nom_balise}` dans le texte à l'endroit où vous souhaitez référencer le document.

### 9.1.2 Compilations spécifiques

Il suffit alors de compiler une première fois le document .tex puis de compiler le fichier .bib au moyen de Bib $\TeX$ , par exemple `bibtex mabiblio` si le fichier .bib s'appelle `mabiblio.bib`, après quoi une paire de compilation  $\LaTeX$  terminent de mettre en place la bibliographie dans le document final. Quelques points importants :

- La référence au fichier contenant la biblio se fait sans préciser son extension. Il faut bien taper `bibtex mabiblio` et non `bibtex mabiblio.bib`. L'extension est ajoutée par Bib $\TeX$ . Idem pour la mise en place de la biblio par `\bibliography{}`.
- La première compilation  $\LaTeX$  crée (ou régénère) le fichier .aux nécessaire (entre autre) à la compilation avec Bib $\TeX$ . Les compilations suivantes sont nécessaires à la mise en place de la biblio dans le document, à la régénération de la numérotation du plan etc.
- J'ai parlé ici essentiellement de compilation avec  $\LaTeX$  mais le fonctionnement est strictement le même avec `pdflatex`. L'avantage de `pdflatex` est, comme toujours, qu'il crée des liens internes au document et permet de sauter immédiatement à la bonne ligne du tableau des références simplement en cliquant sur le numéro de lien qui apparaît dans le texte

En procédant ainsi on voit apparaître, à l'endroit fixé par la balise `\bibliography{}` un tableau dont le titre est "Références" par défaut et qui consigne les enregistrements de la base de données qui sont référencés dans le document (et seulement ceux-ci). Pour tout renseignement relatif à ce sujet je vous renvoie à l'excellent manuel [3] édité par O'REILLY relatif à  $\LaTeX$ . On pourra se rendre compte de la nécessité des 4 compilations décrites ci-dessus en jetant un oeil au contenu du fichier .aux, par exemple avec `cat mon_fichier.aux | grep citation` ou `grep bibtex`. On verra alors apparaître explicitement les liens vers la bibliographie

mis en place dans le document. Bibtex va relire ces liens pour créer la table des références laquelle sera ensuite mise en place dans le document final par deux nouvelles compilations.

### 9.2 Mise en place d'un index

L'index est certainement l'un des aspects les plus importants du document imprimé. En effet, dans sa forme papier, le document ne permet plus les liens hypertexte ni les recherches de mots. Il en découle qu'un index bien construit constitue, avec la table des matières, une aide précieuse pour le lecteur.

La mise en place d'un index passe par plusieurs étapes, la modification du code source bien sûr, mais aussi la compilation selon une procédure spéciale.

#### 9.2.1 Modification code source

**Entête et fin de document** Deux modifications sont ici nécessaires, la première dans l'entête du document doit avertir  $\LaTeX$  qu'il aura à compiler un index. Cette information est donnée en ajoutant les lignes :

```
\usepackage{makeidx}
\makeindex
```

dans l'entête, puis à placer :

```
\printindex{}
```

à l'endroit où on souhaite voir figurer l'index, généralement en fin de document.

**Corps du document** La modification est là aussi assez simple et consiste, pour l'essentiel à insérer des balises `\index{NomIndex}` où `NomIndex` représente l'entrée souhaitée dans l'index. L'emplacement de cette balise est important dans la mesure où l'index fait référence à la page sur laquelle figure l'index. Un positionnement au mot près n'est donc pas nécessaire, pour ma part j'aime bien mettre les balises `\index{}` en fin de ligne de titre de section, sous-section etc.

#### 9.2.2 Compilation spécifique

Une fois le source modifié une première compilation va créer un nouveau fichier auxiliaire d'extension `.idx` comportant toutes les entrées d'index. Une seconde compilation obtenue par `makeindex` va créer un fichier de définition d'index qui sera finalement mis en place dans le document final par une nouvelle compilation  $\LaTeX$  ou `pdf $\LaTeX$` . Vous aurez noté au passage que la démarche est finalement assez proche de celle qui a été décrite plus haut pour la gestion des bibliographies.

#### 9.2.3 Utilisation avancée

Vous vous reporterez utilement au livre  $\LaTeX$  par la pratique [3] pour trouver les finesses ultimes qui permettent de changer la police de certaines entrées d'index qu'on souhaite distinguer ou pour créer des index à plusieurs "profondeurs". Je n'exposerais ici que le point

concernant le classement forcé d'entrée d'index qui, si elles n'étaient pas forcées, se classeraient mal. J'explique en prenant un exemple, celui de cet article. J'expose, entre autre, l'utilisation de `tableofcontents`.

J'ai donc été amené à placer un `\index{\verb+\+tableofcontents}` dans le texte à l'endroit ad-hoc pour créer un ancrage et une entrée d'index. En fait si j'avais simplement utilisé la syntaxe ci-dessus l'entrée `\tableofcontents` aurait été classée tout en début d'index, avant la lettre a, simplement parce que la barre oblique inverse est interprétée comme un caractère dont le classement alpha est placé plus haut que "a". Il faut donc forcer le classement de `\tableofcontents` à la lettre "t" avec une syntaxe légèrement différente : `\index{tableofcontents@\verb+\+tableofcontents}`. cette fois le symbole @ sépare deux champs, le second étant celui qui sert à créer l'index, le premier étant l'adresse employée pour le classement alphabétique des entrées.

### 9.3 Compilation automatique

Entre les compilations spécifiques rendues nécessaires par l'emploi d'une bibliographie et celles qui sont consécutives à la mise en place d'un index il devient assez pénible de recompiler à chaque correction. Pour automatiser tout cela j'ai simplement créé un petit fichier exécutable que j'ai nommé tout simplement `compil`, placé dans le dossier de travail et que j'appelle simplement en tapant `./compil`. Contenu :

```
pdflatex Latex5.tex
bibtex Latex5
makeindex Latex5
pdflatex Latex5.tex
```

Ce tout petit aménagement simplifie bien les choses et permet de faire tout le travail en une seule opération sans rien oublier. J'avoue que, à la fin finale lorsque je souhaite créer le document définitif (jusqu'à la prochaine révision), je relance au moins deux fois la commande pour être certain que toutes les références sont parfaitement à jour.

## 10 Création de documents au format .pdf exploitables sur d'autres systèmes d'exploitation

Tant que vous restez dans le monde LINUX les éléments qui précèdent vous permettent de créer des documents parfaitement lisibles et imprimables. Les difficultés (et les désillusions) vont commencer lorsque vous allez relire vos .pdf avec un logiciel tel qu'Acrobat Reader, un lecteur de .pdf gratuit courant sous Windows. J'ai rencontré essentiellement les difficultés suivantes :

- Les entêtes sont coupées.
- Le rendu de la police de caractère est absolument catastrophique, comme si l'imprimante s'était mise à postillonner.

Je ne prétends pas avoir la solution parfaite, voilà au moins comment je règle (ou je contourne) le problème.

## 10.1 Format de page

Lorsque je sais que le fichier doit être relu par Acrobat Reader, j'évite l'utilisation du package `\fullpage` qui conduit à mettre l'entête très haut dans la page, dans une zone qui n'est pas traitée correctement par Acrobat. J'avais attiré votre attention en son temps au chapitre 5.1 page 14 sur le fait que `\fullpage` ajustait une marge haute négative.

N'utilisant plus les valeurs prédéfinies dans `\fullpage` je suis donc contraint de régler un à un les paramètres de pagination. C'est simplement un peu plus long, mais ça donne d'excellents résultats. Voilà des réglages qui conviennent à une page A4 :

```
\textwidth=16cm % largeur du texte hors note en marge
\oddsidemargin=0pt % marge gauche réduite à 1 pouce
\textheight=22cm % longueur utile de la page
\topmargin=0pt % marge haute réduite à 1 pouce
\headsep=20pt % séparation 20 points entre entête et texte
\footskip=20pt % idem séparation pied de page
```

N'hésitez pas à ajuster éventuellement en fonction de votre sensibilité personnelle. Si vous n'envisagez pas d'utiliser les notes en marges vous pouvez augmenter la largeur du texte à 17 cm pas exemple.

Les commentaires sont suffisants à la compréhension des différentes lignes de réglages qui doivent, est-il besoin de le préciser, trouver leur place dans le préambule du document.

## 10.2 Police de caractères

La police de caractère Computer Modern passe mal lorsqu'elle est relue par Acrobat Reader. L'aspect "postillonné" du texte le rend difficilement lisible. Pour contourner cette difficulté il vaut mieux utiliser une police `pslatex`, prévue d'origine pour la création de documents au format PostScript et qui donne des résultats nettement meilleurs que la police `cm`. Pensez donc à mettre une ligne `\usepackage{pslatex}` dans le préambule de votre source  $\LaTeX$ . Malheureusement pour ceux qui aiment les polices sans serif la police `pslatex` ne fait pas partie de cette famille.

L'emploi de la police `phv` tel qu'il a été expliqué au paragraphe 6.3.3 page 17 règle de façon simple et efficace le souci de trouver une police sans sérif qui "passe" aussi bien en compilation  $\LaTeX$  qu'en compilation `pdflatex`.

Passage  
conservé  
d'une  
an-  
cienne  
version

Ajout  
20 juin  
2004

## 11 Compilation avec `pdflatex`

Nous abordons ici la création de document en suivant la branche de droite du schéma 1 page 5.

Obtenir un document au format `.pdf` nécessite, si on compile avec la commande `latex` (branche gauche examinée jusque là) de passer par l'une des étapes suivantes :

- Convertir le fichier `.dvi` obtenu après compilation en `.pdf` au moyen de la commande `dvipdf`.
- Passer par la création d'un fichier PostScript avec la commande `dvips`, puis convertir le fichier `.ps` ainsi obtenu en `.pdf` au moyen du petit programme `ps2pdf`.

Dans les deux cas l'exploitation du .pdf par xpdf ou Acrobat Reader produira un document dont le fonctionnement sera assez semblable à celui d'une version papier, donc sans liens, vous obligeant à feuilleter, à revenir à la table des matières etc.

Or, comme nous l'avons vu plus haut, le format .pdf permet d'utiliser les liens hyper texte, du moins si on utilise un afficheur qui sait les exploiter ce qui est le cas de Acrobat Reader et de xpdf. Attention gv (Ghostview) ne sait pas suivre ces liens, même s'il les affiche correctement en couleur. La mise en place de liens hyper texte passe donc par une compilation avec pdf $\LaTeX$ . Bien entendu il conviendra d'adapter légèrement le code source de telle sorte à charger les packages qui permettent la gestion des liens. Il conviendra en particulier de charger le package hyperref en plaçant la ligne `\usepackage{hyperref}` dans le préambule. Le compilateur pdf $\LaTeX$  présente une différence notable par rapport au compilateur  $\LaTeX$  dont il a été question jusque là puisqu'il produit directement un fichier .pdf sans passer par la création du format intermédiaire .dvi propre au compilateur  $\LaTeX$ . Par contre il présente l'inconvénient de ne pas savoir interpréter les graphismes au format .ps ! La compilation avec pdf $\LaTeX$  nécessite donc de remplacer les références à des fichiers graphiques en format PostScript par des références à un format reconnu par pdf $\LaTeX$  :

- .png, le format qui monte actuellement, plus compact que le format PostScript il est idéal pour recevoir des images converties de .gif.
- .jpg, là aussi plus compact que le PostScript si vous voulez stocker un fichier bitmap.
- .pdf, ben voyons, y'aurait plus manqué que ça, que pdf $\LaTeX$  ne sache pas exploiter les images au format .pdf.
- .tiff etc..

La première étape va donc consister à mettre à disposition du compilateur les fichiers graphiques dans les bons formats, le cas échéant en les convertissant.

Les conversions des fichiers PostScript (ou eps) dans l'un des formats listés ci-dessus peuvent se faire au moyen de logiciels spécifiques tels que epstopdf. Ce programme effectue aussi bien les conversions de fichiers .eps (encapsuled Postscript) que de fichiers .ps ordinaires. The Gimp représente une autre possibilité très efficace pour effectuer des conversions entre différents formats de fichiers graphiques. N'hésitez pas à explorer ce programme graphique extrêmement puissant et qui vous rendra bien des services.

L'inconvénient majeur de la compilation par pdf $\LaTeX$  est donc la nécessité de modifier profondément le code source et de remplacer toutes les extensions .ps par des extensions qui soient lisibles par pdf $\LaTeX$ , typiquement .png. Et bien entendu cette modification rend le fichier source impropre à une compilation directe par  $\LaTeX$  ! Ne vous impatientez pas, dans le chapitre suivant tout va s'arranger.

### 11.1 Modification élémentaire du source en vue de la compilation par pdf $\LaTeX$

Modifier toutes les lignes qui font une inclusion graphique est la première solution qui vient à l'esprit, c'est aussi la plus mauvaise. Heureusement il y a une solution plus efficace. Celle-ci consiste simplement à préciser une fois pour toutes quels sont les types de fichiers graphiques à prendre en compte lors de la compilation. Cette précision est apportée par la directive `\DeclareGraphicsExtensions{.png}` placée dans le préambule et qui déclare,

dans la syntaxe donnée ici en exemple, que tous les fichiers graphiques seront, par défaut, des types .png. Il suffit alors de supprimer toutes les extensions de noms de fichiers. Le compilateur recomposera les noms de fichiers au moyen de la racine donnée dans les  $\includegraphics$  et de l'extension par défaut précisée dans le préambule. Pour passer d'un système de compilation à l'autre il suffit d'adapter cette unique ligne. Admettez que  $\LaTeX$  est génial non ? Il suffit maintenant, après avoir créé les deux versions de fichier graphiques, de changer la terminaison par défaut pour adapter le source au compilateur que l'on souhaite utiliser.

Notez que l'indication du type d'extension n'est pas obligatoirement unique et que vous pouvez préciser une liste d'extensions qui seront testées les unes après les autres. Le syntaxe  $\DeclareGraphicsExtensions\{.ps, .eps\}$  par exemple demandera au compilateur de chercher d'abord un fichier se terminant par .ps, puis, si ce fichier n'est pas trouvé, de chercher une extension .eps.

### 11.2 Source compilable par $\LaTeX$ et $pdflatex$

$\LaTeX$  permet de créer des fichiers sources qui soient compilables aussi bien par la commande  $\LaTeX$  classique que par  $pdflatex$ . L'astuce consiste à ajouter, dans l'entête du fichier source, une routine qui ajuste le source en fonction du compilateur utilisé. Je vous donne si dessous la syntaxe que j'utilise, trouvée sur Internet. Cette syntaxe fait appel au langage  $\TeX$ . Elle met en place un test identifiant le compilateur puis charge les packages en fonction du résultat de ce test.

```
\newif\ifpdf
  \ifx\pdfoutput\undefined
    \pdffalse
  \else
    \pdfoutput=1
    \pdftrue
  \fi

\ifpdf
  \usepackage[pdfTeX,colorlinks=true,urlcolor=blue,pdfstartview=FitH]{hyperref}
  \pdfcompresslevel=9
  \usepackage[pdfTeX]{graphicx}
  \DeclareGraphicsExtensions{.png, .jpg}
\else
  \usepackage{graphicx}
  \DeclareGraphicsExtensions{.eps, .ps}
\fi
```

L'analyse de la syntaxe détaillée de cette ligne sortirait du cadre de la présente étude (et dépasserait mes compétences à l'heure où j'écris ces lignes) sans apporter d'éléments déterminants au regard de l'objectif poursuivi.

## 11.3 Aléas lors des compilations alternées avec LaTeX et pdflatex

J'ai exposé à différents chapitres de ce document que ma préférence allait à la compilation par pdflatex dès lors qu'il s'agit de créer un document de type article. Je veille toutefois toujours à ce que mes sources restent compilables avec LaTeX. La compilation LaTeX faite après une compilation pdfLaTeX ne pose pas de problème (sous réserve que la syntaxe soit correcte). Par contre, dans le sens opposé, la première compilation pdfLaTeX abouti assez systématiquement à des messages d'erreur, du moins à partir du moment où le document fait appel à des références internes. Ceux-ci semblent causés par des différences dans la composition des fichiers annexes comme .aux ou .toc. Deux solutions sont alors offertes :

- Soit passer en force en conservant le doigt sur la touche "ENTER", jusqu'à la fin des messages d'erreur et faire immédiatement une seconde compilation !
- Soit effacer au préalable les fichiers auxiliaires pour repartir sur une base propre.

Vous aurez certainement deviné que j'utilise la première solution, celle du passage en force ! C'est simple et moyennant une seconde compilation dans la foulée je n'ai jamais constaté le moindre problème.

## 12 Synthèse

En conclusion de ces chapitres consacrés à la rédaction de documents de type rapport je vous livre ci-dessous les constituants de l'entête et les premières lignes du code source qui a donné naissance au document que vous lisez en ce moment. Pour ma part j'utilise cette entête comme point de départ pour les documents que je rédige dans le cadre de mes activités.

Je vous livre ce code, adaptez-le à vos besoins, j'espère qu'il vous rendra service, au moins pour démarrer dans cette technique particulière de création de documents.

```
% INTRODUCTION
\documentclass[a4paper, french, 12pt]{article}
\usepackage[greek, francais]{babel} % Typographie française, date en format F etc..
\usepackage[latin1]{inputenc} % accents standard clavier
\usepackage[T1]{fontenc} % accents dans le dvi
\usepackage{pslatex}
\usepackage[francais, verbose]{layout}
\usepackage{openbib} % bibtex, gestion des bibliographies
\usepackage{fancyhdr} % entêtes et pieds de pages
\usepackage{makeidx}
makeindex
% SELECTION SELON COMPILATEUR
\newif\ifpdf
  \ifx\pdfoutput\undefined
    \pdffalse
  \else
    \pdfoutput=1
```

---

```

\pdftrue
\fi
\ifpdf
  \usepackage[pdftex, colorlinks=true]{hyperref}
  \pdfcompresslevel=9
  \usepackage{graphicx}
  \DeclareGraphicsExtensions{.png, .jpg}
\else
  \usepackage{graphicx}
  \DeclareGraphicsExtensions{.eps, .ps}
\fi

% MISE EN PAGE
\textwidth=16cm
\oddsidemargin=0pt
\textheight=22cm
\topmargin=0pt
\headsep=20pt
\footskip=20pt
\rhead{}
\lfoot{\tiny\textsl{Page \thepage}}
\cfoot{\includegraphics*[width=2cm]{jml4}}
\rfoot{\tiny\textsl{\today}}
\renewcommand{\footrulewidth}{0.4pt}
\pagestyle{fancy}
\usepackage[outerbars]{changebar}

% SYMBOLE EURO
\newcommand{\e}{\begin{LARGE}$\epsilon$\end{LARGE}}
\newfont{\cmdh}{cmdunh10}
\newcommand{\E}{\textgreek{\euro}}

% POLICE PAR DEFAUT
\renewcommand{\familydefault}{phv}
\selectfont

\bibliographystyle{plain}

\begin{document}

% CESURES
\hyphenation{sub-para-graph} % définition des césures souhaitées

\selectlanguage{français}
\title{Introduction à \LaTeX}
\author{Jean-Marc LICHTLE}

```

---

```
\maketitle
\tableofcontents{}
\thispagestyle{fancy}
\renewcommand{\abstractname}{Note de mise à jour}
\newcommand{\logo}{\includegraphics*[width=1.5cm]{jml4} }

* corps du document *

\printindex{}
\bibliography{Latex5}

end{document}
```

## 13 Faire du courrier

### 13.1 Compilation, L<sup>A</sup>T<sub>E</sub>X ou pdfL<sup>A</sup>T<sub>E</sub>X ?

Un courrier est, par définition, un document destiné à être imprimé. Il en découle que les liens hypertexte autorisés par pdfL<sup>A</sup>T<sub>E</sub>X ne présentent plus guère d'intérêt. De même que la possibilité de les lire avec un type de browser présent sur toutes les plateformes imaginables. Bien que la compilation par pdfL<sup>A</sup>T<sub>E</sub>X reste possible je pense que l'emploi classique de la commande L<sup>A</sup>T<sub>E</sub>X s'impose dans une pareille application.

### 13.2 La classe lettre de L'Observatoire de Genève

J'ai choisi de présenter la classe lettre de l'Observatoire de Genève plutôt que la classe «officielle» letter parce qu'elle offre non seulement un format très simple d'utilisation pour rédiger des courriers classiques mais qu'elle permet également de composer des fax très bien construits. De plus cette classe se sert d'un fichier de paramètres par défaut ce qui permet de ne mettre que le minimum d'informations dans le fichier du courrier et donc d'aboutir au résultat final avec un minimum de frappe.

### 13.3 Récupérer les fichiers strictement nécessaires

Le strict minimum consiste à récupérer le fichier lettre.cls et le fichier default.ins (un modèle pour le fichier de paramètres par défauts). Ces fichiers sont accessibles dans le dossier ftp <ftp://obsftp.unige.ch/pub/tex/macros/lettre/inputs/>.

### 13.4 Une documentation complète

Je vous recommande par ailleurs de télécharger le fichier lettre\_2.345.tar.gz ou la dernière variante à jour disponible au jour ou vous téléchargerez. Ce fichier est accessible sur le même serveur dans </pub/tex/macros/>. Le décompactage de ce tar.gz va créer un sous-répertoire </doc/> contenant une documentation très intéressante. Notez que le décompactage

va créer également un sous-répertoire `/inputs/` dans lequel vous trouverez une nouvelle copie des fichiers indispensables qui définissent la classe lettre et les réglages par défaut. Vous trouverez dans le sous répertoire `/doc/`, entre autres petites merveilles, un fichier `let-doc.ps` qui constitue un manuel très clair et très bien documenté construit par M. DENIS MEGEVAND de l'Observatoire de Genève.

### 13.5 Installation

L'installation la plus élémentaire consiste à copier simplement les deux fichiers `lettre.cls` et `default.ins` dans le répertoire dans lequel vous travaillez. Un demi ton plus loin dans la sophistication : copier `lettre.cls` dans `/usr/share/texmf/tex/latex/base/`, sous-répertoire dans lequel se trouve déjà le fichier de définition de la classe `letter` standard de  $\text{\LaTeX}$ . De la sorte  $\text{\LaTeX}$  trouvera immédiatement le fichier de classe quelque soit le sous répertoire depuis lequel la compilation est lancée. Le fichier ayant par défaut des droits positionnée à `rwx` pour tout le monde, tous les utilisateurs de votre PC devraient avoir accès à cette nouvelle classe sans restriction. Pensez éventuellement à modifier les droits d'accès en `r-` avec le `chmod 444` qui va bien.

Le fichier `default.ins` par contre est personnel. C'est la raison pour laquelle il doit obligatoirement être présent dans le sous répertoire à partir duquel vous lancez la compilation. Avantage : vous pouvez parfaitement organiser votre travail dans différents sous-répertoires, courriers privés, courriers professionnels etc, chaque sous répertoire contenant un modèle de fichier `default.ins` différent et adapté au contexte.

### 13.6 Paramétrage du fichier `default.ins`

La syntaxe de ce fichier est très simple, jetez-y un oeil, vous y trouverez immédiatement les 4 lignes qui définissent l'adresse, le téléphone, le fax et le lieu. Mettez-y les valeurs qui vous concernent et vous aurez terminé cette part du travail. N'hésitez pas, lorsque vous serez un peu familiarisé avec l'utilisation de la classe lettre à y mettre d'autres informations personnelles et qui ne changent pas d'un courrier à l'autre, la signature par exemple. La syntaxe de cette ligne est la même que celle qui aurait été employée dans le fichier définissant la lettre. Le fichier `default.ins` est en fait appelé par un `\input{default.ins}` situé vers la fin du fichier de classe. Cet appel « complète » le fichier source de votre courrier en lui ajoutant, pour les besoins de la compilation, les informations contenues dans `default.ins`.

### 13.7 Premier essai

Plutôt que de faire des grands discours je vous propose de passer immédiatement à un premier essai

```
\documentclass[a4paper]{lettre}
\usepackage[français]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\begin{document}
\begin{letter}{\Large Revue \LaTeX\ en folie\\rue de la pompe\\75000 PARIS}
```

## 13.7 Premier essai

---

```
\fontfamily{cmss}
\selectfont
\conc{Article Introduction à \LaTeX}
\signature{JML}
\opening{Cher ami}
Je vous prie de bien vouloir trouver en annexe etc...
\closing{Salutations}
\end{letter}
\end{document}
```

La figure montre le résultat, à la fois très simple et très propre, de la compilation de ce premier courrier. Quelques commentaires relatifs à ces quelques lignes :

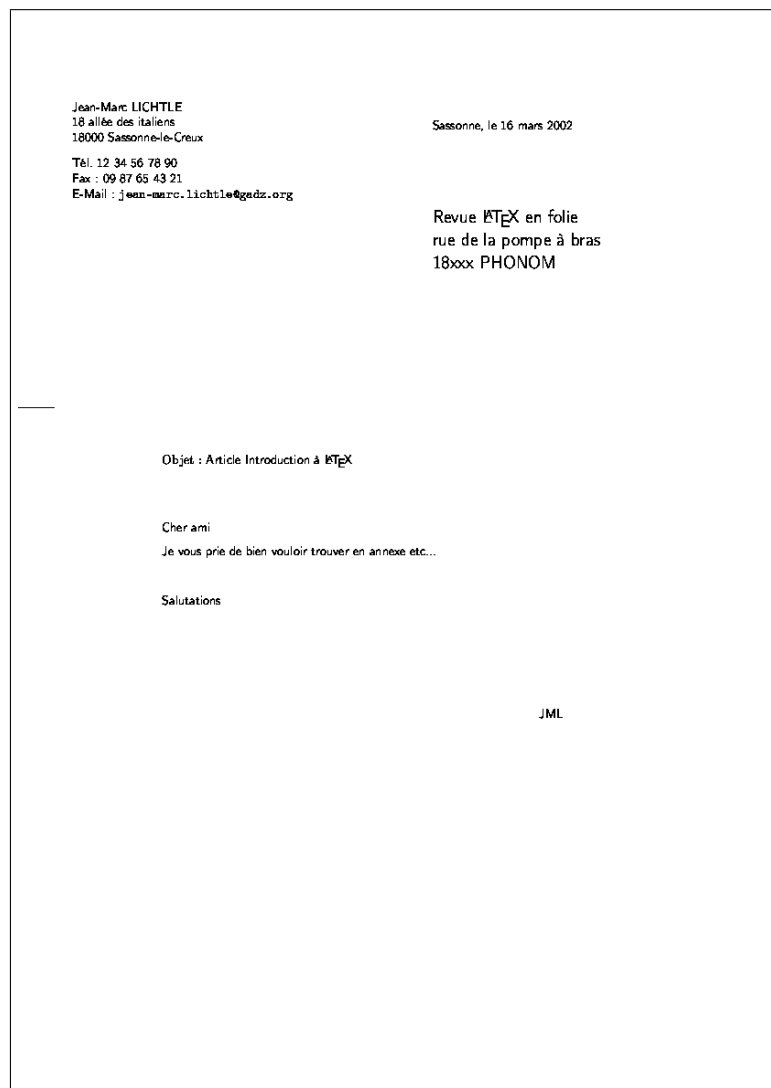


FIG. 6 – Notre premier courrier

- La première ligne réalise l'appel de la classe lettre en lui passant le format A4 comme paramètre.
- Les trois lignes suivantes ne devraient plus avoir de secrets pour vous.
- Un fichier source peut contenir plusieurs courriers. Il comporte donc deux balises `\begin{letter}`, l'une pour le document, l'autre pour le courrier (et autant de balises `\end{letter}`). Si vous souhaitez préparer un deuxième courrier dans le même source ouvrez simplement un deuxième environnement `letter` à la suite du premier. Vous aurez noté au passage l'orthographe à utiliser. La classe lettre est un joyeux mélange de français et d'anglais. L'appel à la classe se fait avec l'orthographe française (`lettre.cls`) alors que la balise d'ouverture du courrier utilise l'orthographe anglaise `\begin{letter}`.
- La balise `\begin{letter}` prends un argument essentiel, l'adresse du destinataire de la lettre. Vous noterez la possibilité d'utiliser les balises de modification de texte. J'ai utilisé ici pour l'exemple une balise `\Large` qui augmente un peu la taille de la police utilisée pour l'adresse. Les balises `\` permettent la mise en place rapide de sauts de ligne qui vont définir les champs de l'adresse, nom du destinataire, rue, ville etc.
- Les deux premières lignes du courrier changent simplement la police de caractère. Pour ma part j'aime les polices sans serif, je ne vais donc pas me priver de les utiliser chaque fois que c'est possible. On pourrait imaginer mettre les commandes de sélection de police entre la balise d'ouverture de document et la balise d'ouverture de lettre, ça ne donne pas le résultat escompté.
- Après la balise qui ouvre la lettre commence la définition des champs de celle-ci. Parmi ces champs `\conc{}` permet de préciser le titre de la lettre qui suivra la mention «Objet : » mise en place automatiquement par  $\text{\LaTeX}$ , `\signature{}` permet de consigner la définition exacte de la signature qui apparaîtra en fin de courrier. Nous verrons plus loin la liste des champs disponibles. Le champ `\opening{}` est obligatoire. A la lecture de ce champ  $\text{\LaTeX}$  va déclencher la mise en place de l'entête de la lettre. Sans `\opening{}` pas d'entête de lettre !
- Le texte de la lettre lui-même accepte à peu près tous les formats imaginables, vous pouvez y définir des tableaux, y placer des figures etc. Attention toutefois, quelques limitations risquent de vous rappeler que vous n'êtes pas dans un environnement article. Les notions de section ne semblent pas être fonctionnelles. Les figures peuvent être mises en place avec `\includegraphics{}` mais l'environnement `\figure{}` semble incompatible avec la classe lettre.
- La balise `\closing` est, comme la balise `\opening` obligatoire. Sans elle pas de pied de lettre !

Comme il avait été évoqué plus haut la classe lettre vous ouvre la possibilité de placer plusieurs courriers dans le même fichier source. Vous pouvez ainsi constituer des fichiers sources qui sont de véritables dossiers contenant tous les courriers que vous avez créé pour une affaire. Faites l'essai en recopiant simplement le bloc défini par `\begin{letter}` - `\end{letter}` à la suite de lui-même en faisant une modification dans la copie. Après compilation vous aurez bien deux courriers différents dans votre fichier `.dvi`. J'en entends d'ici qui font le remarque «c'est bien beau mais comment séparer ces courriers les uns des autres si on ne souhaite pas imprimer le lot complet ?». Je vous donne donc le truc que j'emploie quand je compile avec  $\text{\LaTeX}$ , pas obligatoirement ce qui se fait de mieux mais ça marche :

- Je commence par exporter mon fichier `.dvi` en format PostScript au moyen d'une com-

- mande simple, dvips -o lettre.ps lettre.dvi
- Je feuillette les courriers affichés avec gv lettre.ps
- J'imprime celui que j'ai choisi avec l'option «Print marked».
- Si le courrier tient sur deux pages je suis bien entendu obligé de faire l'opération en deux temps. J'en profite d'ailleurs pour lancer les impressions en commençant par la dernière page ce qui conduit à un document correctement classé directement en sortie d'imprimante.

La compilation avec pdflatex est un peu plus simple dans la mesure où le browser (afficheur) de fichier .pdf, qu'il s'agisse de xpdf ou Acrobat Reader, peut imprimer le fichier page par page ou par séquence de numéro de page selon les réglages de l'utilisateur.

## 13.8 Les champs disponibles pour la zone entête de lettre

Il me semble important de dresser une liste de synthèse des champs disponibles dans la classe lettre.cls. La définition de ces champs est à placer entre la balise `\begin{letter}` et la balise `\opening{}` ou entre la balise `\closing{}` et la balise `\end{}` selon que les informations concernent l'entête ou le pied de lettre. Attention à la prise de tête ! Curieusement la signature ne fait pas partie du pied de lettre, n'essayez donc pas de placer la balise `\signature{}` après `\closing{}`, ça ne marche pas, la signature fait partie des informations d'entête, au moins pour le source du document  $\LaTeX$ .

**\address{}** Ce champ devrait être défini par défaut dans default.ins. Vous pouvez toutefois remplacer cette définition par défaut en précisant un nouveau contenu, par exemple `\address{Frédéric PHONOM\ rue de l'exance\ 51000 CHALONS SUR MARNE}`. Cette nouvelle définition écrase l'ancienne.

**\location{}** Permet de préciser le nom du service expéditeur ou celui de l'expéditeur, par exemple si le champ adresse ne contient que l'adresse de l'entreprise ou de l'association.

**\telephone{}** Défini par défaut dans default.ins, la valeur par défaut est remplaçable par `\telephone{nouveau num.}`. Attention à l'orthographe, telephone ne prend pas d'accents !

**\notelephone** Permet de supprimer l'indication du numéro de téléphone, même s'il est défini par défaut.

**\fax{}** Idem telephone, donc défini par défaut.

**\nofax** Idem notelephone

**\email{}** Le numéro d'email qui apparaîtra dans l'entête

**\lieu{}** Défini par défaut, correspond au lieu qui sera mentionné dans le libellé de la date, par exemple Sassonne-le-Creux, le 16 juillet 1954.

**\nolieu {}** Pour supprimer l'indication du lieu

**\date{}** Permet de forcer la date. Par défaut  $\LaTeX$  utilise la date de la compilation. `\date{16 juillet 1954}` vous permet de forcer la date à une valeur de votre choix. Notez que la valeur forcée sera reproduite à l'identique, si vous passez une valeur `{16/07/54}` c'est bien cette présentation (affreuse) que vous aurez dans votre entête de lettre !

- \nodate** Permet de supprimer l'insertion de la date. Attention à la cohérence de ce que vous faites, si vous employez `\nodate` pensez à mettre aussi `\nolieu` sinon vous risquez de retrouver le nom du bled suspendu et isolé dans l'entête de votre document.
- \vref{}** Précise les références du courrier auquel on fait réponse. Notez que, curieusement, deux orthographes semblent possible, `vref` et `Vref`. Cela ne semble pas être le cas d'autres balises telles que `location` par exemple.
- \nref{}** Nos références. A noter que les deux champs `vref` et `nref` sont associés. La mise en place d'une valeur pour l'un des deux provoque automatiquement l'affichage des deux, même si le deuxième est vide.
- \telex{}** Sans accent! Affiche le numéro de télex dans l'entête après les champs `vref` et `nref`, sur la même ligne. Si ces derniers sont vides ils apparaissent vides, `\telex{}` en force l'affichage.
- \ccp{}** Pour indiquer le numéro de compte courant postal. Comme pour de télex l'indication d'un ccp provoquera l'affichage des champs `vref` et `nref` même vides.
- \conc{}** Permet d'indiquer l'objet du courrier. Le texte passé en paramètre apparaîtra après un libellé «Objet :»

### 13.9 Les champs disponibles pour la zone pied de lettre

Sauf indications contraires ces champs peuvent être définis entre la balise `\closing{}` et la balise `\end{letter}`.

- \name{}** La documentation annonce que ce paramètre est obligatoire, je constate toutefois qu'on peut parfaitement faire du courrier sans préciser ce champ. Notez toutefois que vous obtiendrez une erreur de compilation si vous LaTeXez un courrier pour lequel ni la signature, ni le nom ne sont positionnés. La compilation d'un source qui précise le nom mais ne donne pas de valeur à la signature abouti au remplacement de la signature par la valeur contenue dans le nom.
- \signature{}** Met en place la signature en bas de courrier, du côté droit. Curieusement il ne semble pas possible de préciser cette signature, ni les suivantes d'ailleurs, après la balise `\closing{}`. Les signatures doivent donc être définies entre `\begin{letter}` et `\opening{}` ou alors dans le texte du courrier, en tout état de cause avant la balise `\closing{}`
- \secondsignature{}** S'il est nécessaire d'avoir une seconde signature
- \thirdsignature{}** Dans le cas exceptionnel ou il vous en faudrait une troisième !
- \cc{}** Pour noter les destinataires des copies. Ces destinataires apparaîtrons après un libellé «C.c.».
- \encl{}** Signale les pièces éventuellement jointes au courrier et qui seront listées après un libellé «P.j.».
- \menc{}** Pour rappeler les annexes mentionnées dans le texte. Le libellé devient dans ce cas «Annexe(s) mentionnée(s)».
- \ps{}** Post-scriptum. Tout à fait entre nous, j'ai transpiré un moment avec cette balise, le temps de me rendre compte qu'elle prend deux arguments et non un seul comme

toutes celles qui précèdent. Le premier argument précise le libellé, par exemple «PS : », le suivant donne le post-scriptum. Une des syntaxes possible serait donc : `\ps{PS : }{J'espère lire prochainement votre article sur lea-linux.org}`. C'est très curieux puisque tous les autres libellés sont définis dans la classe en fonction de la langue sélectionnée alors que celui-ci doit être explicitement précisé à chaque fois.

La figure 7 montre ce que devient notre courrier après avoir utilisé une partie des balises complémentaires : La majorité des libellés qui apparaissent dans le courrier final sont mo-

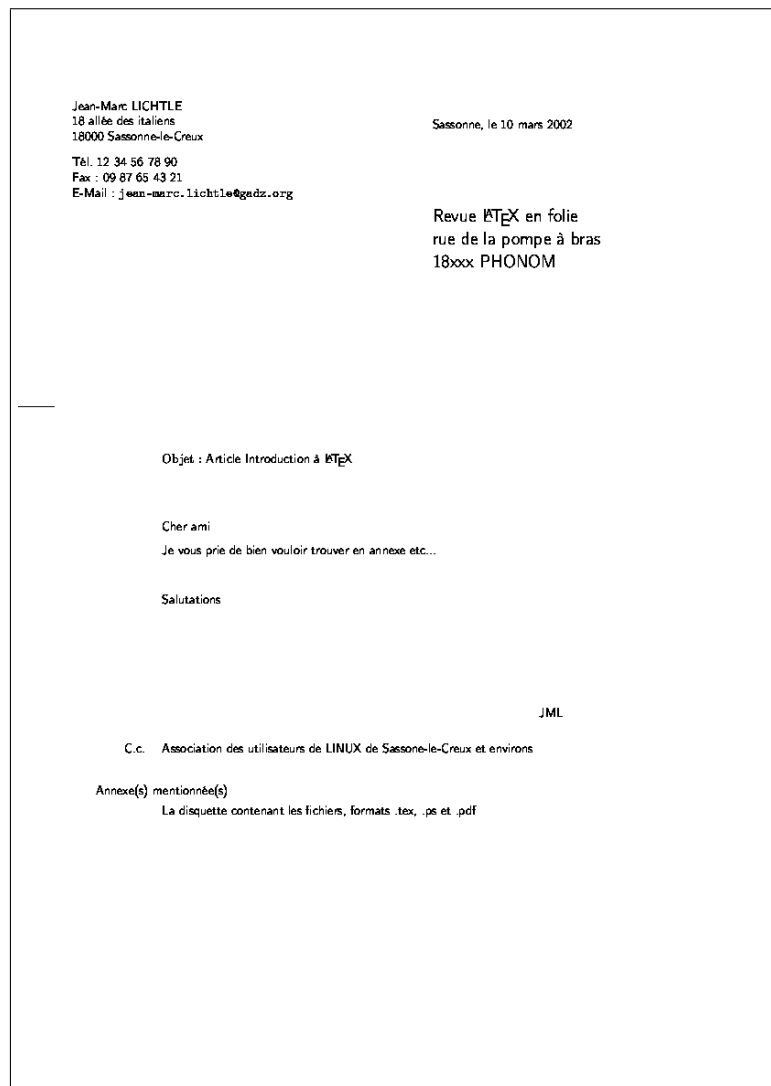


FIG. 7 – Notre courrier s'étoffe

difiabiles à souhait par l'utilisateur. Exemple : le libellé «E-Mail» introduit l'adresse email dans l'entête du courrier. Parce que vous souhaitez «franciser» votre présentation vous souhaitez que le libellé devienne «Mél : » pour message électronique. La modification peut se faire très simplement en insérant une ligne `\renewcommand{\emaillabelname}{Mél :~}`

---

entre `\begin{letter}` et `\opening{}`.

Les libellés ainsi modifiables pour le format lettre sont les suivants :

**tellabelname** Valeur par défaut Tél.~

**faxlabelname** Valeur par défaut Fax :~

**emaillabelname** Valeur par défaut E-Mail :~

**concname** Valeur par défaut Objet :~

**ccname** Valeur par défaut C.c.~

**enclname** Valeur par défaut P.j.~

**mentionname** Valeur par défaut Annexe(s) mentionnée(s)~

**vrefname** Valeur par défaut V./réf. ou plus exactement V./réf.~

**nrefname** Valeur par défaut N./réf. ou plus exactement N./réf.~

Vous noterez que les syntaxes sont intéressantes. Elles se terminent systématiquement par un tilde ~ qui définit un espace insécable. Il serait donc judicieux de penser à terminer les nouveaux libellés de la même façon.

Nota : toutes ces définitions par défaut sont visibles dans le fichier `lettre.cls`, un peu après la ligne 1000 pour la version que j'utilise. Vous n'aurez donc aucun mal à les retrouver si vous souhaitez faire une modification permanente. Deux petits conseils toutefois :

- Faites une copie de sauvegarde de `lettre.cls` avant de mettre les mains dans le cambouis ;
- Méfiez-vous, la classe `lettre.cls` est construite pour deux formes de français, le français de l'hexagone et le romand, forme helvétique de la langue de Molière. Modifiez donc la bonne version faute de quoi vous risquez de ne jamais voire l'effet de celle-ci !

Remarquez que la classe `lettre` peut également prendre en charge :

- l'allemand,
- l'anglais (UK),
- l'américain.

Je n'imagine toutefois pas qu'on puisse confondre les libellés dans ces langues avec l'une des formes du français.

## 14 Utilisation de la classe `lettre` pour la rédaction de téléfax

J'avais signalé en introduction de la section 13.2 que l'un des avantages de la classe `lettre` de l'observatoire de Genève est la possibilité de créer très simplement des fax.

### 14.1 Premier essai

Comme dans le cas de la lettre je vous propose de mettre directement les mains dans le cambouis en essayant le code source suivant :

```
\documentclass[a4paper]{lettre}
\usepackage[français]{babel}
```

```
\usepackage[latin1]{inputenc} % accents standard clavier
\usepackage[T1]{fontenc} % accents dans le dvi
\begin{document}
\begin{telefax}{01 23 45 68 79}{\Large Revue \LaTeX\ en folie\\rue de la pompe à bras\\18}
\fontfamily{cmss}
\selectfont
\name{JML}
\conc{Article Introduction à \LaTeX}
\opening{Cher ami}
Je vous prie de bien vouloir trouver en annexe etc...
\closing{Salutations}
\end{telefax}
\end{document}
```

Le résultat obtenu par la compilation de ce code est présenté dans la figure 8. Les champs sont les mêmes que pour la lettre, il n'y a donc pas lieu de faire de commentaires particuliers sur le code source. Vous aurez remarqué que dans le cas de ce fax je n'ai pas donné de valeur au champ signature et que c'est donc le champ name qui en tient lieu.

Remarques importantes :

- Le document comporte un champ nombre de pages. Pour que le nombre de pages soit transféré dans ce champ il faut réaliser deux compilations successives. A la première  $\LaTeX$  calcule le nombre de pages et le range dans un fichier accessoire. Au deuxième passage le compilateur transfère cette valeur vers le fichier de sortie.
- Il peut se produire (en fait c'est très fréquent) que vous mettiez des annexes avec votre fax. Dans ce cas il faut indiquer à  $\LaTeX$  le nombre de pages ajoutées afin qu'il en tienne compte dans le calcul du nombre total de pages du fax. Ceci s'obtient en précisant `\addpages{x}`, x étant le nombre d'annexes. Cette ligne doit être ajoutée dans l'entête du fax, en clair entre la balise `\begin{telefax}` et la balise `\opening{}`.
- A la différence du format lettre qui permettait de ranger plusieurs courriers dans le même fichier  $\LaTeX$  le format telefax n'autorise qu'un seul fax par fichier.

## 14.2 Les champs disponibles pour la zone entête de telefax

La majorité des champs sont strictement communs avec ceux déjà exposés lors de l'examen du format lettre. Je n'en mentionne ici la liste que pour que l'exposé soit complet.

**\address{}** Voir lettre

**\telephone{}** idem

**\fax{}** idem

**\email{}** idem

**\telex{}** idem

**À : et Télécopie** Ces deux paramètres sont passés en argument lors de l'appel

`\begin{telefax}{Télécopie}{Á}`. La syntaxe de l'adresse est semblable à ce qui a été exposé pour les lettres, vous avez la possibilité de commander les sauts de ligne au moyen de `\\` ou celle d'augmenter la taille de la police de caractères.

## 14.2 Les champs disponibles pour la zone entête de telefax

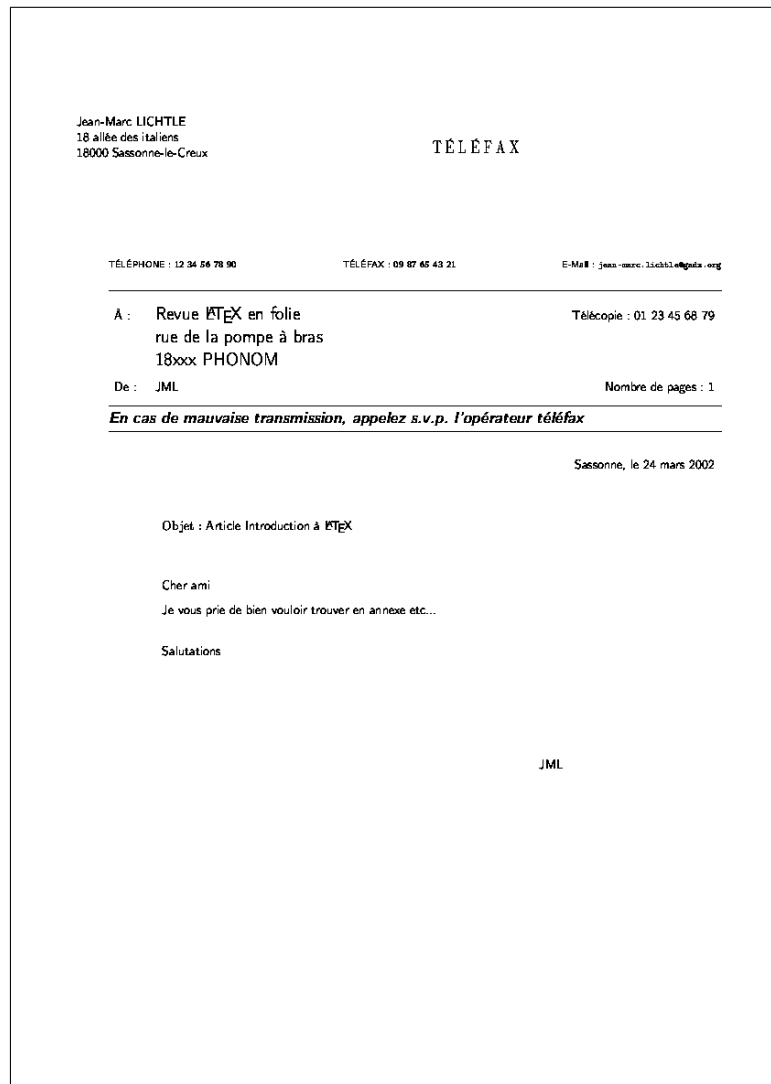


FIG. 8 – Notre premier téléfax

**De :** Il s'agit là du contenu de la variable `\location` déjà examinée lors de l'étude du format courrier.

**\addpages{}** et **nombre de page** Point examiné plus haut sous 14.1.

**\lieu{}** et **\nolieu{}** Voir lettre

**\date{}** et **\nodate{}** idem

**\conc{}** idem

**\opening{}** Marque, comme dans le cas d'une lettre la fin de l'entête, provoque la compilation de celle-ci et ouvre le texte du fax.

Le format telefax contient des chaînes de caractères fixes définies par le format telles de l'indication TÉLÉFAX ou le texte `En cas de mauvaise transmission.....` Dans les deux

cas je trouve les formules assez peu heureuses<sup>4</sup>. A quoi bon préciser au destinataire qu'il s'agit d'un téléfax alors qu'il va s'en rendre compte à 2 mètres rien qu'à voir la mauvaise définition du document. Dans le même ordre d'idée quel est l'opérateur à appeler si la transmission est incomplète ? A quel numéro ? Pour ma part je transforme souvent tout ceci en remplaçant la mention TÉLÉFAX par quelque chose de plus précis, du genre "Demande de prix", "Demande de travaux" etc. Le texte du bandeau est souvent remplacé par quelque chose qui ressemble à "En cas de réception incomplète merci de m'appeler au numéro qui figure dans l'entête". Vous l'aurez deviné, ces transformations se font en agissant sur des variables qui dans ce cas s'appellent `telefaxstring` pour le titre et `faxwarning` pour le bandeau. La figure 9 montre ce qui est obtenu après avoir ajouté les lignes suivantes :

```
\renewcommand{\telefaxstring}{\LARGE MESSAGE URGENT!}
\renewcommand{\faxwarning}{En cas de réception incomplète merci de ...}
dans l'entête du fax c'est à dire entre la balise \begin{telefax} et la balise \opening{}
```

## 14.3 Les champs disponibles pour la zone pied de telefax

`\closing{}` Voir lettre

`\signature{}` idem. Notez que, comme dans le format lettre vous avez possibilité de définir trois signatures, `\signature{}`, `\secondsignature{}` et `\thirdsignature{}`.

`\cc{}` Voir lettre

`\encl{}` idem

`\mencl{}` idem

`\ps{}` Je fais le pari que `\ps` se comporte de façon aussi curieuse dans les fax que dans les courriers. Je le vérifierais dès que le besoin s'en fera sentir..

## 15 Utilisation avancée

### 15.1 Inclusion d'un fichier d'entête

L'entête relative à un courrier est relativement simple. Par contre vous n'aurez pas été sans remarquer que l'entête pour un article est une partie relativement importante du document. Imaginons que vous ayez à taper un document qui ne comporterait que deux pages, la taille de l'entête est presque celle du document ! Bien sûr le copier-coller facilite énormément le travail en permettant de récupérer une entête qui donne satisfaction pour la coller dans un document à créer. Inconvénient de cette méthode, toutes les améliorations futures que vous pourrez trouver pour l'entête ne s'appliqueront qu'aux nouveaux fichiers. Une amélioration "rétroactive" va nécessiter la modification d'autant de fichiers que vous avez de documents. Ne perdez pas de vue par ailleurs que vous risquez, parce que la mémoire est loin d'être infaillible, de "remettre en circuit" une entête qui n'est pas tout à fait à jour, ce qui est vraiment dommage. Le remède ici consiste à créer un fichier d'entête, nommé par exemple "entete.tex" et de l'inclure au tout début de chacun des fichiers documents que vous

Tout le chapitre 15 a été refondu le 20 juin 2004

---

<sup>4</sup>Entendons nous bien, il ne s'agit pas d'une critique, simplement d'un avis personnel

## 15.1 Inclusion d'un fichier d'entête

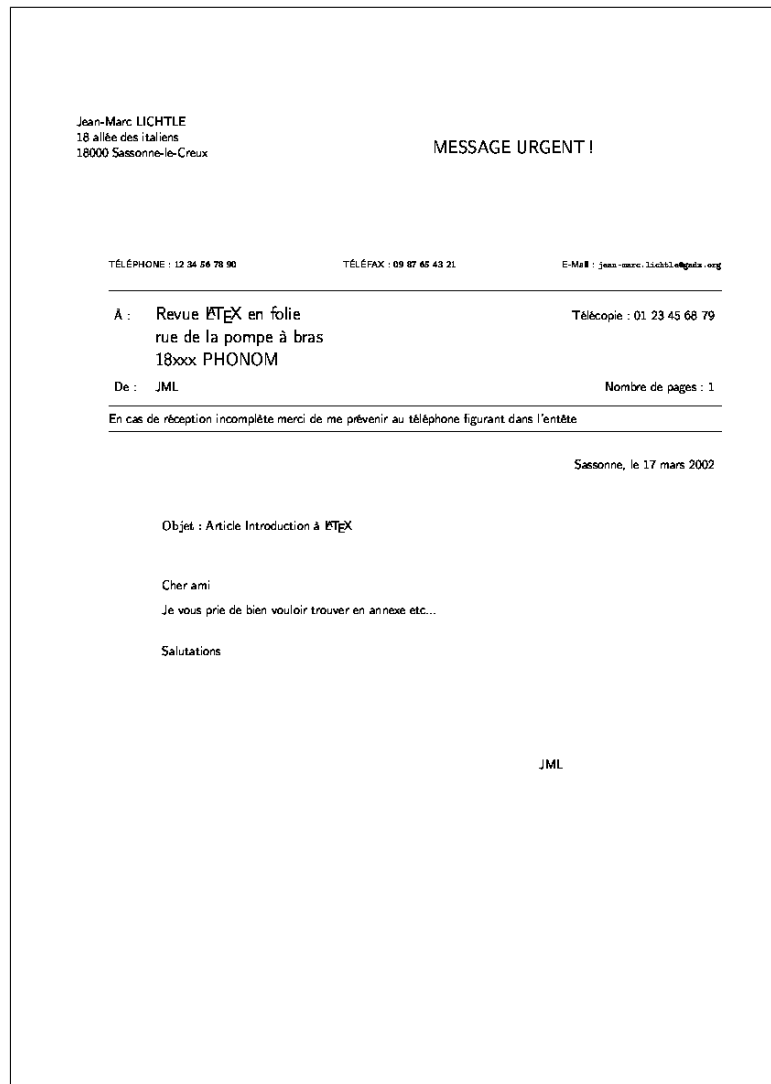


FIG. 9 – Une version plus évoluée

voulez créer. Cette inclusion se fait simplement au moyen de la commande  $\text{\LaTeX} \backslash\text{input}\{\}$ . Les crochets doivent contenir le nom du fichier. Si l'extension du nom de fichier n'est pas précisée celle-ci sera mise en place par  $\text{\LaTeX}$  qui considère par défaut que le fichier a une extension `.tex`.

Une autre méthode consiste à inclure l'entête au moyen de la commande  $\backslash\text{include}\{\}$ . Cette dernière nécessite l'emploi du nom de fichier sans extension. Note : dans ce cas l'extension est obligatoirement `.tex`.

## 15.2 Inclusion de parties de documents stockés dans des fichiers externes

Une autre utilisation de `input` et `include` est la construction d'un document par "briques". Vous pouvez parfaitement créer un document maître qui ne contiendra que peu de choses mais qui fera appel, via `input` ou `include`, à des chapitres stockés dans des fichiers séparés. La différence entre les deux est de deux ordres :

**Format** La commande `input` insère le document ajouté exactement comme s'il avait été tapé dans le corps du texte maître alors que le texte mis en place par `include` provoque deux sauts de page, l'un avant l'inclusion, l'autre juste après.

**Emploi** L'utilisation des deux commandes n'est pas tout à fait la même. `input` réalise simplement un copier coller inconditionnel. `include` peut être associé à une directive `\includeonly{}` mise en place dans l'entête et qui donne la liste des "includes" à valider. En d'autres termes vous pouvez construire un document constitué d'une dizaine de chapitres mais souhaiter, parce que vous ne travaillez pas sur l'intégralité du document à un instant donné ou parce que la taille totale du document ne permet plus de le compiler rapidement, ne compiler que les chapitres récemment modifiés.

## 15.3 Définir une nouvelle macro commande paramétrable

À l'évidence la syntaxe de  $\text{\LaTeX}$  est loin d'être tout à fait agréable à employer. Imaginons par exemple que vous ayez en chantier un document contenant un grand nombre d'illustrations. Pour chacune d'elle vous allez devoir définir le centrage, la taille, le nom du fichier source, le titre, le label etc. chacun de ces éléments faisant appel à sa commande propre. Il serait (il est) beaucoup plus simple de construire une fois pour toutes une commande qui prend par exemple quatre arguments, la largeur de l'image, le nom du fichier, le titre souhaité et le label associé. Nous avons commencé à lever un coin du voile relatifs aux commandes personnalisées dans le chapitre 8 page 25. Nous avons à cet endroit créé des macros qui génèrent le symbole € ou qui permettaient l'inclusion de logos dans le corps du texte. Toutes ces macros étaient définies une fois pour toutes et ne prenaient aucun argument. L'exemple suivant illustre la méthode qui permet de définir une nouvelle commande, cette fois avec prise en charges d'une liste d'arguments :

```
\newcommand{\grafic}[4]
{
\begin{figure}[h!]
\begin{center}
\fbbox{\includegraphics[width=#1]{#2}}
\caption{#3}
\label{#4}
\end{center}
\end{figure}
}
```

ainsi que le résultat obtenu avec la syntaxe suivante :

```
\grafic{5cm}{jml4}{joli essai}{je}
```



FIG. 10 – joli essai

Explications :

- La définition commence par `newcommand` qui indique que nous allons créer une nouvelle commande qui n'existait pas encore. Pour remplacer une commande existante utiliser `renewcommand`. Les deux premiers arguments sont le nouveau nom de cette commande ainsi que le nombre d'arguments.
- Le troisième argument, le plus volumineux, va contenir les commandes individuelles qui vont constituer la nouvelle macro, les arguments étant remplacés dans cette définition par des `#n` ou `n` est un numéro séquentiel commençant à 1 et désignant les arguments dans leur ordre d'apparition dans la future ligne de commande.
- La syntaxe de la ligne de commande est on ne peut plus simple, le nom de la nouvelle commande suivi d'une succession de crochets contenant les arguments dans leur ordre d'appel.

Moyennant la création d'une macro, qui ne prend guère plus de place que la collection de lignes qu'elle remplace, nous venons de nous simplifier énormément la vie. Définie en début de document, ou même dans l'entête de celui-ci cette macro pourra s'appliquer à l'ensemble des éléments graphiques qui seront inclus dans le document. Admettez que du point de vue simplification il est assez difficile de faire mieux. Autre avantage, l'emploi de cette macro permet une présentation unifiée pour l'ensemble du document. Ainsi définie tous les éléments graphiques seront soulignés d'un filet noir, le titre sera toujours en dessous du graphique etc. Qu'il vous vienne l'envie de faire passer les titres au dessus des images, de les mettre en gras et de supprimer le filet servant de cadre et une simple modification de la macro transformera l'aspect de l'intégralité du document.

Vous imaginez bien qu'on peut compliquer à l'infini. Garder toutefois présent à l'esprit que les macros même très simples, peuvent conduire à des économies de frappe assez intéressantes. Un petit exemple tout simple, la ligne :

```
\newcommand{\BI}{\begin{itemize}} si je l'avais employé dès le début de la création de ce document m'aurait permis d'éviter le frappe, une trentaine de fois, de la séquence \begin{itemize}!
```

## 16 Remerciements

Je pourrais distinguer dans une liste impeccable les sources nombreuses que j'ai eu à ma disposition pour me familiariser avec  $\LaTeX$ .  $\LaTeX$  est d'ailleurs très bien construit et permet de dresser de superbes bibliographies. Je m'en garderais toutefois conseillant au lecteur de chercher par soi-même, chez son libraire et sur Internet les sources d'informations qui correspondent à ses besoins. Votre libraire aura certainement peu de livres sur  $\LaTeX$  en rayon, mais en cherchant bien dans ses catalogues il doit trouver, mais si, chez O'REILLY par exemple (Elle est facile celle-là, O'REILLY est LE spécialiste de ce genre de littérature...).

---

Quant à Internet essayez Latex formation ou Latex aide dans votre moteur de recherche, mais attention à l'avalanche !

La table de références qui figure à la fin de ce document est une simple illustration de mon propos sur la mise en place de bibliographies. J'y ai simplement consigné les contributions qui m'ont été les plus précieuses.

Je tiens toutefois à distinguer dans mon propos M. Denis MÉGEVAND [2] de l'Observatoire de Genève qui a rédigé un manuel très complet (il pèse tout de même plus de 70 pages) sur l'utilisation de la classe lettre. Mon exposé, assurément incomplet, laissera obligatoirement le lecteur dans l'embarras sur tel ou tel point non abordé (ou mal expliqué). Qu'il se reporte donc à cette brochure dont la lecture lui apportera tous les renseignements qu'il peut souhaiter. L'adresse à laquelle cette documentation peut être récupérée est précisée au paragraphe 13.4 page 36. Merci donc à M. Denis MÉGEVAND pour la qualité de son travail.

Une mention particulière aussi pour M. Benjamin BAYARD pour son document "Joli manuel pour L<sup>A</sup>T<sub>E</sub>X" [1] qu'on trouvera facilement sur le site de l'ESIEE (mettez esiee latex dans votre moteur de recherche et vrouoummm). Attention, là on est parti pour presque 150 pages, réfléchissez avant d'imprimer !

Je conclurais par un remerciement tout particulier à l'équipe de Lea-linux.org qui fait un gros travail pour assurer la maintenance du site <http://www.lea-linux.org>, et en faire une vraie mine d'informations. Un grand merci à JCC avec qui je mets au point mes papiers

## 17 L'auteur

Jean-Marc LICHTLE, ingénieur Arts et Métiers promotion Chalons 1973-1977. Ouais, comme le temps a passé ! Je me souviens encore avoir travaillé de nuit<sup>5</sup> sur la seule calculatrice programmable qu'on avait à notre disposition à cette époque ! Trois promotions dans l'école et une seule machine qu'on se partageait avec certains professeurs ! Les premières HP et TI programmables sont arrivées peu après... Quand à Linux ou à L<sup>A</sup>T<sub>E</sub>X nous n'imaginions même pas que ça pourrait exister un jour !.

Dans la série "du même auteur" (et sur <http://www.lea-linux.org>)

- Installation d'un serveur APACHE, PHP et MySQL. Partant d'un Linux qui fonctionne et sans connaissances particulières de l'installation d'un serveur web, le fil conducteur qui permet d'aboutir à un serveur fonctionnel.
- Sécurisation du serveur APACHE, PHP et MySQL. La suite de l'installation, on traite ici de la partie sécurisation, mise en place de contrôles d'accès etc.
- RPM, la révision de l'article de JCC sur la commande rpm, révision réalisée à la demande de l'auteur.
- GNUPlot, LyX et L<sup>A</sup>T<sub>E</sub>X, une petite étude sans prétention tirée directement de la pratique sur le terrain et qui explique comment faire des tracés élémentaires, et d'autres qui le sont moins, avec GNUPlot et les intégrer dans un document L<sup>A</sup>T<sub>E</sub>X.

---

<sup>5</sup>Je me souviens des aller-retour entre la salle et le foyer, on faisait déjà du time sharing, fallait tout de même pas qu'on soit tous au foyer en même temps !

---

## 18 Copyleft

On va faire simple, ce texte est mis à disposition de tous ceux qui voudront en faire usage. Il peut être découpé, copié, trituré librement et par tout moyen adéquat, copié-collé, ciseaux, destructeur de document, photocopieur etc. Je remercie simplement le copiste, s'il conserve la plus grande partie de mon document original, d'y adjoindre une mention de copyleft au moins aussi peu contraignante que la présente. Et si en plus mon nom, ou plus simplement mes initiales, pouvaient y subsister alors ce serait parfait. N'hésitez pas à me faire part de vos observations en m'adressant un mail à l'adresse [jean-marc.lichtle@gadz.org](mailto:jean-marc.lichtle@gadz.org).

jml

## Index

- `%`, 8
- `\\`, 16
  
- Acrobat Reader, 7
- Article (type), 8
- Automatisation, 30
  
- Babel, 9
- Balise, 8, 10
- Barre de révision, 22
- Bibliographie, 27
  
- Commentaires, 8
- Compilation, 6
- Copyright, 51
- Courrier, 36
  
- `\date`, 11
- Documentclass, 9
- DVI, 5
  
- Entête, 34
- Espace, 10
- Euro, 26
  
- `\familydefault`, 17
- Fancy, 14
- FreeBSD, 7
- Fullpage, 14
- fullpage, 31
  
- Graphiques, 18
  
- HTML, 5
  
- Images, 18
- `\includegraphics`, 19
- Inclusions, 48
- `\index`, 29
- IRIX, 7
  
- jml, 50
  
- KDE, 7
  
- Langue, 9
  
- `\layout`, 13
- Letter (type), 9
- Lettre (classe), 36
- Liste, 18
- Logo, 27
  
- Macros, 48
- `\makeindex`, 29
- `\maketitle`, 11
- Mathématiques, 25
  
- Notes pied de page, 23
- Numérotation, 12
  
- PDF, 4
- PDF format, 30
- PdfLaTeX, 6
- Pdflatex, 31
- .png, 33
- Police, 16
- Polices de caractères, 31
- PostScript, 4
- .ps, 33
  
- Références, 20
- Résumé, 15
- Révision, 3
- Report (type), 8
  
- `\section`, 12
- `\selectfont`, 17
  
- Tableaux, 21
- `\tableofcontents`, 11
- Tabulations, 24
- `\textbf`, 10

## Références

- [1] Benjamin BAYARD. *Joli manuel pour LaTeX*. ESIEE.
- [2] Denis MEGEVAND. *De la correspondance avec LaTeX 2e*. Observatoire genève, 2002.
- [3] Christian ROLLAND. *LaTeX par la pratique*. O'Reilly, 1999.