




Les bases L^AT_EX 2_ε par l'exemple.

Présentation des notions de base pour utiliser le formateur de
texte L^AT_EX.

Nicolas Sayer
sayer@imagine.fr

23 juin 1998 — Version 1.0)



Ce document a été réalisé uniquement dans le but de diffuser de l'information. L'auteur ne serait être tenu responsable de tout "problème" résultant de l'utilisation d'informations contenues dans ce document. Il a été réalisé pendant mon temps libre sans aucun but commercial. Vous pouvez diffuser et modifier ce document comme bon vous semble, il appartient au domaine public. Je souhaiterais cependant être averti par courrier électronique des erreurs et des modifications afin de pouvoir éventuellement les inclure dans une nouvelle version. Je serais également ravi que les lecteurs de ce document me donnent leur avis, ne serait-ce que pour savoir si il y a quelque qu'intérêt à le faire évoluer. La dernière version de ce document réside actuellement sur: <http://www.imagine.fr/~sayer>

Table des matières

1	Introduction.	3
1.1	Qui est visé?	3
1.2	L ^A T _E X, qu'est ce que c'est?	4
2	Du code à l'imprimante.	5
3	Structure d'un document.	6
3.1	Le préambule.	6
3.2	Le corps.	6
4	Exemple pratique.	7
4.1	Document exemple.	7
4.2	Code source.	9
4.3	Analyse.	12
5	Organiser votre document.	13
5.1	Découpage du document.	13
5.2	La table des matières.	14
5.3	Page de garde.	14
5.4	Références dans un document.	15
5.5	Index, glossaire et bibliographie.	15
6	Jouer avec les caractères.	16
6.1	Changer la police.	16
6.2	Taille des caractères.	17
6.3	Caractères spéciaux.	17
6.3.1	Accents et caractères français.	17
6.3.2	Les autres.	18
7	Formatage du contenu.	19
7.1	Les notes.	19
7.2	Les listes.	19
7.3	Les tableaux.	20
7.4	Les figures et les tables.	21
7.5	Les césures.	22
7.6	Modifier le type de justification.	22
7.7	Quelques derniers tags.	23
8	Commandes du préambule.	23
8.1	Les classes de documents.	23
8.2	Les options.	24
8.3	Entête et numérotation de page	24
8.4	Les packages.	25
8.5	Packages intéressants.	25
8.6	Pour nous: Les Français!	25

<i>TABLE DES MATIÈRES</i>	3
9 whereis L^AT_EX & Outils additionnelles.	26
9.1 Où trouver L ^A T _E X 2 _ε ?	26
9.2 Outils sympas.	26
10 Aller encore plus loin?	27
11 Conclusion.	27

Pour la petite histoire, j'ai découvert \LaTeX alors que je venais de passer une nuit à taper un rapport sous *Word* pour l'école. Trois heures au total, une pour taper le texte, et deux pour se battre et essayer d'avoir un résultat présentable. Tout le monde peut me traiter de fou, mais rien au monde ne me fera quitter \LaTeX pour un truc WYSIWYLTG (*What You See Is What You'd Like To Get*).

1 Introduction.

Ce document est une *zième* présentation de l'outil de formatage de texte \LaTeX . Le but de celui-ci est de vous donner juste assez d'information pour que vous puissiez générer vos premiers documents sous \LaTeX . Je ne présenterais donc que les commandes que j'utilise régulièrement pour la production de mes propres documents. D'autres documents que je citerais en conclusion pourront vous donner une vue beaucoup plus complète du formateur de texte. Pour pouvoir suivre ce tutoriel il vous faut impérativement posséder une version de $\text{\LaTeX} 2_{\epsilon}$ installée (pas $\text{\LaTeX} 2.09$) et savoir utiliser un éditeur de texte de base (vi, edit, emacs, etc.). \LaTeX existe sous beaucoup de plateformes différentes: UNIX, Linux, OS/2, Windows... à vous de voir!

Après une courte description du principe de formatage de texte, je décrirai la structure d'un document \LaTeX de base. S'en suivra un document complet en exemple. C'est dans une troisième partie que je donnerai toutes les commandes nécessaires, ainsi que des exemples, pour que vous puissiez commencer à rédiger vos premières documentations sous \LaTeX .

Ce document sera sûrement appelé à évoluer dans les mois qui viennent pour inclure d'autres commandes intéressantes et corriger d'éventuelles erreurs. Vous pourrez trouver tous les fichiers sources de ce document sur mon site (URL indiqué en première page).

1.1 Qui est visé?

\LaTeX n'est pas et n'a jamais prétendu être l'outil idéal pour remplacer un traitement de texte comme Word ou WordPerfect. Il vise uniquement la production de document complet comme des rapports, thèses, livres, etc. Bien que je l'utilise régulièrement pour tout mes documents, je ne suis quand même pas complètement drogué. Quand il s'agit d'une lettre à la banque, ou une invitation de mariage, un traitement de texte graphique est beaucoup plus simple à utiliser. Mais dès que vos document atteignent une certaine taille et une certaine complexité, \LaTeX reste imbattable.

Si vous êtes un(e) jeune secrétaire souhaitant rajouter une ligne "*compétences \LaTeX* " sur votre CV, vous faite fausse route. En revanche, si vous êtes un étudiant en passe d'écrire votre premier rapport, un mathématicien, ou un

informaticien¹ vous pouvez avoir intérêt à goûter aux joies des beaux documents et du code \LaTeX .

1.2 \LaTeX , qu'est ce que c'est ?

Scientifiquement, \LaTeX est une sur-couche de macros édifée sur le langage \TeX . \TeX est un langage de description de page assez barbare créé originellement pour le monde de l'édition. Concrètement c'est un langage permettant de créer des documents divers: rapports, livres, lettres, articles. . . Sous \LaTeX vous créez votre document sous la forme d'un fichier source, contenant le texte du document et des marqueurs (tags) donnant les informations de formatage. Une fois le fichier terminé il ne reste plus qu'à le compiler². Le fichier source ressemble un peu au langage HTML utilisé sur le web. Sous *Word* il suffirait de sélectionner un **mot** et de cliquer sur le bouton "gras" pour qu'il soit mis en gras, alors que sous \LaTeX il faudra encadrer le mot d'accolades et le précéder d'une commande comme suit: `\textbf{mot}`. Cette dernière commande va sûrement rembarasser un certain nombre de personne, mais les tags de formatage (gras, italique, emphasis, souligné. . .), et ceux d'organisation du document (chapitre, sous-partie, préambule) sont les seuls à connaître pour pouvoir créer une documentation très évoluée.

\LaTeX a plus ou moins été créé en 1985³ par Leslie Lamport. Il a connu depuis un immense succès dans le monde scientifique et informatique en particulier. Il s'est enrichi de beaucoup de modules depuis, mais reste toujours attaché à \TeX . Une des raisons de son succès est sa puissance en formatage et ses possibilités en ce qui concerne les formules mathématiques. Il reste, aujourd'hui encore, le *must* dans le domaine des mathématiques.

\LaTeX s'occupe lui-même du formatage du document (justification, césure, génération de la table des matières, numérotation, etc.), l'utilisateur n'a donc plus qu'à s'occuper du contenu, et plus vraiment du contenant.

The Pros & Cons of \LaTeX . J'ai réalisé un micro-sondage chez des utilisateurs et des anti- \LaTeX , voici les résultats:

Pour:

- Aucun formatage à réaliser;
- La numérotation des pages, chapitres, parties, figures est automatique;
- Fichiers compatibles entre différentes plateformes;
- Documents agréables à regarder;
- \LaTeX est intégralement gratuit.

1. Je veux dire **Informaticien**, pas disciple de la secte du gourou Bile ;))
 2. le mot semble barbare, mais je vous rassure, ça n'a rien de méchant.
 3. Le HTML, c'est 1994.

Contre:

- Langage codifié, à apprendre...
- Les documents se ressemblent tous (moi je trouve que c'est un avantage).

Je viens de dire que \LaTeX est gratuit, je m'explique: les compilateurs d'origine \TeX avaient été écrits par des universitaires et étaient dans le domaine public. La sur-couche de macros \LaTeX , de Leslie Lamport est aussi gratuite. Vous trouvez donc une multitude de compilateurs intégralement exempts de droits d'auteurs. Certains éditeurs ont réalisé des versions payantes de \LaTeX , celles-ci gagnent généralement en support technique et en suivi de bug. Mais de toute façon, quelque soit votre plateforme, il existe une version en *freeware*.

2 Du code à l'imprimante.

La base de votre document sous \LaTeX est donc un fichier de texte pur qui peut être écrit avec n'importe quel éditeur de texte. Evidemment ce fichier ne ressemble en rien au résultat papier que vous voulez obtenir. La première étape dans la quête de la jolie documentation que vous attendez est de compiler le document source \LaTeX . Par habitude les documents sources ont l'extension `.ex`. Pour compiler le document `doc1.tex` vous devrez taper la commande `latex doc1` ou `latex doc1.tex` à la ligne de commande de votre système. Si la compilation a lieu sans erreur \LaTeX devrait générer le fichier: `doc1.dvi`. Faisons un essai: tapez le document source \LaTeX suivant dans un fichier que vous nommerez `doc1.tex`. Essayez ensuite de le compiler comme indiqué ci-dessus:

```
\documentclass[]{article}
\begin{document}
Mon premier document!
\end{document}
```

Si tout se passe correctement et sans erreur vous devriez voir le fichier `doc1.dvi` apparaître dans le répertoire courant. L'extension `dvi` signifie: DeVice Independant. Les fichiers `dvi` sont des fichiers de description de pages indépendant des périphériques. Ils peuvent être aussi bien visualisés qu'imprimés. Sous l'environnement X Window d'UNIX la commande `xdvi "nom-fichier"` permet de les visualiser, et la commande `dvips "nom-fichier"` de les imprimer. Sous DOS/Windows il existe le programme `windvi` qui permet de visualiser et d'imprimer les fichiers `dvi`.

En cas d'erreur(s) lors de la compilation du fichier source, \LaTeX donne un message d'erreur et permet de taper des commandes interactivement. Vous pouvez soit corriger l'erreur à ce niveau, soit quitter \LaTeX en tapant la commande `X` (en majuscule) et la touche *ENTER* pour corriger votre fichier source.

3 Structure d'un document.

Un fichier source \LaTeX peut être décomposé en deux parties: le préambule, et le corps du document à proprement parler. Le corps d'un fichier source est toujours délimité par les commandes `\begin{document}` et `\end{document}`, signifiant respectivement début et fin de document. Tout ce qui se trouve avant la commande `\begin{document}` constitue le préambule. Dans l'exemple précédant, le préambule ne contient qu'une seule ligne.

3.1 Le préambule.

Le but du préambule est de spécifier à \LaTeX des informations sur la façon de formater le document. Il doit contenir un minimum d'une ligne, le tag `\documentclass` (il remplace le tag `\documentstyle`, utilisé sous $\LaTeX 2.09$). Ce tag doit être le premier du document source \LaTeX . Il donne des informations sur le type de document. Dans notre exemple notre document est du type `article`, mais il en existe d'autres:

book type livre, défini en plus des séparateurs de parties, comme le chapitre;
letter type lettre, défini une zone pour l'adresse de l'expéditeur, du destinataire;
slides pour faire des transparents;
report rapports.

En plus des types standards de documents, vous pouvez bien évidemment créer les vôtres.

Entre les crochets de la ligne `\documentclass[]{article}`, vous pouvez spécifier une liste d'options séparées par des virgules. Dans ce document le premier tag est: `\documentclass[12pt,français,a4paper,oneside]{article}`. Les options signifient respectivement: corps de police de caractère 12 points, langue française, papier format A4, impression recto. Il existe d'autres options, nous verrons cela plus tard.

On peut mettre d'autres commandes au niveau du préambule à la suite du tag `documentclass`. Nous verrons plus loin dans ce document les autres options du préambule, elle ne sont pas nécessaires pour commencer à écrire.

3.2 Le corps.

Le corps du document va contenir votre texte à proprement parler. Dans tout les cas celui-ci doit toujours être délimité par les marqueurs suivant:

début de document: `\begin{document}`

fin de document: `\end{document}`

Tout ce qui est entre ces deux tags sera imprimé sur le document final, exceptées les commandes. \LaTeX se charge lui-même de gérer les coupures en fin de ligne et la césure; vous pouvez donc formater votre document comme bon vous semble. Si dans votre éditeur vous séparez les différentes lignes par un retour de chariot (touche “*entrée*”), une fois compilé, vous n’en verrez rien. Pour forcer un retour de chariot dans votre texte, il vous faudra mettre deux anti-slash (\backslash) en fin de ligne. Si vous souhaitez non seulement forcer le passage à la ligne, mais aussi changer de paragraphe, il faudra laisser une ligne vide après la ligne se terminant par le \backslash .

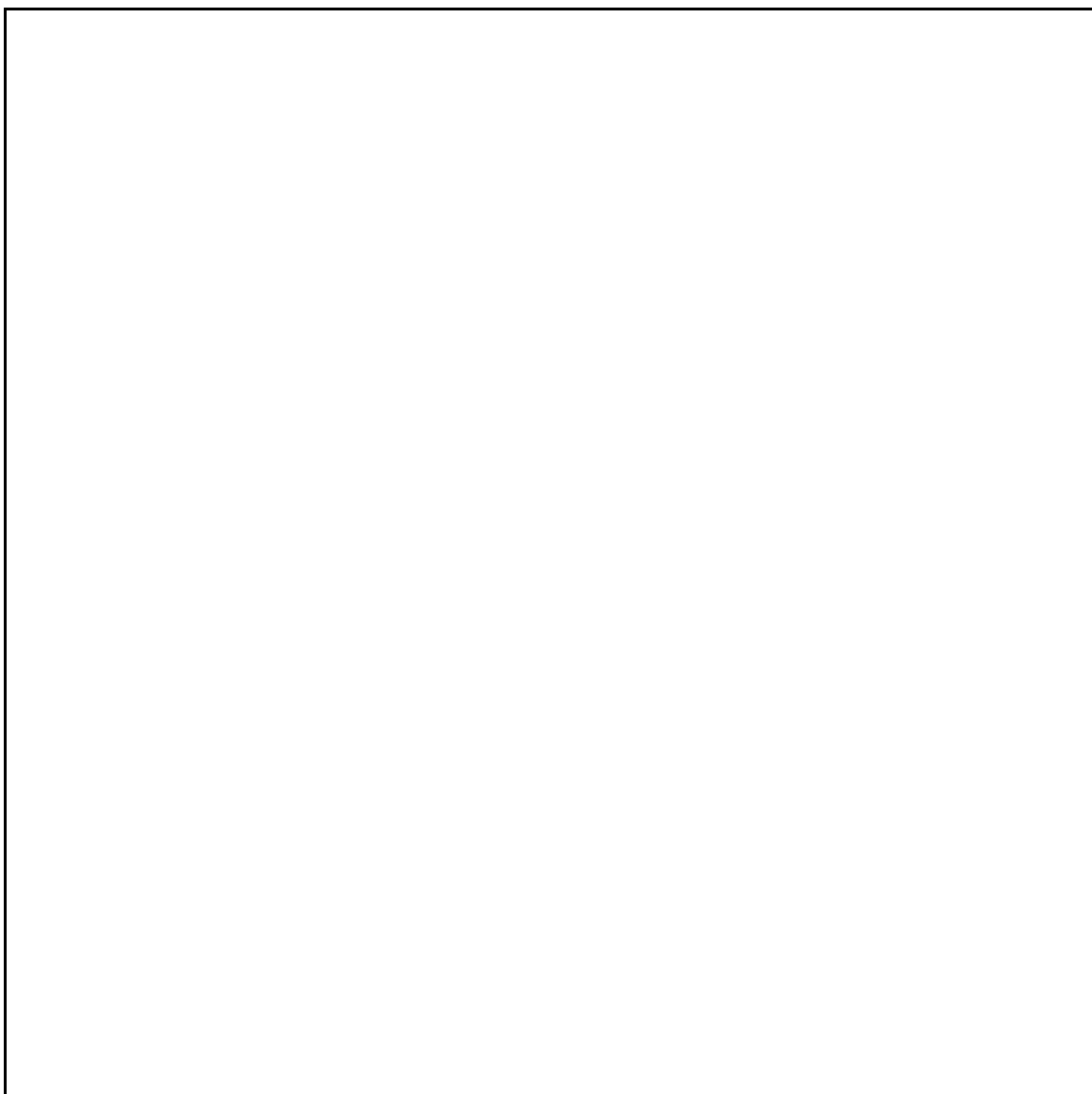
4 Exemple pratique.

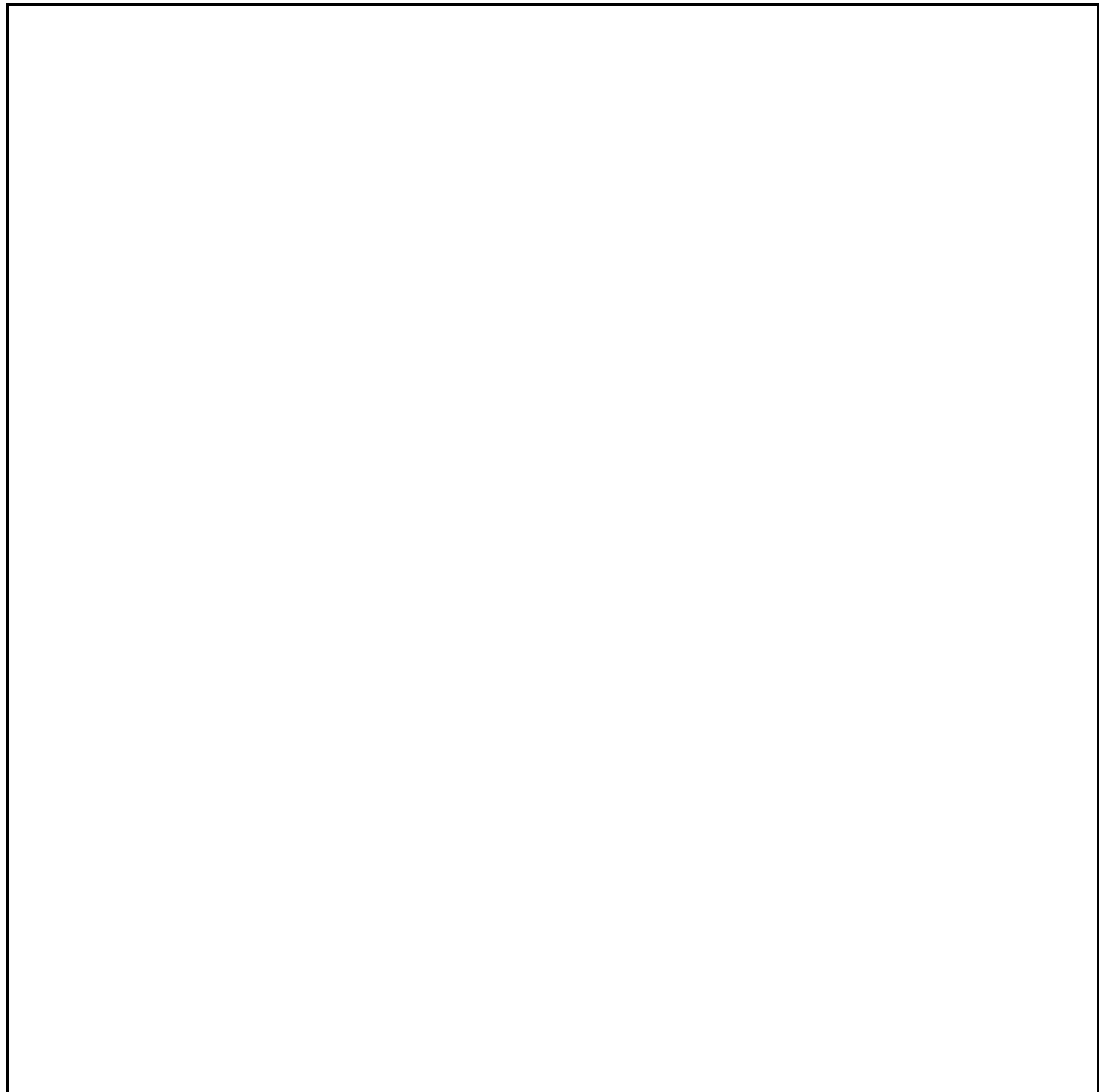
N’ayant pas, à l’époque, les moyens d’acheter de livre ou de trouver de documentation, j’ai appris \LaTeX en regardant les fichiers sources d’autres personnes. Je pense même qu’un tutoriel très bien fait ne remplacera jamais l’apprentissage par l’exemple. C’est pour cela que je tenais à inclure dans celui-ci le code source et le résultat d’un document \LaTeX complet. C’est un petit document de quelques pages regroupant la plupart des marqueurs utilisés couramment. Ce document n’intègre pas la totalité des tags présentés dans ce tutoriel, mais rien de vous empêche de le modifier. Vous pourrez retrouver le fichier source `.tex` de l’exemple sur le site où vous avez récupéré ce document.

Il n’est peut-être pas souhaitable que vous vous amusiez dès maintenant à décrypter tout le code source. Regardez plutôt l’aspect général du fichier. Toute les commandes qui sont utilisées sont décrites en détail plus loin dans ce document.

4.1 Document exemple.

Vous trouverez sur la page suivante notre exemple une fois passé à travers le compilateur \LaTeX :





4.2 Code source.

```
1 \documentclass[10pt,a4,oneside]{article}
  \usepackage[babel]{french}
  \usepackage[latin1]{inputenc}
  \usepackage[T1]{fontenc}
5
  %\selectlanguage{français}
```

```

\title{Présentation par (et de) l'absurde.}
\author{Nicolas Sayer\footnote{aidé par ceux qui composent sa vie.}}
10 \date{\today}

\begin{document}
\maketitle
15 Ce document est non-seulement l'exemple lié au dernier <<best-seller>> de
Nicolas Sayer (qui paraîtra bientôt dans mon \emph{home dir}), mais aussi
une succincte présentation de son auteur. Le lecteur de ces lignes voudra
bien de ne pas porter de jugement sur le personnage principal de cette
histoire, mais d'utiliser ce document pour ce qu'il est: une démonstration
20 de \LaTeXe\ldots

\section{Plus et moins.}
Nous parlons donc d'un homme de 22 ans. Il possède des défauts,
et des qualités\footnote{si si, il paraît que oui!} (comme
25 n'importe qui, je sais). Le problème avec notre homme c'est qu'il
est très fort dans ces qualités. Mais ces défauts son très
{\LARGE très} mauvais.\\

Listons donc les plus:
30 \begin{itemize}
\item{La greffe de mémoire vive sous son cortex s'est très
bien passée;}
\item{Il va bientôt fêter un mois de vie commune avec son
ours en peluche;}
35 \item{Il est très serviable, voir même trop, voir même
brave\ldots}
\end{itemize}

Je ne listerais bien entendu pas les défauts (si vous croyez
40 pouvoir me faire chanter, inspecteur à la con!).

\subsection{Le \emph{very} plus.}
Il est sympa! Il serai prêt à tout pour pouvoir sortir un pote
du caca\footnote{je trouvais ça plus joli que {\sl de la
45 merde}.}. Et il est tellement sympa qu'il serait même capable
d'enfoncer ses ennemis si il pensait que cela pouvait lui ramener
plus de copains.

\subsection{Le \textsl{très} moins.}
50 Son gros problème (ne me dite pas que vous ne l'avez pas
remarqué) c'est l'orthographe. En anglais il se débrouille
correctement, mais en français c'est une catastrophe
complète. Il a donc besoin de trouver des correcteurs.

```

Aujourd'hui qu'il est sorti du monde de `\emph{Bill}` il est
 55 obligé de se tourner vers des joujoux comme `\textsf{ispell}` ou
`\textsf{mon Papa}`.

```
\section{Et l'info dans tout ça ?}
Pour Nicolas, l'informatique commence à devenir une longue
60 histoire d'amour. Il avoue lui-même ne pas avoir été très
fidèle à une certaine époque, mais il tente aujourd'hui
d'être un amant parfait avec son système. Son seul regret est
de ne pas être né plus tôt et d'avoir raté le commencement
de cette merveilleuse aventure. Vous trouverez un résumé de son
65 existence informatique en figure \ref{Fjust_maried}.\
```

Le but de sa vie d'informaticien est la découverte de nouvelles
 choses quelles qu'elles soient! L'avantage, pour lui, des systèmes
 UNIX est qu'il n'est pas près d'avoir fini de découvrir.

```
70 \begin{figure}
\begin{center}
\begin{tabular}{|c|l|l|}
\hline
75 période & jouet & & & jeux découverts \\
\hline
\hline
82-90 & & Radio Shack TRS-80 (4ko) & & Basic \\
\hline
80 85-90 & & Amstrad CPC 464 (64ko) & & Basic, Assembleur Z80, CPM \\
\hline
90-92 & & Apple's Macintosh Classic & & C, Assembleur 68000 \\
\hline
92-94 & & Mon premier PC sous DOS/Windows & & C/C++, Assembleur X86 \\
85 \hline
94---> & & Enfin: Linux & & awk, bash, \LaTeX, HTML, TCP/IP\ldots \\
\hline
\end{tabular}
\end{center}
90 \caption{\emph{Les différents mariages de Nicolas Sayer.}}
\label{Fjust_maried}
\end{figure}
```

```
\section{Conclusion.}
95 Voilà, ce bout de papier ne voulant rien dire est terminé. Il
n'explore qu'un très petit panel des possibilités de \LaTeX,
mais comme base c'est largement suffisant.
```

```
\end{document}
```

4.3 Analyse.

Nous allons analyser ensemble le code source de l'exemple donné précédemment. Je rappelle que cet exemple n'est qu'un complément au reste de ce tutoriel. Je vais donc donner, ligne par ligne, l'utilité de chaque commande.

- 1 Défini la classe du document que nous allons utiliser: police de caractère 10 points, en français, papier A4, recto. J'utilise la classe `article`;
- 2 Utilisation du package `babel` en définissant la langue du document en français;
- 3 Titre de la page de garde;
- 4 Auteur pour la page de garde;
- 5 Date pour la page de garde `\today` donnera la date du jour, à la compilation;
- 8 Fin du préambule, début du corps du document;
- 9 Insertion de la page de garde ici;
- 10 Enfin, le début de notre texte...
- 11 Quelques accents tordus sur cette ligne;
- 12 Utilisation de `emph` pour *mettre en avant*;
- 16 Les macros `\LaTeX` et `\ldots`;
- 18 Un titre de premier niveau;
- 20 Une note en bas de page;
- 23 Agrandissons un peu la police de caractère;
- 26–33 Une petite liste d'éléments;
- 38 Un sous-titre;
- 40 Plein de choses: une note en bas de page, un *c-cédille* et des *italiques*;
- 45 Second sous-titre, avec des *italiques* pour être drôle;
- 51 & 52 `textsf`, une police sans empâtement;
- 54 Nouveau titre de premier niveau;
- 61 Une référence à la figure `Fjust_maried` (sera remplacé par le numéro de la figure en question. La ligne fini par un `\\` pour forcer le passage à la ligne;
- 63 Début d'un nouveau paragraphe;
- 67 Début de l'environnement pour une figure;
- 68 Elle sera centrée;
- 69 Début de l'environnement pour créer un tableau;
- 70–83 Contenu du tableau;
- 84 Fin de l'environnement pour les tableaux;
- 85 Fin de la zone centrée;
- 86 Légende à positionner sous la figure;
- 87 Etiquette correspondant à la référence vue précédemment;
- 88 Fin de la figure;
- 90 Encore un titre;
- 95 Fin du document.

5 Organiser votre document.

La très grande force de \LaTeX se situe dans l'organisation des documents. Grâce à des commandes simples vous allez pouvoir découper votre texte en différentes parties logiques, créer automatiquement la table des matières...

5.1 Découpage du document.

Tout document un tant soit peu organisé possède des chapitres, parties, sous-parties, etc. Comme vous avez pu le voir dans l'exemple, \LaTeX se charge lui-même de créer les titres et de les numéroter. La seule chose que vous ayez à faire est de mettre un marqueur donnant le titre de la section en question entre les différentes parties de votre document.

Le titre de la sous-partie que vous êtes en train de lire a été créé avec la commande: `\subsection{D'ecoupage du document.}`

De la même façon, le titre global précédant est fait grâce à:

`\section{Organiser votre document.}`

Vous remarquerez que \LaTeX s'est chargé lui-même de numéroter ces parties. Je peux donc très facilement insérer un nouveau chapitre en début de document; les suivants seront automatiquement re-numérotés.

<i>partie</i>	<code>\part{...}</code>
<i>chapitre</i>	<code>\chapter{...}</code>
titre	<code>\section{...}</code>
sous-titre	<code>\subsection{...}</code>
sous-sous-titre	<code>\subsubsection{...}</code>

J'ai donné dans ce tableau les séparateurs dans l'ordre logique: un chapitre contient des sections, qui contiennent des sous-sections... Les marqueurs `part` et `chapter` ne sont définis que dans le type de document `book` (livre). Et celui-ci ne connaît pas le tag `subsubsection`.

Grâce à ces tags vous pouvez créer des documents très organisés, qui seront donc beaucoup plus simples à aborder pour le lecteur. Pour ma part, j'essaye d'exploiter au maximum ces possibilités de découpage, de cette façon j'ai moins de mal à écrire car les choses sont claires. Alors que je suis en train d'écrire ces lignes, toutes les parties suivantes de ce document sont déjà définies. Ainsi, je sais exactement ce que j'ai à taper dans chaque section sans me répéter et je n'ai plus qu'à remplir les blancs.

Il existe aussi un tag: `\paragraph{...}` qui, je vous le donne en mille, permet de donner un titre à un paragraphe. Ceux-ci ne sont pas numérotés et n'apparaissent pas dans la table des matières.

5.2 La table des matières.

Grâce aux titres que nous avons mis en place dans la partie précédente, lors de la compilation, \LaTeX sait exactement où se situent ces titres. Il n'est donc pas difficile pour lui de générer une table des matières avec les titres et les numéros des pages où ils commencent. Le tag `\tableofcontents` permet d'insérer cette table des matières. Il est généralement placé en début de document, juste après la page de garde.

Pour être sûr que la table soit correcte (numéros de pages) il faut compiler successivement deux fois votre fichier source (`latex doc.tex` ; `latex doc.tex`). \LaTeX a besoin de passer une première fois sur le fichier pour noter où sont les sections et une seconde pour inclure la table avec les résultats trouvés. En testant, vous pourrez remarquer qu'au lieu d'afficher en titre "Table des matières", vous avez "Table of contents". C'est tout à fait logique si vous n'avez pas indiqué à \LaTeX que vous vouliez écrire en français (voir page 25).

De façon identique vous pouvez demander à \LaTeX de créer une liste des figures et des tables avec la commande `\listoffigures`.

5.3 Page de garde.

La page de garde standard de \LaTeX est très sobre (voir trop), mais vous pouvez très bien la modifier. Cependant ce n'est pas mon but de vous donner tout de suite tout les *trucs*. Pour mettre une page de garde dans votre document, il suffit de placer le tag `\maketitle` juste au début du corps du document. Evidemment, \LaTeX a beau être très intelligent, même s'il sait qu'il faut mettre une page de garde, il ne pourra le faire correctement sans en connaître le contenu. Pour cela quelques tags ont été définis. Voici un document possédant une page de garde:

```
\documentclass[]{article}

\begin{document}

\title{Tintin et Milou dans les KTA}
\author{par Nicolas Sayer}
\date{\today}

\maketitle

Tintin et Milou sont en vadrouille dans un TGV...
\end{document}
```

Remarquez l'utilisation du tag `\today` qui donne la date du jour (on aurait très bien pu mettre une date fixe) ainsi que le tag `\clearpage` pour forcer un changement de page. Encore une fois, si la date est affichée en anglais, aller voir en page 25.

A la suite de la page de garde vous pouvez utiliser l'environnement `abstract` pour écrire un résumé du document (obligatoire sur un long document) de la façon suivante:

```
\abstract{le résumé}
```

5.4 Références dans un document.

Quand un document devient important, il est souvent intéressant dans une phrase de se référer à une autre partie du texte. \LaTeX sait aussi automatiser la gestion des références. Tout ce que vous avez à faire est de mettre un tag: `\label{toto}` à l'endroit que vous voudrez référencer. Puis vous pouvez placer n'importe où dans le texte une phrase du style:

```
pour plus d'info, voir page \pageref{toto}
```

A la compilation le tag `\pageref{toto}` sera remplacé par le numéro de la page où se situe le tag `\label{toto}`. De façon identique vous pourrez donner le numéro de partie où se trouve l'étiquette en utilisant `\ref{toto}`.

Grâce aux références de \LaTeX , je peux vous dire que nous sommes à la page 15, de la section 5.4 en étant sûr de ne pas me tromper. Quand je réalise de gros documents, j'ai pour habitude de mettre un `label`, avec un nom générique, à chaque partie du document. Je peux donc très simplement dire au lecteur d'aller jeter un coup d'œil dans un autre coin.

5.5 Index, glossaire et bibliographie.

\LaTeX peut vous aider à générer automatiquement l'index et le glossaire de votre document. J'ai malheureusement très peu d'expérience dans ce domaine. Je ne vous donnerai donc pas d'information supplémentaire. Votre distribution de \LaTeX inclue sûrement des informations à ce sujet.

Pour ce qui est de la gestion d'une table bibliographique en fin de document, \LaTeX possède le package `BibTeX`. Celui-ci est compris dans la majorité des distributions du marché. Son système de gestion de bibliographie est un des atouts les plus appréciés par les utilisateurs de \LaTeX . Il existe même des tables complètes toutes préparées, possédant les références des livres les plus importants de bon nombre de domaines.

6 Jouer avec les caractères.

Comme n'importe quel traitement de texte, \LaTeX permet de jouer avec les caractères. Dans tout document il est toujours intéressant de pouvoir mettre en avant certaines parties d'un texte ou certains mots.

6.1 Changer la police.

La police de base sous \LaTeX (roman) a été conçue pour être très lisible et agréable à regarder. Je ne m'attarderai donc pas à vous expliquer comment la changer. Dans mes documents, j'aime jouer avec les polices pour améliorer la clarté. Dans les manuels informatique, on a pour habitude de toujours noter les morceaux de code en police de taille fixe, comme `cela`. Quand je fais référence à une commande située dans un menu ou un bouton, j'ai pour habitude d'utiliser une police sans empâtement, comme celle-ci. Pour mettre un mot en avant, ou lorsque j'utilise des mots du langage parlé je mets mes mots en *emphasis*. Enfin, quand je ne sais pas trop comment mettre le mot j'utilise des *italiques*.

du gras	<code>du \textbf{gras}</code>
un peu <i>d'italique</i>	<code>un peu \textsl{d'italique}</code>
pourquoi pas comme ça	<code>pourquoi pas \textsf{comme \c{c}a}</code>
<code>#include <pthread.h></code>	<code>\texttt{\#include <pthread.h>}</code>
une petite <i>Guinness</i>	<code>une petite \emph{Guinness}</code>

Vous pouvez évidemment vous amuser à mélanger les styles... Comme vous l'avez peut-être remarqué, certains caractères ne sont pas correctement gérés par \LaTeX , en l'occurrence tout les caractères de contrôle comme: `\ # & % $` et j'en oublie. Pour la plupart de ces caractères, il suffit de mettre un anti-slash devant pour qu'ils deviennent affichables, mais ce n'est pas toujours possible. Si vous voulez mettre dans votre document tout un morceau de texte bourré de caractères spéciaux (comme un bout de code C) \LaTeX offre l'environnement `verbatim`. Tout ce qui est tapé dans cet environnement est reproduit tel quel dans le document final. Pour pouvoir afficher le code de l'exemple de police ci-dessus, c'est cet environnement que j'ai utiliser.

Quand on fait des threads, il faut:	Quand on fait des threads, il faut:
	<code>\begin{verbatim}</code>
	<code>#include <threads.h></code>
<code>#include <pthread.h></code>	<code>#include <semaphore.h></code>
<code>#include <semaphore.h></code>	<code>\end{verbatim}</code>

Le texte situé dans cet environnement est reproduit de façon identique, que ce soit pour les retours de chariot, les espaces, etc. Comme vous vous en doutez peut-être, il y a un problème quand on souhaite faire apparaître la commande `\end{verbatim}`, car c'est elle qui termine l'environnement. Pour contourner ce

problème on utilise le marqueur `\verb`, c'est d'ailleurs lui que j'utilise pour vous donner tout ces exemples. Il est très pratique car vous définissez vous même le caractère de début et de fin. Je m'explique:

```
\verb+ls -lR > /dev/null+
```

Ce qui donne: `ls -lR > /dev/null`

L^AT_EX a vu que le premier caractère après le `\verb` était un `+`, il attend donc comme caractère de fin un autre `+`. Si jamais vous souhaitez mettre un `+` dans la ligne, vous pouvez utiliser un autre caractère:

```
\verb#g++ -o toto toto.c#
```

6.2 Taille des caractères.

Je ne vais expliquer ici que la façon de modifier la taille de caractère pour une partie du texte. Pour modifier la taille sur le document global allez voir la partie sur les commandes du préambule (section 8). Les marqueurs pour changer la taille s'utilisent de façon identique que ceux des polices:

```
My name is Bond.
```

```
My name is Bond.
```

```
My name is Bond.
```

```
My name is Bond.
```

```
My name is Bond.
```

```
My name is Bond.
```

```
My name is
```

```
Bond.
```

```
{\tiny My name is Bond.}
```

```
My name is Bond.
```

```
{\large My name is Bond.}
```

```
{\Large My name is Bond.}
```

```
{\LARGE My name is Bond.}
```

```
{\huge My name is Bond.}
```

```
{\Huge My name is Bond.}
```

6.3 Caractères spéciaux.

Vous êtes sûrement au courant, mais votre clavier ne possède pas tout les caractères existant⁴. L^AT_EX possède des tags pour tout ces caractères bizarres, ainsi que pour nos accents chéris.

6.3.1 Accents et caractères français.

Si votre système L^AT_EX n'est pas correctement installé sur votre site, quand vous mettez le caractère `é` dans votre fichier source, celui-ci n'apparaîtra pas correctement dans le résultat une fois compilé. Ceci est dû aux différents encodages utilisés selon votre système pour les caractères de code ASCII supérieur à 128. Si vous voulez être sûr que votre fichier source se compile sans aucun problème sous n'importe quelle machine, ou si les accents n'apparaissent pas lorsqu'ils sont tapés directement au clavier, vous pouvez utiliser les tags suivants:

4. Et encore moins tout les caractères de la langue française.

tag	résultat
<code>\'e</code>	é
<code>\'e</code>	é
<code>\^e</code>	ê
<code>\~e</code>	ë
<code>\"e</code>	ë
<code>\'o</code>	ó
<code>\'o</code>	ó
<code>\^o</code>	ô
<code>\~o</code>	õ
<code>\"o</code>	ö

Je suppose que vous comprendrez très facilement comment mettre un accent sur un “a”, c’est la même méthode. A noter que \LaTeX est bête et méchant et qu’une faute de frappe peut très bien donner des résultats très drôles: \acute{s}

Pareillement un c cédille (\grave{c}) est codé: `\c{c}`, et on reste fun: \grave{u} . Pour faire de l’œil: `l'\oe}i1`.

Le seul moment où cela devient moins drôle c’est quand \LaTeX devient trop logique. Tentez de faire un i accent grave, vous m’en direz des nouvelles:

source	résultat
<code>\'i</code>	ì

Pour dire à \LaTeX d’enlever cet idiot de point:

source	résultat
<code>\'{\i}</code>	ì

6.3.2 Les autres.

\LaTeX possède aussi des macros pour certains caractères bizaroïdes:

source	résultat
<code>\LaTeX</code>	\LaTeX
<code>\TeX</code>	\TeX
<code>\LaTeXe</code>	$\text{\LaTeX} 2_{\epsilon}$
<code>\%</code>	%
<code>-</code>	-
<code>--</code>	—
<code>---</code>	—
<code>\ldots</code>	...

7 Formatage du contenu.

7.1 Les notes.

\LaTeX se charge lui-même de toute la mise en page. Entre autre ce qui concerne les notes. Il existe deux types de notes. Les notes en bas de page et les notes en marge.

Une note en bas de page est référencée par un numéro⁵. Pour créer une note en bas de page vous devez utiliser le tag `\footnote`. Voici le code utilisé pour faire une note en bas de page:

```
...les chiens\footnote{voire certaines poules.} sont des animaux...
```

Ça donne: ...les chiens⁶ sont des animaux...

Les notes en marge sont aussi simples à faire:

Comme cela.

```
...ma maman\marginpar{la femme de papa} est une femme...
```

Et cela: ...ma maman est une femme...

la femme de
papa

7.2 Les listes.

Que dire? à part de jeter un coup d'œil à notre exemple.

\LaTeX connaît trois types de listes:

`itemize` Liste simple d'éléments décalés précédés d'un tiret (ou d'un point en format anglais);

`enumerate` Idem, excepté qu'à la place des tirets, le compilateur met des chiffres: 1,2,3, etc. C'est une liste numérotée;

`description` Comme ce que je suis en train de faire, idéal pour les définitions.

Vous allez être heureux, je vais enfin dire du mal de \LaTeX . J'avoue que je suis assez déçu du changement de syntaxe dans les listes. Je m'explique:

– Je suis au début;	<code>\begin{itemize}</code>
	<code>\item{Je suis au d\'ebut;}</code>
– Moi c'est le milieu;	<code>\item{Moi c'est le milieu;}</code>
	<code>\item{Partant pour la fin.}</code>
– Partant pour la fin.	<code>\end{itemize}</code>

Certes, pour une liste numérotés c'est proche:

5. Comme ça.

6. voire certaines poules.

1. Je suis au début;	<code>\begin{enumerate}</code>
	<code>\item{Je suis au d\'ebut;}</code>
2. Moi c'est le milieu;	<code>\item{Moi c'est le milieu;}</code>
	<code>\item{Partant pour la fin.}</code>
3. Partant pour la fin.	<code>\end{enumerate}</code>

Mais alors pour les définitions, c'est à dire les `description`, on peut le dire: C'est la pagaille!

Moi Le type devant l'écran;	<code>\begin{description}</code>
	<code>\item[Moi]Le type devant l\'écran;</code>
Lui Le mec qui tente de me lire;	<code>\item[Lui]Le mec qui tente de me lire;</code>
	<code>\item[Ma femme]Celle que j'aime (elle</code>
Ma femme Celle que j'aime (elle est	<code>est pas au courant, enfin maintenant si</code>
pas au courant, enfin maintenant	<code>(c'est m^eme elle qui corrige!)).</code>
si (c'est même elle qui corrige!)).	<code>\end{description}</code>

Au final on peut quand même dire que ce n'est pas trop sorcier. C'est logique, c'est tout! Bien entendu on peut jouer avec ce schéma; en gardant la syntaxe et la logique vous pourrez imbriquer les listes comme bon vous semble et mélanger des définitions dans les listes d'éléments...

Dans une liste en français, les différents éléments doivent être terminés par des points virgules, excepté pour le dernier, où c'est un point⁷.

7.3 Les tableaux.

`LATEX` n'a aucune prétention de rivaliser avec Excel, il sait pourtant faire des tableaux comme celui que vous avez pu voir dans la demo. L'environnement `tabular` que nous allons étudier est somme toute assez simple, certains packages supplémentaires permettent de créer des tableaux sur plusieurs pages, incluant d'autres environnements.

`tabular` est un environnement. Il commence donc par un `begin` et se termine par un `end`, comme pour les listes. Juste après le `begin` il faut donner à `LATEX` des informations sur les colonnes que vous souhaitez avoir dans votre tableau. Exemple:

```
\begin{tabular}{|l||c|c|}
```

Cette entête définit un tableau à trois colonnes séparées par des lignes verticales. Entre la première et la seconde colonne on a une double ligne. Les `c` et le `l` donnent des informations sur le type de justification que l'on souhaite dans cette colonne:

1 Justifié à gauche (left);

⁷. Ce n'est pas une invention de ma part, c'est comme ça que ça se fait dans notre belle langue.

c Centré;
r Justifié à droite (right).

Au sein de l'environnement `tabular` vous devez écrire le texte qui sera contenu dans le tableau:

	CPU	RAM
Win 3.1	i486	8Mo
Win NT	i586	32Mo
Linux	i386	4Mo

```

~      & CPU  & RAM  \\
\hline
\hline
Win 3.1 & i486 & 8Mo  \\
\hline
Win NT  & i586 & 32Mo \\
\hline
Linux   & i386 & 4Mo  \\
\hline
\end{tabular}

```

```
\begin{tabular}{r||1|1|}
```

Vous remarquerez l'utilisation du tilde (blanc insécable sous \LaTeX) pour signifier que la case est vide. On utilise le tag `\hline` pour mettre une barre horizontale.

7.4 Les figures et les tables.

Les figures et les tables sont généralement utilisées pour inclure des éléments non-text dans votre document. Des schémas, des photos, des tableaux... \LaTeX gère lui-même le placement de ces objets, ainsi que leur numérotation. Vous pourrez même posséder une liste des figures et des tables avec les numéros des pages correspondantes (voir 5.2).

Les tables et les figures sont déclarées par des environnements. Le contenu de celles-ci reste au format \LaTeX classique. Voici comment créer une figure:

```

\begin{figure}
ici je mets le contenu...
\end{figure}

```

Le contenu de la figure peut être du texte, une image, etc. La déclaration d'une table se fait de façon identique: en remplaçant dans le code \LaTeX ci-dessus le mot `figure` par `table`. Les figures et les tables de \LaTeX sont aussi couramment appelées des *float*, car elles flottent dans la page.

Au sein de ces deux environnements, on peut placer d'autres tags particuliers. Le tag `\caption{Une photo}` vous permet de mettre une légende à votre figure. Le tag `\label` placé à l'intérieur de l'un de ces environnements permet

de faire une référence directe à cette figure (voir section 5.4).

Enfin, un petit exemple pour mieux se faire comprendre:

```
\begin{figure}
\includegraphics{lochness.jpg}
\caption{Photo du monstre du Loch-Ness.}
\label{Flochness}
\end{figure}
```

7.5 Les césures.

Comme dit, en première partie du document, \LaTeX s’occupe lui-même des coupures de mot en fin de ligne. Il s’arrangera pour ne couper les mots que quand c’est réellement nécessaire. Si la table de césure française est installé, il coupera les mots selon les règles d’usage de la langue française. Cependant, il se peut que \LaTeX n’arrive pas à couper correctement un mot. Si c’est le cas il donnera une erreur lors de la compilation. Si vous avez mis le mot-clef `draft` dans les options de classe, une “patouille” sera placée à côté du mot mal justifié.

Vous pouvez dire à \LaTeX comment il à le droit de couper un mot si jamais vous voyez qu’il a du mal à le faire:

```
...voici une phrase qu'il faut ap\ -pren\ -dre vite!...
```

Si le compilateur doit couper le mot “apprendre” de la phrase précédente, il ne le fera qu’entre les deux ‘p’ et le ‘n’ et le ‘d’. Si la table de césures françaises est installée sur votre système, vous n’aurez que très exceptionnellement besoin d’indiquer où peuvent être coupés les mots.

7.6 Modifier le type de justification.

Par défaut \LaTeX justifie le texte sur la largeur entre la marge de gauche et la marge de droite. Vous pouvez modifier cette justification en incluant le texte à modifier dans un environnement comme suit:

```
\begin{center}
Je suis au centre!
\end{center}
```

L’environnement donné en exemple centre le texte en milieu de page. Vous pouvez aussi le justifier à gauche ou à droite de façon identique:

`center` Centre le texte;

flushright Justifie à droite;

flushleft Justifie à gauche.

7.7 Quelques derniers tags.

Pour des raisons de présentation vous pouvez souhaiter forcer \LaTeX à changer de page. Le tag `\clearpage` laissera la fin de la page vide et passera à la suivante.

Dans la série des environnements, j'en ai deux petits à vous présenter:

quote Pour les citations, le texte est mis en retrait sur la page;

versus Des vers, pour les poètes.

8 Commandes du préambule.

Le préambule d'un fichier source \LaTeX comprend tout le texte se trouvant avant la ligne `\begin{document}`. On place dans le préambule des indications sur la façon de formater le fichier et d'autres paramètres globaux. Les commandes données dans le préambule agissent sur tout le document, un peu comme dans le langage C au niveau des variables globales.

8.1 Les classes de documents.

La première ligne d'un document \LaTeX (et donc du préambule) est toujours `\documentclass`. Elle spécifie la classe du document que l'on souhaite utiliser. \LaTeX propose, en standard, un certain nombre de classes qui vous permettront de créer la majorité de vos documents. Selon la classe que vous décidez d'utiliser, \LaTeX utilisera des paramètres différents au niveau du formatage du document. Voici une liste des classes existantes en standard sous $\text{\LaTeX 2}_{\epsilon}$. Selon la distribution installée chez vous, d'autres classes peuvent exister. Vous pouvez aussi créer vos propres classes (`.sty`) mais cela sort du domaine de cette documentation.

report Classe utilisée pour créer des rapports. Pour ma part, c'est la classe que j'utilise le plus souvent;

article Utilisé généralement pour des textes plus courts que la classe précédente;

book Classe utilisée pour produire des livres. Cette classe définit en plus les tags `\chapter{}` pour les chapitres, `\openright` forçant \LaTeX à démarrer les chapitres sur la page de droite et quelques autres marqueurs supplémentaires (`\frontmatter`, `\mainmatter`, `\backmatter`).

letter Cette classe est utilisée pour écrire des lettres. Les différentes commandes de sectionnement n'existent plus (défini en 5.1). En revanche, cette classe

possède des tags pour inclure l'adresse de l'envoyeur, du destinataire, les pièces jointes. . .

slides Celle-ci est totalement spécifique à la production de transparents pour rétroprojecteurs. Personnellement je ne l'utilise pas. \LaTeX est surtout pratique pour produire des documentations, pas des Post-It.

Je n'ai pas le temps, et pas vraiment la connaissance, pour donner tous les détails sur chacune de ces classes. Ce tutoriel concerne surtout les classes `article` et `report`. Si vous sentez le besoin d'utiliser une de ces autres classes vous pouvez vous référer aux documents indiqués en 10.

8.2 Les options.

Dans la déclaration `\documentclass` vous pouvez aussi rajouter des options. Celles-ci sont spécifiques à la classe que vous utilisez. Cependant, certaines options sont génériques:

10pt Fixe la taille de la police de caractères de base du document à 10 points (ça marche aussi pour 12. . .);

oneside Document en recto;

twosides Recto-verso;

a4paper Redéfinit les marges du document pour du papier A4 (21x29.7);

français Indique que le document est en français, voir section 8.6;

draft Mode brouillon, n'imprimera pas les images et mettra une "patouille" à la fin des lignes où il y a un problème de justification.

Encore une fois, cette liste est loin d'être exhaustive mais suffisante pour la plupart des besoins. Les options doivent être spécifiées comme suit sur la première ligne du préambule:

```
\documentclass[a4paper,français,twoside]{article}
```

8.3 Entête et numérotation de page

En standard, \LaTeX imprime les numéros de page en bas de page et centré. Vous remarquerez que ce n'est pas le cas sur le document que vous êtes en train de lire. Ici, les numéros sont en haut de la page, ainsi que le titre de la section en cours. Pour changer ce comportement par défaut j'ai rajouté le tag suivant au niveau du préambule:

```
\pagestyle{headings}
```

Alors que le réglage par défaut est:

```
\pagestyle{empty}
```

Bien que cette commande soit placée dans le préambule, elle peut être mise n'importe où dans le fichier source. Vous pouvez donc changer de style en plein milieu de votre document. Il existe un dernier type: `\pagestyle{myheadings}` vous permettant de choisir vous même le contenu de la tête et le pied de page.

8.4 Les packages.

Dans les environnements d'informaticiens, la règle d'or pour quelque système que ce soit est l'extensibilité. \LaTeX ne fait pas exception à la règle. La commande pour inclure un package est: `\usepackage[options]{nom du package}`

Les packages créés pour \LaTeX sont extrêmement nombreux et variés. Ils sont généralement créés par des utilisateurs voulant posséder certaines fonctions en plus. Souhaitant que la taille de cette documentation reste limitée en, je ne donnerai pas de détails sur l'utilisation de packages. En section 8.5, je donne toutefois des informations sur quelques packages intéressants.

8.5 Packages intéressants.

Les packages (inclus au niveau du préambule par la commande `\usepackage`) offrent des possibilités d'extention presque infinies, que ce soit pour inclure des images, faire des figures. . .

epsfig Permet d'inclure des images en PostScript encapsulé;

graphics Inclusion d'images à partir de différents formats;

moreverb Plus de puissance pour l'environnement `verbatim`;

babel \LaTeX multilingue;

tabularx Donne plus de puissance à l'environnement de création de tableau;

multicol Permet de mettre une partie de document sur plusieurs colonnes.

8.6 Pour nous: Les Français!

Originellement \LaTeX a été créé par et pour des anglo-saxons. Si vous souhaitez uniquement taper en anglais vous pouvez donc arracher cette page. En revanche, que les francophones restent! (vous êtes encore tous là?).

Commençons tout d'abord par le plus simple. Si vous avez inséré une table des matières, des chapitres, un résumé, ou la date avec `\today`, vous vous êtes

peut-être rendu compte que L^AT_EX met tout en anglais. Pour remédier à ce problème, il suffit de mettre `français` dans les options de classes.

Mais ce n'est pas tout. Notre exception culturelle est belle et bien réelle! Dans une liste d'éléments, on met des tirets et pas des points. Un point-virgule doit être séparé de la lettre qui le précède par un demi-blanc. . . Pour finir, les règles de césure sont différentes (les mots sont différents d'ailleurs). Le package `babel` permet de définir ces petits "détails". Pour l'inclure dans votre document il suffit: `\usepackage[français]{babel}`

A condition, bien évidemment, que l'installation de L^AT_EX que vous utilisez possède `babel` et la table des césures française.

9 whereis L^AT_EX & Outils additionnelles.

Qu'est-ce que j'appelle des outils additionnels? Et bien se sont des programmes externes à L^AT_EX, qui simplifient son utilisation.

9.1 Où trouver L^AT_EX 2_ε?

UNIX A mon avis la meilleure distribution de L^AT_EX sous UNIX est de loin `teTeX`, mais les avis peuvent diverger. Vous pouvez trouver celle-ci via `ftp.loria.fr/pub/ctan` Ce site regroupe des distributions et des packages L^AT_EX pour à peu près toutes les plateformes possibles et imaginables.

DOS, OS/2, Windows J'ai découvert récemment la distribution `AsTeX` qui est très complète. Elle possède tout les outils de base, ainsi que des "plus" comme `GNU-plot` et `Win-Emacs`. Elle est disponible via `ftp` sur: `ftp.univ.orleans.fr/pub/tex/PC`.

Macintosh Dans le monde Apple, la distribution `Textures` fait référence, je n'ai pas actuellement de pointeurs dessus.

9.2 Outils sympas.

Quelques petits programmes permettent aujourd'hui de ce simplifier la vie avec L^AT_EX. N'ayant pas réellement tripoté d'autre mondes qu'UNIX à ce niveau, je n'ai pas grand chose d'autre à vous proposer (peut-être dans la version 2 de cette doc si vous pouvez me donner des idées):

xL^AT_EX Interface graphique (plus de commandes, que des boutons);

LyX Interface WYSIWYG (plus ou moins);

GNU make Non, je blague, y'a que des dingues comme moi pour utiliser ce genre de dinosaure. Mais c'est quand même très utile pour gérer des gros documents où l'on peut séparer les chapitres et les compiler séparément;

Scientific WorkPlace (sous Windows) paraît que c'est bien, surtout pour ce qui est des formules mathématiques;

L^AT_EX2HTML Permet de convertir vos documents L^AT_EX en HTML pour pouvoir les publier sur le web. Ce n'est pas une conversion brute comme Word, il découpe votre document selon les sections et crée une table des matières contenant des hyper-liens vers les parties respectives. Il ajoute aussi des boutons pour revenir en arrière, au sommaire... Un très joli petit produit;

L^AT_EX2RTF Convertit vos documents sources L^AT_EX en RTF⁸, pour que vos idées soient plus facilement lisibles par les gens moins barbares;

ispell Un logiciel de correction d'orthographe multilingue pour différents types de fichiers, dont L^AT_EX.

10 Aller encore plus loin ?

Enfin, un peu de littérature pour votre table de chevet. Je suis malheureusement assez pauvre en bouquins en français, mais je promets d'en ajouter bientôt:

Leslie Lamport. *L^AT_EX A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, 1994.

Michel Goossens, Frank Mittelbach, and Alexandre Samarin. *The L^AT_EX Companion*. Addison-Wesley, Reading, Massachusetts, 1994.

Donald E. Knuth. *The T_EX Book*. Addison-Wesley, Reading, Massachusetts, 1994.

11 Conclusion.

Voilà, c'est terminé. Je souhaitais réaliser un document donnant les bases de L^AT_EX2_ε et donnant au lecteur la possibilité d'aller plus loin et je crois avoir atteint mon but. En guise d'ultime exemple, vous pourrez retrouver les sources complètes de ce document sur mon site web (URL donnée en première page).

Pour finir je voudrais remercier les quelques personnes qui m'ont dit d'avance qu'elles souhaitaient avoir une copie de ce tutoriel, me permettant ainsi d'avoir un but concret. Merci aussi à Léo, ma re-liseuse en chef. Puis enfin aux quelques allumés de Jussieu, qui un beau jour m'ont donné la solution: L^AT_EX.

8. Rich Text Format, développé par Microsoft pour posséder un format (plus ou moins) standard entre les traitements de textes.